Máster en Big Data

Tecnologías de Almacenamiento

2. Hands-On: Uso de HDFS

- Albert Ripoll

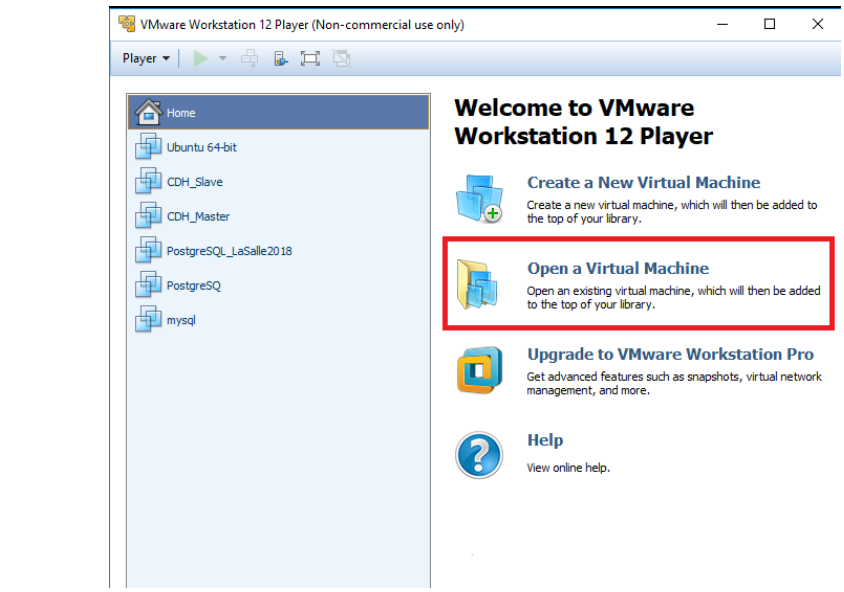
Índice

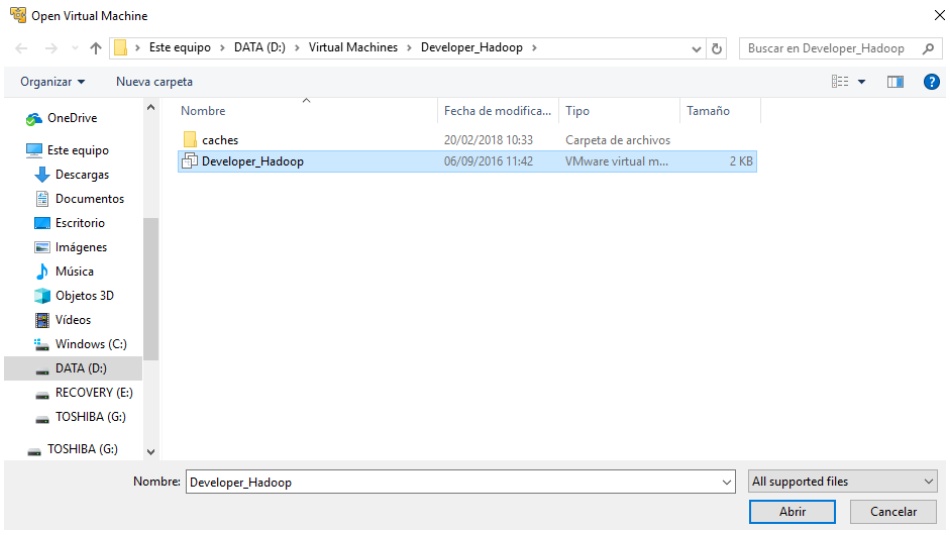
1. Introducción

El objetivo de este Hands-On es ganar confianza con el uso de HDFS por línea de comandos.

2. Desplegar el nodo de Hadoop

Junto con este enunciado se proporciona una instancia de clúster Hadoop pseudodistribuido. Descomprimir la carpeta e importar la máquina a VMWare como ya se ha hecho anteriormente.





3. Uso de HDFS

Todas las instrucciones deben introducirse mediante Shell de Linux.

IMPORTANTE: Adjuntar el comando y captura de que los ficheros afectados por el comando se han creado/modificado/eliminado correctamente.

