17. shell基础（下）

笔记本： 优秀笔记
创建时间： 2018/12/26 21:46 **更新时间：** 2018/12/26 22:21
作者： 306798658@qq.com

shell特殊符号_cut命令

概念：cut命令用来截取某一个字段
格式：cut -d '分割字符' [-cf] n,这里的n是数字，该命令选项有如下几个：
- d 后面跟分割字符，分割字符要用单引号括起来- c 后面接的是第几个字符- f 后面接的是第几个区块

cut命令用法如下

```
[root@localhost do]# cat /etc/passwd |head -2
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
[root@localhost do]# cat /etc/passwd |head -2 |cut -d ':' -f 1
root
bin
[root@localhost do]# cat /etc/passwd |head -2 |cut -d ':' -f 1,2
root:x
bin:x
[root@localhost do]# cat /etc/passwd |head -2 |cut -d ':' -f 1-3
root:x:0

bin:x:1
```

cat passwd这个文件，head只查看前两行，cut -d截取分割符号为“：”， -f 1 表示截取第一段，1,2表示前两段，1-3表示头三段。

sort排序_wc统计行数_uniq删除重复行

sort命令介绍

sort命令用于排序
格式：sort [-t 分隔符] [-kn1,n2] [-nru]
这里-n1和n2指的是数字，其他选项如下：
-t 后面跟分割字符，作用跟cut -d选项一样，截取符号是什么；-n 表示使用纯数字排序，字母及特殊符号表示为0；-r 表示反向排序

- 例：如sort不加任何选项，他是默认升序输出，我们假设看passwd配置文件前五行有什么区别。

```
[root@localhost /]# head -n5 /etc/passwd |sort
#查看passwd前五行，输出给sort(排序)执行，按默认升序排序。
adm:x:3:4:adm:/var/adm:/sbin/nologin
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
```

> root:x:0:0:root:/root:/bin/bash

例2：把passwd配置文件前10行写到3.txt,再加几个特殊符号数字进行排序。

```
[root@localhost do]# vi 3.txt

     1 root:x:0:0:root:/root:/bin/bash
     2 bin:x:1:1:bin:/bin:/sbin/nologin
     3 daemon:x:2:2:daemon:/sbin:/sbin/nologin
     4 adm:x:3:4:adm:/var/adm:/sbin/nologin
     5 lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
     6 sync:x:5:0:sync:/sbin:/bin/sync
     7 shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
     8 halt:x:7:0:halt:/sbin:/sbin/halt
     9 mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
    10 operator:x:11:0:operator:/root:/sbin/nologin
    11 "
    12 :
    13 }
    14 _
    15 1
    16 3
    17 2
    18
~                                                    @51CTO博客
```

