

# Especialização Desenvolvimento de Aplicações Web e Móveis Escaláveis

Turma 2021-2022

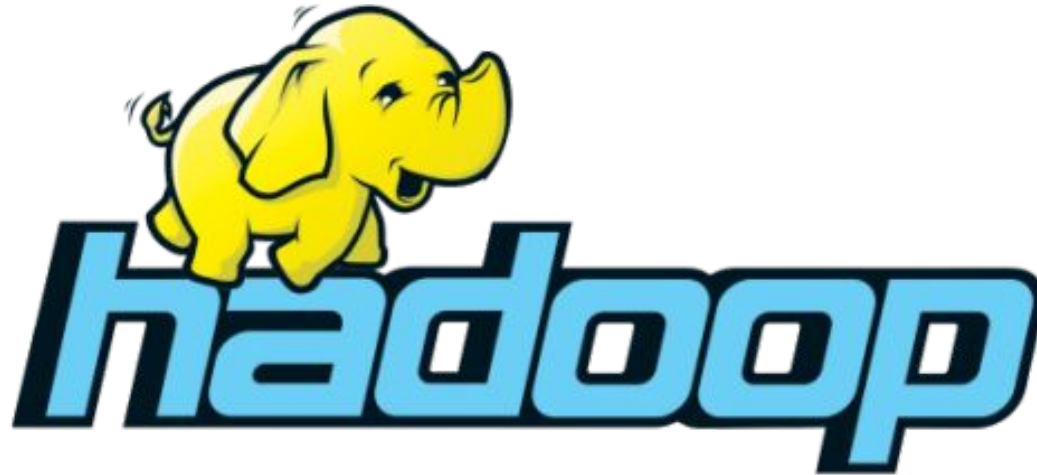
## Big Data com Python

André Morais

*[andre.morais@luizalabs.com](mailto:andre.morais@luizalabs.com)*

09/2022



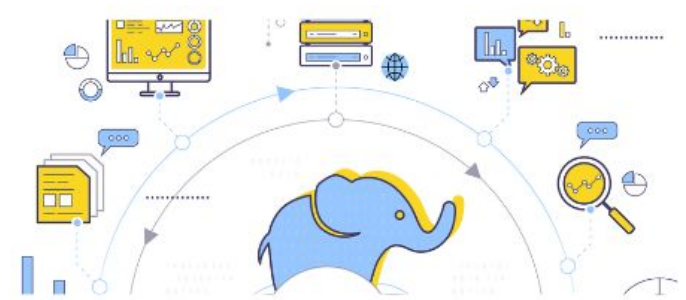


<http://hadoop.apache.org/>

**Apache Hadoop** é um framework Open Source, construído em Java, para armazenamento e processamento paralelo e distribuído e altamente confiável de grandes volumes de dados através de clusters em hardware de baixo custo e Cloud Computing.

# Hadoop

## Um breve histórico



- 2003-2004: Google lança Map Reduce e GFS
- 2005: MR e DFS implementados por [Doug Cutting](#) (Nutch)
- 2006: Hadoop se torna um projeto Apache  
<https://www.apache.org/>
- 2011: Apache disponibiliza Hadoop 1.x (HDFS + MR)
- 2013: Apache lança Hadoop 2.x (HDFS + MR + YARN)
- Atualmente está na versão 3.0.3

# Hadoop

**Por que o Apache Hadoop está se tornando padrão em projetos de Big Data?**

Baixo  
Custo

Escalável

Tolerante a  
Falhas

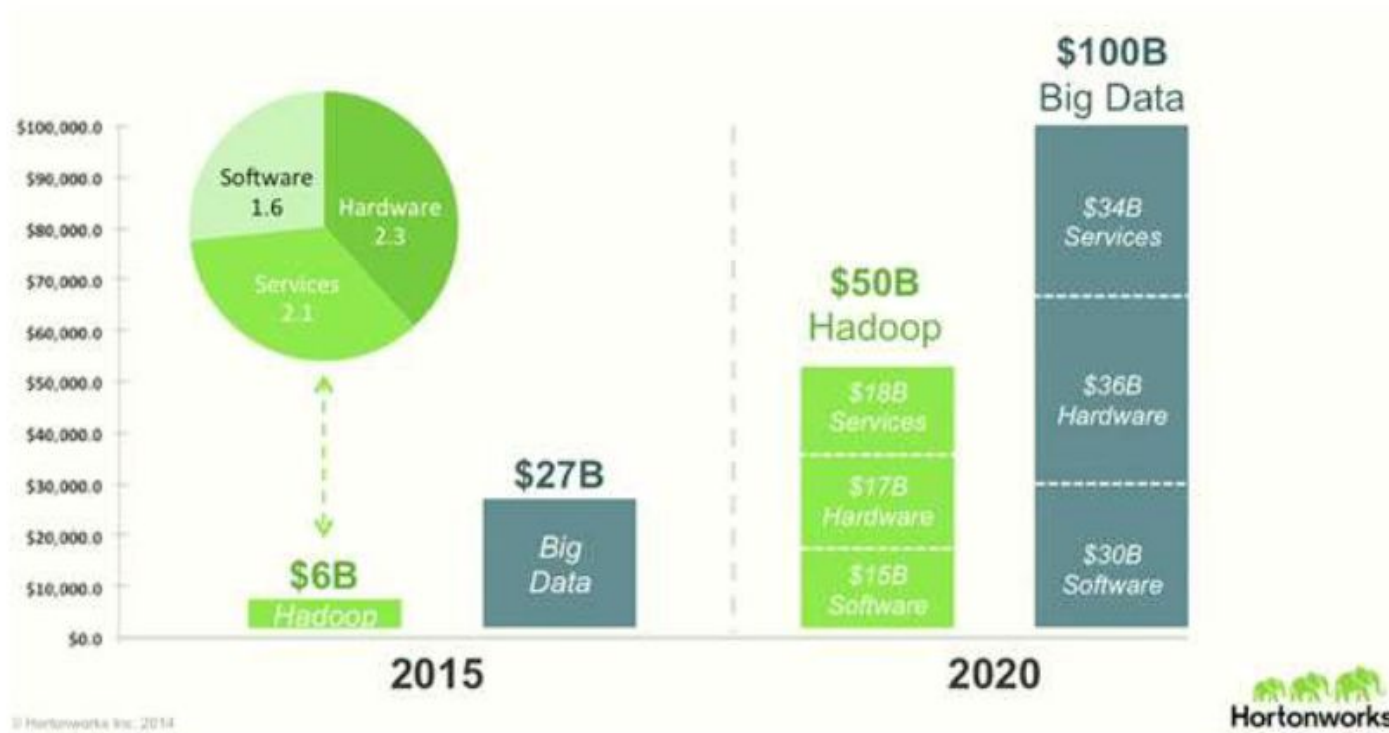
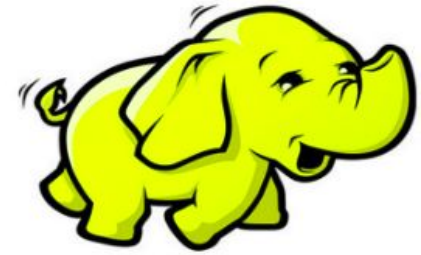
Flexível

