1)linux统计一个文件中特定字符的个数

[root@web ~]# cat 1.log

1 2

2 3

3 4

100 1

[root@web ~]# cat 1.log | awk '{for(i=1;i<=NF;i++)a[$i]++}END{for(j in a) if(j==100) print j,a[j]}'

100 1

[root@web ~]# cat 1.log | awk '{for(i=1;i<=NF;i++)a[$i]++}END{for(j in a) print j,a[j]}'

4 1

100 1

1 2

2 2

3 2

1、统计某个字符的个数，以a为例

grep -o '1' 1.log | wc –l

awk -F "" '{for(i=1;i<=NF;++i) if($i=="a") ++sum}END{print sum}' urfile

#-F ""中间必须要有空格   "a"必须使用双引号，字符串

awk -F "" '{for(i=1;i<=NF;i++)a[$i]++}END{for(j in a) if(j=="c") print a[j]}' urfile

#

2、储存在变量里

echo "0001111" |tr -cd 1 |wc -c

$ grep -o a urfile | wc -l

      4

$ tr -cd a < urfile | wc -c

      4

$ awk -Fa 'NF>1{s+=NF-1}END{print s}' urfile

4

$ awk '{s+=gsub(/a/,"")}END{print s}' urfile

4

$ awk -v RS=a 'END{print NR-1}' urfile

4

awk '{for(i=1; i<=length($0); i++) ++S[substr($0,i,1)]} END {for(a in S) print S[a], a}' aa

awk -v RS='\0' -F'3' '{print NF-1}' 1.log

2)

|  |
| --- |
| zhangsan     80  lisi         81.5  wangwu       93  zhangsan     85  lisi         88  wangwu       97  zhangsan     90  lisi         92  wangwu       88 |

要求输出格式：（average：平均成绩，total：总成绩）   
name#######average#######total   
zhangsan            xxx                        xxx   
lisi                       xxx                        xxx   
wangwu             xxx                        xxx

[root@web ~]# cat 1.log |awk 'BEGIN{printf"%-10s\t%-10s%-10s\n","name","avg","total"}{a[$1]+=$2;b[$1]++}END{for(i in a) printf"%-10s\t%-10f\t%-10i\n",i,a[i]/b[i],a[i]}'

name avg total

zhangsan 85.000000 255

wangwu 92.666667 278

lisi 87.166667 261

3)

AWK笔试题

获取本机的IP地址和掩码

# cat /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE="eth0"

BOOTPROTO="static"

HWADDR="EC:9E:CD:0E:7C:E8"

NM\_CONTROLLED="yes"

ONBOOT="no"

TYPE="Ethernet"

UUID="9a975e2c-dabc-4c5e-91ca-56521a409dd8"

NAME="eth0"

ONBOOT=yes

IPADDR=168.0.190.82

PREFIX=16

GATEWAY=168.0.0.254

#!/bin/bash

#This script print IP and MASK

file="/etc/sysconfig/network-scripts/ifcfg-eth0"

if [ -f $file ]

then

ip=`grep IPADDR $file | awk -F= '{print $2}'`

mask=`grep NETMASK $file | awk -F= '{print $2}'`

if [ -z $mask ]

then

echo "$ip"

else

echo "$ip/$mask"

fi

else

echo "$file is not exist !"

fi

文本分析

取出password中shell出现的次数

第一种方法结果：

4 /bin/bash

1 /bin/sync

1 /sbin/halt

31 /sbin/nologin

1 /sbin/shutdown

第二种方法结果：

/bin/sync 1

/bin/bash 1

/sbin/nologin 30

/sbin/halt 1

/sbin/shutdown 1

# awk -F: '{print $NF}' /etc/passwd | sort |uniq -c

4 /bin/bash

1 /bin/sync

1 /sbin/halt

24 /sbin/nologin

1 /sbin/shutdown

# awk -F: '{print $NF}' /etc/passwd | sort |uniq -c |awk '{print $2,$1}'

/bin/bash 4

/bin/sync 1

/sbin/halt 1

/sbin/nologin 24

/sbin/shutdown 1

[root@web ~]# cat /etc/passwd | awk -F ":" '{a[$NF]++}END{for(i in a)print a[i],i}' | sort -nr

25 /sbin/nologin

2 /bin/bash

1 /sbin/shutdown

1 /sbin/halt

1 /bin/sync

输出记录最多的IP

[腾讯面试题]：一个文本类型的文件，里面每行存放一个登陆者的IP（某些行是重复的），写一个shell脚本输出登陆次数最多的用户。

Ip\_input.txt的内容假设如下：

219.217.49.14

175.43.4.87

87.48.98.1

59.73.38.25

219.217.50.14

59.92.48.32

219.217.49.14

59.72.38.142

59.73.38.25

219.217.49.14

#! /bin/bash

# test.sh

sort ip\_input | uniq -c | sort | tail -1 | awk '{print $2}' 或者：

sort ip\_input | uniq -c | sort -r | head -1 | awk '{print $2}'

uniq -c 选项。它的作用是将重复行去掉并且在每行的前面添加此行重复的次数

sort命令的默认排序方式是升序，使用 -r 选项逆序排序结果，则出现次数最多的IP则排在了第一行

[root@web ~]# cat 1.log | awk '{a[$0]++}END{for(i in a)if(a[i]>max)max=a[i];print a[i],i}'

3 219.217.49.14

处理一下文件内容,将域名取出并进行计数排数,如处理:

http://www.baidu.com/index.html

http://ww.baidu.com/1.html

http://www.baidu.com/2.html

http://post.baidu.com/index.html

http://mp3.baidu.com/index.html

http://www.baidu.com/3.html

http://post.baidu.com/2.html

得到如下结果:域名的出现次数,域名

4 www.baidu.com

2 post.baidu.com

1 mp3.baidu.com

shell程序如下:

方法一:

#cat file | sed -e 's/http:\/\///' -e 's/\/.\*//'| sort | uniq -c | sort -nr | head -10

sort -n 依照数值的大小排序。

方法二:

#awk -F/ '{print $3}' file|sort -r|uniq -c|awk '{print $1 "\t",$2}'

从a.log文件中提取包含"WARNING"或"FATAL",同时不包含"IGNOR"的行,然后提取以":"分割的第5个字段.

grep -E 'WARNING|FATAL' file|grep -v 'IGNOR'|awk -F: '{print $5}'

在shell环境中,如何查看远程Linux系统运行了多少时间?

ssh user@被监控的主机ip "uptime"|awk '{print $3,$4}'

腾讯一shell试题.

假设qq.tel文件内容:

12334:13510014336

12345:12334555666

12334:12343453453

12099:13598989899

12334:12345454545

12099:12343454544

分类如下:

[12334]

13510014336

12343453453

...........

[12099]

13598989899

12343454544

............

实现如下:

cat qq.tel|sort -r |awk -F: '{if(tmp!=$1) {tmp=$1;print "["tmp"]"} print $2}'

cat 1.log |sort -nr| awk -F: '{a[$1]=a[$1]"\n"$NF}END{for(i in a)print "["i"]",a[i]}'

=======================================

[awk 多文件操作2种实现方法](http://www.cnblogs.com/chengmo/archive/2010/10/15/1851983.html)

我们经常会将2个有关联文本文件进行合并处理。分别从不同文件获取需要的列，然后，整体输出到一起。awk进行多文件处理时候，常常会遇到2个方面问题，第一个是怎么样合并多个文件为一个文件。第二个问题就是怎么样将多行合并为一行显示。我这里说下我的处理2种方法，还有实现思路。

实例文本：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | [chengmo@centos5 shell]$ awk 'FNR==1{print "\r\n"FILENAME}{print $0}' a.txt b.txt    a.txt  100     wang    man  200 wangsan woman  300 wangming man  400 wangzheng man    b.txt  100 90 80  200 80 70  300 60 50  400 70 20 |

需要合并得到结果：

100     wang    man 90  80  
200 wangsan woman 80    70  
300 wangming man 60     50  
400 wangzheng man 70    20

