Processamento de dados em Big Data

Parte 2 - Frameworks

Luiz Henrique Zambom Santana, D.Sc.

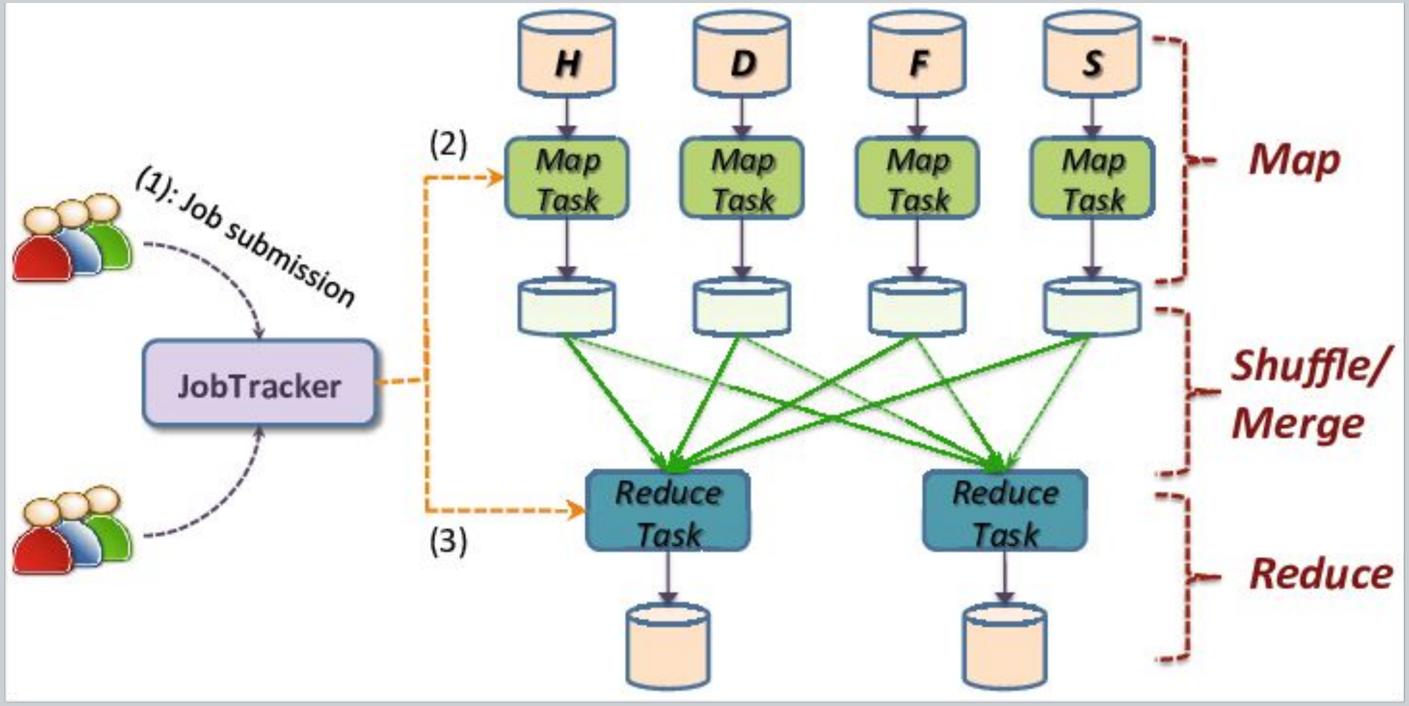
INE | CTC

UNIVERSIDADE FEDERAL
DE SANTA CATARINA

Agenda

- Processamento:
    - Hadoop
    - Apache Spark
        - Core
        - Streaming
        - Machine learning
- Streaming: Apache Kafka e NiFi
- Orquestração: Apache Camel e Apache AirFlow
- Data Lake: Apache Hudi, Delta Lake

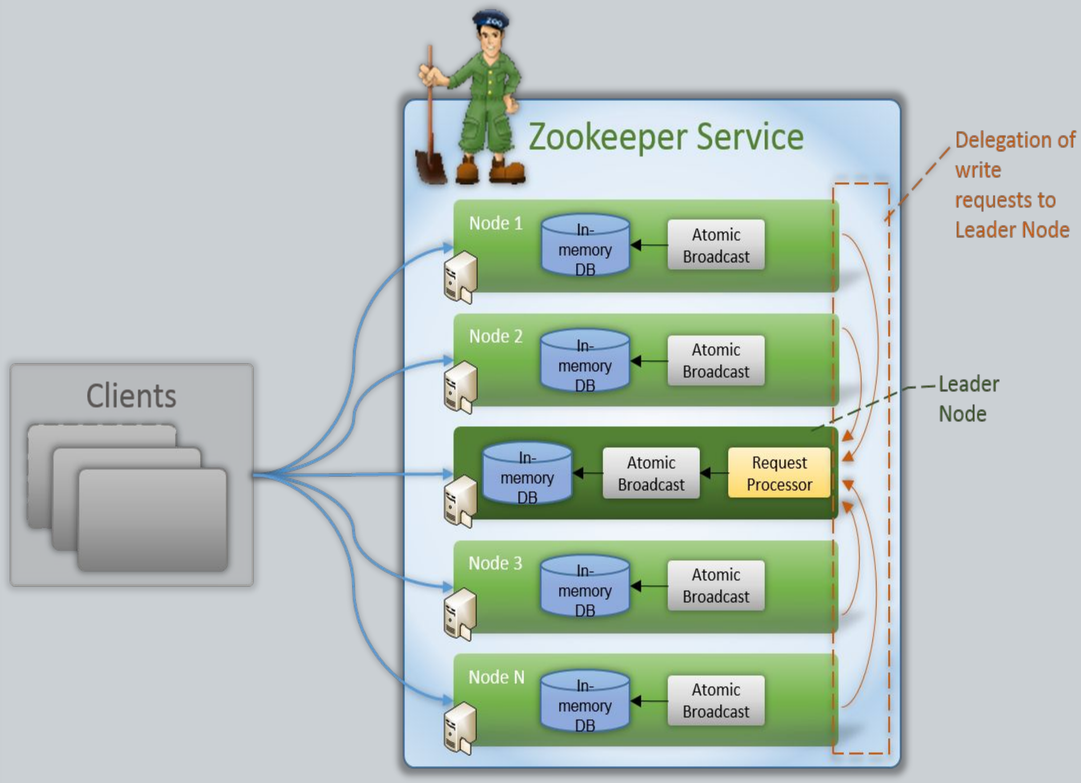
Apache Hadoop



Apache Hadoop

- O Hadoop oferece um ecosistema completo:
  - Zookeeper
  - Hive
  - Pig
  - HDFS

Apache Hadoop - Zookeeper

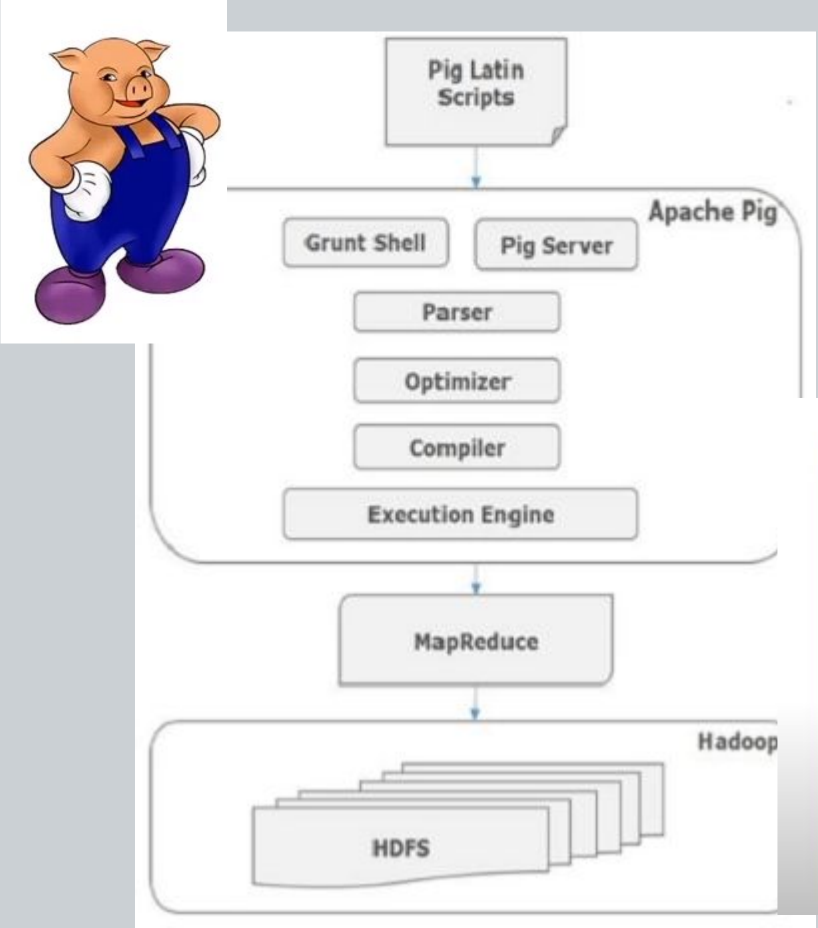


- Serviço de nomeação
- Gerenciamento de configuração:
- Gerenciamento de cluster
- Eleição do líder
- Serviço de bloqueio e sincronização
- Registro de dados altamente confiável
- Kafka, Hadoop, Spark, NiFi…

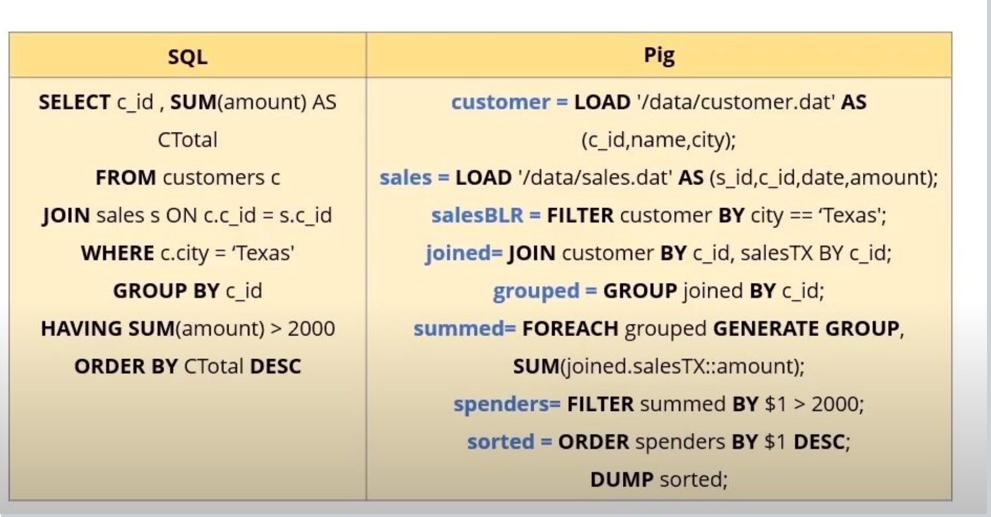
Apache Hadoop - Hive

- HQL: Hive Query Language
- Traduzido para jobs MapReduce no Hadoop
- OLAP, mas não suporta OLTP
- Funcionalidades
    - Suporta vários formatos, incluindo ORC
    - Usa vários tipos de compressão
    - Joins especiais para aumentar o desempenho
    - Schema on Read

