Respostas teste Semantix

1) **Qual o objetivo do comando cache em Spark?**

O comando cache serve para otimizar o algoritmo. Com esse comando o Spark armazena os dados em cache sendo esta uma memoria mais rápida dos que outras como por exemplo o disco rígido.

2) **O mesmo código implementado em Spark é normalmente mais rápido que a implementação equivalente em MapReduce. Por quê?**

O MapReduce lê os dados em uma memória “lenta” como o disco rígido, processa e depois os escreve na memória novamente. Já o Spark é mais rápido pois o processamento em geral é realizado em memória “rápida” como a cache.

3) **Qual é a função do SparkContext?**

Passar as informações de configuração, como alocação de memória, para que o Spark saiba como acessar o cluster.

4) **Explique com suas palavras o que é Resilient Distributed Datasets (RDD).**

É uma representação de um conjunto de dados. Cada conjunto de dados no RDD é dividido entre os diferentes nós do cluster para que possam ser processados em paralelo. O termo “Resilient” significa que são tolerantes a falhas, ou seja, nas falhas dos nós é possível recomputar dados perdidos ou danificados.

5) **GroupByKey é menos eficiente que reduceByKey em grandes dataset. Por quê?**

No ReduceByKey o Spark faz uma redução de dados combinando estes pela chave comum em cada partição antes de enviar os dados para o processamento, por isso o termo “reduce”. Já o GroupByKey não faz essa redução e o conjunto de dados enviados para processamento são maiores.

6) Explique o que o código Scala abaixo faz

```
val textFile = sc.textFile("hdfs://...")
val counts = textFile.flatMap(line => line.split(" "))
                 .map(word => (word, 1))
                 .reduceByKey(_ + _)
counts.saveAsTextFile("hdfs://...")
```

Na primeira linha lê-se um arquivo texto através do comando “sc.textFile” e o armazena na variável *textFile*. Em seguida, o texto é quebrado em palavras com o comando “split” e pela linha 3 é feito um mapeamento, ou seja, cada palavra recebe um atributo que no caso seria uma chave. Na 4 linha é aplicado o comando “reduceByKey” para agregar as palavras por chaves. De certa forma, o comando anterior organiza e faz a contagem das palavras. Por fim, a contagem das palavras é armazenada em um arquivo texto.