UdeA
Analytics &
DataScience

PROGRAMACIÓN SOBRE GRANDES VOLUMENES DE DATOS

FRAMEWORKS PARA BIG DATA

Magister - Efraín Alberto Oviedo
eaoc46@gmail.com

UNIVERSIDAD DE ANTIOQUIA
1 8 0 3

UNIVERSIDAD DE ANTIOQUIA
FACULTAD DE INGENIERÍA
ESPECIALIZACIÓN EN ANALÍTICA Y CIENCIA DE DATOS

AGENDA

UdeA
Analytics &
DataScience

1. **Hadoop**
   - ❑ HDFS
   - ❑ Map Reduce
   - ❑ Otros componentes
2. Spark
   - ❑ Arquitectura
   - ❑ Lenguajes

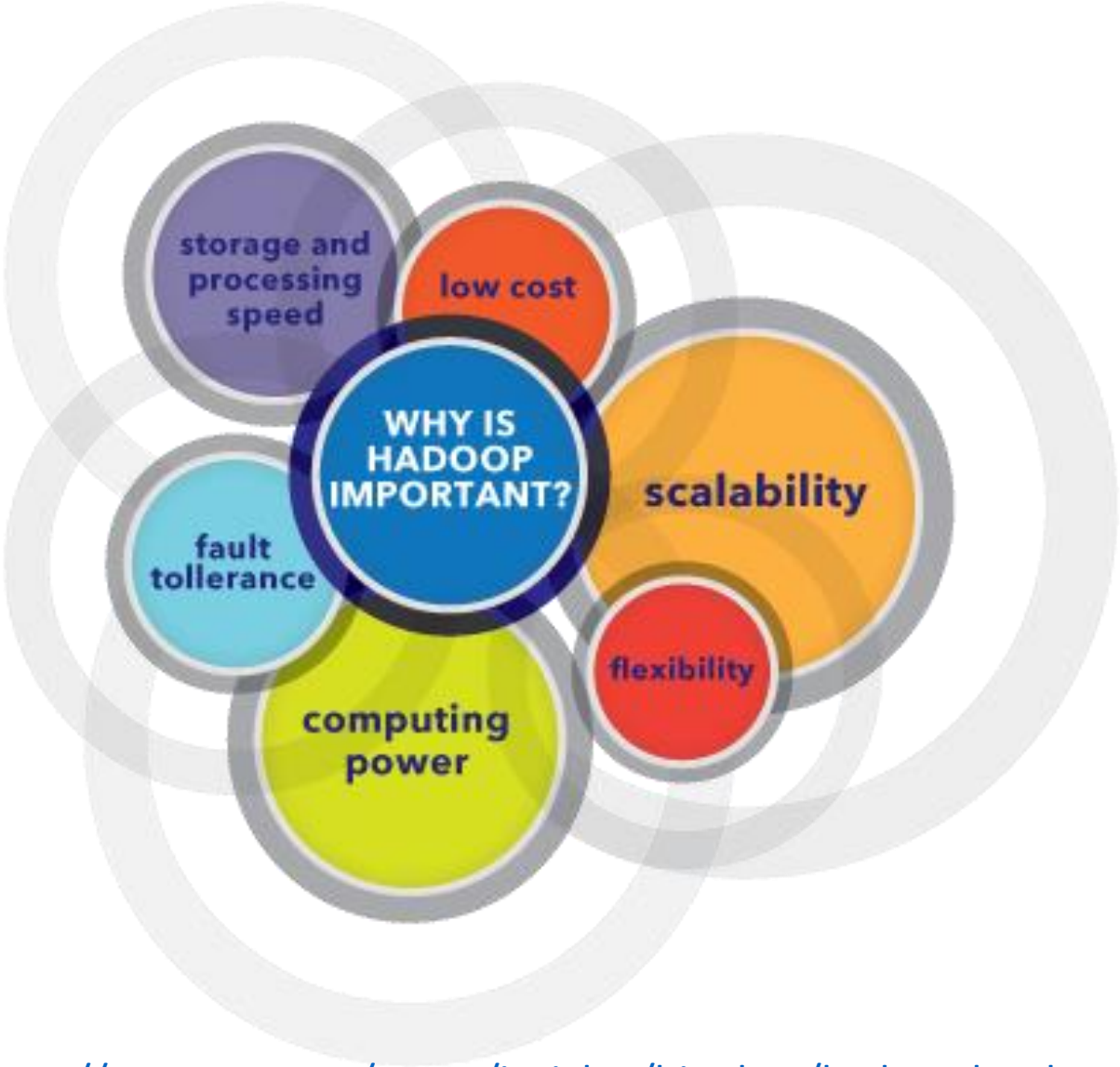
Hadoop

Para muchos es considerado un **sinónimo de Big Data**

- Proyecto de código abierto desarrollado en Lenguaje Java que tuvo sus inicios en 2006
- Plataforma para almacenamiento y procesamiento de grandes volúmenes de datos de forma **escalable** y **distribuida**
- Permite ejecutar código en Clústers

Hadoop era el nombre de un elefante de juguete amarillo propiedad del hijo de uno de sus inventores