**awk多文件操作方法一：**

**实现思路：**

通过外部命令合并文件，然后通过排序，然后通过awk进行合并操作。

**首先：**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | [chengmo@centos5 shell]$ cat a.txt b.txt | sort -n -k1 |awk '{print}'  100 90 80  100     wang    man  200 80 70  200 wangsan woman  300 60 50  300 wangming man  400 70 20  400 wangzheng man |

现在需要把：第一列相同的处理合并到一行，这里需要用“next”语句。它操作，可以参考[awk 多行合并【next 使用介绍】（常见应用4）](http://www.cnblogs.com/chengmo/archive/2010/10/13/1850145.html)

**继续：**

|  |  |
| --- | --- |
| 1  2  3  4  5 | [chengmo@centos5 shell]$  cat a.txt b.txt | sort -n -k1 |awk  'NR%2==1{fd1=$2"\t"$3;next}{print $0"\t"fd1}'  100     wang    man     90      80  200 wangsan woman       80      70  300 wangming man        60      50  400 wangzheng man       70      20 |

需要把几行合并，经常用到方法是：NR%num 然后将行值保存下来，next该行。在输出时候打印出来。

**awk多文件操作方法二 🡺不推荐太复杂了**

**实现思路**

不借助第3放工具打开，直接通过awk 打开多个文件。然后可以通过：FILENAME获得当前处理文件名。NR总记录 FNR当前文件记录,以及ARGC传入参数总数，ARGV是数组，各个参数值。

看下这些实例：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | [chengmo@centos5 shell]$ awk 'BEGIN{print ARGC,ARGV[0],ARGV[1],ARGV[2]}{print FILENAME,NR,FNR,$0}' a.txt b.txt  3 awk a.txt b.txt  a.txt 1 1 100   wang    man  a.txt 2 2 200 wangsan woman  a.txt 3 3 300 wangming man  a.txt 4 4 400 wangzheng man  b.txt 5 1 100 90 80  b.txt 6 2 200 80 70  b.txt 7 3 300 60 50  b.txt 8 4 400 70 20 |

**程序代码：**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | [chengmo@centos5 shell]$ awk '  BEGIN{  if(ARGC<3)  {    exit 1;  }    file="";  }  {  aData[FILENAME,$1]=ARGV[1]==FILENAME?$0:$2"\t"$3;  }  END{  for(k in aData)  {      split(k,idx,SUBSEP);      if(idx[1]==ARGV[1] && (ARGV[2],idx[2]) in aData)      {          print aData[ARGV[1],idx[2]],aData[ARGV[2],idx[2]] | "sort -n -k1";      }  }  }' a.txt b.txt    100     wang    man 90  80  200 wangsan woman 80    70  300 wangming man 60     50  400 wangzheng man 70    20 |

代码说明：

这里用到2维数组，aData[文件名,关联列对应值] ,这种方法可以将多个文件内容。放入一个统一二维数组。然后循环数组，通过if((i,j} in array) 查找对应列值，在其它文件中是否存在。

以上是2种实现方法，其中第一种思路较为简单，很容易理解。第二种处理起来较为复杂。有更好方法大家给我分享。

[#2楼](http://www.cnblogs.com/chengmo/archive/2010/10/15/1851983.html#2036534) 2011-03-02 13:20 [thezwj](http://www.cnitblog.com/r.aspx?url=http://zhangwenjie.net)

gawk 'NR==FNR{a[$1]=$2;b[$1]=$3;};NR>FNR&&a[$1]{print $0, a[$1],b[$1]}' b.txt a.txt

[#3楼](http://www.cnblogs.com/chengmo/archive/2010/10/15/1851983.html#2706786) 2013-06-17 17:19 [li0924](http://www.cnblogs.com/lottu/) 

我也提供两种方法：  
1.awk 'BEGIN{OFS="\t"}NR==FNR{a[$1]=1;b[$1]=$2;c[$1]=$3;next}{if(a[$1]==1)print $1,b[$1],c[$1],$2,$3}' a.txt b.txt  
2. join a.txt b.txt  
对方法2；有点约束性；首先是排好序--[sort]命令；不然也实现不了！

[#4楼](http://www.cnblogs.com/chengmo/archive/2010/10/15/1851983.html#2782098) 2013-09-25 17:08 [DiaoCow](http://www.cnblogs.com/DiaoCow/) 

awk '{printf("%s ", $0); getline<"b.txt" ;print $2,$3}' a.txt

=================================

**cat a.txt  
20110308    21:37:52    13058   aa      VIP1    a账户  
20110308    19:27:35    13057   bb      VIP2    b账户  
20110308    16:22:02    8710    cc      VIP3    b账户  
20110308    15:09:42    13036   dd      VIP2     c账户  
20110307    21:32:45    12997   ee      VIP3     a账户  
20110307    17:04:43    8830    ff      VIP3     a账户  
20110307    16:08:37    13033   gg      VIP8     d账户  
20110306    22:06:51    13032   hh      VIP10    e账户  
  
cat b.txt  
20110308    21:37:52    13058   
20110308    19:27:35    13057   
20110308    16:22:02    8710    
20110306    22:06:51    130323  
20110308    15:09:42    13036   
20110306    22:06:51    130351  
20110307    21:32:45    12997   
20110307    17:04:43    8830    
20110307    16:08:37    13033   
20110306    22:06:51    13032   
  
问大家个shell问题，有两个文件 a.txt、b.txt  
我想实现的是，a.txt中 第三列中，在b.txt中存在，则把a.txt最后两列追加到 b.txt对应第三列的的后面。**

**cat a.log b.log | sort -n -k3 |awk  'NR%2==1{f=$1"\t"$2"\t"$3;next;}{print f"\t"$5"\t"$6}'**

awk 'NR==FNR{a[$3]=$(NF-1)" "$NF;next}{($3 in a)?$3=$3" "a[$3]:$3=$3}1' a.log b.log | column –t

=============================================

在awk进行文本处理时候，我们可能会遇到。将多行合并到一行显示问题。 有点象sql里面，经常遇到的行转列的问题。 这里需要用到next语句。

**awknext语句使用**：**在循环逐行匹配，如果遇到next,就会跳过当前行，直接忽略下面语句。而进行下一行匹配。**

text.txt 内容是：

a  
b  
c  
d  
e

[chengmo@centos5 shell]$ awk 'NR%2==1{next}{print NR,$0;}' text.txt       
2 b  
4 d

当记录行号除以2余 1，就跳过当前行。下面的print NR,$0也不会执行。 下一行开始，程序有开始判断NR%2 值。这个时候记录行号是：2 ，就会执行下面语句块：'print NR,$0'

**awk next使用实例：**

**要求：**

文件：text.txt 格式：  
web01[192.168.2.100]

httpd            ok  
tomcat               ok  
sendmail               ok  
web02[192.168.2.101]  
httpd            ok  
postfix               ok  
web03[192.168.2.102]  
mysqld            ok  
httpd               ok

需要通过awk将输出格式变成：

web01[192.168.2.100]:   httpd            ok  
web01[192.168.2.100]:   tomcat               ok  
web01[192.168.2.100]:   sendmail               ok  
web02[192.168.2.101]:   httpd            ok  
web02[192.168.2.101]:   postfix               ok  
web03[192.168.2.102]:   mysqld            ok  
web03[192.168.2.102]:   httpd               ok

**分析：**

分析发现需要将包含有“web”行进行跳过，然后需要将内容与下面行合并为一行。

[chengmo@centos5 shell]$ awk '/^web/{T=$0;next;}{print T":\t"$0;}' test.txt

web01[192.168.2.100]:   httpd            ok  
web01[192.168.2.100]:   tomcat               ok  
web01[192.168.2.100]:   sendmail               ok  
web02[192.168.2.101]:   httpd            ok  
web02[192.168.2.101]:   postfix               ok  
web03[192.168.2.102]:   mysqld            ok  
web03[192.168.2.102]:   httpd               ok

next在多行合并，以及选择性输出方面，非常方便。大家在使用时候不妨试试。

=========================================================

文件passwd:   
s2002408030068:x:527:527::/home/dz02/s2002408030068:/bin/pw   
s2002408032819:x:528:528::/home/dz02/s2002408032819:/bin/pw   
s2002408032823:x:529:529::/home/dz02/s2002408032823:/bin/pw  
  
