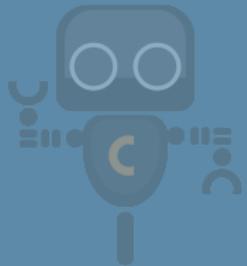


Fundamentos e Arquitetura de Bigdata, Banco de dados não relacionais

Aula 2



cienc

Francisco Nauber Bernardo Gois

Instrutor



F. Nauber Bernardo Gois

Dsc. Informática Aplicada
Líder de Aprendizado de
Máquina na Secretaria de
Saúde do Ceará
Engenheiro de Aprendizado
de Máquina

Instrutor



[https://www.linkedin.com/in/n](https://www.linkedin.com/in/naubergois/)

[https://www.linkedin.com
/in/naubergois/](https://www.linkedin.com/in/naubergois/)

Envie recomendações, de-
poimentos e competÊncias

Instrutor



ciencIA

<https://www.youtube.com/canaldaciencia>

[https://www.youtube.com/
canaldaciencia](https://www.youtube.com/canaldaciencia)



Índice

Índice

Material Utilizado no Curso

Introdução a Big Data

CAP

Hadoop

HDFS

Pig

Conclusão





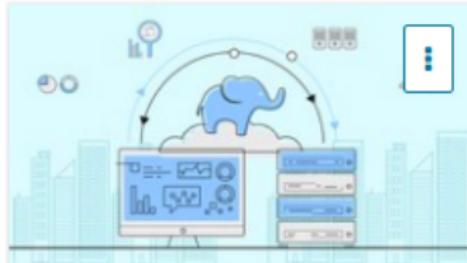
The Ultimate Hands-On Hadoop - Tame your Big Data!

Sundog Education by Frank Kane, Founder, Sundog...

6% Complete



Deixe uma classificação



Learn Big Data: The Hadoop Ecosystem Masterclass

Edward Viaene, DevOps, Cloud, Big Data Specialist

8% Complete



Sua classificação

Cursos Recomendados



Material do Curso

HDP 2.5 sandbox image

Baixe a imagem no endereço <https://www.cloudera.com/downloads/hortonworks-sandbox/hdp.html>

CLOUDERA

Why Cloudera

Products

Solutions

Services & Support

Hortonworks Data Platform (HDP[®]) on Hortonworks Sandbox

The HDP Sandbox makes it easy to get started with Apache Hadoop, Apache Spark, Apache Hive, Apache HBase, Druid and Data Analytics Studio (DAS).

Get Started Now

CHOOSE INSTALLATION TYPE

LET'S GO! →



Material do Curso

Chris Albon

Curso de Aprendizado de Máquina

<https://chrisalbon.com/>

CHRIS ALBON

TECHNICAL NOTES ▾ ARTICLES

Notes On Using

Data Science & Artificial Intelligence

To Fight For Something That Matters

I am a data scientist with a decade of experience applying statistical learning, artificial intelligence, and software engineering to political, social, and humanitarian efforts -- from election monitoring to disaster relief. I lead the data science team at [Devoted Health](#), helping fix America's health care system.

Material do Curso

HDP 2.5 sandbox image

Baixe a imagem no endereço https://www.cloudera.com/downloads/hortonworks-sandbox/hdf.html?utm_source=HDF

Thank you for choosing Cloudera DataFlow (A)

Sandbox HDFS Docker Downloads

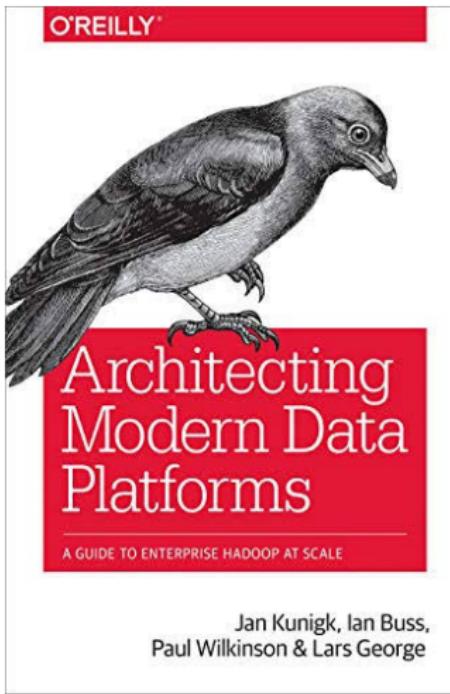
HDF Docker 3.1.1 (Latest)

Install Guide on Docker

Cloudera DataFlow (Ambari)
on Sandbox



Getting Started with Cloudera DataFlow (Ambari) >



Livro Recomendado



Livro

Infolab

Stanford University



Mining of Massive Datasets

Jure Leskovec, Anand Rajaraman, Jeff Ullman

- Home
- Book & Slides
- Stanford Courses
- Supporting Materials

Big-data is transforming the world. Here you will learn data mining and machine learning techniques to process large datasets and extract valuable knowledge from them.

The book



The book is based on Stanford Computer Science course [CS246: Mining Massive Datasets](#) (and [CS345A: Data Mining](#)).

The book, like the course, is designed at the undergraduate computer science level with no formal prerequisites. To support deeper explorations, most of the chapters are supplemented with further reading references.

<http://mmds.org/>

Índice

Índice

Material Utilizado no Curso

Introdução a Big Data

CAP

Hadoop

HDFS

Pig

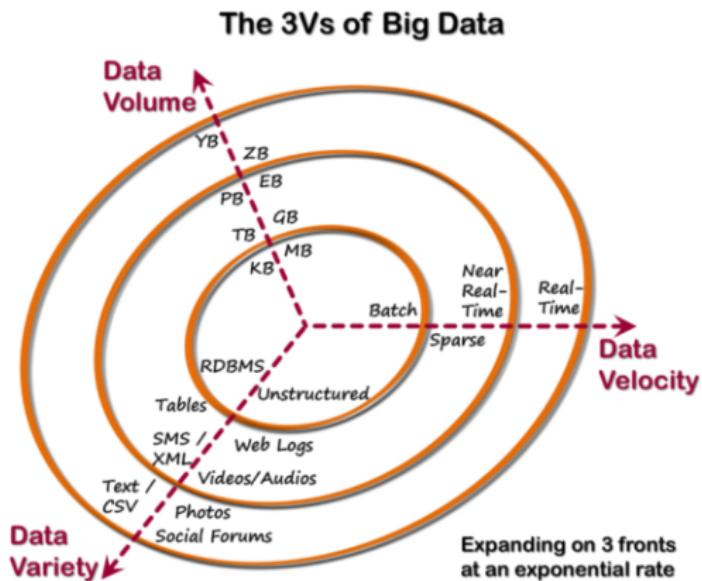
Conclusão



In 60 seconds..._



3 Vs



<http://www.hadooptechs.com/>

Dados necessitam estar armazenados

But to extract the knowledge data
needs to be

- Stored (systems)
- Managed (databases)
- And ANALYZED ← this class

Data Mining ≈ Big Data ≈
Predictive Analytics ≈
Data Science ≈ Machine Learning

[http://web.stanford.edu/class/cs246/slides/
01-intro.pdf](http://web.stanford.edu/class/cs246/slides/01-intro.pdf)



Computação em Larga Escala

- Large-scale computing for data mining problems on commodity hardware
- Challenges:
 - How do you distribute computation?
 - How can we make it easy to write distributed programs?
 - Machines fail:
 - One server may stay up 3 years (1,000 days)
 - If you have 1,000 servers, expect to lose 1/day
 - With 1M machines 1,000 machines fail every day!

<http://web.stanford.edu/class/cs246/slides/01-intro.pdf>



Copiando os dados na rede

- **Issue:**

Copying data over a network takes time

- **Idea:**

- Bring computation to data
- Store files multiple times for reliability

- **Spark/Hadoop address these problems**

- **Storage Infrastructure – File system**

- Google: GFS. Hadoop: HDFS

- **Programming model**

- MapReduce
 - Spark

<http://web.stanford.edu/class/cs246/slides/01-intro.pdf>



Distributed File System

■ **Chunk servers**

- File is split into contiguous chunks
- Typically each chunk is 16-64MB
- Each chunk replicated (usually 2x or 3x)
- Try to keep replicas in different racks

■ **Master node**

- a.k.a. Name Node in Hadoop's HDFS
- Stores metadata about where files are stored
- Might be replicated

■ **Client library for file access**

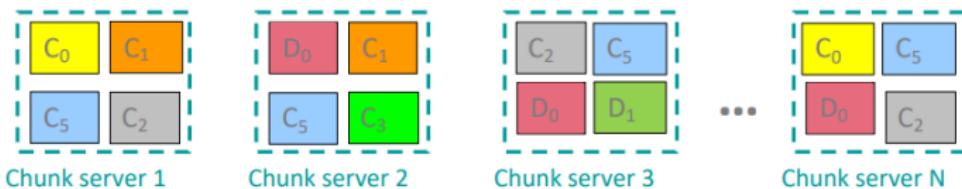
- Talks to master to find chunk servers
- Connects directly to chunk servers to access data

http://web.stanford.edu/class/cs246/slides/01_intro.pdf



Distributed File System

- Reliable distributed file system
- Data kept in “chunks” spread across machines
- Each chunk replicated on different machines
 - Seamless recovery from disk or machine failure

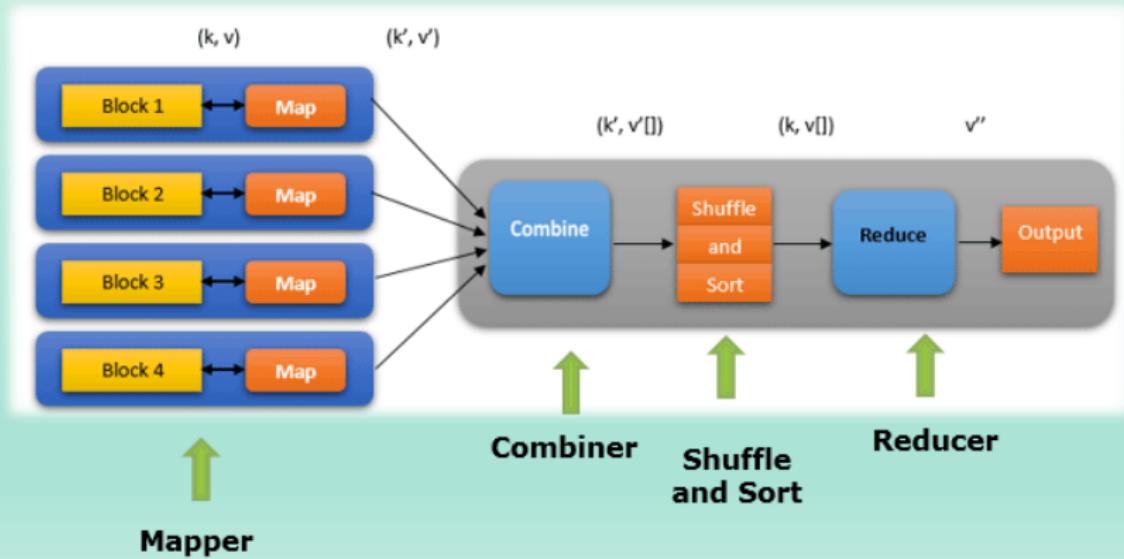


Bring computation directly to the data!