Livre

“Um das principais características do Apache Hadoop é a confiabilidade e sua capacidade de se recuperar de falhas automaticamente”

# Hadoop

## Pesquisas mostram o Crescimento do Hadoop



# Hadoop

## Principais distribuições do Hadoop



Amazon  
Elastic  
MapReduce (EMR)

Soluções em cloud  
computing



**Azure HD Insight**



Google Cloud  
Dataproc

### Players x Implementadores:

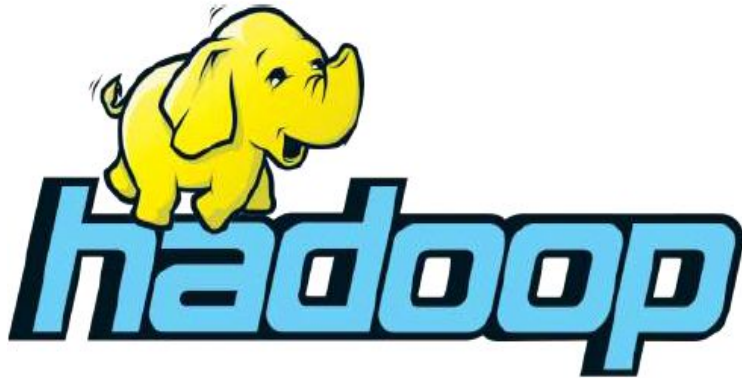
- IBM → InfoSphere → Cloudera
- Oracle → Oracle Big Data → Cloudera
- EMC → GreenPlum → MapR
- Teradata → Hortonworks
- Microsoft → HDInsight → Hortonworks

# Hadoop

## Empresas que utilizam Hadoop







<http://hadoop.apache.org>

O projeto Apache hadoop é composto de 3 módulos principais:

- Hadoop Distributed File System (HDFS)
- Hadoop Yarn
- Hadoop MapReduce

**Hadoop Common** - Conjunto de utilitários que contém a base do hadoop. É usado por toda aplicação. Bibliotecas como para paralelização de dados e manipulação de arquivos.

**Hadoop HDFS** - Sistema de arquivos distribuídos nativo do hadoop. Permite armazenamento e transmissão de grande volume de dados em máquinas de baixo custo.

**Hadoop MapReduce** - Modelo de programação especializado em processamento de conjuntos de dados distribuídos em cluster. Funções paralelas Map e Reduce.

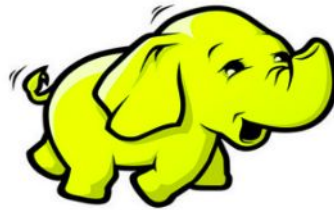
**Hadoop Yarn** – Gerenciador de jobs do Hadoop. O Yarn faz a gestão do MapReduce através dos processos: Resource manager e o Application Master.

# Hadoop

## Componentes do Hadoop

### Master - Nó Mestre

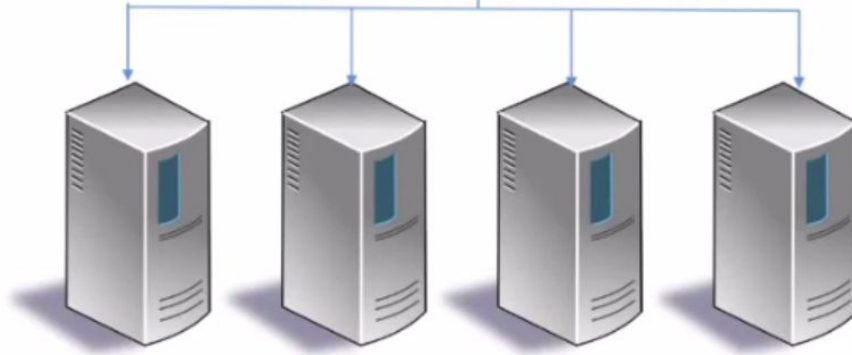
- NameNode
- SecondaryNameNode
- JobTracker



Master

### Slaves - Nós Escravos

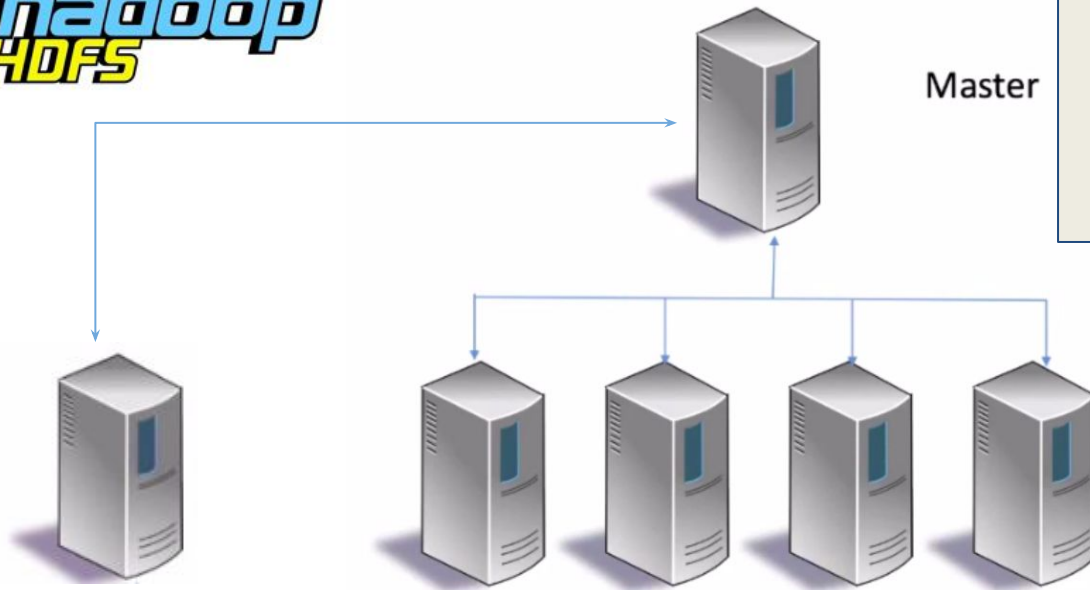
- DataNode
- TaskTracker



Slave

# Hadoop

## Arquitetura HDFS e Componentes



### Namenode

Gerencia a estrutura do file system, e dos metadados do cluster, assim como todos os arquivos e diretórios.