文件shadow:   
s2002408030068:$1$d8NwFclG$v4ZTacfR2nsbC8BnVd3dn1:12676:0:99999:7:::   
s2002408032819:$1$UAvNbHza$481Arvk1FmixCP6ZBDWHh0:12676:0:99999:7:::   
s2002408032823:$1$U2eJ3oO1$bG.eKO8Zupe0TnyFhWX9Y.:12676:0:99999:7:::   
  
用shadow文件中的密文部分替换passwd中的"x",生一个新passwd文件,如下所示  
s2002408030068:$1$d8NwFclG$v4ZTacfR2nsbC8BnVd3dn1:527:527::/home/dz02/s2002408030068:/bin/pw   
s2002408032819:$1$UAvNbHza$481Arvk1FmixCP6ZBDWHh0:528:528::/home/dz02/s2002408032819:/bin/pw   
s2002408032823:$1$U2eJ3oO1$bG.eKO8Zupe0TnyFhWX9Y.:529:529::/home/dz02/s2002408032823:/bin/pw   
=================  
awk -F ":" -vOFS=":" 'NR==FNR{a[$1]=$2;next}{($1 in a)?$2=a[$1]:$2=$2}1' shadow passwd   
s2002408030068:$1$d8NwFclG$v4ZTacfR2nsbC8BnVd3dn1:527:527::/home/dz02/s2002408030068:/bin/pw   
s2002408032819:$1$UAvNbHza$481Arvk1FmixCP6ZBDWHh0:528:528::/home/dz02/s2002408032819:/bin/pw   
s2002408032823:$1$U2eJ3oO1$bG.eKO8Zupe0TnyFhWX9Y.:529:529::/home/dz02/s2002408032823:/bin/pw

=============================================================

cat file1:   
0011AAA 200.00 20050321   
0012BBB 300.00 20050621   
0013DDD 400.00 20050622   
0014FFF 500.00 20050401   
  
cat file2:   
I0011  11111   
I0012  22222   
I0014  55555   
I0013  66666   
  
规则：比较 file1的1-4字符 和 file2的2-5 字符，如果相同，将file2 的第二列 与 file1 合并 file3

[root@web ~]# cat file1 | awk 'NR==FNR{a[$1]=substr($1,1,4);b[$1]=$0;next}{v=substr($1,2,5);for(i in a){if(a[i]==v){print b[i],$2}}}' file1 file2

0011AAA 200.00 20050321 11111

0012BBB 300.00 20050621 22222

0014FFF 500.00 20050401 55555

0013DDD 400.00 20050622 66666

=============================

2015-12-20|2016-01-01

2016-05-20|2016-05-12

2016-06-28|2016-05-31

2016-03-28|2016-07-31

大神们，求助！！第一列减去第二列得到相差几天，awk怎么实现？

[root@web ~]# cat 2.log | awk -F\| '{"date +%s -d "$1""|getline date1;"date +%s -d "$2""|getline date2;print (date1-date2)/86400}'

-12

8

28

-125

[root@web ~]# cat 2.log | awk -F\| '{"date +%s -d" $1|getline date1;"date +%s -d" $2|getline date2;print (date1-date2)/86400}'

===================

[root@web ~]# cat 1.log

drwxr-xr-x.

d-x-x-x.

drwxr-xr-x.

drwxr-xr-x.

-rw-r--r--.

-rw-r--r--.

-rw-r--r--.

drwxr-xr-x.

drwxr-xr-x.

[root@web ~]# cat 1.log | awk '{j=0;split($1,a,"");for(i=1;i<=length(a);i++){if(a[i]=="r")j++}print NR,j}'

1 3

2 0

3 3

4 3

5 3

6 3

7 3

8 3

9 3

===============================================

1. cat file
2. 888    2013/11/25 9:11
3. 888    2013/11/25 12:12
4. 888    2013/11/25 13:29
5. 888    2013/11/25 13:33
6. awk '{"date +%W -d" $2|getline week;"date +%j -d" $2|getline day;print week,day}' file
7. 47 329
8. 47 329
9. 47 329
10. 47 329

========================

文件a

220 34 50 70  
553 556 32 21  
1 1 14 98 33

文件b

10  
8  
2

要求文件a的每行数据与文件b的相对应的行的值相减，得到其绝对值。

[root@web ~]# awk '{"cat b.log"|getline var;for(i=1;i<=NF;i++){$i>var?$i=$i-var:$i=var-$i}}1' a.log

210 24 40 60

545 548 24 13

1 1 12 96 31

[root@web ~]# awk 'NR==FNR{ b[NR]=$1;next }{for(i=1;i<=NF;i++){a[FNR,i]=$i;a[FNR,i]>b[FNR]?$i=a[FNR,i]-b[FNR]:$i=b[FNR]-a[FNR,i]}print $0}' b.log a.log

awk 'ARGIND==1{b[NR]=$1;next}{for(i=1;i<=NF;i++){a[FNR,i]=$i;printf a[FNR,i]-b[FNR]>=0?a[FNR,i]-b[FNR]" ":b[FNR]-a[FNR,i]" "}printf "\n"}' b a

============================

文件a

aaa  
bbb  
ccc  
ddd

文件b

111 xxx  
222 xxx  
333 xxx  
444 xxx

要求文件a里的数据依次替换文件b中的xxx字样。

[root@web ~]# awk '{getline v<"a.log";for(i=1;i<=NF;i++){if(i==2)$i=v};print $0 }' b.log

111 aaa

222 bbb

333 ccc

444 ddd

【总裁】北京-baby神(455416841)  23:16:41  
awk '{getline i < "a";print $1,i}' b

awk '{getline i<"a"}/xxx/{sub("xxx",i,$2)}1' b

=================================================

要求统计每个小时内的访问总和：

2011-07-20 09:57:01  
239  
2011-07-20 11:03:01  
248  
2011-07-20 10:29:01  
250  
2011-07-20 09:56:01  
255  
2011-07-20 10:45:01  
269  
2011-07-20 11:27:01  
272  
2011-07-20 10:28:01  
273  
2011-07-20 11:32:01  
274  
2011-07-20 10:44:01  
303  
2011-07-20 11:36:01  
316

[root@web ~]# cat 1.log | sed 'N;s#\n# #g' | awk -F "[ :]+" '{a[$2]+=$5}END{for(i in a){print $1 i":"$3":"$4" ",a[i]}}'  
2011-07-2009:36:01  494  
2011-07-2010:36:01  1095  
2011-07-2011:36:01  1110

[root@web ~]# cat 1.log | awk -F "[ :]+" '{i=$2;getline;a[i]+=$1}END{for(i in a)print i,a[i]}'

09 494

10 1095

11 1110

【副总】广州-R<r0060207@qq.com>  23:39:24  
awk -F : '{getline b;a[$1]+=b}END{for(i in a)print i,a[i]}' file  
【总裁】北京-baby神(455416841)  23:39:24  
awk -F ':' '/^2011/{getline x;a[$1]+=x}END{for (i in a)print i":00:00",a[i]}' a      
2011-07-20 09:00:00 494  
2011-07-20 10:00:00 1095  
2011-07-20 11:00:00 1110

awk -F':' '{i=$1;getline;a[i]+=$1}END{for(j in a)print j,a[j]}' file

[解析]

  因为数据可以看出是很有规律的一行是时间，下一样就是数值，所以把时间赋值给一个变量，然后读取到下一行，把下一行的数值累加给这个时间段为下标的数组a，最后取出结果。

awk -F':' '/:/{getline x;a[$1]+=x}END{for(i in a) print i,a[i]}' file

[解析]

  这个getline的意义完全不一样了，当匹配到有冒号的行时执行{action}，getline是读取下一行的数据给变量x，然后再根据这个时间的下标把x累加给数组a，最后取出结果。执行的结果都是一样的，大家好好揣摩一下getline的用法区别。

