UFC

DEPARTAMENTO DE COMPUTAÇÃO

MDCC

GREat

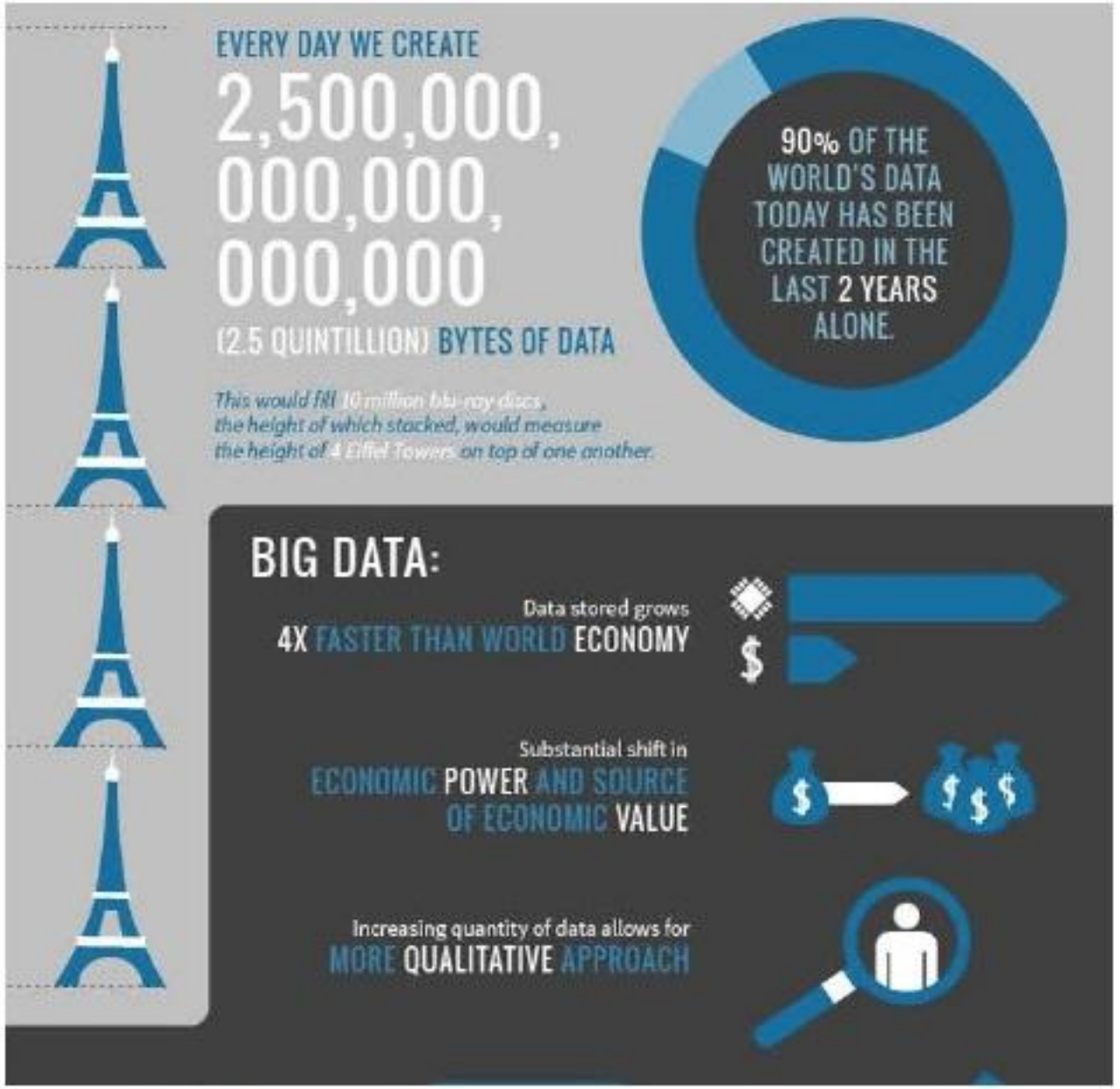
Map/Reduce

Professor:

Fernando Antonio Mota Trinta

Paulo Antonio Rego Leal

Contextualização



Fonte: vcloudnews

Contextualização e mais dados

- Mais de 4 bilhões de usuários na internet
  - 5 bi de usuários únicos de celular no mundo
  - Mais de 334 milhões de smartphones foram vendidos em 2018

Dos dados no mundo hoje foram **90%**

produzidos nos últimos **dois** *anos*

E mais dados...

- Facebook
  - 1B de usuários, 1,13 Trilhões de "likes", 219B de fotos e 140.3B de relacionamentos
- Youtube
  - 100 horas de vídeos adicionado a cada minuto
- Yahoo!
  - + de 650M de usuários, 11B visitas a páginas/mês
- Flick
  - + de 5B de fotos
- Twitter
  - 80 TB e 1B de tweets por dia
- Boeing
  - 640 TB gerados em um voo transatlântico
- Wal-Mart
  - 2,5 PB e 1 milhão de transações/hora
- LHC CERN
  - 15 Petabytes por ano

Comportamento

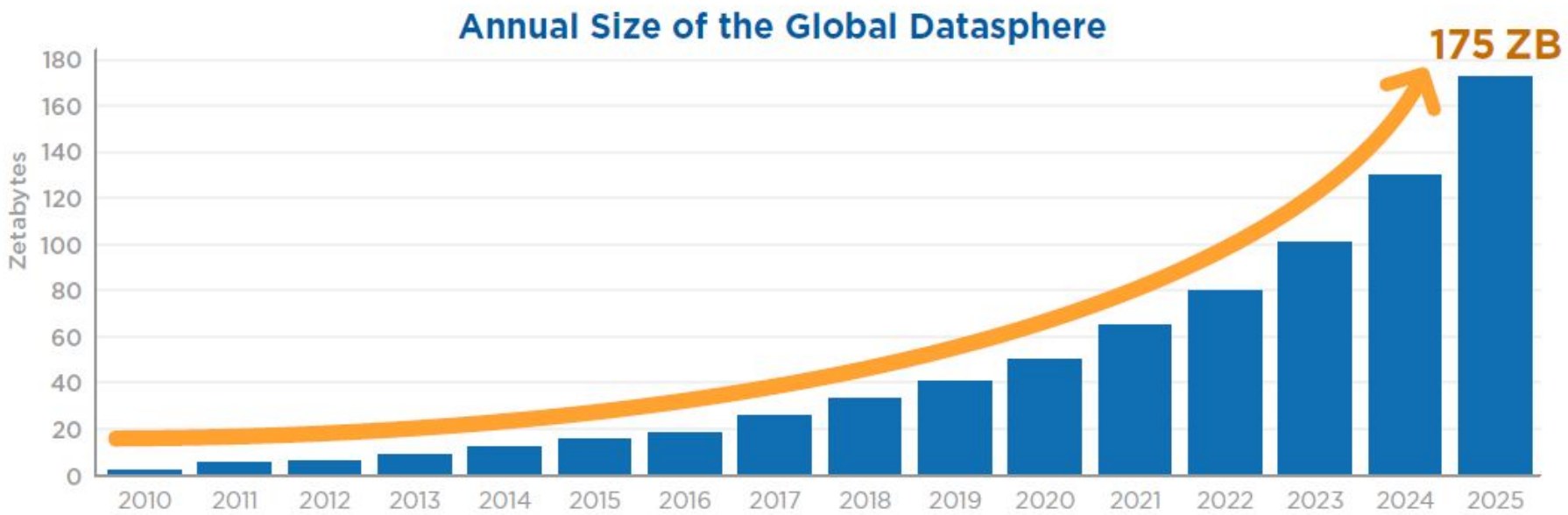
1990s

2010s

Dados x Informação

- "Extracting Value from Chaos" - a informação mundial está dobrando a cada 2 anos - 1.8 zettabytes foram criados em 2011, crescendo mais que a lei de Moore.

Figure 1 – Annual Size of the Global Datasphere

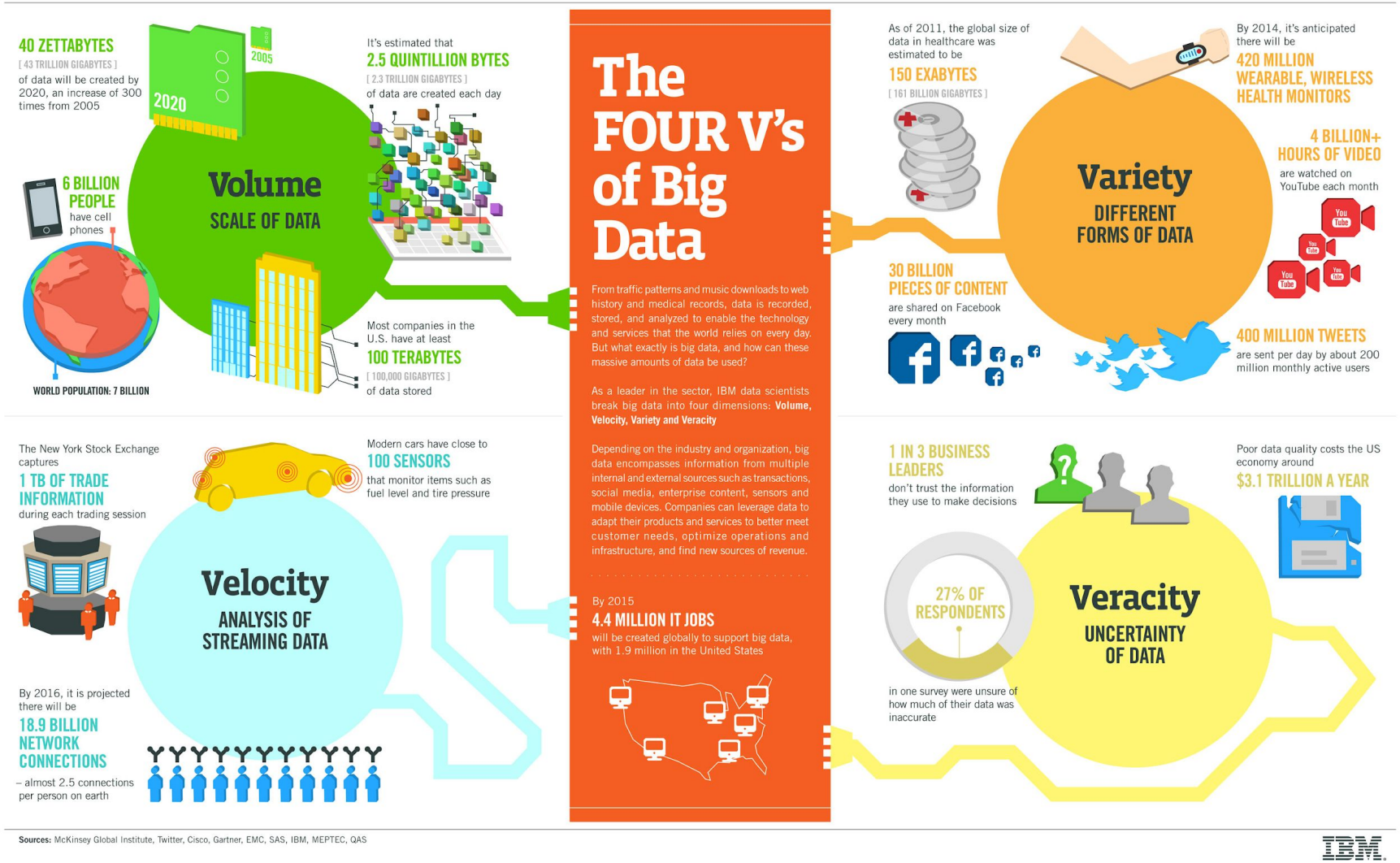


Contextualização

- Grande quantidades de dados geram desafios
  - Armazenamento...
  - Processamento...
  - Análise...

