

# Tema 4. Sistemas de archivos.

## Introducción a las bases de datos.

### Contenidos

- 4.1. Concepto de archivo y directorio.
- 4.2. Organización de archivos.
- 4.3. Bases de datos.
- 4.4. Sistema de gestión de la base de datos

### Objetivos

- Conocer los conceptos básicos de archivo, registro, campo, directorio y estructura jerárquica de archivos.
- Conocer los distintos tipos de organización interna de archivos.
- Conocer las acciones involucradas en el acceso a un registro en cada una de las organizaciones posibles de un archivo.
- Comprender los problemas inherentes a la gestión de archivos frente al uso de una base de datos.
- Conocer las características de una base de datos.
- Conocer el concepto de sistema de gestión de base de datos y las funciones que debe cumplir.

### Bibliografía

**[Prieto 06]** A. Prieto, A. Lloris, J.C. Torres. Introducción a la Informática. 4ª Edición. McGraw-Hill, 2006.

## 4.1 Concepto de archivo y directorio

**Archivo** es un conjunto de información sobre un mismo tema, tratada como una unidad de almacenamiento y organizada de forma estructurada para la búsqueda de un dato individual.

Un archivo está compuesto de **registros** homogéneos que contienen información sobre el tema.

Los **registros** son las estructuras o unidades que forman el archivo y que contienen la información correspondiente a cada elemento individual.

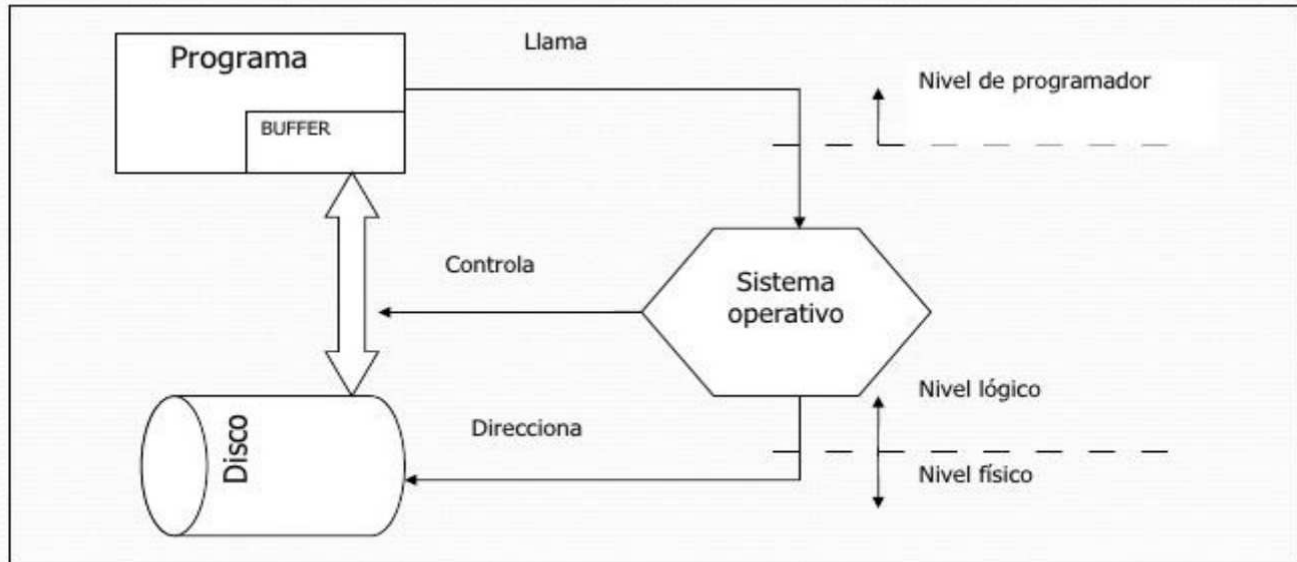
**Campo** es un dato que forma parte de un registro y representa una información unitaria o independiente.

