**2 -- 6. SQL vs NoSQL: Entenda Todas as Diferenças**

**Diferença banco relacional com NoSQL:**

No banco relacional existem as tabelas e Views, no banco NoSQL existem as coleções.

No banco relacional existem as linhas, no banco NoSQL existem os documentos (JSON, BSON).

No banco relacional você pode exigir que uma coluna seja preenchida, no banco NoSQL não é possível exigir.

No banco relacional você pode realizar uma junção, no banco NoSQL você deve salvar o objeto com o conteúdo todo. Ex: Banco relacional teria uma tabela de funcionários e uma tabela de dependentes, no banco NoSQL vai ter só um objeto com funcionário e seus dependentes.

No banco relacional você tem uma chave estrangeira que impede você de apagar os dados caso existam dados relacionados, no banco NoSQL você pode ter uma chave estrangeira mas ela não impede você de apagar os dados caso existam dados relacionados.

---------------------------------------------------

**2 -- 7 Arquitetura do MongoDB**

**Arquitetura para replicar dados em outro servidor:**

Para replicar o dado de um servidor MongoDB “Master” em outros servidores “Slave”, o Servidor “Master” deve ser o responsável por gravar os registros e replicar os registros nos outros servidores “Slave”. E os servidores “Slave” apenas executam a leitura dos registros.

---------------------------------------------------

**3 -- 9. MongoDB University: Conheça as Certificações de DEV e DBA**

**Site para conteúdos e certificações:**

<https://university.mongodb.com/>

---------------------------------------------------

**3 -- 10. Simulado e Preparação para as Certificações**

**Site para simulados das certificações:**

Através do link:

<https://university.mongodb.com/certification/exam-prep>

Ou no site da university.mongodb clique em “Certification” depois em “Exam Prep”.

---------------------------------------------------

**4 -- 12. Utilizando o MongoDB no VSCODE**

**Instalar a extensão do Azure Databases:**

1. Abra o “Visual Studio Code”.
2. Clique no ícone “Extensions” na barra ao lado esquerdo ou pressione “Ctrl” + “Shift” + “X”.
3. Depois digite “Azure Databases” e clique sobre “install”.

---------------------------------------------------

**Criar a conexão com um banco de dados Mongo:**

1. Abra o “Visual Studio Code”.
2. Clique no ícone “Azure” na barra ao lado esquerdo ou pressione “Ctrl” + “Shift” + “A”.
3. Clique em “Attached Database Accounts”.
4. Clique em “Attached Database Account...”.
5. Selecione Mongo e pressione o “Enter”.
6. Informe a string de conexão e pressione o “Enter”. Caso tenha o MongoDB instalado o campo já vira preenchido com o “mongodb://127.0.0.1:27017” que é a URL do “localhost”.
7. Abaixo de “Attached Database Accounts” será exibida a conexão, ao clicar sobre ela são exibidos os bancos.

Obs: A instalação default utiliza as credenciais do Windows, na maquina de desenvolvimento isso não é um problema. Já na maquina de produção é necessário utilizar um usuário e senha.

---------------------------------------------------

**Criar uma Database no Mongo através do Visual Studio Code:**

1. Abra o “Visual Studio Code”.
2. Clique no ícone “Azure” na barra ao lado esquerdo ou pressione “Ctrl” + “Shift” + “A”.
3. Clique em “Attached Database Accounts”.

Clique sobre a conexão com o botão direito do mouse e selecione “Create Database...”.

1. Informe um nome e pressione o “Enter”.

Obs: Igual ao banco de dados no SQL.

---------------------------------------------------

**Criar uma Collection no Mongo através do Visual Studio Code:**

1. Abra o “Visual Studio Code”.
2. Clique no ícone “Azure” na barra ao lado esquerdo ou pressione “Ctrl” + “Shift” + “A”.
3. Clique em “Attached Database Accounts”.
4. Clique sobre a conexão. Serão exibidas todas as “Databases”.
5. Clique sobre a “Database” com o botão direito do mouse e selecione “Create Collection...”.
6. Informe um nome e pressione o “Enter”.

Obs: Igual a uma tabela no SQL.

