

Banco de Dados II

BD baseado em documentos

– **MongoDB**

Compass, coleções e find()



Profa. Damires Souza
damires@ifpb.edu.br



MongoDB Compass

MongoDB Compass - localhost:27017/bd223.minhanovacoção

Connect Edit View Collection Help

localhost:27017

Documents
bd223.minhanov...

My Queries

Databases

Search

Academico

BD2

Doenças

MPTIBD

admin

bd223

minhanovacoção

series

teste

config

local

loja

test

bd223.minhanovacoção

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' }

Reset Find

ADD DATA EXPORT DATA

1 - 5 of 5

5 DOCUMENTS 1 INDEXES

_id: ObjectId('646e61ffcb2027b92ba5ac5b')
nome: "João"

_id: ObjectId('646e628ccb2027b92ba5ac5c')
nome: "Marcia"
idade: 23
hobbies: Array
endereço: Object

_id: ObjectId('646e644dcb2027b92ba5ac5d')
nome: "Alana"
idade: 33
hobbies: Array
endereço: Object

_id: ObjectId('646e644dcb2027b92ba5ac5e')
nome: "Alvaro"
idade: 43
hobbies: Alvaro
endereço: Object

>_MONGOSH

Relembrando “minhanovacoção”

```
{
  "_id" : ObjectId("5e85f1e58f21fd24b7a8d20d"),
  "nome" : "Alana",
  "idade" : 33.0,
  "hobbies" : [
    "volley",
    "filmes"
  ],
  "endereço" : {
    "rua" : "XX",
    "num" : 305.0,
    "apto" : 502.0
  }
}
```


Array

Documento ou objeto embutido

Relacional x Documentos

Conceito	Relacional	Documentos (MongoDB)
Conjunto de instâncias	Tabela ou conjunto de tabelas	Coleção de documentos
Instância	Tupla ou registro	Documento
Atributo Simples	Atributo ou coluna	Campo (field)
Atributo Composto	Dividido em atributos simples ou em tabelas com dependência entre elas	Campos ou documentos embutidos
Atributo Multivalorado	Dividido em atributos ou em tabelas com dependência entre elas	Campos ou documentos embutidos
Valor Nulo	possui	Pode não ter
Identificador	PK	Object identifier (_id)
Referência	FK	Refs ou aninhamento
Integridade Referencial	possui	não possui

Exemplo de aplicação

amazon.com.br

Enviar para Damires
João Pessoa 58033370

Todos ▾ Nosql



Olá, Damires
Contas e Listas ▾

Devoluções
e Pedidos

0 Carrinho

≡ Todos

Venda na Amazon

Ideias para Presente

Eletrônicos

Computadores

Amazon Moda

Atendimento ao Cliente

Comprar novamente

Confira os E-readers Kindle

1-48 de mais de 1.000 resultados para "Nosql"

Ordenar por: Em destaque ▾

☐ Frete Grátis em envios pela Amazon

Todos os clientes têm frete GRÁTIS em pedidos a partir de R\$ 129 em produtos enviados pela Amazon.

Departamento

Livros

Base de dados

Programação de Computadores

Importados de Computação, Informática e Mídias Digitais

Loja Kindle

Computação, internet e mídia digital

eBooks em outros idiomas estrangeiros

Computação, Informática e Mídias Digitais em Línguas Estrangeiras

Computação, internet e mídia digital em inglês

Avaliação do Cliente

★★★★★ e acima

★★★★☆ e acima

★★★☆☆ e acima

★★★★★ e acima

Série de livros

☐ For Mere Mortals

Resultados



NoSQL Essencial: um Guia Conciso Para o Mundo Emergente da Persistência Poliglota

Edição Português por Martin Fowler e Pramod J. Sadalage

★★★★☆ ~ 142

Capa Comum

R\$62⁴⁶ De: R\$73,00

Receba **segunda-feira, 12 de jun.** - quinta-feira, 15 de jun.



NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence (English Edition)

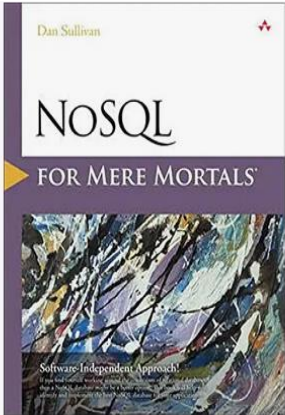
Edição Inglês por Pramod J. Sadalage e Martin Fowler

★★★★☆ ~ 382

Kindle

R\$181⁴⁷

Disponível instantaneamente



NoSQL for Mere Mortals

Parte de: For Mere Mortals (5 livros)

★★★★☆ ~ 44

Capa Comum

R\$465¹²

em até 9x de R\$ 51,68 sem juros

Receba **quarta-feira, 28 de jun.** - **quinta-feira, 13 de jul.**

Frete GRÁTIS

Compra Internacional



NoSQL: Como armazenar os dados de uma aplicação moderna

por David Paniz

★★★★☆ ~ 7

Kindle

R\$49⁹⁰

Disponível instantaneamente

Outro formato: Capa Comum

No MongoDB Compass

MongoDB Compass - localhost:27017/local.minhanovacolecao

Connect Edit View Collection Help

localhost:27017

Documents
local.minhanova...

+

My Queries

Databases

Search

admin

config

local

test

NovaColeçãoTeste3

minhanovacoleção

Documents

Aggregation

Filter

+

ADD DATA

EXPORT

```
{ "_id": ObjectId('656...'), "nome": "João" }
```

```
{ "_id": ObjectId('656...'), "nome": "Marcia", "idade": 23, "hobbies": Array (2), "endereço": Object }
```

```
{ "_id": ObjectId('656...'), "nome": "Alana", "idade": 33, "hobbies": Array (2), "endereço": Object }
```

Create Database

Database Name

bd2

Collection Name


livros


☐ Time-Series
Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

> Additional preferences (e.g. Custom collation, Capped, Clustered collections)


Cancel



Create Database

 **localhost:27017** ...









 **Documents**
bd2.livros

+

 My Queries

 Databases  +

Search

- ▶  admin
- ▼  bd2
 -  **livros** ...
- ▶  config
- ▶  local
- ▼  test
 -  NovaColeçãoTeste3
 -  minhanovacoleção

bd2.livros

Documents Aggregations Schema Indexes Validation

[Filter](#)   Type a query: { field: 'value' } or [Generate query](#) 

