



ARA0168

TÓPICOS DE BIG DATA

EM PYTHON

Aula 2 – Ecossistema Hadoop

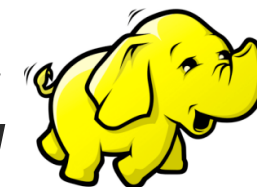
Universidade Estácio de Sá

Prof. Simone Gama

simone.gama@estacio.br



They say an elephant never forgets. Well, you are not an elephant. Take notes, constantly. Save interesting thoughts, quotations, films, technologies... the medium doesn't matter, so long as it inspires you.



Aaron Koblin



BIG DATA: Estruturação de Dados



Dados Estruturados

Os **dados estruturados** são aqueles **organizados** e representados com uma **estrutura rígida**, a qual foi previamente planejada para armazená-los, por exemplo um banco de dados, que é a representação mais típica e comum de dados estruturados.

Em um banco de dados, os dados são estruturados conforme a definição de um esquema, que define como as tabelas e suas respectivas linhas e colunas serão armazenadas. Podemos conceituar o esquema de um banco de dados como sendo uma descrição sobre uma organização, ou sobre o minimundo que se deseja representar, definindo quais dados que serão armazenados.



BIG DATA: Estruturação de Dados



Dados Não Estruturados

Os **dados não-estruturados**, que possuem uma estrutura totalmente inversa dos dados estruturados, sendo **flexíveis** e **dinâmicos** ou, até mesmo, sem **qualquer estrutura**.

Pense em um editor de texto no Word, adicionando quanto textos, sem qualquer preocupação com campos, restrições ou limites, também podemos adicionar imagens, gráficos e fotos, misturando com os textos que já escrevemos, temos nesse cenário um exemplo de dados não-estruturados. Redes sociais são outro exemplo de dados não estruturados.



BIG DATA: Estruturação de Dados



Dados Semi Estruturados

Não possuem estrutura **totalmente rígida** nem estrutura **totalmente flexível**, sendo uma representação heterogênea. Um exemplo típico seria um arquivo em XML (*eXtensible Markup Language*, que significa em português, Linguagem de Marcação Estendida), o qual possui nós, que são rótulas de abertura e fechamento, precedidos de um símbolo “/”, com os dados inseridos entre os nós.





ARA0168

TÓPICOS DE BIG DATA

EM PYTHON

2.1 – Introdução ao Ecossistema Hadoop

Universidade Estácio de Sá

Prof. Simone Gama

simone.gama@estacio.br

BIG DATA: Motores de Busca



Conforme a World Wide Web crescia no final dos anos 1990 e início dos anos 2000, **mecanismos de busca** e **indexadores** foram criados para ajudar a localizar informações relevantes em meio a conteúdos textuais.

No começo, eram os próprios seres humanos quem devolviam os resultados de buscas; mas, à medida que a internet cresceu de dezenas para milhares de páginas, a **automação** se tornou necessária. Rastreadores web foram criados, muitos como projetos de pesquisa liderados por universidades, e *startups* de mecanismos de busca decolaram (Yahoo, AltaVista etc).



BIG DATA: Motores de Busca



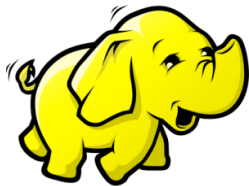
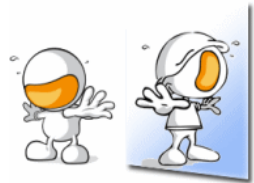
Um desses projetos era um mecanismo de busca *open-source* chamado **Nutch** – idealizado por Doug Cutting e Mike Cafarella. O objetivo era retornar resultados mais rapidamente ao **distribuir dados e cálculos entre computadores distintos** para que diferentes tarefas pudessem ser realizadas simultaneamente.



Durante esse tempo, outro mecanismo de busca chamado **Google** estava em construção. Ele era baseado no mesmo conceito – **armazenar e processar dados de forma distribuída e automatizada** para que resultados de pesquisa relevantes pudessem ser encontrados rapidamente.



BIG DATA: Motores de Busca



Em 2006, Cutting foi contratado pelo **Yahoo** e levou com ele o projeto **Nutch**, bem como ideias baseadas nos trabalhos iniciais do Google de **automatizar o armazenamento e o processamento de dados de modo distribuído**.

O projeto **Nutch** foi dividido – o rastreador web permaneceu como **Nutch** e a **parte de processamento e computação distribuída tornou-se o Hadoop** (nome do elefantinho de brinquedo do filho do Cutting).



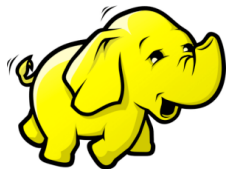
BIG DATA: **Ecosystema Hadoop**



Em 2008, o Yahoo lançou o **Hadoop** como um projeto *open-source*.

Hoje, a estrutura e o ecossistema de tecnologias **Hadoop** são gerenciados e mantidos pela organização sem fins lucrativos **Apache Software Foundation** (ASF), uma comunidade global de desenvolvedores de software e colaboradores.





Ecossistema *Hadoop*

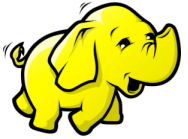


Um **ecossistema de software** (**ECOS**) é um conjunto de atores e artefatos, internos e externos a uma organização ou comunidade, que trocam recursos e informações centrados em uma plataforma tecnológica comum¹.

Este contexto tem **afetado decisões de gerenciamento e desenvolvimento** de tais plataformas, notadamente sobre modelos de arquitetura, de governança e de colaboração, nos mais variados domínios de aplicação.



¹Fonte: Simpósio Brasileiro de Sistemas Multimídia e Web ([98-1 \(sbc.org.br\)](http://98-1.sbc.org.br))

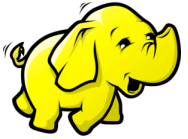


Ecossistema Hadoop - Importância



- **Capacidade de armazenar e processar grandes quantidades de qualquer tipo de dado, e rapidamente.** Com os volumes e tipos de dados disponíveis crescendo constantemente, graças, principalmente, às mídias sociais e à Internet das Coisas (*IoT*), isso é uma consideração importante.
- **Poder computacional.** O modelo computacional distribuído do Hadoop processa big data rapidamente. Quanto maior a quantidade de nós computacionais você usar, mais poder de processamento você terá.
- **Custo baixo.** A estrutura *open-source* é gratuita e utiliza hardwares comuns para armazenar grandes quantidades de dados.



