Almacenamiento y bases de datos

Ramon Costa i Pujol
Jaume Raventós Moret

PID_00153114

UOC
Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

La aplicación práctica de los ordenadores y de los sistemas informáticos es el **tratamiento de la información**. Por eso, al conjunto de sistemas informáticos y mecanismos de comunicación se lo denomina *tecnologías de la información y la comunicación* (TIC).

Ved también

Para saber qué comporta el tratamiento de la información en un sistema informático, podéis consultar el subapartado "Los ordenadores tratan información" en el módulo "Ordenadores y sistemas operativos".

Todos los programas informáticos gestionan datos, que son captados, analizados, tratados y presentados a fin de que el usuario pueda tomar decisiones o llevar a cabo otras acciones sobre ellas.

Por lo tanto, es necesario poder **almacenar los datos** e informaciones que el ordenador y los sistemas de información tienen que tratar.

En un ordenador hay dos tipos **de memoria principal**: RAM y ROM.

Ved también

Para tener más detalles sobre el funcionamiento de la memoria RAM y ROM, podéis consultar el subapartado"Estructura básica de un ordenador" en el módulo "Ordenadores y sistemas operativos".

- La información de la **memoria ROM** no se borra nunca. Almacena el test de fiabilidad del ordenador (un conjunto de instrucciones por medio de las cuales comprueba el estado de sus componentes cada vez que se enciende), las rutinas de inicio y arranque, y la BIOS.

  Sistema básico de entrada y salida

  La BIOS (*basic input / output system*) es un pequeño conjunto de instrucciones que utiliza el procesador para encontrar el sistema operativo y cargarlo a la memoria RAM. También sirve para gestionar el flujo de datos entre el sistema operativo y los dispositivos del ordenador.

ROM

ROM (*read only memory*) es memoria de sólo lectura.

Esta memoria es **permanente**, es decir, la información no se pierde al desconectar el ordenador. Las instrucciones y los datos de la ROM permanecen allí cuando se apaga el ordenador.

- A la **memoria RAM** se guarda temporalmente la información de los programas y procesos que se ejecutan. Se puede leer y escribir. Para que un programa se pueda ejecutar se tiene que cargar (almacenar) en la memoria RAM.

RAM

RAM (*random access memory*) es memoria de acceso aleatorio.

Esta memoria es **volátil**, es decir, en el momento en que se apaga el ordenador se pierde toda la información que hay almacenada.

Estos tipos de memoria, pues, no cumplen un requisito imprescindible para almacenar datos que es el "almacenamiento permanente de gran cantidad de información", por lo que son necesarios **dispositivos de almacenamiento** externo, como, por ejemplo, los discos magnéticos, los CD-ROM, los DVD o los lápices de memoria (*pen drive* o *USB flash drive*).

Ved también

Para conocer los diferentes dispositivos de almacenamiento externo, podéis consultar el subapartado "¿Cuáles son las unidades de almacenamiento más corrientes?" del apartado "Soportes físicos de información" en el módulo "Ordenadores y sistemas operativos".

Por otra parte, esta información que hay que almacenar y gestionar se tiene que estructurar de manera que la puedan tratar fácilmente tanto los ordenadores y programas informáticos como, sobre todo, sus usuarios.

Por ello, existen diferentes **estructuras de datos** que permiten almacenar esta información: los ficheros y las bases de datos. Este módulo trata los principales aspectos relacionados con estas dos maneras de organizar la información.

- En un **fichero** se almacena información que hace referencia al mismo tema de una manera estructurada para poder trabajar con los datos de manera individual.
  Cada fichero se compone de estructuras más simples llamadas **registros**. Cada registro está formado por **campos** que contienen información en lo referente a un elemento o característica en particular dentro del fichero.
  Las principales operaciones que se pueden llevar a cabo sobre un fichero son su creación, la lectura y actualización de la información (registros y campos), la ordenación y consulta de los registros, su eliminación y la generación de informes.
  El tipo de organización de un fichero (secuencial, directa, etc.) condiciona las operaciones que se pueden hacer con él y, por lo tanto, viene determinado por la aplicación que se le dé.
  Vistas sus limitaciones, los ficheros han sido sustituidos, en la mayoría de aplicaciones, por bases de datos.

- Las **bases de datos** surgieron para intentar resolver los problemas que tenían los ficheros (inconsistencia de los datos, redundancia, rigidez en las buscas, dependencia de los programas...).

Una base de datos es una serie de datos organizados de manera que se pueda acceder a sus contenidos y éstos se puedan administrar y actualizar con facilidad.

Un **sistema gestor de bases de datos** (SGBD) es el conjunto de software destinado a la creación, gestión, control y manipulación de la información almacenada en una base de datos.

El SGBD dispone de un **lenguaje de definición de datos** (DDL) que permite definir el esquema de la base de datos, un **lenguaje de manipulación de datos** (DML) que facilita la gestión de la información almacenada, y una **interfaz de usuario** que permite acceder al sistema y trabajar con comodidad.

Existen tipos muy diversos de bases de datos. Esta asignatura se centra en las relacionales (el modelo más utilizado), que permiten gestionar información estructurada, y en las documentales, que gestionan información no estructurada.

- Una **base de datos relacional** está formada por **tablas**, **que son estructuras de filas** (los **registros** de información) y **columnas** (los **campos** de información de los registros).
  El diseño y creación de una base de datos relacional consiste en definir el **modelo relacional**, es decir, las tablas, sus campos y las relaciones entre tablas, principalmente.
  Previo a este diseño, sin embargo, se aconseja **elaborar el modelo conceptual**, que permite plasmar en un esquema entidad-relación las diferentes informaciones (entidades, atributos e interrelaciones) que se quieren almacenar y gestionar en la futura base de datos relacional.

- Una **base de datos documental** se caracteriza porque cada registro se corresponde con un documento de cualquier tipo (publicación, documento gráfico o sonoro...) o con su referencia.
  La información que contiene se estructura en diferentes campos: unos se refieren a la descripción formal del documento, otros tratan sobre su contenido temático, e incluso uno puede guardar el propio documento.
  Según la información contenida y la referencia al documento correspondiente, se puede clasificar en: base de datos de **texto completo** (contiene los propios documentos), **archivo electrónico de imágenes** (contiene imágenes de los documentos originales) y base de datos **referenciales** (contiene referencias para localizar los documentos originales).

Objetivos

Los **objetivos generales** que el estudiante puede alcanzar son los siguientes:

1. Identificar la necesidad de las bases de datos (BD), y sus ventajas en relación con los ficheros.
2. Conocer los elementos de un sistema gestor de bases de datos (SGBD) y sus principales funcionalidades.
3. Adquirir una visión general de las tipologías de BD, especialmente las relacionales y las documentales.
4. Conocer los fundamentos del diseño de BD relacionales.

Estos objetivos generales se desglosan en los **objetivos específicos** siguientes:

1. Identificar la necesidad de almacenar y gestionar la información por parte de las aplicaciones informáticas.
2. Definir los conceptos de fichero, registro, campo, registro lógico frente a registro físico y bloque.
3. Identificar el proceso de intercambio de información entre la memoria externa y la interna de un ordenador.
4. Definir las principales operaciones que se pueden llevar a cabo con un fichero.
5. Clasificar los ficheros según dos criterios: la longitud de los registros y su uso.
6. Describir las diferentes maneras como se puede organizar un fichero.
7. Enumerar las limitaciones de uso y aplicación de los ficheros.
8. Definir las principales funcionalidades de las BD.
9. Enumerar aplicaciones prácticas de las bases de datos.
10. Definir las principales funcionalidades y componentes o elementos de un SGBD.

11. Identificar diferentes SGBD existentes en el mercado.

12. Enumerar los principales tipos de BD y su manera de organizar los datos.

13. Describir los principales conceptos de las bases de datos relacionales y la estructuración de sus datos.

14. Distinguir entre los lenguajes de definición de los datos (DDL) y los de manipulación de estos datos (DML).

15. Identificar las opciones disponibles para trabajar con una BD desde un lenguaje de programación.

16. Identificar los principales elementos de un modelo conceptual de bases de datos según el modelo entidad-relación.

17. Definir una base de datos relacional a partir de un modelo entidad-relación.

18. Enumerar las diferentes opciones para normalizar el diseño de una BD.

19. Conocer los conceptos básicos de bases de datos documentales (BDD).

20. Enumerar los diferentes sistemas de recuperación de la información en bases de datos documentales.

21. Distinguir entre distintas categorías de bases de datos documentales.

22. Identificar tres tipos diferentes de modos de acceso a la información de una base de datos documental.

23. Distinguir entre las BD centradas en un solo tipo de documento y las que incorporan diferentes tipos de documentos.

24. Diferenciar las bases de datos documentales según el modelo de tratamiento documental.

1. Ficheros

Un fichero, también denominado *archivo*, es un conjunto ordenado de datos que mantienen entre sí una relación lógica y se almacenan en un soporte de información para la comunicación con el ordenador.

Ejemplo de estructura de un fichero

El fichero que almacena los datos de los abonados a plazas de aparcamiento de un garaje se puede describir gráficamente de la siguiente manera:

Nombre de abonado	NIF de abonado	Matricula de vehículo	Tipo de vehículo	Plaza de aparcamiento	....
Ramon Pujol	46587875	2587-CDD	monovolumen	25	
Jaume Moret	46585965	5874-AAF	furgoneta	36	
Marc Costa	45898745	2698-CCE	todo terreno	24	
Marta Pou	58954785	2598-BDA	motocicleta	14	
Albert Raventós	54785478	5698-BEE	motocicleta	05	
Montse Molina	58745854	1985-CCA	monovolumen	32	
Núria Camps	47858585	8847-TKM	ciclomotor	18	
....					

En un fichero se almacena información que hace referencia al mismo tema de una manera estructurada para poder trabajar con los datos de manera individual.

Los ficheros se componen de estructuras más simples denominadas *registros*. Todos los registros de un fichero son del mismo tipo.

Cada registro está formado por **campos** que contienen información en lo referente a un elemento o característica en particular dentro del fichero.

Fichero de propietarios de plazas de aparcamiento en un garaje

Gráficamente, podríamos describir un fichero de la manera siguiente:

Propietario	NIF	Matrícula	Plaza de aparcamiento	...
Ramón Pujol	46587875	2587-CDD	25	

Propietario	NIF	Matrícula	Plaza de aparcamiento	...
Alberto Costa	45898745	2698-CCE	24	
Antonio Pou	58954785	2598-BDA	14	
Marcos Cartagena	54785478	5698-BEE	05	
Montse Monte	58745854	6985-CCA	36	
Nuria Camps	47858585	B-5847-TX	14	
Pedro Soler	46585965	5874-AAF	36	
...				
...				

Dentro de un fichero determinado, los registros se identificarán por un campo o un conjunto de campos denominados *clave*. Estos identificadores servirán para distinguir cada registro de los otros, así como para facilitar la localización rápida de los registros dentro de los ficheros (por ejemplo, el NIF). Un fichero puede tener una clave, varias o ninguna.

La mayoría del software (procesadores de texto, gestores de hojas de cálculo, etc.) también trabaja con ficheros para poder almacenar la información que gestiona. Por ejemplo, cualquier documento elaborado con un procesador de texto se guarda como un fichero, así como cualquier foto o vídeo que almacenemos en nuestro disco duro. En estos casos, sin embargo, la información se visualiza por pantalla, de una manera diferente a como se ha guardado. El programa (procesador de texto, sistema operativo, etc.) se encarga de leer el contenido de aquel fichero y mostrarlo de una manera comprensible para el usuario.

Los archivos se almacenan en los dispositivos de **memoria masiva**, también denominados de *memoria auxiliar*, como por ejemplo los discos duros, los disquetes, CD, DVD, lápiz de memoria (*pen drive* o *flash memory*) y otros dispositivos de memoria con conexión USB.

Para saber más

Para conocer las características de los diferentes tipos de soportes de almacenaje, podéis consultar el subapartado 1.4.6 del módulo "Ordenadores y sistemas operativos".

Distintas denominaciones para el almacenaje permanente de datos

Hay muchas maneras de denominar un soporte físico digital de almacenaje permanente de datos, según los términos utilizados para su descripción: dispositivo (unidad o soporte) de almacenaje (memoria) auxiliar (secundario/a o externo/a o masivo/va). Así, es correcto usar, entre otros, cualquiera de las siguientes denominaciones genéricas:

- Dispositivo de almacenamiento secundario.
- Dispositivo de memoria auxiliar.
- Dispositivo de memoria masiva.
- Soporte de almacenaje externo.
- Soporte de almacenaje masivo.

- Unidad de memoria externa.
- Unidad de almacenaje secundario.
- Etc.

Un elemento que se debe considerar en los ficheros, tal y como se comentará más adelante, es su organización, de la que distinguiremos cuatro tipos: **secuencial**, **secuencial encadenada**, **secuencial indexada** y **directa o aleatoria**. Este aspecto, tal como se comenta más adelante, está relacionado con el tipo de acceso a los ficheros.

Para saber más

Para conocer cómo se organizan y cómo es el acceso a los ficheros, podéis consultar el subapartado 1.4 de este módulo.

En los siguientes apartados, también se presentan las principales operaciones que se pueden hacer con los ficheros y las diferentes tipologías en las que éstos se clasifican.

1.1. Proceso de entrada y salida de datos. Almacenaje

Un fichero está compuesto por un conjunto de registros, todos del mismo tipo (por ejemplo, los datos de los propietarios de los coches vendidos en un concesionario), y cada registro está compuesto por campos que contienen información en lo referente a un elemento o característica del registro (por ejemplo, el número de la matrícula, modelo y marca del coche, NIF del propietario, fecha de la venta, etc.).

Proceso de entrada y salida de datos

Los procesos de entrada y salida de datos corresponden a las operaciones de escritura y lectura, respectivamente, mediante las que se añade, modifica y/o se accede a la información de los registros.

El **acceso a los dispositivos** de memoria masiva donde se almacenan los ficheros puede ser de dos tipos: secuencial y directo.

- En los **soportes de acceso secuencial** (por ejemplo, una cinta) para acceder al registro *n* se tienen que leer, uno a uno, los *n-1* registros anteriores en el orden en el que están escritos.

