**Metas del Sistema de archivos**

* Brindar espacio en disco a los archivos del usuario y del sistema.
* Mantener un registro del espacio libre. Cantidad de espacio libre y donde se ubica dentro del disco.

**Conceptos**

* Sector
  + Unidad de almacenamiento utilizada en los discos rígidos.
* Bloque o Cluster
  + Conjuntos de sectores consecutivos.
  + Nos sirve para archivos que necesitan más de un sector.
* File system
  + Define la forma en que los datos se almacenan.
* FAT: File Allocation Table
  + Tabla que contiene información sobre en que lugar están alocados los distintos archivos.
* Se puede provocar fragmentación interna en un disco.

**Pre-asignación: forma de asignar espacio**

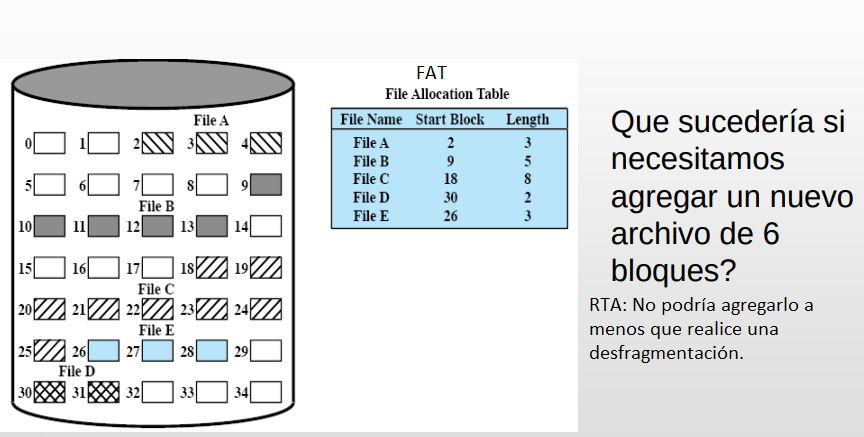
* Es necesario saber cuanto espacio va a ocupar el archivo en el momento de su creación.
* Se tiende a definir espacios mucho más grandes que lo necesario.
* Es posible usar sectores contiguos para almacenar los datos de un archivo.
* Y si el archivo supera el espacio que le asignaron?
* Asignación continua aplica la pre-asignación.

**Asignación dinámica: forma de asignar espacio**

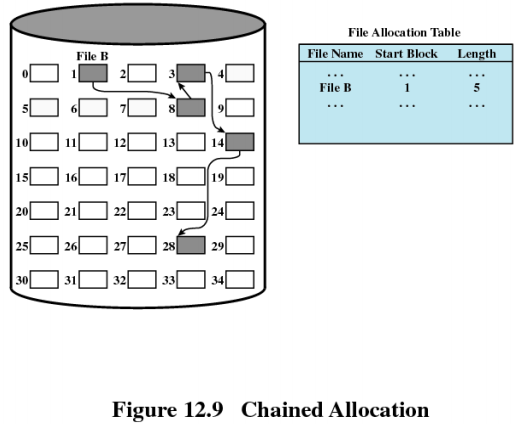
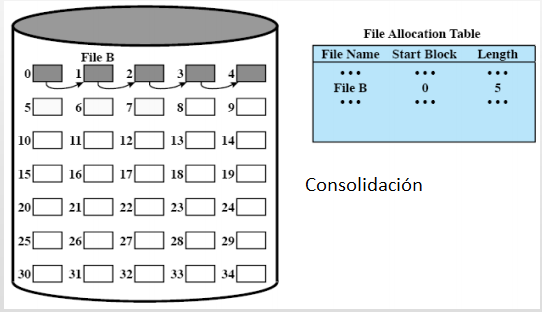
* El espacio es solicitado a medida que se necesita.
* Los bloques de datos pueden quedar de forma no contigua.
* La tendencia de hoy día es para asignaciones dinámicas.

**Veremos formas de asignación**

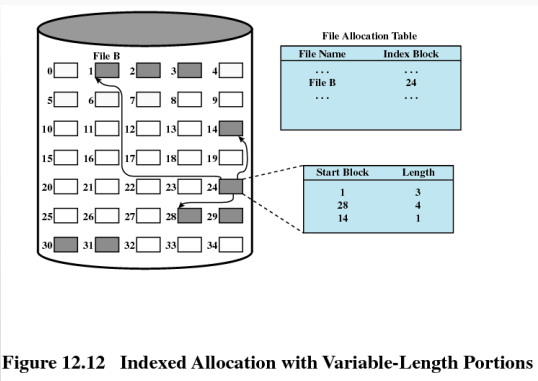
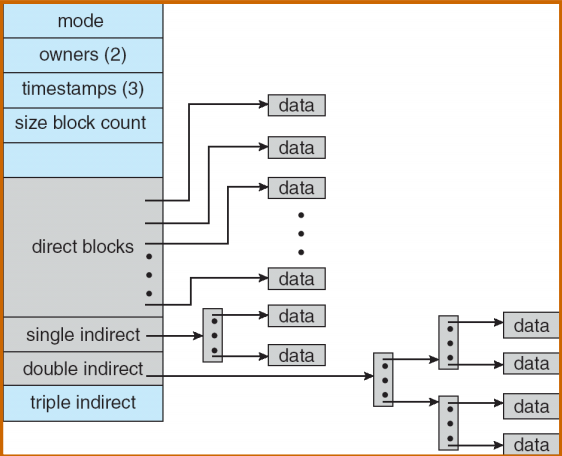
**Forma continua**

