

APACHE SPARK

Sesión 1

¿QUÉ VEREMOS EN ESTA UNIDAD?



1. Introducción a Spark
2. Entornos de trabajo
3. Repaso Python y funciones Lambda
4. Primeros pasos con Spark
 1. Acciones
 2. Transformaciones
5. Parte práctica

Contenido de la sesión 2



1. Repaso sesión 1 y dudas
2. Corrección ejercicios sesión 1
3. RRDs
4. RDDs Clave Valor
5. Persistencia
6. Acumuladores
7. Variables de transmisión
8. Particionado

Contenido de la sesión 3



1. Repaso sesión anterior y dudas
2. Corrección ejercicios sesión anterior
3. Spark SQL
4. Introducción a MLLIB
5. Introduccion GraphX

Contenido de la sesión 4



1. Repaso sesión anterior y dudas
2. Corrección ejercicios
3. Spark en Cluster
4. Paralelización
5. Etapas tareas y Planificación
6. Aplicaciones y rendimiento
7. Spark Streaming

Introducción a Spark



Origen de Spark



Computación tradicional

Pocos datos

Fácil procesamiento

Necesidades reducidas

VS

Computación actual

Gran volumen de datos

Procesamientos complejos

Sistemas distribuidos

Origen de Spark



Surgen los sistemas distribuidos

- + Capacidad de procesamiento
- + Velocidad
- + Cantidad de datos

Origen de Spark



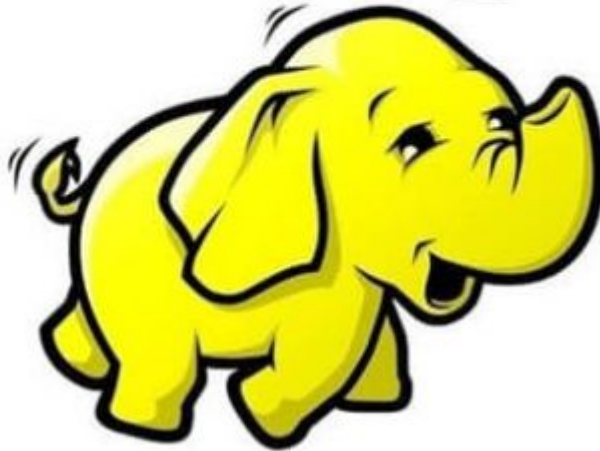
Surgen los problemas de los sistemas distribuidos

- + Dificultad para programar
- + Ancho de banda
- + Probabilidad de fallos

Origen de Spark

Mejoras de los sistemas distribuidos

hadoop



Origen de Spark



Tolerancia a fallos

- Si falla sigue trabajando
- Reasignación de tareas a otro nodo
- Replicación de datos en nodos
- Nodo recuperado vuelve automáticamente al cluster

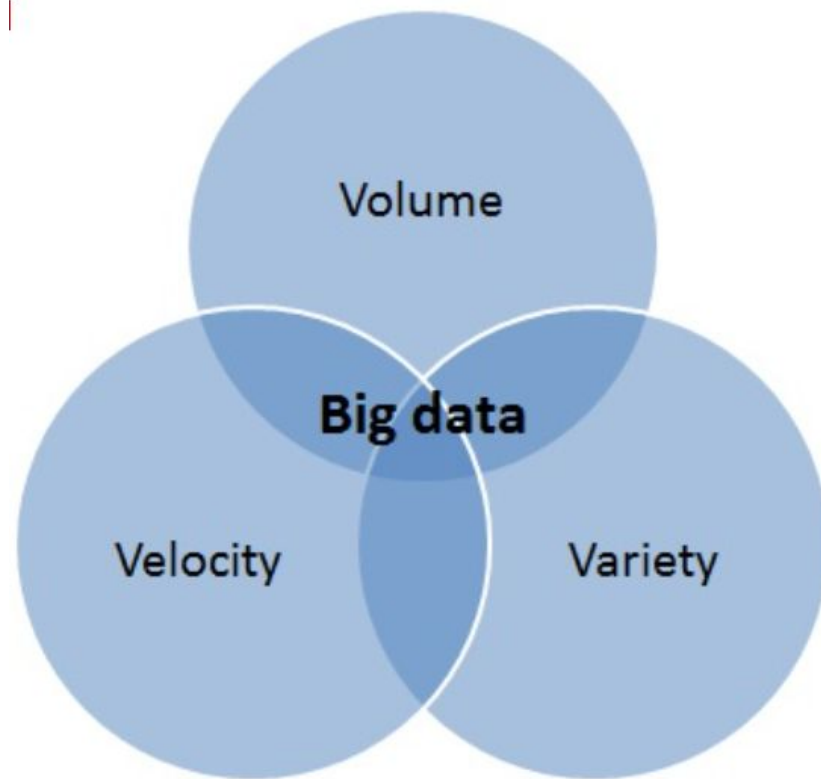
Origen de Spark



¿Para qué se usa?