El sistema operativo permite que el usuario pueda aludir al archivo mediante un nombre, independientemente de la forma en que se almacene en el dispositivo,

y suministra órdenes que realizan operaciones como

- crear/copiar/borrar/renombrar un archivo,
- establecer/obtener atributos de un archivo
- abrir/cerrar un archivo para el procesamiento de su contenido
- leer/escribir un registro de un determinado archivo

Figura 5.1 Gestión del acceso a archivos realizada por el sistema operativo



## Comentarios sobre la figura anterior:

- El sistema operativo transporta, cada vez que se accede al dispositivo, una cantidad fija de información que depende de las características físicas de éste: bloque o registro físico, de longitud distinta al tamaño del registro.
- El sistema operativo transforma la dirección lógica usada en los programas de usuario en la dirección física con la que se direcciona en el soporte.
- Un archivo es una estructura de datos externa al programa; en las operaciones de lectura/escritura se transfiere la información a/desde un buffer en memoria principal asociado a las entradas/salidas sobre el archivo.

## Clasificación de archivos según el tipo de registros:

1. Longitud fija.
2. Longitud variable.
  - con delimitador: un determinado carácter llamado delimitador marca el fin de un registro (suelen usarse como delimitadores el salto de línea, nulo, etc...)
  - con cabecera: cada registro contiene un campo inicial que almacena el número de bytes del registro.
3. Indefinido: el archivo no tiene realmente ninguna estructura interna, el sistema operativo no realiza ninguna gestión sobre la longitud de los registros.

En cada operación de lectura/escritura se transfieren una determinada subcadena del archivo, será el programa de usuario quien se encargue de localizar el principio y final de cada registro

Posibilidades más avanzadas: disponer de un campo clave que permita localizar un registro

## Concepto de directorio

El sistema operativo permite que el conjunto de archivos sea visible para el usuario según una estructura jerárquica en que aparece el concepto de **directorio** como agrupación de otros archivos o directorios.

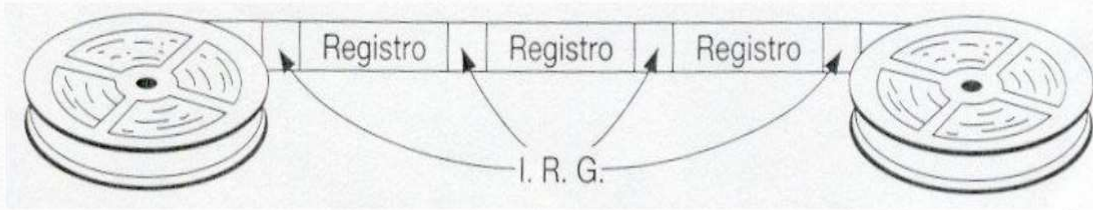
En la visión que el usuario tiene de la estructura jerárquica de archivos se utilizan los los conceptos de...

- Directorio actual o de trabajo
- Directorio inicial
- Rutas absolutas y relativas
- Lista de búsqueda
- Enlace duro, enlace simbólico.

El sistema operativo proporciona operaciones para manejar los conceptos anteriores.

### 5.2.1. Organización secuencial: los registros están almacenados físicamente contiguos.

*Ejemplo de archivo secuencial almacenado en una cinta magnética.*  
(IRG: inter record gap)



Las distintas operaciones o acciones que se pueden realizar sobre archivos con organización secuencial son:

**Añadir.** Sólo es posible escribir al final del archivo. La información se graba en el archivo escribiendo los registros en secuencia, en el orden en que se desea que estén en el archivo.

**Consulta o recuperación.** Se realiza en orden secuencial, es decir, para leer el registro que ocupa la posición **n** en el archivo es necesario leer previamente los **n-1** registros que hay antes que él.

**Inserción, modificación y eliminación.** No es fácil realizar estas operaciones sobre un archivo secuencial, siendo posible que sea necesario crear otro archivo que incorpore las actualizaciones.

La modificación de un registro necesita que no se aumente su longitud.

No es posible eliminar un registro del archivo.

Borrado lógico: marcarlo de tal forma que al leer se identifique como no válido.

Si el archivo se encuentra en un soporte direccionable y los registros son de longitud fija, entonces es posible determinar la dirección de comienzo de cada registro a partir de su posición relativa dentro del archivo.

