**SED**

1. 基础语法

准备文件如下：

$ vi employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

上述数据包括：雇员ID 雇员姓名 雇员职位

1、Sed基本语法：

Sed [options] {sed-commands}{input-files}

Sed每次从input-file中读取一行记录，并在该记录上执行sed-comands；

sed首先从input-file中读取第一行，然后执行所有的sed-commands；在读取第二行，执行所有sed-commands，重复这个过程，知道input-file结束。

通过制定[options]还可以给sed传递一些可选的选项

打印/etc/passwd文件中所有行

sed 'p' /etc/passwd #打印文件所有行，并每行输出两次

sed –n 'p' /etc/passwd #打印文件所有行，每行输出一次

区别如下：

sed ’1,3p’ /etc/passwd #打印结果为所有行，全部输出

[root@cxp ~]# sed -n '1,5p' /etc/passwd #打印要求的行数

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

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

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

adm:x:3:4:adm:/var/adm:/sbin/nologin

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

=🡺具体解释，在下面哦！

2、使用sed脚本的基本语法

sed [options] –f {sed-commands-in-a-file}{input-file}

编写sed脚本：打印/etc/passwd中以root和nobody开头的行

[root@cxp ~]# vim test.sed

/^root/p

/^nobody/p

[root@cxp ~]# sed -n -f test.sed /etc/passwd #-f指定脚本文件-n同上

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

nobody:x:99:99:Nobody:/:/sbin/nologin

3、使用-e选项，执行多个sed命令

sed [options] –e {sed-commands-1} –e{sed-commands-2}{input-file}

sed –n –e ’/^root/p’-e ’/^nobody/p’ /etc/passwd

[root@cxp ~]# sed -n -e '/^root/p' -e '/^nobody/p' /etc/passwd

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

nobody:x:99:99:Nobody:/:/sbin/nologin

也可以这样： #用\分割

Sed –n \

-e ‘/^root/ p’ \

-e ‘/^nobody/ p’ \

/etc/passwd

还可以这样： #使用{}将命令分组

Sed -n ’{

/^root/p

/^nobody/p

}’ /etc/passwd

注意：sed命令绝不会修改源文件，它只是将结果内容输出到标准输出设备。如果要保持变更，应该使用重定向到>filename.txt

1. sed脚本执行流程

sed脚本执行流程遵从下面简单易记的顺序：Read，Execute，Print，Repeat（读取，执行，打印，重复）简称REPR。

分析脚本执行顺序：

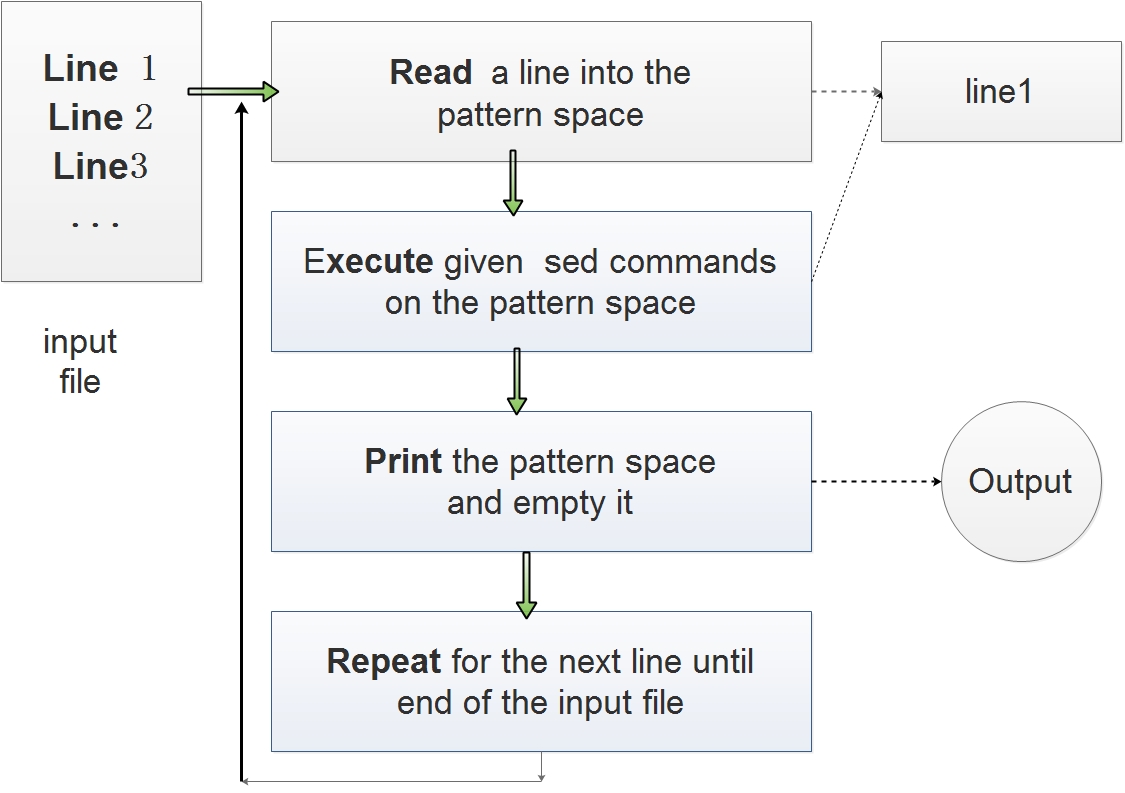
\*读取一行到模式空间（sed内部的一个临时缓存，用于存放读取到的内容）

\*在模式空间中执行命令，如果使用了{}或-e指定了多个命令，sed将依次执行每个命令

\*打印模式空间的内容，然后清空模式空间

\*重复上述过程，直到文件结束

Sed执行流程如下：



三、命令实战

1. 打印模式空间（命令p）

使用命令p，可以打印当前模式空间的内容。

sed在执行完命令后默认打印模式空间的内容，既然如此，那么为何还要命令p呢？

有如下原因：命令p可以控制只输出指定的内容。通常使用p时，还需要使用-n选项来屏蔽sed的默认输出，否则当执行命令p时，每行输出两次：

[root@cxp ~]# sed 'p' employee.txt

101,John Doe,CEO

101,John Doe,CEO

102,Jason Smith,IT Manager

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

105,Jane Miller,Sales Manager

[root@cxp ~]# sed -n 'p' employee.txt #结果对比很明显

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

1. 指定地址范围

