TECNOLOXÍAS DE COMPUTACIÓN PARA DATOS MASIVOS

Entrega 2

ANDRÉS CAMPOS CUIÑA FECHA DE ENTREGA: 31/10/2021

ÍNDICE

1	Probar el comando hdfs dfsadmin	1
2	Probar el comando hdfs fsck	3
3	Probar el uso de códigos de borrado (Erasure Codes)	10

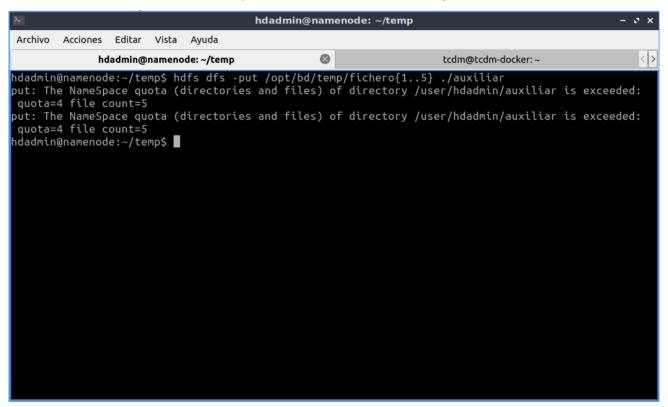
TCDM	doc.: practica2_CamposCuina.pdf
ICDIVI	Dágina 1 do 12
MaBD	Página 1 de 12

1 PROBAR EL COMANDO HDFS DFSADMIN

En este primer apartado lo que haremos es, en el NameNode y como usuario hdadmin, creamos un directorio en el directorio home del usuario hdadmin, al que llamaremos **auxiliar**, mediante el comando hdfs dfs -mkdir ./auxiliar. Una vez creado este directorio le pondremos una cuota de sólo 4 ficheros mediante el comando hdfs dfsadmin -setQuota 4 ./auxiliar.

Ahora lo que haremos es copiar una serie de ficheros de local a esta carpeta del HDFS. Lo primero que haremos es crear un directorio temporal, al que llamaremos **temp**, en el directorio **/opt/bd/** mediante el comando mkdir **/opt/bd/**temp y dentro de este nuevo directorio crearemos cinco ficheros vacíos mediante el comando touch fichero{1..5}.

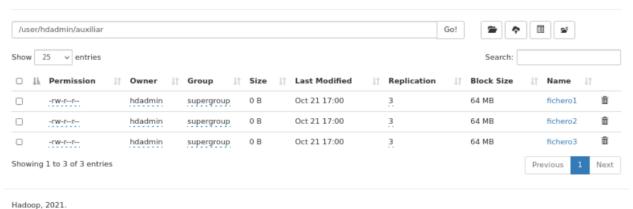
Una vez creados estos ficheros comprobaremos cuántos ficheros podemos copiar al directorio **auxiliar** que creamos anteriormente en el HDFS. Para esto usaremos el comando hdfs dfs -put /opt/temp/fichero{1..5} ./auxiliar. Mediante este comando intentaremos copiar 5 ficheros en el directorio en el que establecimos la cuota de 4 ficheros. El resultado de la ejecución de este comando es el siguiente:



Como podemos observar nos da dos errores, un error para cada uno de los ficheros **fichero4** y **fichero5**, pero copia los otros tres ficheros, como se puede ver si accedemos al *filesytem* a través de la interfaz web:

TCDM	doc.: practica2_CamposCuina.pdf
ICDIVI	Dágina 2 do 12
MaBD Página 2 de 12	Pagina 2 de 12

