Formación senior en Spark

PARTE DE LA FORMACIÓN BIG DATA ENGINEER DE BIG DATA ACADEMY PERÚ

FORMADOR: ALONSO MELGAREJO [alonsoraulmgs@gmail.com]

Concepto

Spark

Es un **motor de procesamiento distribuido paralelo in-memory**. Proporciana **apis en Java, Scala, Python y R**. Spark mantiene la **escalabilidad lineal y la tolerancia a fallos** de MapReduce, pero amplía sus bondades gracias a varias funcionalidades.

Objetivo fundamental

Objetivo fundamental de Spark

Ejecutar procesos **lo más rápido posible**

Naturaleza de funcionamiento

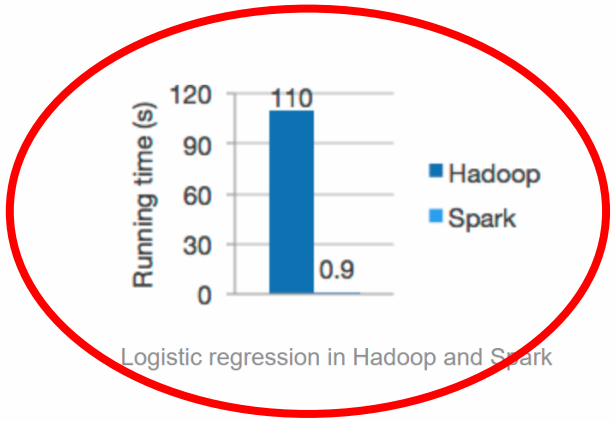
Spark vs Hadoop

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,

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

Latest News

Spark 2.3.3 relea

Spark 2.2.3 relea

Spark+AI Summ
2019, San Franc
(Dec 19, 2018)

Spark 2.4.0 relea

APACI
North Ar
Las Vega

Downl

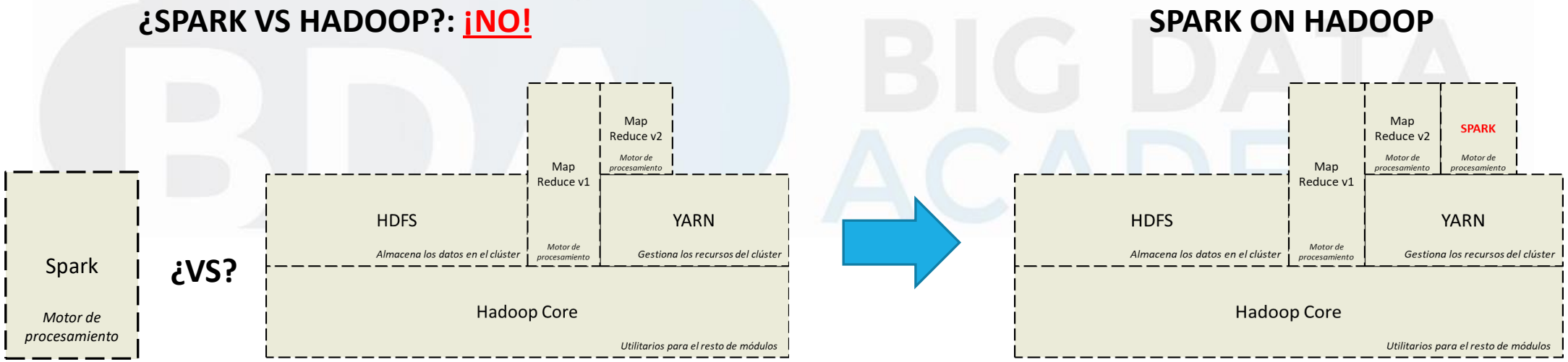
Built-in Libraries

SQL and DataFrar

Spark puede llegar a ser hasta **100 veces más rápido que Hadoop**

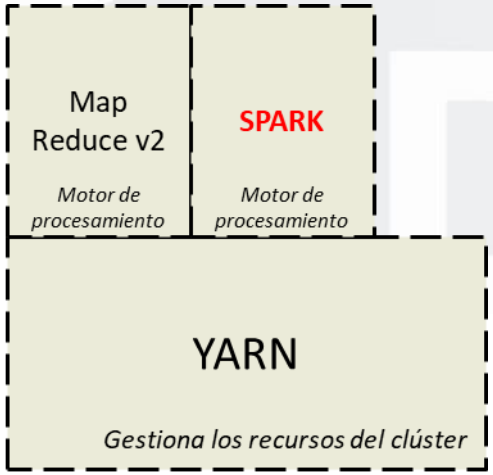
Si Spark es “mejor” entonces, ¿reemplaza a Hadoop?