---------------------------------------------------

**Criar um Document no Mongo através do Visual Studio Code:**

1. Abra o “Visual Studio Code”.
2. Clique no ícone “Azure” na barra ao lado esquerdo ou pressione “Ctrl” + “Shift” + “A”.
3. Clique em “Attached Database Accounts”.
4. Clique sobre a conexão. Serão exibidas todas as “Databases”.
5. Clique sobre a “Database”. Serão exibidas todas as “Collections”.
6. Clique sobre a “Collection” com o botão direito do mouse e selecione “Create Document...”.
7. Informe um nome e pressione o “Enter”.

Obs: Será gerado um arquivo com o “$oid” que é o identificador único gerado pelo Mongo.

Obs: Igual a uma (Linha / Registro) no SQL.

"\_id": {

    "$oid": "5fd156a844ef703d6cddb114"

  }

---------------------------------------------------

**Alterar um Document no Mongo através do Visual Studio Code:**

1. Abra o “Visual Studio Code”.
2. Clique no ícone “Azure” na barra ao lado esquerdo ou pressione “Ctrl” + “Shift” + “A”.
3. Clique em “Attached Database Accounts”.
4. Clique sobre a conexão. Serão exibidas todas as “Databases”.
5. Clique sobre a “Database”. Serão exibidas todas as “Collections”.
6. Clique sobre a “Collection”. Serão exibidos todos os “Documents”.
7. Clique sobre o “Document” e será aberta uma janela com o conteúdo do “Document”.
8. Altere o conteúdo e pressione o botão “Update to Cloud” que fica no canto superior direito para salvar as alterações no banco de dados.

---------------------------------------------------

**4 -- 13. Acessando o MongoDB Utilizando os Scrapbooks no VSCODE**

**Abrir o Scrapbooks no Visual Studio Code:**

1. Abra o “Visual Studio Code”.
2. Clique no ícone “Azure” na barra ao lado esquerdo ou pressione “Ctrl” + “Shift” + “A”.
3. Clique em “New Mongo Scrapbook” na barra superior.
4. Clique sobre “Connect to a database”.
5. Selecione “Attached Database Accounts” e pressione o “Enter”.
6. Selecione a “Account” e pressione o “Enter”.
7. Selecione uma “Database” e pressione o “Enter”.

---------------------------------------------------

**Exibir os dados de uma Collection através do Scrapbooks:**

Dentro do Scrapbook digite “db.[Nome da Collection].find({})” e depois clique em Execute.

db.Usuario.find({})

---------------------------------------------------

**Contar a quantidade de objetos dentro de uma Collection através do Scrapbooks:**

Dentro do Scrapbook digite “db.[Nome da Collection].count()” e depois clique em Execute.

db.Usuario.count()

Obs: Para descobrir todos os comandos basta digitar “db.[Nome da Collection].” e será exibido uma lista com todos os comandos.

---------------------------------------------------

**4 -- 14. Instalação do MongoDB + Compass no WINDOWS**

**Site para download do MongoDB:**

<https://www.mongodb.com/>

Obs: No site localize o “MongoDB Community Server” e realize o download.

---------------------------------------------------

**Instalando o MongoDB:**

1. Execute o arquivo “.msi” do “MongoDB Community Server” que você fez download.
2. Clique em “Next”.
3. Marque a caixa “I accept” e clique em “Next”.
4. Clique em “Complete”.
5. Habilite a opção “Install MongoD as a Service”.
6. Se quiser pode alterar o nome do Serviço e os diretórios do aplicativo.
7. Clique em “Next”.
8. Habilite a opção “Install MongoDB Compass” e clique em “Next”.
9. Clique em “Install”.

Obs: O “Install MongoDB Compass” é o equivalente ao “SQL Server Management Studio”, se você quiser apenas utilizar a parte gráfica e conectar com um banco em outra maquina basta instalar o “Install MongoDB Compass”.

---------------------------------------------------

**4 -- 15. Usando o MongoDB Compass – Parte I**

**Abrir a ferramenta MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.

---------------------------------------------------

**Criar uma Database através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clicar sobre o botão de “+” no lado inferior esquerdo ou sobre o botão “CREATE DATABASE” na barra superior.
5. Informe o nome da “Database”.
6. Informe o nome da “Collection”.
7. Clique sobre o botão “CREATE DATABASE”.