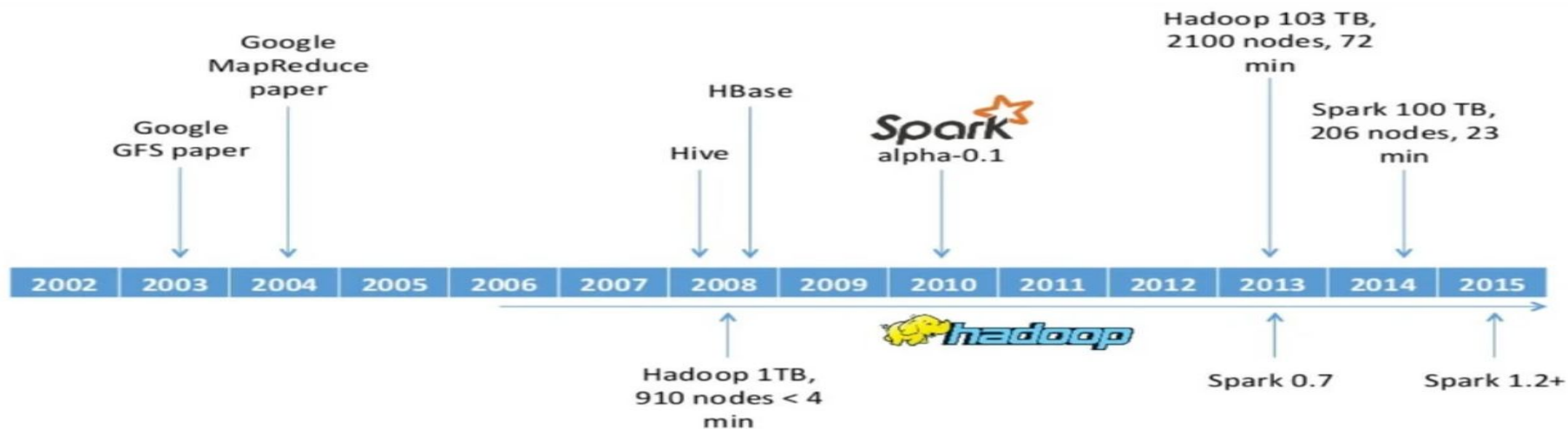
- ETL
- Creación de análisis
- Reconocimiento de patrones
- Modelos de predicción
- Evaluación de riesgos
- ...

Origen de Spark



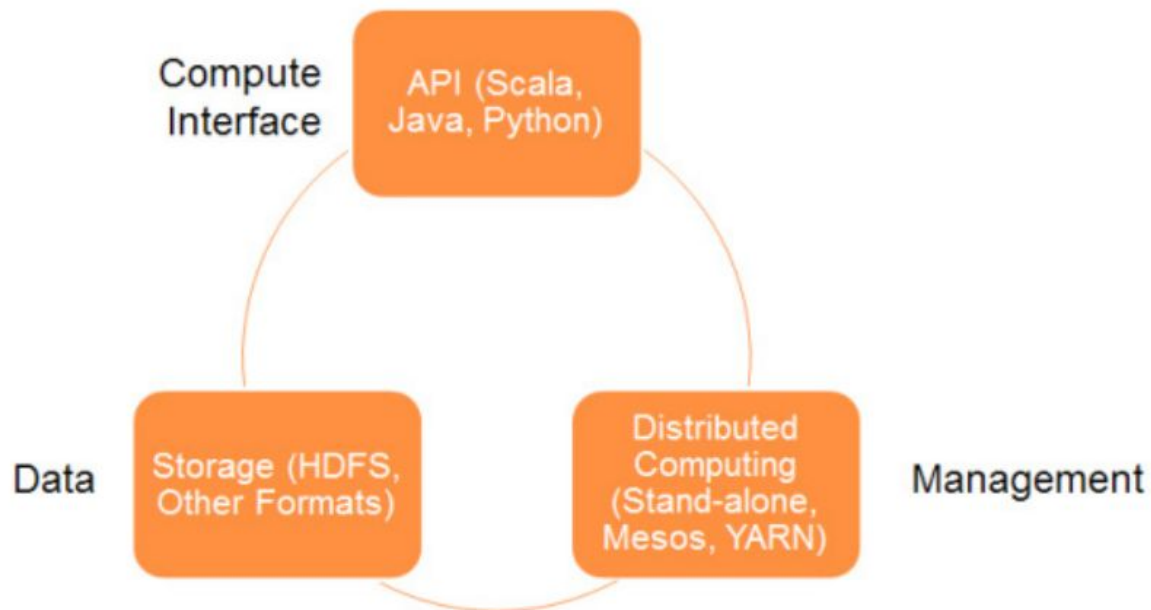
Origen de Spark

- Universidad de Berkeley 2009
- No se pensó como solución Big Data
- Hoy es proyecto open source
- Sistema de procesamiento de datos de alto volumen en cluster



Origen de Spark

Arquitectura de Spark



Origen de Spark



Programación a alto nivel

Computación en el cluster

Almacenamientos distribuido

Datos en memoria

Análisis de datos imposibles a priori

Menor coste

Menor tiempo

Más flexibilidad

Escalabilidad

Origen de Spark



Hadoop Vs Spark

API alto nivel

Baja latencia

Almacenamiento en memoria x100 de rendimiento

Componentes específicos GraphX, Spark Streaming, Spark SQL, Mlib...

Origen de Spark



Hadoop Vs Spark

PILARES BÁSICOS

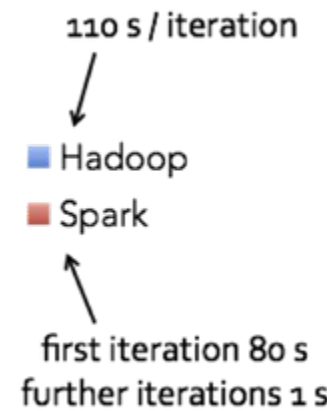
RENDIMIENTO

COSTE

SEGURIDAD

FACILIDAD DE USO

RENDIMIENTO



Origen de Spark



COSTE

Más memoria = Mayor Coste

Coste de memoria > Coste Disco

Origen de Spark



SEGURIDAD

HDFS es quien proporciona la seguridad

Origen de Spark



FACILIDAD DE USO

Spark: API de java, scala,python,R...

