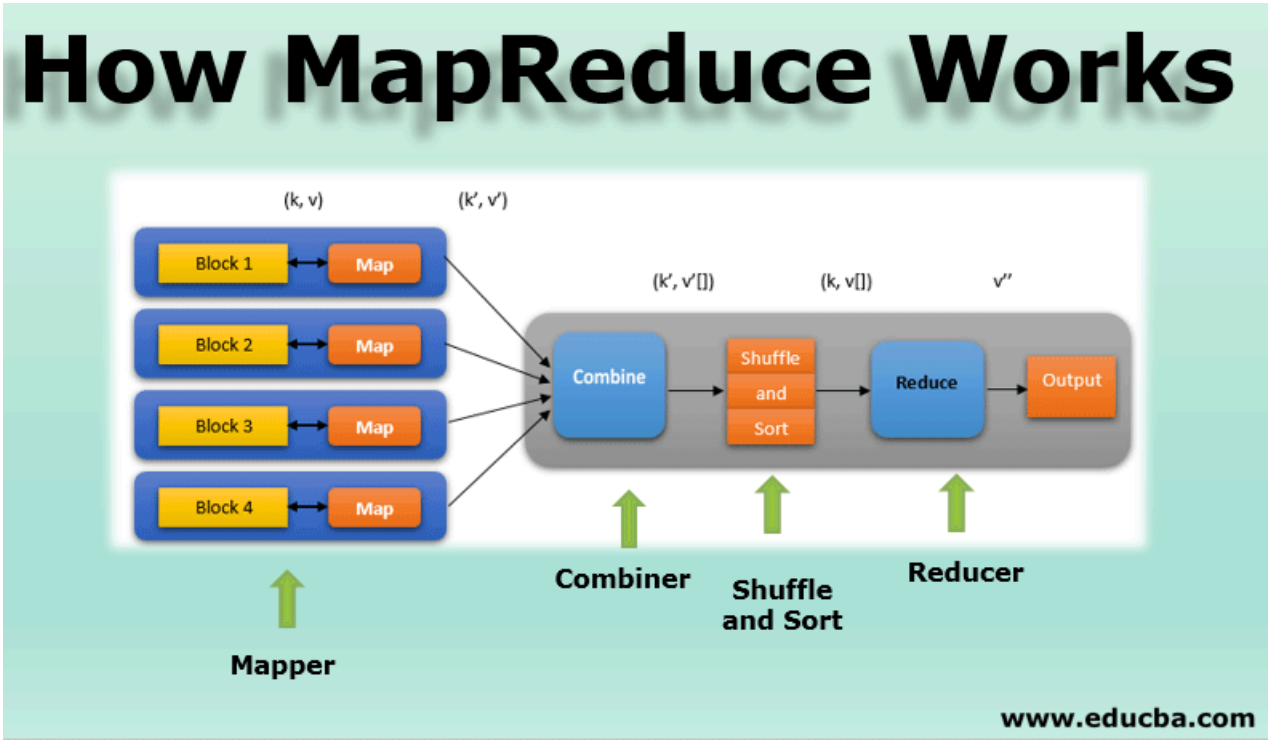
Mapreduce conteo de palabras



Adrián Yared Armas de la Nuez

Contenido

1. Actividad 1

1.1 Enunciado

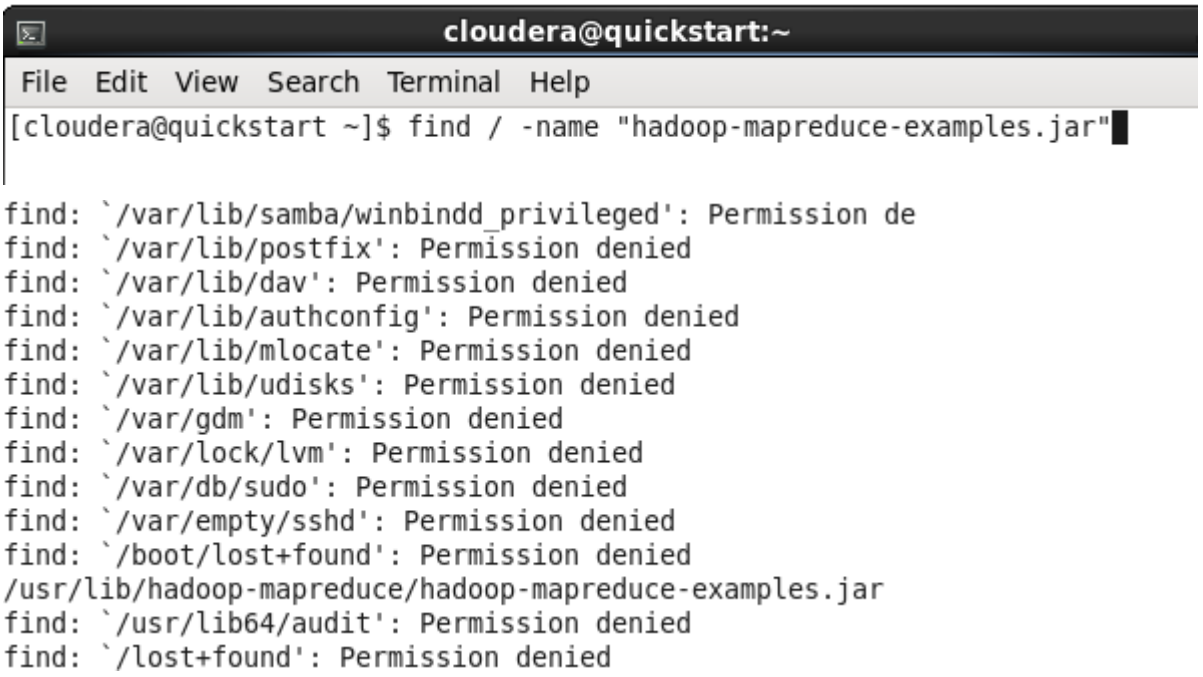
Comprueba la ruta y ejemplos de mapreduce.

1.1 Resolución

Los ejemplos se encuentran en el archivo hadoop-mapreduce-examples.jar, que generalmente se instala con Hadoop. Para verificar su ubicación: bash

1.1.1 Comandos

Compruebo el primer comando y da error de permisos:



Así que pruebo el siguiente y me da la ruta (/usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar):

```
[cloudera@quickstart ~]$ sudo find / -name "hadoop-mapreduce-examples.jar"

/usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar
[cloudera@quickstart ~]$
```

Consulta los ejemplos disponibles:

```
[cloudera@quickstart ~]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar
```

Me devuelve una lista gigante de ejemplos (solo muestro algunos de ellos en la captura):

```
bbp: A map/reduce program that uses Bailey-Borwein-Plouffe to compute exact digits of Pi.
dbcount: An example job that count the pageview counts from a database.
distbbp: A map/reduce program that uses a BBP-type formula to compute exact bits of Pi.
grep: A map/reduce program that counts the matches of a regex in the input.
join: A job that effects a join over sorted, equally partitioned datasets
multifilewc: A job that counts words from several files.
pentomino: A map/reduce tile laying program to find solutions to pentomino problems.
pi: A map/reduce program that estimates Pi using a quasi-Monte Carlo method.
randomtextwriter: A map/reduce program that writes 10GB of random textual data per node.
randomwriter: A map/reduce program that writes 10GB of random data per node.
secondarysort: An example defining a secondary sort to the reduce.
sort: A map/reduce program that sorts the data written by the random writer.
sudoku: A sudoku solver.
teragen: Generate data for the terasort
terasort: Run the terasort
teravalidate: Checking results of terasort
wordcount: A map/reduce program that counts the words in the input files.
wordmean: A map/reduce program that counts the average length of the words in the input files.
wordmedian: A map/reduce program that counts the median length of the words in the input files.
wordstandarddeviation: A map/reduce program that counts the standard deviation of the length of the v
```

