linux中sed的用法

sed命令行格式为：

sed [-nefri] ‘command’ 输入文本/文件

常用选项：

-n：取消默认的输出,使用安静(silent)模式。在一般 sed 的用法中，所有来自 STDIN的资料一般都会被列出到屏幕上。但如果加上 -n 参数后，则只有经过sed 特殊处理的那一行(或者动作)才会被列出来

-e：进行多项编辑，即对输入行应用多条sed命令时使用. 直接在指令列模式上进行 sed 的动作编辑

-f：指定sed脚本的文件名. 直接将 sed 的动作写在一个档案内， -f filename 则可以执行 filename 内的sed 动作

-r：sed 的动作支援的是延伸型正则表达式的语法。(预设是基础正则表达式语法)

-i：直接修改读取的文件内容，而不是由屏幕输出

常用命令：

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，所以 d 后面通常不接任何内容

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： 列印，亦即将某个选择的资料印出。通常 p 会与参数 sed -n 一起用

s： 取代，可以直接进行替换的工作。通常这个 s 的动作可以搭配正则表达式。例如 1,20s/old/new/g

定址

定址用于决定对哪些行进行编辑。地址的形式可以是数字、正则表达式、或二者的结合。如果没有指定地址，sed将处理输入文件的所有行。

[16] 计算机使用与维护(133)
[17] C语言程序设计(111)
[18] 数据结构与算法(70)
[19] 构件技术/组件技术(17)
[20] 实习/毕设/培训(17)
[21] 软件测试/项目管理(27)
[22] 嵌入式开发(16)
[23] 英语(47)
[24] 我的软件(16)
[25] 生物信息学(81)
[26] 生物/医学(18)
[27] Matlab/Perl(37)
[28] 其他(42)
[29] 计算机基础(11)
[30]Shell(61)
[31]R(71)
[32]Python(34)
[33]Linux/Unix(77)
[34]PHP(10)
[35]MySQL(35)
[36]Oracle(25)
[37]SQL Server(68)
[38]Charts/Tables(11)
[39]模式识别/机器学习/优化(1)
[40]Perl(19)

积分与排名

积分 - 1281305

地址是一个数字，则表示行号；是"$"符号，则表示最后一行。例如：

```
sed -n '3p' datafile
只打印第三行
```

只显示指定行范围的文件内容，例如：

```
# 只查看文件的第100行到第200行
sed -n '100,200p' mysql_slow_query.log
```

地址是逗号分隔的，那么需要处理的地址是这两行之间的范围（包括这两行在内）。范围可以用数字、正则表达式、或二者的组合表示。例如：

```
sed '2,5d' datafile
#删除第二到第五行
sed '/My/,/You/d' datafile
#删除包含"My"的行到包含"You"的行之间的行
sed '/My/,10d' datafile
#删除包含"My"的行到第十行的内容
```

举例：（假设我们有一文件名为ab）

删除某行

```
[root@localhost ruby] # sed '1d' ab        #删除第一行
[root@localhost ruby] # sed '$d' ab        #删除最后一行
[root@localhost ruby] # sed '1,2d' ab      #删除第一行到第二行
[root@localhost ruby] # sed '2,$d' ab      #删除第二行到最后一行
```

显示某行

```
[root@localhost ruby] # sed -n '1p' ab      #显示第一行
[root@localhost ruby] # sed -n '$p' ab      #显示最后一行
[root@localhost ruby] # sed -n '1,2p' ab    #显示第一行到第二行
[root@localhost ruby] # sed -n '2,$p' ab    #显示第二行到最后一行
```

使用模式进行查询

```
[root@localhost ruby] # sed -n '/ruby/p' ab    #查询包括关键字ruby所在所有行
[root@localhost ruby] # sed -n '/\$/p' ab      #查询包括关键字$所在所有行，使用反斜线\屏蔽特殊含义
```

增加一行或多行字符串

```
[root@localhost ruby]# cat ab
Hello!
ruby is me,welcome to my blog.
end
[root@localhost ruby] # sed '1a drink tea' ab  #第一行后增加字符串"drink tea"
Hello!
drink tea
ruby is me,welcome to my blog.
end
[root@localhost ruby] # sed '1,3a drink tea' ab #第一行到第三行后增加字符串"drink tea"
Hello!
drink tea
ruby is me,welcome to my blog.
drink tea
end
drink tea
[root@localhost ruby] # sed '1a drink tea\nor coffee' ab   #第一行后增加多行，使用换行符\n
Hello!
drink tea
or coffee
ruby is me,welcome to my blog.
end
```

代替一行或多行

```
[root@localhost ruby] # sed '1c Hi' ab              #第一行代替为Hi
Hi
```

```
ruby is me,welcome to my blog.
end
[root@localhost ruby] # sed '1,2c Hi' ab          #第一行到第二行代替为Hi
Hi
end
```