 **ADD DATA**  **EXPORT DATA**



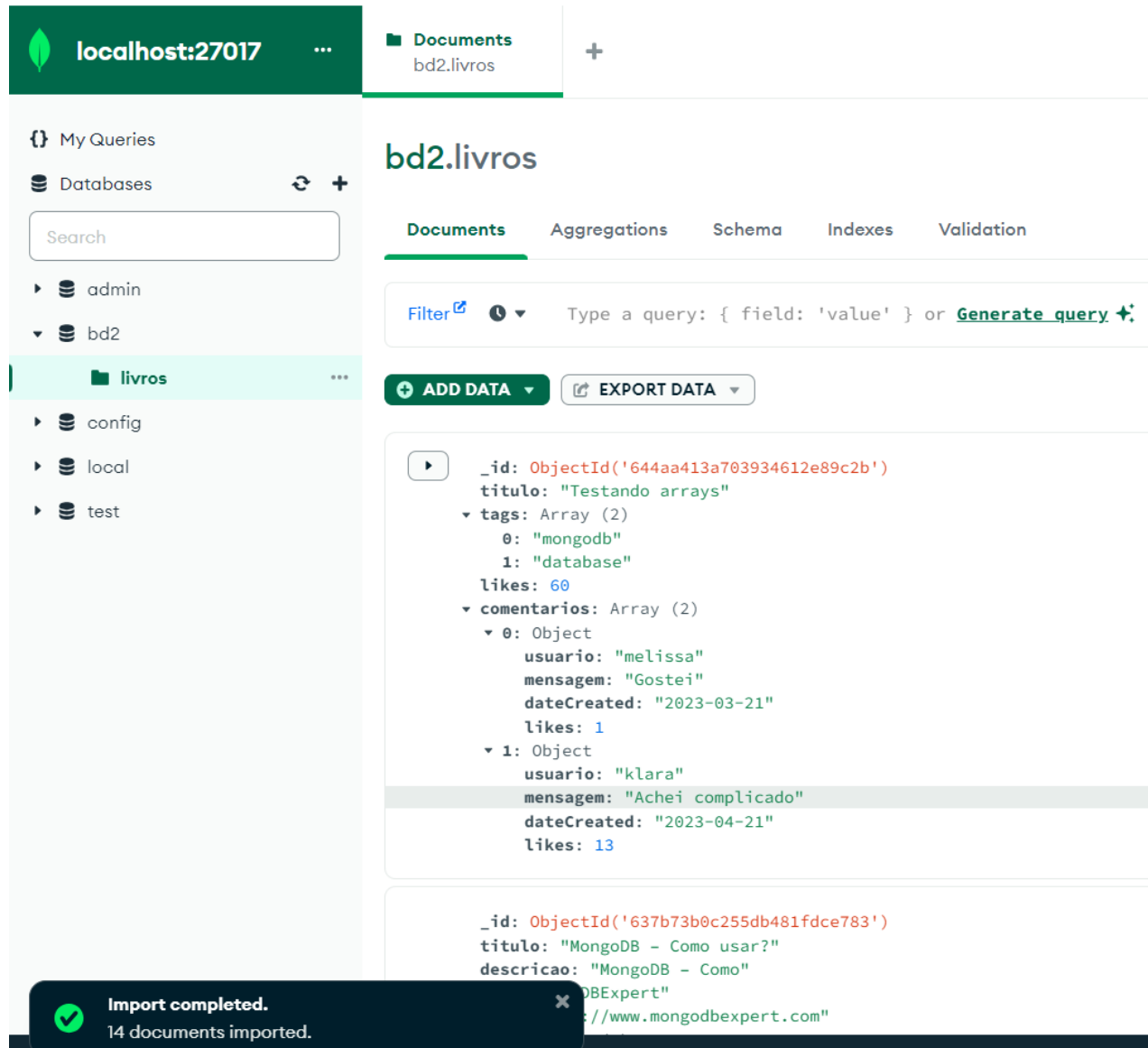
This collection has no data

It only takes a few seconds to import data from a JSON or CSV file.

Importar livros.json

Import Data

Coleção “livros”



localhost:27017 ...

Documents
bd2.livros

My Queries
Databases
Search
admin
bd2
livros
config
local
test

bd2.livros

Documents Aggregations Schema Indexes Validation

Filter ⓘ ⌚ Type a query: { field: 'value' } or [Generate query](#) ⚡

ADD DATA EXPORT DATA

```
{
  "_id": ObjectId('644aa413a703934612e89c2b'),
  "titulo": "Testando arrays",
  "tags": Array (2)
    0: "mongodb"
    1: "database"
  "likes": 60
  "comentarios": Array (2)
    0: Object
      usuario: "melissa"
      mensagem: "Gostei"
      dateCreated: "2023-03-21"
      likes: 1
    1: Object
      usuario: "klara"
      mensagem: "Achei complicado"
      dateCreated: "2023-04-21"
      likes: 13
}
```

```
{
  "_id": ObjectId('637b73b0c255db481fdce783'),
  "titulo": "MongoDB - Como usar?",
  "descricao": "MongoDB - Como usar MongoDBExpert"
}
```

Import completed.
14 documents imported.

Importando um CSV

Create Collection

Collection Name

testeCSV

☐ Time-Series
Time-series collections eff of time. [Learn More](#)

