第 2 章　文件系统管理

学习引导

【本章概述】

Linux 作为开源的操作系统，支持各种各样文件系统类型，文件系统不仅包含着文件中的数据而且还有文件系统的结构，所有 Linux 用户和程序看到的文件、目录及文件保护信息等都存储在其中，是用户使用 Linux 系统的接口。

本章通过四个项目主要介绍 Linux 系统中文件和目录的常用操作命令，vim 编辑器的使用方法以及管道、输入输出重定向等概念。介绍磁盘分区以及移动存储设备的挂载方法。通过本章的学习可以进一步熟悉 Linux 系统操作的环境，掌握常用的操作命令，为后续深入的学习 Linux 系统打下良好的基础。

【本章重点与难点】

重点：

基本的目录及文件操作，vim 编辑器的使用。

难点：

管道和输入输出重定向的使用方法，分区及移动存储设备的挂载方法。

学习建议：

通过对文件、目录相关命令的操作，了解 Linux 系统的文件组织形式；

通过练习学习 vim 编辑器的使用。

2.1　项目一：Linux 下目录及文件管理

【项目描述】

Linux 操作系统安装完成后，李斯安排新进公司的网络部实习生使用 tom 账户访问 Linux 操作系统尽快熟悉 Linux 环境的相关操作，以便更好地协助李斯工作。作为 Linux 系统的初学

者，为了理解 Linux 操作系统中目录及文件的概念，掌握 Linux 环境下目录及文件的相关命令，做了如下操作：

(1)创建目录/home/tom/work1，/home/tom/work2；

(2)将当前目录切换到/home/tom/work1；

(3)显示当前路径；

(4)在/home/tom/work1 目录下生成文件 file1；

(5)显示当前目录下的所有内容(包括隐藏文件)，以确认是否生成文件 file1；

(6)将文件 file1 拷贝到目录/home/tom/work2 下；

(7)查看/home/tom/work2 目录下是否有 file1 文件，以确认拷贝操作是否成功；

(8)删除/home/tom/work1 目录下的 file1 文件；

(9)删除空目录/home/tom/work1；

(10)将 file1 文件改名为 file2；

(11)查找当前目录下所有以“file”开头的文件。

【构思设计】

该项目中涉及大量的与文件操作相关的命令，在掌握命令之前需要对文件系统的概念以及系统中目录和文件的特点做简单了解，本项目的相关知识点如表 2.1 所示。

表 2.1　　本项目相关知识点分析

序号	知识点	详见章节
1	了解 Linux 文件系统概念	2.1.1
2	了解 Linux 中目录和文件的特点	2.1.2
3	掌握文件操作相关命令	2.1.3

【实施运行】

操作步骤：

(1)根据项目描述要求在/home/tom/目录下分别创建 work1 和 work2 两个子目录，由于已经明确所要生成目录的绝对路径，所以可以通过 mkdir 命令直接生成指定的目录，执行命令：

```
$mkdir  /home/tom/work1
$mkdir  /home/tom/work2
```

需要注意的是，在生成目录时，可以使用绝对路径，也可以使用相对路径。如果只写出一个目录的名字，则新的目录将会被创建在当前目录的子目录。

(2)要进入指定的路径，可以直接用 cd 命令加绝对路径的方式进行操作，执行命令：

```
$cd  /home/tom/work1
```

(3)显示当前路径可以验证上一步操作的正确性，执行命令：

```
$pwd
```

(4)由于是在当前位置创建文件 file1，可以使用相对路径的操作方法来实现，执行命令：

```
$touch  file1
```

(5)由于要求显示包括隐藏文件在内的所有文件，因此需要添加选项-a，执行命令：

```
$ls  -a
```

(6)由于要求将当前目录下的文件拷贝到另一个目录中,因此不需要写源文件的绝对路径,只需要写出拷贝的对象名 file1 就可以,执行命令:

```
$cp  file1  /home/tom/work2
```

(7)为了验证 file1 文件拷贝是否成功,查看/home/tom/work2 目录下的内容,命令格式:

```
$ls  /home/tom/work2
```

(8)目前位于系统/home/tom/work1 路径下,所以删除目录/home/tom/work1 中的 file1 文件,执行命令:

```
$rm  file1
```

(9)删除目录/home/tom/work1,首先要切换出当前位置执行命令:

```
$cd  ..
$rm  -r  /home/tom/work1
```

(10)由于需要将文件 file1 改名成 file2,并不涉及到对文件的备份问题,所以直接进行 mv 操作,执行命令:

```
$cd  /home/tom/work2
$mv  file1  file2
```

(11)需要查找的范围是当前目录,所以不需要指明查找目录的路径。而查找所有以“file”开头的文件,则查找的依据是文件的名称,所以条件选项选择-name,执行命令:

```
$find  -name  file*
```

需要注意的是,查找结果是文件存放的相对路径。其中“./file2”,代表的是当前目录下的文件 file2。也就是说“.”代表当前目录。另外,“..”代表上层目录。如果指定从根目录下开始查找,则查找结果是文件存放的绝对路径。

2.1.1　文件系统概述

1. 关于硬盘和硬盘分区的基本概念

硬盘是计算机中用于存储信息的介质。在使用硬盘存储文件之前,需要先对硬盘进行分区、格式化操作,格式化成功以后的存储空间才是有效可用的。硬盘目前由于硬件技术的不同而分为两种:IDE 硬盘、SCSI 硬盘。通常情况下,IDE 硬盘较广泛地应用于个人 PC,而 SCSI 硬盘较广泛地应用于服务器。

硬盘的分区主要分为两种:主分区和扩展分区。

主分区是硬盘中不可再被细化分的分区,一旦划分完成以后,就以一个整体的形式被使用,而且系统引导信息一般存放于主分区中。

扩展分区也是分区的一种。只是还可以被细化分为更小的逻辑分区,而信息的存储真正是存储在逻辑分区中。

现在的硬盘技术中的硬盘分区表,只能记录四个分区信息。也即只能支持四个主分区,如果需求多于四个分区则需要创建一个扩展分区,如图 2.1 所示。

图 2.1　硬盘上的分区

2. 文件系统的基本概念

文件系统是操作系统用于明确磁盘分区上信息存储的一种格式。即文件系统规定了以什么方式在存储设备上存储数据以及如何有效地访问在存储设备上存储的数据。一个文件系统在逻辑上是一个独立的系统，被操作系统管理和控制。

Linux 系统采用了一种称为虚拟文件系统(VFS)的技术，因此 Linux 可以支持多种不同的文件系统类型，如图 2.2 所示。

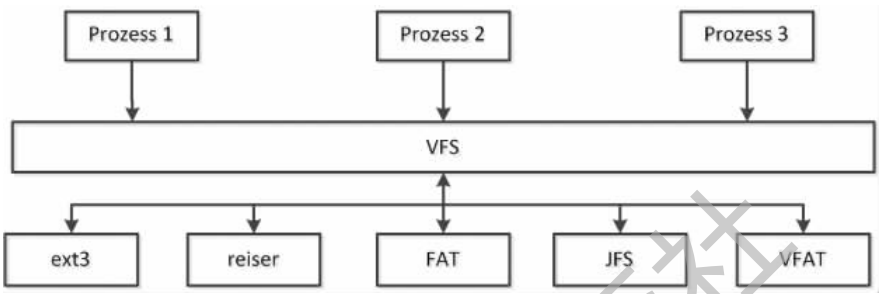

图 2.2 Linux 中的虚拟文件技术

通过 VFS，不同的进程在访问不同文件系统时，可以不需要考虑不同文件系统各自的特点，如同访问单一文件系统下的数据一样进行简单的操作即可。

在 Linux 系统中，所有的程序、系统文件和用户文件都存放在文件系统中。文件系统就是系统中所有文件和目录的集合。在 Linux 系统中，文件系统是以树型的结构来存储文件的，如图 2.3 所示。管理员可以通过设定不同的权限限制用户的访问。

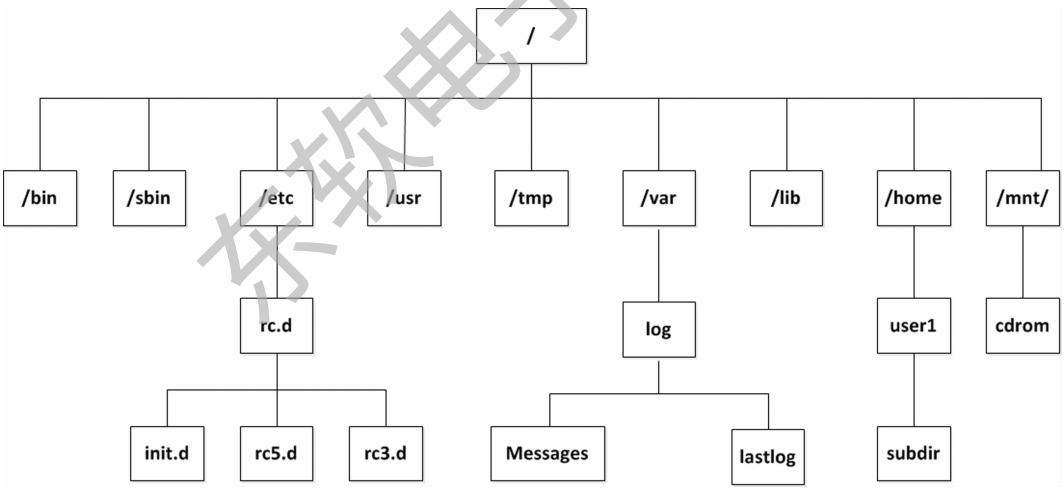

图 2.3 Linux 树形目录结构

3. 文件系统的不同类型

目前 Linux 系统中支持多种常见的文件系统类型，不仅可以使用 Linux 系统自身的文件系统类型，还可以支持使用微软、IBM 等其他操作系统平台下的文件系统类型。下面我们介绍 Linux 中常见的文件系统类型。

ext2 文件系统是 Linux 自带的文件系统类型，Linux 在早期的发行版本中使用 ext2 作为默认使用的文件系统。

ext3 是目前 Linux 中最常使用的文件系统格式，在 ext2 的基础之上发展演变而来，具有

ext2 没有的优点。其中最重要的，也是最明显的区别于 ext2 文件系统的地方是：ext3 文件系统是带有日志功能的一种文件系统格式，它会跟踪对于磁盘的写入操作并将日志记录，这样可以在需要的时候回溯查找跟踪。此时如果系统出现意外掉电重新启动后，就只需要根据日志中记录的信息，对文件系统的处理直接定位到系统出现意外以前处理的部分，而不需要像 ext2 文件系统一样对整个文件系统要进行一番从头到尾的查找，用来检测文件系统的一致性。系统恢复需要的时间少，而且更加安全可靠。