==============================================

[root@web ~]# cat 1.log

r9\_cz3IwGbu642/1 scaffold\_3 1

TL\_tKUFyGbu642/2 scaffold\_2 2

V4\_CUctkGbu642/3 scaffold\_30 3

B6\_SIhaqGbu642/4 scaffold\_3 4

[root@web ~]# cat 1.log| sort -k3 | awk '{while("sort -k3 1.log"|getline){print $0" ",i?$3-i:"";i=$3 }}'

r9\_cz3IwGbu642/1 scaffold\_3 1

TL\_tKUFyGbu642/2 scaffold\_2 2 1

V4\_CUctkGbu642/3 scaffold\_30 3 1

B6\_SIhaqGbu642/4 scaffold\_3 4 1

[root@web ~]# sort -nk3 1.log | awk 'NR>1{print $0,$3-b}NR==1{print $0}{b=$3}'

sort -k3 1.log|awk 'NR>1{$0=$0 FS $3-x}{x=$3}1'

=========================

cat 1.log  
1,2,3,4,5,6  
7,8,9  
10,11,12,13  
变成这样,不够补0  
1 7 10  
2 8 11  
3 9 12  
4 0 13  
5 0 0  
6 0 0  
行和列对调

[root@web ~]# awk -vnu=`awk -F, 'NR==1{print NF}' 1.log` -F, '{for(i=1;i<=nu;i++)if($i==""){$i=0;a[i]=a[i]?a[i]" "$i:$i}else{a[i]=a[i]?a[i]" "$i:$i}}END{for(i=1;i<=nu;i++)print a[i]}' 1.log

[root@web ~]# awk -F, '{l=l>NF?l:NF;for(i=0;i++<l;)a[i]=i>NF?a[i]","0:a[i]?a[i]","$i:$i}END{for(i=0;i++<l;)print a[i]}' 1.

=================================

1  
2  
3  
4  
5  
6  
7  
8  
9  
10 把/etc/passwd中20到30行插入到数值5之下,其中插入的20到30中没有第25行  
题意明白了吧

[root@web ~]# seq 5 |awk '1;/3/{system("sed '25d' /etc/passwd | sed -n '20,30p'")}'

=============================

5个字符切割

[root@web ~]# echo "aaaaaaaaaaaaaaaaaaa" |awk --re-interval -F "" 'gsub(".{5}","&\n")'

aaaaa

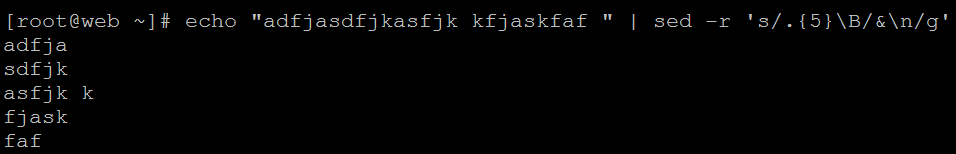
aaaaa

aaaaa

aaaa

echo 1234567 | awk -F '' '{for(i=1;i<=length($0);i++){if(i%5!=0)printf $i;else print $i}}'

echo 1234afdafasdfsa567 | awk -F '' '{for(i=1;i<=length($0);i++){if(i%5!=0)printf $i;else print $i}printf "\n"}'



[root@web ~]# echo "aaaaabcehijk" |grep -Po '.{5}|.{1,4}$'

aaaaa

bcehi

jk

================================

[root@web ~]# echo "1234567890 123" | sed -r ':1;s/([^\n]{5})([^\n])/\1\n\2/;t1'

12345

67890

123

[root@web ~]# echo "00ac45d5082f" | sed -r 's/.{5}/&\n/g'

00ac4

5d508

2f

=======================

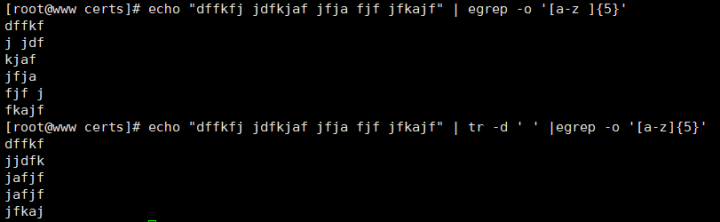
echo aaaaaaaaaabcehijk | sed -n 'l6'

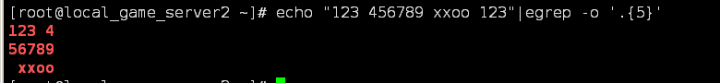
小写L

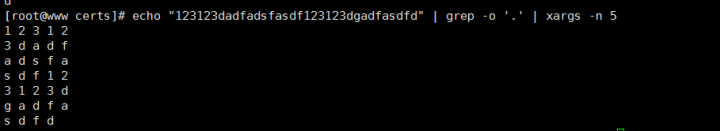
l=list，打印模式空间的内容，用于测试的

sed -n 'l6'|sed -r 's#\\|\$##g'

sed -n 'l6'| sed 's/.$//'







==================================

echo "11111" | awk '{print $0=gensub("1","x",4)}'  
111x1  
  
echo "11111" | awk '{print $0=gensub("1","x","g")}'  
xxxxx  
  
  
echo "unix linux" | awk '{print gensub(/(.+) (.+)/,"\\2 \\1","g")}'  
linux unix  
  
echo "xaax xbx xxx:xaax xbx xxx" | awk -F: -vOFS=":" '{$2=gensub(/x([^x]+)x/,"\\1YY",2,$2)}1'  
xaax xbx xxx:xaax bYY xxx  
这是gensub的例子

【总裁】北京-baby神(455416841)  20:58:38  
echo 1234 | awk '{gsub(/123/,"&hehe")}1'  
123hehe4

[root@web ~]# echo 1234dfdfaaf123 | awk '{print gensub(/(123)/,"&hehe","g",$0)}'

123hehe4dfdfaaf123hehe

[root@web ~]# echo 1234dfdfaaf123 | awk '{print $0=gensub(/(123)/,"&hehe","g",$0)}'

123hehe4dfdfaaf123hehe

echo "2016-03-17 21:06:00"|awk '{print $0=gensub(":|-"," ","g",$0)}

[:|-]

总监】黑郁/wx金香(119794881) 20:40:36

@浙江-带头大哥 有区别。[:|-]的意思是“:”或“|”或“-”

:|- 的意思是“:”或“-”

|  |
| --- |
| 【总裁】北京-baby神(455416841) 20:40:53  seq 4 | awk '$0~"2"' 2 |
| 【总裁】北京-baby神(455416841) 20:41:04  跟这种写法有点像 |

加//和加""一样

===============================

文件内容：  
8613380308589|460110410761112|2016-03-17 21:06:00|460110418680672|2013-06-19 10:28:13  
8613380343213|460110410353535|2015-03-17 21:06:00|460110446352525|2013-06-18 11:28:13  
8617722833295|460110418713828|2016-01-01 09:07:15|460110418713826|2016-01-01 09:07:15|460110418713821|2016-03-01 09:07:15|4601104184242424|2016-06-01 19:07:15  
输出规则：  
每单独一行的时间字段进行比较，如果时间大，则返回第一个字段加时间字段前一列的460开头的数据；（每行时间列数多少不固定）  
输出：  
8613380308589|460110410761112|2016-03-17 21:06:00  
8613380343213|460110410353535|2015-03-17 21:06:00  
8617722833295|4601104184242424|2016-06-01 19:07:15

[root@web ~]# cat 1.log | awk -F "|" '{for(i=3;i<=NF;i=i+2){"date +%s -d \""$i"\""|getline t; if(t >max)max=t;};print $1,$2,strftime("%Y-%m-%d %H%M%S",max)  }'  
8613380308589 460110410761112 2016-03-17 210600  
8613380343213 460110410353535 2016-03-17 210600  
8617722833295 460110418713828 2016-06-01 190715

* -W re-interval or --re-inerval  
  允许间隔正则表达式的使用，参考(grep中的Posix字符类)，如括号表达式[[:alpha:]]。

awk --re-interval '{max=0;for(i=3;i<=NF;i+=2){$i=gensub(":|-"," ","g",$i);m=mktime($i);if(m>max){max=m;str=$(i-1)}};print $1,str,strftime("%F %T",max)}' FS="|" OFS="|" file

优化:

[root@web ~]# cat 1.log | awk -F "|" '{for(i=3;i<=NF;i=i+2){t=mktime(gensub("-|:"," ","g",$i)); if(t >max)max=t;};print $1,$2,strftime("%Y-%m-%d %H%M%S",max) }'