如果在命令行前面不指定地址范围，那么默认会匹配所有行。

[root@cxp ~]# sed -n '2p' employee.txt #只打印第二行

102,Jason Smith,IT Manager

[root@cxp ~]# sed -n '1,4p' employee.txt #打印1,4行

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

[root@cxp ~]# sed -n '2,$p' employee.txt #打印2到最后一行

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

==🡺此处为空行

修改地址范围：

可以使用逗号、加号、和波浪号来修改地址范围。

上面的例子里面，就已经很明显使用了逗号参与地址范围的指定。意思：n，m代表第n至第m行。

加号+配合逗号使用，可以指定相的若干行，而不是绝对的几行。如：n，+m表示从第n行开始后的m行

波浪号~也可以指定地址范围，它指定每次要跳过的行数。如：n~m表示从第n行开始，每次跳过m行。

[root@cxp ~]# sed -n '1~2p' employee.txt #只打印奇数

101,John Doe,CEO

103,Raj Reddy,Sysadmin

105,Jane Miller,Sales Manager

1. 匹配模式

打印匹配模式“Jane”的行

[root@cxp ~]# sed -n '/Jane/p' employee.txt

105,Jane Miller,Sales Manager

打印第一匹配Jason的行到底4行的内容：

[root@cxp ~]# sed -n '/Jason/,4p' employee.txt

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

注意：如果开始的4行中，没有匹配到Jason，那么sed打印第4行以后匹配到的Json的内容

打印从第一次匹配到的Raj的行到最后的所有行：

[root@cxp ~]# sed -n '/Raj/,$p' employee.txt

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

打印匹配Jason的行和其后面的两行：

[root@cxp ~]# sed -n '/Jason/,+2p' employee.txt

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

1. 删除行

命令d用来删除行，需要注意的是它只是删除空间模式的内容，和其他sed命令一样，命令d不会修改原始文件的内容。

如果不提供地址范围，sed默认匹配所有行，所以下面的例子什么都不会输出，因为它匹配了所有行并删除了它们：

sed ’d’ employee.txt

指定删除的地址更有用。

只删除第2行

[root@cxp ~]# sed '2d' employee.txt

101,John Doe,CEO

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

[root@cxp ~]# cat employee.txt #源文件内容并没有改变

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

删除第1至4行：

[root@cxp ~]# sed '1,4d' employee.txt

105,Jane Miller,Sales Manager

删除第2行至最后一行：

[root@cxp ~]# sed '2,$d' employee.txt

101,John Doe,CEO

只删除奇数行：

[root@cxp ~]# sed '1~2d' employee.txt

102,Jason Smith,IT Manager

104,Anand Ram,Developer

删除第一次匹配到的Jason的行至第4行：

[root@cxp ~]# sed '/Jason/,4 d' employee.txt

101,John Doe,CEO

105,Jane Miller,Sales Manager

注意：如果开头的4行中，没有匹配到Jason的行，那么上述命令将删除第4行以后匹配Jason的行

删除所有空行：

[root@cxp ~]# sed '/^$/d' employee.txt #匹配空行

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

删除所有注释行（假定以#开头）

sed ’/^#/d’ employee.txt

注意：如果有多个命令，sed遇到命令d时，会删除匹配到的整行数据，其余的命令将无法操作被删除的行。

1. 把模式空间内容写到文件中（w命令）

命令w可以把当前模式空间的内容保存到文件中。默认情况下模式空间的内容每次都会打印到标准输出，如果要把输出保存到文件同时不显示到屏幕上，还需要使用-n选项。

把employmee.txt的内容保存到output.txt，同时显示在屏幕上：

[root@cxp ~]# sed 'w output.txt' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

