



Fundamentos de Big Data (part 2)

Centro Universitário 7
Setembro - Uni7

**Especialização em
Ciência de Dados com Big
Data, BI e Data Analytics**

Prof. Manoel Ribeiro

Conteúdo da disciplina (20 h/a)

- Dia 2 (8h/a)
 - Introdução ao **OpenStack**
 - Introdução à Arquitetura e ao Ecossistema **Hadoop**.
 - Introdução ao Apache **Spark**.
 - Introdução aos Bancos de Dados **NoSQL**
 - Introdução ao **MongoDB**

Roteiro





openstack™
CLOUD SOFTWARE

DevStack Install

Ubuntu 16.04/17.04, Fedora 24/25, CentOS/RHEL 7

```
$ sudo useradd -s /bin/bash -d /opt/stack -m stack
```

```
$ echo "stack ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/stack
```

```
$ sudo su - stack
```

```
$ git clone https://git.openstack.org/openstack-dev/devstack
```

```
$ cd devstack
```

```
$ nano local.conf
```

```
[[local|localrc]]
```

```
ADMIN_PASSWORD=secret
```

```
DATABASE_PASSWORD=$ADMIN_PASSWORD
```

```
RABBIT_PASSWORD=$ADMIN_PASSWORD
```

```
SERVICE_PASSWORD=$ADMIN_PASSWORD
```

```
$/stack.sh
```


Contribuições

LB-RLT Approach for Load Balancing Heterogeneous Storage Nodes

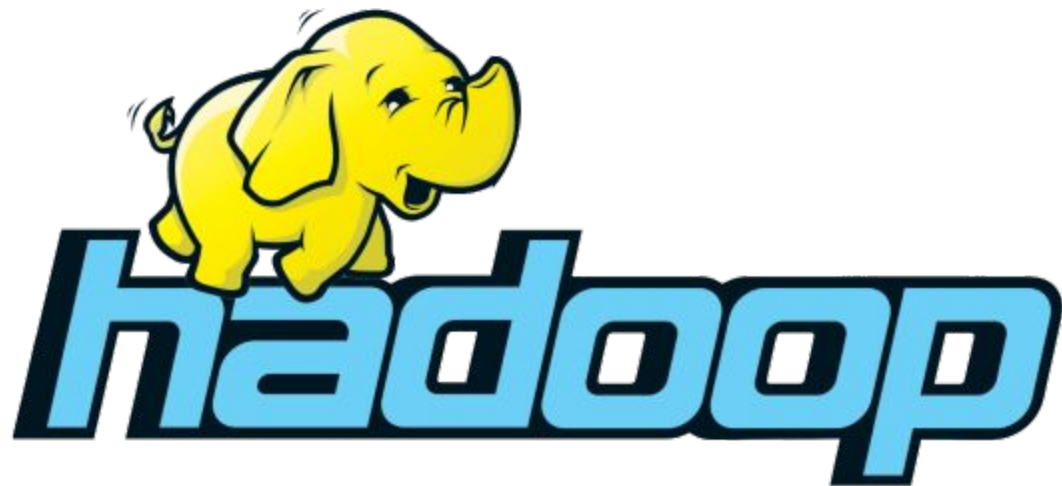
Antonio M. R. Almeida, Denis M. Cavalcante,
Flávio R. C. Sousa e Javam C. Machado ¹

¹ Mestrado e Doutorado em Ciencia da Computacao (MDCC)
Universidade Federal do Ceará (UFC) – Fortaleza, CE – Brasil

{manoel.ribeiro,denis.cavalcante}@lsbd.ufc.br {sousa,javam}@ufc.br

https://www.researchgate.net/publication/305771735_LB-RLT_Approach_for_Load_Balancing_Heterogeneous_Storage_Nodes

Abstract. *Cloud computing is a paradigm of service-oriented computing and has changed the way computing infrastructure is abstracted and used. The cloud is composed by heterogeneous resources and has a variable workload. Thus, load balancing techniques are crucial to distribute workload to processing nodes for enhancing the overall system performance. Conventional algorithms for load balancing have limitations in this environment, or they do not consider specific aspects of the resources simultaneously, e.g., response time and throughput. To address these limitations, this paper presents an approach to load balancing in the cloud. This approach considers cloud infrastructure storage throughput and a heterogeneous storage nodes environment. In order to evaluate this approach, we have conducted experiments that measure the response time, throughput and success rate and compared our results against conventional algorithms. Experimental results confirm that our approach ensures quality of service agreement, while using resources more efficiently.*