### SecondaryNamenode

É o Nó auxiliar do HDFS. Realiza os checkpoints (pontos de montagem) em intervalos predefinidos e ajuda no nível de desempenho do Namenode

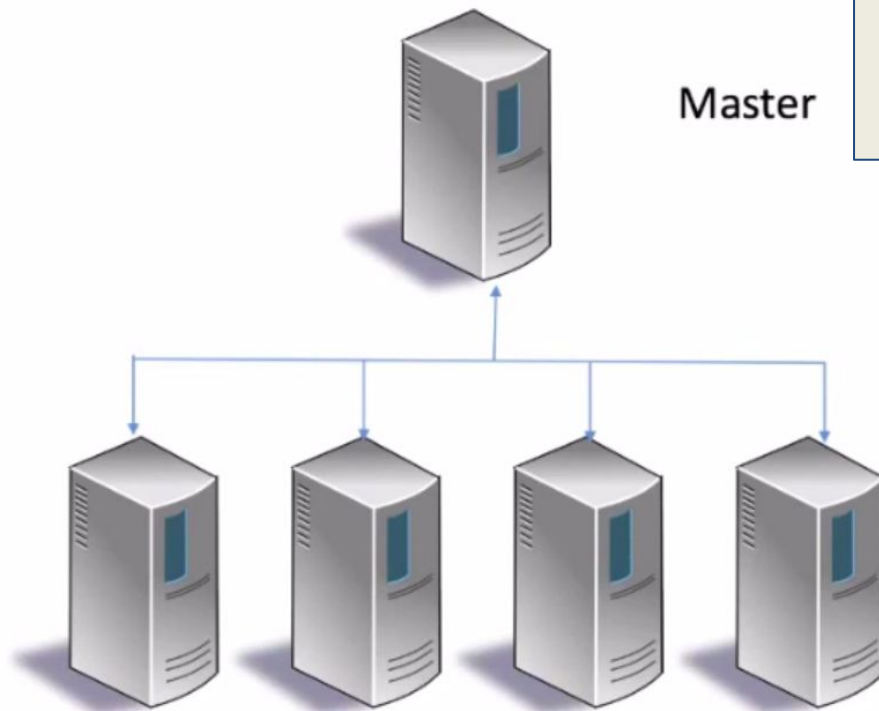
### Slave

### Datanode

Armazena os blocos de dados e responde à aplicação cliente ou Namenode que solicitado. Reporta frequentemente para o Namenode seu status e lista de blocos armazenados.

# Hadoop

## Arquitetura MapReduce e Componentes



Master

### JobTracker

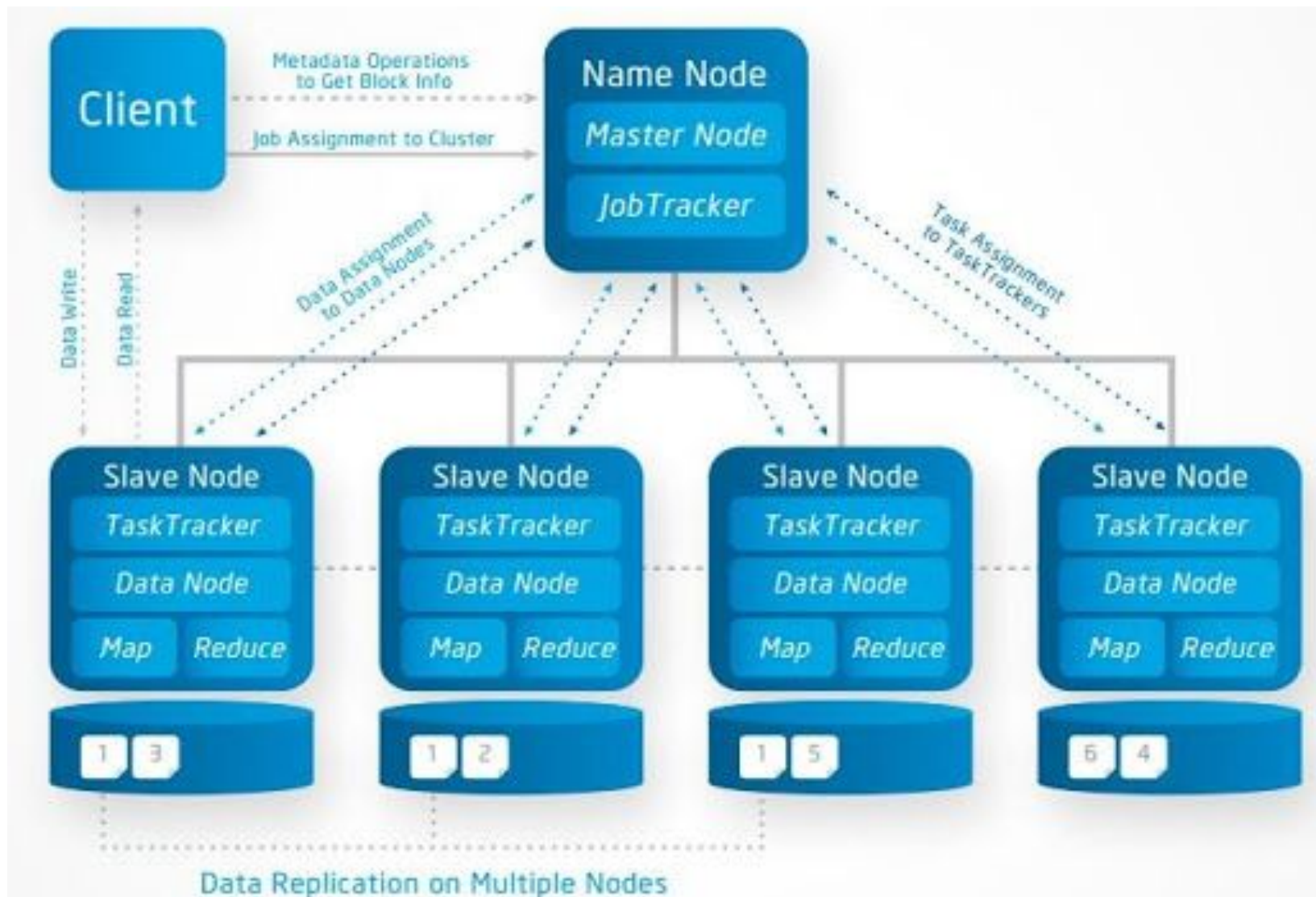
Gerencia o plano de execução das tarefas e delega as tarefas aos nós escravos. Monitora as execuções e atua em caso de falhas.

