mongoDB

MARCH19|ISCTE - IUL
MongoDBLX
in 2015

Introdução ao MongoDB

Gaspar Brogueira

Quinta-Feira, 19 de Março de 2015

O que é o MongoDB?

*MongoDB (from "Hu**mongo**us") is a scalable, high-performance, open source, schema-free, document-oriented database.* http://mongodb.org

- Base Dados NoSQL Open Source
- Orientada ao Documento
- Guarda documentos em JSON (esquemas dinâmicos)
- Queries em JavaScript
- Indexação Flexível
- Replicação
- Desenvolvido em C++
- Multi-Linguagem (C#, Java, Python, Ruby)

mongoDB

Vantagens do MongoDB

- **Storage Approach**
  - formato BSON
- **Horizontal Scale (Sharding)**
  - distribuição do volume de dados por diversos servidores
- **Schemaless**
  - permite a alteração da estrutura dos dados em tempo real
- **Compound Indexes**
  - melhora a performance
- **Cluster management**
  - fácil adição de novos servidores motivado pelo aumento de dados

mongoDB

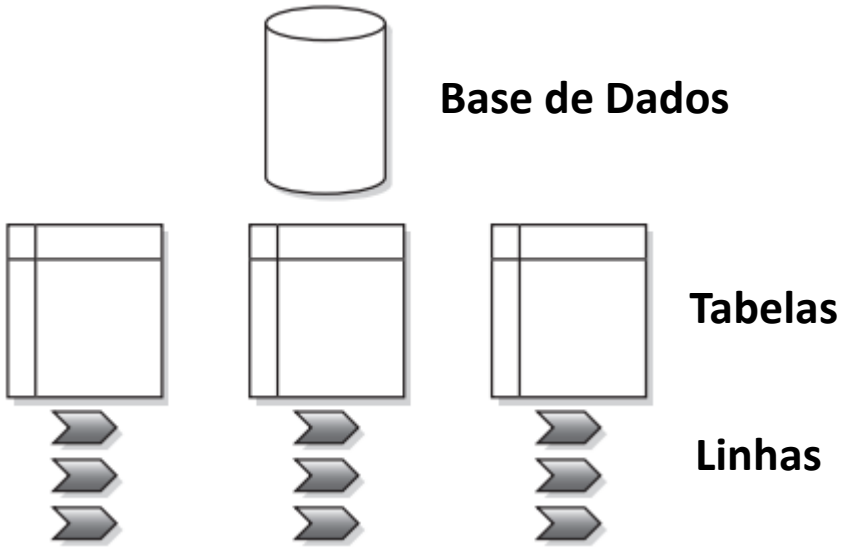
SQL Databases vs NoSQL MongoDB

Anos 1970 como primeira forma de armazenar dados	História	Anos 2000 de acordo com as limitações das SQL DB: replicação, dados não estruturados, escalável
Um tipo (base dados SQL) com poucas variações	Tipos	Diferentes tipos: key-value, colunas, documentos e grafos
Nova informação sobre um item, implica a alteração da BD, ficando indisponível temporariamente	Esquemas	Novos campos relativos a um item, podem ser adicionados sem alteração da estrutura da BD
Misto de Código Aberto (Postgres, MySQL) e Código Fechado (Oracle)	Modelo Desenvolvimento	Código Aberto
Linguagem específica - SQL SELECT campo FROM tabela WHERE...	Manipulação de Dados	Através de APIs orientadas a objectos

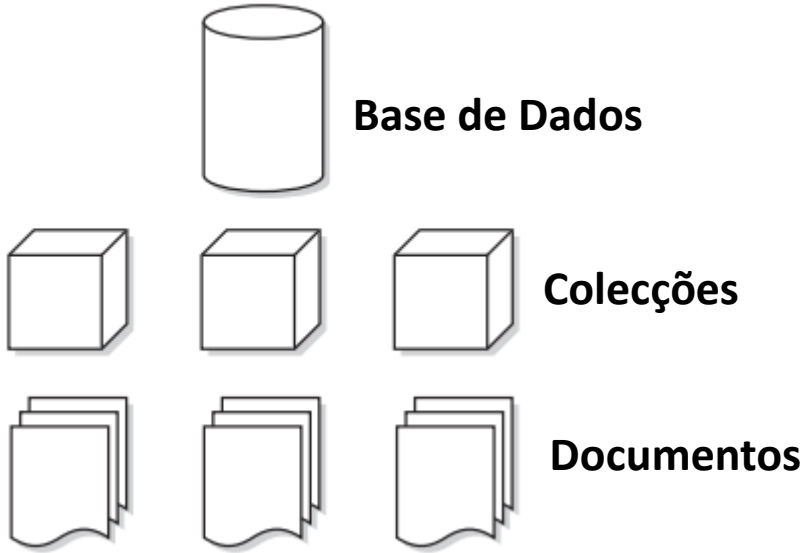
mongoDB

Modelo Relacional vs MongoDB

Modelo Relacional



Modelo MongoDB



mongoDB

Mapeamento de Conceitos

Executável MySQL	Executável Oracle	Executável MongoDB
mysqld	oracle	mongod
mysql	sqlplus	mongo

Termo/Conceito SQL	Termo/Conceito MongoDB
Base de Dados	Base de Dados
Tabela	Colecção
Linha	Documento (BSON)
Coluna	Campo
Índice	Índice
Join de Tabelas	Documentos embebidos e ligados
Chave Primária	Chave Primária
Especificação de uma ou várias colunas como Chave Primária	Chave Primária é colocada automaticamente no campo _id
Agregação (ex. *group by*)	*Aggregation Pipeline*

mongoDB

JavaScript Object Notation

{**JSON** (*JavaScript Object Notation*) **:** formato de transferência de dados}

