**Sed Command in Unix and Linux Examples**

Sed is a Stream Editor used for modifying the files in unix (or linux). Whenever you want to make changes to the file automatically, sed comes in handy to do this. Most people never learn its power; they just simply use sed to replace text. You can do many things apart from replacing text with sed. Here I will describe the features of sed with examples.  
  
Consider the below text file as an input.

>cat file.txt

unix is great os. unix is opensource. unix is free os.

learn operating system.

unixlinux which one you choose.

**Sed Command Examples**

**1.** Replacing or substituting string  
  
Sed command is mostly used to replace the text in a file. The below simple sed command replaces the word "unix" with "linux" in the file.

>sed 's/unix/linux/' file.txt

linux is great os. unix is opensource. unix is free os.

learn operating system.

linuxlinux which one you choose.

Here the "s" specifies the substitution operation. The "/" are delimiters. The "unix" is the search pattern and the "linux" is the replacement string.  
  
By default, the sed command replaces the first occurrence of the pattern in each line and it won't replace the second, third...occurrence in the line.  
  
**2.** Replacing the nth occurrence of a pattern in a line.  
  
Use the /1, /2 etc flags to replace the first, second occurrence of a pattern in a line. The below command replaces the second occurrence of the word "unix" with "linux" in a line.

>sed 's/unix/linux/2' file.txt

unix is great os. linux is opensource. unix is free os.

learn operating system.

unixlinux which one you choose.

**3.** Replacing all the occurrence of the pattern in a line.  
  
The substitute flag /g (global replacement) specifies the sed command to replace all the occurrences of the string in the line.

>sed 's/unix/linux/g' file.txt

linux is great os. linux is opensource. linux is free os.

learn operating system.

linuxlinux which one you choose.

**4.** Replacing from nth occurrence to all occurrences in a line.  
  
Use the combination of /1, /2 etc and /g to replace all the patterns from the nth occurrence of a pattern in a line. The following sed command replaces the third, fourth, fifth... "unix" word with "linux" word in a line.

>sed 's/unix/linux/3g' file.txt

unix is great os. unix is opensource. linux is free os.

learn operating system.

unixlinux which one you choose.

**5.** Changing the slash (/) delimiter  
  
You can use any delimiter other than the slash. As an example if you want to change the web url to another url as

>sed 's/http:\/\//www/' file.txt

In this case the url consists the delimiter character which we used. In that case you have to escape the slash with backslash character, otherwise the substitution won't work.  
  
Using too many backslashes makes the sed command look awkward. In this case we can change the delimiter to another character as shown in the below example.

>sed 's\_http://\_www\_' file.txt

>sed 's|http://|www|' file.txt

**6.** Using & as the matched string  
  
There might be cases where you want to search for the pattern and replace that pattern by adding some extra characters to it. In such cases & comes in handy. The & represents the matched string.

>sed 's/unix/{&}/' file.txt

{unix} is great os. unix is opensource. unix is free os.

learn operating system.

{unix}linux which one you choose.

>sed 's/unix/{&&}/' file.txt

{unixunix} is great os. unix is opensource. unix is free os.

learn operating system.

{unixunix}linux which one you choose.

**7.** Using \1,\2 and so on to \9  
  
The first pair of parenthesis specified in the pattern represents the \1, the second represents the \2 and so on. The \1,\2 can be used in the replacement string to make changes to the source string. As an example, if you want to replace the word "unix" in a line with twice as the word like "unixunix" use the sed command as below.

>sed 's/\(unix\)/\1\1/' file.txt

unixunix is great os. unix is opensource. unix is free os.

learn operating system.

unixunixlinux which one you choose.

The parenthesis needs to be escaped with the backslash character. Another example is if you want to switch the words "unixlinux" as "linuxunix", the sed command is

>sed 's/\(unix\)\(linux\)/\2\1/' file.txt

unix is great os. unix is opensource. unix is free os.

learn operating system.

linuxunix which one you choose.

