



1

BIG DATA Y MINERÍA DE DATOS GEOESPACIALES
3 PROCESAMIENTO DISTRIBUIDO DE DATOS

- Popularizado por Google a partir de la publicación:
J. Dean and, S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Communications of the ACM, Jan 2008, VI 51 No. 1.

3.1 MapReduce

A. Martín

2

BIG DATA Y MINERÍA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

0. **(Splitting)**. Partición de los datos (si es necesario y posible)
1. **(Map)**. Diseñar un conjunto de tareas simples “map” que sobre una partición de los datos de entrada generan un conjunto de resultados intermedios, que son pares (clave,valor)
2. **(Shuffle and Sort)**. Los resultados intermedios (pares) son agrupados y ordenados (por clave).
3. **(Reduce)**. Los pares ordenados por clave son procesados por otro conjunto de tareas simples “reduce” para producir el resultado.

3.1 MapReduce

A. Martín

3

BIG DATA Y MINERÍA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

Fichero de texto donde cada línea contiene medidas meteorológicas de una estación de un país
(id_país, año, temp_media, temp_máx, temp_min)

Fichero	Map	Shuffle & Sort & Group	Reduce	Salida
21 1904 25 37 -24 12 1915 13 42 -11 14 1902 17 44 -46 13 1904 21 39 -54 45 1902 57 41 -93 13 1907 21 37 -31 ...	(1904, 37) (1915, 42) (1902, 44) (1904, 39) (1902, 41) ...	(1902, [44,41]) (1904, [37,39]) (1915, 42) ...	(1902, 44) (1904, 39) (1915, 42) ...	1902, 44 1904, 39 1915, 42 ...

cat data.txt | mapper.py | sort | reducer.py > salida.txt

(1902, 44)
(1902, 41)
(1904, 37)
(1904, 39)
(1915, 42)

3.1 MapReduce

A. Martín

4

BIG DATA Y MINERIA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

- Especially designed to function on enormous quantities of **distributed data**

Supongamos que queremos procesar los datos de una base de datos, ¿dónde ejecutaríamos sus operaciones?

3.1 MapReduce

A. Martín

5

BIG DATA Y MINERIA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

Splitting

Nodo 1

21	1904	25	37	-24
12	1915	13	42	-11
14	1902	17	44	-46

Map

(1904, 37)
(1915, 42)
(1902, 44)

Nodo 2

13	1904	21	39	-54
45	1902	57	41	-93

Map

(1904, 39)
(1902, 41)

Shuffling

Sort & Group

(1902, [44, 41])
(1904, [37, 39])
(1915, 42)
...

Reduce

(1902, 44)
(1904, 39)
(1915, 42)
...

Salida

1902, 44
1904, 39
1915, 42
...

3.1 MapReduce

A. Martín

6

BIG DATA Y MINERIA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

- MapReduce es un entorno de programación distribuida sobre datos distribuidos
- Reduce la complejidad natural de un sistema distribuido
- Aprovecha la localidad de los datos
- Permite procesar gran cantidad de datos de forma eficiente

3.1 MapReduce
NODE 04
A. Martín

7

BIG DATA Y MINERIA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

EJERCICIO: Conjunto de datos de Compras, fichero purchases.txt

```
angel@angel-VirtualBox:~/Documents/BigData/mapreduce/purchases$ head -n 5 purchases.txt
date, time, store, item, cost, payment
2012-01-01,09:00,San Jose,Men's Clothing,214.05,Amex
2012-01-01,09:00,Fort Worth,Women's Clothing,153.57,Visa
2012-01-01,09:00,San Diego,Music,66.08,Cash
2012-01-01,09:00,Pittsburgh,Pet Supplies,493.51,Discover
angel@angel-VirtualBox:~/Documents/BigData/mapreduce/purchases$
```

Calcular los valores totales por tienda (por ciudad)

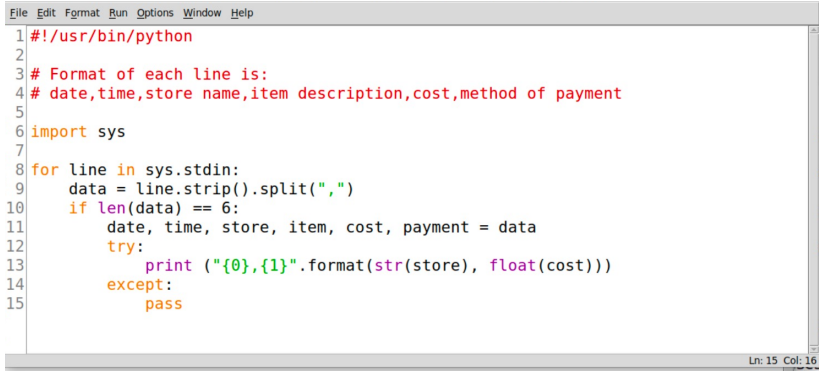
3.1 MapReduce
NODE 04
A. Martín

8

BIG DATA Y MINERIA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

MAPPER.PY



```

1  #!/usr/bin/python
2
3  # Format of each line is:
4  # date,time,store name,item description,cost,method of payment
5
6  import sys
7
8  for line in sys.stdin:
9      data = line.strip().split(",")
10     if len(data) == 6:
11         date, time, store, item, cost, payment = data
12         try:
13             print ("{}},{1}".format(str(store), float(cost)))
14         except:
15             pass

```

Ln: 15 Col: 16

3.1 MapReduce

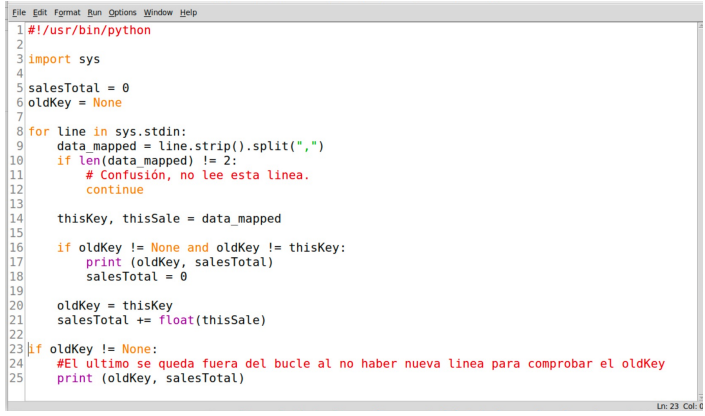

A. Martín

9

BIG DATA Y MINERIA DE DATOS GEOESPACIALES

3 PROCESAMIENTO DISTRIBUIDO DE DATOS

REDUCER.PY



```


1  #!/usr/bin/python
2
3  import sys
4
5  salesTotal = 0
6  oldKey = None
7
8  for line in sys.stdin:
9      data_mapped = line.strip().split(",")
10     if len(data_mapped) != 2:
11         # Confusión, no lee esta línea.
12         continue
13
14     thisKey, thisSale = data_mapped
15
16     if oldKey != None and oldKey != thisKey:
17         print (oldKey, salesTotal)
18         salesTotal = 0
19
20     oldKey = thisKey
21     salesTotal += float(thisSale)
22
23 if oldKey != None:
24     #EL ultimo se queda fuera del bucle al no haber nueva linea para comprobar el oldKey
25     print (oldKey, salesTotal)

```

Ln: 23 Col: 0

cat purchases.txt | python3 mapper.py | sort | python3 reducer.py > salida.txt

3.1 MapReduce


A. Martín

10