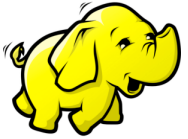


Ecossistema Hadoop - Importância



- **Tolerância a falhas.** O processamento de dados e aplicações é protegido contra falhas de hardware. Se um nó cai, os trabalhos são automaticamente redirecionados para outros nós para garantir que a computação distribuída não falhe. Múltiplas cópias de todos os dados são armazenadas automaticamente.
- **Flexibilidade.** Ao contrário dos bancos de dados relacionais tradicionais, você não precisa pré-processar os dados antes de armazená-los. Você pode armazenar seus dados o quanto quiser e decidir como usá-los depois. Isso inclui dados não-estruturados como texto, imagens e vídeos.
- **Escalabilidade.** Você pode aumentar facilmente o seu sistema para lidar com mais dados ao adicionar nós. Não é preciso muita administração.





Ecossistema Hadoop - Importância



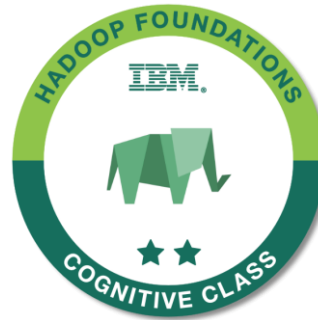
Além da distribuição livre da Apache, o Hadoop possui outras distribuições, como:



**amazon
EMR**

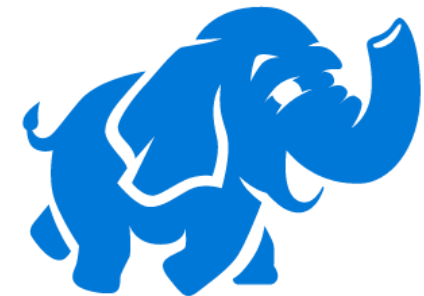
Amazon

Plataforma de big data – Amazon EMR –
Amazon Web Services



IBM Hadoop

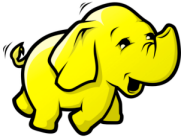
Apache Hadoop | IBM



Azure Hadoop

Azure HDInsight - Hadoop, Spark,
and Kafka | Microsoft Azure





Ecossistema Hadoop - Importância



Além da distribuição livre da Apache, o Hadoop possui outras distribuições, como:



Cloudera

[Cloudera | The Hybrid Data Company](https://www.cloudera.com/)



MapR

[HPE Ezmeral Data Fabric Software | HPE](https://www.mapr.com/)



Pivotal HD

[Tanzu Cloud Native Application Platform for Multi-cloud | VMware Tanzu](https://www.pivotal.io/tanzu)





Estácio



ARA0168

TÓPICOS DE BIG DATA

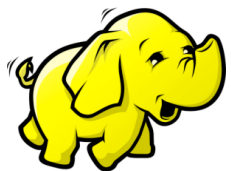
EM PYTHON

2.2 – Ecossistema *Hadoop*: **Composição e Arquitetura**

Universidade Estácio de Sá





Prof. Simone Gama

simone.gama@estacio.br

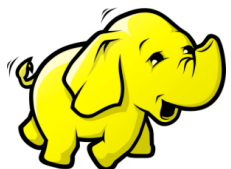


Ecossistema *Hadoop* – *Stack* de App's



CAMADA DE VISUALIZAÇÃO E API'S				RESTful API
FERRAMENTAS DE ANALÍTICA				
CAMADA DE ANÁLISE				 
CAMADA DE PROCESSAMENTO				
CAMADA DE ARMAZENAMENTO				
MOTOR DE EVENTOS				
CAMADA DE INTEGRAÇÃO (ETL)				
FONTES DE INFORMAÇÃO				

