

## Manual de introducción al comando sed

### Introducción

Se trata de un editor de cadenas, que deriva directamente del comando ed, por lo cual es fácil de aprender, una vez conocido el propio comando ed.

Un editor de cadenas es usado para realizar transformaciones de texto básicas de una cadena de entrada (un archivo o la salida de un pipe). A diferencia de otros editores comunes, sed hace solo una revisión sobre cada entrada, por lo cual es mas eficiente, en especial lo es, cuando filtra texto que pasa por una unión pipe.

Cuando sed filtra un archivo, no altera su contenido, todos los cambios los realiza sobre la salida que imprime. Sed despliega cada renglón automáticamente y por ello no se necesita después del comando de sustitución (s, por ejemplo s/old/new/). Las comillas son indispensables pues existen metacaracters de sed que tienen significado especial para el shell sobre el cual se ejecuta.

Sin embargo, sed tiene una limitación que no se encuentra en ed: no maneja números relativos derenglón. En particular, + y - no son entendidos como expresiones para indicar números de renglón, por lo cual resulta imposible de retroceder a uno o mas renglones anteriores o hacer direccionamientos relativos hacia adelante. Esto se debe a que una vez leído un renglón, el anterior se pierde para siempre; no hay manera de identificar el penúltimo renglón.

En la lista de comandos puede existir cualquiera de los listados más adelante. Cuando se hace uso de uno solo, basta con encerrarlo entre comillas simples como sigue:

```
sed 's/old/new/' archivo
```

Pero en el caso de que necesitemos ejecutar varios comandos sobre un mismo archivo, es necesario que estos se encuentren en diferentes lineas, por ejemplo:

```
sed 's/^/ /  
3q' archivo
```

No hay que olvidar que antes de escribir el primer comando, se deben abrir las comillas simples y las mismas se deben cerrar después de teclear el último comando. También es posible almacenar los comandos dentro de un archivo y llamarlo a la hora de que se ejecute sed con el parámetro -f. Por ejemplo:

```
sed -f cmdarchivo archivo_texto
```

Se pueden utilizar otros selectores o patrones antes de cada comando, por ejemplo:

```
sed '/patron/q'   Imprime cada linea de archivo hasta encontrar aquella que contenga el patron  
sed '/patron/d'   Cancela todas las lineas que contengan el patron y no se imprimen
```

Incluso si se desea el resultado inverso a lo que realiza un comando, basta con agregar un '!' antes del comando. Por ejemplo, para los dos ejemplos anteriores se prosigue como sigue:

```
sed '/patron/!q'  Imprime cada linea de archivo hasta encontrar aquella que contenga el patron  
sed '/patron/!d'  Cancela todas las lineas que contengan el patron y no se imprimen
```

## Sintaxis

La sintaxis de sed es la siguiente

```
sed [-n] [-V] [--quiet] [--silent] [--version] [--help]
    [-e script] [--expression=script]
    [-f script-file] [--file=script-file]
    [script-if-no-other-script]
    [file...]
```

-V ó --version

Imprime la versión de sed y nota de copyright.

-h ó --help

Imprime un resumen de todas las opciones de sed.

-n ó --quiet ó --silent

Por defecto, sed imprime cada línea cuando termina de editarla. Esta opción deshabilita la impresión automática y no se produce ninguna salida, excepto que se incluya el comando p.

-e script ó --expression=script

Agrega los comandos de la cadena script, encerrada entre comillas simples, a la lista de comandos para ser ejecutados mientras se procesa la entrada.

-f script-file ó --file=script-file

Agrega los comandos contenidos en el archivo script para ser procesados sobre la entrada.

Si ninguno de los dos parámetros anteriores es utilizado, en cualquiera de sus formas, sed tomará como comandos la primer expresión que no sea opción de sed como argumento para ejecutar como si fuera script.

## Comandos

A continuación se presenta una lista de comandos que se pueden utilizar para filtrar una entrada con sed, así mismo se incluyen algunos ejemplos para una mayor comprensión.