Reiserfs 是 SuSE Linux 默认的文件系统格式。在小文件和大数量的文件处理方面有比较好的处理能力，可以节约存储空间、提高访问执行效率。最重要的优于其他文件系统的方面是可以根据需要动态地分配 inode 节点，而不是在创建文件系统之初就分配完成，这样文件系统的可扩展性比较好，同时支持平衡树的访问原则，所以不论文件系统中文件有多少，访问查找时间都不会有太大的变化。

swap 文件系统在 Linux 中作为交换分区的文件系统使用。交换分区是在硬盘上分配出来的一块存储空间，系统访问该空间的时间比较短，用来弥补物理内存空间的不足。交换空间由操作系统自动管理。因为 Linux 系统有提前读和延后写的操作机制，所以在安装 Linux 系统的过程中，交换分区是必须被分配的，而且其文件系统类型就是 swap。

NFS 是指网络文件系统。在类 UNIX 系统之间进行文件共享的时候会使用到的一种文件系统类型。Window 系统中文件的共享通过简单的设置就可以实现，在 Linux 系统中进行文件共享需要进行特殊的共享设置操作。首先需要设置 NFS 服务器，将需要共享的文件目录设置共享。普通用户可以把网络中 NFS 服务器提供的共享目录挂载到本地目录中，然后可以像操作本地文件系统一样操作 NFS 文件系统中的内容。

ISO9660 是光盘文件使用的标准文件系统，在 Linux 系统中对光盘的支持通过该文件系统实现。该文件系统不仅可以支持读取操作，也支持写入操作。

2.1.2　Linux 的目录与文件介绍

1. Linux 树形目录结构

Linux 操作系统中目录的概念，类似于 Windows 操作系统中的文件夹概念。Linux 操作系统是用树形目录结构来组织和管理文件的，每个文件都有文件名，并编排在目录下，Linux 中，目录也是文件。所有的文件采取分级、分层的方式组织在一起，从而形成了一个树形的层次结构，如图 2.3 所示。

通过文件系统结构图，可以看到在 Linux 文件系统中，有不同的层次，每个层次有不同的目录。这些目录的命名、存放信息遵循一个标准：Linux 文件系统层次结构标准（File System Hierarchy Standard，FHS）。表 2.2 列出了 FHS 规定的存放特定类型的文件的对应位置。

表 2.2　　FHS 规定的文件系统布局

目录名	说明
bin	存放二进制的可执行程序，普通用户都有权限执行
sbin	类似 bin 目录，也存放可执行程序，但存放的是 system 相关的可执行程序。只有 root 可执行其中的程序修改系统属性，普通用户可以通过指定绝对路径执行其中的程序查看系统属性
boot	存放系统引导时使用的文件资源
dev	存放系统的设备配置文件

（续表）

目录名	说明
etc	存放系统配置文件
home	存放所有普通用户文件的主目录。有一个普通用户，默认情况下在该目录下就会有一个和该用户名同名的目录，当用户登录系统后就会进入其主目录
lib	存放文件系统中的程序运行所需要的共享库信息
mnt	挂载临时文件系统的挂载点的目录
root	管理员主目录
tmp	存放临时文件，普通用户正常情况下只有在自己的主目录和 tmp 目录下有写的权限，可以写临时文件在 tmp 目录下
usr	存放系统应用程序
var	存放在系统运行过程中会发生改变的文件，如系统日志等
opt	额外安装的软件包放置的位置
proc	虚拟文件系统，存放在系统内存中的信息。其中的内容不在系统中存储，每次系统启动的时候都会动态搜集，重新生成新的信息

目前所有的 Linux 系统基本都遵循这个标准。我们可以看到遵循 FHS 文件系统层次结构标准以后，在 Linux 系统中文件是归类存放的。这样在不同的 Linux 系统中用户都可以根据分类很快找到自己想要查找的文件。

(1)常用目录

/bin：存放系统所需要的那些命令，比如 ls、cp、mkdir 等命令；这个目录中的文件都是可执行的、普通用户可以使用的命令。

/dev：设备文件存储目录，比如声卡、磁盘。

/boot：这是 Linux 的内核及引导系统程序所需要的文件目录，比如 initrd.img 等文件都位于这个目录中，GRUB 系统引导管理器也位于这个目录。

/etc：系统配置文件的所在，一些服务器的配置文件也在这里；比如用户账号及密码配置文件。

/home：普通用户主目录默认存放的位置。

/lib：库文件存放目录，普通用户无权限执行这个目录下的命令，这个目录和/usr/sbin；/usr/local/sbin目录是相似的。凡是目录 sbin 中包含的都是 root 权限才能执行的。

/tmp：临时文件目录，有时用户运行程序的时候，会产生临时文件。这个目录和/var/tmp 目录相似。

/usr：这个是系统存放程序的目录，比如命令、帮助文件等。当安装一个 Linux 发行版官方提供的软件包时，大多安装在这里。如果有涉及服务器配置文件的，会把配置文件安装在/etc 目录中。/usr 目录下包括设计字体目录/usr/share/fonts，帮助目录/usr/share/man 或/usr/share/doc，普通用户可执行文件目录/usr/bin 或/usr/local/bin；超级权限用户 root 可执行命令存放目录，比如/usr/sbin 或/usr/local/sbin 等，还有程序的头文件存放目录/usr/include。

/var：这个目录的内容是经常变动的，/var 下有/var/log 是用来存放系统日志的目录。

/var/lib 用来存放一些库文件，比如 MySQL。

(2)重要的子目录

/etc/init.d：用来存放系统或服务器启动的脚本。

/etc/X11：X-Window 相关配置文件存放地。

/usr/bin：存放可执行程序的目录，普通用户有权限执行；当从系统自带的软件包安装一个程序时，其可执行文件大多会放在这个目录。

/usr/sbin：存放可执行程序的目录，只有 root 权限才能执行。

/usr/local：存放用户自编译安装软件，一般是通过源码包安装的软件，如果没有特别指定安装目录的话，一般是安装在这个目录中。

/usr/share：存放系统共用的文件。

/usr/src：存放内核源码的目录。

(3)特殊目录

有一些特殊的缩写表示的目录，如表 2.3 所示。

表 2.3　　特殊目录名

符号	意义
.	当前工作目录
..	父目录
～	用户主目录
-	前一个工作目录

2. 文件系统

操作系统中负责管理和存储文件信息的软件机构称为文件管理系统，简称文件系统。它规定了文件的存储方式及文件索引方式等信息。文件系统主要由三部分组成，分别是与文件管理相关的软件、被管理的文件和实施文件管理所需的数据结构。

在 Linux 系统中使用的文件系统，一般会在安装系统的时候创建完成。但通常我们也会遇到调整现有分区大小或者创建新的文件系统的情况。一般情况下，将遵循下列步骤来实现对文件系统的使用：

(1)在新的存储设备(硬盘)上创建分区，使用 fdisk 命令。

(2)在分区上创建文件系统，类似在 windows 下对分区进行格式化的操作，使用 mkfs 命令。

(3)挂载文件系统到现行系统中。在新的分区中创建文件系统以后，将该文件系统挂载到相应目录下即可使用。挂载文件系统使用命令 mount，如果希望在系统启动时文件系统被自动挂载，则需要在/etc/fstab 文件中添加该文件系统的信息。

(4)文件系统使用完成以后需要卸载。当类似于移动硬盘这样的存储设备上的文件系统被使用完成以后，在拿走设备之前，需要使用 umount 命令进行文件系统的卸载。

在 Linux 的树形结构中，只有一个根目录位于根分区，其他目录、文件以及外部设备(包括硬盘、光驱、调制解调器等)文件都是以根目录为起点，挂接在根目录下面的，即整个 Linux 的文件系统，都是以根目录为起点的，其他所有分区都被挂载到了目录树的某个目录中。通过访问

挂载点目录,即可实现对这些分区的访问。

Linux 支持长文件名,最长可以达到 256 个字节。Linux 的文件名中不能含有空格和以下特殊字符:

! @ # ¥ % ~ & × ( ) [ ] { } ' " \ / | ; < > << >>

Linux 操作系统中的文件和命令都需要区分大小写。

3. 绝对路径和相对路径

在计算机上要找到需要的文件或目录就必须知道文件或目录的位置,而表示文件或目录的位置的方式就是路径。表示路径的方式有绝对路径和相对路径两种。

(1)绝对路径

绝对路径是以根目录为起点,完整地表示到目标文件或目录的路径。

(2)相对路径

相对路径是以当前目录为起点,完整地表示到目标文件或目录的路径。

2.1.3 Linux 的目录与文件管理命令

1. 基本命令格式

Linux 操作系统有图形操作界面和字符操作界面。虽然现在很多工作可以在图形下通过简单的操作实现,但复杂的系统管理工作通常还是在字符操作界面下进行,而且所有的可执行程序都可以通过命令方式来执行。

Linux 命令的基本格式:

```
cmd -[options] [arguments]
```

说明:

cmd 是命令名字;

options 是选项,同一个命令可以通过不同的选项实现不一样的操作。

arguments 是参数。

一个简单的 Linux 命令只有命令名字。复杂一些的可以通过不同的选项和参数来实现。命令、选项以及参数之间通过空格(SPACE)键来分隔。

2. 用 mkdir 命令创建目录

Linux 中提供 mkdir 命令(make directory 的缩写)用于创建新的目录,可以同时创建一个或多个目录。

命令格式:

```
mkdir [选项] 目录名称
```

其中选项是可根据命令要完成的功能进行选择的。常用选项及功能见表 2.4。

表 2.4 mkdir 命令选项

选项	说明
-m	在创建目录时设定权限模式
-p	创建目录结构中指定的每一个目录,如果目录不存在则创建目录,如果目录已存在也不会被覆盖
-v	或--verbose:每次创建新目录都显示信息

例 2.1　在当前位置创建目录 web。

```
#mkdir  web
```

例 2.2　在/home 下创建目录 web，假设目前在系统的根目录下。

方法一：

```
#mkdir  home/web  //相对路径方法
```

方法二：

```
#mkdir  /home/web    //绝对路径方法
```

方法三：

```
#cd  home
#mkdir  web
```

例 2.3　在 root 用户主目录中创建目录 work1 和 work2，在/tmp 目录下创建 www 目录，假设目前位于 root 用户主目录中。

```
#mkdir  work1  work2  /tmp/www
```

例 2.4　在 root 用户主目录中创建目录树 web1/web2/web3/web4work1，假设目前位于 root 用户主目录中。

```
#mkdir  -p  web1/web2/web3/web4work1
```

