How can I use PowerShell to help me see my text data?

Authored by

stats writer

June 30, 2024

PowerShell is a powerful command-line interface and scripting language that allows users to efficiently manipulate and manage their data. By utilizing its various commands and functions, users can easily view and analyze their text data in a systematic and organized manner. With the ability to create custom scripts and automate tasks, PowerShell can assist in extracting, filtering, and formatting text data to generate insights and make informed decisions. It also provides a user-friendly environment for working with large amounts of data, making it an essential tool for data analysis and visualization. Overall, using PowerShell can greatly enhance the efficiency and effectiveness of managing and viewing text data for various purposes.

FAQ:

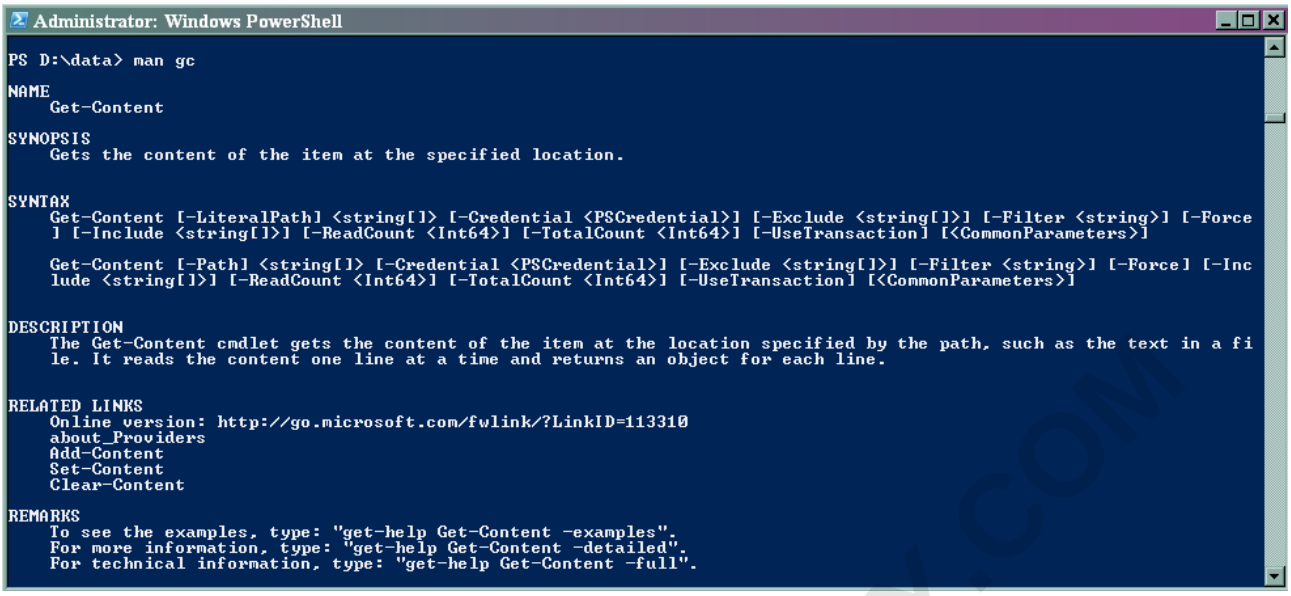
How can I use PowerShell to help me see my text data?

According to Wikipedia, Windows PowerShell is Microsoft's task automation framework, consisting of a command-line shell and associated scripting language built on top of, and integrated with the .NET Framework. PowerShell can be useful when you want to see a few lines of a very large text file. To access PowerShell, you can click on Start, Accessories, Windows PowerShell. This will open a DOS-like command window. As in Unix and DOS, you can issue commands from the prompt. If you do not have PowerShell installed on your computer, you can download it from this Microsoft website .

The commands in PowerShell are called "cmdlets" (pronounced "command-lets", or small commands), and they are case-sensitive. You can find a listing of the available commands here. We will illustrate a few cmdlets that can be useful when you want to see some of the contents of a large text file. The lines that start with a pound sign (#) are comments.

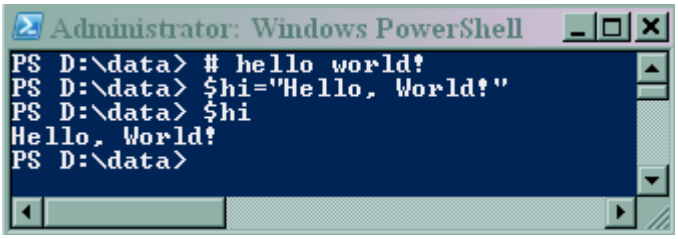
To get help, you can type the following cmdlets.

```
# getting help
#
http://technet.microsoft.com/en-us/library/dd347616.aspx
get-help Get-Content -examples
man gc
```



We will start with the familiar "Hello, World!" to show how variables can be created, and then using "pwd" and "dir" to get information regarding directories.

