**awk**

**awk  数据过滤软件（grep），统计**

**逐行处理**

**命令格式：**

**awk 【选项】 '【条件】{指令} '   文件....                   ##  print打印 是最常用的指令**

**常用命令选项：**

**-F:   指定分隔符，可省略（默认空格或tab位）**

**-v:   调用外部shell变量**

**[root@server0 ~]# awk  '{print $1}' /etc/passwd           ###打印出所有行的内容**

**root:x:0:0:root:/root:/bin/bash  
bin:x:1:1:bin:/bin:/sbin/nologin**

**[root@server0 ~]# awk -F: '{print $1}' /etc/passwd      ###打印出每行的第一列**

**root  
bin**

**[root@server0 ~]# awk -F: '/bash$/{print $1}' /etc/passwd             ### 打印出以“bash”结尾的行的第一列内容  
root  
student**

**示例：查看根分区的可用容量**

**[root@server0 ~]# df -h**  
**[root@server0 ~]# df -h | awk '/vda1/{print $4}'**                **###  $4 代表第四列**  
**7.1G**  
**示例：查看内存的可用容量**  
**[root@room9 桌面]# free | awk '/Mem/{print $4}'**  
**1334920**  
**示例：查看网卡的流量**  
**[root@server0 ~]# ifconfig eth0 | awk '/RX p/{print $5}'**  
**313482**  
**案例：过滤登陆失败的IP地址 （/var/log/secure)**  
**[root@server0 ~]# awk '/Failed p/{print $11}' /var/log/secure**  
**172.25.0.10**

**awk 内置变量**

**$0   一整行**

**$1   第一列**

**$2   第二列**

**NF   当前行有几列？**

**NR**   **当前行的行号**

**示例：**

**[root@server0 ~]# awk -F: '/^root/{print $0}' /etc/passwd**            **###打印以root开头的行的所有内容**  
**root:x:0:0:root:/root:/bin/bash**  
**[root@server0 ~]# awk -F: '/^root/{print $1}' /etc/passwd**              **###打印以root开头的行的第一列**      
**root**  
**[root@server0 ~]# awk -F: '/^root/{print NF}' /etc/passwd**                **###打印以root开头的所在行一共有多少列**  
7  
**[root@server0 ~]# awk -F: '/root/{print NR}' /etc/passwd**              **###打印有root的所在行的行号**  
**1**  
**10**  
**[root@server0 ~]# awk '{print "nb"}' /etc/passwd                          ###有多少行就会执行打印多少行的nb**  
**nb**  
**nb**  
**[root@server0 ~]# awk -F: '{print "用户名:",$1,"UID:",$3}' /etc/passwd**       **###混搭常量与变量**  
**用户名: root UID: 0**  
**用户名: bin UID: 1**  
  
**awk过滤的时机**

**awk 'BEGIN{指令}   条件{指令}   END{指令}'    文件....**  
**1)  BEGIN里的命令，在读取文件之前执行，执行1次；**

**2)  {}里的命令，在读取文件过程中执行，执行n次；**

**3)  END里的命令，在读完文件之后执行，执行1次。**  
**示例：**

**[root@server0 ~]# awk 'BEGIN{print "hello"}' /etc/passwd**  
**hello**  
**[root@server0 ~]# awk 'BEGIN{print "hello"}'**  
**hello**  
**[root@server0 ~]# awk 'BEGIN{x=1;y=2;print x+y;print x\*y}'**  
**3**  
**2**  
**[root@server0 ~]# awk 'BEGIN{print i}'      ### i 是未定义的变量，故输出为空**

**[root@server0 ~]# awk 'BEGIN{print x+y;print x+22}'           ###执行了运算，x y会被定义为0，故输出运算后的结果**  
**0**  
**22**  
**[root@server0 ~]# awk 'END{print NR}' /etc/passwd            ###先执行NR的命令，然后END执行**  
**39**

**[root@server0 ~]# awk 'BEGIN{x=0}/bash$/{x++}END{print x}' /etc/passwd        ###awk的统计功能，统计有多少以bash结尾的行**  
**2**  
**[root@server0 ~]# awk '/bash$/{x++}END{print x}' /etc/passwd        ###效果同上一条命令，统计有多少以bash结尾的行**  
**2**  
**案例：格式化输出/etc/passwd文件，只显示用户名、UID、宿主目录这3列，并加标题，最后输出处理的总行数**