Chunk servers also serve as compute servers

<http://web.stanford.edu/class/cs246/slides/01-intro.pdf>

How MapReduce Works

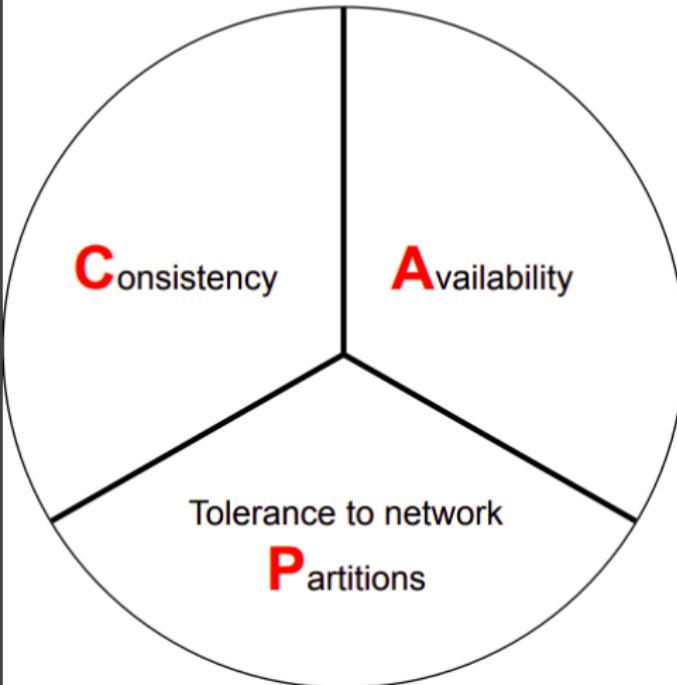


Why MapReduce?

- Distributes the processing of data on your cluster
- Divides your data up into partitions that are MAPPED (transformed) and REDUCED (aggregated) by mapper and reducer functions you define
- Resilient to failure - an application master monitors your mappers and reducers on each partition



The CAP Theorem



Theorem: You can have **at most two** of these invariants for any shared-data system

Corollary: consistency boundary must choose A or P

CAP Theorem

- ▶ 2000: Eric Brewer, PODC conference keynote
- ▶ 2002: Seth Gilbert and Nancy Lynch, ACM SIGACT News 33(2)

*“Of three properties of shared-data systems
(Consistency, Availability and
tolerance to network Partitions) only two can
be achieved at any given moment in time.”*

Índice

Índice

Material Utilizado no Curso

Introdução a Big Data

CAP

Hadoop

HDFS

Pig

Conclusão





What is Hadoop?

Apache Hadoop is an open source software framework for storage and large scale processing of data-sets on clusters of commodity hardware

<https://www.coursera.org/learn/hadoop/lecture/ZbVQf/hadoop-stack-basics>



Hadoop

O que é o Hadoop?

Criado por Doug Cutting em 2005 com o nome do elefante do filho.



Hadoop

O que é o Hadoop?

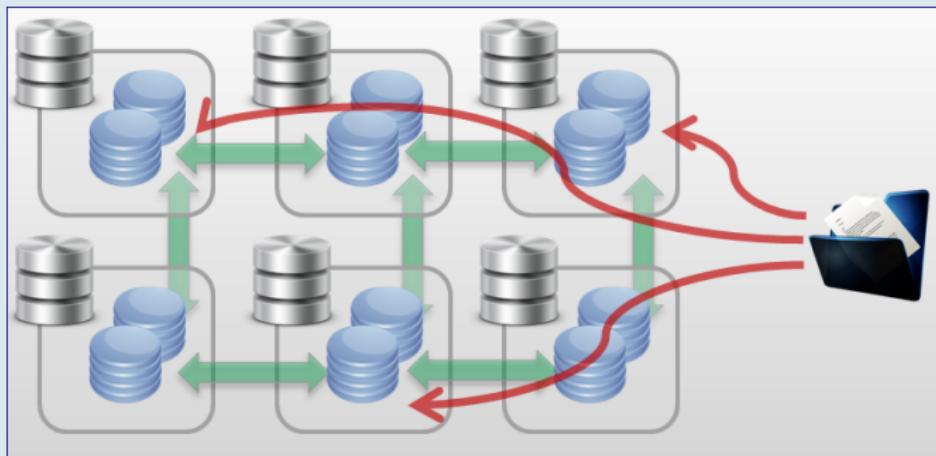
A biblioteca de software Apache Hadoop é uma estrutura que permite o processamento distribuído de grandes conjuntos de dados entre clusters de computadores usando modelos de programação simples. Ele foi projetado para expandir de servidores únicos para milhares de máquinas, cada uma oferecendo computação e armazenamento local. <https://hadoop.apache.org/>



Hadoop

O que é o Hadoop?

Movendo o processamento para os dados.





Why Hadoop?

Data Growth is mind boggling. Forecast for 2020: 40 Trillion GB

Cost effective

Scalable

Fast

Open source



Why Hadoop?



- Data's too darn big - terabytes per day
- Vertical scaling doesn't cut it
 - Disk seek times
 - Hardware failures
 - Processing times
- Horizontal scaling is linear
- Hadoop: It's not just for batch processing

the-ultimate-hands-on-hadoop-tame-your-big-data/learn/lecture/

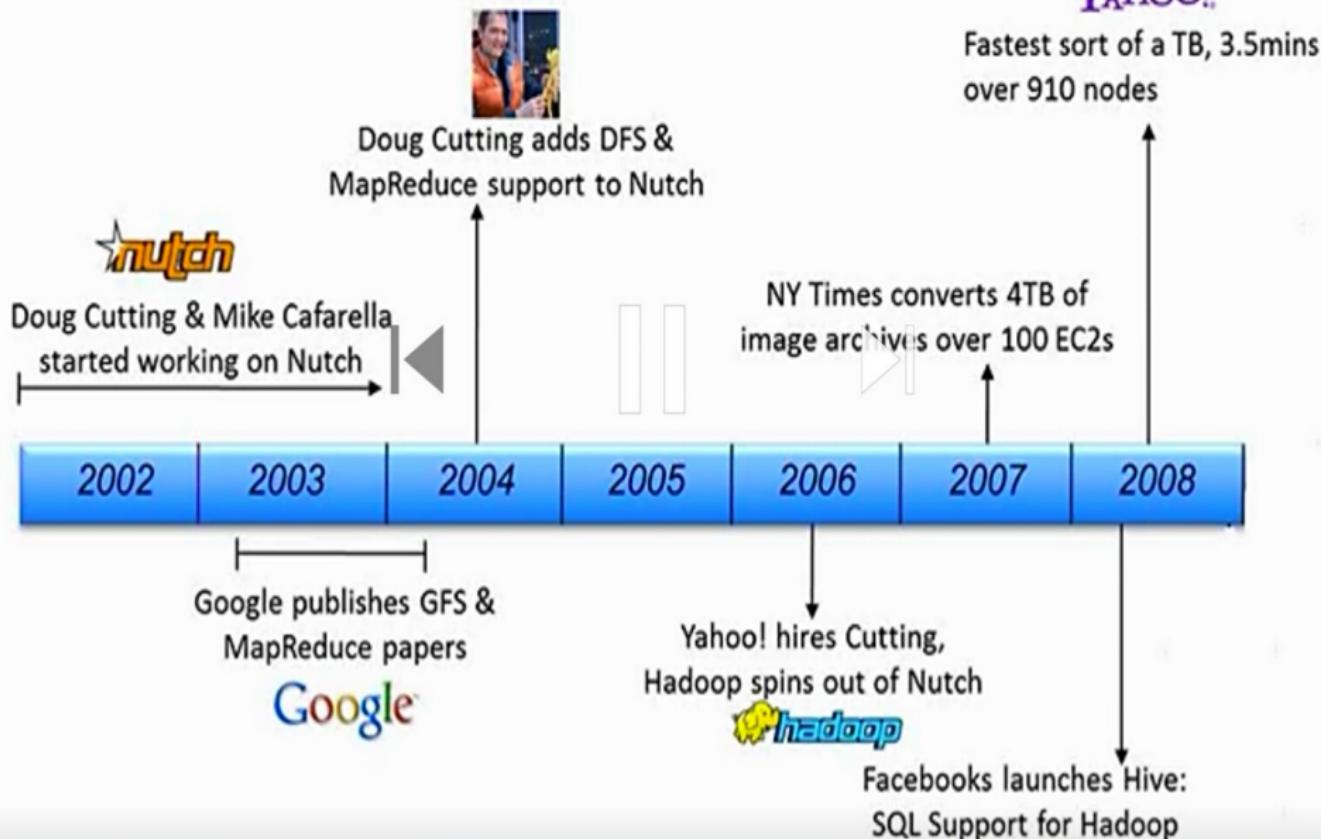
Hadoop

Porque usar o Hadoop?

