Liberando o poder do aggregation framework do MongoDB no Node

Apresentação

Felipe de Alencar Pinheiro

Web developer at LabTrans

- https://github.com/felpin
- https://twitter.com/felipeapinheiro

ROADMAP

- Introdução ao MongoDB
- Como utilizar o MongoDB no Node.js
- Aggregation Framework

Introdução ao MongoDB

NoSQL

- Quebram o conceito de tabela, coluna e registro
- Rompem com as propriedades ACID

NoSQL

Modelos de dados:

documentos	grafos	chave-valor	wide column
			

MongoDB

- Banco de dados orientado a documentos
- Os dados são salvos no formato BSON

Estrutura

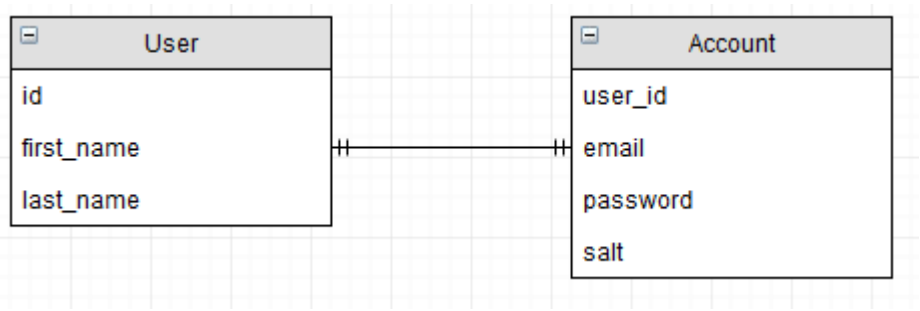
RELACIONAL	NoSQL
banco de dados	banco de dados
tabela	coleção
coluna	~~propriedade~~ *schemaless*
registro	documento

Estrutura

Name	FName	City	Age	Salary
Smith	John	3	35	$280
Doe	Jane	1	28	$325
Brown	Scott	3	41	$265
Howard	Shemp	4	48	$359
Taylor	Tom	2	22	$250

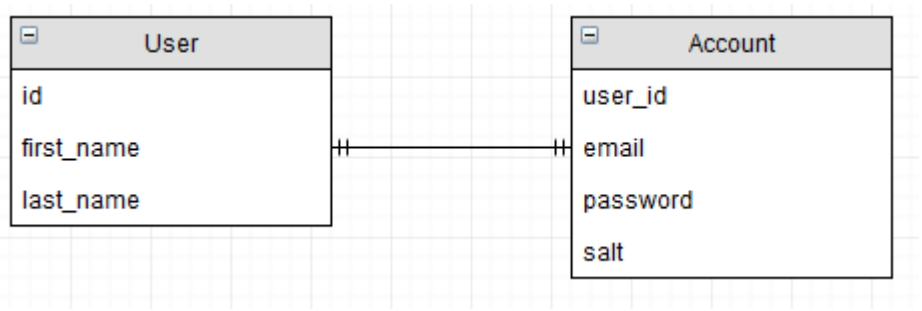

```
"firstName": "John",
"lastName": "Smith",
"age": 25,
"address": {
    "streetAddress": "21 2nd S
    "city": "New York",
    "state": "NY",
    "postalCode": 10021
},
"phoneNumbers": [
    {
        "type": "home",
```

Relações (1:1)




```
{
    "_id": 1,
    "first_name": "John",
    "last_name": "Smith",
    "account": {
        "email":"johnsmith@temporary.com",
        "password":"DwfNUQWtJrL2CHqd",
        "salt": "VOHxR1azr3J7iLoy"
    }
}
```

Relações (1:1)

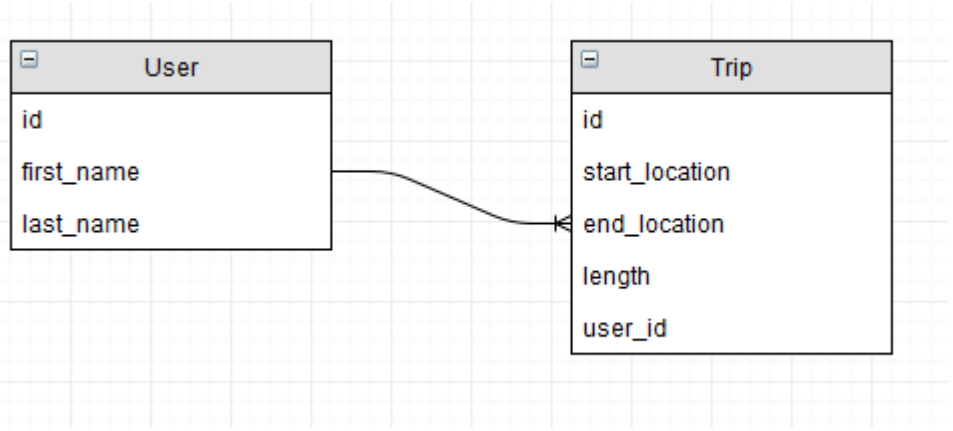


VS

```
// USER
{
    "_id": 1,
    "first_name": "John",
    "last_name": "Smith",
    "account": "johnsmith@temporary.com"
}

// ACCOUNT
{
    "_id": "johnsmith@temporary.com",
    "password":"DwfNUQWtJrL2CHqd",
    "salt": "VOHxR1azr3J7iLoy"
}
```

