

Hadoop: MapReduce

*Máster en Business Intelligence e
Innovación Tecnológica*

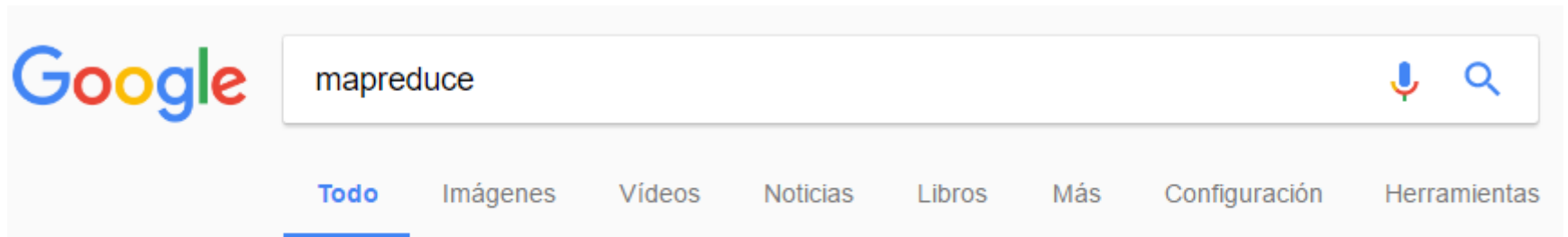
Índice de contenidos

1. Introducción.
2. Ejemplos prácticos

Introducción

- Fue creado por Google para utilizarlo en sus motores de búsqueda
- Partiendo del trabajo de Google, Yahoo y posteriormente Apache integran ese modelo de desarrollo en su framework Hadoop.
- MapReduce realmente es un paradigma de programación, orientado a computación distribuida.
- Sus fundamentos son distribuir la carga de trabajo en pequeñas subtarefas elementales, de forma que se puedan ejecutar por separado en sistemas cloud con múltiples nodos.

Introducción



Aproximadamente 4.580.000 resultados (0,56 segundos)

