**Lista com os comandos MongoDB para começar**

**Link para documentação MongoDB:** <https://www.mongodb.com/docs/manual/>

1. Listar todos os bancos
   1. show dbs ou show databases
   2. Para selecionar algumas “colunas” podemos usar o seguinte comando:
2. Criar um banco
   1. Não há um comando para criar um banco no comand line do mongo. Porém podemos usar o comando **use nome\_do\_banco**;
   2. O banco foi criado, mas só passará a existir quando inserirmos alguma coisa nele. Se dermos o comando **show dbs**; ele não irá aparecer
   3. Então para criar um banco teremos que criar uma “**coleção**”(collection), são como tabelas no sql, mas que na realidade são grupos de documentos.
   4. Podemos criar uma coleção assim:
      1. **db.createCollection(name, option)**
      2. exemplo:**db.createCollection(“produto”)**
   5. Ou podemos criar da seguinte forma: **db.produto.insertOne({nome:”farinha”, preco:4.45})**
3. Apagar banco de dados e tabela
   1. Apagar banco: **db.dropDatabase()**
   2. Apagar coleção: **db.colecao.drop()**
4. Como consultar dados na tabela:
   1. db.nome\_coleção.find(). Exemplo: **db.produto.find()**
   2. Pra exibir de um modo formatado: **db.produto.find().pretty()**
5. Como consultar exibindo “colunas” específicas:
   1. **db.colecao.find({}, { campo1: 1, campo2: 1, \_id: 0 })** Neste exemplo, o primeiro argumento para find é um objeto de filtro vazio {}, o que significa que a consulta retornará todos os documentos na coleção. O segundo argumento é a projeção de campos, onde { campo1: 1, campo2: 1, \_id: 0 } significa que os campos "campo1" e "campo2" serão exibidos, mas o campo "\_id" será excluído dos resultados
6. Contar a quantidade de documentos “registros” que uma coleção possui:
   1. db.coleção.countDocuments() // irá retornar um valor numérico com o total de documentos contabilizados
7. Consultar com filtros semelhante ao Where
   1. **Igualdade:** db.produto.find({nome:”farinha”})
   2. **Menor que:** db.produto.find({estoque:{$lt:30}}) //$ls significa less than
   3. **Maior que:** db.produto.find({estoque:{$gt:30}}) // $gt significa greater than
   4. **Menor ou igual:** db.produto.find({estoque:{$lte:30}}) // $lte significa less than or equal
   5. **Maior ou igual:** db.produto.find({estoque:{$gte:30}})
8. Inserir valores
   1. Inserir muitos valores: db.insertMany([

{nome:”cafe”, preco:2.12},

{nome:”arroz”, preco:4.12}

]);

* 1. Inserir apenas 1 valor: **db.produto.insertOne({nome:”farinha”, preco:4.45})**

1. Atualizar informação
   1. Atualizar 1 informação:
      1. **db.students.updateOne( { id: 3 }, [ { $set: { "nota": 7.5 } } ] )**
      2. Atualizando mais de um campo: **db.students.updateOne( { id: 3 }, [ { $set: { "nota": 7.5, idade:13 } } ] )**
2. Adicionar um campo a um documento já existente:
   1. **db.coleção.updateOne({id:3}, [{$set:{novoCampo:valor}}])** // no 1º argumento {id:3}, você pode usar o **\_id:ObjectId()** mas como os valores são grandes, uma dica é selecionar todo o campo **\_id** e clicar com o botão direito do mouse, os dados serão copiados, em seguida é só clicar novamente com botão direito onde se deseja colar
3. Excluir documentos:
   1. db.coleção.deleteOne({campo:valor}) // irá remover o documento onde o **campo** seja **igual** ao **valor**
   2. db.coleção.deleteOne({}) // Irá excluir o 1º documento
   3. db.coleção.deleteMany({}) // Irá excluir o todos os documentos