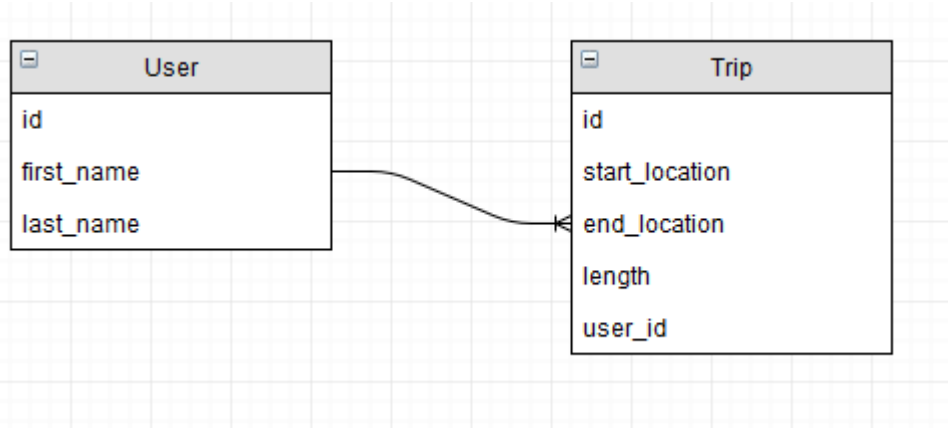
Relações (1:N)



VS

```
{
    "_id": 1,
    "first_name": "John",
    "last_name": "Smith",
    "trips":[
        {
            "start_location": "Florianópolis",
            "end_location": "Porto Alegre",
            "length": "40m"
        },
        {
            "start_location": "Porto Alegre",
            "end_location": "Fortaleza",
            "length": "4h20m"
        }
    ]
}
```

Relações (1:N)

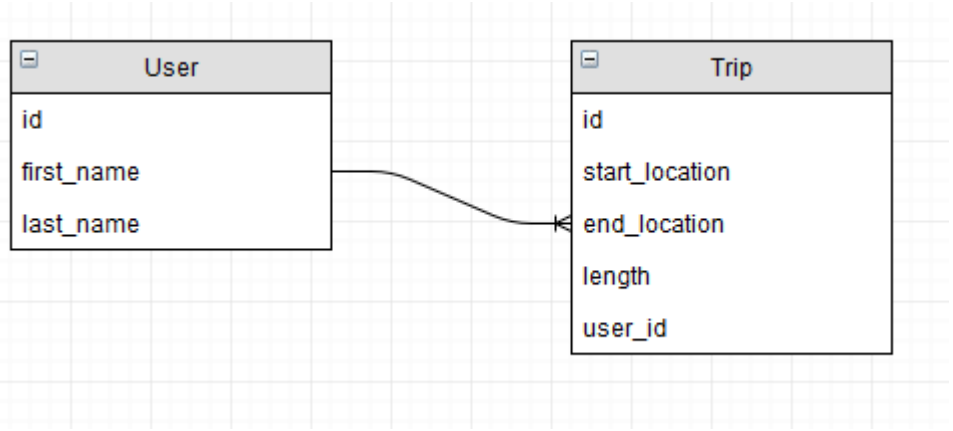


VS

```
// USER
{
    "_id": 1,
    "first_name": "John",
    "last_name": "Smith",
    "trips": [
        "FLN-POA",
        "POA-FOR"
    ]
}
```

```
// TRIP
{
    "_id": "FLN-POA",
    "start_location": "Florianópolis",
    "end_location": "Porto Alegre",
    "length": "40m"
}
{
    "_id": "POA-FOR",
    "start_location": "Porto Alegre",
    "end_location": "Fortaleza",
    "length": "4h20m"
}
```

Relações (1:N)



VS

```
// USER
{
    "_id": 1,
    "first_name": "John",
    "last_name": "Smith"
}
```

```
// TRIP
{
    "_id": "FLN-POA",
    "start_location": "Florianópolis",
    "end_location": "Porto Alegre",
    "length": "40m",
    "user": 1
}
{
    "_id": "POA-FOR",
    "start_location": "Porto Alegre",
    "end_location": "Fortaleza",
    "length": "4h20m",
    "user": 1
}
```

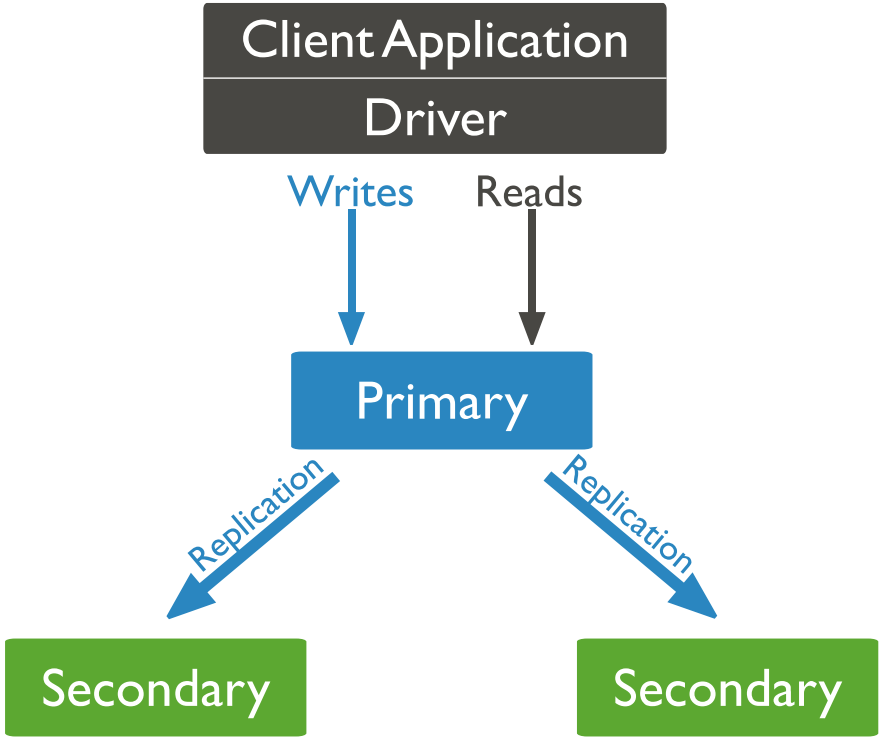
Relações (N:M)

many-to-many?