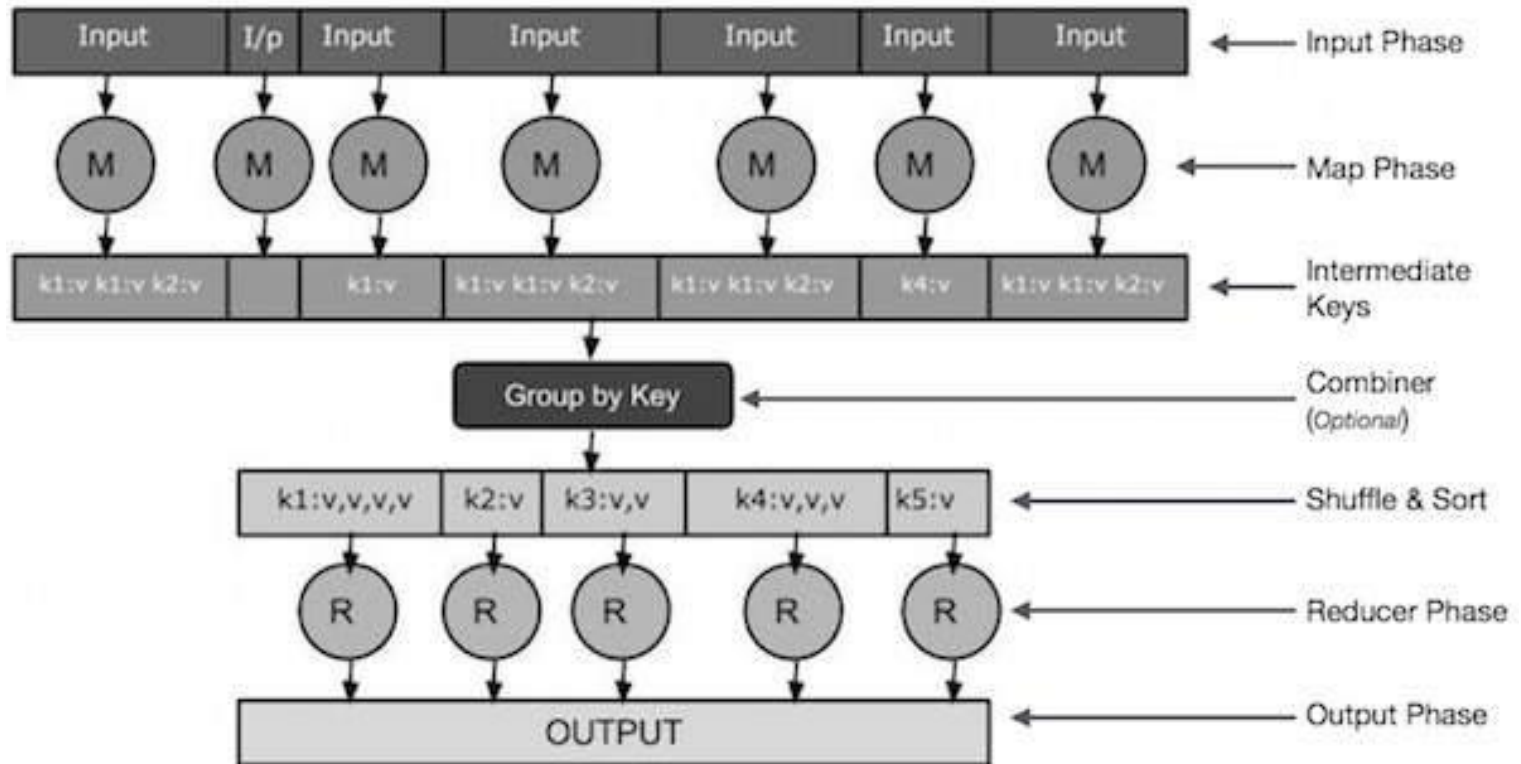
Introducción

- Fue creado por Google para utilizarlo en sus motores de búsqueda
- Partiendo del trabajo de Google, Yahoo y posteriormente Apache integran ese modelo de desarrollo en su framework Hadoop.
- MapReduce realmente es un paradigma de programación, orientado a computación distribuida.
- Sus fundamentos son distribuir la carga de trabajo en pequeñas subtareass elementales, de forma que se puedan ejecutar por separado en sistemas cloud con múltiples nodos.

Resumen:

- Las tareas de Map toman un conjunto de datos y lo transforman en un conjunto de elementos individuales con formato de tuplas clave-valor.
- Las tareas Reduce toma la salida de la tarea Map como entrada y combinan esos pares de clave-valor en los conjuntos de valores resultantes.

