Unix'te bir dosya disk üzerindeki baytların toplamı olarak isimlendirilir.Bu tanımı iki parça şeklinde düşünebiliriz:

- Disk üzerindeki kayıtların toplamı
- Bir adla

Her dosya disk üzerinde belirli bir konumda saklanır.Örneğin; myfile.cshrc my sun Workstation içinde diskte "sd0h" bölümünde tutulur.İşletim sistemlerinde disk üzerindeki her bir dosya veri yapıları ile ilişkilidir.Buna "dosya indisi" denir.Bu dosya indisi dosyanın disk üzerindeki yeri,büyüklüğü,benzersiz bir dosya indis numarası ,koruma modu,dosya sahibi vb. bilgileri içerir.Eğer ilgileniyorsanız,işte tipik dosya indis veri yapısı.Ancak bu veri yapılarının detayları bizim alanımızın ötesinde.

```
struct inode {
        LIST_ENTRY(inode) i_hash;  /*Hash zinciri*/
       struct  vnode  *i_vnode;     /* Vnode bu dosya indisi ile
ilişkilendirilmiş. */
        struct  vnode  *i_devvp;     /* Vnode I/O blokları için */
        u_int32_t i_flag;            /* bayraklar,aşağı bakınız*/
        dev_t     i_dev;             /* Cihaz dosya indisi ile
ilişkilendirilmiş*/
        ino_t     i_number;          /* dosya indis kimliği*/

        union {                      /* dosya sistemiyle
ilişkilendirilmiş.*/
                struct  fs *fs;      /* FFS */
        } inode_u;

        struct   klist i_knotes;     /* Knotes Vnode'da bağlandı */
        struct   dquot *i_dquot[MAXQUOTAS]; /* Dquot yapıları*/
        u_quad_t i_modrev;           /* NFS kiralama için revizyon seviyesi
*/
        struct   lockf *i_lockf;     /* Bayt düzeyinde kilit liste başı*/
        struct   lock__bsd__ i_lock; /* Dosya indis kilidi*/
        /*
         * Yan etkileri;dizin arama sırasında kullanılır
         */
        int32_t   i_count;           /* Dizinde boş slot boyutu. */
        doff_t    i_endoff;          /* Dizindeki yararlı şeyler sonu. */
        doff_t    i_diroff;          /* Offset in dir,son girdimizin
bulunduğu yer*/
        doff_t    i_offset;          /* Offset dizininde boş olan alan*/
        ino_t     i_ino;             /* Dosya indis numarasının bulunduğu
dizin. */
        u_int32_t i_reclen;          /* Dizin girdisinin boyutunun
bulunması*/
        /*
         * The on-disk dinode itself.
         */
        struct  dinode i_din;        /*Diskteki dinode 128 bayt */
};

struct dinode {
        u_int16_t       di_mode;        /*   0:IFMT,izin;aşağıya bakınız */
        int16_t         di_nlink;       /*   2: Dosya bağlantı sayası*/
        union {
```

```
                u_int16_t oldids[2];    /*   4: Ffs:Eski kullanıcı ve grup
'id'leri. */
                int32_t   inumber;      /*   4: Lfs: Dosya dizin numarası*/
        } di u;
        u_int64_t       di_size;        /*   8: Dosya bayt sayısı */
        int32_t         di_atime;       /*  16: Son erişim zamanı. */
        int32_t         di_atimensec;   /*  20: Son erişim zamanı. */
        int32_t         di_mtime;       /*  24:Son modifiye zamanı. */
        int32_t         di_mtimensec;   /*  28: Son modifiye zamanı. */
        int32_t         di_ctime;       /*  32: Son dosya indisi değişiklik
zamanı. */
        int32_t         di_ctimensec;   /*  36: Son dosya indisi değişiklik
zamanı. */
        ufs_daddr_t     di_db[NDADDR];  /*  40: Direkt disk bloğu. */
        ufs_daddr_t     di_ib[NIADDR];  /*  88: Dolaylı disk bloğu. */
        u_int32_t       di_flags;       /* 100: Durum bayrakları(chflags).
*/
        u_int32_t       di_blocks;      /* 104: Gerçekte düzenlenen
bloklar. */
        int32_t         di_gen;         /* 108: Üretim numarası*/
        u_int32_t       di_uid;         /* 112: Dosya sahibi*/
        u_int32_t       di_gid;         /* 116: Dosya grubu*/
        int32_t         di_spare[2];    /* 120: Ayrılmış;şu anda
kullanılmayan */
};
```