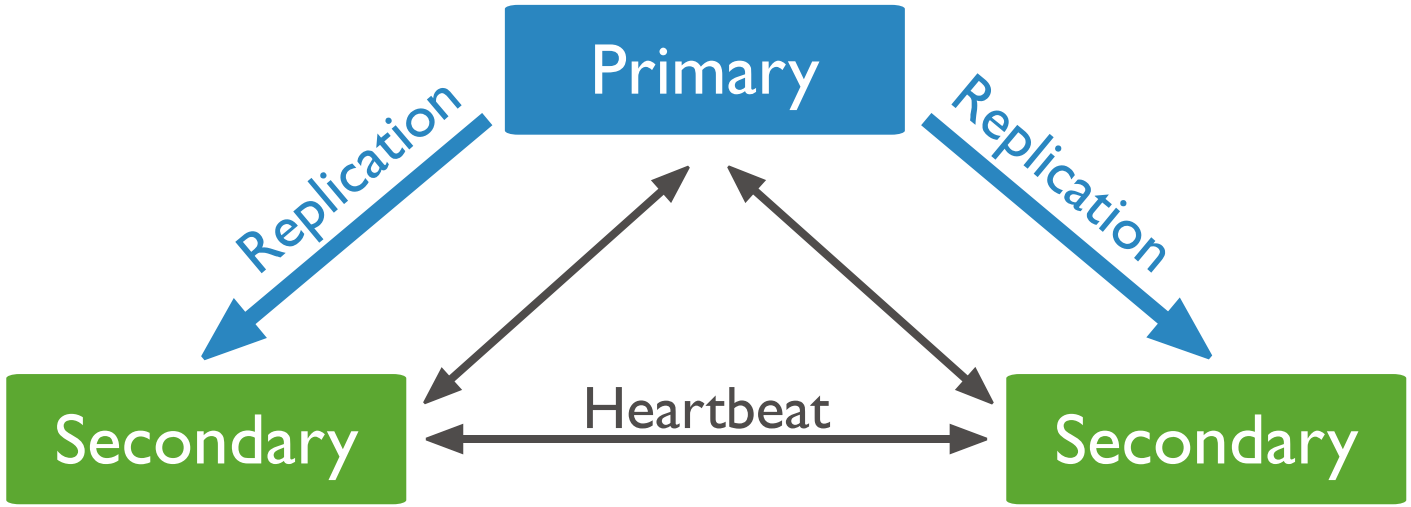
Características

- Todas as operações são atômicas a nível de documento
- Existe apenas o *left outer join* como operador *join* (**$lookup**)
- Utiliza o *WiredTiger* como *storage engine*
- Existe um limite para o documento de 16MB
  - Mídias e arquivos podem ser armazenados com GridFS