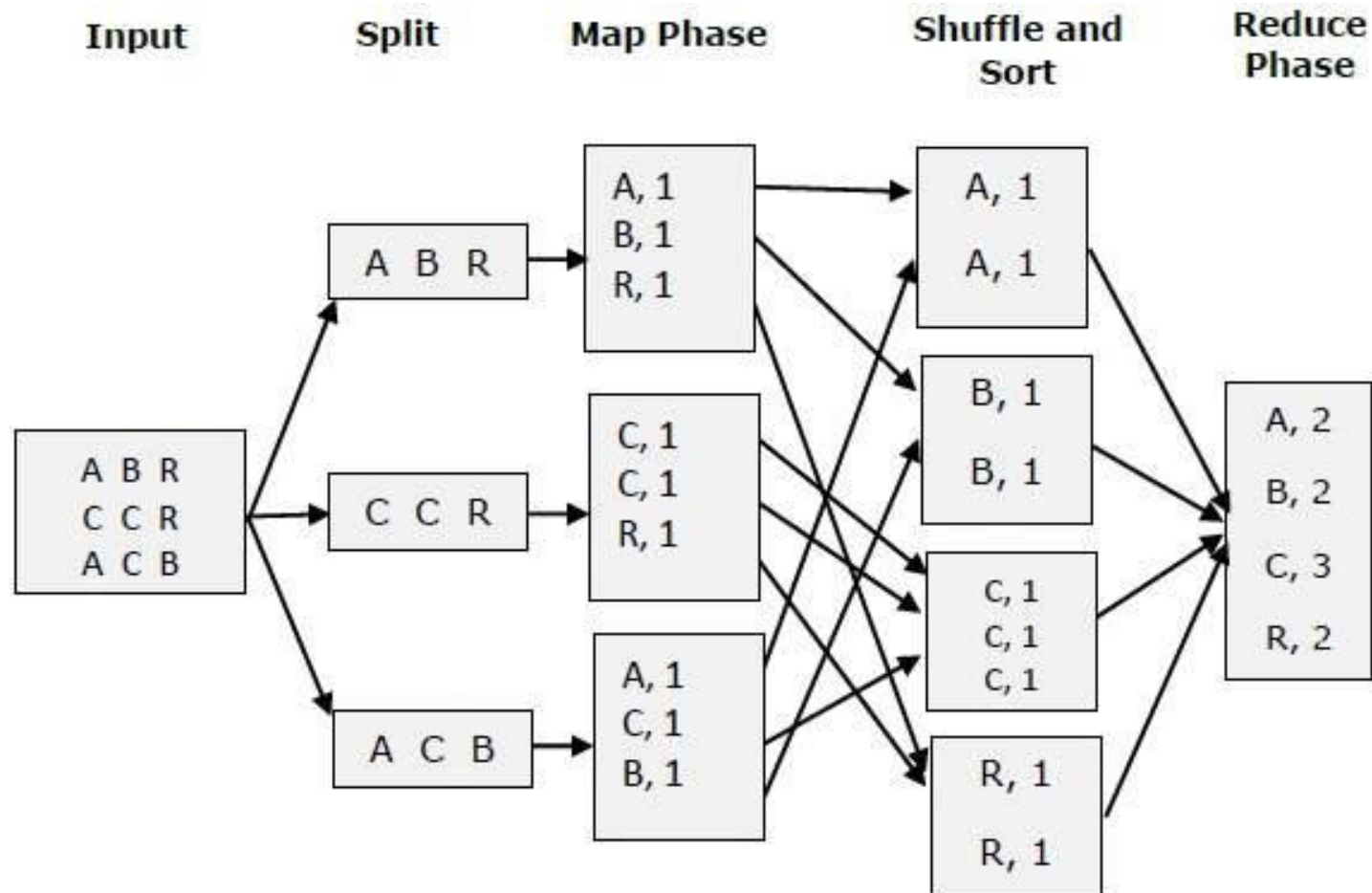
Fases



Fases

- **Input Phase** – Un lector de registros procesa los ficheros a tratar y los envía al mapper en forma de pares “clave-valor”
- **Map** – Map es una función definida por el usuario que procesa los pares clave-valor, aplicando ciertas condiciones, generando otro set de pares resultante.
- **Intermediate Keys** – Los resultados de la función Map.
- **Combiner** – Es un tipo de Reducer que agrupa los pares de datos similares en conjuntos de datos identificables. Toma las Intermediate Keys y aplica funciones de agregación definidas por el usuario. No es una parte fija de MapReduce (es opcional).
- **Shuffle and Sort** – La tarea del Reducer inicia con este paso. Descarga los pares clave-valor agrupadas en la máquina local donde corre el reducer. Los pares individuales se ordenan para poder ser iterados de una forma más fácil en el reducer.
- **Reducer** – Toma los pares agrupados y aplica la función de Reduce definida por el usuario. Aquí, los datos son agregados, filtrados y combinados según los objetivos definidos durante el proceso. Cuando su ejecución ha acabado, los resultados son 0 o más pares de clave-valor que se entregan al paso final.
- **Output Phase** – En esta fase, un formateador de salida convierte los pares de salida del reducer y los escribe en un fichero.

Fases



Funciones HDFS

```
hdfs dfs -mkdir /user/cloudera/input
```

→ Crea un directorio en hdfs

```
hdfs dfs -put testfile1 /user/cloudera/input
```

→ Sube un fichero local a hdfs

```
hdfs dfs -ls /user/cloudera/input
```

→ Muestra el contenido de un directorio hdfs

```
hdfs dfs -cat  
/user/cloudera/output_new/part-00000
```

→ Muestra el contenido de un fichero en hdfs

```
hdfs dfs -getmerge  
/user/cloudera/output_new_0/*  
wordcount_num0_output.txt
```

→ Descarga y agrupa los ficheros de un directorio hdfs en local

```
hdfs dfs -rm -r input
```

→ Borra un directorio hdfs y su contenido

Librería de ejemplos

Disponible en:

<https://github.com/JoanPinol/EAE/tree/master/hadoop>

Aquí encontrarás el código de más ejemplos!