2. Actividad 2

1.1 Enunciado

1.2 Código

Creo el directorio:

```
[cloudera@quickstart ~]$ hdfs dfs -mkdir -p /user/cloudera/input
[cloudera@quickstart ~]$
```

Creo el archivo input.txt y le meto el texto que pide el enunciado:

```
[cloudera@quickstart ~]$ nano input.txt
[cloudera@quickstart ~]$
```

```
  GNU nano 2.0.9                          File: input.txt

Hadoop es una herramienta poderosa.
Hadoop permite el procesamiento distribuido.
El procesamiento distribuido es eficiente.
```

Compruebo que se ha creado correctamente:

```
[cloudera@quickstart ~]$ hdfs dfs -put input.txt /user/cloudera/input/
[cloudera@quickstart ~]$
```

```
[cloudera@quickstart ~]$ hdfs dfs -ls /user/cloudera/input
Found 1 items
-rw-r--r--   1 cloudera cloudera        124 2024-11-26 09:52 /user/cloudera/input/input.txt
[cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/input
Found 1 items
-rw-r--r--   1 cloudera cloudera        124 2024-11-26 09:52 /user/cloudera/input/input.txt
[cloudera@quickstart ~]$
```

3. Actividad 3

1.1 Enunciado

1.2 Código

Ejecuta el comando mapreduce

```
[cloudera@quickstart ~]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount input/input.txt /user/cloudera/output-wordcount
```

Final del comando:

```
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=124
        File Output Format Counters
                Bytes Written=124
[cloudera@quickstart ~]$
```

Compruebo que hizo correctamente el wordcount:

```
[cloudera@quickstart ~]$ hdfs dfs -cat /user/cloudera/output-wordcount/part-r-00000
El      1
Hadoop  2
distribuido     1
distribuido.    1
eficiente.      1
el      1
es      2
herramienta     1
permite 1
poderosa.       1
procesamiento   2
una     1
[cloudera@quickstart ~]$
```

4. Actividad 4

4.1 Enunciado

Repite los pasos anteriores con el Quijote.

4.2 Código

4.2.1 Paso 1

Creo el archivo con el Quijote:

```
[cloudera@quickstart ~]$ hdfs dfs -put DonQuixotedelaMancha.txt /user/cloudera/i
nput/
[cloudera@quickstart ~]$
```

4.2.2 Paso 2

Compruebo que está en la ruta:

```
[cloudera@quickstart ~]$ hdfs dfs -ls /user/cloudera/input
Found 2 items
-rw-r--r--   1 cloudera cloudera    2141519 2024-11-26 10:09 /user/cloudera/inpu
t/DonQuixotedelaMancha.txt
-rw-r--r--   1 cloudera cloudera        124 2024-11-26 09:52 /user/cloudera/inpu
t/input.txt
[cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/input
Found 2 items
-rw-r--r--   1 cloudera cloudera    2141519 2024-11-26 10:09 /user/cloudera/inpu
t/DonQuixotedelaMancha.txt
-rw-r--r--   1 cloudera cloudera        124 2024-11-26 09:52 /user/cloudera/inpu
t/input.txt
[cloudera@quickstart ~]$
```

4.2.3 Paso 3

Elimino el wordcount anterior para poder crear uno nuevo:

```
[cloudera@quickstart ~]$ hdfs dfs -rm -r /user/cloudera/output-wordcount
Deleted /user/cloudera/output-wordcount
```

4.2.4 Paso 4

Ejecuto el wordcount de nuevo:

```
[cloudera@quickstart ~]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-e
xamples.jar wordcount input/DonQuixotedelaMancha.txt /user/cloudera/output-wordc
ount
```

Final del comando:

```
                Virtual memory (bytes) snapshot
                Total committed heap usage (byt
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=2141519
        File Output Format Counters
                Bytes Written=439411
[cloudera@quickstart ~]$
```

