

Bootcamp: Analista de Banco de Dados

Desafio do módulo

Módulo 3	Banco de Dados NoSQL
-----------------	-----------------------------

Objetivos

Exercitar os conceitos vistos em aulas em relação ao banco de dados NoSQL MongoDB. A partir de dados da Força Aérea Brasileira sobre a aviação civil Brasileira (CENIPA - Ocorrências Aeronáuticas na Aviação Civil Brasileira), vamos importar algumas informações no MongoDB para executar análises.

Enunciado

A base de dados de ocorrências aeronáuticas é gerenciada pelo Centro de Investigação e Prevenção de Acidentes Aeronáuticos (CENIPA). Constam nesta base de dados as ocorrências aeronáuticas notificadas ao CENIPA nos últimos 10 anos que ocorreram em solo brasileiro.

Dentre as informações disponíveis estão os dados sobre as aeronaves envolvidas, fatalidades, local, data, horário dos eventos e informações taxonômicas típicas das investigações de acidentes (AIG).

Arquivos com os quais trabalharemos:

- Ocorrenci.csv - Informações sobre as ocorrências.
- Ocorrenci_tipo.csv - Informações sobre o tipo de ocorrência.
- Aeronave.csv - Informações sobre as aeronaves envolvidas nas ocorrências.

Fonte: Sistema DÉDALO.

<https://dados.gov.br/dataset/ocorrencias-aeronauticas-da-aviacao-civil-brasileira>

Alguns ajustes foram executados para facilitar nosso estudo, tais como eliminação de caracteres especiais, acentos e ajustes nos campos data e hora para facilitar a importação no MongoDB.

Etapas do trabalho:

- a) **Abrir o prompt de comando do MongoDB.** Vamos criar o database e as collections por lá.
- b) **Criar o Database chamado “desafio”.**
- c) **Criar as collections com validator:**

Criar a collection “ocorrencia”:

db.ocorrencia.drop() – para o caso de necessitar rodar a criação novamente

```
db.createCollection("ocorrencia", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      properties: {
        id_ocorrencia: {
          bsonType: "int",
          description: "is not required"
        },
        classificacao: {
          bsonType: "string",
          description: "is not required"
        },
        cidade: {
          bsonType: "string",
          description: "is not required"
        },
        uf: {
          bsonType: "string",
          description: "is not required"
        },
        pais: {
          bsonType: "string",
          description: "is not required"
        },
        data: {
```

```

        bsonType: "date",
        description: "is not required"
    },
    num_recomendacoes: {
        bsonType: "int",
        description: "is not required"
    }
}
}
}
})

```

d) Criar a collection “ocorrencia_tipo”

db.ocorrencia_tipo.drop() – para o caso de necessitar rodar a criação novamente

```

db.createCollection("ocorrencia_tipo", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      properties: {
        id_ocorrencia_t: {
          bsonType: "int",
          description: "is not required"
        },
        tipo: {
          bsonType: "string",
          description: "is not required"
        }
      }
    }
  }
})

```

e) Criar a collection “aeronave”

```

db.aeronave.drop()

db.createCollection("aeronave", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["assentos", "ano_fabricacao"],
      properties: {
        id_ocorrencia_a: {
          bsonType: "int",
          description: "is not required"
        }
      }
    }
  }
})

```



```

    matricula: {
      bsonType: "string",
      description: "is not required"
    },
    operador_categoria: {
      bsonType: "string"
    },
    tipo_veiculo: {
      bsonType: "string",
      description: "is not required"
    },
    fabricante: {
      bsonType: "string",
      description: "is not required"
    },
    modelo: {
      bsonType: "string",
      description: "is not required"
    },
    motor_tipo: {
      bsonType: "string"
    },
    motor_quantidade: {
      bsonType: "string"
    },
    assentos: {
      bsonType: "int",
      minimum: 1,
      maximum: 1000,
      description: "must be an integer in [ 1, 1000 ] and is required"
    },
    ano_fabricacao: {
      bsonType: "int",
      minimum: 1950,
      maximum: 2030,
      description: "must be an integer in [ 1950, 2030 ] and is required"
    },
    pais_fabricante: {
      bsonType: "string"
    },
    registro_segmento: {
      bsonType: "string"
    },
    voo_origem: {
      bsonType: "string"
    },
    voo_destino: {
      bsonType: "string"
    },
    fase_operacao: {
      bsonType: "string"
    }
  }