3. 用 cd 命令切换目录

该命令用于改变当前目录，使用户进入指定的目录，并使该目录成为当前目录。

命令格式：

cd ［目录名称］

需要注意的是，如果 cd 命令（change directory 缩写）后加绝对路径，则可以直接进入指定的路径。如果加的是相对路径，则进入的是当前目录下的子目录。

例 2.5　切换当前位置为/home

```
#cd /home
```

例 2.6　目前位于系统根目录，创建/home/student 目录，并进入 student 目录。

```
#mkdir  home/student
#cd  home/student
```

例 2.7　切换到当前目录的父目录。

```
#cd  ..
```

例 2.8　返回当前用户的主目录

方法一：

```
#cd
```

方法二：

```
#cd ~    //~表示主目录
```

4. 用 pwd 命令查看当前路径

pwd(print working directory)命令用于显示当前目录的绝对路径。

命令格式：

pwd

5. 用 ls 命令查看当前目录信息

该命令用于列出一个或多个目录下的内容（目录或文件）。这是一个应用非常广泛的命令，

支持很多的选项，以实现更详细的控制。

命令格式：

```
ls [选项] [目录名称]
```

常用选项及功能见表 2.5。

表 2.5　　ls 命令选项

选项	说明
-a	列出目录下的所有文件，包括以“.”开头的隐含文件(all)
-d	将目录像文件一样显示，而不是显示其下的文件(directory)
-e	输出时间的全部信息
-i	输出文件的 i 节点的索引信息(inode)
-l	列出文件的详细信息(long)
-m	横向输出文件名，并以“,”作分格符
-x	按列输出，横向排序
-R	列出所有子目录下的文件
-S	以文件大小排序 (Size)
-1	一行只输出一个文件 (1 个文件)
--help	在标准输出上显示帮助信息

例 2.9　查看当前目录下的所有文件，含隐藏文件。

```
#ls  -a
```

例 2.10　以长格式方式查看/home 目录下的所有文件信息。

方法一：

```
#ls  -l -a  /home
```

方法二：

```
#ll  -a  /home
```

6. 用 touch 命令创建和更新文件

该命令用于创建一个新的空文本文件。

命令格式：

```
touch  文件名
```

如果命令后面接的是绝对路径，则以指定的文件名在指定路径下创建一个空文件。如果命令后面只接了新的文件名，则在当前路径下生成一个空文件。

如果指定的文件已经存在，则 touch 命令可以用来更新指定文件，使该文件被访问和修改的时间更新为系统当前的日期和时间。查看当前系统日期和时间，可以使用 date 命令。

例 2.11　在当前位置创建文件 file1 和 file2。

```
#touch  file1  file2
```

7. 用 cp 命令复制文件及目录

cp 是英文 copy 的缩写，可用于目录或文件的复制。

命令格式：

cp ［选项］ 源文件　目标文件

常用的选项及功能见表 2.6。

表 2.6　　cp 命令选项

选项	说明
-a	保留链接、文件属性，复制目录时可递归的复制目录
-f	如果目标文件或目录已经存在，则将其覆盖，并不做提示（force）
-i	如果目标文件或目录已经存在，则对用户进行提示，可以用字母 y 确认，其他字母都是否认
-r	复制目录，实现将源目录下的文件和子目录一起复制到目标目录中
-p	复制文件时保留修改时间和访问权限（permission）

例 2.12　将文件 file1 复制到/home 目录下并重命名为 file2。

```
#cp  file1  /home/file2
```

例 2.13　将文件 file1 复制到/home 目录下并重命名为 file2，如果/home 下已经存在 file2 文件，则备份原 file2 文件。

```
#cp  -b  file1 /home/file2
```

例 2.14　将 work1 目录复制到/home 目录下。

```
#cp  -r  work1  /home
```

8. 用 mv 命令移动和重命名文件及目录

英文 move 的缩写，该命令用于移动或重命名目录或文件。Linux 操作系统中没有单独的重命名命令，因此，可利用该命令来间接实现。

命令格式：

mv ［选项］ 源文件　目标文件

使用该命令可将文件移动到另一个目录之下，若目标文件已经存在，可以使用选项-b，则在覆盖已经存在的文件前，系统会自动创建一个原文件的备份，备份文件名为原名称后附加一个“～”符号。

例 2.15　将文件 file1 重命名为 file2。

```
#mv  file1  file2
```

例 2.16　将文件 file2 移动到/tmp 目录下并重命名为 file3，如果/tmp 下已经存在 file3 文件，则备份原 file3 文件。

```
#mv  -b  file2  /tmp/file3
```

例 2.17　将目录 work2 移动到/tmp 目录下。

```
#mv  work2  /tmp
```

例 2.18　将文件 file2 移动到/tmp 目录下并重命名为 file3

```
#mv  file2  /tmp/file3
```

9. 用 rm 命令删除文件

rm 命令是 Remove 的缩写，该命令用来删除文件或目录。可以删除一个目录中的一个或多个文件或目录，也可以将某个目录及其下的所有文件及子目录均删除。对于链接文件，只是断开了链接，原文件保持不变。

命令格式：

```
rm ［选项］ 目标文件
```

常用的选项见表 2.7。

表 2.7　　rm 命令选项

选项	说明
-f	强制删除文件或目录 (force)
-i	对用户进行提示,(inform)可以用字母 y 确认,其他字母都是否认
-r	目录删除,将指定目录下的所有文件及其子目录一并删除

需要说明的是,如果没有使用-r 选项,则 rm 不能进行删除目录操作。

例 2.19　删除文件 file。

```
#rm  file
```

例 2.20　删除目录 work。

```
#rm  -r  work
```

例 2.21　强制删除目录 work。

```
#rm  -rf  work
```

10. rmdir 删除目录

rmdir 命令是 Remove directory 的缩写,该命令的删除对象必须是空目录,且必须在上级目录进行删除操作。

命令格式：

```
rmdir 选项 目录名
```

常用选项见表 2.8。

表 2.8　　rmdir 命令选项

选项	说明
-p	删除指定的目录树
-v	--verbose 删除目录过程中输出诊断信息
--help	显示命令帮助信息

添加-p 选项后,如果删除指定的目录后,该目录的上层目录变成空目录,则可将上层目录一起删除。

11. 用 find 命令查找文件

该命令用于在相应路径下查找满足条件的文件。

命令格式：

```
find  查找目录的路径      查找条件选项   对查找条件的设定
```

其中“查找目录的路径”指查找的范围。“查找条件选项”指进行查找的依据,例如利用文件名、用户名、文件类型等条件进行查找。“对查找条件的设定”指的是将查找条件具体化,比如说利用文件名进行查找时,文件名应该满足什么样的条件才是查找的对象。查找文件时经常使用到的查找条件见表 2.9。

表 2.9　　find 命令查找条件选项

选项	说明
-name	通过文件名查找文件
-user	通过用户名查找文件
-type	通过文件类型查找文件
-size	通过文件大小查找文件
-atime	通过文件的最后访问日期查找文件(单位:天)
-mtime	通过文件的最后修改日期查找文件(单位:天)
-newer	查找比指定文件更新的文件
-amin	查找在指定时间内曾被存取过的文件(单位:分钟)
-cmin	查找在指定时间内被更改过的文件(单位:分钟)
-group	查找符合指定群组名称的文件
-perm	查找符合指定权限数值的文件

例 2.22　在/root 目录下,查找所有文件拥有者是 root 的文件。

```
#find  /root  -user  root
```

例 2.23　查找/etc 目录下所有以 pass 字符串开头的文件。

```
#find  /etc  -name  pass*
```

12. 获取帮助

Linux 系统中命令数量众多,而且不同的 options 可以决定一个命令完全不同的执行结果,同时系统也提供了大量的管理工具。作为系统管理员,记住全部的操作要耗费掉大量的精力,而且不经常使用的管理操作也没有记忆的必要。在这种情况下,Linux 系统从设计初始就为几乎所有的程序、工具、命令、系统调用、配置文档等编制和提供了多种帮助文档,可以让用户在用到该命令或操作的时候随时获得帮助信息。

(1)在字符界面下经常使用 man 命令获得帮助

在命令行下,通过输入 man cmd 即可获得该命令的所有信息。

例如,输入$man ls 命令,可看到如图 2.4 所示信息。

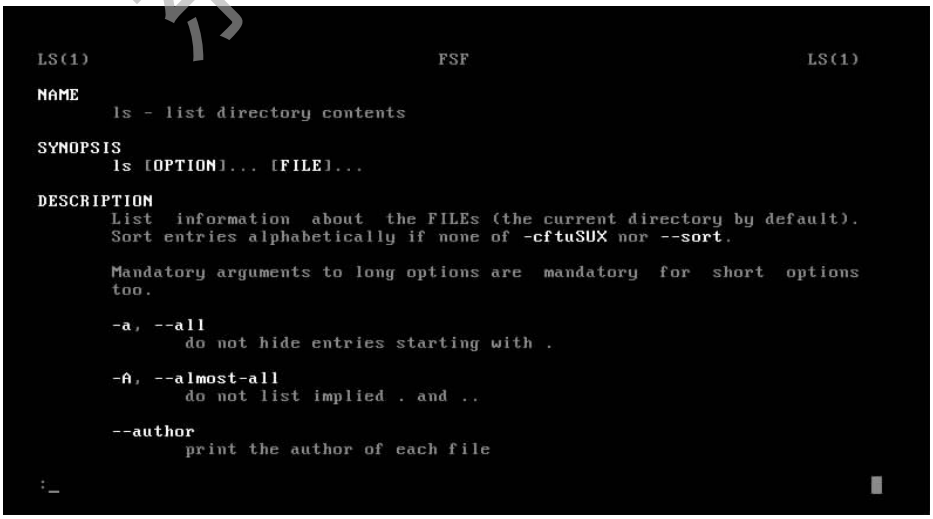

图 2.4　使用 man 命令获取帮助信息

在帮助文档页面中，可以看到关于命令的详细介绍。用户可以通过上下箭头进行翻页查询阅读。按 q 退出帮助文档。

一般情况下，帮助文档中包含表 2.10 所示的常规信息。

表 2.10　　man 手册常规信息

组成部分	说明
Name	命令名称含义的解释
Synopsis	命令使用的基本语法规则
DescrIPtion	对命令和每个 options 的详细解释
Author	命令程序编写者的信息
Files	和该命令有关的文件信息
Reporting Bugs	发现问题以后的报告地址
Diagnostics	该命令运行可能的错误信息
Copyright	版权信息
See also	和该命令有关系的其他命令信息清单
Example	该命令的例示
Bugs	程序中发现的问题