### TaskTracker

Executa o processamento das tarefas MapReduce, com um instância em cada nó escravo.

Slave

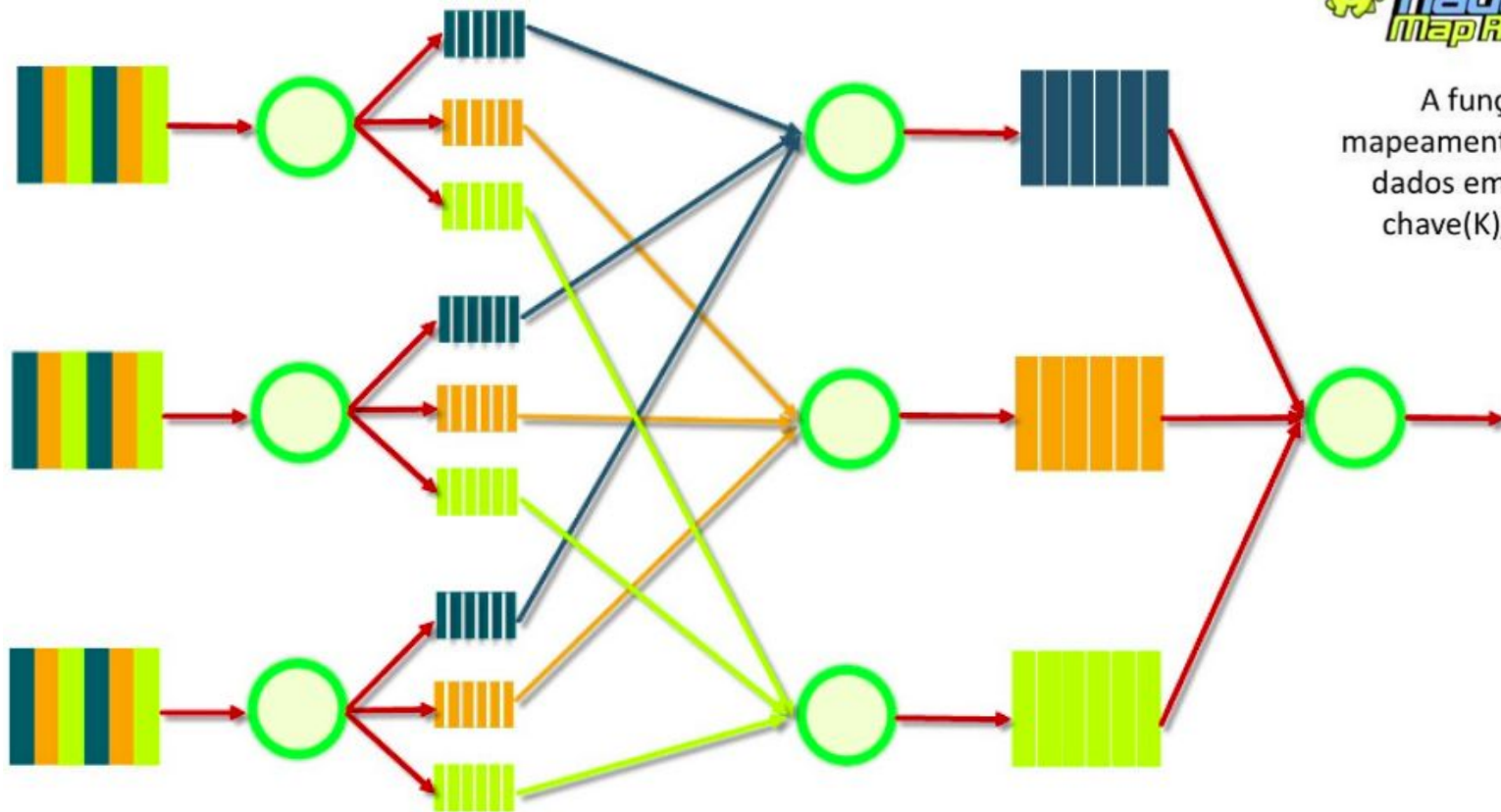
# Hadoop Arquitetura Visão Geral



# Hadoop Processamento MapReduce



A função de mapeamento, converte dados em pares de chave(K)/valor(V)



## Map

Trabalha em um conjunto de dados de entrada dos blocos produzindo uma lista de chaves e valores

## Shuffle

O processamento intermediário é feito com persistência em disco no Hadoop

## Reduce

Trabalha em um conjunto de entrada de dados dos blocos, produzindo uma lista de chaves e valores.



# Hadoop Ecosystem

## Provisioning, Managing and Monitoring Hadoop Clusters




  
**Scoop**  
Data Exchange



**Zookeeper**  
Coordination

  
**Oozie**  
Workflow

  
**PIG**  
Scripting

  
**mahout**  
Machine Learning

  
**R Connectors**  
Statistics

  
**Hive**  
SQL Query



**YARN Map Reduce v2**  
Distributed Processing Framework

**Hadoop Distributed File System**



  
**APACHE HBASE**  
Column Store

  
**Flume**  
Log Collector

# Hadoop Principais projetos



**Zookeeper** - Criado pelo Yahoo em 2007 para coordenar aplicações distribuídas de alto desempenho. Possui recursos como Configuração de nodes do cluster, sincronização de processos distribuídos, e grupos de serviço.



**Hive** - Desenvolvido pela equipe de funcionários do facebook. Tornou-se código aberto em 2008. Possui uma infraestrutura que permite utilizar o **HSQL** ou **HiveQL** similar ao SQL e conceito de bases relacionais visando análise complexas em dados não relacionais.



# Hadoop Principais projetos



**Hbase** - Banco NoSql criado pela Power7 em 2007. Posteriormente incorporado a Apache. Considerado uma versão de código aberto do Big Table criado pela Google. Distribuído e escalável para armazenamento estruturado para grandes tabelas.



