小標題的圖示硬碟組成與分割的複習

磁碟盤上的物理組成則為(假設此磁碟為單碟片，磁碟盤圖示請參考[第三章圖2.2.1的示意](http://linux.vbird.org/linux_basic/0130designlinux.php#fig2.2.1))：

* 磁區(Sector)為最小的物理儲存單位，每個磁區為512 bytes；
* 將磁區組成一個圓，那就是磁柱(Cylinder)，磁柱是分割槽(partition)的最小單位；
* 第一個磁區裡有主要開機區(Master boot record(MBR), 佔446 bytes)及分割表(partition table, 64 bytes)。

各種介面的磁碟在Linux中的檔案名稱分別為：

* /dev/sd[a-p][1-15]：為SCSI, SATA, USB, Flash隨身碟等介面的磁碟檔名；
* /dev/hd[a-d][1-63]：為 IDE介面的磁碟檔名；

磁碟分割意即指定分割槽的啟始與結束磁柱就是了。

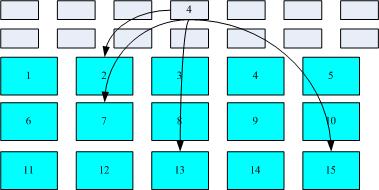
小標題的圖示檔案系統特性

磁碟分割後要進行格式化(format)，之後作業系統才能使用該分割槽。每種作業系統所設定的檔案屬性/權限並不同，為了存放這些檔案，就需將分割槽格式化，以成為作業系統能夠利用的『檔案系統格式(filesystem)』。Linux的正統檔案系統為Ext2(Linux second extended file system, ext2fs)。在預設的情況下，windows是不會認識Linux的Ext2。傳統的磁碟與檔案系統應用中，一個分割槽只能被格式化成為一個檔案系統，即一個filesystem就是一個partition。較新的作業系統的檔案資料除了檔案實際內容外，通常含有其它屬性，如Linux的檔案權限(rwx)與屬性(擁有者、群組、時間參數等)。

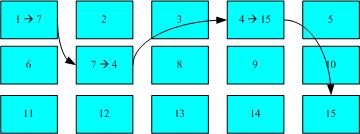
每個inode與block都有編號，至於這三個資料的意義可以簡略說明如下：

* superblock：記錄filesystem的整體資訊，包括inode/block的總量、使用量、剩餘量，以及檔案系統的格式等；
* inode：記錄檔案的屬性，一個檔案佔用一個inode，同時記錄此檔案的資料所在的block號碼；
* block：實際記錄檔案的內容，若檔案太大時，會佔用多個block。

每個inode與block都有編號，而每個檔案會佔一個inode，inode內則有檔案資料放置的block號碼。如下圖所示，檔案系統先格式化出inode與block的區塊，假設某一個檔案的屬性與權限資料是放在inode 4號(下圖較小方格內)，而這個inode記錄了檔案資料的實際放置點為2, 7, 13, 15四個block號碼，此時作業系統就能據此來排列磁碟的讀取順序，將四個block內容讀出來！資料的讀取就如同下圖中的箭頭所指定的模樣。這種資料存取方法稱為索引式檔案系統(indexed allocation)。

  
圖1.2.1、inode/block 資料存取示意圖

隨身碟使用的檔案系統一般為FAT格式。FAT格式的檔案系統並沒有inode存在，所以FAT沒法將這個檔案的所有block在一開始就讀取出來。每個block號碼都記錄在前一個block 當中，他的讀取方式有點像底下這樣：

  
圖1.2.2、FAT檔案系統資料存取示意圖

上圖中假設檔案的資料依序寫入1->7->4->15號這四個block號碼中，但這檔案系統沒法一次就知道四個block的號碼，得要一個個的將block讀出後，才會知道下一個block在何處。如果同一個檔案資料寫入的block太分散，則磁碟讀取頭將無法在磁碟轉一圈就讀到所有的資料，磁碟就會多轉好幾圈才能完整的讀取到檔案的內容！

磁碟重組的原因就是檔案寫入的block太過離散，檔案讀取效能變差。透過磁碟重組將同一個檔案所屬的blocks彙整在一起，資料的讀取會比較容易！

* data block (資料區塊)

data block是用來放置檔案內容資料地方，在Ext2檔案系統中所支援的block大小有1K, 2K及4K三種。格式化時block的大小就固定了，且每個block都有編號以便inode記錄。

* inode table (inode 表格)

如前所述inode的內容在記錄檔案的屬性以及該檔案實際資料是放置在哪幾號block內！基本上，inode的數量與大小也是在格式化時就已經固定了，除此之外inode還有些什麼特色呢？

* 每個inode大小均固定為128 bytes；
* 每個檔案都僅佔用一個inode，因此檔案系統能夠建立的檔案數量與inode的數量有關；
* 系統讀取檔案時需先找到inode，並分析inode所記錄的權限與使用者是否符合，若符合才能實際讀取block的內容。

* Superblock (超級區塊)

Superblock是記錄整個filesystem資訊的地方，沒有Superblock，就沒這個filesystem。他記錄的主要資訊有：

* block與inode的總量；
* 未使用與已使用的inode/block數量；
* block與inode的大小(block為1, 2, 4K，inode為128 bytes)；
* filesystem的掛載時間、最近一次寫入資料的時間、最近一次檢驗磁碟(fsck)的時間等檔案系統的相關資訊；
* 一個valid bit數值，若此檔案系統已被掛載，則valid bit為0，若未被掛載，則valid bit為1。

Superblock非常重要，因為檔案系統的基本資訊都寫在這裡，如果superblock死掉了，檔案系統可能就花費很多時間去挽救！一般來說，superblock的大小為1024bytes。