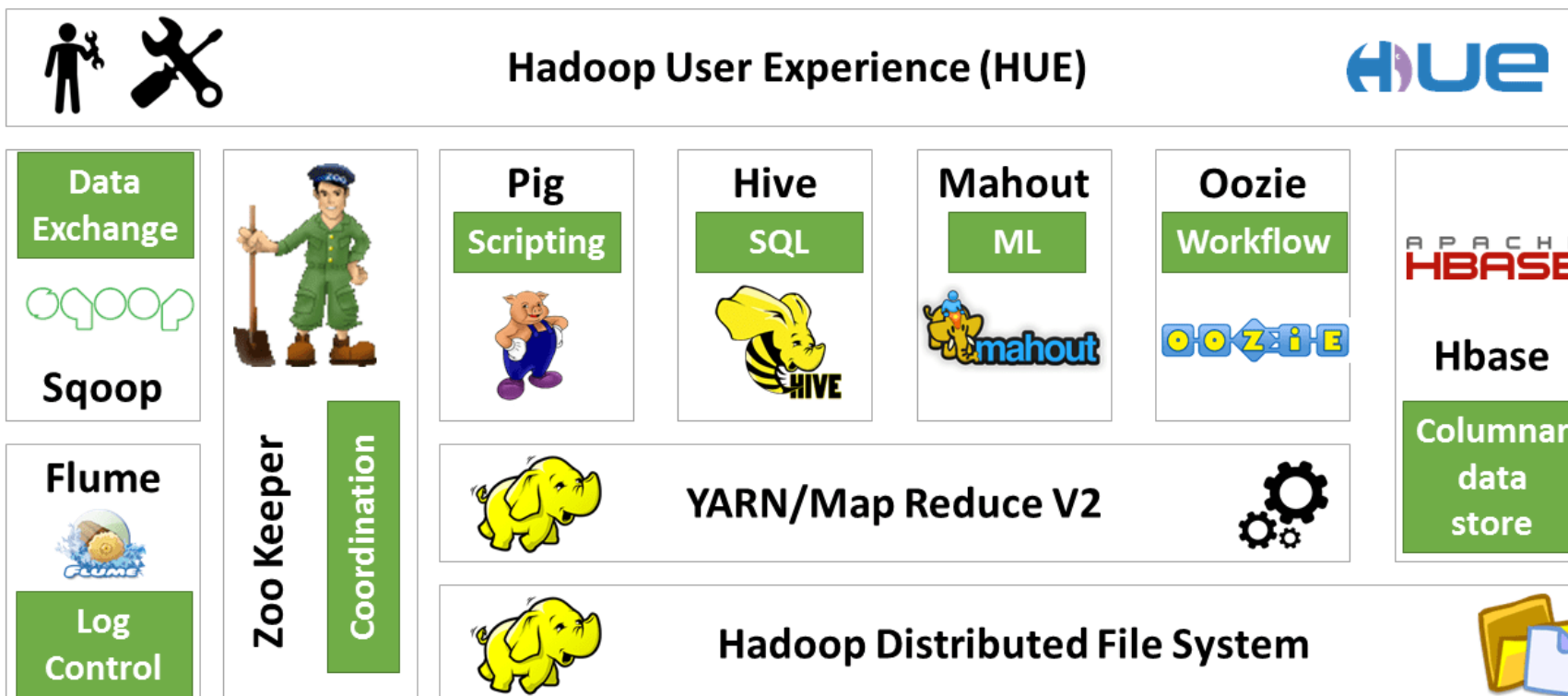


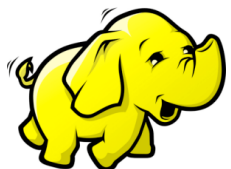


Ecossistema *Hadoop* – *Stack de App's*

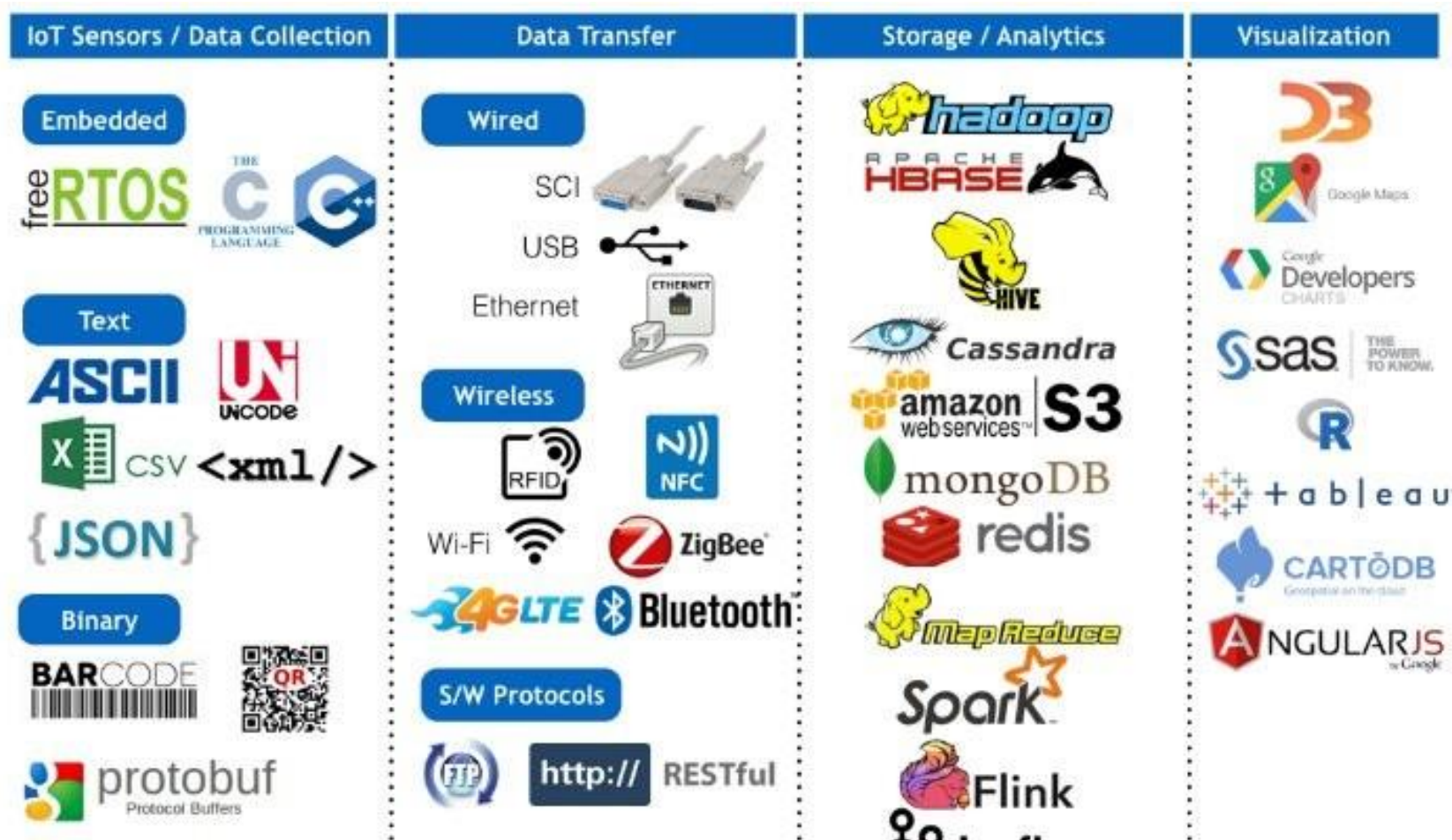


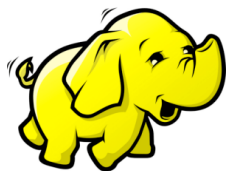
The Apache Hadoop Stack





Ecossistema *Hadoop* – *Stack* de App's

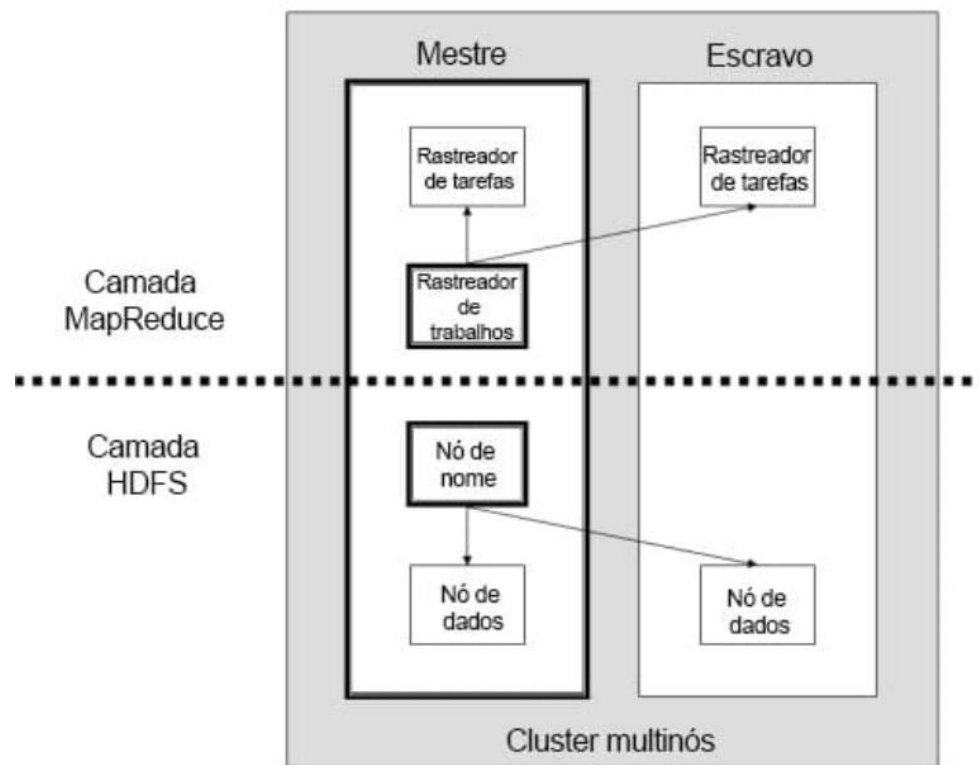


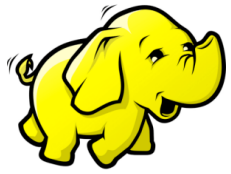


Ecossistema *Hadoop*: **Arquitetura**



O “*core*” do *Hadoop*





Ecossistema *Hadoop*: **HDFS**

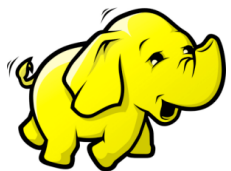


HDFS (**Sistema de Arquivos Distribuídos Hadoop**)

O sistema de arquivos distribuídos do Hadoop, mais conhecido como **HDFS** (*Hadoop Distributed File System*), é responsável pelo armazenamento de dados em um *cluster* do Hadoop.

O HDFS foi projetado para trabalhar com grandes volumes de dados em hardware comum, isto é, em dispositivos baratos, como computadores pessoais.





