**PROJECT 4: REPORT**

**[Maaz Syed (0003544185) & Venkat Sambandhan (0003544188)]**

HBase WordCount is a simple program which counts the number of occurrences of each word in a given text input dataset. Below is the logic behind our implementation:

### Mapper

A Mapper overrides the “map” function from the Class "org.apache.hadoop.hbase.mapreduce.TableMapper<Text, LongWritable>" which provides <key, value> pairs as the input. <key, value> of this map function is <rowkey, content>, where the key is the rowkey of an HBase record related to a specified URI, and the content is the stored text of that URI. The Map task outputs <word, frequency> for each word in the content of text.

We use the "getWordFreq" function to count the frequencies of words in content.

We use the hashmap to go through all the words.

HashMap<String, Long> wordFrequency= getWordFreq(content);

for (Map.Entry<String, Long> wordFrequencyEntry : wordFrequencyEntry.entrySet()) {

context.write(new Text(wordFrequencyEntry.getKey()), new LongWritable(wordFrequencyEntry.getValue()));

}

### Reducer

A Reducer collects the intermediate <key, value> output from multiple map tasks and assembles a single result. Here, the reducer function will sum up the occurrence of each word to pairs as <word, occurrence>, then write it back to an HBase table with put operations which contain the key-value pair information of each word.

long totalFreq = 0;

for(LongWritable frequencyValue: freqs){

totalFreq = frequencyValue +( frequencyValue.get();

}

Now we write this into the HBase table:

Put p = new Put(Bytes.toBytes(word.toString()));

put.add(Constants.CF\_FREQUENCIES.getBytes(), Constants.QUALIFIER\_COUNT.getBytes(), Bytes.toBytes(totalFreq));

context.write(null, p);

### Main program

The main function has been provided as standard initialization. The provided code is designed for using put operations in the reducer content.write() function.