```

```

    }
  }
}
})

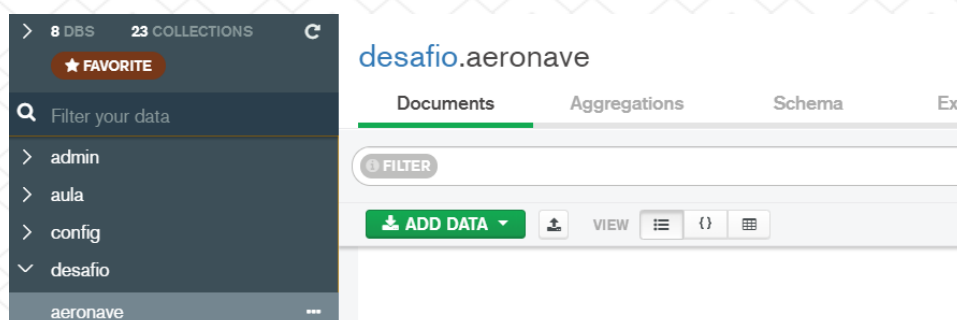
```

f) Abrir o MongoDB Compass para fazer as importações dos dados.

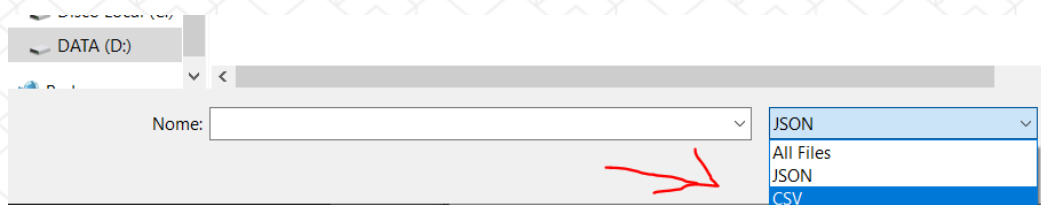
ATENÇÃO: se você editar e salvar o arquivo csv no excel, ele cria uma última linha em branco no arquivo o que vai gerar uma indicação de erro na importação. Abra os arquivos csv com o notepad (bloco de notas), por exemplo, e verifique a última linha. Se não estiver preenchida com dados, você deve apagá-la.

g) Carregar a collection “aeronave”:

Pelo MongoDB Compass, escolha o database Desafio e a collection “aeronave”. Clique no botão “ADD DATA” para inserir os dados na collecion.



Selecione o arquivo “aeronave.csv”. Repare que para o arquivo csv aparecer na seleção é necessário que você mude o tipo de arquivo.



Selecione a opção de tipo “CSV”.

Selecione o delimitador “ponto e vírgula” (SEMICOLON).

Não se esqueça de informar corretamente os tipos dos fields da collection.

- Id_ocorrendia_a, assentos, ano_fabricacao são fields do tipo **Int32**.

- Os demais são do tipo **String**.