Ecossistema *Hadoop*: **HDFS**

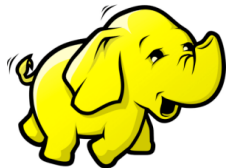


HDFS (**Sistema de Arquivos Distribuídos Hadoop**)

O sistema de arquivos distribuídos do Hadoop, mais conhecido como **HDFS** (***Hadoop Distributed File System***), é **responsável pelo armazenamento de dados em um *cluster* do Hadoop.**

O HDFS foi projetado para trabalhar com grandes volumes de dados em hardware comum, isto é, em dispositivos baratos, como computadores pessoais.



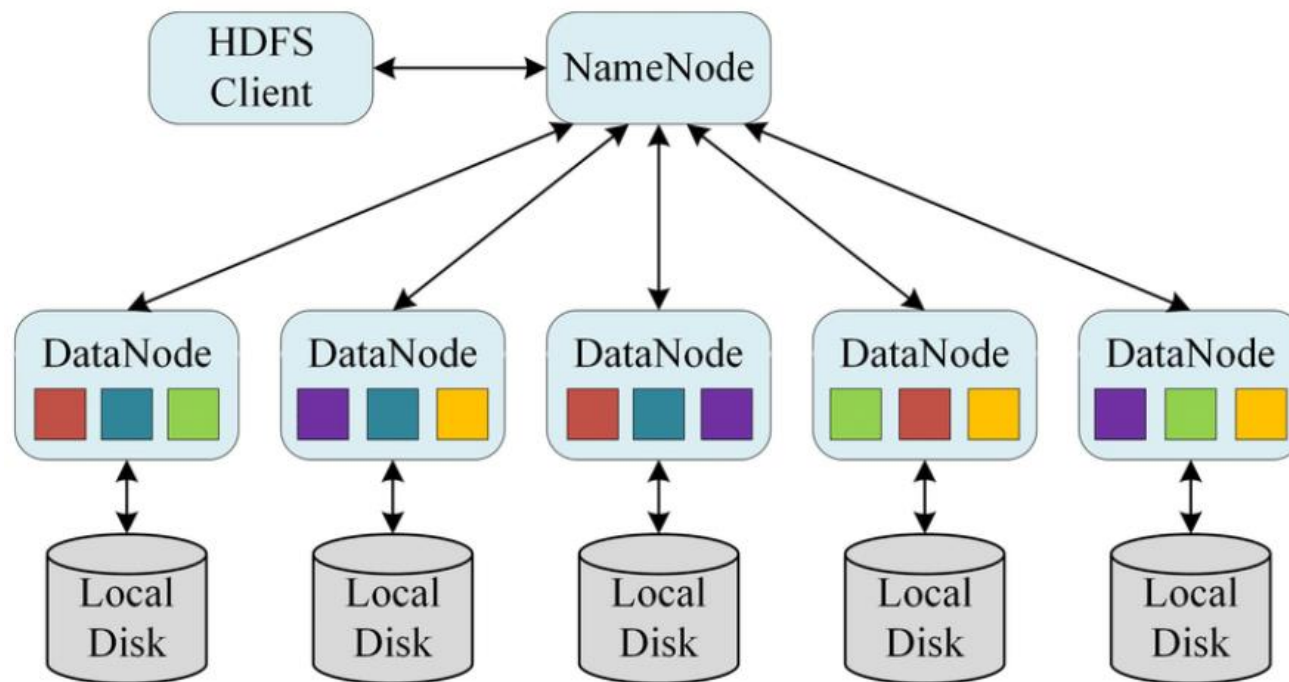


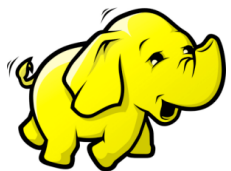
Ecossistema *Hadoop*: **HDFS**



HDFS (**Sistema de Arquivos Distribuídos Hadoop**)

A arquitetura do HDFS, por meio dos componentes **NameNode** e **DataNode**, utiliza um sistema de arquivos distribuídos que proporciona alto desempenho no acesso aos dados em clusters Hadoop.