如图所示，我们用sort来进行排序，以数字排序 -n

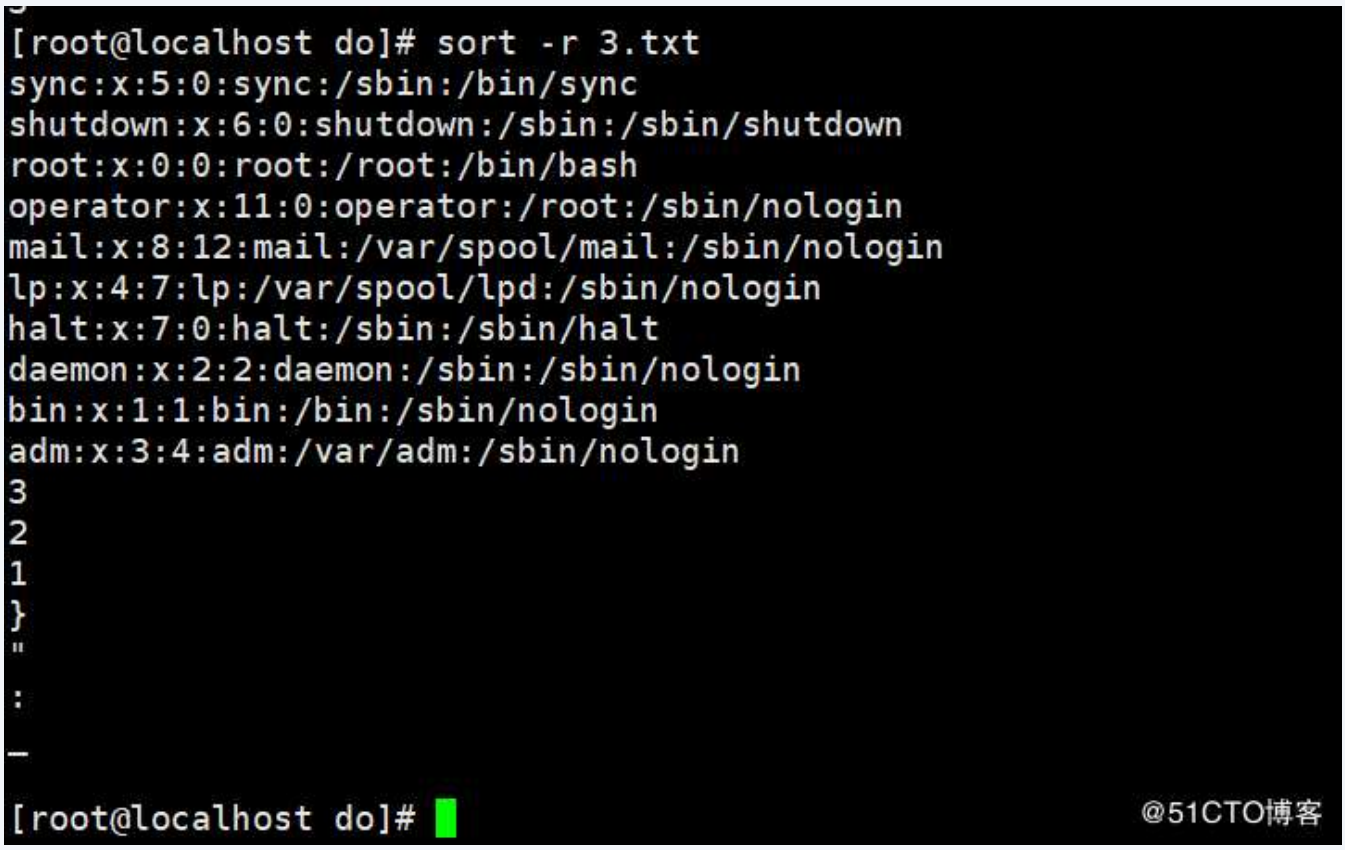
[root@localhost do]# sort -n 3.txt #-n表示正序，特殊符号及字母表示为0

```
[root@localhost do]# sort -n 3.txt


_
:
"
}
adm:x:3:4:adm:/var/adm:/sbin/nologin
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
halt:x:7:0:halt:/sbin:/sbin/halt
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
root:x:0:0:root:/root:/bin/bash
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
sync:x:5:0:sync:/sbin:/bin/sync
1
2
3
[root@localhost do]#                                 @51CTO博客
```

如第一张图所示，我们-r 给反向排序看看

[root@localhost do]# sort -r 3.txt #-r表示反序



wc 命令介绍，用于来统计

概念：wc命令用于统计文档的行数，字符数或词数。
选项：
-l 统计行数-m 统计字符数-w 统计词数，以空格作为区分

- 具体操作如下：

```
[root@localhost do]# wc /etc/passwd #查看这个文档的行数，词数及字数
19 27 846 /etc/passwd
[root@localhost do]# wc -l /etc/passwd #统计行数
19 /etc/passwd
[root@localhost do]# wc -m /etc/passwd #统计字数
846 /etc/passwd
[root@localhost do]# wc -w /etc/passwd #统计行数

27 /etc/passwd
```

命令uniq 删除重复的行

概念：uniq命令用来删除重复的行，改名了只有-c选项比较常用；
它表示统计重复的行数，并把行数写在前面。

- 编写一个文件，示例如下：

```
[root@localhost do]# touch 4.txt #创建一个文本
[root@localhost do]# vi 4.txt #编辑内容
111
222
333

111
```

- 使用uniq前，必须献给文件排序，否则不管用，示例如下：

```
[root@localhost do]# cat 4.txt #查看里面的内容
111
222
333
111
[root@localhost do]# uniq 4.txt #测试一下看看不排序管用么，不管用的
111
222
333
111
[root@localhost do]# sort 4.txt |uniq #sort排序4.txt文件，再输出给uniq删除重复行
111
222
333
[root@localhost do]# sort 4.txt |uniq -c #统计重复的有行，有多少
2 111
1 222

1 333
```

命令tee，重定向并显示内容

```
tee命令后面跟文件名，起作用类似于重定向>，但它比重定向多一个功能；
把echo输出的结果，通过管道符“|” tee输出给后面的文件并在屏幕上显示。
```

- 具体操作如下：

```
[root@localhost do]# touch 1.txt #创建一个文本
[root@localhost do]# echo "1321asd123" | tee 1.txt #把echo的内容通过管道符号，输出给tee命令，到1.txt文件，并且在屏幕显示结果。

1321asd123
```

命令tr，用于替换字符

```
tr命令用于替换字符，常用来处理文档中出现的特殊符号；
如DOS文档中出现的符号^M，该命令常用的选项如下：
-d 表示删除某个字符，后面跟要删除的字符；
-s 表示删除重复的字符。
```