No, Spark es un motor de procesamiento, Hadoop es un ecosistema que incluye un motor de procesamiento llamado MapReduce v2 y otros componentes (HDFS para almacenar en disco duro, YARN para gestionar recursos). **Spark no reemplaza a Hadoop, sino que lo potencia.**



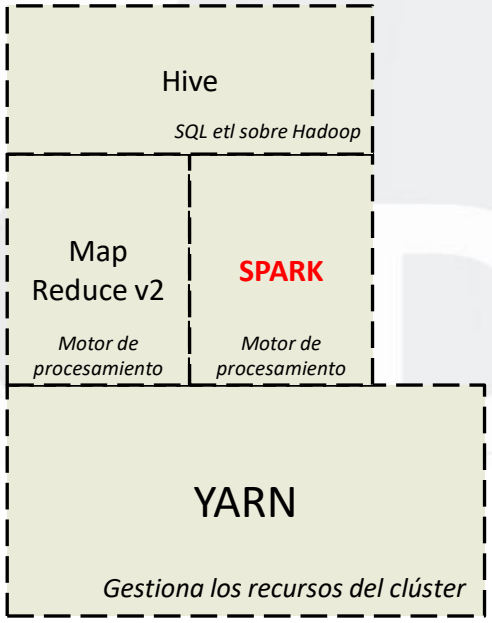
Seleccionando el motor de procesamiento: Spark vs MapReduce

Dependerá del caso de uso que se esté implementado. Spark es más rápido que MapReduce, pero no por eso "mejor". Spark es muy rápido, pero utiliza mucha memoria RAM. MapReduce es más lento, pero ahorra mucha más RAM.



¿Tienes **procesos prioritarios** que deben acabar lo más rápido posible? **SPARK**
¿Tienes **procesos del tipo ETL**? **MAPREDUCE**
¿Tienes procesos que requieren una **lógica tipo SQL**?: **MAPREDUCE**
¿Tienes procesos que requieren de una **programación funcional**?: **SPARK**
¿Tienes procesos de **minería de datos**?: **SPARK**

Hive on Spark



Hive es una "fachada" (facade) que traduce código SQL al código equivalente de algún motor de procesamiento. Por defecto está configurado para funcionar sobre MapReduce, pero **podemos cambiarlo para que se ejecute sobre SPARK.**

SCRIPT HIVE

```
SELECT
    p.name Product_Name,
    s.name Supplier_Name
FROM
    products_suppliers ps
        JOIN products p ON ps.productID = p.pr
            JOIN suppliers s ON ps.supplierID
WHERE
    p.name = 'Pencil 3B'
UNION
SELECT
    products.name Product_Name,
    suppliers.name Supplier_Name
FROM
    products_suppliers
        JOIN products ON products_suppliers.pr
            JOIN suppliers ON products_supplie
WHERE
    price < 0.6
UNION
SELECT
    p.name Product_Name,
    s.name Supplier_Name
FROM
    products p,
    products_suppliers ps,
    suppliers s
WHERE
    p.productID = ps.productID AND
    ps.supplierID = s.supplierID AND
    s.name = 'ABC Traders';
```