1- FLEXÍVEL Nenhum outro meio de armazenamento e/ou processamento de dados tem a capacidade de armazenar desde conteúdos estruturados como tabelas e arquivos texto, até vídeos, fotos, etc. Com sua estrutura baseada em file system, seu armazenamento tem as características do sistema operacional Linux e comandos POSIX.



Hadoop Creation History



Security



Meta Data Management

Apache Atlas

Data Format

Parquet,
Avro,
ORC, Arrow

Coordinate & Management

In-Memory Processing

Stream Processing

SQL Over Hadoop

NoSQL Database

Search Engine

Data Piping

Machine Learning

Scripting

Scheduler



Zookeeper



Storm



Ambari

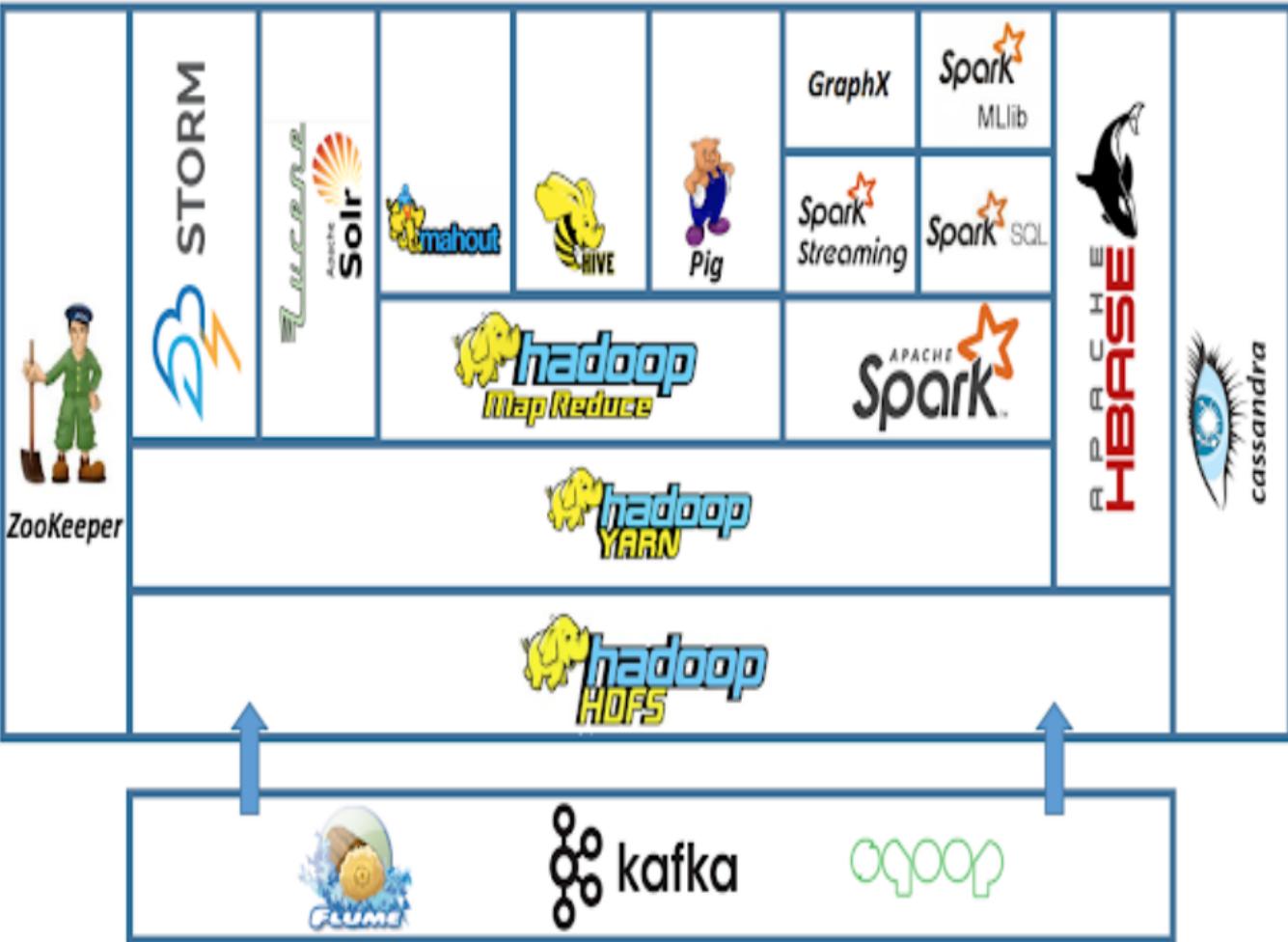
Resource Management



Airflow

Storage





MapReduce



YARN



HDFS

the-ultimate-hands-on-hadoop-tame-your-big-data/learn/lecture/



HDFS

HDFS

O HDFS é um projeto da Apache Software Foundation e um subprojeto do projeto Apache Hadoop (veja Recursos). O Hadoop é ideal para armazenar grandes quantidades de dados, do porte de terabytes e pentabytes, e usa o HDFS como sistema de armazenamento. O HDFS permite a conexão de nós (computadores pessoais padrão) contidos nos clusters por meio dos quais os arquivos de dados são distribuídos.

<https://www.ibm.com/developerworks/br/library/wa-introhdfs/index.html>



Hive



O Apache Hive ™ facilita a leitura, gravação e gerenciamento de grandes conjuntos de dados residentes no armazenamento distribuído usando SQL.

<https://hive.apache.org/>



GENERAL

[Home](#)

[Downloads](#)

[License](#)

[Privacy Policy](#)

DOCUMENTATION

[Language Manual](#)

[Javadoc](#)

[Wiki](#)

COMMUNITY

[Becoming a Committer](#)

[Edit Website](#)

[How to Contribute](#)

[Resources for contributors](#)

[Issue Tracking](#)

[Mailing Lists](#)

[People](#)

DEVELOPMENT

[Builds](#)

[Design Docs](#)

[FAQ](#)

APACHE HIVE TM

The Apache Hive ™ data warehouse software facilitates reading, writing, and managing large datasets residing in distributed storage using SQL. Structure can be projected onto data already in storage. A command line tool and JDBC driver are provided to connect users to Hive.

Getting Started With Apache Hive Software

- Check out the [Getting Started Guide](#) on the [Hive wiki](#).
- Learn more [About Hive's Functionality](#) on [our wiki](#)
- Read the [Getting Started Guide](#) to learn how to install Hive
- The [User and Hive SQL documentation](#) shows how to program Hive

Getting Involved With The Apache Hive Community

Apache Hive is an open source project run by volunteers at the Apache Software Foundation. Previously it was a subproject of [Apache® Hadoop®](#), but has now graduated to become a top-level project of its own. We encourage you to learn about the project and contribute your expertise.

Módulos do Hadoop

Pig

O Apache Pig altera isto criando uma abstração de linguagem de procedimentos mais simples sobre o MapReduce para expor uma interface mais parecida com Structured Query Language (SQL) para aplicativos Hadoop.

<https://pig.apache.org/>





Project

Wiki

Project

[Welcome](#)[Releases](#)[About](#)[Mailing Lists](#)[Who We Are](#)[Bylaws](#)[Pig Tools](#)[Privacy Policy](#)[Sponsorship](#)[Thanks](#)

› Documentation

› Developers

[LEARN MORE](#)

Welcome to Apache Pig!

News

[Apache Pig 0.17.0 is released!](#)

Getting Started

[Getting Involved](#)

Apache Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for executing these programs on large clusters of commodity machines. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

At the present time, Pig's infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs, for which large-scale parallel execution is provided by the Hadoop subproject. Pig's language layer currently consists of a textual language called Pig Latin, which has the following key properties:

- **Ease of programming.** It is trivial to achieve parallel execution of simple, "embarrassingly parallel" data analysis tasks. Complex tasks comprising multiple parallel steps and data transformations are explicitly encoded as data flow sequences, making them easy to write, understand, and maintain.
- **Optimization opportunities.** The way in which tasks are encoded permits the system to optimize their execution automatically, allowing the user to focus on the logic of the program rather than its efficiency.
- **Extensibility.** Users can create their own functions to do special-purpose processing.

Apache Pig is released under the [Apache 2.0 License](#).

Ambari



Apache Ambari

O projeto Apache Ambari tem como objetivo simplificar o gerenciamento do Hadoop, desenvolvendo software para provisionar, gerenciar e monitorar clusters do Apache Hadoop. O Ambari fornece uma UI da Web de gerenciamento Hadoop intuitiva e fácil de usar, apoiada por suas APIs RESTful. <https://ambari.apache.org/>



AMBARI

Overview

[What's New?](#)[Project Team](#)[IRC Channel](#)[Mailing Lists](#)[Issue Tracking](#)[User Group](#)[Project License](#)

DOCUMENTATION

[Wiki](#)[Quick Start Guide](#)[Features + Roadmap](#)[API Reference](#)

Introduction

The Apache Ambari project is aimed at making Hadoop management simpler by developing software for provisioning, managing, and monitoring Apache Hadoop clusters. Ambari provides an intuitive, easy-to-use Hadoop management web UI backed by its RESTful APIs.

Ambari enables System Administrators to:

- Provision a Hadoop Cluster
 - Ambari provides a step-by-step wizard for installing Hadoop services across any number of hosts.
 - Ambari handles configuration of Hadoop services for the cluster.
- Manage a Hadoop Cluster
 - Ambari provides central management for starting, stopping, and reconfiguring Hadoop services across the entire cluster.
- Monitor a Hadoop Cluster
 - Ambari provides a dashboard for monitoring health and status of the Hadoop cluster.
 - Ambari leverages [Ambari Metrics System](#) for metrics collection.
 - Ambari leverages [Ambari Alert Framework](#) for system alerting and will notify you when your attention is needed (e.g., a node goes down, remaining disk space is low, etc).

Ambari enables Application Developers and System Integrators to:

Mesos



O Apache Mesos abstrai a CPU, a memória, o armazenamento e outros recursos de computação das máquinas (físicas ou virtuais), permitindo que sistemas distribuídos elásticos e tolerantes a falhas sejam facilmente construídos e executados com eficiência.