Apache Hadoop - Pig



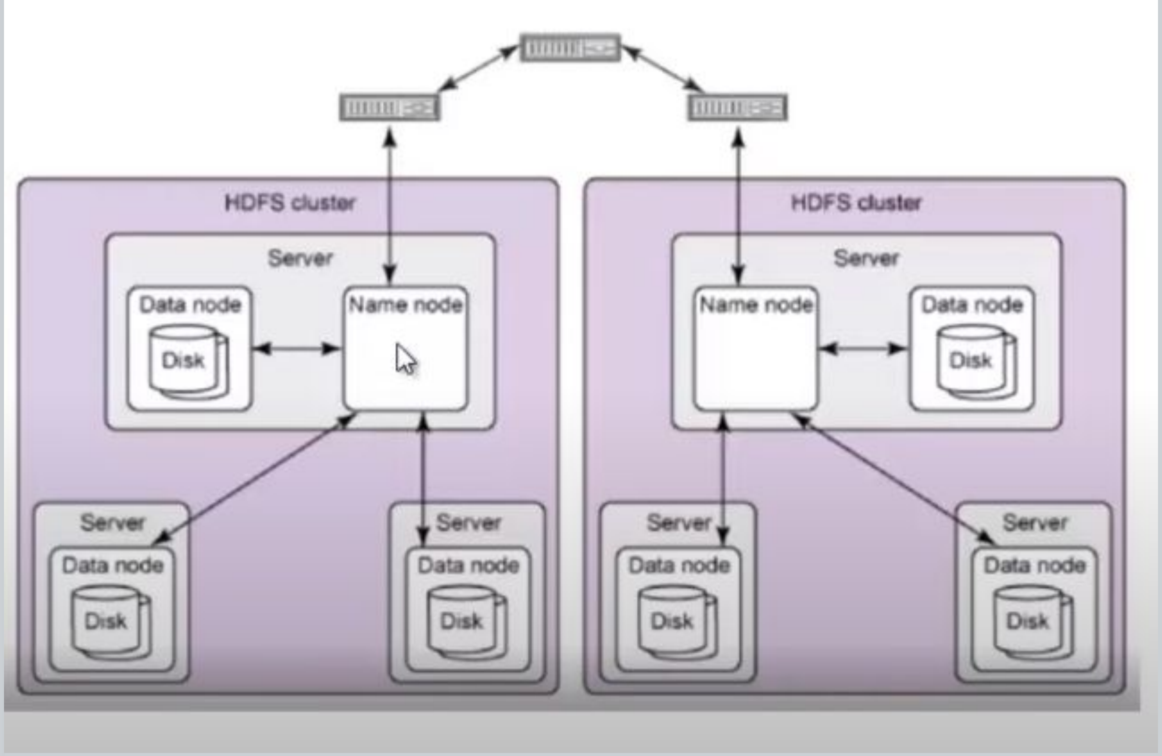
- Script para acessar Hadoop e fazer análise de dados
- Bom para dados desestruturados



SQL	Pig

SELECT c_id , **SUM**(amount) AS CTotal
FROM customers c
JOIN sales s ON c.c_id = s.c_id
WHERE c.city = 'Texas'
GROUP BY c_id
HAVING SUM(amount) > 2000
ORDER BY CTotal **DESC**
	customer = **LOAD** '/data/customer.dat' **AS** (c_id,name,city);
	sales = **LOAD** '/data/sales.dat' **AS** (s_id,c_id,date,amount);
	salesBLR = **FILTER** customer **BY** city == 'Texas';
	joined= **JOIN** customer **BY** c_id, salesTX BY c_id;
	grouped = **GROUP** joined **BY** c_id;
	summed= **FOREACH** grouped **GENERATE GROUP**, **SUM**(joined.salesTX::amount);
	spenders= **FILTER** summed **BY** $1 > 2000;
	sorted = **ORDER** spenders **BY** $1 **DESC**;
	DUMP sorted;

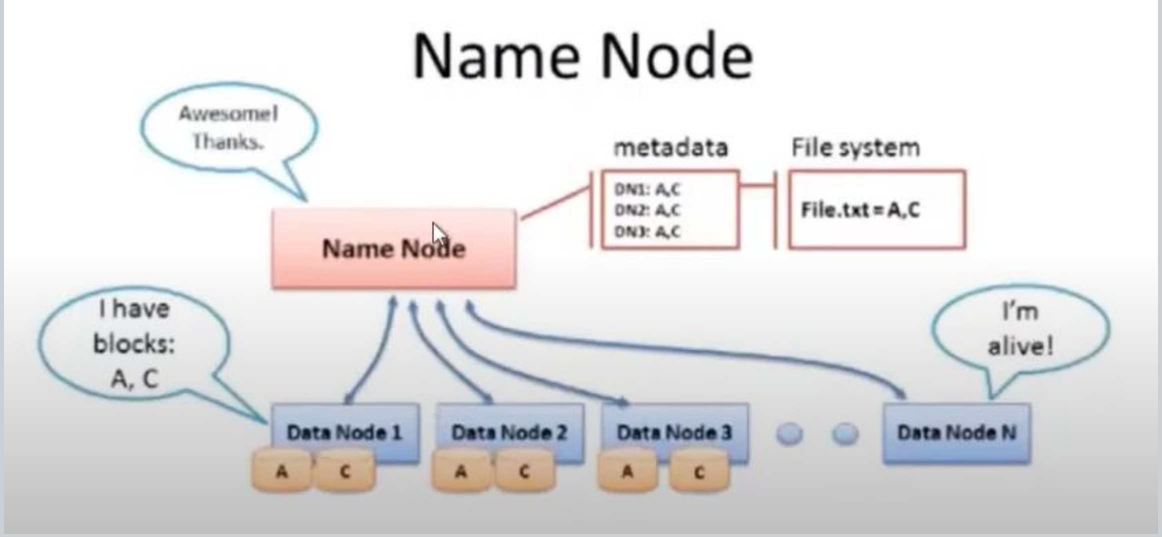
Apache Hadoop - HDFS



- Hadoop File System
- Sistema de arquivos distribuídos
- Forma uma abstração
- Baseado no Google File System