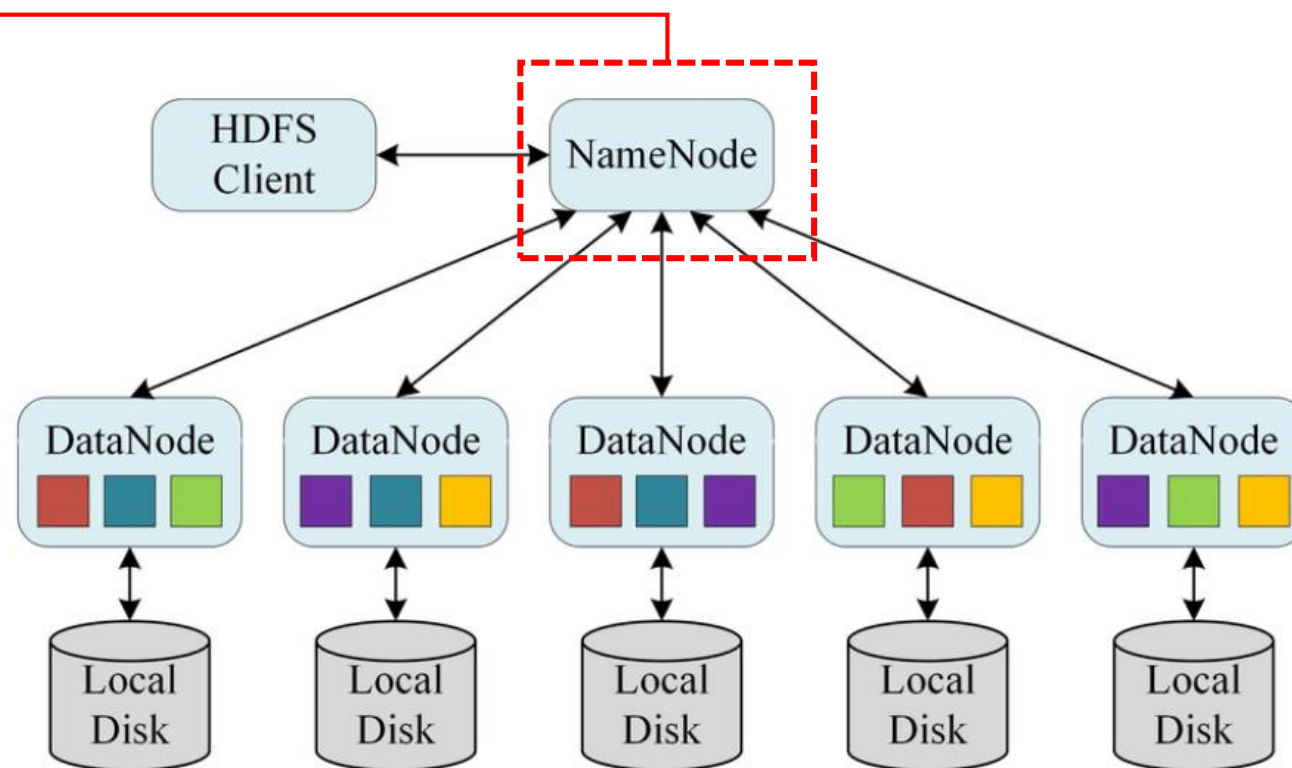


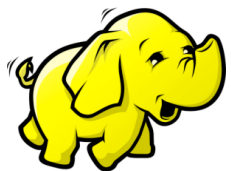
Ecossistema *Hadoop*: **HDFS**



HDFS (**Sistema de Arquivos Distribuídos Hadoop**)

O **NameNode** gerencia o *namespace* do sistema de arquivos. Isto mantém a árvore do sistema de arquivos e os metadados de todos os arquivos e diretórios na árvore.



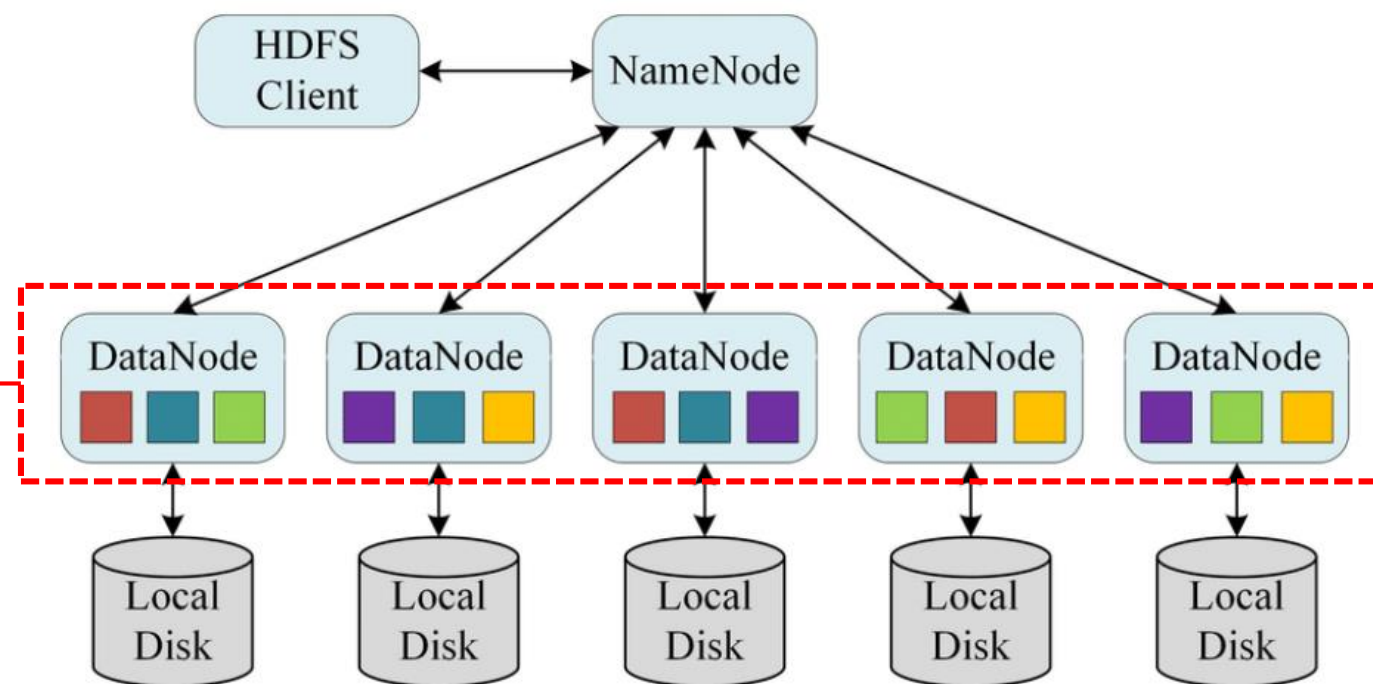


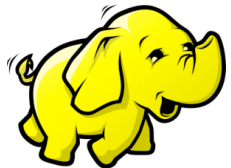
Ecossistema *Hadoop*: **HDFS**



HDFS (**Sistema de Arquivos Distribuídos Hadoop**)

Os **DataNodes** são os trabalhadores do sistema de arquivos. O *DataNode* realiza a **criação, exclusão e cópia** do bloco sob o comando do *NameNode*.