Man 帮助文档，根据内容的不同分别放在不同的 section 中。不同类型用不同的数字标识，各种不同 section 含义如表 2.11 所示。

表 2.11　　man 手册的类型

类型	说明
man1	可执行程序和普通用户权限，可执行 Shell 命令
man2	系统调用，内核命令说明
man3	子程序，库函数说明
man4	设备文件的参考说明
man5	配置文件格式说明，/etc 下配置文件的格式描述
man6	游戏说明
man7	宏包的相关信息
man8	系统管理工具，只有超级用户 root 才可以使用的

帮助文档的相关信息，根据类型的不同存放在系统的不同目录下。通常情况下，帮助文档一般都存放在/usr/share/man 下。

在 man 帮助页面中有一系列的帮助命令，如表 2.12 所示。

表 2.12　　man 帮助文档的操作命令

主要命令	说明
Space(空格)	下翻一页
b	回翻一页
PageDown	下翻一页
PageUp	回翻一页
Enter	下翻一行
Down-arrow	下翻一行
Up-arrow	回翻一行
End	到帮助页的结尾页
Home	回到帮助页的开始页
/expression	在帮助文档中查找字符串,从当前位置开始向下查找,匹配到的第一个结果在屏幕的第一行显示
? expression	在帮助文档中查找字符串,从当前位置开始向前查找,匹配到的第一个结果在屏幕的第一行显示
n	查找字符串过程中向后查找下一个匹配项
N	查找字符串过程中向前查找下一个匹配项
q	结束帮助文档的显示,退出

(2)在命令行下使用 info 获取帮助

随着开源的发展,有相当数量的 GNU 程序不再提供手册式的帮助文档,取而代之出现了信息文件,可以通过 info 命令来查看相关帮助信息。输入命令 #info cmd,结果如图 2.5 所示。

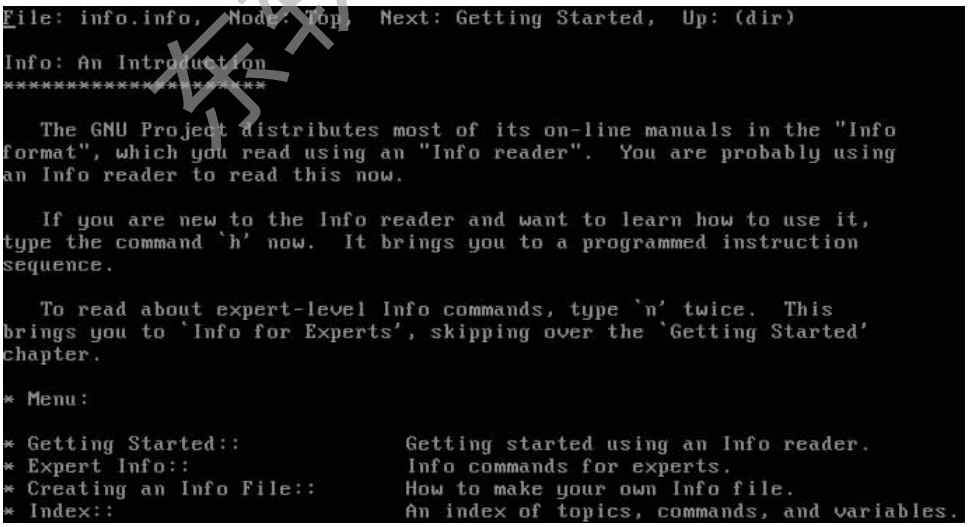

图 2.5　info 帮助信息页

使用信息文件的好处:使用有结构的文档设置、可以从目录直接到达特定节,可以链接到特定节。

帮助信息文件在目录/usr/share/info 下。常用的 info 命令，如表 2.13 所示。

表 2.13　　info 常用命令

主要命令	说明
Space 和 PageDown	搜索信息页，下翻
BackSpace 和 PageUp	前翻一页
b	光标回到当前信息页开头
e	光标移到当前信息页结尾
Tab	光标移动到下一个引用
Enter	跟在引用之后
n	移动到同一层的下一个信息页
p	移动到同一层的上一个信息页
u	移动到上一级
?	列出命令的概要
q	退出帮助信息文档

(3)图形界面下获取帮助

在 GNOME 桌面环境下，系统提供了帮助程序 yelp，如图 2.6 所示。通过该程序，用户可以在图形环境下查看 man 手册页，info 信息页，还可以浏览 GNOME 桌面自己的联机帮助文档。在此基础之上，yelp 还提供了索引的功能，可以在各种文档中进行快速查找，有效地提高用户查找帮助文档的效率。

图 2.6　GNOME 桌面的帮助浏览器

说明：在桌面环境下，主菜单中选择“帮助”，在终端命令行下运行“yelp”程序，在桌面下的应用程序中点击“帮助”→“目录”。

2.2　项目二：vim 文本编辑器的使用

【项目描述】

tom 用户在 tty1 控制台使用 vim 编辑器制定自己今天的项目学习计划，内容如下：

Data access：8a. m.

Reading program：10a. m.

Project design：13p. m.

Discussion items：15p. m.

计划制定完成后，tom 对文件的内容进行了浏览和确认，最后将内容输出给在 tty2 控制台的用户参考。

【构思设计】

本项目主要涉及文档内容的编辑、查看、输出重定向等知识点。文档的编辑采用 vim 编辑器完成，需要掌握 vim 的使用方法。文档查看的命令有很多，各有优势，可以根据具体情况进行选择使用。输出重定向可以帮助用户改变命令的输出位置，本项目需要输出到其他的控制台，方便信息共享。本项目相关知识点如表 2.14 所示。

表 2.14　　本项目相关知识点分析

序号	知识点	详见章节
1	掌握 vim 编辑器的使用方法	2.2.1
2	理解查看文件内容的方法	2.2.2
3	了解输出重定向	2.2.3

【实施运行】

操作步骤：

```
$cd     //切换至 tom 用户主目录
$vim tom.plan     //vim 命令新建 tom.plan 文件并打开，详见 2.2.1
```

切换到输入模式，输入：

```
Data access：8a. m.
Reading program：10p. m.
Project design：13a. m.
Discussion items：15p. m.
```

输入完成后，切换至末行模式，保存退出。

```
$less  tom.plan   //查看确认 tom.plan 内容，详见 2.2.2
$less  tom.plan>>/dev/tty2   //将 tom.plan 内容输出到/dev/tty2，详见 2.2.3
```

2.2.1　vim 编辑器的使用

vim 编辑器是 Linux 系统中功能最为强大的全屏幕文本编辑器。它可以完成输出、删除、

查找、替换、块操作等文本操作，而且用户可以根据自己的需要对其进行定制，这是其他编辑程序所没有的。但是 vim 不是一个排版程序，它不能像 Word 或 WPS 那样可以对字体、格式、段落等其他属性进行编排，它只是一个文本编辑程序。

在使用时，只需在命令提示符后面，直接输入 vim 或者 vim 加文件名，就可以启动 vim 编辑器。vim 命令后若指定了文件名，则打开或创建该文件(如果指定的文件不存在)。如果没有指定文件名，则创建一个未命名的新文件。

vim 编辑器共有三种模式，分别是命令模式、输入模式和末行模式，相互之间的转换关系如图 2.7 所示。

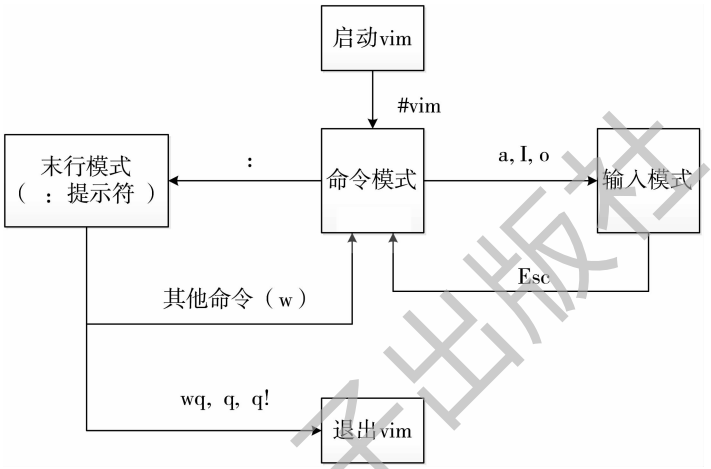

图 2.7 vim 编辑器的工作模式转换

命令模式：不管用户当前处于何种模式，只要按 Esc 键，则立即进入命令模式。在命令模式下，允许输入 vim 的命令，以对文件进行管理，输入的 vim 命令通常都是单个字母并且所输的命令都不回显，若输入的不是合法的 vim 命令，系统则会发出蜂鸣声进行提示。在命令模式下，控制光标移动的命令如表 2.15 所示。

表 2.15 控制光标移动的命令

移动	命令
←，↓，↑，→	h，j，k，l
到下一行的首字符	+
到上一行的首字符	−
到单词的尾部	e 或 E
按单词前移	w 或 W
按单词后移	b 或 B
到行尾	$
到行首	0

命令模式下的常用命令如表 2.16 所示。

表 2.16　　命令模式下的常用命令

命令		说明
光标的定位	G	跳到文件最后一行，光标停于行首
	0	移动光标到当前行的行首
	$	移动光标到当前行的行尾
字符检索	/str	从当前页开始向后搜索字符串
	? str	从当前页开始向前搜索字符串
	n	往相同的方向移动到被搜索的字符串所在的位置
	N	往相反的方向移动到被搜索的字符串所在的位置
文本的复制、粘贴和删除	dd	删除当前光标所在行
	ndd	删除从光标所在行开始的 n 行
	yy	复制光标所在当前行的文本信息到缓冲区
	p	粘贴在缓冲区中的内容到当前光标所在行的下一行
	nyy	将当前开始的 N 行内容复制到缓冲区
	x	删除当前光标所在位置的一个字符
	nx	删除从光标所在位置开始向右的 n 个字符
撤销和重复	u	取消刚刚发生的误操作或者不适合的操作对文件造成的影响，恢复文件到操作之前的状态
	.	重新执行一遍刚刚执行完的操作
保存和退出	ZZ	存盘退出
	ZQ	不保存此次关于文本的修改退出编辑

输入模式：也称插入模式或编辑模式。在该模式下，用户输入的内容成为文件正文，并显示在屏幕上。在命令模式下，输入 i、a、o 命令都可以进入到输入模式，实现对文件内容的输入或修改。进入输入模式的命令如表 2.17 所示。

