Máster en Big Data

Tecnologías de Almacenamiento

5. Hands-On: Desarrollo MapReduce Avanzado

Presentado por:

Albert Ripoll y Jose David Angulo

Índice

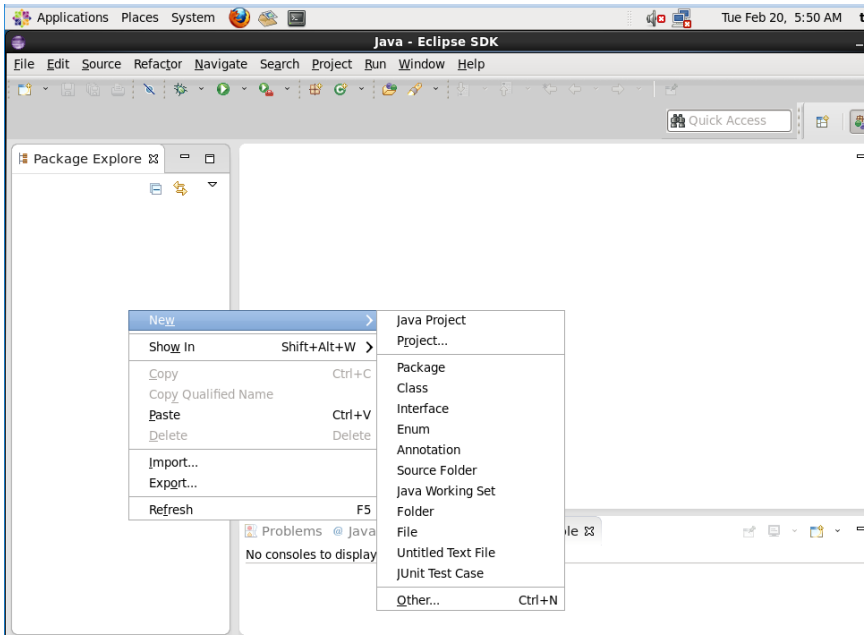
1. Introducción

El objetivo de este Hands-On es poner en práctica conceptos avanzados en el desarrollo de Jobs de MapReduce

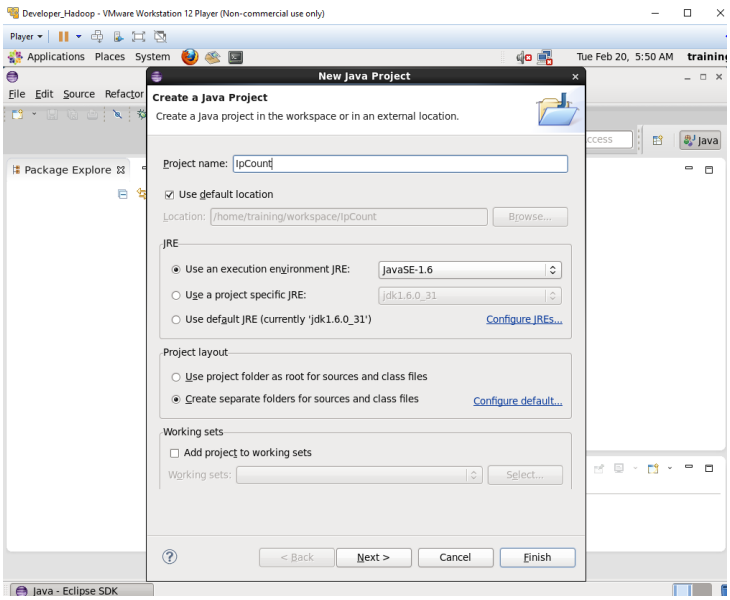
2. Entorno de desarrollo

Para realizar el desarrollo lo haremos mediante el IDE Eclipse de la máquina virtual importada en ejercicios anteriores.

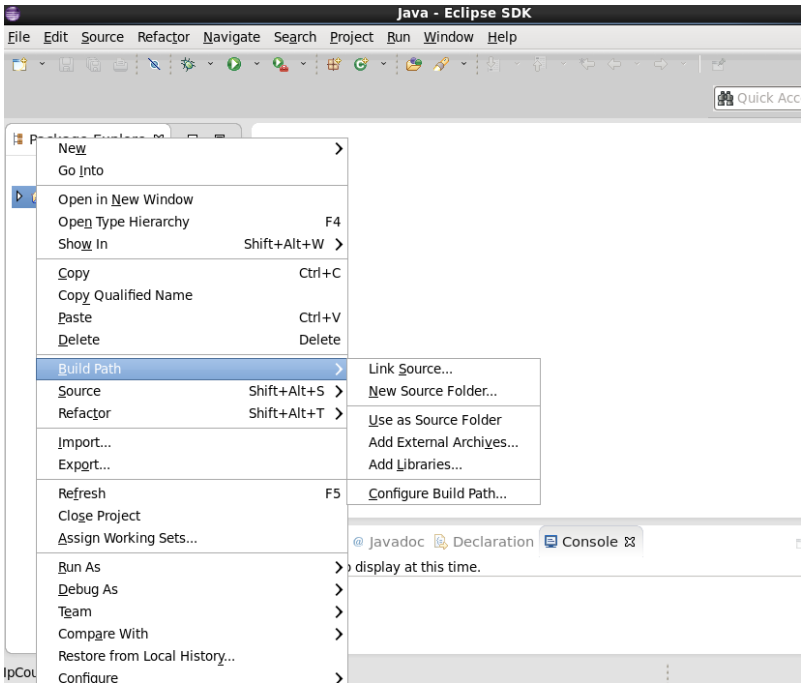
Para crear un nuevo proyecto, haremos click derecho sobre el package explorer New → Java Project