### Sustitución

Su función es sustituir una cadena old por una nueva new en cada línea que procese. Su sintaxis es la siguiente:

```
s/old/new/
```

Ejemplo:

```
sed 's/casa/carro/' contrato.txt
```

Sustituye la cadena 'casa' por 'carro' en contrato.txt

```
sed 's/^./-' arch.txt
```

Agrega la cadena '-' al inicio de cada línea de arch.txt

```
sed 's/^\\
```

```
/' arch.txt
```

Agrega un salto de línea al final de cada línea de arch.txt

Otro comando que puede ser útil es y, el cual reemplaza cada carácter procedente de old con el carácter correspondiente proveniente de new. Aquí no se admiten rangos. Su sintaxis es:

```
y/old/new/
```

### Borrado

Su función es borrar a la salida, las líneas que correspondan al patrón escrito. Su sintaxis es la siguiente:

```
/patron/d
```

Ejemplo:

```
sed '/calvo/d' arch.txt    Elimina las líneas que contienen la cadena 'calvo' en arch.txt
sed '/amor/d' carta.txt   Elimina las líneas que no tengan la palabra amor en carta.txt
sed '3d' forma.txt        Elimina la línea 3 del archivo forma.txt
```

### Impresión

Su función es imprimir la salida de sed. En realidad, sed imprime automáticamente la entrada filtrada, entonces si se agrega el comando p, cada línea de salida se imprimirá doble. En caso de que se agregue el parámetro -n a sed, pero se quiera imprimir la salida, se agrega el comando p. Su sintaxis es la siguiente:

```
/patron/p
```

Ejemplo:

```
who | sed -n '/root/p'    Imprime solamente las sesiones de root
who | sed '/root/p'       Imprime dos veces cada sesión abierta de root
who | sed -n '2p'         Imprime solamente la segunda sesión que se inició en el sistema
```

### Abandonar filtrado

Otro comando útil para imprimir una parte de la entrada es q. Este comando termina inmediatamente las operaciones de sed, en base a la correspondencia del patrón con las líneas que este procesando. Su sintaxis es la siguiente:

```
/patron/q
```

Ejemplo:

```
sed '3q' forma.txt        Imprime hasta la línea 3 del archivo forma.txt
sed '/clavo/q' arch.txt    Imprime desde la línea 1 de arch.txt hasta encontrar aquella que contiene la palabra 'clavo'
```

### Inserción de texto

Este comando inserta o agrega texto a cada línea que procesa sed. El texto debe estar localizado en las siguientes líneas que proceden después del carácter '\'. Su sintaxis es la siguiente:

```
i \      y      a\
[ texto a agregar ]
```

Ejemplo:

```
sed 'i \
.' oraciones.txt        Agrega un punto final a cada línea de oraciones.txt
sed 'a \
' doble.txt             Agrega un salto de línea al final de cada línea de doble.txt
```

**Guardar la entrada filtrada en un archivo**

Este comando permite salvar la entrada filtrada en otro archivo. Su sintaxis es la siguiente:

```
/patron/w archivo
```

Ejemplo:

```
sed '/pat/w patos.txt
```

```
    /pat!/w otros.txt animales.txt
```

Guarda las líneas que concuerden con 'pat' en un archivo llamado patos.txt y los que no concuerden en el archivo otros.txt

```
sed 's/UNIX/UNIX(TM)/gw u.out' libros.txt > salida.out
```

Almacena toda la salida en el archivo salida.out, pero las líneas que fueron modificadas, se almacenan en u.out

**Funciones****Etiqueta**

Permite identificar una etiqueta label, si se realiza un cambio en la línea actual. Esta es especificada por ':' y seguida del nombre de la etiqueta. Su sintaxis es como sigue:

```
: label
```

**Grupo**

Indica un grupo de comandos, que inicia con un caracter '{' y termina con un '}'.

**Lectura de archivos**

Con el comando r, lee un archivo y agrega todo su contenido al final de cada línea de entrada de sed. Su sintaxis es:

```
nr archivo
```

Donde n es el número de línea donde pegar el texto de archivo.

**Listar línea**

El comando es l y su función es hacer visibles todos los caracteres no imprimibles de cada línea de entrada.

**Línea actual**

Se puede imprimir el número de la línea actual, si se agrega el comando '='. Su sintaxis es:

```
/patron/=
```

**Salto**

Permite saltar o ir a la línea donde se encuentra una etiqueta label definida y continuar la ejecución de sed desde ese comando. Su sintaxis es:

```
b label
```

**Usos de sed**

En realidad sed tiene bastante utilidad como la comprobación de condiciones, creación de ciclos y

ramificaciones, retención de líneas precedentes y muchos de los comandos de ed. Su uso es similar al que se ha descrito en este manual, comandos sencillos de edición y no tanto de secuencias largas o complejas.

Sed es adecuado para manejar entradas arbitrariamente grandes de forma rápida, pero ofrece una forma bastante reducida de memoria, puesto que al filtrar una siguiente línea, ya no recuerda la anterior, por lo que no se puede regresar a filtrar la anterior y tampoco proporciona recursos para manipular números. Solo es un editor de texto.