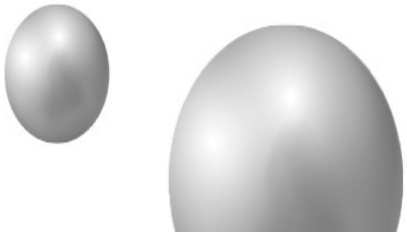

UNITAU
Universidade de Taubaté


MongoDB
Find

Pesquisa em Documentos Embutidos

```
> db.vendas.insertMany([
   {_id: "001", "data": "2020-02-14",
     "cliente": {"nome": "Laura", "sexo":
"F", "idade": 21},
     "produtos": [
                 {"nome": "Caderno",
"quantidade": 2, "preco": 10.50},
                 {"nome": "Lapis",
"quantidade": 3, "preco": 1.15}
                 ]
   },
```



Pesquisa em Documentos Embutidos

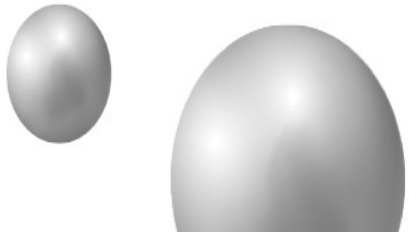
```
   {_id: "002", "data": "2020-02-18",
     "cliente": {"nome": "Paulo", "sexo":
"M", "idade": 25},
     "produtos": [
                 {"nome": "Caderno",
"quantidade": 3, "preco": 14.00},
                 {"nome": "Caneta",
"quantidade": 2, "preco": 3.20},
                 {"nome": "Post-it",
"quantidade": 3, "preco": 12.15}
                 ]
   },
```

Pesquisa em Documentos Embutidos

```
   {_id: "003", "data": "2020-02-18",
     "cliente": {"nome": "Henrique", "sexo":
"M", "idade": 19},
     "produtos": [
                 {"nome": "Lapis",
"quantidade": 4, "preco": 1.30},
                 {"nome": "Borracha",
"quantidade": 1, "preco": 5.90},
                 {"nome": "Marca Texto",
"quantidade": 2, "preco": 3.25}
                 ]
   },
```

Pesquisa em Documentos Embutidos

```
   {_id: "004", "data": "2020-02-22",
     "cliente": {"nome": "Marta", "sexo":
"F", "idade": 23},
     "produtos": [
                 {"nome": "Caderno",
"quantidade": 1, "preco": 16.20},
                 {"nome": "Lapiseira",
"quantidade": 3, "preco": 14.30},
                 {"nome": "Borracha",
"quantidade": 2, "preco": 5.90}
                 ]
   },
```



Pesquisa em Documentos Embutidos

```
   {_id: "005", "data": "2020-03-02",
     "cliente": {"nome": "Laura", "sexo":
"F", "idade": 17},
     "produtos": [
                 {"nome": "Fichario",
"quantidade": 2, "preco": 25.10},
                 {"nome": "Caderno",
"quantidade": 3, "preco": 14.30},
                 {"nome": "Caneta",
"quantidade": 4, "preco": 3.20},
                 {"nome": "Corretivo",
"quantidade": 1, "preco": 5.00},
                 {"nome": "Post-it",
"quantidade": 4, "preco": 14.10}
                 ]
   },
```

Pesquisa em Documentos Embutidos

```
   {_id: "006", "data": "2020-03-06",
     "cliente": {"nome": "Renata", "sexo":
"F", "idade": 32},
     "produtos": [
                 {"nome": "Fichario",
"quantidade": 3, "preco": 27.00},
                 {"nome": "Caderno",
"quantidade": 4, "preco": 15.60},
                 {"nome": "Caneta",
"quantidade": 6, "preco": 3.10},
                 {"nome": "Lapiseira",
"quantidade": 4, "preco": 12.50},
                 {"nome": "Borracha",
"quantidade": 5, "preco": 6.10},
```