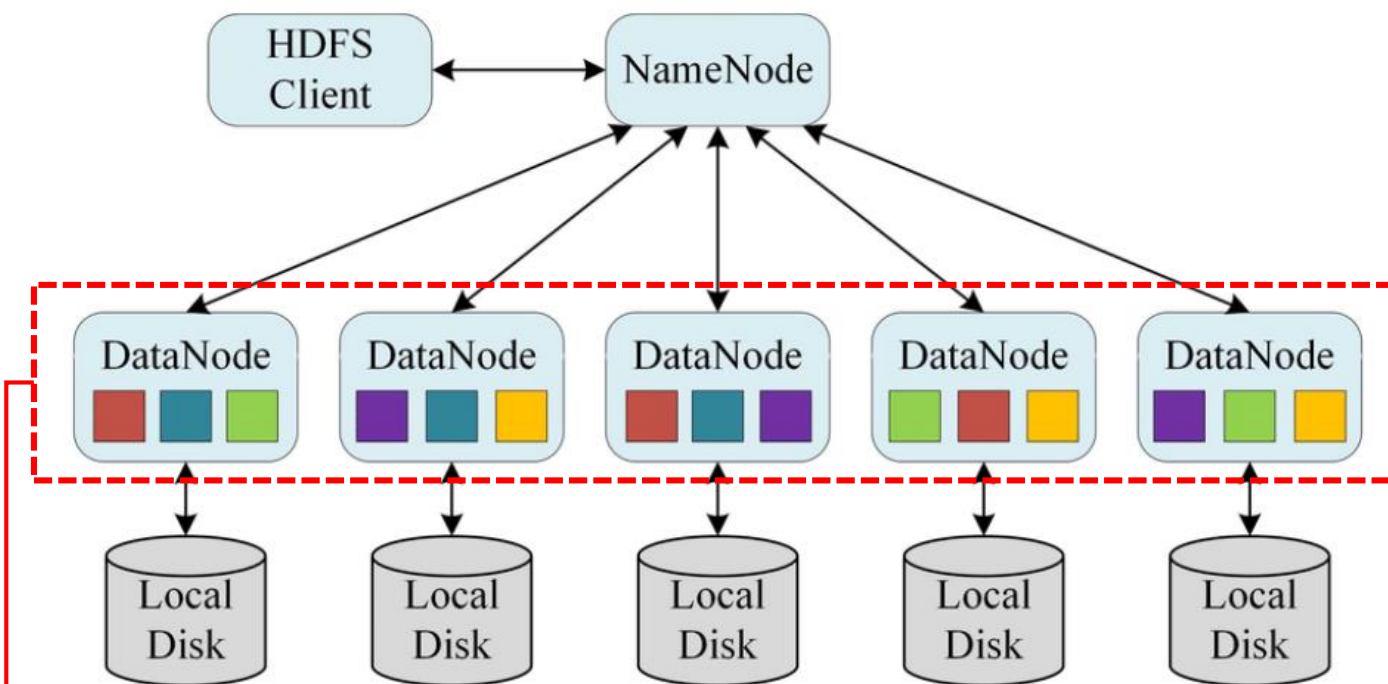


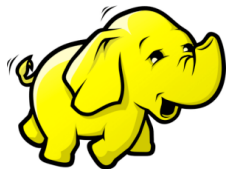
Ecossistema *Hadoop*: **HDFS**



HDFS (**Sistema de Arquivos Distribuídos Hadoop**)

Eles armazenam e recuperam blocos quando solicitados (pelos clientes ou pelo *NameNode*) e reportam ao *NameNode* periodicamente com listas de blocos que estão armazenando.





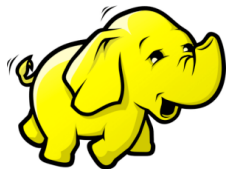
Ecossistema *Hadoop* : MapReduce



MapReduce (Mapear e Reduzir)

Para manejar grandes volumes de dados e extrair o máximo do *big data*, o Hadoop conta com um algoritmo, também introduzido pelo Google, chamado **MapReduce** que facilita a distribuição e execução de uma tarefa paralelamente no *cluster*. Sua missão é basicamente dividir uma tarefa em várias e processa em máquinas diferentes.





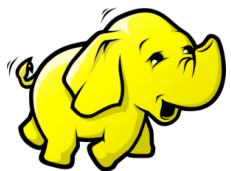
Ecossistema *Hadoop* : MapReduce



MapReduce (Mapear e Reduzir)

Para manejar grandes volumes de dados e extrair o máximo do *big data*, o Hadoop conta com um algoritmo, também introduzido pelo Google, chamado ***MapReduce*** que facilita a distribuição e execução de uma tarefa paralelamente no *cluster*. **Sua missão é basicamente dividir uma tarefa em várias e processa em máquinas diferentes.**





Ecossistema *Hadoop* : **MapReduce**

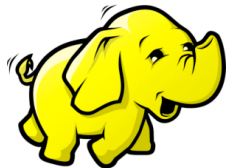


MapReduce (**Mapear** e **Reduzir**)

Em um modelo de programação, o *MapReduce* pode ser definido por três fases principais, a saber:

1. **Mapeamento** (*Map*)
2. **Embaralhar e Classificar** (*Shuffle and Sort*)
3. **Reduzir** (*Reduce*)