Hadoop - setup for single node

>PATH=%hadoop_home%\bin

edit %hadoop_home%/etc/hadoop/core-site.xml

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

edit %hadoop_home%/etc/hadoop/hdfs-site.xml:

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

<https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html>

Hadoop - Start

```
>cd %hadoop_home%\bin
```

```
>hadoop namenode -format
```

```
** check pasta /tmp/hadoop-<user>
```

```
>cd %hadoop_home%\sbin
```

```
>start_yarn.cmd
```

- Resource manager
- Node Manager

Hadoop - Start

>start_dfs.cmd

- **Hadoop namenode**
- **Hadoop datanode**

HADOOP LIVE!

<http://localhost:8088>

RM Home



All Applications

▼ Cluster

[About](#)[Nodes](#)[Node Labels](#)[Applications](#)[NEW](#)[NEW SAVING](#)[SUBMITTED](#)[ACCEPTED](#)[RUNNING](#)[FINISHED](#)[FAILED](#)[KILLED](#)[Scheduler](#)

► Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total
0	0	0	0	0	0 B	8 GB

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes
1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:1>

Show 20 ▼ entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCore
▼	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕

No data available in table

Showing 0 to 0 of 0 entries

Hadoop - Shutdown

```
>cd %hadoop_home%\sbin
```

```
>stop_yarn.cmd
```

```
>stop_dfs.cmd
```



Spark - Dataset para Demo

Portal da transparência - Brasil - Gastos

<http://www.portaldatransparencia.gov.br/downloads/mensal.asp?c=GastosDiretos#meses02>

Despesas - Gastos Diretos - Pagamentos - fev/2017 - 392 MB

Spark - Demo

```
>iconv -f ISO-8859-1 -t UTF-8 201702_GastosDiretos.csv >  
201702_GastosDiretos-utf8.csv
```

```
>pyspark
```

```
>>> df=spark.read.format("com.databricks.spark.csv").option("header",  
"true").option("delimiter", "\t").option("inferSchema",  
"True").load('/bigdata/dataset/201702_GastosDiretos-utf.csv' )
```

```
>>>df.count()
```

Spark - Demo

```
>>> df.printSchema()
root
|-- Código Órgão Superior: integer (nullable = true)
|-- Nome Órgão Superior: string (nullable = true)
|-- Código Órgão: integer (nullable = true)
|-- Nome Órgao: string (nullable = true)
|-- Código Unidade Gestora: integer (nullable = true)
|-- Nome Unidade Gestora: string (nullable = true)
|-- Código Grupo Despesa: integer (nullable = true)
|-- Nome Grupo Despesa: string (nullable = true)
|-- Código Elemento Despesa: integer (nullable = true)
|-- Nome Elemento Despesa: string (nullable = true)
|-- Código Função: integer (nullable = true)
|-- Nome Função: string (nullable = true)
|-- Código Subfunção: integer (nullable = true)
|-- Nome Subfunção: string (nullable = true)
|-- Código Programa: integer (nullable = true)
|-- Nome Programa: string (nullable = true)
|-- Código Ação: string (nullable = true)
|-- Nome Ação: string (nullable = true)
|-- Linguagem Cidadã: string (nullable = true)
|-- Código Favorecido: string (nullable = true)
|-- Nome Favorecido: string (nullable = true)
|-- Número Documento: string (nullable = true)
|-- Gestão Pagamento: string (nullable = true)
|-- Data Pagamento: string (nullable = true)
|-- Valor: double (nullable = true)
```

Spark - Demo

```
>>>df.select("Nome Órgão Superior").show(10, truncate=False)
```

[illegible]