每個區段與superblock的資訊都可以使用dumpe2fs指令來查詢！方法如下：

|  |
| --- |
| [root@www ~]# dumpe2fs [-bh] 裝置檔名  選項與參數：  -b ：列出保留為壞軌的部分(一般用不到吧！？)  -h ：僅列出superblock的資料，不會列出其他的區段內容！  範例：找出我的根目錄磁碟檔名，並觀察檔案系統的相關資訊  [root@www ~]# df <==這個指令可以叫出目前掛載的裝置  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/hdc2 9920624 3822848 5585708 41% / <==就是這個光！  /dev/hdc3 4956316 141376 4559108 4% /home  /dev/hdc1 101086 11126 84741 12% /boot  tmpfs 371332 0 371332 0% /dev/shm  [root@www ~]# dumpe2fs /dev/hdc2  dumpe2fs 1.39 (29-May-2006)  Filesystem volume name: /1 <==這個是檔案系統的名稱(Label)  Filesystem features: has\_journal ext\_attr resize\_inode dir\_index  filetype needs\_recovery sparse\_super large\_file  Default mount options: user\_xattr acl <==預設掛載的參數  Filesystem state: clean <==這個檔案系統是沒問題的(clean)  Errors behavior: Continue  Filesystem OS type: Linux  Inode count: 2560864 <==inode的總數  Block count: 2560359 <==block的總數  Free blocks: 1524760 <==還有多少個 block 可用  Free inodes: 2411225 <==還有多少個 inode 可用  First block: 0  Block size: 4096 <==每個 block 的大小啦！  Filesystem created: Fri Sep 5 01:49:20 2008 |

* 目錄

Linux下的ext2檔案系統建立一個目錄時，ext2會分配一個inode與至少一塊block給該目錄。其中，inode記錄該目錄的相關權限與屬性，並可記錄分配到的那塊block號碼；而block則是記錄在這個目錄下的檔名與該檔名佔用的inode號碼資料。也就是說目錄所佔用的block內容在記錄如下的資訊：

  
圖1.4.1、目錄佔用的 block 記錄的資料示意圖

如果想要實際觀察root家目錄內的檔案所佔用的inode號碼時，可以使用ls -i來處理：

|  |
| --- |
| [root@www ~]# ls –li  total 92  654683 -rw------- 1 root root 1474 Sep 4 18:27 anaconda-ks.cfg  648322 -rw-r--r-- 1 root root 42304 Sep 4 18:26 install.log  648323 -rw-r--r-- 1 root root 5661 Sep 4 18:25 install.log.syslog |

* 檔案：

在Linux下的ext2建立一個一般檔案時，ext2會分配一個inode與相對於該檔案大小的block數量給該檔案。例如：假設一個block為4 Kbytes，而要建立一個100 KBytes的檔案，那麼linux將分配一個inode與25個block來儲存該檔案！ 但同時請注意，由於inode僅有12個直接指向，因此還要多一個block來作為區塊號碼的記錄！

* 目錄樹讀取：

inode本身不記錄檔名，檔名的記錄是在目錄的block中，因此要讀取某個檔案時，必會經過目錄的inode與block，然後再找到那待讀取檔案的inode號碼，最終才會讀到正確檔案block內的資料。由於目錄樹是由根目錄開始讀起，因此系統透過掛載的資訊可找到掛載點的inode號碼，此時就能夠得到根目錄的inode內容，並依該inode讀取根目錄的block內的檔名資料，再一層層的往下讀到正確的檔名。舉例來說，如果要讀取/etc/passwd這個檔案時，系統是如何讀取的呢？

* filesystem 大小與磁碟讀取效能：

關於檔案系統的使用效率上，當檔案系統規劃的很大時，如100GB時，硬碟上的資料總是來來去去，整個檔案系統上的檔案通常無法連續寫在一起(block號碼不會連續)，而是填入式的將資料填入沒有被使用的block當中。如果檔案寫入的block分的很散，會有所謂的檔案資料離散。因此partition的規劃並不是越大越好，而是真的要針對主機用途來進行規劃！

* 日誌式檔案系統 (Journaling filesystem)

為了避免檔案系統不一致的情況發生，因此如果在filesystem中規劃一區塊，該區塊專門在記錄寫入或修訂檔案時的步驟，就可以簡化一致性檢查的步驟了？也就是說：

1. 預備：當系統要寫入一個檔案時，會先在日誌記錄區塊中紀錄某個檔案準備要寫入的資訊；
2. 實際寫入：開始寫入檔案的權限與資料；開始更新metadata的資料；
3. 結束：完成資料與 metadata 的更新後，在日誌記錄區塊當中完成該檔案的紀錄。

萬一資料的紀錄過程當中發生了問題，那系統只要去檢查日誌記錄區塊，就可以知道哪個檔案發生了問題，針對該問題來做一致性的檢查即可，而不必針對整塊filesystem去檢查，就可以達到快速修復filesystem的能力！ext3是ext2的升級版本，可向下相容ext2版本！[dumpe2fs](http://linux.vbird.org/linux_basic/0230filesystem.php#dumpe2fs)輸出的訊息，可發現superblock裡面含有底下這樣的資訊：

|  |
| --- |
| Journal inode: 8  Journal backup: inode blocks  Journal size: 128M |

透過inode 8號記錄journal區塊的block指向，而且具有128MB的容量在處理日誌呢！

小標題的圖示掛載點的意義 (mount point)：

每個filesystem都有獨立的inode/block/superblock等資訊，檔案系統要能夠連結到目錄樹才能被使用。將檔案系統與目錄樹結合的動作稱為『掛載』。關於掛載的重點是：掛載點一定是目錄，該目錄為進入該檔案系統的入口。因此並不是任何檔案系統都能使用，必須要『掛載』到目錄樹的某個目錄後，才能使用該檔案系統。