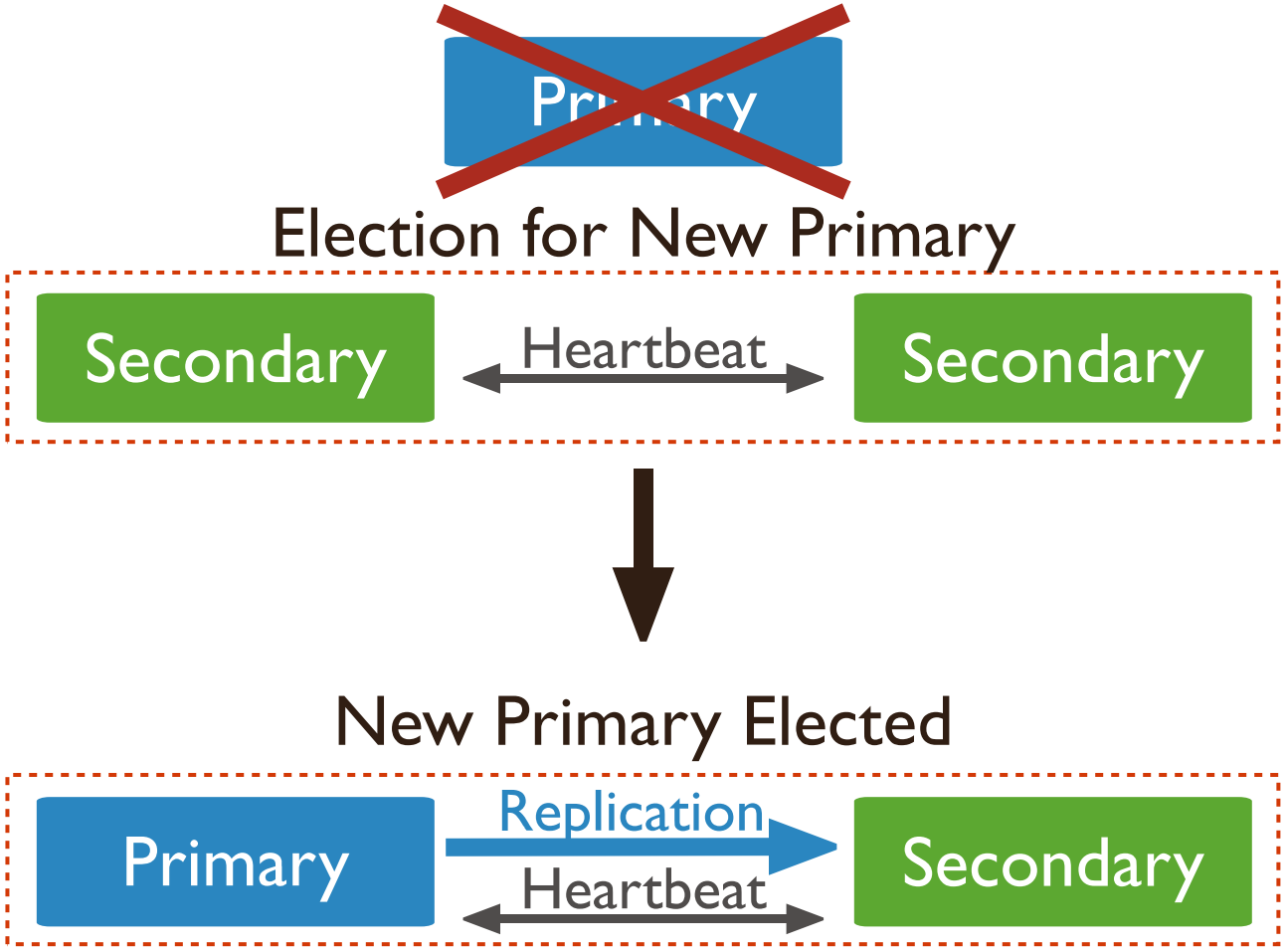
Replication



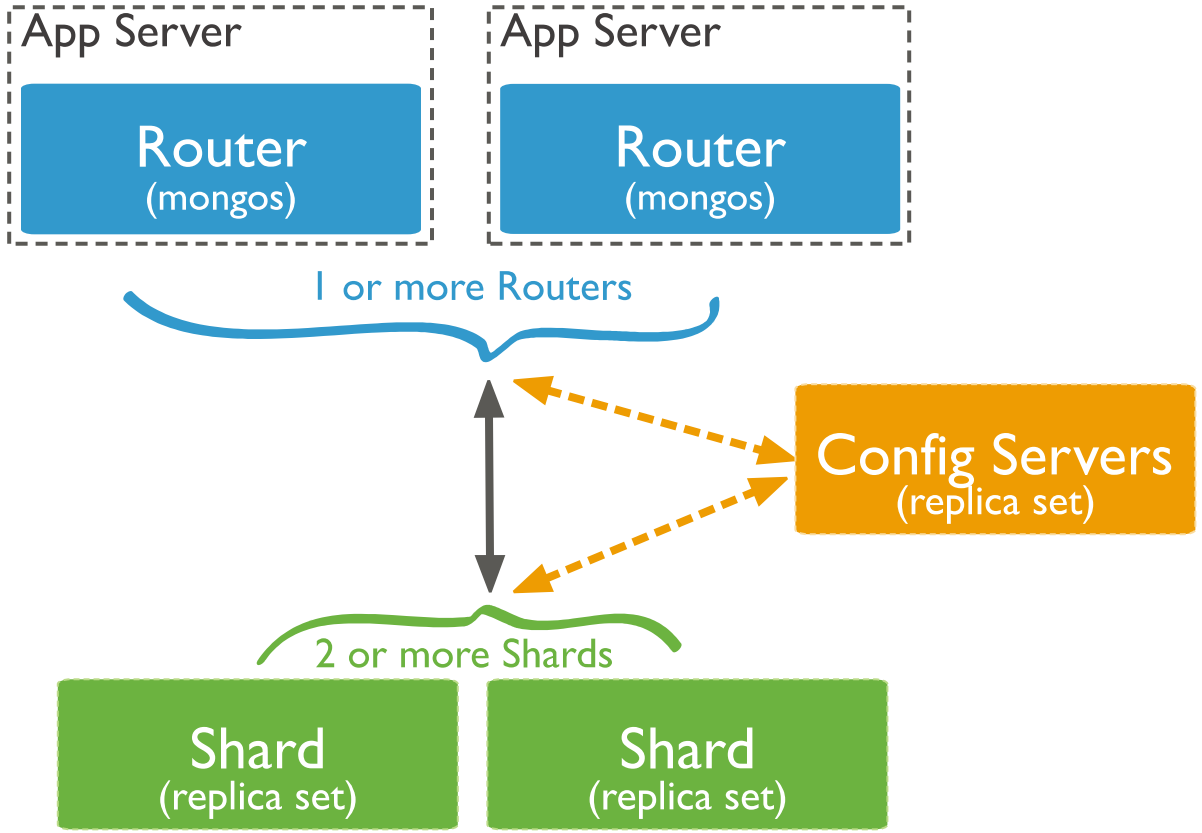
Replication



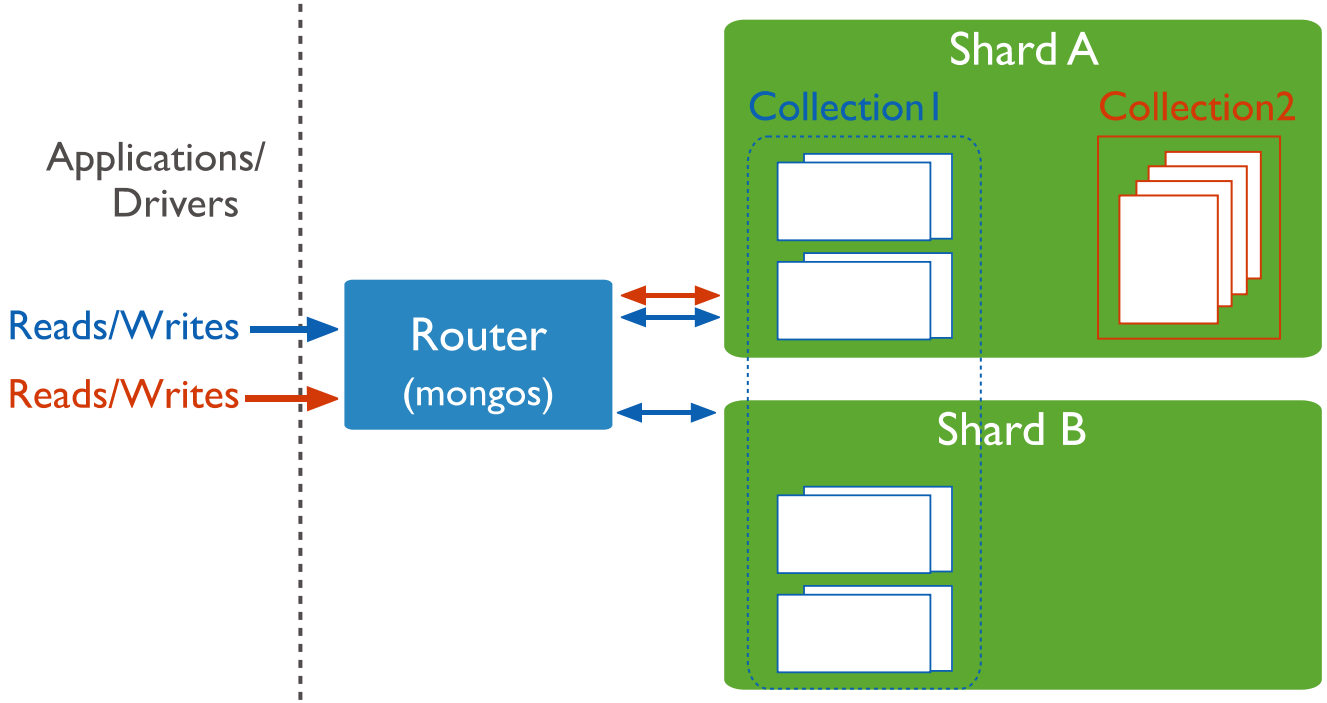
Replication



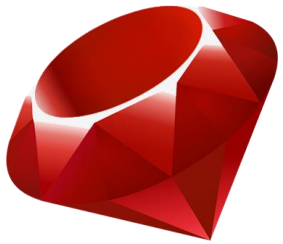
Sharding


App Server
Router
(mongos)
App Server
Router
(mongos)
1 or more Routers
Config Servers
(replica set)
2 or more Shards
Shard
(replica set)
Shard
(replica set)

Sharding


Applications/
Drivers
Reads/Writes
Reads/Writes
Router
(mongos)
Shard A
Collection1
Collection2
Shard B

Drivers



Ferramentas

- MongoDB Compass - GUI for MongoDB
- MongoDB Atlas – Database as a Service
- MongoDB Stitch – Backend as a Service

Who uses?


Adobe


amadeus


amazon.com


BOSCH


CERN


CISCO


comcast


eHarmony


GitHub


KPMG


MetLife


Saks Fifth Avenue


SAP

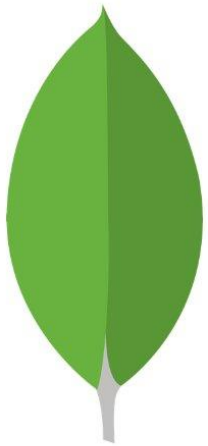

stripe


The New York Times


vmware


Royal Bank of Scotland

Como utilizar o MongoDB no Node.js



npm install mongodb

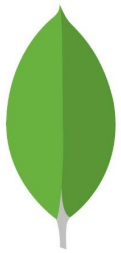
mongoose

npm install mongoose

Comparação

Object Document Mapping (ODM)

mongoose Object Relational Mapping (ORM)

PERFORMANCE	DEVELOPMENT TIME	LEARNING CURVE
	mongoose	

Conclusão...?

Utilização

Nativo

```
const mongodb = require('mongodb');

const url = 'mongodb://localhost:27017/floripajs';

mongodb.MongoClient
    .connect(url)
    .then(db => {
        console.log('YAY... There is a connection');

        // TODO: Do something

        db.close();
    })
    .catch(error => {
        console.error(error);
    });
```

Mongoose

```
const mongoose = require('mongoose');

mongoose.Promise = global.Promise;

const url = 'mongodb://localhost:27017/floripajs';

mongoose
    .connect(url, { useMongoClient: true })
    .then((db) => {
        console.log('YAY... There is a connection');

        // TODO: Do something
    })
    .catch(error => {
        console.error(error);
    });
```

Aggregation Framework

Aggregation Framewrok

- Single Purpose Aggregation Operations
- Map-Reduce
- Aggregation Pipeline

Single Purpose Aggregation Operations

COUNT

```
db.collection('main').count().then((qty) => console.log(qty));
```

```
PS D:\Felipe\Projects\floripajs-17> node .\native\single-count.js
YAY... There is a connection
1000
```