Spark - Demo

```
>>>df.select("Nome Órgão Superior").distinct().show(10,truncate=False)
```

```
+-----+
|Nome Órgão Superior|
+-----+
|MINISTERIO DA PREVIDENCIA SOCIAL|
|MINISTERIO DA SAUDE|
|PRESIDENCIA DA REPUBLICA|
|MINISTERIO DA DEFESA|
|MINIST. DA AGRICUL..PECUARIA E ABASTECIMENTO|
|MINIST. DO PLANEJAMENTO. DESENVOLV. E GESTAO|
|MINISTERIO DAS RELACOES EXTERIORES|
|MINISTERIO DO MEIO AMBIENTE|
|MINISTERIO DO DESENVOLVIMENTO SOCIAL|
|MINIST.DA CIENCIA.TECNOL..INOV.E COMUNICACOES|
+-----+
only showing top 10 rows
```

Spark - Demo

```
>>>df.select("Nome Órgão Superior").distinct().show(10,truncate=False)
```

```
+-----+
|Nome Órgão Superior|
+-----+
|MINISTERIO DA PREVIDENCIA SOCIAL|
|MINISTERIO DA SAUDE|
|PRESIDENCIA DA REPUBLICA|
|MINISTERIO DA DEFESA|
|MINIST. DA AGRICUL..PECUARIA E ABASTECIMENTO|
|MINIST. DO PLANEJAMENTO. DESENVOLV. E GESTAO|
|MINISTERIO DAS RELACOES EXTERIORES|
|MINISTERIO DO MEIO AMBIENTE|
|MINISTERIO DO DESENVOLVIMENTO SOCIAL|
|MINIST.DA CIENCIA.TECNOL..INOV.E COMUNICACOES|
+-----+
only showing top 10 rows
```

Spark - Demo

```
>>>df.filter(df["Nome Órgão Superior"]=="MINISTERIO DO  
TRABALHO").count()
```

4410

Spark - Demo

```
>>> df.groupBy("Nome Órgão  
Superior").count().orderBy("count").show(5, False)
```

```
+-----+-----+  
|Nome Órgão Superior|count|  
+-----+-----+  
|MINIST. DA TRANSPARENCIA, FISCALIZACAO E CGU|218|  
|MINISTERIO DO ESPORTE|368|  
|MINISTERIO DAS RELACOES EXTERIORES|779|  
|MINISTERIO DO TURISMO|803|  
|MINIST. DA INDUSTRIA, COM.EXTERIOR E SERVICOS|1403|  
+-----+-----+
```

only showing top 5 rows

Spark - Demo

```
>>> df.groupBy("Nome Órgão Superior").sum("Valor").show(10, False)
```

```
+-----+-----+
|Nome Órgão Superior                |sum(Valor)                |
+-----+-----+
|MINISTERIO DA PREVIDENCIA SOCIAL   |1.9915093517999896E8|
|MINISTERIO DA SAUDE                 |1.8933997643299787E9|
|PRESIDENCIA DA REPUBLICA            |1.1735994061000007E8|
|MINISTERIO DA DEFESA                |6.417606468800002E8 |
|MINIST. DA AGRICUL..PECUARIA E ABASTECIMENTO |1.2012305850999987E8|
|MINIST. DO PLANEJAMENTO. DESENVOLV. E GESTAO |4.9010021770000115E7|
|MINISTERIO DAS RELACOES EXTERIORES  |1.0862974240000006E7|
|MINISTERIO DO MEIO AMBIENTE         |4.239037645999985E7 |
|MINISTERIO DO DESENVOLVIMENTO SOCIAL |1.9672350602999988E8|
|MINIST.DA CIENCIA.TECNOL..INOV.E COMUNICACOES|4.0943230055000174E8|
+-----+-----+
```

only showing top 10 rows

Spark - Demo

Responda: Qual o órgão Superior que mais gastou em fevereiro/2017?

