# sed

MengChunlei

February 24, 2021

## 1 基本用法

sed 全称是 stream editor. 简单说, 它是按照行来处理文本的一个工具. 本文将按照以下 12 个部分来总结它的用法:

- 替换
- 地址使用
- 删除
- 插入
- 整行替换和转换
- 打印
- 读写文件
- 多行用法
- 空间保持和排除命令
- 流改变
- 模式替代
- 用例示范

## 2 替换

替换的命令: s/origin/replacement/. 多个替换命令可以串联, 用分号隔开.

Listing 1: Substitution

```
[chunleimeng]$ echo "hello world" | sed 's/hello/Hello/'
Hello world
[chunleimeng]$ cat file1
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
[chunleimeng]$ sed -e 's/brown/green/; s/dog/cat/' file1
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
[chunleimeng]$
```

命令可以放在一个文件里面. 也可以用 $-e$ 以写在多行上.

Listing 2: Substitution

```
[chunleimeng]$ cat script1.sed
s/brown/green/
s/dog/cat/
[chunleimeng]$ sed -f script1.sed file1
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
[chunleimeng]$ sed -e '
> s/brown/green/
> s/dog/cat/' file1
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
The quick green fox jumps over the lazy cat.
```

替换还有四种控制:

- 数字, 替换第几处
- g, 全部替换
- p, 打印替换的行
- w file, 将替换的结果写到文件

Listing 3: Substitution

```
[chunleimeng]$ cat file2
this is test a test
this is test
no key
[chunleimeng]$ sed 's/test/new/' file2   /*默认替换第一次出现的地方*/
this is new a test
this is new
no key
[chunleimeng]$ sed 's/test/new/2' file2  /*替换第二次出现的地方*/
this is test a new
this is test
no key
[chunleimeng]$ sed 's/test/new/g' file2
this is new a new
this is new
no key
[chunleimeng]$ sed -n 's/test/new/p' file2   /*-n表示不产生输出*/
this is new a test
this is new
[chunleimeng]$ sed -n 's/test/new/2p' file2
this is test a new
[chunleimeng]$ sed -n 's/test/new/w result.txt' file2
[chunleimeng]$ cat result.txt
this is new a test
this is new
```

# 3 地址使用

如果只想把命令作用于某些行或者不作用于某些行, 需要使用地址. 有两种方式:

- 以数字表示区间
- 用本文模式找到特定的行

使用数字表示区间的样例:

Listing 4: Address

```
[chunleimeng]$ sed '2s/dog/cat/' file1  /*只替换第二行*/
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy cat.
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
[chunleimeng]$ sed '2,3s/dog/cat/' file1 /*替换第二行到第三行*/
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy cat.
The quick brown fox jumps over the lazy cat.
The quick brown fox jumps over the lazy dog.
[chunleimeng]$ sed '2,$s/dog/cat/' file1 /*替换第二行到最后一行*/
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy cat.
The quick brown fox jumps over the lazy cat.
The quick brown fox jumps over the lazy cat.
[chunleimeng]$ sed '2{
> s/fox/elephant/
> s/dog/cat/}' file1
The quick brown fox jumps over the lazy dog.
The quick brown elephant jumps over the lazy cat.
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
[chunleimeng]$
```

使用文本模式示例:

Listing 5: Address

```
[chunleimeng]$ cat file2
this is test a test
this is test
no key
[chunleimeng]$ sed '/test a/s/test/abc/g' file2
this is abc a abc
this is test
no key
[chunleimeng]$
```

# 4 删除

删除的命令是 d.

Listing 6: Delete

```
[chunleimeng]$ cat file3
this is line number 1
this is line number 2
this is line number 3
this is line number 4
this is line number 1 again
line six
line seven
[chunleimeng]$ sed 'd' file3  /*删除所有行*/
[chunleimeng]$ sed '3d' file3 /*删除第三行*/
this is line number 1
this is line number 2
this is line number 4
this is line number 1 again
line six
line seven
[chunleimeng]$ sed '3,$d' file3 /*删除第三行到结尾*/
this is line number 1
this is line number 2
[chunleimeng]$ sed '/number 1/d' file3 /*删除包含'number 1'的行*/
this is line number 2
this is line number 3
this is line number 4
line six
line seven
[chunleimeng]$ sed '/1/,/3/d' file3 /*删除包含1的行到包含3的行. 数字1第二次出现的时候没有*
this is line number 4               /*对应的3的行,所以后面的都删除了*/
```

## 5 插入

插入有两种, i 表示在指定行前面插入, a 表示在指定行后面插入.

Listing 7: Insert

```
[chunleimeng]$ cat file4
this is line number 1
this is line number 2
[chunleimeng]$ sed '2i\insert line' file4 /*在第二行前面插入*/
this is line number 1
insert line
this is line number 2
[chunleimeng]$ sed '$a\insert line' file4 /*在最后一行后面插入*/
this is line number 1
this is line number 2
insert line
[chunleimeng]$ sed '2i\   /*插入多行*/
> insert line1\
> insert line2' file4
this is line number 1
insert line1
insert line2
this is line number 2
```

# 6 整行替换和转换

整行替换的命令是 c.