Hadoop: Hive,pig,sqoop...

Origen de Spark



Hadoop MapReduce vs Spark

```
public class WordCount {  
    public static void main(String[] args) throws Exception {  
        Job job = new Job();  
        job.setJarByClass(WordCount.class);  
        job.setJobName("Word Count");  
        FileInputFormat.setInputPaths(job, new Path(args[0]));  
        FileOutputFormat.setOutputPath(job, new Path(args[1]));  
        job.setMapperClass(WordMapper.class);  
        job.setReducerClass(SumReducer.class);  
        job.setMapOutputKeyClass(Text.class);  
        job.setMapOutputValueClass(IntWritable.class);  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(IntWritable.class);  
        boolean success = job.waitForCompletion(true);  
        System.exit(success ? 0 : 1);  
    }  
}
```

```
public class WordMapper extends Mapper<LongWritable, Text, Text, IntWritable> {  
    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {  
        String line = value.toString();  
        for (String word : line.split("\\W+")) {  
            if (word.length() > 0)  
                context.write(new Text(word), new IntWritable(1));  
        }  
    }  
}
```

```
public class SumReducer extends Reducer<Text, IntWritable, Text, IntWritable> {  
    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {  
        int wordCount = 0;  
        for (IntWritable value : values) {  
            wordCount += value.get();  
        }  
        context.write(key, new IntWritable(wordCount));  
    }  
}
```

Origen de Spark



MapReduce vs Spark

Python:

Python:

```
sc.textFile(file)
    .flatMap(lambda s: s.Split())
    .map(lambda w: (w,1))
    .reduceByKey(lambda v1,v2: v1+v2)
    .saveAsTextFile(output)
```

Scala:

Scala:

```
sc.textFile(file)
    .flatMap(s => s.split(" "))
    .map(w => (w,1))
    .reduceByKey(_ + _)
    .saveAsTextFile(output)
```

Origen de Spark



1. Está integrado con Apache Hadoop.
2. Trabaja en memoria, con lo que se **consigue mucha mayor velocidad de procesamiento**
3. También permite trabajar en disco
4. Nos proporciona API para Java, Scala, Python y R
5. Permite el **procesamiento en tiempo real, Spark Sql y otras funcionalidades**

Origen de Spark



Los componentes del framework son los siguientes:

- **Spark Core** : Es la base o conjunto de librerías donde se apoya el resto de módulos. Es el núcleo del framework.
- **Spark SQL**: Es el módulo para el procesamiento de datos estructurados y semi-estructurados
- **Spark Streaming** : Es el que permite la ingesta de datos en tiempo real.
- **Spark Graph** : Permite el procesamiento de grafos
- **Spark MLlib** : Es una librería muy completa que contiene numerosos algoritmos de Machine Learning



Comprendiendo Spark

¿Cuál es su relación con Hadoop?

¿Son competencia?

¿Son lo mismo?

¿Que tiene que ver con Hadoop?



Comprendiendo Spark

¿Cómo funciona?



¿Cómo funciona Spark?



Comprendiendo Spark

¿Cuales son sus funciones?

¿Para qué sirve?

¿Qué puedo hacer con Spark?



Entornos de trabajo



Spark Shell

- Python (pyspark)
- Scala (spark-shell)
- Java (no tiene shell)

Instalación en Windows y Linux

Cloudera

Máquina virtual

Entornos de trabajo



¿Qué vamos a necesitar?

- Windows o Linux
- Python
- Jupyter notebook

Instalación Pyspark Windows



PASO 1. Instalar la distribución Python o Anaconda

El primer paso es descargar e instalar Python desde [Python.org](https://python.org) o la distribución Anaconda que incluye Python, Spyder IDE y Jupyter notebook.

Descargamos el ejecutable y lo instalamos, debemos asegurarnos de que las variables de entorno están correctamente establecidas.

```
JAVA_HOME = C:\Program Files\Java\jdk1.8.0_201
```

```
PATH = %PATH%;C:\Program Files\Java\jdk1.8.0_201\bin
```

Instalación Pyspark Windows



PASO 2. Instalación de PySpark en Windows

En la página de descarga de Spark(<https://spark.apache.org/downloads.html>)

Una vez lo hemos descargado, descomprime el binario usando 7zip o winrar y copia la carpeta de dentro spark-3.0.0-bin-hadoop2.7 en la ruta que quieras por ejemplo **c:\apps**

Ahora crearemos las variables de entorno.

```
SPARK_HOME = C:\apps\spark-3.0.0-bin-hadoop2.7
```

```
HADOOP_HOME = C:\apps\spark-3.0.0-bin-hadoop2.7
```

```
PATH=%PATH%;C:\apps\spark-3.0.0-bin-hadoop2.7\bin
```

Instalación Pyspark Windows



PASO 3. Instale winutils.exe en Windows

Descargue el archivo winutils.exe de winutils y cópielo en la %SPARK_HOME%\bin. Winutils son diferentes para cada versión de Hadoop, por lo tanto, descargue la versión correcta de <https://github.com/steveloughran/winutils>

PASO 4. Ejecuta PYSPARK

Ya lo tendríamos todo, ahora solo tendremos que abrir una consola y ejecutar:

pyspark

```
C:\Users\fedev>pyspark
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
21/03/22 19:50:10 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to

  ____      _
 / ___|  _ \| | | |
 \___ \| |_) | |_| |
  ___) | |_) | | | |
 |____|_|_|\___|_|_|_|

 version 3.1.1

Using Python version 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021 13:44:55)
Spark context Web UI available at http://DESKTOP-58ICH65:4040
Spark context available as 'sc' (master = local[*], app id = local-1616439012124).
SparkSession available as 'spark'.
>>>
```