https://www.youtube.com/watch?v=Z4htZMwIfDs

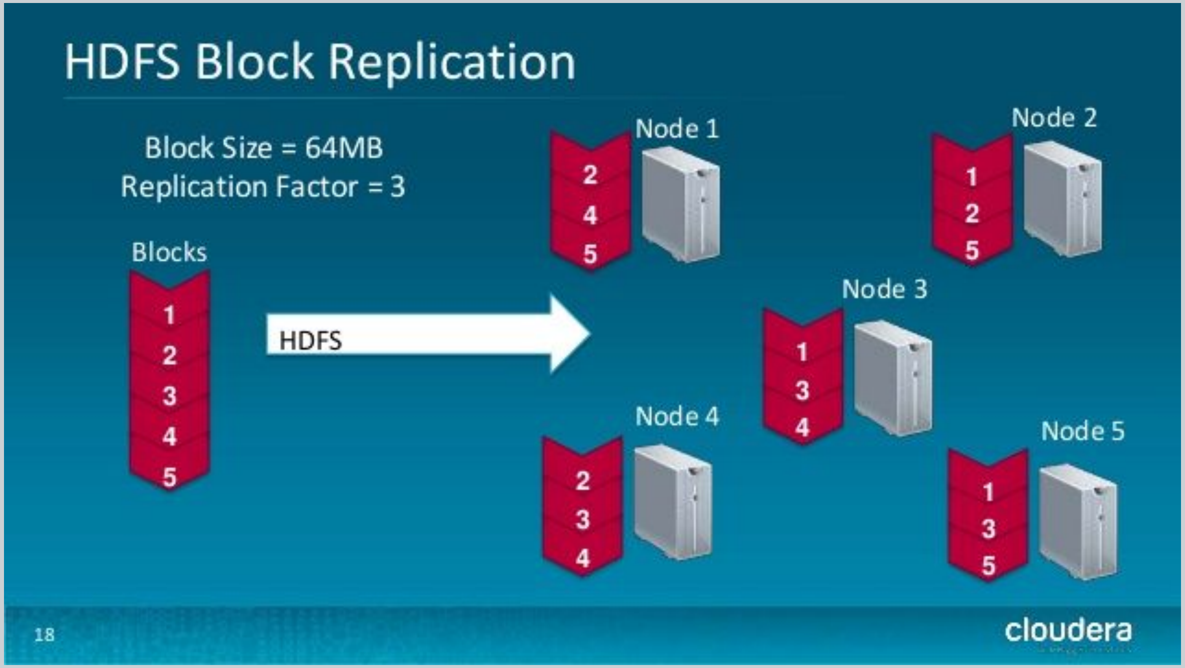
Apache Hadoop - HDFS



- Um grande índice de metadados

https://www.youtube.com/watch?v=Z4htZMwIfDs

Apache Hadoop - HDFS

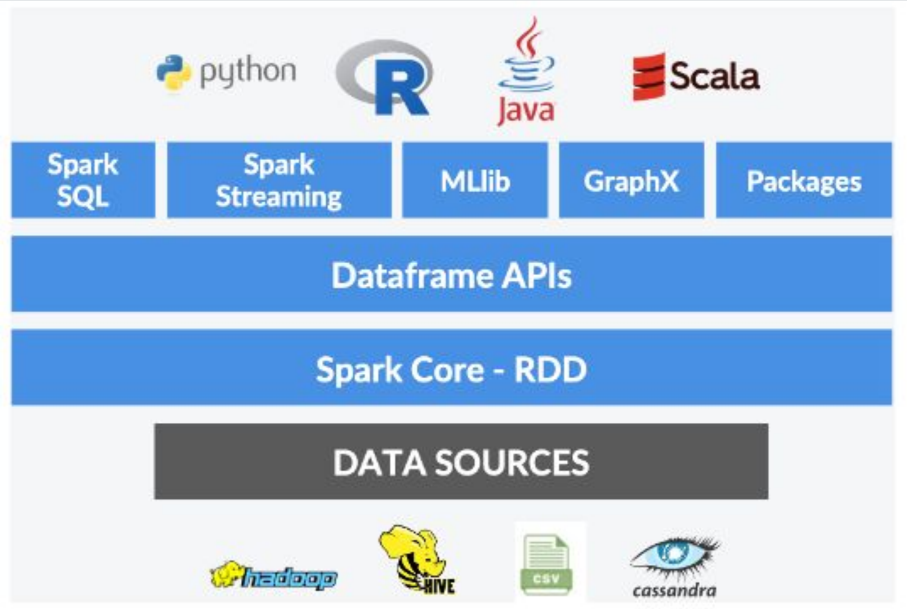


- Utiliza sempre blocos sequenciais
- Replicação de blocos

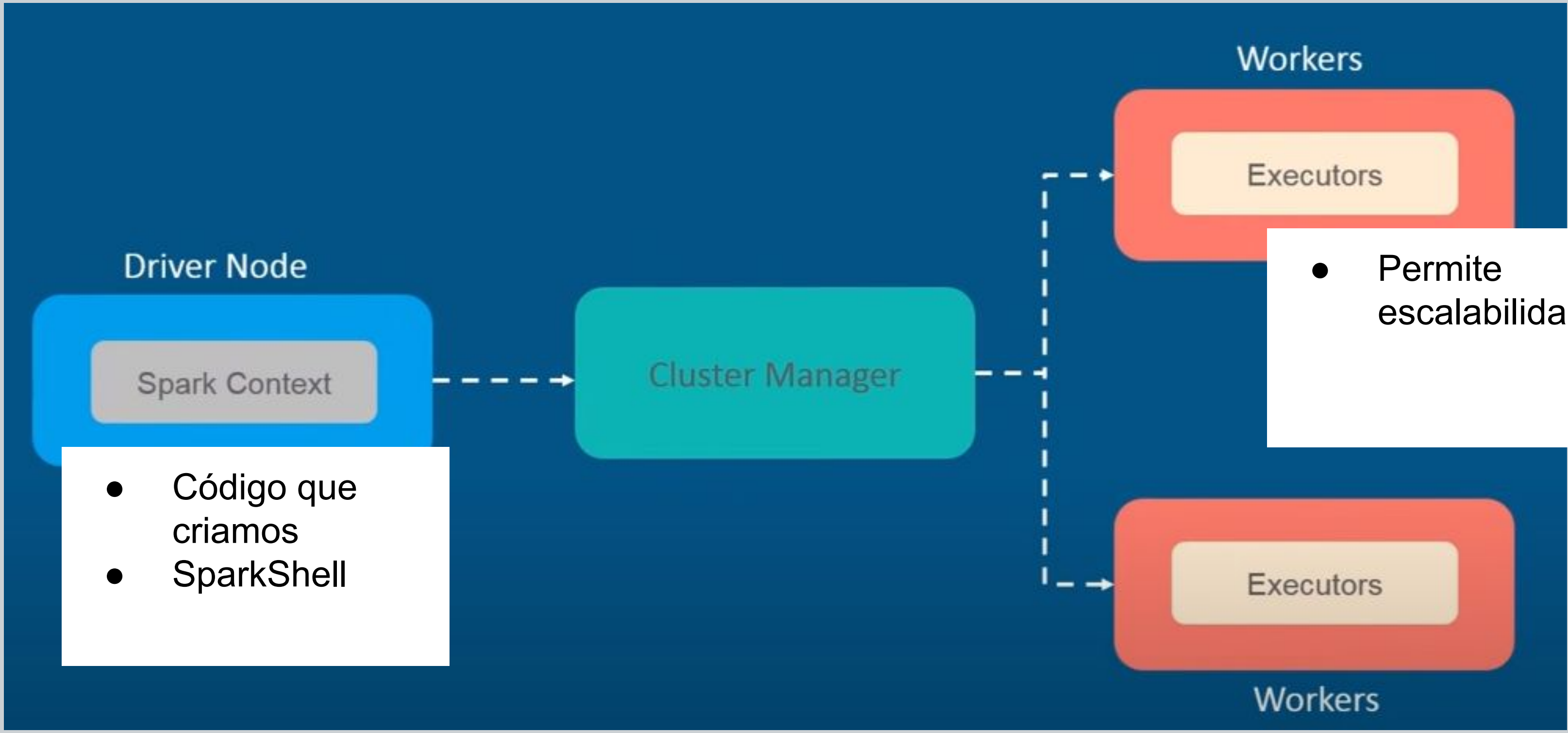
https://bigishere.wordpress.com/2016/08/03/configuring-replication-factor-and-block-size-in-hdfs/

Apache Spark

- Apache Spark
    - Core
        - RDD
            - Transformation
            - Actions
        - DAG
    - Streaming
    - Machine learning



Apache Spark

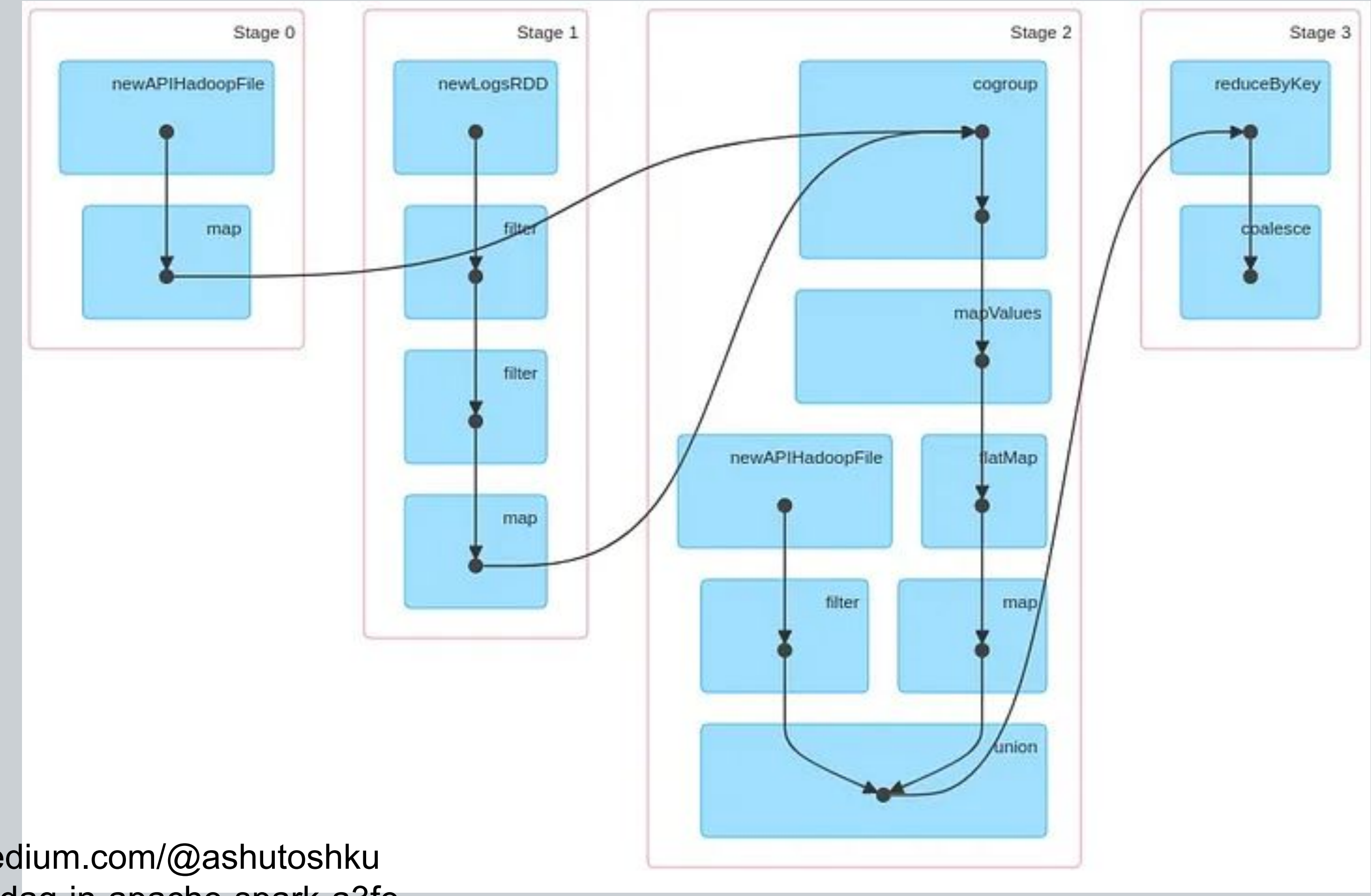


- Código que criamos
- SparkShell

- Permite escalabilidade

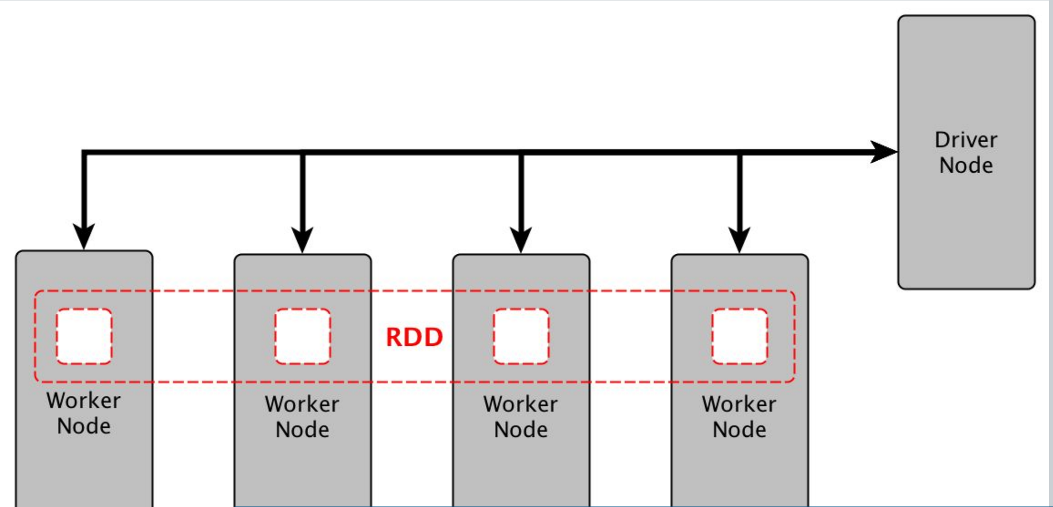
https://www.youtube.com/watch?v=jffQhcweGwY

Apache Spark -
Directed Acyclic Graph (DAG)

