EJERCICIO SESIÓN 2 SPARK - LOGS - RDD PARES

El objetivo de la siguiente práctica es que el alumno se familiarice con las operaciones básicas sobre HDFS y YARN. Para ello se propone una serie de ejercicios que deben ir resolviéndose gradualmente para ir adquiriendo el conocimiento y destrezas necesarias para operar en el entorno distribuido de Hadoop.

Sumario

Solución Ejercicio de Logs (RDD Pares)

Dados unos logs de una web, queremos saber cuantas veces ha accedido cada usuario, y obtener los 10 que más han accedido

a)Carga los logs de la carpeta weblogs(por hacerlo más rápido puedes coger solo uno de ellos)

SOLUCIÓN	**#Podemos cargar todos los ficheros de esta forma**
	`logs=sc.textFile("recursos/weblogs/*")`
	#O para hacer el ejercicio y que sea más ligero podemos coger solo uno de ellos
	`logs=sc.textFile("recursos/weblogs/2013-09-15.log")`

b)Crea un RDD de pares a partir de estos ficheros con el usuario como clave y un valor 1.

userid1,1

userid2,1

userid3,1

...

SOLUCIÓN	**#Obtenemos cada usuario en formato tupla, con el valor = 1** `usuarios_pair = logs.map(lambda linea:`

	`linea.split()).map(lambda words: (words[2].strip(),1))`
	
	#Consultamos que el resultado sea correcto
	`usuarios_pair.take(10)`

c) Suma los valores de cada ID de usuario

useid1,4

userid2,9

SOLUCIÓN	**#Usamos la función reduceByKey para sumar y obtener el total de veces que ha accedido cada usuarios**
	
	`usuarios_pair_sum = usuarios_pair.reduceByKey(lambda count1,count2: count1 + count2)`
	#Consultamos el resultado
	`usuarios_pair_sum.sortByKey(True).take(10)`

d) Muestra los 10 usuarios con máximas apariciones

SOLUCIÓN	**#Si hacemos un sortByKey, nos va a ordenar por clave pero nos interesa ordenar por valor:** `usuarios_pair_sum.sortByKey(True).take(10)`

	#Le damos la vuelta a la tupla, ordenamos, y le volvemos a dar el formato correcto `usuarios = usuarios_pair_sum.map(lambda pair: (pair[1],pair[0])).sortByKey(False).map(lambda pair: (pair[1],pair[0]))` **#false es orden descendente**

e) Crea un rdd de pares donde la clave sea el usuario y el valor una lista de Ips desde donde se ha conectado el usuario

SOLUCIÓN	userips = logs.map(lambda line: line.split()).map(lambda words: (words[2],words[0])).groupByKey() **#Consultamos, nos devuelve el usuario y una lista de ips iterable** userips.take(10) **#Consultamos las listas iterables con un bucle** usuarios_ips = [(x,list(y)) for x,y in userips.collect()]

f) En el cvs de cuentas, tenemos información de los clientes, podrías añadir el nombre y apellido del usuario junto a las apariciones de ese usuario?

SOLUCIÓN	**#Obtenemos la información que nos interesa del excel** usuario_nombre = sc.textFile("cuentas.csv") \ .map(lambda s: s.split(',')) \ .map(lambda account: (account[0],account[3] +' ' + account[4])) usuario_nombre.collect() **#Cruzamos ambos RDDs por nuestra key** usuario_nombre.join(usuarios).collect()