SELECCIÓN DEL MOTOR

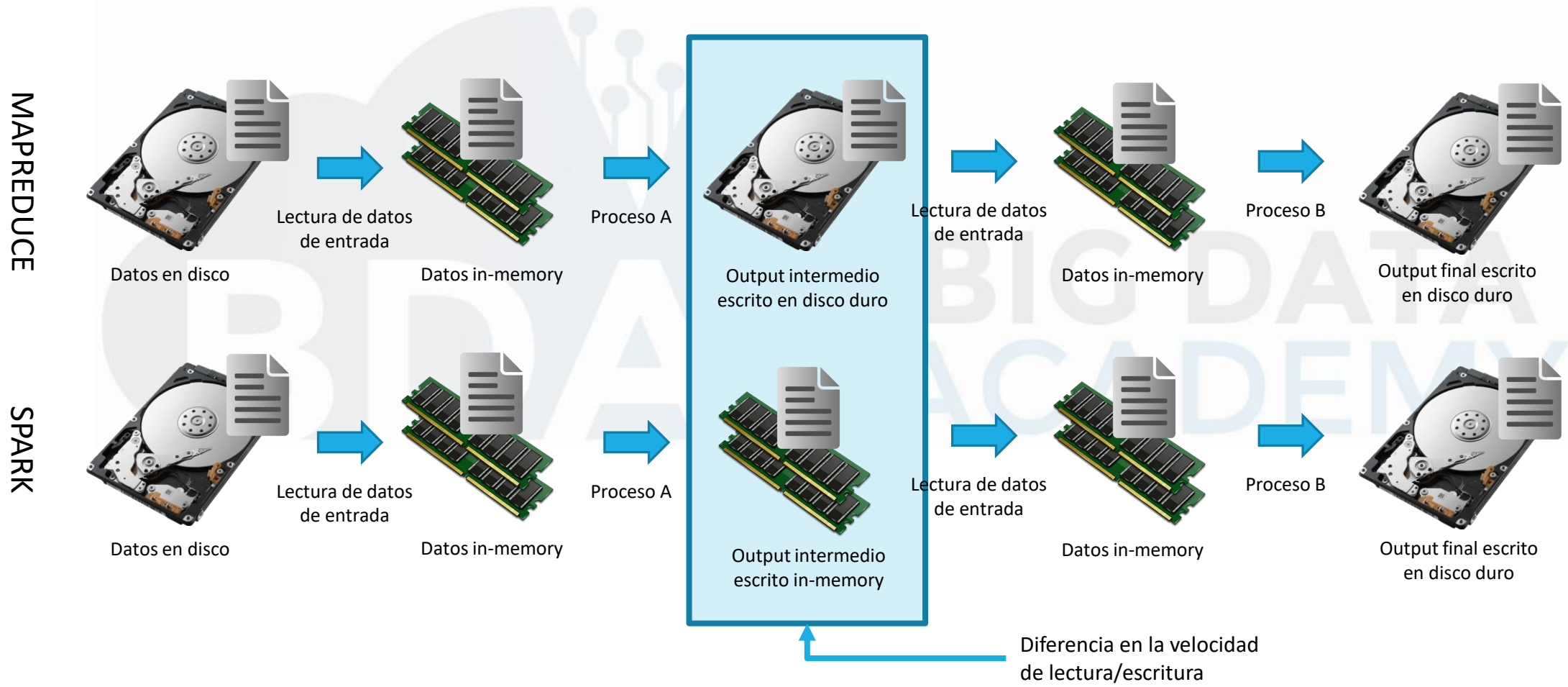
Ejecución sobre MAPREDUCE

SET hive.execution.engine=mr;

Ejecución sobre SPARK

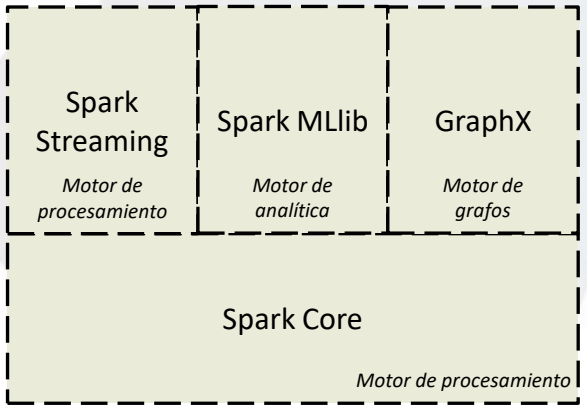
SET hive.execution.engine=spark;

¿Por qué SPARK es más rápido que MapReduce?



Módulos de SPARK

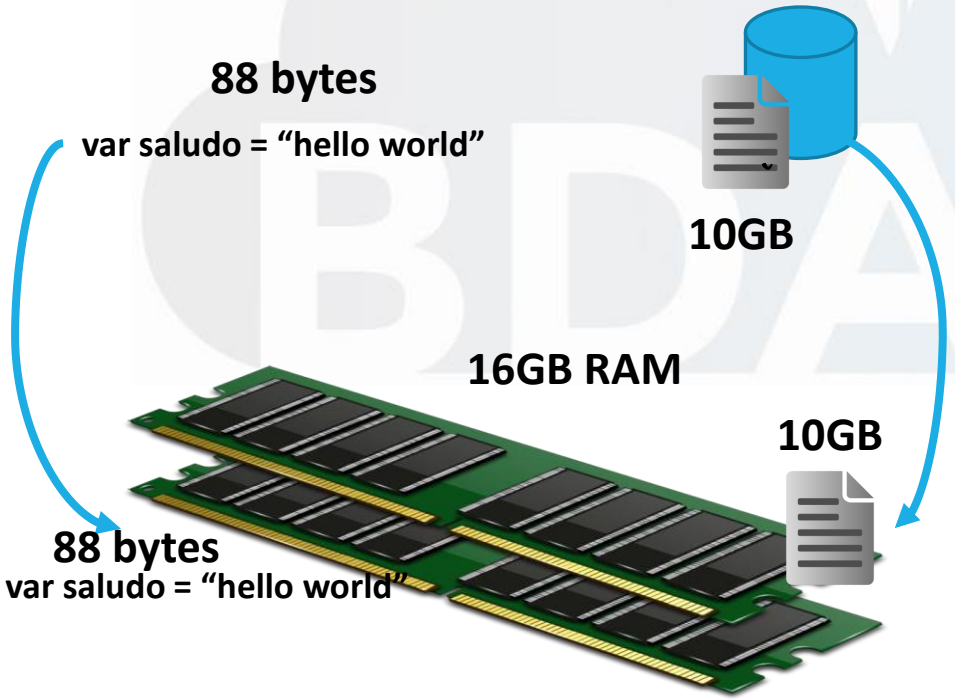
Spark tiene cuatro módulos, cada uno de ellos está enfocado en cierto tipo de programación:



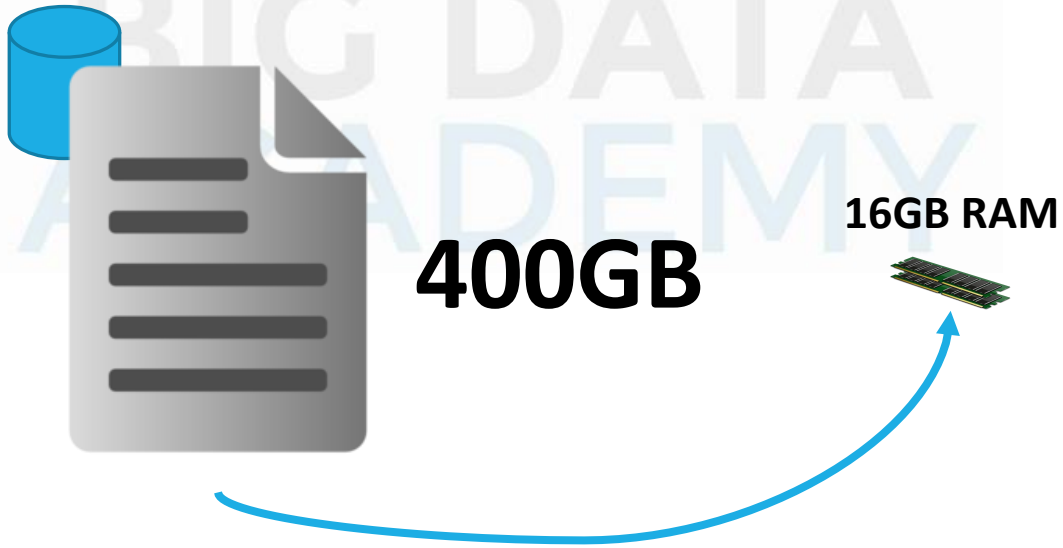
1. **Spark Core**: Programación **batch** in-memory
2. **Spark Streaming**: Programación **micro-batch**
3. **Spark Mllib**: Programación **analítica**
4. **GraphX**: Programación sobre **grafos**