- BigData & Data Analytics
  - ciência de examinar os dados brutos com a finalidade de tirar conclusões sobre essa informação...
  - Dados analisados são valiosos hoje

Big Data (Definição dos 4V's)



Big Data

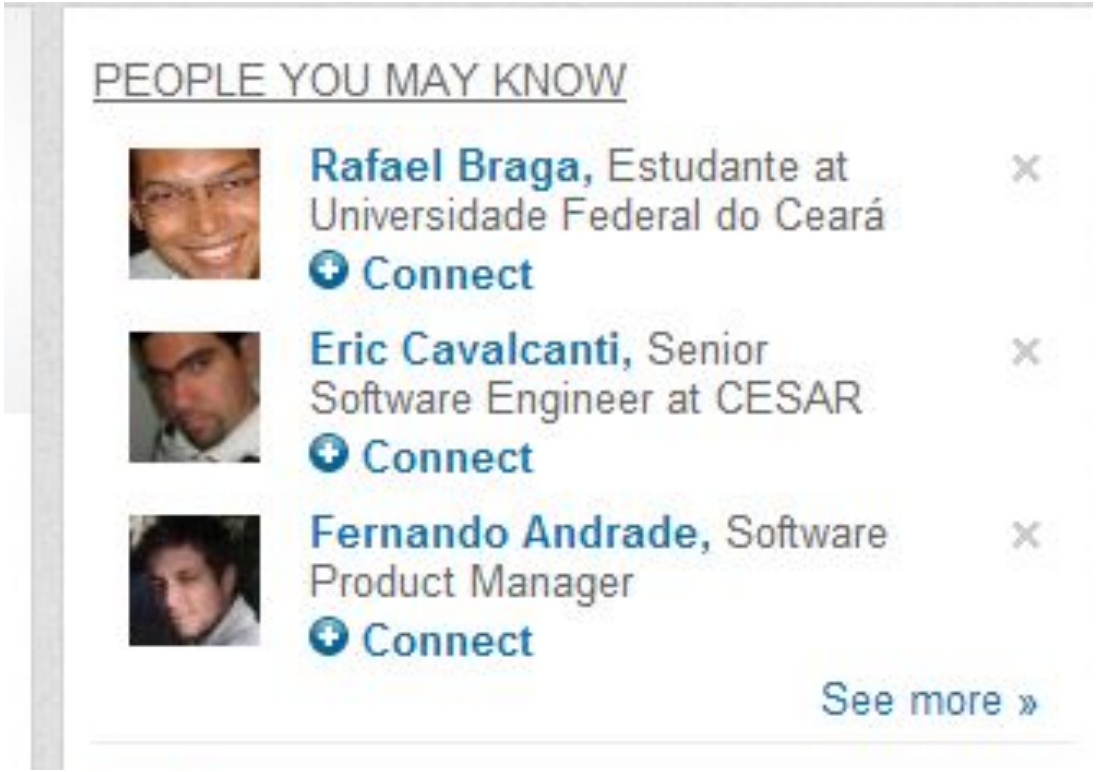
- Definição
  - Big data é alto volume, grande velocidade, e/ou grande variedade de informações que demandam formas inovadoras e economicamente viáveis de processar informações/dados, a fim de possibilitar melhoria de inferências, tomada de decisão e automação de processos. (Gartner 2012)

Big Data

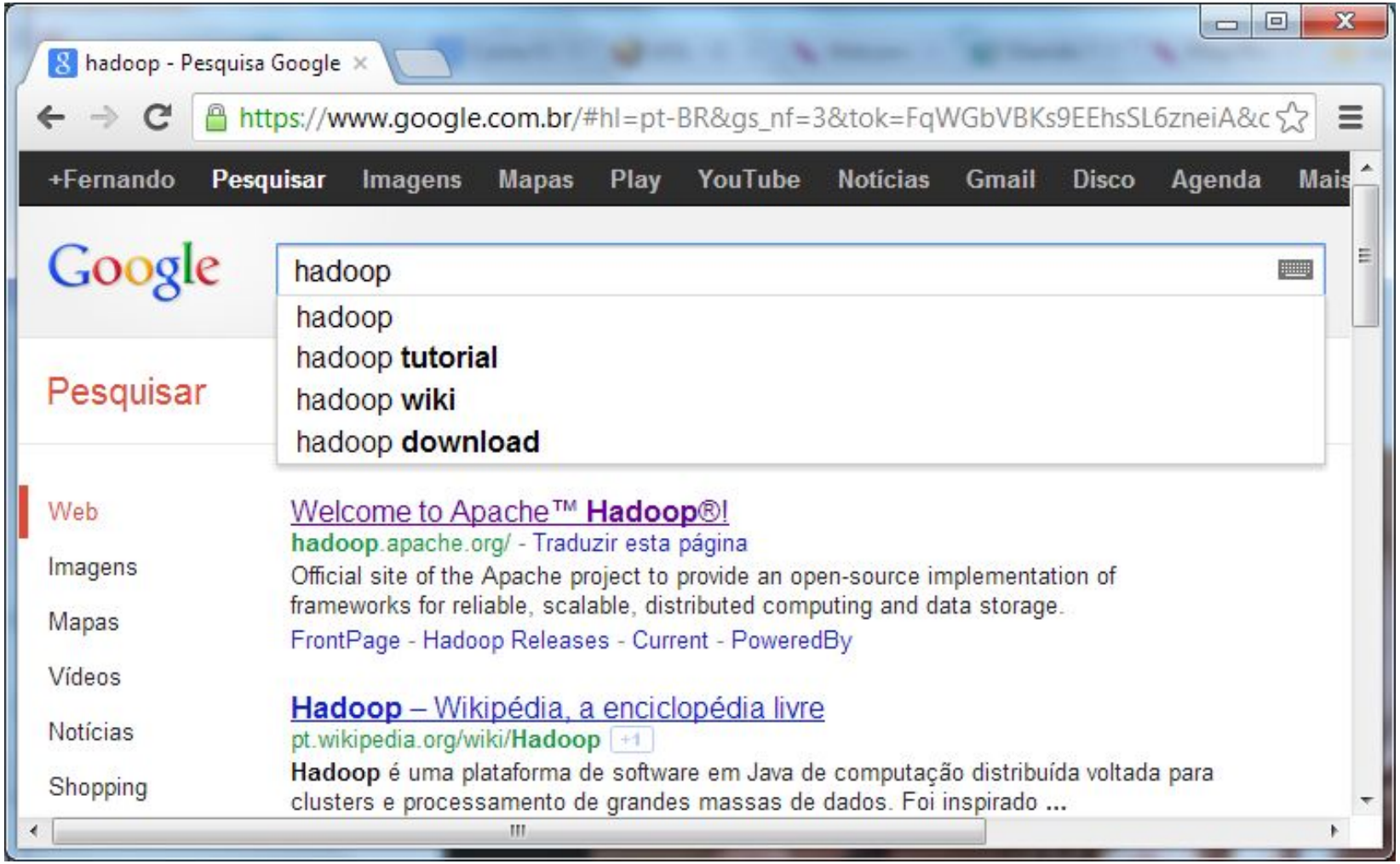
A habilidade de alcançar grande Valor através de ideias/conhecimentos obtidos com análise de dados

Exemplos

LinkedIn



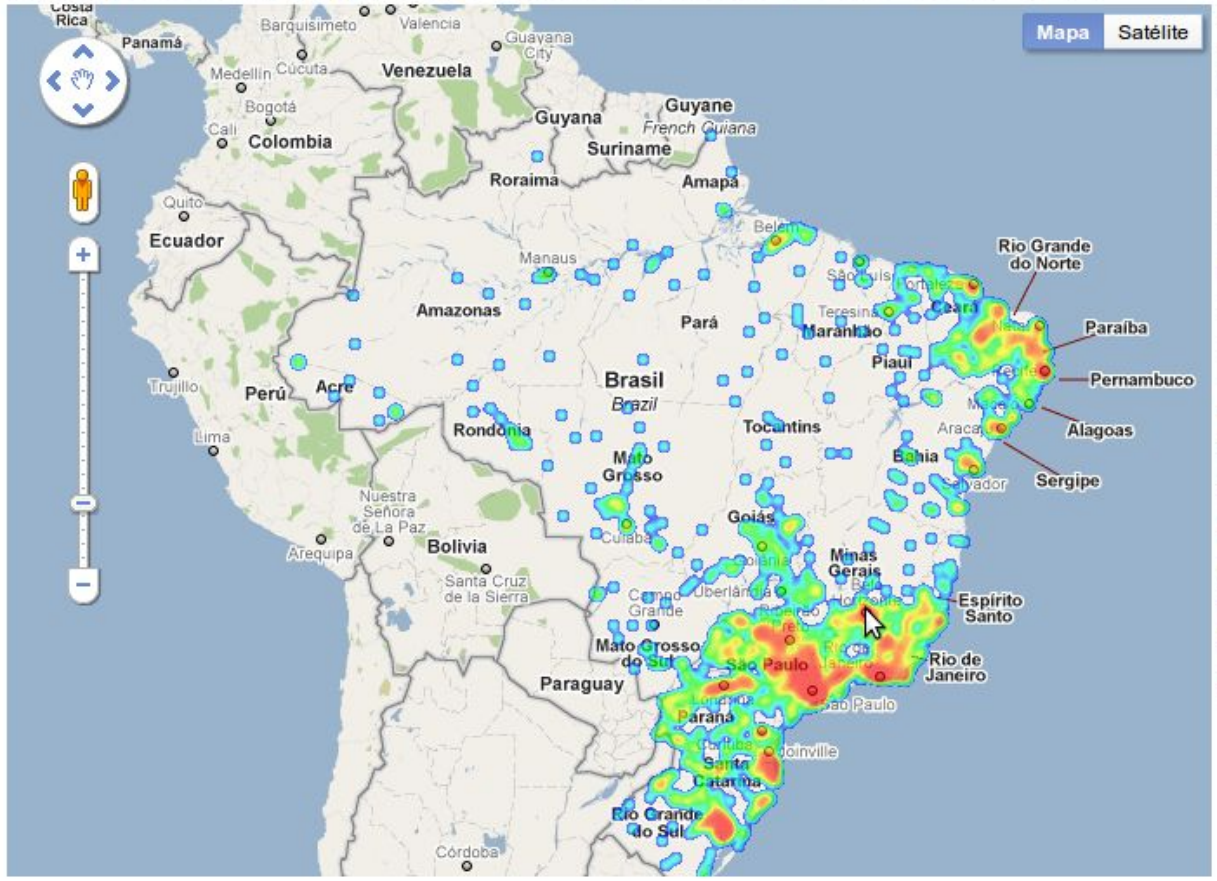
Exemplos


hadoop - Pesquisa Google
https://www.google.com.br/#hl=pt-BR&gs_nf=3&tok=FqWGbVBKs9EEhsSL6zneiA&c
+Fernando
Pesquisar
Imagens
Mapas
Play
YouTube
Notícias
Gmail
Disco
Agenda
Google
hadoop
hadoop
hadoop tutorial
hadoop wiki
hadoop download
Pesquisar
Web
Imagens
Mapas
Vídeos
Notícias
Shopping
Welcome to Apache™ Hadoop®!
hadoop.apache.org/ - Traduzir esta página
Official site of the Apache project to provide an open-source implementation of frameworks for reliable, scalable, distributed computing and data storage.
FrontPage - Hadoop Releases - Current - PoweredBy
Hadoop – Wikipédia, a enciclopédia livre
pt.wikipedia.org/wiki/Hadoop
Hadoop é uma plataforma de software em Java de computação distribuída voltada para clusters e processamento de grandes massas de dados. Foi inspirado ...

Dengue Watch: Heat Map

observatóriodadengue

Menções à dengue no Twitter no mês de fev/2011



Clique nos pontos do mapa para informações
Cidades: 11 Tweets: 59 População: 1925450 Tx. Inc. Méd.: 1.5334e-04

Cidade	Pop.	Tweets	Tx.Inc
Betim	377547	14	1.8230e-04
Brumadinho	34013	1	1.4289e-04
Contagem	603048	22	1.7922e-04
Ibirite	159026	4	1.2110e-04
Itabira	109551	6	2.7305e-04
Itauna	85396	1	5.2124e-05
Joao Monlevade	73451	4	2.7146e-04
Matozinhos	32973	1	1.4765e-04
Ribeirao das Neves	296376	2	2.6665e-05
Sabara	126219	3	1.1399e-04
Santa Barbara	27850	1	1.7627e-04

Campos de Aplicação

- Sistemas de Recomendação
- Processamento de Linguagem Natural
- Data Warehousing
- Pesquisa de Mercado
- Análise Financeira
- Máquinas de inferência
- Processamento de Vídeo/Imagens
- Análise de Logs

Campos de Aplicação

- Ciências da Saúde
- Gestão governamental
- Redes Sociais
- Telecomunicações

Como armazenar e processar este grande volume de dados?

Computação em Nuvem

"Grandes poderes trazem grandes responsabilidades!"