Por tanto se puede acceder a un registro conociendo su posición relativa sin necesidad de leer el archivo secuencialmente desde el principio.

**La organización secuencial es adecuada cuando....**

- se necesita únicamente un acceso secuencial (los registros se recorren en la misma secuencia en que están almacenados)
- y se procesarán la mayor parte de los registros del archivo.

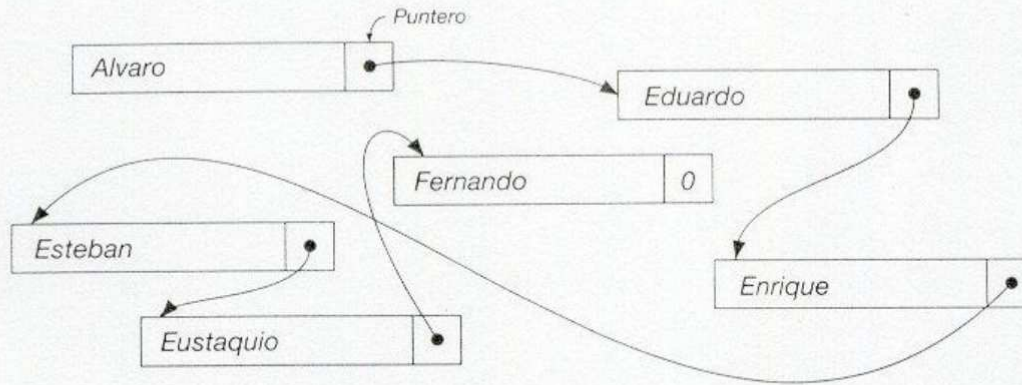
**Ventajas** de la organización secuencial:

buen aprovechamiento del espacio

es sencilla de utilizar

se puede utilizar con dispositivos secuenciales que son de bajo precio.

**4.2.2. Organización secuencial encadenada:** junto a cada registro se almacena un puntero con la dirección del registro siguiente.



Las distintas operaciones o acciones que se pueden realizar sobre archivos con organización secuencial encadenada son

**Recuperación o consulta.** La consulta es secuencial, al igual que en un archivo con organización secuencial pura.

Cada vez que se lee un registro se lee la posición del siguiente, lo que permite seguir la secuencia lógica del archivo.

**Inserción.** Para insertar un registro.....

- primero hemos de localizar la posición donde se desea insertar; esto es, entre qué dos registros se quiere que aparezca al leer el archivo.
- se escribe el registro en una zona libre y como puntero asignamos la dirección del registro que debe ser el registro siguiente.
- se modifica el registro anterior para actualizar el valor de su puntero, de forma que contenga la dirección del nuevo registro

*Ejemplo de inserción en un archivo secuencial encadenado. Se inserta un registro con llave "Gato".*



**Borrado.** Para borrar un registro se asigna al puntero del registro anterior la dirección del registro siguiente al que se desea borrar.

**Modificación.** Si el cambio no implica un aumento de longitud del registro, éste puede reescribirse en el mismo espacio. En caso contrario se debe insertar el registro, y luego, borrar la versión anterior al cambio.

La principal ventaja de la organización secuencial encadenada es la facilidad de inserción y borrado de registros;

su principal inconveniente es, al igual que en la organización secuencial pura, su limitación a consulta secuencial.

### 4.2.3. Organización secuencial indexada.

Un archivo con organización secuencial indexada está formado por dos estructuras o zonas distintas: zona de registros y zona de índices

**zona de registros:** se encuentran los registros en sí, ordenados según el valor de una llave; en esta zona se pueden direccionar los registro; está dividida en determinados tramos (conjunto de registros consecutivos)

Por cada tramo de la zona de registros hay un registro en la **zona de índices** que contiene:

- \* el valor mayor de la llave del tramo (valor de llave del último registro del tramo)
- \* la dirección del primer registro del tramo



La gestión de la estructura la realiza el sistema operativo o un paquete software especial, por lo que el usuario de esta estructura no necesita conocer la existencia de ambas zonas, pudiendo contemplar ambas como un todo.