<http://mesos.apache.org/>



Program against your datacenter like it's a single pool of resources

Apache Mesos abstracts CPU, memory, storage, and other compute resources away from machines (physical or virtual), enabling fault-tolerant and elastic distributed systems to easily be built and run effectively.

[Download Mesos](#)[Mesos 1.9.0 Changelog](#)

Spark



Apache Spark™ is a unified analytics engine for large-scale data processing.

<https://spark.apache.org/>

Tez



O projeto Apache TEZ® visa criar uma estrutura de aplicativos que permita um gráfico acíclico direcionado complexo de tarefas para o processamento de dados. Atualmente, ele é construído sobre o Apache Hadoop YARN.
<https://tez.apache.org/>

HBase



HBase é um banco de dados distribuído open-source orientado a coluna, modelado a partir do Google BigTable e escrito em Java. O Hbase tem fácil integração com o Hadoop, sendo assim, pode utilizar o MapReduce para distribuir o processamento dos dados, podendo processar facilmente vários terabytes de dados.

<https://hbase.apache.org/>



Storm



O Apache Storm é um sistema de computação em tempo real distribuído de código aberto e gratuito. O Apache Storm facilita o processamento confiável de fluxos de dados ilimitados, fazendo para o processamento em tempo real o que o Hadoop fez no processamento em lote.

<https://storm.apache.org/>



Oozie



Oozie é um sistema de agendador de fluxo de trabalho para gerenciar tarefas do Apache Hadoop.

Os trabalhos do Oozie Workflow são gráficos acíclicos direcionados (DAGs) de ações.

Os trabalhos do Oozie Coordinator são trabalhos recorrentes do Oozie Workflow acionados por tempo (frequência) e disponibilidade de dados.

<https://oozie.apache.org/>



Zookeeper



O ZooKeeper é um serviço centralizado para manter informações de configuração, nomear, fornecer sincronização distribuída e fornecer serviços de grupo. Todos esses tipos de serviços são usados de uma forma ou de outra por aplicativos distribuídos.

<https://zookeeper.apache.org/>



SQOOP



O Apache Sqoop (TM) é uma ferramenta projetada para transferir eficientemente dados em massa entre o Apache Hadoop e datastores estruturados, como bancos de dados relacionais.

<https://sqoop.apache.org/>

Flume



O Flume é um serviço distribuído, confiável e disponível para coletar, agrregar e mover com eficiência grandes quantidades de dados de log. Possui uma arquitetura simples e flexível, baseada no fluxo de dados de streaming. É robusto e tolerante a falhas, com mecanismos de confiabilidade ajustáveis e muitos mecanismos de failover e recuperação.

<https://flume.apache.org/>



Kafka



O Kafka é usado para criar pipelines de dados em tempo real e aplicativos de streaming. É escalável horizontalmente, tolerante a falhas, extremamente rápido e é executado em produção em milhares de empresas.

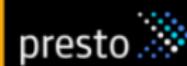
<https://kafka.apache.org/>

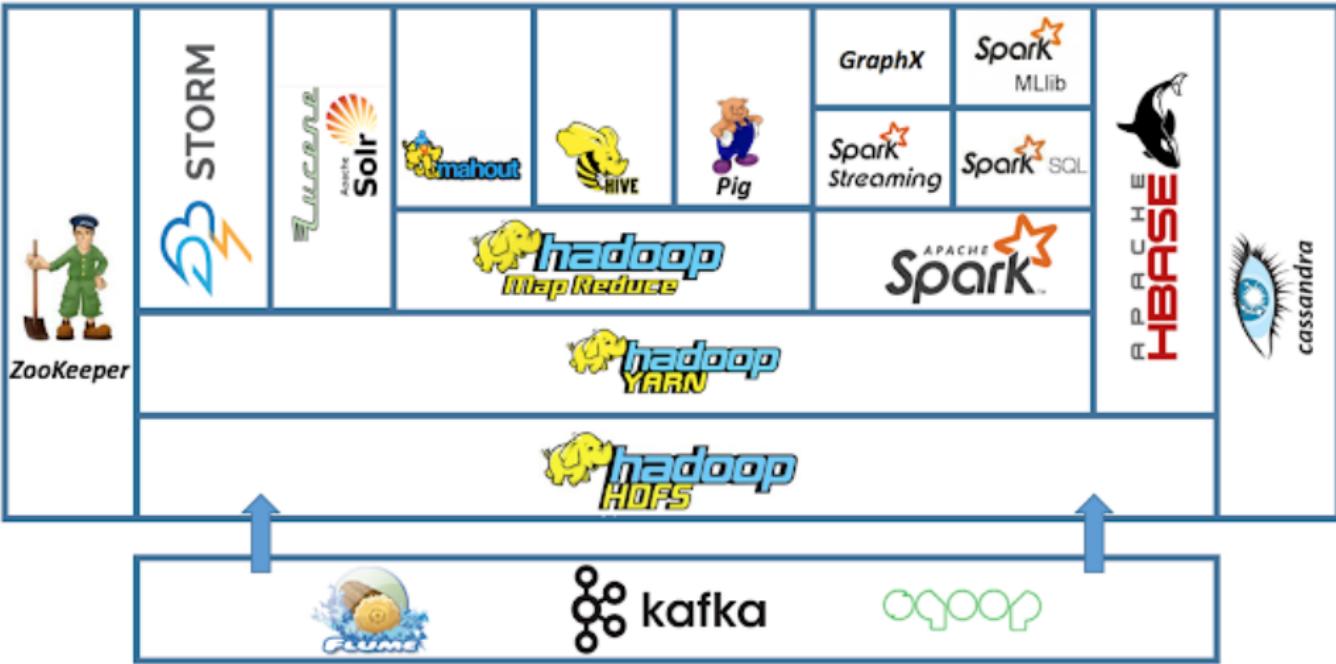


External Data Storage



Query Engines





Hadoop

Modulos do Hadoop

Hadoop Common: The common utilities that support the other Hadoop modules.

Hadoop Distributed File System (HDFS™): A distributed file system that provides high-throughput access to application data.

Hadoop YARN: A framework for job scheduling and cluster resource management.

<https://hadoop.apache.org/>



Hadoop

Modulos do Hadoop

Hadoop MapReduce: A YARN-based system for parallel processing of large data sets.

Hadoop Ozone: An object store for Hadoop.

Hadoop Submarine: A machine learning engine for Hadoop. <https://hadoop.apache.org/>



Índice

Índice

Material Utilizado no Curso

Introdução a Big Data

CAP

Hadoop

HDFS

Pig

Conclusão





Why DFS?

Read 1 TB Data



1 Machine

4 I/O Channels
Each Channel – 100 MB/s



10 Machines

4 I/O Channels
Each Channel – 100 MB/s



Slide 15

Why DFS?