[root@cxp ~]# cat output.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

把employmee.txt的内容保存到output.txt，同时不显示在屏幕上：

[root@cxp ~]# sed -n 'w output.txt' employee.txt

#空行

只保存第2行：

[root@cxp ~]# sed '2 w output.txt' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

[root@cxp ~]# cat output.txt

102,Jason Smith,IT Manager

#其他情况依此类推

1. sed替换命令

基础语法：

sed’address-range|pattern-range]s/original-string/replacement-string/[substitute-flags]’ input file

上面提到的语法为：

·address-range或pattern-range（即地址范围和模式范围）是可选的。如果没有指定，那么sed将在所有行进行替换。

·s即执行替换命令substitue

·original-string是被sed搜索然后被替换的字符串，它可以是一个正则表达式

·replacement-string替换后的字符串

·substitue-flags是可选的，下面具体解释

注意：原始文件的内容不会被修改，sed只在模式空间中执行替换命令，然后输出模式空间的内容。

用Director替换所有行中的Manager：

[root@cxp ~]# sed 's/Manager/Director/' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Director

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Director

只把包含Sales的行中的Manager替换为Director

[root@cxp ~]# sed '/Sales/s/Manager/Director/' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Director

7、g代表全局（global）默认情况下，sed至会替换每行中第一次出现的original-string。

用大写的A替换第一次出现的小写a：

[root@cxp ~]# sed 's/a/A/' employee.txt

101,John Doe,CEO

102,JAson Smith,IT Manager

103,RAj Reddy,Sysadmin

104,AnAnd Ram,Developer

105,JAne Miller,Sales Manager

把所有的小写字母a替换为大写的A：

[root@cxp ~]# sed 's/a/A/g' employee.txt

101,John Doe,CEO

102,JAson Smith,IT MAnAger

103,RAj Reddy,SysAdmin

104,AnAnd RAm,Developer

105,JAne Miller,SAles MAnAger

1. 数字标志（1,2,3....）

使用数字可以指定original-string出现的次序。只有第n次出现的original-string才会触发替换。每行的数字从1开始，最大为512.

把第2次出现的小写字母a替换为大写字母A：

[root@cxp ~]# sed 's/a/A/2' employee.txt

101,John Doe,CEO

102,Jason Smith,IT MAnager

103,Raj Reddy,SysAdmin

104,Anand RAm,Developer

105,Jane Miller,SAles Manager

为了方便下面的示例，请先建立如下文件：

[root@cxp ~]#vim substitute-locate.txt

locate command is used to locate files

locate command uses database to locate files

locate command can also use regex for searching

使用刚才建立的文件，把每行中第二次出现的locate替换为find：

[root@cxp ~]# sed 's/locate/find/2' substitute-locate.txt

locate command is used to find files

locate command uses database to find files

locate command can also use regex for searching

1. 打印标志（print）

命令p代表打印print。当替换操作完成后，打印替换的行，与其他打印命令类似，sed中比较有用的方法是和-n一起使用以抑制默认的打印操作。

只打印替换后的行：

[root@cxp ~]# sed -n 's/John/johnny/p' employee.txt

101,johnny Doe,CEO

在之前的数字标志的例子中，使用/2来替换第二次出现的locate。第3行中locate只出现了一次，所以没有替换任何内容。使用p标志可以只打印替换过的两行。

把每行中第2次出现的locate替换为find并打印出来：

[root@cxp ~]# sed -n 's/locate/find/2p' substitute-locate.txt

locate command is used to find files

locate command uses database to find files

1. 写标志

标志w代表write。当替换操作执行成功后，它把替换的结果保存到文件中。多数人更倾向于使用p打印内容，然后重定向到文件中。为了sed标志有个完整的描述，在这里把这个标志也提出来。