➤ Additional preferences (

My Queries

Databases

Search

- admin
- bd2
 - livros
 - testeCSV**
- config
- local
- test

bd2.testeCSV

Documents Aggregations Schema Indexes Validation

Filter ⚙️ ⌚ Type a query: { field: 'value' } or [Generate query](#) ⚡

ADD DATA EXPORT DATA

```
{
  "_id": ObjectId('656db7b951590c2666e2915b'),
  "ano": 2023,
  "mes": "03-marco",
  "processo_numero": 8024764523,
  "solicitacao_data": "29/03/2023",
  "solicitacao_hora": "08:50",
  "solicitacao_descricao": "Colocação de lonas plásticas e capinação na barreira.",
  "solicitacao_regional": "NORTE",
  "solicitacao_bairro": "AGUA FRIA",
  "solicitacao_localidade": "CGO. SAO SEBASTIAO",
  "solicitacao_endereco": "RUA JOSÉ MANUEL DE LIMA, N87 A E B",
  "solicitacao_roteiro": "PRÓXIMO AO TERMINAL DE ONIBUS DE ÁGUA FRIA",
  "rpa_codigo": 2,
  "rpa_nome": "2-NORTE",
  "solicitacao_microrregiao": 2.2,
  "solicitacao_plantao": "Não",
  "solicitacao_origem_chamado": "Ocorrências",
  "solicitacao_vitimas": "Não",
  "solicitacao_vitimas_fatais": "Não",
  "processo_situacao": "execucacao",
  "processo_tipo": "ATENDIMENTO",
  "processo_localizacao": "SEDEC - GERÊNCIA REGIONAL NORTE",
  "status": "TRAMITAÇÃO"
}
```



Tabela flat?

Import completed.
168 documents imported.

Voltando a Livros

Insert Document

To collection bd2.livros

VIEW  


```
1  /**
2  * Paste one or more documents here
3  */
4  {
5    "_id": {
6      "$oid": "656db81551590c2666e29204"
7    }
8  }
```


**titulo: "Verificando o Insert",
tags: ["mongodb","nosql"],
Likes: 23**

Cancel **Insert**

bd2.livros

Documents Aggregation

Filter   Type a query

ADD DATA  **EXPORT**

`_id: ObjectId('644...')`
`titulo: "Testando"`
`tags: Array (2)`
`likes: 60`
`comentarios: Array`

`_id: ObjectId('637...')`
`titulo: "MongoDB -"`
`descricao: "MongoD"`
`tags: "MongoDBF"`

`_id: ObjectId('656db81551590c2666e29204')`
`titulo: "Verificando o Insert"`
`tags: Array (2)`
`0: "mongodb"`
`1: "nosql"`
`likes: 23`

Novo Import em livros

Importe um dataset em **JSON sem Ids**

**** Use livros_semID.json**

O que ocorre?

Principais tipos de dados

- **String:** tipo de dados mais comumente usado. String no mongodb deve ser um UTF-8 válido.
- **Integer:** para armazenar um valor numérico inteiro. Integer pode ser 32 bits ou 64 bits dependendo do servidor.
- **Boolean:** valor booleano (verdadeiro/falso).
- **Double:** usado para armazenar valores de ponto flutuante.
- **Array:** usado para armazenar arrays, listas ou múltiplos valores dentro de uma chave (key) ou propriedade.
- **Object:** usado para incorporar documentos/objetos (documentos embutidos).

Principais tipos de dados

- ***Null***: usado para armazenar um valor nulo (null).
- ***Symbol***: usado de forma idêntica ao String, porém ele é geralmente reservado para linguagens que usam um tipo de símbolo específico.
- ***Date***: data pode ser representada como uma string ou um objeto *Date* representado em UTC.
- ***Object ID***: tipo de dados usado para armazenar os identificadores(ID) dos documentos.
- ***Binary data***: usado para armazenar um dado binário.

Tipo Date

```
var mydateobj = new Date()  
mydateobj.toString()  
mydateobj.getMonth()
```

```
var date1 = new Date("2022-04-  
16")  
var date2 = new Date("2022-04-  
16T10:05:44")  
print(date1)  
print(date2)
```

```
var myDateString = Date();
```

```
db.produto.update(  
  { _id: 1 },  
  {  
    $set: { item: "apple" },  
    $setOnInsert: { dateAdded:  
      new Date() }  
  },  
  { upsert: true }  
)
```