**[root@server0 ~]# awk -F: 'BEGIN{print "ID 用户名 解释器"}{print $3,$1,$7}END{print "总用户量："NR}' /etc/passwd |column -t**  
**ID            用户名               解释器                                                           ### column -t 对齐各项**  
**0             root                 /bin/bash**  
**1             bin                  /sbin/nologin**  
**2             daemon               /sbin/nologin**

**总用户量：39**  
  
**awk处理条件**  
**条件的表现形式：**

**1）——正则表达式；（ ～ 匹配 ；！～ 不匹配）**

[root@server0 ~]# awk -F: '/root/' /etc/passwd               **#### 对整行进行匹配，只要有root就打印**  
root:x:0:0:root:/root:/bin/bash  
operator:x:11:0:operator:/root:/sbin/nologin  
[root@server0 ~]# awk -F: '$1~/root/' /etc/passwd             **####对每行的第一列进行匹配，是root才打印**  
root:x:0:0:root:/root:/bin/bash

**2）  ——数值/字符串比较**

**==等于；！=不等于；->大于；>=大于等于；-<小与；<=小与等于**

**[root@server0 ~]# awk -F: '$3>1000' /etc/passwd            ###查看普通用户的数量  （UID<=1000时，为系统用户，大于的为普通用户)**  
**[root@server0 ~]# awk -F: '$3<=1000' /etc/passwd           ###查看系统用户的数量**

**3） ——逻辑比较**

**[root@server0 ~]# awk -F: '$3>=1 && $3<10' /etc/passwd**             **###查看UID在1-10之间的所在行**  
**[root@server0 ~]# awk -F: '$3>1000 || $3<10' /etc/passwd**              **###查看UID大于1000或者小于10的所在行**

**4） ——运算符**

**支持：+-\*/% ++ -- += -= \*= /=**

**案例：把200以内能被7整出或者包含7的数找出来**

**[root@server0 ~]# seq 200 | awk '$1%7==0 || $1~/7/'**  
**案例：**

**1，从/etc/passwd 中把能登入的用户名取出来**

**2，从/etc/shadow中把他们的密码取出来**

awk -F: '/bash$/{print $1}' /etc/passwd

 awk -F: '{print $2}' /etc/shadow

**[root@server0 ~]# vim shell1.sh**  
**#! /bin/bash**  
**user=`awk -F: '/bash$/{print $1}' /etc/passwd`**  
**for i in $user ; do**  
**pass1=`grep "$i" /etc/shadow`**  
**pass2=${pass1#\*:}**  
**pass=${pass2%%:\*}**  
**echo $i $pass**  
**done**  
**[root@server0 ~]# vim shell1.sh**  
**#! /bin/bash**  
**user=`awk -F: '/bash$/{print $1}' /etc/passwd`**  
**for i in $user**  
**do**  
**awk -F: -v x=$i '$1==x{print $1,$2}'/etc/shadow**  
**done**

**awk流程控制**

**分支结构**

**单分支：——if (条件){指令}**

**双分支：——if (条件){指令1}else{指令2}**

**单分支：——if (条件){指令}else if(条件){指令2}....else{指令N}**

**案例： 判断普通用户和系统用户分别是多少**

**[root@desktop0 ~]# vim xy.sh**

**#! /bin/bash**  
**x=0**  
**y=0**  
**awk -F: '{if($3>1000){let x++}else{let y++}} END{print"普通用户"x,"系统用户"y}' /etc/passwd**

**[root@desktop0 ~]# awk -F"[ ]" '{i=1;while (i<=NF){if($i=="root"){x++};i++}} END{print "共 有"x"个root"}' test.txt**  **##注意定义 空格**

**awk 数组**

**[root@desktop0 ~]# awk 'BEGIN{a[0]=1;a[1]=2;print a[0],a[1]}'**  
**1 2**  
**遍历数组**  
**for {变量 in  数组名}{print 属组名[变量]}**

**示例：统计ip.txt里ip出现的次数**

**[root@desktop0 ~]# awk '{a[$1]++}END{for(i in a){print a[i],i}}' ip.txt**  
**6 192.168.4.2**                                  **### 次数  IP地址**  
**1 192.168.4.3**  
  
**[root@room9 ~]# ab -c 10 -n 1000 http://172.25.0.11/          ###模拟10个用户，每个用户访问1000次**

**#!/bin/bash**  
**jindu(){**  
**while :**  
**do**  
**echo -ne '\033[43m \033[0m'**  
**sleep 0.5**  
**done**  
**}**  
**jindu &**  
**cp -r $1 $2**  
**kill $!**

**案例：**

**多分割符的awk过滤(以空格和.为分割符)**

**#awk -F[:’ ’’.’] ‘{print $2,$3}’ a.txt**