只把替换的内容写到output.txt中：

[root@cxp ~]# sed -n 's/John/hello/w output.txt' employee.txt

[root@cxp ~]# cat output.txt

101,hello Doe,CEO

把每行第2次出现的locate替换为find，把替换的结果保存到文件中，同时显示输入文件所有内容：

[root@cxp ~]# sed 's/locate/find/2w output.txt' substitute-locate.txt

locate command is used to find files

locate command uses database to find files

locate command can also use regex for searching

注意：上面两个是否输出，只是有无-n的区别

1. 忽略大小写标志i（ignore）

替换标志i代表忽略大小写，可以用i来以小写字符的模式匹配original-string。该标志值有GNU sed中才可以使用。

下面的例子不会把John替换为Johnny，因为替换original-string字符串是小写形式：

[root@cxp ~]# sed 's/John/Johnny/' employee.txt

101,Johnny Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

把John或john替换为Johnny：

[root@cxp ~]# sed 's/john/johnny/i' employee.txt

101,johnny Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

1. 执行命令标志e（excute）

替换标志e代表excute。该标志可以将模式空间中的任何内容当做shell命令执行，并把命令执行的结果返回到模式空间。该标志只有GNU sed中才可以使用。

为了下面例子，先建立如下文件：

[root@cxp ~]# cat files.txt

/etc/passwd

/etc/group

在files.txt文件中的每行前面添加ls –l 并把结果作为命令执行：

[root@cxp ~]# sed 's/^/ls -l /' files.txt

ls -l /etc/passwd

ls -l /etc/group

在files.txt文件中的每行前面添加ls –l并把结果作为命令执行：

[root@cxp ~]# sed 's/^/ls -l /e' files.txt #注意命令后面添加空格

-rw-r--r--. 1 root root 1230 Mar 27 20:59 /etc/passwd

-rw-r--r--. 1 root root 617 Mar 27 20:59 /etc/group

1. 使用替换标志组合

根据需要可以把一个或多个替换标志组合起来使用。

把每行中的出现的所有Manager或manager替换为Director，然后把替换后的内容打印到屏幕上，同时把这些内容保存到output.txt文件中：

使用g，l，p，w的组合：

[root@cxp ~]# sed -n 's/manager/Director/igpw output.txt' employee.txt

102,Jason Smith,IT Director

105,Jane Miller,Sales Director

[root@cxp ~]# cat output.txt

102,Jason Smith,IT Director

105,Jane Miller,Sales Director

1. 替换命令分界符

上面所有的例子中，我们都是使用sed的默认分界符/，即s/original-string/replace-string/g

如果在original-string或replace-string中有/，那么需要使用\来转义。为了方便示例，请先建立下面的文件：

[root@cxp ~]#vim path.txt

reading /usr/local/bin directory

限制使用sed把/usr/local/bin替换为/usr/bin。在下面例子中，sed默认的分界符/都被\转移了：

[root@cxp ~]# sed 's/\/usr\/local\/bin/\/usr\/bin/' path.txt

reading /usr/bin directory

很难看！如果要替换一个很长的路径，每个/前面都使用\转义，会显的很混乱。幸运地是，你可以使用任何一个字符来作为sed替换命令的分界符，如：|^@!

下面的例子就比较易读了：

[root@cxp ~]# sed -n 's@/usr/local/bin@/usr/bin@p' path.txt

reading /usr/bin directory

1. 单行内容上执行多个命令

Sed执行的过程是读取内容、执行命令、打印结果、重复循环，其中执行命令部分，可以由多个命令执行，sed将一个一个地依此执行它们。

例如：你有两个命令，sed将在模式空间中执行第一个命令，然后再执行第二个命令。如果第一个命令改变了模式空间的内容，第二个命令会在改变后的模式空间上执行（此时模式空间的内容已经不是最开始的读取内容了）。

下面演示了在模式空间内执行两个替换命令的过程：

把Developer替换为IT Manager，然后把Manager替换为Director：

[root@cxp ~]# sed -e 's/Developer/IT Manager/' -e 's/Manager/Director/' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Director

103,Raj Reddy,Sysadmin

104,Anand Ram,IT Director

105,Jane Miller,Sales Director

还可以这样：  
[root@cxp ~]# sed '{ #不需要加\换行符

> s/Developer/IT Manager/

> s/Manager/Director/

> }' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Director

103,Raj Reddy,Sysadmin

104,Anand Ram,IT Director

105,Jane Miller,Sales Director

分析一下第4行的执行过程：

A、读取数据：在这一步，sed读取内容到模式空间，此时模式空间的内容为：104,Anand Ram,IT Developer

B、执行命令：第一个命令，s/Developer/IT Manager/执行后，模式空间的内容为：104,Anand Ram,IT Manager

现在在模式空间上执行第二个命令s/Manager/Director/，执行后，模式空间内容为：104,Anand Ram,IT Director

谨记：sed在第一个命令执行的结果上，执行第二个命令

C、打印内容：打印当前模式空间的内容，如下：

104,Anand Ram,IT Director

D、重复循环：移动的输入文件的下一行，然后重复执行第一步，即读取数据

16、&的作用----获取匹配到的模式

当replace-string中使用&时，它会替换成匹配到的original-string或正则表达式，这是很有用的东西。

给雇员ID（即第一列的3个数字）加上[]，如101改成[101]:

[root@cxp ~]# sed 's/^[0-9][0-9][0-9]/[&]/g' employee.txt

[101],John Doe,CEO

[102],Jason Smith,IT Manager

[103],Raj Reddy,Sysadmin

[104],Anand Ram,Developer

[105],Jane Miller,Sales Manager

把每一行放进<>中：

[root@cxp ~]# sed 's/^.\*/<&>/' employee.txt

<101,John Doe,CEO>

<102,Jason Smith,IT Manager>

<103,Raj Reddy,Sysadmin>

<104,Anand Ram,Developer>

<105,Jane Miller,Sales Manager>

1. 分组替换（单个分组）

跟在正则表达式中一样，sed中也可以使用分组。分组以\(开始，以\)结束。分组可以用在回溯引用中。

回溯引用即重新使用分组选择的部分正则表达式，在sed替换命令的replace-string中和正则表达式中，都可以使用回溯引用。

单个分组：

[root@cxp ~]# sed 's@\([^,]\*\).\*@\1@g' employee.txt

101

102

103

104

105

上面例子中：

·正则表达式\([^,]\*\)匹配字符串从头开始到第一个逗号之间的所有字符（并将其放入第一组中）

·replace-string中的\1将替代匹配到的分组

·g是全局标志

下面这个例子只会显示/etc/passwd的第一列，即用户名：

[root@cxp ~]# sed 's/\([^:]\*\).\*/\1/g' /etc/passwd

root

bin

daemon

adm

lp

sync

shutdown

………省略

如果单词第一个为大写，那么会给这个大写字符加上（）:

[root@cxp ~]# echo "The Geek Stuff"|sed 's/\(\b[A-Z]\)/\(\1\)/g'

(T)he (G)eek (S)tuff # \b边界符

请先建立下面文件，以便示例使用：

[root@cxp ~]#vim numbers.txt

1

12

123

1234

12345

123456

格式化数字，增加其可读性：

[root@cxp ~]# sed 's/\(^\|[^0-9.]\)\([0-9]\+\)\([0-9]\{3\}\)/\1\2,\3/g' numbers.txt

1

12

123

1,234

12,345

123,456

1. 分组替换（多个分组）

可以使用多个\(和\)划分分组，使用多个分组时，需要在replace-string中使用\n来指定第n个分组。

只打印第一列（雇员ID）和第三列（雇员职位）：

[root@cxp ~]# sed 's/^\([^,]\*\),\([^,]\*\),\([^,]\*\)/\1,\3/g' employee.txt

101,CEO

102,IT Manager

103,Sysadmin

104,Developer

105,Sales Manager

在这个例子中，可以看到，original-string中，划分了3个分组，以逗号分隔。

·\([^,]\*\)第一个分组，匹配员工ID

* ，为分段分隔符
* \([^,]\*\)第二个分组，匹配雇员姓名
* ，为分段分隔符
* \([^,]\*\)第三个分组，匹配雇员职位
* ，为字段分隔符，上面的例子演示了如何使用分组
* \1代表一个分组（雇员ID）
* ，出现在第一个分组之后的逗号
* \3代表第二个分组（雇员职位）

注意：sed最多能处理9个分组，分别用\1至\9表示。

交换第一列（雇员ID）和第二列（雇员姓名）：

[root@cxp ~]# sed 's/^\([^,]\*\),\([^,]\*\),\([^,]\*\)/\2,\1,\3/' employee.txt

John Doe,101,CEO

Jason Smith,102,IT Manager

Raj Reddy,103,Sysadmin

Anand Ram,104,Developer

Jane Miller,105,Sales Manager

1. 正则表达式

行开头的（^） 匹配每一行的开头

显示以103开头的行：

[root@cxp ~]# sed -n '/^103/p' employee.txt

103,Raj Reddy,Sysadmin

行的结尾（$） 匹配行的结尾

显示以字符r结尾的行：

[root@cxp ~]# sed -n '/r$/p' employee.txt

102,Jason Smith,IT Manager

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

单点字符（.） 匹配除换行符之外的任意单个字符

.匹配单个字符

..匹配两个字符

依此类推

模式“J后面跟三个字符串和一个空格”将被替换为“Jason后面一个空格”。所以“J。。”同时匹配employee.txt文件中的“Jhon”和“Jane”，替换结果如下：

[root@cxp ~]# sed -n 's/J.../Jason/p' employee.txt

101,Jason Doe,CEO

102,Jasonn Smith,IT Manager

105,Jason Miller,Sales Manager

匹配零次或多次（\*）--星号\*匹配零个或多个其前面的字符串。

先建立下面文件：

[root@cxp ~]# vim log.txt

log: input.txt

log:

log: testing resumed

log:

log:output created

假设你想查看那些包含log并且后面有信息的行，log和信息之间可能有0个或多个空格，同时不想查看那些log：后面没有任何信息的行。

显示包含log：并且后面log后面有信息的行，log和信息之间可能有空格：

[root@cxp ~]# sed -n '/log:\*../p' log.txt #注意空格的个数

log: input.txt

log: testing resumed

log:output created

匹配一次或多次（\+） “\+”匹配一次或多次它前面的字符，例如：空格\+或

“\+”匹配至少一个或多个空格

显示包含log：并且log：后面有一个或多个空格的所有行：

[root@cxp ~]# sed -n '/log: \+/p' log.txt #注意\+前面加空格

log: input.txt

log: testing resumed #没有匹配到log：和log:output created这行

零次或一次匹配（\?） \?匹配0次或一次它前面的字符。

[root@cxp ~]# sed -n '/log:\?/p' log.txt

log: input.txt

log:

log: testing resumed

log:

log:output created

转义字符（\）

[root@cxp ~]# sed -n '/127\.0\.0\.1/p' /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