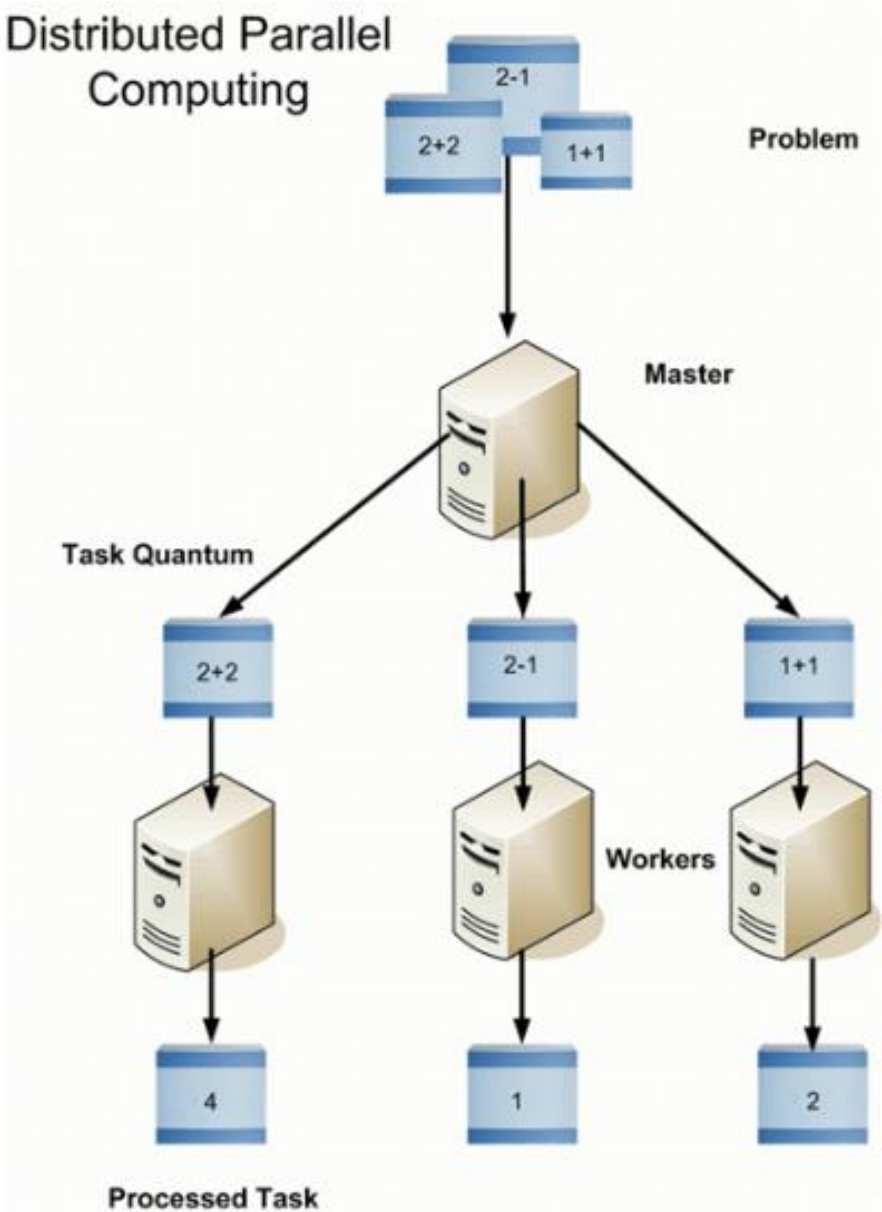
Hadoop

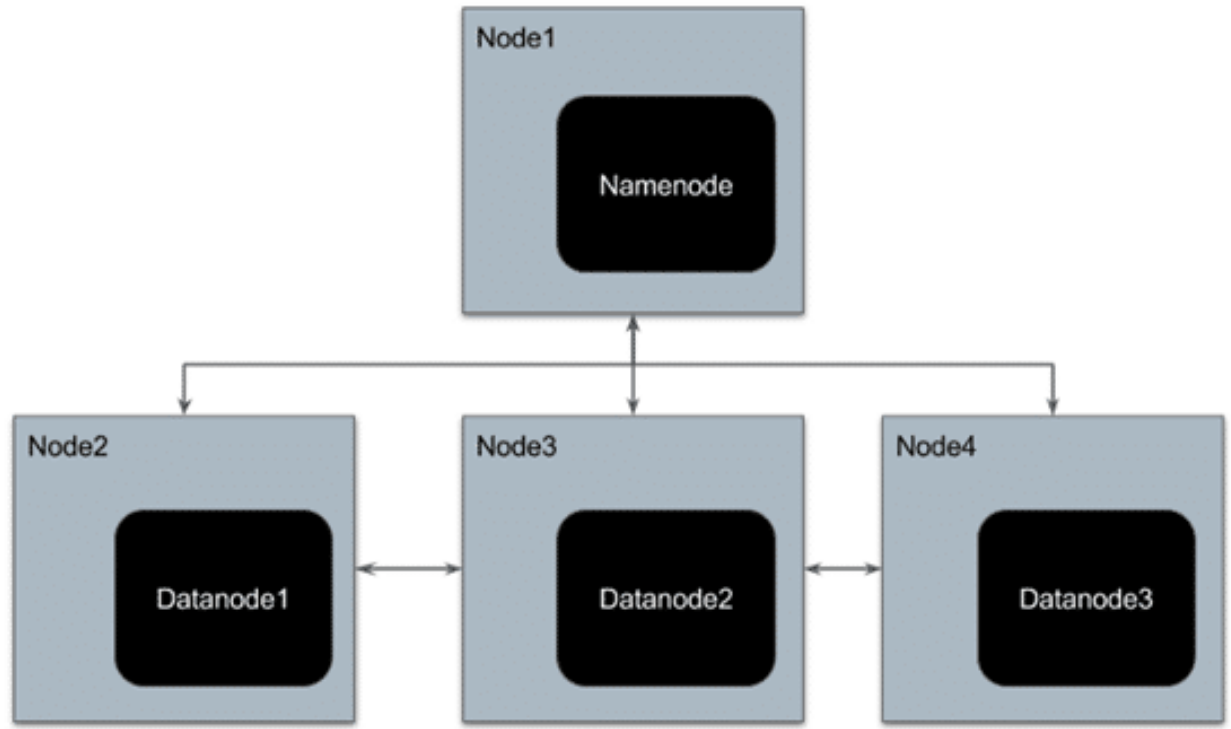


https://www.sas.com/es_pe/insights/big-data/hadoop.html

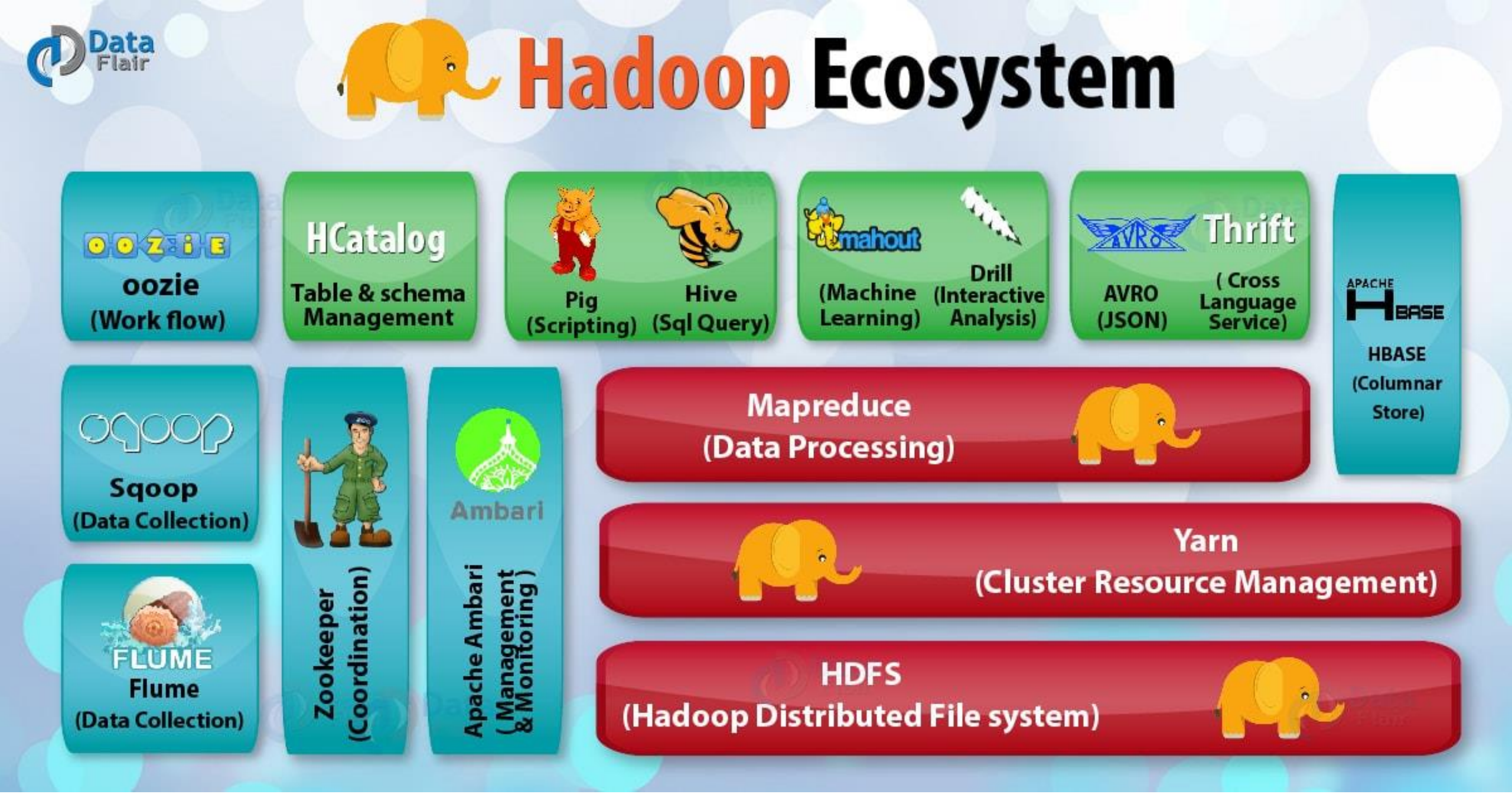
- **Capacidad de almacenar y procesar** enormes cantidades de cualquier tipo de datos, al instante.
- **Poder de cómputo.** procesa big data a gran velocidad. Cuantos más nodos de cómputo utiliza usted, mayor poder de procesamiento tiene.
- **Tolerancia a fallos.** Si falla un nodo, los trabajos son redirigidos automáticamente a otros modos para asegurarse de que no falle el procesamiento distribuido.
- **Flexibilidad.** Puede almacenar tantos datos como desee y decidir cómo utilizarlos más tarde. Eso incluye datos no estructurados como texto, imágenes y videos.
- **Bajo costo.** La estructura de código abierto es gratuita y emplea hardware comercial para almacenar grandes cantidades de datos.
- **Escalabilidad.** Puede hacer crecer fácilmente su sistema para que procese más datos son sólo agregar nodos. Se requiere poca administración.

Hadoop





https://dimensionless.in/know-how-to-install-and-run-hadoop-on-windows-for-beginners/