Pesquisa em Documentos Embutidos

```
                    {"nome": "Corretivo",
"quantidade": 2,    "preco": 5.20}
                    ]
   }
 ])
{
        "acknowledged" : true,
        "insertedIds" : [
                "001",
                "002",
                "003",
                "004",
                "005",
                "006"
        ]
}
```

Pesquisa em Documentos Embutidos

Para realizar um pesquisar por um documento embutido, a pesquisa deve ser feita por todos os campos do documento embutido, na ordem exata dos campos dentro do documento embutido.

```
> db.editora.find( {sede: {pais: "Brasil",
estado: "SP", cidade: "Sao Paulo"}} )
{ "_id" : "novatec", "nome" : "Novatec
Editora Ltda", "sede" : { "pais" : "Brasil",
"estado" : "SP", "cidade" : "Sao Paulo" } }
> db.editora.find( {sede: {pais: "Brasil",
estado: "SP"}} )
>
> db.editora.find( {sede: {estado: "SP",
pais: "Brasil", cidade: "Sao Paulo"}} )
>
```

Pesquisa em Documentos Embutidos

Para realizar um pesquisar por um campo de um documento embutido deve ser usada a notação com ponto (.) para especificar o campo do documento embutido.

```
> db.editora.find( {"sede.estado": {$ne: "SP"}} )
{ "_id" : "oreilly", "nome" : "O'Reilly Media", "sede" : { "pais" : "Estados Unidos", "estado" : "CA", "cidade" : "Sebastopol" } }
{ "_id" : "cm", "nome" : "Editora Ciencia Moderna", "sede" : { "pais" : "Brasil", "estado" : "RJ", "cidade" : "Rio de Janeiro" } }
```

Projeção em Documentos Embutidos

Para especificar um campo de um documento embutido na lista de campos a retornar/não retornar deve ser usada a notação com ponto (.) para especificar o campo do documento embutido.

```
> db.acervo.find( {}, {_id: 0, editora: 1,
"sede.pais": 1} )
{ "editora" : "Novatec Editora Ltda",
"sede" : { "pais" : "Brasil" } }
{ "editora" : "O'Reilly Media", "sede" :
{ "pais" : "Estados Unidos" } }
{ "editora" : "Editora Ciencia Moderna",
"sede" : { "pais" : "Brasil" } }
```

Projeção em Documentos Embutidos

```
> db.acervo.find( {}, {_id: 0, livros: 0,
"sede.cidade": 0} )
{ "editora" : "Novatec Editora Ltda",
"sede" : { "pais" : "Brasil", "estado" : "SP"
} }
{ "editora" : "O'Reilly Media", "sede" :
{ "pais" : "Estados Unidos", "estado" :
"CA" } }
{ "editora" : "Editora Ciencia Moderna",
"sede" : { "pais" : "Brasil", "estado" : "RJ"
} }
```

Projeção em Documentos Embutidos

A projeção de campos de um documento embutido pode ser feita mesmo para campos embutidos em arrays.

```
> db.acervo.find( {}, {_id: 0, editora: 1,
"livros.titulo": 1} )
{ "editora" : "Novatec Editora Ltda",
"livros" : [ { "titulo" : "Introducao ao
MongoDB" }, { "titulo" : "Arduino Basico" } ]
}
{ "editora" : "O'Reilly Media", "livros" :
[ { "titulo" : "Programming With QT" },
{ "titulo" : "SQL Tuning" } ] }
{ "editora" : "Editora Ciencia Moderna",
"livros" : [ { "titulo" : "Programacao em
Linguagem C" } ] }
```

Pesquisando em Arrays

```
> db.alunos.insertMany( [
    {_id:1, nome:"Joao", notas:[8,9,5]},
    {_id:2, nome:"Maria", notas:[8,10,7]},
    {_id:3, nome:"Jose", notas:[10,10,9]},
    {_id:4, nome:"Marcia", notas:[7,6,9]},
    {_id:5, nome:"Paulo", notas:[5,4,5]},
    {_id:6, nome:"Ana", notas:[5,8,9]}
  ] )
{ "acknowledged" : true, "insertedIds" : [ 1,
2, 3, 4, 5, 6 ] }
```