小標題的圖示其他Linux支援的檔案系統與VFS

Linux支援很多檔案系統格式的，如Windows的FAT檔案系統等，都被Linux所支援！常見的支援檔案系統有：

* 傳統檔案系統：ext2/minix/MS-DOS/FAT (用 vfat 模組)/iso9660 (光碟)等等；
* 日誌式檔案系統： ext3/ReiserFS/Windows' NTFS/IBM's JFS/SGI's XFS
* 網路檔案系統： NFS/SMBFS

想要知道Linux支援的檔案系統有哪些，可以察看底下這個目錄：

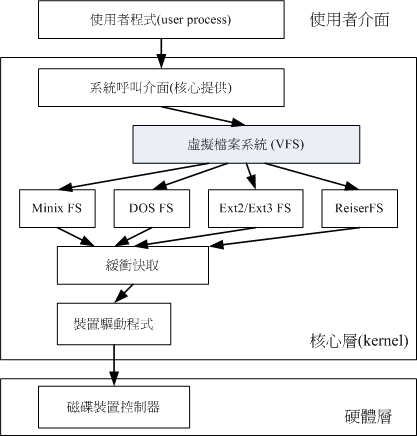
|  |
| --- |
| [root@www ~]# ls -l /lib/modules/$(uname -r)/kernel/fs |

系統目前已載入到記憶體中支援的檔案系統則有：

|  |
| --- |
| [root@www ~]# cat /proc/filesystems |

* Linux VFS (Virtual Filesystem Switch)

整個Linux的系統都是透過Virtual Filesystem Switch的核心功能去讀取filesystem的。Linux認識的filesystem其實都是VFS在進行管理，使用者不需要知道partition上的filesystem是什麼～VFS會主動做好讀取的動作～假設/是/dev/hda1，用ext3，而/home用/dev/hda2，那麼你取用/home/dmtsai/.bashrc時，沒有特別指定要用的什麼檔案系統的模組來讀取！這個就是VFS的功能！整個VFS可以約略用下圖來說明：

  
圖 1.8.1、VFS 檔案系統的示意圖

小標題的圖示磁碟與目錄的容量：

底下就讓我們來談一談這兩個指令：df和du。

* df：列出檔案系統的整體磁碟使用量

|  |
| --- |
| [root@www ~]# df [-ahikHTm] [目錄或檔名]  選項與參數：  -a ：列出所有的檔案系統，包括系統特有的/proc等檔案系統；  -k ：以 KBytes 的容量顯示各檔案系統；  -m ：以 MBytes 的容量顯示各檔案系統；  -h ：以人們較易閱讀的GBytes, MBytes, KBytes等格式自行顯示；  -H ：以 M=1000K 取代 M=1024K 的進位方式；  -T ：連同該 partition 的 filesystem 名稱 (例如 ext3) 也列出；  -i ：不用硬碟容量，而以 inode 的數量來顯示  範例一：將系統內所有的filesystem列出來！並以易讀的容量格式顯示出來  [root@www ~]# df -h  Filesystem Size Used Avail Use% Mounted on  /dev/hdc2 9.5G 3.7G 5.4G 41% /  /dev/hdc3 4.8G 139M 4.4G 4% /home  /dev/hdc1 99M 11M 83M 12% /boot  tmpfs 363M 0 363M 0% /dev/shm  # 都是在記憶體當中，例如/proc這個掛載點。因此，這些特殊的檔案系統不會佔據硬碟空間！  範例二：將目前各個 partition 當中可用的 inode 數量列出  [root@www ~]# df -ih  Filesystem Inodes IUsed IFree IUse% Mounted on  /dev/hdc2 2.5M 147K 2.3M 6% /  /dev/hdc3 1.3M 46 1.3M 1% /home  /dev/hdc1 26K 34 26K 1% /boot  tmpfs 91K 1 91K 1% /dev/shm  # 這個範例主要列出可用的inode剩餘量與總容量。分析一下與範例一的關係，通常inode的數量剩餘都比block還要多 |

先來說明一下範例一所輸出的結果訊息為：

* Filesystem：代表該檔案系統是在哪個partition，所以列出裝置名稱；
* 1k-blocks：說明底下的數字單位是1KB呦！可利用-h或-m來改變容量；
* Used：顧名思義，就是使用掉的硬碟空間啦！
* Available：也就是剩下的磁碟空間大小；
* Use%：就是磁碟的使用率啦！如果使用率高達 90% 以上時， 最好需要注意一下了，免得容量不足造成系統問題喔！(例如最容易被灌爆的 /var/spool/mail 這個放置郵件的磁碟)
* Mounted on：就是磁碟掛載的目錄所在！(掛載點！)

df讀取的資料幾乎是針對一整個檔案系統，讀取的範圍主要是在Superblock內的資訊，所以顯示結果非常快速！如果使用-a參數時，系統會出現/proc這個掛載點，但是裡面的東西都是0！/proc是Linux系統所需要載入的系統資料，而且是掛載在『記憶體當中』的，所以沒有佔任何的硬碟空間！至於/dev/shm/目錄，是利用記憶體虛擬出來的磁碟空間！由於透過記憶體模擬出來的磁碟，因此這個目錄下建立任何資料檔案時，存取速度是非常快速的！(在記憶體內工作)不過也由於是記憶體模擬出來的，因此這個檔案系統的大小在每部主機上都不一樣，且建立的東西在下次開機時就消失了！因為在記憶體中！