8613380308589 460110410761112 2016-03-17 210600

8613380343213 460110410353535 2016-06-18 112813

8617722833295 460110418713828 2016-07-01 190715

知识点: gensub

时间函数

|  |  |
| --- | --- |
| **函数名** | **说明** |
| mktime( YYYY MM DD HH MM SS[ DST]) | 生成时间格式 |
| strftime([format [, timestamp]]) | 格式化时间输出，将时间戳转为时间字符串  具体格式，见下表. |
| systime() | 得到时间戳,返回从1970年1月1日开始到当前时间(不计闰年)的整秒数 |

创建指定时间(mktime使用）

[chengmo@centos5 ~]$ awk 'BEGIN{tstamp=mktime("2001 01 01 12 12 12");print strftime("%c",tstamp);}'  
2001年01月01日 星期一 12时12分12秒

[chengmo@centos5 ~]$ awk 'BEGIN{tstamp1=mktime("2001 01 01 12 12 12");tstamp2=mktime("2001 02 01 0 0 0");print tstamp2-tstamp1;}'  
2634468

求2个时间段中间时间差,介绍了strftime使用方法

[chengmo@centos5 ~]$ awk 'BEGIN{tstamp1=mktime("2001 01 01 12 12 12");tstamp2=systime();print tstamp2-tstamp1;}'   
308201392

strftime日期和时间格式说明符

| **格式** | **描述** |
| --- | --- |
| %a | 星期几的缩写(Sun) |
| %A | 星期几的完整写法(Sunday) |
| %b | 月名的缩写(Oct) |
| %B | 月名的完整写法(October) |
| %c | 本地日期和时间 |
| %d | 十进制日期 |
| %D | 日期 08/20/99 |
| %e | 日期，如果只有一位会补上一个空格 |
| %H | 用十进制表示24小时格式的小时 |
| %I | 用十进制表示12小时格式的小时 |
| %j | 从1月1日起一年中的第几天 |
| %m | 十进制表示的月份 |
| %M | 十进制表示的分钟 |
| %p | 12小时表示法(AM/PM) |
| %S | 十进制表示的秒 |
| %U | 十进制表示的一年中的第几个星期(星期天作为一个星期的开始) |
| %w | 十进制表示的星期几(星期天是0) |
| %W | 十进制表示的一年中的第几个星期(星期一作为一个星期的开始) |
| %x | 重新设置本地日期(08/20/99) |
| %X | 重新设置本地时间(12：00：00) |
| %y | 两位数字表示的年(99) |
| %Y | 当前月份 |
| %Z | 时区(PDT) |
| %% | 百分号(%) |

strftime

将时间戳转换成“年月日时分秒”的格式

awk '{print strftime("%Y%m%d%H%M%S",$1)}'

strftime 将时间戳转日期  
awk 'BEGIN{print strftime("%Y-%m-%d",systime())}'

将日期转为时间戳  
awk 'BEGIN {printf("%d\n",mktime(2006" "8" "5" "15" "09" "0))}'

===============

有这么一个文本a.txt,内容如下  
20160101|42012319891122492X|602|10000|100|10000|  
20160101|42012319891122492X|605|10000|200|20000|  
20160101|420123197911215144|612|10000|30000|200|  
1、红色部分为关键列  
2、要求前两列关键列相同的，第三列求最小的值，后面的列求和  
3、这只是个例子，实际上有N行、N列所以脚本不能写死  
4、速度不能太慢（几分钟之内）  
  
结果：  
  
20160101|42012319891122492X|602|20000|300|30000|  
20160101|420123197911215144|612|10000|30000|200|

awk -F '|' '{b[$1"|"$2] = 0 ; if (a[$1"|"$2,3] == 0 || $3 < a[$1"|"$2,3]) a[$1"|"$2,3]=$3; for(i=4;i<+NF;i++) a[$1"|"$2,i] += $i} END { for (i in b) {printf(i"|");for(j=3;j<NF;j++) printf(a[i,j]"|" );printf"\n"}}' 1.log

==========================

[root@web ~]# awk -vRS='//\n' -F '\n' '$0{for(i=3;i<NF;i++)print $1,$2,$i}' 1.log

AC MIPF0002122 ID mir-9457 MI MI0029079 nve-mir-9457-1

AC MIPF0002122 ID mir-9457 MI MI0029080 nve-mir-9457-2

AC MIPF0002123 ID MIR9552 MI MI0029296 bra-MIR9552a

AC MIPF0002123 ID MIR9552 MI MI0029297 bra-MIR9552b

[root@web ~]# awk -vRS='//\n' -F '\n' '$0{for(i=3;i<NF;i++)print $1,$2,$i}' 1.log

AC MIPF0002122 ID mir-9457 MI MI0029079 nve-mir-9457-1

AC MIPF0002122 ID mir-9457 MI MI0029080 nve-mir-9457-2

AC MIPF0002123 ID MIR9552 MI MI0029296 bra-MIR9552a

AC MIPF0002123 ID MIR9552 MI MI0029297 bra-MIR9552b

【副总】广州-R<r0060207@qq.com> 11:22:53

@龙啸九天 awk -vRS='//\n' -F '\n' '$0{for(i=2;i++<NF-1;)print $1,$2,$i}'  file

============================================

[root@centos ~]# echo "" |awk '{print "I am:",v1,"he is:",v2}' v1=$var1 v2=$var2

I am: tom he is: tom

[root@centos ~]# awk 'BEGIN{print "I am:",v1,"he is:",v2}' v1=$var1 v2=$var2

I am: he is:

总结:在后面定义的变量,对BEGIN里不用引用到变量的值,而主体部分是可以引用到其值

[root@centos ~]# awk -vv1=hello -vv2=world 'BEGIN{print "I am:",v1,"he is:",v2}'

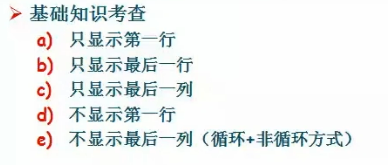
I am: hello he is: world

[root@centos ~]# echo "" |awk -vv1=hello -vv2=world '{print "I am:",v1,"he is:",v2}'

I am: hello he is: world

总结:通过申明变量这种方式BEGIN里面或者主体里面都是可以获得变量的值

===================================



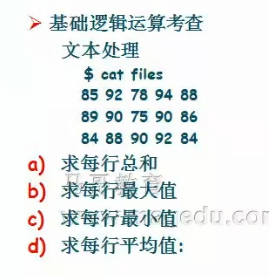
a) awk 'NR==1{print $0}' /etc/passwd

b) awk 'END{print $0}' /etc/passwd

c) awk -F: '{print $NF}' /etc/passwd

d) awk 'NR>1' /etc/passwd

e) awk -F: '{for(i=1;i<NF;i++){printf $i" ";}}{print "\n"}' /etc/passwd



a)

[root@centos ~]# awk '{for(i=1;i<=NF;i++){ sum+=$i}print sum}' 1.log

437

867

1305

b)

awk '{for(i=1;i<=NF;i++){ if($i > max){max=$i} }print max }' 1.log

c)

awk '{for(i=1;i<=NF;i++){min=100; if($i < min){min=$i} }print min }' 1.log/

d)

[root@centos ~]# awk '{for(i=1;i<=NF;i++){sum+=$i}print sum/NF}' 1.log

87.4

173.4

261



a)

column -t 1.log | awk '/sun/'

b)

column -t 1.log | awk '$2 > 40'

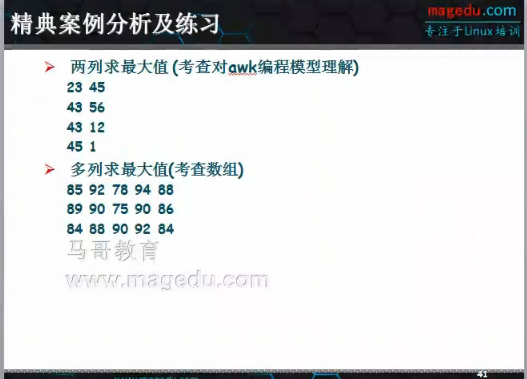
c)

column -t 1.log | awk 'length($1) > 4 && $2 > 50 '

d)

cat 1.log | awk '/^$/{++a}END{print a}'

