File System Checking (fsck)

The built in UNIX feature of delaying the updating of the disk superblock and the inode block by their memory copies leaves scope for file system inconsistency. If the power goes of before the superblock written to disk, the file system loses its integrity. There are many discrepancies that could lead to file system corruption, and the most common ones are listed below:

- Two or more inodes claiming the same disk block
- A block marked as free but not listed in the superblock
- A used block marked as free
- An inode neither marked free nor in use, or having a bad block number that is out of range.
- Mismatch between the file size specified in the inode and the number of data blocks specified in the address array.
- A corrupt superblock containing erroneous summary data.
- A file not having at least one directory entry or having an invalid file type specified in the inode.

The fsck command is used to check and repair a damaged file system. It is generally run when a file system fail to mount. File systems are marked as “dirty” or “clean”. Fsck then cheks only the dirty file systems during the net startup. The command can also be used with the name of the file system as argument.

```
# fsck  /dev/rdsk/c0t3d0s5

**  /dev/rdsk/c0t3d0s5

** Phase 1  -  Check Blocks and Sizes

** Phase 2 – Check Pathnames

** Phase 3 – Check Connectivity

** Phase 4 – Check Reference Counts
```

fsck conducts a check in five phases, and the output above is obtained when the file system is consistent. However, when it is corrupt, messages and questions are seen on the system console, which you have to answer correctly. The five phases are:

- Phase 1 Validates the inode for correctness of format and the block numbers for bad and duplicate blocks. fsck declares a block BAD if the block number is out of range and DUP if it is claimed by another inode.
- Phase 2 Check all directory entries, starting from root, for OUT OF RANGE inode numbers detected in Phase 1. fsck corrects the errors either removing the entire directory or the file.

- Phase 3 Looks for unreferenced directories and stores their files in /lost+found for later examination. The files here are named after their inode numbers. You must make sure this directory is always available on every file system because fsck would not create it on its own.
- Phase 4 Check the link count as stored in the inode with the directory entries, and prompts for the file's removal or reconnection (to the /lost+found directory), depending on the extent of damaged caused. fsck then compares the free inode count it computes with the figure stored in the super block.
- Phase 5 Finally, fsck's free-block count is compared with the figure maintained in the superblock. A salvage operation may be carried out with the user's approbal, which will replace the erroneous free block list with a newly computed one.
- When used without options, fsck prompts you before repairing any damage that it has detected. It is generally safe to answer every question with a **y**. **fsck –y** assumes that all answers are in the affirmative, and proceeds without waiting for a response.

Communicating with Users (wall)

The wall command addresses all users simultaneously. Most unix system don't permit users to run this command, and reserve it for sole use of the administrator.

```
$ wall

The machine will be shut down today

At 13:30 hrs. The backup will be at 13:00 hrs

[Ctrl-d]
```

All users currently logged in will receive this message on their terminal. This command is routinely executed by the administrator –especially before shutdown of the system.

Changing the time stamps (touch)

The touch command is used to set the modification and access times to predefined values.

Syntax

touch options expression filename(s)

When touch used without options or expression, both times are set to current time. The file is created if it does not exist.

```
$ touch student.lst                create if file does not exist, set both time to current time
```

When touch is used without options but with expression, it changes both times. The expression consists of and eight digit number using the format MMDDhhmm (month, day, hour and minute). Optinally you can suffix a two- or four-digit year string:

```
$ touch 05151620 student.lst

$ ls –l student.lst

-rw-r—r--  1  students  students  900  May  15  16:20  student.lst
```

Change the modification time

```
$ touch  -m  01300830  student.lst

$ ls –l student.lst

-rw-r—r--  1  students  students  900  Jan  30  08:30  student.lst
```

Change the access time

```
$ touch  -a  02250930  student.lst

$ ls –l student.lst

-rw-r—r--  1  students  students  900  Feb  25  09:30  student.lst
```

The system administrator often uses the touch to “touch up” these times so a file may be included in or excluded from an incremental backup (that backs up only changed files).

Managing Disk Space

The administrator must regularly scan the disk and locate files that have no longer required. He needs the df and du commands for this task.