Con archivos que usen organización secuencial indexada se pueden realizar las siguientes operaciones a nivel de registro:

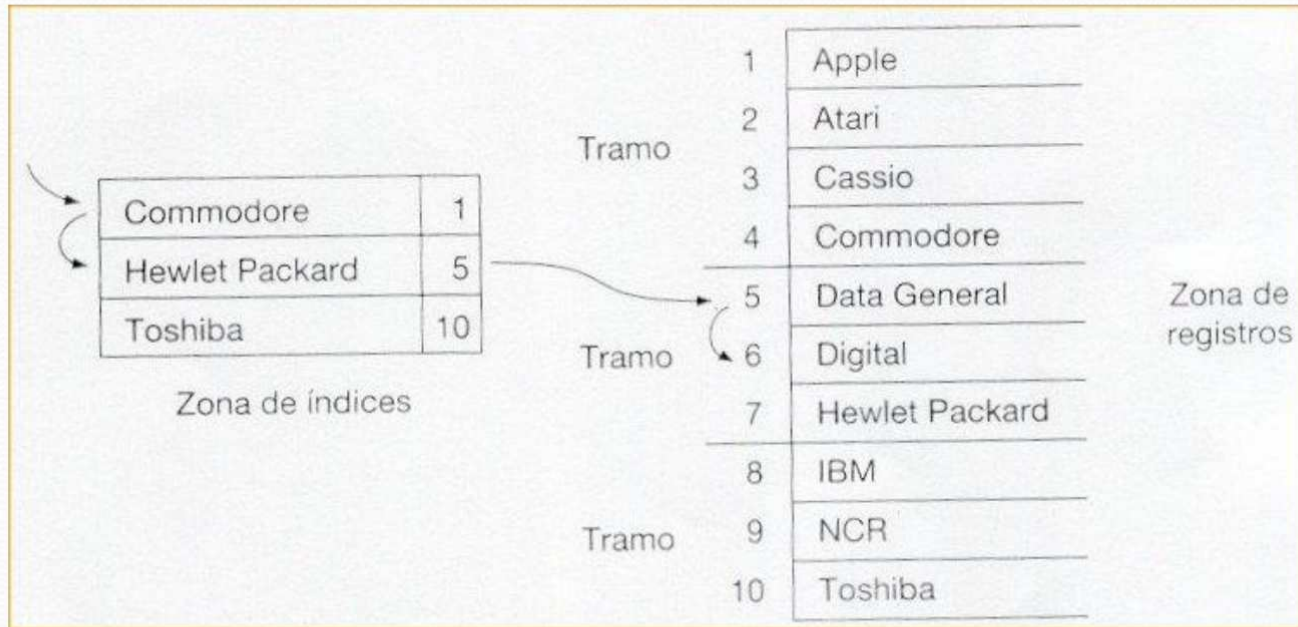
**Consulta.** Esta organización de archivo permite realizar consultas por llave (esto es, localizar un registro conocida su llave) sin necesidad de leer los registros que le anteceden en el archivo.

El procedimiento a seguir para realizar una consulta por llave es:

Leer en secuencia las llaves en la zona de índices hasta encontrar una mayor o igual a la del registro buscado. Obtener la dirección de comienzo del tramo donde está el registro.

Leer en secuencia en la zona de registros a partir de la dirección obtenida en la zona de índices hasta encontrar el registro buscado o uno con valor de llave mayor que el buscado. (En este último caso el registro no se encuentra en el archivo)

*Ejemplo: Consulta de un registro por llave, se busca el registro con llave "Digital".*



**Inserción.** Dado que ambas zonas son secuenciales, no es posible insertar un registro en archivos con esta organización.

En algunos casos se permite la escritura de nuevos registros al final de la zona de registros. Estos registros, como es lógico, no podrán ser consultados por llave con el procedimiento antes descrito.

**Borrado y modificación.** Al estar los registros escritos en secuencia no es posible borrar un registro.

La única forma de eliminar la información contenida en un registro es marcándolo, lo que se conoce como **borrado lógico**.

