Hala Mallak

1. **hadoop fs -mkdir my_input**

2. **hadoop fs -put input/* my_input/**
3.

```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
org.apache.hadoop.mapreduce.lib.input.FileSplit

public class WordCount {

  public static class TokenizerMapper extends Mapper<Object, Text, Text,
Text>{

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

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

  public void map(Object key, Text value, Context context) throws
IOException, InterruptedException {
    String docId = context.getInputSplit()).getPath().toString();
    StringTokenizer itr = new StringTokenizer(value.toString());
    while (itr.hasMoreTokens()) {
      word.set(itr.nextToken());
      context.write(word, new Text(docId));
    }
  }
 }

 public static class IntSumReducer extends Reducer<Text, Text, Text, Text>
{

  public void reduce(Text key, Iterable<Text> values, Context context)
throws IOException, InterruptedException {
    int frequency = 0;
    StringBuilder docIds = new StringBuilder();
    for (Text val : values) {
      docIds.append(val.toString()).append(",");
      frequency++;
    }
    docIds.deleteCharAt(docIds.length() - 1);
```

```
      context.write(key, new Text(docIds.toString() + "," + frequency));
    }
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);
    FileInputFormat.addInputPath(job, new Path("my_input"));
    FileOutputFormat.setOutputPath(job, new Path("output"));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

```
documents = LOAD ' output '  USING PigStorage(',') AS (word:chararray, docId:chararray, frequency:int);

documents_filtered = FILTER documents BY word == 'MapReduce';

documents_sorted = ORDER documents_filtered BY frequency DESC;

top_10 = LIMIT documents_sorted 10;

output = FOREACH top_10 GENERATE docId, frequency;

STORE output INTO 'top_10_output' USING PigStorage('\t');
```