Tipo null


```
db.inventory.insertMany([ {  
  _id: 1, item: null },  
  { _id: 2 } ])
```

```
/* 1 */  
{  
  "_id" : 1.0,  
  "item" : null  
}  
  
/* 2 */  
{  
  "_id" : 2.0  
}
```

```
db.inventory.find( { item: null } )
```



```
db.inventory.find( {  
  item: null, item: { $exists: true } } )
```



```
/* 1 */  
{  
  "_id" : 1.0,  
  "item" : null  
}
```

Coleção com validação de esquema

- O MongoDB é **flexível** em termos de **esquema**
 - Pode até não ter um definido previamente
 - Não é necessário especificar o número ou tipo de colunas/propriedades antes de inserir os dados
- Mas, é possível definir um esquema JSON e usá-lo para impor **regras de validação** para os dados

Exemplo






```
db.createCollection("students", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: [ "name", "year", "major", "address" ],
      properties: {
        name: {
          bsonType: "string",
          description: "must be a string and is required" },
        year: {
          bsonType: "int",
          minimum: 2017, maximum: 3017,
          description: "must be an integer in [ 2017, 3017 ] and is required"
        },
        address: {
          bsonType: "object",
          required: [ "city" ],
          properties: {
            street: { bsonType: "string", description: "must be a string "
            },
            city: { bsonType: "string", description: "must be a string " } } } } } } } }
```

Voltando ao Find()

bd2.livros

15 1
DOCUMENTS INDEXES

Documents Aggregations Schema Indexes Validation

Filter   [Generate query](#)  [Explain](#) [Reset](#) [Find](#)  [Options](#) 

```
_id: ObjectId('656db81551590c2666e29204')  
titulo: "Verificando o Insert"  
▸ tags: Array (2)  
likes: 23
```

Sort

cursor.sort(sort)

- especifica a ordem na qual a consulta retorna os documentos

```
db.livros.find().sort( { "titulo": 1 } )
```

```
db.livros.find().sort( { "titulo": -1 } )
```

```
db.livros.find().sort( { "titulo": 1, likes : -1 } )
```

= Order by do SQL

Limit e Skip

`cursor.limit()`

`db.collection.find(<query>).limit(<number>)`

`db.livros.find().limit(3);`

`db.livros.find.sort(
{“titulo”:1,likes:-1}).limit(3).skip(3);`

Usando a interface de consultas

bd2.livros

15
DOCUMENTS

1
INDEXES

Documents Aggregations Schema Indexes Validation

Filter ⓘ ⌚ ▼ Type a query: { field: 'value' } or [Generate query](#) ⚡

Explain

Reset

Find

⌕

Options ▼

Project { field: 0 }

Sort titulo

MaxTimeMS 60000

Collation { locale: 'simple' }

Skip 0

Limit 0

```
>_MONGOSH
```

```
> use bd2
```

```
< switched to db bd2
```

```
> db.livros.find().sort( { "titulo": 1 } )
```

```
< {
```

```
  _id: ObjectId("62c5956744d10d4b78ae3b12"),
```

```
  titulo: 'Cassandra - The Definitive Guide',
```

```
  descricao: 'Introduction to Cassandra',
```

Verifique

```
db.livros.find(  
  { titulo: /^V/ },  
  { titulo: 1, likes: 1, _id: 0 } ).sort({titulo:-  
1}).limit(2)
```

Índice

- `db.livros.createIndex({ titulo: 1 })`



ascendente

- `db.livros.createIndex({ titulo:1, likes: -1 })`



descendente

SEM índices: collection scan

`db.livros.find().explain()`

Critérios para criação equivalentes aos de BDR

1) Exercício: Find na coleção de livros

```
db.livros.find( { likes: 100 } )
```

```
db.livros.find( { }, { "titulo": 1, "descricao": 1 } )
```

```
db.livros.find( { }, { "titulo": 1, "descricao": 1 } ).limit(2)
```

```
db.livros.find({}, {_id:0,titulo:1});
```

```
db.livros.find( { tags: "database" } )
```

```
db.livros.find( { "comentarios.likes": { $gt: 2 } } )
```

Teste usando o shell (mongosh) e a interface de consultas do Compass

2) Exercício: Modelo conceitual

Monte um modelo conceitual E-R para este banco com a coleção “livros”.