字符集（[0-9]） 字符集匹配方括号中出现的任意一个字符

匹配包含2、3或者4的行：

[root@cxp ~]# sed -n '/[234]/p' employee.txt

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

在方括号中，可以使用连接符-指定一个字符范围。如[0123456789]可以用[0-9]表示，字母可以用[a-z][A-Z]表示，等等。

或操作符（|） 管道符号|用来匹配两遍任意一个子表达式。

打印包含101或者包含102的行：

[root@cxp ~]# sed -n '/101\|102/p' employee.txt #注意转义|

101,John Doe,CEO

102,Jason Smith,IT Manager

精准匹配m次（{m}）

打印包含任意数字的行（这个命令打印所有行）：

[root@cxp ~]# sed -n '/[0-9]/p' numbers.txt

1

12

123

1234

12345

123456

打印包含5个数字的行：

[root@cxp ~]# sed -n '/^[0-9]\{5\}$/p' numbers.txt

12345

或者

打印包含5个数字的行：

[root@cxp ~]# sed -n 's/\([0-9]\{5\}\).\+/\1/p' numbers.txt

12345

匹配m至n次（{m，n}） 至少m次，最多n次，m、n不能是负数，并且要小于255.

打印由3至5个数字组成的行：

[root@cxp ~]# sed -n '/^[0-9]\{3,5\}$/p' numbers.txt

123

1234

12345

字符边界（\b） \b用来匹配单词开头（\bxx）或结尾（xx\b）的任意字符，因此\bthe\b将匹配the，但不匹配they，\bthe将匹配the或they。

请先建立如下文件：

[root@cxp ~]# vim words.txt

word matching using: the

word matching using: thethe

word matching using: they

匹配包含the作为整个单词的行：

[root@cxp ~]# sed -n '/\bthe\b/p' words.txt

word matching using: the

匹配以the开头的行：

[root@cxp ~]# sed -n '/\bthe/p' words.txt

word matching using: the

word matching using: thethe

word matching using: they

回溯引用（\n）---不是换行符

只匹配重复the两次的行：

[root@cxp ~]# sed -n '/\(the\)\1/p' words.txt

word matching using: thethe

同理：，“\([0-9]\)\1”匹配连续两个相同的数字，如：11,22,33。。。

sed中使用正则表达式：

把employee.txt中每行最后两个字符替换为“Not Defined”：

[root@cxp ~]# sed -n 's/..$/,Not Defined/p' employee.txt

101,John Doe,C,Not Defined

102,Jason Smith,IT Manag,Not Defined

103,Raj Reddy,Sysadm,Not Defined

104,Anand Ram,Develop,Not Defined

105,Jane Miller,Sales Manag,Not Defined

建立html文件：

[root@cxp ~]#vim test.html

<html><body><h1>Hello World!</h1></body></html>

清除test.html文件中的所有HTML标签：

[root@cxp ~]# sed 's/<[^>]\*>//g' test.html

Hello World!

删除所有注释行和空行：

[root@cxp ~]# sed -e 's/#.\*//;/^$/d' /etc/profile

pathmunge () {

case ":${PATH}:" in

\*:"$1":\*)

;;

\*)

if [ "$2" = "after" ] ; then

PATH=$PATH:$1

else

PATH=$1:$PATH

fi

esac

}

省略。。。。。

只删除注释行：

[root@cxp ~]# sed '/#.\*/d' /etc/profile

pathmunge () {

case ":${PATH}:" in

\*:"$1":\*)

;;

\*)

if [ "$2" = "after" ] ; then

PATH=$PATH:$1

else

PATH=$1:$PATH

fi

esac

}

省略。。。。

使用sed把DOS格式的文件转换为Unix格式：

sed ’s/.$//’ filename

1. sed追加命令（a）

使用命令a可以在指定位置的后面插入新行。

语法：sed ’[address] a the-line-to-append’ input-file

在后面两行追加一行：

[root@cxp ~]# sed '2 a 203,jack johnson,Engineer' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

203,jack johnson,Engineer

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

在文件结尾追加一行：

[root@cxp ~]# sed '$ a 106,jack johnson,Engineer' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