Las modificaciones son posibles tan sólo si el registro no aumenta de longitud al ser modificado y no se altera el valor de la llave del mismo.

#### 4.2.4. Organización directa o aleatoria.

Un archivo con **organización directa o aleatoria (“random”)** *es un archivo escrito sobre un soporte de acceso directo para el cual existe una transformación conocida que genera la dirección de cada registro en el archivo a partir de un campo que se usa como llave.*

El nombre de “aleatorio” se debe a que normalmente no existe ninguna vinculación aparente entre el orden lógico de los registros y su orden físico.

Un problema fundamental de esta organización es elegir adecuadamente de la transformación o **método de direccionamiento** que se ha de utilizar.

##### **Situaciones no deseadas que pueden ocurrir:**

Hay direcciones que no se corresponden con ninguna llave y, por tanto, habrá zonas de disco sin utilizar.

Hay direcciones que corresponden a más de una llave. En este caso se dice que las llaves son sinónimas para esa transformación.

Hay dos **formas de resolver el problema de los sinónimos**, siempre a costa de complicar la estructura del archivo:

Cuando se asocia a una llave una dirección ya ocupada por un registro distinto (esto es, por un sinónimo de esta llave ), se busca en el archivo hasta encontrar una posición libre donde escribir el registro.

Se reserva una **zona de desbordamientos** donde se escribirán los registros que no se pueden escribir en la posición que les corresponde según la transformación, a la que denominaremos zona principal

Hay tres **métodos de direccionamiento** para los archivos de organización directa:

### 1) Direccionamiento directo.

Se utiliza como dirección la propia llave

Sólo es factible cuando la llave es numérica y su rango de valores no es mayor que el rango de direcciones en el archivo.

En algunos casos pueden quedar lagunas de direcciones sin utilizar, en lugares conocidos de antemano. En este caso se pueden ocupar dichas direcciones desplazando las direcciones superiores.

Por ejemplo, el archivo de habitaciones de un hotel puede organizarse en forma aleatoria con direccionamiento directo, sin más que hacer la dirección igual al número de habitación.

## 2) Direccionamiento asociado.

El direccionamiento asociado se puede utilizar para cualquier tipo de llave. Si se utiliza este método debe construirse una tabla en la que se almacena para cada llave la dirección donde se encuentra el registro correspondiente.

## 3) Direccionamiento calculado ("hashing").

La dirección de cada registro se obtiene realizando una transformación sobre la llave.

Se utiliza cuando....

- \* la llave no es numérica; se usará una conversión previa para obtener un número a partir de ella.

Por ejemplo se usa el equivalente decimal al propio código binario del carácter (al carácter **A** le correspondería el 65, ...)

- \* la llave es numérica pero toma valores en un rango inadecuado para usarse directamente como dirección.

Operaciones sobre un archivo con organización directa:

**Consulta.** La consulta se realiza por llave.

Para leer un registro debe aplicarse a la llave el algoritmo de transformación y posteriormente leer el registro en la dirección resultante.

Si el registro con la llave buscada no se encuentra allí se procederá según cómo se haya resuelto gestionar los sinónimos.

**Borrado.** Siempre se realiza un borrado lógico, pudiéndose reutilizar el espacio del registro eliminado.

**Modificación e inserción.** Siempre se puede modificar o insertar un nuevo registro, realizando la transformación de la llave correspondiente.

La organización directa es útil para archivos donde los accesos deben realizarse por llave, accediéndose siempre a registros concretos.

Si la información se va a procesar en conjunto, con frecuencia puede ser más rentable una organización secuencial indexada.

En una aplicación convencional con archivos aparecen los siguientes **problemas**:

**1.Dificultad de mantenimiento.** Si hay archivos con información parcialmente duplicada, realizar las actualizaciones necesarias es un problema complejo y costoso. Normalmente, es necesario actualizar varios archivos con diferentes organizaciones. Si la actualización no se realiza correctamente se tendrán informaciones incoherentes.