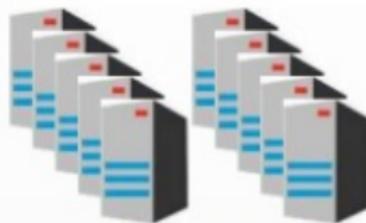
Read 1 TB Data



1 Machine

4 I/O Channels
Each Channel – 100 MB/s

45 Minutes



10 Machines

4 I/O Channels
Each Channel – 100 MB/s

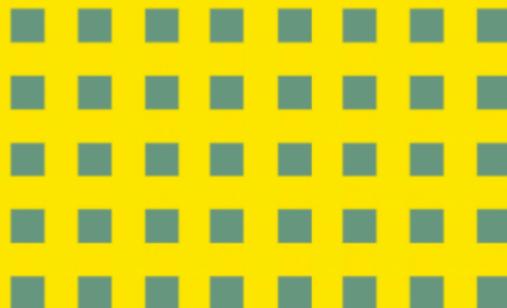
4.5 Minutes

Handles big files

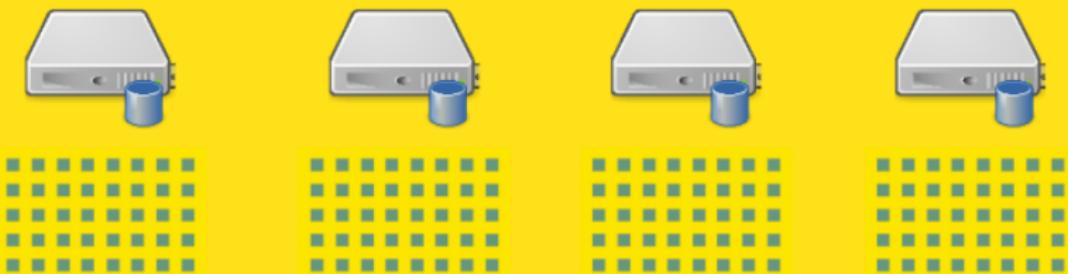


<https://www.udemy.com/course/the-ultimate-hands-on-hadoop-tame-your-big-data>

By breaking them into blocks



**Stored across several
commodity computers**



HDFS Architecture

Name Node



Data Node



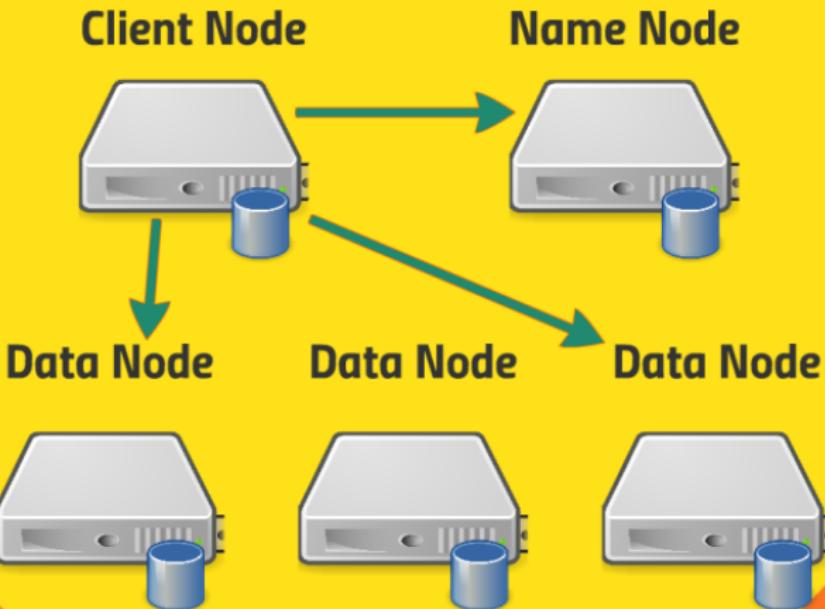
Data Node



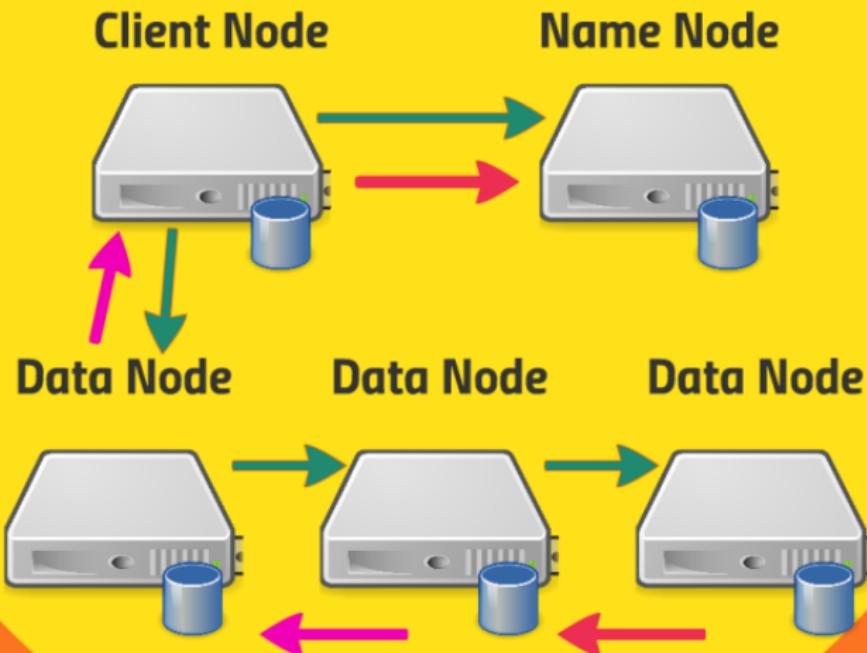
Data Node



Reading a File



Writing a File





Back Up Metadata

**Namenode writes to local
disk and NFS**

Secondary Namenode

**Maintains merged copy
of edit log you can
restore from**

HDFS Federation

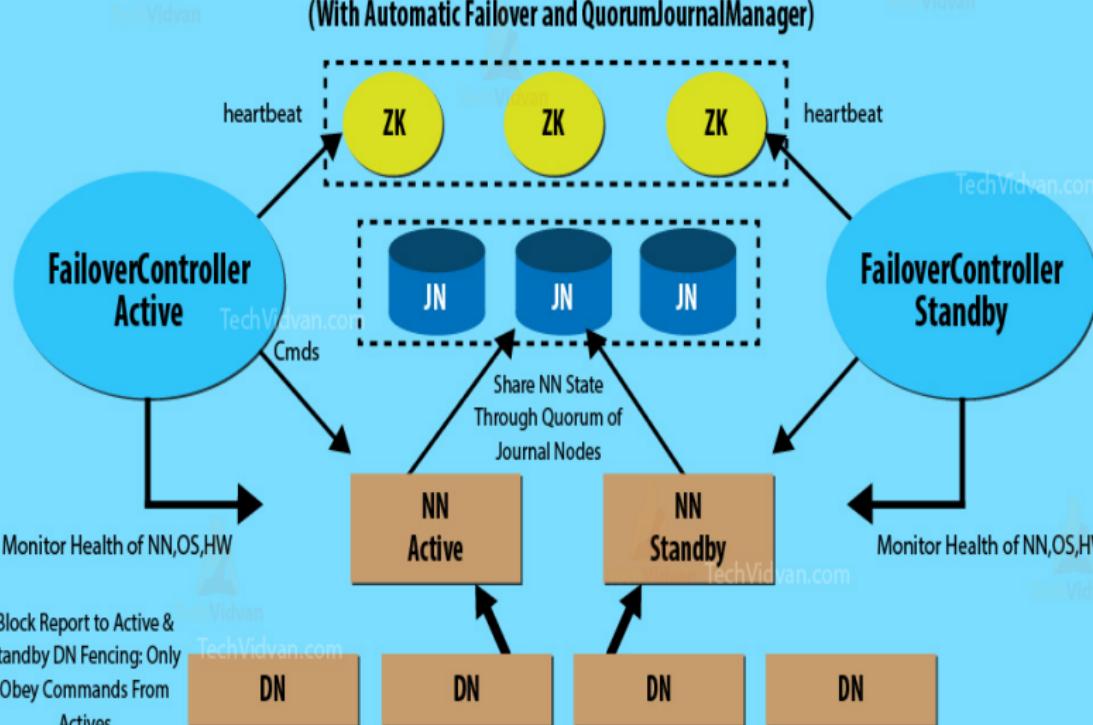
**Each namenode
manages a specific
namespace volume**

2. What is Hadoop HDFS High Availability?

Hadoop HDFS is a distributed file system. HDFS distributes data among the nodes in the Hadoop cluster by creating a replica of the file. Hadoop framework store these replicas of files on the other machines present in the cluster. So, when an HDFS client wants to access his data, he can easily access that data from a number of machines present in the cluster. Data is easily available in the closest node in the cluster. At some unfavorable conditions like a failure of a node, the client can easily access their data from the other nodes. This feature of Hadoop is called **High Availability**.



Hadoop High Availability



Using HDFS

UI (Ambari)

Command-Line Interface

HTTP / HDFS Proxies

Java interface

NFS Gateway

Importando MovieLens DataSet

Importando MovieLens

Vamos importar o dataset MovieLens no Hadoop

<https://grouplens.org/datasets/movielens/>

The screenshot shows the GroupLens website with a blue header bar containing the logo "grouplens" and navigation links for "about", "datasets", "publications", and "blog". The main content area has a white background. On the left, there's a section titled "MovieLens" with a brief description of the dataset collection process and a link to "take a short survey". Below this is a section titled "recommended for new research". On the right, there's a sidebar titled "Datasets" with a list of available datasets: "MovieLens" (which is highlighted in a blue box), "WikiLens", "Book-Crossing", "Jester", "EachMovie", "HetRec 2011", "Serendipity 2018", and "Personality 2018". At the bottom of the sidebar, there's a small "GROUPLENS" logo.

grouplens about datasets publications blog

MovieLens

GroupLens Research has collected and made available rating data sets from the MovieLens web site (<http://movielens.org>). The data sets were collected over various periods of time, depending on the size of the set. Before using these data sets, please review their README files for the usage licenses and other details.

Help our research lab: Please [take a short survey](#) about the MovieLens datasets

Seeking permission? If you are interested in obtaining permission to use MovieLens datasets, please first read the terms of use that are included in the README file. Then, [contact us](#), including a description of how you will (a) use the datasets, (b) redistribute the datasets, and (c) provide attribution. We typically do not permit public redistribution (see [Kaggle](#) for an alternative download location if you are concerned about availability).

recommended for new research

Datasets

MovieLens

WikiLens

Book-Crossing

Jester

EachMovie

HetRec 2011

Serendipity 2018

Personality 2018

//grouplens.org



Executando em um container Docker

```
docker run -m 4G --memory-reservation 2G  
--memory-swap 8G  
--hostname=quickstart.cloudera --privileged=true  
-t -i -v $(pwd):/zaid  
--publish-all=true -p8888 -p8088  
cloudera/quickstart  
/usr/bin/docker-quickstart
```



Importando MovieLens DataSet

Importando MovieLens

Entrar na url `http://localhost:8080/` e digitar o login `maria_dev` e senha `maria_dev`



Sign in

Username

Password

Sign in



- ✓ HDFS
- ✓ YARN
- ✓ MapReduce2
- Tez
- ✓ Hive
- HBase
- Pig
- Sqoop
- ✓ Oozie
- ✓ ZooKeeper
- Falcon
- Storm
- ✓ Flume
- Ambari Infra
- Atlas
- ✓ Kafka
- Knox

Metrics Heatmaps Config History

Metric Actions ▾

Last 1 hour ▾

HDFS Disk Usage



DataNodes Live

1/1

HDFS Links

NameNode

Secondary NameNode

1 DataNodes

More... ▾

Memory Usage

No Data Available

Network Usage

No Data Available

CPU Usage

No Data Available

Cluster Load

No Data Available

NameNode Heap



NameNode RPC

1.28 ms

NameNode CPU WIO

n/a

NameNode Uptime

17.7 min

HBase Master Heap

n/a

HBase Links

No Active Master

1 RegionServers

n/a

HBase Ave Load

n/a

HBase Master Uptime

n/a



- ✓ HDFS
 - ✓ YARN
 - ✓ MapReduce2
 - Tez
 - ✓ Hive
 - HBase
 - Pig
 - Sqoop
 - ✓ Oozie
 - ✓ ZooKeeper
 - Falcon
 - Storm
 - ✓ Flume
 - Ambari Infra
 - Atlas
 - ✓ Kafka
 - Knox
- 1:8080/#/main/dashboard

[Summary](#)[Heatmaps](#)[Configs](#)[Quick Links ▾](#)[Service Actions ▾](#)

Summary

No alerts[NameNode](#) ✓ Started No alerts

Disk Remaining 78.4 GB / 106.0 GB (73.97%)

[SNameNode](#) ✓ Started No alerts

Blocks (total) 1121

[DataNodes](#) 1/1 Started

Block Errors 0 corrupt replica / 0 missing / 0 under replicated

DataNodes Status 1 live / 0 dead / 0 decommissioning

Total Files + Directories 1352

[JournalNodes](#) 0/0 JournalNodes Live

Upgrade Status No pending upgrade

[NFSGateways](#) 0/0 Started

Safe Mode Status Not in safe mode

NameNode Uptime 19.57 mins

NameNode Heap 82.0 MB / 240.0 MB (34.2% used)