表 2.17　　进入输入模式的命令

编辑行为	命令
在当前位置插入文本	i
在行首插入文本	I
在当前位置追加文本	a
在行尾追加文本	A
在光标所在行的上面新建一行，等待输入	o
在光标所在行的下面新建一行，等待输入	O

末行模式：虽然在命令模式下可以实现大部分的文件操作功能，但是由于命令模式输入的命令不回显，所以当操作命令表达比较复杂时，需要回显确认输入，例如要将文件内容保存到指

定的文件里或者将指定文件的内容读入到当前位置等操作。这时就要用到 vim 提供的末行模式来完成。

在命令模式下按 Shift+“:”键，即可切换到末行模式。此时，在编辑器屏幕的最后一行将显示“:”提示符，在此行中输入命令，按回车键后即可开始执行。命令执行结束后，自动切换到命令模式。末行模式的常用命令见表 2.18。

表 2.18　　末行模式的常用命令

命令		说明
光标的定位	:n	输入要移动的行号，光标即可到达目的行
字符串检索、替换	/str/	从当前光标开始向后移动到被搜索的字符串的位置
	? str?	从当前光标开始向前移动到被搜索的字符串的位置
	:/str/ w file	将包含有 str 的行都写到文件 file 中
	:/str1/,/str2/w file	将从 str1 开始到 str2 结束的内容都写到文件 file 中
	:s/str1/str2/g	将所有 str1 替换成 str2
文件相关	:w	将当前标记的内容存盘
	:w file	将当前编辑的内容写到文件 file 中
	:n1,n2 w file	将从 n1 开始到 n2 结束的信息写到文件 file 中
文本的复制、粘贴和删除	:d	删除当前光标所在行
	:nd	删除从光标所在行开始的 n 行
	:n1,n2 co n3	将从 n1 开始到 n2 结束的所有内容复制到 n3 后面
	:n1,n2 m n3	将从 n1 开始到 n2 结束的所有内容移动到 n3 后面
	:n1,n2 d	删除 n1 开始到 n2 结束的所有内容
	:.,$d	删除从当前行到结尾的所有内容
	:/str1/,/str2/d	删除从 str1 开始到 str2 为止的所有内容
保存和退出	:wq	存盘退出
	:q!	不保存此次关于文本的修改退出编辑
	:q	退出 vi

例 2.24　使用 vim 编辑器创建并编辑文件 hello.c。

```
#vim  hello.c
```

直接在命令提示符后输入命令 vim hello.c，执行后自动生成文件 hello.c，并以全屏的方式打开该文件，进入命令模式。然后可以使用 i 命令切换到输入模式，对文件的内容进行编辑。进入输入模式后，光标停在第一行第一个字符的位置上，其他每一行都有一个“～”符号，表明该行是一空行。最后一行是状态行，显示当前正在编辑的文件名以及文件的状态。如果被编辑文件已经存在，则在屏幕上显示文件的内容，光标停在第一行的首位，在状态行除了可以看到文件的名字以外，还可以看到文件的行数、字数，如图 2.8 所示。

```
#include<stdio.h>
main()
{
    printf("Hello world!\n");



}
~
~
~
-- 插入 --                          4,30          全部
```

图 2.8　vim 编辑器编辑文件 hello. c

完成了对文件的编辑之后，进入末行模式，保存文件并退出 vim 编辑。需要注意的是，vim 编辑器不能从输入模式直接切换到末行模式。因此需要先按 Esc 键，进入命令模式，然后再按 Shift＋“:”键，切换到末行模式。进入末行模式之后，输入 wq 命令，对文件进行保存，并退出 vim 编辑器。执行结果如图 2.9 所示。

```
#include<stdio.h>
main()
{
    printf("Hello world!\n");



}
~
~
:wq
```

图 2.9　保存文件 hello. c

2.2.2　查看文件内容命令

1. more/less 浏览文件全部内容

当文件内容过多时，可以用 more 或 less 命令来查看。

命令格式：

more 文件名

less 文件名

使用 more 命令时可以查看相应文件第一屏的内容。同时在屏幕左下角出现一个百分比，它表示的是当前显示的内容占整个文件内容的百分比。按空格键，系统会自动显示下一屏的内容，到达文件末尾后，命令执行结束。

less 命令比 more 命令功能更强大，在查看文件内容时，按 z 键，可以向下翻页，显示下一屏的内容。按 w 键，可以向上翻页，显示上一屏的内容。到达文件尾部时会出现提示符“End”。结束浏览时，直接按 q 键退出。同时支持光标键和翻页键。

例 2.25　使用 vim 编辑器创建并编辑文件 info，文件内容不少于 30 行，使用 more 和 less 命令分别查看文件内容。

```
#vim  info   //输入不少于 30 行
#more  info  //注意底部的百分比显示
#less  info  //注意退出使用“q”
```

2. cat 查看文件内容

该命令用于将文件的内容打印输出到显示器或终端窗口上。

命令格式：

```
cat ［选项］文件名
```

如果文件的内容超过一屏，则只显示最后一屏的内容，因此常用于查看内容不太多的文件的内容。如果被查看的文件内容过多，则会因滚动太快而无法阅读。常用的选项见表 2.19。

表 2.19　　cat 命令选项

选项	说明
-n	由 1 开始对所有输出的行数进行编号
-b	由 1 开始对所有输出的非空白行的行数进行编号
-s	当遇到连续两行以上的空白行时，用一个空白行代替

3. head/tail 显示文件头部/尾部信息

head 命令用来查看文件前若干行，tail 命令用来查看文件后若干行。

命令格式：

```
head ［选项］ 文件名
tail ［选项］ 文件名
```

默认情况下，查看 10 行的内容，可以通过对选项的设置来决定要查看的行数。

例 2.26　查看文件 hello.c 文件第一行的内容和后两行的内容。

```
#head  -1  hello.c
#tail  -2  hello.c
```

4. grep 查询字符串

grep 命令常规情况下，在指定文本文件中匹配字符串，输出匹配字符串所在行的全部内容。

命令格式：

```
grep  关键字 查找范围
```

例 2.27　在当前目录下的 myfile 文件中查找字符串 this。

```
#less  myfile
#grep  this  myfile
```

2.2.3　管道与重定向命令

1. 管道命令

有时在操作 Linux 命令时，需要将一个命令的执行结果作为另一个命令输入来执行，这时候就需要用到管道命令。

管道命令是“|”。管道命令可以将多个命令连接在一起，每一个命令都独立运行，每一个命令的运行结果都作为下一个命令的输入。管道的单向性决定了命令处理的单向性。

命令格式：

```
cmd1 | cmd2 | cmd3 |……|cmd(n)
```

管道命令可以实现将一个命令的输出当作另一个命令的输入，后者的输出又可作为第三条命令的输入，以此类推。这样，管道命令行中最后一条命令的输出才会显示输出在屏幕上。因此，可以利用管道操作，将多条相关的命令连接起来。在使用的时候，第一个命令正常写，后面的命令都只写操作，不写操作的对象，因为操作的对象就是前一个命令的输出结果。

例 2.28　查看/etc 目录下的内容。

```
#ls  /etc | less
```

本例中，如果执行 #ls /etc 命令，会发现由于/etc 目录下内容比较多，所以无法看到全部的内容，一种解决方案就是使用|命令，将 ls /etc 的执行结果作为 less 命令的输入，达到分页显示 ls /etc 内容的功能。

例 2.29　显示/etc 目录下文件名以 pass 字符串开头的文件。

```
#ls  /etc | grep pass*
```

2. 重定向

Linux 命令在执行的时候，常规下都会有输入，命令处理完成后会有结果的输出。输入通常使用标准输入设备，输出通常使用标准输出设备和标准错误设备端口。

stdin 表示标准输入设备端口，命令的输入都从其获取。默认是键盘。

stdout 表示标准输出设备端口，命令执行的结果都向其输出。默认是控制台的显示屏。

stderr 表示标准错误设备端口，命令执行过程中出现的错误信息都向其输出。默认是控制台的显示屏。

所谓重定向，指不使用系统的标准输入、输出、错误端口进行信息的获取或输出，而是通过重新的指定，让命令从非默认的输入输出设备获取或输出信息。所以重定向分为：输入重定向、输出重定向和错误重定向。

重定向功能是通过重定向符号来实现的。Shell 执行命令时，会通过检查命令行中是否含有重定向符号来决定本次命令的执行是否需要重定向。表 2.20 列出了重定向的常用符号。

表 2.20　　**重定向的常用符号**

命令	说明
>	输出重定向。如果原来目标文件存在，则新的内容会覆盖文件中原有的内容
>>	输出重定向。如果原来目标文件存在，则新的内容会追加在原来内容的后面，不覆盖文件中的原有内容
<	输入的重定向。即命令的输入不通过键盘来完成
2>	错误重定向
&>	输出重定向和错误重定向同时实现

例 2.30　将 ls /命令的执行结果记录到 list 的文件中。

```
#ls  / > list
```

执行后在本目录下会新建一个 list 文件，文件内容为 ls / 命令的执行结果。

例 2.31　如果命令 useradd lolo 在执行时发生错误，则将错误信息保存到当前目录下的 errlog 文件中。

```
#useradd lolo  2>errlog
```

如果命令执行正常，没有错误出现，则该错误重定向操作不会被执行。只有在命令执行出现错误的情况下，该错误重定向才会被执行。错误信息重定向经常用于对程序的调试。

例 2.32 将程序 hello 的执行结果和错误信息全部都重定向输出到/tmp 目录下的 outfile 文件中。

```
#./hello  &> /tmp/outfile
```

2.3 项目三：使用 U 盘备份文件

【项目描述】

tom 用户在 Linux 系统中工作了一天，在工作结束前，打算把今天的工作文档 tfile 备份到自己的 U 盘中。

【构思设计】

本项目要求完成 Linux 环境下对 U 盘的访问。首先需要让 Linux 系统识别到 U 盘，然后进行挂载，挂载成功后就对 U 盘进行操作，使用完成后，要进行正确的卸载操作，最后拔出 U 盘，本项目相关知识点如表 2.21 所示。

表 2.21 本项目相关知识点分析

序号	知识点	详见章节
1	使用 mkdir 命令创建挂载点	2.3.1
2	查询系统分配给 U 盘的设备名	2.3.2
3	使用 mount 命令进行挂载	2.3.3
4	使用 umount 命令卸载 U 盘	2.3.4

【实施运行】

操作步骤：

(1)根据项目描述中的要求，先在 Windows 主机中查看 U 盘中的内容。

(2)按照图 2.10 中的示例，点击 VMware 中的“断开连接”，使 U 盘断开与 Windows 主机的连接，并与虚拟机中的 Linux 操作系统连接起来。在 Windows 主机的右下角出现“安全地移除硬件”字样。