**Pig** - Linguagem de alto nível orientada a fluxo de dados e de execução de computação paralela. Modo Client side, não altera a configuração do Hadoop. Usa a linguagem Pig Latim e é compilada para utilizar as funcionalidades da programação Map Reduce.

# Hadoop Principais projetos



**Sqoop** - Ferramenta de transferência de dados. Importar dados para HDFS, Hive ou Hbase. E exportar para outras bases de dados externas. Paraleliza transferência de dados otimizando desempenho, fazendo melhor uso de recursos e redes. Usa a linguagem SQL para as bases relacionais e importação no Hadoop. Possui conectores para MySql, PostgreSQL, Oracle, MS SQL Server, IBM DB2.



**Mahout** - Primeira versão em 2009. Utilizar o MapReduce aplicando algoritmos complexos de machine learning para grandes volumes de dados.

# Hadoop Principais projetos



**Flume** - Criado em 2011 pela Cloudera. Sistema distribuído, confiável e disponível para coletar, agregar e mover grandes quantidades de dados de várias fontes diferentes. Uso para coleta de logs, mas também para transportar grandes volumes de dados de outras fontes como dados de redes sociais, e-mails, dados de streaming e outros.



**Oozie** - É um sistema de fluxo de trabalho e coordenação que gerencia jobs do hadoop, é integrado à pilha do hadoop. Suporte ao MapReduce, Pig, Hive ou Sqoop. Pode ser usado para agendar jobs específicos para o sistema, como escritos em Java, Python e Shell Scripts.





É um framework para processamento de Big Data construído com foco em velocidade, facilidade de uso e análises sofisticadas.

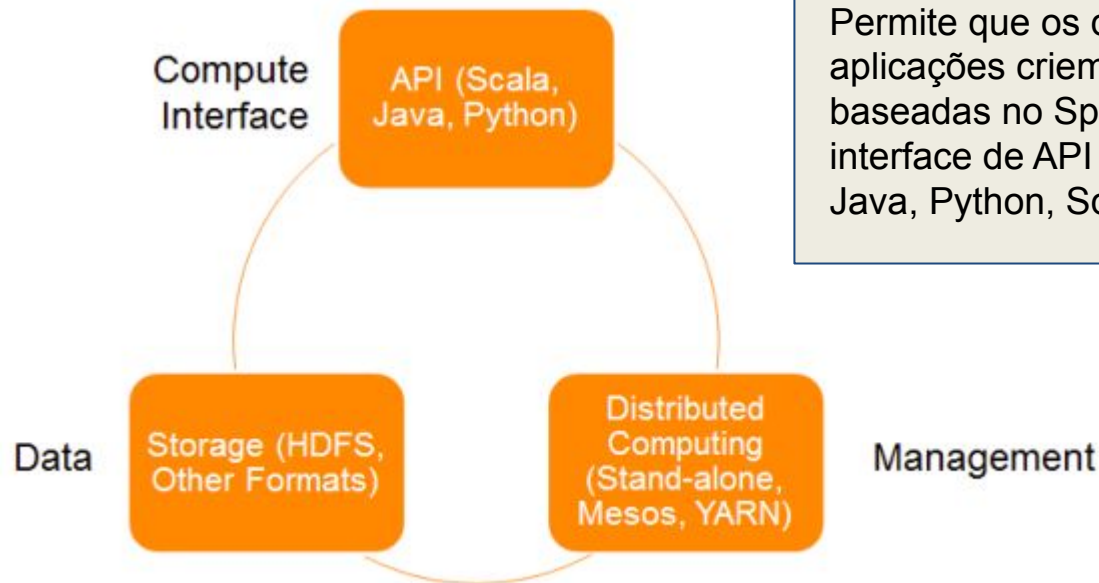
<https://spark.apache.org/docs/latest/api/python/index.html>

# Spark Algumas características



- Foi desenvolvido na UC Berkeley em 2009. A mesma equipe fundou a Databricks em 2013: <https://databricks.com/spark/about>
- Desenvolvido em Scala. Porém fornece APIs de alto nível em Java, Scala, Python, SQL e na linguagem R.
- Possui um ecossistema de bibliotecas que permitem trabalhar de forma integrada em uma mesma aplicação, usando linguagem SQL, streaming e análises complexas para lidar com uma variedade de situações de processamento de dados.
- Realiza processamento em memória, suprimindo a deficiência do Hadoop MapReduce, processando 100x mais rápido em memória e 10x mais rápido em disco.
- Não possui armazenamento próprio e trabalha muito bem com Hadoop e outras tecnologias.

# Spark Componentes da Arquitetura



## API

Permite que os desenvolvedores de aplicações criem aplicações baseadas no Spark usando uma interface de API padrão para Scala, Java, Python, Sql e R.

