Portafolio CBD

Alumno: Javier Meliá Sevilla

Práctica de MapReduce y Apache Hadoop

Los ejercicios 1,2 y 3 no son evaluables por eso solo se entregan de 4 al 8 si hiciera falta entregar alguno de los anteriores están también realizados.

Ejercicio 4:

Mapper:

```python
#!/usr/bin/python
# Format of each line is:
# date\ttime\tstore name\titem description\tcost\tmethod of payment
# We want elements 2 (store name) and 4 (cost)
# We need to write them out to standard output, separated by a tab
import sys
for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 6:
        date, time, store, item, cost, payment = data
        print "{0}\t{1}".format(item, cost)
```

Reducer:

```python
#!/usr/bin/python
import sys
ventasTotales = 0
oldKey = None
# Loop around the data
# It will be in the format key\tval
# Where key is the store name, val is the sale amount
# All the sales for a particular store will be presented,
# then the key will change and we'll be dealing with the next store

for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue
    thisKey, thisSale = data_mapped
    if oldKey and oldKey != thisKey:
        print oldKey, "\t", ventasTotales
        oldKey = thisKey;
        ventasTotales = 0
    oldKey = thisKey
    ventasTotales += float(thisSale)

if oldKey != None:
    print oldKey, "\t",
```

Ejercicio 5:

Mapper:

```
#!/usr/bin/python

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 6:
        date, time, store, item, cost, payment = data
        print "{0}\t{1}".format(store, cost)
```

Reducer:

```
#!/usr/bin/python

import sys
valorMaximo = 0
oldKey = None
for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisKey, thisSale = data_mapped

    if oldKey and oldKey != thisKey:
        print oldKey, "\t", valorMaximo
        oldKey = thisKey;
        valorMaximo = 0

    oldKey = thisKey

    if valorMaximo < float(thisSale):
        valorMaximo = float(thisSale)

if oldKey != None:
    print oldKey, "\t", valorMaximo
```

Ejercicio 6:

Mapper:

```python
#!/usr/bin/python

import sys

for line in sys.stdin:
    data = line.strip().split(" ")
    if len(data) == 10:
        ip, client, user, date, time, req, path, prot, res, size = data
        print "{0}\t{1}".format(path, 1)
```

Reducer:

```python
#!/usr/bin/python

import sys

salesTotal = 0
oldKey = None
for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisKey, thisSale = data_mapped

    if oldKey and oldKey != thisKey:
        print oldKey, "\t", salesTotal
        oldKey = thisKey;
        salesTotal = 0

    oldKey = thisKey
    salesTotal += int(thisSale)

if oldKey != None:
    print oldKey, "\t", salesTotal
```

Ejercicio 7:

Mapper:

```python
#!/usr/bin/python
import sys

for line in sys.stdin:
    data = line.strip().split(" ")
    if len(data) == 10:
        ip, client, user, date, time, req, path, prot, res, size = data
        if ip == '10.223.157.186':
            print "{0}\t{1}".format(ip, 1)
```

Reducer:

```python
#!/usr/bin/python

import sys

salesTotal = 0
oldKey = None

for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisKey, thisSale = data_mapped

    salesTotal += int(thisSale)

print  salesTotal
```

Ejercicio 8:

Mapper:

```python
#!/usr/bin/python
import sys

for line in sys.stdin:
    data = line.strip().split(" ")
    if len(data) == 10:
        ip, client, user, date, time, req, path, prot, res, size = data
        print "{0}\t{1}".format(path, 1)
```

Reducer:

```python
#!/usr/bin/python

import sys

salesTotal = 0
oldKey = None
maxx = 0
maxPath = ''

for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisKey, thisSale = data_mapped

    if oldKey and oldKey != thisKey:
        #print oldKey, "\t", salesTotal
        if salesTotal > maxx:
        maxPath = oldKey
            maxx = salesTotal
        oldKey = thisKey;

        salesTotal = 0

    oldKey = thisKey
    salesTotal += int(thisSale)

if oldKey != None:
    if salesTotal > maxx:
        maxPath = oldKey
        maxx = salesTotal

print maxPath, "\t", maxx
```

