Aplicações MapReduce – Top N HashTags

PROFESSORA

Rosangela de Fátima Pereira – rpereira@larc.usp.br

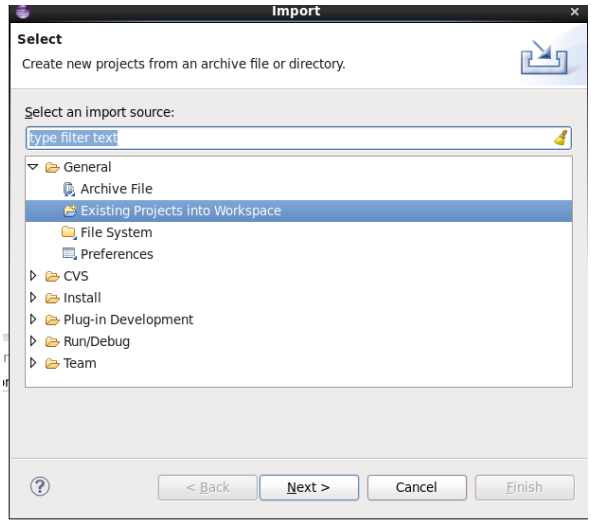
DESCRIÇÃO

Esse documento descreve os passos necessários para a implementação e execução de uma aplicação MapReduce que permite a contagem das top n *hashtags* encontradas em uma base de dados do Twitter.