Disk Usage (DFS Used) 2.1 GB / 106.0 GB (1.95%)

Disk Usage (Non DFS Used) 25.5 GB / 106.0 GB (24.08%)

Metrics

Last 1 hour ▾

NameNode GC count

No Data Available

NameNode GC time

No Data Available

NN Connection Load

No Data Available

NameNode Heap

No Data Available

NameNode Host Load

No Data Available



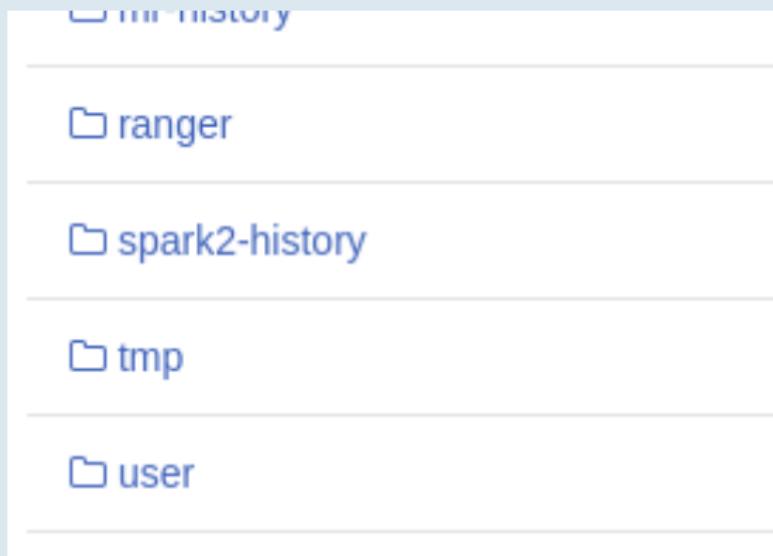
			Total: 11 files or folders			
Name	Size	Last Modified	Owner	Group	Permission	
app-logs	--	2018-06-18 12:18	yarn	hadoop	drwxrwxrwx	
apps	--	2018-06-18 13:13	hdfs	hdfs	drwxr-xr-x	
ats	--	2018-06-18 11:52	yarn	hadoop	drwxr-xr-x	
hdp	--	2018-06-18 11:52	hdfs	hdfs	drwxr-xr-x	
livy2-recovery	--	2018-06-18 12:11	livy	hdfs	drwx-----	
mapred	--	2018-06-18 11:52	mapred	hdfs	drwxr-xr-x	
mr-history	--	2018-06-18 11:52	mapred	hadoop	drwxrwxrwx	
ranger	--	2018-06-18 12:59	hdfs	hdfs	drwxr-xr-x	
spark2-history	--	2020-02-12 22:12	spark	hadoop	drwxrwxrwx	
27.0.0.1:8080/#/main/dashboard	--	2018-06-18 13:06	hdfs	hdfs	drwxrwxrwx	

Search in current directory...

Importando MovieLens DataSet

Importando MovieLens

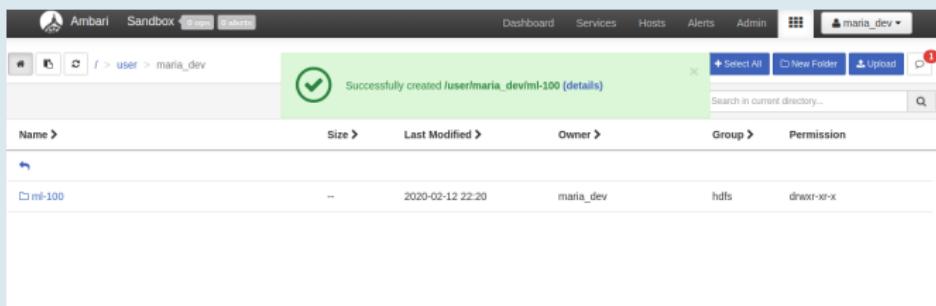
Selecione a pasta user e depois maria_dev



Importando MovieLens DataSet

Importando MovieLens

Crie o diretório e realize o upload do arquivo



The screenshot shows the Ambari Sandbox interface. At the top, there are tabs for Dashboard, Services, Hosts, Alerts, Admin, and a user dropdown for 'maria_dev'. Below the tabs, there's a breadcrumb navigation: 'Ambari' > 'Sandbox' > 'user' > 'maria_dev'. A green success message box displays the text 'Successfully created /user/maria_dev/ml-100 (details)'. To the right of the message are buttons for 'Select All', 'New Folder', 'Upload', and a trash bin icon. Below the message is a search bar with the placeholder 'Search in current directory...'. The main area is a table with columns: Name, Size, Last Modified, Owner, Group, and Permission. The table contains one row for the file 'ml-100', which has a size of '--', was last modified on '2020-02-12 22:20', belongs to 'maria_dev' as owner and 'hdfs' as group, and has permissions 'drwxr-xr-x'.

Name	Size	Last Modified	Owner	Group	Permission
ml-100	--	2020-02-12 22:20	maria_dev	hdfs	drwxr-xr-x



The screenshot shows a file browser interface with the following details:

Header:

- Left sidebar icons: Home, Back, Forward, Refresh, Stop.
- Path: / > user > maria_dev > ml-100
- Total: 1 files or folders
- Buttons: Select All, New Folder, Upload, Help (with a red dot).

Search bar: Search in current directory... with a magnifying glass icon.

Table Headers:

Name	Size	Last Modified	Owner	Group	Permission
------	------	---------------	-------	-------	------------

Data Row:

movies.csv	482.8 kB	2020-02-12 22:24	maria_dev	hdfs	-rW-r--r--
------------	----------	------------------	-----------	------	------------

Importando MovieLens DataSet

Importando MovieLens

Clique em open e veja o preview do arquivo

The screenshot shows a file browser window with the following details:

- Toolbar: Includes icons for file operations like Open, Save, and Copy.
- Path Bar: Shows the current path: / > user > maria_dev > ml-100.
- Status Bar: Displays "1 Files, 0 Folders selected".
- Left Panel: A sidebar with "Open" and "Rename" buttons.
- Table View: A list of files under "Name" column, showing "movies.csv" which is currently selected.
- Preview Area: Titled "File Preview" with the path "/user/maria_dev/ml-100/movies.csv". It displays the first 17 rows of the CSV file content.

Name
movies.csv

File Preview
/user/maria_dev/ml-100/movies.csv

```
movieId,title,genres
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
4,Waiting to Exhale (1995),Comedy|Drama|Romance
5,Father of the Bride Part II (1995),Comedy
6,Heat (1995),Action|Crime|Thriller
7,Sabrina (1995),Comedy|Romance
8,Tom and Huck (1995),Adventure|Children
9,Sudden Death (1995),Action
10,GoldenEye (1995),Action|Adventure|Thriller
11,"American President, The (1995)",Comedy|Drama|Romance
12,Dracula: Dead and Loving It (1995),Comedy|Horror
13,Balto (1995),Adventure|Animation|Children
14,Nixon (1995),Drama
15,Cutthroat Island (1995),Action|Adventure|Romance
16,Casino (1995),Crime|Drama
17,Sense and Sensibility (1995),Drama|Romance
```

Importando MovieLens DataSet

Importando MovieLens

Importando via linha de comando

```
Hortonworks Sandbox HDP 2.6.5 (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Hortonworks HDP Sandbox
https://hortonworks.com/products/sandbox

To quickly get started with the Hortonworks Sandbox, follow this tutorial:
https://hortonworks.com/tutorial/hadoop-tutorial-getting-started-with-hdp/

To initiate your Hortonworks Sandbox session, open a browser to this address:
For VirtualBox:
  Welcome screen: http://localhost:1080
  SSH: http://localhost:4200

For VMWare:
  Welcome screen: http://10.0.2.15:1080
  SSH: http://10.0.2.15:4200
```

Executando comandos no Hadoop

Executar o comando ssh e se conectando ao shell
ssh maria_dev@127.0.0.1 -p 2222



Executando comandos no Hadoop

Listando o diretório

```
hadoop fs -ls
```

```
.
maria_dev@127.0.0.1's password:
[maria_dev@sandbox-hdp ~]$ hadoop fs -ls
Found 2 items
drwxr-xr-x    - maria_dev hdfs          0 2020-02-13 01:24 ml-100
-rw-r--r--    1 maria_dev hdfs  494431 2020-02-13 01:23 movies.csv
[maria_dev@sandbox-hdp ~]$ █
```



Executando comandos no Hadoop

Criando o diretório

```
hadoop fs -mkdir ml-100-new
```

```
[maria_dev@sandbox-hdp ~]$ hadoop fs -mkdir ml-100-new
[maria_dev@sandbox-hdp ~]$
```



Executando comandos no Hadoop

Baixando arquivo de dados

```
wget http://media.sundog-soft.com/hadoop/ml-100k/u.data
```

Dica: veja o link <https://github.com/srafay/Hadoop-hands-on>



Executando comandos no Hadoop

Copiando para a pasta de destino

```
hadoop fs -copyFromLocal source Destination
```

Dica: hadoop fs -copyFromLocal u.data ml-100



Let's illustrate with an example

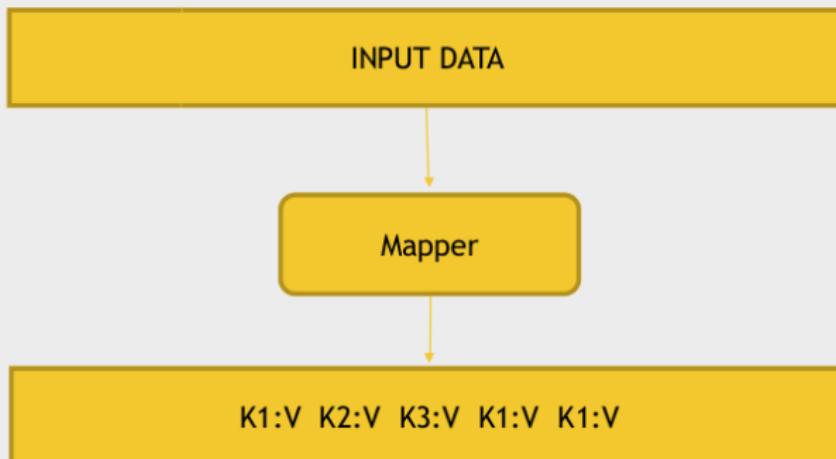
- How many movies did each user rate in the MovieLens data set?