Variables en memoria

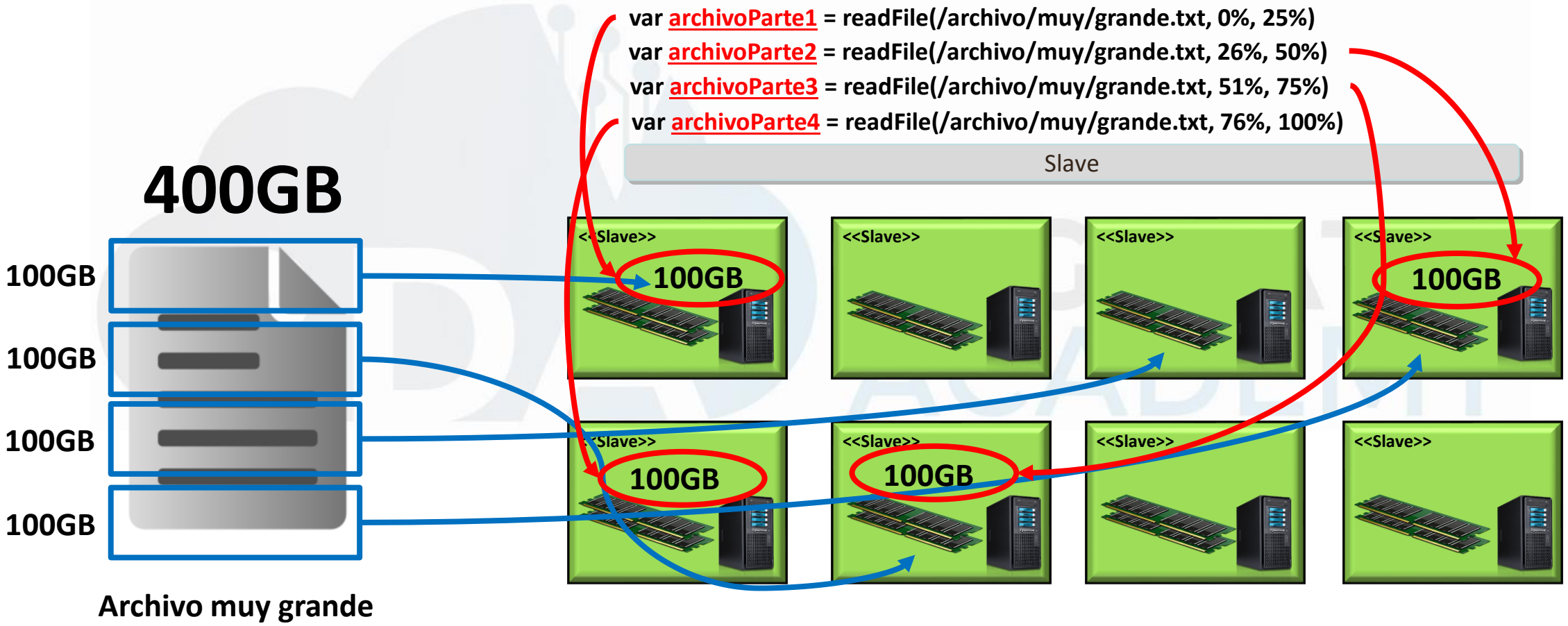
¿Cómo se crea una variable en memoria?



¿Y si tengo un archivo muy grande?