ATIVIDADES

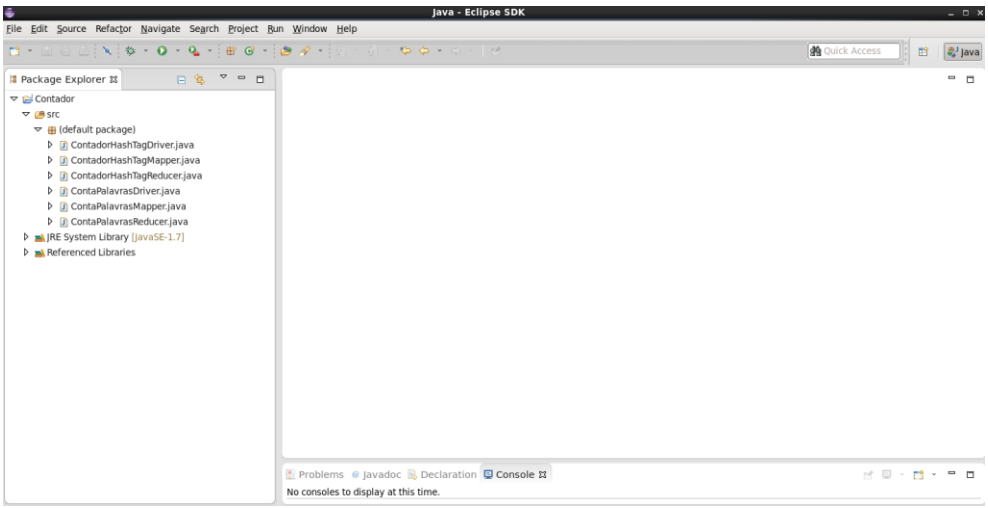
1. Abra a IDE Eclipse

2. Selecione a opção “File>>Import”. Essa ação deverá abrir uma janela similar à Figura a seguir:



3. Selecione a opção “Existing Projects into Workspace” e pressione o botão Next.

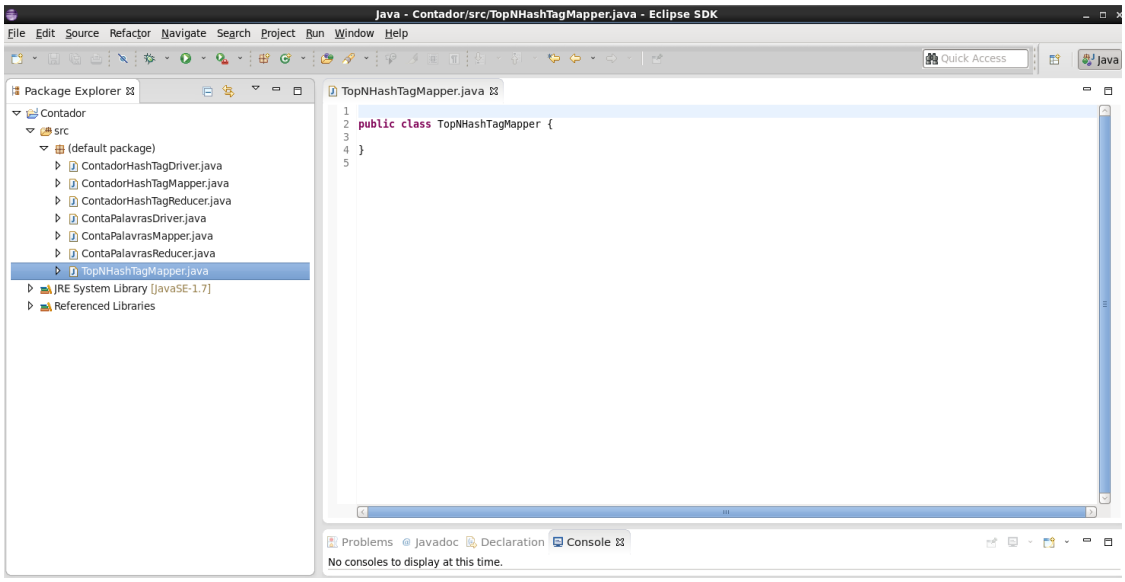
4. No campo “Select root directory” indique o seguinte caminho: /home/training/workspace/Contador Feito isso, pressione o botão Finish. O projeto deverá aparecer conforme imagem a seguir:



5. Esse projeto possui as classes necessárias para a aplicação que realiza a contagem de palavras (ContaPalavrasDriver, ContaPalavrasMapper, ContaPalavrasReduce) e para a aplicação que realiza a contagem de hashtags (ContadorHashTagDriver, ContadorHashTagMapper, ContadorHashTagReducer). Você deverá implementar agora as classes responsáveis pela aplicação que busca as top n hashtags mais encontradas na base de dados do Twitter, onde “n” é um parâmetro que deverá ser indicado pelo usuário.

6. Primeiramente deverá ser criado a classe TopNHashTagMapper. Para isso, clique com o botão direito do mouse sobre o nome do projeto e selecione a opção: New >> Class.

7. No campo “Name”, indique a classe TopNHashTagMapper. Após isso, pressione o botão Finish. Deverá aparecer a seguinte imagem:



8. Na classe TopNHashTagMapper, inserir o código a seguir:

```
import java.io.IOException;
import java.util.TreeMap;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;


public class TopNHashTagMapper extends Mapper<Object, Text, NullWritable, Text>
{
   private TreeMap<Integer, Text> topN = new TreeMap<>();

   private Text palavra = new Text();
```

```
    public void map(Object key, Text value, Context context)throws IOException, InterruptedException {

        //capturando o parametro qtd informado pelo usuario
      Configuration conf = context.getConfiguration();
      int qtd = Integer.parseInt(conf.get("qtd"));

      String[] listaHashtags = value.toString().toLowerCase().split("\t") ;

      //Entada: hashtag - qtd de hashtags
      if (listaHashtags.length < 2) {
            return;
      }

      topN.put(Integer.parseInt(listaHashtags[1].toLowerCase()), new Text(value));

      if (topN.size() > qtd) {
            topN.remove(topN.firstKey());
      }
    }

    protected void cleanup(Context context) throws IOException,
          InterruptedException {
      for (Text t : topN.values()) {
        context.write(NullWritable.get(), t);
      }
    }

}
```

9. Repita o mesmo procedimento da Etapa 5, para criar a classe TopNHashTagReducer, inserindo o código a seguir:

```
import java.io.IOException;
import java.util.TreeMap;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;



public class TopNHashTagReducer extends Reducer<NullWritable, Text, NullWritable, Text> {

        private TreeMap<Integer, Text> topN = new TreeMap<>();

        public void reduce(NullWritable key, Iterable<Text> values,
                        Context context) throws IOException, InterruptedException {

                for (Text value : values) {
                  String[] hashtags = value.toString().toLowerCase().split("\t") ;
```

