Exercícios:

I. Coleções no MongoDB;

II. Inserir documentos na coleção;

III. Ler documentos da coleção;

No MongoDB, os dados são organizados hierarquicamente da seguinte forma:

- BD:
  - É a unidade mais alta de armazenamento de dados, análogo a um BD no SGBD-R;
  - No MongoDB podem existir vários bancos e eles são independentes entre si;
  - Um BD pode conter várias coleções.
- Coleção (collection):
  - Uma coleção é análoga a uma tabela no SGBD-R;
  - Uma coleção é um grupo de documentos;
  - As tabelas de um SGBD-R definem um esquema fixo para todos os registros, mas as coleções não impõem um esquema fixo aos documentos.
- Documento (document):
  - Um documento é uma unidade básica de dados no MongoDB e é representado em formato BSON (Binary JSON -formato binário JSON-like);
  - Um documento é análogo a uma linha (registro) de uma tabela do SGBD-R;
  - Enquanto todos os registros de uma tabela no SGBD-R compartilham o mesmo esquema (estrutura de colunas), os documentos de uma coleção não compartilham o mesmo esquema (propriedades do JSON). Desta forma, não se tem um esquema fixo;
  - Um documento no MongoDB é composto por campos e valores. Cada campo contém um valor associado, e estes campos podem ser de diferentes tipos de dados, como strings, inteiros, arrays, objetos, entre outros.

i. Exercícios 1

- Crie um banco de dados com o nome bdaula. O comando `use` criará o BD se ele não existir.
  - Comando: **use bdaula**
- Liste os bancos de dados. O comando `show databases` lista os BDs no MongoDB.
  - Comando: **show databases**

ii. Exercícios 2

- Crie uma coleção chamada `users`. Faça a inserção de um documento com os campos: `nome`, `idade` e `genero`. Use o comando para inserir apenas os dados de um aluno.

users
_id: ObjectId
nome: string
idade: number
genero: string

  - Comando: `db.users.insertOne`

```
db.users.insertOne({
  nome: "Maria",idade: 21,genero: "F"
})
```

- Novamente, liste os bancos de dados. Verifique se está visualizando o bd criado.

- Liste as coleções do bd atual bdaula. Os comandos `show tables` e `show collections` são usados para listar as coleções do BD atual.
    - Comando: **show tables**

iii. Exercício 3

- Remova a coleção `users`. O método **drop** é usado para remover a coleção do BD.
    - **Comando**: db.users.drop()
- Use o comando para visualizar as coleções em bdaula.
    - **Comando**: show collections
- Faça a inserção dos documentos na coleção `users`. O método `insertMany()` recebe um array onde cada elemento é um objeto JSON com os campos e valores do documento a ser inserido.
    - **Comando**: `db.users.insertMany`

```
db.users.insertMany([
  {nome: "Pedro", idade: 25, genero: "M"},
  {nome: "Ana", idade: 20, genero: "F"},
  {nome: "Maria", idade: 21, genero: "F"},
  {nome: "Lucas", idade: 28, genero: "M"},
  {nome: "João", idade: 22, genero: "M"},
  {nome: "Renata", idade: 24, genero: "F"},
  {nome: "Paulo", idade: 23, genero: "M"},
  {nome: "Bruna", idade: 27, genero: "F"},
  {nome: "Irene", idade: 20, genero: "F"},
  {nome: "Yuri", genero: "M"}
])
```

iv. Exercício 4

- Faça uma consulta que retorne todos os documentos da coleção `users`.
    - O método `find` seleciona documentos em uma coleção.

      Comando: `db.users.find()`
- Faça uma consulta que retorne um array com todos os documentos da coleção `users` onde o campo `genero` é igual a `F` **e** campo `idade` é igual a `20`.
    - O método `find` recebe um objeto JSON com os critérios de filtragem.
    - **Comando**:

```
db.users.find({
  genero:"F", idade:20
})
```

- Faça uma consulta que retorne somente o campo `nome`.

- O método `find()` faz **consulta com projeção de campos** retornando um array com todos os documentos da coleção `users`, mas mostrará somente o campo especificado. O método `find` recebe um objeto JSON com os critérios de filtragem.
- **Comando:**

```
db.users.find(
  {},{ nome:true, _id:false }
)
```

- Faça uma consulta que retorne um array com documentos da coleção `users` ordenados pelo campo `idade`, ignorando os 2 primeiros documentos e mostrando 4 documentos.
  - Para fazer **consulta com opções** entre outras opções temos: **sort** (ordenação), **limit** (número de documentos no resultado) e **skip** (salta um número especificado de documentos antes de começar a retornar os resultados).
  - Comando:

```
db.users.find(
{},{  nome:true,  idade:true,  _id:false },
{  sort:{ idade:1 },  skip: 2,  limit: 4 }
)
```

- Faça uma consulta que retorne um array com documentos da coleção `users` ordenados pelo campo `idade`, ignorando os 2 primeiros documentos e mostrando 4 documentos.
  - **Comando**:

```
db.users.find(
{},{  nome:true,  idade:true,  _id:false },
{  sort:{ idade:1 },  skip: 2,  limit: 4 }
)
```

- Faça uma consulta que retorne o 1º documento da coleção.
  - O método `findOne` retorna apenas o 1º documento selecionado pelos critérios da consulta.
  - **Comando**:

```
db.users.findOne()
```

- Faça uma consulta que retorne o 1º documento da coleção `users` que o `gênero` é `F` e a `idade` seja `20`.
  - O método `findOne` retorna apenas o 1º documento selecionado pelos critérios da consulta e com projeção de campos.
  - **Comando**:

```
db.users.findOne(
{genero:"F", idade:20 },
{ nome:true, _id:false }
)
```