**2.Redundancia.** Se dice que hay redundancia si un dato se puede deducir a partir de otros datos (se dan los problemas explicados en el caso anterior)

**3.Rigidez de búsqueda.** El archivo se concibe para acceder a los datos de un determinado modo. Sin embargo, en la mayoría de los casos es necesario (o al menos deseable) combinar acceso secuencial y directo por varias claves.

- 4. Dependencia con los programas.** En un archivo no están reflejadas las relaciones existentes entre campos y registros. Es el programa, que trabaja con el archivo, el que determina en cada caso dichas relaciones.

Esto implica que cualquier modificación de la estructura de un archivo obliga a modificar todos los programas que lo usen.

Esto ocurre aun en el caso de que la alteración sea ajena al programa. Así por ejemplo, si se aumenta la longitud de un campo habrá que modificar incluso los programas que no lo usan.

- 5. Seguridad.** Uno de los mayores problemas de cualquier sistema de información es mantener la seguridad necesaria sobre los datos que contiene.

Si se está trabajando con archivos el control deberá realizarlo el propio programa.

El aspecto de la seguridad es particularmente deficitario en los sistemas de archivos.

### Concepto de base de datos.

Las bases de datos surgen como alternativa a los sistemas de archivos, intentando eliminar o al menos reducir sus inconvenientes.

Podemos definir una base de datos así:

Una **base de datos** es un sistema formado por un conjunto de datos y un paquete software para gestión de dicho conjunto de datos de tal modo que:

*se controla el almacenamiento de datos redundantes,  
los datos resultan independientes de los programas que los usan,  
se almacenan las relaciones entre los datos junto con éstos y  
se puede acceder a los datos de diversas formas.*

En una base de datos se almacenan las relaciones entre datos junto a los datos.

Esto, y el utilizar como unidad de almacenamiento el campo además del registro, es el fundamento de la independencia con los programas de aplicación.

Requisitos que debe cumplir un buen sistema de base de datos:

- 1. Acceso múltiple.** Diversos usuarios pueden acceder a la base de datos, sin que se produzcan conflictos, ni visiones incoherentes.
- 2. Utilización múltiple.** Cada usuario podrá tener una imagen o visión particular de la estructura de la base de datos.
- 3. Flexibilidad.** Se podrán usar distintos métodos de acceso, con tiempos de respuesta razonablemente pequeños.
- 4. Seguridad.** Se controlará el acceso a los datos (a nivel de campo), impidiéndoselo a los usuarios no autorizados.
- 5. Protección contra fallos.** Deben existir mecanismos concretos de recuperación en caso de fallo de la computadora.

- 6. Independencia física.** Se puede cambiar el soporte físico de la base de datos sin que esto repercuta en la base de datos ni en los programas que la utilizan.
- 7. Independencia lógica.** Se pueden modificar los datos contenidos en la base, las relaciones existentes entre ellos o incluir nuevos datos, sin afectar a los programas que la usan.
- 8. Redundancia controlada.** Los datos se almacenan una sola vez.
- 9. Interfaz de alto nivel.** Existe una forma sencilla y cómoda de utilizar la base al menos desde un lenguaje de programación de alto nivel.
- 10. Interrogación directa (“query”).** Existe una utilidad que permite el acceso a los datos de forma interactiva o conversacional.

### Estructura de una base de datos.

En una base de datos se almacena información de una serie de objetos o elementos. Estos objetos reciben el nombre de **entidades**, siendo una entidad cualquier ente sobre el que se almacena información.

De cada entidad se almacenan una serie de datos que se denominan **atributos** de la entidad. Puede ser atributo de una entidad cualquier característica o propiedad de ésta.

Las entidades y los atributos son conceptos abstractos. En una base de datos la información de cada entidad se almacena en **registros**, y cada atributo en **campos** de dicho registro, de forma análoga al almacenamiento en archivos. Sin embargo, en una base de datos **hay diferentes tipos de registros**, uno por cada entidad.

Normalmente se reserva el nombre "registro" para especificar un "tipo de registro", utilizándose otros nombres para especificar cada una de las apariciones de ese registro en la base de datos, tales como: **elemento**, **valor actual de registro**, u **ocurrencia de registro**.