3.1. Explorar HDFS

Para explorar HDFS (Hadoop Distributed File System) usamos el comando "hadoop fs".

a) Muestra el comando de ayuda de Hadoop

```
[training@localhost bin]$ hadoop fs -help
Usage: hadoop fs [generic options]
        [-cat [-ignoreCrc] <src> ...]
        [-chgrp [-R] GROUP PATH...]
        [-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...]
        [-chown [-R] [OWNER][:[GROUP]] PATH...]
        [-copyFromLocal <localsrc> ... <dst>]
        [-copyToLocal [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-count [-q] <path> ...]
        [-cp <src> ... <dst>]
        [-df [-h] [<path> ...]]
        [-du [-s] [-h] <path> ...]
        [-expunge]
        [-get [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-getmerge [-nl] <src> <localdst>]
        [-help [cmd ...]]
        [-ls [-d] [-h] [-R] [<path> ...]]
        [-mkdir [-p] <path> ...]
        [-moveFromLocal <localsrc> ... <dst>]
        [-moveToLocal <src> <localdst>]
        [-mv <src> ... <dst>]
        [-put <localsrc> ... <dst>]
        [-rm [-f] [-r|-R] [-skipTrash] <src> ...]
        [-rmdir [--ignore-fail-on-non-empty] <dir> ...]
        [-setrep [-R] [-w] <rep> <path/file> ...]
        [-stat [format] <path> ...]
        [-tail [-f] <file>]
        [-test -[ezd] <path>]
        [-text [-ignoreCrc] <src> ...]
        [-touchz <path> ...]
        [-usage [cmd ...]]

-cat [-ignoreCrc] <src> ...:    Fetch all files that match the file pattern <src
>
                and display their content on stdout.

-chgrp [-R] GROUP PATH...:      This is equivalent to -chown ... :GROUP ...

-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...:       Changes permissions of a
 file.
                        This works similar to shell's chmod with a few exception
s.

                -R      modifies the files recursively. This is the only option
                        currently supported.

                MODE    Mode is same as mode used for chmod shell command.
```

```
                MODE    Mode is same as mode used for chmod shell command.
                        Only letters recognized are 'rwxXt'. E.g. +t,a+r,g-w,+rw
x,o=r

                OCTALMODE Mode specifed in 3 or 4 digits. If 4 digits, the first
 may
                be 1 or 0 to turn the sticky bit on or off, respectively.  Unlik
e shell command, it is not possible to specify only part of the mode
                        E.g. 754 is same as u=rwx,g=rx,o=r

                        If none of 'augo' is specified, 'a' is assumed and unlik
e
                        shell command, no umask is applied.

-chown [-R] [OWNER][:[GROUP]] PATH...:  Changes owner and group of a file.
                        This is similar to shell's chown with a few exceptions.

                        -R      modifies the files recursively. This is the only
 option
                        currently supported.

                        If only owner or group is specified then only owner or
                        group is modified.

                        The owner and group names may only cosists of digits, al
phabet,
                        and any of '-_.@/' i.e. [-_.@/a-zA-Z0-9]. The names are
case
                        sensitive.

                        WARNING: Avoid using '.' to separate user name and group
 though
                        Linux allows it. If user names have dots in them and you
 are
                        using local file system, you might see surprising result
s since
                        shell command 'chown' is used for local files.

-copyFromLocal <localsrc> ... <dst>:    Identical to the -put command.

-copyToLocal [-ignoreCrc] [-crc] <src> ... <localdst>:  Identical to the -get co
mmand.

-count [-q] <path> ...: Count the number of directories, files and bytes under t
he paths
                that match the specified file pattern.  The output columns are:
                DIR_COUNT FILE_COUNT CONTENT_SIZE FILE_NAME or
                QUOTA REMAINING_QUATA SPACE_QUOTA REMAINING_SPACE_QUOTA
                      DIR_COUNT FILE_COUNT CONTENT_SIZE FILE_NAME

-cp <src> ... <dst>:    Copy files that match the file pattern <src> to a
```

```
-df [-h] [<path> ...]:  Shows the capacity, free and used space of the filesyste
m.
                If the filesystem has multiple partitions, and no path to a
