W6

Irlanda Ayon-Moreno

4/15/2018

Inspecting Data in the Terminal

Command	Description	R alternative
`wc nba2017-players.csv`	count lines, words, and bytes	`object.size()`, `nrow()`
`wc -l nba2017-players.csv`	count number of lines	`nrow()`
`head nba2017-players.csv`	inspect first 10 rows	`head()`
`tail nba2017-players.csv`	inspect last 10 rows	`tail()`
`less nba2017-players.csv`	see contents with a paginator	`View()`

Manipulation of Data

Command	Description	R alternative
`head -n 11 nba2017-players.csv > data10.csv`	redirection to new file	`sink()`
`cat data10.csv`	display contents on screen	
`cut -d "," -f 3 data10.csv`	select (third) column	
`sort positions10.txt`	sort the lines of a stream of data	
`cut -d "," -f 3 data10.csv | tail +2`	excludes the 1st value (of the col)	

/ is a pipe operator. takes the output of a command and sends it as the input of another command

Filters

Extracting columns with `cut`

`cut` operates based either on character position within the column when using the `-c` flag, or on delimited fields when using the `-f` flag. By default, `cut` expects tabs as the delimiter. If a file separates fields with spaces or commas or any other delimiter, you need to use the option `-d` indicating the character used as field delimiter between quote marks.

Option	Description
`-f` 1,5	return columns 1 and 5, delimited by tabs.
`-f` 1-5	return columns 1 through 5, delimited by tabs.
`-d ","`	use commans as the delimiters.
`-c 2-7`	return characters 2 through 7 from the file.

Sorting lines with `sort`

Option	Description
`-n`	sort in numerical order rather than alphabetically.
`-r`	sort in reverse order, z to a or decreasing numbers.
`-f`	fold uppercase into lowercase (i.e. ignore case).
`-u`	return a unique representative of repeated items.
`-k 3`	sort lines based on column 3 (tab or space delimiters)
`-t ","`	use commas for delimiters.
`-b`	ignore leading blanks.
`-d`	sort in dictionary order.

Isolating unique lines with `uniq`

This command removes consecutive identical lines from a file, leaving one unique representative. More precisely, what `uniq` does is compare each line it reads with the previous line. If the lines are the same, `uniq` does not list the second line.

Option	Description
`-c`	adds a count of how many times each line occurred.
`-u`	lists only lines that are not repeated.
`-d`	lists only lines that are duplicated.
`-i`	ignore case when determining uniqueness
`-f 4`	ignore the first 4 fields (space delimiter)

To get a single representative of each unique line from the entire file, in most cases you would need to first sort the lines with the `sort` command to group matching lines together. Interestingly, `uniq` can be used with the flag `-c` to count the number of occurrences of a line. This gives a quick way, for example, to assess the frequencies of values in a given column.

Redirecting

Description	bash example
Send *standard output* to *file*	`cmd > file`
Send *standard error* to *file*	`cmd 2> file`
Take *standard input* from *file*	`cmd < file`
Send *standard output* to end of *file*	`cmd >> file`