Browse Directory



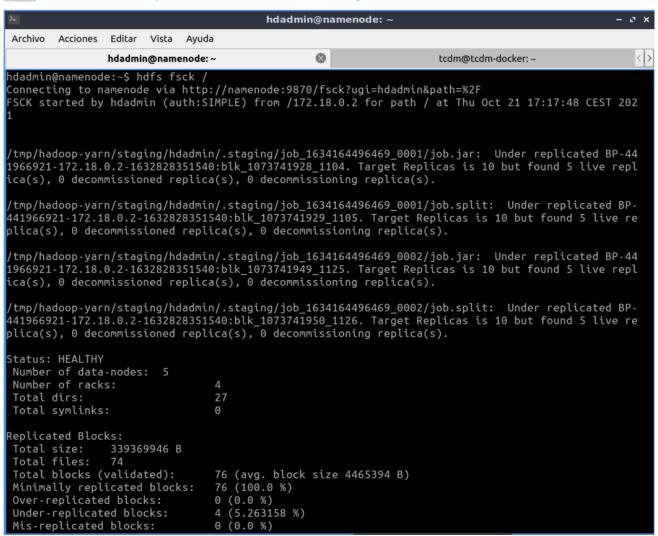
Consultando en la documentación del comando hdfs dfsadmin -setQuota encontramos que el propio directorio cuenta para su cuota, por lo que si queremos que se puedan guardar en un directorio un máximo de 4 ficheros, en realidad deberíamos ejecutar el comando hdfs dfsadmin -setQuota 5 <path_to_folder>, de esta manera se podrán almacenar 4 ficheros a mayores del propio directorio.

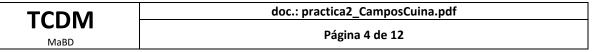
Finalmente, eliminamos tanto de local como del HDFS estos directorios creados de forma temporal.

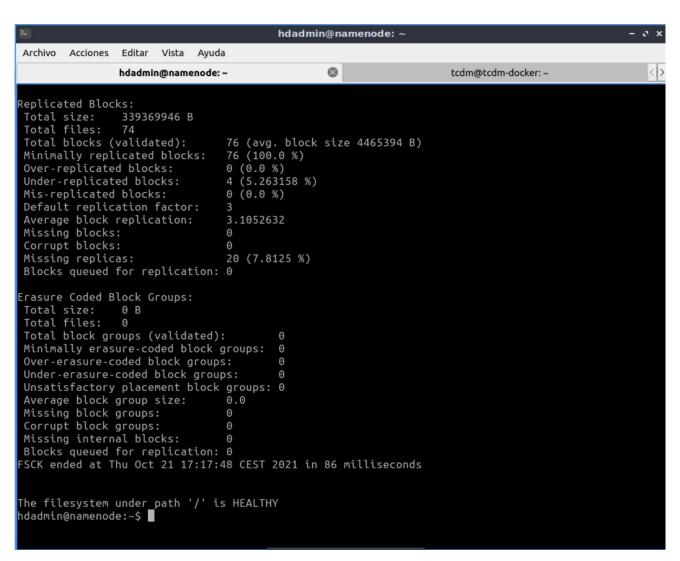
TCDM	doc.: practica2_CamposCuina.pdf
ICDIVI	Décino 2 de 13
MaBD	Página 3 de 12

2 PROBAR EL COMANDO HDFS FSCK

En el NameNode y como usuario hdadmin hacemos un chequeo de todo el HDFS mediante el comando hdfs fsck /. El resultado de la ejecución de este comando es el siguiente:





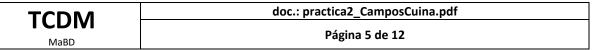


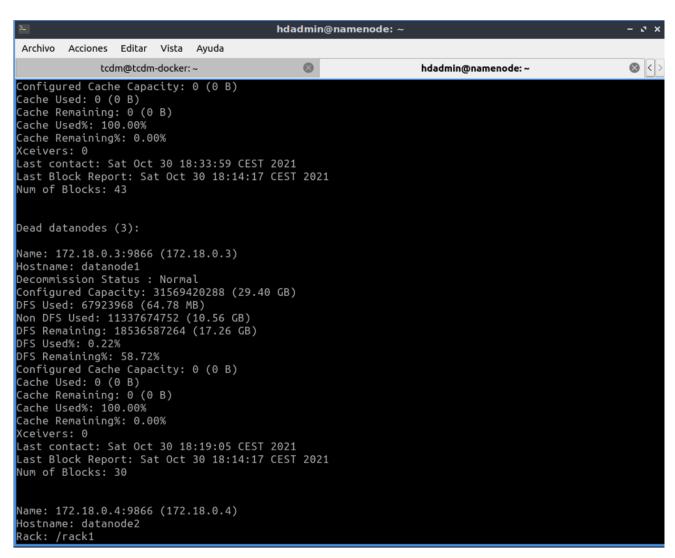
Como se puede ver hay 4 ficheros para los cuales el número de réplicas de sus bloques es inferior al que debería ser. Esto lo podemos ver en el mensaje de error que da: **Under replicated block. Target Replicas is 10 but found 5 live replica(s)**.