- En los **soportes de acceso directo** (por ejemplo, un disco duro o un disquete), se puede acceder directamente a un registro físico solamente dando su dirección física.

Se diferencia entre **registro lógico**, entendido como la información correspondiente a uno de los elementos del fichero, y **registro físico** (o bloque), que es el conjunto de información que se puede leer y escribir al mismo tiempo en el soporte físico.

Los registros físicos se almacenan en el dispositivo correspondiente, y el sistema operativo es el encargado de escribir y leer los datos que componen el fichero. El sistema operativo transporta, cada vez que se accede al dispositivo (para leer o escribir), una cantidad fija de información (bloque o registro físico) que depende de las características físicas o del hardware.

Para saber más

Para saber más sobre el proceso de escritura y lectura de datos en un fichero, podéis consultar el sistema de ficheros y la gestión de E/S del sistema operativo en el subapartado 2.2 del módulo "Ordenadores y sistemas operativos".

En general, un bloque puede tener un número variable de registros lógicos, es decir, se pueden transferir distintos registros lógicos en una sola operación de escritura/lectura. Este hecho recibe el nombre de **bloqueo**. El número de registros lógicos contenidos en un bloque (longitud del bloque) recibe el nombre de **factor de bloqueo**.

El bloqueo de registro aporta dos grandes ventajas:

- Más velocidad en los procesos de entrada/salida (E/S), ya que cuanto mayor es el factor de bloqueo, menos accesos serán necesarios hacer en el dispositivo físico para tratar la información.

- Mejor aprovechamiento de la capacidad de soporte del almacenamiento.

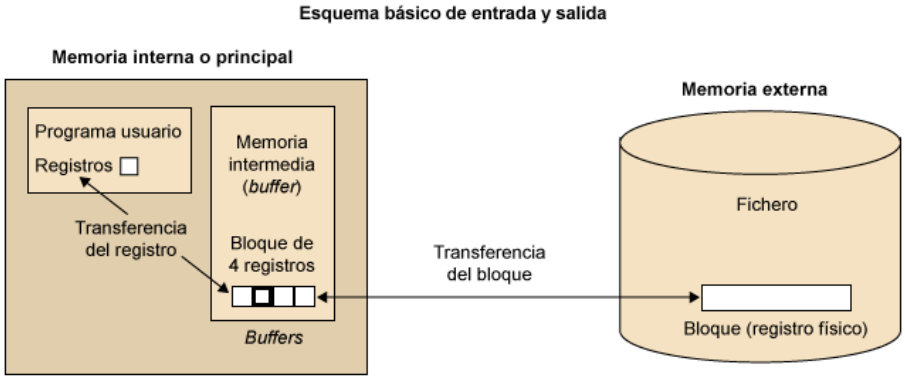
La **dirección lógica** (dirección software) de un registro es la posición relativa que ocupa en el fichero, mientras que la **dirección física** es la posición real donde se encuentra el registro en el soporte de información (dirección hardware).

En el fichero, los registros se aparecen al usuario de manera lógica, es decir, ordenados según su dirección lógica. En cambio, el orden físico de los registros de un fichero en el disco puede no tener ninguna relación con la información que contiene.

El **sistema operativo** es el responsable de realizar la transformación de **la dirección lógica**, utilizada en los programas, en **la dirección física** con la que se almacena en el dispositivo.

Desde un programa se accede a un fichero para leer y modificar uno de sus registros o escribir. Al leer, se transfiere de bloque a bloque (registro físico) la información del fichero a un área de memoria principal denominada *memoria intermedia* (*buffer*).

Desde esta memoria intermedia se transfiere la información que el programa puede programar. Y de la misma manera, el programa puede transferir información desde esta zona al fichero y modificar su contenido.



Aquí el factor de bloqueo (número de registros contenidos en un bloque) es cuatro.

Desde la memoria intermedia, los datos son procesados por el programa.

De este modo, el programa transfiere información desde esta memoria al fichero, con lo cual se actualiza su contenido.

1.2. Operaciones con ficheros

Los programas, o aplicaciones informáticas, acceden a los ficheros para llevar a cabo un conjunto de **operaciones** con la información almacenada. A continuación, se describen las operaciones más habituales:

a) Creación. Es la primera operación que se debe hacer sobre un fichero. Consiste en definir las características de los datos. A partir de este momento, ya se puede añadir registros al fichero y trabajar con ellos.

b) Lectura. Consiste en recuperar la información del fichero en el nivel del registro.

c) Mantenimiento o **actualización**. La modificación de un fichero incluye tres acciones posibles:

- **Inserción** de un registro nuevo.
- **Modificación** de un registro, cambiando uno o más valores.
- **Eliminación** de un registro.

d) Ordenación. Consiste en alterar el orden de los registros de un fichero según uno o más criterios, y a partir de los valores de uno o más campos, que reciben el nombre de claves de ordenación.

e) Búsqueda. Operación que consiste en localizar dentro de un fichero un registro concreto. Hay diferentes procedimientos para llevar a cabo la búsqueda de uno o más registros en un fichero:

- **Búsqueda secuencial**. Representa examinar cada uno de los registros del fichero hasta que se encuentra el solicitado.

- **Búsqueda dicotómica o binaria**. Aplicable sólo en el caso de ficheros ordenados. Consiste en dividir los ficheros en subficheros cada vez más y más pequeños. Se empieza leyendo el registro del medio y, si el valor que se ha de buscar es superior, se descartan los registros de la primera mitad del fichero. Si el valor que se busca es inferior, se descartan los de la segunda mitad del fichero. Con los registros elegidos se repite el proceso de búsqueda, hasta encontrar el registro buscado o comprobar que no está.

- **Búsqueda por bloques**. Con este método se realiza el mismo recorrido sobre las claves que en un fichero secuencialmente indexado. En primer lugar, se determinará el bloque en el que se encuentra el registro. Para hacerlo, se lee el último registro de cada bloque, hasta encontrar el buscado o uno mayor. En este último caso, se pasará a buscar el registro en el bloque anterior y se efectúa una búsqueda secuencial.

Para saber más

Para tener más información sobre la ordenación de ficheros, podéis consultar el apartado "Organización de ficheros" en este módulo.

f) Fusión. En esta operación se unen dos o más ficheros para integrarlos en uno solo.

g) División. Operación contraria a la fusión y que divide la información de un fichero en dos o más.

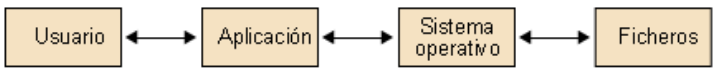
h) Generación de informes a partir de datos contenidos en el fichero. Los informes pueden ser impresos, por pantalla, en formato web, etc.

i) Destrucción. Eliminación del fichero del dispositivo correspondiente.

Las aplicaciones informáticas y los sistemas operativos incluyen una serie de programas útiles para llevar a cabo las operaciones básicas con ficheros: creación, eliminación, lectura, etc. Estos programas reciben el nombre de **sistemas de gestión de ficheros**.

En general, un fichero utilizado por un usuario desde un lenguaje de alto nivel, por ejemplo Visual Basic, C++ o Java, no lo gestiona directamente el mismo programa, sino el sistema operativo, responsable de realizar los accesos necesarios al dispositivo en el que se archiva y de transferir la información solicitada del fichero al programa y a la inversa.

Gestión del sistema operativo



1.3. Tipos de ficheros

Los ficheros se pueden clasificar según dos criterios principales:

- Según la longitud de los registros que contienen. En este caso, se distingue entre los ficheros de longitud fija, los de longitud variable, los delimitados y los indefinidos.

- Según el uso que se hará de éstos. Se pueden clasificar entre ficheros permanentes y temporales, y en cada uno de estos casos se distinguen entre ficheros maestros o de situación, ficheros constantes o ficheros históricos e intermedios, de maniobras o de resultados.

A continuación, se describen las características principales de cada uno de estos tipos de ficheros.

1.3.1. Según la longitud de los registros

Los registros que forman parte de un fichero pueden tener la misma longitud o una diferente, tanto si se debe a la existencia de campos de longitud variable como al hecho de que contienen campos de longitud fija pero que se repiten un número variable de veces, así como si es por las dos causas.

Si se sigue este criterio, los ficheros se pueden clasificar o distinguir entre los tipos siguientes:

- **Ficheros de longitud fija**. Son aquéllos en los que la suma de los caracteres de todos los campos de los registros es constante. Todos los registros, por lo tanto, tienen la misma longitud.

Ficheros de longitud fija

R	A	M	O	N					C	O	S	T	A					3	6	5	4	-	C	D	A		
P	E	D	R	O					P	U	J	O	L					B	-	2	5	4	5	-	T	X	
...																											
...																											
...																											

- **Ficheros de longitud variable**. Son los ficheros en los que cada registro puede tener una longitud diferente en el fichero, que oscila entre una pequeña variación entre un mínimo y un máximo. En estos casos, el sistema reserva una palabra al inicio de cada registro, en la que anota su longitud.

Ficheros de longitud variable

#18#	R	A	M	O	N	C	O	S	T	A	3	6	5	4	-	C	D	A	#19	P	E	D	R	O
P	U	J	O	L	S	B	-	2	5	4	5	-	T	X	#	-	-	-	-		-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

El número que acompaña al símbolo # indica la longitud de cada registro.

- **Ficheros delimitados**. En estos tipos de ficheros, la longitud del registro es variable y no se puede saber en qué medida difieren unos de otros. El sistema incluye un carácter especial para indicar el final del registro. En estos casos, se dice que los ficheros son de tipo texto.

Ficheros delimitados

R	A	M	O	N	C	O	S	T	A	3	6	5	4	-	C	D	A	;	P	E	R	E	P
U	J	O	L	S	B	-	2	5	4	5	-	T	X	;	-	-	-	-	-		-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

En este caso, el caracter punto y coma (";") indica dónde acaba cada registro.

- **Ficheros indefinidos**. Son aquéllos en los que la longitud es totalmente variable. En estos casos, el sistema operativo no realiza ninguna gestión sobre la longitud de los registros del fichero. Será el programa el encargado de localizar el principio y el final de cada registro.

Ficheros indefinidos

R	A	M	O	N	C	O	S	T	A	3	6	5	4	-	C	D	A	P	E	D	R	O	P	U
J	O	L	S	B	-	2	5	4	5	-	T	X	-	-	-	-	-	-	-		-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

1.3.2. Según el uso

Según el uso que se hace de un fichero, los podemos clasificar en diferentes tipos. Para poder organizar y diseñar un fichero, es imprescindible saber qué función ejercerá.

a) Ficheros permanentes. Contienen la información necesaria para el funcionamiento de una aplicación. Tienen una vida larga y normalmente no se pueden generar de manera inmediata a partir de otros ficheros. Se distinguen tres tipos:

- **Ficheros maestros o de situación**. Contienen información que refleja el estado actual de los datos. Se actualizan periódicamente para adaptarse a cada situación nueva. Los registros se modifican muy a menudo, pero la estructura no varía. La mayoría de las aplicaciones informáticas están orientadas a actualizar el fichero maestro o a obtener resultados a partir de este fichero.

Ejemplo de ficheros maestros

Un ejemplo de fichero maestro sería los datos de los clientes de una cadena de supermercados que tienen su tarjeta de fidelización.

- **Ficheros constantes**. Mantienen datos fijos para la aplicación. Estos datos son prácticamente inamovibles (tienen pocas modificaciones) y se utilizan como ficheros de consulta.

Ejemplo ficheros constantes

Un fichero que guarde la relación de los códigos de identificación de categorías de productos sería un ejemplo de fichero con datos constantes.

- **Ficheros históricos**. Contienen datos que fueron actuales en tiempos anteriores. Están constituidos por registros que reúnen cronológicamente las diferentes modificaciones que ha experimentado el fichero en el tiempo. Se conservan con el objetivo de poder reconstruir situaciones anteriores. Algunas veces pueden estar formados, simplemente, por los registros borrados del fichero maestro.

Ejemplo ficheros históricos

Un ejemplo de este tipo de ficheros sería la lista de todos los clientes que se han dado de baja de la cadena de supermercados que tenían su tarjeta de fidelización.

b) Ficheros temporales. Estos ficheros contienen información necesaria para un proceso específico dentro de una aplicación. Se generan a partir de los datos de los ficheros permanentes y se utilizan para obtener resultados o actualizar la información. Una vez hecha la operación, se eliminan. Se distinguen tres tipos:

- **Ficheros intermedios**. Generados a partir de los resultados de un programa y utilizados como la entrada en otro dentro de la misma tarea o proceso. Sólo se utilizan para pasar información de un proceso a otro.

Ejemplo ficheros intermedios

Un ejemplo de éstos sería un programa que necesitara hacer una operación de cálculo con los pedidos de todos los clientes de un área geográfica concreta. Se podría generar el fichero con estos datos a partir del fichero maestro, llevar a cabo el tratamiento sobre este fichero intermedio y eliminarlo posteriormente.

- **Ficheros de maniobras**. Este tipo de ficheros se utiliza para no perder la información generada por un proceso que no se puede conservar, por falta de espacio, en la memoria principal.

Ejemplo ficheros de maniobras

Un ejemplo de aplicación de un fichero de maniobras sería el de un proceso de una entidad financiera que quisiera calcular todos los intereses que ha cobrado a todos los clientes durante los últimos tres años.

- **Ficheros de resultados**. Se generan a partir de los resultados finales de un proceso, que se transferirán a un dispositivo de salida.

Ejemplo ficheros de resultados

Un ejemplo de este tipo sería un fichero de impresión con el listado de todos los clientes del supermercado que han hecho compras superiores a un importe determinado durante los dos meses anteriores, y que después se transferirá a la impresora.

1.4. Organización de ficheros

El ordenador debe tener acceso a los ficheros creados por los usuarios, tanto para leer la información almacenada como para grabarla o modificarla.

El **acceso a un fichero** está íntimamente relacionado con su **organización**. Esta organización indica cómo están dispuestos los registros en el **soporte material** para conseguir una utilización más eficiente.

Para saber más

Para tener más información sobre los registros en el soporte material, podéis consultar el apartado "Soportes físicos de información" en el módulo "Ordenadores y sistemas operativos".