https://data-flair.training/blogs/hadoop-ecosystem-components/

AGENDA

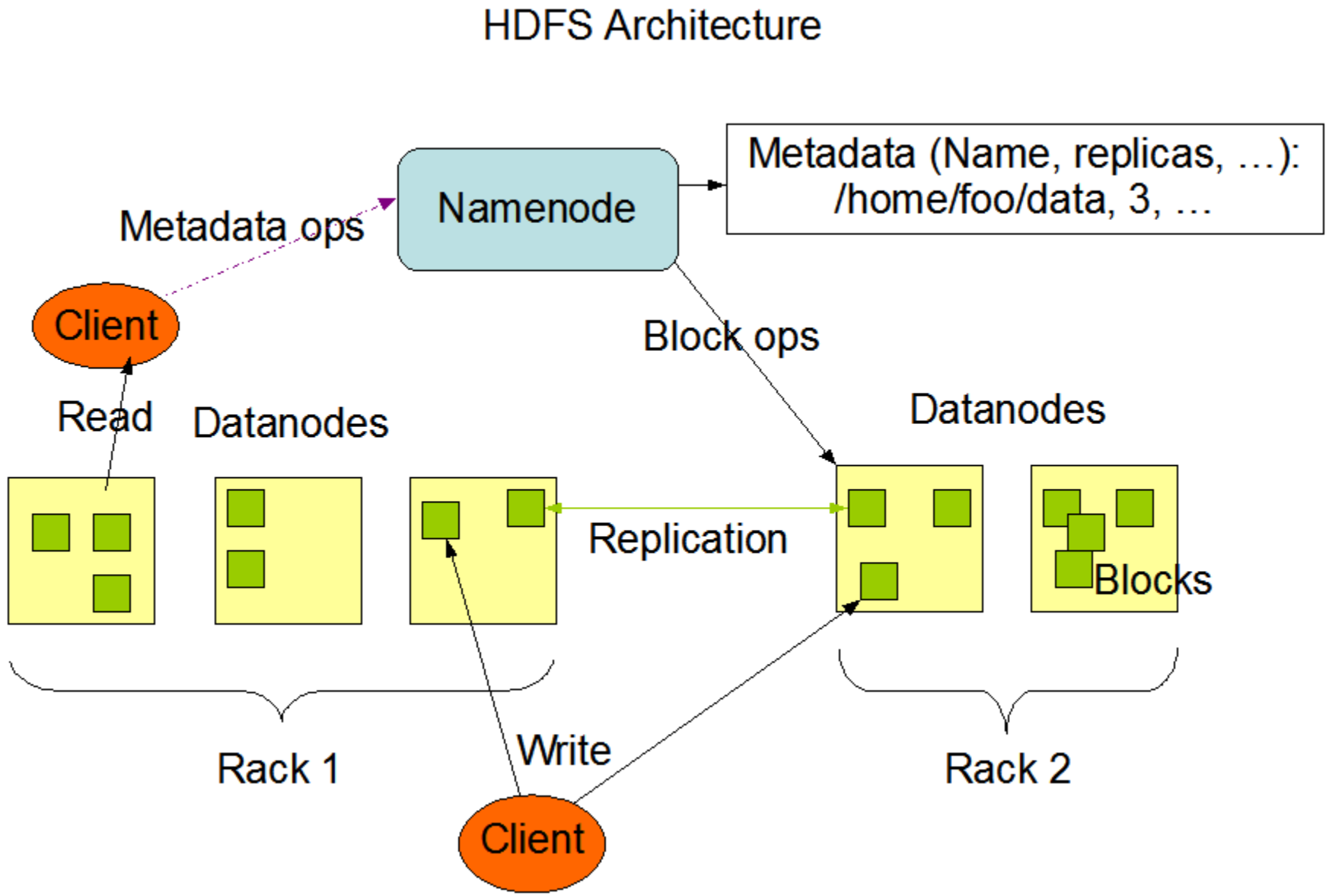
UdeA
Analytics & DataScience

1. Hadoop
   - ❑ **HDFS**
   - ❑ Map-reduce
   - ❑ Otros componentes
2. Spark
   - ❑ Arquitectura
   - ❑ Lenguajes

HDFS

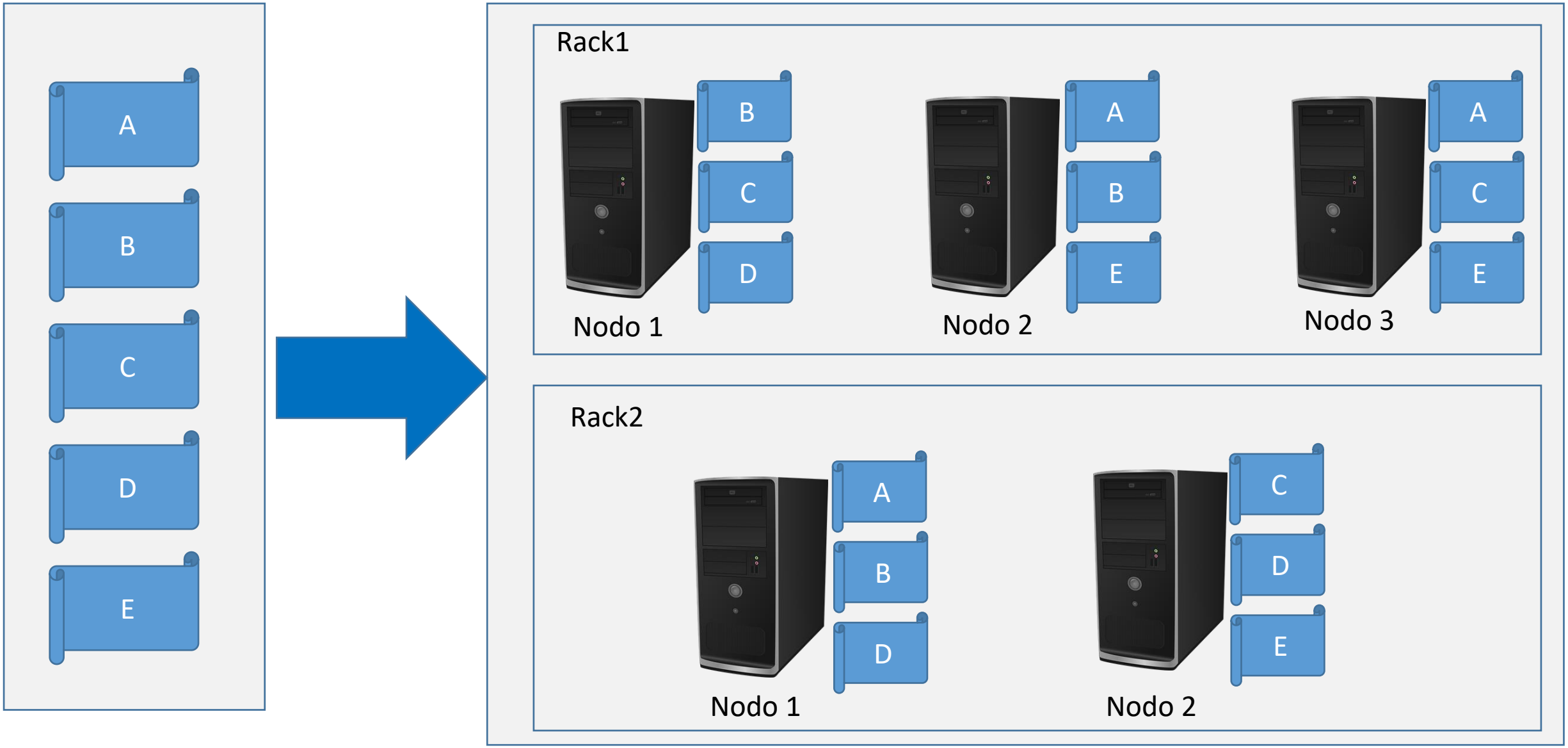
Hadoop Distributed File System (HDFS)

- Permite almacenar grandes volúmenes de datos con redundancia
- Se pueden almacenar muchos mas datos de forma distribuida que en una sola máquina
- Si uno de las máquinas falla el sistema lo resuelve

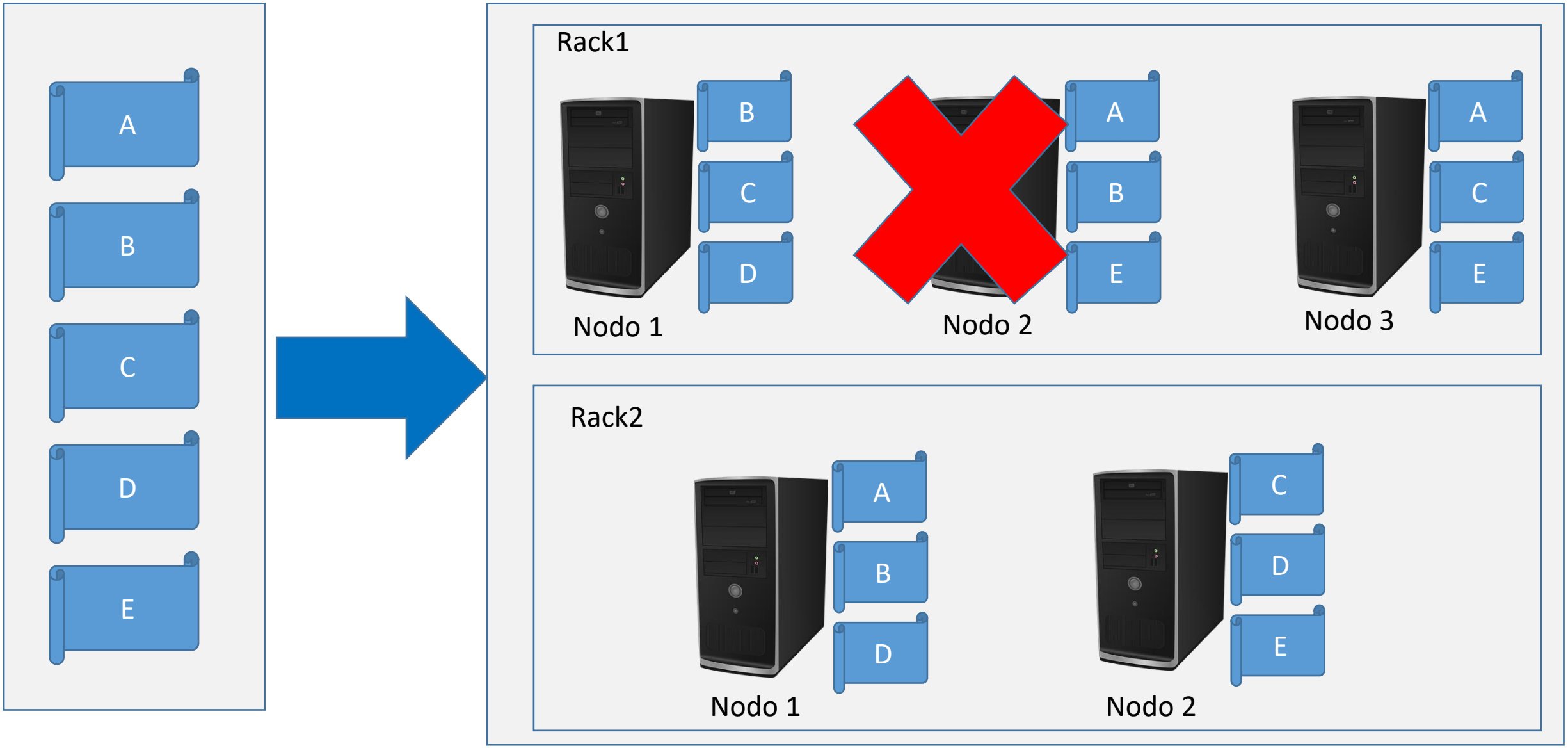


http://hadoop.apache.org/docs/stable1/hdfs_design.pdf

HDFS - Ejemplo



HDFS - Ejemplo



HDFS

Ejercicio

Lea el documento de arquitectura de HDFS disponible en:

http://hadoop.apache.org/docs/stable1/hdfs_design.pdf

Y responda el cuestionario

HDFS

Cuestionario:

1. Normalmente, de qué tamaño son los archivos que se almacenan en el sistema HDFS?
2. Qué es mas recomendado, mover la aplicación hacia donde están los datos o mover los datos hacia donde está la aplicación?
3. Cuál es el sistema operativo utilizado típicamente para soluciones HDFS?
4. Si un usuario solicita un archivo que tiene réplicas en varios nodos, cual es el nodo preferencial para responder la solicitud?
5. El DataNode almacena todos los archivos en el mismo directorio? Porque?
6. Cuáles son las fallas mas comunes en un sistema HDFS?
7. Cómo se comprueba la integridad de un archivo HDFS?
8. Al almacenar un archivo quién se encarga de transferir los bloques de datos a los DataNode que van a contener réplicas?
9. Es posible recuperar un archivo eliminado en el sistema HDFS?

AGENDA

UdeA
Analytics & DataScience

1. Hadoop
   - ❑ HDFS
   - ❑ **Map-reduce**
   - ❑ Otros componentes
2. Spark
   - ❑ Arquitectura
   - ❑ Lenguajes

Map Reduce

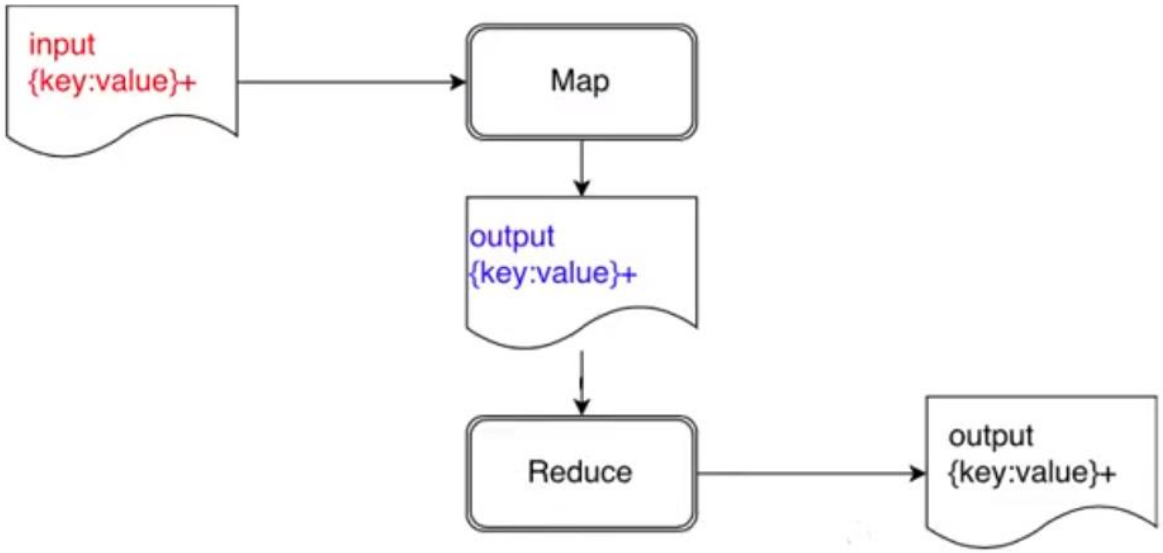
- Metodología de programación dada a conocer por Google orientada a procesamiento en paralelo.

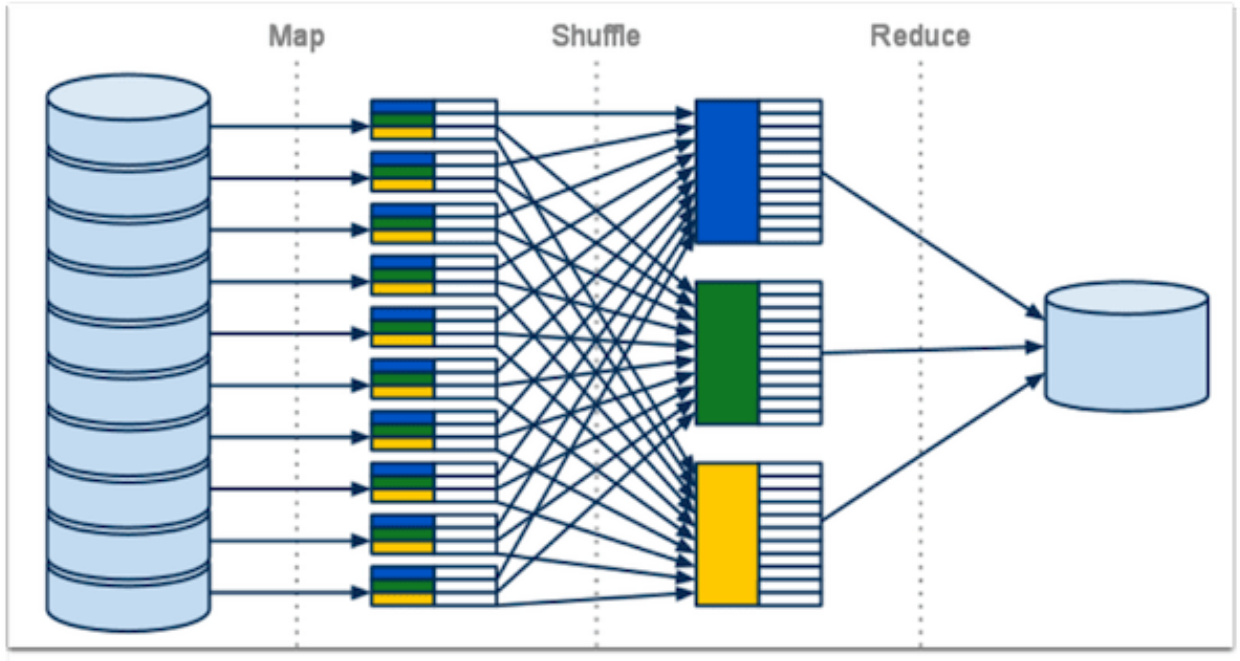
- Consiste en dividir la aplicación en pequeños fragmentos de trabajo donde cada uno de ellos puede ser ejecutado en un nodo

- Etapas
  - Split: Divide los datos en múltiples fracciones de datos
  - Map: Procesamiento de cada nodo
  - Shuffle & Sort: Ordenar los resultados
  - Redduce: Compactar y resumir los resultados