Spark - Demo

```
>>> to_float = lambda x: float(x.replace(",","."))
```

```
>>> from pyspark.sql import functions as SQL
```

```
>>> from pyspark.sql.types import FloatType
```

```
>>> udf_to_float = SQL.udf(to_float, FloatType())
```

```
>>> df2=df.withColumn("ValorFloat",udf_to_float(df["Valor"]))
```

Spark - Demo

```
>>>df2.printSchema()
```

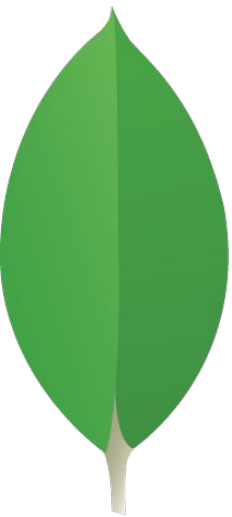
```
root
```

```
|-- Código Órgão Superior: string (nullable = true)  
|-- Nome Órgão Superior: string (nullable = true)  
|-- Código Órgão: string (nullable = true)  
|-- Nome Órgao: string (nullable = true)  
|-- Código Unidade Gestora: string (nullable = true)  
|-- Nome Unidade Gestora: string (nullable = true)  
|-- Código Grupo Despesa: string (nullable = true)  
|-- Nome Grupo Despesa: string (nullable = true)
```

```
***
```

```
|-- Nome Favorecido: string (nullable = true)  
|-- Número Documento: string (nullable = true)  
|-- Gestão Pagamento: string (nullable = true)  
|-- Data Pagamento: string (nullable = true)  
|-- Valor: string (nullable = true)  
|-- ValorFloat: float (nullable = true)
```

Not
Only SQL



mongoDB®

MongoDB - Demo

MongoDB Community Server

<https://www.mongodb.com/community>

<https://www.mongodb.com/download-center?jmp=nav#community>

Install custom to c:\bigdata\mongodb

MongoDB - Demo

```
>cd \bigdata\mongodb\
```

```
>mkdir data
```

```
>cd data
```

```
>mkdir db
```

```
>cd \bigdata\mongodb\
```

```
>mkdir log
```

```
>mongod -directoryperdb -dbpath=c:\bigdata\mongodb\data\db  
-logpath=c:\bigdata\mongodb\log\mongo.log -logappend -rest -install
```

MongoDB - Demo

>net start MongoDB

O serviço de MongoDB está sendo iniciado..

O serviço de MongoDB foi iniciado com êxito.

**** Verificar mongo.log**

***** [initandlisten] Service running**

*****[thread1] waiting for connections on port 27017**

MongoDB - Demo

```
>cd \bigdata\mongodb\
```

```
>mongo
```

```
connecting to: mongodb://127.0.0.1:27017
```

```
MongoDB server version: 3.4.9
```

```
Server has startup warnings:
```

```
2017-10-19T09:44:25.519-0300 I CONTROL [initandlisten]
```

```
2017-10-19T09:44:25.519-0300 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
```

```
2017-10-19T09:44:25.523-0300 I CONTROL [initandlisten] **      Read and write access to data and configuration is unrestricted.
```

```
2017-10-19T09:44:25.524-0300 I CONTROL [initandlisten]
```

```
>
```

MongoDB - Demo

>help

<code>db.help()</code>	help on db methods
<code>db.mycoll.help()</code>	help on collection methods
<code>sh.help()</code>	sharding helpers
<code>rs.help()</code>	replica set helpers
<code>help admin</code>	administrative help
<code>help connect</code>	connecting to a db help
<code>help keys</code>	key shortcuts
<code>help misc</code>	misc things to know
<code>help mr</code>	mapreduce
<code>show dbs</code>	show database names
<code>show collections</code>	show collections in current database
<code>show users</code>	show users in current database
<code>show profile</code>	show most recent system.profile entries with time >= 1ms
<code>show logs</code>	show the accessible logger names
<code>show log [name]</code>	prints out the last segment of log in memory, 'global' is default
<code>use <db_name></code>	set current database
<code>db.foo.find()</code>	list objects in collection foo
<code>db.foo.find({ a : 1 })</code>	list objects in foo where a == 1
<code>it</code>	result of the last line evaluated; use to further iterate
<code>DBQuery.shellBatchSize = x</code>	set default number of items to display on shell
<code>exit</code>	quit the mongo shell

MongoDB - Demo

> show dbs

admin 0.000GB

local 0.000GB

> use bigdata

switched to db bigdata

> db

bigdata

> db.ListaTelefonica.insert({Nome:"Paulo", Telefone:"(85) 929292"})