                particular partition is specified, then the status of the root
                partitions will be shown.
                  -h   Formats the sizes of files in a human-readable fashion
                       rather than a number of bytes.

-du [-s] [-h] <path> ...:       Show the amount of space, in bytes, used by the
files that
                match the specified file pattern. The following flags are option
al:
                  -s   Rather than showing the size of each individual file that
                       matches the pattern, shows the total (summary) size.
                  -h   Formats the sizes of files in a human-readable fashion
                       rather than a number of bytes.

                Note that, even without the -s option, this only shows size summ
aries
                one level deep into a directory.
                The output is in the form
                        size    name(full path)

-expunge:       Empty the Trash

-get [-ignoreCrc] [-crc] <src> ... <localdst>:  Copy files that match the file p
attern <src>
                to the local name.  <src> is kept.  When copying multiple,
                files, the destination must be a directory.

-getmerge [-nl] <src> <localdst>:       Get all the files in the directories tha
t
                match the source file pattern and merge and sort them to only
                one file on local fs. <src> is kept.
                  -nl   Add a newline character at the end of each file.

-help [cmd ...]:        Displays help for given command or all commands if none
                is specified.

-ls [-d] [-h] [-R] [<path> ...]:        List the contents that match the specifi
ed file pattern. If
                path is not specified, the contents of /user/<currentUser>
                will be listed. Directory entries are of the form
                        dirName (full path) <dir>
                and file entries are of the form
                        fileName(full path) <r n> size
                where n is the number of replicas specified for the file
                and size is the size of the file, in bytes.
                  -d  Directories are listed as plain files.
                  -h  Formats the sizes of files in a human-readable fashion
                      rather than a number of bytes.
```

```
-rmdir [--ignore-fail-on-non-empty] <dir> ...:  Removes the directory entry spec
ified by each directory argument,
                provided it is empty.

-setrep [-R] [-w] <rep> <path/file> ...:        Set the replication level of a f
ile.
                The -R flag requests a recursive change of replication level
                for an entire tree.

-stat [format] <path> ...:      Print statistics about the file/directory at <pa
th>
                in the specified format. Format accepts filesize in blocks (%b),
 group name of owner(%g),
                filename (%n), block size (%o), replication (%r), user name of o
wner(%u), modification date (%y, %Y)

-tail [-f] <file>:      Show the last 1KB of the file.
                        The -f option shows appended data as the file gr
ows.

-test -[ezd] <path>:    If file exists, has zero length, is a directory
                then return 0, else return 1.

-text [-ignoreCrc] <src> ...:   Takes a source file and outputs the file in text
 format.
                The allowed formats are zip and TextRecordInputStream.

-touchz <path> ...:     Creates a file of zero length
                at <path> with current time as the timestamp of that <path>.
                An error is returned if the file exists with non-zero length

-usage [cmd ...]:       Displays the usage for given command or all commands if
none
                is specified.

Generic options supported are
-conf <configuration file>     specify an application configuration file
-D <property=value>            use value for given property
-fs <local|namenode:port>      specify a namenode
-jt <local|jobtracker:port>    specify a job tracker
-files <comma separated list of files>    specify comma separated files to be co
pied to the map reduce cluster
-libjars <comma separated list of jars>    specify comma separated jar files to
include in the classpath.
-archives <comma separated list of archives>    specify comma separated archives
 to be unarchived on the compute machines.
```