* du：評估檔案系統的磁碟使用量(常用在推估目錄所佔容量)

|  |
| --- |
| [root@www ~]# du [-ahskm] 檔案或目錄名稱  選項與參數：  -a ：列出所有的檔案與目錄容量，因為預設僅統計目錄底下的檔案量而已。  -h ：以人們較易讀的容量格式 (G/M) 顯示；  -s ：列出總量而已，而不列出每個各別的目錄佔用容量；  -S ：不包括子目錄下的總計，與 -s 有點差別。  -k ：以 KBytes 列出容量顯示；  -m ：以 MBytes 列出容量顯示；  範例一：列出目前目錄下的所有檔案容量  [root@www ~]# du  8 ./test4 <==每個目錄都會列出來  8 ./test2  ....中間省略....  12 ./.gconfd <==包括隱藏檔的目錄  220 . <==這個目錄(.)所佔用的總量  # 沒加任何選項時，du會分析『目前所在目錄』的檔案與目錄佔用的硬碟空間。但實際顯示時僅會顯示目錄容量(不含檔案)，  # . 目錄有很多檔案沒被列出來，所以全部的目錄相加不會等於 . 的容量！此外，輸出的數值資料為1K大小的容量單位。  範例二：同範例一，但是將檔案的容量也列出來  [root@www ~]# du -a  12 ./install.log.syslog <==有檔案的列表了  8 ./.bash\_logout  8 ./test4  8 ./test2  ....中間省略....  12 ./.gconfd  220 . |

與df不一樣，du會直接到檔案系統內去搜尋所有的檔案資料，所以指令的運作會執行一小段時間！此外，在預設的情況下，容量的輸出是以KB來設計的，如果想知道目錄佔了多少MB，就使用-m即可！如果只想知道該目錄佔了多少容量的話，使用-s就可以！至於-S這個選項部分，由於du預設會將所有檔案的大小均列出，因此假設你在/etc底下使用du時，所有的檔案大小，包括/etc底下的次目錄容量也會被計算一次。然後最終的容量(/etc)也會加總一次， 因此很多朋友都會誤會du分析的結果不太對勁。所以如果想要列出某目錄下的全部資料，或許也可以加上-S的選項，減少次目錄的加總喔！

小標題的圖示實體連結與符號連結： ln

在Linux底下的連結檔有兩種，一種是類似Windows的捷徑功能的檔案，可快速的連結到目標檔案(或目錄)；另一種則是透過檔案系統的inode連結來產生新檔名，而不是產生新檔案！這種稱為實體連結(hard link)。

* Hard Link (實體連結, 硬式連結或實際連結)

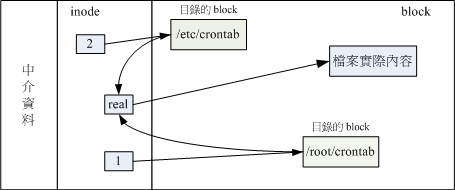
在前一小節當中，我們知道幾件重要的資訊，包括：

* 每個檔案都會佔用一個inode，檔案內容由inode的記錄來指向；
* 想要讀取該檔案，必須要經過目錄記錄的檔名來指向到正確的inode號碼才能讀取。

有沒可能多個檔名對應到同一個inode號碼呢？有的！那就是hard link的由來。簡單的說：hard link是在某個目錄下新增一筆檔名連結到某inode號碼的關連記錄。舉例來說，假設系統有個/root/crontab是/etc/crontab的實體連結，也就說這兩個檔名連結到同一個inode，自然這兩個檔名的所有相關資訊都會一樣(除了檔名之外)。實際的情況可以如下所示：

|  |
| --- |
| [root@www ~]# ln /etc/crontab . <==建立實體連結的指令  [root@www ~]# ll -i /etc/crontab /root/crontab  1912701 -rw-r--r-- 2 root root 255 Jan 6 2007 /etc/crontab  1912701 -rw-r--r-- 2 root root 255 Jan 6 2007 /root/crontab |

兩個檔名都連結到1912701這個inode號碼，所以檔案的權限/屬性完全一樣。第二個欄位由原本的1變成2了！那個欄位稱為『連結』，意義為：『有多少個檔名連結到這個inode號碼』。成示意圖類似如下畫面：

  
圖 2.2.1、實體連結的檔案讀取示意圖

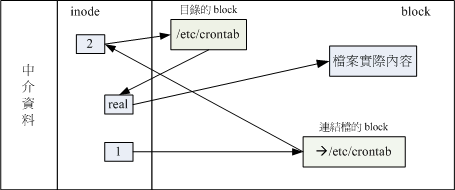
上圖的意思是，透過1或2的目錄之inode指定的block找到兩個不同的檔名，不管哪個檔名均可指到real那個inode去讀最終資料！如果將任一個『檔名』刪除，其實inode與block都還是存在的！透過另一個『檔名』來讀取到正確的檔案資料！不論使用哪個『檔名』來編輯，最終的結果都會寫入到相同的inode與block中，因此均能進行資料的修改！

* Symbolic Link (符號連結，亦即是捷徑)

Symbolic link是建立一個獨立的檔案，而這個檔案會讓資料的讀取指向他link那個檔案的檔名！由於只是利用檔案來做為指向的動作，所以，當來源檔被刪除之後，symbolic link的檔案會『開不了』，會一直說『無法開啟某檔案！』。實際上就是找不到原始『檔名』而已啦！舉例來說，我們先建立一個符號連結檔連結到 /etc/crontab 去看看：

|  |
| --- |
| [root@www ~]# ln -s /etc/crontab crontab2  [root@www ~]# ll -i /etc/crontab /root/crontab2  1912701 -rw-r--r-- 2 root root 255 Jan 6 2007 /etc/crontab  654687 lrwxrwxrwx 1 root root 12 Oct 22 13:58 /root/crontab2 -> /etc/crontab |

上表的結果可知兩個檔案指向不同的inode號碼，當然就是兩個獨立的檔案存在！而且連結檔的重要內容就是他會寫上目標檔案的『檔名』，為什麼上表中連結檔的大小為12 bytes呢？因為箭頭(-->)右邊的檔名『/etc/crontab』共有12個英文，每個英文佔用1個bytes，所以檔案大小就是12bytes了！我們以如下圖示來解釋：