Pesquisando em Arrays

```
> db.alunos.find()
{ "_id" : 1, "nome" : "Joao", "notas" : [ 8,
9, 5 ] }
{ "_id" : 2, "nome" : "Maria", "notas" : [ 8,
10, 7 ] }
{ "_id" : 3, "nome" : "Jose", "notas" : [ 10,
10, 9 ] }
{ "_id" : 4, "nome" : "Marcia", "notas" :
[ 7, 6, 9 ] }
{ "_id" : 5, "nome" : "Paulo", "notas" : [ 5,
4, 5 ] }
{ "_id" : 6, "nome" : "Ana", "notas" : [ 5,
8, 9 ] }
```

Pesquisando em Arrays

É possível pesquisar documentos que tenham como valor de um campo exatamente um array indicado usando usado o filtro `{ <field>: <array> }` onde `<array>` é o array a ser pesquisado.

```
> db.alunos.find( {notas: [8, 9, 5]} )
{ "_id" : 1, "nome" : "Joao", "notas" : [ 8,
9, 5 ] }
```

O valor do campo deve ser exatamente igual ao array indicado, inclusive com todos elementos na mesma ordem indicada.

```
> db.alunos.find( {notas: [8, 9]} )
>
```

Pesquisando em Arrays

Para pesquisar documentos que contenham um elemento específico, mesmo que contenham outros elements e independente da ordem deve ser usado o filtro { <field>: <value> } onde <value> é o elemento a pesquisar.

```
> db.alunos.find( {notas: 10} )
{ "_id" : 2, "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
{ "_id" : 3, "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
```

Pesquisando em Arrays

Para pesquisar array que contenha ao menos um elemento que atenda a condição com um operador deve ser usado o filtro { <array field>: { <operator1>: <value1>, ... } }.

```
> db.alunos.find( {notas: {$gte: 9}} )
{ "_id" : 1, "nome" : "Joao", "notas" : [ 8, 9, 5 ] }
{ "_id" : 2, "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
{ "_id" : 3, "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
{ "_id" : 4, "nome" : "Marcia", "notas" : [ 7, 6, 9 ] }
{ "_id" : 6, "nome" : "Ana", "notas" : [ 5, 8, 9 ] }
```

Pesquisando em Arrays

Para pesquisar arrays que tenham elementos que atendam uma condição composta mesmo que não seja o mesmo elemento atendendo a todas condições deve ser usado o filtro { <array field>: { <operator1>: <value1>, <operator2>: <value2>… } }.

```
> db.alunos.find( { notas: {$lt: 8, $gt: 9}} )
{ "_id" : 2, "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
```

Pesquisando em Arrays

Para aplicar a pesquisa ao elemento de uma posição específica do array deve ser usada a notação de ponto para indicar a posição do elemento no array. O primeiro elemento de um array é o de posição 0.

```
> db.alunos.find( {"notas.0": 8} )
{ "_id" : 1, "nome" : "Joao", "notas" : [ 8, 9, 5 ] }
{ "_id" : 2, "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
```

$all

O operador $all verifica se o array contém todos os elementos especificados, independente da ordem.

```
{ <field>: { $all: [ <value1> , <value2> ... ] } }
```

```
> db.alunos.find( {notas: {$all: [9, 10]}} )
{ "_id" : 3, "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
```

Esta pesquisa é equivalente a:

```
> db.alunos.find( {$and: [{notas: 9}, {notas: 10}]} )
{ "_id" : 3, "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
```

$elemMatch

O operador $elemMatch verifica se algum elemento do array atende a todos critérios estabelecidos.

```
{ <field>: { $elemMatch: { <query1>, <query2>, ... } } }
```

```
> db.alunos.find( {notas: {$elemMatch: {$gt: 5, $lt: 7}}} )
{ "_id" : 4, "nome" : "Marcia", "notas" : [ 7, 6, 9 ] }
```