DISTINCT

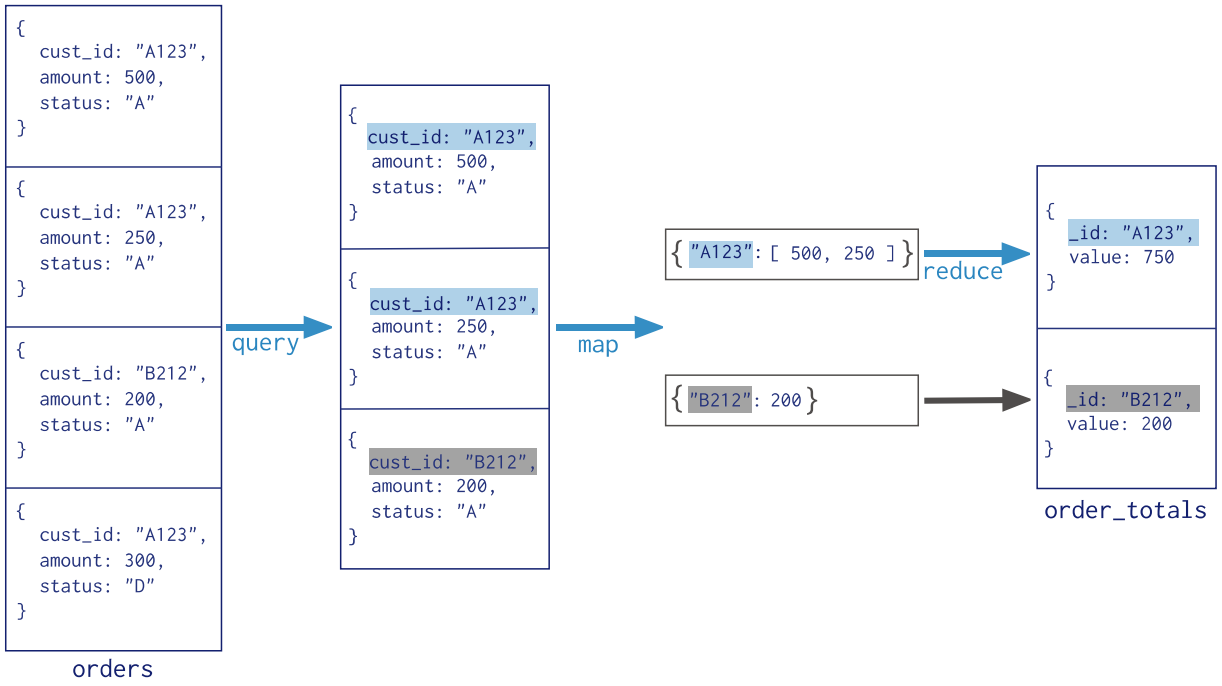
```
db.collection('main').distinct('a').then(r => console.log(r));
```

```
PS D:\Felipe\Projects\floripajs-17> node .\native\single-distinct.js
YAY... There is a connection
[ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ]
```

Map-Reduce

Collection

```
db.orders.mapReduce(
          map      →   function() { emit( this.cust_id, this.amount ); },
          reduce   →   function(key, values) { return Array.sum( values ) },
                       {
          query    →     query: { status: "A" },
          output   →     out: "order_totals"
                       }
                   )
```



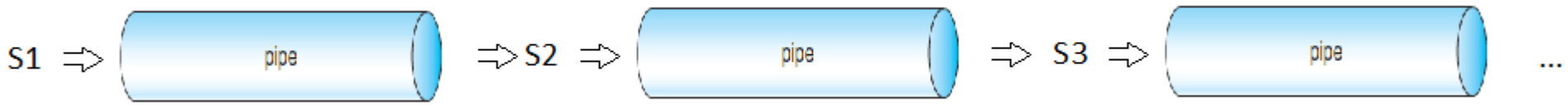
Map-Reduce

```
db.collection('main').mapReduce(
    // MAP
    function () {
        emit(this.a, this.arr.length);
    },
    // REDUCE
    function (key, values) {
        return Array.sum(values)
    },
    // PARAMS
    { out: { replace: 'secondary' } }
)
    .then(() => db.collection('secondary').find().forEach(item => console.log(item)))
```

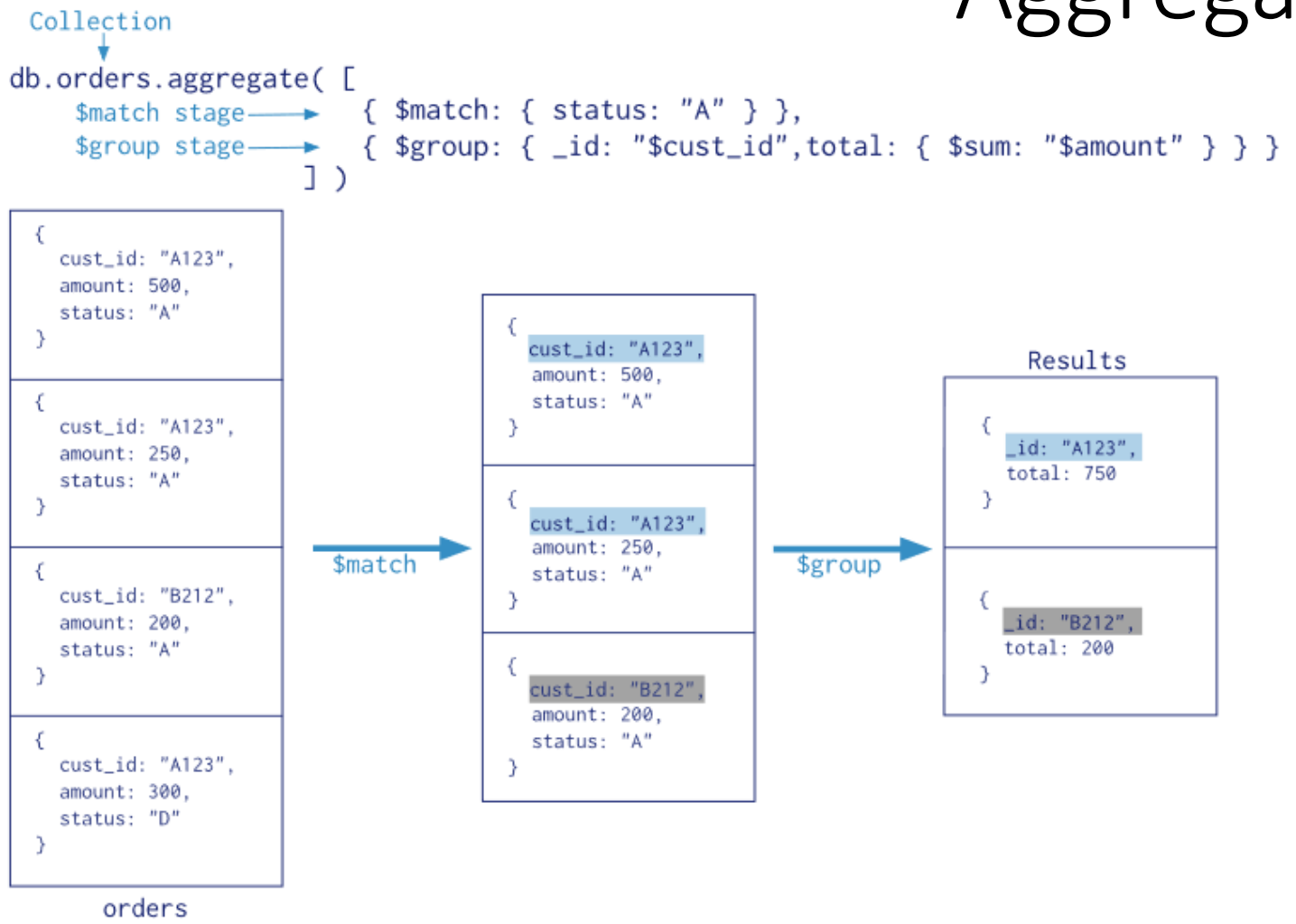
```
PS D:\Felipe\Projects\floripajs-17> node .\native\map-reduce.js
YAY... There is a connection
{ _id: 0, value: 501 }
{ _id: 1, value: 489 }
{ _id: 2, value: 499 }
{ _id: 3, value: 506 }
{ _id: 4, value: 478 }
{ _id: 5, value: 504 }
{ _id: 6, value: 488 }
{ _id: 7, value: 495 }
{ _id: 8, value: 488 }
{ _id: 9, value: 464 }
```