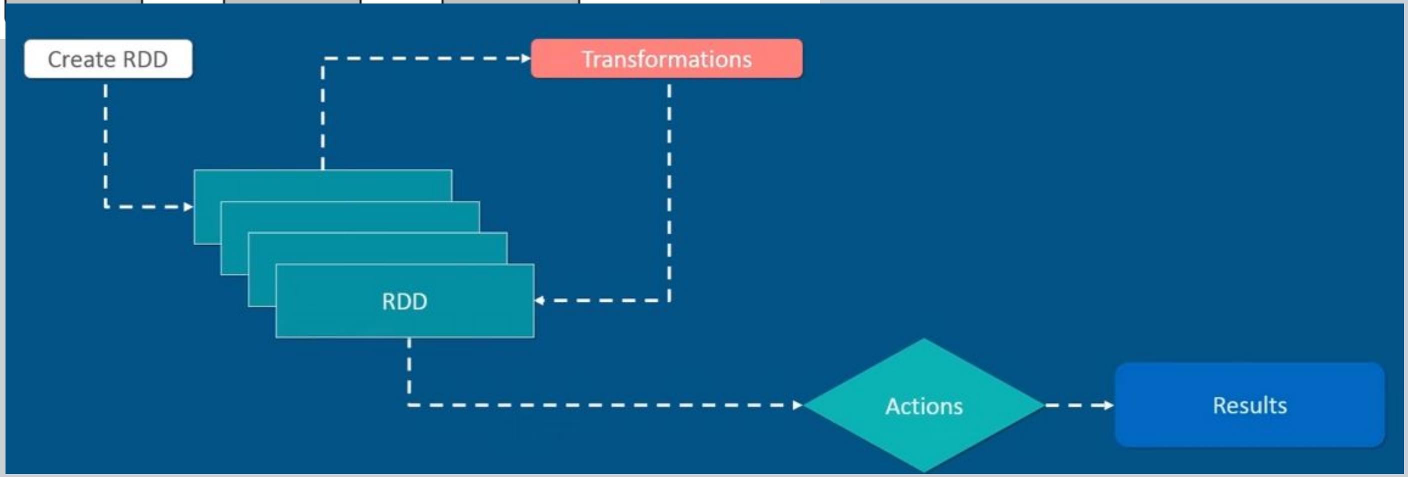


https://medium.com/@ashutoshkumar2048/dag-in-apache-spark-a3fee17f7494

Apache Spark

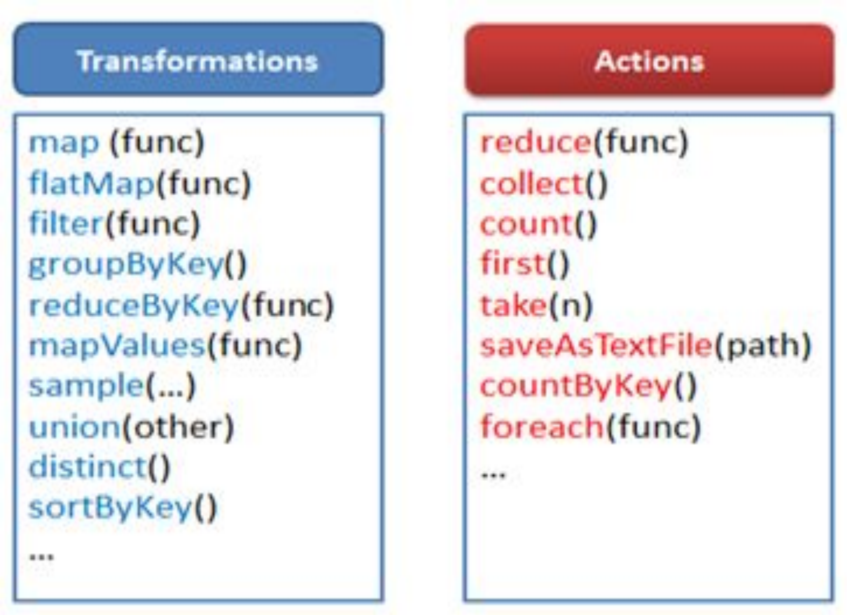


- Resilient Distributed Datasets
- Memory organization
- Immutable objects in JVM



Apache Spark

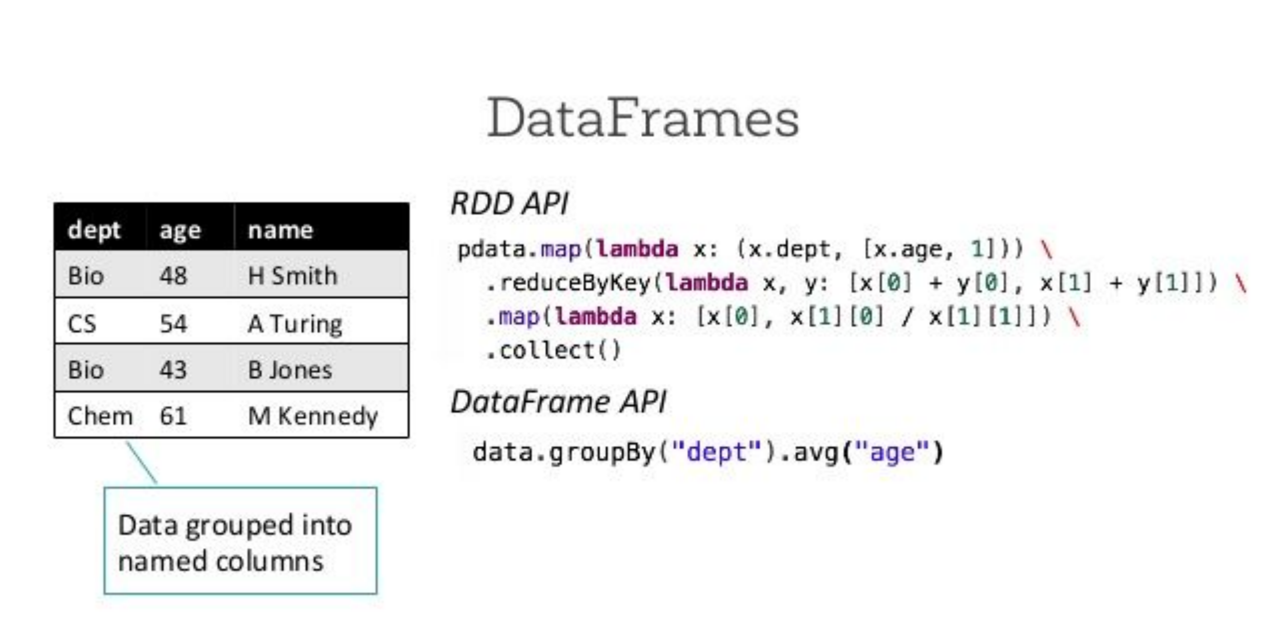
- Transformations e Actions no RDD



Transformations	Actions
map (func)	reduce(func)
flatMap(func)	collect()
filter(func)	count()
groupByKey()	first()
reduceByKey(func)	take(n)
mapValues(func)	saveAsTextFile(path)
sample(...)	countByKey()
union(other)	foreach(func)
distinct()	...
sortByKey()	
...	

Exercício

- Instalar Spark localmente
- Usar transformations e actions



DataFrames

dept	age	name
Bio	48	H Smith
CS	54	A Turing
Bio	43	B Jones
Chem	61	M Kennedy

Data grouped into named columns

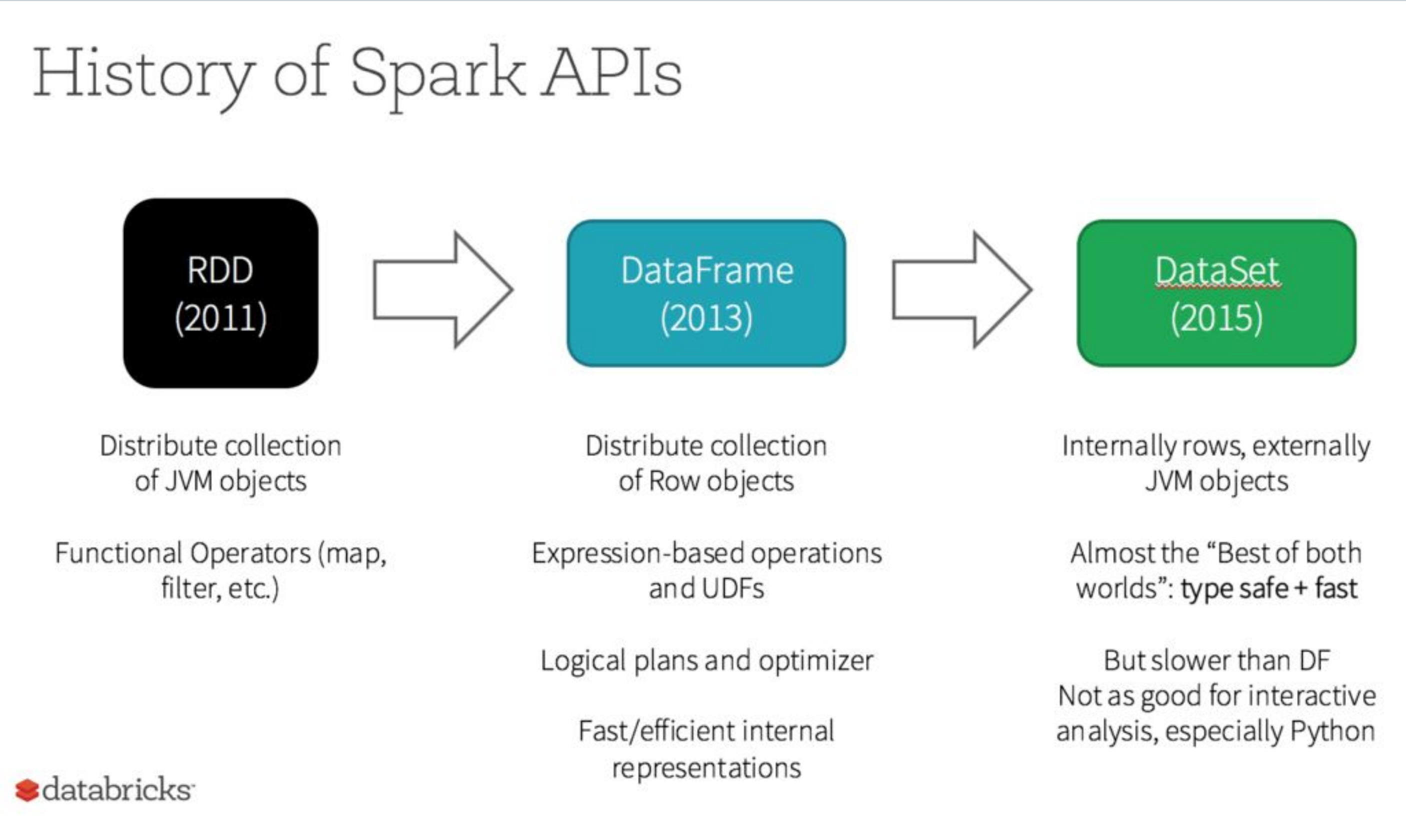
RDD API

```
pdata.map(lambda x: (x.dept, [x.age, 1])) \
  .reduceByKey(lambda x, y: [x[0] + y[0], x[1] + y[1]]) \
  .map(lambda x: [x[0], x[1][0] / x[1][1]]) \
  .collect()
```

DataFrame API

```
data.groupBy("dept").avg("age")
```

Apache Spark



History of Spark APIs

RDD
(2011)

Distribute collection
of JVM objects

Functional Operators (map,
filter, etc.)

DataFrame
(2013)

Distribute collection
of Row objects

Expression-based operations
and UDFs

Logical plans and optimizer

Fast/efficient internal
representations

DataSet
(2015)

