Descrição da atividade: fazer os comandos para listar dados de uma coleção do MongoDB.

Nota: 0,25 pts. na média final.

Data de entrega: na aula de 09/maio.

Forma de entrega: individual e presencial.

Objetivos:

- Modelo de dados incorporados;
- Consultas usando os métodos find e aggregate do MongoDB.

Sobre os dados: Banco de Dados Meteorológicos do INMET – os dados foram baixados de https://portal.inmet.gov.br/dadoshistoricos, uma breve explicação se encontra em https://bdmep.inmet.gov.br. Foram baixados os dados das estações de coletas de dados meteorológicos dos estados do Espírito Santo, Rio de Janeiro e São Paulo do período de 01/01/2024 a 31/03/2024.

Passos para carregar os dados:

1. Descompacte o arquivo.zip no local de seu interesse do computador. Evite descompactar no desktop (área de trabalho), visto que precisaremos endereçar esse arquivo no comando para importar o JSON numa coleção do MongoDB;
2. Ignore esse passo se você já tiver instalado o programa mongoimport.
   Normalmente os programas mongoexport e mongoimport não são instalados com o MongoBD, então precisaremos instalar eles no nosso computador.
   Acesse MongoDB Command Line Database Tools Download (https://www.mongodb.com/try/download/database-tools) e faça o download do pacote MSI. Sugere-se instalar no caminho sugerido pelo instalador, que provavelmente seja na mesma pasta onde se encontra o MongoDB Server;
3. Os programas estão instalados no caminho /MongoDB/Tools/100/bin.
   No CMD (prompt de comando) acesse a pasta onde se encontra o programa mongoimport e submeta o comando a seguir. Lembre-se antes de colocar o caminho correto para o arquivo.json:

```
mongoimport ^
    --uri "mongodb://127.0.0.1:27017/bdmeteorologico" ^
    --collection "estacoes" ^
    --file "C:\pasta\pasta\arquivo.json" ^
    --jsonArray
```

Em caso de sucesso o comando exibirá a seguinte resposta dizendo que foram importados 78 documentos.

```
2024-04-30T17:22:09.814-0300    connected to: mongodb://127.0.0.1:27017/bdmeteorologico
2024-04-30T17:22:11.280-0300    78 document(s) imported successfully. 0 document(s) failed to import.
```

Observações:

- Será criado o `bdmeteorologico` se ele não existir;
- O símbolo de circunflexo (^) quebra o comando em várias linhas no terminal do CMD para facilitar a digitação;
- Curiosidade: o `arquivo.json` foi gerado usando o seguinte comando, ou seja, os documentos da coleção `dados` foram exportados para o `arquivo.json`:

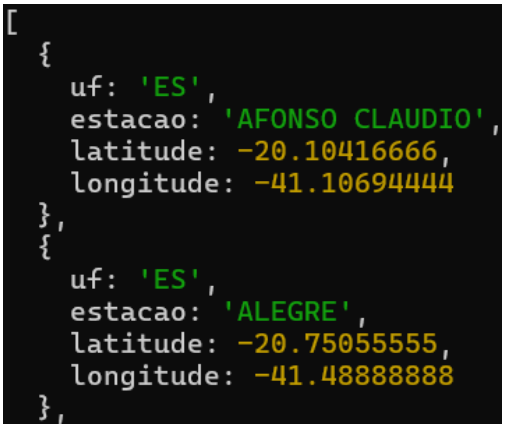
```
mongoexport ^
    --uri "mongodb://127.0.0.1:27017/bdmeteorologico" ^
    --collection "dados" ^
    --out "C:\pasta\pasta\arquivo.json" ^
    --jsonArray
```

4. Execute os comandos a seguir no MongoDB Shell para verificar se os dados foram carregados corretamente na coleção `estacoes` do `bdmeteorologico`;

```
bdmeteorologico> show tables
estacoes
bdmeteorologico> db.estacoes.countDocuments()
78
```

Exercícios

1) Fazer o comando para listar os campos `uf`, `estacao`, `latitude` e `longitude` de todas as estações. Apresente o resultado ordenado pelo campo `estacao`.
O resultado será um array com 78 documentos.
Dicas:
   - Use o método `find(query, projection, options);`
   - No parâmetro `projection` forneça os campos `uf`, `estacao`, `latitude` e `longitude`, e retire o campo obrigatório `_id`;
   - Forneça a propriedade `sort` no objeto do parâmetro `options`.



2) Fazer o comando para listar a quantidade de leituras por estação. Apresente o resultado ordenado pelo campo `estacao`. Apresente o resultado ordenado pelo campo `estacao`.
O resultado será um array com 78 documentos.
Dicas:

- Use o método `aggregate([{$project}, {$sort}])` com os estágios `$project` e `$sort`. Cada estágio precisa estar em um objeto, ou seja, delimitados por chaves;
- No estágio `$project`, use o operador `$size` para obter a quantidade de elementos do array que está no campo `leituras`;
- No estágio `$sort`, use o campo `estacao` para ordenar o resultado.