Import To Collection desafio.aeronave

Select File

D:\Meus Documentos\Aulas\IGTI\Bootcamp Banco de Dados\Modulo 3\De

BROWSE

Select Input File Type

JSON

CSV

Options

Select delimiter SEMICOLON

☒ Ignore empty strings

☒ Stop on errors

Specify Fields and Types

	<input checked="" type="checkbox"/> id_ocorrencia_a Int32	<input checked="" type="checkbox"/> matricula String	<input checked="" type="checkbox"/> operador_categoria String	<input checked="" type="checkbox"/> tipo_veiculo String	<input checked="" type="checkbox"/> fabric Strin
1	39115	PTNQX	***	AVIAO	NEIVA INC
2	39155	PTLVI	***	AVIAO	BEECH AIF
3	39156	PPPTO	***	AVIAO	AEROSPATJ
4	39158	PRLGJ	REGULAR	AVIAO	BOEING CC
5	39176	PRMAA	REGULAR	AVIAO	AIRBUS IN
6	39178	PTMZU	REGULAR	AVIAO	AIRBUS IN
7	39235	PTWKN	***	AVIAO	CESSNA AI
8	39275	PTYRE	***	HELICOPTERO	EUROCOPT
9	39295	PUFLK	EXPERIMENTAL	ULTRALEVE	***
10	39315	PTHLE	***	HELICOPTERO	HELIBRAS

CANCEL

IMPORT

<input checked="" type="checkbox"/> motor_tipo String	<input checked="" type="checkbox"/> motor_quantidade String	<input checked="" type="checkbox"/> assentos Int32	<input checked="" type="checkbox"/> ano_fabricacao Int32	<input checked="" type="checkbox"/> pais_fabricar String
PISTAO	MONOMOTOR	4	1979	BRASIL
TURBOELICE	BIMOTOR	8	1979	BRASIL

Ao executar a importação, você terá uma tela como a abaixo indicando erros na importação. Alguns documentos seão importados, outros não.

Divulgar Documentos

Aulas

Tutoriais

Boletim

Banco de Dados

Módulo 01

Select Input File Type

JSON

CSV

Options

Select delimiter

SEMICOLON ▼

☒ Ignore empty strings
☒ Stop on errors

	<input checked="" type="checkbox"/> motor_quantidade <div>String ▼</div>	<input checked="" type="checkbox"/> assentos <div>Int32 ▼</div>	<input checked="" type="checkbox"/> ano_fabricacao <div>Int32 ▼</div>	<input checked="" type="checkbox"/> pais_fabricante <div>String ▼</div>	<input checked="" type="checkbox"/> registro_ <div>String ▼</div>
	MONOMOTOR	4	1979	BRASIL	PARTICULAR
	BIMOTOR	8	1979	BRASIL	PARTICULAR
	BIMOTOR	73	2008	BRASIL	REGULAR
	BIMOTOR	5	1984	BRASIL	REGULAR
	BIMOTOR	184	2001	BRASIL	REGULAR
	BIMOTOR	184	2001	BRASIL	REGULAR
	MONOMOTOR	1	1976	BRASIL	AGRICOLA
	MONOMOTOR	6	1994	BRASIL	PARTICULAR
	MONOMOTOR	2	2004	BRASIL	EXPERIMENTAL
	MONOMOTOR	6	1981	BRASIL	TAXI AEREO

Error importing

5.303 (100%)

Document failed validation

CLOSE

IMPORT

Por que isso acontece?

Verifique que por default as ações na validação e o nível de validação estão conforme imagem abaixo, portanto, se o flag “Stop on errors” estiver marcado, vai parar a carga dos documentos em caso de linhas que não se enquadrem na validação. Se o flag “Stop on errors” não estiver marcado, vai carregar apenas os documentos que passarem na validação.

Validation Action ⓘ

ERROR ▼

Validation Level ⓘ

OFF ▼

```
1 {
2   $jsonSchema: {
3     bsonType: 'object',
4     required: [
```

Se você abrir o arquivo com os dados “aeronave.csv”, verá que algumas informações estão sendo barradas pelo validator dos fields “assentos” e “ano_fabricacao”.