---------------------------------------------------

**Criar um Document através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clique sobre a “Database” e serão exibidas todas as “Collections”.
5. Clique sobre a “Collection”.
6. Clique sobre “Documents” na barra superior e serão exibidos todos os “Documents”.
7. Clique sobre o botão “ADD DATA” na barra superior.
8. Selecione “Insert Document”.
9. Insira o “JSON”.
10. Clique em “INSERT”.

Obs: Ao clicar em uma “Collection” serão exibidos todos os “Documents”, é possível alterar o tipo de exibição entre “Hierárquico”, “JSON” e “Tabular (Igual ao SQL)” através da barra superior.

Obs: Ao clicar em “Insert Document” será gerado um arquivo com o “$oid” que é o identificador único gerado pelo Mongo.

Obs: Ao inserir o “JSON” se alterar para “Lista” na barra superior, é mais fácil inserir um novo parâmetro e é possível informar o tipo de parâmetro.

---------------------------------------------------

**Editar um Document através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clique sobre a “Database” e serão exibidas todas as “Collections”.
5. Clique sobre a “Collection”.
6. Clique sobre “Documents” na barra superior e serão exibidos todos os “Documents”.
7. Clique sobre o “Document”.
8. Clique sobre o botão “Edit Document”.
9. Altere o “Document”.
10. Clique em “UPDATE”.

---------------------------------------------------

**Clonar um Document através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clique sobre a “Database” e serão exibidas todas as “Collections”.
5. Clique sobre a “Collection”.
6. Clique sobre “Documents” na barra superior e serão exibidos todos os “Documents”.
7. Clique sobre o “Document”.
8. Clique sobre o botão “Clone Document”.
9. Clique em “INSERT”.

---------------------------------------------------

**Filtrar os Documents exibidos através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clique sobre a “Database” e serão exibidas todas as “Collections”.
5. Clique sobre a “Collection”.
6. Clique sobre “Documents” na barra superior e serão exibidos todos os “Documents”.
7. Clique sobre “OPTIONS” na barra superior.
8. E serão exibidas as opções para filtrar os “Documents” exibidos.
9. Preencha as opções e clique em “FIND”.

**Opções de Filtro:**

* **FILTER:**

Nesse campo deve ser informado um objeto com os valores que deseja encontrar.

Ex: “{ nome:"Rafael" }”.

* **PROJECT:**

Nesse campo deve ser informado um objeto com os valores que deseja retornar na consulta. Informado “[Nome Propriedade]: Valor”. Valor “1” para exibir e “0” para ocultar.

Ex: “{ nome: 1, idade: 1 }” ou “{ nome: 0, }”.

* **SORT:**

Nesse campo deve ser informado um objeto com os valores que deseja ordenar a lista.

Informado “[Nome Propriedade]: Valor”. Valor “1” para crescente e “-1” para decrescente.

Ex: “{ nome: -1 }” ou “{ nome: 1 }”.

---------------------------------------------------

**Obter o resumo de uma consulta sobre a Collection através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clique sobre a “Database” e serão exibidas todas as “Collections”.
5. Clique sobre a “Collection”.
6. Clique sobre “Explain Plan” na barra superior.
7. Informar as opções de filtro para os “Documents”.
8. Clique sobre “Execute Explain”.

Obs: Muito importante por que retorna o tempo de execução de uma consulta com os parâmetros informados.

Obs: Será exibido um relatório com a quantidade de documentos, a quantidade de chaves examinadas, a quantidade de documentos examinados, o tempo de execução, o aviso caso a “Collection” não possua um index, etc...

---------------------------------------------------

**Criar um Index na Collection através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clique sobre a “Database” e serão exibidas todas as “Collections”.
5. Clique sobre a “Collection”.
6. Clique sobre “Indexes” na barra superior e serão exibidos todos os “Indexes”.
7. Clique sobre “CREATE INDEX”.
8. Informe um nome para o “INDEX”. Ex: “idxIdade”
9. Escolha a propriedade para o “INDEX”. Ex: “idade”.
10. Escolha o tipo de “INDEX”. Ex: “1 (asc)”.
11. Clique em “CREATE INDEX”.

Obs: O “Index” sobre o campo “\_id” é criado automaticamente pelo Mongo. Uma vez que para inserir o objeto ele tem que verificar se esse “\_id” já existe, tornando a inserção muito mais rápida.