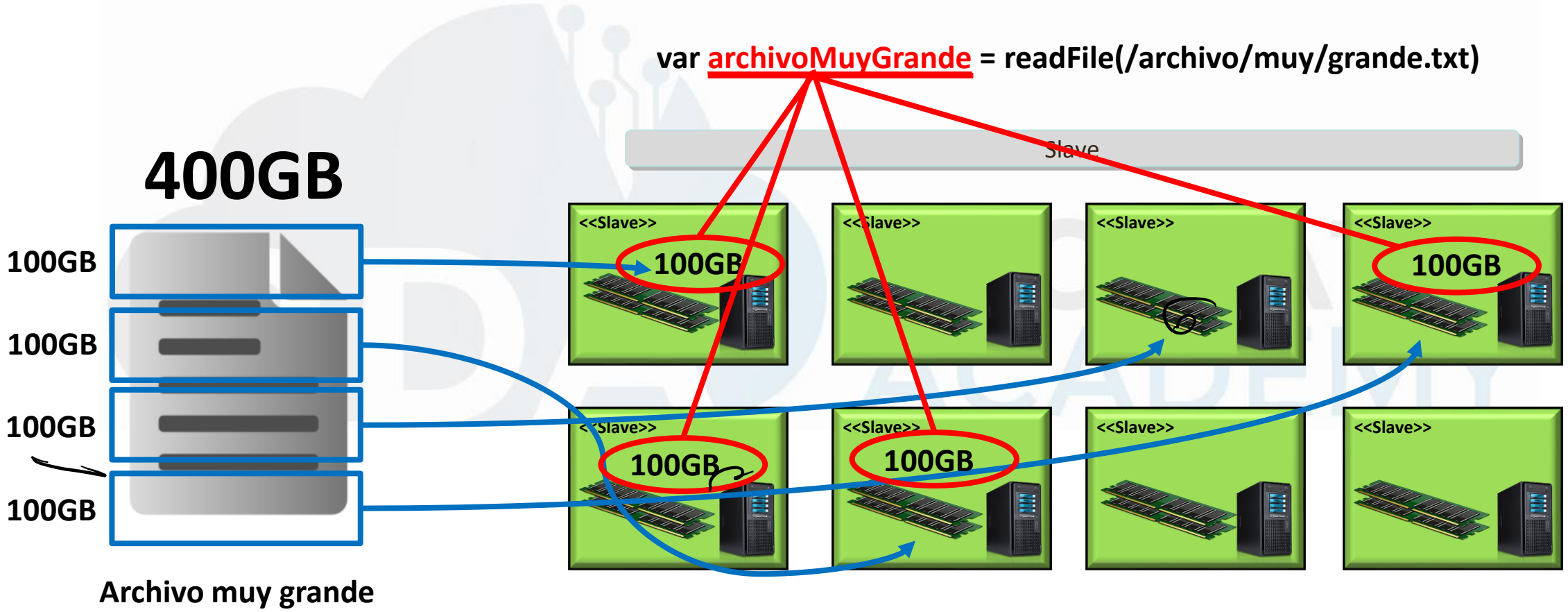


En un clúster clásico



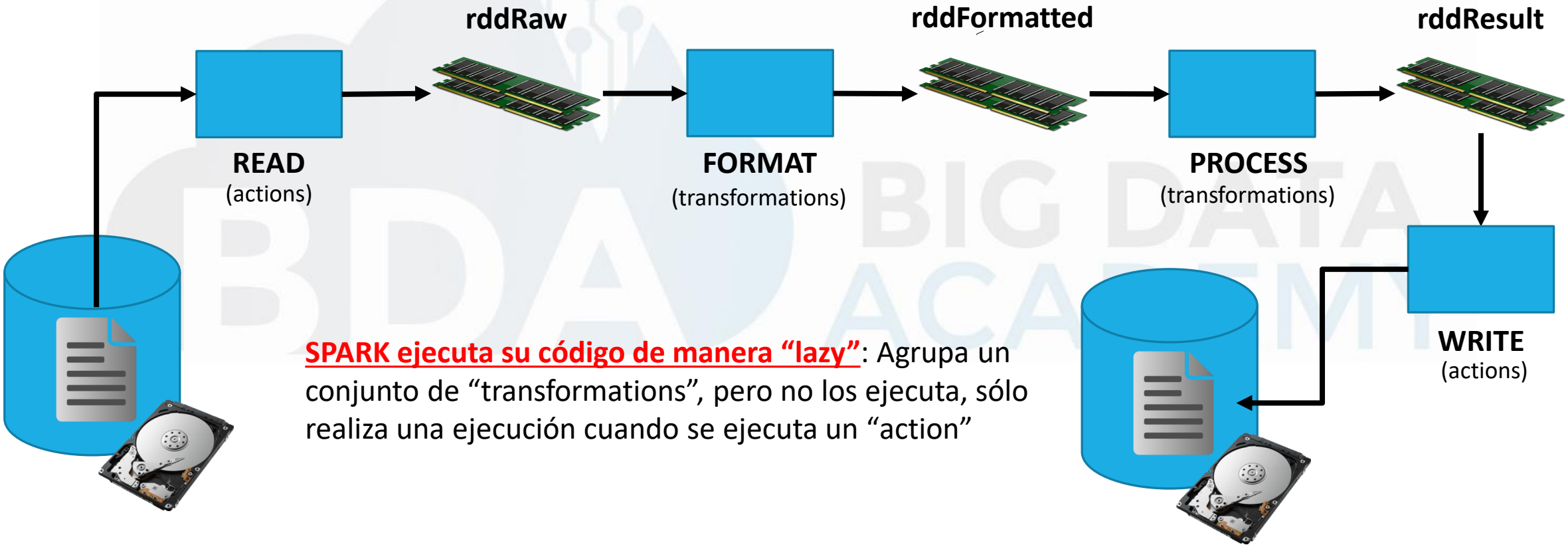
Servidores de 256GB de RAM cada uno

RDD: Resilient Distributed Dataset



Servidores de 256GB de RAM cada uno

Arquetipo de procesamiento en SPARK



SPARK ejecuta su código de manera "lazy": Agrupa un conjunto de "transformations", pero no los ejecuta, sólo realiza una ejecución cuando se ejecuta un "action"

Equivalente de un GROUP BY

```
rddPersonaGroupBy = sc.textFile("/dataset/persona.data"). \
map(lambda line : line.split("|")). \
filter(lambda register : register[0] != "ID"). \
map(lambda register : ( \
int(register[7]), \
( \
1, \
float(register[6]), \
float(register[6]), \
float(register[6]), \
int(register[5]) \
) \
) \
). \
reduceByKey(lambda v1, v2 : ( \
v1[0] + v2[0], \
v1[1] + v2[1], \
max(v1[2], v2[2]), \
min(v1[3], v2[3]), \
v1[4] + v2[4] \
)). \
map(lambda register : (( \
register[0], \
( \
register[1][0], \
register[1][1], \
register[1][2], \
register[1][3], \
register[1][4]/register[1][0] \
) \
))). \
sortBy(lambda register : register[0])
```

```
df5 = dfData.\
groupBy("EDAD").\
agg(\
f.count("EDAD"), \
f.min("FECHA_INGRESO"), \
f.sum("SALARIO"), \
f.max("SALARIO")\
)
```