2





a)

[root@centos ~]# cat 1.log | awk '{a[$1]=$1;b[$2]=$2}END{for(i in a){if(a[i]>max){max=a[i]}if(b[i]>max2)max2=b[i] }print max,max2 }'

56 56

b)

[root@centos ~]# cat 2.log| awk '{a[$1]++}END{for(i in a)print i,a[i]}'

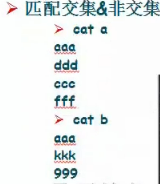
aaa 3

ccc 1

bbb 1







[root@centos ~]# awk 'NR==FNR{a[$1]=$1}{if (!($1 in a)){print $0} }' a b

kkk

999



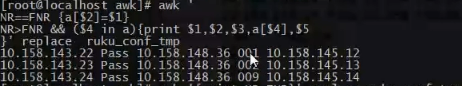
[root@centos ~]# awk 'NR==FNR{a[$2]=$1}NR>FNR{for(i in a){if($4==i) $4=a[i]};print $0}' 2.log 1.log

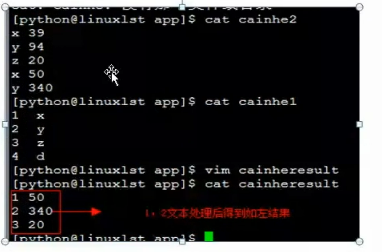
10.158.143.22 pass 10.158.148.36 001 10.158.145.12

10.158.143.23 pass 10.158.148.36 002 10.158.145.13

10.158.143.24 pass 10.158.148.36 009 10.158.145.13

awk 'NR==FNR{a[$2]=$1;next}{if($4 in a)$4=a[$4]}1' 2.log 1.log





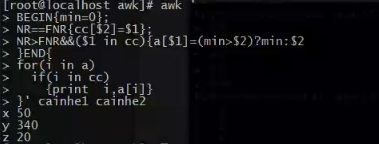
awk 'NR==FNR{a[$1]=$2;next}{if($2 in a && a[$2]>max?max=a[$2]:max=a[$2])print $1,max}' 1.log 2.log

[root@centos ~]# awk 'NR==FNR{a[$1]=a[$1]>$2?a[$1]:$2;next}NR>FNR && ($2 in a){print $1,a[$2]}' 1.log 2.log

1 50

2 340

3 20



行转列

[root@centos ~]# awk 'NF>max{max=NF}{for (i=1;i<=max;i++)a[i]=a[i]$i"\t"}END{for (i=1;i<=max;i++)print a[i]}' 1.log

1 2 3 4

2 3 4 5

3 4 5 6

4 5 6 7

5 6 7 8

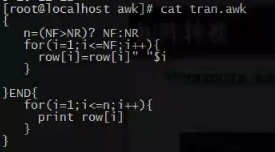
[root@centos ~]# cat 1.log

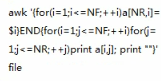
1 2 3 4 5

2 3 4 5 6

3 4 5 6 7

4 5 6 7 8









这个[shell](http://www.ha97.com/category/shell)用awk[统计](http://www.ha97.com/tag/%e7%bb%9f%e8%ae%a1)history中的命令出现的次数，去掉 “./” 这样不算命令的情况，最后排序输出，列出前20个。 我最常用的前20个命令如下：

==========================

原文:

2013-09-13 00:06:09:adid:10000254

2013-09-13 00:06:14:adid:10000254

2013-09-13 01:06:43:adid:10000253

2013-09-13 01:06:48:adid:10000254

按照小时和最后一个字段来统计:

想要的结果:

2013-09-13 00 10000254 2

2013-09-13 01 10000253 1

2013-09-13 01 10000254 2

【副总】海(675951805)  13:28:34  
[root@node1 ~]# cat test | awk -v FS='[ |:]' '{print $1,$2,$NF}' | sort -n | uniq -c  
      2 2013-09-13 00 10000254  
      1 2013-09-13 01 10000253  
      2 2013-09-13 01 10000254

[root@centos ~]# awk -F '[ :]+' '{all=$1" "$2" "$NF}{count[all]++}END{for(i in count)print i,count[i]}'

2.log

2013-09-13 01 10000253 1

2013-09-13 01 10000254 2

2013-09-13 00 10000254 2

[root@centos ~]# awk -F '[ :]+' '{all=$1" "$2" "$NF}{count[all]++}END{for(i in count)print i,count[i]}'

2.log

2013-09-13 01 10000253 1

2013-09-13 01 10000254 2

2013-09-13 00 10000254 2

【副总】海(675951805)  13:47:28  
[root@node1 ~]# cat test |  awk -F ':' '{D[$1,$NF]++}END{for(i in D) print i,D[i]}'  
2013-09-13 0110000253 1  
2013-09-13 0110000254 2  
2013-09-13 0010000254 2

【总监】浙江-带头大哥(120117282)  13:56:28  
[root@centos ~]# cat 2.log | awk -F: '{a[$1,$NF]++}END{for(i in a){split(i,c,SUBSEP);print c[1],c[2],a[i]} }'  
2013-09-13 01 10000253 1  
2013-09-13 01 10000254 2  
2013-09-13 00 10000254 2

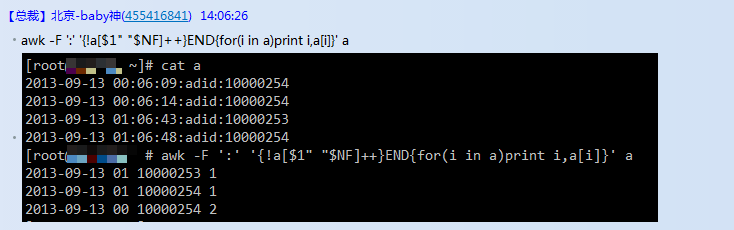
【经理】北京-悠久之翼<youjiu\_linux@qq.com>  13:56:37

awk -F "[ :]" '{array[$1" "$2" "$NF]++}END{for(key in array)print key,array[key]}' sort\_multi.txt

2013-09-13 01 10000253 1

2013-09-13 01 10000254 1

2013-09-13 00 10000254 2



===================================================

[root@centos ~]# awk -v x=5 'BEGIN{for(i=0;i++<x;){a=i\*2-1;b=a>x?x\*2-a:a;c=(x-b)/2;for(j=0;j++<c;)printf " ";for(j=0;j++<b;)printf "\*";print ""}}'

\*

\*\*\*

\*\*\*\*\*

\*\*\*

\*

[root@centos ~]# awk -v x=20 'BEGIN{for(i=0;i++<x;){a=i\*2-1;b=a>x?x\*2-a:a;c=(x-b)/2;for(j=0;j++<c;)printf " ";for(j=0;j++<b;)printf "\*";print ""}}'

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

awk 'BEGIN{for(i=1;i<=9;i++){for(j=1;j<=i;j++){arry[i,j]=i\*j;printf "%s\*%s=%s ",i,j,arry[i,j];}print}}'

1\*1=1

2\*1=2 2\*2=4

3\*1=3 3\*2=6 3\*3=9

4\*1=4 4\*2=8 4\*3=12 4\*4=16

5\*1=5 5\*2=10 5\*3=15 5\*4=20 5\*5=25

6\*1=6 6\*2=12 6\*3=18 6\*4=24 6\*5=30 6\*6=36

7\*1=7 7\*2=14 7\*3=21 7\*4=28 7\*5=35 7\*6=42 7\*7=49

8\*1=8 8\*2=16 8\*3=24 8\*4=32 8\*5=40 8\*6=48 8\*7=56 8\*8=64

9\*1=9 9\*2=18 9\*3=27 9\*4=36 9\*5=45 9\*6=54 9\*7=63 9\*8=72 9\*9=81

[root@centos ~]# awk 'BEGIN{for(i=1;i<=9;i++){for(j=1;j<=i;j++){printf i"\*"j"="i\*j"\t"}print }}'