Listing 8: Change

```
[chunleimeng]$ cat file3
this is line number 1
this is line number 2
this is line number 3
this is line number 4
this is line number 1 again
line six
line seven
[chunleimeng]$ sed '3c\changed line' file3  /*替换第三行*/
this is line number 1
this is line number 2
changed line
this is line number 4
this is line number 1 again
line six
line seven
[chunleimeng]$ sed '/number 1/c\changed line' file3 /*替换包含'number 1'的行*/
changed line
this is line number 2
this is line number 3
this is line number 4
changed line
line six
line seven
[chunleimeng]$ sed '2,3c\changed line' file3 /*替换第2行到第3行*/
this is line number 1
changed line
this is line number 4
this is line number 1 again
line six
line seven
```

转换的命令是: y/inchars/outchars, 其中 inchars 和 outchars 要一样长. 它会把每一个 inchars 替换为对应的 outchars.

Listing 9: Transform

```
[chunleimeng]$ sed 'y/123/789/' file3
this is line number 7
this is line number 8
this is line number 9
this is line number 4
this is line number 7 again
line six
line seven
[chunleimeng]$ echo '111222333' | sed 'y/123/789/'  /*会替换所有的字符*/
777888999
```

# 7 打印

控制打印的有三个命令:

- p, 打印本文行
- =, 打印行号
- l, 列出行

Listing 10: Print

```
[chunleimeng]$ cat file5
this is line number 1
this is line number 2
this is line number 3
this is line number 4
[chunleimeng]$ sed -n '/number 3/p' file5
this is line number 3
[chunleimeng]$ sed -n '2,3p' file5
this is line number 2
this is line number 3
[chunleimeng]$ sed -n '/3/{
> p
> s/line/test/p
> }' file5
this is line number 3
this is test number 3
[chunleimeng]$ sed '=' file5
1
this is line number 1
2
this is line number 2
3
this is line number 3
4
this is line number 4
[chunleimeng]$ sed -n '/number 4/{
> =
> p
> }' file5
4
this is line number 4
[chunleimeng]$ cat file6
tag     test    end
[chunleimeng]$ sed -n 'l' file6  /*tab会显示为\t*/
tag\ttest\tend$
[chunleimeng]$
```

# 8 读写文件

- w, 写文件
- r, 读文件

Listing 11: File

```
[chunleimeng]$ cat file5
this is line number 1
this is line number 2
this is line number 3
this is line number 4
[chunleimeng]$ sed '1,2w result.txt' file5
this is line number 1
this is line number 2
this is line number 3
this is line number 4
[chunleimeng]$ cat result.txt
this is line number 1
this is line number 2
[chunleimeng]$ cat file7
insert line1
insert line2
[chunleimeng]$ sed '3r file7' file5 /*将file7中的内容插入第三行后*/
this is line number 1
this is line number 2
this is line number 3
insert line1
insert line2
this is line number 4
[chunleimeng]$ sed '$r file7' file5
this is line number 1
this is line number 2
this is line number 3
this is line number 4
insert line1
insert line2
```

# 9 多行用法

- next 命令
- 多行删除
- 多行打印

next 命令有两种, 小写 n 表示移动到下一行, 大写 N 表示将下一行的内容附加到当前行的后面.

Listing 12: next-n

```
[chunleimeng]$ cat file8
first line
second line
third line
forth line
[chunleimeng]$ sed '/first/{n ; d}' file8 /*删除first行的后面一行*/
first line
third line
forth line
[chunleimeng]$
```

Listing 13: next-N

```
[chunleimeng]$ cat file9
first line hello
world second line
third line
abc hello world ef
we hello world
[chunleimeng]$ sed 'N; s/hello world/HELLO WORLD/' file9 /*可以发现，第一行第二行换行的*/
first line hello                                         /*hello world没有替换成功*/
world second line
third line
abc HELLO WORLD ef
we hello world
[chunleimeng]$ sed 'N; s/hello.world/HELLO WORLD/' file9 /*上面的问题解决了，但是第一行*/
first line HELLO WORLD second line                       /*第二行合并了*/
third line
abc HELLO WORLD ef
we hello world
[chunleimeng]$ sed '                  /*分两种情况处理.问题是最后一行没有成功.因为没有*/
> N                                   /*下一行，所以N命令失败了，从而结束. 后面一节会*/
> s/hello\nworld/HELLO\nWORLD/        /*解决这个问题*/
> s/hello world/HELLO WORLD/' file9
first line HELLO
WORLD second line
third line
abc HELLO WORLD ef
we hello world
```

接下来是多行删除. 命令是 D.

Listing 14: Delete-D

```
[chunleimeng]$ cat file9
first line hello
world second line
third line
abc hello world ef
we hello world
[chunleimeng]$ sed 'N; /hello.world/d' file9 /*文件的第1–2行删除了一次,第3–4行删除了一次*/
we hello world
[chunleimeng]$ sed 'N; /hello.world/D' file9 /*删除目标串所在行的前一行*/
world second line
third line
we hello world
```

接下来是多行打印.

