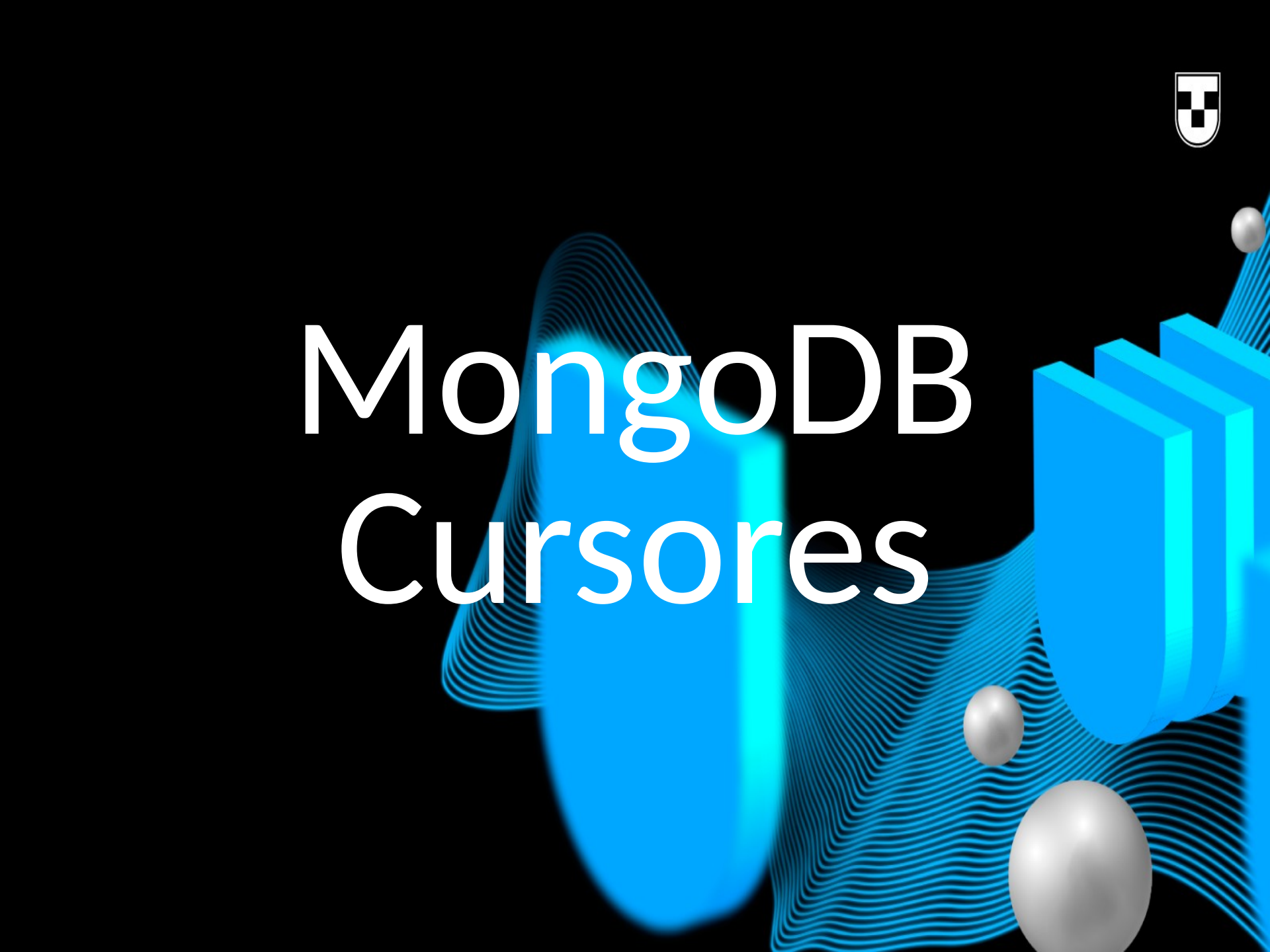

UNITAU
Universidade de Taubaté


MongoDB
Cursores

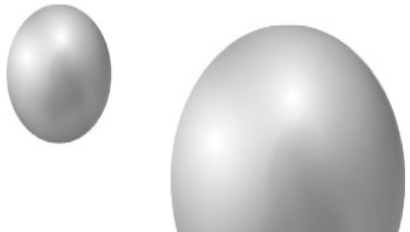
Cursores

O método **db.collection.find()** retorna um cursor que pode ser utilizado com outros métodos de cursores ou utilizados em scripts.

```
> var myCursor =
db.produtos.find({quantidade:20})
> while(myCursor.hasNext()) {
        printjson(myCursor.next())
    }
{ "_id" : 1, "item" : "caderno", "quantidade"
: 20 }
{ "_id" : 3, "item" : "envelope",
"quantidade" : 20, "minimo" : 10 }
```

