The Ninth International Conference on Advances in Databases, Knowledge, and Data Applications, Mai 21 - 25, 2017 - Barcelona, Spain

Tutorial: Data Manipulation and Data Transformation using the Shell
Andreas Schmidt, Steffen Scholz

Command Overview

Command	Description	Popular options
cat	concatenate files and print on the standard output	**-n**: number all output lines
head	output the first part of files	**-n***<num>: print the first <num> lines* **-n** -*<num>*: print all but the last *<num>* lines
tail		**-n***<num>: print the last <num> lines* **-n** + *<num>: print, starting from line <num>*
wc	print newline, word, and byte counts for each file	**-c**: print byte counts **-m**: print the character counts **-w**: print the word counts **-l**: print the newline counts
grep	print lines matching a pattern	**-E**: support extended regexp **-i**: ignore case **-v**: invert match **-o**: print only the matched part (output: one line per match) **-f**: obtain patterns from file **-l**: files with match **-L**: files without match **-c**: suppress normal output, instead count matching lines. **-m***<num>*: Stop reading a file after *<num>* matching lines
seq	print a sequence of numbers	**-s**: separator (default: \n)
split	split a file into pieces	
cut	remove sections from each line of files	**-d***<delim>*: Use *<delim>* instead oft ab as field separator **-f***<field-list>:* select only fields in *<field-list>* **--output-delimiter**=*<delim>* : use *<delim>* as output delimiter
paste	merge lines of files	**-d**<char>: use *<char>* as output delimiter
tr	Translate, squeeze, and/or delete characters	**-c**: use the complement of set1 **-d**: delete the characters in set1 **-s**: replace each sequence of a repeated character with a single character
sort	sort lines of text files	**-n**: numeric sort **-r**: reverse sort **-R**: random shuffle **-c**: check, if sorted, do not sort **-t**: field separator **-k***<keydef>* : sort according to keydef *<keydef>*: F[.C][OPTS][,F[.C][OPTS]] **-u**: output only the first of equal lines
join	join lines of two files on a common field Remark: files must be sorted on join column	**-t***<char>* : Use *<char>* as input, output separator **-1***<field>* : join on this FIELD of file 1 **-2***<field>* : join on this FIELD of file 2 **-o***<format>* : obey *<format>* while constructing output line

		<format> : filenum.field[[filenam.field][…]]
Command	**Description**	**Popular options**
comm	compare two sorted files line by line	**-1:** suppress column 1 (lines unique to FILE1) **-2:** suppress column 1 (lines unique to FILE2) **-3:** suppress column 1 (lines unique to FILE3) **--total** : output a summary
uniq	report or omit repeated lines	**-c :** prefix lines by the number of occurrences **-d** : only print duplicate lines, one for each group **-i**: ignore case **-u** : only print unique lines

sed – stream editor for filtering and transfering text

Command	**Description**	**Popular options**
sed	stream editor for filtering and transforming text	**-n :** suppress automatic printing of pattern space **-f** <script-file> : scripts with commands tob e executed **-i** : edit inplace **-E, -r** : support extended regexp
		<address> <start-address>,<end-address> <start-address>, + <number-of-lines> <address> can be: • line-number (i.e. 1,5,7, …) • $ (represent last line of file) • regular-expression

Sed commands	**Description**
a *<text>*	append text
i *<text>*	insert text
c *<text>*	replace the selected lines with text
p	print
d	delete pattern space
s/regexp/replacement/	regexp-replace

sed-Examples:

- sed -i '/Aachen/ d' city.csv # delete line(s) containing Aachen (inplace)
- sed '2i Karlsruhe,D,"Baden Wuerttemberg",301452,49.0,6.8' city.csv
  # insert ‚Karlsruhe …' at line 2
- sed -Ei '/<script>/,/<\/script>/d' jaccard.html # remove all script-sections
- sed -i 's/\bNULL\b/\\N/g' city.csv # replace NULL ->\n
- sed -n '5,10p;23p;56,71p' city.csv # print lines 5-10, 23, 56-71