Prezado candidato.

Gostaríamos de fazer um teste que será usado para sabermos a sua proficiência nas habilidades para a vaga. O teste consiste em algumas perguntas e exercícios práticos sobre Spark e as respostas e códigos implementados devem ser armazenados no GitHub. O link do seu repositório deve ser compartilhado conosco ao final do teste.

Quando usar alguma referência ou biblioteca externa, informe no arquivo README do seu projeto. Se tiver alguma dúvida, use o bom senso e se precisar deixe isso registrado na documentação do projeto.

Qual o objetivo do comando **cache** em Spark?
Ao realizar ações, são gerados processos de computação de dados, mas podemos persistir os dados em CACHE para ser usado em outras ações, sem a necessidade de novos processamentos, tendo assim um ganho de performance.

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

Qual é a função do **SparkContext**?
Executar aplicações, chamar as "funções" e também o contexto geral, dando acesso aos recursos do Spark.

Explique com suas palavras o que é **Resilient Distributed Datasets** (RDD).
Estrutura principal do Spark, é como se fosse um dataframe do pandas, a diferença é que a RDD foi criada para rodar em ambiente distribuído.

GroupByKey é menos eficiente que **reduceByKey** em grandes dataset. Por quê?
Devido ao método de agrupamento das Keys. => reduceByKey faz o agrupamento das Keys primeiro, já no GroupByKey, o agrupamento, é feito somente depois. Para grandes volumes, o ganho de desempenho com reduceByKey é maior.

Explique o que o código Scala abaixo faz. Faz a contagem de palavras de um dataset

```
val textFile = sc.textFile("hdfs://...")        leitura dos dados/input
val counts = textFile.flatMap(line => line.split(" "))        Separação por espaços e submetidos a uma sequencia de caracteres pelo flatMap
             .map(word => (word, 1))        Utilizando o método map é atribuído um valor para cada chave
             .reduceByKey(_ + _)        utilizando o reduceByKey onde o agrupamento por chave é feito primeiro, e logo em seguida é feito a somatória dos "itens"
counts.saveAsTextFile("hdfs://...")        salvando o processo em um sistema distribuído.
```

HTTP requests to the NASA Kennedy Space Center WWW server

Fonte oficial do dateset: http://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html

Dados:

- Jul 01 to Jul 31, ASCII format, 20.7 MB gzip compressed, 205.2 MB.
- Aug 04 to Aug 31, ASCII format, 21.8 MB gzip compressed, 167.8 MB.

Sobre o dataset: Esses dois conjuntos de dados possuem todas as requisições HTTP para o servidor da NASA Kennedy Space Center WWW na Flórida para um período específico.

Os logs estão em arquivos ASCII com uma linha por requisição com as seguintes colunas:

- **Host fazendo a requisição**. Um hostname quando possível, caso contrário o endereço de internet se o nome não puder ser identificado.
- **Timestamp** no formato "DIA/MÊS/ANO:HH:MM:SS TIMEZONE"
- **Requisição (entre aspas)**
- **Código do retorno HTTP**
- **Total de bytes retornados**

Questões

Responda as seguintes questões devem ser desenvolvidas em Spark utilizando a sua linguagem de preferência.

1. Número de hosts únicos.
2. O total de erros 404.
3. Os 5 URLs que mais causaram erro 404.
4. Quantidade de erros 404 por dia.
5. O total de bytes retornados.