**格式：**

　　sed [OPTION]... {script-only-if-no-other-script} [input-file]...

　　sed [OPTION]... ‘地址定界+[高级]编辑命令’  file

**常用选择：**

　　i　　　　：修改内容

　　i.bak　　：先备份，然后再修改内容

　　r　　　　：支持正则表达式

　　e　　　　：支持多点编辑

　　n　　　　：不输出模式空间内容到屏幕，即不自动打印

**'地址定界+[高级]编辑命令'**

　　地址定界

　　　　 1.不给地址：

　　　　　　对全文进行处理

　　　　 2.单地址：

　　　　　　#：指定的行 ， $：最后一行

　　　　　　/pattern/：被此处模式所能够匹配到的每一行

　　　　 3.地址范围

　　　　　　#,#

　　　　　　#,+#

　　　　　　/pat1/,/pat2/

　　　　　　#,/pat1/

　　　　 4. ~：步进

　　　　　　1~2 奇数行

　　　　　　2~2 偶数行

　　编辑命令

　　　　d：删除模式空间匹配的行，并立即启用下一轮循环

　　　　p：打印当前模式空间内容，追加到默认输出之后

　　　　a[\]text：在指定行后面追加文本，支持使用\n实现多行追加

　　　　i[\]text：在行前插入文本

　　　　c[\]text：替换行为单行或多行文件

　　　　w /path/somefile：保存模式匹配的行至指定文件

　　　　r /path/somefile：读取指定文件的文本至模式空间中匹配到的行后

　　　　=：为模式空间中的行打印行号 !：模式空间中匹配行取反处理

　　高级编辑命令 P：打印模式空间开端至\n内容，并追加到默认输出之前

　　　　h：把模式空间中的内容覆盖至保持空间中 H：把模式空间中的内容追加至保持空间中

　　　　g：从保持空间取出数据覆盖至模式空间 G：从保持空间取出数据追加至模式空间

　　　　x：把模式空间中的内容与保持空间中的内容进行交换

　　　　n：读取匹配到的行的下一行覆盖至模式空间

　　　　N：读取匹配到的行的下一行追加至模式空间

　　　　d：删除模式空间中的行

　　　　D：如果模式空间包含换行符，则删除知道第一个换行符的模式空间中的文本， 并不会读取新的输入行，而使用合成的模式空间重新启动循环。如果模式空间不包含换行符，则会像发 出d命令那样启动正常的新循环

**基本正则模式匹配**

**匹配字符：**

　　　　　　. ：任意一个字符。

　　　　　　[abc] ：表示匹配一个字符

　　　　　　[a-Z] ：表示匹配一个字符

　　　　　　[^123] ：匹配一个字符，这个字符是除了1、2、3以外的所有字符。

　　　　　　对于一些常用的字符集，系统做了定义：

　　　　　　[a-Z] 等价于 [[:alpha:]]

　　　　　　[0-9] 等价于 [[:digit:]]

　　　　　　[a-Z0-9] 等价于 [[:alnum:]]

　　　　　　tab,space 等空白字符 [[:space:]]

　　　　　　[A-Z] 等价于 [[:upper:]]

　　　　　　[a-z] 等价于 [[:lower:]]

　　　　　　标点符号 [[:punct:]]

**匹配次数：**

　　　　　　\{m,n\} ：匹配其前面出现的字符至少m次，至多n次。  
　　　　　　\? ：匹配其前面出现的内容0次或1次，等价于\{0,1\}。  
　　　　　　\* ：匹配其前面出现的内容任意次，等价于\{0,\}，所以 ".\*" 表述任意字符任意次，即无论什么内容全部匹配。