id_ocor	matricu	operad	tipo_ve	fabrica	modelo	motor	motor_q	assento	ano_fab	pais_fab	registro	voo_ori	voo_de	fase_op	acao
43790	N300R	***	AVIAO	EMBRAER	E55P	***	SEM TRAC	0	1900	ESTADOS UNIDOS	***	FORA DE A	FORA DE A	POUSO	
43869	PUFAT	EXPERIME	ULTRALEV	FABRICAC	FOX V5 SU	PISTAO	MONOMC	0	1900	BRASIL	EXPERIME	FORA DE A	FORA DE A	INDETERMINADA	
43994	PRABM	***	AVIAO	CESSNA AI	210L	PISTAO	MONOMC	0	1900	BRASIL	PARTICUL	AERODRO	GENERAL I	DECOLAGEM	
40269	PTWYD	***	AVIAO	CESSNA AI	310R	PISTAO	BIMOTOR	6	1979	BRASIL	PARTICUL	PRESIDEN	ANAPOLIS	POUSO	
40270	PREJM	***	AVIAO	CESSNA AI	152	PISTAO	MONOMC	2	1985	BRASIL	INSTRUCA	FORA DE A	FORA DE A	PARTIDA DO MOTOR	
40271	PPRTO	***	AVIAO	CIA AERON	CAP-4	PISTAO	MONOMC	2	1946	BRASIL	INSTRUCA	ENCANTA	FAZENDA I	DECOLAGEM	
40272	PTWYD	***	AVIAO	CESSNA AI	310R	PISTAO	BIMOTOR	6	1979	BRASIL	PARTICUL	PRESIDEN	ANAPOLIS	POUSO	

Para verificar isso, você deve clicar na aba validation.

desafio.aeronave

Documents 133
TOTAL SIZE 54.9KB
AVG. SIZE 423B
INDEXES 1
TOTAL SIZE 4.0KB
AVG. 4.0

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER
OPTIONS FIND RESET

ADD DATA VIEW

Displaying documents 1 - 20 of 133
REFRES

```
_id: ObjectId("5ffb3f315bdcda1890cd9cd5")
id_ocorencia_a: 39115
matricula: "P1HQK"
operador_categoria: "****"
tipo_veiculo: "AVIAO"
fabricante: "NEIVA INDUSTRIA AERONAUTICA"
modelo: "EWB-711A"
motor_tipo: "PISTAO"
motor_quantidade: "MONOMOTOR"
assentos: 4
ano_fabricacao: 1979
pais_fabricante: "BRASIL"
registro_segmento: "PARTICULAR"
voo_origem: "BRIGADEIRO ABRAZPE MACEDO"
voo_destino: "CORRENTINA"
fase_operacao: "DECOLAGEM"
```

```
_id: ObjectId("5ffb3f315bdcda1890cd9cd6")
id_ocorencia_a: 39155
matricula: "P1LVI"
operador_categoria: "****"
tipo_veiculo: "AVIAO"
fabricante: "BEECH AIRCRAFT"
modelo: "C90"
```

Os limites da validação estão barrando a importação de alguns documentos.

desafio.aeronave

DOCUMENTS 0 TOTAL SIZE 0B AVG. SIZE 0B INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

```

31 },
32 motor_tipo: {
33   bsonType: 'string'
34 },
35 motor_quantidade: {
36   bsonType: 'string'
37 },
38 assentos: {
39   bsonType: 'int',
40   minimum: 1,
41   maximum: 1000,
42   description: 'must be an integer in [ 1, 1000 ] and is required'
43 },
44 ano_fabricacao: {
45   bsonType: 'int',
46   minimum: 1950,
47   maximum: 2030,
48   description: 'must be an integer in [ 1950, 2030 ] and is required'
49 },
50 pais_fabricante: {
51   bsonType: 'string'
52 },
53 registro_segmento: {
54   bsonType: 'string'
55 },
56 voo_origem: {
57   bsonType: 'string'
58 },
59 voo_destino: {
60   bsonType: 'string'
61 },
62 fase_operacao: {
63   bsonType: 'string'
64 }
65 }
66 }
67 }

```

Para acertar isso, você precisa alterar os limites mínimos de validação dos fields assentos e ano_fabricacao.