  
圖 2.2.2、符號連結的檔案讀取示意圖

由1號inode讀取到連結檔的內容僅有檔名，根據檔名連結到正確的目錄去取得目標檔案的inode，最終就能讀到正確的資料。如果目標檔案(/etc/crontab)被刪除了，那麼整個環節就會無法繼續進行下去，所以就無法透過連結檔讀取！ Symbolic Link與Windows的捷徑可以劃上等號。由於Hard Link 的限制太多了，包括無法做『目錄』的 link，在用途上面是比較受限的！反而Symbolic Link的使用方面較廣喔！

大標題的圖示磁碟的分割、格式化、檢驗與掛載：

對於一個系統管理者(root)而言，如果想要在系統裡面新增一顆硬碟時，應該有哪些動作需要做的呢：

1. 對磁碟進行分割，以建立可用的partition；
2. 對該partition進行格式化(format)，以建立系統可用的filesystem；對剛剛建立好的filesystem進行檢驗；
3. 在Linux系統上，建立掛載點(亦即是目錄)，並將他掛載上來；

小標題的圖示磁碟分割： fdisk

|  |
| --- |
| [root@www ~]# fdisk [-l]裝置名稱  選項與參數：  -l ：輸出後接裝置所有的partition內容。若僅有fdisk -l時，系統將會把整個系統內能夠搜尋到裝置的partition均列出來。  範例：找出你系統中的根目錄所在磁碟，並查閱該硬碟內的相關資訊  [root@www ~]# df / <==注意：重點在找出磁碟檔名而已  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/hdc2 9920624 3823168 5585388 41% /  [root@www ~]# fdisk /dev/hdc <==仔細看，不要加上數字喔！  The number of cylinders for this disk is set to 5005.  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)  Command (m for help): <==等待你的輸入！ |

先用df找出可用磁碟檔名，然後再用fdisk查閱。進入fdisk的工作畫面後，如果硬碟太大(通常指磁柱數量多於1024以上)，會出現如上訊息。這訊息僅是在告知，因為某些舊版的軟體與作業系統並無法支援大於1024磁柱(cylinter)後的磁區使用，不過新版Linux沒問題！不管進行什麼動作，只要離開fdisk時按下『q』，所有的動作『都不會生效！』相反的，按下『w』就是動作生效的意思。

* 刪除磁碟分割槽

如果想要測試一下如何將你的/dev/hdc全部的分割槽刪除，應該怎麼做？

1. fdisk /dev/hdc ：先進入 fdisk 畫面；
2. p ：先看一下分割槽的資訊，假設要殺掉 /dev/hdc1；
3. d ：這個時候會要選擇一個partition，就選1囉！
4. w (or) q ：按w可儲存到磁碟資料表中，並離開fdisk；如果反悔了，直接按下q就可以取消剛剛的刪除動作了！

|  |
| --- |
| # 練習一： 先進入fdisk的畫面當中去！  [root@www ~]# fdisk /dev/hdc  # 練習二： 先看看整個分割表的情況是如何  Command (m for help): p  Disk /dev/hdc: 41.1 GB, 41174138880 bytes  255 heads, 63 sectors/track, 5005 cylinders  Units = cylinders of 16065 \* 512 = 8225280 bytes  Device Boot Start End Blocks Id System  /dev/hdc1 \* 1 13 104391 83 Linux  /dev/hdc2 14 1288 10241437+ 83 Linux  /dev/hdc3 1289 1925 5116702+ 83 Linux  /dev/hdc4 1926 5005 24740100 5 Extended  /dev/hdc5 1926 2052 1020096 82 Linux swap / Solaris  # 練習三： 按下 d 給他刪除吧！  Command (m for help): d  Partition number (1-5): 4  Command (m for help): d  Partition number (1-4): 3  Command (m for help): p  Disk /dev/hdc: 41.1 GB, 41174138880 bytes  255 heads, 63 sectors/track, 5005 cylinders  Units = cylinders of 16065 \* 512 = 8225280 bytes  Device Boot Start End Blocks Id System  /dev/hdc1 \* 1 13 104391 83 Linux  /dev/hdc2 14 1288 10241437+ 83 Linux  # 因為/dev/hdc5是由/dev/hdc4衍生出的邏輯分割槽，因此/dev/hdc4被刪除，/dev/hdc5就自動不見了！  Command (m for help): q |

* 練習新增磁碟分割槽

新增磁碟分割槽有好多種情況，因為新增Primary/Extended/Logical的顯示結果都不太相同。

|  |
| --- |
| # 練習一： 開始新增，先新增一個Primary的分割槽，且指定為4號看看！  Command (m for help): n  Command action <==因為是全新磁碟，因此只會問extended/primary而已  e extended  p primary partition (1-4)  p <==選擇 Primary 分割槽  Partition number (1-4): 4 <==設定為 4 號！  First cylinder (1-5005, default 1): <==直接按下[enter]按鍵決定！  Using default value 1 <==啟始磁柱就選用預設值！  Last cylinder or +size or +sizeM or +sizeK (1-5005, default 5005): +512M  # 我們知道partition是由n1到n2的磁柱號碼(cylinder)，但磁柱的大小每顆磁碟都不相同，這時可填入+512M來讓  # 系統自動幫我們找出『最接近512M的那個cylinder 號碼』！不可能剛好等於512MBytes！  # 如上所示：這個地方輸入的方式有兩種：  # 1) 直接輸入磁柱的號碼，你得要自己計算磁柱/分割槽的大小才行；  # 2) 用 +XXM 來輸入分割槽的大小，讓系統自己捉磁柱的號碼。+與M是必須要有的，XX為數字  Command (m for help): p  Disk /dev/hdc: 41.1 GB, 41174138880 bytes  255 heads, 63 sectors/track, 5005 cylinders  Units = cylinders of 16065 \* 512 = 8225280 bytes  Device Boot Start End Blocks Id System  /dev/hdc4 1 63 506016 83 Linux  # 注意！只有 4 號！ 1 ~ 3 保留下來了！ |