WriteResult({ "nInserted" : 1 })

> db.ListaTelefonica.insert({Nome:"Policia", Telefone:"190"})

WriteResult({ "nInserted" : 1 })

MongoDB - Demo

> show dbs

admin 0.000GB

bigdata 0.000GB

local 0.000GB

> show collections

ListaTelefonica

> db.ListaTelefonica.find()

```
{ "_id" : ObjectId("59e8b136dfa9a3fbf0fd8238"), "Nome" : "Paulo", "Telefone" :  
"(85) 929292" }
```

```
{ "_id" : ObjectId("59e8b146dfa9a3fbf0fd8239"), "Nome" : "Policia", "Telefone" :  
"190" }
```


MongoDB - Demo

```
> db.ListaTelefonica.find().pretty()
```

```
{  
  "_id" : ObjectId("59e8b136dfa9a3fbf0fd8238"),  
  "Nome" : "Paulo",  
  "Telefone" : "(85) 929292"  
}  
{  
  "_id" : ObjectId("59e8b146dfa9a3fbf0fd8239"),  
  "Nome" : "Policia",  
  "Telefone" : "190"  
}
```

MongoDB - Demo

```
> db.ListaTelefonica.update({Nome:"Paulo"},{Nome:"Paulo", Telefone:"(85) 923873727", tipo:"Celular"})
```

```
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
> db.ListaTelefonica.find().pretty()
```

```
{
  "_id" : ObjectId("59e8b3ccdfa9a3fbf0fd823a"),
  "Nome" : "Paulo",
  "Telefone" : "(85) 923873727",
  "tipo" : "Celular"
}
{
  "_id" : ObjectId("59e8b3d3dfa9a3fbf0fd823b"),
  "Nome" : "Policia",
  "Telefone" : "190"
}
```

MongoDB - Demo

```
> db.ListaTelefonica.update({Nome:"Paulo"},{$set:{ Telefone:"(85) 988888888"}})
```

```
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
> db.ListaTelefonica.find().pretty()
```

```
{
  "_id" : ObjectId("59e8b3ccdfa9a3fbf0fd823a"),
  "Nome" : "Paulo",
  "Telefone" : "(85) 988888888",
  "tipo" : "Celular"
}
{
  "_id" : ObjectId("59e8b3d3dfa9a3fbf0fd823b"),
  "Nome" : "Policia",
  "Telefone" : "190"
}
```

MongoDB - Demo

```
> db.ListaTelefonica.find({Nome:"Paulo"}).pretty()
```

```
{
  "_id" : ObjectId("59e8b3ccdfa9a3fbf0fd823a"),
  "Nome" : "Paulo",
  "Telefone" : "(85) 988888888",
  "tipo" : "Celular"
}
```

```
> db.ListaTelefonica.remove({Nome:"Paulo"})
```

```
WriteResult({ "nRemoved" : 1 })
```

```
> db.ListaTelefonica.find().pretty()
```

```
{
  "_id" : ObjectId("59e8b3d3dfa9a3fbf0fd823b"),
  "Nome" : "Policia",
  "Telefone" : "190"
}
```