Rode novamente o comando de criação com as validações corrigidas pelo prompt de comando ou faça isso pela própria aba Validation no MongoDB Compass.

Prompt de comando:

```

    assentos: {
      bsonType: "int",
      minimum: 0,
      maximum: 1000,
      description: "must be an integer in [ 0, 1000 ] and is required"
    },
    ano_fabricacao: {
      bsonType: "int",
      minimum: 1900,
      maximum: 2030,
      description: "must be an integer in [ 1900, 2030 ] and is required"
    },

```

Ou aba Validation no MongoDB Compass:

Documents
Aggregations
Schema
Explain Plan
Indexes
Validation

```

38  assentos: {
39    bsonType: 'int',
40    minimum: 0,
41    maximum: 1000,
42    description: 'must be an integer in [ 0, 1000 ] and is required'
43  },
44  ano_fabricacao: {
45    bsonType: 'int',
46    minimum: 1900,
47    maximum: 2030,
48    description: 'must be an integer in [ 1900, 2030 ] and is required'
49  },
50  pais_fabricante: {
51    bsonType: 'string'
52  },
53  registro_segmento: {
54    bsonType: 'string'
55  },
56  voo_origem: {
57    bsonType: 'string'
58  },
59  voo_destino: {
60    bsonType: 'string'
61  },
62  fase_operacao: {
63    bsonType: 'string'
64  }
65  }
66  }
67  }

```

Validation modified
CANCEL
UPDATE

Faça a importação novamente, mas antes limpe a tabela, porque alguns documentos foram importados na última tentativa.

`db.aeronave.remove({})`

Ao executar a nova importação, lembre-se de alterar os tipos dos campos.

Options
Select delimiter SEMICOLON
Ignore empty strings
Stop on errors
Specify Fields and Types

id_ocorrencia_a	matricula
Int32	String

A mensagem será **Import completed** em verde.

Select Input File Type

JSON

CSV

Options

Select delimiter SEMICOLON ▼

☒ Ignore empty strings

☒ Stop on errors

<input checked="" type="checkbox"/> motor_quantidade	<input checked="" type="checkbox"/> assentos	<input checked="" type="checkbox"/> ano_fabricacao	<input checked="" type="checkbox"/> pais_fabricante	<input checked="" type="checkbox"/> registro_s
String ▼	Int32 ▼	Int32 ▼	String ▼	String
MONOMOTOR	4	1979	BRASIL	PARTICULAR
BIMOTOR	8	1979	BRASIL	PARTICULAR
BIMOTOR	73	2008	BRASIL	REGULAR
BIMOTOR	5	1984	BRASIL	REGULAR
BIMOTOR	184	2001	BRASIL	REGULAR
BIMOTOR	184	2001	BRASIL	REGULAR
MONOMOTOR	1	1976	BRASIL	AGRICOLA
MONOMOTOR	6	1994	BRASIL	PARTICULAR
MONOMOTOR	2	2004	BRASIL	EXPERIMENTAL
MONOMOTOR	6	1981	BRASIL	TAXI AEREO

Import completed

5.303 / 5.303

DONE

h) Carregar a collection “ocorrencia_tipo”.

Não se esqueça de informar corretamente os tipos dos fields da collection.

- Id_ocorrencia_t é do tipo **Int32**.
- Os demais são do tipo **String**.

Options

Select delimiter SEMICOLON ▼

☒ Ignore empty strings

☒ Stop on errors

Specify Fields and Types

<input checked="" type="checkbox"/> id_ocorrencia_t	<input checked="" type="checkbox"/> tipo
Int32 ▼	String ▼
1 39115	PANE SECA
2 39155	VAZAMENTO DE COMBUSTIVEL
3 39156	FOGO EM VOO

i) Carregar a collection “ocorrencia”.

Não se esqueça de informar corretamente os tipos dos fields da collection.