$elemMatch

O operador $elemMatch também pode ser utilizado para pesquisar documentos dentro de um array.

```
> db.acervo.find( {livros: {$elemMatch: {ano: "2010", edicao: "2"}}} )
{ "_id" : "oreilly", "editora" : "O'Reilly Media", "sede" : { "pais" : "Estados Unidos", "estado" : "CA", "cidade" : "Sebastopol" }, "livros" : [ { "titulo" : "Programming With QT", "isbn" : "9781449390938", "ano" : "2010", "edicao" : "2" }, { "titulo" : "SQL Tuning", "isbn" : "9780596552367", "ano" : "2009", "edicao" : "1" } ] }
```

$elemMatch

Não é necessário usar $elemMatch para pesquisar documentos dentro de um array se for usada apenas uma condição.

```
> db.acervo.find( {livros: {$elemMatch: {ano: "2010"}}}, {editora:1, "livros.titulo":1, "livros.ano":1} )
{ "_id" : "oreilly", "editora" : "O'Reilly Media", "livros" : [ { "titulo" : "Programming With QT", "ano" : "2010" }, { "titulo" : "SQL Tuning", "ano" : "2009" } ] }
{ "_id" : "cm", "editora" : "Editora Ciencia Moderna", "livros" : [ { "titulo" : "Programacao em Linguagem C", "ano" : "2010" } ] }
```

$elemMatch

```
> db.acervo.find( {"livros.ano": "2010"},
{editora:1, "livros.titulo":1,
"livros.ano":1} )
{ "_id" : "oreilly", "editora" : "O'Reilly
Media", "livros" : [ { "titulo" :
"Programming With QT", "ano" : "2010" },
{ "titulo" : "SQL Tuning", "ano" : "2009" } ]
}
{ "_id" : "cm", "editora" : "Editora Ciencia
Moderna", "livros" : [ { "titulo" :
"Programacao em Linguagem C", "ano" :
"2010" } ] }
```

$elemMatch

O operador $elemMatch pode ser usado na projeção para filtrar documentos embutidos em um array .

```
> db.acervo.find( {editora: "O'Reilly Media",
livros: {$elemMatch: {ano: "2010"}}} )
{ "_id" : "oreilly", "editora" : "O'Reilly
Media", "sede" : { "pais" : "Estados Unidos",
"estado" : "CA", "cidade" : "Sebastopol" },
"livros" : [ { "titulo" : "Programming With
QT", "isbn" : "9781449390938", "ano" :
"2010", "edicao" : "2" }, { "titulo" : "SQL
Tuning", "isbn" : "9780596552367", "ano" :
"2009", "edicao" : "1" } ] }
```

$elemMatch

```
> db.acervo.find( {editora: "O'Reilly
Media"}, {livros: {$elemMatch: {ano:
"2010"}}} )
{ "_id" : "oreilly", "livros" :
[ { "titulo" : "Programming With QT",
"isbn" : "9781449390938", "ano" : "2010",
"edicao" : "2" } ] }
```

$elemMatch

Quando usado na projeção, o operador $elemMatch retorna apenas o primeiro documento embutido que atenda ao critério do operador .

```
> db.acervo.find( {}, {livros: {$elemMatch: {ano: "2015"}}} )
{ "_id" : "novatec", "livros" :
[ { "titulo" : "Introducao ao MongoDB", "isbn" : "8575224220", "ano" : "2015", "edicao" : "1" } ] }
{ "_id" : "oreilly" }
{ "_id" : "cm" }
```

$size

O operador $size permite testar o tamanho de um array.

```
> db.megasena.find( {uf: {$size: 15}} )
{ "_id" : 529, "data" : "14/01/2004",
"dezenas" : [ 56, 45, 13, 33, 38, 1 ],
"sena_ganhadores" : 15, "uf" : [ "BA", "CE",
"CE", "PB", "PB", "PB", "PE", "PE", "PE",
"PE", "PE", "PI", "PI", "RN", "RN" ],
"sena_rateio" : 348732.75, "quina_ganhadores"
: 87, "quina_rateio" : 8565.12,
"quadra_ganhadores" : 4758, "quadra_rateio" :
156.03 }
```