{ Fácil de ler e escrever **:** por humanos }

{ Fácil de analisar e gerar **:** por máquinas }

{ Formato de texto **:** completamente independente do idioma }

{

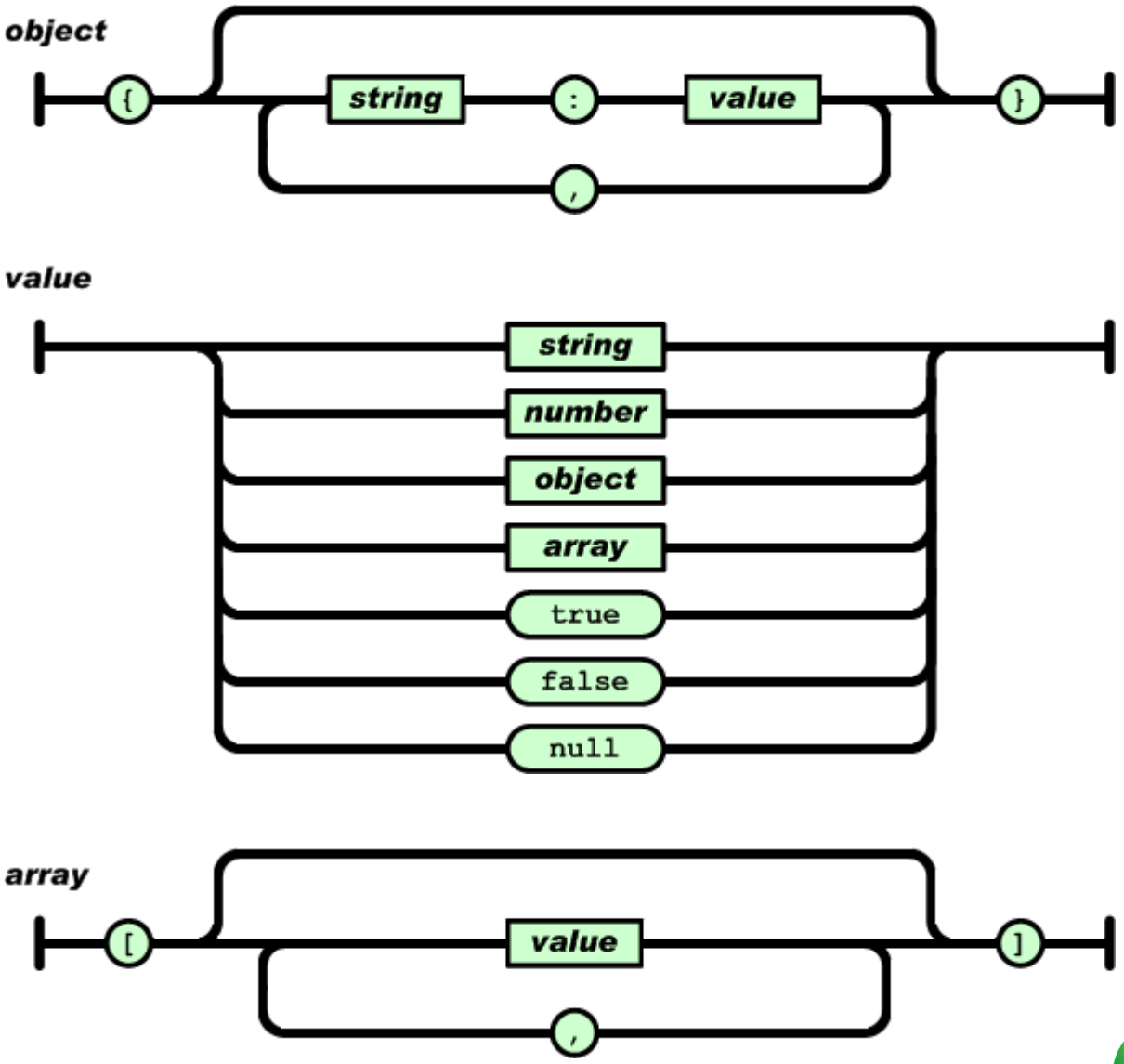
Convenções **:** familiares aos programadores de diversas linguagens,

Linguagens **:** { C , C ++, C # , Java, JavaScript, Perl, Python }

}

mongoDB

JavaScript Object Notation



mongoDB

Documentos JSON


```
{
    "_id" : "547ceb8770928122a8814c98",
    "contributors" : null,
    "truncated" : false,
    "text" : "tal zuka? já pitei",
    "in_reply_to_status_id" : null,
    "id" : "535583499500400642",
    "favorite_count" : 0,
    "source" : "<a href=\"http://twitter.com\"
rel=\"nofollow\">Twitter Web Client</a>",
    "retweeted" : false,
    "coordinates" : null,
    "timestamp_ms" : "1416528031598",
    "entities" : {
        "user_mentions" : [ ],
        "symbols" : [ ],
        "trends" : [ ],
        "hashtags" : [ ],
        "urls" : [ ]
    },
    "in_reply_to_screen_name" : null,
    "id_str" : "535583499500400642",
    "retweet_count" : 0,
    "in_reply_to_user_id" : null,
    "favorited" : false,
    "lang" : "pt",
    "created_at" : "2014-11-21 00:00:31",
    "place" : {
        "country_code" : "PT",
        "url" : "https://api.twitter.com/1.1/geo/id/c1430b24da8e9229.json",
        "country" : "Portugal",
        "place_type" : "city",
        "bounding_box" : {
            "type" : "Polygon",
            "coordinates" : [
                [
                    [
                        -9.2298264,
                        38.6913748
                    ],
                    [
                        -9.2298264,
                        38.7958529
                    ],
                    [
                        -9.0901639,
                        38.7958529
                    ],
                    [
                        -9.0901639,
                        38.6913748
                    ]
                ]
            ]
        },
        "full_name" : "Lisbon",
        "attributes" : {
        },
        "id" : "c1430b24da8e9229",
        "name" : "Lisbon"
```