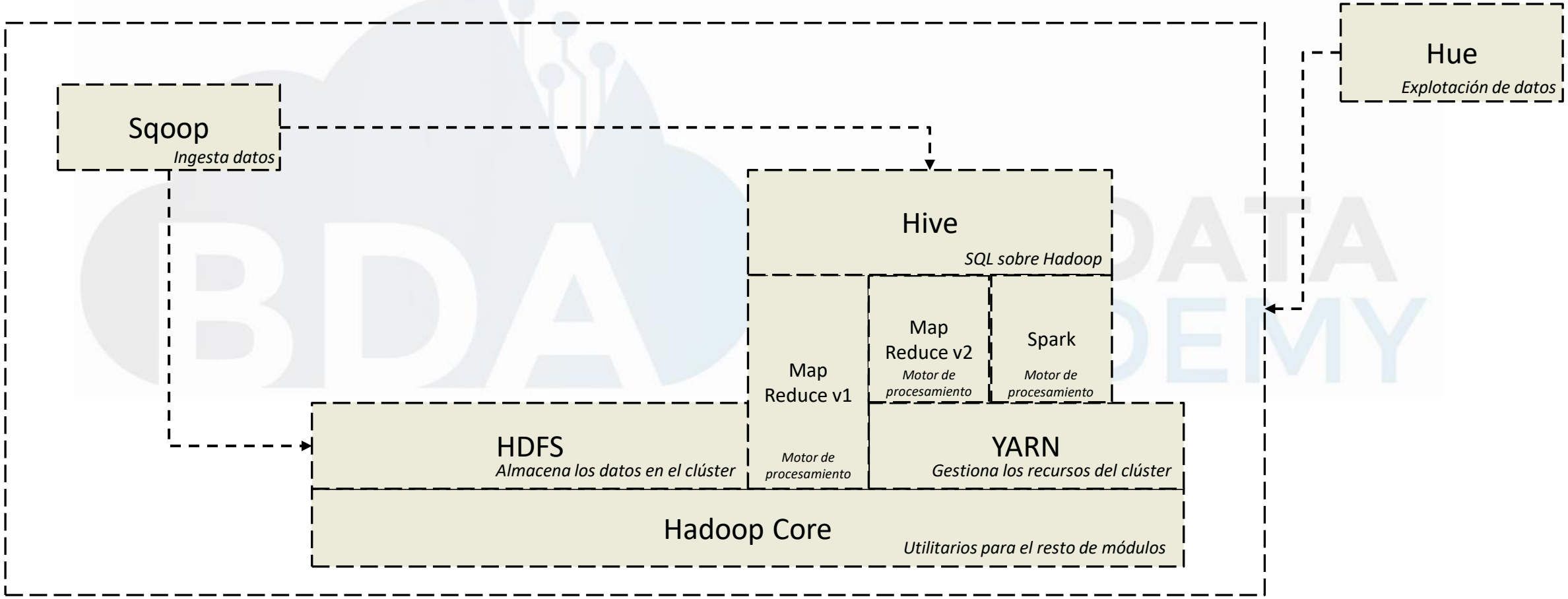
Agregando estructura a los RDD: Los Dataframes

RDD + **METADATA** = **DATAFRAME**

(CAMPO1 STRING,
CAMPO2 INT,
CAMPO3 DOUBLE,
…)