1\*1=1

2\*1=2 2\*2=4

3\*1=3 3\*2=6 3\*3=9

4\*1=4 4\*2=8 4\*3=12 4\*4=16

5\*1=5 5\*2=10 5\*3=15 5\*4=20 5\*5=25

6\*1=6 6\*2=12 6\*3=18 6\*4=24 6\*5=30 6\*6=36

7\*1=7 7\*2=14 7\*3=21 7\*4=28 7\*5=35 7\*6=42 7\*7=49

8\*1=8 8\*2=16 8\*3=24 8\*4=32 8\*5=40 8\*6=48 8\*7=56 8\*8=64

9\*1=9 9\*2=18 9\*3=27 9\*4=36 9\*5=45 9\*6=54 9\*7=63 9\*8=72 9\*9=81

=================================

一，类似[**数据库**](http://www.jbxue.com/db/)的left join查询

[root@centos ~]# cat 1.log

111 aaa

222 bbb

333 cccc

444 ddd

[root@centos ~]# cat 2.log

111 123 456

2 abc cbd

444 rts 786

[root@centos ~]# awk 'NR==FNR{a[$1]=$2" "$3;next}{if($1 in a)print $0,a[$1]}' 2.log 1.log

111 aaa 123 456

444 ddd rts 786

======================

二，去除重复

root@ecms mytest]# cat repea   //文件repea    
 a b    
 c d    
 e f    
 b d    
 b a    
 f e    
 1 2    
 2 1  

如果有a,b和b,a这样的情况，就删除b,a，当然数字也一样;

awk '++a[$1,$2]&&!a[$2,$1]{print $0}' 1.log

a b

c d

e f

b d

1 2

[root@centos ~]# awk '!a[$1\_$2]++&&!a[$2\_$1]++' 1.log

a b

c d

e f

b d

1 2

=================

1       2    
 2       3    
 a       b    
 3       2    
 4       1    
 3       r  

把都是数字的行，最大的那个数字加起来，第一行是2，第二行是3，每四行是3，第五行是4，总和是12

[root@centos ~]# cat 2.log| awk '$1~/[0-9]+/ && $2~/[0-9]+/{max=($1>$2)?$1:$2;a[NR]=max}END{for(i in a){sum+=a[i]} print sum}'

12

======================

一文本内容如下：

tb|cname

tb|objldn

tb|date

tb\_q|cname

tb\_q|objldn

需求：

每一行增加一个域,其值为：第一个域相同的行数

希望结果：

tb|3|cname

tb|3|objldn

tb|3|date

tb\_q|2|cname

tb\_q|2|objldn

awk 'BEGIN{FS="|";while((getline < "a")>0)a[$1]++}{print $1"|"a[$1]"|"$2}' a

[root@centos ~]# awk -F\| 'NR==FNR{a[$1]++;next}{$2=a[$1]FS$2}1' OFS=\| a a

tb|3|cname

tb|3|objldn

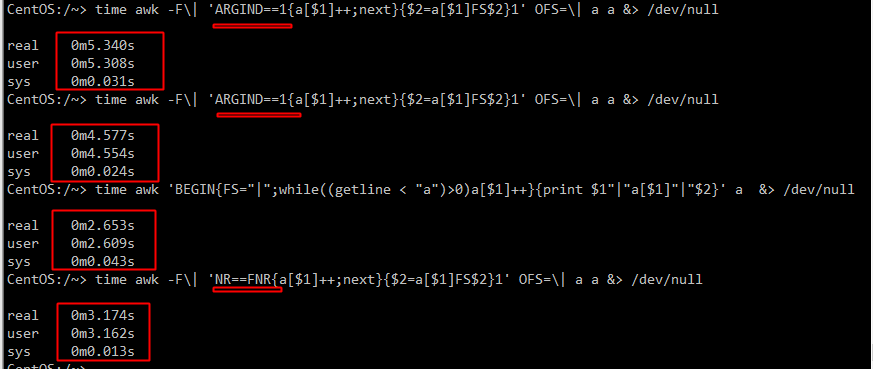
tb|3|date

tb\_q|2|cname

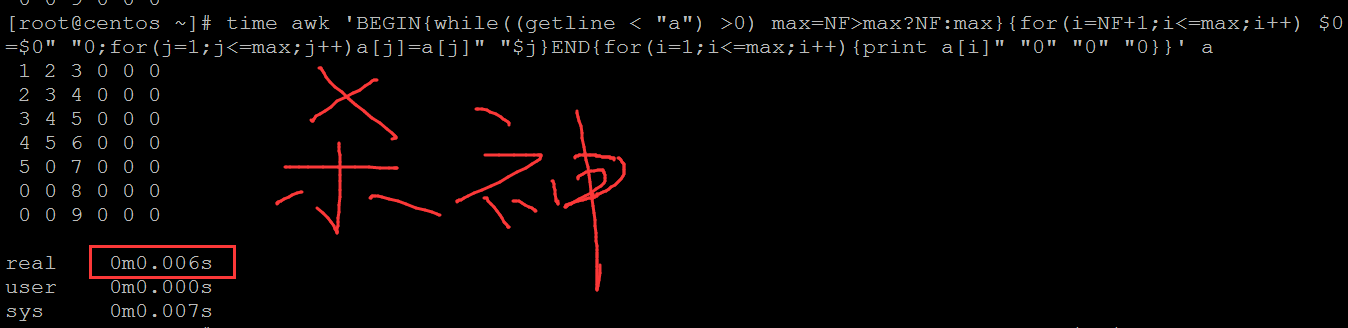
tb\_q|2|objldn

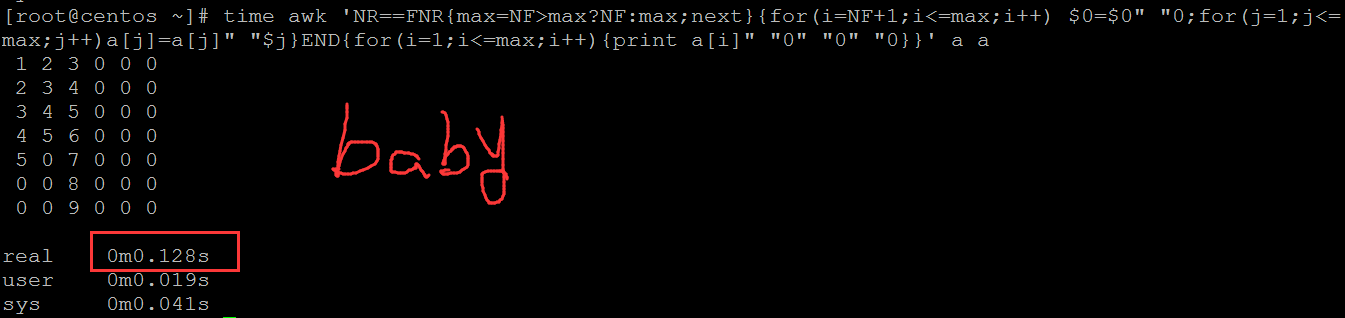
**cat m | awk -F "|" '{a[$1]++;b[$1,$2]=$0}END{for(j in a){for(k in b) {split(k,q,SUBSEP);if(j==q[1]){print q[1]"|"a[j]"|"q[2]} } }}'  
tb\_q|2|cname  
tb\_q|2|objldn  
tb|3|date  
tb|3|objldn  
tb|3|cname**

ARGING和NR的效率对比



NR要优一点





总结: NR==FNR 效率要比ARGIND 高 , getline 加 while 比读两个文件要快

=============================

1 2 3 4 5

2 3 7

2 4 2 3 5 6 8 7

3 4 5 7 8

不足的宽度补0

1 2 3 4 5 0 0 0

2 3 7 0 0 0 0 0

2 4 2 3 5 6 8 7

3 4 5 7 8 0 0 0

[root@centos ~]# cat a | awk 'BEGIN{while((getline < "a")>0) l=NF>l?NF:l}{for(i=NF+1;i<=l;i++){$0=$0" "0}print $0}'

1 2 3 4 5 0 0 0

2 3 7 0 0 0 0 0

2 4 2 3 5 6 8 7

3 4 5 7 8 0 0 0

awk 'NR==FNR{l=l>=NF?l:NF;next}{for(i=NF;++NF<=l;)$0=$0" "0}1' file file

===================

文件１：  
A 1 5  
B 2 4  
C 1 6  
D 3 7  
E 4 9  
  
文件２：  
1 2  
3 5  
4 6  
7 12  
  
文件１有３列，第１列为标示，第２列为起始值，第３列为终止值。  
文件２有２列，第２列为起始值，第２列为终止值。  
要求对文件２与文件１进行对比，如果文件１起始值到终止值之间有介于文件２中的，则输出文件２和对应标示。  
两文件对比输出结果：  
1 2 A B C  
3 5 E A B C D  
4 6 E A B C D  
7 12 E D