---------------------------------------------------

**4 -- 16. Usando o MongoDB Compass – Parte II**

**Criar uma Collection através do MongoDB Compass:**

1. Abra a ferramenta “MongoDB Compass”.
2. Informar a string de conexão e clicar em “Connect”.
3. Caso consiga realizar a conexão serão exibidas todas as “Databases” do servidor.
4. Clique sobre a “Database” e serão exibidas todas as “Collections”.
5. Clique sobre o botão “+” do lado direito do nome da “Database” ou sobre o botão “CREATE COLLECTION” na barra superior.
6. Informe um nome para sua “Collection”.
7. Clique sobre o botão “CREATE COLLECTION”.

---------------------------------------------------

**4 -- 18. Instalação da Ferramenta ROBO3T no WINDOWS**

**Site para download do ROBO3T:**

<https://robomongo.org/download>

Obs: Muito bom porque ele tem um console para realizar operações, lembrando que o “MongoDB Compass” não tem console.

Obs: Não realizar o download do “Studio 3T” que é a versão paga. Realizar o download do “Robo 3T” que é a versão gratuita.

Obs: Ao executar o portable do “Robo 3T“ pela primeira vez ou instalar o “Robo 3T“ não insira o nome e e-mail apenas clique em “Finish”.

---------------------------------------------------

**Conectar o ROBO3T ao MongoDB:**

1. Abra a ferramenta “Robo 3T”.
2. Clique sobre o botão Create.
3. Informe o tipo de conexão. Ex: “Direct”
4. Informe o nome da conexão. Ex: “Teste”.
5. Informe o endereço. Ex: “localhost”.
6. Informe a porta. Ex: “27017”.
7. Clique em “Save”.
8. Clique sobre a conexão.
9. Clique sobreo botão “Connect”.

Obs: Os campos de endereço e porta já vem preenchidos com os valores padrão.

Obs: Após preencher todos os campos é possível clicar sobre o botão “Test” para verificar se é possível realizar uma conexão com as configurações informadas.

---------------------------------------------------

**4 -- 19. Conexão e Recursos na Ferramenta ROBO3T**

**Visualizar os Documents de uma Collection através do ROBO3T:**

1. Abra a ferramenta “Robo 3T”.
2. Clique sobre a conexão.
3. Clique sobreo botão “Connect” e serão exibidas todas as “Databases” na janela do lado esquerdo.
4. Clique sobre a “Database”.
5. Clique sobre “Collections” e serão exibidas todas as “Collections”.
6. Clique sobre a “Collection” com o botão direito e selecione “View Documents”.

Obs: É possível alterar o tipo de exibição entre “Hierárquico”, “Tabular (Igual ao SQL)” e “JSON” através da barra superior no lado direito.

Obs: Ao clicar sobre o “View Documents” na janela exibida aparece o console com o comando para exibir os dados da “Collection”, é possível alterar o comando passando parâmetros.

---------------------------------------------------

**4 -- 20. HANDS ON: A Primeira Manipulação de Dados via Console na Ferramenta ROBO3T**

**Abrir um console no ROBO3T:**

1. Abra a ferramenta “Robo 3T”.
2. Clique sobre a conexão.
3. Clique sobreo botão “Connect” e serão exibidas todas as “Databases” na janela do lado esquerdo.
4. Clique sobre a “Database” com o botão direito do mouse e selecione “Open Shell”.

Obs: Caso já tenha um console aberto também pode utilizar o atalho “Ctrl” + “T” para abrir outro console.

---------------------------------------------------

**Comandos para o console:**

* **Mostrar as Databases disponíveis no servidor**

No console digite o comando “show dbs” e execute o comando.

---------------------------------------------------

* **Usar uma determinada Database**

No console digite o comando “use [Nome da Database]” e execute o comando.

Ex: “use Teste”.

---------------------------------------------------

* **Criar uma Collection**

No console digite o comando “db.createCollection('[Nome da Collection]')” e execute o comando.

Ex: use Teste

Ex: db.createCollection('usuarios')

Obs: A “Collection” será criada na “Database” que esta em contexto.

---------------------------------------------------

* **Remover uma Database**

No console digite “db.dropDatabase()” e execute o comando.