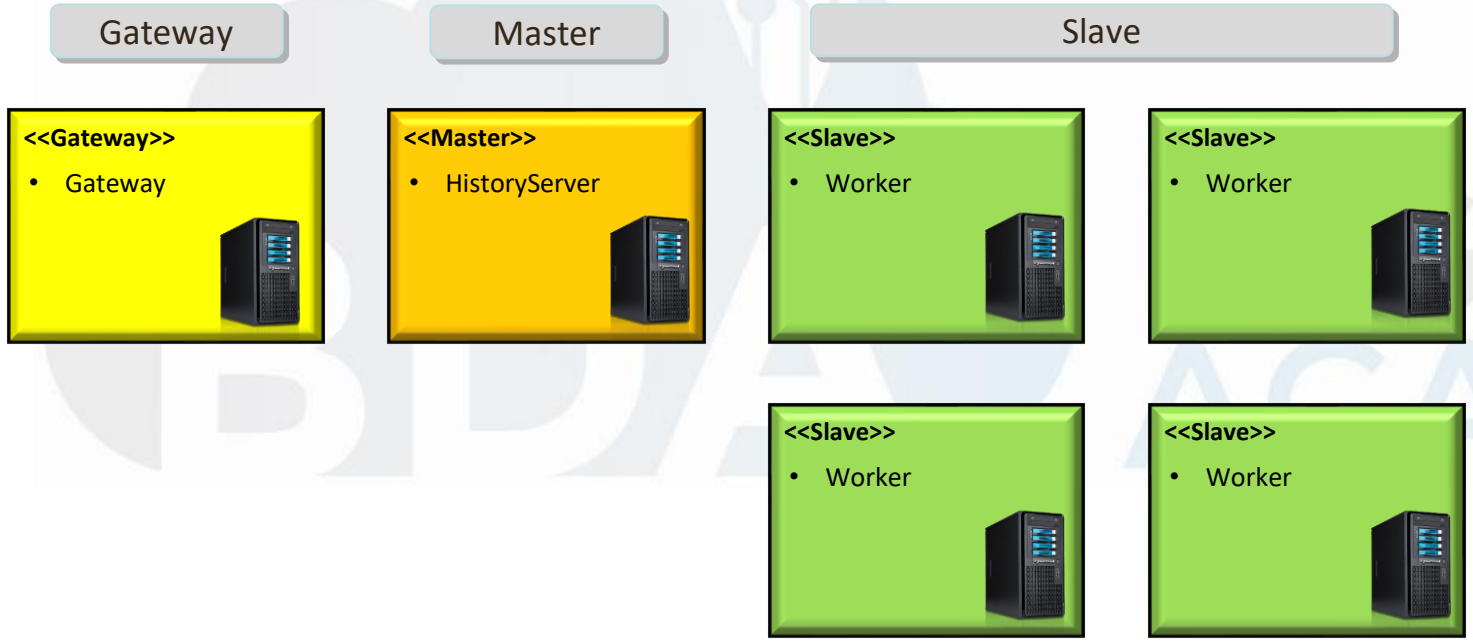
Arquitectura de componentes

Arquitectura de componentes



Arquitectura de servicios

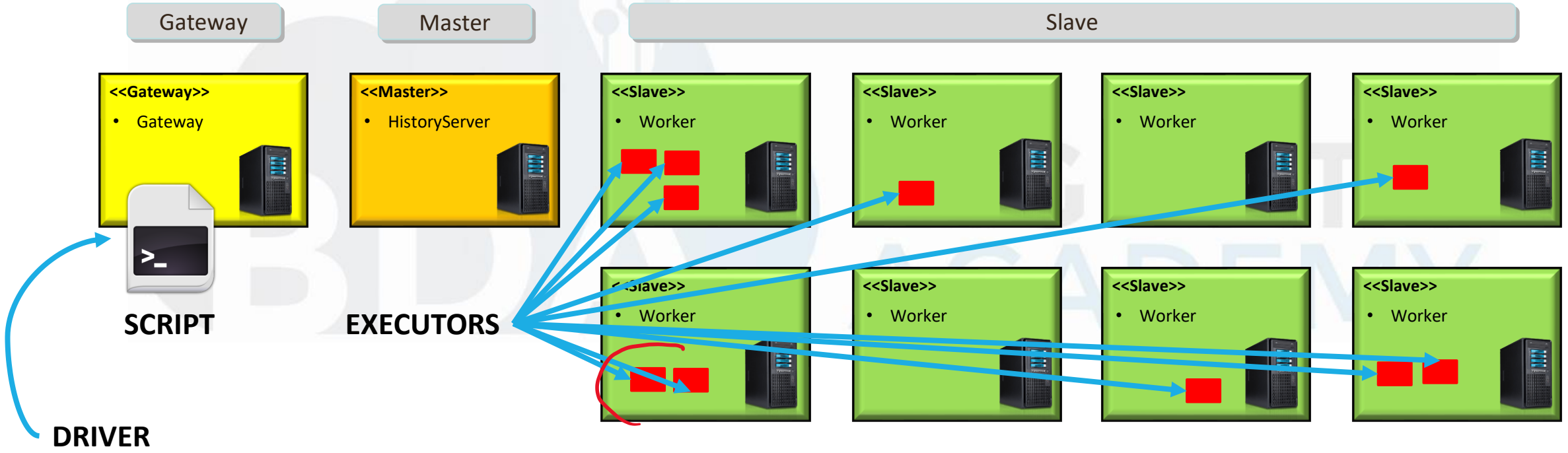
Arquitectura de servicios



DESCRIPCIÓN

- **Gateway**: Nodo con la configuración de SPARK y acceso a las shells
- **HistoryServer**: Almacena información sobre los programas ejecutados sobre SPARK
- **Worker**: Realiza el procesamiento de las cargas de trabajo

Drivers y executors



Un executor es un contenedor de recursos computacionales (RAM y CPU)

Resumen

Hablemos…