Al crear un fichero, se especifica cuál será su organización, dado que determinará el tipo de acceso que podremos utilizar.

Hay **dos tipos de acceso** a un fichero:

- **Acceso secuencial**. Para acceder al registro *n*, primero debemos recurrir a los *n – 1* anteriores en el mismo orden en el que se escribieron, hasta encontrar el registro buscado.

- **Acceso relativo**. A partir de una clave (identificador), se puede acceder directamente al registro sin tener que recurrir a los anteriores.

Clave de un fichero

La clave (o identificador) de un fichero es un campo o un conjunto de campos que identifican, de manera única, cada uno de los registros del mismo. Un ejemplo de clave es el NIF, tal como se muestra en el ejemplo de estructura de un fichero que podéis ver al principio del apartado "Ficheros".

Los tipos de organización de ficheros son básicamente cuatro: **secuencial**, **secuencial encadenada**, **secuencial indexada** y **directa**. En este apartado, se describe cada uno de estos cuatro tipos de organización.

1.4.1. Organización secuencial

En este primer tipo de organización, los registros se almacenan de manera **continua**, uno a continuación del otro, sin espacio entre los mismos y sin ningún índice que indique sus direcciones.

Ésta es la única organización que se puede gestionar en un dispositivo no direccionable como, por ejemplo, las cintas magnéticas, lo cual no implica que no se pueda utilizar en soportes de acceso directo.

Esta organización se suele utilizar con ficheros en los que en cada proceso se debe acceder a la mayor parte de los registros. Tienen la **ventaja** de que aprovechan muy bien el espacio, son sencillos de utilizar y se pueden utilizar con dispositivos secuenciales.

Su principal **inconveniente** es la falta de flexibilidad y la lenta velocidad de acceso, por lo cual no se recomiendan para ser utilizados en procesos interactivos.

En este tipo de ficheros se pueden realizar las operaciones de añadir registros al final del fichero y de consultarlos. En cambio, para insertar, modificar o borrar registros se deben utilizar ficheros temporales.

Ejemplo de organización secuencial

Un ejemplo de aplicación de esta organización sería un fichero que recogiera y almacenara, en tiempo real, las condiciones ambientales (temperatura, humedad, etc.) a inter-

valos de diez segundos. Posteriormente, esta información se podría tratar si se quisiera generar alguna media.

1.4.2. Organización secuencial encadenada

Los registros de un fichero con esta organización almacenan, además de su información, un **puntero** con la dirección del registro siguiente, según el orden lógico del fichero. Desde este punto de vista lógico, el fichero se ordenará según el valor de una clave, si bien los registros se colocan en direcciones físicas totalmente arbitrarias. Los punteros garantizan la secuencia lógica del fichero.

Puntero de un registro

Un puntero (o apuntador) es una variable cuyo valor es la dirección física donde se almacena un dato. Se dice que la variable *p* de tipo puntero (que contiene la dirección donde se almacena el valor *v*) apunta a *v*. Esta marca interna que tiene un registro siempre apunta al registro lógico activo (es decir, al registro lógico que se procesa en la siguiente operación del fichero) y se incrementa automáticamente cada vez que se procesa un registro (se lee o se escribe).

Esta organización es adecuada en ficheros que utilizan procesos interactivos con una actualización frecuente, pero en los que cada operación afecta a pocos registros.

Presentan la **ventaja** de una gran flexibilidad, ya que se pueden realizar todo tipo de operaciones (añadir, consultar, insertar, modificar y borrar), pero en cambio tienen el **inconveniente**, igual que los ficheros con organización secuencial para acceder a un registro concreto, de que hay que seguir de manera secuencial, todos los registros según los punteros.

Ejemplo de organización secuencial encadenada

Un fichero de ejemplo con una organización secuencial encadenada sería la lista de los socios de una entidad deportiva de volumen medio en la que no haya que llevar a cabo muchas operaciones sobre estos datos.

1.4.3. Organización secuencial indexada

Un fichero con organización secuencial indexada está compuesto de dos zonas:

- La zona de **registros**, que contiene todos los registros ordenados según el valor de alguna clave (uno o más campos del registro que la identifican). Se puede considerar una estructura secuencial pura.

- La zona de **índices**, formada por un número de registros inferior al total de registros del fichero.

Los registros de la zona de índices tienen una estructura particular que no tiene nada que ver con los registros reales del fichero. Tienen el **campo clave** (que contiene algunos valores de la clave del fichero) y el **campo dirección** (que contiene la dirección de un registro del fichero).

La zona de registros se considera dividida en una serie de **tramos lógicos** o **segmentos**, cada uno formado por registros consecutivos. Para cada tramo en la zona de registros, hay un registro en la zona de índices que contendrá en su campo clave el valor de la clave del último registro del tramo y, en el campo dirección, la dirección del primer registro del tramo.

Ejemplo de fichero con organización secuencial indexada

Ejemplo de fichero con organización secuencial indexada

Zona de índices	
Clave	Dirección
34874644	1
45674635	4
47657462	7
59263527	10

Zona de registros				
Núm. reg.	NIF	Nombre	...	
1	32564736			Segmento I
2	34563546			
3	34874644			
4	44756575			Segmento II
5	45673653			
6	45674635			
7	45674651			Segmento III
8	46587689			
9	47657462			
10	56734521			Segmento IV
11	58374632			
12	59263527			

En la zona de índices, el **campo clave** corresponde al NIF, que es la clave del fichero en la zona de registros, y el **campo dirección** corresponde al número de registro del fichero en la zona de registros.

Una analogía de esta organización es el índice de un libro de lectura.

La **ventaja** de este tipo de organización es que resulta muy útil cuando hay que combinar consultas a registros concretos y el procesamiento secuencial de todo el fichero, ya que permite el acceso directo a los registros en determinadas aplicaciones y, secuencialmente, en otras.

Su principal **inconveniente** es la imposibilidad de introducir nuevos registros o actualizaciones en los registros que hay en el fichero sin tener que llevar a cabo una reorganización con las modificaciones efectuadas.

Una solución a este problema es reservar una zona de memoria complementaria, donde se almacenen los registros nuevos y que se suele denominar *zona de desbordamiento* o *ciclo de trabajo*.

Una mejora para esta estructura consiste en utilizar la organización denominada *secuencial indexada encadenada* (una mezcla de secuencial indexada y secuencial encadenada).

Ejemplo de organización secuencial indexada

Una aplicación de esta estructura organizativa sería la de un fichero que contuviera todos los datos de las reservas de productos de un almacén, consultado de manera muy frecuente, pero que se actualiza una vez cada semana.

1.4.4. Organización directa o aleatoria

En este tipo de organización no hay ninguna relación lógica entre los registros y su ubicación física.

Cada registro se sitúa en una dirección de memoria que se calcula para cada uno aplicando una fórmula o algoritmo matemático. Estos métodos toman el valor de un campo del registro, aplican una transformación y obtienen su dirección.

Cuando se detecta que la dirección asignada a un registro ya está ocupada por otro, se puede optar por dos alternativas:

- Buscar una nueva dirección libre, secuencialmente, en el mismo fichero hasta encontrar una **posición libre** donde almacenar los registros. Este proceso es lento y provoca una mala ocupación de la memoria (degrada el fichero), pues quedan espacios vacíos.

- Reservar una **zona de desbordamiento** en la que almacenar, de manera consecutiva, los sinónimos (registros que obtienen el mismo valor de la dirección, y por lo tanto, pueden ocupar una misma posición), a medida que éstos aparecen.

Nota

- **Colisión**: situación que se produce cuando se quiere poner un registro en una posición del fichero que ya está ocupada por otro registro.
- **Sinónimos**: registros que entran en colisión ya que, por transformación, obtienen un mismo valor de la posición o dirección física.
- **Zona de desbordamiento**: zona donde se almacenan los registros sinónimos que no se pueden colocar en la posición que les corresponde de la zona de registros por estar ocupada por otro registro. También se denomina área de excedentes, zona de saturación u *overflow*.

Los ficheros con esta organización tienen como principales **ventajas** la gran flexibilidad y la rapidez de consulta.

Entre los **inconvenientes**, hay que destacar el desperdicio del espacio (ya que se debe reservar todo el espacio necesario, aunque no haya datos) y la necesidad de utilizar soportes direccionables.

Ejemplo de organización directa o aleatoria

El fichero con los datos de los vehículos que tienen contratada una plaza en un aparcamiento privado de 2.000 plazas. Como no se puede guardar un registro para cada matrícula posible, habrá que "generar" la posición del registro en función, por ejemplo, de los números de la matrícula. Una posible fórmula (o algoritmo de conversión) es coger las cuatro cifras de la matrícula; si la primera cifra está entre 0 y 4, ésta se cambia por un 0 y si está entre 5 y 9, se cambia por un 1. Esta nueva combinación de cifras indicará la posición en la que se almacenará el registro en el fichero. Además, se reserva espacio para 200 registros adicionales, por ejemplo, en una zona de desbordamiento al final del registro.

En este ejemplo, una posible distribución sería la siguiente:

Núm. registro	Matrícula	Propietario	Núm. plaza	...
...				
0233	T-3233-BC			
...				
0323	L-3323-CD			

Núm. registro	Matrícula	Propietario	Núm. plaza	...
...				
0333	T-4333-BB			
...				
0367	G-3367-AQ			
...				
0421	T-3421-EF			
...				
0454	B-3454-ZY			
...				
0456	B-3456-SZ			
...				
0543	L-4543-AA			
...				
0565	L-4565-AB			
...				
0567	T-4567-AB			
...				
0576	B-4576-VU			
...				
0587	G-4587-AW			
...				
1458	B-5458-TY			
...				
1675	G-8675-DD			
1676	G-5676-ED			
...				
1786	T-9786-DX			
1787	B-6787-VB			
...				
Zona de ciclo de trabajo				
2000	T-3454-CC			

Núm. registro	Matrícula	Propietario	Núm. plaza	...
2001	L-9786-DD			
...				
2199	...			

Los dos registros almacenados en la zona de desbordamiento no se pueden ubicar en la zona normal de registros, dado que su posición natural está ocupada. Son sinónimos de otros registros que ya ocupan el espacio que les correspondería a ellos.

1.5. Limitaciones de los ficheros

Aunque en su momento los ficheros fueron un elemento importante para el almacenamiento de la información (y de hecho, todavía lo son como soporte de muchas aplicaciones informáticas), su uso y trabajo supone, sin embargo, una serie de inconvenientes, y los principales son:

- **Inconsistencia de la información**. Al tener información parcial o totalmente duplicada en varios ficheros, o al tener organizaciones diferentes, la actualización de los datos puede ser costosa. Un error en la actualización de esta información puede provocar inconsistencias en los datos del fichero.

Ejemplo de inconsistencia de la información

Por ejemplo, si queremos almacenar la información de los pacientes de un hospital con su médico asociado, y también los datos de los médicos, habrá que disponer de dos ficheros. Parte de la información de los médicos estará repetida en los dos ficheros.

- **Redundancia**. Este problema ocurre al tener datos que no aportan información y que se pueden calcular a partir de otros datos.

Ejemplo de redundancia

Muchas veces, con el objetivo de facilitar la rapidez del acceso a la información, por ejemplo, se pueden guardar datos en el fichero que se podrían calcular a partir de otros campos almacenados. Por ejemplo, el importe con IVA de las facturas del fichero de facturas de proveedores de la empresa.

- **Rigidez de busca**. Cada fichero tiene una organización determinada, según el tipo de acceso para el cual se ha definido.

Ejemplo rigidez de busca

Si se ha creado un fichero de tipo secuencial, por ejemplo, siempre se deberá acceder a él de esta manera, aunque su contenido pueda evolucionar y se pueda aplicar otro método de busca.

- **Dependencia de los programas**. Las relaciones entre los datos no se almacenan a su lado. El hecho de conocer y mantener estas relaciones es responsabilidad del programa que las gestiona. Cualquier cambio en la estructura del fichero representa modificar los programas que lo utilizan.

Ejemplo dependencia de los programas

Los programas informáticos desarrollados para gestionar la información han de conocer la estructura y la disposición de los campos de los ficheros que tratan. Añadir, por ejemplo, un nuevo campo al fichero de los pacientes del hospital significaría tener que modificar todos los programas que trabajan con este fichero para que puedan consultar la información de este nuevo campo.

Por este motivo, en la mayoría de las aplicaciones informáticas el uso de los ficheros se ha sustituido por el trabajo con bases de datos.

1.6. Resumen

Un fichero es un conjunto ordenado de datos que mantienen una relación lógica unos con otros y se almacenan en un soporte de información para que los puedan utilizar un programa o una aplicación.

En un fichero, se almacena información que hace referencia al mismo tema de una manera estructurada para poder trabajar con los datos de modo individual.

Los ficheros están compuestos por estructuras más simples denominadas *registros*.

Cada registro está formado por **campos** que contienen información en lo referente a un elemento o característica particular dentro del fichero.

Las principales operaciones que se pueden llevar a cabo sobre un fichero son su creación, la lectura, el mantenimiento y la actualización de la información (registros y campos), la ordenación de los registros, la búsqueda o consulta de datos, la fusión de dos o más ficheros en uno solo, la división de un fichero en uno o más, su destrucción o eliminación y la generación de informes o impresos a partir de la información contenida.

Los archivos se almacenan en los **dispositivos de memoria masiva**, también denominados **de memoria auxiliar**.

Estos dispositivos pueden ser de dos tipos: secuencial y de acceso directo.

- En los **soportes secuenciales**, por ejemplo una cinta, para acceder al registro n se deben leer los $n - 1$ registros anteriores.

- En cambio, en los **soportes de acceso directo**, por ejemplo un disco duro o un disquete, se puede acceder directamente a un registro físico sólo dando la dirección física.

Los ficheros se pueden clasificar según dos criterios principales:

- Según la **longitud de los registros** que contiene. En este caso, se distingue entre los ficheros de longitud fija, los de longitud variable, los delimitados y los indefinidos.

- Según el **uso que se hará de éstos**. Se pueden clasificar entre ficheros permanentes y temporales. En el primer tipo distinguimos entre ficheros

maestros o de situación, ficheros constantes o ficheros históricos; el segundo tipo pueden ser intermedios, de maniobras o de resultados.