Map Reduce

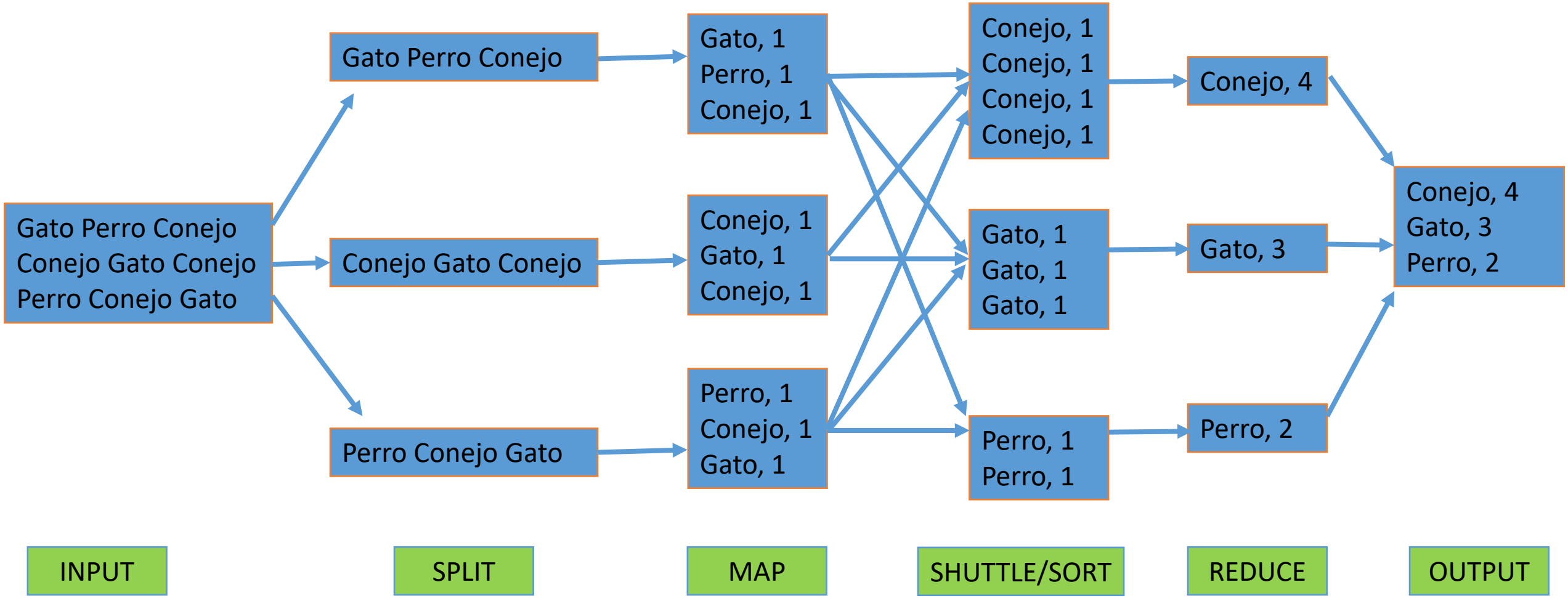
Esquema Map Reduce





Map Reduce

Ejemplo: Contador de Palabras



```
public static void main(String[] args) throws Exception {
  Configuration conf = new Configuration();
  Job job = Job.getInstance(conf, "word count");
  job.setJarByClass(WordCount.class);
  job.setMapperClass(TokenizerMapper.class);
  job.setCombinerClass(IntSumReducer.class);
  job.setReducerClass(IntSumReducer.class);
  job.setOutputKeyClass(Text.class);
  job.setOutputValueClass(IntWritable.class);
  //arg[0] es el nombre de la Aplicación
  FileInputFormat.addInputPath(job, new Path(args[1]));
  FileOutputFormat.setOutputPath(job, new Path(args[2]));
  System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```

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

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

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
}
```

```
public static class IntSumReducer
    extends Reducer<Text,IntWritable,Text,IntWritable> {
  private IntWritable result = new IntWritable();

  public void reduce(Text key, Iterable<IntWritable> values,
                     Context context
                     ) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
      sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
  }
}
```

Map Reduce

What is MapReduce used for?

- At Google
  - index construction for Google Search
  - article clustering for Google News
  - statistical machine translation
  - distributed grep, distributed sort
  - web access log stats
  - web link-graph reversal
  - inverted index construction
- At Yahoo!
  - "Web map" powering Yahoo! Search
  - Spam detection for Yahoo! Mail
- At Facebook
  - Data mining
  - Ad optimization
  - Spam detection

https://ethz.ch/content/dam/ethz/special-interest/gess/computational-social-science-dam/documents/education/Spring2017/ML/big_data.pdf

AGENDA

UdeA
Analytics &
DataScience

1. Hadoop
   - ❑ HDFS
   - ❑ Map-reduce
   - ❑ **Otros componentes**
2. Spark
   - ❑ Arquitectura
   - ❑ Lenguajes

HBase

- Base de datos NoSQL open source desarrollada en Java
- Hace parte del proyecto Hadoop, se ejecuta sobre HDFS
- Es una base de datos Key-value orientada a columnas
- El acceso a los datos se puede hacer desde Pig, Hive

Pig

- Plataforma para analizar y consultar grandes volúmenes de datos almacenados en HDFS. Permite crear programas Map Reduce

- Utiliza el lenguaje PigLatin (similar a SQL)

- Al escribir un Script en PigLatin este es automáticamente paralelizado y distribuido en un clúster

Pig Latin

Operador	Descripción
FILTER	Selecciona un conjunto de tuplas de una relación basado en una condición.
FOREACH	Itera las tuplas de una relación generando una transformación de datos.
GROUP	Agrupa los datos en una o más relaciones
JOIN	Une dos o más relaciones (unión interna o externa)
LOAD	Carga datos desde el sistema de archivos
ORDER	Clasifica una relación basado en uno o más campos.
SPLIT	Particiona una relación en una o más relaciones.
STORE	Almacena datos en el sistema de archivos

https://www.ibm.com/developerworks/ssa/library/l-apachepigdataquery/index.html

Pig – Contador de Palabras

HADOOP

```
hdfs dfs -copyFromLocal C:\BigData\test_data\animales.txt /ejemplo2
```

PIG

```
//Lee el archivo
data = load '/ejemplo2/animales.txt' as (line);
//lee por palabras
words = foreach data generate flatten (TOKENIZE(line)) as word;
//agrupa cada palabra
grpd = group words by word;
//Cuenta las palabras
cntd = foreach grpd generate group,COUNT(words);
//muestra el resultado
dump cntd;
```

Hive

- Infraestructura de data warehouse de Hadoop (No es BD)
- Realiza funciones de agrupación, consulta y análisis de datos, almacenados bajo HDFS
- Implementa las queries mediante HiveQL (HQL)
- Traduce consultas tipos SQL a trabajos Map Reduce
- Instalación:

https://mauricioanderson.com/hive-introduccion-instalacion/

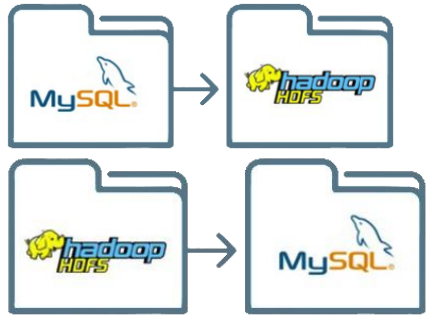
Impala

- Motor de consultas implementado de forma nativa sobre Hadoop
- Utiliza lenguaje SQL
- Soporta almacenamiento de datos en HDF y Hbase
- Proporciona alto rendimiento y baja latencia
- No tiene tolerancia a fallas

Sqoop

- Sqoop (SQL + Hadoop): Herramienta de línea de comandos para transferir datos entre bases de datos relacionales y Hadoop

- Permite importar y exportar datos
  - Transformar datos de una BD relacional a HDFS (Hive o Hbase)
  - Transformar datos de HDFS a BD relacionales (MySQL, Oracle, etc)



- Instalación:

https://www.youtube.com/watch?v=eePrk-UA_gg

Collaborative Filtering *with CLI drivers*

User-Based Collaborative Filtering

Item-Based Collaborative Filtering

Matrix Factorization with ALS

Matrix Factorization with ALS on Implicit Feedback

Weighted Matrix Factorization, SVD++

Clustering *with CLI drivers*

Canopy Clustering

k-Means Clustering

Fuzzy k-Means

Streaming k-Means

Spectral Clustering

Classification *with CLI drivers*

Logistic Regression - trained via SGD

Naive Bayes / Complementary Naive Bayes

Hidden Markov Models

https://mahout.apache.org/users/basics/algorithms.html

Hue

- Editor Open Source que permite consultar y visualizar datos en el ecosistema de Hadoop

- Permite interactuar con aplicaciones HDFS y Map Reduce

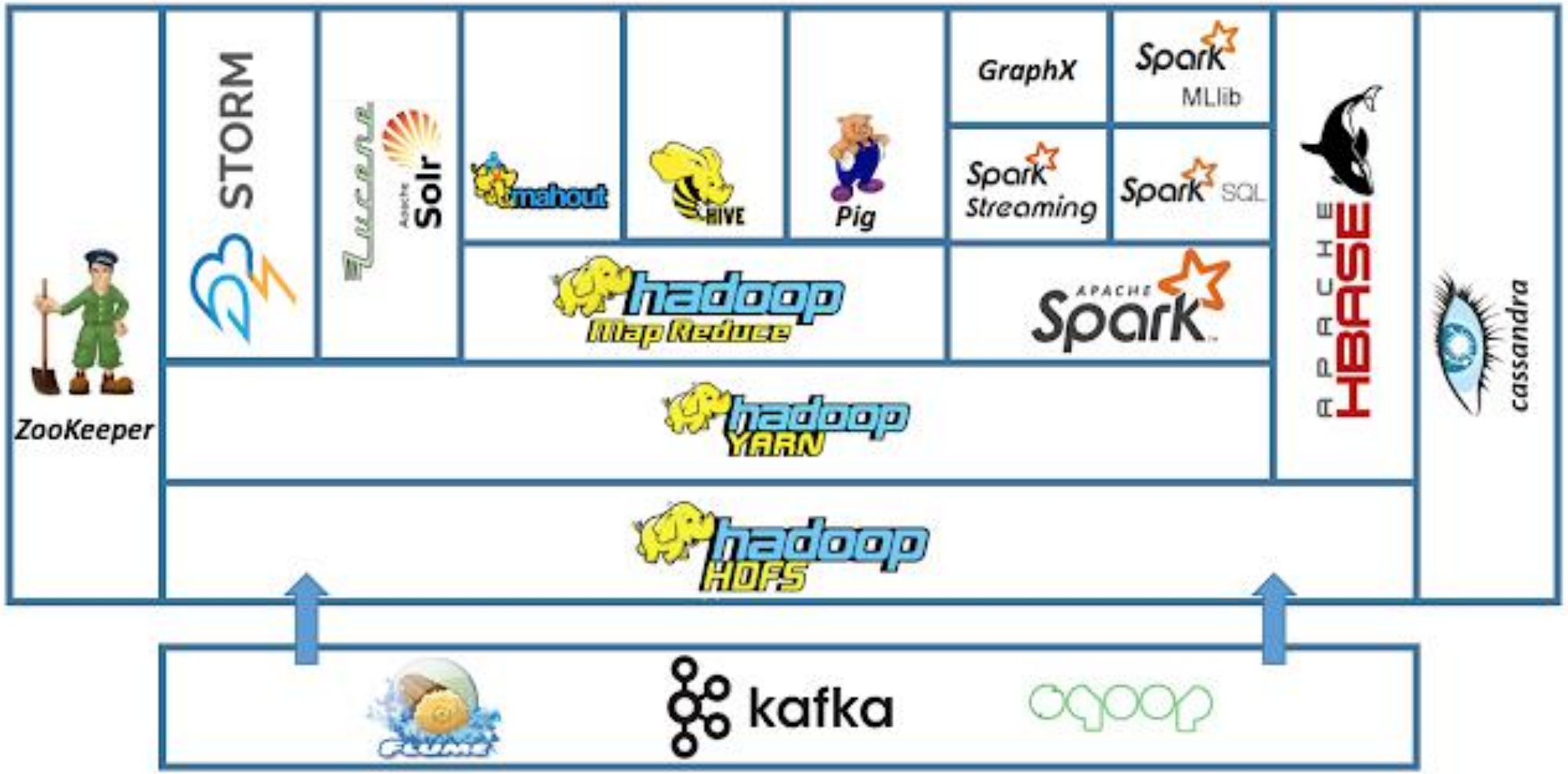
- Demo: https://demo.gethue.