Instalación Pyspark Windows



Paso 5. PySpark con Jupyter notebook

- Pip install jupyter
- conda install -c conda-forge Findspark (anconda)
- Pip install findpark (cmd)

Instalación Pyspark Windows

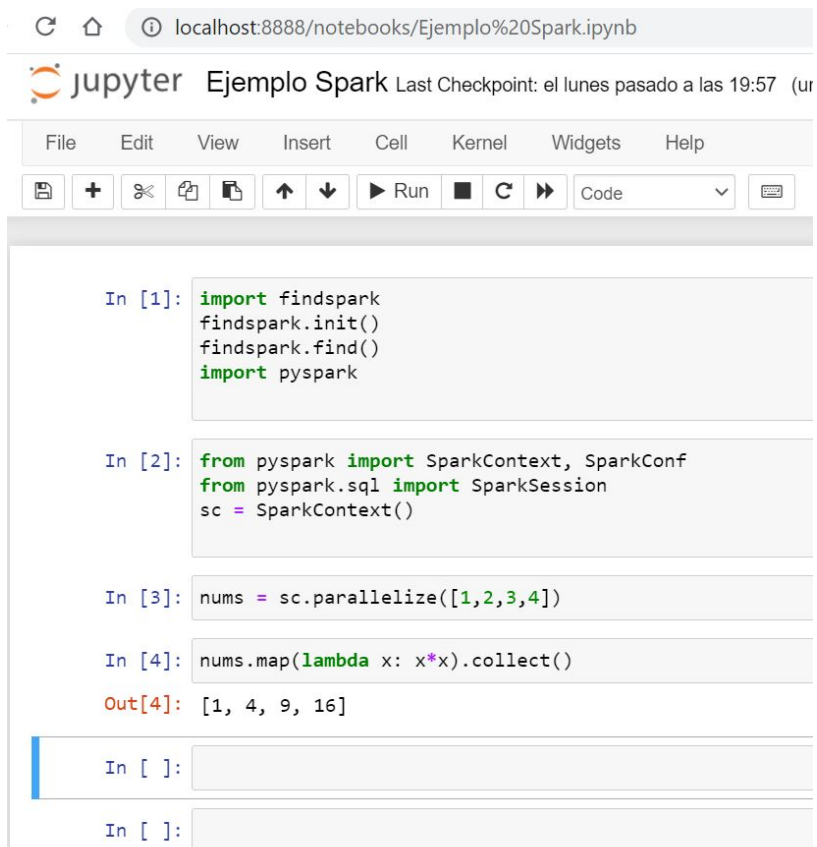


Las siguientes instrucciones deberían funcionar en Windows 7, Windows 8.1 y Windows 10:

1. Descarga el script del [instalador get-pip.py](#). Si estás en Python 3.2, necesitarás esta versión de [get-pip.py](#). En caso de tener Python 3.3 o 3.4 usar estas versiones de PiP correspondientemente [Python 3.3 get-pip.py](#) o [Python 3.4 get-pip.py](#). De cualquier manera, haga clic derecho en el enlace y seleccione Guardar como y guárdelo en cualquier carpeta del pc, como su carpeta de Descargas.
2. Abra el símbolo del sistema y navegue hasta el archivo get-pip.py.
3. Ejecute el siguiente comando: `python get-pip.py`

Instalación Pyspark Windows

Jupyter notebook



The image shows a Jupyter Notebook interface in a web browser. The address bar shows the URL: localhost:8888/notebooks/Ejemplo%20Spark.ipynb. The notebook title is "Ejemplo Spark" and it shows the last checkpoint time as "el lunes pasado a las 19:57". The interface includes a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu bar is a toolbar with icons for saving, adding cells, undo, redo, and running code. The notebook content consists of four input cells and one output cell. The first cell imports findspark and pyspark. The second cell imports SparkContext, SparkConf, SparkSession, and creates a SparkContext object. The third cell creates a parallelized list of numbers. The fourth cell maps a lambda function to the list and collects the results. The output of the fourth cell is displayed as [1, 4, 9, 16].

localhost:8888/notebooks/Ejemplo%20Spark.ipynb

Jupyter Ejemplo Spark Last Checkpoint: el lunes pasado a las 19:57 (ur

File Edit View Insert Cell Kernel Widgets Help

Run

```
In [1]: import findspark
        findspark.init()
        findspark.find()
        import pyspark
```

```
In [2]: from pyspark import SparkContext, SparkConf
        from pyspark.sql import SparkSession
        sc = SparkContext()
```

```
In [3]: nums = sc.parallelize([1,2,3,4])
```

```
In [4]: nums.map(lambda x: x*x).collect()
```

Out[4]: [1, 4, 9, 16]

```
In [ ]:
```

```
In [ ]:
```

Instalación Pyspark Linux



- Instalar openjdk-8-jdk
- *Revisar la versión java -version*
- *Descarga Apache spark*
- *Modificamos la variable de entorno*
- *Instalar spark*
- *Instalar jupyter notebook*
- *Sincronizar spark y jupyter*

CLOUDERA



¿Qué es cloudera?

“Cloudera es la plataforma Apache Hadoop más rápida, fácil de usar y segura con la que solucionar los problemas de datos más complejos. Y es que Hadoop se ha convertido en el nuevo paradigma de arquitectura de computación en paralelo.”

Repaso python



Es un lenguaje de programación de código abierto. Su filosofía es que su sintaxis favorezca la legibilidad. También la favorece el hecho de que el contenido de los bloques de código está delimitado mediante indentación (espacios o tabuladores). Esto hace que sea simple, versátil y de desarrollo rápido.