替换一行中的某部分

格式：sed 's/要替换的字符串/新的字符串/g' （要替换的字符串可以用正则表达式）

```
[root@localhost ruby] # sed -n '/ruby/p' ab | sed 's/ruby/bird/g'    #替换ruby为bird
[root@localhost ruby] # sed -n '/ruby/p' ab | sed 's/ruby//g'       #删除ruby
```

插入

```
[root@localhost ruby] # sed -i '$a bye' ab        #在文件ab中最后一行直接输入"bye"
[root@localhost ruby]# cat ab
Hello!
ruby is me,welcome to my blog.
end
bye
```

替换：

-e是编辑命令，用于sed执行多个编辑任务的情况下。在下一行开始编辑前，所有的编辑动作将应用到模式缓冲区中的行上。

```
sed -e '1,10d' -e 's/My/Your/g' datafile
```

#选项-e用于进行多重编辑。第一重编辑删除第1-3行。第二重编辑将出现的所有My替换为Your。因为是逐行进行这两项编辑（即这两个命令都在模式空间的当前行上执行），所以编辑命令的顺序会影响结果。

```
# 替换两个或多个空格为一个空格

sed 's/[ ][ ]*/ /g' file_name


# 替换两个或多个空格为分隔符:
```

```
sed 's/[ ][ ]*/:/g' file_name

# 如果空格与tab共存时用下面的命令进行替换

# 替换成空格

sed 's/[[:space:]][[:space:]]*/ /g' filename

# 替换成分隔符:

sed 's/[[:space:]][[:space:]]*/:/g' filename
```

==========

sed命令的调用:

在命令行键入命令;将sed命令插入脚本文件,然后调用sed;将sed命令插入脚本文件,并使sed脚本可执行

```
sed [option] sed命令 输入文件                在命令行使用sed命令,实际命令要加单引号
sed [option] -f sed脚本文件 输入文件         使用sed脚本文件
sed脚本文件 [option] 输入文件                第一行具有sed命令解释器的sed脚本文件
```

option如下:

```
n 不打印; sed不写编辑行到标准输出,缺省为打印所有行(编辑和未编辑),p命令可以用来打印编辑行
c 下一命令是编辑命令,使用多项编辑时加入此选项
f 如果正在调用sed脚本文件,使用此选项,此选项通知sed一个脚本文件支持所用的sed命令,如
    sed -f myscript.sed input_file  这里myscript.sed即为支持sed命令的文件
```

使用重定向文件即可保存sed的输出

使用**sed**在文本中定位文本的方式:

```
x          x为一行号,比如1
x,y        表示行号范围从x到y,如2,5表示从第2行到第5行
/pattern/      查询包含模式的行,如/disk/或/[a-z]/
```

/pattern/pattern/ 查询包含两个模式的行,如/disk/disks/

/pattern/,x 在给定行号上查询包含模式的行,如/disk/,3

x,/pattern/ 通过行号和模式查询匹配行,如 3,/disk/

x,y! 查询不包含指定行号x和y的行

基本sed编辑命令:

p	打印匹配行	c/	用新文本替换定位文本
=	显示文件行号	s	使用替换模式替换相应模式
a/	在定位行号后附加新文本信息	r	从另一个文本中读文本
i/	在定位行号后插入新文本信息	w	写文本到一个文件
d	删除定位行	q	第一个模式匹配完成后退出或立即退出
l	显示与八进制ASCII代码等价的控制字符	y	传送字符
n	从另一个文本中读文本下一行,并附加在下一行	{}	在定位行执行的命令组
g	将模式2粘贴到/pattern n/		

基本sed编程举例:

使用p(rint)显示行: sed -n '2p' temp.txt 只显示第2行,使用选项n

打印范围: sed -n '1,3p' temp.txt 打印第1行到第3行

打印模式: sed -n '/movie/'p temp.txt 打印含movie的行

使用模式和行号查询: sed -n '3,/movie/'p temp.txt 只在第3行查找movie并打印

显示整个文件: sed -n '1,$'p temp.txt $为最后一行

任意字符: sed -n '/.*ing/'p temp.txt 注意是.*ing,而不是*ing

打印行号: sed -e '/music/=' temp.txt

附加文本:(创建sed脚本文件)chmod u+x script.sed,运行时./script.sed temp.txt

```
#!/bin/sed -f
/name1/ a/              #a/表示此处换行添加文本
HERE ADD NEW LINE.      #添加的文本内容
```

插入文本: /name1/ a/ 改成 4 i/ 4表示行号,i插入

修改文本: /name1/ a/ 改成 /name1/ c/ 将修改整行,c修改

删除文本: sed '1d' temp.txt 或者 sed '1,4d' temp.txt

替换文本: sed 's/source/OKSTR/' temp.txt 将source替换成OKSTR

sed 's//$//g' temp.txt 将文本中所有的$符号全部删除

sed 's/source/OKSTR/w temp2.txt' temp.txt 将替换后的记录写入文件temp2.txt

替换修改字符串: sed 's/source/"ADD BEFORE" &/p' temp.txt

结果将在source字符串前面加上"ADD BEFORE",这里的&表示找到的source字符并保存

sed结果写入到文件: sed '1,2 w temp2.txt' temp.txt

sed '/name/ w temp2.txt' temp.txt

从文件中读文本: sed '/name/r temp2.txt' temp.txt

在每列最后加文本: sed 's/[0-9]*/& Pass/g' temp.txt

从shell向sed传值: echo $NAME | sed "s/go/$REP/g" 注意需要使用双引号

快速一行命令:

's//.$//g'	删除以句点结尾行
'-e /abcd/d'	删除包含abcd的行
's/[][][]*/[]/g'	删除一个以上空格,用一个空格代替
's/^[][]*//g'	删除行首空格
's//.[][]*/[]/g'	删除句号后跟两个或更多的空格,用一个空格代替
'/^$/d'	删除空行
's/^.//g'	删除第一个字符,区别 's//.//g'删除所有的句点
's/COL/(.../)//g'	删除紧跟COL的后三个字母
's/^////g'	删除路径中第一个/

//

!!!!!!!!!!!

、使用句点匹配单字符 句点“.”可以匹配任意单字符。“.”可以匹配字符串头，也可以是中间任意字符。假定正在过滤一个文本文件，对于一个有10个字符的脚本集，要求前4个字符之后为X C，匹配操作如下：....X C....

2、在行首以^匹配字符串或字符序列 ^只允许在一行的开始匹配字符或单词。在行首第4个字符为1，匹配操作表示为：^ . . . 1

3、在行尾以$匹配字符串或字符 可以说$与^正相反，它在行尾匹配字符串或字符， $符号放在匹配单词后。如果在行尾匹配单词jet01，操作如下：jet01$ 如果只返回包含一个字符的行，操作如下：^ .$

4、使用*匹配字符串中的单字符或其重复序列 使用此特殊字符匹配任意字符或字符串的重复多次表达式。

5、使用/屏蔽一个特殊字符的含义 有时需要查找一些字符或字符串，而它们包含了系统指定为特殊字符的一个字符。如果要在正则

表达式中匹配以*.pas结尾的所有文件，可做如下操作：/*/.pas

6、使用**[]**匹配一个范围或集合　使用[ ]匹配特定字符串或字符串集，可以用逗号将括弧内要匹配的不同字符串分开，但并不强制要求这样做（一些系统提倡在复杂的表达式中使用逗号），这样做可以增 加模式的可读性。使用“-”表示一个字符串范围，表明字符串范围从“-”左边字符开始，到“-”右边字符结束。假定要匹配任意一个数字，可以使用：[0123456789]　要匹配任意字母，则使用：[A-Za-z]表明从A-Z、a-z的字母范围。

7、使用**/{/}**匹配模式结果出现的次数　使用*可匹配所有匹配结果任意次，但如果只要指定次数，就应使用/{/}，此模式有三种形式，即：

pattern/{n/} 匹配模式出现n次。

pattern/{n,/} 匹配模式出现最少n次。

pattern/{n,m} 匹配模式出现n到m次之间，n，m为0-255中任意整数。

匹配字母A出现两次，并以B结尾，操作如下：A/{2/}B匹配值为AAB　匹配A至少4次，使用：A/{4,/}B

==========

替换单引号为空：

可以这样写：

```
sed 's/'"'"'//g'

sed 's/'\''//g'

sed s/\'//g
```

==========

在文件的第一行前面插入一行**abc**

```
sed -i '1i\abc' urfile
```

分类: [33]Linux/Unix,[30]Shell

emanlee
关注 - 79
粉丝 - 412
+加关注

5 0

« 上一篇：eclipse+pydev (python) 配置出错
» 下一篇：linux 用grep匹配横线

posted @ 2013-09-07 18:10 emanlee 阅读(97894) 评论(2) 编辑 收藏

评论列表

#1楼 2015-09-29 16:56 winwill2012

下面这篇文章有更详细的讲解与示例，推荐大家看看。
http://qifuguang.me/2015/09/21/sed%E5%91%BD%E4%BB%A4%E8%AF%A6%E8%A7%A3/

支持(0) 反对(1)

#2楼 2017-11-17 14:33 rayinnight

Mark

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。

最新**IT**新闻:

- 微博针对"杠精"采取新规定：一人拉黑 禁评3天
- 微软在丹麦建立用于量子计算的材料实验室
- 科大讯飞AI遭质疑市值蒸发410亿 应收账款达25.5亿
- 创业后的"阿里人"，怎么就变成了"腾讯人"？| 前操盘手回忆（六）
- 赚了10亿美金，陌陌却不是阿里的成功案例？| 前操盘手回忆（五）

» 更多新闻...

最新知识库文章:

- 为什么说 Java 程序员必须掌握 Spring Boot ？
- 在学习中，有一个比掌握知识更重要的能力
- 如何招到一个靠谱的程序员
- 一个故事看懂"区块链"
- 被踢出去的用户

» 更多知识库文章...