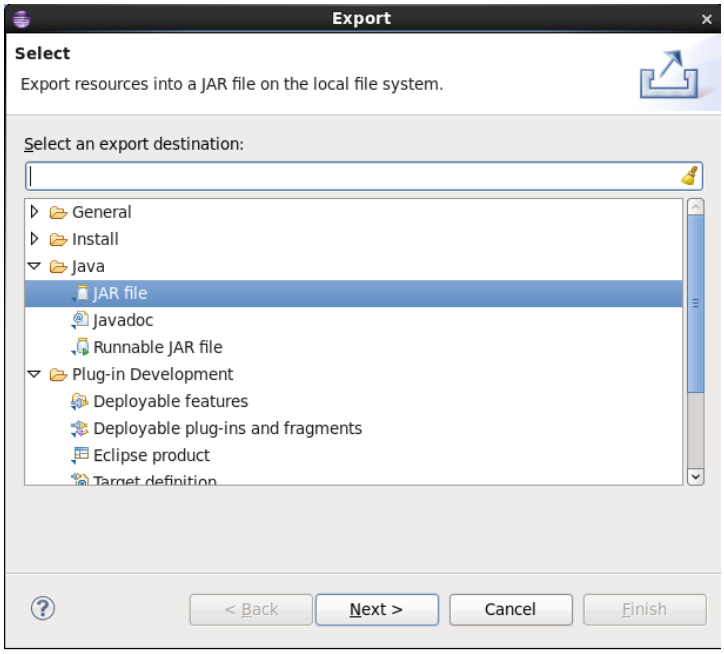
```
            //capturando o parametro qtd informado pelo usuario
            Configuration conf = context.getConfiguration();
            int qtd = Integer.parseInt(conf.get("qtd"));

            topN.put(Integer.parseInt(hashtags[1].toLowerCase()),
              new Text(value));

            if (topN.size() > qtd) {
              topN.remove(topN.firstKey());
            }
          }

          for (Text word : topN.descendingMap().values()) {
            context.write(NullWritable.get(), word);
          }
    }
}
```

10. Repita o mesmo procedimento da Etapa 5, para criar a classe TopNHashTagDriver, inserindo o código a seguir:

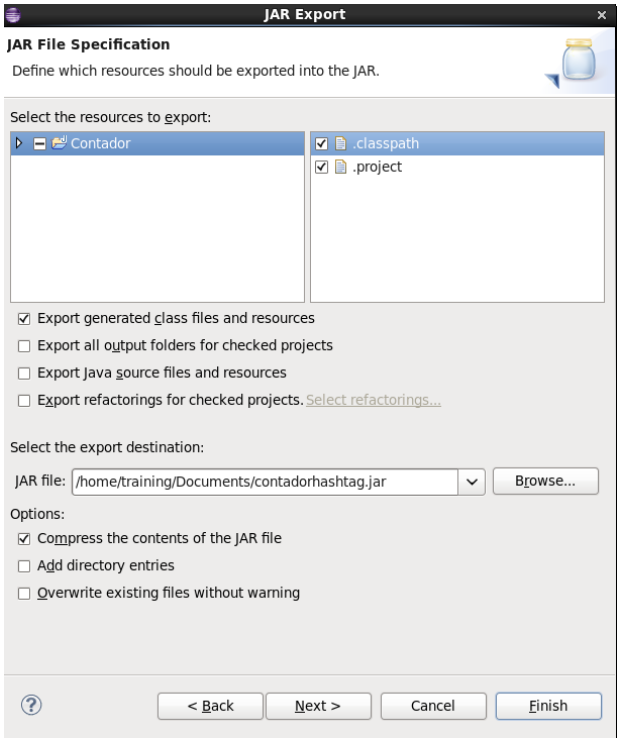
```
import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.Job;


public class TopNHashTagDriver {

    public static void main(String[] args) throws IOException, ClassNotFoundException,
InterruptedException{

        Configuration conf = new Configuration();

      //recebendo o parametro qtd do usuario
      conf.set("qtd", args[2]);

      Job job = Job.getInstance(conf);
      job.setJarByClass(TopNHashTagDriver.class);
      job.setMapperClass(TopNHashTagMapper.class);
      job.setReducerClass(TopNHashTagReducer.class);
      job.setOutputKeyClass(NullWritable.class);
      job.setOutputValueClass(Text.class);
      job.setNumReduceTasks(1);

      FileInputFormat.addInputPath(job, new Path(args[0]));
      FileOutputFormat.setOutputPath(job, new Path(args[1]));
```

```
            System.exit(job.waitForCompletion(true) ? 0 : 1);
        }

    }
```