- O grande volume de dados demanda
- Grande poder de processamento
- Paralelizar e Distribuir tarefas
- Facilidade de processamento

Computação em Nuvem

- Para tratar problemas de larga escala, idealmente não gostaríamos de se preocupar com:
  - Paralelização e distribuição automática
  - Tolerância a falhas
  - Escalonamento de I/O
  - Monitoramento de tarefas
- Então é necessário se adequar a um modo de facilitar o uso da nuvem para tarefas de manipulação de dados

Modelo de Programação para Nuvem

- Forma, abordagem ou maneira específica de como se programar, dentro do contexto de uma aplicação ou domínio
  - Abstrações adequadas
  - Eficiência
- Duas abordagens:
  - Estender um modelo existente
  - Propor um novo modelo

Map/Reduce

- Um modelo de programação e uma implementação associada para processamento e geração de grandes conjuntos de dados
  - Inspirado pelas primitivas map e reduce encontrados na Lisp e em outras linguagens funcionais
- Permite paralelizar grandes computações facilmente e usar re-execução como mecanismo para tolerância a falhas

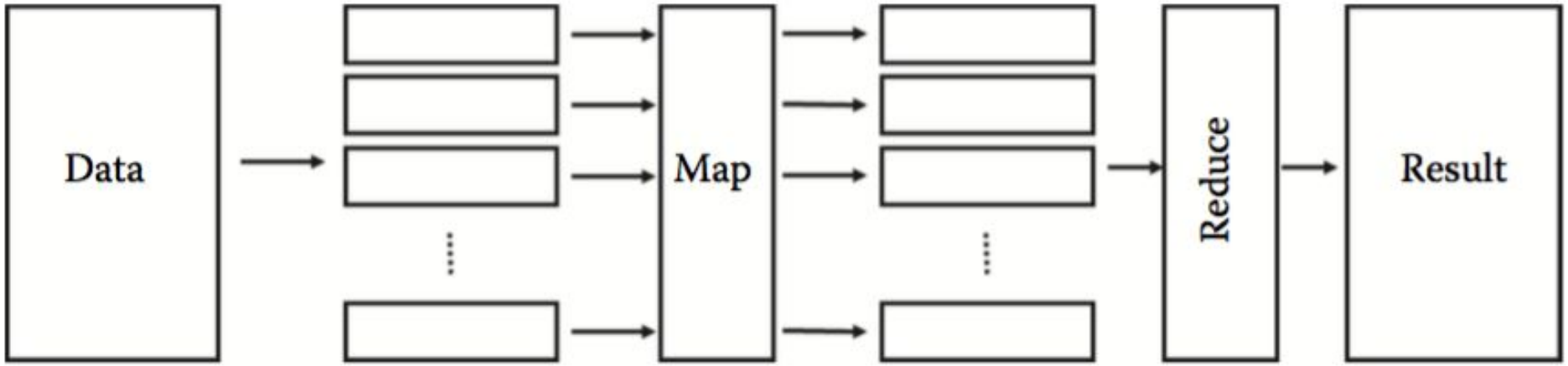
Map/Reduce - História

- Originalmente desenvolvido pela Google
  - Jeffrey Dean e Sanjay Ghemawat
  - MapReduce: Simplified Data Processing on Large Clusters. OSDI'04: Sixth Symposium on Operating System Design and Implementation (December 2004)
- Usado no Search Engine para tratar a quantidade de dados a serem processados

Map/Reduce - Ideia Geral

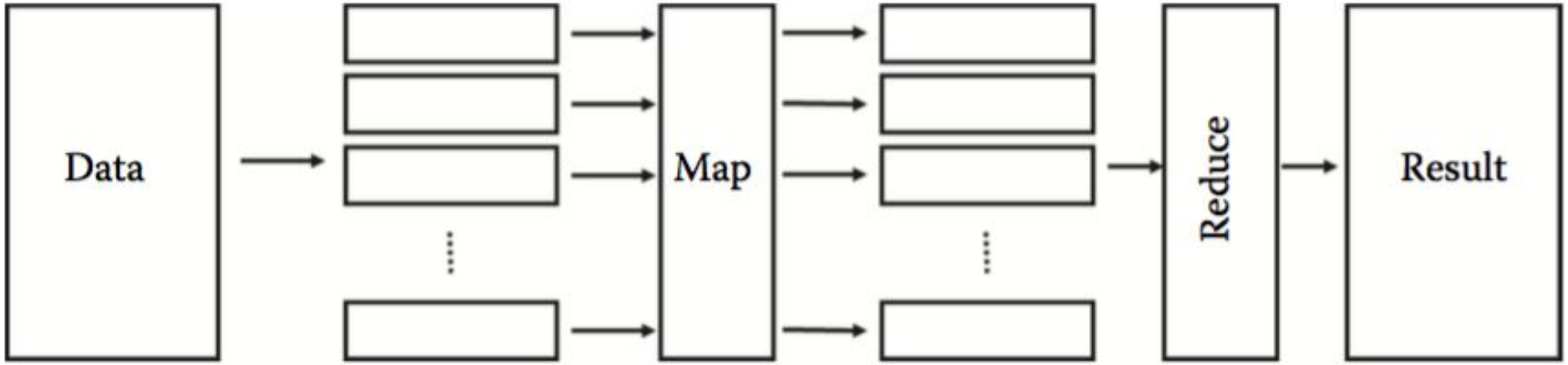
- Divida uma tarefa que deve processar um grande conjunto de dados em partes
- Cada parte deve ser responsável por uma parte pequena do conjunto
- Cada parte é independente
- Após o término do processamento das partes individuais, junte os resultados das partes

Map/Reduce



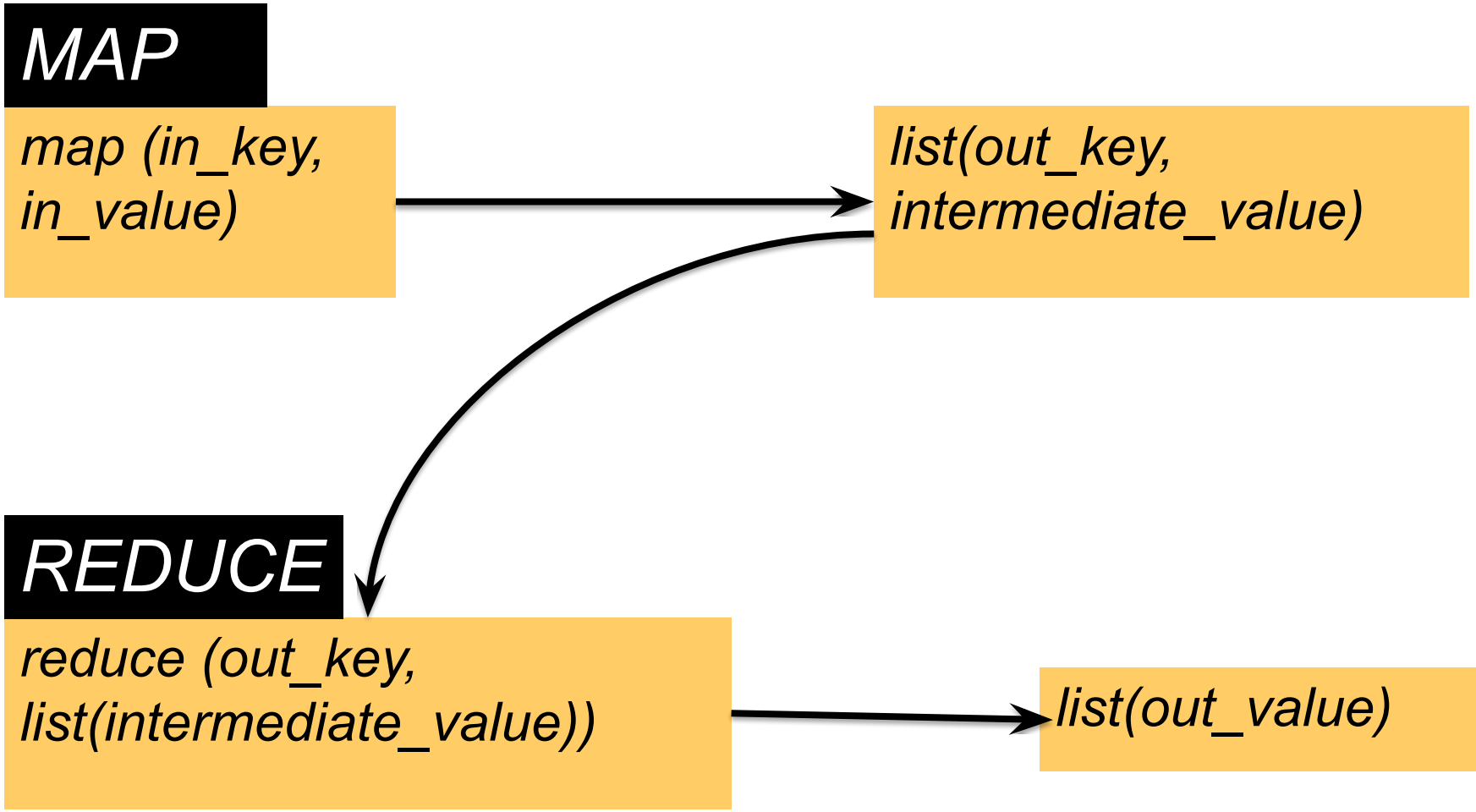
- MAP:
  - Toma-se uma tarefa complexa ou custosa
  - Quebra tal tarefa em sub-problemas menores
  - Delega a resolução desta tarefas a nós distribuídos

Map/Reduce



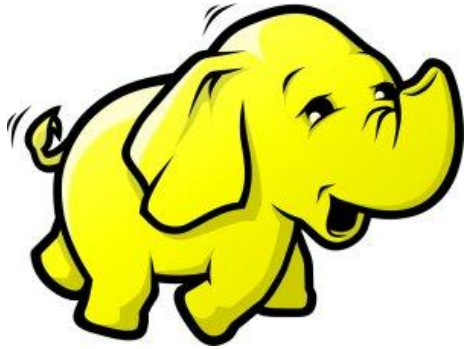
- REDUCE:
  - Coleta as respostas dos nós distribuídos
  - Agrega tais respostas em uma saída que representa a solução do problema complexo

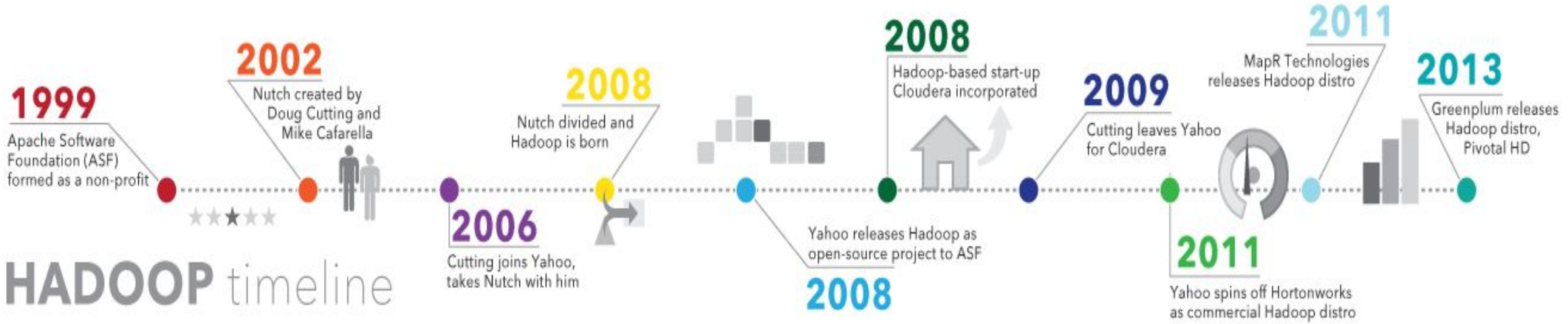
Map/Reduce


MAP
map (in_key, in_value)
list(out_key, intermediate_value)
REDUCE
reduce (out_key, list(intermediate_value))
list(out_value)

Apache Hadoop

- Inspirado em iniciativas da Google
  - BigTable e Map/Reduce
- Criado por Doug Cutting em 2002
  - Criador do Lucene



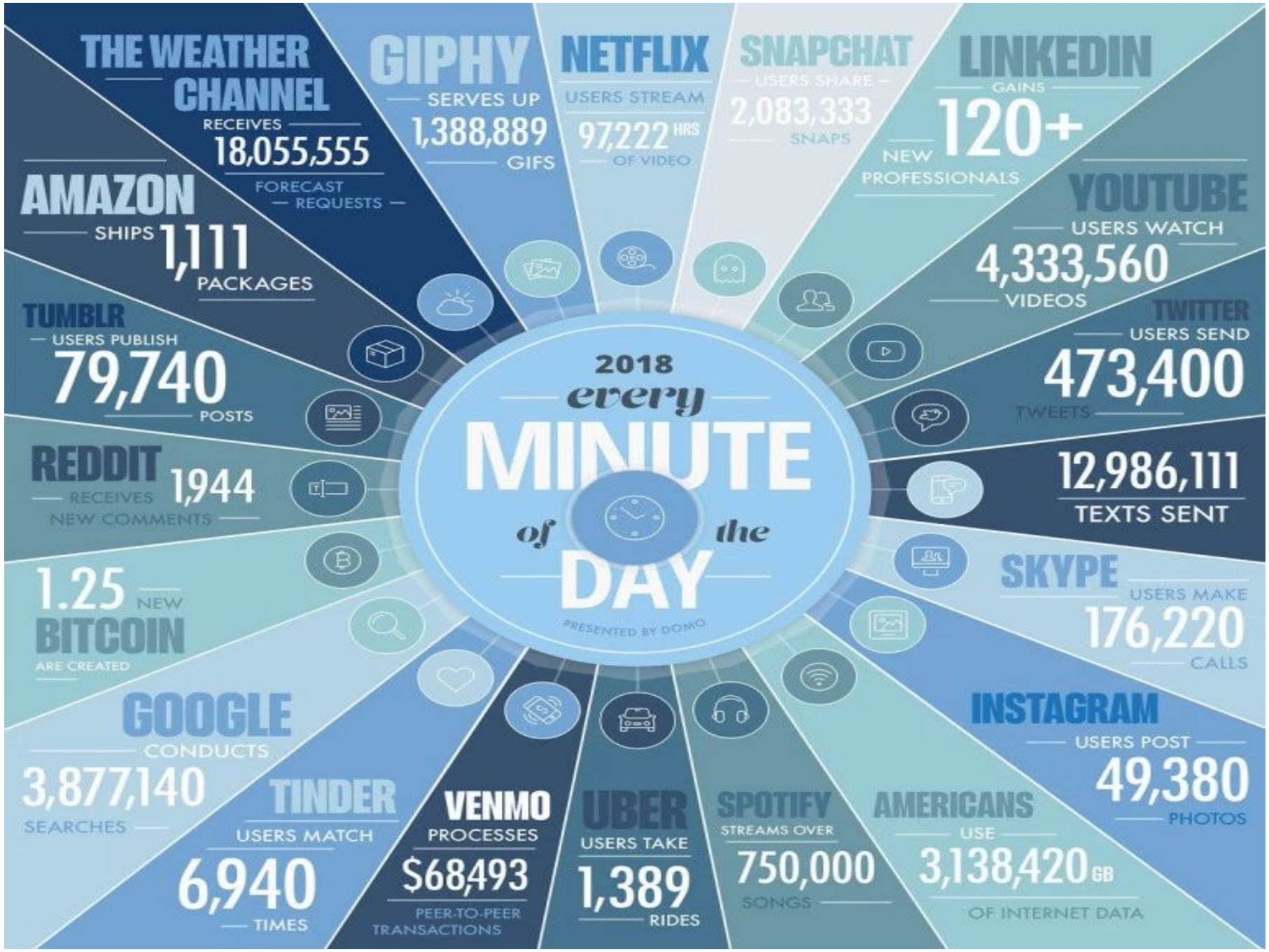


Hadoop

Um framework open-source de propósito geral, orientado a processamento em lote/offline, de uso intensivo de dados (I/O intensive) utilizado para criação de aplicações que processam uma grande quantidade de dados.