Como se puede observar, el target del número de réplicas para estos bloques es de 10 pero se han encontrado sólo cinco réplicas, por lo que nos da el error de que están *under-replicated*. Sin embargo, como podemos ver el número de réplicas por defecto en el sistema es de tres, por lo que estos bloques en realidad tienen más réplicas que el número de réplicas por defecto del sistema, sólo que estos ficheros tienen un número de réplicas establecido mayor (de 10).

Para arreglar este problema podríamos forzar a estos bloques a adaptarse al factor de replicación por defecto del resto del clúster mediante el comando hdfs dfs -setrep -w 3 <path to file>.

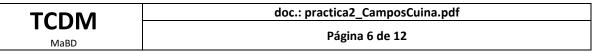
A continuación, paramos los datanodes de forma brusca ejecutando el comando docker container stop datanode1 datanode2 datanode4 para que sólo queden dos datanodes activos en dos racks diferentes. Tras esperar 10 minutos, mediante el comando hdfs dfsadmin -report podemos comprobar que sólo quedan dos datanodes vivos:

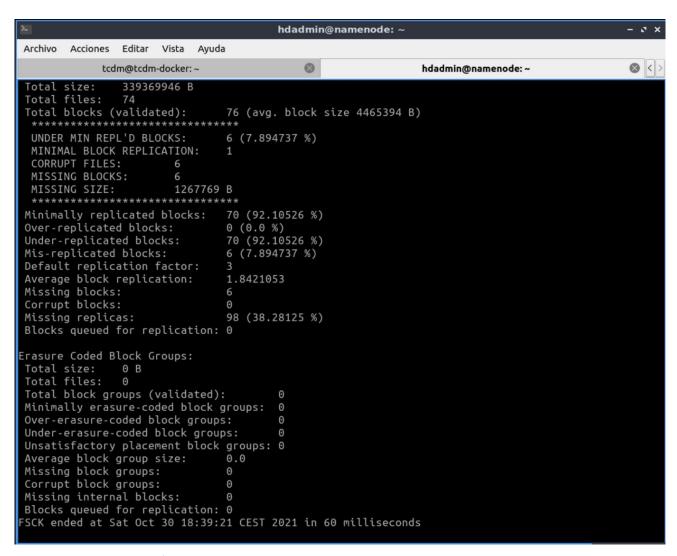




Como se puede ver en la salida del comando anterior, hay 3 datanodes muertos que se corresponden con los que paramos de forma deliberada.

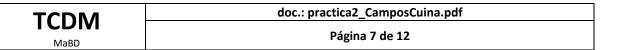
Volvemos a ejecutar el comando hdfs fsck /, el resultado de este comando esta vez es el siguiente:

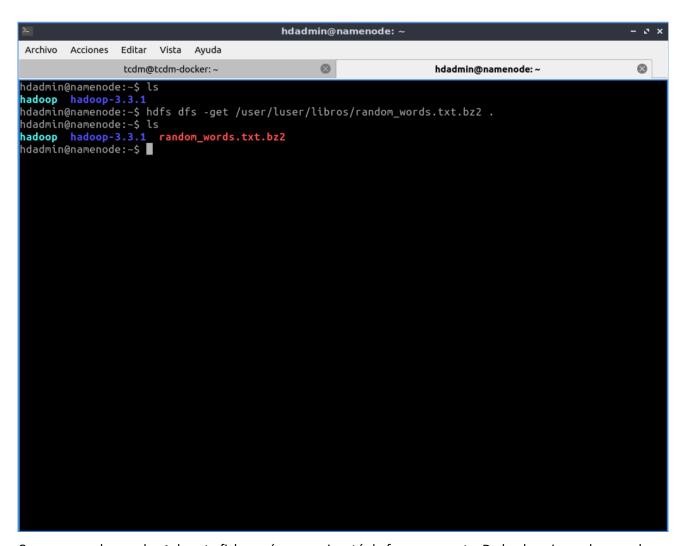




Como se puede ver, el número de bloques que aparecen ahora como *under-replicated* es igual a 70. Esto es un número de bloques *under-replicated* muy superior al anterior, debido a los nodos que hemos matado. La salida de este comando también nos muestra que el *filesystem* ahora está corrupto.