(3)为了能挂载使用 U 盘，首先需要设置一个挂载点目录。创建目录/mnt/usb 作为挂载点。然后查询系统为 U 盘分配的设备名，最后使用 mount 命令来完成挂载，实现的命令为：

```
$mkdir  /home/tom/usb   //创建目录/home/tom/usb 作为挂载点
$fdisk  -l              //查询系统为 U 盘分配的设备号
```

```
$mount     /dev/sdb1    /home/tom/usb //使用 mount 命令来完成挂载
```

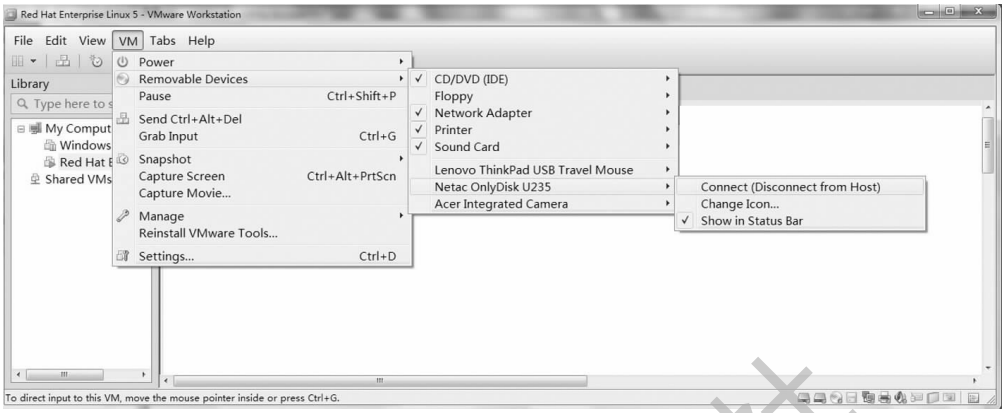

图 2.10　断开 U 盘与 Windows 主机的连接

执行挂载命令时，只要未输出错误信息，则意味着挂载成功。进入到对应挂载目录/home/tom/usb，就可以查看到 U 盘中的内容。需要注意的是，在 Linux 环境下查看 U 盘中的内容，汉字部分可能不会正常显示，出现的是“????”形式。

(4)根据项目描述中的要求，需要对 tfile 文件进行备份，然后卸载 U 盘，实现的命令为：

```
$cd   /home/tom/usb              //切换目录
$cp   /home/tom/tfile  ..        //复制文件 tfile 到 U 盘进行备份
$umount     /dev/sdb1            //卸载/dev/sdb1 设备
```

注意卸载时先要退出挂载点。

2.3.1　挂载点

由于 Linux 操作系统只有一个根目录，所以当向系统中添加新的存储设备时，不会像 Windows 操作系统那样出现一个新的根目录。因此在 Linux 下访问新存储设备时需要首先创建挂载点。

所谓的挂载点就是文件系统中存在的一个目录，通常情况下，创建在/mnt 目录下，挂载成功后，访问挂载点就是访问新的存储设备。

通常情况下，挂载点应该是空目录，否则原来该挂载点中存在的文件将会被隐藏。而且，挂载点在实施挂载操作之前就应该存在。

2.3.2　查询设备名

在 Linux 系统中，挂载设备之前需要使用 fdisk -l 命令查询系统自动分配的设备名，如图 2.11 所示。

```
[root@localhost ~]# fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          13      104391   83  Linux
/dev/sda2              14        1305    10377990   8e  Linux LVM

Disk /dev/sdb: 8011 MB, 8011120640 bytes
247 heads, 62 sectors/track, 1021 cylinders
Units = cylinders of 15314 * 512 =7840768 bytes


   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1  ?       50813      125353   570754815+  72  Unknown
```

图 2.11　fdisk -l 命令的执行结果

可以看到，现在系统中两块硬盘/dev/sda 和/dev/sdb，但是第二块硬盘/dev/sdb 大概 8G，设备名为/dev/sdb1。注意，通常情况下，由于 U 盘只有一个盘，是即插即用设备，所以系统可以直接分配设备名字。如果挂载的是硬盘那么执行 fdisk -l 命令后将会发现出现的新硬盘由于没有分区而不能显示设备名，因此需要先分区再创建文件系统然后才可以挂载详见 2.4。

2.3.3　挂载文件系统

挂载由 mount 命令来完成，可以灵活地挂载各种类型的文件系统。

命令格式：

mount ［选项］［设备名］［挂载点］

常用选项及功能见表 2.22。

表 2.22　　mount 命令选项

选项	说明
-t fstype	指定要挂载的文件系统的类型，如果不清楚，可以使用-t auto 让系统自己选择最合适的文件系统类型挂载
-r	以只读的方式挂载文件系统
-w	以可写的方式挂载文件系统
-o	设置挂载属性
-a	挂载/etc/fstab 文件中记录的设备

使用 mount 命令，不带任何选项，可以列出当前系统中所有已经挂载的文件系统，如图 2.12 所示。

```
[root@localhost ~]# mount
/dev/mapper/VolGroup00-LogVol00 on / type ext3 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sda1 on /boot type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
.host:/ on /mnt/hgfs type vmhgfs (rw,ttl=1)
none on /proc/fs/vmblock/mountPoint type vmblock (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
```

图 2.12　mount 命令执行结果

图 2.12 信息显示每个磁盘分区在目录树中的挂载点，文件系统的类型以及挂载权限。例如，/dev/sda3在文件系统中的挂载点是/usr，文件系统是 ext3，挂载权限是可读写。mount 命令可以用来挂载新的文件系统。

2.3.4　卸载文件系统

文件系统可以被挂载，在不使用的时候就可以被卸载。卸载文件系统的命令是 umount，这个命令可以把文件系统从 Linux 系统中的挂载点上分离，将原来建立的文件系统和挂载点的连接断掉。

命令格式：

```
umount [设备名或者挂载点]
```

在卸载一个文件系统的时候，需要指定要卸载的文件系统的挂载点或者设备名。例如：

```
#umount /dev/sdb1
#umount /mnt/sdb1
```

可以将/dev/sdb1 上的文件系统卸载。

但是，当文件系统处于“busy”状态的时候，即在被使用的时候，不能进行卸载操作。

文件系统的“busy”状态可能由于：文件系统上缓冲文件被使用；文件系统上有文件被打开；文件系统上有目录被使用。

当在文件系统的挂载点下对文件系统进行卸载时，也会看到“busy”的状态。如果看到文件系统处于“busy”状态，可以使用 fuser 命令查看正在使用该文件系统的用户信息。

命令格式：

```
fuser [options]  文件系统名或者文件名
```

常用选项及功能见表 2.23。

表 2.23　　fuser 命令选项

选项	说明
-a	显示所有在命令行中指定的文件系统信息。默认情况下，至少被一个进程访问的文件才会被显示
-k	杀死访问文件的进程
-i	在杀死使用文件的用户进程之前，提示确认
-u	显示使用文件系统的进程的所有者信息
-v	按进程查看命令 ps 的显示模式显示文件的使用者信息，包括 PID、USER、COMMAND 等等
-m	显示在访问该文件系统的文件的所有进程

例 2.33　查看哪个进程打开了当前目录下的 myfile 文件。

```
#fuser  myfile
myfile:  943
```

结果显示访问 myfile 文件的进程 PID 为 943。

如果要杀死在/usr 分区上打开文件的进程，则用命令如下：

```
#fuser  -km /usr
```

所以，当一个文件系统处于“busy”状态时，可以先杀死所有在该文件系统上打开文件的进程，然后卸载文件系统。

2.4 项目四:扩充系统的硬盘空间

【项目描述】

管理员 root 发现 Linux 系统硬盘资源不足,因此在服务器上新增加了一块硬盘,想在 Linux 系统下使用它。

【构思设计】

在 Linux 的安装过程中,会自动创建分区的文件系统,但是如果硬盘不够用了就需要向系统添加新的硬盘来扩充硬盘的可用空间。通常遵循以下步骤:

(1)向系统中添加一块硬盘
(2)由系统识别硬盘名
(3)对新硬盘进行分区
(4)创建文件系统
(5)挂载使用

【实施运行】

操作步骤:

(1)向系统中添加一块硬盘

如果是真实的 Linux 操作系统,则直接向系统添加一块新硬盘即可。

如果是在虚拟机中则需要进行以下过程。

在虚拟机关机的前提下,打开虚拟机属性面板,选中“Hard Disk(SCSI)”一项,如图 2.13 所示。

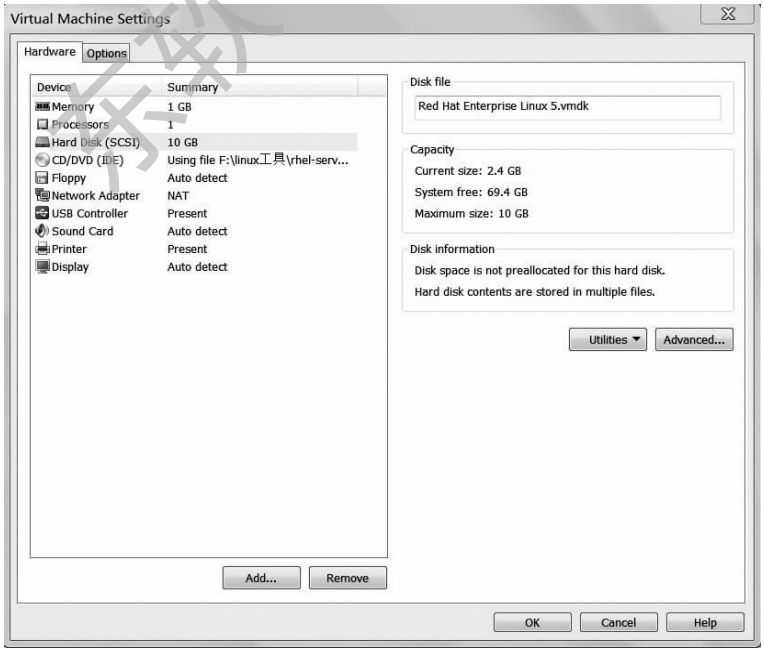

图 2.13 虚拟机属性面板

点击“Add”按钮，如图 2.14 所示。

图 2.14　选择添加硬盘

在图 2.14 中选择“Hard Disk”一项，单击“Next”，如图 2.15 所示。

图 2.15　创建新硬盘

在图 2.15 中选择“Create a new virtual disk”，单击“Next”，如图 2.16 所示。

图 2.16　创建 SCSI 硬盘

在图 2.16 中选择“SCSI”，单击“Next”，如图 2.17 所示。

图 2.17　设置硬盘容量

在图 2.17 中设置新硬盘的容量，单击“Next”，如图 2.18 所示。

图 2.18　新文件系统的存储命令

单击“Finish”,完成新硬盘的添加,如图 2.19 所示。

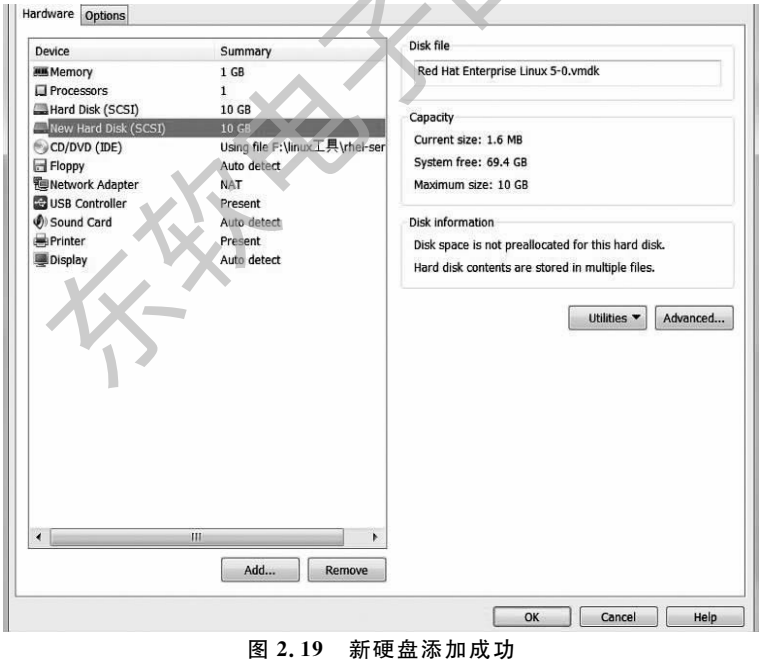

图 2.19　新硬盘添加成功

(2)由系统识别硬盘名

硬盘添加成功后,重新启动计算机,进入 Linux 操作系统。首先使用 fdisk -l 命令确认添加的第二块硬盘在系统中被识别,如图 2.20 所示。

```
[root@localhost ~]# fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          13      104391   83  Linux
/dev/sda2              14        1305    10377990   8e  Linux LVM

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/sdb doesn't contain a valid partition table
```

图 2.20 使用 fdisk -l 命令查看硬盘情况

可以看到：现在系统中两块硬盘/dev/sda 和/dev/sdb，其中第一块硬盘/dev/sda 容量10.7GB是目前正在使用的硬盘，划分了两个分区，分别是/dev/sda1 和/dev/sda2。第二块硬盘/dev/sdb 容量10.7GB，目前还没有磁盘分区表。

(3)对新硬盘进行分区

第 1 步：进行分区操作。

新的存储设备使用之前，需要进行磁盘存储设备的分区操作，可以使用磁盘分区工具 fdisk 完成。在 Linux 中，每一个硬件设备都映射到一个系统文件，对于硬盘、光驱等 IDE 或 SCSI 设备也不例外。Linux 给 IDE 设备分配由 hd 前缀组成的文件；而对于 SCSI 设备，则分配由 sd 前缀组成的文件。例如，第一个 IDE 设备，Linux 系统就定义为 hda；第二个 IDE 设备就定义为 hdb；下面以此类推。而 SCSI 设备则依次定义为 sda、sdb、sdc 等。

目前新添加的硬盘为/dev/sdb，分区使用 fdisk 命令，在后面直接加上要分区的硬盘作为选项，如图 2.21 所示。

```
[root@localhost ~]# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.


The number of cylinders for this disk is set to 1305.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help):
```

图 2.21 分区操作第 1 步

Command 后面输入命令就可以对硬盘进行分区操作。输入 m 可以列出所有可供选择的子命令，如表 2.24 所示。

表 2.24　　fdisk 命令的子命令

命令	说明
a	调整硬盘启动分区
d	删除一个硬盘分区
n	创建新的硬盘分区
l	列出所有分区类型
m	列出所有命令
p	列出硬盘分区表
t	更改分区类型
u	切换分区的显示单元
w	将所有的修改写入硬盘分区表，退出 fdisk
q	不保存更改退出 fdisk
x	列出额外选项

第 2 步：创建新分区。

在 Command 后面输入“n”命令，回车，如图 2.22 所示。

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1305, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-1305, default 1305): +4G

Command (m for help):
```

图 2.22　创建新分区

说明：

n 表明创建新分区。

p 创建主分区。

1 创建分区号为 1 的第一个主分区。

First cylinder 分区的起始位置，默认不修改。

Last cylinder 分区的结束位置，可以指定所要创建新分区的大小。

创建完成，会重新回到命令行下。+4G 表示创建主分区容量是 4G。

可以通过 p 查看分区表，如图 2.23 所示。

```
Command (m for help): p

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1         487     3911796   83  Linux

Command (m for help):
```

图 2.23　创建新分区完成以后的分区表情况

第 3 步:重复第 2 步操作创建第二个主分区,如图 2.24 所示。

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (488-1305, default 488): 488
Last cylinder or +size or +sizeM or +sizeK (488-1305, default 1305): 1305

Command (m for help):
```

图 2.24　创建第二个分区

分区结束以后,输入“p”命令查看分区情况,然后输入“w”命令将分区信息写入分区表并退出 fdisk,如图 2.25 所示。

```
Command (m for help): p

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1         487     3911796   83  Linux
/dev/sdb2             488        1305     6570585   83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

图 2.25　显示并保存分区表

第 4 步:查看分区以后的情况,如图 2.26 所示。

```
[root@localhost ~]# fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          13      104391   83  Linux
/dev/sda2              14        1305    10377990   8e  Linux LVM

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1         487     3911796   83  Linux
/dev/sdb2             488        1305     6570585   83  Linux
```

图 2.26　分区结束后硬盘上的分区情况

至此可以看到第二块硬盘上已经存在两个分区，分别为/dev/sdb1 和/dev/sdb2。

(4)创建文件系统

完成分区后，要在分区上创建文件系统，该分区才可以被使用。由于刚才创建了两个分区，所以要依次创建文件系统。

在第一个分区上创建文件系统(ext3 类型的文件系统)，如图 2.27 所示。

```
[root@localhost ~]# mkfs.ext3  /dev/sdb1
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
489600 inodes, 977949 blocks
48897 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1002438656
30 block groups
32768 blocks per group, 32768 fragments per group
16320 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 27 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

图 2.27　创建文件系统

说明：在通过 mkfs 创建文件系统的操作中，用户可以指明创建何种文件系统类型。如果不指定特定的文件系统类型，mkfs 默认以 ext2 的文件系统类型格式化指定的分区。

在第二个分区上创建文件系统(ext3 类型的文件系统)，如图 2.28 所示。

```
[root@localhost ~]# mkfs.ext3  /dev/sdb2
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
822528 inodes, 1642646 blocks
82132 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1686110208
51 block groups
32768 blocks per group, 32768 fragments per group
16128 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 37 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

图 2.28　创建指定类型的文件系统

(5)挂载使用

首先创建挂载点,命令如下:

```
#mkdir  /mnt/sdb1
#mkdir  /mnt/sdb2
```

创建两个挂载点,每个文件系统一个,然后挂载文件系统,命令如下:

```
#mount  /dev/sdb1  /mnt/sdb1
#mount  /dev/sdb2  /mnt/sdb2
```

查看已经挂载的文件系统信息,如图 2.29 所示。

```
[root@localhost ~]# mount
/dev/mapper/VolGroup00-LogVol00 on / type ext3 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sda1 on /boot type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
.host:/ on /mnt/hgfs type vmhgfs (rw,ttl=1)
none on /proc/fs/vmblock/mountPoint type vmblock (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
/dev/hdc on /media/RHEL_5.5 i386 DVD type iso9660 (ro,noexec,nosuid,nodev,uid=0)
/dev/sdb1 on /mnt/sdb1 type ext3 (rw)
/dev/sdb2 on /mnt/sdb2 type ext3 (rw)
```

图 2.29　已经挂载的文件系统信息

可以看到,最后两行信息显示,两个新的文件系统挂载成功,同时也可以验证两个分区的文件系统都是 ext3 文件系统。

使用 df 命令查看文件系统的磁盘空间占用情况,如图 2.30 所示。

```
[root@localhost ~]# df
文件系统                1K-块      已用      可用 已用% 挂载点
/dev/mapper/VolGroup00-LogVol00
                      7998912   2686408   4899628  36% /
/dev/sda1              101086     12186     83681  13% /boot
tmpfs                  517552         0    517552   0% /dev/shm
.host:/              102826312  25721916  77104396  26% /mnt/hgfs
/dev/hdc               3038672   3038672         0 100% /media/RHEL_5.5 i386 DVD
/dev/sdb1              3850292     73248   3581456   3% /mnt/sdb1
/dev/sdb2              6467288    145668   5993092   3% /mnt/sdb2
```

图 2.30　查看文件系统的磁盘空间占用情况

最后可以切换至新的硬盘下，执行应用操作，例如编辑文件 hello.c：

```
#cd  /mnt/sdb1
#vim  hello.c
```

这样在/mnt/sdb1 中的操作就是在新的硬盘文件系统中进行操作。

至此，已经向系统中增加了一块新的硬盘，并且可以正常使用。

2.5　知识扩展

2.5.1　命令行下提高工作效率的方法

常规情况下，在执行一个命令时，需要在命令提示符下完整地输入该命令。在命令或者操作对象名字很复杂的情况下，输入操作就会占用大量的时间并且容易出错。在 Linux 下提供了几种方法既可以简化操作，又可以使操作的准确度提高，从而提高工作效率，节省时间。在这里介绍一下。

1. Linux 下 Shell 命令补全功能

命令补全是指在当前目录下，当用户在命令行键入的字符足以确定操作目录下一个唯一的文件时，只需要按 Tab 键，Shell 就会自动实现命令的补全操作，把命令的剩余部分自动补齐。

通常在 bash(Linux 默认 Shell)下进行操作的时候，不需要把命令名全部都输入完整，只需要输入一两个首字符 bash 就能帮我们补全。

例如当前目录下的文件如图 2.31 所示。

```
err   hello        hello.java  j2sdk-1_4_2_04-Linux-i586-rpm.bin  list
file  hello.class  hello.sh    jakarta                            passwd
```

图 2.31　显示当前目录下文件列表

如果想看文件 list，可以输入完整命令：

```
#more  list
```

也可使用 bash 自动补全的功能，因为 list 是当前目录里唯一以字母 l 开头的文件，bash 在用户输入 l 后就能判断用户要操作哪个文件：