**位置锚定：**

　　　　　　^ ：锚定行首

　　　　　　$ ：锚定行尾。技巧："^$"用于匹配空白行。

　　　　　　\b或\<：锚定单词的词首。如"\blike"不会匹配alike，但是会匹配liker

　　　　　　\b或\>：锚定单词的词尾。如"\blike\b"不会匹配alike和liker，只会匹配like

　　　　　　\B ：与\b作用相反。

**分组及引用：**

　　　　　　\(string\) ：将string作为一个整体方便后面引用

　　　　　　　　\1 ：引用第1个左括号及其对应的右括号所匹配的内容。

　　　　　　　　\2 ：引用第2个左括号及其对应的右括号所匹配的内容。

　　　　　　　　\n ：引用第n个左括号及其对应的右括号所匹配的内容。

**扩展正则模式匹配**

**匹配字符：**

　　　　　　跟基本正则用法一样

**匹配次数：**

　　　　　　{m,n} ：匹配其前面出现的字符至少m次，至多n次。  
　　　　　　? ：匹配其前面出现的内容0次或1次，等价于\{0,1\}。  
　　　　　　\* ：匹配其前面出现的内容任意次，等价于\{0,\}，所以 ".\*" 表述任意字符任意次，即无论什么内容全部匹配。

　　　　　　+：匹配前面出现的内容1次或1次以上

**位置锚定：**

　　　　　　跟基本正则用法一样

**分组及引用：**

　　　　　　(string) ：将string作为一个整体方便后面引用

　　　　　　　　\1 ：引用第1个左括号及其对应的右括号所匹配的内容。

　　　　　　　　\2 ：引用第2个左括号及其对应的右括号所匹配的内容.

　　　　　　　　\n ：引用第n个左括号及其对应的右括号所匹配的内容。

**扩展增用法：**

　　　　　　str | num：匹配 str 或 num 内容

**例题**  
1.删除/etc/grub2.cfg文件中所有以空白开头的行行首的空白字符

|  |  |
| --- | --- |
| 1 | [root@cloud ~]# sed -r "/^[[:space:]]\*$/d" /etc/grub2.cfg |

2.删除/etc/fstab文件中不以#开头，后面至少跟一个空白字符的行的行首的和空白字符

[root@cloud ~]# sed -r "/^# \*|^$/d" /etc/fstab

3.在/etc/fstab文件中不以#开头的行的行首增加#号

[root@cloud ~]# sed -r -n "/^[^#]/s/(.\*)/#\1/p" /etc/fstab

4.处理/etc/fstab路径使用sed命令取出其目录和基名

[root@cloud ~]# echo /etc/fstab | sed -r "s@(.\*/)(.\*$)@\2@"

5.利用sed取出ifconfig命令中本机的IPv4地址

[root@cloud ~]# ifconfig eth0 | sed -r -n "s/.\*inet (.\*) net.\*/\1/p"

6.统计centos安装光盘中Package目录下的所有rpm文件的以. 分割倒数第二个字段的重复次数

[root@cloud ~]# mount /dev/cdrom /opt/dvd

[root@cloud ~]# ls /opt/dvd/Packages/ | sed -r "s/.rpm$//" | sed -r "s/.\*\.(.\*)/\1/" | sort | uniq -c

[root@cloud ~]# ls /opt/dvd/Packages/ | sed -r "s/.\*\.(.\*).rpm/\1/" | sort | uniq -c

7.将文本文件的n和n+1行合并为一行，n为奇数行

[root@cloud ~]# seq 10|xargs -n2

[root@cloud ~]# seq 10|sed 'N;s/\n/ /'

Linux sed 命令是利用脚本来处理文本文件。

sed 可依照脚本的指令来处理、编辑文本文件。

Sed 主要用来自动编辑一个或多个文件、简化对文件的反复操作、编写转换程序等。

**语法**

sed [-hnV][-e<script>][-f<script文件>][文本文件]

**参数说明**：

* -e<script>或--expression=<script> 以选项中指定的script来处理输入的文本文件。
* -f<script文件>或--file=<script文件> 以选项中指定的script文件来处理输入的文本文件。
* -h或--help 显示帮助。
* -n或--quiet或--silent 仅显示script处理后的结果。
* -V或--version 显示版本信息。

**动作说明**：

* a ：新增， a 的后面可以接字串，而这些字串会在新的一行出现(目前的下一行)～
* c ：取代， c 的后面可以接字串，这些字串可以取代 n1,n2 之间的行！
* d ：删除，因为是删除啊，所以 d 后面通常不接任何咚咚；
* i ：插入， i 的后面可以接字串，而这些字串会在新的一行出现(目前的上一行)；
* p ：打印，亦即将某个选择的数据印出。通常 p 会与参数 sed -n 一起运行～
* s ：取代，可以直接进行取代的工作哩！通常这个 s 的动作可以搭配正规表示法！例如 1,20s/old/new/g 就是啦！

**实例**

在testfile文件的第四行后添加一行，并将结果输出到标准输出，在命令行提示符下输入如下命令：

sed -e 4a\newLine testfile

首先查看testfile中的内容如下：

$ cat testfile #查看testfile 中的内容

HELLO LINUX!