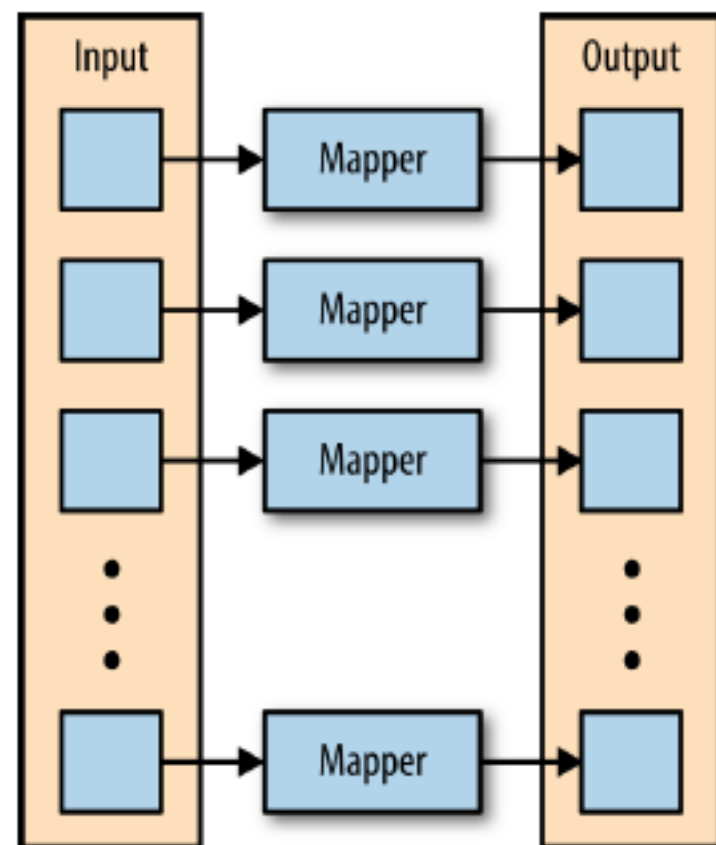
Ecossistema *Hadoop* : MapReduce

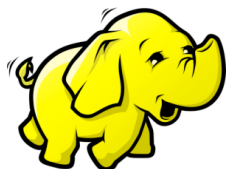


MapReduce – 1ª Fase: Mapeamento



- Uma função chamada mapeador “roteia” uma série de **pares chave-valor** dentro do estágio do mapa.
- O mapeador processa em série cada par de chave-valor separadamente, criando zero ou mais pares de chave-valor de saída.





Ecossistema *Hadoop* : MapReduce

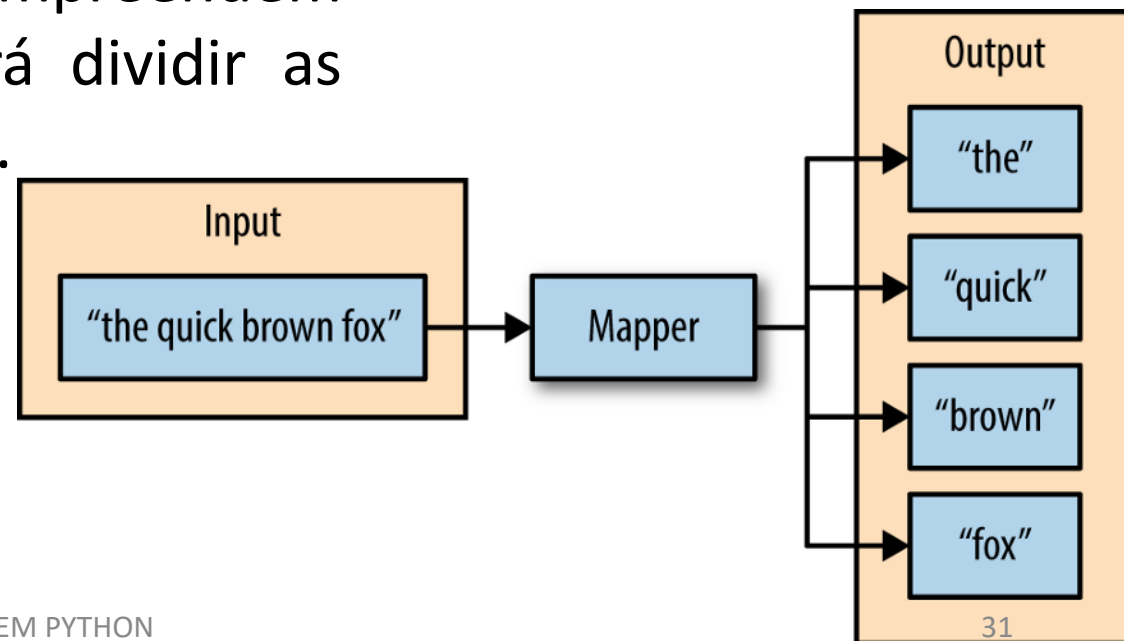


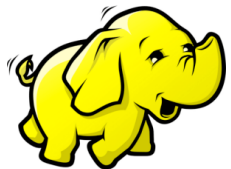
MapReduce – 1ª Fase: Mapeamento



Exemplo:

Considere um *mapeador* cujo impulso é **transformar frases em palavras**, por exemplo. A entrada para este *mapeador* serão *strings* que compreendem sentenças. A função do *mapeador* será dividir as frases em palavras e produzir as palavras.

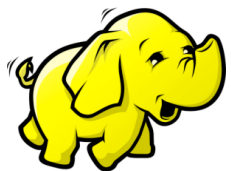




MapReduce – 2ª Fase: Embaralhar e Classificar

- As saídas intermediárias do estágio de mapeamento são movidas para os redutores à medida que os *mapeadores* são concluídos.
- Esse processo de mover a saída dos *mapeadores* para os redutores é reconhecido como **embaralhamento** e utiliza algum particionador.
- O particionador padrão é identificado como hash.
- Finalmente esses dados são **classificados** de acordo com algum critério estabelecido (ordem crescente, ordem de importância, etc).