hasNext

O método **cursor.hasNext()** retorna **true** se o cursor tem documentos remanescentes.



next

O método **cursor.next()** retorna o próximo documento do cursor.

```
> var myCursor =
db.produtos.find( {quantidade: 20}, {_id:
0} )
> myCursor.hasNext()
true
> myCursor.next()
{ "item" : "caderno", "quantidade" : 20 }
> myCursor.hasNext()
true
> myCursor.next()
{ "item" : "envelope", "quantidade" : 20,
"minimo" : 10 }
> myCursor.hasNext()
false
```

next

```
> myCursor.next()
2018-09-05T04:02:39.856-0300 E QUERY
[thread1] Error: error hasNext: false :
DBQuery.prototype.next@src/mongo/shell/
query.js:305:1
@(shell):1:1
```

size

O método **cursor.size()** retorna o número de documentos do cursor.

```
> var myCursor =
db.produtos.find( {quantidade: 20}, {_id:
0} )
> myCursor.size()
2
```

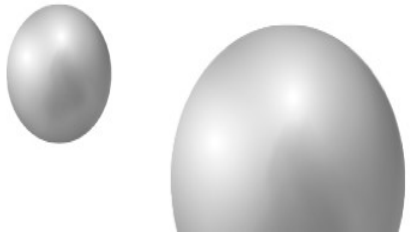
objsLeftInBatch

O método **cursor.objsLeftInBatch()** retorna o número de documentos remanescentes do cursor.

```
> var myCursor =
db.produtos.find( {quantidade: 20}, {_id:
0} )
> myCursor.size()
2
> myCursor.objsLeftInBatch()
2
> myCursor.next()
{ "item" : "caderno", "quantidade" : 20 }
> myCursor.size()
2
> myCursor.objsLeftInBatch()
1
```

objsLeftInBatch

```
> myCursor.next()
{ "item" : "envelope", "quantidade" : 20,
"minimo" : 10 }
> myCursor.size()
2
> myCursor.objsLeftInBatch()
0
```



map

O método **cursor.map()** aplica uma função JavaScript de mapeamento aos documentos do cursor e retorna o resultado da função em um array.

```
cursor.map(function)
```

```
> 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 ] }
```

map

```
> db.alunos.find().map(
    function(u) {
      var soma=0;
      for (var i=0; i<u.notas.length; i++) {
        soma += u.notas[i];
      }
      return soma;
    } )
[ 22, 25, 29, 22, 14, 22 ]
```

forEach

O método cursor.forEach() aplica uma função JavaScript a cada documento do cursor.

```
cursor.forEach(function)
```

```
> db.alunos.find().forEach(
    function(u) {
      var soma=0;
      for (var i=0; i<u.notas.length; i++) {
        soma += u.notas[i]; }
      print( u.nome+" "+soma);
    } )
Joao 22
Maria 25
Jose 29
Marcia 22
Paulo 14
Ana 22
```

toArray

O método **cursor.toArray()** retorna um array que contém todos os documentos do cursor.

```
> var prd = db.produtos.find().toArray()
> if (prd.length > 0) { printjson (prd[0]); }
{ "_id" : 1, "item" : "caderno", "quantidade"
: 20 }
```

pretty

O método **cursor.pretty()** configura o cursor para apresentar os resultados em um formato mais fácil para leitura.

```
> db.acervo.find( {_id: "cm"} ).pretty()
{
        "_id" : "cm",
        "editora" : "Editora Ciencia
Moderna",
        "sede" : {
                "pais" : "Brasil",
                "estado" : "RJ",
                "cidade" : "Rio de Janeiro"
        },
```