Bu yolla bir dosya indisine "link" bağlayarak dosya isimlendiriyoruz.
Örneğin şöyle dediğimizde ;

```
UNIX> cat > f1
This is f1
^D
UNIX>
Bu disk üzerinde içeriği bayt olan bir dosya oluşturur.
"This is f1\n"
Bir dosyanın dosya indis numarasını görmek için '-i' ls bayrağını
kullanabilirsiniz.
UNIX> ls -i f1
34778 f1
UNIX>
```

Bir dosyaya birden fazla bağlantı noktası olabilir.Yukarıda f1 dosyasını yaptığımızı farzedelim ve şimdi aşağıdakileri yapalım.

```
UNIX> ln f1 f2
```

Bu f1'den yeni bir link oluşturmak ve bunu f2 olarak isimlendirmektir…Ve şimdi bir dosyamız için 2 pointerımız oldu..Bunu listelediğimizde :

```
UNIX> ls -li f1 f2
34778  -rw-r--r--  2 plank          11 Sep 16 10:12 f1
34778  -rw-r--r--  2 plank          11 Sep 16 10:12 f2
UNIX> cat f1
This is f1
UNIX> cat f2
This is f1
UNIX>
```

Dosyaların bağlantılarının farklı olması yanı sıra, aynı dosyalar olduğunu görüyoruz. Eğer bu dosyalardan birinde değişiklik yaparsak , örneğin vi editor yardımı ile f2 deki “This” sözcüğü yerine “That” yazalım.. Aynı değişikliklerin f2’nin yanı sıra f1 dosyasındada olmuş olduğunu göreceğiz…

```
UNIX> vi f2
...
UNIX> cat f2
That is f1
UNIX> cat f1
That is f1
UNIX> ls -li f1 f2
34778  -rw-r--r--  2 plank          11 Sep 16 10:14 f1
34778  -rw-r--r--  2 plank          11 Sep 16 10:14 f2
UNIX>
```

F2’nin değiştirme zamanını değiştirsek, f1 için olan değiştirme zamanıda değişir. Dosya değiştirme zamanı,dosya düğümünün bir parçasıdır..Böylece f2 değiştirildiğinde,aynı değişiklikler f1 de de görülür... Aynı şeyler,Dosya değiştirildiğindede görülür. Eğer f1 için korumayı değiştirirsek, göreceğiz ki aynı değişiklikler f2 yi de etkilemiş…

```
UNIX> chmod 0400 f1
UNIX> ls -li f1 f2
34778  -r--------  2 plank          11 Sep 16 10:14 f1
34778  -r--------  2 plank          11 Sep 16 10:14 f2
UNIX>
```

Ls komutunun 3. sutununa dikkat edin.. bu dosya linklerinin numaralarıdır..Eğer f1 e farklı bir link ile bağlarsak, bu sutün güncellenecektir..

```
UNIX> ln f1 f3
UNIX> ls -li f1 f2 f3
34778  -r--------  3 plank          11 Sep 16 10:14 f1
34778  -r--------  3 plank          11 Sep 16 10:14 f2
34778  -r--------  3 plank          11 Sep 16 10:14 f3
```

Rm komutunu kullandığımızda,aslında linkleri silmiş oluruz..
Örneğin:

```
UNIX> chmod 0644 f1
UNIX> rm f1
UNIX> ls -li f*
34778  -rw-r--r--  2 plank          11 Sep 16 10:14 f2
34778  -rw-r--r--  2 plank          11 Sep 16 10:14 f3
UNIX>
```

Dosyanın son linki silindikten sonra dosya kendisini düğümleri ve sonra her şeyisiler. Dosya link işaret edicisi olduğu sürece dosya kalır.. Hadi dosya linkinin üzerine yazdığımızda ne olduğuna bakalım..Örneğin şu adımları takip edelim:

```
UNIX> cat > f2
This is now file f2
^D
UNIX> cat f2
This is now file f2
UNIX> cat f3
```