Los tipos de organización de ficheros son básicamente cuatro: **secuencial**, **secuencial encadenada**, **secuencial indexada** y **directa**. Su organización condicionará qué tipos de operaciones pueden hacer con el fichero. En función de la aplicación del fichero, será más aconsejable una organización u otra.

Finalmente, debemos remarcar que vistas las limitaciones de los ficheros (inconsistencia, redundancia, rigidez en las búsquedas, dependencia de los programas...), se han sustituido, en la mayoría de las aplicaciones, por bases de datos.

2. Bases de datos y sistemas gestores de bases de datos

Las bases de datos surgieron para intentar resolver los problemas que tenían los ficheros.

Una base de datos es un sistema formado por un conjunto de datos organizados de manera que se evitan los datos redundantes, son independientes de los programas que los utilizan, almacenan los datos junto con las relaciones entre ellos y se puede acceder a éstos de diferentes maneras.

Dicho de otra manera, una **base de datos (BD)** es una colección de datos organizados de manera que se pueda acceder a sus contenidos, administrarlos y actualizarlos con facilidad.

El **sistema gestor de bases de datos (SGBD)** es el conjunto de software destinado a la creación, la gestión, el control y la manipulación de la información almacenada en una base de datos.

Acrónimos más comunes en bases de datos

En este módulo, se utilizan los siguientes acrónimos, de una manera amplia usados en catalán (y en castellano):

Concepto	Acrónimo	Comentario
Base de datos	**BD**	En inglés, *DB* por *data base*
Sistema gestor de bases de datos	**SGBD**	En inglés, *DBMS* por *data base management system*
Lenguaje de definición de datos	**DDL**	Del inglés, *data definition language*
Lenguaje de manipulación de datos	**DML**	Del inglés, *data management language*
Lenguaje de consulta estructurado	**SQL**	Del inglés, *structured query language*

Los SGBD disponen de un lenguaje de definición de datos **(DDL)** que permite definir el esquema de la base de datos, un lenguaje de manipulación de datos **(DML)** que facilita la gestión de la información almacenada y una interfaz de usuario que permite el acceso y trabajo en el sistema de una manera cómoda.

Cronológicamente, las bases de datos se han clasificado en tres grupos: **jerárquicas**, **en red** y **relacionales**.

Hacia los años setenta, aparecieron las **bases de datos** (BD) relacionales para obtener más flexibilidad en el tratamiento de los datos y son, hoy día, las más extendidas.

Ejemplo de gestor de BD

Algunos ejemplos de gestores de BD relacionales serían el programa Microsoft Access, el servidor de BD Oracle, Ms SQL Server, MySQL, etc.

Una **BD relacional** está formada por tablas, que son una estructura de filas (los registros de información) y columnas (los campos de información de los registros).

El diseño y la creación de una base de datos relacional consistirá en definir el **modelo relacional**, es decir, las tablas, sus campos y las relaciones entre tablas, principalmente.

Previamente al diseño de este modelo relacional, sin embargo, se aconseja hacer el **modelo conceptual**, que permite "plasmar" en un modelo de entidad-relación las diferentes informaciones que se quieren llegar a almacenar y gestionar con la futura base de datos relacional.

Un tipo de bases de datos especial son las documentales **(BDD)**, caracterizadas por el hecho de que cada registro se corresponde con un documento de cualquier tipo (publicación, documento gráfico o sonoro, etc.) o su referencia.

Las BDD se pueden clasificar en distintas categorías, según la información que contienen y la referencia al documento correspondiente: bases de datos de texto completo, archivos electrónicos de imagen y bases de datos referenciales. También se pueden clasificar por otras tipologías: según el modo de acceso a la información, la cobertura documental o según el modelo de tratamiento documental.

En los próximos apartados, se detallará cada uno de estos aspectos.

2.1. Conceptos básicos

En los siguientes subapartados se exponen algunos conceptos básicos sobre las bases de datos y los ficheros y las bases de datos.

2.1.1. Bases de datos

Una **base de datos** es una colección de información almacenada de manera organizada en formato electrónico.

Un **sistema gestor de bases** de datos es un programa que permite organizar su almacenamiento y facilitar su recuperación.

Las bases de datos ofrecen distintas **ventajas**:

- Facilitan el almacenamiento de grandes cantidades de información.
- Facilitan la recuperación rápida y flexible de información.
- Facilitan la organización y reorganización de la información.
- Facilitan la impresión y distribución de información de diferentes maneras.

Ejemplos de bases de datos

Las bases de datos son una colección de información de cualquier tipo como, por ejemplo, un directorio telefónico, un tarjetero de recetas, un catálogo de fichas bibliográficas, el inventario de productos y servicios de la compañía, los registros de calificaciones escolares de un estudiante, la lista de las asignaturas de un curso, la relación de los trabajadores de la organización, la lista de los clientes de la compañía, los pacientes de un centro asistencial, la lista de las lecturas de la temperatura de un termómetro, etc.

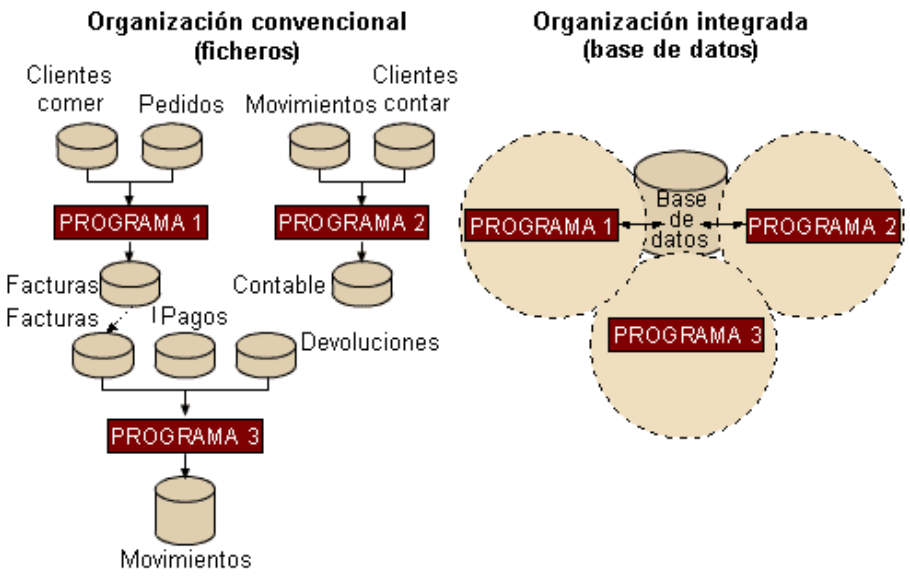
La información de algunos de estos ejemplos se puede gestionar en forma de simples ficheros, ya que contienen un único tipo de entidad (clientes, pacientes, temperaturas, calificaciones, etc.), mientras que otros integran información de diferentes tipos de entidad que sólo puede ser tratada como bases de datos.

Prácticamente, cualquier colección de información se puede convertir en una base de datos.

2.1.2. Ficheros y bases de datos

Las bases de datos aparecieron como una alternativa a los sistemas de ficheros para solucionar los inconvenientes y desventajas que presentaban.

La figura siguiente muestra la diferente forma de organizar los datos en un sistema de ficheros y en una base de datos. El primero precisa múltiples ficheros para organizar la misma información que la segunda integra en un único archivo.



Para saber más

Para tener más información sobre los inconvenientes y las desventajas que presentaban los ficheros, podéis consultar el apartado "Limitaciones de los ficheros" en este mismo módulo.

Actualmente, el uso de ficheros se limita básicamente a aplicaciones en las que no es necesario un acceso directo a la información o hay pocos tipos de datos relacionados entre diferentes ficheros, y como soporte para almacenar la información de programas (por ejemplo, procesadores de textos) o del sistema operativo.

Aun así, físicamente, en el ordenador, las bases de datos se almacenan y gestionan como ficheros por medio de los sistemas gestores de bases de datos (SGBD).

Aunque de manera muy simplificada, se pueden identificar las diferencias siguientes entre los ficheros tradicionales y las bases de datos:

- **Entidades tipo**. Los ficheros tienen registros de una sola entidad tipo (por ejemplo, pacientes), mientras que las BD tienen datos de distintas entidades tipo (pacientes, tratamientos, médicos, etc.).

  El concepto de entidad

  Tanto los ficheros como las BD almacenan información de *objetos* del mundo real, denominados "entidades", que tienen una función importante en las organizaciones. Se pueden distinguir dos niveles de abstracción de una entidad:

  - **Entidad instancia:** es un objeto real o abstracto con existencia independiente y características que lo diferencian de otros objetos, aunque sean del mismo tipo. Por ejemplo, un paciente concreto, un médico concreto, un alumno concreto, una enfermedad concreta o una asignatura concreta. En este módulo, también se usa el término **ocurrencia** para referirse a este concepto.

  - **Entidad tipo:** es un conjunto de entidades instancia que comparten las mismas características. Por ejemplo, la abstracción *paciente*, formada por todos los pacientes de un hospital, o la abstracción *alumno*, formada por todos los alumnos de una escuela. En este módulo, también se usa el término **entidad** (a secas) para referirse a este concepto.

- **Interrelaciones**. En el caso de los ficheros, el sistema no interrelaciona ficheros, mientras que en el caso de las BD el sistema tiene herramientas previstas que interrelacionan entidades.

- **Redundancia**. Mientras se crean ficheros a la medida de cada aplicación con todos los datos necesarios –aunque algunos sean redundantes con respecto a otros ficheros–, en las BD todas las aplicaciones trabajan con la misma BD y la integración de los datos es básica, de manera que se evita la redundancia.

- **Usuarios**. Los ficheros sirven para un solo usuario o una sola aplicación, mientras que las BD son compartidas por muchos usuarios de diferentes tipos.

2.2. Aplicaciones prácticas de BD

En la actualidad, la mayoría de los sistemas de información de una organización están basados en bases de datos.

Ejemplos de aplicaciones prácticas de BD

- Por ejemplo, toda la información gestionada por el departamento comercial (clientes, productos, ventas, etc.) estará en bases de datos.
- También lo estará toda la información tratada por el departamento de marketing y comunicación (campañas, actuaciones, etc.).
- Toda la información relativa a los empleados y gestionada por el Departamento de Recursos Humanos también estará en una o más bases de datos, así como toda la información utilizada por los departamentos de finanzas y compras.
- Todos los datos de un sistema de gestión de alumnos de un centro de formación, por ejemplo, también estarán en una base de datos.
- Otro ejemplo de BD sería toda la información sobre las películas de un videoclub, sus clientes y sus actuaciones (alquileres, devoluciones, etc.).
- Todas las bibliotecas basan su gestión sobre bases de datos.

Y, de esta manera, encontraríamos infinidad de ejemplos.

2.2.1. Características y objetivos de las BD

Tal y como hemos comentado en el apartado anterior, las bases de datos surgen para resolver los problemas que presentan los ficheros: inconsistencia de la información, redundancia, rigidez de busca y dependencia de los programas.

Una base de datos es un sistema formado por un conjunto de datos organizados de manera que se controla el almacenamiento de los que son redundantes. Los datos resultan independientes de los programas que los utilizan; se almacenan de manera conjunta los datos y las relaciones entre sí, por lo que se puede acceder a la información de distintas maneras.

Los requerimientos principales que debe cumplir un buen sistema de bases de datos (BD) son los siguientes:

- Distintos usuarios pueden acceder simultáneamente a la base de datos, cada uno a una información determinada.
- Se controla el acceso de los usuarios a los datos y se garantiza la confidencialidad y seguridad.
- Los datos se almacenan sin redundancia (excepto en algunos casos en los que interesa mejorar los accesos y el rendimiento).

- Permiten utilizar diferentes métodos de acceso y asegurar flexibilidad en las investigaciones.
- Disponen de mecanismos de recuperación de información.
- Se puede cambiar el apoyo físico de la base de datos sin que se resientan la base de datos ni los programas que la utilizan.
- La modificación del contenido y las relaciones entre los datos no afecta a los programas que las utilizan.
- Disponen de una interfaz de usuario que permite utilizar la base de datos de manera cómoda y flexible.
- Toda base de datos se gestiona por medio de un sistema de gestión de bases de datos (SGBD).

2.3. Sistemas gestores de BD

Se denomina *sistema gestor de base de datos* (DBMS o SGBD) al conjunto de software destinado a la creación, la gestión, el control y la manipulación de la información sobre una base de datos. Los SGBD tienen como finalidad registrar y mantener información.

Un SGBD debe permitir llevar a cabo las operaciones siguientes:

a) Definición del esquema de la base de datos. Una vez diseñado el esquema de la base de datos, lo hemos de describir mediante un conjunto de instrucciones. Esto se lleva a cabo mediante un lenguaje específico, denominado *lenguaje de descripción de datos* (DDL), y que, como cualquier lenguaje de alto nivel, necesitará un traductor para generar el código objeto a partir del código fuente.

Para saber más

Para tener más información sobre el esquema de la base de datos, podéis consultar el apartado "Definición conceptual de BD" en este mismo módulo.

b) Acceso a la información desde un lenguaje de alto nivel. Esto se realiza mediante un lenguaje específico, denominado *lenguaje de manipulación de datos* (DML). El DML se puede utilizar de dos maneras diferentes:

Para saber más

Para saber más sobre los lenguajes de definición (DDL) y de manipulación (DML) de datos, podéis consultar el apartado "Lenguajes de bases de datos relacionales", en este mismo módulo.

- Como lenguaje huésped, incluyendo sentencias DML dentro de un programa anfitrión escrito en un lenguaje de alto nivel (como, por ejemplo, Cobol, Pascal, Basic, C, etc.).
- De manera interactiva, por medio de programas que contengan exclusivamente sentencias propias del DML.

c) Acceso a la información en modo conversacional. El SGBD incorpora una interfaz de usuario por medio de la cual el usuario puede introducir sentencias DDL o DML directamente desde un terminal y obtener información interactiva.

d) Gestión de archivos. Función realizada por un módulo denominado *gestor de archivos* que se encarga de la comunicación con el sistema operativo.

e) Otras funciones: controles de usuarios, recuperación de la información después de errores del sistema, organización física de la base de datos, control de seguridad y privacidad de información o la gestión de accesos concurrentes.