  Para mais detalhes sobre o operador $size acesse https://www.mongodb.com/docs/manual/reference/operator/aggregation/size.

```
[
  { estacao: 'AFONSO CLAUDIO', quantidade: 2184 },
  { estacao: 'ALEGRE', quantidade: 2184 },
  { estacao: 'ALFREDO CHAVES', quantidade: 2184 },
  { estacao: 'ANGRA DOS REIS', quantidade: 2184 },
```

3) Fazer o comando para listar a média, a mínima e a máxima do campo `temperaturaAr` das leituras da estação de `VITORIA`.
O resultado será um array com 1 documento.
Dicas:

```
[
  {
    media: 27.426597582037996,
    minima: 21.5,
    maxima: 37.9,
    estacao: 'VITORIA'
  }
]
```

- Use o método `aggregate([{$match},{$unwind},{$group},{$project}]);`
- No estágio `$match` use a condição `estacao:NomeDaEstacao`;
- Forneça o campo `"$leituras"` para o estágio `$unwind`. Para calcular a média do campo `temperaturaAr` dos subdocumentos que estão no campo `leituras`, foi necessário usar o operador `$unwind` para "desconstruir" o array `leituras` em documentos individuais antes de calcular a média;
- No estágio `$group`,
  - Será necessário agrupar pelo campo `estacao`;
  - Use os operadores `$avg`, `$min` e `$max` para obter, respectivamente, a média, o mínimo e o máximo no campo `"$leituras.temperaturaAr"`.

4) Fazer o comando para listar as leituras de `temperaturaAr` no intervalo [`ISODate("2024-01-02T00:00:00.000Z"), ISODate("2024-01-03T23:00:00.000Z")`].
O resultado será um array com 1 objeto e dentro desse objeto deverá ter as propriedades leituras e estacao.
A propriedade leituras do resultado será um array com 48 documentos.

```
[
  {
    leituras: [
      { leitura: 23, data: ISODate('2024-01-02T00:00:00.000Z') },
      { leitura: 23.2, data: ISODate('2024-01-02T01:00:00.000Z') },
      { leitura: 23, data: ISODate('2024-01-02T02:00:00.000Z') },
      { leitura: 23, data: ISODate('2024-01-02T03:00:00.000Z') },
      { leitura: 23, data: ISODate('2024-01-02T04:00:00.000Z') },
      ...
      { leitura: 27, data: ISODate('2024-01-03T20:00:00.000Z') },
      { leitura: 26.4, data: ISODate('2024-01-03T21:00:00.000Z') },
      { leitura: 25.9, data: ISODate('2024-01-03T22:00:00.000Z') },
      { leitura: 26.1, data: ISODate('2024-01-03T23:00:00.000Z') }
    ],
    estacao: 'VITORIA'
  }
]
```

Dicas:

- Use o método `aggregate([{$unwind},{$match},{$group},{$project}]);`
- No estágio `$unwind` forneça o campo `"$leituras"` para descontruir o array de leituras;
- No estágio `$match` use a condição

```
leituras.datahora:{
    $gte: ISODate("2024-01-02T00:00:00.000Z"),
    $lte: ISODate("2024-01-03T23:00:00.000Z")
}
```

- No estágio `$group`:
  - Será necessário agrupar pelo campo `estacao`;
  - Crie o campo `leituras` com o operador `$push` para concatenar as respostas em um array.

    Esse recurso criará a série temporal de `leituras` com as propriedades `leitura` e `data`:

```
leituras: {
  $push: {
    leitura: "$leituras.temperaturaAr",
    data: "$leituras.datahora",
  },
},
```

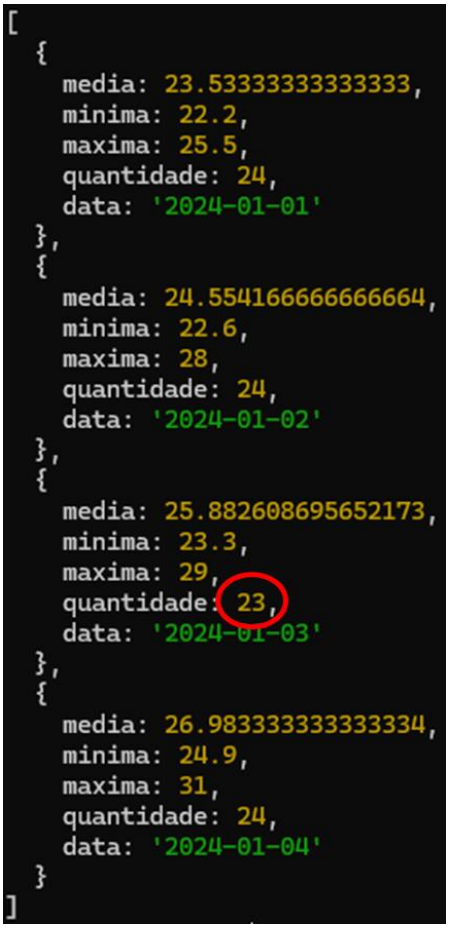
5) Fazer o comando para listar as leituras de `temperaturaAr` da estação de `VITORIA` agrupadas por dia.

Apresente a média, a mínima, máxima e quantidade de valores do campo `temperaturaAr` por dia.

Apresente o resultado ordenado pela data e limitado aos 4 primeiros documentos do resultado.

Observe que dia 2024-01-03 tem apenas 23 documentos com o campo `temperaturaAr`.

O resultado será um array com 4 documentos.



Dicas:

- Use o método `aggregate([{$unwind}, {$match}, {$project}, {$group},{$project}, {$sort}, {$limit}])`. Observe que será necessário usar o estágio `$project` duas vezes;
- No 1º $project: projete o campo `dia` extraindo apenas o dia da `datahora`. O resultado desse estágio será projetado para o próximo estágio:

```
{
  $project: {
     // Extrai o dia da datahora
     dia: { $dateToString: { format: "%Y-%m-%d", date: "$leituras.datahora" } },
     temperaturaAr: "$leituras.temperaturaAr"
  }
},
```

- No 2º $project serão projetados os campos para o resultado.