CPSC 4330 Big Data Analytics

In-class Exercise 7 - Create an Inverted Index

In this exercise, you will write a MapReduce job that produces an inverted index.

For this exercise you will use the input *invertedIndexInput.tgz* posted on Canvas. When decompressed, this archive contains a directory of files; each is a Shakespeare play formatted as follows:

```
0	HAMLET
1
2
3	DRAMATIS  PERSONAE
4
5
6	CLAUDIUS    king of Denmark. (KING CLAUDIUS: )
7
8	HAMLET      son to the late, and nephew to the present king.
9
10	POLONIUS    lord chamberlain.  (LORD  POLONIUS: )

…
```

Each line contains:

Line number

Separator: a tab character

Value: the line of text

This format can be read directly using the *KeyValueTextInputFormat* class provided in the Hadoop API. This input format presents each line as one record to your Mapper, with the part before the tab character as the key, and the part after the tab as the value.

Given a body of text in this form, your program should produce an index of all the words in the text. For each word, the index should have a list of all the locations where the word appears. For example, for the word 'honeysuckle' your output should look like this (e.g. in the following example, 2kinghenryiv is filename, and 1038 is line number)

```
honeysuckle	2kinghenryiv@1038, midsummernightsdream@2175, …
```

The index should contain such an entry for every word in the text.

Step 1: Prepare the Input Data

1. Copy the file *invertedIndexInput.tgz* to Docker.

2. Extract the invertedIndexInput directory and upload to HDFS.

```
# tar  zxvf   invertedIndexInput.tgz

# hadoop   fs  -put  invertedIndexInput   /exercise/invertedIndexInput
```

Step 2: Start Eclipse

1. Download the zip file "ex7.zip" from Canvas and unzip it. There are three files after unzipping:

   - IndexMapper.java (Mapper)
   - IndexReducer.java (Reducer)
   - InvertedIndex.java (Driver)

2. Review the instructions in step 2 of exercise 3 to create a new java project "invertedindex" and a package "stubs" for this exercise, and then copy all the java files from the previous step to the eclipse workspace on your computer for the project "invertedindex" under the "stubs" folder

(e.g. /Users/linli/eclipse-workspace/invertedindex/src/stubs).

Configure the Build Path of the project as how we did in step 2 of exercise 3.

Step 3: Write the Program in Java

1. IndexMapper.java (mapper)

Remember that for this program you use a special input format *KeyValueTextInputFormat* to suit the form of your data, so the input key and value types of Mapper are Text.

Note that the exercise requires you to retrieve the file name – since that is the name of the play. The Context object can be used to retrieve the name of the file like this:

```
FileSplit    fileSplit   =   (FileSplit) context.getInputSplit();

Path    path  =  fileSplit.getPath();

String   filename  =  path.getName();
```

You need to add the following two lines at the beginning of your Mapper code to import the classes FileSplit and Path.

```
import org.apache.hadoop.mapreduce.lib.input.FileSplit;

import org.apache.hadoop.fs.Path;
```

2. IndexReducer.java (reducer)

The reduce function needs to get each location in the list of all locations and add a separator (“,”) in between.

3. InvertedIndex.java (driver)

Remember that for this program you use *KeyValueTextInputFormat* to read data, so your driver class will need to include a line like:

```
job.setInputFormatClass(KeyValueTextInputFormat.class);
```

You also need to add the following line to import the *KeyValueTextInputFormat* class.

```
import org.apache.hadoop.mapreduce.lib.input.KeyValueTextInputFormat;
```

Step 3: Compile and Test your Solution

Build and test your code against the *invertedIndexInput* data you loaded above.