Tras esto, probamos a hacer un **get** del fichero **random_words.txt.bz2** para ver si se realiza de forma correcta. Para esto hacemos uso del comando hdfs dfs -get /user/luser/libros/random_words.txt.bz2 . que nos trae este fichero del HDFS a local. La salida de la ejecución de este comando es la siguiente:





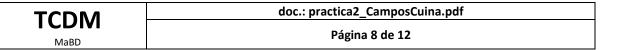
Como se puede ver el **get** de este fichero sí que se ejecutó de forma correcta. De hecho, si accedemos a la interfaz gráfica del NameNode podemos ver como los ficheros que están corruptos son los siguientes:

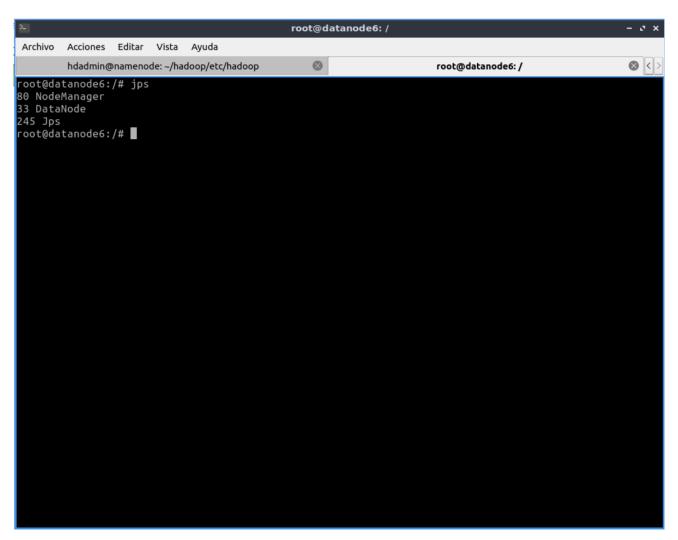
```
There are 6 missing blocks. The following files may be corrupted:

blk_1073741835 /user/luser/libros/pg18005.txt.gz
blk_1073741836 /user/luser/libros/pg2000.txt.gz
blk_1073741921 /user/hdadmin/QuasiMonteCarlo_1634166760970_1289019487/in/part9
blk_1073741931 /tmp/hadoop-yarn/staging/hdadmin/.staging/job_163416496469_0001/job.xml
blk_1073741938 /user/hdadmin/QuasiMonteCarlo_1634166988512_741858546/in/part5
blk_1073741942 /user/hdadmin/QuasiMonteCarlo_1634166988512_741858546/in/part9

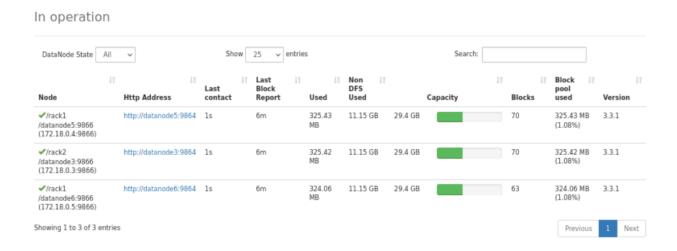
Please check the logs or run fsck in order to identify the missing blocks. See the Hadoop FAQ for common causes and potential solutions.
```

Seguimos ahora los pasos indicados en la práctica 1 para añadir un nuevo datanode, el **datanode6**. Una vez hemos añadido este datanode de forma correcta volvemos a ejecutar el comando hdfs fsck / para realizar un chequeo de todo el *filesystem*. De esta manera podremos ver como los bloques se van replicando en el nuevo datanode que acabamos de añadir al clúster. Lo primero de todo será comprobar en el datanode6, el cual acabamos de crear, que los demonios están funcionando de forma correcta. Esto lo hacemos mediante el comando jps:





Después, desde la interfaz web del NameNode podemos ver como los bloques se van replicando en el datanode6:



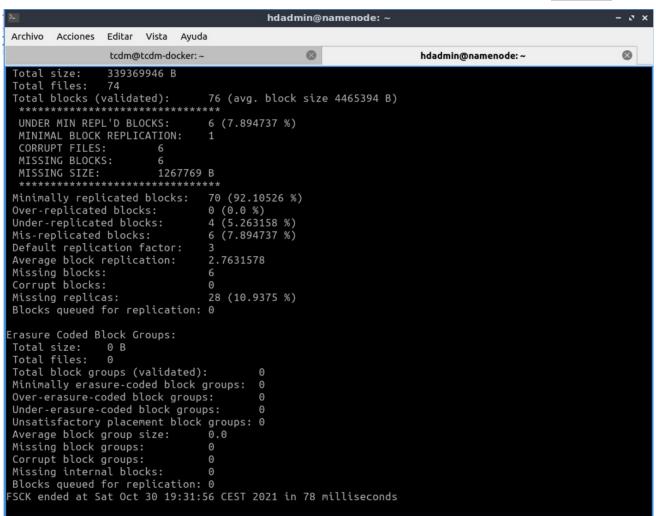
Finalmente, el número de bloques en el datanode6 iguala el número de bloques en los datanodes que no fueron detenidos, por lo que se completa el replicado de los bloques:

TCDM	doc.: practica2_CamposCuina.pdf
ICDIVI	Dágina O da 13
MaBD	Página 9 de 12

In operation



También lo podemos comprobar haciendo un chequeo del filesystem mediante el comando hdfs fsck /:

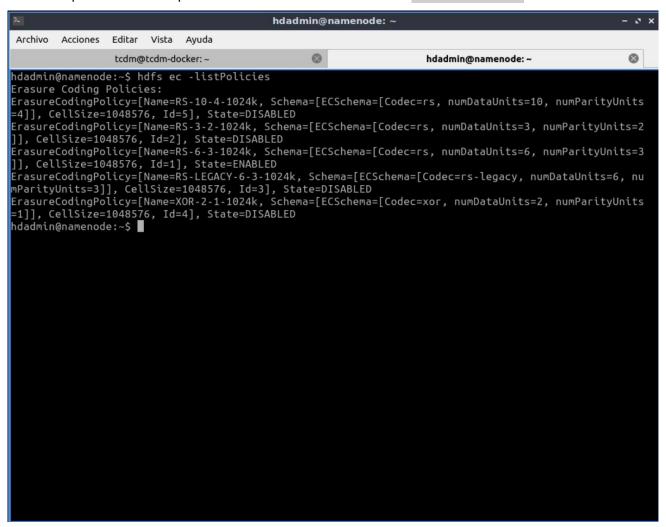


Como se puede ver en esta captura, el número de bloques *under-replicated* vuelve a ser el que era antes de matar a los datanodes.

TCDM	doc.: practica2_CamposCuina.pdf
ICDIVI	Dágina 10 da 13
MaBD Página 10 de 12	ragina 10 de 12

3 PROBAR EL USO DE CÓDIGOS DE BORRADO (ERASURE CODES)

Vemos las políticas de EC disponibles mediante el uso del comando hdfs ec -listPolicies:

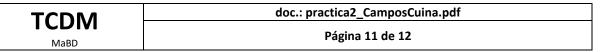


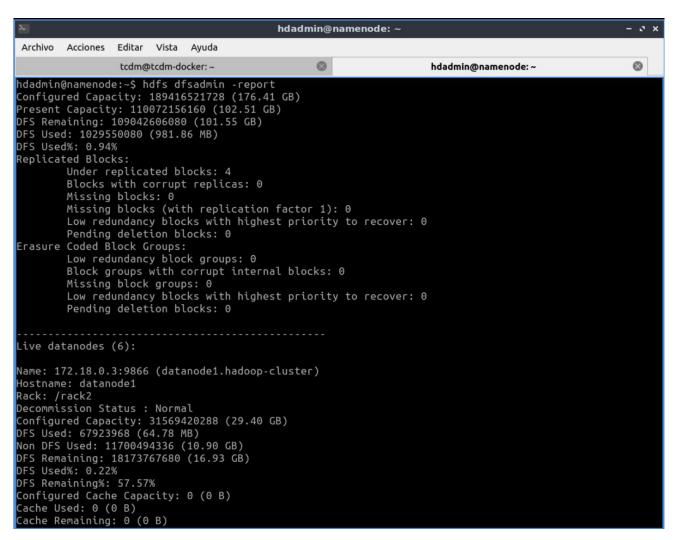
Como se puede ver, tenemos activada la política de EC por defecto, de nombre 'RS-6-3-1024k'. Tras haber iniciado los dockers necesarios para tener al menos 5 datanodes (en realidad iniciamos los 6 datanodes que tenemos creados para hacer uso del clúster al completo), en el NameNode, como usuario hdadmin habilitamos la política 'RS-3-2-1024k' ejecutando el comando hdfs ec -enablePolicy -policy RS-3-2-1024k.