mongoDB


mongoDB


start

Introdução ao MongoDB

- Criar uma Base de Dados

```
use DATABASE_NAME
```

- Mostrar Bases de Dados disponíveis

```
show dbs / show databases
```

- Criar uma Collection

```
db.createCollection( name )
```

- Mostrar Colecções disponíveis

```
show collections
```

- Mostrar utilizadores da Base de dados

```
show users
```

mongoDB

Create, Read, Update & Delete

CRUD	MySQL	MongoDB
CREATE	INSERT	INSERT
READ	SELECT	FIND
UPDATE	UPDATE	UPDATE
DELETE	DELETE	REMOVE

- INSERT » `db.collection.insert(<document>)`
- FIND » `db.collection.find(<query>, <projection>)`
- UPDATE » `db.collection.update(<query>, <update>, <options>)`
- REMOVE » `db.collection.remove(<query>, <justOne>)`

mongoDB

Insert

- **db.collection.insert(<document>)**

```
db.alunos.insert({nome: "Ricardo", idade: 25, curso: "MOSS",
                  disciplinas: ["A", "B", "C"]})

db.alunos.insert({nome: "Joao", idade: 29, curso: "MOSS",
                  disciplinas: ["A", "B", "D"]})

db.alunos.insert([

        {nome: "Maria", idade: 24, curso: "EI",
               disciplinas: ["A", "C", "E"]},

        {nome: "Carlos", idade: 30, curso: "MOSS",
               disciplinas: ["C", "D", "E"]},

        {nome: "Claudia", idade: 27, curso: "IGE",
               disciplinas: ["A", "C", "E"]}
])
```

mongoDB

Find

- **db.collection.find(<query>, <projection>)**

  - **db.alunos.find()**

```
{ "_id": ObjectId("547cf6e75895bbeed405c32a"), "nome": "Ricardo",
 "idade": 25, "curso": "MOSS", "disciplinas": [ "A", "B", "C" ] }

{ "_id": ObjectId("547cf6ef5895bbeed405c32b"), "nome": "João",
 "idade": 29, "curso": "MOSS", "disciplinas": [ "A", "B", "D" ] }

{ "_id": ObjectId("547cf6f95895bbeed405c32c"), "nome": "Maria",
 "idade": 24, "curso": "MOSS", "disciplinas": [ "A", "C", "E" ] }

{ "_id": ObjectId("547cf6fe5895bbeed405c32d"), "nome": "Carlos",
 "idade": 30, "curso": "MOSS", "disciplinas": [ "C", "D", "E" ] }

{ "_id": ObjectId("547cf7045895bbeed405c32e"), "nome": "Cláudia",
 "idade": 27, "curso": "MOSS", "disciplinas": [ "A", "C", "E" ] }
```

Find

- **db.alunos.find().pretty()**

```
{"_id" : ObjectId("547cf6e75895bbeed405c32a"),
        "nome" : "Ricardo",
        "idade" : 25,
        "curso" : "MOSS",
        "disciplinas" : [
                "A",
                "B",
                "C"
 ]}

{"_id" : ObjectId("550a2a9f8f8b74b28c769fbf"),
        "nome" : "Joao",
        "idade" : 29,
        "curso" : "MOSS",
        "disciplinas" : [
                "A",
                "B",
                "D"
]}
                (…)
```

mongoDB

Find

- **db.alunos.findOne()**

```
{"_id" : ObjectId("547cf6e75895bbeed405c32a"),
        "nome" : "Ricardo",
        "idade" : 25,
        "curso" : "MOSS",
        "disciplinas" : [
                "A",
                "B",
                "C"
 ]}
```

- **db.alunos.find({ nome: "João" })**

```
{ "_id": ObjectId("547cf6ef5895bbeed405c32b"),
  "nome": "João", "idade": 29, "curso": "MOSS",
  "disciplinas": [ "A", "B", "D" ] }
```

mongoDB

Find

- **db.alunos.find({nome: "Joao"}, {_id: 0, nome: 1, idade: 1})**

```
{ "nome" : "Joao", "idade" : 29 }
```

- **db.alunos.find({ idade: { $gt: 26 } }, { nome: 1, _id: 0 })**

```
{ "nome" : "Joao" }
{ "nome" : "Carlos" }
{ "nome" : "Cláudia" }
```

- **db.alunos.distinct("nome")**

```
[ "Ricardo", "Joao", "Maria", "Carlos", "Claudia" ]
```

mongoDB

Find

- `db.alunos.find({ nome: {$regex: "ar" }},{ nome: 1, _id: 0 })`

```
{ "nome" : "Ricardo" }
{ "nome" : "Carlos" }
```

- `db.alunos.find({ profissao: { $exists: true } } )`

-
```
db.alunos.find({ $or: [
                        { name  : { $regex: "C" }},
                        { idade : { $lt: 27 }}
                      ]
                },{ nome: 1, idade: 1, _id: 0}
               )
```

```
{ "nome" : "Maria", "idade" : 24 }
{ "nome" : "Ricardo", "idade" : 25 }
{ "nome" : "Carlos", "idade" : 30 }
{ "nome" : "Cláudia", "idade" : 27 }
```

mongoDB

Operadores de Query e Projecções

- **Comparação**
  - `$gt`
  - `$gte`
  - `$in`
  - `$lt`
  - `$lte`
  - `$ne`
  - `$nin`
- **Lógicos**
  - `$and`
  - `$nor`
  - `$not`
  - `$or`
- **Avaliação**
  - `$mod`
  - `$regex`
  - `$text`
  - `$where`
- **Geoespacial**
  - `$geoIntersects`
  - `$geoWithin`
  - `$nearSphere`
  - `$near`
- **Array**
  - `$all`
  - `$elemMatch`
  - `$size`
- **Projecções**
  - `$`
  - `$elemMatch`
  - `$meta`
  - `$slice`

mongoDB

Operadores de Update

- **$inc** : Incrementa o valor de um campo, com um determinada quantidade

```
db.collection.update( { field: value }, { $inc: { field1: amount } } )
```