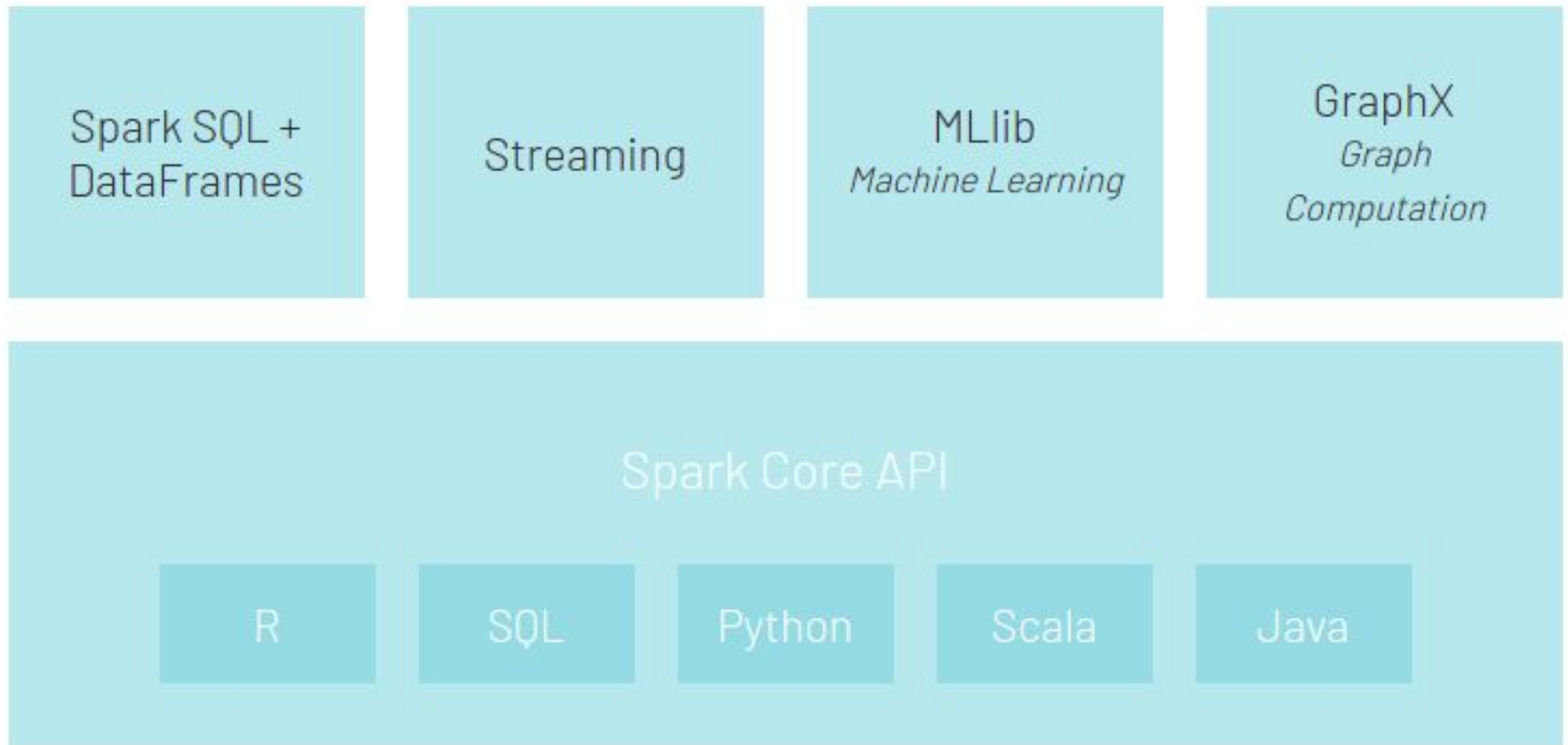
## Armazenamento

Utiliza sistema de arquivos distribuídos como HDFS, e outros em Cloud como GCS e S3. Funciona com outras fontes de dados como HBase, Cassandra, etc.

## Gerenciamento (recursos)

Sua arquitetura pode ser implantada em servidor autônomo ou em uma estrutura de computação distribuída como o Mesos ou o YARN (Hadoop)

# Spark Ecossistema

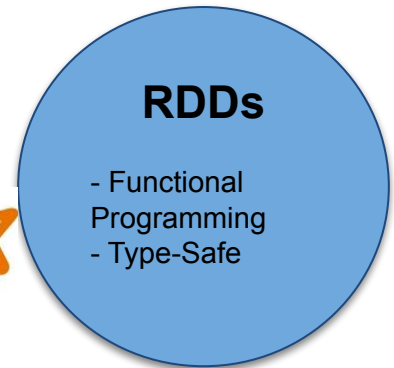


**Existem três interfaces principais do Apache Spark:** Os **RDDs** - Resilient Distributed Dataset, os **DataFrames** e os **Datasets**.



# Spark INTERFACES

## RDD - Resilient Distributed Datasets



- RDD é a primeira abstração, a estrutura original e fundamental do Apache Spark. São a API de “nível mais baixo”.
- É a infraestrutura que permite que o Spark seja executado com rapidez e forneça a linhagem de dados.
- Como uma tabela em um banco de dados, ele pode conter qualquer tipo de dados.
- O armazenamento de dados em RDD é feito em diferentes partições.
- São tolerantes e resilientes a falhas.
- São imutáveis.
- O RDD suporta dois tipos de operações: Transformação e Ação.

# Spark INTERFACES

## DataFrames



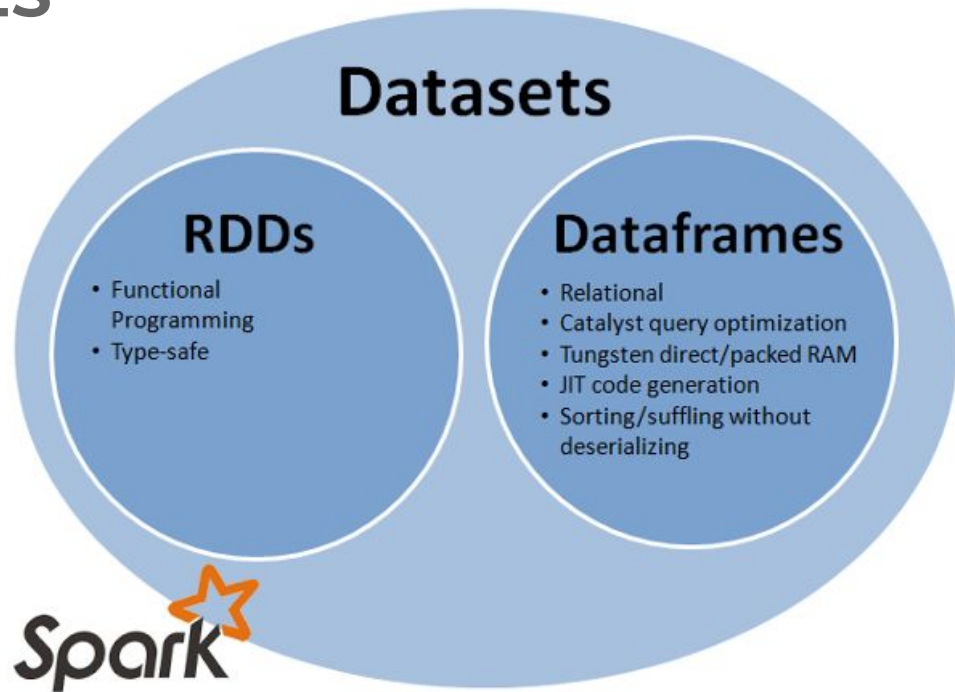
### DataFrames

- Relacional
- Catalyst query optimization
- Tungsten direct/packed RAM
- JIT code generation
- Sorting/shuffling without deserializing

- DataFrame é um superconjunto da funcionalidade RDD. Os Dataframes estão disponíveis nas linguagens Java, Python e Scala.
- Um DataFrame possui metadados adicionais devido ao seu formato tabular, o que permite ao Spark executar certas otimizações na consulta finalizada.
- São semelhantes ao conceito utilizado nas bibliotecas pandas no Python e na linguagem R.
- Em geral é aconselhado a utilização do DataFrame, especialmente com as otimizações de desempenho incorporadas no mesmo.