AGENDA

UdeA
Analytics &
DataScience

1. Hadoop
   - ❑ HDFS
   - ❑ Map-reduce
   - ❑ Otros componentes
2. **Spark**
   - ❑ Arquitectura

Ecosistema de Hadoop con Spark



http://blog.newtechways.com/2017/10/apache-hadoop-ecosystem.html

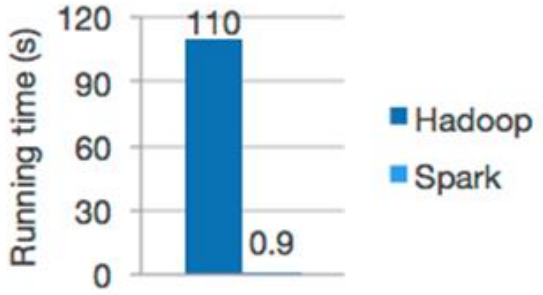
Spark

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

Speed

Run workloads 100x faster.

Apache Spark achieves high performance for both batch and streaming data, using a state-of-the-art DAG scheduler, a query optimizer, and a physical execution engine.



Logistic regression in Hadoop and Spark

Ease of Use

Write applications quickly in Java, Scala, Python, R, and SQL.

Spark offers over 80 high-level operators that make it easy to build parallel apps. And you can use it *interactively* from the Scala, Python, R, and SQL shells.

```
df = spark.read.json("logs.json")
df.where("age > 21")
  .select("name.first").show()
```

Spark's Python DataFrame API
Read JSON files with automatic schema inference

http://spark.apache.org/

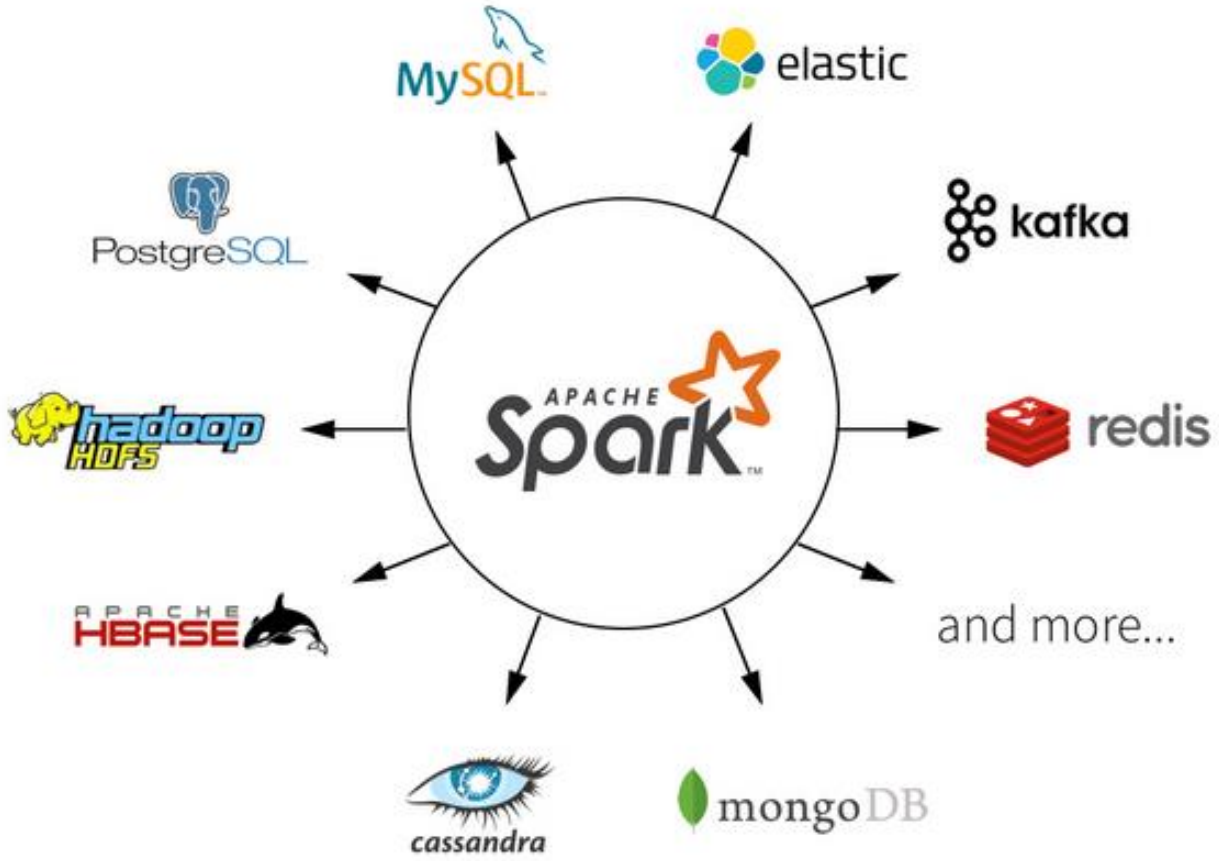
Spark vs Hadoop

	Hadoop MR Record	Spark Record	Spark 1 PB
Data Size	102.5 TB	100 TB	1000 TB
Elapsed Time	72 mins	23 mins	234 mins
# Nodes	2100	206	190
# Cores	50400 physical	6592 virtualized	6080 virtualized
Cluster disk throughput	3150 GB/s (est.)	618 GB/s	570 GB/s
Sort Benchmark Daytona Rules	Yes	Yes	No
Network	dedicated data center, 10Gbps	virtualized (EC2) 10Gbps network	virtualized (EC2) 10Gbps network
Sort rate	**1.42 TB/min**	**4.27 TB/min**	**4.27 TB/min**
Sort rate/node	**0.67 GB/min**	**20.7 GB/min**	**22.5 GB/min**

https://databricks.com/blog/2014/11/05/spark-officially-sets-a-new-record-in-large-scale-sorting.html

Spark - Almacenamiento

Spark puede utilizar datos de diferentes fuentes como:

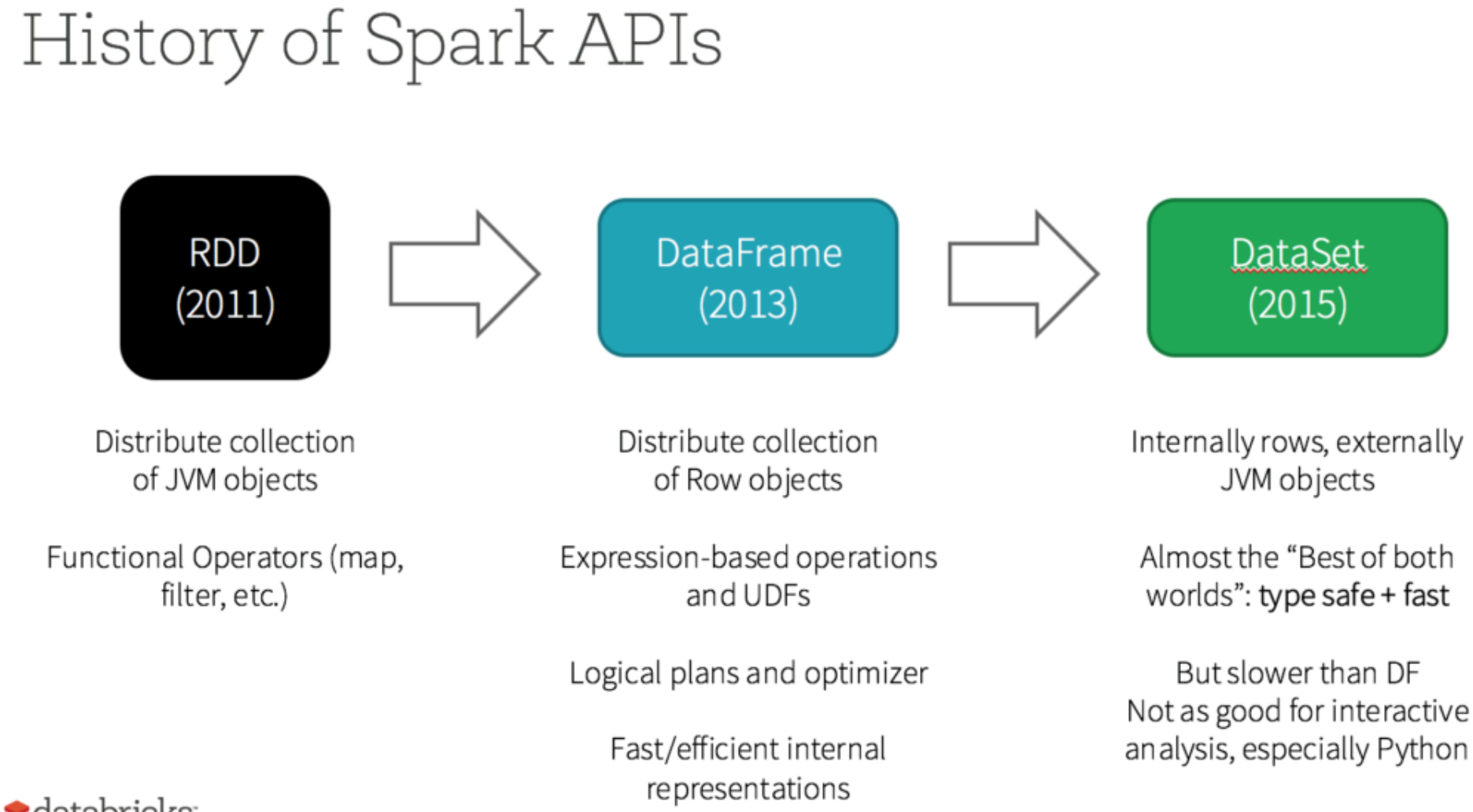


Otras:

- Sistema de archivos local
- Hive
- Nube (Amazon S3)

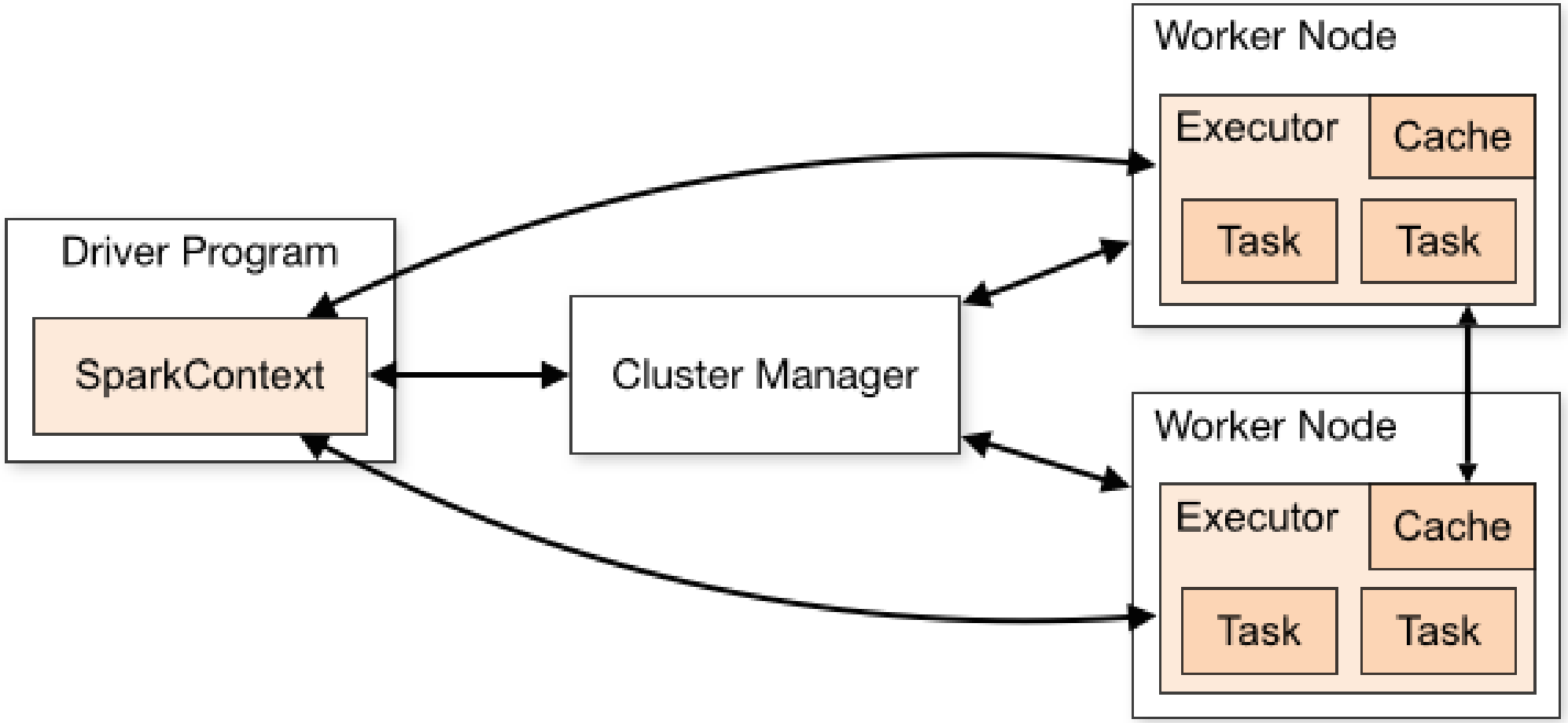
https://sitiobigdata.com/2019/12/24/apache-spark-introduccion-para-principiantes/

Spark - APIs



https://databricks.com/glossary/what-is-rdd

Clusters en Spark



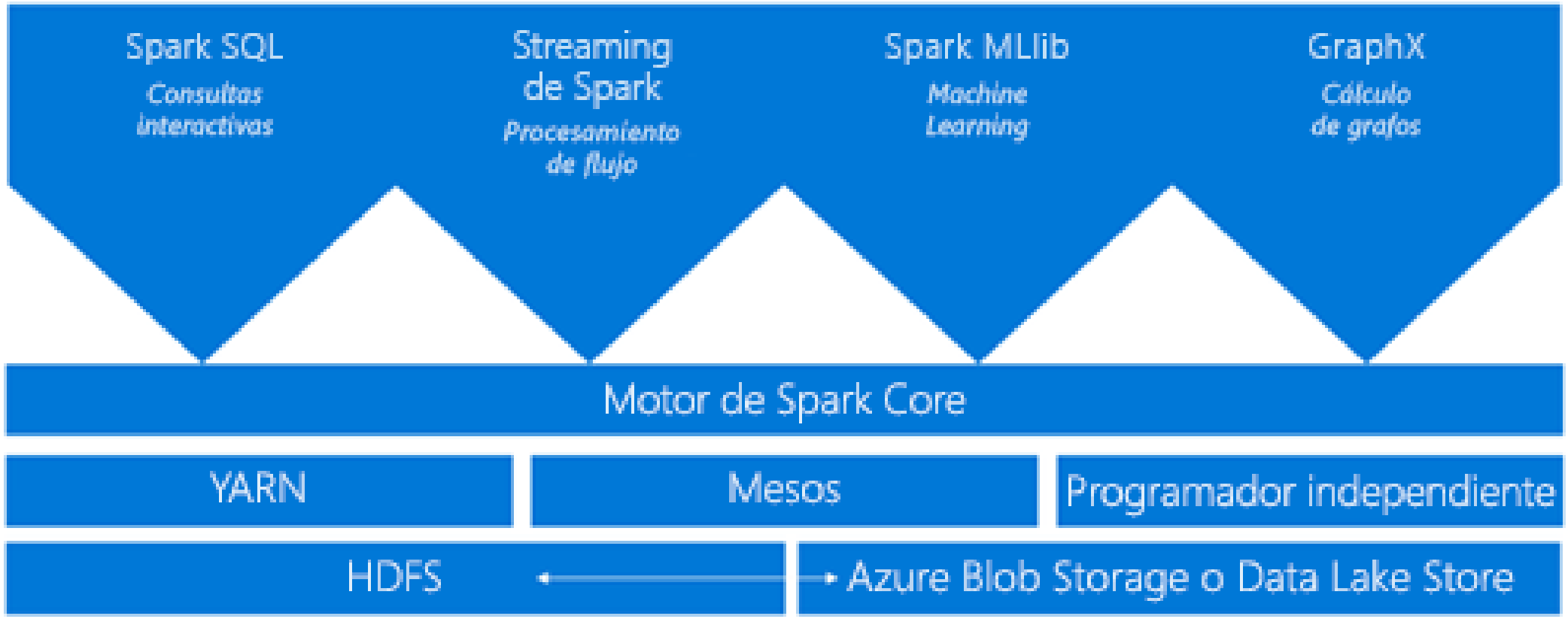
http://spark.apache.org/docs/latest/cluster-overview.html

AGENDA

UdeA
Analytics &
DataScience

1. Hadoop
   - ❑ HDFS
   - ❑ Map-reduce
   - ❑ Otros componentes
2. Spark
   - ❑ **Arquitectura**
   - ❑ Lenguajes

Arquitectura



https://docs.microsoft.com/es-es/azure/hdinsight/spark/apache-spark-overview

Spark - SQL

Spark SQL is Apache Spark's module for working with structured data.

Integrated

Seamlessly mix SQL queries with Spark programs.

