Qual o objetivo do comando cache em Spark?

- O comando cache faz com que os arquivos carregados persistam em memória, é bastante útil em processos iterativos que utilizam os mesmos dados mais que uma vez.

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

- O Spark é, em geral, mais rápido que o MapReduce porque ele prioriza o processamento em memória RAM, que é mais rápido, evitando operações em disco que são mais lentas.

Qual é a função do SparkContext?

- SparkContext é a interface com ambiente Spark. O SparkContext permite a conexão a um cluster do Spark e a inicialização dos serviços internos. Também permite a criação de RDDs e o acesso aos jobs em execução.

Explique com suas palavras o que é Resilient Distributed Dataset (RDD).

- RDD é uma abstração fornecida pelo Spark para a manipulação de dados que estão particionados em diversos nós de um cluster e que podem ser processados de forma paralela. A classe RDD fornece as ferramentas básicas para a manipulação dos dados de forma paralela.

GroupByKey é menos eficiente que ReduceByKey em grandes datasets. Por quê?

- O ReduceByKey combina os dados dentro do nó e só envia apenas uma chave pela rede, por isso é mais eficiente em grandes datasets que o GroupByKey que envia todos os dados pela rede.

Explique o que o código scala abaixo faz

O código lê um arquivo no hdfs, conta a quantidade de ocorrências de cada palavra e salva o resultado no hdfs.

val textFile = sc.textFile("hdfs://...") // Lê um arquivo gravado em hdfs

val counts = textFile.flatMap(line=>line.split("")) // Cria um RDD em que cada elemento é uma palavra de textFile

.map(word=>(word,1)) // Cria pares chave-valor, em que a chave é cada palavra no RDD e o valor é 1

.reduceByKey(_+_) //Agrupa as chaves iguais, somando os valores. Esse valor é registrado na variável counts

counts.saveAsTextFile("hdfs://...") // Salva o conteúdo de counts no hdfs