O que seria Grande?

- 25K máquinas
- Dezenas de clusters
- 3 Pb de dados
- 10000 jobs/semana


2018
every
MINUTE
of the
DAY
PRESENTED BY DOMO
THE WEATHER CHANNEL
RECEIVES
18,055,555
FORECAST REQUESTS
GIPHY
SERVES UP
1,388,889
GIFS
NETFLIX
USERS STREAM
97,222 HRS
OF VIDEO
SNAPCHAT
USERS SHARE
2,083,333
SNAPS
LINKEDIN
GAINS
120+
NEW PROFESSIONALS
AMAZON
SHIPS 1,111
PACKAGES
YOUTUBE
USERS WATCH
4,333,560
VIDEOS
TUMBLR
USERS PUBLISH
79,740
POSTS
TWITTER
USERS SEND
473,400
TWEETS
REDDIT
RECEIVES 1,944
NEW COMMENTS
12,986,111
TEXTS SENT
1.25 NEW
BITCOIN
ARE CREATED
SKYPE
USERS MAKE
176,220
CALLS
GOOGLE
CONDUCTS
3,877,140
SEARCHES
INSTAGRAM
USERS POST
49,380
PHOTOS
TINDER
USERS MATCH
6,940
TIMES
VENMO
PROCESSES
$68,493
PEER-TO-PEER TRANSACTIONS
UBER
USERS TAKE
1,389
RIDES
SPOTIFY
STREAMS OVER
750,000
SONGS
AMERICANS
USE
3,138,420 GB
OF INTERNET DATA

Quem usa mais

http://wiki.apache.org/hadoop/PoweredBy

O Hadoop não é...

... Um banco de dados relacional

... Um sistema online de processamento de transações

... Um sistema de armazenamento estruturado de qualquer tipo

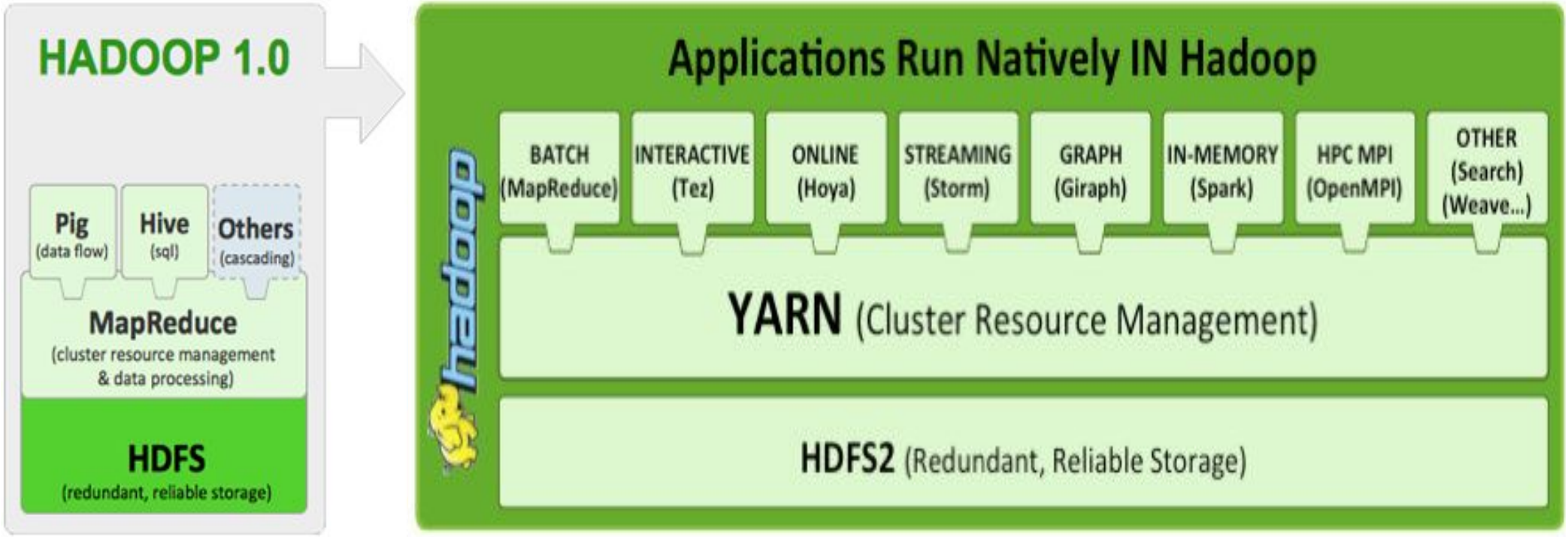
Hadoop x Relacional

Hadoop	*Relacional*
Pares de chave/valor	*Tabelas*
Informa-se como processar dados	*Informa-se como se deseja obter os dados (SQL)*
Offline/Lote	*Online/Tempo Real*
Escalabilidade Horizontal	*Escalabilidade Vertical*

Hadoop - Módulos principais

- Hadoop Common
  - Utilitários que suportam os outros módulos.
- Sistema distribuído de arquivos
  - Hadoop Distributed File System (HDFS)
    - Provê acesso de alta velocidade aos dados
- Hadoop YARN (Yet Another Resource Negotiator)
  - Framework para escalonamento de tarefas e gerenciamento de recursos do cluster
- Modelo de programação MapReduce
  - Framework baseado em YARN para facilitar a escrita de aplicações que processam de maneira paralela e distribuída grande quantidade de dados

Hadoop


HADOOP 1.0
Pig
(data flow)
Hive
(sql)
Others
(cascading)
MapReduce
(cluster resource management
& data processing)
HDFS
(redundant, reliable storage)
Applications Run Natively IN Hadoop
hadoop
BATCH
(MapReduce)
INTERACTIVE
(Tez)
ONLINE
(Hoya)
STREAMING
(Storm)
GRAPH
(Giraph)
IN-MEMORY
(Spark)
HPC MPI
(OpenMPI)
OTHER
(Search)
(Weave...)
YARN (Cluster Resource Management)
HDFS2 (Redundant, Reliable Storage)

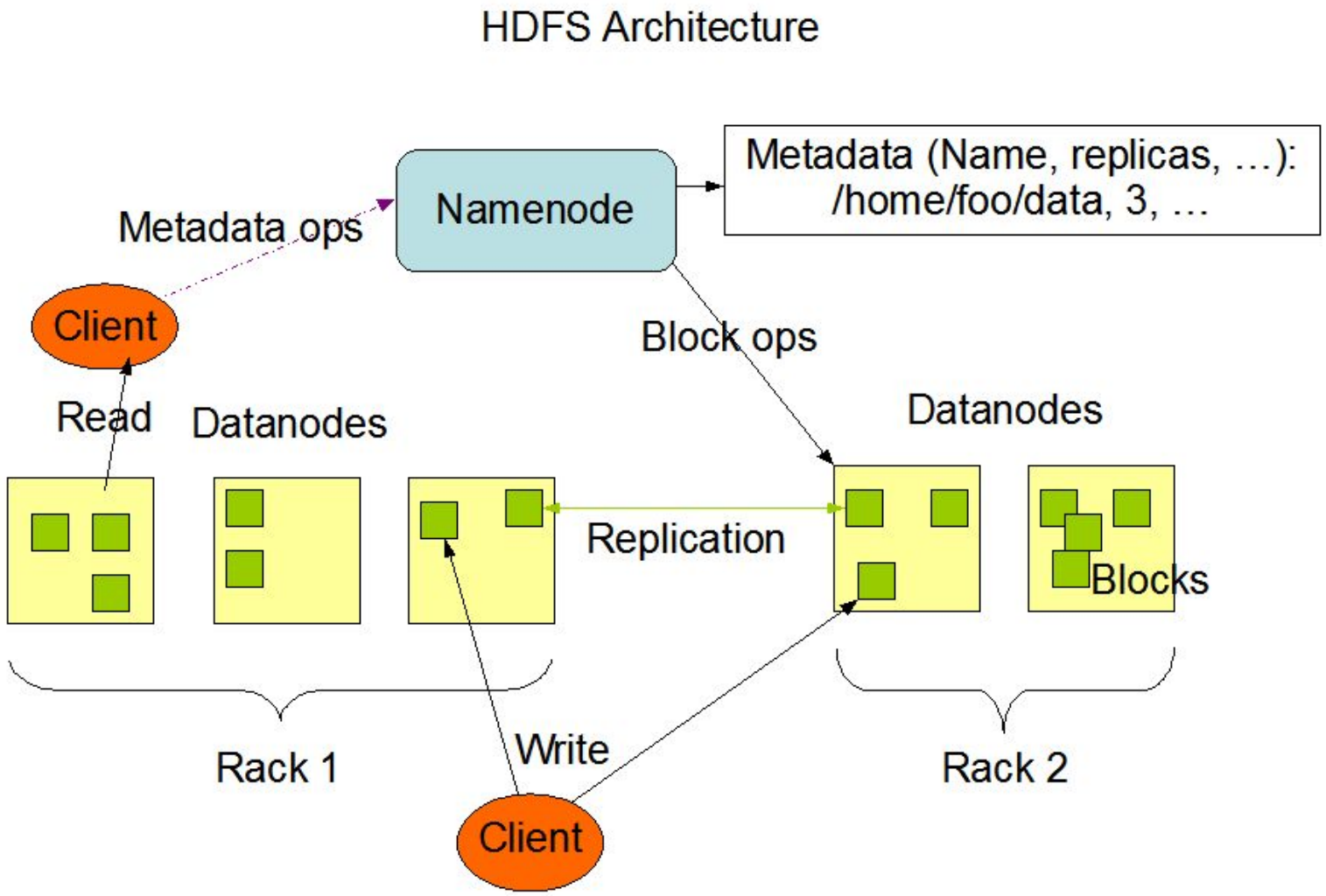
Hadoop - Ideia geral

- Um job MapReduce divide os dados de entrada em blocos independentes que são processados pelos Maps de maneira totalmente paralelizada.
- O framework ordena a saída dos Maps, que servem como entrada para os Reduces.
- Geralmente tanto os dados de entrada quanto de saída do job são armazenados no sistema de arquivo.
- O framework cuida do escalonamento e monitoramento das tarefas, reexecutando-as em casos de falhas.

HDFS

- Dados são replicados e distribuídos em vários nós
  - Fator de replicação: 3
- Projetado para arquivos grandes
  - Terabytes
- Orientado a blocos
- Comandos a lá Linux
  - ls, cp, mv, rm, etc

HDFS


HDFS Architecture
Namenode
Metadata (Name, replicas, …):
/home/foo/data, 3, …
Metadata ops
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client

NameNode

- Gerente do HDFS
  - Um cluster HDFS consiste em um único NameNode, que gerencia o namespace do sistema de arquivos e regula o acesso a arquivos pelos clientes.
  - Determina o mapeamento de blocos para os DataNodes.
- O NameNode é o árbitro e o repositório de todos os metadados do HDFS.
  - O sistema é projetado de tal forma que os dados do usuário nunca passam pelo NameNode.
  - Recomenda-se não executar jobs nesse servidor.