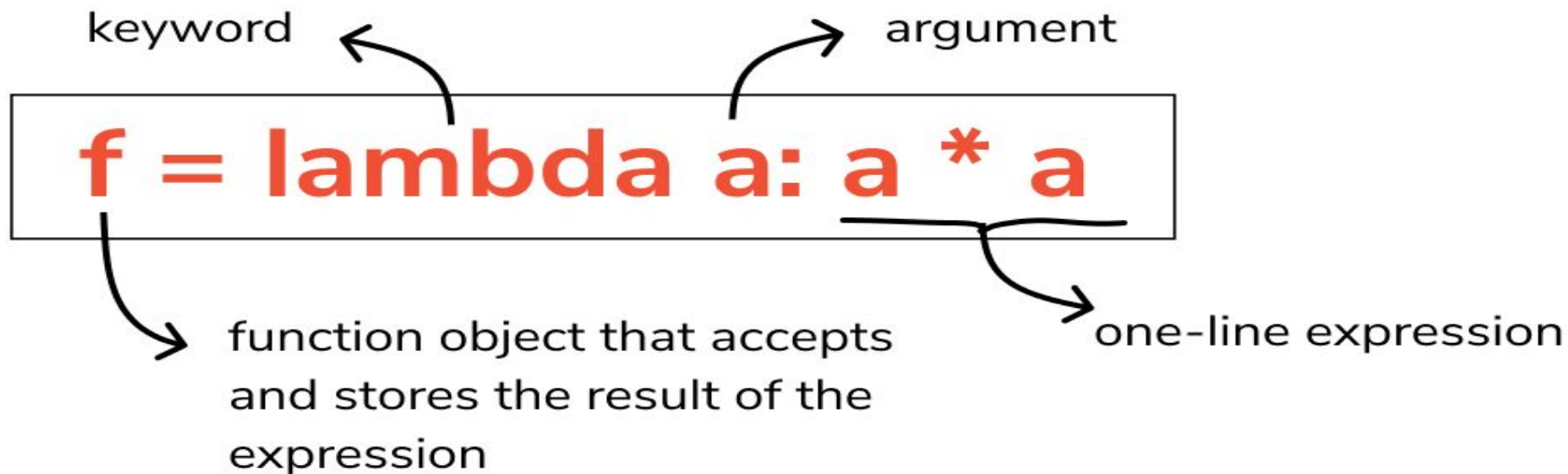
Repaso expresiones Lambda



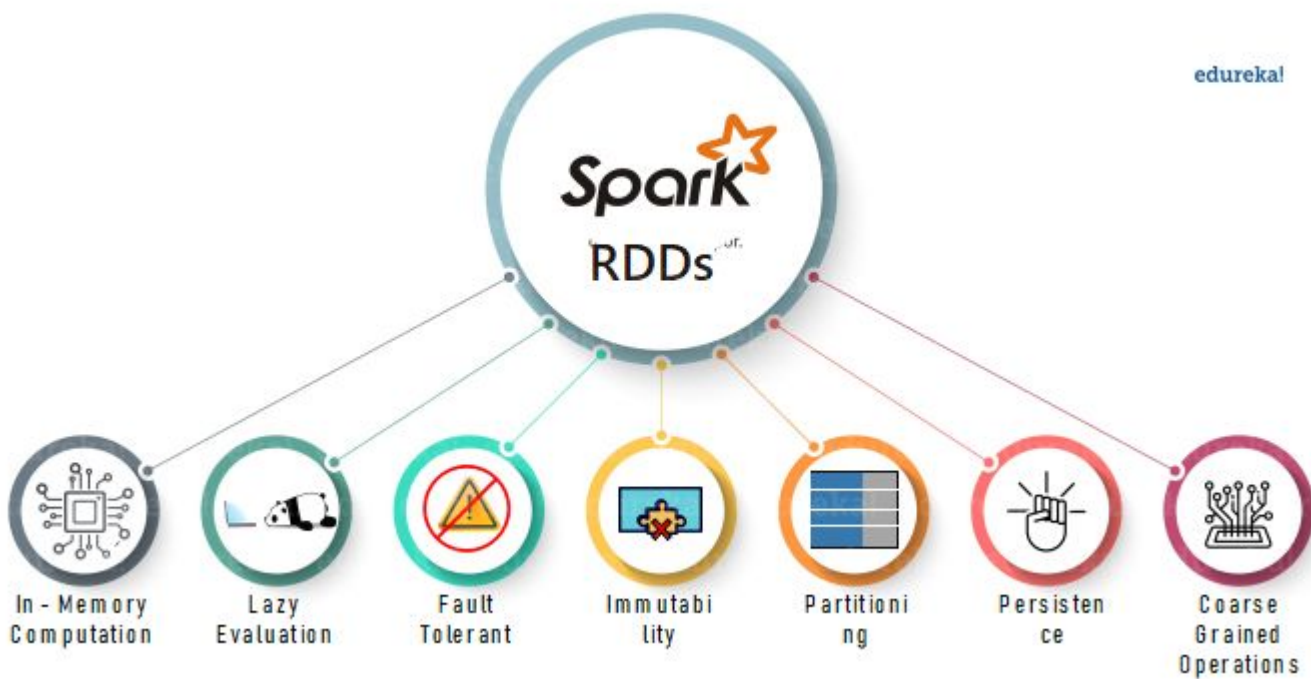
- Funciones anónimas
- Código claro y conciso
- Tiempo de compilación

Repaso expresiones Lambda

Funciones anónimas



Primeros pasos con Spark



Primeros pasos con Spark



Introducción a RDDs

- Resilient
- Distributed
- Dataset

Es una colección distribuida inmutable de objetos. Cada conjunto de datos en RDD se divide en particiones lógicas, que se pueden calcular en diferentes nodos del clúster. Los RDD pueden contener cualquier tipo de objetos Python, Java o Scala, incluidas las clases definidas por el usuario.