Mapper

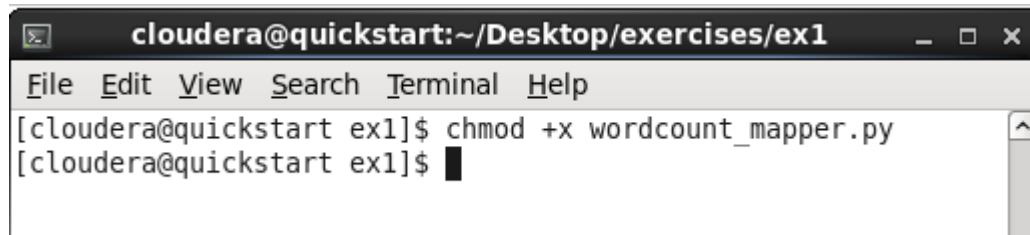
```
1  #!/usr/bin/env python
2
3  import sys                #a python module with system functions for this OS
4
5  for line in sys.stdin:
6
7      line = line.strip()    #strip is a method, ie function, associated
8                             # with string variable, it will strip
9                             # the carriage return (by default)
10     keys = line.split()    #split line at blanks (by default),
11                             # and return a list of keys
12     for key in keys:       #a for loop through the list of keys
13         value = 1
14         print('{0}\t{1}'.format(key, value) ) #the {} is replaced by 0th,1st items in format list
15                                             #also, note that the Hadoop default is 'tab' separates key from the value
16
```

Reducer

```
1  #!/usr/bin/env python
2
3  import sys
4
5  last_key      = None          #initialize these variables
6  running_total = 0
7
8  for input_line in sys.stdin:
9      input_line = input_line.strip()
10
11     this_key, value = input_line.split("\t", 1) #the Hadoop default is tab separates key value
12                                     #the split command returns a list of strings, in this case into 2 variables
13     value = int(value)             #int() will convert a string to integer (this program does no error checking)
14
15     if last_key == this_key:       #check if key has changed ('==' is logical equality check
16         running_total += value     # add value to running total
17
18     else:
19         if last_key:               #if this key that was just read in
20                                     # is different, and the previous
21                                     # (ie last) key is not empty,
22                                     # then output
23                                     # the previous <key running-count>
24         print( "{0}\t{1}".format(last_key, running_total) )
25                                     # hadoop expects tab(ie '\t')
26                                     # separation
27         running_total = value      #reset values
28         last_key = this_key
29
30 if last_key == this_key:
31     print( "{0}\t{1}".format(last_key, running_total))
32
```

Ejecución

Permitir modo ejecución:



```
cloudera@quickstart:~/Desktop/exercises/ex1
File Edit View Search Terminal Help
[cloudera@quickstart ex1]$ chmod +x wordcount_mapper.py
[cloudera@quickstart ex1]$
```

Crear directorio en hdfs y subir los ficheros:



```
cloudera@quickstart:~/Desktop/exercises/ex1
File Edit View Search Terminal Help
[cloudera@quickstart ex1]$ hdfs dfs -mkdir /user/cloudera/input
[cloudera@quickstart ex1]$ hdfs dfs -put input/* /user/cloudera/input
[cloudera@quickstart ex1]$
```

Ejecución

Test en local:

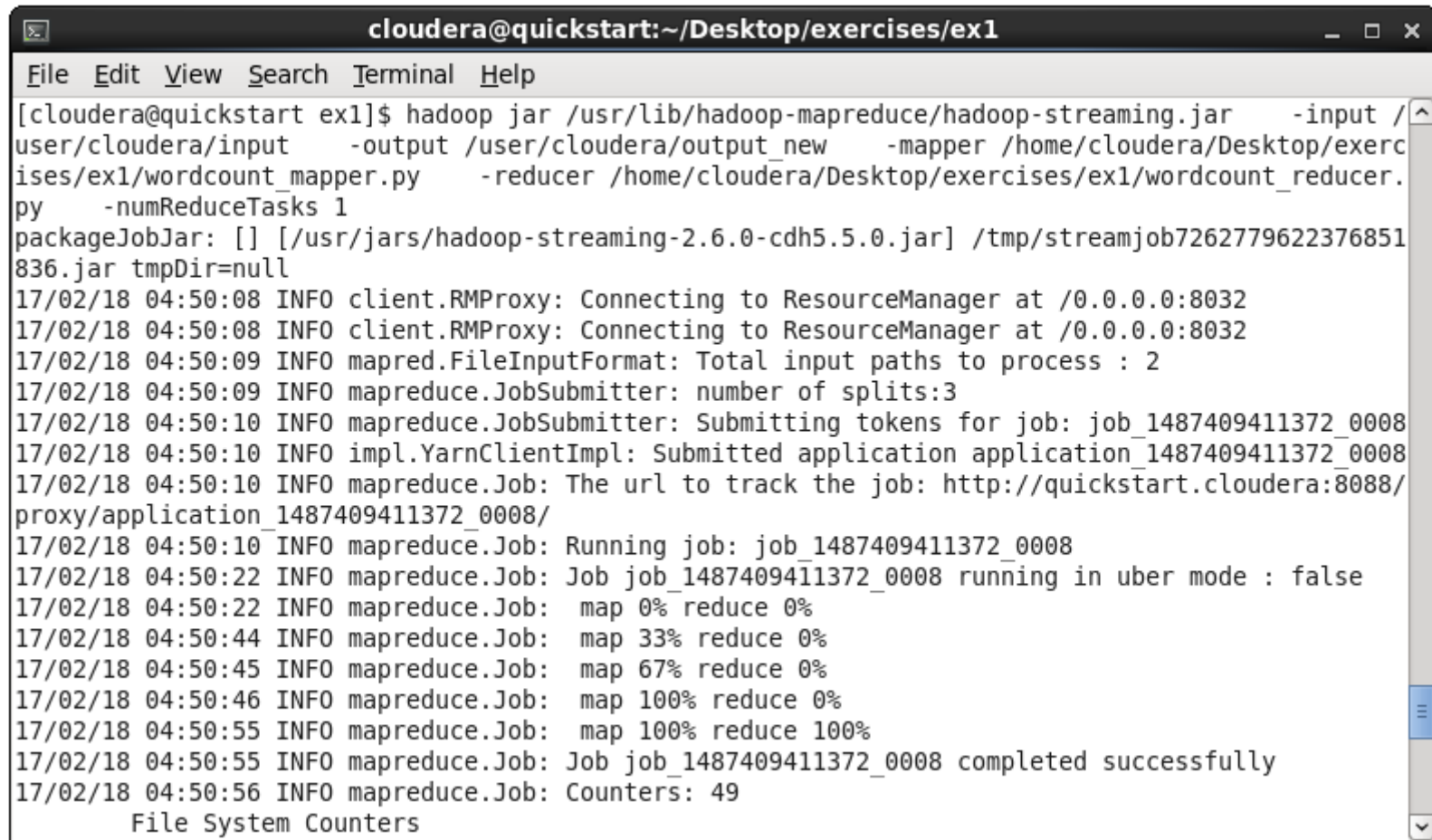


```
cloudera@quickstart:~/Desktop/exercises/ex1
File Edit View Search Terminal Help
[cloudera@quickstart ex1]$ ls
[cloudera@quickstart ex1]$ cat input/* | ./wordcount_mapper.py | sort | ./wordcount_reducer.py
a      1
A      1
ago    1
Another 1
away   1
episode 1
far     2
galaxy  1
in      1
long    1
of      1
Star    1
time    1
Wars    1
[cloudera@quickstart ex1]$
```

Sentencia: `cat input/* | ./wordcount_mapper.py | sort | ./wordcount_reducer.py`

Ejecución

Ejecutar en hdfs:


A terminal window titled 'cloudera@quickstart:~/Desktop/exercises/ex1' showing the execution of a Hadoop streaming job. The command used is 'hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -input /user/cloudera/input -output /user/cloudera/output_new -mapper /home/cloudera/Desktop/exercises/ex1/wordcount_mapper.py -reducer /home/cloudera/Desktop/exercises/ex1/wordcount_reducer.py -numReduceTasks 1'. The output shows the job being submitted and running successfully, with progress updates for map and reduce tasks. The job ID is 'job_1487409411372_0008'.

```
cloudera@quickstart:~/Desktop/exercises/ex1
File Edit View Search Terminal Help
[cloudera@quickstart ex1]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -input /
user/cloudera/input -output /user/cloudera/output_new -mapper /home/cloudera/Desktop/exerc
ises/ex1/wordcount_mapper.py -reducer /home/cloudera/Desktop/exercises/ex1/wordcount_reducer.
py -numReduceTasks 1
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.5.0.jar] /tmp/streamjob7262779622376851
836.jar tmpDir=null
17/02/18 04:50:08 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032
17/02/18 04:50:08 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032
17/02/18 04:50:09 INFO mapred.FileInputFormat: Total input paths to process : 2
17/02/18 04:50:09 INFO mapreduce.JobSubmitter: number of splits:3
17/02/18 04:50:10 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1487409411372_0008
17/02/18 04:50:10 INFO impl.YarnClientImpl: Submitted application application_1487409411372_0008
17/02/18 04:50:10 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/
proxy/application_1487409411372_0008/
17/02/18 04:50:10 INFO mapreduce.Job: Running job: job_1487409411372_0008
17/02/18 04:50:22 INFO mapreduce.Job: Job job_1487409411372_0008 running in uber mode : false
17/02/18 04:50:22 INFO mapreduce.Job: map 0% reduce 0%
17/02/18 04:50:44 INFO mapreduce.Job: map 33% reduce 0%
17/02/18 04:50:45 INFO mapreduce.Job: map 67% reduce 0%
17/02/18 04:50:46 INFO mapreduce.Job: map 100% reduce 0%
17/02/18 04:50:55 INFO mapreduce.Job: map 100% reduce 100%
17/02/18 04:50:55 INFO mapreduce.Job: Job job_1487409411372_0008 completed successfully
17/02/18 04:50:56 INFO mapreduce.Job: Counters: 49
File System Counters
```