Para saber más

Para tener más información sobre sistemas operativos, podéis consultar la unidad "Sistemas operativos" del módulo "Ordenadores y sistemas operativos".

En un sistema tradicional de ficheros, cada una de las aplicaciones deberá realizar la descripción de los registros, así como determinar la organización de ficheros, los tipos de acceso, etc.

Para saber más

Para tener más información sobre ficheros, podéis consultar la unidad "Ficheros" de este mismo módulo.

En un entorno de base de datos, estas especificaciones las realizan los SGBD y es el administrador de la base de datos el encargado de garantizar que se efectúen éstas y otras funciones sobre el sistema.

2.3.1. Características y objetivos de los SGBD

A continuación, se detallan algunas de estas características o requisitos de los sistemas gestores de bases de datos (SGBD):

- **Consultas no predefinidas y complejas**. El objetivo fundamental de los SGBD es permitir que se hagan consultas no predefinidas (*ad hoc*) y complejas. El usuario debe poder formular la consulta con un lenguaje sencillo, que se quede en el nivel lógico, y el sistema lo debe interpretar directamente. Los usuarios podrán hacer consultas de cualquier tipo y complejidad directamente al SGBD, que deberá responder inmediatamente sin que estén preestablecidas, es decir, sin que se deba escribir, compilar y ejecutar un programa específico para cada consulta. En cambio, en los ficheros tradicionales, cada vez que se quería realizar una consulta se debía escribir un programa a medida.

- **Flexibilidad e independencia**. La complejidad de las BD y la necesidad de ir adaptándolas a la evolución del sistema de información hacen que un objetivo básico de los SGBD sea dar flexibilidad a los cambios. Interesa obtener la máxima independencia posible entre los datos y los procesos y programas para que se puedan efectuar todo tipo de cambios tecnológicos y variaciones en la descripción de la BD, sin que se deban modificar los programas de aplicación ya escritos ni se haya de cambiar la manera de escribir las consultas directas. En el mundo de los ficheros ya había independencia física en un grado determinado, pero en el mundo de las BD suele ser mucho mayor.

- **Mínima redundancia**. Uno de los objetivos de los SGBD es facilitar la eliminación de la redundancia, ya que suponía un riesgo de inconsistencia o incoherencia de los datos. Conviene hacer que un dato figure una sola vez en la BD. En los ficheros tradicionales, cada aplicación utilizaba su fichero. Sin embargo, puesto que había mucha coincidencia de datos entre aplicaciones, se producía mucha redundancia entre los ficheros.

- **Concurrencia de usuarios**. Un objetivo fundamental de los SGBD es permitir que diferentes usuarios puedan acceder de manera concurrente a la misma BD. Con el objetivo de tratar los accesos concurrentes, los SGBD utilizan el concepto de transacción de BD, concepto de especial utilidad para todo aquello que hace referencia a la integridad de los datos. Se denomina *transacción de BD* o simplemente *transacción* a un conjunto de operaciones simples que se ejecutan como una unidad. Los SGBD han de conseguir que el conjunto de operaciones de una transacción nunca se ejecute parcialmente. O se ejecutan todas, o no se ejecuta ninguna.

- **Mecanismos de seguridad**. En el campo de los SGBD, el término *seguridad* se suele utilizar para hacer referencia a los temas relativos a la confidencialidad, las identificaciones, las autorizaciones, los derechos de acceso, etc. Los SGBD permiten definir autorizaciones o derechos de acceso en diferentes ámbitos (globalmente en toda la BD, en la entidad y en el atributo). Estos mecanismos de seguridad requieren que el usuario se pueda identificar. Se suelen utilizar códigos de usuario (y grupos de usuarios) acompañados de contraseñas (*passwords*), pero también se utilizan tarjetas magnéticas, identificación por reconocimiento de la voz, etc.

2.3.2. SGBD del mercado

Hoy día, hay muchos paquetes en el mercado que reducen costes y esfuerzos y hacen de las bases de datos una aplicación práctica para la mayoría de los usuarios de ordenador.

Estos paquetes son gestores de bases de datos sofisticados y poderosos que tratan de satisfacer las necesidades de dos grupos de usuarios:

- **Usuarios finales**, que son los que quieren utilizar la base de datos para almacenar, organizar y recuperar los datos.

Ejemplo de SGBD para usuarios finales

Entre los SGBD para el entorno Windows en nivel de usuario final, destaca Microsoft Access.

- **Programadores**, que la quieren utilizar para desarrollar aplicaciones específicas para los negocios u organizaciones.

Ejemplo de SGBD para programadores

Entre los SGBD servidores, pensados para departamentos de organizaciones y empresas, destacan los productos comerciales Informix, Microsoft SQL Server y el servidor Oracle. Los principales SGBD de software libre son MySQL y PostgreSQL.

Entre los paquetes líderes para Windows, en un ámbito de usuario destaca Microsoft Access.

Entre los sistemas gestores de bases de datos servidores, es decir, los pensados para departamentos de organizaciones y empresas, destacan los servidores Oracle, Microsoft SQL Server, Informix, MySQL, etc.

2.4. Tipos de bases de datos

Si atendemos al modelo lógico de datos, hay varios tipos de BD: **jerárquicas, en red, relacionales y orientados a objetos**, o combinaciones entre diferentes tipos. Por ejemplo, hoy en día Oracle es un SGBD para bases de datos relacionales y orientados a objetos.

Para saber más

Para saber más sobre modelos de datos y sus elementos constitutivos, podéis consultar el subapartado "Modelos de datos: conceptos básicos".

Modelo de datos

Un **modelo de datos** es, por una parte, la descripción del contenedor de datos (donde se guardará la información) y por la otra, el conjunto de métodos para almacenar y recuperar información de los contenedores.

Todo modelo de datos proporciona una serie de elementos (objetos, asociaciones, propiedades, operaciones, restricciones, etc.) que permiten, mediante la abstracción de una parte del mundo real, la obtención de un conjunto estructurado de datos y un conjunto de operaciones definidas sobre ellas.

A continuación, se detallan los tipos de BD atendiendo a su **modelo de datos**:

1) BD jerárquica

El primer modelo de BD en aparecer fue el **modelo jerárquico**, a principios de los años sesenta. Tal como indica su nombre, almacena la información en una estructura jerárquica.

Los datos se organizan en registros interrelacionados en forma similar a un árbol invertido. Un árbol se compone de un conjunto de registros (*nodos*) unidos por medio de asociaciones (*arcos*) padre-hijo que pueden ser de uno a uno o de uno a muchos. Es decir, un nodo "padre" puede tener uno o más nodos "hijos". El nodo que no tiene padres se denomina *raíz*, y los nodos que no tienen hijos, *hojas*. Estas asociaciones entre nodos se implementan, físicamente, utilizando punteros.

Para saber más

Podéis encontrar una pequeña descripción de cómo funciona el puntero de un registro en el subapartado "Organización secuencial encadenada".

El término *nodo*, cuando hablamos de bases de datos, tiene las siguientes acepciones:

- En una BD jerárquica o en red, **cada uno de los objetos** (entidades o registros) que conforman la estructura de datos (en árbol o en red, respectivamente).

- En una BD distribuida, **cada uno de los ordenadores** o lugares interconectados donde se emplazan los datos.

Ejemplos de SGBD jerárquicos

El IMS (*information management system*) de IBM, diseñado en 1966 y aparecido en 1968, fue líder de los SGBD jerárquicos durante los años setenta. A pesar de ser superado por otros modelos de datos, es el único que ha sobrevivido, pues desarrollos posteriores han permitido que soporte aplicaciones en Java, JDBC, XML y servicios web.

Otros SGBD de la época, actualmente en desuso, son System-2000 (de MRI, después comercializado por SAS Institute), Mark IV (de Control Data Corporation) y TDMS (de System Development Corporation).

El modelo jerárquico y las estructuras en árbol se utilizan en otros sistemas de almacenamiento, como el registro de Windows, los documentos XML o los repositorios LDAP (basados en el protocolo de acceso a servicios de directorio).

Su principal **inconveniente** es la incapacidad de representar la redundancia de datos (duplicidad de registros) y la poca flexibilidad.

2) BD en red

A finales de los años sesenta, aparecieron SGBD basados en un **modelo en red**. Como en el modelo jerárquico, hay registros e interrelaciones, pero un registro (*nodo*) ya no está limitado a ser "hijo" de un solo "padre". Un nodo puede tener diferentes padres, permitiendo así representar interrelaciones más complejas que con el modelo jerárquico.

Tiene la ventaja de que permite representar cualquier sistema y disponer de datos redundantes, pero su administración es muy compleja, por lo que su uso se ha limitado siempre a los programadores y no a los usuarios finales.

El comité CODASYL-DBTG propuso, en 1971, un estándar basado en este modelo, que fue adoptado por muchos productores de SGBD; sin embargo, encontró la oposición de IBM, la empresa dominante por entonces.

Ejemplos de SGBD basados en el modelo en red

El primero en aparecer fue el IDS (*integrated data store*), desarrollado por Charles W. Bachman a mitad de los años sesenta para la compañía General Electric. Versiones posteriores, como el IDS/2, fueron comercializadas por Honeywell-Bull.

El SGBD comercial más destacado fue el IDMS (*integrated database management system*) de Cullinet (actualmente, de Computer Associates), a pesar de que no cumple plenamente el estándar CODASYL.

Otros sistemas conocidos incluyen VAX-DBMS (de Digital), IMAGE (de Hewlet-Packard) y DMS_1100 (de Univac).

Tanto los sistemas del modelo en red como del jerárquico –que se puede considerar un caso particular del primero– presentaban lenguajes procedimentales que obligaban al programador a navegar, registro a registro, por la BD, no disponiendo de suficiente independencia físico-lógica, lo cual comportaba escasa flexibilidad.

3) BD relacional

La introducción de la teoría matemática del álgebra relacional, en el campo de las BD por parte de Edgar F. Codd (de IBM), en 1970, le llevó a proponer el **modelo relacional**.

Para saber más

Para tener más detalles sobre el modelo relacional, podéis consultar el subapartado siguiente, "Bases de datos relacionales".

Se basa en el concepto matemático de relación (que tiene una apariencia similar a una tabla de valores) y en la teoría de conjuntos. Una **tabla** es una estructura bidimensional donde se almacenan los datos como un conjunto de registros del mismo tipo, que se dividen horizontalmente en filas y verticalmente en columnas. Una **fila** representa un registro o grupo de valores de campos, y una **columna** contiene información en lo referente a un único campo o atributo de diferentes registros.

Este modelo supuso un paso importante para el desarrollo de los SGBD. Durante los años ochenta, se extendió el uso de los sistemas relacionales, la mayoría de los cuales utilizan como lenguaje nativo para la manipulación de los datos el SQL (estandarizado en 1986).

Para saber más

Para saber más sobre el SQL, podéis consultar el subapartado "Lenguajes de bases de datos relacionales".

Ejemplos de SGBDR

El prototipo System R (de IBM) fue el origen de los primeros SGBD relacionales como Ingres (creado en 1974 por la Universidad de Berkeley, y comercializado en 1980 por Ingres Inc. y después por Computer Associates).

Hay centenares de SGBD relacionales, tanto para PC (monousuario) como para sistemas *mainframe* (multiusuario), a pesar de que muchos no son completamente fieles al modelo relacional.

Los primeros SGBD comerciales multiusuario fueron Oracle (1979) y DB2 de IBM (1982). Otros sistemas multiusuario son Informix (después Informix Dynamic Server, de IBM), Sybase (SQL Server, después Adaptive Server Enterprise), Microsoft SQL Server, Allbase (de Hewlet-Packard), Interbase (de Borland) y los SGBD de código abierto MySQL, Firebird (basado en Interbase) y OpenIngres (de Computer Associates).

Los SGBD relacionales para microordenadores, inicialmente monousuario, después ofrecieron arquitectura cliente-servidor y se han ido adaptando al estándar ODBC (*open database connectivity*) de Microsoft, que permite la utilización de herramientas de tipo *front-end*. Son ejemplos de ello: Paradox y dBase (de Borland), FoxPro y R:base (de Microverso) y Access de Microsoft.

La mayoría de sistemas informáticos hoy en funcionamiento utilizan un SGBD relacional (o extensiones del mismo), aunque en algunas organizaciones todavía se usan los jerárquicos.

4) BD de objetos u orientada a objetos

Estructura la información en clases y subclases de objetos completos (estado y comportamiento) e incorpora los conceptos importantes de la programación orientada a objetos (encapsulación, herencia, polimorfismo, etc.). Todos los objetos pertenecen a una clase de objetos generalizados que comparten las mismas funciones. Por medio de la herencia, los objetos de una clase pueden adquirir las funciones de la clase.

Los siguientes conceptos del enfoque orientado a objetos son importantes para los sistemas de BD:

- **Objeto**: entidad discreta que tiene, básicamente, dos componentes:
  - Estado (valor) o estructura con sus atributos.
  - Comportamiento, que queda determinado por las **operaciones** (procedimientos, métodos y funciones) que se le pueden aplicar externamente.

- **Clase**: categoría generalizada que describe las características y el comportamiento de un grupo de objetos que pueden existir dentro de la misma.

- **Herencia**: transferencia de las propiedades y el comportamiento de una clase existente a una nueva subclase que se deriva de ella.

- **Método**: implementación de una operación que se aplica externamente a los objetos.

- **Mensaje**: código de programa que se envía al objeto para llamar o invocar un método.

- **Encapsulación**: acción de ocultar la estructura y el comportamiento interno de un objeto detallando con rigor su comportamiento externo, cosa que permite impedir conflictos y accesos incorrectos.

- **Polimorfismo**: capacidad para que varias clases de objetos tengan un comportamiento diferente ante una misma operación.

Correspondencia entre los términos de OO y los de programación tradicional

		OO	Programación
Términos	Objeto		Variable
	Clase		Tipo
	Método		Procedimiento, rutina (secuencia de instrucciones que ejecutan una única función)
	Mensaje		Llamamiento a rutina o procedimiento

Ejemplos de SGBDO

Destacan: GemStone (de ServiLogic), ObjectStore (de Object Design), Ardent (antes O2), Objectivity, Versant, Ontos y Poet.