4.2.5 Paso 5

Leo el resultado:

```
[cloudera@quickstart ~]$ hdfs dfs -cat /user/cloudera/output-wordcount/part-r-00000
```

4.2.6 Paso 6

Resultado sin filtrar:

```
íbades  1
íbamos  3
íbamos, 2
íbase   1
ídolo   1
ímpetu  3
ímpetu, 1
ímpetus 1
ímpetus.        1
ínclitas        1
ínclito 2
índice  1
ínfima  1
ínsula  65
ínsula, 37
ínsula. 7
ínsula: 1
ínsula; 1
ínsula? 4
ínsulas 19
ínsulas,        5
ínsulas.        1
ínsulas:        1
ínsulas?        2
ínsulo, 1
ínsulos 2
ínsulos.        1
ínterin 1
íntimo  2
ñudos,  1
ñudos;  1
óiganme 1
órdenes 10
órdenes.        1
órgano  1
última  10
última, 1
últimamente     4
últimamente,    1
últimas 4
último  25
último, 1
último: 1
últimos 3
única   5
única,  1
única.  1
único   11
único,  1
únicos  1
```

4.2.7 Paso 7

Elimino el wordcount de nuevo:

```
[cloudera@quickstart ~]$ hdfs dfs -rm -r /user/cloudera/output-wordcount
Deleted /user/cloudera/output-wordcount
```

Ç

4.2.8 Paso 8

Creo un filtro de palabras que no me interesan (adverbios, preposiciones y pronombres):

echo -e "el\nla\nde\ny\nque\nlos\nlas\nun\nuna\nunos\nunas\ncon\ndel\na\npor\npara\nen\no\nse\nsu\nal\nlo\nmuy\nmas\nmenos\nsiempre\nnunca\naqui\nallí\nahora\nentonces\nhoy\nmanana\ntarde\nbien\nmal\nasi\nya\ncasi\ntan\ndemasiado\ntodo\nnada\nalgo\ncomo" > stopwords.txt

```
[cloudera@quickstart ~]$ echo -e "el\nla\nde\ny\nque\nlos\nlas\nun\nuna\nunos\nunas\ncon\ndel\na\npor\npara\nen\no\nse\nsu\nal\nlo\nmuy\nmás\nmenos\nsiempre\nnunca\naquí\nallí\nahora\nentonces\nhoy\nmañana\ntarde\nbien\nmal\nasí\nya\ncasi\ntan\ndemasiado\ntodo\nnada\nalgo\ncomo" > stopwords.txt
```

4.2.9 Paso 9

Filtro sin alfanuméricos y paso todo a minúscula:

hadoop fs -cat /user/cloudera/input/DonQuixotedelaMancha. txt | grep -o \b[A-Z][a-zA-Z]*\b' | tr "[: upper:]' "[: lower:]' | grep -v -w -f stopwords.txt > filtered input.txt

```
[cloudera@quickstart ~]$ hadoop fs -cat /user/cloudera/input/DonQuixotedelaMancha.txt | grep -o '\b[A-Z][a-zA-Z]*\b' | tr '[:upper:]' '[:lower:]' | grep -v -w -f stopwords.txt > filtered input.txt
```

4.2.10 Paso 10

Lo subo a HDFS:

```
[cloudera@quickstart ~]$ hadoop fs -put filtered input.txt /user/cloudera/input/filtered input.txt
```

4.2.11 Paso 11

Creo el wordcount:

```
[cloudera@quickstart ~]$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount \
> /user/cloudera/input/filtered_input.txt /user/cloudera/output-wordcount
```

4.2.12 Paso 12

Muestro las 5 palabras más frecuentes:

```
[cloudera@quickstart ~]$ hadoop fs -cat /user/cloudera/output-wordcount/part-r-00000 | \
> sort -k2 -nr | head -5
quijote 2175
sancho  2148
no      548
dios    518
panza   328
```