Kabuk performansı çıkışa yönlendirildiğinde f2 nin anlamı budur, dosyaları silip yeniden oluşturmak yerine,onları açar ve onları değiştirir..
Bu yaptığımızın yerine,aşağıdakini düşünürsek:

```
UNIX> gcc -o f2 ls1.c
UNIX> ls -li f*
34779  -rwxr-xr-x  1 plank       24576 Sep 16 10:16 f2
34778  -rw-r--r--  1 plank          20 Sep 16 10:16 f3
UNIX>
```

C de derlerken f2 çalıştırılabilir dosyasını oluşturmadan önce “rm f2 komutunu çalıştırmamaya dikkat etmeliyiz…

Not:tüm dizinlerin en az 2 bağlantısı vardır:

```
UNIX> mkdir test
UNIX> ls -li | grep test
34800  drwxr-xr-x  2 plank         512 Sep 16 10:17 test
UNIX>
```

Bunun nedeni her iki dizinin alt dizinleri içermesidir"." ve ".." Önce kendisi için bir link ve ikinci dizin için bir link olusturur.Böylece, dizin dosyası için iki bağlantı vardır."test": "test" ve "test/."
Benzer şekilde, Deney bir alt dizin olduğunu varsayalım:

```
UNIX> mkdir test/sub
UNIX> ls -li | grep test
34800  drwxr-xr-x  3 plank         512 Sep 16 10:17 test
UNIX>
```

Şimdi "test etmek" için üç bağlantı olsun:test", "test/.", ve "test/sub/.."
Sizin için otomatik olarak oluşturulan bu bağlantıların yanı sıra, Manuel olarak dizinlere bağlantı oluturamazsınız. Bunun yerine, bir "soft link" adlı özel bir bağlantı vardır, içine sizin "ln-s" komutu kullanmanıza imkan verir.
Örneğin, Aşağıdaki gibi test dizininde soft bir link oluşturabilirsiniz:

```
UNIX> ln -s test test-soft
UNIX> ls -li | grep test
34800  drwxr-xr-x  3 plank         512 Sep 16 10:17 test
34801  lrwxrwxrwx  1 plank           4 Sep 16 10:18 test-soft -> test
```

Soft linklerin dizin listesinde farklı bir tür olduğunu dikkat edin.
Ayrıca, "test" soft bir linkin oluşturuldugunda testin düğümünün bağlantı alanını güncelleştirmek

olmadığına dikkat edilmesi gerekir.Bu sadece kayıtları düzenli, veya "hard" linkleri.
Soft bir link dosyanın düğüm değiştirmeden bir dosyaya işaret etmesinin bir yoludur.
Ancak, Soft linklerle hard linklerin yapabileceğinden çok şey yapabilirsiniz:

```
UNIX> cat > f1
This is f1
UNIX> ln -s f1 f2
UNIX> cat f2
This is f1
UNIX> cat > f2
This is f2
UNIX> cat f1
This is f2
UNIX> ls -l f*
-rw-r--r--  1 plank          11 Sep 16 10:19 f1
lrwxrwxrwx  1 plank           2 Sep 16 10:18 f2 -> f1
UNIX> chmod 0600 f2
UNIX> ls -l f*
-rw-------  1 plank          11 Sep 16 10:19 f1
lrwxrwxrwx  1 plank           2 Sep 16 10:18 f2 -> f1
UNIX>
```

Sonra hard ve soft Linkler arasındaki temel fark nedir?
Bir uygulamanın ,bütün hard dosya linkleri, soft Linklerin hepsini silmesenizde,daha sonra dosya yine silinir.

```
UNIX> rm f1
UNIX> ls -l f*
lrwxrwxrwx  1 plank           2 Sep 16 10:18 f2 -> f1
UNIX> cat f2
cat: f2: No such file or directory
UNIX>
```

Link "kararsız(unresolved)" olarak çağırılır.

Unixde, başka bir dosya sistemindeki bir dizin, bir dosya sistemindeki bir dosyayı hard linkle bağlayamaz. Yani, öğrenci hesaplarından, sizin için örnek bir komut yapamazsınız:

UNIX> ln /blugreen/homes/plank/cs360/notes/Links/lecture.html ~/lecture.html

Çünkü evinizin dizini başkasıyla aynı dosya sistemi üzerinde değil.Ancak,soft bir bağlantı ile yapabilirsiniz:

UNIX> ln -s /blugreen/homes/plank/cs360/notes/Links/lecture.html ~/lecture.html