Linux is a free unix-type opterating system.

This is a linux testfile!

Linux test

使用sed命令后，输出结果如下：

$ sed -e 4a\newline testfile #使用sed 在第四行后添加新字符串

HELLO LINUX! #testfile文件原有的内容

Linux is a free unix-type opterating system.

This is a linux testfile!

Linux test

newline

**以行为单位的新增/删除**

将 /etc/passwd 的内容列出并且列印行号，同时，请将第 2~5 行删除！

[root@www ~]# nl /etc/passwd | sed '2,5d'

1 root:x:0:0:root:/root:/bin/bash

6 sync:x:5:0:sync:/sbin:/bin/sync

7 shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown

.....(后面省略).....

sed 的动作为 '2,5d' ，那个 d 就是删除！因为 2-5 行给他删除了，所以显示的数据就没有 2-5 行罗～ 另外，注意一下，原本应该是要下达 sed -e 才对，没有 -e 也行啦！同时也要注意的是， sed 后面接的动作，请务必以 '' 两个单引号括住喔！

只要删除第 2 行

nl /etc/passwd | sed '2d'

要删除第 3 到最后一行

nl /etc/passwd | sed '3,$d'

在第二行后(亦即是加在第三行)加上『drink tea?』字样！

[root@www ~]# nl /etc/passwd | sed '2a drink tea'

1 root:x:0:0:root:/root:/bin/bash

2 bin:x:1:1:bin:/bin:/sbin/nologin

drink tea

3 daemon:x:2:2:daemon:/sbin:/sbin/nologin

.....(后面省略).....

那如果是要在第二行前

nl /etc/passwd | sed '2i drink tea'

如果是要增加两行以上，在第二行后面加入两行字，例如 **Drink tea or .....** 与 **drink beer?**

[root@www ~]# nl /etc/passwd | sed '2a Drink tea or ......\

> drink beer ?'

1 root:x:0:0:root:/root:/bin/bash

2 bin:x:1:1:bin:/bin:/sbin/nologin

Drink tea or ......

drink beer ?

3 daemon:x:2:2:daemon:/sbin:/sbin/nologin

.....(后面省略).....

每一行之间都必须要以反斜杠『 \ 』来进行新行的添加喔！所以，上面的例子中，我们可以发现在第一行的最后面就有 \ 存在。

**以行为单位的替换与显示**

将第2-5行的内容取代成为『No 2-5 number』呢？

[root@www ~]# nl /etc/passwd | sed '2,5c No 2-5 number'

1 root:x:0:0:root:/root:/bin/bash

No 2-5 number

6 sync:x:5:0:sync:/sbin:/bin/sync

.....(后面省略).....

透过这个方法我们就能够将数据整行取代了！

仅列出 /etc/passwd 文件内的第 5-7 行

[root@www ~]# nl /etc/passwd | sed -n '5,7p'

5 lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin

6 sync:x:5:0:sync:/sbin:/bin/sync

7 shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown

可以透过这个 sed 的以行为单位的显示功能， 就能够将某一个文件内的某些行号选择出来显示。

**数据的搜寻并显示**

搜索 /etc/passwd有root关键字的行

nl /etc/passwd | sed '/root/p'

1 root:x:0:0:root:/root:/bin/bash

1 root:x:0:0:root:/root:/bin/bash

2 daemon:x:1:1:daemon:/usr/sbin:/bin/sh

3 bin:x:2:2:bin:/bin:/bin/sh

4 sys:x:3:3:sys:/dev:/bin/sh

5 sync:x:4:65534:sync:/bin:/bin/sync

....下面忽略

如果root找到，除了输出所有行，还会输出匹配行。

使用-n的时候将只打印包含模板的行。

nl /etc/passwd | sed -n '/root/p'