Ex: use Teste

Ex: db.dropDatabase()

Obs: Se não executar a linha “use [Nome da Database]” junto com o comando “dropDatabase” será excluído o banco sobre o qual o console foi aberto. \*\*\* IMPORTANTE \*\*\* Sempre selecione as duas linhas “use” + “dropDatabase”.

---------------------------------------------------

* **Mostrar as Collections disponíveis na Database**

No console digite “show collections” e execute o comando.

Obs: Serão exibidas as Collections da “Database” que esta em contexto.

---------------------------------------------------

* **Remover todos os Documents de uma Collection**

No console digite “db.[Nome da Collection].remove({})” e execute o comando.

Ex: db.Usuario.remove({})

Obs: Muito cuidado ao utilizar o remove passando um objeto vazio, equivale ao Delete no SQL sem o Where.

---------------------------------------------------

* **Remover uma Collection da Database**

No console digite “db.[Nome da Collection]. drop ()” e execute o comando.

Ex: db.Usuario.drop()

---------------------------------------------------

Obs: “db” é a variável que aponta para o banco que esta em contexto.

Obs: Muito cuidado ao utilizar o comando “drop” para não remover a “Database” ou “Collection” erada.

Obs: Para executar qualquer comando basta selecionar as linhas e clicar sobre o botão “Execute Query” ou pressionar o botão “F5” ou “Ctrl” + “Enter”.

---------------------------------------------------

**5 -- 23. Transações Distribuídas**

**Uso de transações:**

O MongoDB recentemente incluiu o controle transacional.

Obs: O controle transacional deve ser utilizado apenas em uma parte especifica do código, aonde é necessário uma consistência mais rígida dos dados.Lembrando que ao utilizar o uso de transações, você ira perder desempenho.

---------------------------------------------------

**Uso de transações em maquinas distribuídas:**

Nas novas versões o MongoDB disponibilizou o uso de transações em maquinas distribuídas.

---------------------------------------------------

**5 -- 24. Chaves de Particionamento Mutáveis**

**Uso de Shard Keys Mutaveis:**

“Shard Keys” é a chave criada para particionar os dados, ou seja responsável por indicar em qual maquina aquele dado estará armazenado.

Ex: É possível criar uma “Shard Key” sobre a coluna “Estado” das notas fiscais, toda vez que inserir uma nova nota fiscal e o “Estado” for São Paulo os dados vão ser armazenadas em um servidor e se for do Rio de Janeiro em outro servidor.

Obs: Nas novas versões do MongoDB é possível atualizar o valor da chave “Shard Keys”. Antes não era possível era necessário apagar o dado e inserir novamente.

---------------------------------------------------

**5 -- 25. Evolução Constante dos Operadores de Agregação**

**Operador Lookup:**

Nas novas versões o MongoDB sempre vai disponibilizando novos operadores de agregação como o “sum” ,“lookup”, etc...

O “sum” funciona como o sum do SQL.

O “lookup” funciona como uma junção do SQL.

Obs: Vale lembrar que os bancos no SQL foram deixando de lado varias dessas funções para conseguir aumentar a produtividade e desempenho.

---------------------------------------------------

**5 -- 27. Visões Materializadas no MongoDB**

**Visões materializadas:**

Uma “Visão” é uma camada de abstração que permite realizar algumas operações antes de retornar o dado. Ex: Realizando uma junção entre varias tabelas para retornar um cliente ativo.

Uma “Visão materializada” é o mesmo que uma “Visão”, só que após a primeira execução os dados ficam armazenados em tabelas temporárias, porque caso os dados precisem ser consultados novamente o banco apenas retorna os registros da tabela temporária e não precisa realizar toda a consulta novamente.

Obs: Você pode criar uma “Visão materializada” através do “$merge”, com ele é possível obter o resultado de uma consulta e inserir ou atualizar o dado em outra “Collection”.

---------------------------------------------------

**5 -- 28. Índices Wildcard**

**Índices Wildcard:**

Nas novas versões o MongoDB permite criar índices “Wildcard” que permite indexar atributos e sub atributos. Ex: Endereço que dentro tem endereço residencial e endereço comercial.

Obs: O índice pode tornar a inserção e a atualização mais lenta.