pretty

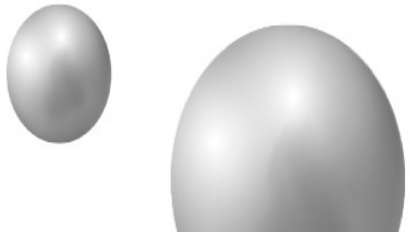
```
        "livros" : [
                {
                        "titulo" :
"Programacao em Linguagem C",
                        "isbn" :
"8573939494",
                        "ano" : "2010",
                        "edicao" : "1"
                }
        ]
}
```

count

O método **cursor.count()** pode ser aplicado a um cursor para contar os documentos do cursor.

```
> db.megasena.find( {sena_ganhadores: {$gt: 0}} ).count()
486
```

Em clusters fragmentados, cursores obtidos sem um filtro no método **find()** pode ter um resultado incorreto para o método **count()** em determinadas circunstâncias.



sort

O método **cursor.sort()** ordena os documentos do cursor. Deve ser passado como parâmetro um documento no formato com os campos a ordenar com o valor **1** para ordem ascendente ou **-1** para ordem descendente.

```
cursor.sort( { field1: value1, field2: value2, ... } )
```

```
> db.livro.find( {}, {_id:0, titulo:1, editora_id:1} ).sort( {editora_id: 1, titulo: -1} )
{ "titulo" : "Programacao em Linguagem C", "editora_id" : "cm" }
{ "titulo" : "Introducao ao MongoDB", "editora_id" : "novatec" }
{ "titulo" : "Arduino Basico", "editora_id" : "novatec" }
```

sort

```
{  "titulo"  :  "SQL  Tuning",  "editora_id"  :
"oreilly" }
{ "titulo" : "Programming With QT",
"editora_id" : "oreilly" }
```

sort

Quando comparando valores de diferentes tipos, o BSON usa a seguinte ordem de comparaçao, do menor para o maior:

1. **MinKey**
2. **Null**
3. **Numbers** (ints, longs, doubles, decimals)
4. **Symbol, String**
5. **Object**
6. **Array**
7. **BinData**
8. **ObjectId**
9. **Boolean**
10. **Date**
11. **Timestamp**
12. **Regular Expression**
13. **MaxKey**

limit

O método **cursor.limit()** determina o número máximo de documentos que o cursor deve retornar.

```
cursor.limit(<number>)
```

```
> 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 ] }
> db.alunos.find({},{_id:0}).limit(3)
{ "nome" : "Joao", "notas" : [ 8, 9, 5 ] }
{ "nome" : "Maria", "notas" : [ 8, 10, 7 ] }
{ "nome" : "Jose", "notas" : [ 10, 10, 9 ] }
```

skip

O método **cursor.skip()** determina o número de documentos que deve ser ignorado pelo cursor.

```
cursor.skip(<offset>)
```

```
> 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 ] }
> db.alunos.find({},{_id:0}).skip(4)
{ "nome" : "Paulo", "notas" : [ 5, 4, 5 ] }
{ "nome" : "Ana", "notas" : [ 5, 8, 9 ] }
```

close

O método **cursor.close()** fecha o cursor e libera os recursos do servidor usados para o cursor.

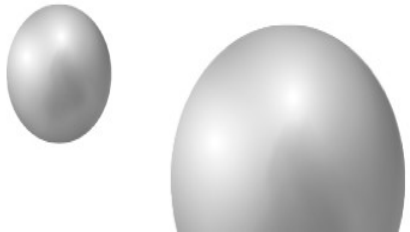
O servidor encerra automaticamente cursores que não tem mais resultados remanescentes ou que estejam inativos por um período de tempo.

Para evitar que o cursor seja fechado automaticamente é necessário utilizar o método **cursor.noCursorTimeout()**.

isClosed

O método **cursor.isClosed()** retorna **true** se o cursor esta fechado.

Um cursor fechado ainda pode ter documentos remanescentes.



isExhausted

O método **cursor.isExhausted()** retorna **true** se o cursor esta fechado e não tem mais documentos remanescentes.

```
> var myCursor = db.alunos.find().limit( 1 )
> myCursor.isClosed()
true
> myCursor.isExhausted()
false
> myCursor.next()
{ "_id" : 1, "nome" : "Joao", "notas" : [ 8,
9, 5 ] }
> myCursor.isClosed()
true
> myCursor.isExhausted()
true
```