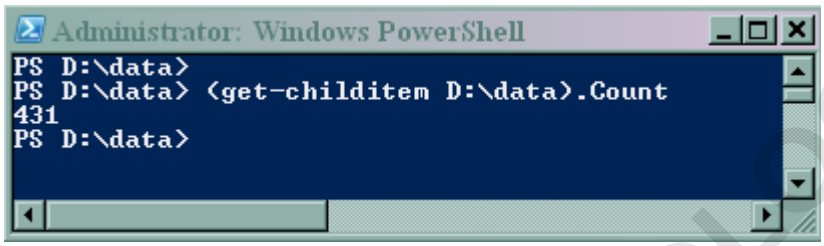
```
# hello world!
$hi="Hello, World!"
$hi
```



pwd

dir

 # counting the number of files in a folder
get-childitem D:data
get-childitem D:data -name
(get-childitem D:data).Count



To record the script (or series of cmdlets that you type), you can use
the following cmdlets.

start-transcript "D:datatranscript.txt"
#things in between will be recorded
stop-transcript

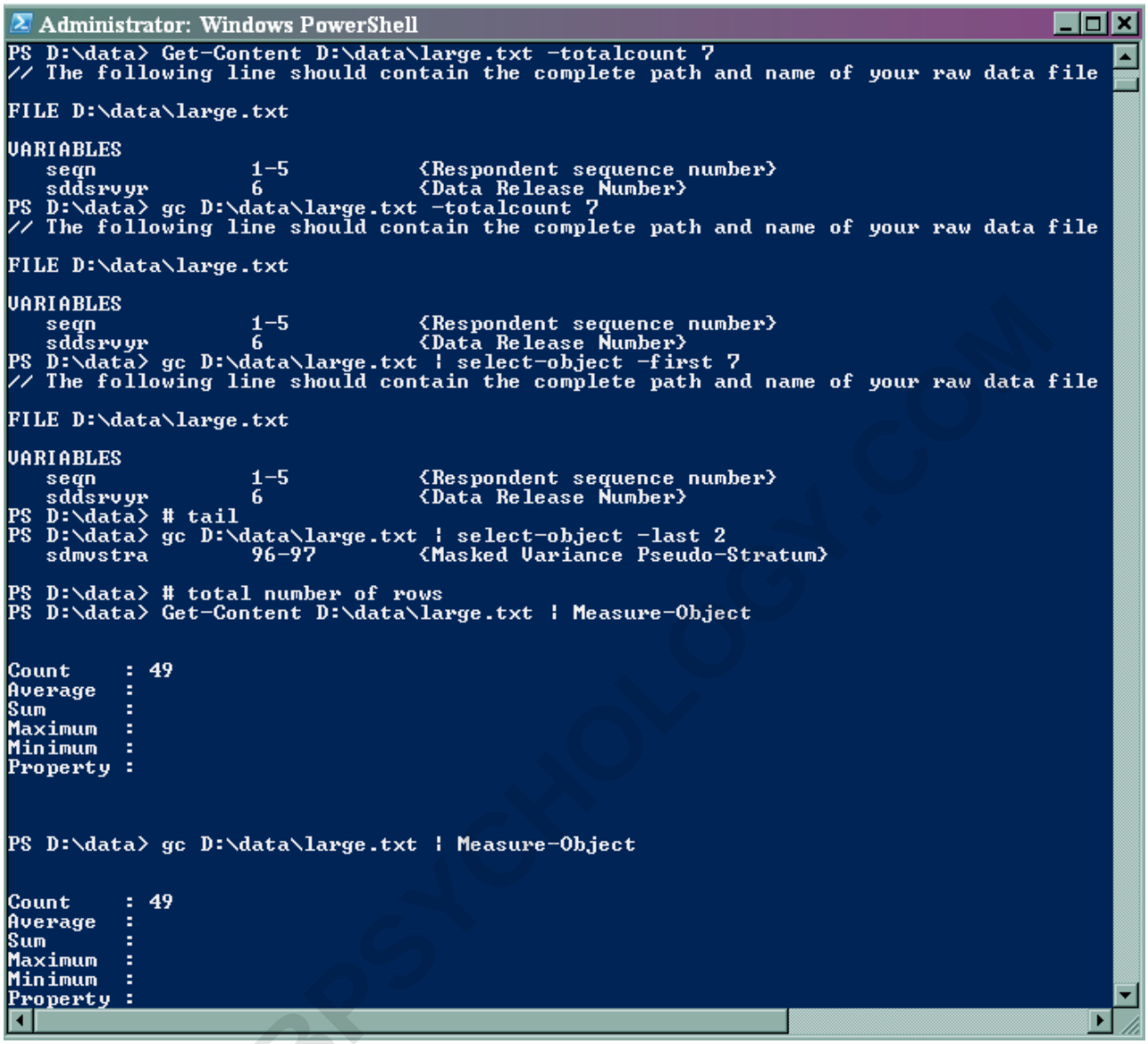
The "Get-Content" cmdlet is very useful for accessing text files.
In the first example, we access the first 7 rows of the text file, which we