* Para guardar un archivo se utiliza un conjunto contiguo de bloques.
* Es necesario preasignar, ósea conocer el tamaño del archivo al ser creado.
* El formato de la FAT es simple, ya que utiliza una sola entrada que incluye el bloque en el cuál empieza el archivo y la longitud (bloques usados).
* El archivo puede ser leído con una única operación.
* Puede existir fragmentación externa que se puede resolver compactando aunque esto es MUY costoso.
* La técnica presenta problemas:
  + Encontrar bloques libres continuos en el disco (y que tengan el tamaño necesario).
  + El incremento del tamaño de un archivo.
* 

**Forma encadenada**

* Se asigna en base a bloques individuales.
* Cada bloque tiene un puntero a la ubicación del próximo bloque del archivo 🡪 se debe implementar una estructura de enlaces.
* FAT tiene una única entrada, bloque inicial y tamaño del archivo.
* No hay fragmentación externa al usar esta forma.
* Útil para acceso secuencial.
* Los archivos pueden crecer bajo demanda (asignación dinámica).
* No se requieren bloques contiguos.
* 
* Es posible mover los bloques de un mismo archivo de forma que queden contiguamente, esto ayuda a que haya una cercanía de bloques para agilizar operaciones a la hora de leer el disco (desfragmentadores de discos).
  + 

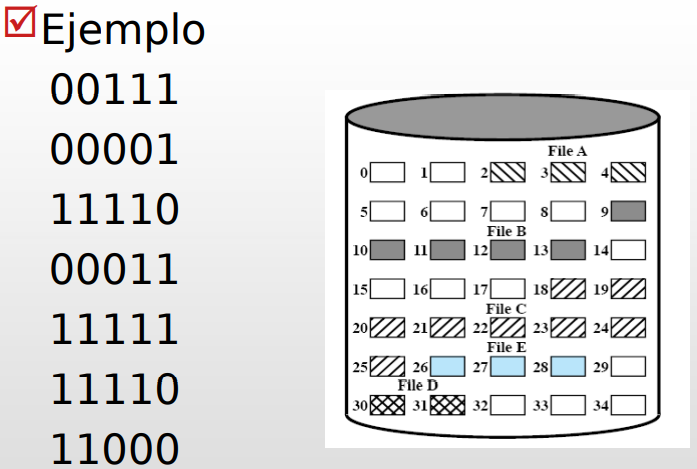
**Forma indexada**:

* Se asigna en base a bloques individuales.
* No se produce fragmentación externa.
* El acceso random a un archivo se vuelve eficiente.
* FAT
  + Una sola entrada con la dirección del bloque de índices (puntero).
  + El bloque índice NO contiene datos del archivo, sino que contiene una lista de punteros a los bloques que componen el archivo. (índice)
* Tiene dos variantes
* **ASIGNACIÓN POR SECCIONES**
  + A cada entrada del bloque índice se agrega el campo longitud.
  + Cada índice del bloque índice apunta al primer bloque de un conjunto que se almacena sí o sí de manera contigua.
  + 
* **NIVELES DE INDIRECCIÓN**
  + Existen bloques directos con datos guardados dentro.
  + Otros bloques son considerados como bloque índice (apuntan a varios bloques de datos).
    - Direccionamiento indirecto simple: un bloque índice solo que apunta a los bloques de datos.
    - Direccionamiento indirecto doble: un bloque índice que apunta a otro bloque índice y este último apunta a los bloques de datos.
    - Pueden haber varios niveles de indirección
  + 
  + Conocido como INODO

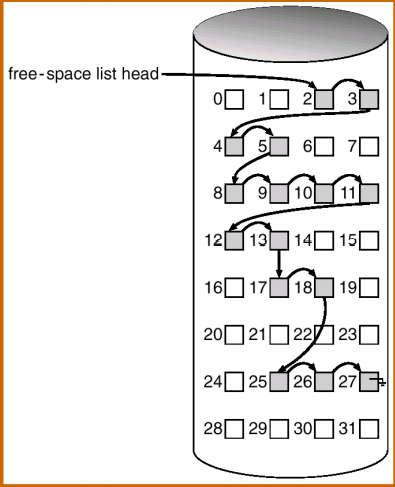
**Gestión de espacio libre**

* Consiste en el control sobre cuáles de los bloques de disco están disponibles.
* Tres alternativas:
  + Tablas de bits.
  + Bloques libres encadenados.
  + Indexación.

**Tablas de bits**

* Vector con 1 bit por CADA bloque de disco.
  + 0 = bloque libre, 1 = bloque ocupado.
* Ventaja
  + Es fácil halalr un bloque o grupo de bloques que estén libres.
* Desventaja
  + Tamaño que llega a ocupar el vector en la memoria.
    - Se calcula: tamaño del disco bytes / tamaño bloque en sistema de archivos.
    - Ej: Disco de 16 GB con bloques de 512 bytes cada uno 🡪 el vector pesa 32 MB.
* 

**Bloques encadenados**

* Se tiene un puntero al primer bloque que esté libre.
* Cada bloque libre tiene un puntero al siguiente bloque libre.
* Resulta ser ineficiente para la búsqueda de bloques libres ya que hay que realizar varias operaciones de E/S para obtener un grupo libre.
* Si se pierde un enlace se generan problemas.
* Es difícil hallar bloques libres consecutivos.
* 

**Indexación (agrupamiento)**

* Variante de bloques libres encadenados.
* El primer bloque libre contiene las direcciones de N bloques libres.
* Las N-1 primeras direcciones son bloques libres.
* La N-ésima dirección referencia otro bloque con N direcciones de bloques libres.