Another example is switching the first three characters in a line

>sed 's/^\(.\)\(.\)\(.\)/\3\2\1/' file.txt

inux is great os. unix is opensource. unix is free os.

aelrn operating system.

inuxlinux which one you choose.

**8.** Duplicating the replaced line with /p flag  
  
The /p print flag prints the replaced line twice on the terminal. If a line does not have the search pattern and is not replaced, then the /p prints that line only once.

>sed 's/unix/linux/p' file.txt

linux is great os. unix is opensource. unix is free os.

linux is great os. unix is opensource. unix is free os.

learn operating system.

linuxlinux which one you choose.

linuxlinux which one you choose.

**9.** Printing only the replaced lines  
  
Use the -n option along with the /p print flag to display only the replaced lines. Here the -n option suppresses the duplicate rows generated by the /p flag and prints the replaced lines only one time.

>sed -n 's/unix/linux/p' file.txt

linux is great os. unix is opensource. unix is free os.

linuxlinux which one you choose.

If you use -n alone without /p, then the sed does not print anything.  
  
**10.** Running multiple sed commands.  
  
You can run multiple sed commands by piping the output of one sed command as input to another sed command.

>sed 's/unix/linux/' file.txt| sed 's/os/system/'

linux is great system. unix is opensource. unix is free os.

learn operating system.

linuxlinux which one you chosysteme.

Sed provides -e option to run multiple sed commands in a single sed command. The above output can be achieved in a single sed command as shown below.

>sed -e 's/unix/linux/' -e 's/os/system/' file.txt

linux is great system. unix is opensource. unix is free os.

learn operating system.

linuxlinux which one you chosysteme.

**11.** Replacing string on a specific line number.  
  
You can restrict the sed command to replace the string on a specific line number. An example is

>sed '3 s/unix/linux/' file.txt

unix is great os. unix is opensource. unix is free os.

learn operating system.

linuxlinux which one you choose.

The above sed command replaces the string only on the third line.  
  
**12.** Replacing string on a range of lines.  
  
You can specify a range of line numbers to the sed command for replacing a string.

>sed '1,3 s/unix/linux/' file.txt

linux is great os. unix is opensource. unix is free os.

learn operating system.

linuxlinux which one you choose.

Here the sed command replaces the lines with range from 1 to 3. Another example is

>sed '2,$ s/unix/linux/' file.txt

linux is great os. unix is opensource. unix is free os.

learn operating system.

linuxlinux which one you choose.

Here $ indicates the last line in the file. So the sed command replaces the text from second line to last line in the file.  
  
**13.** Replace on a lines which matches a pattern.  
  
You can specify a pattern to the sed command to match in a line. If the pattern match occurs, then only the sed command looks for the string to be replaced and if it finds, then the sed command replaces the string.

>sed '/linux/ s/unix/centos/' file.txt

unix is great os. unix is opensource. unix is free os.

learn operating system.

centoslinux which one you choose.

Here the sed command first looks for the lines which has the pattern "linux" and then replaces the word "unix" with "centos".  
  
**14.** Deleting lines.  
  
You can delete the lines a file by specifying the line number or a range or numbers.

>sed '2 d' file.txt

>sed '5,$ d' file.txt

**15.** Duplicating lines  
  
You can make the sed command to print each line of a file two times.

>sed 'p' file.txt

**16.** Sed as grep command  
  
You can make sed command to work as similar to grep command.

>grep 'unix' file.txt

>sed -n '/unix/ p' file.txt

Here the sed command looks for the pattern "unix" in each line of a file and prints those lines that has the pattern.  
  
You can also make the sed command to work as grep -v, just by using the reversing the sed with NOT (!).

>grep -v 'unix' file.txt

>sed -n '/unix/ !p' file.txt

The ! here inverts the pattern match.  
  