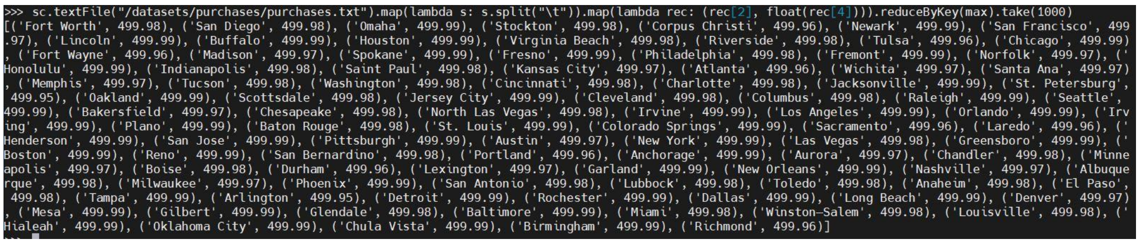
En Principio todos los resultados obtenidos por los códigos correspondientes son los mismos que los que se indican en el boletín.

Ejercicios evaluables de Apache Spark

Para estos ejercicios primero se adjunta la línea de código que se ha utilizado para obtener el resultado y un pantallazo de este mismo.

Ejercicio 0: Para cada tienda, obtener la transacción de máximo importe.

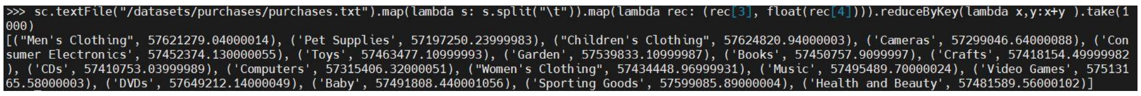
```
sc.textFile("/datasets/purchases/purchases.txt").map(lambda s:
s.split("\t")).map(lambda rec: (rec[2], float(rec[4]))).reduceByKey(max).take(1000)
```



```
>>> sc.textFile("/datasets/purchases/purchases.txt").map(lambda s: s.split("\t")).map(lambda rec: (rec[2], float(rec[4]))).reduceByKey(max).take(1000)
[('Fort Worth', 499.98), ('San Diego', 499.98), ('Omaha', 499.99), ('Stockton', 499.98), ('Corpus Christi', 499.96), ('Newark', 499.99), ('San Francisco', 499
.97), ('Lincoln', 499.99), ('Buffalo', 499.99), ('Houston', 499.99), ('Virginia Beach', 499.98), ('Riverside', 499.98), ('Tulsa', 499.96), ('Chicago', 499.99)
, ('Fort Wayne', 499.96), ('Madison', 499.97), ('Spokane', 499.99), ('Fresno', 499.99), ('Philadelphia', 499.98), ('Fremont', 499.99), ('Norfolk', 499.97), ('
Honolulu', 499.99), ('Indianapolis', 499.98), ('Saint Paul', 499.98), ('Kansas City', 499.97), ('Atlanta', 499.96), ('Wichita', 499.97), ('Santa Ana', 499.97)
, ('Memphis', 499.97), ('Tucson', 499.98), ('Washington', 499.98), ('Cincinnati', 499.98), ('Charlotte', 499.98), ('Jacksonville', 499.99), ('St. Petersburg',
 499.95), ('Oakland', 499.99), ('Scottsdale', 499.98), ('Jersey City', 499.99), ('Cleveland', 499.98), ('Columbus', 499.98), ('Raleigh', 499.99), ('Seattle',
499.99), ('Bakersfield', 499.97), ('Chesapeake', 499.98), ('North Las Vegas', 499.98), ('Irvine', 499.99), ('Los Angeles', 499.99), ('Orlando', 499.99), ('Irv
ing', 499.99), ('Plano', 499.99), ('Baton Rouge', 499.98), ('St. Louis', 499.99), ('Colorado Springs', 499.99), ('Sacramento', 499.96), ('Laredo', 499.96), ('
Henderson', 499.99), ('San Jose', 499.99), ('Pittsburgh', 499.99), ('Austin', 499.97), ('New York', 499.99), ('Las Vegas', 499.98), ('Greensboro', 499.99), ('
Boston', 499.99), ('Reno', 499.99), ('San Bernardino', 499.98), ('Portland', 499.96), ('Anchorage', 499.99), ('Aurora', 499.97), ('Chandler', 499.98), ('Minne
apolis', 499.97), ('Boise', 499.98), ('Durham', 499.96), ('Lexington', 499.97), ('Garland', 499.99), ('New Orleans', 499.99), ('Nashville', 499.97), ('Albuque
rque', 499.98), ('Milwaukee', 499.97), ('Phoenix', 499.99), ('San Antonio', 499.98), ('Lubbock', 499.98), ('Toledo', 499.98), ('Anaheim', 499.98), ('El Paso',
 499.98), ('Tampa', 499.99), ('Arlington', 499.95), ('Detroit', 499.99), ('Rochester', 499.99), ('Dallas', 499.99), ('Long Beach', 499.99), ('Denver', 499.97)
, ('Mesa', 499.99), ('Gilbert', 499.99), ('Glendale', 499.98), ('Baltimore', 499.99), ('Miami', 499.98), ('Winston-Salem', 499.98), ('Louisville', 499.98), ('
Hialeah', 499.99), ('Oklahoma City', 499.99), ('Chula Vista', 499.99), ('Birmingham', 499.99), ('Richmond', 499.96)]
```