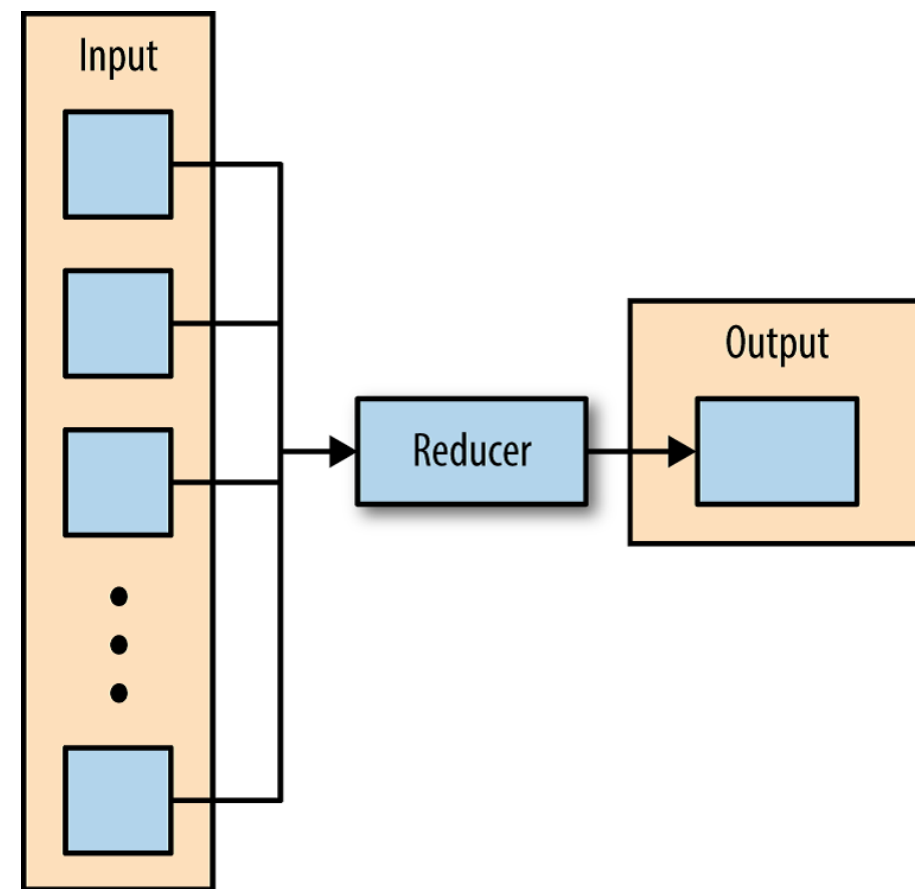


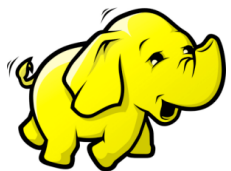
Ecossistema *Hadoop* : MapReduce



MapReduce – 3ª Fase: Reduzir

- Um *iterador* de valores é dado a uma função conhecida como redutor dentro da fase redutora.
- O iterador de valores é um conjunto não único de valores para cada chave exclusiva da saída do estágio de mapa.
- O redutor soma os valores de cada chave e produz zero ou mais pares de valores-chave de saída, ou utiliza algum outro critério que tenha por objetivo reduzir as chaves e/ou seus valores



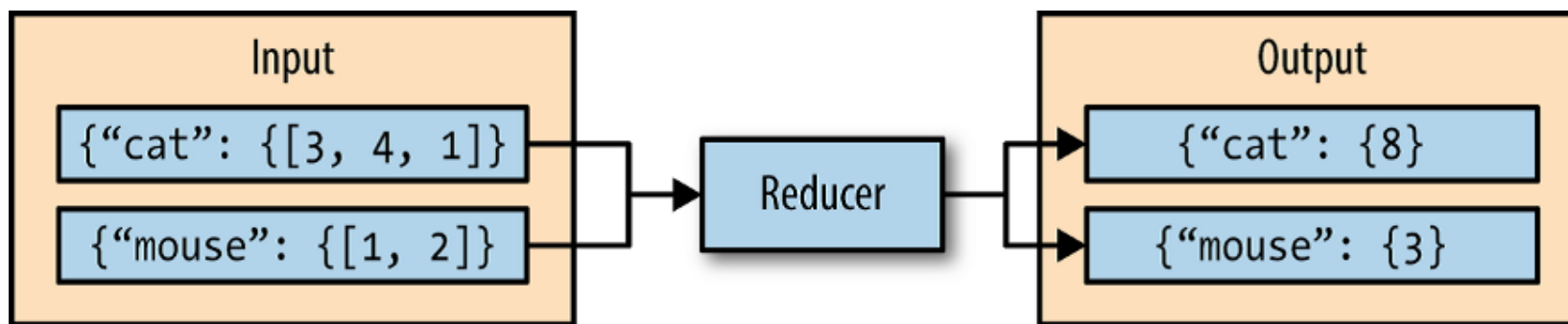


MapReduce – 3ª Fase: Reduzir

Exemplo:

Considere um **redutor** cujo objetivo é somar todos os valores de uma chave como uma instância. A entrada para esse redutor é um iterador de todos os valores de uma chave.

O redutor calcula todos os valores e então emite um **par chave-valor** que compreende a chave de entrada e a soma dos valores da chave de entrada.



BIG DATA: **Treinamento *MapReduce***



O objetivo é simular com o Python algoritmos aplicados em *MapReduce*.

1. Implemente o exemplo da 3ª fase do *MapReduce*, inserindo em um dicionário a soma dos valores de suas respectivas chaves.
2. Leia o arquivo “`mapa.txt`” e faça a contagem de palavras do arquivo, inserindo esses dados em um dicionário chamado ‘***contador***’. Exemplo:

```
coração    paixão    amor
paixão    alegria    tristeza
amor       amor       coração
alegria    tristeza    alegria
paixão
```

Arquivo `mapa.txt`



```
{‘coração’: 2, ‘paixão’:  
3, ‘amor’:3, ‘alegria’:3,  
‘tristeza’:2}
```

Dicionário `contador`



BIG DATA: **Treinamento *MapReduce***



O objetivo é simular com o Python algoritmos aplicados em *MapReduce*.

3. Leia o resultado de um mapeamento gerado em um arquivo “`mapa_2.txt`” e faça a classificação (*Sort*) das chaves de forma ascendente e decrescente.

```
{"nome1": "João",  
"nome2": "Mathew",  
"nome4": "Roxanne",  
"nome3": "David"}
```

Arquivo `mapa_2.txt`



```
{  
'nome1': 'João',  
'nome2': 'Mathew',  
'nome3': 'David',  
'nome4': 'Roxanne'  
}
```

Dicionário **crescente**

```
{  
'nome4': 'Roxanne',  
'nome3': 'David',  
'nome2': 'Mathew',  
'nome1': 'João'  
}
```

Dicionário **Decrescente**



BIG DATA: **Treinamento *MapReduce***



O objetivo é simular com o Python algoritmos aplicados em *MapReduce*.

4. Um mapeamento foi gerado, porém seus pares-chaves ficaram isolados em dois arquivos: “`keys.txt`” e “`values.txt`”. Deseja-se unificar esses itens em um dicionário. Exemplo:

Maçã pêra uva morango

`keys.txt`

3 4 10 2

`values.txt`

{'maçã': 3, 'pêra': 4,
'uva': 10, 'morango': 2}

Dicionário Contador
Frutas





Dúvidas?