---------------------------------------------------

**6 -- 30. Conheça a Anatomia de um Documento Armazenado no MongoDB**

**Document:**

No MongoDB um “Document” sempre começa com “{”, possui uma lista de atributos e valores separados por “,” e termina com “}”.

Ex:

{

“\_id”: “ObjectId(“123456456”)”

“nome”: “Teste”,

“idade”: 25

}

Obs: Caso não especifique o valor do atributo “\_id” o MongoDB vai gerar um “ObjectId” para identificar de forma única esse objeto. Caso você informe o valor do atributo “\_id” o MongoDB não faz nada.

Obs: O valor de um atributo pode ser um tipo primitivo, um “Array” ou outro “Document”.

{

“\_id”: “ObjectId(“123456456”)”

“nome”: “Teste”, **-- Tipo Primitivo**

“hobby”: [“musica”, “futebol”], **-- Array**

“recomendacoes” : { **-- Documento embutido**

“cor”: “azul”,

“esporte”: “futebol”,

}

}

---------------------------------------------------

**6 -- 31. Arrays e Documentos Aninhados (Nested Documents) no MongoDB**

**Array:**

No MongoDB um atributo sempre começa com “[”, possui uma lista de valores separados por “,” e termina com “]”.

{

“hobby”: [ “Futebol”, “Musica”],

}

Obs: Principal diferença entre “Array” e “Document”, é que o “Array” possui uma lista de valores, enquanto o “Document” possui uma lista de atributos e valores.

---------------------------------------------------

**6 --32. Resumo das Principais Características de um Documento no MongoDB**

**Documento embutido ou documento aninhado ou sub documentos:**

É um objeto que possui um atributo com valor de outro documento.

Ex:

{

“\_id”: “ObjectId(“123456456”)”

“recomendacoes” : {

“cor”: “azul”,

“esporte”: “futebol”,

}

}

Obs: Muito importante porque já salva o objeto completo, em um banco de dados relacional seria necessário utilizar uma junção para obter o objeto completo.

---------------------------------------------------

**7 -- 34. Visão Geral dos Tipos de Dados no MongoDB**

**Tipos de Dados:**

Todos os tipos de dados são dinâmicos. Ex: Em um “Document” a idade pode ser um inteiro e em outro “Document” a idade pode ser um texto.

O MongoDB suporta todos os tipos de dados utilizados no BSON. Entre eles estão:

**String:** Utilizado para armazenar valor texto.

**Inteiro:** Utilizado para armazenar valor numérico. Pode ser de 32 ou 64 bits.

**Booleano:** Utilizado para armazenar verdadeiro ou falso.

**Double:** Utilizado para armazenar valor de ponto flutuante.

**Min / Max Keys:** Utilizado para comparar valor com os elementos BSON maior e menor.

**Arrays:** Utilizado para armazenar vários valores em uma chave.

**Timestamp:** Utilizado para armazenar a data e hora.

**Object:** Utilizado para armazenar documentos aninhados.

**Null:** Utilizado para armazenar valor nulo.

**Symbol:** Utilizado de forma idêntica a “string”, no entanto geralmente é reservado para idiomas que utilizam um tipo de símbolo especifico.

**Date:** Utilizado para armazenar a data e hora atual no formato de hora UNIX. Você pode especificar sua própria data e hora.

**Object ID:** Utilizado para armazenar o ID do documento.

**Binady Data:** Utilizado para armazenar dados binários.

**Code:** Utilizado para armazenar código JavaScript no “Document”.

**Regular Expression:** Utilizado para armazenar expressões regulares.

---------------------------------------------------

**7 -- 35. HANDS ON: Tipos de Dados no MongoDB - Parte I**

**Abrir um console para inserir comandos no MongoDB através do Prompt de Comando:**

1. Abra o “Prompt de Comando”.
2. Através do comando “cd” navegue ate a pasta “C:\Program Files\MongoDB\Server\4.4\bin”.
3. Digite “mongo” e pressione o “Enter”.

Obs: É possível realizar os mesmos passos através do terminal do Visual Studio Code.