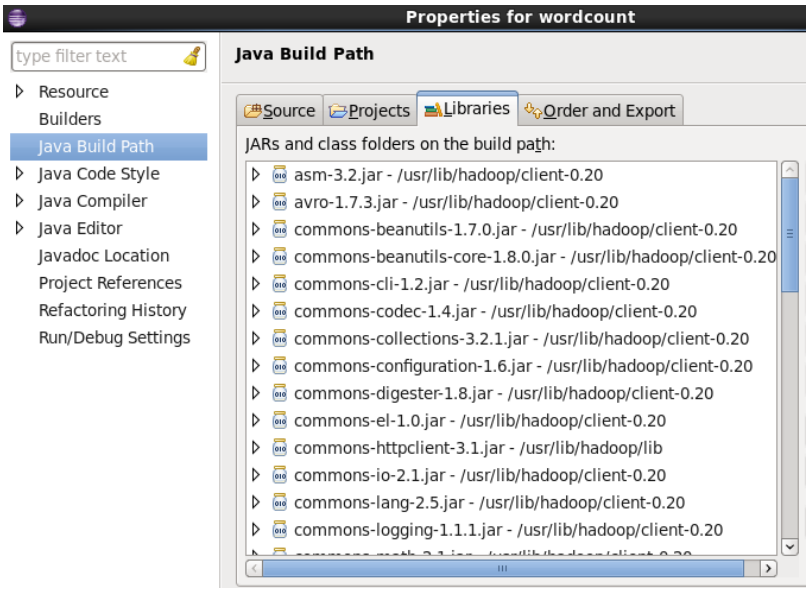
Introducimos el nombre del proyecto y click en Finish



Importamos manualmente las lilbrerías necesarias haciendo click derecho sobre el proyecto que acabamos de crear y seleccionamos Build Path → Configure Build Path



En la pestaña de libraries, seleccionamos Add Extertnal Jars e importamos todo el contenido de la carpeta /usr/lib/hadoop/client-0.20/



3. Tool Runner y parámetros

Desarrollar y ejecutar el siguiente MapReduce:

Aprovechando el ejercicio del Hands-On anterior (**AvarageWordLength)** realizar las siguientes modificaciones:

- La clase driver use ToolRunner

- Modificar el Mapper para referenciar una variable booleana llamada caseSensitive. Si esta variable es true, el mapper no diferenciara entre mayúsculas ni minúsculas, si es false, hará una conversión de todas las letras a minúscula.

Creación del IPCOUNT5:

```
package LAB5;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.FloatWritable;
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;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

//user/training/weblog/access_log


public class IPCOUNT5 extends Configured implements Tool {

        public static void main(String[] args) throws Exception {
```

```
		//int exitCode = ToolRunner.run(new Configuration(), new WordCount(), args);
		int exitCode = ToolRunner.run(new Configuration(), new IPCOUNT5(), args);
		System.exit(exitCode);
		}

	public int run(String[] args) throws Exception {
		if (args.length != 2) {
			System.out.printf (
					"Usage: %s [generic options] <input dir> <output dir>\n",
getClass().getSimpleName());
			return -1;
		}

		Job job = new Job (getConf());
		job.setJarByClass (IPCOUNT5.class);
		//job.setJobName ("Word Count");

		job.setJobName ("Ip Driver");

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

		job.setMapperClass (IPMAPER5.class) ;
		job.setReducerClass (IPREDUCER5.class) ;

		job.setMapOutputKeyClass (Text.class);
		job.setMapOutputValueClass (FloatWritable.class);

		job.setOutputKeyClass (Text.class) ;
		job.setOutputValueClass (FloatWritable.class);

		boolean success = job.waitForCompletion(true);
		return success ? 0 : 1;
	}
}
```

Creación del IPMAPER5:

```
package LAB5;


import java.io.IOException;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce .Mapper;



public class IPMAPER5 extends Mapper<LongWritable, Text, Text,
FloatWritable> {


	private boolean caseSensitive;// estamos obteniendo el valor de la variable que es caseSensitive
	protected void setup(Context context) throws IOException, InterruptedException {

		caseSensitive = context.getConfiguration().getBoolean("caseSensitive",true);
	}

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