DataNode

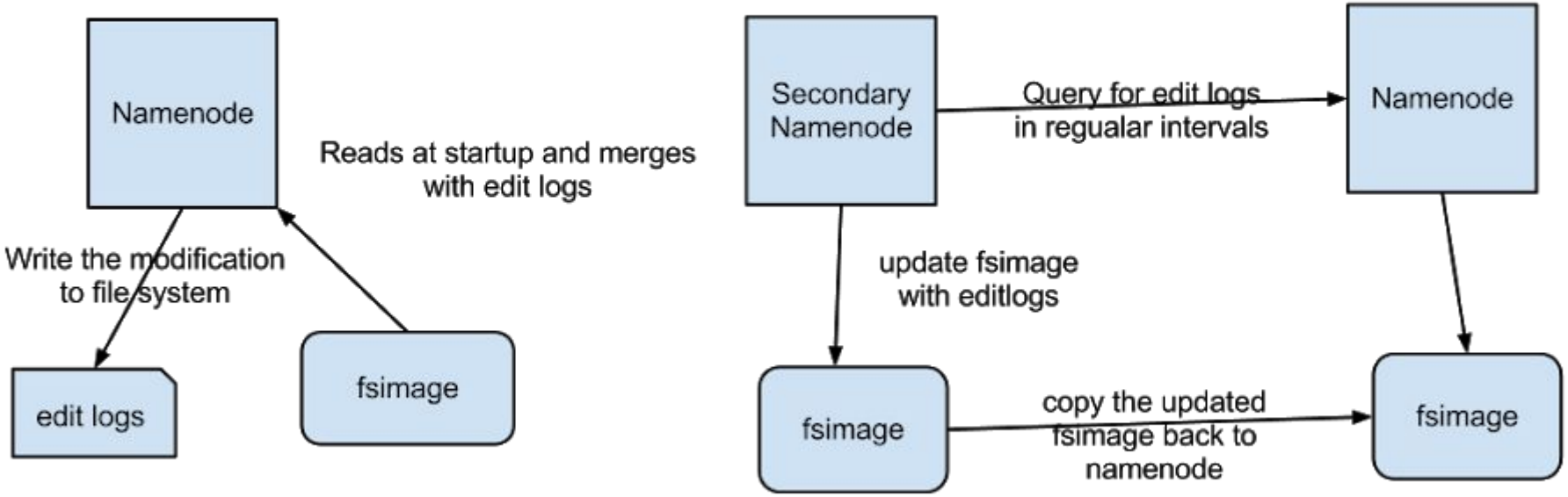
- São responsáveis por atender solicitações de leitura e gravação dos clientes do sistema de arquivos.
- Armazena os blocos de arquivos
  - Executam a criação, exclusão e replicação de blocos sob instruções do NameNode.
  - Não armazena arquivos contíguos
- Informa dados do bloco ao NameNode

Secondary NameNode

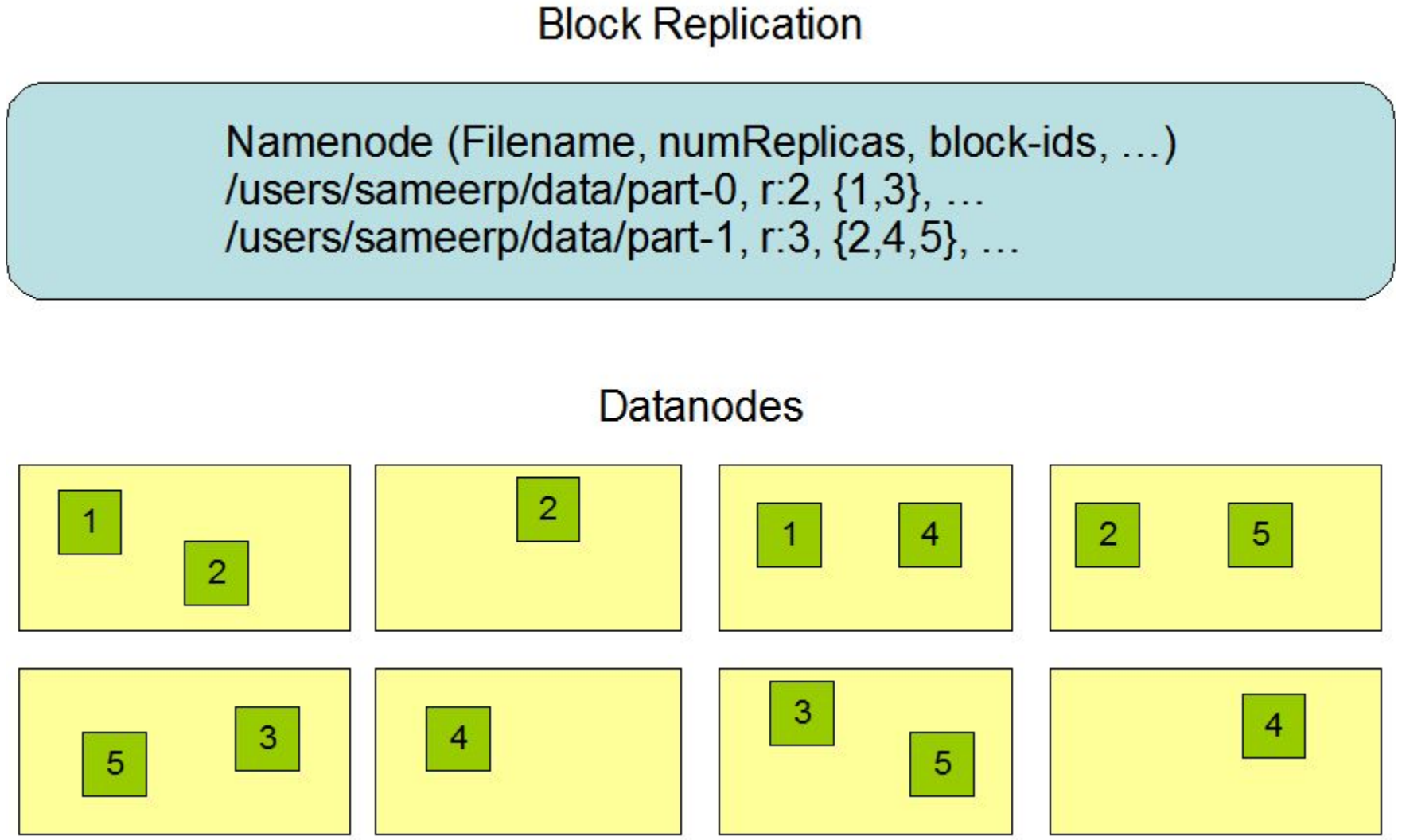
- Sua principal função é obter pontos de verificação dos metadados do sistema de arquivos presentes no Namenode.
- É um auxiliar do Namenode, não é um Namenode de backup.
- Em caso de falha, o Namenode pode reiniciar mais rapidamente por causa dos pontos de verificação.
  - Além de minimizar a perda de dados em caso de falha do NameNode.

Secondary NameNode

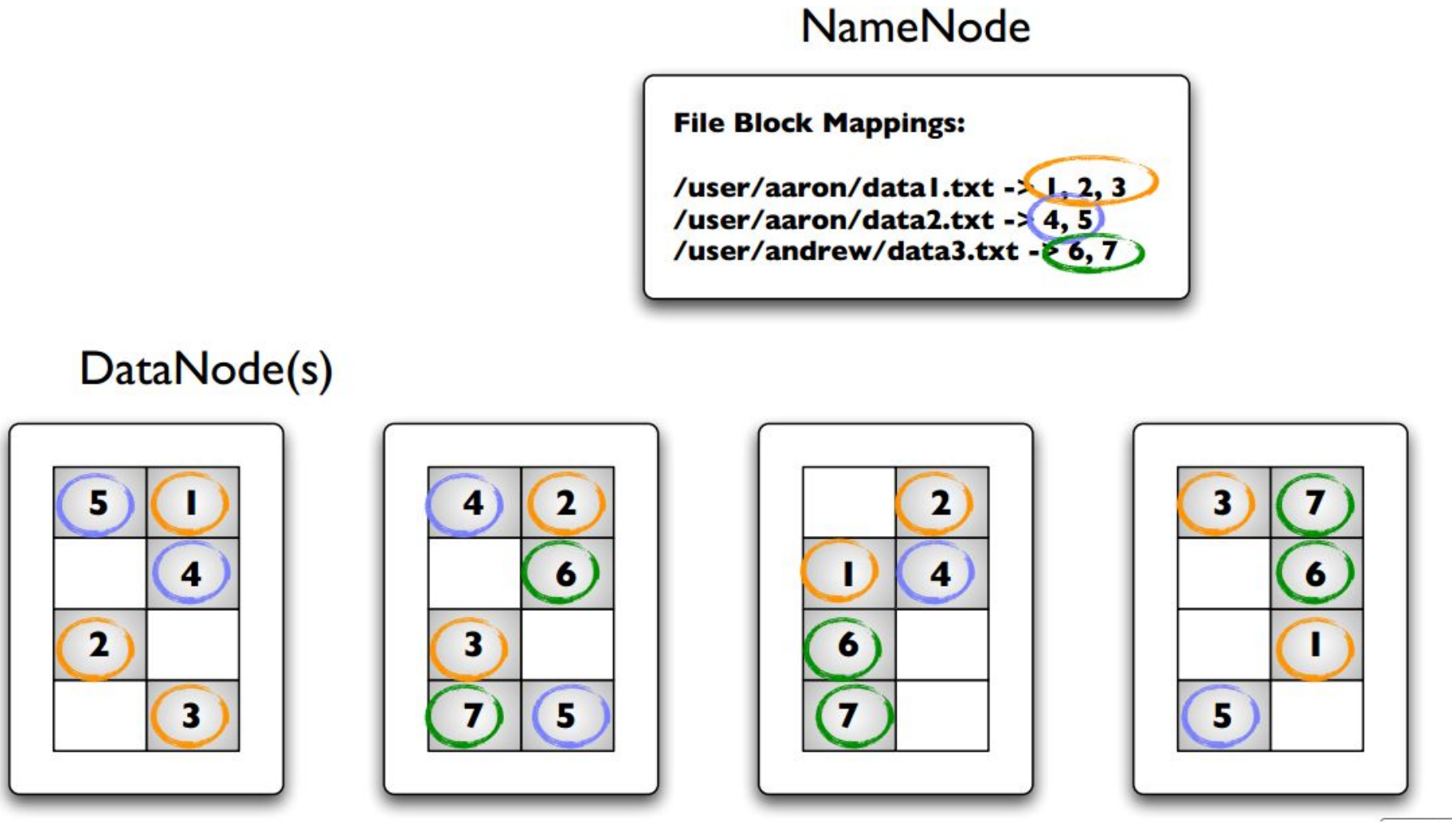
- fsimage: é uma snapshot do sistema de arquivos quando o Namenode started
- Edit logs: É a sequência de alterações feitas no sistema de arquivos após o Namenode ser iniciado



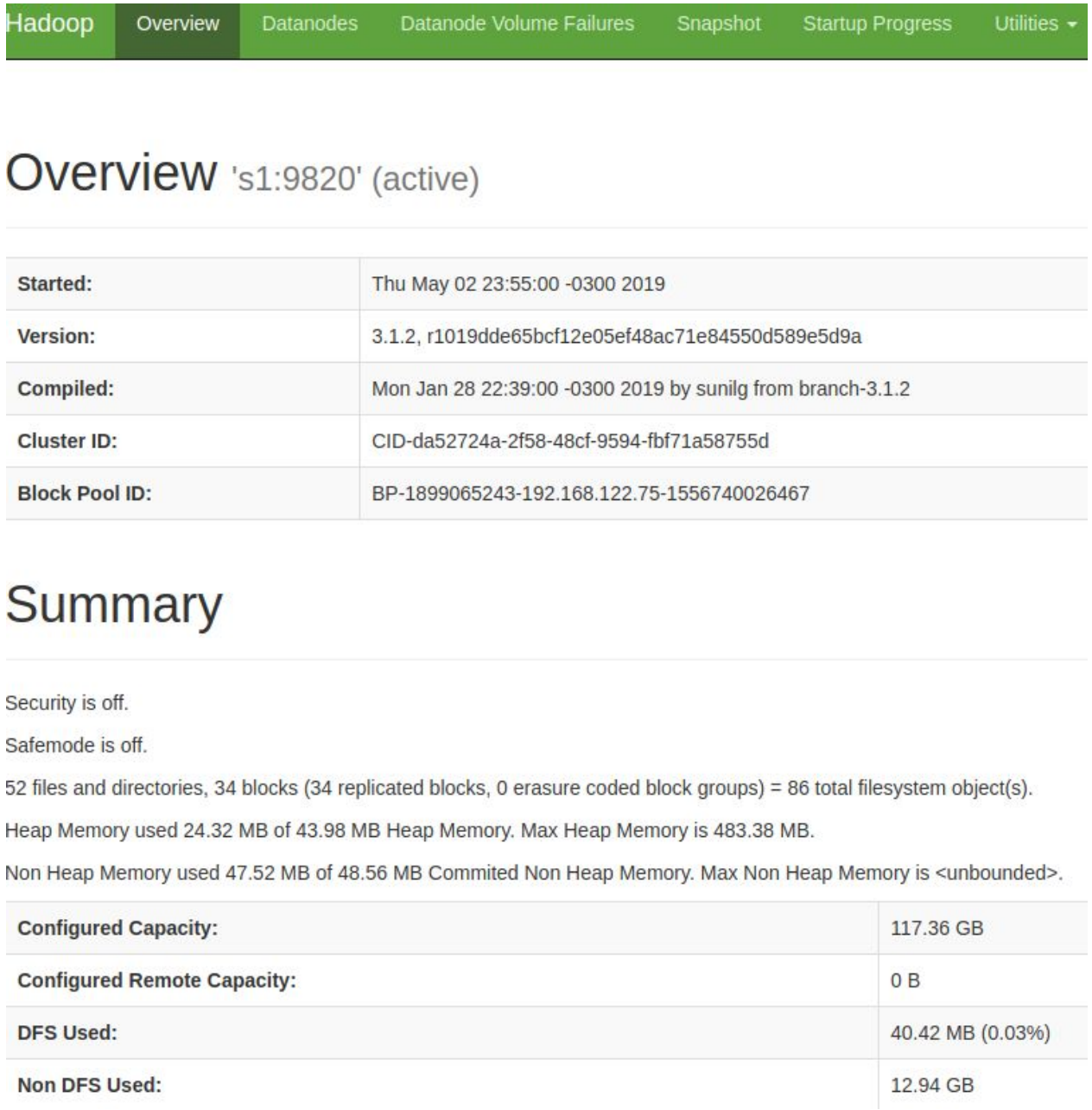
HDFS


Block Replication
Namenode (Filename, numReplicas, block-ids, …)
/users/sameerp/data/part-0, r:2, {1,3}, …
/users/sameerp/data/part-1, r:3, {2,4,5}, …
Datanodes
1
2
2
1
4
2
5
5
3
4
3
5
4

