TP2 - Spark

Marion Le Béchec

Question 1

After cutting and pasting the code given in the subject, I have made two changes.

First, I changed the value in the map function of the definition of the RDD « pairs » : I put 1 instead of 2 (*line 4)*, in order to have the right values in the result. (With the value 2, each count is multiplied by two).

Second, at the line 7, I put the command « words.count( ) » for the argument of the .take( ) function, in order to take all the words of the text into account and not just the first 5 words (as the command « counts.take(5) only takes the 5 first elements of my RDD « counts »).

Here is the result when I run my cell :

```
1  text_file = sc.textFile("/FileStore/tables/dataset/steve.txt")
2  words = text_file.flatMap(lambda line: line.split(" "))
3
4  pairs = words.map(lambda s: (s, 1))
5
6  counts = pairs.reduceByKey(lambda a, b: a + b)
7  for (count, word) in counts.take((words.count())) :
8    print ("%s: %i" % (count, word))
```

▸ (3) Spark Jobs

```
Steven: 1
Jobs: 23
(/dʒɒbz/;: 1
was: 33
an: 10
American: 2
business: 2
inventor,: 1
chief: 1
executive: 1
officer: 1
co-founder: 2
of: 41
Apple: 11
Inc.;: 1
CEO: 3
shareholder: 1
The: 6
Walt: 1
board: 1
following: 1
```

```
"without: 1
upped: 1
move: 1
anyone: 1
bring: 1
thought: 1
this: 1
best: 1
"doctor: 1
sheltered: 1
quietly: 1
gave: 1
well-educated,: 1
wealthy."[18]: 1
changed: 1
however,: 1
adopt: 1
boy: 1
placed: 2
Jobs,: 1
neither: 1
```

Question 2

We want to get the 5 words with the most occurrences in the text. To that end, we are going to use the function sortByKey( ). This function sorts as per the first argument : here, it will sort the

words in an alphabetical order. To avoid this problem, we are going to invert the two arguments of the RDD « counts », then we will sort the RDD in a descending order (we want to have the 5 biggest occurrences !) and finally we will invert against the key and the value of each pair.

Here is the result :

```
counts_desc = counts.map(lambda s : (s[1], s[0])).sortByKey(False).map(lambda s : (s[1], s[0]))
for (count, word) in counts_desc.take(5) :
  print ("%s: %i" % (count, word))
```

▸ (3) Spark Jobs

```
the: 66
and: 53
a: 45
to: 42
of: 41
```

Question 3

In order to get the top 5 words with the largest number of occurrences among the words containing at least 5 characters, we use the function filter on the RDD « counts_desc » and keep the words that have 5 characters or more.

Here is the result :

```
most_words = counts_desc.filter(lambda s : len(s[0])>4)
for (count, word) in most_words.take(5) :
  print ("%s: %i" % (count, word))
```

▸ (1) Spark Jobs

```
Apple: 11
Jobs's: 8
Jandali: 8
Clara: 8
Schieble: 8
```

Question 4

First, I created the RDD « pages » that contains the id of each reference and the reference it self. The data are stored as : (id, reference)
Then, I created the RDD « newedge » that contains the all id of the references, form this RDD I created the RDD « occurences » that counts the occurrences of each reference. The data are stored as : (id, number of occurrences).

Finally, after joining the two RDDs, I sorted the occurrences in a descending order.

Here is the result :

```
labels = sc.textFile("/FileStore/tables/dataset/idslabels.txt")
edgelist = sc.textFile("/FileStore/tables/dataset/edgelist.txt")
pages = labels.map(lambda line : (line.split(" ")[0], (" ").join(line.split(" ")[1:])))
newedge = edgelist.map(lambda line: (line[2:])).flatMap(lambda line: line.split(" "))
occurences = newedge.map(lambda s : (s,1)).reduceByKey(lambda a, b: a + b)
results = occurences.join(pages).map(lambda s: (s[1][0], (s[0],s[1][1]))).sortByKey(False)
print (results.take(10))

for  (count, (id_name, name)) in results.take(10) :
  print ("%s: %i" % (name, count))
```

▸ (4) Spark Jobs

```
[(8145, ('60589', 'United States')), (7799, ('30594', 'France')), (5740, ('24449', 'Communes of France')), (5299, ('26539', 'Departments of France')), (4064, ('51359', 'Regions of France')), (3832, ('23683', 'City')), (3527, ('52174', 'Romania')), (2978, ('20409', 'Category:Rivers in Romania')), (2799, ('59931', 'Tributary')), (2277, ('28563', 'England'))]
United States: 8145
France: 7799
Communes of France: 5740
Departments of France: 5299
Regions of France: 4064
City: 3832
Romania: 3527
Category:Rivers in Romania: 2978
Tributary: 2799
England: 2277
```