Gestores de objetos

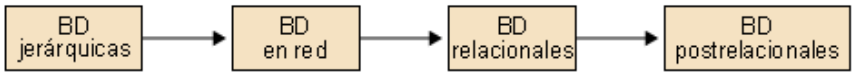
Sistemas con características similares a los SGBDO, pero con un modelo de datos limitado, que son extensiones de sistemas de ficheros o de gestión de memoria virtual. Por ejemplo, Mneme (gestor de memorias de traducción) y LOOM (sistema gestor de bases del conocimiento).

El modelo orientado al objeto surgió a finales de los años ochenta, por la falta de capacidad semántica del modelo relacional a la hora de atender determinados tipos de aplicaciones que requieren modelar objetos e interrelaciones complejos, almacenar información no estructurada, etc.

Esto ha hecho que los conceptos de orientación a objetos (OO) se hayan incorporado a los sistemas relacionales.

Tradicionalmente, las BD se han clasificado de modo cronológico en generaciones:

- 1.ª generación: **BD prerrelacionales**: jerárquicas y en red.
- 2.ª generación: **BD relacionales**.
- 3.ª generación: **BD postrelacionales**: orientadas al objeto, relacionales extendidas, distribuidas, documentales, etc.



Evolución cronológica de las bases de datos.

Las **BD postrelacionales** incluyen muchas tipologías y combinaciones de BD, algunas de las cuales no responden a un modelo de datos definido. A continuación, se exponen las más destacadas:

1) BD relacionales extendidas

Sistemas aparecidos, a partir de los años ochenta, fruto de la evolución del modelo relacional para superar sus limitaciones y obtener una mayor capacidad expresiva que represente más fielmente el mundo real, incorporando funcionalidades de otros modelos de datos como orientación a objetos, deductivas, con mecanismos de actividad, etc.

Se pueden distinguir diferentes enfoques:

a) BD relacional con frontal OO: añade un nivel o capa de OO superpuesta a un SGBD relacional preexistente.

b) BD objeto-relacional (SGBDOR): sigue el enfoque integrador de los dos modelos, incorporando nuevas capacidades:

- Nuevos tipos de datos (definidos por el usuario, multimedia, objetos grandes BLOB) que permiten gestionar aplicaciones más complejas con una gran riqueza de dominios.

- Operaciones que permiten gestionar el comportamiento de los tipos de datos.

- Mayor capacidad expresiva para la semántica de los datos (disparadores, procedimientos almacenados y funciones definidas por el usuario).

- Reusabilidad de librerías de clases ya existentes.

- Mejor capacidad consultiva (consultas anidadas, recursivas, almacenadas, prefabricadas, etc.).

En realidad, los actuales SGBD objeto-relacionales suelen incorporar capacidades deductivas, mecanismos de actividad, integración de documentos XML, etc.

c) BD activa: permite definir "reglas activas" que se activan, automáticamente, cuando suceden determinados "acontecimientos", que inician la ejecución de una "acción" (disparador o *trigger*) si se dan unas "condiciones" específicas.

Permite modelar mejor las reglas de funcionamiento interno de una organización: mantener automáticamente datos derivados, notificar determinadas situaciones, garantizar el cumplimiento de restricciones de integridad (como las reglas de negocio en aplicaciones organizativas complejas), etc.

d) BD deductiva: incluye mecanismos de inferencia (basados en "reglas deductivas" de la lógica matemática) que permiten generar información adicional a partir de los hechos almacenados en la BD. Se relaciona con la inteligencia artificial y las bases de conocimiento.

Permite la comprobación de hipótesis, el descubrimiento de conocimiento deductivo (nuevas relaciones entre los datos), la gestión de procesos gobernados por reglas, el modelado de la estructura y los procesos de la empresa, el establecimiento de perfiles de clientes en comercio electrónico, etc.

Los intentos de proporcionar un modelo de datos que represente más fielmente el mundo real han dado lugar a los **modelos de datos semánticos**.

Extensión del modelo relacional de Codd

A la hora de intentar enmendar algunas de las deficiencias de su modelo relacional, Codd presentó una versión extendida denominada RM/T (1979) y, más recientemente, RM/V2 (1990).

El *RM/T* aborda algunos aspectos similares al modelo entidad-relación (de Chen, 1976) pero de manera más cuidada. Por ejemplo, clasifica las entidades en tres categorías (núcleos, características y asociaciones), los aspectos estructurales y de integridad son más amplios y están definidos con más precisión, incluye operadores especiales adicionales, incorpora soporte para la dimensión tiempo y para distintas clases de agregación de datos.

Ejemplos de SGBDR extendidos

- **BD relacionales con frontal OO**: evoluciones de SGBDR preexistentes son Oracle 8, Informix 9, DB2/UBD (*universal database*, de IBM) y Sybase. En la década del 2000, aparecen sistemas de código abierto como OpenODB (de Hewlet-Packard, basado en el SGBDR Allbase/SQL) y PostgreSQL (basado en Ingres).

- **BD objeto-relacionales** (SGBDOR): en 1990, aparece UniSQL/X, el primer **SGBDO con SQL** partiendo de cero. A mediados de los años noventa, aparecen Illustra (después adquirido por Informix y presentado como Informix Universal Server) y Omniscience.

- **BD activas:** muchos SGBD relacionales actuales (Oracle, DB2, Sybase) cuentan con esta funcionalidad, que proporcionan en forma de disparadores. Starburst es un prototipo experimental.

- **BD deductivas:** a mediados de los años ochenta, aparece LDL (*logic data language*) y los sistemas experimentales Coral y Nail.

- A pesar de no ser extensiones relacionales, hay que mencionar las BDOO con extensión deductiva (denominadas BDDOO) creadas a finales de los años ochenta, como Validity (de Hervor) y Coral++.

2) BD temporal

Soporta la variable tiempo y otros conceptos temporales. Permite almacenar un historial de cambios y consultar el estado (actual o pasado) de la BD. En algunos casos, incluso, se almacena información futura prevista. La información temporal se puede referir a hechos puntuales (que se producen en un punto del tiempo) o de duración (en la que se considera que son verdaderos).

Aplicaciones de las BD temporales

- **Asistencia médica**, donde hay que guardar los historiales médicos de los pacientes.

- **Seguros**, donde se necesitan los historiales de reclamaciones y notificaciones de accidentes, así como información sobre los periodos de vigencia de las pólizas.

- **Reservas** (compañías aéreas, servicios de transporte, alquiler de coches, hostelería, espectáculos, etc.), donde hace falta información sobre fechas y periodos de validez de las reservas.

- **Recursos humanos**, donde hay que mantener los historiales laborales de los trabajadores de una empresa (referentes a los salarios, puestos de trabajo y proyectos en los que han trabajado).

- **Investigación científica**, donde hay que guardar los valores y periodos de tiempo en los que se miden los datos recogidos en los experimentos.

- **Gestión académica**, donde hay que incluir el semestre y/o el año de las calificaciones de las asignaturas en expedientes académicos de los alumnos y en información relativa a becas.

3) BD espacial

Proporciona conceptos que permiten seguir la pista de objetos que se encuentran en un espacio multidimensional. Facilita interpretar y especificar las características espaciales de los objetos mediante conceptos geométricos bidimensionales y tridimensionales, y gestionar las relaciones espaciales entre los objetos.

Los diferentes tipos de bases de datos espaciales son los siguientes:

- **BD cartográficas**, que almacenan datos de mapas. Incluyen conceptos geométricos bidimensionales (puntos, líneas, arcos, polígonos, etc.) y tienen en cuenta operaciones espaciales (cálculo de distancias) y operaciones booleanas (verificación de superposiciones). Se usan en gestión medioambiental, de emergencias, de conflictos bélicos.

- **BD meteorológicas**, que almacenan datos de temperaturas y otra información del tiempo atmosférico relacionada con puntos espaciales tridimensionales.

4) Sistema de información geográfica (SIG)

Permite almacenar, analizar y representar gráficamente información que describe diferentes propiedades geográficas y es gestionada por una BD, la cual contiene dos tipos de datos:

- Datos espaciales: de tipo físico (orografía, topografía y meteorología), de fronteras políticas y administrativas, de redes de transporte y comunicaciones.
- Datos no espaciales: demográficas, económicas, comerciales.

Los sistemas de información geográfica cubren disciplinas tan distintas como cartografía animada, mapeo interactivo, herramientas de ayuda a la decisión espacial, estándares de intercambio de datos geográficos, control de calidad y reingeniería, BD espacio-temporales, servicios de localización. Se pueden agrupar en tres categorías:

Ejemplos de SIG

Suelen ser sistemas que operan con SGBD relacionales o de objetos. ARC/Info, aparecido en 1981, integra la funcionalidad de un SGBD relacional. ARC/Storm (*arc store manager*) maneja BD distribuidas y se integra con SGBD relacionales como Oracle, Informix y Sybase.

- **Aplicaciones cartográficas**: estudios del terreno y el paisaje, análisis estructurales de tráfico y redes de transporte, prospección marina y petrolera, control y optimización de plantaciones agrícolas. Representan los datos mediante atributos espaciales (como la densidad de cultivos) y comportan funciones como la superposición de varias capas de mapas para combinar atributos que permitan la medida de distancias en un espacio tridimensional.

- **Aplicaciones para el modelado del terreno**: análisis del sol, gestión de recursos del agua, control de inundaciones, estudios de contaminación e impacto ambiental. Representan los datos mediante atributos espaciales (como las características del sol o la calidad del aire) y requieren la representación digital de elevaciones del terreno en puntos de muestreo que se interconectan por interpolación, dando lugar a un modelo de la superficie en 3D.

- **Aplicaciones de objetos geográficos**: análisis geográfico de mercados, análisis de pautas de voto, distribución y consumo de servicios públicos, sistemas de navegación de vehículos, gestión del catastro y localización de mobiliario urbano. Representan los objetos de interés (como distritos electorales, edificios, semáforos, farolas, centrales eléctricas) y necesitan funciones espaciales adicionales para manejar los datos referentes a un dominio físico concreto como redes viarias, de telecomunicaciones, de suministro, y de recursos hídricos; es decir, carreteras, cables de fibra óptica y de alta tensión, conductas de gas y de agua, ríos, etc.

5) BD multidimensional

Organiza los datos en estructuras matriciales de varias dimensiones y dispone de lenguajes especiales para realizar consultas complejas mediante el acceso multidimensional a los datos, de forma más adecuada y eficiente que una BD estructurada convencional como la relacional (orientada a transacciones y de naturaleza volátil).

Esta disposición proporciona una forma más natural de análisis de información, pues facilita el acceso y visualización de datos en cualquier combinación de dimensiones (las dimensiones pueden ser, por ejemplo, los productos, las zonas comerciales o los periodos impositivos de una empresa) mediante diferentes criterios y jerarquías (niveles de agrupamiento dentro de las dimensiones, subtotales previamente calculados). Representaciones jerárquicas como la exploración ascendente y la exploración descendente permiten agrupar (o resumir) los datos en categorías más generales (por ejemplo, ventas trimestrales para familias de producto) o disgregarlas en categorías más concretas.

Se implementa sobre un motor relacional con tablas, donde se almacena un volumen de datos muy elevado, a los que se aplican operaciones de cálculo intensivo (agregación, suma, media) y técnicas de indexación especiales que permiten consultas rápidas predefinidas.

Procesamiento de datos en línea: tecnologías OLAP y OLTP

A veces, las BD **multidimensionales** se denominan BD de **procesamiento analítico en línea** (OLAP, *on-line analytical processing*), ya que utilizan esta tecnología de análisis de datos que considera diferentes dimensiones y variables al mismo tiempo.

Los denominados **almacenes de datos** (*data warehouse*) utilizan toda la potencia de procesamiento analítico en línea de las BD multidimensionales.

Las BD convencionales (como las **relacionales**) soportan **procesamiento de transacciones en línea** (OLTP, *on-line transaction processing*), puesto que están optimizadas para procesar un gran volumen de consultas y transacciones concurrentes (inserciones, actualizaciones y supresiones de algunos registros), que pueden abarcar una pequeña parte de la BD, con una frecuencia elevada y una alta velocidad de respuesta.

Empresas del sector servicios (hoteles, bancos, líneas aéreas, compañías de seguros, empresas de servicio público y de comunicaciones) lo utilizan como sistema operacional de consultas y/o reservas durante 24 horas al día, 7 días a la semana.

La información se puede organizar en varias dimensiones:

a) Tabla relacional (una dimensión). Estructura los datos en registros (filas) formados por campos (columnas) de información relacionada de una única entidad u objeto del mundo real; es decir, de una sola dimensión (y una variable). A menudo, sin embargo, interesa gestionar más de una dimensión en una misma tabla.

Ejemplo de tabla relacional

La tabla siguiente muestra información de una única dimensión, la correspondiente a las zonas comerciales de una empresa (zona) con la variable *ventas*.

ZONA	Distribuidor	Dirección	Teléfono	*Ventas*
Zona1	Norte	Pl. Cataluña	93.345.89.78	103.600
Zona2	Sur	C/ Marina	93.123.98.78	250.204
Zona3	Centro	Av. Riera	93.333.88.77	303.667
Zona4	Oeste	C/ Mayor	93.123.55.65	238.423
...				

En este caso, puede interesar que haya otra dimensión (por ejemplo, producto) para almacenar las ventas de cada uno de los productos de la empresa, tal como muestra el ejemplo siguiente, de matriz bidimensional.

b) Matriz de datos bidimensional (dos dimensiones). Permite almacenar, conjuntamente, información de dos dimensiones; es decir, integra datos de dos entidades.

Ejemplo de matriz de datos bidimensional

Una hoja de cálculo con las ventas realizadas en cada zona, para cada producto, durante un periodo de tiempo determinado, ejemplifica una matriz de datos bidimensional. Las zonas se pueden mostrar en filas, y los productos, en columnas; los datos de las ventas de cada producto en cada zona están contenidos en las celdas. En este caso, hay dos dimensiones (zona y producto) y una variable (ventas). Esta es la forma multidimensional de almacenar los datos.