HDFS


NameNode
File Block Mappings:
/user/aaron/data1.txt -> 1, 2, 3
/user/aaron/data2.txt -> 4, 5
/user/andrew/data3.txt -> 6, 7
DataNode(s)
5 1
4
2
3
4 2
6
3
7 5
2
1 4
6
7
3 7
6
1
5

HDFS


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Overview 's1:9820' (active)
Started: Thu May 02 23:55:00 -0300 2019
Version: 3.1.2, r1019dde65bcf12e05ef48ac71e84550d589e5d9a
Compiled: Mon Jan 28 22:39:00 -0300 2019 by sunilg from branch-3.1.2
Cluster ID: CID-da52724a-2f58-48cf-9594-fbf71a58755d
Block Pool ID: BP-1899065243-192.168.122.75-1556740026467
Summary
Security is off.
Safemode is off.
52 files and directories, 34 blocks (34 replicated blocks, 0 erasure coded block groups) = 86 total filesystem object(s).
Heap Memory used 24.32 MB of 43.98 MB Heap Memory. Max Heap Memory is 483.38 MB.
Non Heap Memory used 47.52 MB of 48.56 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.
Configured Capacity: 117.36 GB
Configured Remote Capacity: 0 B
DFS Used: 40.42 MB (0.03%)
Non DFS Used: 12.94 GB

HDFS

Non DFS Used:	12.94 GB
DFS Remaining:	98.34 GB (83.79%)
Block Pool Used:	40.42 MB (0.03%)
DataNodes usages% (Min/Median/Max/stdDev):	0.03% / 0.03% / 0.03% / 0.00%
Live Nodes	3 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	8
Number of Blocks Pending Deletion	0
Block Deletion Start Time	Thu May 02 23:55:00 -0300 2019
Last Checkpoint Time	Thu May 02 23:55:02 -0300 2019

NameNode Journal Status

Current transaction ID: 594

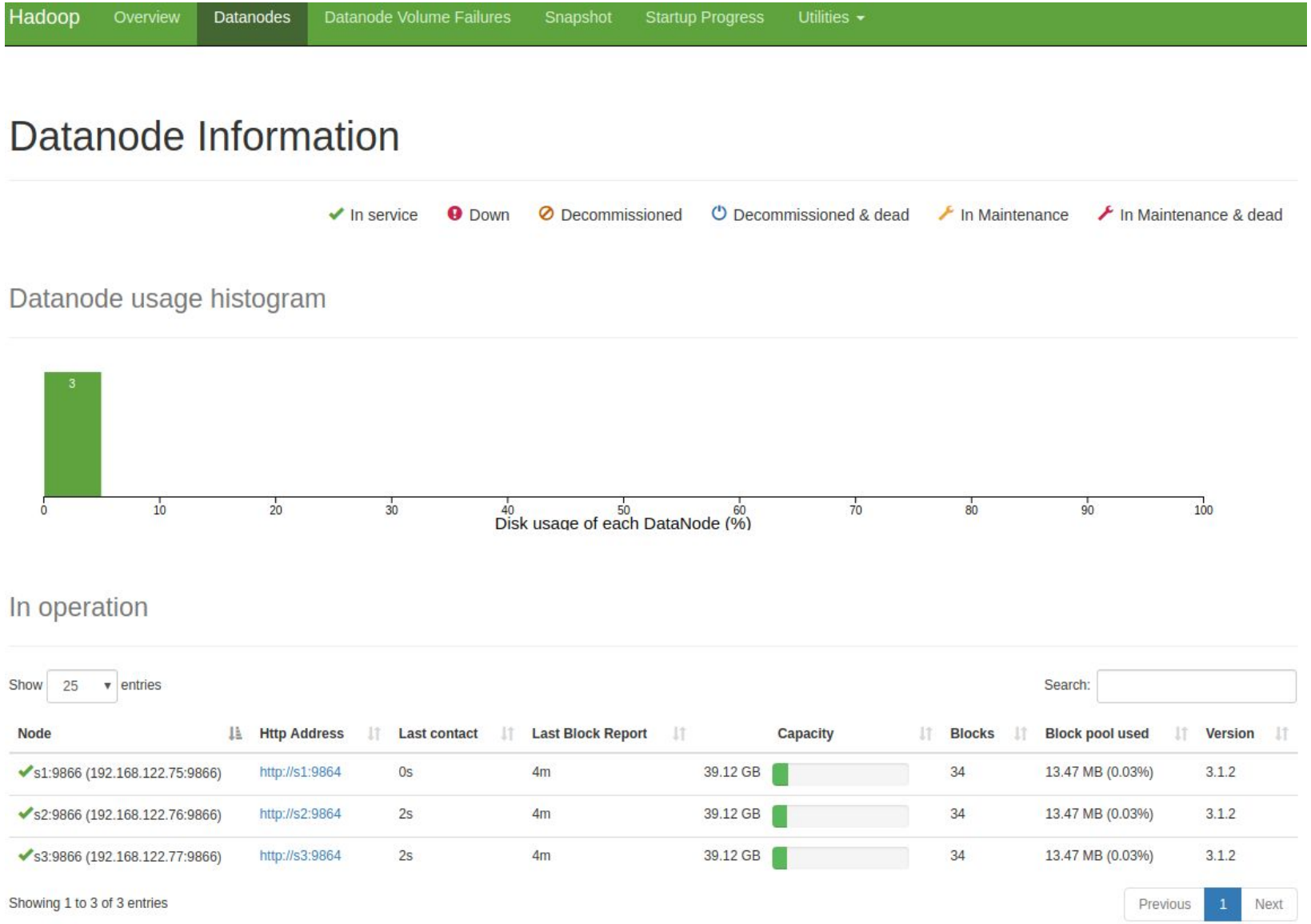
Journal Manager	State
FileJournalManager(root=/opt/hdfs/namenode)	EditLogFileOutputStream(/opt/hdfs/namenode/current/edits_inprogress_0000000000000000594)

NameNode Storage

Storage Directory	Type	State
/opt/hdfs/namenode	IMAGE_AND_EDITS	Active

DEPARTAMENTO DE COMPUTAÇÃO

HDFS



Hadoop

- MapReduce e HDFS são executados nos mesmos nós.
  - Isso permite que o framework escalone as tarefas para os nós que contenham os dados, resultado em uma alta largura de banda agregada em todo o cluster.
- Hadoop trabalha com modos de cluster:
  - Standalone: modo menos usado. Apenas para MapReduce acessando arquivos locais.
  - Pseudo-distributed: usado para desenvolvimento/ estudo. Os daemons rodam na mesma máquina.
  - Fully-distributed: usado em produção. Os daemons são distribuídos e executados em máquinas do cluster.

Coisas Boas

- Tolerância a falhas
  - Sistema ativo mesmo no caso de falha de alguns nós
- Self-Healing
  - Auto balanceamento dos arquivos
- Escalável
  - Adição de novos nós do cluster

Coisas Boas

- Código Aberto
  - Comunidade ativa
  - Apoio de grandes corporações
- Economia
  - Software livre
  - Uso de máquinas convencionais
- Separação da Lógica de Negócios
  - Trabalho duro fica com o Hadoop

Coisas não tão Boas

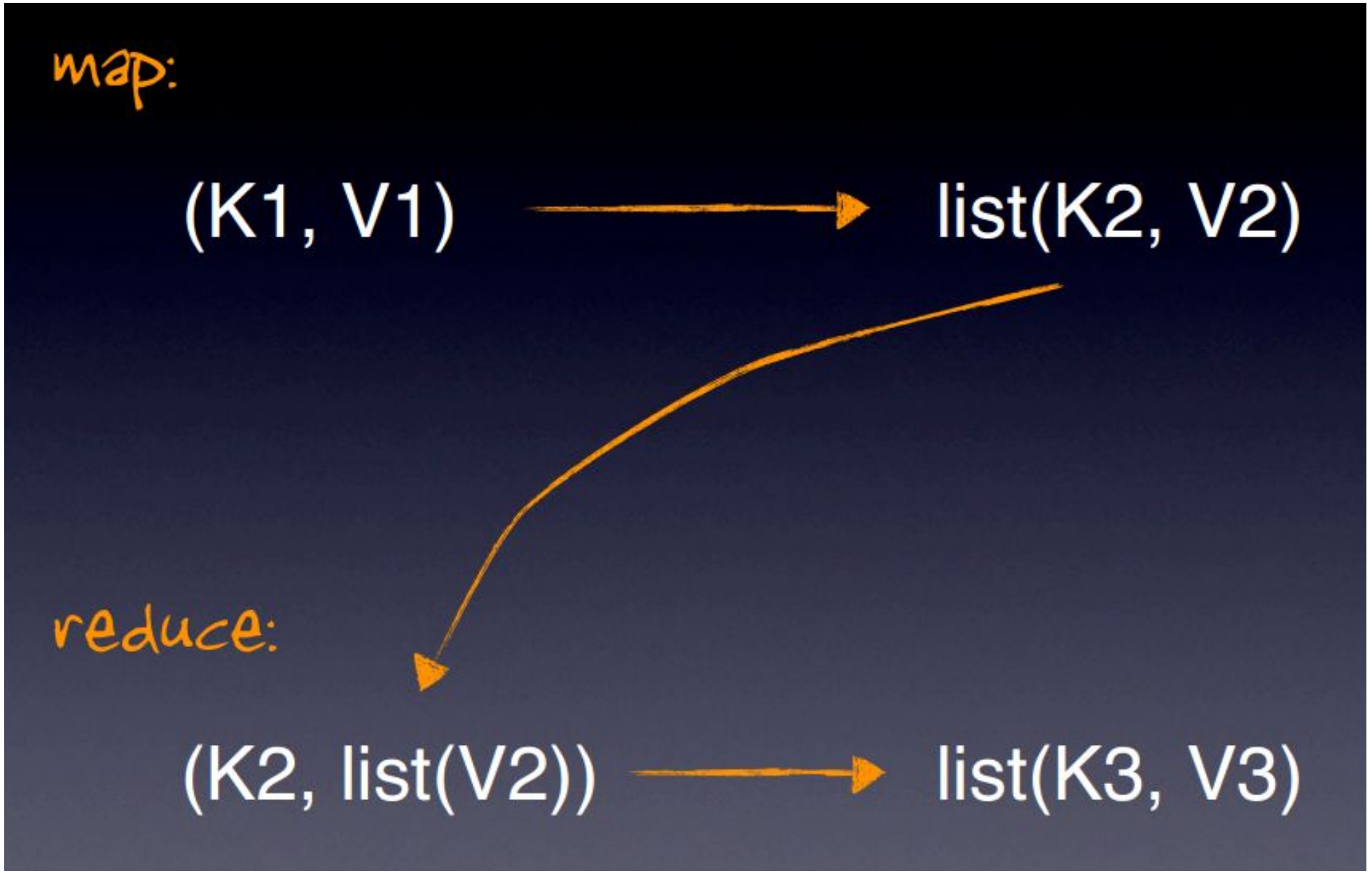
- Único nó mestre
  - Ponto crítico de falha
- Paralelização de aplicações
  - Problemas não paralelizáveis
  - Processamento de arquivos pequenos
    - Muito processamento & poucos dados

Map/Reduce

- Base na programação funcional
  - Manipulação de dados (tipicamente listas)
  - Funções que transformam dados
- Modelo de Programação proposto pelo Google
  - Duas funções básicas: MAP e REDUCE

Map/Reduce

- MAP:
  - Toma-se uma tarefa complexa ou custosa
  - Quebra tal tarefa em sub-problemas menores
  - Delega a resolução desta tarefas a nós distribuídos (workers)
- REDUCE:
  - Coleta as respostas dos workers
  - Agrega tais respostas em uma saída que representa a solução do problema complexo


map:
(K1, V1)
list(K2, V2)
reduce:
(K2, list(V2))
list(K3, V3)

Apache Hadoop

- Divide arquivos
  - Em geral, blocos de 128MB
- Usa pares de chave/valor
- Mappers
  - filtram e transformam o dado de entrada
- Reducers
  - Agregam a saída dos mappers

Importante premissa

- MOVE-SE CÓDIGO, NÃO DADOS
  - Arquivos são grandes, código não
  - Rede é um gargalo

Clássico Exemplo

- Word Count
  - Contar o número de ocorrência de uma palavra em um arquivo

Fernando Antonio Mota Trinta
Ian Gabriel Braga Trinta
Ivana Régia Braga

Map(K1,V1)

- (0, “Fernando Antonio Mota Trinta”)
- (29, “Ian Gabriel Braga Trinta”)
- (53, “Ivana Régia Braga”)

Map list(K2,V2)