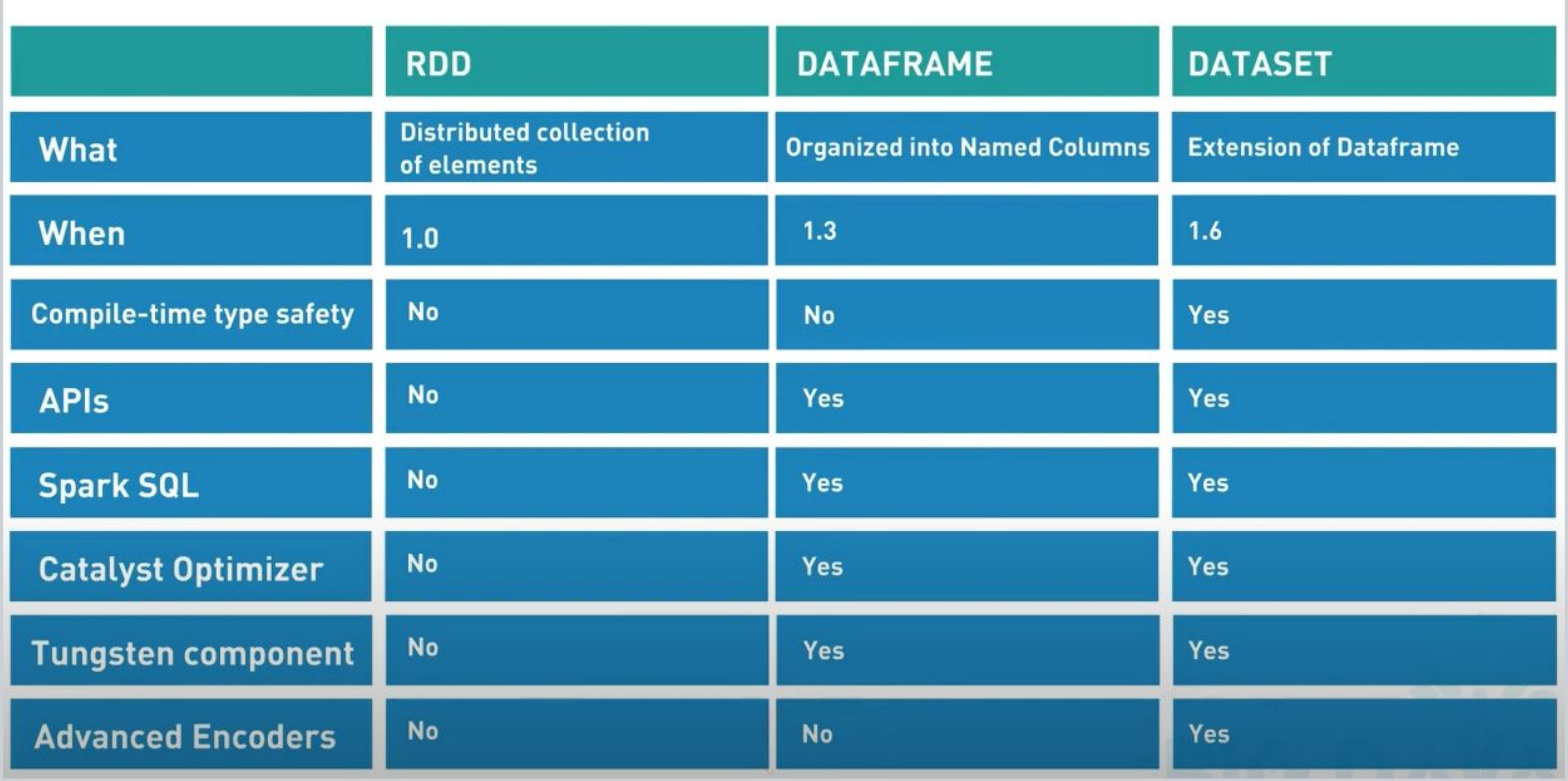
Internally rows, externally
JVM objects

Almost the "Best of both
worlds": **type safe + fast**

But slower than DF
Not as good for interactive
analysis, especially Python

databricks

Apache Spark



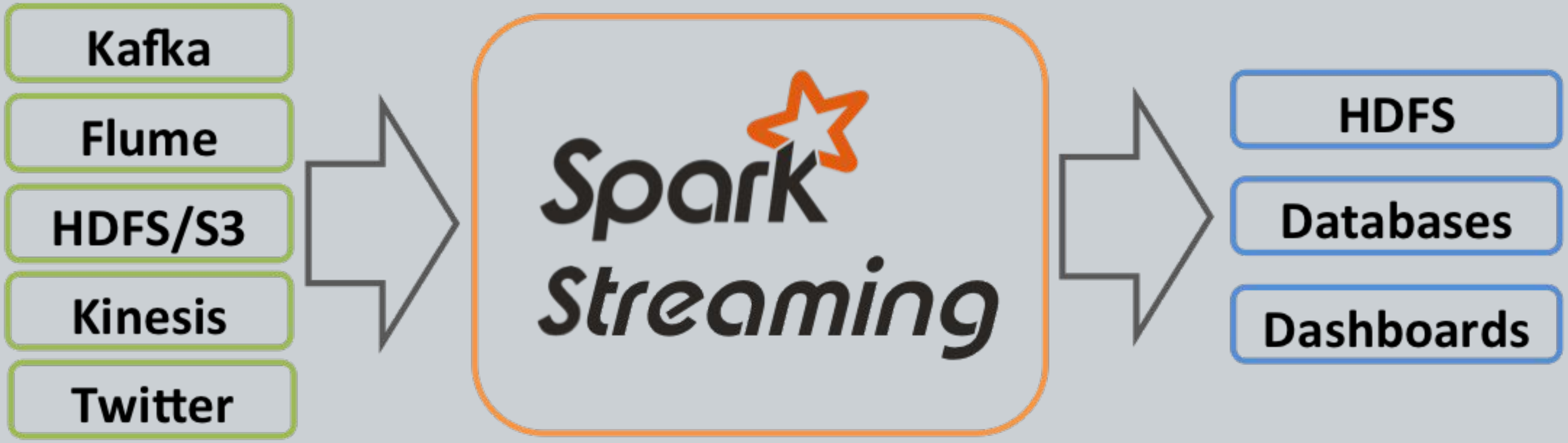
	RDD	DATAFRAME	DATASET
What	Distributed collection of elements	Organized into Named Columns	Extension of Dataframe
When	1.0	1.3	1.6
Compile-time type safety	No	No	Yes
APIs	No	Yes	Yes
Spark SQL	No	Yes	Yes
Catalyst Optimizer	No	Yes	Yes
Tungsten component	No	Yes	Yes
Advanced Encoders	No	No	Yes

https://www.youtube.com/watch?v=
26zl50iNBp8

Exercício

- Executar SQL sobre um DataSet

Spark Streaming



Kafka

Flume

HDFS/S3

Kinesis

Twitter

Spark Streaming

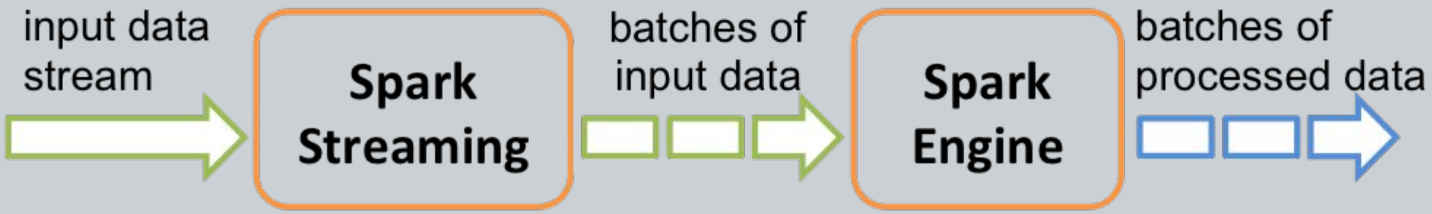
HDFS

Databases

Dashboards

https://spark.apache.org/docs/latest/streaming-programming-guide.html#discretized-streams-dstreams

Spark Streaming



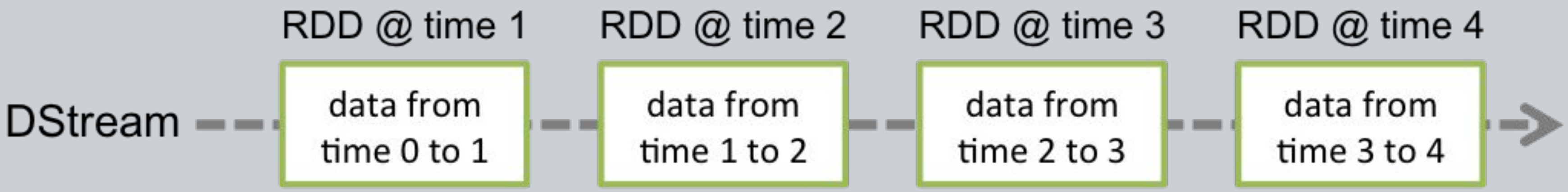
input data stream

Spark Streaming

batches of input data

Spark Engine

batches of processed data



DStream

RDD @ time 1

data from time 0 to 1

RDD @ time 2

data from time 1 to 2

RDD @ time 3

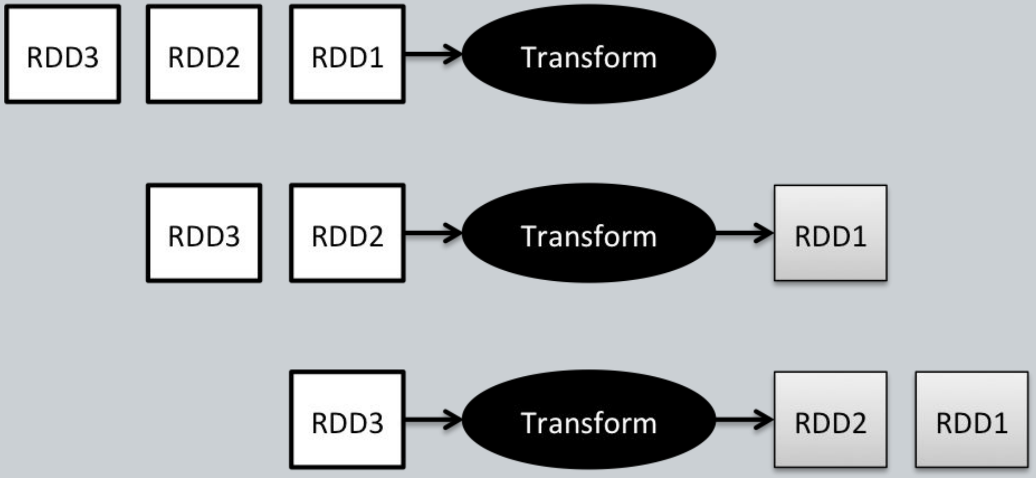
data from time 2 to 3

RDD @ time 4

data from time 3 to 4

https://spark.apache.org/docs/latest
/streaming-programming-guide.html
#discretized-streams-dstreams

Spark Streaming

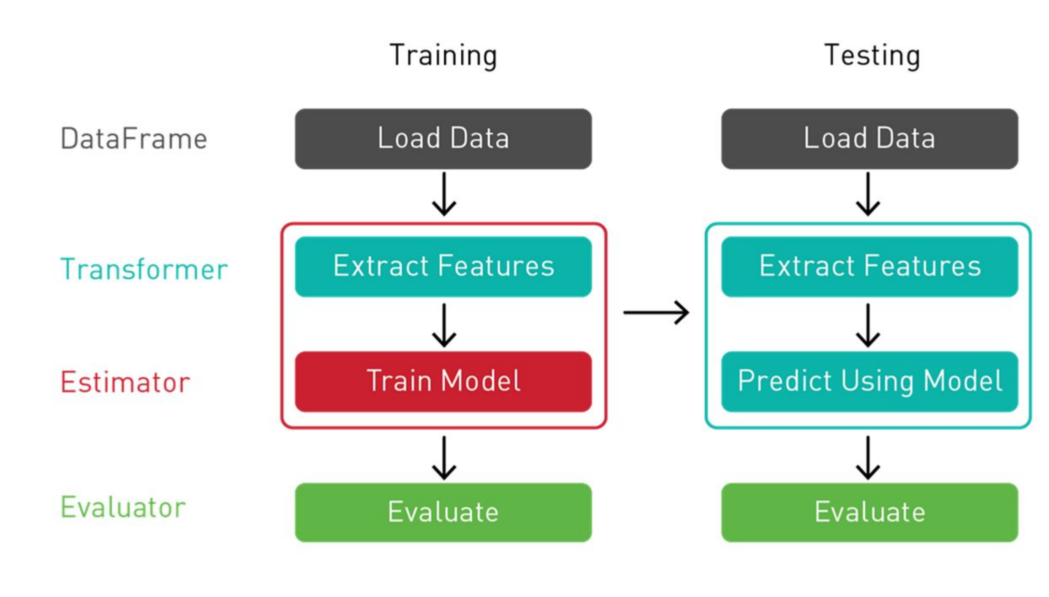


- Transform (RDD -> RDD)
  - Map
  - FlatMap
  - Filter
  - Count
  - CountByValue
  - GroupByKey
  - Reduce
  - ReduceByKey
  - Join
  - Cogroup
  - Transform
  - UpdateStateByKey