```
            String line = value.toString();
            for (String word: line.split("\\W+")){
                        if (word.length() > 0){
                                    String letra;
                                    if (caseSensitive){
                                                letra = word.substring(0,1);
                                    } else {
                                                letra = word.substring(0,1).toLowerCase();
                                    }
                                    float longitud = word.length();
                                    context.write(new Text(letra), new FloatWritable(longitud));
            }
}
}
}
```

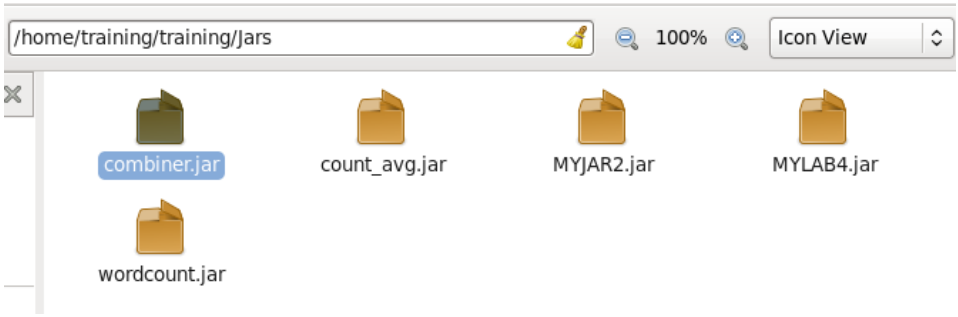
Creación del IPREDUCER5:

```
package LAB5;

import java.io.IOException;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce .Reducer;

public class IPREDUCER5 extends Reducer<Text, FloatWritable, Text, FloatWritable>
{
@Override
public void reduce (Text key, Iterable<FloatWritable> values, Context context)
throws IOException, InterruptedException {
float wordCount = 0;
float i=0;

for (FloatWritable value : values) {
            wordCount += value.get();
            i+=1;
            }
float promedio = wordCount / i;
context.write (key, new FloatWritable (promedio));
}
}
```

1) Para ejecutar primero creamos los JAR.export /training/jars/count_avg.jar

Para hacerlo: Click derecho en el IPCOUNT5 >Export>Jar>Buscamos directorio donde está > y ponemos el nombre del jar como JAR.export /training/jars/count_avg.jar

Esto nos quita tres pasos de ejecución respecto a los Hands On anteriores.



2) En la consola para ejecutar con el –DcaseSensitive=false:

```
[training@localhost ~]$ hadoop jar /home/training/training/Jars/count_avg.jar LAB5.IPCOUNT5 -DcaseSensitive=false /user/training/Shakespeare /home/training/OUTPUTS_LAB5/PUNTO3
24/04/08 15:38:43 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
24/04/08 15:38:43 INFO input.FileInputFormat: Total input paths to process : 4
24/04/08 15:38:43 INFO mapred.JobClient: Running job: job_202404081505_0002
24/04/08 15:38:44 INFO mapred.JobClient:  map 0% reduce 0%
24/04/08 15:38:51 INFO mapred.JobClient:  map 50% reduce 0%
24/04/08 15:38:56 INFO mapred.JobClient:  map 100% reduce 0%
24/04/08 15:38:58 INFO mapred.JobClient:  map 100% reduce 100%
24/04/08 15:38:58 INFO mapred.JobClient: Job complete: job_202404081505_0002
24/04/08 15:38:58 INFO mapred.JobClient: Counters: 32
24/04/08 15:38:58 INFO mapred.JobClient:   File System Counters
24/04/08 15:38:58 INFO mapred.JobClient:     FILE: Number of bytes read=15029594
24/04/08 15:38:58 INFO mapred.JobClient:     FILE: Number of bytes written=23710628
24/04/08 15:38:58 INFO mapred.JobClient:     FILE: Number of read operations=0
24/04/08 15:38:58 INFO mapred.JobClient:     FILE: Number of large read operations=0
24/04/08 15:38:58 INFO mapred.JobClient:     FILE: Number of write operations=0
24/04/08 15:38:58 INFO mapred.JobClient:     HDFS: Number of bytes read=5284706
24/04/08 15:38:58 INFO mapred.JobClient:     HDFS: Number of bytes written=366
24/04/08 15:38:58 INFO mapred.JobClient:     HDFS: Number of read operations=8
24/04/08 15:38:58 INFO mapred.JobClient:     HDFS: Number of large read operations=0
24/04/08 15:38:58 INFO mapred.JobClient:     HDFS: Number of write operations=1
24/04/08 15:38:58 INFO mapred.JobClient:   Job Counters
24/04/08 15:38:58 INFO mapred.JobClient:     Launched map tasks=4
24/04/08 15:38:58 INFO mapred.JobClient:     Launched reduce tasks=1
24/04/08 15:38:58 INFO mapred.JobClient:     Data-local map tasks=4
24/04/08 15:38:58 INFO mapred.JobClient:     Total time spent by all maps in occupied slots (ms)=20238
24/04/08 15:38:58 INFO mapred.JobClient:     Total time spent by all reduces in occupied slots (ms)=7334
24/04/08 15:38:58 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving slots (ms)=0
24/04/08 15:38:58 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving slots (ms)=0
24/04/08 15:38:58 INFO mapred.JobClient:   Map-Reduce Framework
24/04/08 15:38:58 INFO mapred.JobClient:     Map input records=173126
24/04/08 15:38:58 INFO mapred.JobClient:     Map output records=964453
24/04/08 15:38:58 INFO mapred.JobClient:     Map output bytes=5786718
24/04/08 15:38:58 INFO mapred.JobClient:     Input split bytes=475
24/04/08 15:38:58 INFO mapred.JobClient:     Combine input records=0
24/04/08 15:38:58 INFO mapred.JobClient:     Combine output records=0
24/04/08 15:38:58 INFO mapred.JobClient:     Reduce input groups=35
24/04/08 15:38:58 INFO mapred.JobClient:     Reduce shuffle bytes=7715648
24/04/08 15:38:58 INFO mapred.JobClient:     Reduce input records=964453
24/04/08 15:38:58 INFO mapred.JobClient:     Reduce output records=35
24/04/08 15:38:58 INFO mapred.JobClient:     Spilled Records=2843147
24/04/08 15:38:58 INFO mapred.JobClient:     CPU time spent (ms)=3640
24/04/08 15:38:58 INFO mapred.JobClient:     Physical memory (bytes) snapshot=883212288
24/04/08 15:38:58 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=3618459648
24/04/08 15:38:58 INFO mapred.JobClient:     Total committed heap usage (bytes)=557989888
[training@localhost ~]$
```

3) En la consola para leer con el false:

```
[training@localhost ~]$ hadoop fs -cat /home/training/OUTPUTS_LAB5/PUNTO3/part-r-00000
1       1.02
2       1.0588236
3       1.0
4       1.5
5       1.5
6       1.5
7       1.0
8       1.5
9       1.0
a       3.2758996
b       4.4367685
c       6.2040734
d       4.3062005
e       5.3072386
f       4.87379
g       5.163682
h       3.9661317
i       2.1290417
j       5.1489835
k       4.622563
l       4.4545455
m       3.9906976
n       3.7499645
o       2.8046207
p       6.209215
q       5.8527956
r       5.8549657
s       4.4860196
t       3.7721033
u       4.5886965
v       5.540628
w       4.3730965
x       3.1650486
y       3.517174
z       5.0533333
[training@localhost ~]$
```

4) En la consola para ejecutar con el –DcaseSensitive=true:

```
[training@localhost ~]$ hadoop jar /home/training/training/Jars/count_avg.jar LAB5.IPCOUNT5 -DcaseSensitive=true /user/training/Shakespeare /home/training/OUTPUTS_LAB5/PUNTO3true
24/04/08 15:56:26 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
24/04/08 15:56:26 INFO input.FileInputFormat: Total input paths to process : 4
24/04/08 15:56:27 INFO mapred.JobClient: Running job: job_202404081505_0003
24/04/08 15:56:28 INFO mapred.JobClient:  map 0% reduce 0%
24/04/08 15:56:34 INFO mapred.JobClient:  map 50% reduce 0%
24/04/08 15:56:39 INFO mapred.JobClient:  map 100% reduce 0%
24/04/08 15:56:42 INFO mapred.JobClient:  map 100% reduce 100%
24/04/08 15:56:42 INFO mapred.JobClient: Job complete: job_202404081505_0003
24/04/08 15:56:42 INFO mapred.JobClient: Counters: 32
24/04/08 15:56:42 INFO mapred.JobClient:   File System Counters
24/04/08 15:56:42 INFO mapred.JobClient:     FILE: Number of bytes read=15029594
24/04/08 15:56:42 INFO mapred.JobClient:     FILE: Number of bytes written=23710643
24/04/08 15:56:42 INFO mapred.JobClient:     FILE: Number of read operations=0
24/04/08 15:56:42 INFO mapred.JobClient:     FILE: Number of large read operations=0
24/04/08 15:56:42 INFO mapred.JobClient:     FILE: Number of write operations=0
24/04/08 15:56:42 INFO mapred.JobClient:     HDFS: Number of bytes read=5284706
24/04/08 15:56:42 INFO mapred.JobClient:     HDFS: Number of bytes written=647
24/04/08 15:56:42 INFO mapred.JobClient:     HDFS: Number of read operations=8
24/04/08 15:56:42 INFO mapred.JobClient:     HDFS: Number of large read operations=0
24/04/08 15:56:42 INFO mapred.JobClient:     HDFS: Number of write operations=1
24/04/08 15:56:42 INFO mapred.JobClient:   Job Counters
24/04/08 15:56:42 INFO mapred.JobClient:     Launched map tasks=4
24/04/08 15:56:42 INFO mapred.JobClient:     Launched reduce tasks=1
24/04/08 15:56:42 INFO mapred.JobClient:     Data-local map tasks=4
24/04/08 15:56:42 INFO mapred.JobClient:     Total time spent by all maps in occupied slots (ms)=19368
24/04/08 15:56:42 INFO mapred.JobClient:     Total time spent by all reduces in occupied slots (ms)=7283
24/04/08 15:56:42 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving slots (ms)=0
24/04/08 15:56:42 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving slots (ms)=0
24/04/08 15:56:42 INFO mapred.JobClient:   Map-Reduce Framework
24/04/08 15:56:42 INFO mapred.JobClient:     Map input records=173126
24/04/08 15:56:42 INFO mapred.JobClient:     Map output records=964453
24/04/08 15:56:42 INFO mapred.JobClient:     Map output bytes=5786718
24/04/08 15:56:42 INFO mapred.JobClient:     Input split bytes=475
24/04/08 15:56:42 INFO mapred.JobClient:     Combine input records=0
24/04/08 15:56:42 INFO mapred.JobClient:     Combine output records=0
24/04/08 15:56:42 INFO mapred.JobClient:     Reduce input groups=60
24/04/08 15:56:42 INFO mapred.JobClient:     Reduce shuffle bytes=7715648
24/04/08 15:56:42 INFO mapred.JobClient:     Reduce input records=964453
24/04/08 15:56:42 INFO mapred.JobClient:     Reduce output records=60
24/04/08 15:56:42 INFO mapred.JobClient:     Spilled Records=2843147
24/04/08 15:56:42 INFO mapred.JobClient:     CPU time spent (ms)=3570
24/04/08 15:56:42 INFO mapred.JobClient:     Physical memory (bytes) snapshot=883695616
24/04/08 15:56:42 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=3618459648
24/04/08 15:56:42 INFO mapred.JobClient:     Total committed heap usage (bytes)=557989888
```

5) En la consola para leer con el true:

```
[training@localhost ~]$ hadoop fs -cat /home/training/OUTPUTS_LAB5/PUNTO3true/part-r-00000
1	1.02
2	1.0588236
3	1.0
4	1.5
5	1.5
6	1.5
7	1.0
8	1.5
9	1.0
A	3.8913946
B	5.1393027
C	6.6296945
D	5.2018347
E	5.5142636
F	5.2555285
G	5.809792
H	4.4210725
I	1.4526861
J	4.9840083
K	4.657107
L	5.1158814
M	5.4464655
N	3.9848387
O	2.8794768
P	6.5057406
Q	5.5216427
R	5.929275
S	5.293126
T	3.9591436
U	5.325
V	5.1945376
W	4.464014
X	3.1650486
Y	3.4432244
Z	6.1
a	3.0776556
b	4.2453966
c	6.0414414
d	4.1463876
e	5.182466
f	4.778552
g	4.9389167
h	3.877788
i	2.7292957
j	5.329446
k	4.607203
l	4.2727776
m	3.718217
n	3.7032013
```

4. Combiner

Desarrollar y ejecutar el siguiente MapReduce:

Añadir un combiner al proyecto **IpCount** realizado en el Hands-On anterior

Copiamos el codigo del IPMAPER y IPREDUCER del Hands-On de la práctica 3 en un nuevo IPCOMBMAPER5 y IPCOMBREDUCER5. También copiamos el codigo del IPDRIVER de la práctica 3 en un nuveo IPCOMBINER5. A esa classe le añadimos la siguiente linea de código *Job.setCombinerClass(IPCOMBREDUCER5.class);*
De esa forma tenemos:

• IPCOMBINER5:

```
package LAB5COMB;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
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;
//user/training/weblog/access_log

public class IPCOMBINER5 {
public static void main(String[] args) throws Exception {
if (args.length != 2) {
System. out.printf ("Usage: WordCount <input dir> <output dir>n");
System.exit(-1);
}
Job job = new Job();
job.setJarByClass (IPCOMBINER5.class);
job.setJobName ("Ip Driver");

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

job.setMapperClass (IPCOMBMAPER5.class) ;
job.setReducerClass (IPCOMBREDUCER5.class) ;
job.setCombinerClass(IPCOMBREDUCER5.class);

job.setMapOutputKeyClass (Text.class);
job.setMapOutputValueClass (IntWritable.class);

job.setOutputKeyClass (Text.class) ;
job.setOutputValueClass (IntWritable.class);

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

• IPCOMBMAPER5:

```
package LAB5COMB;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce .Mapper;

public class IPCOMBMAPER5 extends Mapper<LongWritable, Text, Text,
IntWritable> {
@Override
public void map(LongWritable key, Text value, Context context)
throws IOException, InterruptedException {

String line = value.toString();

String[] parts = line.split(" - - ");
String word = parts[0];

context.write(new Text (word), new IntWritable(1));
}
}
```

• IPCOMBREDUCER5:

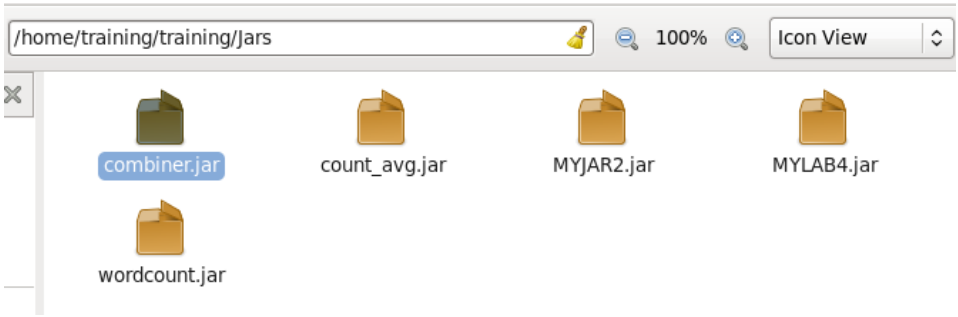
```
package LAB5COMB;
```

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce .Reducer;

public class IPCOMBREDUCER5 extends Reducer<Text, IntWritable, Text, IntWritable>
{
@Override
public void reduce (Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
int wordCount = 0;

for (IntWritable value : values) {
wordCount += value.get();
}
context.write (key, new IntWritable (wordCount));
}
}
```

1) Crear los JAR.export /training/jars/count_avg.jar

   Para hacerlo: Click derecho el el IPCOMBINER>Export>Jar>Buscamos directorio donde está > y ponemos el nombre del jar como JAR.export /training/jars/combiner.jar

Esto nos quita tres pasos de ejecución respecto a los Hands On anteriores.



2) En la consola para ejecutar:

```
[training@localhost ~]$ hadoop jar /home/training/training/Jars/combiner.jar LAB5COMB.IPCOMBINER5 /user/training/Shakespeare /home/training/OUTPUTS_LAB5/COMBINER5
24/04/08 14:46:16 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
24/04/08 14:46:16 INFO input.FileInputFormat: Total input paths to process : 4
24/04/08 14:46:16 INFO mapred.JobClient: Running job: job_202404081430_0002
24/04/08 14:46:17 INFO mapred.JobClient:  map 0% reduce 0%
24/04/08 14:46:24 INFO mapred.JobClient:  map 50% reduce 0%
24/04/08 14:46:28 INFO mapred.JobClient:  map 75% reduce 0%
24/04/08 14:46:29 INFO mapred.JobClient:  map 100% reduce 0%
24/04/08 14:46:31 INFO mapred.JobClient:  map 100% reduce 100%
24/04/08 14:46:32 INFO mapred.JobClient: Job complete: job_202404081430_0002
24/04/08 14:46:32 INFO mapred.JobClient: Counters: 32
24/04/08 14:46:32 INFO mapred.JobClient:   File System Counters
24/04/08 14:46:32 INFO mapred.JobClient:     FILE: Number of bytes read=5897757
24/04/08 14:46:32 INFO mapred.JobClient:     FILE: Number of bytes written=12758518
24/04/08 14:46:32 INFO mapred.JobClient:     FILE: Number of read operations=0
24/04/08 14:46:32 INFO mapred.JobClient:     FILE: Number of large read operations=0
24/04/08 14:46:32 INFO mapred.JobClient:     FILE: Number of write operations=0
24/04/08 14:46:32 INFO mapred.JobClient:     HDFS: Number of bytes read=5284706
24/04/08 14:46:32 INFO mapred.JobClient:     HDFS: Number of bytes written=5410988
24/04/08 14:46:32 INFO mapred.JobClient:     HDFS: Number of read operations=8
24/04/08 14:46:32 INFO mapred.JobClient:     HDFS: Number of large read operations=0
24/04/08 14:46:32 INFO mapred.JobClient:     HDFS: Number of write operations=1
24/04/08 14:46:32 INFO mapred.JobClient:   Job Counters
24/04/08 14:46:32 INFO mapred.JobClient:     Launched map tasks=4
24/04/08 14:46:32 INFO mapred.JobClient:     Launched reduce tasks=1
24/04/08 14:46:32 INFO mapred.JobClient:     Data-local map tasks=4
24/04/08 14:46:32 INFO mapred.JobClient:     Total time spent by all maps in occupied slots (ms)=19724
24/04/08 14:46:32 INFO mapred.JobClient:     Total time spent by all reduces in occupied slots (ms)=7412
24/04/08 14:46:32 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving slots (ms)=0
24/04/08 14:46:32 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving slots (ms)=0
24/04/08 14:46:32 INFO mapred.JobClient:   Map-Reduce Framework
24/04/08 14:46:32 INFO mapred.JobClient:     Map input records=173126
24/04/08 14:46:32 INFO mapred.JobClient:     Map output records=173126
24/04/08 14:46:32 INFO mapred.JobClient:     Map output bytes=5976735
24/04/08 14:46:32 INFO mapred.JobClient:     Input split bytes=475
24/04/08 14:46:32 INFO mapred.JobClient:     Combine input records=173126
24/04/08 14:46:32 INFO mapred.JobClient:     Combine output records=120469
24/04/08 14:46:32 INFO mapred.JobClient:     Reduce input groups=120185
24/04/08 14:46:32 INFO mapred.JobClient:     Reduce shuffle bytes=5897775
24/04/08 14:46:32 INFO mapred.JobClient:     Reduce input records=120469
24/04/08 14:46:32 INFO mapred.JobClient:     Reduce output records=120185
24/04/08 14:46:32 INFO mapred.JobClient:     Spilled Records=240938
24/04/08 14:46:32 INFO mapred.JobClient:     CPU time spent (ms)=3360
24/04/08 14:46:32 INFO mapred.JobClient:     Physical memory (bytes) snapshot=872341504
24/04/08 14:46:32 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=3615301632
24/04/08 14:46:32 INFO mapred.JobClient:     Total committed heap usage (bytes)=557989888
[training@localhost ~]$
```

3) En la consola para leer/-cat:

```
[training@localhost ~]$ hadoop fs -cat /home/training/OUTPUTS_LAB5/COMBINER5/part-r-00000 |tail -n 50
government changed from kings to consuls.       1
hastily dispatcheth messengers, one to Rome for her father,     1
have done is yours; what I have to do is yours; being part in   1
he makes!'      1
heart's content; which I wish may always answer your own wish   1
his passions for the present, departed with the rest back to the        1
his wife, though it were late in the night, spinning amongst her        1
hounds. 1
hours, till I have honoured you with some graver labour. But if 1
in several disports. Whereupon the noblemen yielded Collatinus  1
intending, by their secret and sudden arrival, to make trial of 1
into her chamber, violently ravished her, and early in the      1
maids: the other ladies were all found dancing and revelling, or        1
morning speedeth away. Lucrece, in this lamentable plight,      1
myself highly praised, and vow to take advantage of all idle    1
not requiring or staying for the people's suffrages, had        1
of BUCKINGHAM   [To KING RICHARD III]   1
of King Henry VI        [To KING RICHARD III]   1
of Prince Edward        [To KING RICHARD III]   1
of her sorrow. She, first taking an oath of them for her        1
of my untutored lines, makes it assured of acceptance. What I   1
of young Princes        [To KING RICHARD III]   1
people were so moved, that with one consent and a general       1
possessed himself of the kingdom, went, accompanied with his sons       1
raining?        1
red;    1
revenge, revealed the actor, and whole manner of his dealing, and       1
show greater; meantime, as it is, it is bound to your lordship, 1
smelling.       1
sorry it had so noble a god-father, and never after ear so      1
store,  1
supper every one commended the virtues of his own wife: among   1
surnamed HOTSPUR        his son. (HENRY PERCY:) 1
that which every one had before avouched, only Collatinus finds 1
the Eighth      (KING HENRY VIII:)      1
the Sixth       (KING HENRY VI:)        1
the first heir of my invention prove deformed, I shall be       1
the people with the doer and manner of the vile deed, with a    1
the principal men of the army meeting one evening at the tent of        1
the victory, and his wife the fame. At that time Sextus 1
they all vowed to root out the whole hated family of the        1
this pamphlet, without beginning, is but a superfluous moiety.  1
thunder,        1
to King Richard (QUEEN:)        1
to whom I wish long life, still lengthened with all happiness.  1
unpolished lines to your lordship, nor how the world will       1
was, according to his estate, royally entertained and lodged by 1
whom Collatinus extolled the incomparable chastity of his wife  1
```

5. Partitioner

Desarrollar y ejecutar el siguiente MapReduce:

Aprovechando el proyecto original **IpCount** realizar los cambios pertinentes para escribir un Job con múltiples reducers e implementar un partitioner que redirija la salida según el mes del año hacia un reducer concreto.

Es decir, en total habrán 12 reducers (uno para cada mes del año) y el partitioner será el encargado de redirigir esa clave/valor hacia el reducer correcto.

La salida final consistirá en 12 ficheros, uno para cada mes del año, y contendrán el número de veces que se ha repetido la ip en ese mes del año.

Solución:

Input: `96.7.4.14 - - [24/Apr/2011:04:20:11 -0400] "GET /cat.jpg HTTP/1.1" 200 12433`

Output key: `96.7.4.14`

Output value: `Apr`

El código base original para el partitioner, el partitioner maper y el partitioner reducer es el siguiente. Hace falta modificarlo para hacer las 12 particiones que pide el enunciado.

• IPPARTITIONER5:

```
package LAB5PART;


import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
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;
//user/training/weblog/access_log

public class IPPARTITIONER5 {
public static void main(String[] args) throws Exception {
if (args.length != 2) {
System. out.printf ("Usage: WordCount <input dir> <output dir>n");
System.exit(-1);
}
Job job = new Job();
job.setJarByClass (IPPARTITIONER5.class);
job.setJobName ("Ip Driver");

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

job.setMapperClass (IPPARTMAPER5.class) ;
job.setReducerClass (IPPARTREDUCER5.class) ;

job.setMapOutputKeyClass (Text.class);
job.setMapOutputValueClass (IntWritable.class);

job.setOutputKeyClass (Text.class) ;
job.setOutputValueClass (IntWritable.class);

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

• IPPARTMAPER5:

```
package LAB5PART;


import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce .Mapper;

public class IPPARTMAPER5 extends Mapper<LongWritable, Text, Text,
IntWritable> {
@Override
public void map(LongWritable key, Text value, Context context)
throws IOException, InterruptedException {

String line = value.toString();

String[] parts = line.split(" - - ");
String word = parts[0];

context.write(new Text (word), new IntWritable(1));
}
}
```

• IPPARTREDUCER5:

```
package LAB5PART;


import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce .Reducer;

public class IPPARTREDUCER5 extends Reducer<Text, IntWritable, Text, IntWritable>
{
@Override
public void reduce (Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
int wordCount = 0;

for (IntWritable value : values) {
wordCount += value.get();
}
context.write (key, new IntWritable (wordCount));
}
}
```

La modificación del código anterior para hacer las 12 particiones es:

• MONTHPARTITIONERTEST

```
package solution;

import static org.junit.Assert.assertEquals;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.io.Text;
import org.junit.Test;

public class MonthPartitionTest {

	static String[] months = {"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"};
	MonthPartitioner<Text,Text> mpart;

	@Test
	public void testMonthPartition() {

		mpart=new MonthPartitioner<Text, Text>();
		mpart.setConf(new Configuration());
		int result;
		for (int i = 0; i < months.length; i++) {
			result = mpart.getPartition(new Text("foo"), new Text(months[i]), 12);
			assertEquals(result,i);
		}

	}

}
```

• MONTHPARTITIONER

```
package solution;

import java.util.HashMap;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.conf.Configurable;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Partitioner;

public class MonthPartitioner<K2, V2> extends Partitioner<Text, Text> implements
   Configurable {

 private Configuration configuration;
 HashMap<String, Integer> months = new HashMap<String, Integer>();

 /**
  * Set up the months hash map in the setConf method.
  */
 @Override
 public void setConf(Configuration configuration) {
  this.configuration = configuration;
  months.put("Jan", 0);
  months.put("Feb", 1);
  months.put("Mar", 2);
  months.put("Apr", 3);
  months.put("May", 4);
  months.put("Jun", 5);
  months.put("Jul", 6);
  months.put("Aug", 7);
  months.put("Sep", 8);
  months.put("Oct", 9);
  months.put("Nov", 10);
  months.put("Dec", 11);}

 /**
  * Implement the getConf method for the Configurable interface.
  */
 @Override
 public Configuration getConf() {
  return configuration;  }

 /**
  * You must implement the getPartition method for a partitioner class.
  * This method receives the three-letter abbreviation for the month
  * as its value. (It is the output value from the mapper.)
  * It should return an integer representation of the month.
  * Note that January is represented as 0 rather than 1.
  *
  * For this partitioner to work, the job configuration must have been
  * set so that there are exactly 12 reducers.
  */
 public int getPartition(Text key, Text value, int numReduceTasks) {
  return (int) (months.get(value.toString()));
 }
}
```

• LOGMONTHMAPPER

```
package solution;

import java.io.IOException;
import java.util.Arrays;
import java.util.List;

import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class LogMonthMapper extends Mapper<LongWritable, Text, Text, Text> {

        public static List<String> months =
Arrays.asList("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");

 /**
  * Example input line:
  * 96.7.4.14 - - [24/Apr/2011:04:20:11 -0400] "GET /cat.jpg HTTP/1.1" 200 12433
  *
  */
 @Override
 public void map(LongWritable key, Text value, Context context)
    throws IOException, InterruptedException {

   /*
    * Split the input line into space-delimited fields.
    */
   String[] fields = value.toString().split(" ");

   if (fields.length > 3) {

     /*
      * Save the first field in the line as the IP address.
      */
     String ip = fields[0];

     /*
      * The fourth field contains [dd/Mmm/yyyy:hh:mm:ss].
      * Split the fourth field into "/" delimited fields.
      * The second of these contains the month.
      */
     String[] dtFields = fields[3].split("/");
     if (dtFields.length > 1) {
       String theMonth = dtFields[1];

       /* check if it's a valid month, if so, write it out */
       if (months.contains(theMonth))
            context.write(new Text(ip), new Text(theMonth));
     }
   }
 }
}
```

• PROCESSLOG

```
package solution;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
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 ProcessLogs {

 public static void main(String[] args) throws Exception {

  if (args.length != 2) {
   System.out.printf("Usage: ProcessLogs <input dir> <output dir>\n");
   System.exit(-1);
  }

  Job job = new Job();
  job.setJarByClass(ProcessLogs.class);
  job.setJobName("Process Logs");

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

  job.setMapperClass(LogMonthMapper.class);
  job.setReducerClass(CountReducer.class);

  job.setMapOutputKeyClass(Text.class);
  job.setMapOutputValueClass(Text.class);

  job.setOutputKeyClass(Text.class);
  job.setOutputValueClass(IntWritable.class);

  /*
   * Set up the partitioner. Specify 12 reducers - one for each
   * month of the year. The partitioner class must have a
   * getPartition method that returns a number between 0 and 11.
   * This number will be used to assign the intermediate output
   * to one of the reducers.
   */
  job.setNumReduceTasks(12);

  /*
   * Specify the partitioner class.
   */
  job.setPartitionerClass(MonthPartitioner.class);

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

• COUNTREDUCER

```
package solution;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/* Counts the number of values associated with a key */

public class CountReducer extends Reducer<Text, Text, Text, IntWritable> {

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

		/*
		 * Iterate over the values iterable and count the number
		 * of values in it. Emit the key (unchanged) and an IntWritable
		 * containing the number of values.
		 */

		int count = 0;

		/*
		 * Use for loop to count items in the iterator.
		 */

		/* Ignore warnings that we
		 * don't use the value -- in this case, we only need to count the
		 * values, not use them.
		 */
		for (@SuppressWarnings("unused")
		Text value : values) {

			/*
			 * for each item in the list, increment the count
			 */
			count++;
		}

		context.write(key, new IntWritable(count));
	}
}
```