HADOOP & NATURAL LANGUAGE PROCESSING (DATA SCIENCE FLAVOR) ALL IN ONE COURSE

GITHUB · https://github.com/nitinkaushik01/Hadoop-Natural-Language-Processing

TRANSFER FILE FROM LOCAL MACHINE TO SANDBOX

```
SCP -P 2222 <LOCAL_DIRECTORY_FILE> ROOT@SANDBOX.HORTONWORKS.COM:<SANDBOX_DIRECTORY_FILE>
```

TRANSFER FILE FROM SANDBOX TO LOCAL MACHINE

```
SCP -P 2222 ROOT@SANDBOX.HORTONWORKS.COM:<SANDBOX_DIRECTORY_FILE>  <LOCAL_DIRECTORY_FILE>
```

TRANSFERRING FILE FROM HADOOP LOCAL TO HDFS

Assuming we have data in the file called file.txt in the local system which is ought to be saved in the HDFS file system. Follow the steps given below to insert the required file in the Hadoop file system.

Step 1

You have to create an input directory.

```
HDFS DFS -MKDIR /USER/INPUT
```

Step 2

Transfer and store a data file from local systems to the Hadoop file system using the put command.

```
HDFS DFS -PUT /HOME/FILE.TXT /USER/INPUT
```

Step 3

You can verify the file using ls command.

```
HDFS DFS -LS /USER/INPUT
```

RETRIEVING FILE FROM HADOOP LOCAL TO HDFS

Assume we have a file in HDFS called outfile. Given below is a simple demonstration for retrieving the required file from the Hadoop file system.

Get the file from HDFS to the local file system using get command.

```
HDFS DFS -GET /USER/OUTPUT/ /HOME/
```

THERE ARE MANY OTHER COMMANDS WHICH WE UTILIZE IN DOING DIFFERENT HADOOP RELATED ACTIVITIES OR OPERATIONS.

1. Lists the contents of the directory specified by path, showing the names, permissions, owner, size and modification date for each entry.
   ls <path>

2. Behaves like -ls, but recursively displays entries in all subdirectories of path.
   lsr <path>

3. Shows disk usage, in bytes, for all the files which match path; filenames are reported with the full HDFS protocol prefix.
   du <path>

4. Like -du, but prints a summary of disk usage of all files/directories in the path.
   dus <path>

5. Moves the file or directory indicated by source to destination, within HDFS.
   mv <source><destination>

6. Copies the file or directory identified by source to destination, within HDFS.
   cp <source> <destination>

7. Removes the file or empty directory identified by path.
   rm <path>

8. Removes the file or directory identified by path. Recursively deletes any child entries (i.e., files or subdirectories of path).
   rmr <path>

9. Copies the file or directory from the local file system identified by LocalSource to Destination within the DFS.
   put <LocalSource> <Destination> OR
   copyFromLocal <LocalSource> <Destination>

10. Copies the file or directory from the local file system identified by LocalSource to Destination within HDFS, and then deletes the local copy on success.
    moveFromLocal <LocalSource> <Destination>

11. Copies the file or directory in HDFS identified by source to the local file system path identified by LocalDestination.
    get [-crc] <source> <LocalDestination> OR
    copyToLocal <source> <LocalDestination>

12. Retrieves all files that match the path src in HDFS, and copies them to a single, merged file in the local file system identified by localDest.
    getmerge <src> <localDest>

13. Displays the contents of filename on stdout.
    cat <filename>

14. Works like -get, but deletes the HDFS copy on success.
    moveToLocal <src> <localDest>

15. Creates a directory named path in HDFS.
    mkdir <path>

16. Sets the target replication factor for files identified by path to rep. (The actual replication factor will move toward the target over time)
    setrep [-R] [-w] rep <path>

17. Creates a file at path containing the current time as a timestamp. Fails if a file already exists at path, unless the file is already size 0.
    touchz <path>

18. Returns 1 if path exists; has zero length; or is a directory or 0 otherwise.
    test -[ezd] <path>

19. Prints information about path. Format is a string which accepts file size in blocks (%b), filename (%n), block size (%o), replication (%r), and modification date (%y, %Y).
    stat [format] <path>

20. Shows the last 1KB of file on stdout.
    tail [-f] <file2name>

21. Changes the file permissions associated with one or more objects identified by path.... Performs changes recursively with R. mode is a 3-digit octal mode, or {augo}+/-{rwxX}. Assumes if no scope is specified and does not apply an umask.
    chmod [-R] mode,mode,... <path>...

22. Sets the owning user and/or group for files or directories identified by path.... Sets owner recursively if -R is specified.
    chown [-R] [owner][:[group]] <path>...

23. Sets the owning group for files or directories identified by path.... Sets group recursively if -R is specified.
    chgrp [-R] group <path>...

24. Returns usage information for one of the commands listed above. You must omit the leading '-' character in cmd.
    help <cmd-name>