- **$rename** : Altera o nome de um campo

```
{$rename: { <old name1>: <new name1>, <old name2>: <new name2>, ... } }
```

- **$set** : Altera o valor de um determinado campo

```
db.collection.update( { field: value1 }, { $set: { field1: value2 } } )
```

- **$unset** : Remove determinado campo

```
db.collection.update( { field: value1 }, { $unset: { field1: "" } } )
```

mongoDB

Update

- **db.collection.update(<query>, <update>, <options>)**

  - **db.alunos.update({nome: "Joao"},{ $set: { idade : 30 }})**

    ```
    { "nome" : "Joao", "idade" : 30 }
    ```

  - **db.alunos.update({nome: "Joao"},{ $inc: { idade : 1 }})**

    ```
    { "nome" : "Joao", "idade" : 31 }
    ```

  - **db.alunos.update({nome: "Maria"}, { $unset: { curso : 1 }})**

    ```
    { "nome": "Maria", "idade": 24,
      "disciplinas": [ "A", "C", "E" ]}
    ```

  - **db.alunos.update({nome: "João"},{$push: {disciplinas: "F" }})**

    ```
    { "nome" : "João", "idade" : 30, "curso" : "MOSS",
      "disciplinas" : [ "A", "B", "D", "F" ] }
    ```

Update

Nota: Os campos não incluídos no Update são apagados!

- **db.alunos.update(**
  { nome: "Maria"},
  { nome: "Ana", profissao: "Estudante" }
)

```
{ "nome" : "Ana", "profissao" : "Estudante" }
```

- **db.alunos.update(**
  { nome: "Maria"},
  { nome: "Ana", idade: 24, curso: "MOSS",
  disciplinas: ["A", "C", "E"],
  profissao: "Estudante" }
)

```
{
    "nome": "Ana", "idade": 24, "curso": "MOSS",
    "disciplinas" : [ "A", "C", "E" ],
    "profissao" : "Estudante"
}
```

mongoDB

Remove

- **db.collection.remove(<query>, <justOne>)**

  - **db.alunos.remove({nome:"Maria"})**

    > Remove a aluna Maria

  - **db.alunos.remove()**

    > Remove todos os documentos da collection

mongoDB

Arrays

- **db.arrays.insert({ _id :0, a: [1,2,3,4]})**
- **db.arrays.update({ _id: 0 }, { $set: { "a.2" : 5 } })**

```
{ "_id" : 0, "a" : [ 1, 2, 5, 4 ] }
```

- **db.arrays.update({ _id: 0 }, { $push: { a : 6 } })**

```
{ "_id" : 0, "a" : [ 1, 2, 5, 4, 6 ] }
```

- **db.arrays.update({ _id: 0 }, { $push: { a : 6 } })**

```
{ "_id" : 0, "a" : [ 1, 2, 5, 4, 6, 6 ] }
```

- **db.arrays.update({ _id: 0 }, { $pop: { a : 1 } })**

```
{ "_id" : 0, "a" : [ 1, 2, 5, 4, 6 ] }
```

- **db.arrays.update({ _id: 0 }, { $pop: { a : -1 } })**

```
{ "_id" : 0, "a" : [ 2, 5, 4, 6 ] }
```

Arrays

- **db.arrays.update({ _id: 0 }, { $pushAll: { a : [7,8,9] } })**

```
{ "_id" : 0, "a" : [ 2, 5, 4, 6, 7, 8, 9 ] }
```

- **db.arrays.update({ _id: 0 }, { $pull: { a : 5 } })**

```
{ "_id" : 0, "a" : [ 2, 4, 6, 7, 8, 9 ] }
```

- **db.arrays.update({ _id: 0 }, { $pullAll: { a : [2,4,8] } })**

```
{ "_id" : 0, "a" : [ 6, 7, 9 ] }
```

- **db.arrays.update({ _id: 0 }, { $addToSet: { a : 5 } })**

```
{ "_id" : 0, "a" : [ 6, 7, 9, 5 ] }
```

- **db.arrays.update({ _id: 0 }, { $addToSet: { a : 5 } })**

```
{ "_id" : 0, "a" : [ 6, 7, 9, 5 ] }
```

Outras operações úteis

- Contar número de documentos de uma Collection

```
db.collection.count()
```

- Limitar o número de documentos retornado por uma query

```
db.collection.find().limit( < MAX_RESULTS > )
```