b) Listar el contenido de la raíz

```
[training@localhost bin]$ hadoop fs -ls /
Found 4 items
drwxr-xr-x   - hbase supergroup          0 2024-03-06 14:26 /hbase
drwxrwxrwt   - hdfs  supergroup          0 2016-09-05 12:00 /tmp
drwxrwxrwx   - hue   supergroup          0 2016-09-05 12:01 /user
drwxr-xr-x   - hdfs  supergroup          0 2016-09-05 12:01 /var
```

Vemos que dentro de Hadoop lo que hay son 4 carpetas: hbase, tmp, usr, var creadas en las fechas y horas listadas.

c) Listar el contenido del directorio */user*

```
[training@localhost bin]$ hadoop fs -ls /user
Found 3 items
drwxr-xr-x   - hue      supergroup          0 2016-09-05 12:01 /user/hive
drwxr-xr-x   - hdfs     supergroup          0 2016-09-05 12:01 /user/hue
drwxr-xr-x   - training supergroup          0 2016-09-05 12:01 /user/training
```

Accedemos a la carpeta user y vemos lo que hay. Hay 3 carpetas: hive, hue, training creadas en las fechas y horas listadas.

3.2. Insertar Archivos

a) En el directorio */home/training/training_materials/developer/data* del filesystem local descomprimir el archivo s*hakespeare.tar.gz* e insertar en HDFS el contenido de la carpeta descomprimida en */user/training/Shakespeare*

Para descomprimir el archivo shakespeare.tar.gz en el directorio local:

• Primero accedemos al directorio que nos ha dicho el enunciado con cd (*change directory*).

• Después usamos la función *tar* que es la que se utiliza para manipular archivos comprimidos. Su nombre proviene de "tape archive" de cuando las copias eran con cintas magnéticas.

-z: Indica a tar que descomprima el archivo utilizando gzip.

-x: Le indica a tar que extraiga el contenido del archivo. (como si fuera cntrl+x; cortar)

-v: Hace que tar imprima información detallada sobre el proceso en la consola (es lo que se conoce como modo verbose).

-f: Indica el nombre del archivo con el que se trabajará que es el *shakespeare.tar.gz*

```
[training@localhost bin]$ cd /home/training/training_materials/developer/data
[training@localhost data]$ tar -zxvf shakespeare.tar.gz
shakespeare/
shakespeare/comedies
shakespeare/glossary
shakespeare/histories
shakespeare/poems
shakespeare/tragedies
[training@localhost data]$
```

• Finalmente copiamos el contenido descomprimido a HDFS. En primer lugar creamos un directorio (*-mkdir*) en /user/training/Shakespeare. La opción "-p" indica a Hadoop que cree todos los directorios necesarios en la ruta especificada, incluso si algunos de los directorios intermedios no existen. Es decir, si la carpeta de training dentro de user no existe, la opción "-p" hará que Hadoop cree todos los directorios intermedios automàticamente. En segundo lugar, copiamos del local "-copyFromLocal" toda (*) la carpeta de shakespeare en la carpeta "Shakespeare" que acabamos de crear.

```
[training@localhost data]$ hadoop fs -mkdir -p /user/training/Shakespeare

[training@localhost data]$ hadoop fs -copyFromLocal shakespeare/* /user/training/Shakespeare/
[training@localhost data]$
[training@localhost data]$
```

• Otra forma de hacerlo seria con la función "put"

```
[training@localhost data]$ hadoop fs -mkdir -p /user/training/Shakespeare1
[training@localhost data]$ hadoop fs -put shakespeare/* /user/training/Shakespeare1/
[training@localhost data]$
```

b) Listar el directorio donde se ha realizado la importación

Se usa la función "ls" para listar el contenido de dicho directorio.

```
[training@localhost data]$ hadoop fs -ls /user/training/Shakespeare

Found 5 items
-rw-r--r--   1 training supergroup    1784616 2024-03-06 14:47 /user/training/Shakespeare/comedies
-rw-r--r--   1 training supergroup      58976 2024-03-06 14:47 /user/training/Shakespeare/glossary
-rw-r--r--   1 training supergroup    1479035 2024-03-06 14:47 /user/training/Shakespeare/histories
-rw-r--r--   1 training supergroup     268140 2024-03-06 14:47 /user/training/Shakespeare/poems
-rw-r--r--   1 training supergroup    1752440 2024-03-06 14:47 /user/training/Shakespeare/tragedies
[training@localhost data]$
[training@localhost data]$
```

c) Crear un directorio llamado *weblog* en HDFS en la ruta siguiente: */user/training*

```
[training@localhost data]$ hadoop fs -mkdir -p /user/training/weblog

[training@localhost data]$
[training@localhost data]$
```

d) Importar el archivo *access_log.gz*. (gunzip –c descomprimirá el archivo y volcará todo el contenido a la salida estándar y – en hadoop leerá los datos de la salida estándar)

```
[training@localhost data]$ gunzip -c access_log.gz | hadoop fs -put - /user/training/weblog/access_log

[training@localhost data]$
[training@localhost data]$
```

Utilizamos gunzip -c para descomprimir el archivo access_log.gz y luego hadoop fs -put - para leer desde la salida estándar (-) y colocar los datos en HDFS en la ruta /user/training/weblog/access_log.