106,jack johnson,Engineer

注意：原文没有加行号

也可以追加多行：

[root@cxp ~]# sed '/Jason/a\ #也可以用\n来换行

> 203，jack johnson,Engineer\

> 204, Mark Smith,Sales' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

203，jack johnson,Engineer

204, Mark Smith,Sales

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

1. 插入命令（i）

插入命令insert命令和追加命令类似，只不过是在指定位置之前插入行。

语法：sed ’[address] i the-line-to-insert’ input-file

在employee.txt的第2行之前插入一行：

[root@cxp ~]# sed '2 i 203,jack johnson,Engineer' employee.txt

101,John Doe,CEO

203,jack johnson,Engineer

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

也可以插入多行，同追加一样！

1. 修改命令（c）

修改命令change可以用新行取代旧行。

语法：sed ’[address] c the-line-to-insert’ input-file

用新数据取代第2行：

[root@cxp ~]# sed '2 c 202,jack johnson' employee.txt

101,John Doe,CEO

202,jack johnson

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

多行取代一行：

[root@cxp ~]# sed '/Raj/c \

> 203,jack johnson\

> 240,smith ready' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

203,jack johnson

240,smith ready

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

1. 命令a，i，c组合使用

在“Jason” 后面追加“Jack johnson”，在前面插入“Mark Smith”，用“Joe Mason”替代“Jason”：

[root@cxp ~]# sed '/Jason/{

a\

204,Jack Johnson,Engineer

i\

202,Mark Smith,Sales Engineer

c\

203,Joe Mason,Sysadmin

> }' employee.txt #换行输入“}” ，不然会报错

101,John Doe,CEO

202,Mark Smith,Sales Engineer

203,Joe Mason,Sysadmin

204,Jack Johnson,Engineer

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

1. 打印不可见字符（l）

命令l可以打印不可见的字符，比如制表符\t行尾标志$等。

请先建立下面文件用于后续测试，请确保字段之间使用制表符（tab键）分开：

[root@cxp ~]#vim tabfile.txt

fname First Name

lname Last Name

mname Middle Name

使用命令l，把制表符显示为\t，行尾标志显示为EOL:

[root@cxp ~]# sed -n 'l' tabfile.txt

fname\tFirst Name$

lname\tLast Name$

mname\tMiddle Name$

如果在l后面指定了数字，那么会在第n个字符串处使用一个不可见自动折行，效果如下：

[root@cxp ~]# sed -n 'l 20' employee.txt

101,John Doe,CEO$

102,Jason Smith,IT \

Manager$

103,Raj Reddy,Sysad\

min$

104,Anand Ram,Devel\

oper$

105,Jane Miller,Sal\

es Manager$

注意：这个功能只有GNU sed才有

1. 打印行号（=）

命令=会在每一行后面显示该行的行号。

打印所有行号：

[root@cxp ~]# sed '=' employee.txt

1

101,John Doe,CEO

2

102,Jason Smith,IT Manager

3

103,Raj Reddy,Sysadmin

4

104,Anand Ram,Developer

5

105,Jane Miller,Sales Manager

只打印1,2,3行的行号：

[root@cxp ~]# sed '1,3=' employee.txt

1

101,John Doe,CEO

2

102,Jason Smith,IT Manager

3

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

打印包含关键字“Jane”的行的行号，同时打印输入文件的内容：

[root@cxp ~]# sed '/Jane/=' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

5

105,Jane Miller,Sales Manager

只显示行号不显示内容，那么使用-n选项来配合命令=：

[root@cxp ~]# sed -n '1=' employee.txt

\1

[root@cxp ~]# sed -n '/Jane/=' employee.txt

5

打印文件的总行数：

[root@cxp ~]# sed -n '$=' employee.txt

6

1. 转换字符（y）

命令y根据对应的位置转换字符，好处之一便是大写字符转换为小写，反之亦然。

将a换为A，b换位B，c换位C，依此类推：

[root@cxp ~]# sed 'y/abcde/ABCDE/' employee.txt

101,John DoE,CEO

102,JAson Smith,IT MAnAgEr

103,RAj REDDy,SysADmin

104,AnAnD RAm,DEvElopEr

105,JAnE MillEr,SAlEs MAnAgEr #注意是逐个替换

把所有小写字符替换为大写字符：

[root@cxp ~]# sed 'y/abcdefghijklmnopqrstuvwxyz/ABCDEFGHIJKLMNOPQRSTUVWXYZ/' employee.txt

101,JOHN DOE,CEO

102,JASON SMITH,IT MANAGER

103,RAJ REDDY,SYSADMIN

104,ANAND RAM,DEVELOPER

105,JANE MILLER,SALES MANAGER

[root@cxp ~]# sed 'y/[a-z]/[A-Z]/' employee.txt #注意此方式不满足逐个替换

101,John Doe,CEO

102,JAson Smith,IT MAnAger

103,RAj Reddy,SysAdmin

104,AnAnd RAm,Developer

105,JAne Miller,SAles MAnAger

1. 操作多个文件

同时在/etc/passwd和/etc/group中搜索root：

[root@cxp ~]# sed -n '/root/p' /etc/passwd /etc/group

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

operator:x:11:0:operator:/root:/sbin/nologin

root:x:0:

1. 退出sed（q）

命令终止正在执行的命令并退出sed。

之前提到，正常的sed执行流程：读取数据、执行命令、打印结果、重复循环。

当sed遇到q命令，便立刻退出，当前循环中的后续命令不会被执行，也不会继续循环。

打印第1行后退出：

[root@cxp ~]# sed 'q' employee.txt

101,John Doe,CEO

打印5行后退出：

[root@cxp ~]# sed '5 q' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

打印所有行，知道遇到关键字Manager退出的行：

[root@cxp ~]# sed '/Manager/q' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