|  |
| --- |
| # 練習三： 繼續新增一個，這次我們新增 Extended 的分割槽好了！  Command (m for help): n  Command action  e extended  p primary partition (1-4)  e <==選擇的是 Extended 喔！  Partition number (1-4): 1  First cylinder (64-5005, default 64): <=[enter]  Using default value 64  Last cylinder or +size or +sizeM or +sizeK (64-5005, default 5005): <=[enter]  Using default value 5005  # 延伸分割最好能包含所有未分割的區間；在開始/結束磁柱的位置上，按下兩個[enter]用預設值即可！  Command (m for help): p  Disk /dev/hdc: 41.1 GB, 41174138880 bytes  255 heads, 63 sectors/track, 5005 cylinders  Units = cylinders of 16065 \* 512 = 8225280 bytes  Device Boot Start End Blocks Id System  /dev/hdc1 64 5005 39696615 5 Extended  /dev/hdc4 1 63 506016 83 Linux  # 如上所示，所有的磁柱都在 /dev/hdc1 裡面囉！ |

|  |
| --- |
| # 練習四： 這次我們隨便新增一個 2GB 的分割槽看看！  Command (m for help): n  Command action  l logical (5 or over) <==因為已有 extended ，所以出現 logical 分割槽  p primary partition (1-4)  p <==偷偷玩一下，能否新增主要分割槽  Partition number (1-4): 2  No free sectors available <==肯定不行！因為沒有多餘的磁柱可供配置  Command (m for help): n  Command action  l logical (5 or over)  p primary partition (1-4)  l <==乖乖使用邏輯分割槽吧！  First cylinder (64-5005, default 64): <=[enter]  Using default value 64  Last cylinder or +size or +sizeM or +sizeK (64-5005, default 5005): +2048M  Command (m for help): p  Disk /dev/hdc: 41.1 GB, 41174138880 bytes  255 heads, 63 sectors/track, 5005 cylinders  Units = cylinders of 16065 \* 512 = 8225280 bytes  Device Boot Start End Blocks Id System  /dev/hdc1 64 5005 39696615 5 Extended  /dev/hdc4 1 63 506016 83 Linux  /dev/hdc5 64 313 2008093+ 83 Linux  # 這樣就新增了2GB的分割槽，且由於是 logical，所以由5號開始！  Command (m for help): q |

* 操作環境的說明

以root的身份進行硬碟的partition時，最好是在單人維護模式底下比較安全一些，在進行fdisk的時候，如果該硬碟某個 partition還在使用當中，那麼很有可能系統核心會無法重新載入硬碟的partition table，解決的方法就是將該使用中的 partition卸載，然後再重新進入fdisk一遍，重新寫入partition table，那麼就可以成功囉！

小標題的圖示磁碟格式化

分割完畢後就要進行檔案系統的格式化！格式化的指令是『make filesystem, mkfs』！這個指令是個綜合指令，他會去呼叫正確的檔案系統格式化工具軟體！

* mkfs

|  |
| --- |
| [root@www ~]# mkfs [-t 檔案系統格式] 裝置檔名  選項與參數：  -t ：可以接檔案系統格式，例如 ext3, ext2, vfat 等(系統有支援才會生效)  範例一：請將上個小節當中所製作出來的/dev/hdc6格式化為ext3檔案系統  [root@www ~]# mkfs -t ext3 /dev/hdc6  mke2fs 1.39 (29-May-2006)  Filesystem label= <==這裡指的是分割槽的名稱(label)  OS type: Linux  Block size=4096 (log=2) <==block 的大小設定為 4K  Fragment size=4096 (log=2)  251392 inodes, 502023 blocks <==由此設定決定的inode/block數量  25101 blocks (5.00%) reserved for the super user  First data block=0  Maximum filesystem blocks=515899392  16 block groups  32768 blocks per group, 32768 fragments per group  15712 inodes per group  Superblock backups stored on blocks:  32768, 98304, 163840, 229376, 294912  Writing inode tables: done  Creating journal (8192 blocks): done <==有日誌記錄  Writing superblocks and filesystem accounting information: done  This filesystem will be automatically checked every 34 mounts or 180 days, whichever comes first. Use tune2fs -c or -i to override.  # 這樣就建立起來我們所需要的Ext3檔案系統了！簡單明瞭！  [root@www ~]# mkfs[tab][tab]  mkfs mkfs.cramfs mkfs.ext2 mkfs.ext3 mkfs.msdos mkfs.vfat  # 按下兩個[tab]，會發現mkfs支援的檔案格式如上所示！可以格式化vfat喔！ |

mkfs是綜合指令，當使用『mkfs -t ext3 ...』時，系統會呼叫mkfs.ext3進行格式化！vfat可用在Windows/Linux共用的USB隨身碟。

|  |
| --- |
| 例題：將剛剛的/dev/hdc6格式化為Windows可讀的vfat格式吧！  答：mkfs -t vfat /dev/hdc6 |