- Renomear uma Collection

```
db.collection.renameCollection('novo_nome')
```

- Eliminar uma Collection

```
db.collection.drop()
```

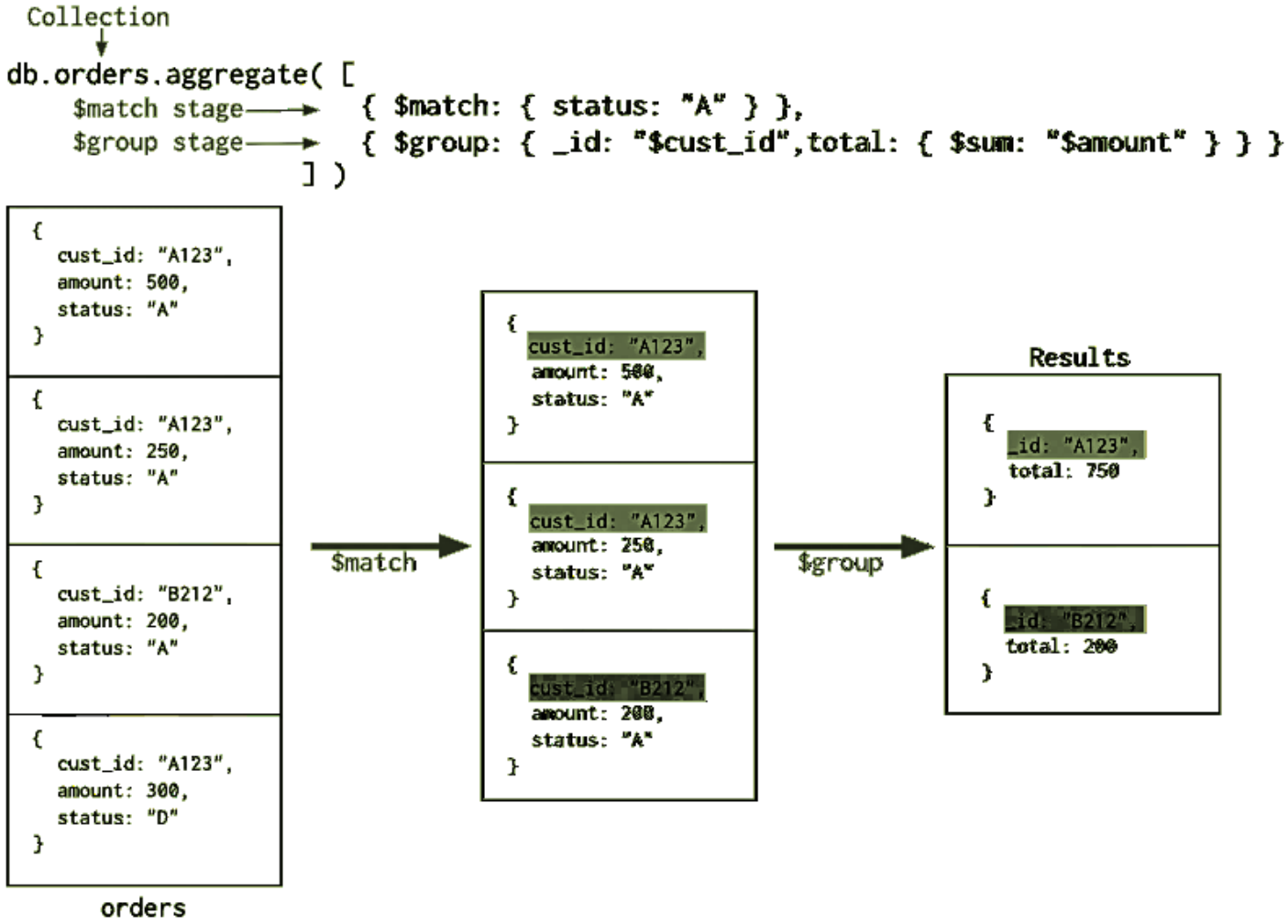
- Eliminar uma Base de Dados

```
use DATABASE_NAME
db.dropDatabase()
```

mongoDB

Agregação

- Agregações são operações que efectuam **processamento dos dados** e **retornam o resultado desse processamento**



Fonte : http://docs.mongodb.org/manual/core/aggregation-introduction/

mongoDB

Operadores de Agregação em Pipeline

- `db.collection.aggregate( [ { <stage> }, ... ] )`
- `{ <operator>: [ <argument1>, <argument2> ... ] }`

- $group
- $limit
- $match
- $project
- $skip
- $sort
- $unwind

- $and
- $not
- $or
- $cmp
- $eq
- $gt

- $add
- $divide
- $mod
- $multiply
- $subtract

- $concat
- $strcasecomp
- $substr
- $toLower
- $toUpper

- $dayOfMonth
- $dayOfWeek
- $dayOfYear
- $hour
- $millisecond
- $minute
- $month
- $second
- $week
- $year

mongoDB

Exemplo de Agregação

Calcular o número de disciplinas de cada aluno:

-
```
db.alunos.aggregate([

        { $unwind : "$disciplinas" },
        { $group  : { _id: "$nome", count: { $sum: 1 } } },
        { $sort   : { count: -1 } }
    ])
```

-
```
db.alunos.aggregate([

        { $match : { nome: "Joao" } },
        { $unwind : "$disciplinas" }
    ])

    {"nome" : "Joao", (…) , "disciplinas" : "A" }
    {"nome" : "Joao", (…) , "disciplinas" : "B" }
    {"nome" : "Joao", (…) , "disciplinas" : "D" }
    {"nome" : "Joao", (…) , "disciplinas" : "F" }
```

mongoDB

Exemplo de Agregação

Calcular o número de disciplinas de cada aluno:

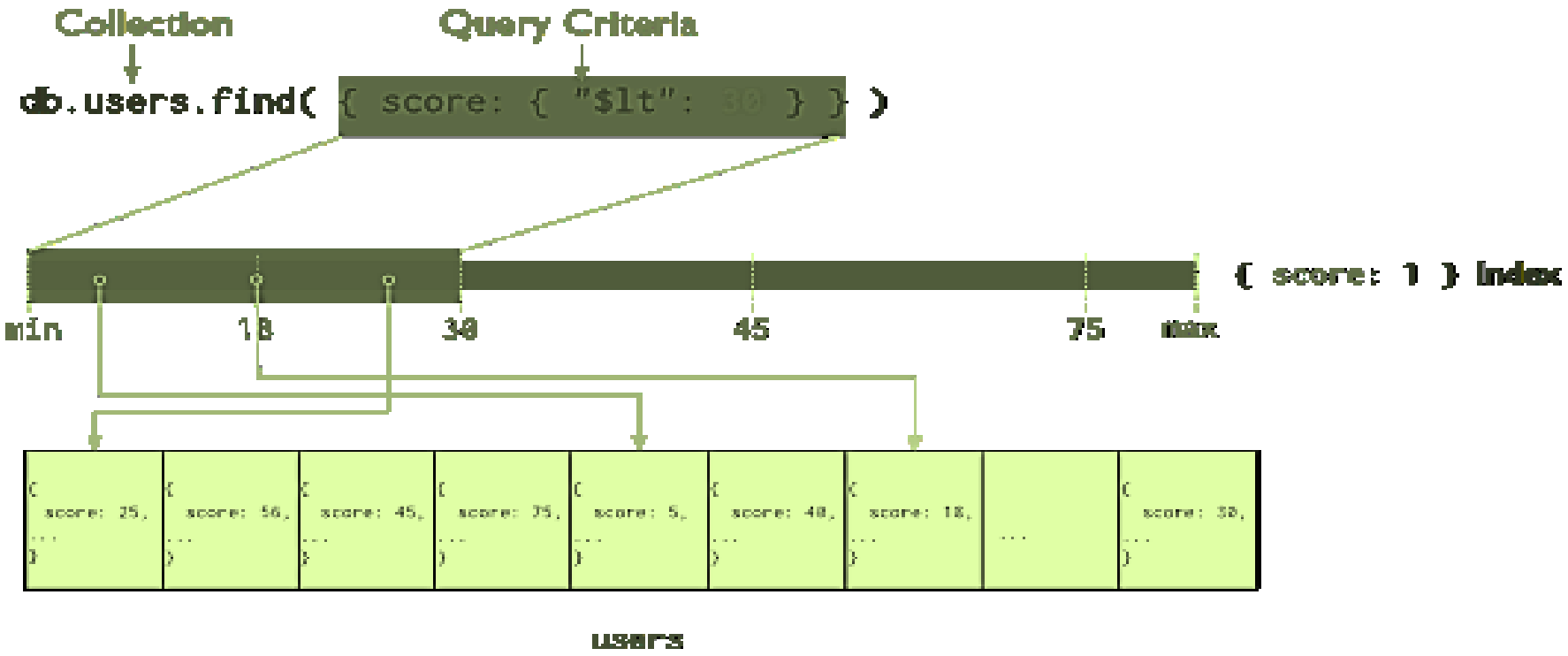
```
db.alunos.aggregate(
    [
        { $unwind : "$disciplinas" },
        { $group  : { _id: "$nome", count: { $sum: 1 } } },
        { $sort   : { count: -1 } }
    ]
)
```

```
{ "_id" : "João",     "count" : 4 }
{ "_id" : "Ana",      "count" : 3 }
{ "_id" : "Carlos",   "count" : 3 }
{ "_id" : "Cláudia",  "count" : 3 }
{ "_id" : "Ricardo",  "count" : 3 }
```

mongoDB

Performance

- Os índices permitem a execução de queries de forma bastante mais eficiente
- Sem a utilização de índices é necessário pesquisar em todos os documentos da collection
- A acção de *collection scan* requer bastante processamento por parte do servidor do MongoDB, devido ao elevado volume de dados a processar



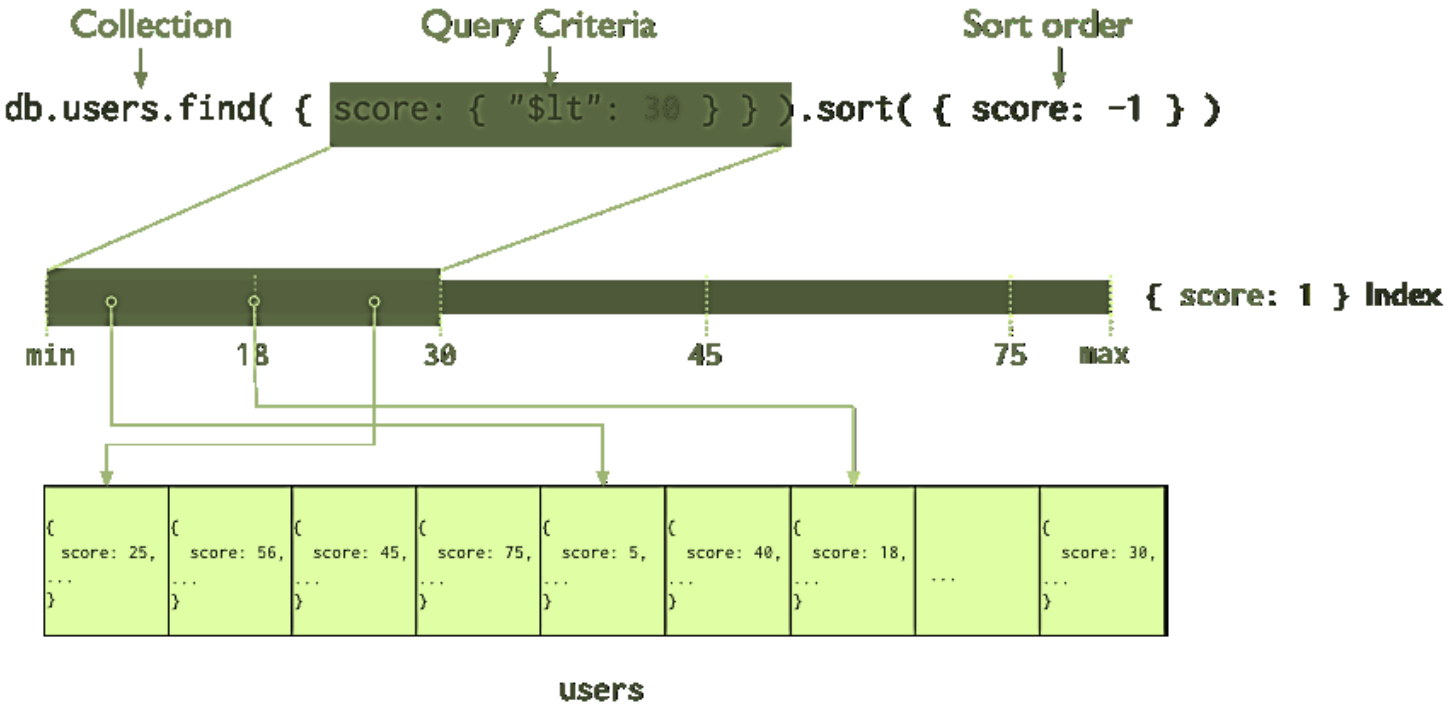
Fonte : http://docs.mongodb.org/manual/core/indexes-introduction/

mongoDB

Performance

- **Resultados Ordenados**

  Os resultados são retornados pela ordem do índice, sem a necessidade de consulta dos documentos