Verificamos que que el archivo access_log se haya importado correctamente a HDFS leyendo la carpeta weblog:

```
[training@localhost data]$ hadoop fs -ls /user/training/weblog/
Found 1 items
-rw-r--r--   1 training supergroup  504941532 2024-03-06 14:56 /user/training/weblog/access_log
[training@localhost data]$
```

3.3. Manipular Archivos

a) Listar el contenido de la carpeta *Shakespeare*

```
[training@localhost data]$ hadoop fs -ls /user/training/Shakespeare

Found 5 items
-rw-r--r--   1 training supergroup    1784616 2024-03-06 14:47 /user/training/Shakespeare/comedies
-rw-r--r--   1 training supergroup      58976 2024-03-06 14:47 /user/training/Shakespeare/glossary
-rw-r--r--   1 training supergroup    1479035 2024-03-06 14:47 /user/training/Shakespeare/histories
-rw-r--r--   1 training supergroup     268140 2024-03-06 14:47 /user/training/Shakespeare/poems
-rw-r--r--   1 training supergroup    1752440 2024-03-06 14:47 /user/training/Shakespeare/tragedies
[training@localhost data]$
[training@localhost data]$
```

b) Borra el archivo *glossary*

```
[training@localhost data]$ hadoop fs -rm /user/training/Shakespeare/glossary
Deleted /user/training/Shakespeare/glossary
[training@localhost data]$
[training@localhost data]$
```

c) Muestra las últimas 50 líneas del archivo *histories*

d) hadoop fs -cat /user/training/Shakespeare/histories: Esto imprime el contenido del archivo histories ubicado en el directorio /user/training/Shakespeare en HDFS. hadoop fs -cat simplemente muestra el contenido del archivo en la salida estándar. | tail -n 50: Utiliza el operador de tubería (|) para pasar la salida del comando anterior (las líneas completas del archivo histories) al comando tail -n 50. Este último comando muestra las últimas 50 líneas de su entrada.

```
[training@localhost data]$ hadoop fs -cat /user/training/Shakespeare/histories | tail -n 50
RICHMOND        God and your arms be praised, victorious friends,
        The day is ours, the bloody dog is dead.

DERBY   Courageous Richmond, well hast thou acquit thee.
        Lo, here, this long-usurped royalty
        From the dead temples of this bloody wretch
        Have I pluck'd off, to grace thy brows withal:
        Wear it, enjoy it, and make much of it.

RICHMOND        Great God of heaven, say Amen to all!
        But, tell me, is young George Stanley living?

DERBY   He is, my lord, and safe in Leicester town;
        Whither, if it please you, we may now withdraw us.

RICHMOND        What men of name are slain on either side?

DERBY   John Duke of Norfolk, Walter Lord Ferrers,
        Sir Robert Brakenbury, and Sir William Brandon.

RICHMOND        Inter their bodies as becomes their births:
        Proclaim a pardon to the soldiers fled
        That in submission will return to us:
        And then, as we have ta'en the sacrament,
        We will unite the white rose and the red:
        Smile heaven upon this fair conjunction,
        That long have frown'd upon their enmity!
        What traitor hears me, and says not amen?
        England hath long been mad, and scarr'd herself;
        The brother blindly shed the brother's blood,
        The father rashly slaughter'd his own son,
        The son, compell'd, been butcher to the sire:
        All this divided York and Lancaster,
        Divided in their dire division,
        O, now, let Richmond and Elizabeth,
        The true succeeders of each royal house,
        By God's fair ordinance conjoin together!
        And let their heirs, God, if thy will be so.
        Enrich the time to come with smooth-faced peace,
        With smiling plenty and fair prosperous days!
        Abate the edge of traitors, gracious Lord,
        That would reduce these bloody days again,
        And make poor England weep in streams of blood!
        Let them not live to taste this land's increase
        That would with treason wound this fair land's peace!
        Now civil wounds are stopp'd, peace lives again:
        That she may long live here, God say amen!

        [Exeunt]
        [END]
[training@localhost data]$
```

e) Descarga el archivo *poems* en el filesystem local

Para coger archivo de Hadoop se usa el -get.