Primeros pasos con Spark



RDDs

- Son la principal abstracción de datos en spark
- Los RDDs están particionados en los nodos del cluster
- Se suelen crear a partir de un fichero del HDFS
- Usan la evaluación perezosa

Primeros pasos con Spark



EVALUACIÓN PEREZOSA (Lazy evaluation)

- Los RDD usan evaluación perezosa
- Mantiene todas las transformaciones en un DAG
- Cuando se lanza una acción, se resuelve el grafo

Primeros pasos con Spark



Evaluación perezosa:

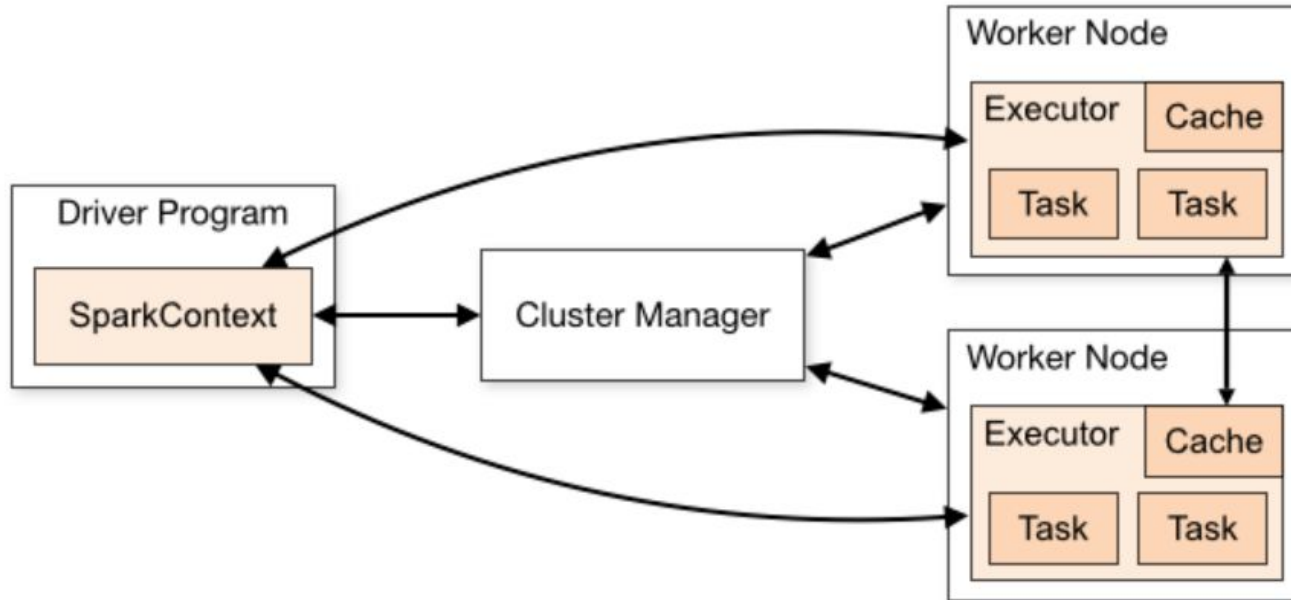
Es una estrategia de evaluación que retrasa el cálculo de una expresión hasta que su valor sea necesario, y que también evita repetir la evaluación en caso de ser necesaria en posteriores ocasiones. Esta compartición del cálculo puede reducir el tiempo de ejecución de ciertas funciones de forma exponencial, comparado con otros tipos de evaluación.

Los beneficios de la evaluación perezosa son:

- El incremento en el rendimiento al evitar cálculos innecesarios, y en tratar condiciones de error al evaluar expresiones compuestas.
- La capacidad de construir estructuras de datos potencialmente infinitas.
- La capacidad de definir estructuras de control como abstracciones, en lugar de operaciones primitivas.

Primeros pasos con Spark

SparkContext



Primeros pasos con Spark



SPARKCONTEXT()

- Punto de entrada a Spark
- La mayoría de las operaciones que usamos vienen de SparkContext
- Sólo se puede crear una instancia SparkContext

Se crea del siguiente modo:

```
from pyspark import SparkContext
```

```
sc = SparkContext()
```

ó

```
sc = SparkContext.getOrCreate()
```

Primeros pasos con Spark



RDD desde fichero

- `sc.textFile('MIFICHERO.txt')`
- Un registro una línea
- Ruta Absoluta o relativa

Primeros pasos con Spark



Tipos de operaciones

- Acciones

Operaciones que devuelven un valor

- Transformaciones

Devuelven otro RDD

Primeros pasos con Spark



TRANSFORMACIONES

- Filter()

Filter: Toma una función que devuelve un nuevo RDD formado por los elementos que pasen una función filtro. Esta función se puede utilizar para limpiar un RDD de entrada

