

1. mkdir:

similar to Unix mkdir command, it is used for creating directories in HDFS.

```
hdfs dfs -mkdir /user/data/
```

Notes:

- In order to create a sub directory /user/hadoop, its parent directory /user must already exist. Otherwise *'No such file or directory'* error message will be returned.

2. ls:

similar to Unix ls command, it is used for listing directories in HDFS.

```
hdfs dfs -ls /
```

Options:

and file entries are of the form:

- d Directories are listed as plain files.
- h Formats the sizes of files in a human-readable fashion rather than a number of bytes.
- R Recursively list the contents of directories.

```
hdfs dfs -ls -R /
```

The **-lsr** command can be used for recursive listing.

permissions - userId groupId sizeOfDirectory(in bytes) modificationDate(yyyy-MM-dd HH:mm)
directoryName

3. put:

Copies files from local file system to HDFS. This is similar to -copyFromLocal command.

Copying fails if the file already exists,

```
hdfs dfs -put sample.txt /user/data/
```

Or

```
hdfs dfs -copyFromLocal sample.txt /user/data
```

Options:

- p preserves access and modification times, ownership and the mode. Passing
- f overwrites the destination if it already exists.

4. get:

Copies files from HDFS to local file system. This is similar to -copyToLocal command.

```
hdfs dfs -get /user/data/sample.txt /BDA
```

Or

```
hdfs dfs -copyToLocal /user/data/sample.txt /BDA
```

5. cat:

similar to Unix cat command, it is used for displaying contents of a file.

```
hdfs dfs -cat /user/data/sample.txt
```

6. cp:

similar to Unix cp command, it is used for copying files from one directory to another within HDFS.

```
hdfs dfs -cp /user/data/sample.txt /user/test
```

7. mv:

similar to Unix mv command, it is used for moving a file from one directory to another within HDFS.

```
hdfs dfs -mv /user/data/sample.txt /user/test/
```

8. rm:

similar to Unix rm command, it is used for removing a file from HDFS.

```
hdfs dfs -rm /user/test/sample.txt
```

Options:

The command -rmr can be used for recursive delete.

-skipTrash option bypasses trash, if enabled, and immediately deletes <src>

-f If the file does not exist, do not display a diagnostic message or modify the exit status to reflect an error.

-[rR] Recursively deletes directories

```
hdfs dfs -rm -r /user/test/sample.txt
```

Note:

- Directories can't be deleted by -rm command. We need to use -rm -r (recursive remove) command to delete directories and files inside them. Only files can be deleted by -rm command.

We can use

```
hdfs dfs -rmdir
```

 command to delete directories.

9. test:

This command can be used to test a hdfs file's existence or zero length or is it a directory.

Syntax is

```
hdfs dfs -test -[defsz] /user/test/test.txt
```

Options:

-d return 0 if <path> is a directory.

-e return 0 if <path> exists.

-f return 0 if <path> is a file.

-s return 0 if file <path> is greater than zero bytes in size.

-z return 0 if file <path> is zero bytes in size, else return 1.

10. expunge:

This command is used to empty the trash in hadoop file system.

```
hdfs dfs -expunge
```

11. appendToFile:

Appends the contents of all the given local files to the given destination file on HDFS. The destination file will be created if it does not exist.

If <localSrc> is –, then the input is read from stdin.

```
hdfs dfs -appendToFile derby.log data.tsv /in/appendfile
```

```
hdfs dfs -cat /in/appendfile
```

12. tail:

Shows the last 1KB of the file. Syntax is:

Syntax:

```
hdfs dfs -tail [-f] <file>
```

Options:

-f Shows appended data as the file grows.

```
hdfs dfs -tail /in/appendfile
```

13. stat:

This option prints statistics about the file/directory at <path> in the specified format.

Syntax:

```
hdfs dfs -stat [format] <path>
```

```
hdfs dfs -stat /in/appendfile
```

```
2014-11-26 04:57:04
```

Examples of Format:

modification date (**%y**, **%Y**)

```
hdfs dfs -stat %Y /in/appendfile
```

```
1416977824841
```

file size in blocks (**%b**)

```
hdfs dfs -stat %b /in/appendfile
```

```
20981
```

replication (**%r**)

```
hdfs dfs -stat %r /in/appendfile
```

```
1
```

block size (**%o**)

```
hdfs dfs -stat %o /in/appendfile
```

```
134217728
```

Other Format accepts: group name of owner(**%g**), file name (**%n**), user name of owner(**%u**),

14. df:

Shows the capacity, free and used space of the filesystem. 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.

Syntax:

```
hdfs dfs -df [-h] [<path> ...]
```

Options:

-h Formats the sizes of files in a human-readable fashion rather than a number of bytes.

15. du:

Show the amount of space, in bytes, used by the files that match the specified file pattern.

Syntax:

hdfs dfs -du [-s] [-h] <path>

Options:

The following flags are optional:

-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 summaries one level deep into a directory.

The output is in the form:

size name(full path)

16. count:

Count the number of directories, files and bytes under the paths that match the specified file pattern.

Syntax:

hdfs dfs -count [-q] <path>

The output columns are:

DIR_COUNT FILE_COUNT CONTENT_SIZE FILE_NAME or

QUOTA REMAINING_QUOTA SPACE_QUOTA REMAINING_SPACE_QUOTA

DIR_COUNT FILE_COUNT CONTENT_SIZE FILE_NAME