awk

awk '{print $1,$3}' log.txt #输出log.txt文件的1, 3项, 每行默认按空格或TAB分割

awk -F ',' '{print $1,$3}' log.txt #参数 –F, 指定分割符, 以逗号为分隔符

awk -F '[ ,]' '{print $1,$3}' log.txt #指定多个分隔符, 第一个是空格, 第二个逗号

awk -F '[;,]' '{print}' log.txt #多个分隔符, 输出每行所有项目

算术运算

awk '$1>2' log.txt #输出第一项大于2的行

awk '$1=2 && $2=="is" ' log.txt #第1项等于2, 且第2项等于is的行

正则, 字符串匹配

awk '/th/' log.txt #输出包含th字符的行, 匹配 / / 之间的模式

awk '$1 ~ /th/' log.txt #输出第1项包含th的行, ~ 表示模式开始

awk 'length>100' log.txt #输出长度大于100的行

* grep 更适合单纯的查找或匹配文本
* sed 更适合编辑匹配到的文本
* awk 更适合格式化文本，对文本进行较复杂格式处理

sed

参数说明:

a 新增, 出现在下一行

c 取代, 取代两行之间的行

d 删除

i 插入, 出现在上一行

p 选择要输出的行, 与sed –n 同时用

s 取代, 直接取代

sed 4a\newLine log.txt #在文件第4行后面新增一行

cat log.txt | sed 4i\newline #在第4行前新增一行

sed 1,3d log.txt #删除1-3行

sed 1,$d log.txt #删除1到最后一行

sed 1,2c line12 log.txt #用line12 取代第1,2行

sed /root/d log.txt #搜寻包含root的行, 并删除

sed s/root/newroot/g log.txt #搜寻包含root的行, 并用newroot代替

sed -n 1,2p log.txt #输出1-2行

一次多个编辑

sed 1d log.txt | sed s/hello/hi/g #删除第一行, 并将hello用hi代替

sed -e 1d -e s/hello/hi/g log.txt #同上, -e表示多点编辑

cut

显示每行从开头算起 num1 到 num2 的文字

cut -b 1-2 log.txt #输出前两个字节

cut -c 1-2 log.txt #输出前两个字符

grep

用于查找文件里符合条件的字符串. 若不指定任何文件名称，或是所给予的文件名为"-"，则grep指令会从标准输入设备读取数据

常用参数:

-i 忽略大小写

-n 列出行号

-r 递归查询, 指定目录及其子目录

-v 反转查询, 输出不符合条件的行

* 搜寻特定字符串；
* 利用 [ ] 来搜寻集合字符；
* 行首字符 ^ 与行尾字符 $ ；
* 任意一个字符 . 与重复字符 \* ；
* 限定连续字符范围 { }。

wc

参数:

-c 显示字节数

-l 显示行数

-w 显示单词数

wc log.txt #默认情况下, 显示行数, 单词数, 字节数

wc -l log.txt #行数

wc -w log.txt #单词数

wc -c log.txt #字节数

date

常用显示格式:

%n 换行

%t 跳格

%H 小时

%M 分钟

%S 秒

%s 从1970/1/1开始的秒数

%N 纳秒部分

%T 直接显示时间,24小时制, 相当于 %H:%M:%S

%A 星期几

%B 月份

%d 日

%Y 年

%x 直接显示日期 mm/dd/yy

%D 直接显示日期 yy年mm月dd日

%c 显示日期与时间

常用用法:

date +%c #显示日期与时间

date +%T #时间

date +%x #日期

date --date '12:34:56' #设定时间

tr

用于转换或删除文件中的字符, 用法: tr [参数] [SET1][SET2]

参数:

-c 反选设定字符, 符合SET1的不作处理

-d 删除设定字符

-s 删除连续重复的字符成单个

字符集范围:

CHAR1-CHAR2 两个字符之间的所有字符

[:alnum:] 所有字母与数字

[:alpha:] 所有字母

[:cntrl:] 所有控制字符

[:digit:] 所有数字

[:punct:] 所有标点

[=CHAR=] 指定字符

cat log.txt | tr a-z A-Z #将文件中所有小写字母转换成大写

cat log.txt | tr -d A-Z #删除所有的大写字母

cat log.txt | tr –dc A-G #删除非A-G之间的所有字符

cat log.txt | tr -d [=';'=] #删除所有分号”;”

expr

expr length “this is a test” #获取字符串长度

expr substr “this is a test” 3 5 #子串, 从第3开始后的5个字符

expr index "sarasara" ra #字符串ra第一次出现的位置

expr $v1 + $v2

expr $v1 - $v2

expr $v1 \\* $v2

expr $v1 / $v2

expr $v1 % $v2

字符串截取

var=https://www.baidu.com/index.html

1. **# 删除左边字符，保留右边字符**

**echo ${var#\*/} #从左边开始删除第一个 / 号及左边的所有字符,**

**#结果:** /www.baidu.com/index.html

1. **## 删除左边字符，保留右边字符**

**echo ${var##\*/} #从左边开始删除最后一个 / 号及左边的所有字符,**

**#结果:**  index.html

1. **% 删除右边字符，保留左边字符**

**echo ${var%/\*} #从右边开始删除第一个 / 号及右边的所有字符,**

**#结果:** /www.baidu.com

1. **%% 删除右边字符，保留左边字符**

**echo ${var%%/\*} #从右边开始删除最后一个 / 号及右边的所有字符,**

**#结果:** http:

1. **从左边第几个字符开始，及字符的个数**

echo ${var:0:5} #从左边第1个字符开始的5个字符

#结果: https

1. **从左边第几个字符开始，一直到结束**

echo ${var:8} #从左边第9个字符开始到末尾的所有字符

#结果: www.baidu.com/index.html

1. **从右边第几个字符开始，及字符的个数**

echo ${var:0-10:5} #从右边第10个字符开始的5个字符

#结果: index

1. **从右边第几个字符开始，一直到结束**

echo ${var:0-10} #从右边第10个字符开始的字符

#结果: index.html

find

常见用法：

find /home –name “\*.txt” 指定目录下以.txt结尾的文件

find /home -name "\*.txt" -o -name "\*.pdf" 以.txt和.pdf结尾的文件

find /home ！ –name “\*.txt” 指定目录下不是以.txt结尾的文件

find /home -type f 普通文件，d 目录，c 字符设备，b 块设备等

find /home –type f -atime -7 七天内被访问过的文件，atime访问时，mtime 修改时间

find /home –type f -atime 7 七天前（当天）被访问过的文件

find /home –type f -atime +7 超过七天被访问过的文件

find /home –type f -amin +10 10分钟之前被访问过的文件

find /home –type f -newer file.log 比 file.log 修改时间更早的文件

find /home –type f -mtime +7 –exec rm {}\; 删除7天前修改的文件

find /home –type f -mtime +7 –delete 删除7天前修改的文件

find /home –type f –size +10k 大于10kb 的文件； 10kb 等于；-10kb 小于