$in

O operador $in quando usado com um array permite verificar se algum dos elementos de um array está contido no array de elementos do operador.

```
> db.megasena.find( {uf: {$in: ["TO", "AC"]}}
)
{ "_id" : 287, "data" : "15/08/2001",
"dezenas" : [ 60, 47, 3, 33, 25, 13 ],
"sena_ganhadores" : 1, "uf" : [ "TO" ],
"sena_rateio" : 1931474.8, "quina_ganhadores"
: 110, "quina_rateio" : 4034.34,
"quadra_ganhadores" : 5633, "quadra_rateio" :
78.49 }
```

$in


```
{ "_id" : 481, "data" : "19/07/2003",
"dezenas" : [ 42, 6, 24, 44, 22, 13 ],
"sena_ganhadores" : 3, "uf" : [ "AC", "MT",
"SP" ], "sena_rateio" : 6796888.03,
"quina_ganhadores" : 359, "quina_rateio" :
5670.32, "quadra_ganhadores" : 19465,
"quadra_rateio" : 104.19 }
```

$

O operador $ pode ser usado na projeção quando no filtro de seleção existe um critério de seleção por elementos de um array. Desta forma, apenas o primeiro elemento do array que atende o critério de seleção será retornado.

```
> db.alunos.find( {notas: {$gt: 8}} )
{ "_id" : 1, "nome" : "Joao", "notas" : [ 8,
9, 5 ] }
{ "_id" : 2, "nome" : "Maria", "notas" : [ 8,
10, 7 ] }
{ "_id" : 3, "nome" : "Jose", "notas" : [ 10,
10, 9 ] }
{ "_id" : 4, "nome" : "Marcia", "notas" :
[ 7, 6, 9 ] }
{ "_id" : 6, "nome" : "Ana", "notas" : [ 5,
8, 9 ] }
```

$

```
> db.alunos.find( {notas: {$gt: 8}}, {"notas.
$": 1} )
{ "_id" : 1, "notas" : [ 9 ] }
{ "_id" : 2, "notas" : [ 10 ] }
{ "_id" : 3, "notas" : [ 10 ] }
{ "_id" : 4, "notas" : [ 9 ] }
{ "_id" : 6, "notas" : [ 9 ] }
```

$

O uso do operador $ tem algumas condições:

- Apenas um operador $ pode ser usado na projeção.
- Apenas um campo de array pode aparecer no filtro de seleção e apenas do array usado com o operador $. Campos de array adicionais no filtro podem causar comportamento indefinido.
- O filtro deve conter uma única condição sobre o campo do array que participa da projeção. Múltiplas condições podem levar a comportamento indefinido.

$slice

O operador $slice permite retornar apenas uma quantidade limitada de elementos de um array.

```
> db.alunos.find({}, {_id: 0} )
{ "nome" : "Joao", "notas" : [ 8, 9, 5 ] }
{ "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
{ "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
{ "nome" : "Marcia", "notas" : [ 7, 6, 9 ] }
{ "nome" : "Paulo", "notas" : [ 5, 4, 5 ] }
{ "nome" : "Ana", "notas" : [ 5, 8, 9 ] }
```

$slice

```
> db.alunos.find( {}, {_id:0, notas: {$slice: 2}} )
{ "nome" : "Joao", "notas" : [ 8, 9 ] }
{ "nome" : "Maria", "notas" : [ 8, 10 ] }
{ "nome" : "Jose", "notas" : [ 10, 10 ] }
{ "nome" : "Marcia", "notas" : [ 7, 6 ] }
{ "nome" : "Paulo", "notas" : [ 5, 4 ] }
{ "nome" : "Ana", "notas" : [ 5, 8 ] }
```

$slice

O operador $slice pode ser usado com um índice negativo para retornar os últimos elementos do array.

```
> db.alunos.find({}, {_id: 0} )
{ "nome" : "Joao", "notas" : [ 8, 9, 5 ] }
{ "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
{ "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
{ "nome" : "Marcia", "notas" : [ 7, 6, 9 ] }
{ "nome" : "Paulo", "notas" : [ 5, 4, 5 ] }
{ "nome" : "Ana", "notas" : [ 5, 8, 9 ] }
```