Después de ejecutar este comando, creamos una carpeta /user/grandes en el HDFS mediante el comando hdfs dfs -mkdir /user/grandes para después indicar que se le aplique la política de EC 'RS-3-2-1024k' mediante el comando hdfs ec -setPolicy -path /user/grandes -policy RS-3-2-1024k.

Las políticas EC (*Erasure Codes*), como dice en la documentación de hadoop, se nombran siguiendo el siguiente esquema: *codec-num data blocks-num parity blocks-cell size*. Estas políticas encapsulan como codificar y decodificar los ficheros.

Ahora para ver que funciona seguiremos los pasos indicados en el enunciado. En primer lugar, con el comando hdfs dfsadmin -report comprobamos el espacio ocupado en DFS:

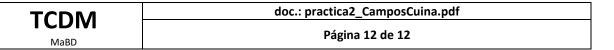


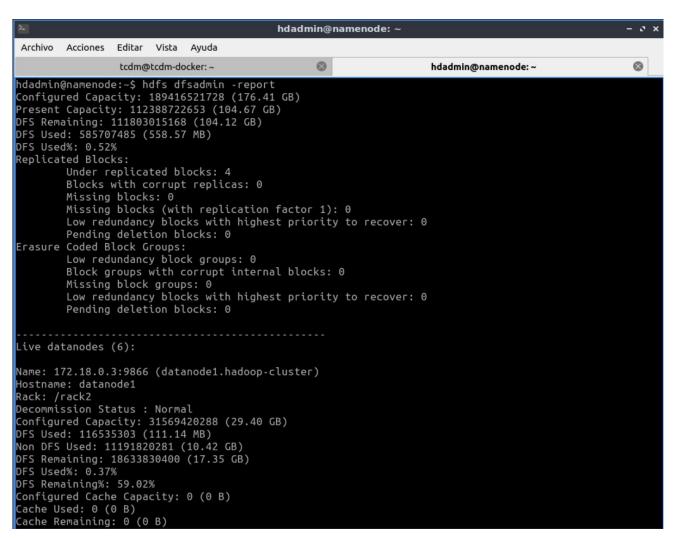


Como se puede ver, el espacio ocupado en DFS es de 981,86MB, siendo esta memoria el 0,94% del espacio total del DFS.

Ejecutamos hdfs dfs -get /user/luser/libros/random_words.txt.bz2 para obtener ese fichero grande desde HDFS. Después borramos ese fichero de HDFS mediante el comando hdfs dfs -rm /user/luser/libros/random words.txt.bz2 y borramos la papelera mediante el comando hdfs dfs -expunge.

Ponemos el fichero en la carpeta /user/grandes mediante el comando hdfs dfs -put random_words.txt.bz2 /user/grandes y comprobamos de nuevo el espacio ocupado en el DFS:





Ahora se puede ver como el espacio total usado del DFS es de 558,57MB, que es menor que el espacio usado anteriormente (981,86MB). Esto se debe a que el fichero **random_words.txt.bz2**, que ocupa 315MB aproximadamente, antes se dividía en 5 bloques, cada uno de los cuales se replicaba 3 veces, por lo que ocupaba un total de 5 bloques. Mediante el uso de esta política de EC el fichero se divide en 3 celdas de datos más 2 celdas de paridad, además de ir codificado mediante el codificado Reed-Salomon, por lo que ocupa menos espacio en el DFS.