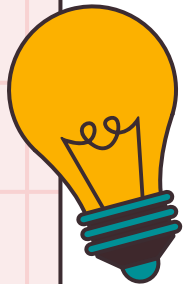
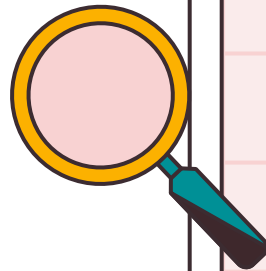




1º DAW

U.T. 1

Ficheros: tipos, propiedades y organización. Sistemas de Ficheros



Pablo Berciano Posada

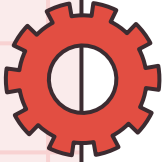


Tabla de contenidos

01

Ficheros

02

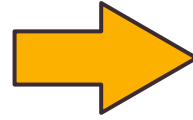
**Propiedades de los
ficheros**

03

**Organización lógica
de ficheros**

04

**Sistemas de
ficheros**



01

Ficheros





¿Qué es un fichero?



Un **fichero** o **archivo** es una **colección ordenada de datos** que tienen entre sí una **relación** y que se almacenan de forma **permanente** en un dispositivo de memoria no volátil. Suelen tener un **nombre** y una **extensión** que determina el formato de la información que contiene.

Con **permanente** queremos decir que, salvo fallos catastróficos o hasta que sean borrados de forma voluntaria, estos datos **permanecen** en el medio en que se almacenan y continúan existiendo después de que el programa que los creó deja de ejecutarse, incluso después de apagar el ordenador.

Esto marca la diferencia con los datos que son **provisionalmente** almacenados en la memoria RAM, la **memoria volátil** del ordenador, que no sobreviven al programa que los crea y mucho menos a la desconexión del ordenador.



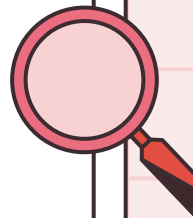
Tipos de ficheros

Tradicionalmente, los ficheros se han **clasificado** de muchas formas:

- Según su **contenido**: texto, binario.
- Según su **organización**: secuencial, directa, indexada. La organización de un fichero dicta la forma en que se han de acceder a los datos.
- Según su **utilidad**: maestro, movimiento, histórico. La utilidad de un fichero indica qué uso se va a hacer de él.

Hoy en día estas dos últimas clasificaciones han quedado en **desuso**.

Actualmente un sistema operativo trata un fichero desde dos puntos de vista:

- Según su **contenido**: texto o binario.
 - Según su **tipo**: imágenes, ejecutables, videos, etc.
- 

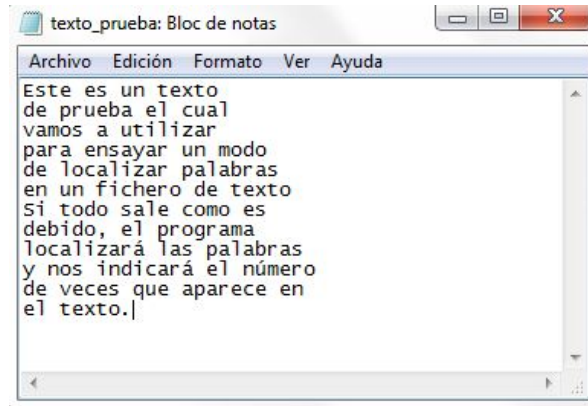


Ficheros de texto



Los **ficheros de texto** se denominan planos ya que no tienen ningún tipo de formato. Únicamente contienen **texto**, **saltos de línea** y la **marca de fin de fichero**. Suelen estar escritos en formato **ASCII**, cada carácter se representa mediante un valor numérico. Estos ficheros son generados mediante un **editor de texto**, NO un "procesador" de textos como Word. Los ficheros de texto pueden tener una **extensión** que identifica su contenido, por ejemplo:

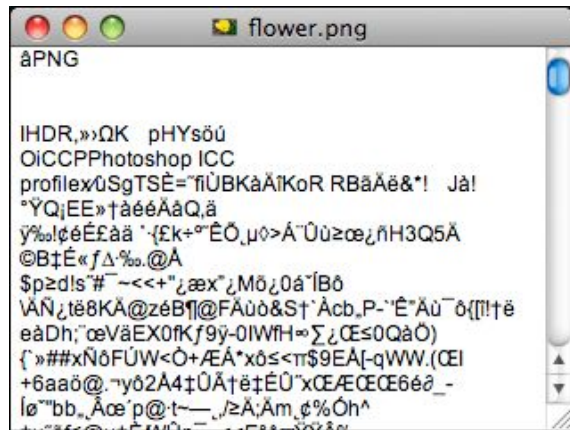
- Ficheros de **configuración**: .ini, .inf, .conf, etc.
- Ficheros de **código fuente**: .sql, .c, .java, etc.
- Ficheros de **páginas web**: .html, .php, .xml, .css, etc.