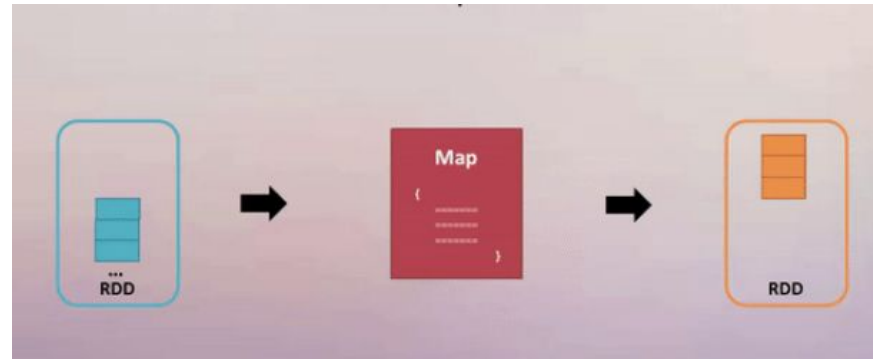
- Map()

Map : Toma una función y la aplica a cada elemento del RDD, el resultado de la función será utilizado para crear el nuevo elemento en el nuevo RDD

- flatMap()

FlatMap: En términos generales, transforma un RDD de longitud N en una colección de N colecciones, luego las aplanas en un solo RDD de resultados.

Primeros pasos con Spark



Primeros pasos con Spark



Takes one element and produces zero, one, or more elements



EJERCICIO PROPUESTO



Ejercicio propuesto 1.

A partir de la lista siguiente:

`['Alicante','Elche','Valencia','Madrid','Barcelona','Bilbao','Sevilla']`

- a) Usando lo que hemos aprendido hasta ahora, quédate sólo con las ciudades que tengan la letra 'e' en su nombre y muestralas
- b) Muestra las ciudades que tienen la letra 'e' y muestra el número de veces que aparece en cada nombre. Por ejemplo (Elche,2)
- c) Podrías quedarte solo con las ciudades que tengan una sola e?
- d) Han pasado una nueva lista pero no han separado bien las ciudades.. podrias volver a contar cuantas e hay en cada ciudad?

`ciudades_mal =`

`[['Alicante.Elche','Valencia','Madrid.Barcelona','Bilbao.Sevilla'],
['Murcia','San Sebastián','Melilla.Merida']]`

Primeros pasos con Spark



TRANSFORMACIONES

- `Sample()`
- `Distinct()`
- `GroupBy()`

EJERCICIO PROPUESTO



Ejercicio propuesto 2.

a) Dada una lista de nombres, agruparlos según su inicial

Lista de nombres:

["Juan", "Jimena", "Luis", "Cristian", "Laura", "Lorena", "Cristina", "Jacobo", "Jorge", "Lorena"]

Por ejemplo de la lista: Laura, Luis, Juan el resultado sería:

L Laura,Luis

J Jua

b) De la lista original obten una muestra sin repetir valores

c) Devuelve una muestra de datos de aproximadamente la mitad de registros que la lista original con datos que no se repitan

Primeros pasos con Spark



TRANSFORMACIONES

- Union()
- Intersection()
- Subtract()
- Cartesian()

EJERCICIO PROPUESTO



Ejercicio propuesto 3.

Dada las listas

Inglés: hello, table, angel, cat, dog, animal, chocolate, dark, doctor, hospital, computer

Español: hola, mesa, angel, gato, perro, animal, chocolate, oscuro, doctor, hospital, ordenador

Usando las transformaciones que hemos aprendido realiza las siguientes tareas:

- 1.- Obtén las palabras que se dicen igual en inglés y en español
- 2.- Obtén las palabras que en español son distintas que en inglés
- 3.- Obtén una única lista con las palabras en ambos idiomas, que son distintas entre ellas
- 4.- Haz dos grupos con todas las palabras, uno con las que empiezan por vocal y otro que empiecen por consonante

Primeros pasos con Spark



ACCIONES

- `Collect()`
- Alternativas `Collect()`

EJERCICIO PROPUESTO



Ejercicio propuesto 4

Dada una lista de elementos desordenados y algunos repetidos, devolver una muestra de 5 elementos, que estén en la lista,