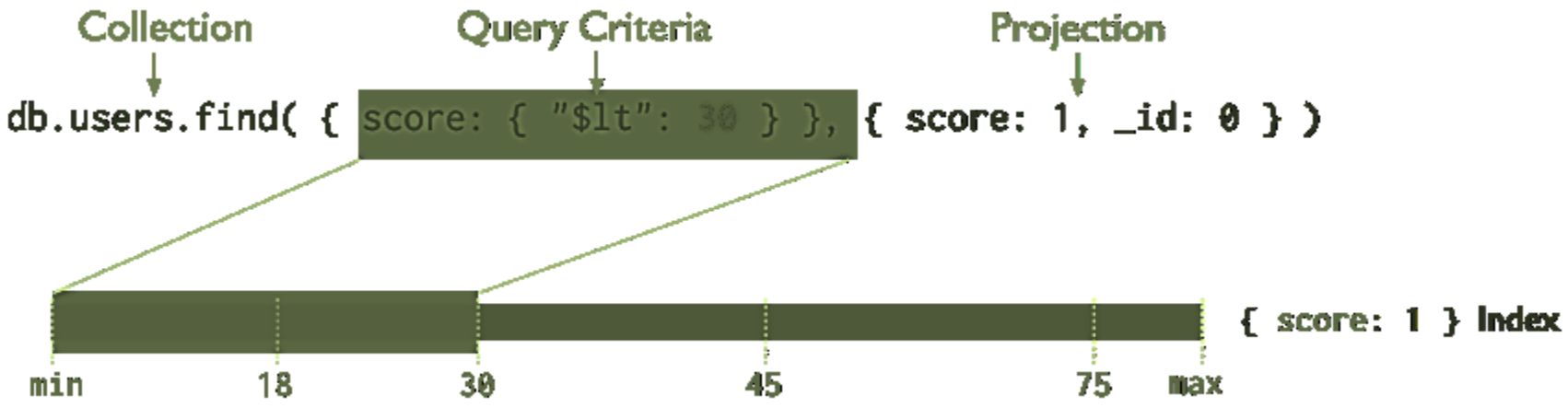
Fonte : http://docs.mongodb.org/manual/core/indexes-introduction/

mongoDB

Performance

- **Cobertura dos Resultados**

  Quando os critérios de pesquisa e a projecção da query incluem apenas os campos do índice, os resultados são retornados sem qualquer pesquisa nos documentos



Fonte : http://docs.mongodb.org/manual/core/indexes-introduction/

mongoDB

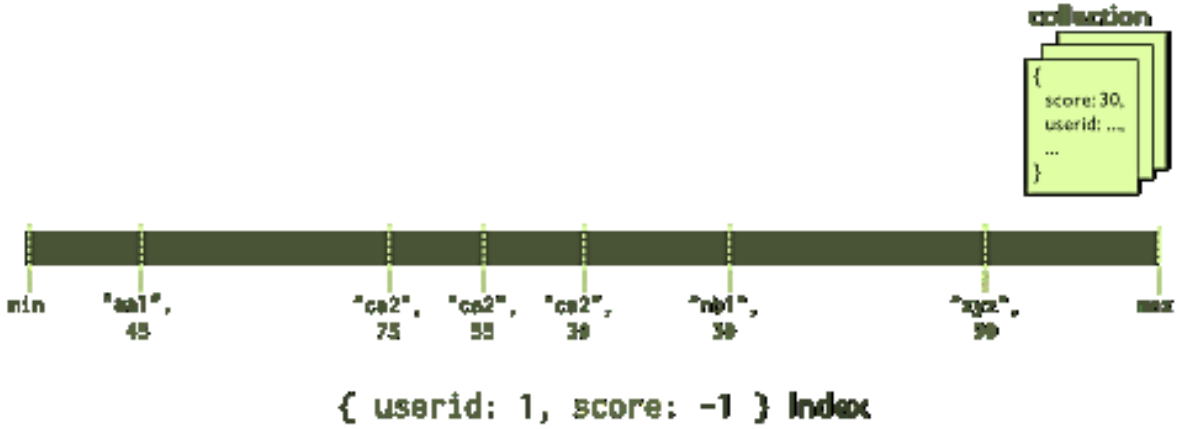
Tipos de Índices

- **_id**

  Todas as *collections* em MongoDB têm por defeito, um índice com o campo **_id**

- **Índice Sim**

- **Índice Com**



Fonte : http://docs.mongodb.org/manual/core/indexes-introduction/

mongoDB

Índices

- **db.collection.ensureIndex()**

```
db.alunos.ensureIndex(  {  idade: 1 } )

db.alunos.ensureIndex( { nome: 1, idade: 1 } )

db.alunos.ensureIndex(
        { nome: 1 },
        { unique: true, dropDups: true }
)

db.collection.getIndexes()

[
        {
                "v" : 1,
                "key" : {
                        "_id" : 1
                },
                "name" : "_id_",
                "ns" : "ISCTE.alunos"
        }
]
```

mongoDB

Python e MongoDB

```
import pymongo
from pymongo import MongoClient

#Conexão ao Servidor do MongoDB
connection = MongoClient('localhost', 27017)

#Ligação à Base de Dados
db = connection.ISCTE

#Selecção da Collection
aluno = db.alunos

#Query
item = aluno.find_one()

print "Nome:" + str(item['nome'])
print "Idade:" + str(item['idade'])
print "Curso:" + str(item['curso'])
```

mongoDB

PHP e MongoDB

```php
<?php

    //$connection = new MongoClient("mongodb://USER:USER@HOST/DATABASE");
    $connection = new MongoClient("mongodb://localhost/DATABASE");

    $collection = $connection->DATABASE->COLLECTION;

    $document = $collection->findOne();

    echo print_r($document);

?>
```