由於沒有詳細指定檔案系統的細部項目，因此系統會使用預設值來進行格式化。其中比較重要的部分為：檔案系統的標頭(Label)、Block的大小以及inode的數量。

小標題的圖示磁碟掛載與卸載

進行掛載前先確定單一檔案系統不應該被重複掛載在不同的掛載點(目錄)中；單一目錄不應該重複掛載多個檔案系統；要作為掛載點的目錄，理論上應該是空目錄。如果掛載的目錄不是空的，掛載檔案系統後，原目錄下的東西會暫時消失。假設/home原本與根目錄(/)在同一個檔案系統中，底下有/home/test與/home/vbird兩個目錄。想要加入新的硬碟，並且直接掛載/home下，當掛載新分割槽時，/home目錄顯示新分割槽內的資料，至於原先test與vbird目錄會暫時被隱藏，並不是被覆蓋，等新分割槽被卸載後，/home原本的內容就再次出來！將檔案系統掛載到Linux系統上，要使用 mount這個指令！

* 掛載Ext2/Ext3檔案系統

|  |
| --- |
| 範例一：用預設的方式，將剛剛建立的/dev/hdc6掛載到/mnt/hdc6上面！  [root@www ~]# mkdir /mnt/hdc6  [root@www ~]# mount /dev/hdc6 /mnt/hdc6  [root@www ~]# df  Filesystem 1K-blocks Used Available Use% Mounted on  .....中間省略.....  /dev/hdc6 1976312 42072 1833836 3% /mnt/hdc6 |

那麼系統有沒有指定哪些類型的filesystem才需要進行上述的掛載測試呢？主要是參考底下這兩個檔案：

* /etc/filesystems：系統指定的測試掛載檔案系統類型；
* /proc/filesystems：Linux系統已經載入的檔案系統類型。

怎知Linux有沒有相關檔案系統類型的驅動程式呢？Linux支援的檔案系統之驅動程式都寫在如下的目錄中：

* /lib/modules/$(uname -r)/kernel/fs/

例如vfat的驅動程式就寫在『/lib/modules/$(uname -r)/kernel/fs/vfat/』目錄下！

|  |
| --- |
| 範例二：觀察目前『已掛載』的檔案系統，包含各檔案系統的Label名稱  [root@www ~]# mount -l  /dev/hdc2 on / type ext3 (rw) [/1]  proc on /proc type proc (rw)  sysfs on /sys type sysfs (rw)  devpts on /dev/pts type devpts (rw,gid=5,mode=620)  /dev/hdc3 on /home type ext3 (rw) [/home]  /dev/hdc1 on /boot type ext3 (rw) [/boot]  tmpfs on /dev/shm type tmpfs (rw)  none on /proc/sys/fs/binfmt\_misc type binfmt\_misc (rw)  sunrpc on /var/lib/nfs/rpc\_pipefs type rpc\_pipefs (rw)  /dev/hdc6 on /mnt/hdc6 type ext3 (rw) [vbird\_logical]  # 除了實際的檔案系統外，很多特殊的檔案系統(proc/sysfs...)也會被顯示出來！  # 值得注意的是，加上 -l 選項可以列出如上特殊字體的標頭(label)喔 |

以/dev/hdc2這個裝置的意義是：『/dev/hdc2是掛載到/目錄，檔案系統類型為ext3，掛載為可讀寫(rw)，這個filesystem 有標頭，名字(label)為 /1』

* 掛載 CD 或 DVD 光碟

|  |
| --- |
| 範例三：將你用來安裝 Linux的CentOS原版光碟拿出來掛載！  [root@www ~]# mkdir /media/cdrom  [root@www ~]# mount -t iso9660 /dev/cdrom /media/cdrom  [root@www ~]# mount /dev/cdrom /media/cdrom  # 可指定-t iso9660光碟片格式掛載，也可讓系統自己測試掛載！上述指令只要一個就夠！但目錄的建立初次掛載時必須要！  [root@www ~]# df  Filesystem 1K-blocks Used Available Use% Mounted on  .....中間省略.....  /dev/hdd 4493152 4493152 0 100% /media/cdrom  # 因為我的光碟機使用的是 /dev/hdd 的 IDE 介面之故！ |

光碟機一掛載之後就無法退出光碟片了！除非你將他卸載才能夠退出！從上面的資料你也可以發現光碟的磁碟使用率達到 100%，你無法直接寫入任何資料到光碟當中！另外，其實 /dev/cdrom 是個連結檔，以鳥哥為例，我的光碟機接在/dev/hdd，所以正確的掛載應該是『mount /dev/hdd /media/cdrom』比較正確喔！

* 掛載隨身碟

隨身碟不能夠是NTFS的檔案系統喔！ 接下來讓我們測試測試吧！

|  |
| --- |
| 範例五：找出你的隨身碟裝置檔名，並掛載到 /mnt/flash 目錄中  [root@www ~]# fdisk -l  .....中間省略.....  Disk /dev/sda: 8313 MB, 8313110528 bytes  59 heads, 58 sectors/track, 4744 cylinders  Units = cylinders of 3422 \* 512 = 1752064 bytes  Device Boot Start End Blocks Id System  /dev/sda1 1 4745 8118260 b W95 FAT32  # 從上的特殊字體，可得知磁碟的大小以及裝置檔名，知道是 /dev/sda1  [root@www ~]# mkdir /mnt/flash  [root@www ~]# mount -t vfat -o iocharset=cp950 /dev/sda1 /mnt/flash  [root@www ~]# df  Filesystem 1K-blocks Used Available Use% Mounted on  .....中間省略.....  /dev/sda1 8102416 4986228 3116188 62% /mnt/flash |

如果帶有中文檔名的資料，那麼可以在掛載時指定一下掛載檔案系統所使用的語系資料。在 man mount找到vfat檔案格式當中可以使用 iocharset 來指定語系，而中文語系是 cp950，所以也就有了上述的掛載指令項目囉。

* umount (將裝置檔案卸載)

|  |
| --- |
| [root@www ~]# umount [-fn] 裝置檔名或掛載點  選項與參數：  -f ：強制卸載！可用在類似網路檔案系統 (NFS) 無法讀取到的情況下；  -n ：不更新 /etc/mtab 情況下卸載。 |