- (“Fernando”, 1)
- (“ Antonio”, 1)
- (“ Mota ”, 1)
- (“ Trinta ”, 1)
- (“ Ian ”, 1)
- (“ Gabriel ”, 1)
- (“ Braga ”, 1)
- (“ Trinta ”, 1)
- (“ Ivana ”, 1)
- (“ Régia ”, 1)
- (“ Braga ”, 1)

Reduce (K2, list(V2))

- (“Fernando”, 1)
- (“ Antonio”, 1)
- (“ Mota ”, 1)
- (“ Trinta ”, (1,1))
- (“ Ian ”, 1)
- (“ Gabriel ”, 1)
- (“ Braga ”, (1,1))
- (“ Ivana ”, 1)
- (“ Régia ”, 1)

Reduce list(V3,K3)

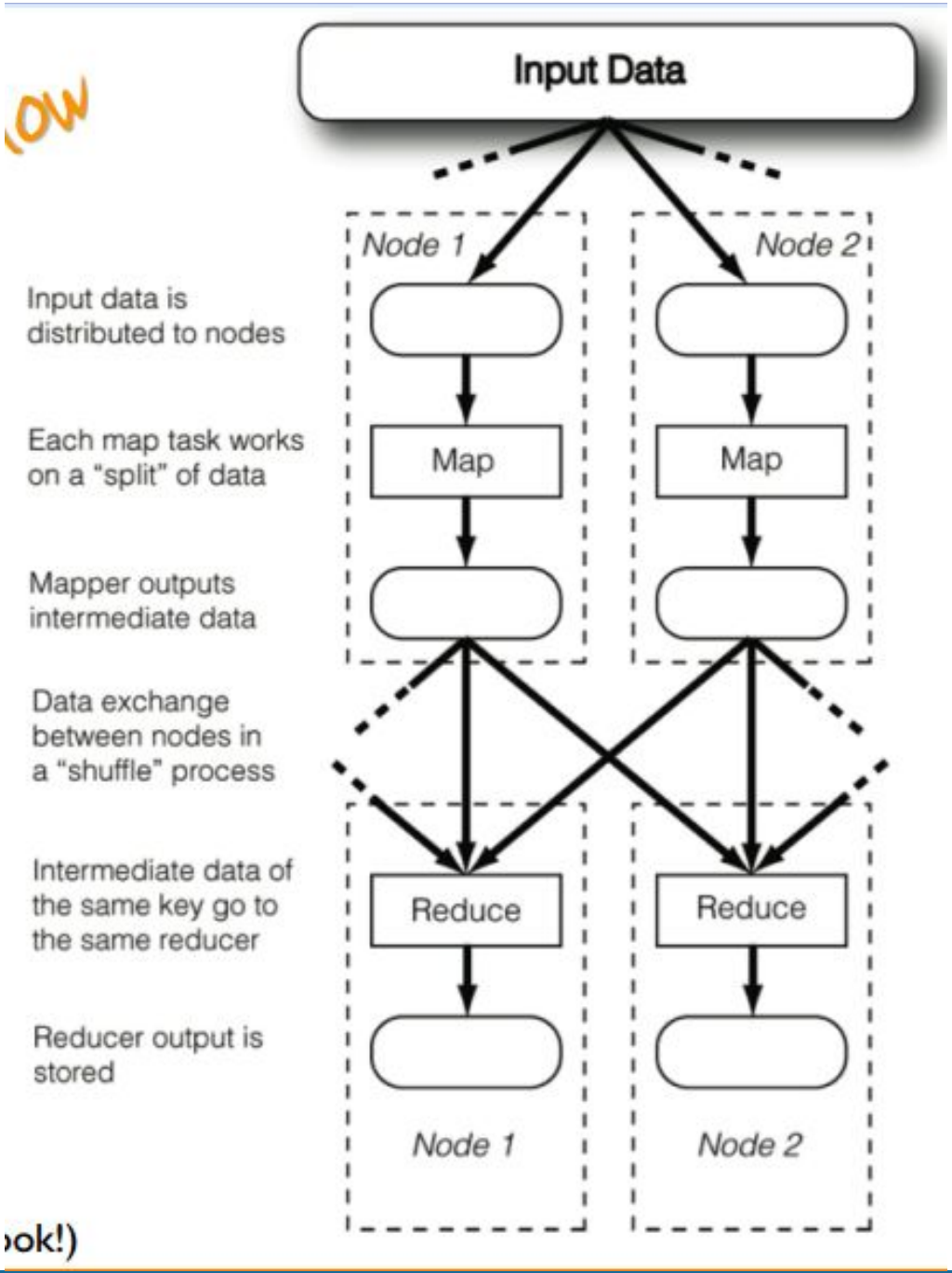
- (“Fernando”, 1)
- (“ Antonio”, 1)
- (“ Mota ”, 1)
- (“ Trinta ”, 2)
- (“ Ian ”, 1)
- (“ Gabriel ”, 1)
- (“ Braga ”, 2)
- (“ Ivana ”, 1)
- (“ Régia ”, 1)

```
public class SimpleWordCount
  extends Configured implements Tool {

  public static class MapClass
    extends Mapper<Object, Text, Text, IntWritable> {
    ...
  }

  public static class Reduce
    extends Reducer<Text, IntWritable, Text, IntWritable> {
    ...
  }

  public int run(String[] args) throws Exception { ... }

  public static void main(String[] args) { ... }
}
```

```
public static class MapClass
  extends Mapper<Object, Text, Text, IntWritable> {

  private static final IntWritable ONE = new IntWritable(1L);
  private Text word = new Text();

  @Override
  protected void map(Object key, Text value, Context context)
    throws IOException, InterruptedException {

    StringTokenizer st = new StringTokenizer(value.toString());
    while (st.hasMoreTokens()) {
      word.set(st.nextToken());
      context.write(word, ONE);
    }
  }
}
```

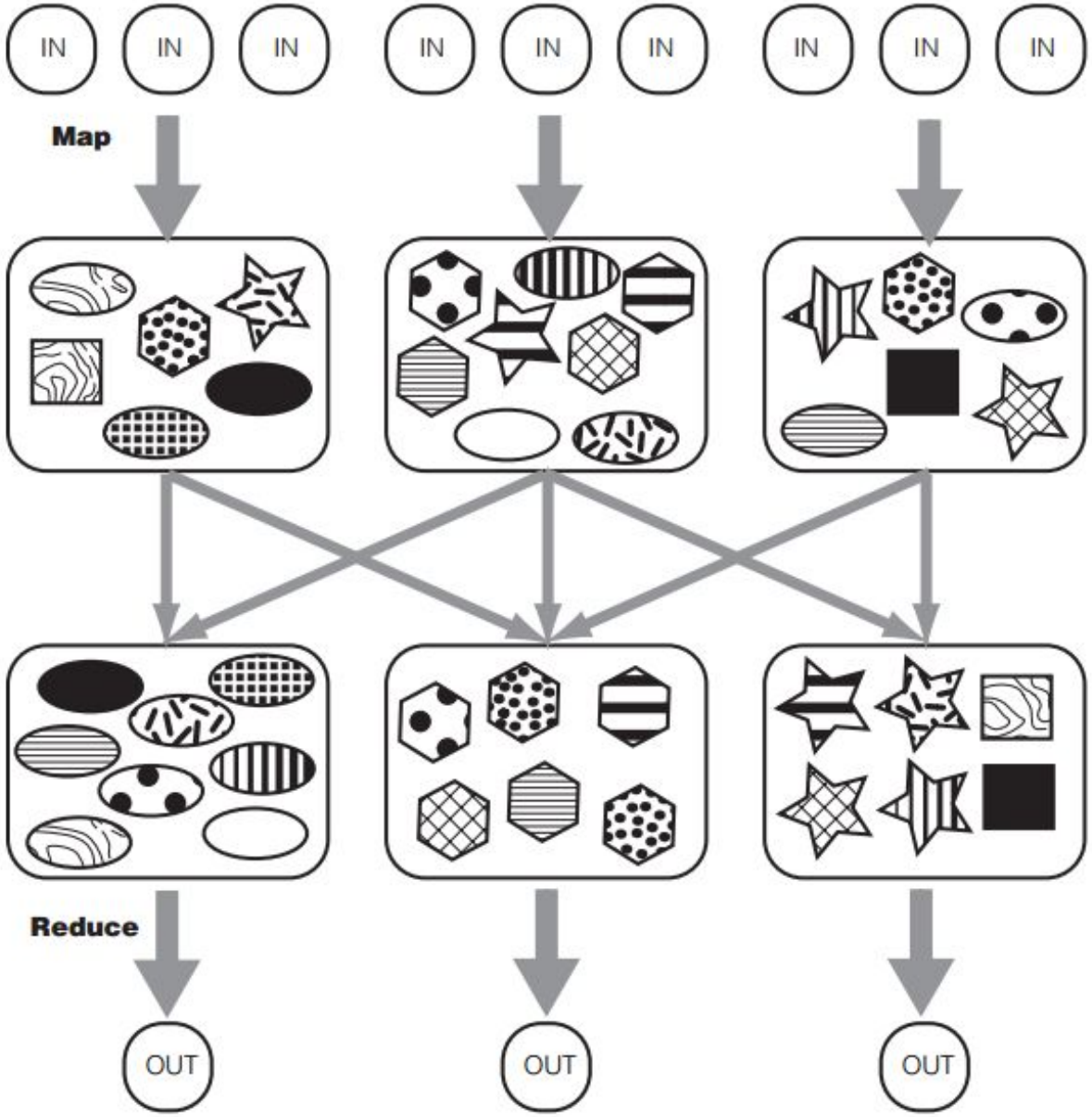
```
public static class Reduce
  extends Reducer<Text, IntWritable, Text, IntWritable> {

  private IntWritable count = new IntWritable();

  @Override
  protected void reduce(Text key, Iterable<IntWritable> values,
                        Context context)
    throws IOException, InterruptedException {

    int sum = 0;
    for (IntWritable value : values) {
      sum += value.get();
    }
    count.set(sum);
    context.write(key, count);
  }
}
```

```
public int run(String[] args) throws Exception {
  Configuration conf = getConf();

  Job job = new Job(conf, "Counting Words");
  job.setJarByClass(SimpleWordCount.class);
  job.setMapperClass(MapClass.class);
  job.setReducerClass(Reduce.class);
  job.setOutputKeyClass(Text.class);
  job.setOutputValueClass(Text.class);

  FileInputFormat.setInputPaths(job, new Path(args[0]));
  FileOutputFormat.setOutputPath(job, new Path(args[1]));

  return job.waitForCompletion(true) ? 0 : 1;
}
```


Input Data
Node 1
Node 2
Input data is distributed to nodes
Each map task works on a "split" of data
Map
Map
Mapper outputs intermediate data
Data exchange between nodes in a "shuffle" process
Intermediate data of the same key go to the same reducer
Reduce
Reduce
Reducer output is stored
Node 1
Node 2

Particionamento

- Decisão de quais chaves vão para qual reducer
- Mundo ideal
  - Distribuição uniforme entre reducers
- Evitar a sobrecarga de um único reducer

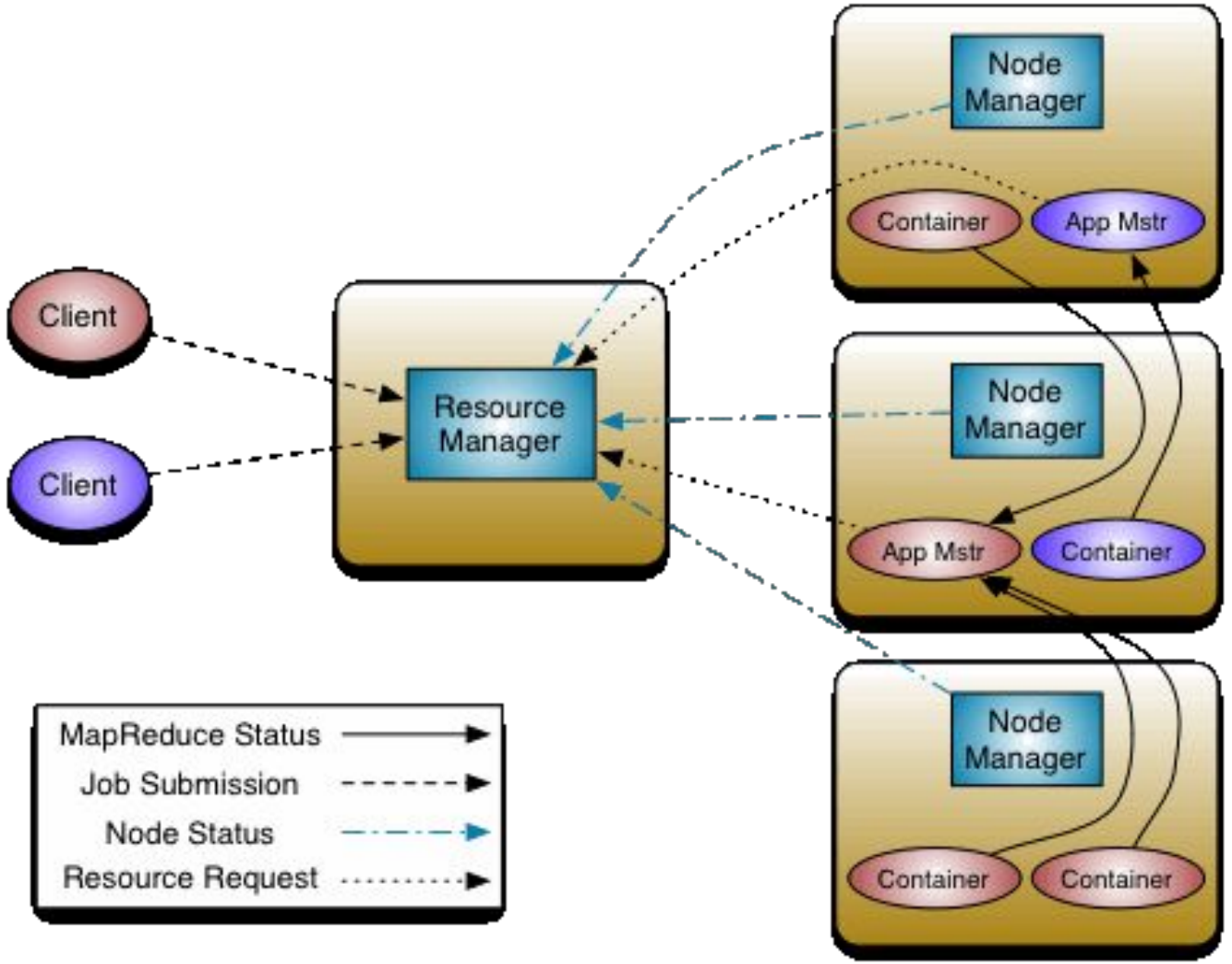

IN
IN
IN
IN
IN
IN
IN
IN
IN
Map
Reduce
OUT
OUT
OUT

Melhorando o Desempenho

- Combinando localmente
- Reduzir o processo de agregação nos mappers
  - Aka "Local Reduce"