Spark SQL lets you query structured data inside Spark programs, using either SQL or a familiar DataFrame API. Usable in Java, Scala, Python and R.

```
results = spark.sql(
  "SELECT * FROM people")
names = results.map(lambda p: p.name)
```

Apply functions to results of SQL queries.

Uniform Data Access

Connect to any data source the same way.

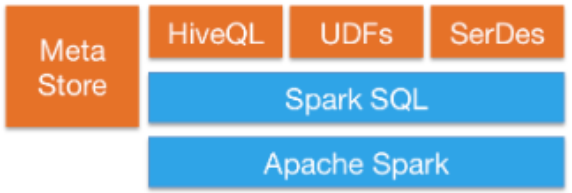
DataFrames and SQL provide a common way to access a variety of data sources, including Hive, Avro, Parquet, ORC, JSON, and JDBC. You can even join data across these sources.

```
spark.read.json("s3n://...")
  .registerTempTable("json")
results = spark.sql(
  """SELECT *
     FROM people
     JOIN json ...""")
```

Hive Integration

Run SQL or HiveQL queries on existing warehouses.

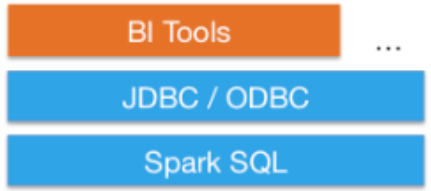
Spark SQL supports the HiveQL syntax as well as Hive SerDes and UDFs, allowing you to access existing Hive warehouses.



Spark SQL can use existing Hive metastores, SerDes, and UDFs.

Standard Connectivity

Connect through JDBC or ODBC.



Use your existing BI tools to query big data.

http://spark.apache.org/sql/

Spark - Streaming

Spark Streaming makes it easy to build scalable fault-tolerant streaming applications.

Ease of Use

Build applications through high-level operators.

Spark Streaming brings Apache Spark's language-integrated API to stream processing, letting you write streaming jobs the same way you write batch jobs. It supports Java, Scala and Python.

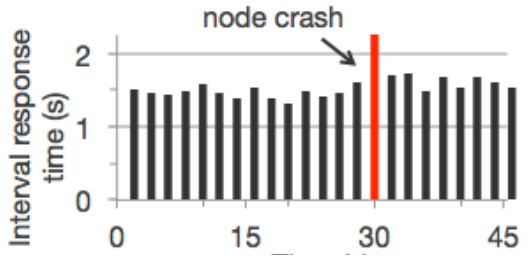
```
TwitterUtils.createStream(...)
    .filter(_.getText.contains("Spark"))
    .countByWindow(Seconds(5))
```

Counting tweets on a sliding window

Fault Tolerance

Stateful exactly-once semantics out of the box.

Spark Streaming recovers both lost work and operator state (e.g. sliding windows) out of the box, without any extra code on your part.



Spark Integration

Combine streaming with batch and interactive queries.

By running on Spark, Spark Streaming lets you reuse the same code for batch processing, join streams against historical data, or run ad-hoc queries on stream state. Build powerful interactive applications, not just analytics.

```
stream.join(historicCounts).filter {
  case (word, (curCount, oldCount)) =>
    curCount > oldCount
}
```

Find words with higher frequency than historic data

http://spark.apache.org/streaming/

Spark - MLlib

- Data types
- Basic statistics
  - summary statistics
  - correlations
  - stratified sampling
  - hypothesis testing
  - random data generation
- Dimensionality reduction
  - singular value decomposition (SVD)
  - principal component analysis (PCA)
- Clustering
  - k-means
  - Gaussian mixture
  - power iteration clustering (PIC)
  - latent Dirichlet allocation (LDA)
  - streaming k-means
- Classification and regression
  - linear models (SVMs, logistic regression, linear regression)
  - naive Bayes
  - decision trees
  - ensembles of trees (Random Forests and Gradient-Boosted Trees)
  - isotonic regression
- Collaborative filtering
  - alternating least squares (ALS)
- Feature extraction and transformation
- Frequent pattern mining
  - FP-growth
- Optimization (developer)
  - stochastic gradient descent
  - limited-memory BFGS (L-BFGS)
- PMML model export

https://spark.apache.org/docs/1.4.1/mllib-guide.html

Spark - GraphX

GraphX is Apache Spark's API for graphs and graph-parallel computation.

Flexibility

Seamlessly work with both graphs and collections.

GraphX unifies ETL, exploratory analysis, and iterative graph computation within a single system. You can view the same data as both graphs and collections, transform and join graphs with RDDs efficiently, and write custom iterative graph algorithms using the Pregel API.

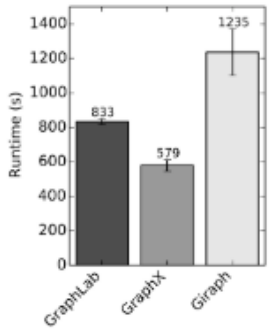
```
graph = Graph(vertices, edges)
messages = spark.textFile("hdfs://...")
graph2 = graph.joinVertices(messages) {
  (id, vertex, msg) => ...
}
```

Using GraphX in Scala

Speed

Comparable performance to the fastest specialized graph processing systems.

GraphX competes on performance with the fastest graph systems while retaining Spark's flexibility, fault tolerance, and ease of use.



End-to-end PageRank performance (20 iterations, 3.7B edges)

http://spark.apache.org/graphx/

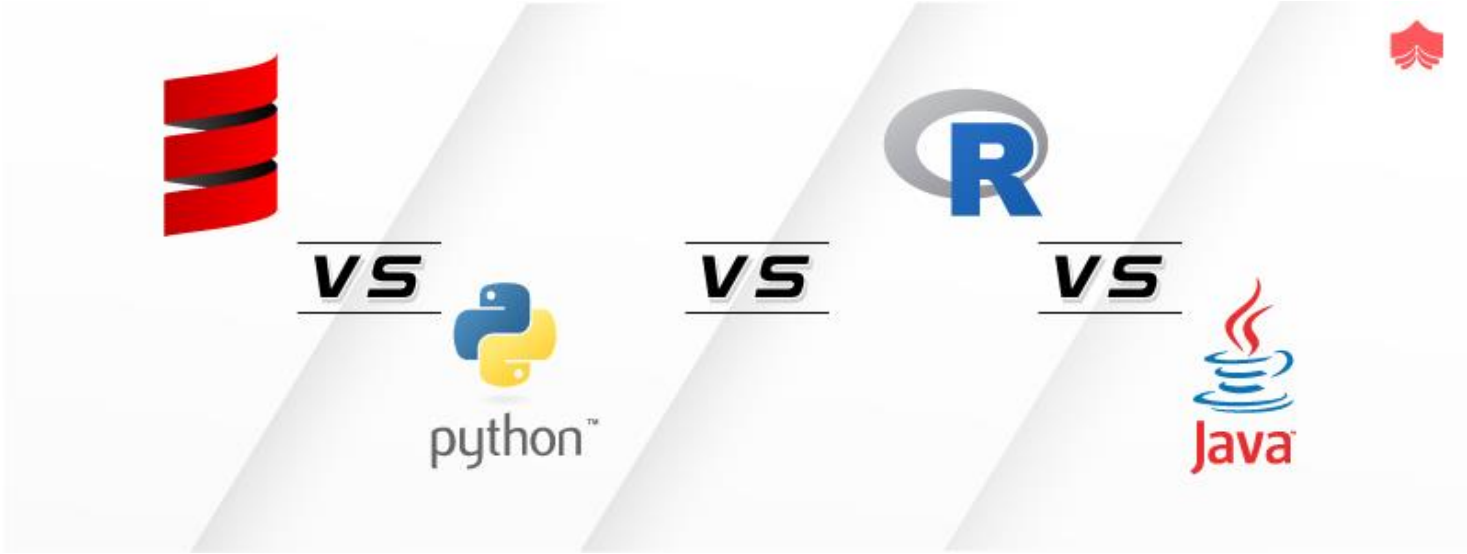
AGENDA

UdeA
Analytics & DataScience

1. Hadoop
   - ❑ HDFS
   - ❑ Map-reduce
   - ❑ Otros componentes
2. Spark
   - ❑ Arquitectura
   - ❑ **Lenguajes**

Spark - Lenguaje

- Scala
- Python
- R
- Java



https://www.knowledgehut.com/blog/programming/scala-vs-python-vs-r-vs-java

Spark - Lenguaje

Característica	Java	Scala	Python	R
Rendimiento	Rápido	**10 veces mas rápido que Python**	Lento	Lento
Curva de Aprendizaje	Más difícil que Python	Más difícil que Java y Python	**Fácil**	Moderado
Grupos de Usuarios	Programadores Web /Hadoop	Programadores de Big Data	Principiantes e Ingenieros de Datos	Científicos de Datos /Estadísticos
Uso	Desarrollo Web Hadoop Nativo	Spark Nativo	Ingeniería de Datos, Visualización	Visualización, Estadística
Soporta Concurrencia	**Si**	**Si**	No	No Aplica
Lenguaje Interpretado	No	No	Si	Si

https://www.knowledgehut.com/blog/programming/scala-vs-python-vs-r-vs-java

PySpark

- API de Python que soporta Apache Spark
- Gratuito y de código abierto
- Puede operar en computación distribuida
- Tolerancia a fallos