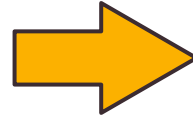


Ficheros binarios

Los **ficheros binarios** son los que contienen información de cualquier tipo **codificada en binario** cuyo fin es el almacenamiento y procesamiento en ordenadores. Por ejemplo, los archivos informáticos que almacenan texto formateado o fotografías, así como los archivos ejecutables que contienen programas, vídeos. Así tenemos **extensiones** como:

- **Procesadores de texto:** .doc, .odt, etc.
- **Imagen:** .bmp, .gif, .jpg, .png, etc.
- **Ejecutables:** .exe, etc.
- **Vídeo:** .avi, .mov, .mpg, etc.
- **Comprimidos:** .tar, .zip, etc.





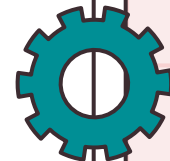
02

Propiedades de los ficheros





Campos



Un **campo** se define como la **menor unidad de información con significado lógico** en un fichero. Hay distintos **tipos de campos**:

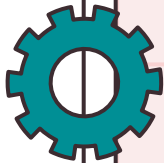
- **Campos de tamaño fijo:** siempre ocupan lo mismo, por lo que para leerlos es necesario solo un pequeño cálculo.
- **Campos de tamaño variable:** utilizan algún indicador de la medida del campo como:
 - Indicador de longitud de campo al inicio
 - Delimitador al final

Un **registro** está construido como un **conjunto de campos que permanecen juntos** cuando se observa el fichero con un mayor nivel de abstracción.



Parámetros de utilización

Los **parámetros de utilización** marcarán la **organización interna del fichero**, y se tendrán en cuenta para optimizar el tipo de operaciones a las que se van a someter. Son los siguientes:

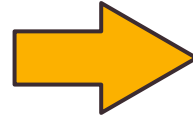
- **Volumen:** Es el número de caracteres o bytes que ocupa el archivo, o el límite que puede llegar a ocupar.
 - **Crecimiento:** Especifica el número de altas y bajas en cada tratamiento del fichero.
 - **Actividad:** Se refiere al número de procesos de consulta y modificación que sufrirá el fichero a lo largo de su vida. Se usan dos parámetros: la **tasa** de consulta o modificación y la **frecuencia** de consulta o modificación.
 - **Volatilidad:** Se refiere al número de inserciones y borrados en el tratamiento del fichero. Si se estima alta volatilidad se debe elegir una organización flexible. Hay dos parámetros para medir la volatilidad: la **tasa** de renovación y la **frecuencia** de renovación.
- 



Operaciones con ficheros

Los sistemas informáticos incluyen una serie de programas de utilidad para efectuar **operaciones básicas con ficheros**. Algunas operaciones son:

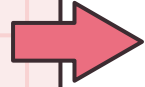
- Creación
 - Apertura
 - Cierre
 - Borrado del fichero
 - Ordenación
 - Fusión
 - Intersección
 - Partición
 - Compactación
 - Copia
 - Consulta
 - Actualización (Inserción, Modificación, Borrado lógico y Borrado real)
- 



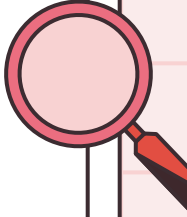
03

Organización lógica de ficheros





Organización lógica de ficheros



**Organización
secuencial**



**Organización
secuencial
encadenada**



**Organización de
acceso directo o
aleatorio**



**Organización
secuencial
indexada**

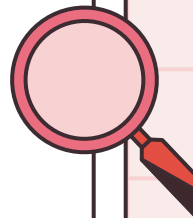


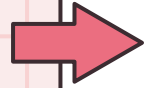
Organización secuencial

En este caso la información del fichero se escribe en **posiciones físicamente contiguas**. Para acceder a un dato es necesario **recorrer todos** los anteriores. Las **operaciones** que se pueden realizar son:

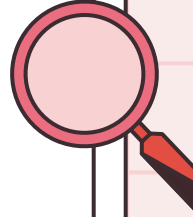
- **Añadir:** sólo al final del fichero.
- **Consulta:** en orden secuencial.
- **Actualización:** inserción, modificación y eliminación, se llevan a cabo en un **fichero de movimientos independiente**, donde se almacenan los cambios, y después se escribe la información en un fichero nuevo y se borra el antiguo.