	Dado	*# pares k/v agregados*
Sem combinação	*("Maria", 1)*	*1000*
Com combinação	*("Maria", 1000)*	*1*

Arquitetura Hadoop YARN



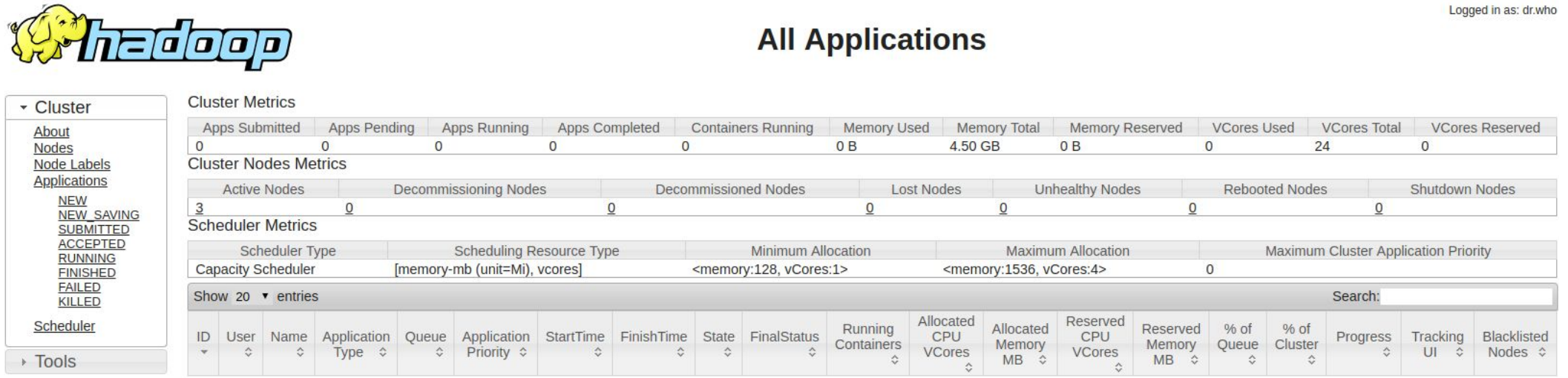
Resource Manager

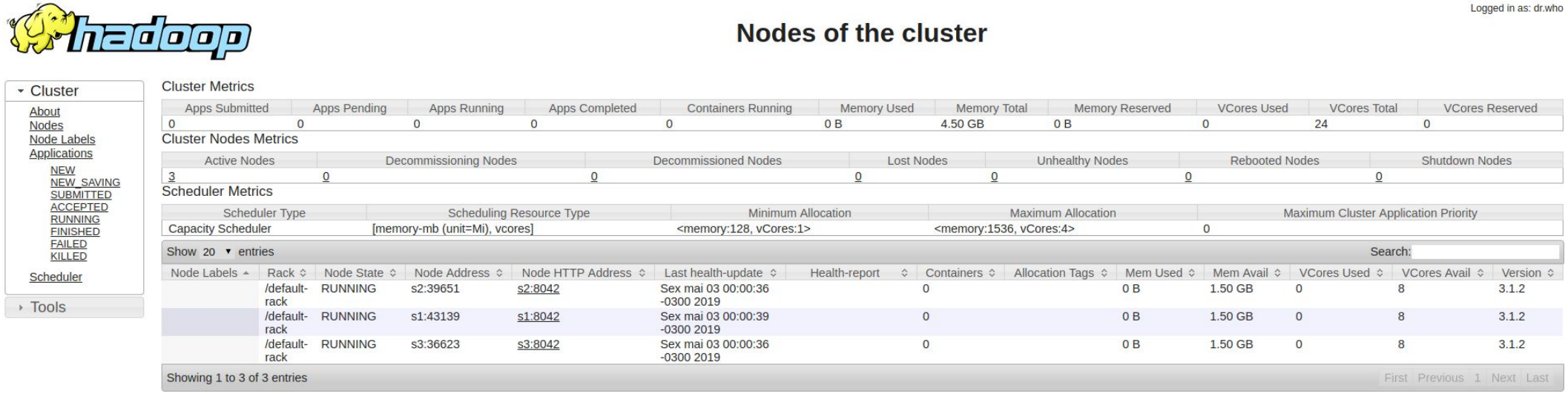
- Nó principal do YARN, responsável por fazer o inventário de recursos disponíveis no cluster e executar vários serviços críticos, sendo o mais importante deles o Scheduler.
- Scheduler aloca recursos para executar os jobs. Mas não monitora ou rastreia o status/ progresso do aplicativo.
  - Não garante reinício das tarefas caso falhem.
- Funciona em conjunto com um NodeManager por nó e o ApplicationMaster por aplicativo.

NodeManagers

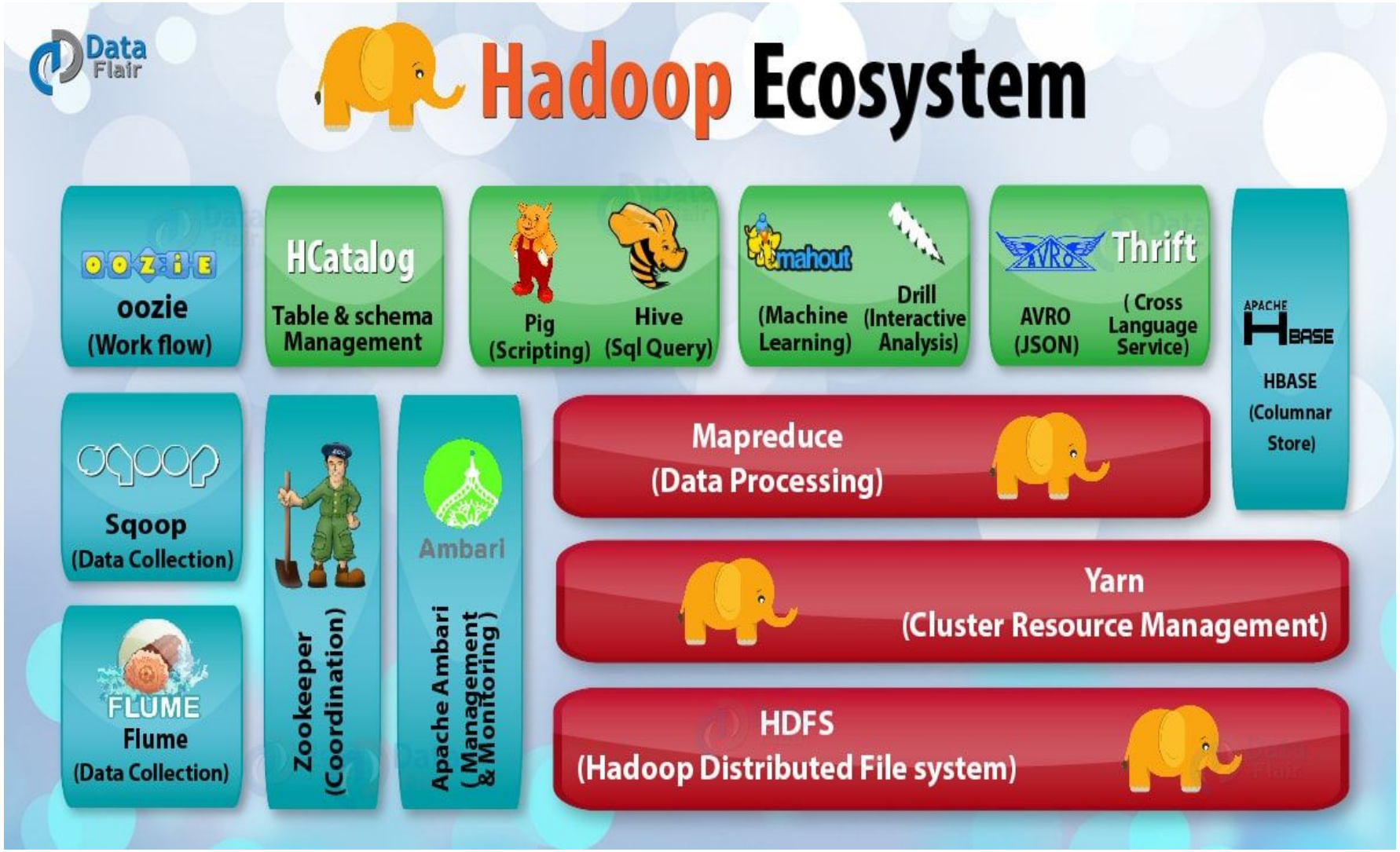
- Os NodeManagers recebem instruções do ResourceManager e gerenciam recursos disponíveis em um único nó.
- ApplicationMasters são responsáveis por negociar recursos com o ResourceManager e por trabalhar com os NodeManagers para iniciar os contêineres.
  - Ele gerencia o ciclo de vida da aplicação.
  - Cada aplicativo tem um ApplicationMaster.

YARN


Logged in as: dr.who
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 | Memory Reserved | VCores Used | VCores Total | VCores Reserved
0 | 0 | 0 | 0 | 0 | 0 B | 4.50 GB | 0 B | 0 | 24 | 0
Cluster Nodes Metrics
Active Nodes | Decommissioning Nodes | Decommissioned Nodes | Lost Nodes | Unhealthy Nodes | Rebooted Nodes | Shutdown Nodes
3 | 0 | 0 | 0 | 0 | 0 | 0
Scheduler Metrics
Scheduler Type | Scheduling Resource Type | Minimum Allocation | Maximum Allocation | Maximum Cluster Application Priority
Capacity Scheduler | [memory-mb (unit=Mi), vcores] | <memory:128, vCores:1> | <memory:1536, vCores:4> | 0
Show 20 entries
Search:
ID | User | Name | Application Type | Queue | Application Priority | StartTime | FinishTime | State | FinalStatus | Running Containers | Allocated CPU VCores | Allocated Memory MB | Reserved CPU VCores | Reserved Memory MB | % of Queue | % of Cluster | Progress | Tracking UI | Blacklisted Nodes


Logged in as: dr.who
Nodes of the cluster
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 | Memory Reserved | VCores Used | VCores Total | VCores Reserved
0 | 0 | 0 | 0 | 0 | 0 B | 4.50 GB | 0 B | 0 | 24 | 0
Cluster Nodes Metrics
Active Nodes | Decommissioning Nodes | Decommissioned Nodes | Lost Nodes | Unhealthy Nodes | Rebooted Nodes | Shutdown Nodes
3 | 0 | 0 | 0 | 0 | 0 | 0
Scheduler Metrics
Scheduler Type | Scheduling Resource Type | Minimum Allocation | Maximum Allocation | Maximum Cluster Application Priority
Capacity Scheduler | [memory-mb (unit=Mi), vcores] | <memory:128, vCores:1> | <memory:1536, vCores:4> | 0
Show 20 entries
Search:
Node Labels | Rack | Node State | Node Address | Node HTTP Address | Last health-update | Health-report | Containers | Allocation Tags | Mem Used | Mem Avail | VCores Used | VCores Avail | Version
/default-rack | RUNNING | s2:39651 | s2:8042 | Sex mai 03 00:00:36 -0300 2019 | | 0 | | 0 B | 1.50 GB | 0 | 8 | 3.1.2
/default-rack | RUNNING | s1:43139 | s1:8042 | Sex mai 03 00:00:39 -0300 2019 | | 0 | | 0 B | 1.50 GB | 0 | 8 | 3.1.2
/default-rack | RUNNING | s3:36623 | s3:8042 | Sex mai 03 00:00:36 -0300 2019 | | 0 | | 0 B | 1.50 GB | 0 | 8 | 3.1.2
Showing 1 to 3 of 3 entries
First Previous 1 Next Last

Ecossistema Hadoop



Dúvidas?