Usamos “ls” para leer (listar) lo que tenemos en el archivo local. No ponemos “hadoop fs” al inicio porque ahora este archivo no está en hadoop sino que está en nuestra computadora local ya que lo acabamos de traer. En la salida vemos que aparece el archivo “poems” en verde. Los distintos colores son para indicar distintos formatos de archivos.

```
[training@localhost data]$ hadoop fs -get /user/training/Shakespeare/poems
[training@localhost data]$ ls
access_log.gz  auctiondata.csv  bible.tar.gz  invertedIndexInput.tgz  movielens.readme  movielens-small.sql  movielens.sql  nameyeartestdata  poems  shakespeare  shakespeare-stream.tar.gz  shakespeare.tar.gz
[training@localhost data]$
```

4. Aprendizajes

4.1. Ejercicio b) Listar directorio de donde se ha realizado la importación

```
[training@localhost ~]$ hadoop fs -ls
Found 3 items
drwxr-xr-x   - training supergroup          0 2024-03-06 15:33 Shakespeare
drwxr-xr-x   - training supergroup          0 2024-03-06 15:54 Shakespeare1
drwxr-xr-x   - training supergroup          0 2024-03-06 14:56 weblog
```

Comandos para escribir:

Hadoop fs → Se actúa en hadoop

-ls → List directory contents (listar el contenido que hay)

Respuesta en pantalla:

• En la primera columna hay los permisos (RWX).

d → Indica que es un directorio.

r → Readable (permiso sobre el archivo que indica se puede leer)

w → Writeable (permiso sobre el archivo que indica se puede escribir)

x → Executable (permiso sobre el archivo que indica se puede ejecutar)

Las siguientes 3 letras son los permisos del usuario dueño del archivo (rwx). Las siguientes 3 letras los permisos del grupo al que pertenece el usuario (r-x). Y las 3 siguientes letras es para cualquier otro grupo (r-x)

De esa forma en este caso el usuario dueño puede leer, escribir y ejecutar sus propios archivos. El grupo de usuarios puede leer y ejecutar pero no (-) escribir. Los otros usuarios de otro grupo puede lee pueden leer y ejecutar pero no (-) escribir

• En la segunda columna hay el nombre de usuario y el grupo.

Training -> es como se llama nuestro usuario

Supergroup -> es como se llama nuestro grupo de usuario

• En la tercera columna hay el nombre de usuario y el grupo.

0 → Cuando es un directorio

Otro número -> El tamaño en bytes que tiene el archivo

• En la cuarta columna hay la fecha que se creó la carpeta en formato yyyy-mm-dd hh-mm.

• En la quinta columna hay el nombre de la carpeta o archivo.

Comandos para escribir:

Hadoop fs → Se actúa en hadoop

-ls → List directory contents (listar el contenido que hay)

/user/training/Shakespeare → Ruta de carpetas que nos lleva a la carpeta de Shakespeare

```
[training@localhost ~]$ hadoop fs -ls /user/training/Shakespeare
Found 5 items
-rw-r--r--   1 training supergroup    1784616 2024-03-06 14:47 /user/training/Shakespeare/comedies
-rw-r--r--   1 training supergroup      58976 2024-03-06 15:33 /user/training/Shakespeare/glossary
-rw-r--r--   1 training supergroup    1479035 2024-03-06 14:47 /user/training/Shakespeare/histories
-rw-r--r--   1 training supergroup     268140 2024-03-06 14:47 /user/training/Shakespeare/poems
-rw-r--r--   1 training supergroup    1752440 2024-03-06 14:47 /user/training/Shakespeare/tragedies
```

Respuesta en pantalla:

Es del mismo estilo que la respuesta anterior.

4.2. Ejercicio c) Listar directorio de donde se ha realizado la importación