Esta organización **aprovecha al máximo el soporte** y **tiene acceso secuencial muy rápido**. Sus **problemas** son la **búsqueda de registros concretos**, además de no ser útil con ficheros que se **modificarán** mucho.





Organización secuencial encadenada



Son ficheros cuyos **registros no son contiguos**, almacenan tanto su **dirección y contenido**, como la **dirección del siguiente** registro. Las **operaciones** que se pueden realizar en ellos son:

- **Añadir:** se busca el último registro, se escribe el nuevo registro en un espacio libre de memoria y se guarda la dirección en el puntero del antiguo último registro.
- **Consulta:** la consulta es secuencial, saltando de registro en registro.
- **Inserción:** hay que actualizar el puntero del registro anterior y direccionar el puntero del nuevo al registro al que apuntaba el registro anterior.
- **Modificación:** si no se modifica longitud del registro, se busca y se cambia el valor. Si no, es necesario insertar un nuevo registro con más espacio y borrar el viejo.
- **Borrado:** Para eliminar un registro, se destruye su dirección del puntero anterior, y se hace que este apunte al sucesor del registro que se quiere eliminar.

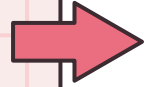


Organización directa o aleatoria

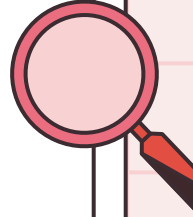


No tiene por qué existir ninguna relación lógica entre los registros y su dirección física. Su dirección se decide mediante una fórmula matemática. Algunos de los métodos para elegir dirección son:

- **Direccionamiento directo:** asigna a cada clave una dirección lógica. Debe haber tantas direcciones como registros, y las llaves deben ser numéricas. No produce sinónimos.
- **Direccionamiento asociado:** cuando son claves alfanuméricas, se construye una tabla y se asocia a cada clave una dirección. Requiere de espacio para la tabla.
- **Direccionamiento calculado (hashing):** se aplica una serie de cálculos matemáticos con la clave, puede dar sinónimos. Algunos ejemplos son **la división por un primo, el cuadrado de la clave, el truncamiento o el plegamiento.**



Organización directa o aleatoria



Las operaciones que se pueden realizar son:

- **Consulta:** Se realiza por clave, de manera directa. Si no está el registro en su dirección, se resuelven los posibles sinónimos.
- **Inserción:** se calcula la nueva dirección del nuevo registro y si ya está ocupada se resuelve el sinónimo.
- **Modificación:** Se localiza el registro y se modifica.
- **Borrado:** Siempre es un borrado lógico.

Tiene una **gran flexibilidad y rapidez de consulta**, pero como inconvenientes destacan el **mal aprovechamiento del espacio y la necesidad de usar dispositivos direccionables**.

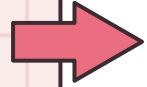


Organización secuencial indexada

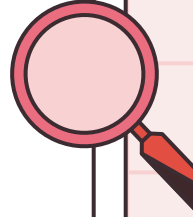
Tiene dos estructuras:

- **Zona de registros:** contiene todos los registros del fichero, ordenados por clave.
- **Zona de índices:** Se procesa de forma secuencial, cada registro tiene dos campos, clave y dirección.

La zona de registros está **dividida en tramos lógicos**. Cada tramo tiene una serie de registros consecutivos y para cada tramo hay un registro en la zona de índices, que **guarda el valor mayor de cada tramo, y la primera dirección del tramo**.