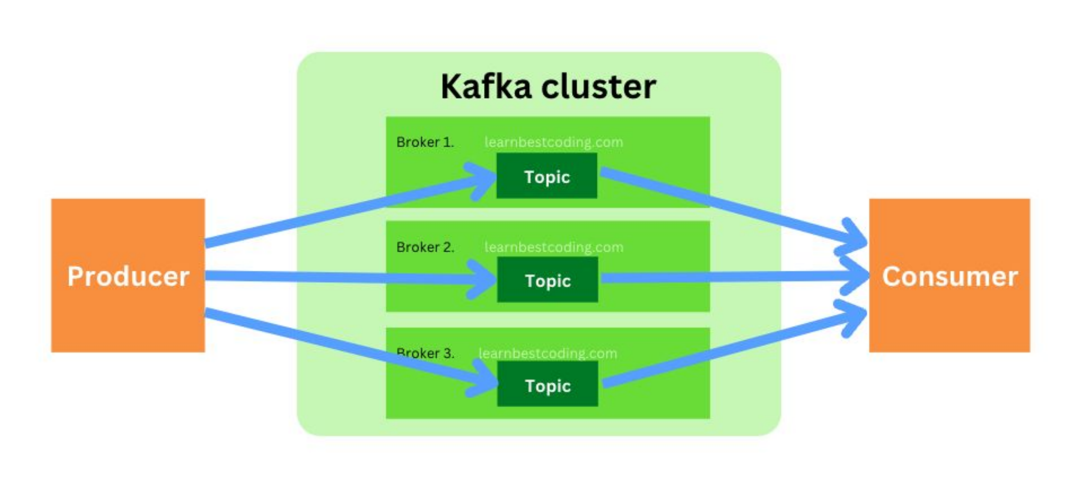
https://spark.apache.org/docs/latest
/streaming-programming-guide.html
#discretized-streams-dstreams

SparkML

- Permite usar as estruturas do Spark para executar algoritmos de Machine Learning
- Tem ferramentas para treinamento e uso de modelos



Apache Kafka



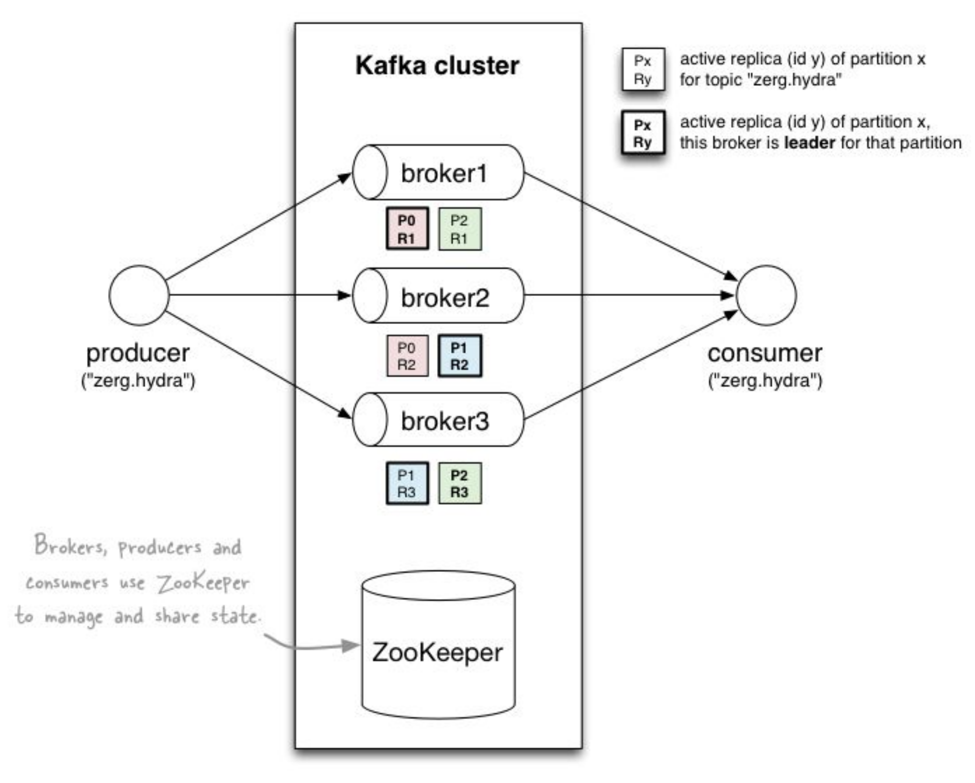
Apache Kafka
e Spring Boot

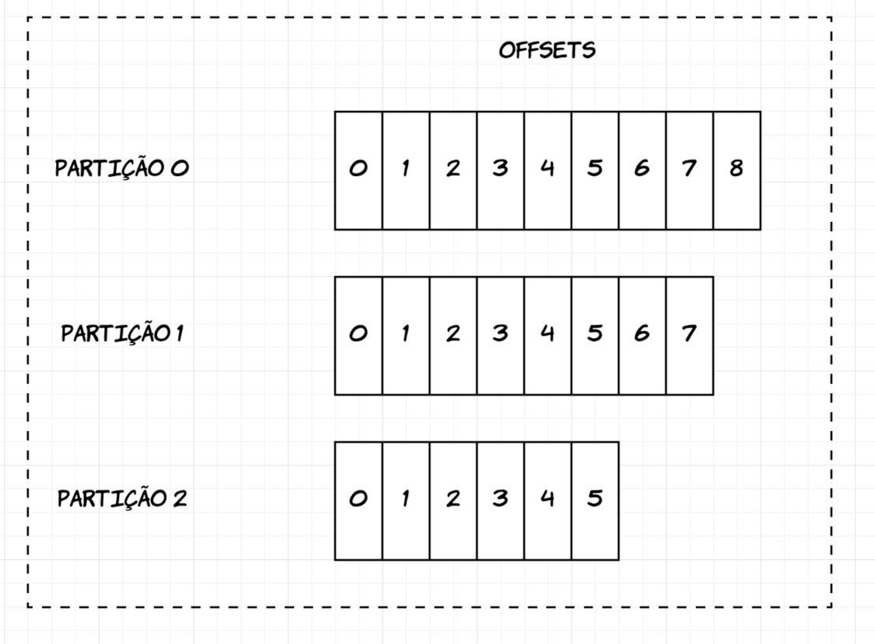
Comunicação assíncrona entre microsserviços

Casa do Código | alura

EDUARDO FELIPE ZAMBOM SANTANA

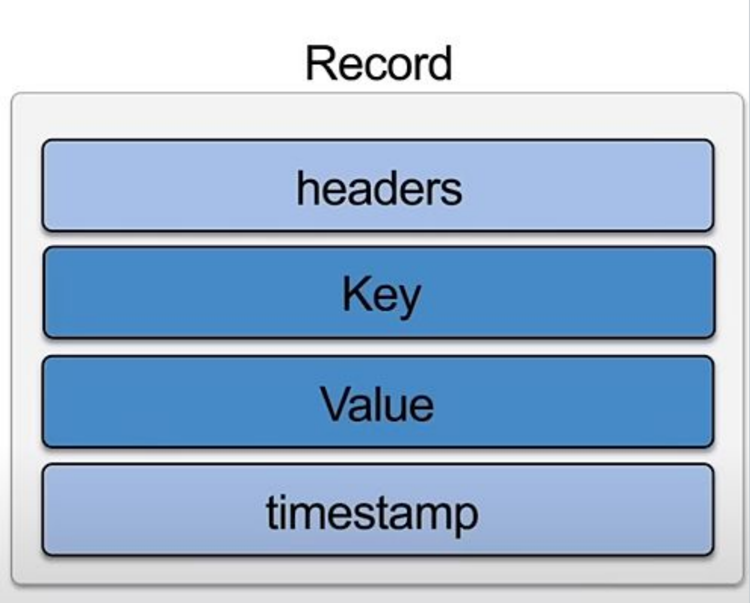
Apache Kafka


Kafka cluster
Px
Ry
active replica (id y) of partition x for topic "zerg.hydra"
Px
Ry
active replica (id y) of partition x, this broker is **leader** for that partition
broker1
P0
R1
P2
R1
broker2
P0
R2
P1
R2
broker3
P1
R3
P2
R3
producer
("zerg.hydra")
consumer
("zerg.hydra")
Brokers, producers and consumers use ZooKeeper to manage and share state.
ZooKeeper



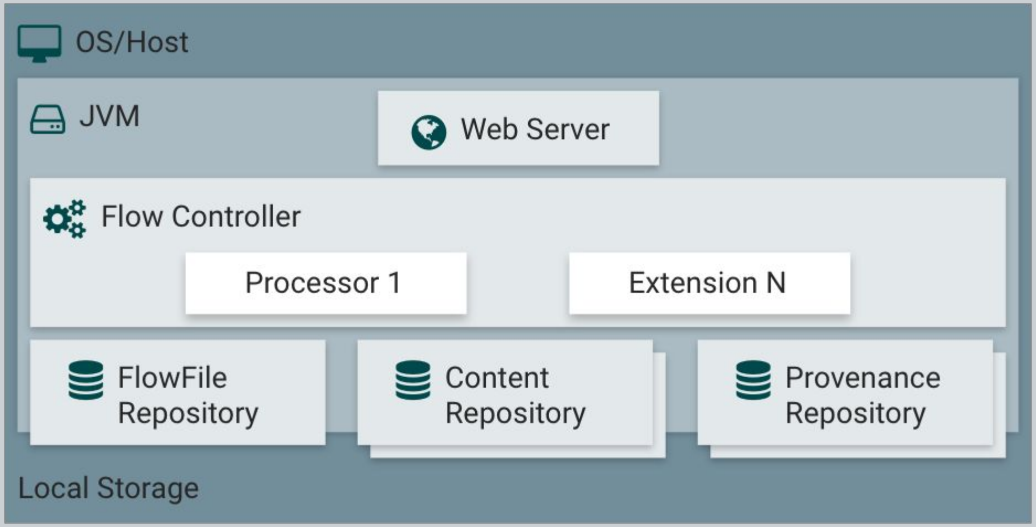
Exercício

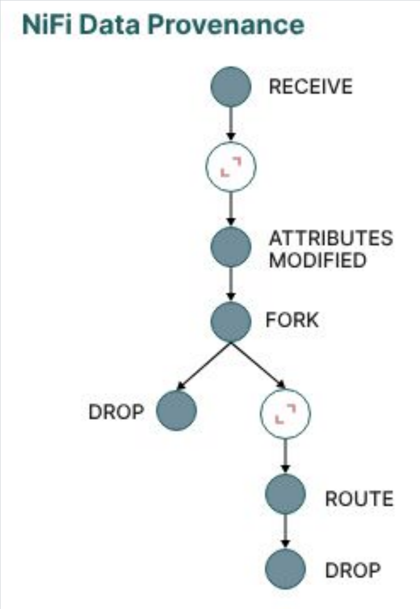
- Instalar Apache Kafka
- Criar um tópico
- Enviar uma mensagem neste tópico