call "large.txt". Next, we access the last 2 rows, and then we
get the total number of rows. Note that the "Get-Content" cmdlet can
be shortened to "gc".

```
# head
Get-Content D:datalarge.txt -totalcount 7
gc D:datalarge.txt -totalcount 7
gc D:datalarge.txt | select-object -first 7

# tail
gc D:datalarge.txt | select-object -last 2

# total number of rows
Get-Content D:datalarge.txt | Measure-Object
gc D:datalarge.txt | Measure-Object
```



You can access a specific column, and you can get date and time

information.

 # get a specific column

#

http://stackoverflow.com/questions/2503010/extracting-

columns-from-text-file-using-powershell

gc D:datalarge.txt | Foreach {($_ -split 's+', 8)}

gc D:datalarge.txt | Foreach {"$(($_ -split 's+', 8))"}

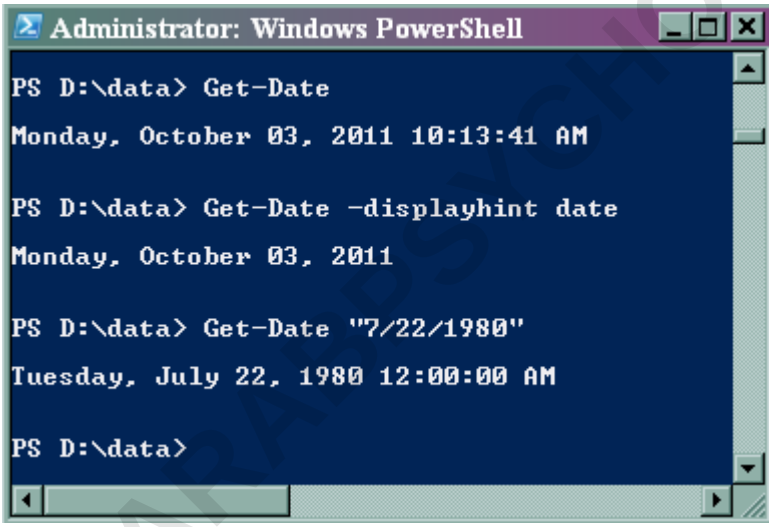
gc D:datalarge.txt | Foreach {"$(($_ -split 's+', 8))"} > col12.txt

 # date and time

Get-Date

Get-Date -displayhint date

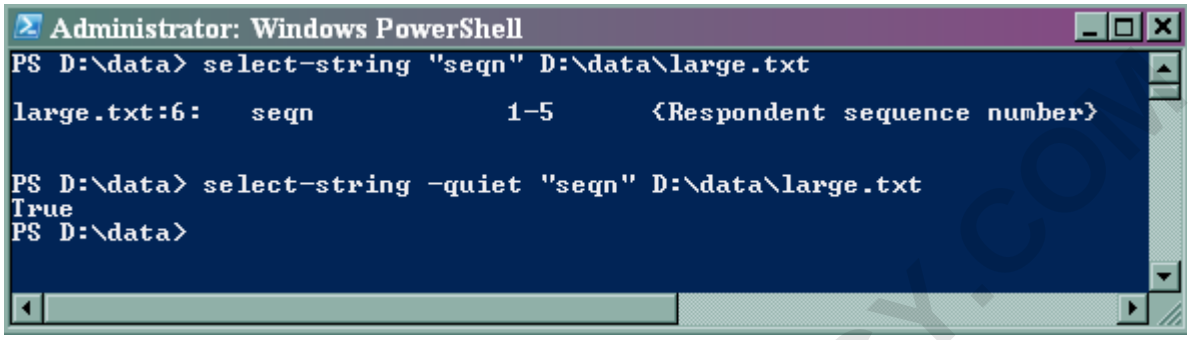
Get-Date "7/22/1980"



Below are a few examples of searching, subsetting and appending.

 # searching and subsetting

```
more D:datalarge.txt
select-string "seqn" D:datalarge.txt
select-string -quiet "seqn" D:datalarge.txt
```




```
# appending
add-content -value "addthis" simple.txt
add-content -value (get-content D:datalarge.txt)
simple.txt
```