將已掛載的檔案系統卸載！卸載之後，可用df或mount -l看是否還存在目錄樹中？卸載的方式，可以下達裝置檔名或掛載點！

|  |
| --- |
| 範例八：將本章之前自行掛載的檔案系統全部卸載：  [root@www ~]# mount  .....前面省略.....  /dev/hdc6 on /mnt/hdc6 type ext3 (rw)  /dev/hdd on /media/cdrom type iso9660 (rw)  /dev/sda1 on /mnt/flash type vfat (rw,iocharset=cp950)  /home on /mnt/home type none (rw,bind)  # 先找一下已經掛載的檔案系統，如上所示，特殊字體即為剛剛掛載的裝置囉！  [root@www ~]# umount /dev/hdc6 <==用裝置檔名來卸載  [root@www ~]# umount /media/cdrom <==用掛載點來卸載  [root@www ~]# umount /mnt/flash <==因為掛載點比較好記憶！  [root@www ~]# umount /dev/fd0 <==用裝置檔名較好記！  [root@www ~]# umount /mnt/home <==一定要用掛載點！因為掛載的是目錄 |

由於通通卸載了，此時你才可以退出光碟片、軟碟片、USB隨身碟等設備喔！如果你遇到這樣的情況：

|  |
| --- |
| [root@www ~]# mount /dev/cdrom /media/cdrom  [root@www ~]# cd /media/cdrom  [root@www cdrom]# umount /media/cdrom  umount: /media/cdrom: device is busy  umount: /media/cdrom: device is busy |

由於目前在/media/cdrom/目錄內，也就是說其實『該檔案系統正在使用』！所以無法卸載這個裝置！那該如何是好？就『離開該檔案系統的掛載點』即可。以上述的案例來說，你可以使用『cd /』回到根目錄，就能夠卸載/media/cdrom囉！

* mknod

Linux所有裝置都以檔案代表！透過檔案的major與minor數值來代表裝置～所以major與minor數值是有特殊意義的！

|  |
| --- |
| [root@www ~]# mknod 裝置檔名 [bcp] [Major] [Minor]  選項與參數：  裝置種類：  b ：設定裝置名稱成為一個周邊儲存設備檔案，例如硬碟等；  c ：設定裝置名稱成為一個周邊輸入設備檔案，例如滑鼠/鍵盤等；  Major ：主要裝置代碼；  Minor ：次要裝置代碼；  範例一：由上述的介紹我們知道 /dev/hdc10 裝置代碼 22, 10，請建立並查閱此裝置  [root@www ~]# mknod /dev/hdc10 b 22 10  [root@www ~]# ll /dev/hdc10  brw-r--r-- 1 root root 22, 10 Oct 26 23:57 /dev/hdc10 |

* 建立大檔案以製作 loop 裝置檔案！

能不能製作一個大檔案，然後將這檔案格式化後掛載呢？這能解決很多系統分割不良的情況！舉例來說，如果當初分割時只分割出一根目錄，假設已沒有多餘的容量可進行額外的分割！偏偏根目錄的容量還很大！此時就能夠製作出一個大檔案，然後將這檔案掛載！感覺上就多了個分割槽！底下在/home下建立一個512MB左右的大檔案，然後將這檔案格式化並且實際掛載！

* 建立大型檔案: 首先得先有一個大的檔案！怎麼建立這個大檔案呢？在Linux底下有一支很好用的程式[dd](http://linux.vbird.org/linux_basic/0240tarcompress.php#dd)！可以用來建立空的檔案。假設我要建立一個空的檔案在 /home/loopdev，那可以這樣做：

|  |
| --- |
| [root@www ~]# dd if=/dev/zero of=/home/loopdev bs=1M count=512  512+0 records in <==讀入 512 筆資料  512+0 records out <==輸出 512 筆資料  536870912 bytes (537 MB) copied, 12.3484 seconds, 43.5 MB/s  # 這個指令的簡單意義如下：  # if是input file ，輸入檔案。那個/dev/zero是會一直輸出0的裝置！  # of是output file ，將一堆零寫入到後面接的檔案中。  # bs是每個block大小，就像檔案系統那樣的block意義；  # count則是總共幾個bs的意思。  [root@www ~]# ll -h /home/loopdev  -rw-r--r-- 1 root root 512M Oct 28 02:29 /home/loopdev |

dd像在疊磚塊，將512塊每塊1MB的磚塊堆疊成一大檔案(/home/loopdev)！最終會出現一個512MB的檔案！

* 格式化: 很簡單就建立起一個512MB的檔案了吶！接下來當然是格式化囉！

|  |
| --- |
| [root@www ~]# mkfs -t ext3 /home/loopdev  mke2fs 1.39 (29-May-2006)  /home/loopdev is not a block special device.  Proceed anyway? (y,n) y <==由於不是正常的裝置，所以這裡會提示你！  Filesystem label=  OS type: Linux  Block size=1024 (log=0)  Fragment size=1024 (log=0)  131072 inodes, 524288 blocks  26214 blocks (5.00%) reserved for the super user  .....以下省略..... |

* 掛載: 那要如何掛載啊？利用 mount 的特殊參數: -o loop參數來處理！

|  |
| --- |
| [root@www ~]# mount -o loop /home/loopdev /media/cdrom/  [root@www ~]# df  Filesystem 1K-blocks Used Available Use% Mounted on  /home/loopdev 507748 18768 462766 4% /media/cdrom |

這方法可在原本分割槽不更動下製作出想要的分割槽！尤其是想要玩Linux上的『虛擬機器』，也就是以一部Linux主機再切割成為數個獨立的主機系統時，類似VMware這類的軟體！