$slice

```
> db.alunos.find( {}, {_id:0, notas: {$slice: -2}} )
{ "nome" : "Joao", "notas" : [ 9, 5 ] }
{ "nome" : "Maria", "notas" : [ 10, 7 ] }
{ "nome" : "Jose", "notas" : [ 10, 9 ] }
{ "nome" : "Marcia", "notas" : [ 6, 9 ] }
{ "nome" : "Paulo", "notas" : [ 4, 5 ] }
{ "nome" : "Ana", "notas" : [ 8, 9 ] }
```

$slice

O operador $slice pode ser usado para retornar uma faixa de elementos do array quando usado no formato $slice: [skip , limit] onde skip é o número de elementos a ignorar e limit é o número de elementos a retornar.

```
> db.alunos.find({}, {_id: 0} )
{ "nome" : "Joao", "notas" : [ 8, 9, 5 ] }
{ "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
{ "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
{ "nome" : "Marcia", "notas" : [ 7, 6, 9 ] }
{ "nome" : "Paulo", "notas" : [ 5, 4, 5 ] }
{ "nome" : "Ana", "notas" : [ 5, 8, 9 ] }
```

$slice

```
> db.alunos.find( {}, {_id:0, notas: {$slice: [1,1]}} )
{ "nome" : "Joao", "notas" : [ 9 ] }
{ "nome" : "Maria", "notas" : [ 10 ] }
{ "nome" : "Jose", "notas" : [ 10 ] }
{ "nome" : "Marcia", "notas" : [ 6 ] }
{ "nome" : "Paulo", "notas" : [ 4 ] }
{ "nome" : "Ana", "notas" : [ 8 ] }
```

$slice

O operador $slice também pode ser usado para retornar uma faixa dos últimos elementos do array. Porém neste caso é informada a posição de início e não o número de elementos a ignorar.

```
> db.alunos.find({}, {_id: 0} )
{ "nome" : "Joao", "notas" : [ 8, 9, 5 ] }
{ "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
{ "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
{ "nome" : "Marcia", "notas" : [ 7, 6, 9 ] }
{ "nome" : "Paulo", "notas" : [ 5, 4, 5 ] }
{ "nome" : "Ana", "notas" : [ 5, 8, 9 ] }
```

$slice

```
> db.alunos.find( {}, {_id:0, notas: {$slice:
[-1,1]}} )
{ "nome" : "Joao", "notas" : [ 5 ] }
{ "nome" : "Maria", "notas" : [ 7 ] }
{ "nome" : "Jose", "notas" : [ 9 ] }
{ "nome" : "Marcia", "notas" : [ 9 ] }
{ "nome" : "Paulo", "notas" : [ 5 ] }
{ "nome" : "Ana", "notas" : [ 9 ] }
```

Exercícios

1) Encontrar a venda para a cliente Laura de 17 anos

2) Relacionar o id, data e dados do cliente das vendas para clientes do sexo masculino

3) Relacionar a data da venda, nome e idade do cliente

4) Relacionar a data da venda, nome do cliente e nome dos produtos comprados

5) Encontrar o sorteiro da megasena que teve as dezenas 6,44,58,41,36,48 sorteadas

6) Encontrar os aeroportos que tem conexão com BOS

7) Encontrar os sorteios da megasena em que o estado do sexto ganhador seja SP

Exercícios

8) Encontrar os sorteios da megasena que tenham ao mesmo tempo ganhadores de São Paulo, Minas Gerais e Rio de Janeiro

9) Relacionar o nome dos cliente que compraram Borracha

10) Relacionar o nome dos cliente que compraram Fichário ou Post-it

11) Encontrar as vendas onde o produto Caderno foi vendido por 14.00

12) Encontrar as vendas onde o produto Caderno foi vendido por um preço maior que 15.00

13) Relacionar o código e data da venda e dados do produto Corretivo, para as vendas que tem Corretivo