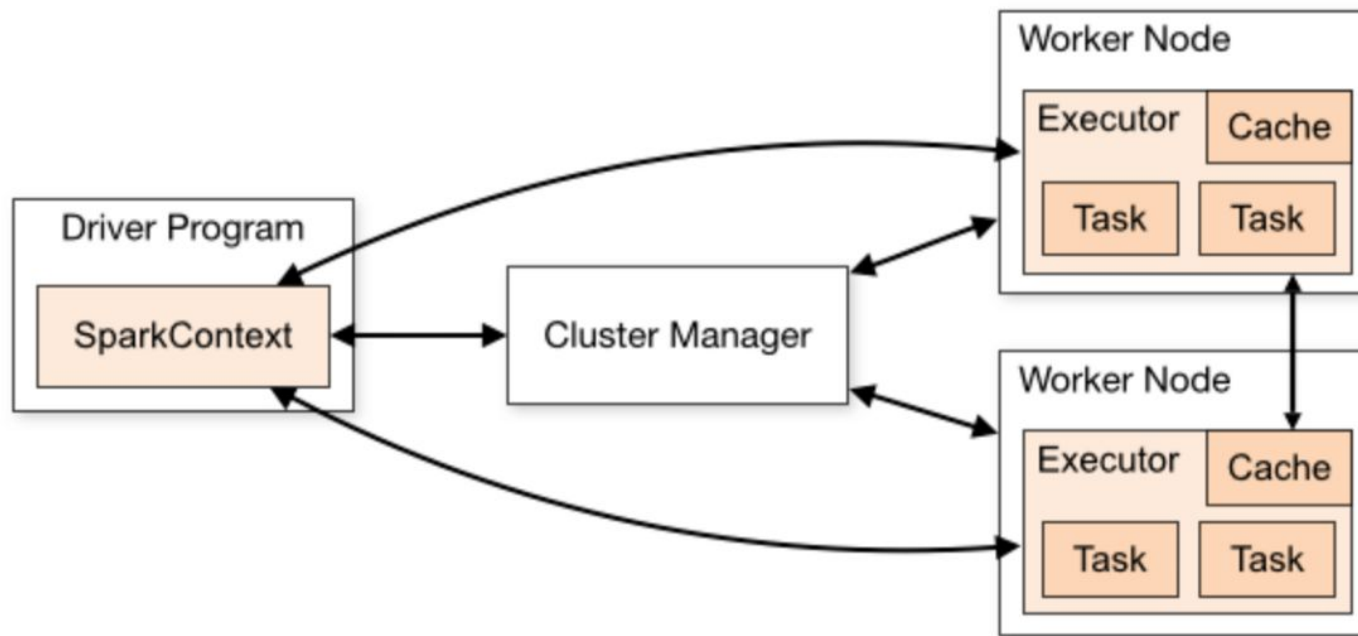
# Spark INTERFACES

## Datasets



- DataSet é uma combinação do Dataframe e RDD.
- Fornece a interface que está disponível em RDDs e ao mesmo tempo a conveniência do Dataframe.
- A API Dataset está disponível apenas nas linguagens Java e Scala.

# Spark Arquitetura e funcionalidade Core



1. O **Driver Program** é a aplicação principal que gerencia a criação e quem executará o processamento;
2. O **Cluster Manager** é um componente opcional que só é necessário se o Spark for executado de forma distribuída, responsável por administrar os workers;
3. Os **Workers** ou **Executores**, que são as máquinas que realmente executarão as tarefas que são enviadas pelo Driver Program.

# Spark

## Bibliotecas do Ecossistema Spark



O Spark SQL utiliza SQL na realização de consultas e processamento sobre os dados no Spark. Utiliza a interface de DataFrame para manipulação de dados, possibilitando a construção de ETLs e análises complexas sobre grandes volumes.

# Spark

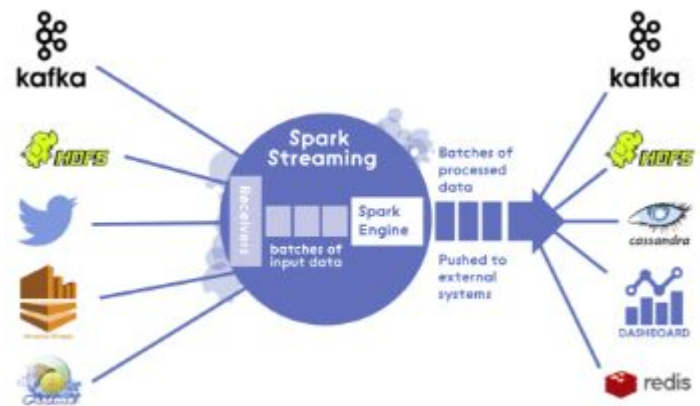
## Bibliotecas do Ecosystema Spark



O Spark Streaming possibilita o processamento de fluxos em tempo real. Se baseia na arquitetura de processamento em micro-batch. O intervalo é configurável na aplicação. Utiliza o DStream, que é basicamente uma série de RDDs, para processar os dados em tempo real.

# Spark

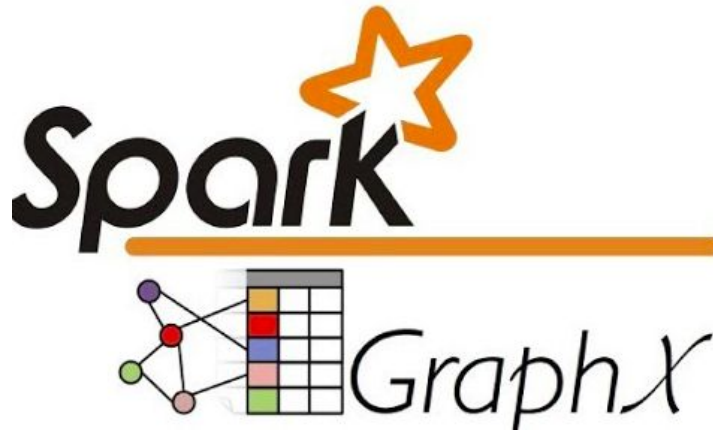
## Bibliotecas do Ecossistema Spark



A MLlib é a biblioteca de aprendizado de máquina, com diversos algoritmos para as mais diversas atividades, incluindo classificação, regressão, clustering, filtragem colaborativa, redução de dimensionalidade, bem como primitivas de otimização subjacentes.