Organización secuencial indexada



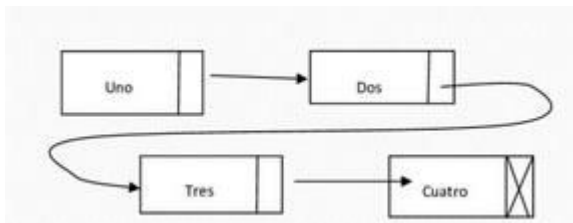
Las **operaciones** que se pueden realizar son:

- **Consulta:** Pueden ser secuenciales, o por llave. Para la segunda vertiente es necesario leer secuencialmente las llaves de la zona de índices hasta encontrar una igual o mayor que la buscada y obtener la dirección de inicio de tramo, para después leer secuencialmente el tramo en la zona de registros hasta encontrar el buscado.
- **Inserción:** no es posible insertar sin actualizar la zona de índices.
- **Eliminación:** sólo está permitido el marcado o borrado lógico.
- **Modificación:** sólo están permitidos los cambios si no aumenta la longitud del registro.

Es muy útil para **combinar consultas a registros concretos y procesamiento secuencial del fichero**. Su inconveniente principal es la **imposibilidad de introducir nuevos registros sin reorganizar los índices**.

Ejemplos

00789521T#Paula#Sanz#González#619554687\$50687452Y#José Luis#García#Viñals#
667859621\$38546998X#Javier#Peinado#Martín#666932541\$09653801B#Ruth#Lázaro#
Cardenal#689330247%



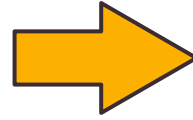
00789521T1	→	00789521TPaula	Sanz	...
09653801B4	→	50687452YJosé Luis	García	...
38546998X3	→	38546998XJavier	Peinado	...
50687452Y2	→	09653801BRuth	Lázaro	...

ZONA DE ÍNDICES

Llave	Dirección
Cáceres	1
Logroño	4
Oviedo	8
Zaragoza	10

ZONA DE REGISTROS

1	Almería
2	Álava
3	Cáceres
4	Cuenca
5	Huesca
6	Jalón
7	Logroño
8	Madrid
9	Oviedo
10	Zaragoza



04

Sistemas de ficheros

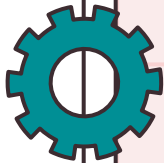




Sistemas de ficheros

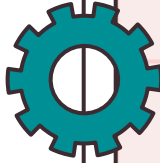
Antes de aparecer los SGBD en los setenta, la información se trataba y gestionaba usando los **sistemas de gestión de ficheros**. Surgieron al tratar de informatizar el manejo de los archivadores manuales con objeto de proporcionar un acceso más eficiente a los datos. En ellos **la definición de los datos se encuentra codificada dentro de los programas de aplicación** en lugar de almacenarse de forma independiente. Además, el control del acceso y la manipulación de los datos viene impuesto por los programas de aplicación.

Esto supone un **gran inconveniente** a la hora de tratar grandes volúmenes de información. Surge así la idea de **separar los datos contenidos en los archivos de los programas que los manipulan**, es decir, que se pueda modificar la estructura de los datos sin tener que modificar los programas con los que trabajan. Se trata de estructurar y organizar los datos de forma que se pueda acceder a ellos con independencia de los programas que los gestionan.





Inconvenientes



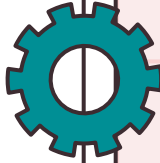
Separación y aislamiento de los datos: cuando los datos se separan en distintos ficheros, es más complicado acceder a ellos, ya que el programador de aplicaciones debe sincronizar el procesamiento de los distintos ficheros implicados para asegurar que se extraen los datos correctos.

Duplicación de datos: la redundancia de datos existente en los sistemas de ficheros hace que se desperdicie espacio de almacenamiento y puede llevar a que se pierda la consistencia de los datos. Se produce una inconsistencia cuando copias de los mismos datos no coinciden.

Dependencia de datos: la estructura física de los datos se encuentra codificada en los programas de aplicación. Para hacer un cambio el programador debe identificar todos los programas afectados por este cambio, modificarlos y volverlos a probar.



Inconvenientes



Formatos de ficheros incompatibles: al definirse la estructura de los ficheros en los programas de aplicación, es completamente dependiente del lenguaje de programación. La incompatibilidad entre ficheros generados por distintos lenguajes hace que los ficheros sean difíciles de procesar de modo conjunto.

Los sistemas de ficheros son muy **dependientes del programador** de aplicaciones, cualquier consulta o informe que se quiera realizar debe ser programado por él.

Todos estos inconvenientes **fomentaron** el desarrollo de las Bases de Datos. Se puede decir que **los sistemas de ficheros tienen una orientación al proceso**, ya que lo que condiciona los datos es el programa en sí, mientras que **las bases de datos están orientadas a los datos**, ofreciendo esa independencia entre los datos y el programa.