1 root:x:0:0:root:/root:/bin/bash

**数据的搜寻并删除**

删除/etc/passwd所有包含root的行，其他行输出

nl /etc/passwd | sed '/root/d'

2 daemon:x:1:1:daemon:/usr/sbin:/bin/sh

3 bin:x:2:2:bin:/bin:/bin/sh

....下面忽略

#第一行的匹配root已经删除了

**数据的搜寻并执行命令**

搜索/etc/passwd,找到root对应的行，执行后面花括号中的一组命令，每个命令之间用分号分隔，这里把bash替换为blueshell，再输出这行：

nl /etc/passwd | sed -n '/root/{s/bash/blueshell/;p;q}'

1 root:x:0:0:root:/root:/bin/blueshell

最后的q是退出。

**数据的搜寻并替换**

除了整行的处理模式之外， sed 还可以用行为单位进行部分数据的搜寻并取代。基本上 sed 的搜寻与替代的与 vi 相当的类似！他有点像这样：

sed 's/要被取代的字串/新的字串/g'

先观察原始信息，利用 /sbin/ifconfig 查询 IP

[root@www ~]# /sbin/ifconfig eth0

eth0 Link encap:Ethernet HWaddr 00:90:CC:A6:34:84

inet addr:192.168.1.100 Bcast:192.168.1.255 Mask:255.255.255.0

inet6 addr: fe80::290:ccff:fea6:3484/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

.....(以下省略).....

本机的ip是192.168.1.100。

将 IP 前面的部分予以删除

[root@www ~]# /sbin/ifconfig eth0 | grep 'inet addr' | sed 's/^.\*addr://g'

192.168.1.100 Bcast:192.168.1.255 Mask:255.255.255.0

接下来则是删除后续的部分，亦即： 192.168.1.100 Bcast:192.168.1.255 Mask:255.255.255.0

将 IP 后面的部分予以删除

[root@www ~]# /sbin/ifconfig eth0 | grep 'inet addr' | sed 's/^.\*addr://g' | sed 's/Bcast.\*$//g'

192.168.1.100

**多点编辑**

一条sed命令，删除/etc/passwd第三行到末尾的数据，并把bash替换为blueshell

nl /etc/passwd | sed -e '3,$d' -e 's/bash/blueshell/'

1 root:x:0:0:root:/root:/bin/blueshell

2 daemon:x:1:1:daemon:/usr/sbin:/bin/sh

-e表示多点编辑，第一个编辑命令删除/etc/passwd第三行到末尾的数据，第二条命令搜索bash替换为blueshell。

**直接修改文件内容(危险动作)**

sed 可以直接修改文件的内容，不必使用管道命令或数据流重导向！ 不过，由於这个动作会直接修改到原始的文件，所以请你千万不要随便拿系统配置来测试！ 我们还是使用文件 regular\_express.txt 文件来测试看看吧！

regular\_express.txt 文件内容如下：

[root@www ~]# cat regular\_express.txt

runoob.

google.

taobao.

facebook.

zhihu-

weibo-

利用 sed 将 regular\_express.txt 内每一行结尾若为 . 则换成 !

[root@www ~]# sed -i 's/\.$/\!/g' regular\_express.txt

[root@www ~]# cat regular\_express.txt

runoob!

google!

taobao!

facebook!

zhihu-

weibo-

:q:q

利用 sed 直接在 regular\_express.txt 最后一行加入 **# This is a test**:

[root@www ~]# sed -i '$a # This is a test' regular\_express.txt

[root@www ~]# cat regular\_express.txt

runoob!

google!

taobao!

facebook!

zhihu-

weibo-

# This is a test

由於 $ 代表的是最后一行，而 a 的动作是新增，因此该文件最后新增 **# This is a test**！

sed 的 **-i** 选项可以直接修改文件内容，这功能非常有帮助！举例来说，如果你有一个 100 万行的文件，你要在第 100 行加某些文字，此时使用 vim 可能会疯掉！因为文件太大了！那怎办？就利用 sed 啊！透过 sed 直接修改/取代的功能，你甚至不需要使用 vim 去修订！