- 假设把一个文件的小l，改成大L

```
[root@localhost do]# echo "linux" > 1.txt
[root@localhost do]# cat 1.txt
linux
[root@localhost do]# echo "linux" | tr '[l]' '[L]' #把前面的小l，改成L

Linux
```

命令split，切割文档

```
split命令用于切割文档，常用的选项为-b和-l
-b 表示依据大小来分割文档，默认单位为byte（字节）
-l 表示依据行数来分割文档
```

- 先搜索出来个内容，并且所有的内容cat出来，追加重定向到一个文件去

```
[root@localhost /]# find /etc/ -type f -name "*conf" -exec cat {} >> /tmp/do/1.txt \;
#find 搜索 /etc/ 下的文件 名字问conf结尾的文件，cat {}查看出来的内容,追加到1.txt文件中。
[root@localhost /]# du -sh /tmp/do/1.txt #查看这个文件的大小

212K /tmp/do/1.txt
```

- 示例1：这个文件有212K，我们给他切割指定大小为100K试试，示例如下：

```
[root@localhost do]# split -b 100K 1.txt
[root@localhost do]# ls
1.txt  2.txt  3.txt  4.txt  a.txt  xaa  xab  xac @51CTO博客
[root@localhost do]#

#这里，-b 指定带下100K，不加单位默认是以字节显示。
```

- 示例2：指定目标文件名123开头的文件。

```
[root@localhost do]# split -b 100M 1.txt 123
[root@localhost do]# ls
123aa  1.txt  2.txt  3.txt  4.txt  a.txt  xaa  xab  xac                @51CTO博客
[root@localhost do]#

#x开头的文件，是示例1的结果，不需要看他，看123开头的文件，形成一个对比。
```

shell特殊符号

```
重点章节，以后会经常用到
```

特殊符号 $

- 符号$可以用作变量前面的标识符，还可以和！结合起来使用，示例如下：

```
[root@localhost do]#
[root@localhost do]# ls /tmp/do/
123aa 1.txt 2.txt 3.txt 4.txt a.txt xaa xab xac
[root@localhost do]# !ls
ls /tmp/do/
123aa 1.txt 2.txt 3.txt 4.txt a.txt xaa xab xac

!$表示上条命令中的最后一个参数。
```

特殊符号；

- 如果想在一行中运行两个或两个以上的命令，需要在命令之间加符号；示例如下：

```
[root@localhost dior1]# mkdir 123 ; touch 1.txt ; touch 2.txt ; ls
123 1.txt 2.txt

如上含义，创建了个目录，又创建了2个文件，并且最后ls查看执行
```

特殊符号 ~

- 符号~ 表示用户的家目录，root用户的家目录是/root，普通用户是/home/username

特殊符号 &

- 如果想把一条命令放到后台执行，则需要加上符号&，它通常用于命令运行时间较长的情况，可以用在sleep（休眠），示例如下：

```
[root@localhost ~]# sleep 30 &
[1] 40966
[root@localhost ~]# jobs

[1]+ 运行中 sleep 30 &
```

重定向符号> 、>> 、2> 、 2>> 、 &> 的用法

```
概念：>、>>他们分别代表取代(>)和追加(>>)的意思；
当我们运行一个命令报错时，报错信息会输出到当前屏幕；
如果想重定向到一个文本，则需要用重定向符号2>或2>>；
他们分别表示错误重定向和错误追加重定向。
&>表示错误和正确的重定向输入到一个文件里去
```

中括号[]

- 中括号内为字符组合，代表字符组合中的任意一个。指定一个范围。（上一章节有提到）

特殊符号 &&和||

- 使用||时，表示或者，意思是说 如果两条命令用||分割的话，第一条执行成功后，第二条就不会执行，假如第一条命令是错误的话，执行失败就会执行第二条。

> 示例1，假设两个命令都是正确的
> [root@localhost do]# ls 1.txt || wc -l 2.txt
>
> 1.txt

> 示例2： 假设第一条命令是错误的，而第二条命令是正确的，就会执行第二条
> [root@localhost do]# ls 111111.txt || wc -l 2.txt #在这里 根本就没有一串1的txt文件
> ls: 无法访问111111.txt: 没有那个文件或目录
>
> 0 2.txt

- 使用&& 表示前面的命令执行成功以后，才会执行后面的命令，如果前面命令执行不成功，后面的命令就不会执行。用&&分割，用来判断的。

> 示例1，两条命令都是正确的情况下
> [root@localhost do]# ls 1.txt && wc -l 2.txt #ok，在这里两条命令都生效
> 1.txt
>
> 0 2.txt

> 示例2，假如第一条命令不成功，后面的命令就不会执行。
> [root@localhost do]# ls 111.txt && wc -l 2.txt
>
> ls: 无法访问111.txt: 没有那个文件或目录

普通特殊符号介绍

```
* 任意个任意字符
? 任意一个字符
#注释字符
\ 脱义字符
| 管道符
```