**17.** Add a line after a match.  
  
The sed command can add a new line after a pattern match is found. The "a" command to sed tells it to add a new line after a match is found.

>sed '/unix/ a "Add a new line"' file.txt

unix is great os. unix is opensource. unix is free os.

"Add a new line"

learn operating system.

unixlinux which one you choose.

"Add a new line"

**18.** Add a line before a match  
  
The sed command can add a new line before a pattern match is found. The "i" command to sed tells it to add a new line before a match is found.

>sed '/unix/ i "Add a new line"' file.txt

"Add a new line"

unix is great os. unix is opensource. unix is free os.

learn operating system.

"Add a new line"

unixlinux which one you choose.

**19.** Change a line  
  
The sed command can be used to replace an entire line with a new line. The "c" command to sed tells it to change the line.

>sed '/unix/ c "Change line"' file.txt

"Change line"

learn operating system.

"Change line"

**20.** Transform like tr command  
  
The sed command can be used to convert the lower case letters to upper case letters by using the transform "y" option.

>sed 'y/ul/UL/' file.txt

Unix is great os. Unix is opensoUrce. Unix is free os.

Learn operating system.

UnixLinUx which one yoU choose.

Here the sed command transforms the alphabets "ul" into their uppercase format "UL"

**sed examples**

| **Learning Linux sed command with examples** | |
| --- | --- |
| **Linux command syntax** | **Linux command description** |
|  | |
| sed 's/Nick/John/g' report.txt | Replace every occurrence of Nick with John in report.txt |
| sed 's/Nick|nick/John/g' report.txt | Replace every occurrence of Nick or nick with John. |
| sed 's/^/ /' file.txt >file\_new.txt | Add 8 spaces to the left of a text for pretty printing. |
| sed -n '/Of course/,/attention you \ pay/p' myfile | Display only one paragraph, starting with "Of course"  and ending in "attention you pay" |
| sed -n 12,18p file.txt | Show only lines 12-18 of file.txt |
| sed 12,18d file.txt | Show all of file.txt *except* for lines from 12 to 18 |
| sed G file.txt | Double-space file.txt |
| sed -f script.sed file.txt | Write all commands in script.sed and execute them |
| sed '5!s/ham/cheese/' file.txt | Replace ham with cheese in file.txt except in the 5th line |
| sed '$d' file.txt | Delete the last line |
| sed '/[0-9]\{3\}/p' file.txt | Print only lines with three consecutive digits |
| sed '/boom/!s/aaa/bb/' file.txt | Unless boom is found replace aaa with bb |
| sed '17,/disk/d' file.txt | Delete all lines from line 17 to 'disk' |
| echo ONE TWO | sed "s/one/unos/I" | Replaces one with unos in a case-insensitive manner,  so it will print "unos TWO" |
| sed 'G;G' file.txt | Triple-space a file |
| sed 's/.$//' file.txt | A way to replace dos2unix :) |
| sed 's/^[ ^t]\*//' file.txt | Delete all spaces in front of every line of file.txt |
| sed 's/[ ^t]\*$//' file.txt | Delete all spaces at the end of every line of file.txt |
| sed 's/^[ ^t]\*//;s/[ ^]\*$//' file.txt | Delete all spaces in front and at the end of every line  of file.txt |
| sed 's/foo/bar/' file.txt | Replace foo with bar only for the first instance in a line. |