MongoDB - Import DataSet

```
>mongoimport -d bigdata -c things --type tsv --file c:\bigdata\dataset\201702_gastosDiretos-utf.csv  
--headerline
```

```
2017-10-19T11:37:50.922-0300    connected to: localhost  
2017-10-19T11:37:53.876-0300    [#.....] bigdata.things      20.6MB/392MB (5.3%)  
2017-10-19T11:37:56.876-0300    [##.....] bigdata.things      40.4MB/392MB (10.3%)  
2017-10-19T11:37:59.876-0300    [###.....] bigdata.things      60.6MB/392MB (15.4%)  
2017-10-19T11:38:02.876-0300    [####.....] bigdata.things      81.0MB/392MB (20.7%)  
2017-10-19T11:38:05.876-0300    [#####.....] bigdata.things      94.1MB/392MB (24.0%)  
2017-10-19T11:38:08.877-0300    [#####.....] bigdata.things     115MB/392MB (29.2%)  
2017-10-19T11:38:11.877-0300    [#####.....] bigdata.things     136MB/392MB (34.6%)  
2017-10-19T11:38:14.879-0300    [#####.....] bigdata.things     157MB/392MB (40.0%)  
2017-10-19T11:38:17.877-0300    [#####.....] bigdata.things     178MB/392MB (45.4%)  
2017-10-19T11:38:20.876-0300    [#####.....] bigdata.things     198MB/392MB (50.4%)  
2017-10-19T11:38:23.876-0300    [#####.....] bigdata.things     218MB/392MB (55.6%)  
2017-10-19T11:38:26.877-0300    [#####.....] bigdata.things     238MB/392MB (60.8%)  
2017-10-19T11:38:29.876-0300    [#####.....] bigdata.things     257MB/392MB (65.4%)  
2017-10-19T11:38:32.876-0300    [#####.....] bigdata.things     273MB/392MB (69.5%)  
2017-10-19T11:38:35.876-0300    [#####.....] bigdata.things     293MB/392MB (74.8%)  
2017-10-19T11:38:38.878-0300    [#####.....] bigdata.things     313MB/392MB (79.8%)  
2017-10-19T11:38:41.882-0300    [#####.....] bigdata.things     334MB/392MB (85.1%)  
2017-10-19T11:38:44.877-0300    [#####.....] bigdata.things     354MB/392MB (90.3%)  
2017-10-19T11:38:47.876-0300    [#####.....] bigdata.things     375MB/392MB (95.5%)  
2017-10-19T11:38:50.450-0300    [#####.....] bigdata.things     392MB/392MB (100.0%)  
2017-10-19T11:38:50.451-0300    imported 927919 documents
```

MongoDB - Demo

> **db.gastos.count()**

927919

> **db.gastos.findOne()**

```
{
  "_id" : ObjectId("59e8bcfdcdd0fefa043a2ea1"),
  "Código Órgão Superior" : 20000,
  "Nome Órgão Superior" : "PRESIDENCIA DA REPUBLICA",
  "Código Órgão" : 20101,
  "Nome Órgao" : "PRESIDENCIA DA REPUBLICA",
  "Código Unidade Gestora" : 110001,
  "Nome Unidade Gestora" : "SECRETARIA DE ADMINISTRACAO/PR",
  "Código Grupo Despesa" : 3,
  "Nome Grupo Despesa" : "Outras Despesas Correntes",
  ****
  "Nome Ação" : "Assistência Pré-Escolar aos Dependentes dos Servidores Civis, Empregados e Militares",
  "Linguagem Cidadã" : "Assistência Pré-Escolar",
  "Código Favorecido" : 191,
  "Nome Favorecido" : "BANCO DO BRASIL SA [DIRECAO GERAL]",
  "Número Documento" : "2017OB801077",
  "Gestão Pagamento" : 1,
  "Data Pagamento" : "24/02/2017",
  "Valor" : "11156,55"
}
```

MongoDB - Demo

```
> db.gastos.aggregate([ {"$group" : {_id:"$Nome Órgao", count:{$sum:1}}} ])
```

```
{ "_id" : "COMPANHIA BRASILEIRA DE TRENS URBANOS - CBTU", "count" : 1640 }  
{ "_id" : "FUNDO NACIONAL DE ASSISTENCIA SOCIAL", "count" : 4 }  
{ "_id" : "INSTITUTO NACIONAL DO SEGURO SOCIAL", "count" : 15970 }  
{ "_id" : "INSTITUTO BRASILEIRO DE TURISMO", "count" : 433 }  
{ "_id" : "SUPERINT. DE DESENV. DO CENTRO-OESTE - SUDECO", "count" : 96 }  
{ "_id" : "SUPERINTEND.DO DESENVOLV.DA AMAZONIA-SUDAM", "count" : 129 }  
{ "_id" : "EPU - REC. SOB SUPERVISAO DO MIN. AERONAUTICA", "count" : 71 }  
{ "_id" : "FUNDO NAVAL", "count" : 5646 }  
{ "_id" : "AMAZONIA AZUL TECNOLOGIAS DE DEFESA S.A.", "count" : 182 }  
{ "_id" : "CAIXA DE CONST.DE CASAS DO PESSOAL DA MARINHA", "count" : 97 }  
{ "_id" : "FUNDACAO OSORIO", "count" : 50 }  
{ "_id" : "DEPARTAMENTO NAC. DE OBRAS CONTRA AS SECAS", "count" : 682 }  
{ "_id" : "INDUSTRIA DE MATERIAL BELICO DO BRASIL-IMBEL", "count" : 751 }  
{ "_id" : "COMANDO DA MARINHA", "count" : 6426 }  
{ "_id" : "COMANDO DA AERONAUTICA", "count" : 11437 }  
{ "_id" : "MINISTERIO DA DEFESA", "count" : 607 }  
{ "_id" : "AUTORIDADE PUBLICA OLIMPICA - APO/FEDERAL", "count" : 73 }  
{ "_id" : "INST.CHICO MENDES DE CONSER.DA BIODIVERSIDADE", "count" : 1826 }  
{ "_id" : "AGENCIA NACIONAL DE AGUAS ± ANA", "count" : 301 }  
{ "_id" : "AGENCIA NACIONAL DO CINEMA ± ANCINE", "count" : 330 }
```