Ejercicio 1: Suma total de ventas para cada categoría de producto.

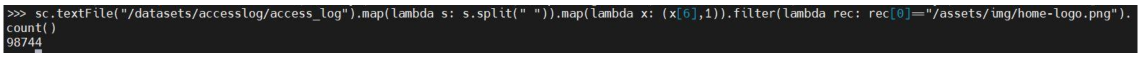
```
sc.textFile("/datasets/purchases/purchases.txt").map(lambda s:
s.split("\t")).map(lambda rec: (rec[3], float(rec[4]))).reduceByKey(lambda x,y:x+y
).take(1000)
```



```
>>> sc.textFile("/datasets/purchases/purchases.txt").map(lambda s: s.split("\t")).map(lambda rec: (rec[3], float(rec[4]))).reduceByKey(lambda x,y:x+y ).take(1
000)
[("Men's Clothing", 57621279.04000014), ('Pet Supplies', 57197250.23999983), ("Children's Clothing", 57624820.94000003), ('Cameras', 57299046.64000088), ('Con
sumer Electronics', 57452374.130000055), ('Toys', 57463477.10999993), ('Garden', 57539833.10999987), ('Books', 57450757.9099997), ('Crafts', 57418154.49999982
), ('CDs', 57410753.03999989), ('Computers', 57315406.32000051), ("Women's Clothing", 57434448.96999931), ('Music', 57495489.70000024), ('Video Games', 575131
65.58000003), ('DVDs', 57649212.14000049), ('Baby', 57491808.440001056), ('Sporting Goods', 57599085.89000004), ('Health and Beauty', 57481589.56000102)]
```

Ejercicio 2: Número total de accesos al recurso "/assets/img/home-logo.png"

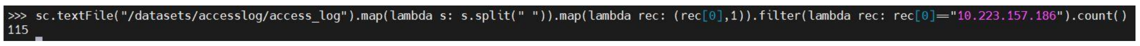
```
sc.textFile("/datasets/accesslog/access_log").map(lambda s: s.split(" ")).map(lambda
x: (x[6],1)).filter(lambda rec: rec[0]=="/assets/img/home-logo.png").count()
```



```
>>> sc.textFile("/datasets/accesslog/access_log").map(lambda s: s.split(" ")).map(lambda x: (x[6],1)).filter(lambda rec: rec[0]=="/assets/img/home-logo.png").
count()
98744
```

Ejercicio 3: Número total de accesos desde la misma dirección IP: 10.223.157.186

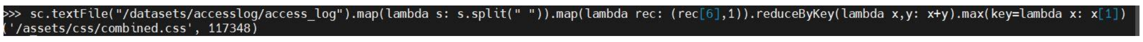
```
sc.textFile("/datasets/accesslog/access_log").map(lambda s: s.split(" ")).map(lambda
rec: (rec[0],1)).filter(lambda rec: rec[0]=="10.223.157.186").count()
```



```
>>> sc.textFile("/datasets/accesslog/access_log").map(lambda s: s.split(" ")).map(lambda rec: (rec[0],1)).filter(lambda rec: rec[0]=="10.223.157.186").count()
115
```

Ejercicio 4: Recurso web con mayor número de accesos

```
sc.textFile("/datasets/accesslog/access_log").map(lambda s: s.split(" ")).map(lambda
rec: (rec[6],1)).reduceByKey(lambda x,y: x+y).max(key=lambda x: x[1])
```



```
>>> sc.textFile("/datasets/accesslog/access_log").map(lambda s: s.split(" ")).map(lambda rec: (rec[6],1)).reduceByKey(lambda x,y: x+y).max(key=lambda x: x[1])
('/assets/css/combined.css', 117348)
```