| sed 's/foo/bar/4' file.txt | Replace foo with bar only for the 4th instance in a line. |
| sed 's/foo/bar/g' file.txt | Replace foo with bar for all instances in a line. |
| sed '/baz/s/foo/bar/g' file.txt | Only if line contains baz, substitute foo with bar |
| sed '/./,/^$/!d' file.txt | Delete all consecutive blank lines except for EOF |
| sed '/^$/N;/\n$/D' file.txt | Delete all consecutive blank lines, but allows  only top blank line |
| sed '/./,$!d' file.txt | Delete all leading blank lines |
| sed -e :a -e '/^\n\*$/{$d;N;};/\n$/ba' \ file.txt | Delete all trailing blank lines |
| sed -e :a -e '/\$/N; s/\\n//; ta' \ file.txt | If a file ends in a backslash, join it with the next (useful  for shell scripts) |
| sed '/regex/,+5/expr/' | Match regex plus the next 5 lines |
| sed '1~3d' file.txt | Delete every third line, starting with the first |
| sed -n '2~5p' file.txt | Print every 5th line starting with the second |
| sed 's/[Nn]ick/John/g' report.txt | Another way to write some example above.  Can you guess which one? |
| sed -n '/RE/{p;q;}' file.txt | Print only the first match of  RE (regular expression) |
| sed '0,/RE/{//d;}' file.txt | Delete only the first match |
| sed '0,/RE/s//to\_that/' file.txt | Change only the first match |
| sed 's/^[^,]\*,/9999,/' file.csv | Change first field to 9999 in a CSV file |
| s/^ \*\(.\*[^ ]\) \*$/||/; s/" \*, \*/"|/g; : loop s/| \*\([^",|][^,|]\*\) \*, \*/||/g; s/| \*, \*/||/g; t loop s/ \*|/|/g; s/| \*/|/g; s/^|\(.\*\)|$//; | sed script to convert CSV file to bar-separated  (works only on some types of CSV,  with embedded "s and commas) |
| sed ':a;s/\(^\|[^0-9.]\)\([0-9]\+\)\ ([0-9]\{3\}\)/,/g;ta' file.txt | Change numbers from file.txt from 1234.56 form to 1.234.56 |
| sed -r "s/\<(reg|exp)[a-z]+/\U&/g" | Convert any word starting with reg or exp to uppercase |
| sed '1,20 s/Johnson/White/g' file.txt | Do replacement of Johnson with White only on  lines between 1 and 20 |
| sed '1,20 !s/Johnson/White/g' file.txt | The above reversed (match all except lines 1-20) |
| sed '/from/,/until/ { s/\<red\>/magenta/g; \ s/\<blue\>/cyan/g; }' file.txt | Replace only between "from" and "until" |
| sed '/ENDNOTES:/,$ { s/Schaff/Herzog/g; \ s/Kraft/Ebbing/g; }' file.txt | Replace only from the word "ENDNOTES:" until EOF |
| sed '/./{H;$!d;};x;/regex/!d' file.txt | Print paragraphs only if they contain regex |
| sed -e '/./{H;$!d;}' -e 'x;/RE1/!d;\ /RE2/!d;/RE3/!d' file.txt | Print paragraphs only if they contain RE1,  RE2 *and*RE3 |
| sed ':a; /\$/N; s/\\n//; ta' file.txt | Join two lines in the first ends in a backslash |
| sed 's/14"/fourteen inches/g' file.txt | This is how you can use double quotes |
| sed 's/\/some\/UNIX\/path/\/a\/new\ /path/g' file.txt | Working with Unix paths |
| sed 's/[a-g]//g' file.txt | Remove all characters from a to g from file.txt |
| sed 's/\(.\*\)foo/bar/' file.txt | Replace only the last match of foo with bar |
| sed '1!G;h;$!d' | A tac replacement |
| sed '/\n/!G;s/\(.\)\(.\*\n\)/&\ /;//D;s/.//' | A rev replacement |
| sed 10q file.txt | A head replacement |
| sed -e :a -e '$q;N;11,$D;ba' \ file.txt | A tail replacement |
| sed '$!N; /^\(.\*\)\n$/!P; D' \ file.txt | A uniq replacement |