Reporting free space (df)

Your operating system is supported by multiple file systems. The df (disk free) command reports the amount of free space available for each file system separately.

```
#df

/                  (/dev/dsk/c0t0d0s0) :    3491876  blocks          483932 files
```

```
/usr        (/dev/dsk/c0t0d0s4) :   2434820  blocks        458466 files
/proc       (/proc              ) :         0  blocks             0 files
/dev/fd     (fd                 ) :         0  blocks             0 files
```

There are several file systems in machine. The first column shows the directory where the file system is attached. The second column shows the device name of the file system. The last two columns show the number of 512-byte blocks available and the number of files that you can create.

The first line refers to the root file system (/), which has 3,491,876 blocks of disk space free. It also has 483,932 inode free, which means upto that many additional files can be created on this file system. The system will continue to function till there are free blocks or inodes.

-t (total) option

The –t (total) option includes the above output, as well as total amount of disk space in the file system.

-k option

The –k option is used to display file system information and usage in 1024 byte (KB) blocks.

```
#df  -k  /   /usr
Filesystem              Kbytes     used     available    capacity        Mounted on
(/dev/dsk/c0t0d0s0) :   1986439   240501   1686345      13%             /
(/dev/dsk/c0t0d0s4) :   2025076   807666   1156658      42%             /usr
```

Disk Usage (du)

du command display usage by recursive examination of the directory tree.

Report the usage of /home/sales/tml

```
$ du  /home/sales/tml
11554   /home/sales/tml/forms
12820   /home/sales/tml/data
.
.
```

25170 /home/sales/tml	Also Report a summary at the end

By default du lists the usage of each subdirectory of its argument , and finally produces a summary.

-s option

-s option is used to report only summary.

du -s /home/sales/tml

25170 /home/sales/tml

Accessing Spaces consumed by users :

Most of the dynamic space in a system is consumed by users' home directories and data files. You should use du –s to report on each user's home directory.

# du -s /home/*	
144208	/home/amit
98290	/home/sumit
13834	/home/students

-a option

-a option of du can also report on each file in a directory. But the list would be too big to be of any use.

grep family

The grep family includes egrep and fgrep.

egrep

egrep stands for extended grep. It extends grep's pattern-matchine capabilities in two ways

- It admits alternates
- It enables regular expressions to be bracketed / grouped using the pair of parentheses (i.e (…)), also known as factoring.

It offers all the options and regular expression meta characters of grep, but its most useful feature is the facility to specify more than one pattern for search. While grep uses some characters that are not recognized by egrep, egrep includes some additional extended metacharacters not used by either grep or sed utilities, given in following table:

Table : egrep's extended metacharacters

Expression	Meaning
ch+	It matches one or more occurrences of character ch.
ch?	It matches zero or one occurances of character ch.
exp1 | exp2	It matches expression exp1 or exp2
(x1 | x2) x3	It matches expression x1x3 or x2x3.

Example:

To display record in which m occurs one or more time.

```
$ egrep m+  f2

Unix and Shell Programming

Program and process

$
```

Example:

To locate multiple pattern in one command Unix and Linux

```
$ grep  "Unix|Linux"  f1

Unix is operation system

Red hat Linux
```

Example:

To locate lines contain hardware or software

```
$ grep  "(soft|hard)ware"  f1
```

-f option

egrep provides a facility to take pattern from a file.

```
$ cat patfile

Unix|Linux

$
```

Now use patfile to search in a file.

```
$ egrep  -f  patfile  f1

Unix is operation system

Red hat Linux
```

fgrep

fgrep stands for fixed/fast grep. The fgrep utility can normally search for fixed strings. i.e., character string without embedded metacharacters. Frep accepts multiple patterns, both from the command line and a file, but unlike grep and egrep, does not accept regular expressions. To search a simple string fgrep is recommended.

Example:

Multiple pattern from command line

```
$ fgrep “Unix

> Linux’  f1

Unix is operation system

Red hat Linux
```

Example:

Taking multiple pattern from file

```
$ cat patfile

Unix

Linux

[Ctrl-d]

$

Now run patfile on f1 file using –f option

$ fgrep  -f  patfile f1
```