mongoDB

PHP e MongoDB

```
//db.tweets.aggregate( [ {
        $group: { _id : {
                        year:  { $year: "$created_at" },
                        month: { $month: "$created_at" },
                        day:   { $dayOfMonth: "$created_at" }},
                  count: { $sum: 1 } } }
])

$pipeline = array(
    array(
        '$group' => array(
                '_id' => array(
                        'year'  => array('$year' => '$created_at'),
                        'month' => array('$month' => '$created_at'),
                        'day'   => array('$dayOfMonth' => '$created_at')
                ),
                'soma'  => array('$sum' => 1 )
                )
          ),
    array(
        '$sort' => array("_id" => 1),
    )
);
$out = $collection->aggregate($pipeline);
```

mongoDB

NodeJS e MongoDB

```
var Hapi = require('hapi');

var server = new Hapi.Server();
server.connection({ port: 3000 });

server.route({
    method: 'GET',
    path: '/{name}',
    handler: function (request, reply) {

    var util = require('util');
    var mongo = require('mongodb');
    var serverInstance = new mongo.Server('localhost', 27017, {safe:false});
    var dbref = new mongo.Db('ISCTE', serverInstance, {w:1});

    dbref.open(function(err, dbref) {
        dbref.collection('alunos', function(err, collectionref) {

            var newStudent = { "nome":encodeURIComponent(request.params.name) };
            collectionref.insert(newStudent, function (err, result) {
                reply(util.inspect(newStudent, {false, null}));
            });
        });
    });
    }
});
```

mongoDB

NodeJS e MongoDB

```
server.route({
    method: 'GET',
    path: '/printAll',
    handler: function (request, reply) {

    var util = require('util');
    var mongo = require('mongodb');
    var serverInstance = new mongo.Server('localhost', 27017, {safe:false});
    var dbref = new mongo.Db('ISCTE', serverInstance, {w:1});

    dbref.open(function(err, dbref) {
        dbref.collection('alunos', function(err, collectionref) {
            collectionref.find().toArray(function(err, docs) {
                docs.forEach(function(doc) {
                        console.dir(doc);
                });
            });
        });
    });
    }
});
```

mongoDB

NodeJS e MongoDB

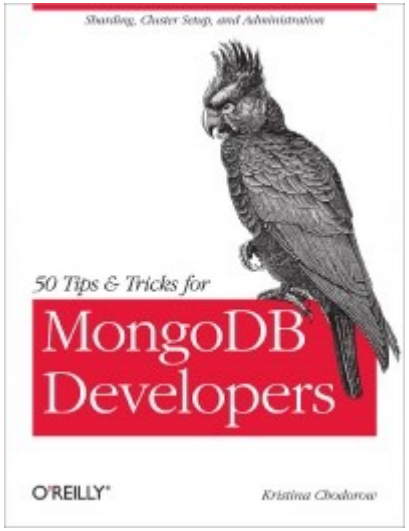
```
server.route({
    method: 'GET',
    path: '/count',
    handler: function (request, reply) {

    var mongo = require('mongodb');
    var serverInstance = new mongo.Server('localhost', 27017, {safe:false});
    var dbref = new mongo.Db('ISCTE', serverInstance, {w:1});

    dbref.open(function(err, dbref) {
        dbref.collection('alunos', function(err, collectionref) {

            var cursor = collectionref.find();
            cursor.count(function (err, amount) {
                reply("Numero de Alunos: " + amount);
            });

        });
    });
    }
});
```

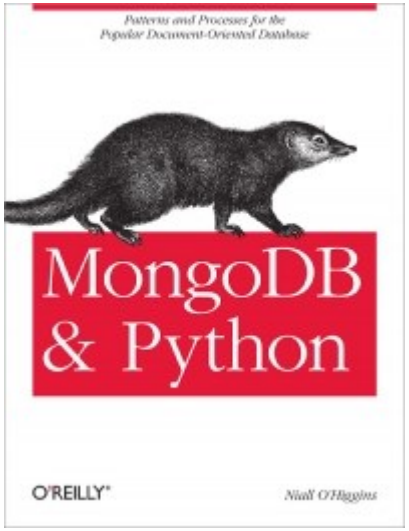
mongoDB

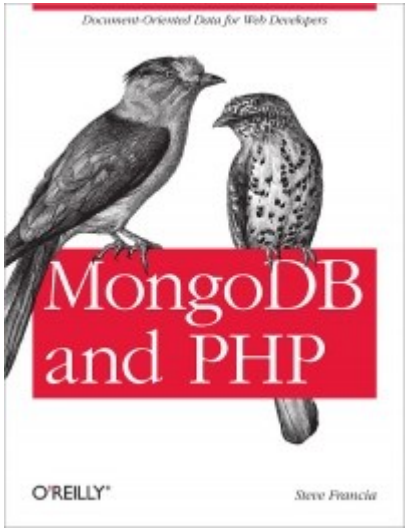
Referências Uteis

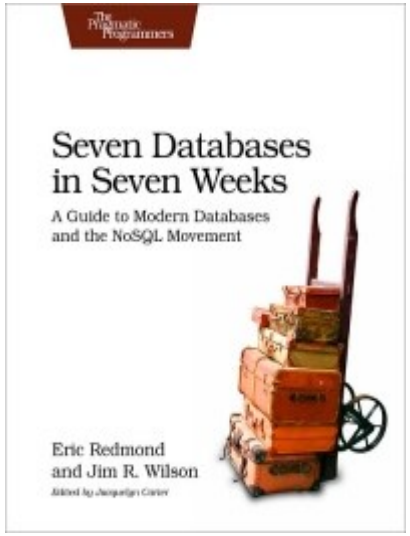
mongoDB

mongolab









mongoDB

{ Obrigado : Thanks }

{ Questões : ??? }

{
Nome : Gaspar Brogueira ,
Email : gmrba@iscte.pt
}