Apache NiFi

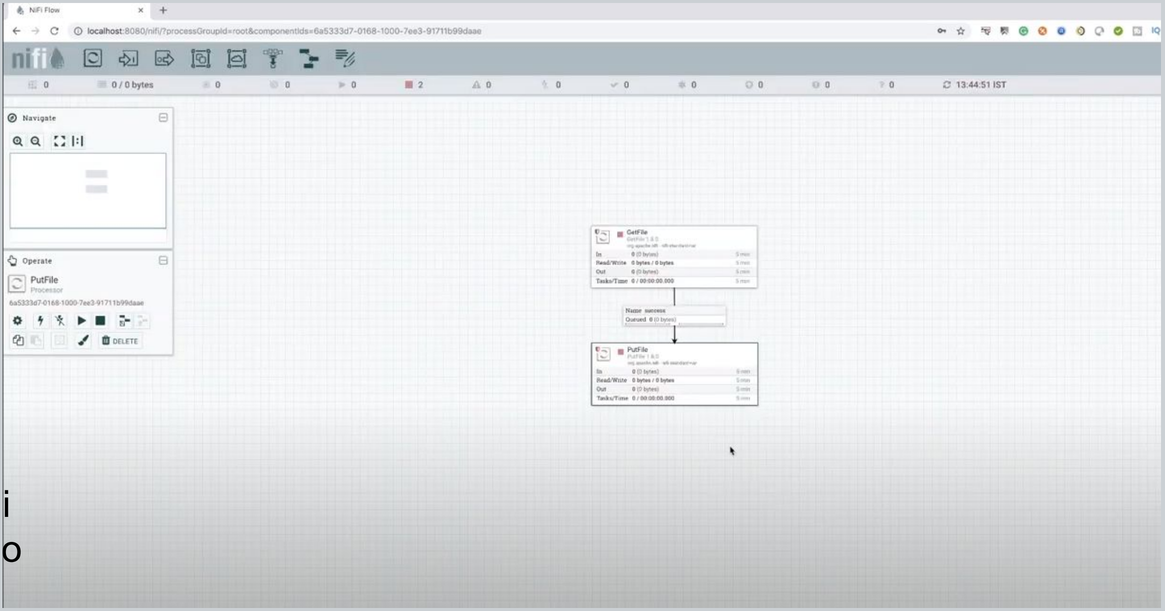
- NiFi was built to automate the flow of data between systems





Apache NiFi

- Configuration over coding
- Flow based programming
  - Processors (GetFile)
  - Data Source e Data Sink (Kafka)



NiFi Flow
localhost:8080/nifi/?processGroupId=root&componentIds=6a5333d7-0168-1000-7ee3-91711b99daae
nifi
0
0 / 0 bytes
0
0
0
2
0
0
0
0
0
0
0
13:44:51 IST
Navigate
Operate
PutFile
Processor
6a5333d7-0168-1000-7ee3-91711b99daae
DELETE

GetFile

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Name success
Queued 0 (0 bytes)

PutFile

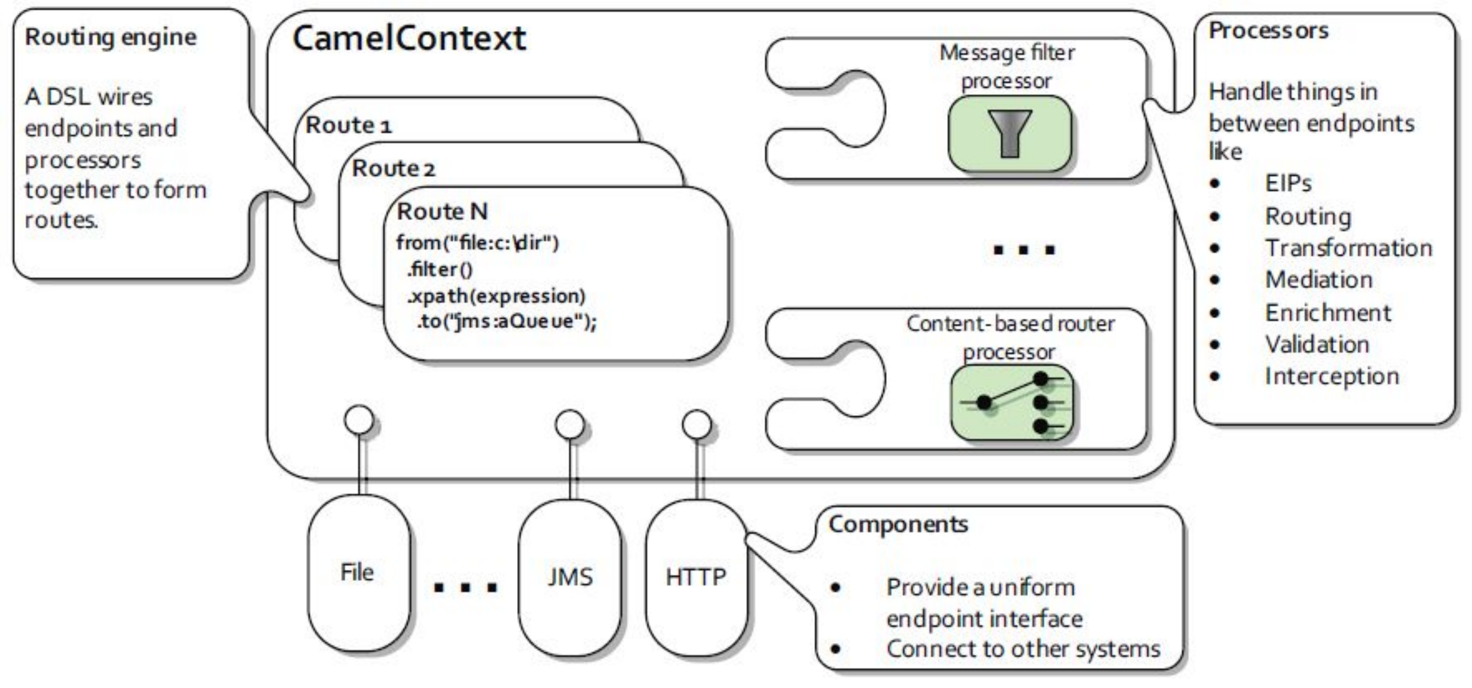
In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

https://www.youtube.com/playlist?list=PL55symSEWBbMBSnNW_Aboh2TpYkNIFMgb

Apache Camel

- Ferramenta para integração
- Muito interessante para sistemas legados
- Domain Specific Language (DSL):
  - from("file:data/inbox").to("jms:queue:order");
- Oferece um motor de roteamento altamente escalável

Apache Camel



```
from("file:c:\dir")
  .filter()
  .xpath(expression)
   .to("jms:aQueue");
```

Routing engine

A DSL wires endpoints and processors together to form routes.

Processors

Handle things in between endpoints like

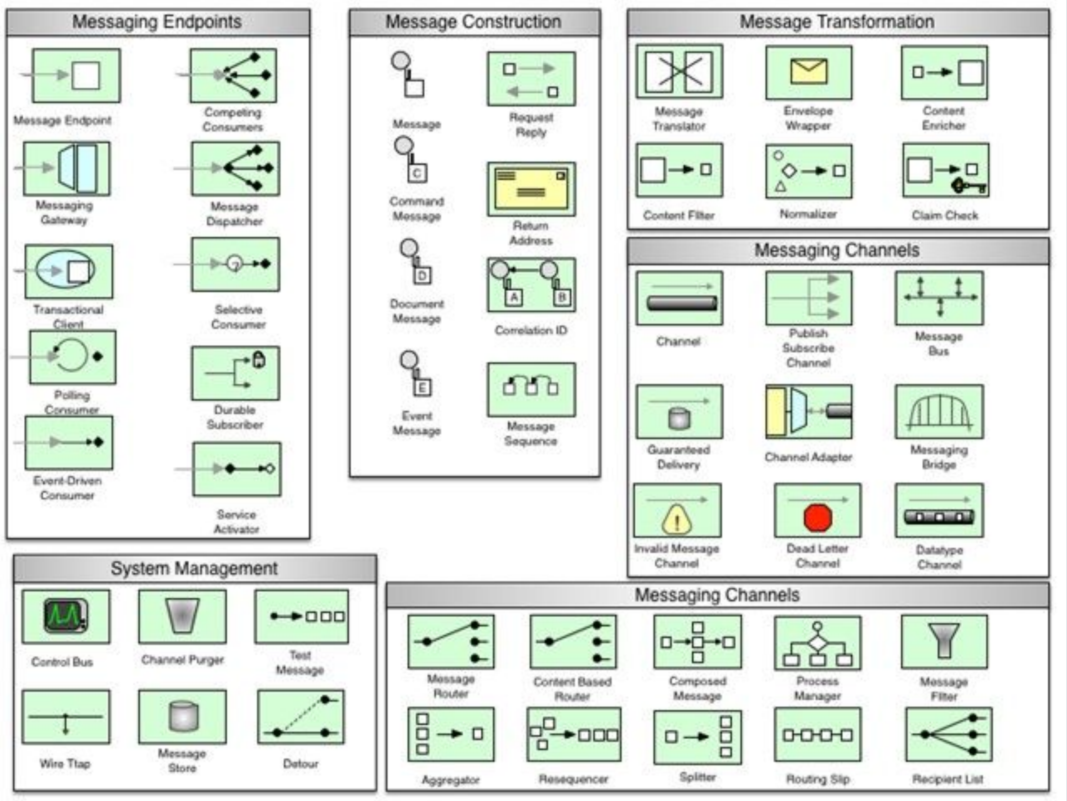
- EIPs
- Routing
- Transformation
- Mediation
- Enrichment
- Validation
- Interception