注意：q命令不能指定地址范围（或模式范围），只能用于单个地址（或单个模式）。

1. 从文件中读取数据（r）

在处理输入文件时，命令r会从另外一个文件读取内容，并在指定的位置打印出来。

将读取log.txt的内容，并在打印employee.txt最后一行之后，把读取的内容打印出来。（事实上它把两个合并然后打印）：

[root@cxp ~]# sed '$ r log.txt' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

log: input.txt

log:

log: testing resumed

log:

log:output created

也可以指定一个模式

将读取log.txt的内容，并且在匹配’Raj’的行后面打印出来：

[root@cxp ~]# sed '/Raj/ r log.txt' employee.txt

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

log: input.txt

log:

log: testing resumed

log:

log:output created

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

1. 命令选项（-i）

sed不会修改输入文件，只会把内容打印到标准输出，或则使用w命令把内容写到不同的文件中，下面使用-i来直接修改输入文件。

在原始文件employee.txt中，用Johnny替换John：

[root@cxp ~]# cat employee.txt #更改前

101,John Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

[root@cxp ~]# sed -i 's/John/Johnny/' employee.txt

[root@cxp ~]# cat employee.txt #更改后

101,Johnny Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

执行和上面相同的命令，但在修改前备份原始文件：

[root@cxp ~]# sed -ibak 's/John/Johnny/' employee.txt

[root@cxp ~]# cat employee.txt

101,Johnny Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

[root@cxp ~]# ls

anaconda-ks.cfg output.txt

employee.txt path.txt

employee.txtbak substitute-locate.txt #备份文件

files.txt tabfile.txt

install.log test.html

install.log.syslog test.sed

log.txt words.txt

numbers.txt

1. 命令选项（-c）

该选项和-i配合使用。使用-i时，通常在名利执行完成后，sed使用临时文件来保持更改后的内容，然后把该临时文件重命名为输入文件。但这样会改变文件的所有者（奇怪的是测试结果不会改变文件所有者），配合c选项，可以保持文件所有者不变。也可以使用—copy来代替。

下面的命令是等价的：

[root@cxp ~]#sed -ibak -c 's/John/Johnny/' employee.txt

[root@cxp ~]#sed --in-place=bak --copy ‘s/John/Johnny/’ employee.txt

1. 选项（-l）length

指定行的长度，需要和l命令配合使用（注意：选项-l和命令-l，不要弄混了，上面提到的命令i和选项i也不要搞错）使用-l选项即指定行长度。也可以使用—line-length来代替。

下面是等价的：

[root@cxp ~]# sed -n --line-length=20 'l' employee.txt

101,Johnnynyny Doe,\

CEO$

102,Jason Smith,IT \

Manager$

103,Raj Reddy,Sysad\

min$

104,Anand Ram,Devel\

oper$

105,Jane Miller,Sal\

es Manager$

$

[root@cxp ~]# sed -n -l 20 'l' employee.txt 注意：第一个-l表示的是长度—line-length，第二个l表示list列出当前行的内容

101,Johnnynyny Doe,\

CEO$

102,Jason Smith,IT \

Manager$

103,Raj Reddy,Sysad\

min$

104,Anand Ram,Devel\

oper$

105,Jane Miller,Sal\

es Manager$

$

如下：

[root@cxp ~]# sed -n 'l' employee.txt

101,Johnnynyny Doe,CEO$

102,Jason Smith,IT Manager$

103,Raj Reddy,Sysadmin$

104,Anand Ram,Developer$

105,Jane Miller,Sales Manager$

注意：不适用-l选项同样可以列出相同的输出：

[root@cxp ~]# sed -n 'l 20' employee.txt

101,Johnnynyny Doe,\

CEO$

102,Jason Smith,IT \

Manager$

103,Raj Reddy,Sysad\

min$

104,Anand Ram,Devel\

oper$

105,Jane Miller,Sal\

es Manager$

$

1. 打印模式空间（n）

命令n打印当前模式空间的内容，然后从输入文件中读取下一行。如果在命令执行过程中遇到n，那么它会改变正常的执行流程。

打印每一行的内容：

[root@cxp ~]# sed -n 'p' employee.txt

101,Johnnynyny Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

[root@cxp ~]# sed n employee.txt #如果使用单一的-n选项没有输出

101,Johnnynyny Doe,CEO

102,Jason Smith,IT Manager

103,Raj Reddy,Sysadmin

104,Anand Ram,Developer

105,Jane Miller,Sales Manager

注意：不要把-n和n弄混了！

sed正常流程是：读取数据、执行命令、打印输出、重复循环。

命令n可以改变这个流程，打印当前模式空间的内容，然后清除模式空间，读取下一行进来，然后继续执行后面的命令。

假设命令n前后各有两个其他命令，如下：

sed-command-1

sed-command-2

n

sed-command-3

sed-command-4

这种情况下，sed-command-1和sed-command-2会在当前模式空间中执行，然后遇到n，它打印当前模式空间的内容，并清空模式空间，读取下一行的，然后把sed-command-3和sed-command-4应用于新的模式空间的内容。