Listing 15: Print

```
[chunleimeng]$ sed -n 'N; /hello\nworld/p' file9 /*打印合并后的所有*/
first line hello
world second line
[chunleimeng]$ sed -n 'N; /hello\nworld/P' file9 /*打印前面一行*/
first line hello
[chunleimeng]$
```

# 10 空间保持和排除命令

sed 命令工作的空间称作模式空间, 还有一块缓冲区叫做保持空间. 有一些命令可以在这两个之间交换数据:

- h: 从模式空间复制到保持空间
- H: 将模式空间附加到保持空间
- g: 从保持空间复制到模式空间
- G: 将保持空间附加到模式空间
- x: 交换模式空间和保持空间

Listing 16: Hold Space

```
1 [chunleimeng]$ cat file9
2 first line hello
3 world second line
4 third line
5 abc hello world ef
6 we hello world
7 [chunleimeng]$ sed -n '/first/{h; n; p; g; p}' file9 /*先打印first行的下一行*/
8 world second line                                    /*再打印first行*/
9 first line hello
```

排除命令是感叹号!, 它表示相反的操作.

Listing 17: Negate

```
1 [chunleimeng]$ cat file9
2 first line hello
3 world second line
4 third line
5 abc hello world ef
6 we hello world
7 [chunleimeng]$ sed -n '/hello/!p' file9 /*包含hello的行不输出*/
8 world second line
9 third line
10 [chunleimeng]$ sed '$!N                  /*最后一行不执行N操作*/
11 > s/hello\nworld/HELLO\nWORLD/
12 > s/hello world/HELLO WORLD/' file9
13 first line HELLO
14 WORLD second line
15 third line
16 abc HELLO WORLD ef
17 we HELLO WORLD
18 [chunleimeng]$ sed -n '{1!G; h; $p}' file9  /*倒序输出文件:*/
19 we hello world                             /*     1: 第一行不执行G操作*/
20 abc hello world ef                         /*     2: 到最后一行的时候才执行p操作*/
21 third line
22 world second line
23 first line hello
```

# 11 流改变

正常的执行规则对于文本的每一行, 顺序执行脚本中的每个命令. 流改变的意思是可以改变这个规则, 包括两个命令, 一个命令类似 C 语言的 goto, 即满足某个条件的话会跳转到某个地方去执行之后的命令; 另一个也是类似 goto, 不过不是某个条件成立, 而是之前的命令执行成功时跳转. 第一个命令是 b, 第二个命令是 t.

Listing 18: Stream Change-b

```
[chunleimeng]$ cat file9
first line hello
world second line
third line
abc hello world ef
we hello world
[chunleimeng]$
[chunleimeng]$ sed '2,3b; s/world/WORLD/' file9   /*第2-3行直接跳转到脚本的最后*/
first line hello                                  /*其他行不跳转, 去执行后面的替换命令*/
world second line
third line
abc hello WORLD ef
we hello WORLD
[chunleimeng]$ sed '/hello/b label; s/world/WORLD/; :label s/hello/HELLO/' file9
first line HELLO                    /*包含hello的行跳转到label去替换hello*/
WORLD second line                   /*不包含hello的会顺序执行所有的命令*/
third line
abc HELLO world ef
we HELLO world
[chunleimeng]$ echo "this, is, a, test, to, remove, commas" | sed -n '
:start
s/,//1p             /*去掉逗号并打印*/
/,/b start'         /*替换后还有逗号就跳转,否则结束*/
this is, a, test, to, remove, commas
this is a, test, to, remove, commas
this is a test, to, remove, commas
this is a test to, remove, commas
this is a test to remove, commas
this is a test to remove commas
[chunleimeng]$ sed '
> s/hello/HELLO/                    /*替换hello成功, 则跳转到脚本最后*/
> t                                 /*否则不跳转, 替换world*/
> s/world/WORLD/' file9
first line HELLO
WORLD second line
third line
abc HELLO world ef
we HELLO world
[chunleimeng]$ echo "this, is, a, test, to, remove, commas" | sed -n '
:start
s/,//1p
t start'          /*替换逗号成功则跳转到start*/
this is, a, test, to, remove, commas
this is a, test, to, remove, commas
this is a test, to, remove, commas
this is a test to, remove, commas
```

```
47 this is a test to remove, commas
48 this is a test to remove commas
```

## 12 模式替代

这个跟正则表达式中的反向引用类似, 就是使用匹配的模式.

Listing 19: Stream Change-b

```
1 [chunleimeng]$ echo 'The cat sleeps in his hat' | sed 's/\(.at\)/"\1"/g'
2 The "cat" sleeps in his "hat"   /*将cat和hat加上引号*/
3 [chunleimeng]$ echo 'abc def ghij afc' | sed 's/a\(.\)c/XX\1XX/g'
4 XXbXX def ghij XXfXX  /*将abc和afc两边替换成XX*/
5 [chunleimeng]$ echo '1234567' | sed '    /*三个一组加上逗号*/
6 > :start
7 > s/\(.*[0-9]\)\([0-9]\{3\}\)/\1,\2/
8 > t start'
9 1,234,567
```