Ventas	**Producto**				
ZONA		**Producto1**	**Producto2**	**Producto3**	**...**
	Zona1	120.000	188.050	123.000	
	Zona2	111.000	229.005	128.976	
	Zona3	230.400	234.000	90.000	
	Zona4	162.050	222.000	89.640	
	...				

A continuación, se muestra un ejemplo de matriz bidimensional con **datos totalizados**.

Ventas	**Producto**					
Zona		**Producto1**	**Producto2**	**Producto3**	**...**	**Total**
	Zona1	120.000	188.050	123.000		**431.050**
	Zona2	111.000	229.005	128.976		**468.981**
	Zona3	230.400	234.000	90.000		**554.400**

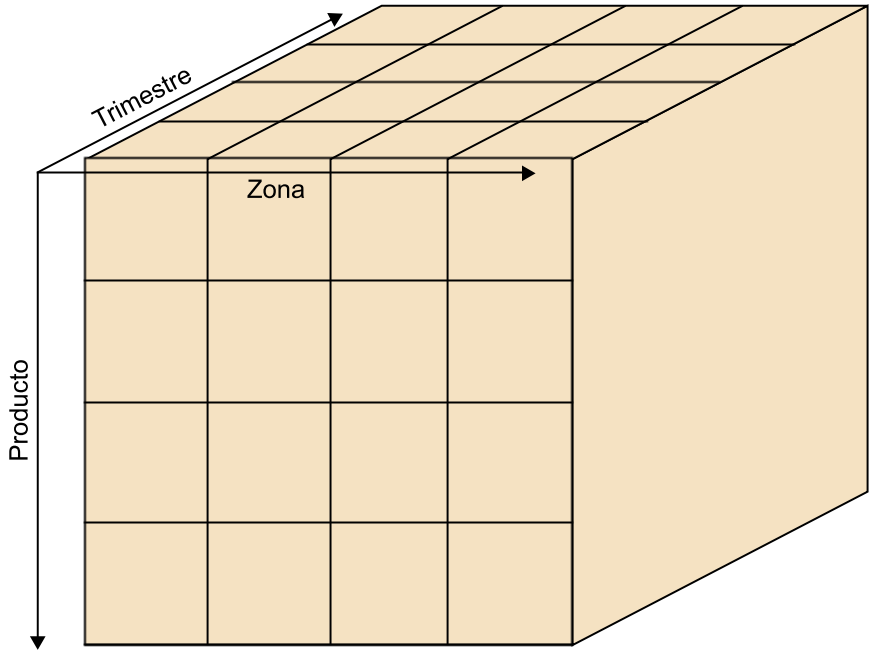
Ventas	Producto					
	Zona4	162.050	222.000	89.640		**473.690**
	...					
	Total	**623.450**	**873.055**	**431.616**		**1.928.121**

Así es como se mostrarían estos datos en un informe de un **almacén de datos** (*data warehouse*).

c) Cubo de datos (tres dimensiones). Corresponde a una matriz de datos tridimensional.

Ejemplo de cubo de datos

Si se añade una tercera dimensión temporal como los trimestres impositivos, se obtiene un cubo de datos que organiza las ventas de productos por trimestres impositivos y zonas comerciales. Cada celda contiene los datos de ventas de un producto concreto, un trimestre concreto y una zona concreta.



El cambio de orientación de un cubo de datos se consigue mediante la rotación (pivotación) sobre uno de los ejes. Por ejemplo, la rotación sobre el eje de zonas permitiría mostrar las ventas por zona en las filas, las ventas por trimestre en las columnas, y las ventas por producto en la tercera dimensión.

d) Hipercubo de datos (más de tres dimensiones). Corresponde a una matriz de datos n-dimensional ($n > 3$) a pesar de que no se puede representar gráficamente ni visualizar de forma sencilla.

6) BD paralela

SGBD que aprovecha el paralelismo de unidades de proceso o de memoria para el procesamiento paralelo de un número significativo de tareas de consulta y actualización, y de funciones de servicio asociadas (registro de transacciones, manejo de entrada/salida y almacenaje en búfer de datos) mediante la utilización concurrente de dos o más procesadores y/o dispositivos de almacenamiento. Eso proporciona mejoras significativas en el tiempo de transacción y de respuesta en BD de grandes dimensiones.

Procesamiento paralelo

Es la descomposición de las unidades de procesamiento (instrucciones individuales, secuencias de instrucciones o bloques de proceso) en partes que el sistema operativo o el SGBD asignan, para su ejecución simultánea, a procesadores diferentes que trabajan coordinadamente en un ordenador con multiprocesadores.

7) BD distribuida

Los datos están repartidos en diferentes ordenadores (nodos) de una red informática. La gran ventaja de este modelo es que permite unir información de diferentes localizaciones y acceder a ella sin tenerlo todo centralizado en un único ordenador. A pesar de la descentralización de los datos, soportada en una arquitectura básica de tipo cliente-servidor, tiene el inconveniente de que su gestión es complicada.

El desarrollo de entornos de Internet ha potenciado el acceso universal a BD distribuidas a la Web desde cualquier ordenador por medio de páginas web. Las **BD web** permiten el acceso a BD distribuidas y heterogéneas almacenadas en servidores web en archivos compartidos. Una capa de software intermedia (*middleware*) entre el usuario y el SGBD, denominada pasarela web o interfaz de pasarela común (CGI*, common gateway interface*) permite ejecutar programas externos para obtener la información que, una vez procesada, vuelve al servidor en formato HTML, que es enviada de nuevo al navegador web en el que puede ser visualizada.

Muchos SGBD actuales incorporan prestaciones para facilitar el acceso a BD por medio de Internet. Hay dos métodos principales:

- **Acceso mediante *scripts* CGI. Permite al servidor de la BD relacionarse con el servidor web mediante la especificación CGI.**

- **Acceso mediante JDBC** (*java data base connectivity*). Esta interfaz de programación de aplicaciones (API) permite a los programas en lenguaje Java el acceso a BD relacionales mediante la ejecución de consultas o sentencias SQL. Otra posibilidad es la utilización del estándar de Microsoft ODBC (*open database connectivity*).

Para saber más

Podéis ver una definición de ODBC cuando se habla de BD accesibles por Internet en el subapartado "Tipos de bases de datos".

Los SGBD que dan acceso a BD vía web proporcionan las siguientes **ventajas**:

- Permiten un acceso sencillo, flexible, homogéneo, eficiente, barato y seguro a los datos.

- Permiten el acceso a datos actualizados continuamente.

- Dan soporte a muy distintos tipos de datos difíciles y costosos de recoger por otros medios. Por ejemplo, datos multimedia como imagen, audio y vídeo.

- Evitan los problemas de compatibilidad de formatos y sistemas, facilitando el acceso desde prácticamente cualquier plataforma.

- Proporcionan herramientas de busca, indexación y relación que mejoran las capacidades y potencialidades de las BD.

8) BD XML

Sistema aparecido a principios del siglo XXI por la necesidad de almacenar y recuperar documentos XML (que son estructuras de datos semiestructurados). Hay dos enfoques de SGBD que soportan con eficiencia documentos XML:

- **Extensiones de BD para XML**. Desglosan un documento XML y lo integran en su correspondiente modelo de base de datos (relacional o de objetos).

- **SGBD XML nativos**. Respetan la estructura del documento, permiten hacer consultas sobre ésta y recuperan el documento tal como fue guardado originalmente.

Ejemplos de SGBD XML

a) Extensiones de SGBD para XML:

- SGBDR para XML: Microsoft SQLXML y Oracle XML DB son productos comerciales.
- SGBDOO para XML: Ozone/XML es de código abierto.

b) SGBD XML nativos: Tamino y X-Hive/DB son comerciales, mientras dbXML, eXist y Xindice son de código abierto.

9) BD multimedia

Ofrece características que permiten almacenar y consultar información multimedia que incluye imágenes (fotografías, dibujos, etc.), vídeo (videoclips, vídeos domésticos, películas, noticiarios o programas de televisión), audio (canciones, audiolibros, discursos o mensajes telefónicos) y documentos de texto (libros, artículos, etc.).

Para la localización de fuentes multimedia, se usan consultas de recuperación basada en contenido.

Ejemplo de recuperación basada en contenido

En una BD de vídeos, puede interesar localizar todos los vídeos que contengan una determinada persona o un tipo de actividad o acontecimiento como, por ejemplo, riadas, incendios forestales o los goles marcados por un jugador o un equipo.

Las BD multimedia abarcan disciplinas muy variadas: gestión de documentos y registros, educación y difusión de conocimientos, comunicación y entretenimiento, control de actividades en tiempo real. Se pueden clasificar en:

- **Aplicaciones de almacenaje** de gran cantidad de datos multimedia (imágenes de satélite, fotografías del espacio) mediante almacenes centrales organizados en niveles.

- **Aplicaciones de presentación** de vídeo o de audio en tiempo real, que se ven o se escuchan a medida que son enviados por el SGBD.

- **Aplicaciones de trabajo colaborativo** para analizar información multimedia en tiempo real. Se usan en campos como la telemedicina o la ingeniería.

Ejemplos de SGBD multimedia

La mayoría son SGBDO con soporte multimedia: Informix Dynamic Server, DB2 Universal Database UDB) de IBM, Oracle 8 o superior, Sybase. Otros sistemas permiten la recuperación de imágenes basada en el contenido: QBIC (*query by image content*) de IBM, Virage, Excalibur, etc.

10) BD documental

Almacena referencias y/o texto completo de documentos de propósito general (libros, artículos, cartas, contratos, etc.), normalmente sin formato predefinido, que pueden contener texto extenso y datos multimedia. Estos documentos se indexan identificando palabras clave significativas (que aparecen en el texto) y sus frecuencias relativas. Las consultas se realizan gracias a estas palabras clave o por medio de un *thesaurus* de estructura jerarquizada. A veces, se llama BD textual o sistema de búsqueda documental.

Según si incluyen o no el contenido completo de los documentos descritos, las BD documentales se clasifican en:

a) BD referencial. No contiene los documentos originales, sino información descriptiva y referencias que permiten localizarlos en otro servicio. También puede incluir enlaces para obtenerlos por medio de otro programa.

b) BD factual. Almacena los documentos fuente u originales de manera completa o contiene toda la información necesaria para dar respuesta a las necesidades del usuario.

- **BD de texto completo**. Constituida por los propios documentos en formato digital. También puede incorporar campos con información complementaria para facilitar la descripción y el acceso. Permite localizar términos presentes en el texto del documento. Se puede considerar un caso específico de BD multimedia donde sólo se almacenan y manipulan fuentes de texto.

- **Archivo electrónico de imágenes**. Constituido por referencias que tienen un enlace a la imagen del documento original. No permite localizar términos presentes en el texto original.

Ejemplos de sistemas de BD documentales

Algunos de los sistemas de BD documentales más usados son Knosys, Verity y Excalibur.

A modo de resumen, a continuación os presentamos una tabla con todos los tipos de bases de datos:

Para saber más

Esta tipología de BD se desarrolla en el apartado "Bases de datos documentales", al final de este módulo. También se estudia en otras asignaturas de la licenciatura.

Tipo de BD	Acrónimo	Estructura de datos	Estándar del modelo	
Jerárquicas	BDJ	**Árbol**		IMS, System 2000, SABRE
En red	BDX	**Red**	Codasyl-DBTG	IDS
Relacionales	BDR	**Tabla**	SQL	DB2, dBASE, Ingres, Oracle, Informix, Paradox, Access, Sybase, MySQL
De objetos	BDO/ BDOO	**Objeto**	ODMG	ObjectStore, Ardent (O2), Objectivity, Versant, Ontos
Objeto-relacionales	BDOR	Tabla, objeto		UniSQL/X, Illustra, OpenODB, PostgreSQL, Ominiscience
Activas	BDA	Tabla con *triggers*		Oracle, DB2 y Sybase (con *triggers*)
Deductivas	BDD*	Tabla con reglas lógicas		LDL, Validity
Temporales	BDT	Tabla u objeto con series de tiempo y versiones de *tuples* o de atributos	TSQL	Informix Universal Server con *datablades* de series de tiempo
Espaciales	BDE			
Información geográfica	SIG			ARC/Info, ARC/Storm
Multidimensionales	BDMD	Hipercubo		UniVerse (de IBM)
Paralelas	BDP			
Distribuidas	BDD*	Heterogénea	ODBC, JDBC	
Web	BD-Web	Heterogénea	ODBC, JDBC	
XML	BDXML	Documentos XML (semiestructurados)		Tamino, X-Hive/DB, dbXML, eXist, Ozone XML DB
Multimedia	BDMM	Objetos LOB, TDA		Sybase, Oracle, DB2, ODB II, CA-Jasmine, DS Informix, QBIC
Documentales	BDD*	Documentos extensos (textuales o no)		Knosys, Verity, Excalibur

El acrónimo BDD se utiliza para una u otra tipología según el entorno del que se trate. Los dos primeros tipos son obsoletos.

2.5. Bases de datos relacionales

Para saber más

Para tener más información sobre los diferentes tipos de bases de datos, podéis consultar el apartado "Tipos de bases de datos" de este mismo módulo.

Tal y como se ha comentado en los apartados anteriores, las bases de datos más extendidas actualmente son las relacionales, sobre todo con respecto al uso de particulares y departamentos de organizaciones.

Algunas entidades bancarias y hospitalarias todavía trabajan, sin embargo, en el ámbito corporativo con bases de datos prerrelacionales.

En este apartado se presentan los conceptos principales de una base de datos relacional y sus componentes: tablas, registros, campos, etc.

En una base de datos relacional, la información se almacena en tablas, compuestas por registros. Cada registro contiene distintos valores, almacenados en campos, que son de diferentes tipos de datos. Todos los registros se identifican con una clave. Las diferentes relaciones entre los registros de dos o más tablas se implementan por medio de claves foráneas.

2.5.1. Tablas

Una tabla es un conjunto de datos que agrupan todas las ocurrencias o elementos de un mismo tipo.

Para cada entidad de información diferente se utiliza una tabla distinta. Por ejemplo, todos los alumnos de la universidad, las asignaturas de la carrera o la lista de profesores.