Aggregation pipeline

- Consiste em um ou mais estágios
- O *output* de um estágio é o *input* do próximo estágio



Aggregation pipeline



Operadores

- $match
- $project
- $sort
- $skip
- $limit
- $count
- $group
- $unwind
- $sample
- $sortByCount

$match

```
db.collection('main')
    .aggregate([
        { $match: { _id: "a6b5c9" } }
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17> node .\native\pipe-match.js
YAY... There is a connection
{ _id: 'a6b5c9', a: 6, b: 5, c: 9, arr: [ 3, 4, 5, 7 ] }
```

```
db.collection('main')
    .aggregate([
        { $match: { arr: { $in: [5] } } }
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17> node .\native\pipe-match.js
YAY... There is a connection
{ _id: 'a6b5c9', a: 6, b: 5, c: 9, arr: [ 3, 4, 5, 7 ] }
{ _id: 'a0b0c3', a: 0, b: 0, c: 3, arr: [ 0, 5, 6, 8, 9 ] }
{ _id: 'a0b0c1', a: 0, b: 0, c: 1, arr: [ 0, 1, 2, 3, 5, 8, 9 ] }
{ _id: 'a0b0c2', a: 0, b: 0, c: 2, arr: [ 0, 2, 3, 4, 5, 8, 9 ] }
{ _id: 'a0b0c6', a: 0, b: 0, c: 6, arr: [ 0, 2, 4, 5, 6, 8, 9 ] }
{ _id: 'a0b0c7', a: 0, b: 0, c: 7, arr: [ 0, 1, 2, 4, 5, 6, 9 ] }
{ _id: 'a0b1c1', a: 0, b: 1, c: 1, arr: [ 3, 5, 7, 8 ] }
{ _id: 'a0b1c3', a: 0, b: 1, c: 3, arr: [ 1, 2, 4, 5, 6, 8, 9 ] }
{ _id: 'a0b1c4', a: 0, b: 1, c: 4, arr: [ 0, 2, 5, 6, 7 ] }
```

$project

```
db.collection('main')
    .aggregate([
        { $project: { _id: 0, a: 1, c: 1 } }
    ])
    .forEach(item => console.log(item));
```

```
{ a: 0, c: 8 }
{ a: 0, c: 9 }
{ a: 0, c: 0 }
{ a: 0, c: 1 }
{ a: 0, c: 2 }
{ a: 0, c: 3 }
{ a: 0, c: 4 }
{ a: 0, c: 5 }
{ a: 0, c: 6 }
{ a: 0, c: 7 }
{ a: 0, c: 8 }
{ a: 0, c: 9 }
{ a: 0, c: 0 }
{ a: 0, c: 1 }
{ a: 0, c: 2 }
{ a: 0, c: 3 }
{ a: 0, c: 4 }
```

```
db.collection('main')
    .aggregate([
        { $match: { a: { $in: [3, 8] }, b: 7 } },
        { $project: { _id: 0, a: 1, c: 1 } }
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripa
YAY... There is a connection
{ a: 3, c: 0 }
{ a: 3, c: 1 }
{ a: 3, c: 2 }
{ a: 3, c: 3 }
{ a: 3, c: 4 }
{ a: 3, c: 5 }
{ a: 3, c: 6 }
{ a: 3, c: 7 }
{ a: 3, c: 8 }
{ a: 3, c: 9 }
{ a: 8, c: 0 }
{ a: 8, c: 1 }
{ a: 8, c: 2 }
{ a: 8, c: 3 }
{ a: 8, c: 4 }
{ a: 8, c: 5 }
{ a: 8, c: 6 }
{ a: 8, c: 7 }
{ a: 8, c: 8 }
{ a: 8, c: 9 }
```

$sort

```
db.collection('main')
    .aggregate([
        { $match: { a: { $in: [3, 8] }, b: 7 } },
        { $project: { _id: 0, a: 1, c: 1 } },
        { $sort: { c: 1 } }
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17>
YAY... There is a connection
{ a: 3, c: 0 }
{ a: 8, c: 0 }
{ a: 3, c: 1 }
{ a: 8, c: 1 }
{ a: 3, c: 2 }
{ a: 8, c: 2 }
{ a: 3, c: 3 }
{ a: 8, c: 3 }
{ a: 3, c: 4 }
{ a: 8, c: 4 }
{ a: 3, c: 5 }
{ a: 8, c: 5 }
{ a: 3, c: 6 }
{ a: 8, c: 6 }
{ a: 3, c: 7 }
{ a: 8, c: 7 }
{ a: 3, c: 8 }
{ a: 8, c: 8 }
{ a: 3, c: 9 }
{ a: 8, c: 9 }
```