[root@centos ~]# awk 'NR==FNR{a[$1]=$2FS$3;next}{t="";for(i in a){split(a[i],b,FS);t=($1>b[2]||$2<b[1])?t:t" "i}print $0,t;}' a b

1 2 A B C

3 5 A B C D E

4 6 A B C D E

7 12 D E

=================

初始文本：  
port trunk allow-pass vlan 3 to 19 21 to 25 29 32 to 35 37 to 41 43 to 44 48 52 to 56 59 to 64 66 to 70  
  
目标结果：  
port trunk allow-pass vlan 3  
port trunk allow-pass vlan 4  
port trunk allow-pass vlan 5  
port trunk allow-pass vlan 6  
port trunk allow-pass vlan 7  
port trunk allow-pass vlan 8  
port trunk allow-pass vlan 9  
port trunk allow-pass vlan 10  
port trunk allow-pass vlan 11  
port trunk allow-pass vlan 12  
port trunk allow-pass vlan 13  
port trunk allow-pass vlan 14  
port trunk allow-pass vlan 15  
port trunk allow-pass vlan 16  
port trunk allow-pass vlan 17  
port trunk allow-pass vlan 18  
port trunk allow-pass vlan 19  
port trunk allow-pass vlan 21  
port trunk allow-pass vlan 2

=============

要求第三列和第四列都是从大到小的排序

[root@centos ~]# cat a

192.168.0.151 00:0F:AF:85:6C:F6

192.168.0.152 00:0F:AF:83:1F:65

192.168.0.153 00:0F:AF:85:70:03

192.168.1.1 00:0F:AF:81:19:1F

192.168.1.10 00:30:15:A2:3B:B6

192.168.1.11 00:30:15:A3:23:B7

192.168.1.12 00:30:15:A2:3A:A1

192.168.1.21 00:0F:AF:85:6C:09

192.168.1.152 00:0F:AF:83:1F:65

192.168.2.2 00:0F:AF:85:6C:25

192.168.2.20 00:0F:AF:85:55:DE

192.168.2.21 00:0F:AF:85:6C:09

192.168.2.22 00:0F:AF:85:5C:41

192.168.3.1 00:0F:AF:81:19:1F

192.168.3.2 00:0F:AF:85:6C:25

192.168.3.3 00:0F:AF:85:70:42

192.168.3.10 00:30:15:A2:3B:B6

192.168.3.12 00:30:15:A2:3A:A1

sort -t . -k 1,1n -k 2,2n -k 3,3n -k 4,4n

[root@centos ~]# cat a| sort -t. -k 3,3n -k 4.1,4.3n

192.168.0.151 00:0F:AF:85:6C:F6

192.168.0.152 00:0F:AF:83:1F:65

192.168.0.153 00:0F:AF:85:70:03

192.168.1.1 00:0F:AF:81:19:1F

192.168.1.10 00:30:15:A2:3B:B6

192.168.1.11 00:30:15:A3:23:B7

192.168.1.12 00:30:15:A2:3A:A1

192.168.1.21 00:0F:AF:85:6C:09

192.168.1.152 00:0F:AF:83:1F:65

192.168.2.2 00:0F:AF:85:6C:25

192.168.2.20 00:0F:AF:85:55:DE

192.168.2.21 00:0F:AF:85:6C:09

192.168.2.22 00:0F:AF:85:5C:41

192.168.3.1 00:0F:AF:81:19:1F

192.168.3.2 00:0F:AF:85:6C:25

192.168.3.3 00:0F:AF:85:70:42

192.168.3.10 00:30:15:A2:3B:B6

192.168.3.12 00:30:15:A2:3A:A1

================

1 2 3 4 5  
2 3 4 5  
3 4 5 6 7 8 9  
  
行和列对调，不足补0，位不足也补0  
变成  
1 2 3 0 0 0 0  
2 3 4 0 0 0 0  
3 4 5 0 0 0 0  
4  5 6 0 0 0 0  
5 0 7 0 0 0 0  
0 0 8 0 0 0 0  
0 0 9 0 0 0 0

awk 'ARGIND==1{max=max>NF?max:NF}ARGIND==2{for(i=1;i<=max;++i)a[FNR,i]=$i?$i:0}END{for(i=1;i<=max;++i){for(j=1;j<=max;++j)printf (a[j,i]?a[j,i]:0) FS;print b}}' file file

[root@centos ~]# awk 'BEGIN{while((getline < "a") >0) max=NF>max?NF:max}{for(i=NF+1;i<=max;i++) $0=$0" "0;for(j=1;j<=max;j++)a[j]=a[j]" "$j}END{for(i=1;i<=max;i++){print a[i]" "0" "0" "0}}' a  
 1 2 3 0 0 0  
 2 3 4 0 0 0  
 3 4 5 0 0 0  
 4 5 6 0 0 0  
 5 0 7 0 0 0  
 0 0 8 0 0 0  
 0 0 9 0 0 0

=====================

**源文件：  
# cat xx  
1.0000  
20020  
3.0230  
56.020  
0.23450  
0.23  
7.001  
10.10  
10.101  
0.00000  
  
目标文件：  
1  
20020  
3.023  
56.02  
0.2345  
0.23  
7.001  
10.1  
10.101  
0**

[root@centos ~]# cat 1.log | awk '{print +$0}' 相当于0+$0 此时也可以写成是$0/1

1

20020

3.023

56.02

0.2345

0.23

7.001

10.1

10.101

0

下面二个有问题,20020 后面的一个0没有了

总裁】北京-村里来的Linuxer

sed -r 'sS([[:digit:]]\*)\.{0,1}(0+)$S\1S'

【总监】郑州-小白(573758972) 21:24:30

[root@m01 ~]# sed -r 's@(\.0+|0+)$@@g' a.txt

【经理】雨夜人(1085514029) 21:25:25

sed -r '/\./s/\.?0+$//'

副总】上海--杀神(1042412709) 21:39:25

grep -Po '^\d+\.\d+(?<!0)|^\d+$|\d+(?=\.)' file

============================

[awk打印除第二/五列 之外的 所有列？](http://bbs.chinaunix.net/thread-3603410-1-1.html)

awk '{$2=$5=""}1' file

=============================

======================

select a,b,c from A where t1='a' and t2='b' and t3='c';

select d,' ',e from A where t1='d' and t2='L2' and t3='e';

select ' ',f,g from A where t1='L1' and t2='f' and t3='g';

select h,i,' ' from A where t1='h' and t2='i' and t3='L3';

源文件内如：

a||b||c

d||||e

||f||g

h||i||

即：源文件有3个域，若是第一个域$1有空的，则select 后面取空值，where条件中对应的值替换为特定的字符L1，非空则原字符替换输出；同样对于第二个域$2是空的，则select 后面取空值，where条件中对应的值替换为特定的字符L2，非空则原字符替换输出；

=============

\047是单引号    \012是换行符

输出单引号

[root@centos ~]# awk 'BEGIN{print "'\''"}'   
'  
[root@centos ~]# awk 'BEGIN{print "\x27"}'

[root@centos ~]# awk 'BEGIN{print "\047"}'

'

==============

3 5 6 7  
2 3 1 0  
4 5 6 9  
2 3 4 4  
2 2 1 0  
4 5 0 9  
假如把第2列和第3列的值之合作为新的第5列，第5列的平均值为avg5,求第5列中大于avg5的行数。

[root@centos ~]# cat 1 | awk '{a[$0]+=$2+$3;f+=$2+$3}END{avg=f/NR;for(i in a){if(a[i]>avg)print i,a[i]} }'

2 3 4 4 7

3 5 6 7 11

4 5 6 9 11

[root@centos ~]# cat 1 | awk '{a[$0]+=$2+$3;f+=$2+$3}END{avg=f/NR;for(i in a){if(a[i]>avg)m++}print m }'

3