# Spark

## Bibliotecas do Ecosistema Spark



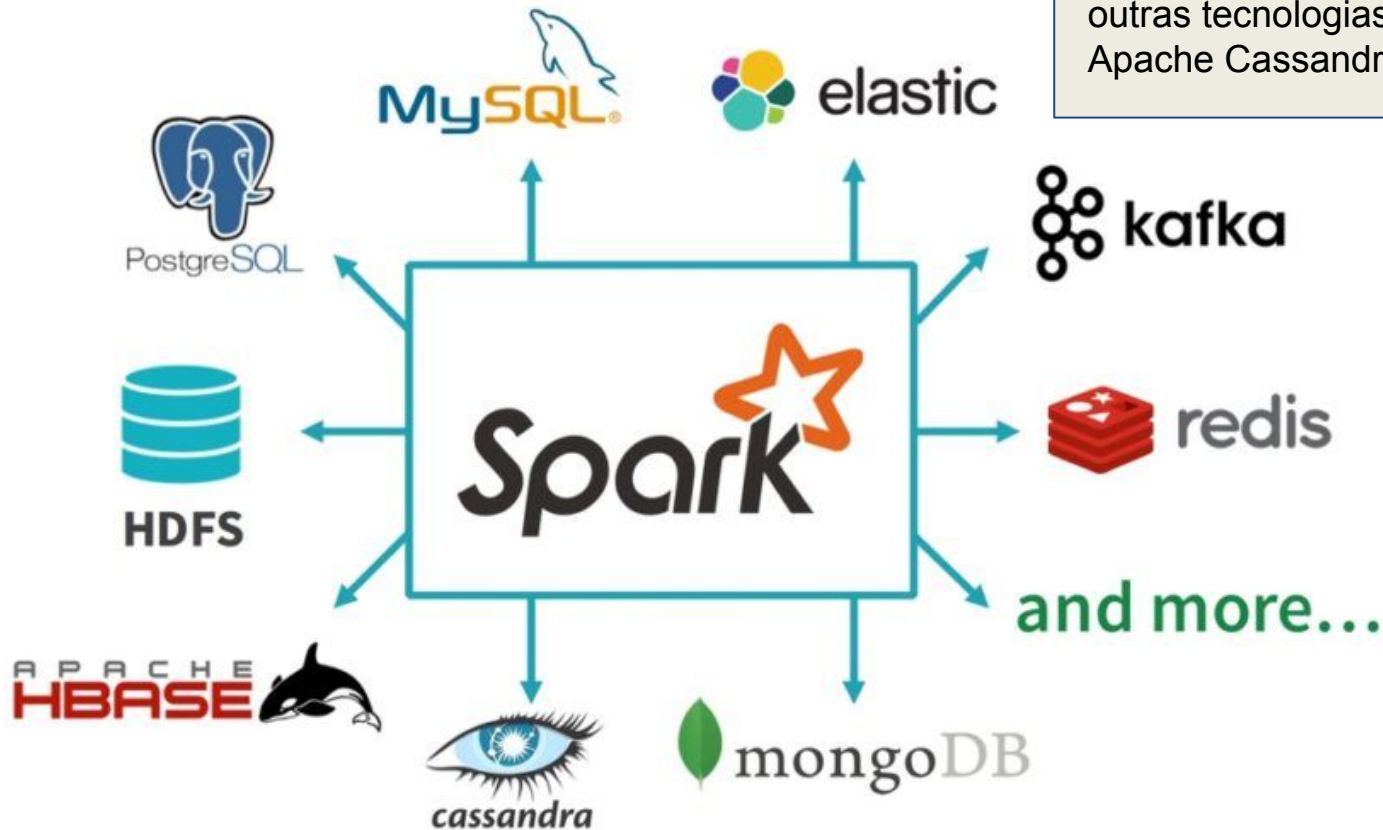
O GraphX realiza o processamento sobre grafos. Ele estende as funcionalidades do RDD em alto nível, e inclui uma coleção crescente de algoritmos e construtores de grafos para simplificar as tarefas de análises.



# Spark Integração com outras Tecnologias

## Integração

Com vários recursos e adaptadores, é possível combinar Spark com outras tecnologias, como Kafka e Apache Cassandra, entre outras.



# Spark

## O Spark substitui o Hadoop?



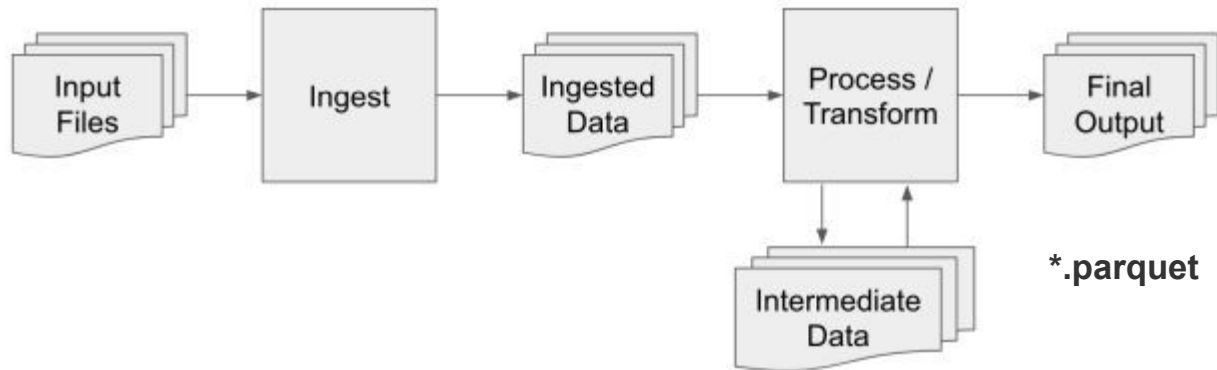
Como vimos, o Apache Spark pode ser utilizado independentemente do Hadoop, mas se complementam através do Yarn e HDFS. Além disso, o ecossistema Hadoop é muito rico, sendo viável em muitas arquiteturas de Big Data. Atualmente o Spark 3.0+ já roda em kubernetes, e com as arquiteturas em nuvem, viabiliza uma “carreira solo” para o Spark.



PySpark é uma API Python para Spark. Foi construído em cima da API Java, e é apenas uma fina camada de software Python que repassa as chamadas de funções para o core Java. Foi lançado a fim de apoiar a colaboração do Apache Spark e Python.

<https://spark.apache.org/docs/latest/api/python/index.html#>

# Parquet



**Apache Parquet** é um formato de armazenamento colunar disponível para qualquer projeto no ecossistema Hadoop e Spark. Independe da escolha da estrutura de processamento de dados, modelo de dados ou linguagem de programação. É o formato padrão para o Spark.

**\*.parquet**

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



**Trabalhando com PySpark**