<https://www.udemy.com/course/the-ultimate-hands-on-hadoop-tame-your-big-data>

How MapReduce Works: Mapping

- The MAPPER converts raw source data into **key/value** pairs



Example: MovieLens Data (u.data file)

USER ID	MOVIE ID	RATING	TIMESTAMP
196	242	3	881250949
186	302	3	891717742
196	377	1	878887116
244	51	2	880606923
166	346	1	886397596
186	474	4	884182806
186	265	2	881171488

Map users to movies they watched

USER ID | MOVIE ID | RATING | TIMESTAMP

196	242	3	881250949
186	302	3	891717742
196	377	1	878887116
244	51	2	880606923
166	346	1	886397596
186	474	4	884182806
186	265	2	881171488

Mapper

196:242 186:302 196:377 244:51 166:346 186:274 186:265

Extract and Organize What We Care About

196:242 186:302 196:377 244:51 166:346 186:474 186:265



<https://www.udemy.com/course/the-ultimate-hands-on-hadoop-tame-your-big-data>

The REDUCER Processes Each Key's Values

166:346 186:302,474,265 196:242,377 244:51



len(movies)



166:1 186:3 196:2 244:1

Putting it All Together

USER ID	MOVIE ID	RATING	TIMESTAMP
---------	----------	--------	-----------

196	242	3	881250949
186	302	3	891717742
196	377	1	878887116
244	51	2	880606923
166	346	1	886397596
186	474	4	884182806
186	265	2	881171488



MAPPER



196:242 186:302 196:377 244:51 166:346 186:474 186:265



SHUFFLE AND SORT



166:346 186:302,474,265 196:242,377 244:51

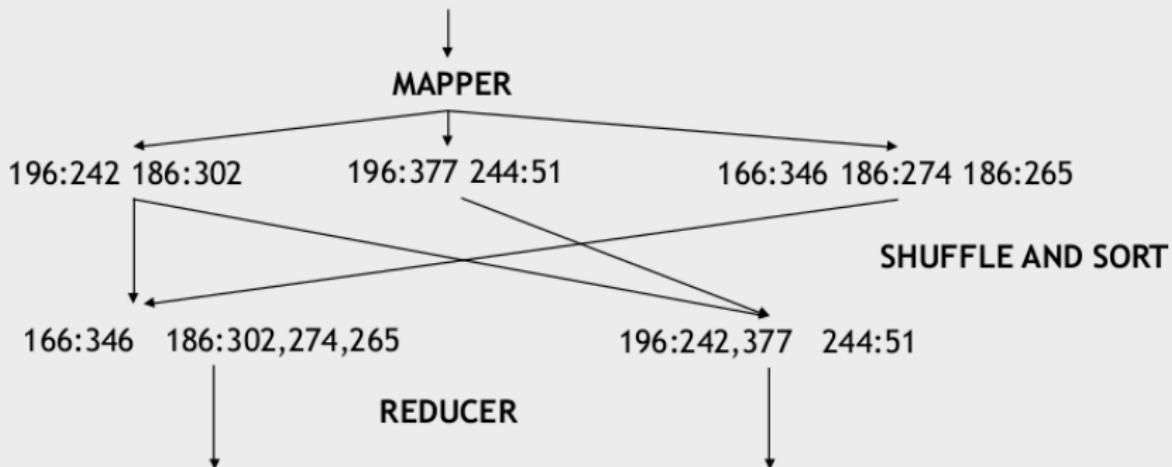
REDUCER

<https://www.udemy.com/course/the-ultimate-hands-on-hadoop-tame-your-big-data>

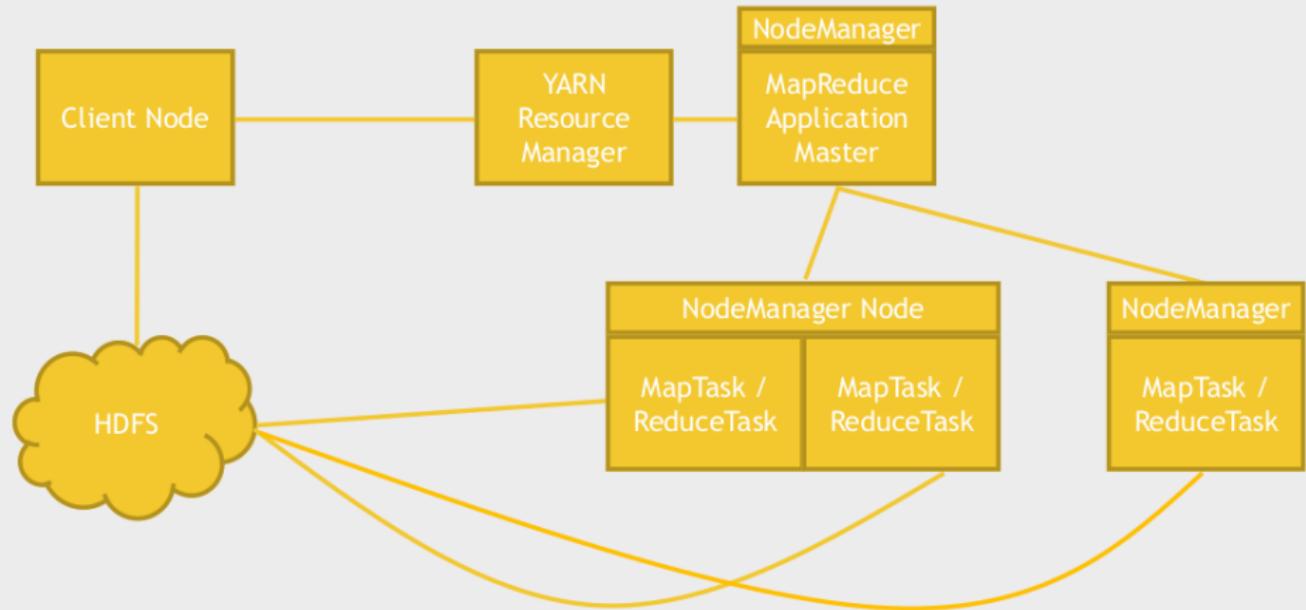
Putting it All Together

USER ID | MOVIE ID | RATING | TIMESTAMP

196	242	3	881250949
186	302	3	891717742
196	377	1	878887116
244	51	2	880606923
166	346	1	886397596
186	474	4	884182806
186	265	2	881171488

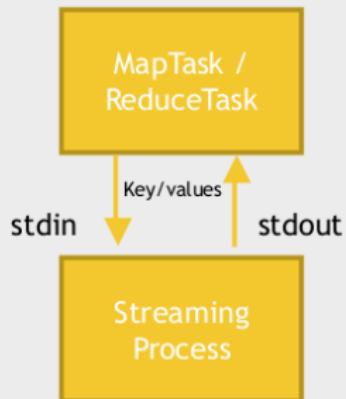


What's Happening



How are mappers and reducers written?

- MapReduce is natively Java
- STREAMING allows interfacing to other languages (ie Python)



<https://www.udemy.com/course/the-ultimate-hands-on-hadoop-tame-your-big-data>

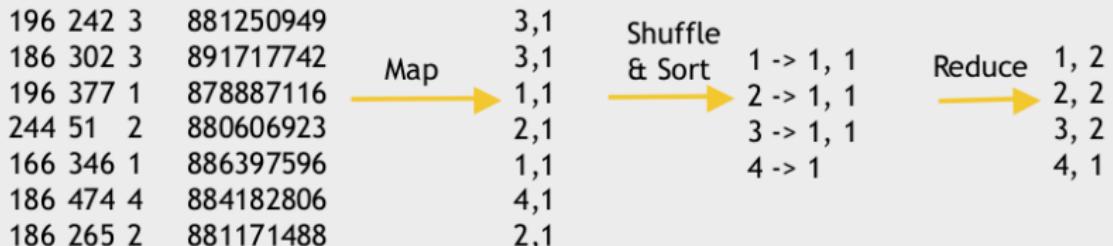
How many of each movie rating exist?



Making it a MapReduce problem

- MAP each input line to (rating, 1)
- REDUCE each rating with the sum of all the 1's

USER ID | MOVIE ID | RATING | TIMESTAMP



```
# RatingsBreakdown.py
from mrjob.job import MRJob
from mrjob.step import MRStep

class RatingsBreakdown(MRJob):

    def steps(self):
        return [
            MRStep( mapper=self.mapper_get_ratings,
                    reducer=self.reducer_count_ratings)
        ]

    def mapper_get_ratings(self, _, line):
        (userID, movieID, rating, timestamp) = line.split('\t')
        yield rating, 1

    def reducer_count_ratings(self, key, values):
        yield key, sum(values)

if __name__ == '__main__':
    RatingBreakdown.run()
```

Executando comandos no Hadoop

Instalando requisitos necessários

```
su root  
password is "hadoop"  
yum install python-pip
```

```
[root@sandbox-hdp maria_dev]# yum install python-pip  
Loaded plugins: fastestmirror, ovl  
HDP-2.6-repo-1 | 2.9 kB 00:00  
HDP-SOLR-2.6-100 | 2.9 kB 00:00  
HDP-UTILS-1.1.0.22-repo-1 | 2.9 kB 00:00  
ambari-2.6.2.0 | 2.9 kB 00:00  
base | 3.6 kB 00:00  
epel/x86_64/metalink | 54 kB 00:00  
epel | 5.3 kB 00:00  
extras | 2.9 kB 00:00
```