11. Tendo implementado todas as classes, gere um arquivo JAR clicando com o botão direito do mouse no nome do projeto, selecionando a opção export. Será aberto a seguinte janela:



12. Selecione a opção JAR file e pressione o botão “Next”. Deverá aparecer a seguinte janela:



13. No campo JAR file, indique o seguinte caminho: /home/training/Documents/contadorhashtag.jar e pressione o botão Finish. Essa ação criará o jar do job implementado.

14. Antes de executar a aplicação desenvolvida, é preciso enviar a base de dados para o HDFS e executar a aplicação de contagem de hashtag, uma vez que utilizaremos o resultado dessa aplicação como entrada da aplicação desenvolvida. Primeiramente iremos criar um diretório conforme o comando a seguir:

    ```
    $ hadoop fs -mkdir base_hashtag
    ```

15. Após criado o diretório, realize o envio dos dados da base local para o HDFS, executando o comando a seguir:

    ```
    $ hadoop fs -put ~/bases/base_tw.txt  base_hashtag
    ```

16. Agora já é possível executar a aplicação de contagem de hashtags. Faça a chamada do job por meio do seguinte comando:

    ```
    $ hadoop jar ~/Documents/contadorhashtag.jar ContadorHashTagDriver base_hashtag saida/contador
    ```

17. Verifique se o job foi executado corretamente listando os arquivos de saída:

    ```
    $ hadoop fs -ls  base_hashtag saida/contador
    ```

18. Deverá aparecer um resultado similar ao exemplo a seguir:

    ```
    Found 3 items
    -rw-rw-rw-   1 training supergroup          0 2016-06-26 22:17 saida/contador/_SUCCESS
    drwxrwxrwx   - training supergroup          0 2016-06-26 22:16 saida/contador/_logs
    -rw-rw-rw-   1 training supergroup      12780 2016-06-26 22:17 saida/contador/part-r-00000
    ```

19. Caso o job tenha sido executado com sucesso, faça a chamada da aplicação de contagem de top n hashtags por meio do comando a seguir (perceba que essa aplicação requer um parâmetro indicando o valor n de hashtags que deverá ser apresentado):

    ```
    $ hadoop  jar ~/Documents/contadorhashtag.jar TopNHashTagDriver   saida/contador  saida/topn   5
    ```

20. No comando anterior indicamos ao job que buscasse as top 5 hashtags mais encontradas na base. Verifique se o job foi executado com sucesso por meio do comando a seguir:

    ```
    $ hadoop fs -ls  saida/topn
    ```

21. O comando anterior deverá apresentar um resultado similar ao exemplo a seguir:

    ```
    Found 3 items
    -rw-rw-rw-   1 training supergroup          0 2016-06-26 22:25 saida/topn/_SUCCESS
    drwxrwxrwx   - training supergroup          0 2016-06-26 22:24 saida/topn/_logs
    -rw-rw-rw-   1 training supergroup         77 2016-06-26 22:25 saida/topn/part-r-00000
    ```

22. Por fim, visualize o conteúdo do arquivo para verificar se o resultado foi apresentado corretamente, por meio do seguinte comando:

    ```
    $ hadoop fs -cat  saida/topn/part-r-00000
    ```

23. O comando anterior deverá apresentar um resultado similar ao exemplo a seguir:

```
#DataScience 1454
#BigData     1038
#datascience 834
#Analytics   488
#dataviz     417
```

Parabéns! Você concluiu as etapas de implementação e execução de uma aplicação para realizar operações no HDFS utilizando a biblioteca Java do Hadoop.