MongoDB - Demo

```
> db.gastos.aggregate([ {"$group" : { "_id" : "$Nome Órgão Superior", "valor" : { "$sum" : "$Valor" } } } ])
```

```
{ "_id" : "MINISTERIO DAS CIDADES", "valor" : 127264010.74000001 }  
{ "_id" : "MINISTERIO DO TURISMO", "valor" : 22155796.56 }  
{ "_id" : "MINISTERIO DA CULTURA", "valor" : 41833812.02 }  
{ "_id" : "MINISTERIO DA INTEGRACAO NACIONAL", "valor" : 188860096.22 }  
{ "_id" : "MINISTERIO DO ESPORTE", "valor" : 27842774.69 }  
{ "_id" : "MINIST.DOS TRANSP.,PORTOS E AVIACAO CIVIL", "valor" : 686267918.95 }  
{ "_id" : "MINISTERIO DAS RELACOES EXTERIORES", "valor" : 10862974.24 }  
{ "_id" : "MINISTERIO DA PREVIDENCIA SOCIAL", "valor" : 199150935.17999998 }  
{ "_id" : "MINISTERIO DA JUSTICA E SEGURANA PUBLICA", "valor" : 200284115.09 }  
{ "_id" : "MINISTERIO DA FAZENDA", "valor" : 1721389684.59 }  
{ "_id" : "MINISTERIO DE MINAS E ENERGIA", "valor" : 172570439.39000002 }  
{ "_id" : "MINISTERIO DA DEFESA", "valor" : 641760646.88 }  
{ "_id" : "MINIST. DA AGRICUL.,PECUARIA E ABASTECIMENTO", "valor" : 120123058.51 }  
{ "_id" : "MINIST. DA TRANSPARENCIA, FISCALIZACAO E CGU", "valor" : 2576314.75 }  
{ "_id" : "MINISTERIO DA EDUCACAO", "valor" : 2165150592.07 }  
{ "_id" : "MINISTERIO DO TRABALHO", "valor" : 7025840035.13 }  
{ "_id" : "MINIST. DO PLANEJAMENTO, DESENVOLV. E GESTAO", "valor" : 49010021.77 }  
{ "_id" : "MINIST.DA CIENCIA,TECNOL.,INOV.E COMUNICACOES", "valor" : 409432300.55 }  
{ "_id" : "MINIST. DA INDUSTRIA, COM.EXTERIOR E SERVICOS", "valor" : 13828517.06 }  
{ "_id" : "MINISTERIO DO DESENVOLVIMENTO SOCIAL", "valor" : 196723506.03 }
```


MongoDB - Convertendo Valores para Float

```
db.gastos.find({"Valor": {$exists: true}}).forEach(function(doc){  
  if(doc.Valor.length > 0){  
    var newVal = doc.Valor.replace(',', '.');  
    var valString = parseFloat(newVal).toFixed(2);  
    doc.Valor = parseFloat(valString);  
    db.gastos.save(doc);  
  } // End of If Condition  
}) // End of foreach
```

Fim