sin repetir y ordenados descendientemente

lista = 4,6,34,7,9,2,3,4,4,21,4,6,8,9,7,8,5,4,3,22,34,56,98

b)Selecciona el elemento mayor de la lista resultante

Primeros pasos con Spark



ACCIONES

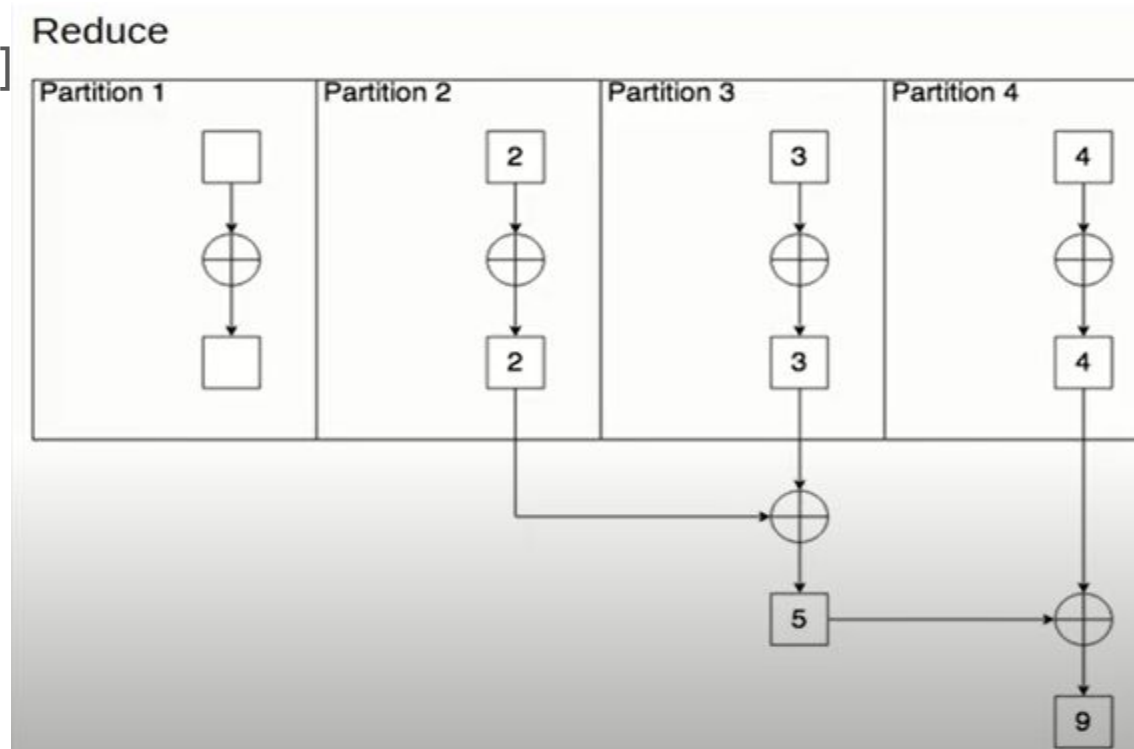
- `Reduce()`
- `Fold()`
- `Aggregate()`

Primeros pasos con Spark

REDUCE

```
rdd = sc.parallelize([2,3,4])
```

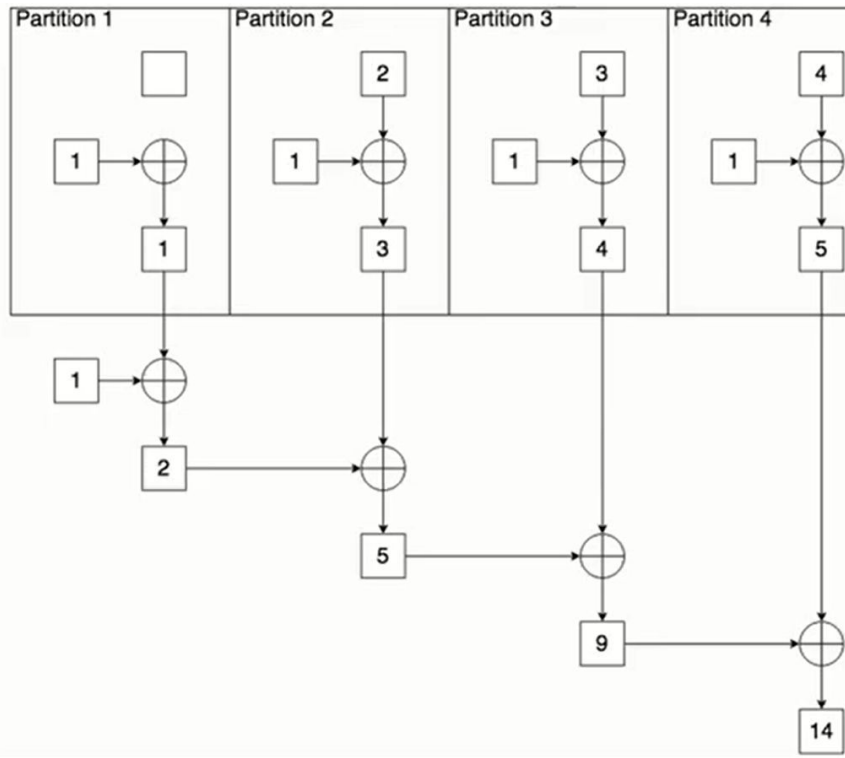
```
rdd.reduce(sumar))
```



Primeros pasos con Spark

```
RDD = sc.parallelize([2,3,4])
```

```
fold(1,add)
```



EJERCICIO PROPUESTO



Ejercicio propuesto 5

a) Dado una lista de vocales: vocales = a,e,i,o,u

Une todas las letras en un único valor dando como resultado::

A-E-I-O-U

b) Sabiendo que la 'a' suma 1 la 'e' suma 2 la 'i' suma 3 la 'o' suma 4 y la 'u' suma 5, y que cualquier consonante resta 1...

Dada esta lista:

'a','x','r','e','i','i','a','m','p','o','u','y','r'

Se pide sumar el total de cada partición y además que por cada partición se sumen 10 extra.

Primeros pasos con Spark



OTRAS ACCIONES

- Contar elementos
 - `count()`
 - `countApprox()`
 - `countApproxDistinct()`

EJERCICIO PROPUESTO



Ejercicio propuesto 6

a) Dada la frase ¿Hola que tal estas?

Cuenta cuantas veces se repite cada carácter

Por ejemplo:

H 1, a 3...

b) Crea un rdd a partir del fichero 'texto_grande.txt' como podriamos hacer una aproximación de las lineas que hay?

c) Y una aproximación de las frases distintas?

Práctica Repaso



Proyecto 1

Esta práctica se debe entregar en: fvalero@teralco.com

Dado el libro de El Quijote en txt:

- a) Crear un RDD a partir del fichero
- b) ¿Cómo verías una muestra del RDD?
- c) Contar cuantas veces aparece la palabra
 - Dulcinea
 - Quijote
 - Rocinante
- d) Devuelve una lista ordenada según el número de veces que sale cada palabra de menos a más

Práctica Repaso



PISTA

Tendrás que investigar cómo usar los métodos

`string.split()` y `string.strip()`

Las palabras pueden aparecer también en mayúsculas o minúsculas, tendrás que usar una función que te pase a mayusculas o minusculas.

APACHE SPARK

Sesión 1