- Id_ocorrencia e num_recomendacoes são do tipo **Int32**.
- Data é do tipo **date**.
- Os demais são do tipo **String**.

Options

Select delimiter **SEMICOLON**

☒ Ignore empty strings

☒ Stop on errors

Specify Fields and Types

☒ id_ocorrencia Int32

☒ classificacao String

<input checked="" type="checkbox"/> uf String	<input checked="" type="checkbox"/> pais String	<input checked="" type="checkbox"/> data Date	<input checked="" type="checkbox"/> num_recomendacoes Int32
BA	BRASIL	2010-02-07T17:40:00Z	2
MG	BRASIL	2010-02-05T12:55:00Z	0
DR	BRASIL	2010-01-10T22:15:00Z	2

Atividades práticas

Execute os comandos das práticas abaixo e **anote os resultados**.

1. **Verifique o número de documentos carregados na collection “ocorrencia”.**
Você pode usar a função `count()` ou `db.collection.aggregate` com `{ $sum:1 }`.
2. **Verifique o número de documentos carregados na collection “ocorrencia_tipo”.** Você pode usar a função `count()` ou `db.collection.aggregate` com `{ $sum:1 }`.
3. **Verifique o número de documentos carregados na collection “aeronave”.**
Você pode usar a função `count()` ou `db.collection.aggregate` com `{ $sum:1 }`.
4. **Execute um comando `find()` na collection `aeronave` com `modelo= 'AB-115'` OU `tipo_veiculo = 'AVIAO'`.**

```
db.collection.find({ $or: [{ , } ]})
```

Limite a consulta para trazer apenas os 5 primeiros documentos.

5. **Execute um comando `find()` na collection `aeronave` onde o `tipo_veiculo` não são os seguintes tipos: `['AVIAO', 'HELICOPTERO', 'HIDROAVIAO', 'PLANADOR', 'ANFIBIO']`**

Limite o resultado da consulta para trazer apenas os 10 primeiros documentos.

A dica é que podemos usar a condição `IN` para retornar apenas valores que estão dentro de uma lista e o `NOT IN` para retornar os valores que não estão dentro de uma lista.

6. **Execute um comando `aggregate()` na collection `aeronave` para agrupar os documentos pelo campo `tipo_veiculo` fazendo uma contagem (`$sum:1`) para cada `tipo_veiculo`.**

7. Execute um comando find() na collection ocorrencia para buscar os documentos com o field num_recomendacoes menor ou igual (\$lte) a 2 (<=2). Limite o resultado da sua consulta para 10 documentos.

8. Execute um comando aggregate() na collection ocorrencia para buscar os documentos conforme abaixo.

- Field num_recomendacoes menor ou igual (\$lte) a 2 (<=2).
- Agrupar pelo field uf (\$uf)
- Fazer a contagem (\$sum:1)
- Ordenar de forma descendente

Dica:

```
db.ocorrencia.aggregate([  
  { $match: { },  
    { $group: { },  
    { $sort:  { }  
  ])
```

9. Execute um comando aggregate() na collection ocorrencia para buscar a média dos números de recomendações (\$num_recomendacoes).

10. Execute um comando lookup aggregate() na collection aeronave fazendo uma junção com a collection ocorrencia. Limite o resultado do lookup aggregate() em dois documentos para facilitar a visão do que acontece.

Dica:

```
db.aeronave.aggregate([  
  {  
    $lookup:  
    {  
      from: "ocorrencia",  
      localField: ( " "),  
      foreignField: ( " "),  
      as: "ocorrencia_aeronave"  
    }  
  },
```



```
{  
  $limit: 2  
}  
]).pretty()
```

11. Execute um comando `lookup aggregate()` na collection `ocorrencia` fazendo uma junção com a collection `ocorrencia_tipo`. Limite o resultado do `lookup aggregate()` em dois documentos para facilitar a visão do que acontece.

Chegamos ao final da prática do nosso desafio.