---------------------------------------------------

**Criar uma variável com valor Doble no console:**

Após abrir o console através do “Prompt de Comando” ou “Visual Studio Code”, digite:

var valorDouble=1100.07;

Obs: Para exibir o valor da variável basta digitar o nome da variável.

valorDouble;

---------------------------------------------------

**Inserir um “Document” com uma String no Mongo**

Após abrir o console através do “Prompt de Comando” ou “Visual Studio Code”, digite:

db.collectionString.insert({ Registro: "Tipo String no Mongo" });

Obs: Para visualizar o dado digite:

db.collectionString.find();

Obs: Utilize o comando “use [Nome Database]” para selecionar a “Database”.

Se você apontar para uma “Database” ou uma “Collection” que não existe, ao inserir um dado em uma “Collection” o MongoDB vai criar automaticamente a “Database” e a “Collection”.

---------------------------------------------------

**Inserir um “Document” com um objeto no Mongo**

Após abrir o console através do “Prompt de Comando” ou “Visual Studio Code”, digite:

var localizacao = {

endereco: "avenida teste",

numero: 50,

complemento: "sala 928" };

db.collectionObjeto.insert( { empresa: "Teste", ramo: "Teste", local: localizacao } );

Obs: Para visualizar o dado digite:

db.collectionObjeto.find();

---------------------------------------------------

**Inserir um “Document” com um Array no Mongo**

Após abrir o console através do “Prompt de Comando” ou “Visual Studio Code”, digite:

var cidadesBrasil = [ "São Paulo", "Rio de Janeiro", "Amapa"];

var cidadesMexico = [ "Guadalajara", "Tijuana", "Cancun" ];

db.collectionArray.insert( { cidades1: cidadesBrasil, cidades2: cidadesMexico } );

Obs: Para visualizar o dado digite:

db. collectionArray.find();

---------------------------------------------------

**8 -- 37. Inserção de Dados**

**Comandos no MongoDB:**

Todos os comandos podem ser utilizados abrindo o console através do “Prompt de Comando” ou “Visual Studio Code”.

Os comandos aceitam expressões regulares nas “Querys”.

* **Insert**

Para inserir um “Document” na “Collection”. Utilize o comando “db.[Nome da Collecion].insert([Document])”.

Ex: “db.funcionario.insert({ primeiro: "Rafael", ultimo: "Silva", idade:33 })”

Obs: Caso a “Collection” não exista será criada uma.

---------------------------------------------------

* **Save**

Para atualizar ou inserir um “Document” na “Collection”. Utilize o comando “db.[Nome da Collecion].save([Document])”.

Obs: Caso seja passado o “\_id” no “Document” como parâmetro e ele já exista o dado será atualizado, caso contrario o dado será inserido.

---------------------------------------------------

* **Update**

Para atualizar um “Document” da “Collection”. Utilize o comando “db.[Nome da Collecion].update([Query], [Update], [Options])”.

Ex: db.funcionario.update(

{ "\_id" : ObjectId("5fdbddba07c46cca368fa952") },

{

$set: { idade: 32, salario: 1000 }

})

**Query:** Objeto com as condições para a atualização. Igual ao “WHERE” do SQL.

Ex: { idade: 30 }

**Update:** Objeto com os dados que serão atualizados. Igual ao “SET” do SQL.

Ex: { salario: 1000 }

Obs: No “Update” ao utilizar o “$set” será inserido os novos parâmetros, atualizado os parâmetros que já existem e mantem os parâmetros antigos, se passar um novo “Document” como parâmetro remove todo o conteúdo antigo.

**Options:** Objeto com os dados de opções. Existem varias opções.

Ex: { upsert: true }

Obs: Caso seja informado o “upsert” e o valor seja “true” e não exista nenhum objeto com a condição informada ele insere um novo objeto na “Collection”.

---------------------------------------------------

* **Remove**

Para remover os “Documents” da “Collection”. Utilize o comando “db.[Nome da Collecion].remove([Query], [JustOne])”.

Ex: db.funcionario.remove({ "nome": /^R/})

**Query:** Objeto com as condições para remover o dado. Igual ao “WHERE” do SQL.

Ex: { idade: 30 }

**JustOne:**

---------------------------------------------------

* **Find**

Para exibir os “Documents” da “Collection”. Utilize o comando “db.[Nome da Collecion].find([Query], [Projection])”.