```
#more  l
```

在输入 l 后，唯一的可能就是 list 文件。想让 bash 自动补全，按下 Tab 键。

```
#more  l<tab>
```

当按下 Tab 键时，bash 将帮助补全命令并显示在屏幕上。但命令的执行需要按下回车键

来实现，在命令执行前 bash 会让用户确认自动补全的命令是否正确。

同样情况，执行程序 j2sdk-1_4_2_04-Linux-i586-rpm.bin，只需要输入：

```
$./j2<tab>
```

用户输入命令时，不论何种情况下按下 Tab 键，bash 都会尽力去补全命令，如果在不成功的情况下，bash 会发出提示声提示用户输入更多信息。用户需要输入更多字符，并在此按 Tab 键重新实现命令的补全。上例中，执行命令时，当前目录下 j 打头的文件多于一个，所以只输入 j<tab> bash 会提示输入更多信息，如图 2.32 所示：

```
[root@localhost test]# ./ j
j2sdk-1_4_2_04-Linux-i586-rpm.bin  jakarta
```

图 2.32 <tab>执行结果

输入 j2〈tab〉即可完整补全命令并执行。

2. 命令别名

命令别名通常情况下是其他一系列长命令的缩写，用来减少用户的输入。同时通过命令别名，用户可以为命令取一个适合自己的，而且习惯使用的名字。

命令格式：

```
alias alias_name='original_command'
```

alias_name 是用户给命令取的别名，original_name 是原来的命令。

说明：

(1)在定义别名的语法中，注意"="左右两边不允许留空格。

(2)原始命令是在''单引号中包含的。

(3)在 Linux 系统中，命令行定义的所有属性值在当前 Shell 状态下生效，在下次用户登录时就失效。所以，关于别名的定义，如果用户想让这个属性长久存在，就需要用户在自己的 home 目录下的系统文件.bashrc 中添加记录。

如果用户在命令行输入 alias 不带任何参数，则会显示系统当前所有已经定义的别名信息，包括每个命令的原始命令含义。以 Red Hat Enterprise Linux 5 为例，输出结果如图 2.33 所示。

```
[root@localhost ~]# alias
alias cp='cp -i'
alias l.='ls -d .* --color=tty'
alias ll='ls -l --color=tty'
alias mv='mv -i'
alias rm='rm -i'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
```

图 2.33 alias 命令执行结果

如果用户习惯了 DOS 下的命令，可以定义下列命令别名，使用户可以像使用 DOS 环境一样使用 Linux 下的资源，进行相应的操作。

```
#alias dir='ls'
#alias md='mkdir'
#alias rd='rmdir'
#alias type='cat'
```

别名定义完成以后，就可以直接使用了。

说明：

在定义别名的时候需要注意，如果定义的别名在原来系统中已经有该命令，则定义完成的功能则会优先于原来系统的本意执行，类似于局部变量的定义覆盖全局变量的值。在这种情况下，如果需要执行系统本身关于该命令的本意时，则需要加转义字符“\”。

例如，系统中原有命令“ls”显示目录下的内容。如果自己定义别名如下：

```
alias  ls='more /etc/inittab'
```

定义完成以后，当输入 ls 命令时，系统会执行命令：more /etc/inittab，如图 2.34 所示，而不是显示当前目录下的内容。

```
[root@localhost ~]# ls
# inittab       This file describes how the INIT process should set up
#               the system in a certain run-level.
#
# Author:       Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#               Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

--More--(39%)
```

图 2.34　more /etc/inittab 执行结果

如果希望系统按 ls 命令原有的含义执行该命令，则应该在命令行做如下操作：

```
#\ls
mydir
```

系统会显示当前目录下的内容。

当用户要取消关于别名的定义时，使用 unalias 命令。

命令格式：

```
#unalias alias_name
```

alias_name 是用户用 alias 命令定义的命令别名。例如，可以通过下面的语句取消别名 ls 的定义：

```
#unalias ls
```

3. 命令历史

bash 自身可以记录一定数目的、以前在 Shell 中执行过的命令。Shell 能够记录的 Shell 命令的个数由环境变量 HISTSIZE 的值所指定，该变量在配置文件/etc/profile 中定义，在 Red Hat Enterprise Linux 5 中默认该值为 1000。记录这些执行过历史命令的文本文件是由环境变量 HISTFILE 指定的，默认情况下都是用户自己 home 目录下的系统文件. bash_history。

Bash 将历史命令记录以后，用户能够通过两种方式使用这些命令：

使用上下方向键、PgUp、PgDn 键来查询执行。

使用 history 命令显示历史命令，通过命令行的操作 $！＜命令编号＞能够重新执行历史命令。

查看历史命令方法如图 2.35 所示。

```
[root@localhost ~]# history
    1  service httpd restart
    2  cd /etc
    3  cd httpd
    4  cd conf
    5  less http
    6  less http*
    7  less httpd.conf
    8  cd /mnt
    9  ld
   10  ls
   11  cd ..
   12  cd media/
   13  ls
   14  cd RHEL_5.5\ i386\ DVD/
   15  ls
   16  cd Server/
   17  ls
   18  rpm -q  httpd
   19  ls
   20  cd red5/
   21  ls
```

图 2.35　history 命令执行结果

重新执行，查看环境变量的命令，只需在命令行输入：

#!12

就可以成功快速执行，而无需翻屏查找命令。

2.5.2　文件系统的自动挂载

通过 mount 命令手动挂载的文件系统，在系统关机的时候会被自动卸载，在下次系统启动以后，该文件系统不能被自动挂载。如果需要文件系统被自动挂载，则在系统配置文件/etc/fstab 中添加对于该文件系统的挂载信息。

/etc/fstab 是系统实现自动挂载的配置文件。该文件记录了在系统启动的过程中需要自动挂载的文件系统、挂载点、文件系统类型、挂载权限等，如图 2.36 所示。

```
[root@localhost ~]# more /etc/fstab
/dev/VolGroup00/LogVol00 /                       ext3    defaults        1 1
LABEL=/boot              /boot                   ext3    defaults        1 2
tmpfs                    /dev/shm                tmpfs   defaults        0 0
devpts                   /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                    /sys                    sysfs   defaults        0 0
proc                     /proc                   proc    defaults        0 0
/dev/VolGroup00/LogVol01 swap                    swap    defaults        0 0
```

图 2.36　/etc/fstab 文件

文件中每一项的含义分别按顺序说明如下：

(1)要挂载的设备，一般情况下是设备文件，比如/dev/sda5。

(2)挂载点，一般在/mnt/目录下。

(3)挂载时候的文件系统类型，例如 ext3。

(4)挂载时的权限，rw，ro 等。

(5)使用 dump 命令备份文件系统的频率,如果该值为 0 或者空白,表明该文件系统不需要备份。

(6)系统开机时 fsck 命令会对文件系统进行检查,最后一位的数值规定了检查的优先顺序。挂载到分区的文件系统,该位为 1。其他文件系统该位为 2,0 表明不需要对该文件系统进行检查。

如果需要在系统启动的时候自动挂载某个文件系统,则可以在/etc/fstab 中做相应的设置。例如,如果想将项目 2.5 中新添加的硬盘配置成开机自动挂载,则需要在文件/etc/fstab 中添加如图 2.37 所示最后两行内容。

```
/dev/VolGroup00/LogVol00 /                       ext3    defaults        1 1
LABEL=/boot              /boot                   ext3    defaults        1 2
tmpfs                    /dev/shm                tmpfs   defaults        0 0
devpts                   /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                    /sys                    sysfs   defaults        0 0
proc                     /proc                   proc    defaults        0 0
/dev/VolGroup00/LogVol01 swap                    swap    defaults        0 0
/dev/sdb1                /mnt/sdb1               auto    defaults        0 0
/dev/sdb2                /mnt/sdb2               auto    defaults        0 0
```

图 2.37　/etc/fstab 中添加的内容

这样,在系统启动以后,会自动把新硬盘上的两个分区分别按正确的文件系统类型挂载到/mnt/sdb1 和/mnt/sdb2下,进入系统后直接在挂载点下使用新硬盘即可。

2.6　习题与项目训练

1.选择题

(1)以下说法正确的是(　　)

A. Linux 命令是不区分大小写的　　B. Linux 操作系统的文件系统类型是 NTFS

C. Linux 内核代码是开源的　　D. Linux 操作系统有多个根目录

(2)以下文件中表示 Linux 隐藏文件的是(　　)

A. test　　B. /tmp/book

C. . filebrc　　D. /mylog

(3)Linux 操作系统的根目录是(　　)

A. C:\　　B. /

C. /home　　D. root

2.在/home 目录下创建/home/user 目录,并在该目录下创建文件 test1. txt,test2. txt,并将 test2 拷贝到/home/share/test 目录下,然后删除 home/user 下的 test2. txt 文件。

3.找出/var/log 目录下所有后缀是. log 的文件,并将这些文件移动到/home/dustbin 目录下,并将/home/bustbin 目录删除。

4.上机完成下面的操作

(1)以 root 身份登录,完成以下操作:

添加两个用户 user1 和 user2。两个用户均加入到 student 组,如果 student 组不存在,创建 sutdent 组,如果用户已存在,将用户及其主目录一并删除,再添加用户。

为 user1 和 user2 设置初始密码 1q2w3a4f。

在 / 目录下创建名为 usershare 的目录，并将此目录的访问权限修改为对于所有用户都可读、写、执行。

(2)以 user1 身份登录，完成以下操作：

修改密码(修改成功即可)。

在/usershare 目录下创建 bin.c 文件，内容如下：

```
#include <stdio.h>
int main()
{
    printf("user1's bin\n");
    return 0;
}
```

(3)以 user2 身份登录，完成以下操作：

①修改 bin.c 文件内容为：

```
#include <stdio.h>
int main()
{
    printf("user2's bin\n");
    return 0;
}
```

②创建/usershare/user2home/目录。

③将 bin.c 文件拷贝到/usershare/user2home/目录下，并更名为 user2bin.c。

(4)再次以 root 身份登录系统，完成以下操作：

删除/usershare 目录。

删除用户 user1、user2，并同时删除它们的主目录。

5.完成通信录的编辑：

管理员 root 需要创建一个 users 用户组，该组拥有 usera，userb，userc 三个用户，在/tmp 下创建 users 文件。

各用户完成下列功能：

登录系统后，在统一的 users 文件中注册个人信息(包括：用户名，真实姓名，性别，联系方式，家庭住址等)。

注册完成后，各用户将该文件备份到各自的主目录中。

root 用户取消其他用户对 users 文件的写权限。