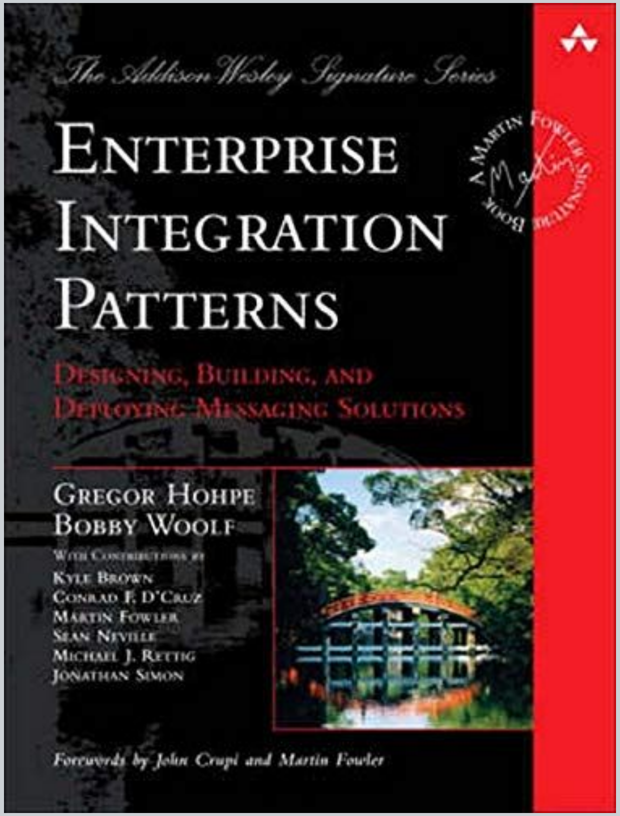
Components

- Provide a uniform endpoint interface
- Connect to other systems

Apache Camel

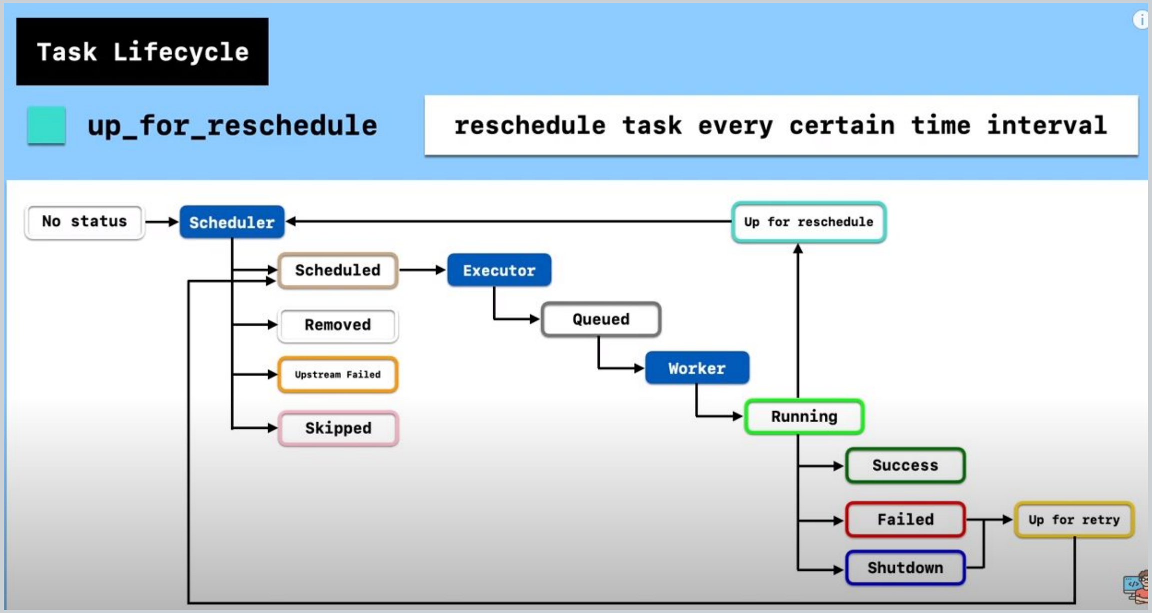
- Baseado no EIPs:




The Addison Wesley Signature Series
ENTERPRISE INTEGRATION PATTERNS
A MARTIN FOWLER SIGNATURE BOOK
DESIGNING, BUILDING, AND DEPLOYING MESSAGING SOLUTIONS
GREGOR HOHPE
BOBBY WOOLF
WITH CONTRIBUTIONS BY
KYLE BROWN
CONRAD F. D'CRUZ
MARTIN FOWLER
SEAN NEVILLE
MICHAEL J. RETTIG
JONATHAN SIMON
Forewords by John Crupi and Martin Fowler

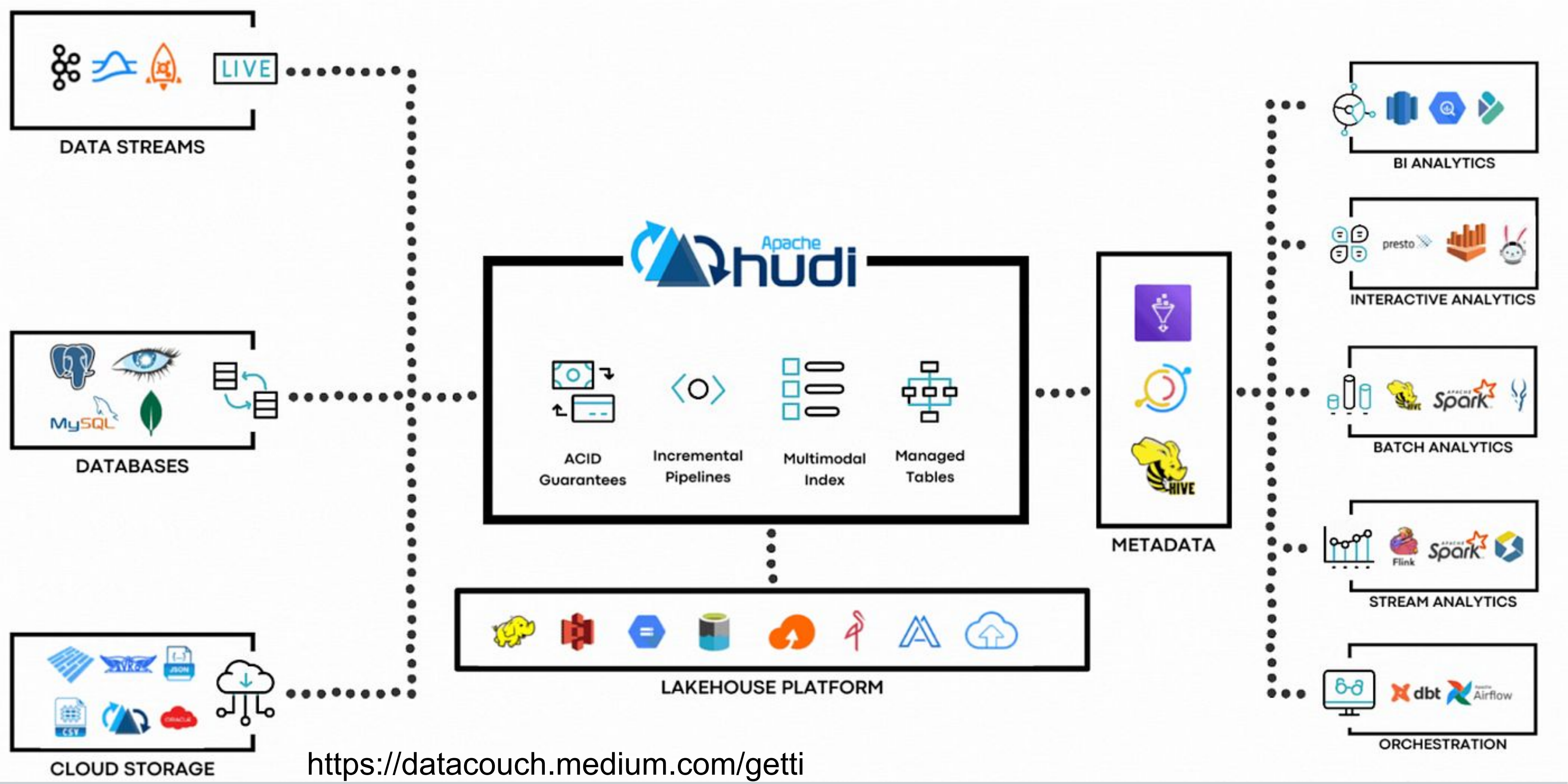
Apache AirFlow

- Criado para o gerenciamento de workflows complexos
- Criado no AirBnb
- Também usa do conceito de DAG
  - Tarefas escritas em Python
  - Operadores



https://www.youtube.com/watch?v=K9AnJ9_ZAXE&list=PLwFJcsJ61oujAqYpMp1kdUBcPG0sE0QMT

Apache Hudi



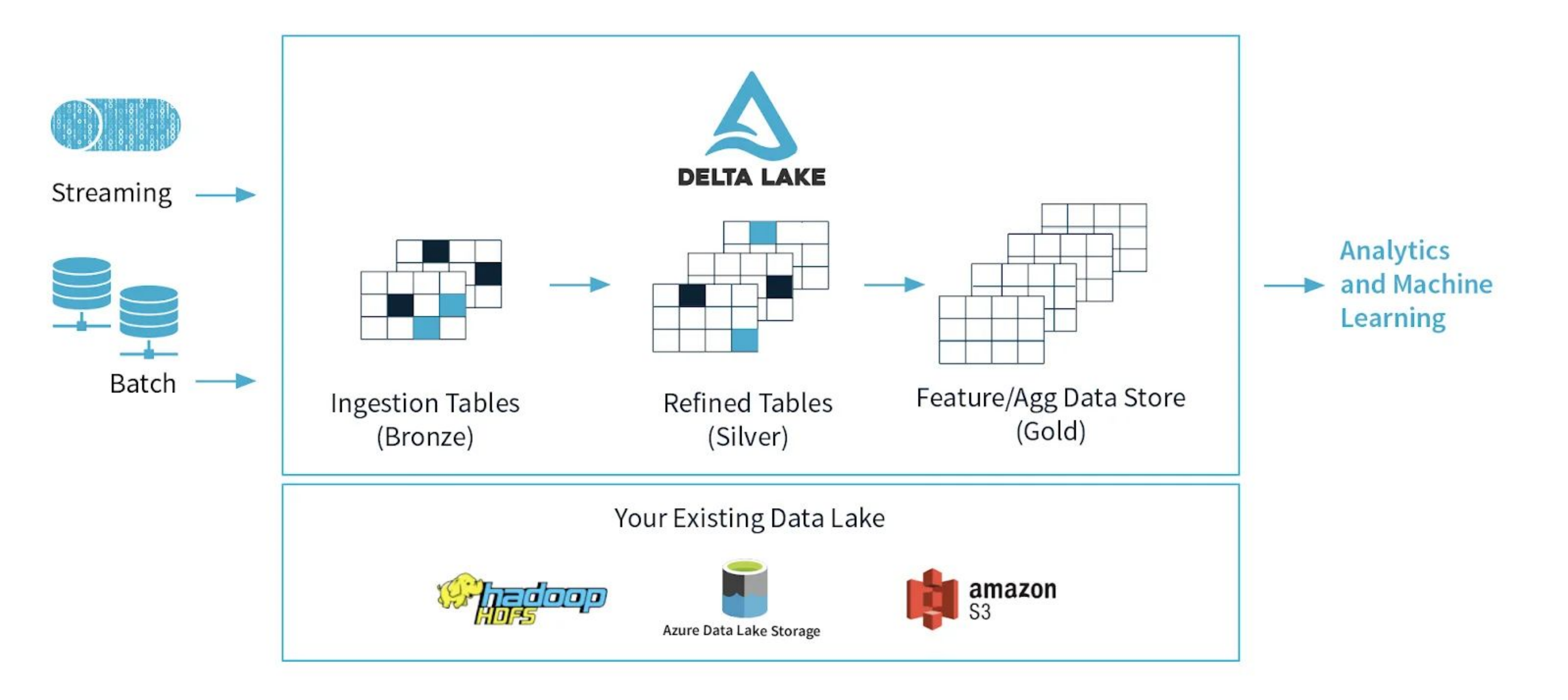
https://datacouch.medium.com/getting-started-with-apache-hudi-711b89c107aa

Delta Lake

- Camada open-source em cima de data lakes
- Trasações ACID
- Time travel

https://www.youtube.com/watch?v=
LJtShrQqYZY

Delta Lake



https://www.youtube.com/watch?v=LJtShrQqYZY

Delta Lake

- Camada open-source em cima de data lakes
- Trasações ACID
- Time travel

https://www.youtube.com/watch?v=
LJtShrQqYZY

Luiz Henrique Zambom Santana

lhzsantana@gmail.com

https://www.linkedin.com/in/luizsantana/

UNIVERSIDADE FEDERAL
DE SANTA CATARINA