Ejecución

Sentencia:


```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \  Librería mapreduce  
-input /user/cloudera/input \ Directorio hdfs de entrada  
-output /user/cloudera/output_new \ Directorio hdfs de salida  
-mapper /home/cloudera/Desktop/exercises/ex1/wordcount_mapper.py \ Mapper (py)  
-reducer /home/cloudera/Desktop/exercises/ex1/wordcount_reducer.py \ Reducer (py)  
-numReduceTasks 1 Número de reducers
```



Se puede especificar tanto el numero de mappers como de reducers. Generalmente suelen haber tantos mappers como sets de datos y tantos reducers como nodos. Con 0 reducers, no se aplica “reducción”

Ejecución

Ver resultados:

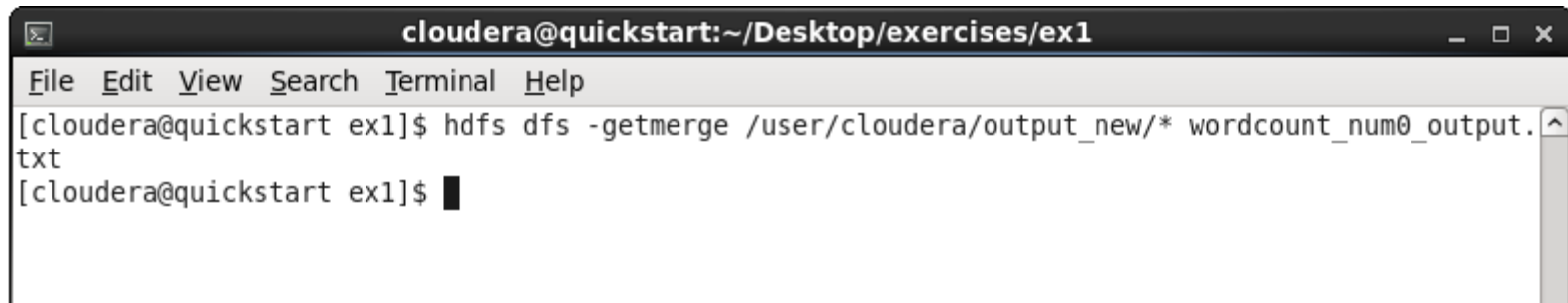


A terminal window titled "cloudera@quickstart:~/Desktop/exercises/ex1" with a menu bar (File, Edit, View, Search, Terminal, Help). The command executed is "hdfs dfs -cat /user/cloudera/output_new/part-00000". The output is a list of words and their counts:

```
[cloudera@quickstart ex1]$ hdfs dfs -cat /user/cloudera/output_new/part-00000
A      1
Another 1
Star   1
Wars   1
a      1
ago    1
away   1
episode 1
far    2
galaxy 1
in     1
long   1
of     1
time   1
[cloudera@quickstart ex1]$
```

Ejecución

Descargar resultados:



```
cloudera@quickstart:~/Desktop/exercises/ex1
File Edit View Search Terminal Help
[cloudera@quickstart ex1]$ hdfs dfs -getmerge /user/cloudera/output_new/* wordcount_num0_output.txt
[cloudera@quickstart ex1]$
```