| sed '$!N; s/^\(.\*\)\n$//;\  t; D' file.txt | The opposite (or uniq -d equivalent) |
| sed '$!N;$!D' file.txt | Equivalent to tail -n 2 |
| sed -n '$p' file.txt | ... tail -n 1 (or tail -1) |
| sed '/regexp/!d' file.txt | grep equivalent |
| sed -n '/regexp/{g;1!p;};h' file.txt | Print the line before the one matching regexp, but  not the one containing the regexp |
| sed -n '/regexp/{n;p;}' file.txt | Print the line after the one matching the regexp, but  not the one containing the regexp |
| sed '/pattern/d' file.txt | Delete lines matching pattern |
| sed '/./!d' file.txt | Delete all blank lines from a file |
| sed '/^$/N;/\n$/N;//D' file.txt | Delete all consecutive blank lines  except for the first two |
| sed -n '/^$/{p;h;};/./{x;/./p;}'\  file.txt | Delete the last line of each paragraph |
| sed 's/.\x08//g' file | Remove nroff overstrikes |
| sed '/^$/q' | Get mail header |
| sed '1,/^$/d' | Get mail body |
| sed '/^Subject: \*/!d; s///;q' | Get mail subject |
| sed 's/^/> /' | Quote mail message by inserting a  "> " in front of every line |
| sed 's/^> //' | The opposite (unquote mail message) |
| sed -e :a -e 's/<[^>]\*>//g;/</N;//ba' | Remove HTML tags |
| sed '/./{H;d;};x;s/\n/={NL}=/g'\  file.txt | sort \ | sed '1s/={NL}=//;s/={NL}=/\n/g' | Sort paragraphs of file.txt alphabetically |
| sed 's@/usr/bin@&/local@g' path.txt | Replace /usr/bin with /usr/bin/local in path.txt |
| sed 's@^.\*$@<<<&>>>@g' path.txt | Try it and see :) |
| sed 's/\(\/[^:]\*\).\*//g' path.txt | Provided path.txt contains $PATH, this will  echo only the first path on each line |
| sed 's/\([^:]\*\).\*//' /etc/passwd | awk replacement - displays only the users  from the passwd file |
| echo "Welcome To The Geek Stuff" | sed \ 's/\(\b[A-Z]\)/\(\)/g' (W)elcome (T)o (T)he (G)eek (S)tuff | Self-explanatory |
| sed -e '/^$/,/^END/s/hills/\ mountains/g' file.txt | Swap 'hills' for 'mountains', but only on blocks  of text beginning  with a blank line, and ending with a line beginning  with the three characters 'END', inclusive |
| sed -e '/^#/d' /etc/services | more | View the services file without the commented lines |
| sed '$s@\([^:]\*\):\([^:]\*\):\([^:]\*\ \)@::@g' path.txt | Reverse order of items in the last line of path.txt |
| sed -n -e '/regexp/{=;x;1!p;g;$!N;p;D;}'\  -e h file.txt | Print 1 line of context before and after the line matching,  with a line number where the matching occurs |
| sed '/regex/{x;p;x;}' file.txt | Insert a new line above every line matching regex |
| sed '/AAA/!d; /BBB/!d; /CCC/!d' file.txt | Match AAA, BBB and CCC in any order |
| sed '/AAA.\*BBB.\*CCC/!d' file.txt | Match AAA, BBB and CCC in that order |
| sed -n '/^.\{65\}/p' file.txt | Print lines 65 chars long or more |
| sed -n '/^.\{65\}/!p' file.txt | Print lines 65 chars long or less |
| sed '/regex/G' file.txt | Insert blank line below every line |
| sed '/regex/{x;p;x;G;}' file.txt | Insert blank line above and below |
| sed = file.txt | sed 'N;s/\n/\t/' | Number lines in file.txt |
| sed -e :a -e 's/^.\{1,78\}$/\  &/;ta' file.txt | Align text flush right |
| sed -e :a -e 's/^.\{1,77\}$/ &/;ta' -e \ 's/\( \*\)//' file.txt | Align text center |