Se dice que un conjunto de atributos de una entidad es un **identificador** o **clave primaria** de dicha entidad si el valor de dichos atributos determina de forma unívoca cada uno de los elementos de dicha entidad, y no existe ningún subconjunto de él que sea identificador de la entidad.

En una base de datos se almacenan además de las entidades, las **relaciones** existentes entre ellas.

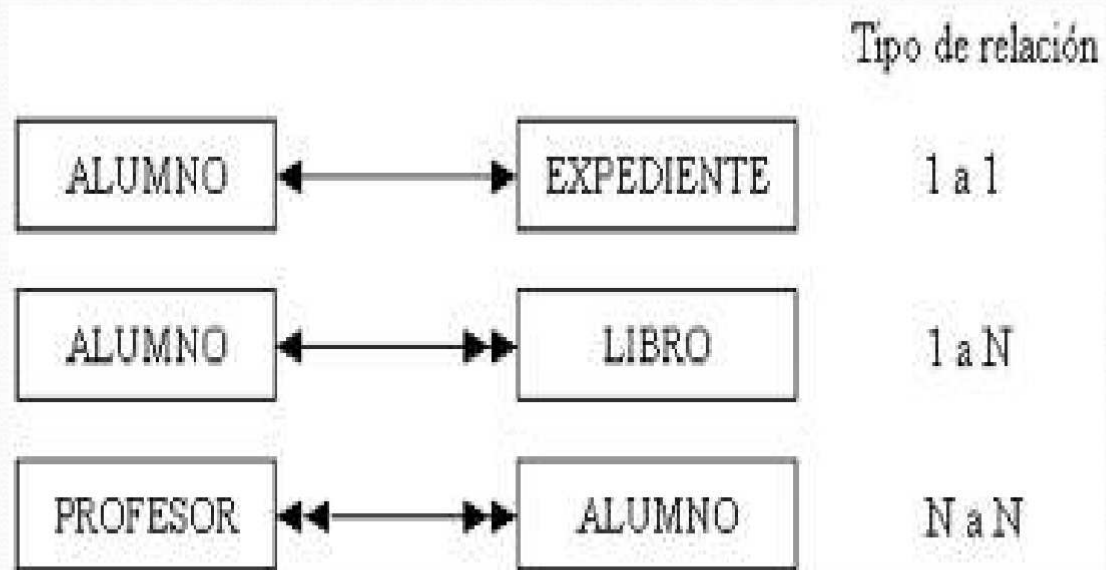
Estas relaciones se almacenan con punteros que inserta automáticamente el sistema de gestión de la base de datos (esta representación mediante de punteros es "transparente" al usuario)

Las relaciones entre entidades pueden ser de estos tipos:

**Relación 1 a 1:** Entre las entidades A y B existe una relación 1 a 1 si a cada elemento de una entidad A le corresponde un único elemento de B y viceversa.

**Relación 1 a muchos (o 1 a N):** Entre las entidades A y B existe una relación 1 a N si a cada elemento de una entidad A le pueden corresponder más de un elemento de B; a cada elemento de B le corresponde un único elemento de A.

**Relación muchos a muchos (o N a N):** Entre las entidades A y B existe una relación 1 a N si a cada elemento de una entidad A le pueden corresponder más de un elemento de B, y a cada elemento de B le puede corresponder más de un elemento de A.



Al definir la base de datos se deben especificar cada uno de los registros que la integran, indicando los campos que los componen y las relaciones que los ligan.

Tal conjunto de especificaciones recibe el nombre de esquema de la base de datos.

El **esquema de una base de datos** es pues la definición de su estructura lógica; suele representarse en forma gráfica o por simple enumeración.

Para que un programa concreto pueda acceder a una base de datos es necesario especificar la estructura lógica de la parte de base de datos que éste va a usar.

La descripción de la estructura lógica de una parte de la base de datos que va a ser utilizada por uno o más programas recibe el nombre de **vista** o **subesquema**.

El subesquema permite que varios usuarios utilicen distintas “visiones” de una misma base de datos.

Permite también limitar el acceso a distintas partes de la base de datos, para realizar tan solo determinadas acciones.