Ex: db.funcionario.find({ idade:33 }, { primeiro: 1, ultimo: 1 })

**Query:** Objeto com as condições para exibir o dado. Igual ao “WHERE” do SQL.

Ex: { idade: 30 }

**Projection:** Objeto com os parâmetros que serão retornados. Igual ao “\*” do SQL.

Ex: { idade:1 }

Obs: Permiti utilizar paginação.

---------------------------------------------------

* **FindOne**

Para exibir um “Document” da “Collection”. Utilize o comando “db.[Nome da Collecion].findOne([Query], [Projection])”.

Ex: db.postagens.findOne({\_id: ObjectId("5fdd0ce266c79f48e8fff4ac")});

**Query:** Objeto com as condições para exibir o dado. Igual ao “WHERE” do SQL.

Ex: { idade: 30 }

**Projection:** Objeto com os parâmetros que serão retornados. Igual ao “\*” do SQL.

Ex: { idade:1 }

Obs: Bastante utilizado passando como parâmetro um identificador único por exemplo, passando o “\_id” como parâmetro na “Query”.

---------------------------------------------------

**8 -- 41. HANDS ON: Inserção e Consulta de Dados - Parte I**

**Exibir dados como JSON no console:**

Para exibir os “Documents” como arquivos JSON através do console utilize o método “pretty”.

Ex: db.postagens.find().pretty();

Retorno:

{

"\_id" : ObjectId("5fdd0ce266c79f48e8fff4ac"),

"titulo" : "Primeira Postagem",

"conteudo" : "Conteudo 01",

"tags" : [ ]

}

Obs: Através da ferramenta “Robo3T” não é possível ver diferença no retorno, uma vez que a ferramenta já trata o retorno das “Querys”.

---------------------------------------------------

**Limitando a quantidade de registros retornados:**

Para limitar a quantidade de “Documents” retornados utilize o método “limit”.

Ex: db.postagens.find().limit(2).pretty();

---------------------------------------------------

**Ordenando retorno dos registros:**

Para ordenar os “Documents” retornados utilize o método “sort”.

Ex: db.postagens.find().sort({titulo: 1});

---------------------------------------------------

**Concatenando comandos:**

É possível concatenar vários comandos para obter um dado.

Ex: db.postagens.find().sort({titulo: 1}).limit(2);

---------------------------------------------------

**8 -- 42. HANDS ON: Inserção e Consulta de Dados - Parte II**

**Removendo o \_id do retorno da Query:**

Por padrão, o MongoDB retorna o atributo "\_id" nas consultas, caso não queira exibir este atributo basta adicionar o atributo "\_id" na projeção com o seu valor para "false".

Ex: db.postagens.find({}, {titulo: true, \_id: false}).sort({titulo: 1})

---------------------------------------------------

**8 -- 43. HANDS ON: Inserção e Consulta de Dados no MongoDB - Parte III**

**Condição IN:**

Utilize o “$in” para verificar se o titulo esta entre as opções informadas.

Ex: db.postagens.find(

{

titulo: { $in: ["Primeira Postagem", "Segunda Postagem"]}

});

---------------------------------------------------

**Operador AND**

Utilize a “,”para verificar se o objeto corresponde a ambas as alternativas.

Ex: db.postagens.find( { titulo: "Primeira Postagem", conteudo: "Conteudo 01" } );

---------------------------------------------------

**Operador OR**

Utilize o “$or” para verificar se o objeto corresponde a uma das alternativas.

Ex: db.postagens.find(

{

$or: [ { titulo: "Primeira Postagem" }, { titulo: "Segunda Postagem" } ]

} );

---------------------------------------------------

**Condição Menor Que**

Utilize o “$lt” para verificar se o valor é menor que.

Ex: db.postagens.find( { quantidade: { $lt: 20}} );

---------------------------------------------------

**Condição Maior Que**

Utilize o “$gt” para verificar se o valor é maior que.

Ex: db.postagens.find( { quantidade: { $gt: 2}} );

---------------------------------------------------

**Retornar a quantidade de “Documents”**

Utilize o “count” para verificar a quantidade de registros.

Ex: db.postagens.count();

---------------------------------------------------

**Consultar atributo de um Documento embutido**

Utilize “[Nome atributo].[Nome atributo Documento embutido]”.

Ex: db.postagens.find( { "redeSocial.nome": "Facebook" });

---------------------------------------------------