Dica: hadoop fs -copyFromLocal u.data ml-100



- If you have **HDP 2.6.5**, follow these instructions
 - `yum install python-pip`
 - `pip install mrjob==0.5.11`
 - `yum install nano`
 - `wget http://media.sundog-soft.com/hadoop/ml-100k/u.data`
 - `wget http://media.sundog-soft.com/hadoop/RatingsBreakdown.py`
- If you have **HDP 2.5**, follow these instructions
 - `cd /etc/yum.repos.d`
 - `cd sandbox.repo /tmp`
 - `rm sandbox.repo`
 - `cd ~`
 - `yum install python-pip`
 - `pip install google-api-python-client==1.6.4`
 - `pip install mrjob==0.5.11`
 - `yum install nano`
 - `wget http://media.sundog-soft.com/hadoop/ml-100k/u.data`
 - `wget http://media.sundog-soft.com/hadoop/RatingsBreakdown.py`
- <https://github.com/srafay/Hadoop-hands-on>

Executando comandos no Hadoop

Executando Map Reduce localmente

```
python RatingsBreakdown.py u.data
```

<https://github.com/srafay/Hadoop-hands-on>



Executando comandos no Hadoop

Executando Map Reduce no cluster

```
python RatingsBreakdown.py -r hadoop  
--hadoop-streaming-jar  
/usr/hdp/current/hadoop-mapreduce-client/  
hadoop-streaming.jar u.data
```

–hadoop-streaming-jar is for telling mrjob where to find the jar file for hadoop streaming

here the file (u.data) is on our machine thus we can access it directly if it's a big file, it will most probably be on the HDFS thus you will give the link of the file on your HDFS system as hdfs://filepath



④ A Simple Map Reduce Job

Code for this example and more live in `mrjob/examples`.

```
"""The classic MapReduce job: count the frequency of words.
"""

from mrjob.job import MRJob
import re

WORD_RE = re.compile(r"[\w']+")

class MRWordFreqCount(MRJob):

    def mapper(self, _, line):
        for word in WORD_RE.findall(line):
            yield (word.lower(), 1)

    def combiner(self, word, counts):
        yield (word, sum(counts))

    def reducer(self, word, counts):
        yield (word, sum(counts))

if __name__ == '__main__':
    MRWordFreqCount.run()
```

<https://github.com/Yelp/mrjob>



Pig

O que é o Pig

[https://www.linkedin.com/learning/search?
keywords=pig](https://www.linkedin.com/learning/search?keywords=pig)

What Is Pig?

- ETL library for Hadoop
 - Generates MapReduce jobs
 - Developed at Yahoo!
- Uses the Pig Latin language

Pig

O que é o Pig

[https://www.linkedin.com/learning/search?
keywords=pig](https://www.linkedin.com/learning/search?keywords=pig)

When Do You Use Pig?

- For ETL-like jobs

Transform data

Clean data

Process data

Pig

O que é o Pig

[https://www.linkedin.com/learning/search?
keywords=pig](https://www.linkedin.com/learning/search?keywords=pig)

How Does Pig Work?

- ETL process and flow

LOAD <file>

FILTER, JOIN, GROUP BY, FOREACH, GENERATE
<values>

DUMP <to screen for testing>

STORE <new file>

Pig

O que é o Pig

[https://www.linkedin.com/learning/search?
keywords=pig](https://www.linkedin.com/learning/search?keywords=pig)

Pig Concepts: Data

- Field: a piece of data
- Tuple: a set of fields
- Bag: a collection of tuples



Running Pig

- Grunt
- Script
- Ambari / Hue



An example

- Find the oldest 5-star movies



```
ratings = LOAD '/user/maria_dev/ml-100k/u.data' AS  
    (userID:int, movieID:int, rating:int, ratingTime:int);
```

This creates a *relation* named “ratings” with a given *schema*.

- (660,229,2,891406212)
- (421,498,4,892241344)
- (495,1091,4,888637503)
- (806,421,4,882388897)
- (676,538,4,892685437)
- (721,262,3,877137285)

Use PigStorage if you need a different delimiter.

```
metadata = LOAD '/user/maria_dev/ml-100k/u.item' USING  
    PigStorage('|')AS (movieID:int, movieTitle:chararray,  
    releaseDate:chararray, videoRelease:chararray,  
    imdbLink:chararray);  
  
DUMP metadata;
```

```
(1,Toy Story (1995),01-Jan-1995,,http://us.imdb.com/M/title-exact?Toy%20Story%20(1995))  
(2,GoldenEye (1995),01-Jan-1995,,http://us.imdb.com/M/title-exact?GoldenEye%20(1995))  
(3,Four Rooms (1995),01-Jan-1995,,http://us.imdb.com/M/title-exact?Four%20Rooms%20(1995))  
(4,Get Shorty (1995),01-Jan-1995,,http://us.imdb.com/M/title-exact?Get%20Shorty%20(1995))  
(5,Copycat (1995),01-Jan-1995,,http://us.imdb.com/M/title-exact?Copycat%20(1995))
```

Creating a relation from another relation; FOREACH / GENERATE

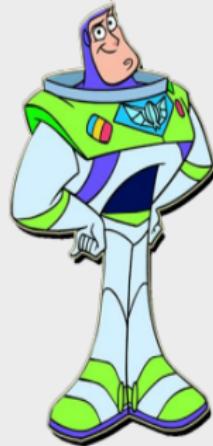
```
metadata = LOAD '/user/maria_dev/ml-100k/u.item' USING PigStorage('|')
AS (movieID:int, movieTitle:chararray, releaseDate:chararray,
videoRelease:chararray, imdbLink:chararray);

nameLookup = FOREACH metadata GENERATE movieID, movieTitle,
ToUnixTime(ToDate(releaseDate, 'dd-MMM-yyyy')) AS releaseTime;
```

(1,Toy Story (1995),01-Jan-1995,,http://us.imdb.com/M/title-exact?Toy%20Story%20(1995))



(1,Toy Story (1995),788918400)



Group By

```
ratingsByMovie = GROUP ratings BY movieID;  
DUMP ratingsByMovie;
```

```
(1,{(807,1,4,892528231),(554,1,3,876231938),(49,1,2,888068651), ... }  
(2,{(429,2,3,882387599),(551,2,2,892784780),(774,2,1,888557383), ... }
```

```
avgRatings = FOREACH ratingsByMovie GENERATE group AS movieID,  
          AVG(ratings.rating) AS avgRating;  
  
DUMP avgRatings;
```

```
(1,3.8783185840707963)  
(2,3.2061068702290076)  
(3,3.033333333333333)  
(4,3.550239234449761)  
(5,3.302325581395349)
```

```
DESCRIBE ratings;  
DESCRIBE ratingsByMovie;  
DESCRIBE avgRatings;
```

```
ratings: {userID: int,movieID: int,rating: int,ratingTime: int}
```

```
ratingsByMovie: {group: int,ratings: {[userID: int,movieID: int,rating: int,ratingTime: int]}}
```

```
avgRatings: {movieID: int,avgRating: double}
```

FILTER

```
fiveStarMovies = FILTER avgRatings BY avgRating > 4.0;
```

```
(12,4.385767790262173)  
(22,4.151515151515151)  
(23,4.1208791208791204)  
(45,4.05)
```

JOIN

```
DESCRIBE fiveStarMovies;  
DESCRIBE nameLookup;
```

```
fiveStarsWithData = JOIN fiveStarMovies BY movieID, nameLookup BY movieID;
```

```
DESCRIBE fiveStarsWithData;
```

```
DUMP fiveStarsWithData;
```

```
fiveStarMovies: {movieID: int, avgRating: double}
```

```
nameLookup: {movieID: int, movieTitle: chararray, releaseTime: long}
```

```
fiveStarsWithData: {fiveStarMovies::movieID: int, fiveStarMovies::avgRating: double,
```

```
nameLookup::movieID: int, nameLookup::movieTitle: chararray, nameLookup::releaseTime: long}
```

```
(12, 4.385767790262173, 12, Usual Suspects, The (1995), 808358400)
```

```
(22, 4.151515151515151, 22, Braveheart (1995), 824428800)
```

```
(23, 4.1208791208791204, 23, Taxi Driver (1976), 824428800)
```

ORDER BY

```
oldestFiveStarMovies = ORDER fiveStarsWithData BY  
    nameLookup::releaseTime;  
  
DUMP oldestFiveStarMovies;
```

```
(493,4.15,493,Thin Man, The (1934),-1136073600)  
(604,4.012345679012346,604,It Happened One Night (1934),-1136073600)  
(615,4.0508474576271185,615,39 Steps, The (1935),-1104537600)  
(1203,4.0476190476190474,1203,Top Hat (1935),-1104537600)
```





Scripts

New Script



Name

Script HDFS Location (optional)

Leave empty to create file automatically.

Actions

[History](#) [Copy](#) [Delete](#)

Show: 10 ▾ 1 - 1 of 1

Test

 Execute on Tez

Execute



PIG helper ▾

UDF helper ▾

/user/maria_dev/pig/scripts/test-2020-02-15_05-39.pig

```
1 ratings = LOAD '/user/maria_dev/ml-100k/u.data' AS (userID:int, movieID:int, rating:int, ratingTime:int);
2 metadata = LOAD '/user/maria_dev/ml-100k/u.item' USING PigStorage('|')
3   AS (movieID:int, movieTitle:chararray, releaseDate:chararray, videoRealese:chararray,
4       imdblink:chararray);
5 nameLookup = FOREACH metadata GENERATE movieID, movieTitle,
6   ToUnixTime(ToDate(releaseDate, 'dd-MMM-yyyy')) AS releaseTime;
7
8 ratingsByMovie = GROUP ratings BY movieID;
9 avgRatings = FOREACH ratingsByMovie GENERATE group as movieID, AVG(ratings.rating) as avgRating;
10 fiveStarMovies = FILTER avgRatings BY avgRating > 4.0;
11 fiveStarsWithData = JOIN fiveStarMovies BY movieID, nameLookup BY movieID;
12 oldestFiveStarMovies = ORDER fiveStarsWithData BY nameLookup::releaseTime;
13 DUMP oldestFiveStarMovies;
```

How Do You Run Pig Scripts?

- Script/Batch mode: run from the Hadoop shell
- Grunt/Interactive mode: start with the Pig shell
- Embedded mode: within Java

Conclusão

Você está apto a:

- Compreender o conceito de Hadoop
- Aplicar o conceito de MapReduce
- Utilizar a ferramenta PIG



Bibliografia I