$skip

```
db.collection('main')
    .aggregate([
        { $match: { a: { $in: [3, 8] }, b: 7 } },
        { $project: { _id: 0, a: 1, c: 1 } },
        { $sort: { c: 1 } },
        { $skip: 10 },
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17>
YAY... There is a connection
{ a: 3, c: 5 }
{ a: 8, c: 5 }
{ a: 3, c: 6 }
{ a: 8, c: 6 }
{ a: 3, c: 7 }
{ a: 8, c: 7 }
{ a: 3, c: 8 }
{ a: 8, c: 8 }
{ a: 3, c: 9 }
{ a: 8, c: 9 }
```

$limit

```
db.collection('main')
    .aggregate([
        { $match: { a: { $in: [3, 8] }, b: 7 } },
        { $project: { _id: 0, a: 1, c: 1 } },
        { $sort: { c: 1 } },
        { $skip: 10 },
        { $limit: 4 }
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17> node .\native\pipe-limit.js
YAY... There is a connection
{ a: 3, c: 5 }
{ a: 8, c: 5 }
{ a: 8, c: 6 }
{ a: 3, c: 6 }
```

$sample

```
db.collection('main')
    .aggregate([
        { $sample: { size: 3 } }
    ])
    .forEach(item => console.log(item));
```

```
{ _id: 'a8b4c7', a: 8, b: 4, c: 7, arr: 8 }
PS D:\Felipe\Projects\floripajs-17> node .\native\pipe-sample.js
YAY... There is a connection
{ _id: 'a2b9c1', a: 2, b: 9, c: 1, arr: [ 0, 1, 3, 5, 8, 9 ] }
{ _id: 'a1b3c4', a: 1, b: 3, c: 4, arr: [ 0, 1 ] }
{ _id: 'a6b7c6', a: 6, b: 7, c: 6, arr: [ 0, 5, 8 ] }
PS D:\Felipe\Projects\floripajs-17> node .\native\pipe-sample.js
YAY... There is a connection
{ _id: 'a9b0c9', a: 9, b: 0, c: 9, arr: [ 4, 8 ] }
{ _id: 'a7b5c0', a: 7, b: 5, c: 0, arr: [ 2, 4, 8, 9 ] }
{ _id: 'a7b9c3', a: 7, b: 9, c: 3, arr: [ 0, 1, 3, 4 ] }
PS D:\Felipe\Projects\floripajs-17>
```

$count

```
db.collection('main')
    .aggregate([
        { $match: { arr: { $all: [0, 2, 4, 6, 8] } } },
        { $count: "even" }

    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17>
YAY... There is a connection
{ even: 23 }
```

$group

```
db.collection('main')
    .aggregate([
        { $match: { a: 7 } },
        { $project: { _id: 0, a: 1, b: 1, lengthOfArr: { $size: '$arr' } } },
        {
            $group: {
                _id: '$b',
                count: { $sum: 1 },
                averageLength: { $avg: '$lengthOfArr' }

            }
        },
        { $sort: { averageLength: 1 } }
    ])
    .forEach(item => console.log(item));
```

```
YAY... There is a connection
{ _id: 9, count: 10, averageLength: 3.9 }
{ _id: 8, count: 10, averageLength: 4.5 }
{ _id: 7, count: 10, averageLength: 4.5 }
{ _id: 2, count: 10, averageLength: 4.5 }
{ _id: 3, count: 10, averageLength: 4.9 }
{ _id: 1, count: 10, averageLength: 5 }
{ _id: 6, count: 10, averageLength: 5.1 }
{ _id: 4, count: 10, averageLength: 5.4 }
{ _id: 0, count: 10, averageLength: 5.7 }
{ _id: 5, count: 10, averageLength: 6 }
PS D:\Feline\Projects\florianis-17>
```

$unwind

```
db.collection('main')
    .aggregate([
        { $match: { _id: 'a8b4c7' } },
        { $unwind: '$arr' }
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17> node .\nat
YAY... There is a connection
{ _id: 'a8b4c7', a: 8, b: 4, c: 7, arr: 0 }
{ _id: 'a8b4c7', a: 8, b: 4, c: 7, arr: 1 }
{ _id: 'a8b4c7', a: 8, b: 4, c: 7, arr: 3 }
{ _id: 'a8b4c7', a: 8, b: 4, c: 7, arr: 5 }
{ _id: 'a8b4c7', a: 8, b: 4, c: 7, arr: 8 }
PS D:\Felipe\Projects\floripajs-17>
```

$sortByCount

```
db.collection('main')
    .aggregate([
        { $unwind: '$arr' },
        { $sortByCount: '$arr' }
    ])
    .forEach(item => console.log(item));
```

```
PS D:\Felipe\Projects\floripajs-17> node
YAY... There is a connection
{ _id: 5, count: 511 }
{ _id: 4, count: 504 }
{ _id: 6, count: 496 }
{ _id: 3, count: 496 }
{ _id: 7, count: 493 }
{ _id: 8, count: 491 }
{ _id: 9, count: 489 }
{ _id: 2, count: 482 }
{ _id: 1, count: 481 }
{ _id: 0, count: 469 }
PS D:\Felipe\Projects\floripajs-17>
```

Outros operadores

- $collStats
- $redact
- $geoNear
- $lookup
- $out
- $indexStats
- $facet
- $bucket
- $bucketAuto
- $addFields
- $replaceRoot
- $graphLookup

Agradecimentos

PERGUNTAS?