**Questões – Spark**

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

O Spark permite a inserção de conjuntos de dados em um cache de memória. Isso acaba sendo muito útil quando os dados são acessados repetidas vezes, reduzindo a sobrecarga de computação. Ele ajuda a salvar resultados parciais intermediários para que possam ser reutilizados em estágios subsequentes.

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

Uma das principais diferenças entre o Spark e o MapReduce é como eles processam os dados. O Spark faz tudo na memória enquanto o MapReduce persiste os dados no disco após realizar o map ou reduce dos trabalhos. Por isso o Spark acaba superando o MapReduce com facilidade. Portanto, desde que se tenha memória suficiente para ajustar os dados, o Spark se sairá melhor que o MapReduce.

1. **Qual é a função do SparkContext?**

O SparkContext configura serviços internos e estabelece uma conexão com o ambiente de execução do Spark. Nele podemos passar configurações para alocar recursos, como memória e processadores. Depois que o SparkContext é criado, você pode usá-lo para criar RDDs, acumuladores e variáveis ​​de broadcast, acessar serviços Spark e executar tarefas.

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

O RDD possui três características importantes: É resiliente, o que significa que ele é tolerante a falhas, tornando possível o recálculo em caso de falha em um dos nós; é distribuído, ou seja, os conjuntos de dados residem em vários nós e, também, possui conjunto de dados porque contém dados.

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

Ao utilizar a opção de groupByKey faz com que todos os pares de key-values trafeguem pelo cluster fazendo com que muitos dados desnecessários ocupem a rede e pode acontecer da memória principal não suportar a quantidade de dados, quando todos os pares são transmitidos. Já a opção de reduceByKey reduz primeiramente a lista de pares, combinando os dados antes de enviar para a rede, dessa maneira teremos menos tráfego e uso de memória é reduzido, por isso torna-se mais eficiente quando trabalha principalmente com grandes conjuntos de dados.

1. **Explique o que o código Scala abaixo faz.**

A close up of a logo

Description automatically generated

Primeiro é lido um arquivo do tipo texto que está salvo no HDFS. Depois o script quebra as linhas em palavras, transformando em uma única coleção de palavras. Depois cada elemento da coleção utiliza um contador começando em 1, basicamente conta a quantidade de vezes que a palavra aparece no arquivo. E, por último, o resultado é salvo no HDFS em formato texto.