2.5.2. Registros

Cada fila de la tabla es un registro de información y describe un elemento de este conjunto de informaciones.

Ejemplos de registros

Ejemplos de registros serían cada alumno de la universidad, cada una de las asignaturas o cada uno de los profesores, almacenados en las tablas de alumnos, asignaturas y profesores.

2.5.3. Campos

Cada valor o información de los registros posible es un campo.

Ejemplos de campos

Por ejemplo, en las tablas que hemos mencionado antes de alumnos y profesores de la universidad, el nombre, el apellido, el NIF, la dirección, el código postal, la población o el teléfono serían campos.

O, por ejemplo, en la tabla de asignaturas, el nombre de la asignatura, su código o los créditos (carga lectiva) serían campos.

2.5.4. Claves

La clave de una tabla son uno o más campos que identifican, de manera única, los registros de la tabla.

A continuación, se describen los diferentes **tipos de claves** que puede tener una tabla:

- **Claves candidatas** (*candidat*e key): son todas las posibles claves que permiten identificar, de manera única, a cada uno de los registros de una tabla.

- **Clave primaria** (*primary key*) o principal: es la clave escogida, de entre las claves candidatas de una tabla, para identificar de manera única sus registros. Se acostumbra a escoger la clave candidata formada por el menor número de campos, y se suele poner al principio de la tabla. También recibe el nombre de identificador.

- **Claves alternativas** (*alternative* key) o secundarias: son las claves candidatas que no son escogidas como clave primaria de una tabla.

- **Clave foránea** (*foreig*n key), ajena o externa: es el campo (o conjunto de campos) que puede formar parte o no de la clave primaria de una tabla (denominada tabla hija o referenciante) que, a la vez, es clave primaria o clave alternativa en otra tabla (llamada tabla maestra o referenciada) con la que se relaciona. Permite establecer relaciones en cascada entre tablas y puede no existir en una tabla. Su objetivo principal es mantener la integridad de la información y es parte importante en la normalización de la BD.

Además, una **clave simple** está formada por un único campo; una **clave compuesta**, por más de un campo.

Aunque los registros de una tabla se pueden identificar por más de una posible combinación de sus campos, es decir, pueden existir varias claves candidatas, sólo una de éstas es escogida como clave primaria. El resto quedan como claves alternativas.

Una tabla puede tener N claves candidatas, 1 clave primaria, N-1 claves alternativas y M claves foráneas, donde N es igual o mayor que 1 y M igual o mayor que 0. Dicho de otra manera, una tabla tiene, como mínimo, una clave candidata (que, en el peor de los casos, estará formada por todos los campos de la tabla y será escogida como clave primaria). La clave primaria, normalmente, es un pequeño subconjunto de los campos de la tabla. Además, una tabla puede tener ninguna o varias claves foráneas procedentes de diferentes tablas referenciadas.

Para saber más

El ejemplo de la relación uno a muchos, expuesto en el siguiente subapartado, "Relaciones", muestra la clave foránea CódigoDepartamento de la tabla Profesor necesaria para implementar la relación con la clave primaria de la tabla Departamento.

Ejemplos de claves

Siguiendo con los ejemplos anteriores, los alumnos se pueden identificar, de manera única, por su NIF o por el código interno en la universidad. Ambos campos serían claves candidatas. Una de ellas sería escogida como clave primaria y la otra quedaría como clave alternativa. Además, el nombre y apellidos no pueden ser clave, ya que no se puede garantizar que no haya dos alumnos que se llamen igual.

La clave primaria de una tabla permite relacionar los registros de ésta con los de otras tablas por medio de las claves foráneas, tal como se explica a continuación.

2.5.5. Relaciones y claves foráneas

Las relaciones entre los registros de las diferentes tablas se implementan mediante la vinculación de valores. Esta vinculación se establece entre la clave foránea de una tabla y la clave primaria de otra.

El **tipo de relación** entre dos tablas se indica mediante el número de registros de una tabla, que se pueden relacionar con un registro de la otra tabla y viceversa. Eso se puede expresar separando por el signo de dos puntos (":") estos valores. Es decir, *1:1*, *1: n* o *m: n*, según el caso, donde las variables *n* y *m* simbolizan *muchos*.

Los tipos de relación entre dos tablas entroncan con el concepto de cardinalidad de una relación entre entidades (modelo conceptual), que es perfectamente aplicable a las relaciones entre tablas en el modelo relacional.

Para saber más

En el subapartado "El modelo entidad-relación", del apartado "Diseño de bases de datos", se desarrolla y se dan ejemplos de este concepto aplicado a las relaciones entre entidades.

Entre dos tablas se pueden establecer tres **tipos de relación**:

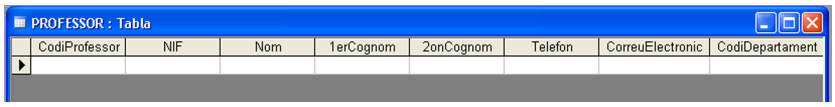
a) Uno a uno. A cada registro de una tabla le corresponde un solo registro de la otra. Se representa por *1:1*.

b) Uno a muchos. A cada registro de la primera tabla (la tabla maestra) le corresponde uno o más registros de la segunda (la tabla hija). Se representa por *1: n o n: 1* (la simbología utilizada por Microsoft Access es *1: ¥ y ¥ :1*).

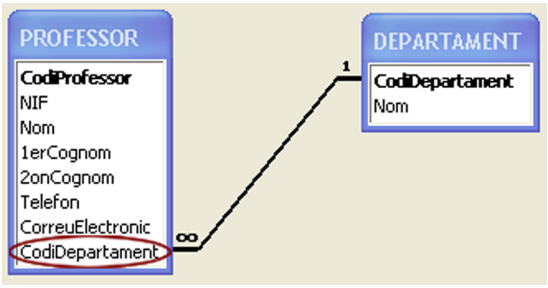
Ejemplo de relación uno a muchos (y de clave foránea)

Para implementar la relación que hay entre los profesores de un centro de estudios y el departamento al que están adscritos, se define, en la tabla Profesor, una clave foránea que se relaciona con la clave primaria de la tabla Departamento.

De esta manera, en el registro de cada profesor habrá un campo donde se almacenará el código del departamento en el que está adscrito, tal como muestra la tabla Profesor creada con el SGBD Microsoft Access de la siguiente figura.



Eso permite relacionar las tablas Profesor y Departamento, tal como se ve a continuación.

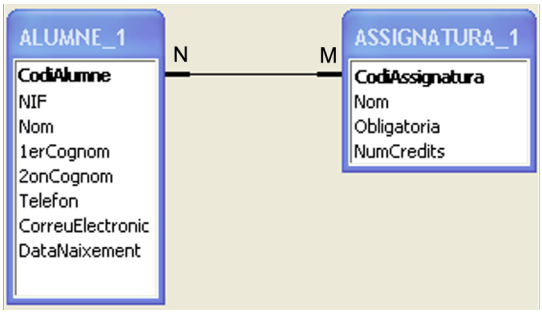


La figura muestra las claves primarias en negrita, y la relación uno a muchos (*1: ¥*) entre las dos tablas mediante una línea que une la clave primaria de la tabla maestra Departamento (en este extremo, se pone el número 1) y la clave foránea de la tabla hija Profesor (en este extremo, se pone el signo ¥).

c) Muchos en muchos. A cada registro de la primera tabla le pueden corresponder varios registros de la segunda y viceversa. Se representa por *m: n*.

Ejemplo de relación muchos a muchos

La relación que hay entre los alumnos de un centro de estudios y las asignaturas en las que se matriculan (inscriben) es del tipo muchos a muchos, ya que un alumno se puede inscribir en distintas asignaturas y, en cada asignatura, se pueden inscribir diferentes alumnos.



La figura muestra este tipo de relación *m:n*, no permitida en un SGBD relacional.

Los SGBD relacionales no permiten definir este tipo de relación entre dos tablas. Si se presenta el caso, hay que crear una tercera tabla, denominada **tabla intermedia**, que se relacione con las otras dos mediante relaciones *1: n* y *n: 1*.

Ejemplo de generación de una tabla intermedia

Para implementar la relación muchos a muchos del ejemplo anterior, hay que elaborar la tabla intermedia "Matriculación", que tendrá como mínimo dos campos: el identificador

de la asignatura y el identificador del alumno, siendo el conjunto de los dos campos su clave primaria.

Si, por ejemplo, un alumno se matricula más de una vez en una misma asignatura, ya no hay bastante con esta clave primaria. Habrá que añadir un tercer campo que ayude a identificar las distintas matriculaciones. Este tercer campo puede ser, por ejemplo, la fecha de matriculación. Ahora, la clave primaria estará formada por tres campos.

La figura creada con Microsoft Access muestra las relaciones *1: n* y *n: 1* existentes entre las tres tablas. La tabla intermedia aparece rodeada, y los campos que forman la clave primaria, en negrita.

2.5.6. Tipos de datos

Los campos de una tabla separan los tipos de información que contiene.

La persona que crea la tabla (diseñador de la base de datos) define los campos que tendrá esta tabla, es decir, qué informaciones se almacenarán para cada registro.

Los diferentes sistemas de gestión de bases de datos (SGBD) ofrecen una variedad de diferentes tipos de datos.

A continuación, definimos los principales.

- **Texto**: cadena o serie de caracteres alfanuméricos no muy larga. Puede ser información textual (el nombre de una persona o de una empresa, una dirección) que incluya letras, dígitos decimales, signos de puntuación, espacios y/o otros símbolos imprimibles. También puede ser un número (teléfono, código postal, etc.) que será tratado como una serie de dígitos y no como el valor numérico que representa y que, por lo tanto, no se podrá utilizar para hacer cálculos.

- **Numérico**: número que requiere cálculos matemáticos (exceptuando si está relacionado con un valor monetario; en tal caso, se utiliza el tipo "moneda").

- **Moneda**: valor numérico con un formato de presentación configurado para representar valores monetarios con decimales. Se utiliza para evitar el redondeo de decimales en los cálculos (A MS Access, tiene una precisión de hasta 4 decimales).

Formato de visualización

Es la forma en que se presentan (por pantalla o impresos) los valores de los diferentes tipos de datos. Los SGBD suelen proporcionar formatos de visualización predefinidos para los tipos: numérico, fecha/hora, moneda y sí/no. Además, se pueden definir formatos personalizados para todos los tipos de datos, excepto para el tipo binario.

- **Fecha y hora**: información temporal que representa fechas y horas.

- **Lógico**: cualquier tipo de dato en el que sólo haya dos valores posibles (sí o no, verdadero o falso, activado o desactivado, encendido o apagado, aprobado o suspenso, etc.). Se almacena uno de los dos valores posibles como 1 o 0, respectivamente.

- **Memo**: información textual alfanumérica más o menos extensa de longitud variable y de tipo memorando. Por ejemplo, descripciones, comentarios, notas, resúmenes, etc.

- **Binario**: objeto o archivo binario como una imagen, audio, vídeo, documento de texto con formato (creado con un procesador de texto), hoja de trabajo (creada con un gestor de hojas de cálculo) o cualquier otro tipo de archivo binario creado en otros programas que se vincula o se incrusta como dato en el campo de la BD mediante el protocolo OLE (*object linking and embedding*) de vinculación e incrustación de objetos.

- **Autonumérico**: valor numérico entero único que se genera, automáticamente, al agregar cada nuevo registro. Puede ser secuencial (con incrementos de una unidad; el primer registro tiene el valor 1, el segundo el valor 2, y así sucesivamente) o aleatorio. Una vez generada, no se puede actualizar y, si se elimina, el valor no se puede volver a usar. El tipo incremental, también denominado contador, es el más común y adecuado para utilizar como identificador único o clave primaria.

- **Hipervínculo**: direcciones URL o rutas de acceso según la convención universal de asignación de nombres, UNC).

El **tipo de dato** para utilizar en un campo de una tabla (valores del campo) se puede **decidir en función de**:

- El tipo de valores que se quieren permitir en el campo. Por ejemplo, no se puede almacenar texto en un campo definido para tipos de datos numérico.

- El tipo de operaciones que se quiere realizar con los valores del campo. Por ejemplo, se pueden sumar los valores de campos de tipo numérico y moneda, pero no los de campos de tipo texto o binario.

- La posibilidad de ordenar los valores del campo. Por ejemplo, los valores de un campo de tipo binario no se pueden ordenar.

- El sistema de ordenación de los valores del campo. Por ejemplo, en un campo de tipo texto, los números se ordenan como cadenas de caracteres (por ejemplo, 1, 10, 100, 2, 20, 200), no como valores numéricos (1, 2, 10, 20, 100, 200). Por un lado, para ordenar números como valores numéricos

conviene utilizar el tipo numérico o moneda. Por el otro, muchos formatos de fecha tampoco se ordenan correctamente si se utiliza un campo de tipo texto. Para garantizar un orden correcto, hay que usar un campo de tipo fecha/hora.

- El espacio de almacenaje que ocupan los valores del campo. Según las características de los datos textuales, se pueden usar tres tipos de datos de diferente tamaño:
  - Texto para almacenar texto corto: datos como nombres, direcciones y cualquier número que no necesite cálculos, como números de teléfono, números de producto o códigos postales. (En Access, hasta 255 caracteres).
  - Memo para almacenar texto extenso. (En Access, hasta 64.000 caracteres).
  - Binario para almacenar documentos de texto con formato más o menos largos.

En los campos con datos numéricos, se puede especificar la medida del campo en un rango de valores. Por ejemplo, en Access, para números enteros (sin decimales): byte: de 0 a 255, entero: entre -32.768 y 32.767, entero largo (predeterminado): entre -2.147.483.648 y 2.147.483.647.

2.5.7. Consultas

En este apartado, se comenta cómo los SGBD tratan la información almacenada en una BD, qué tipo de herramientas ofrecen para llevar a cabo esta tarea y las funcionalidades de las que disponen para consultar los datos.

Las consultas tienen funciones para seleccionar registros de las tablas y campos, o para hacer acciones de cálculo. Una prestación especial es la sentencia JOIN (de SQL), que permite combinar los registros de diferentes tablas, es decir, hacer consultas en las que intervienen varias tablas a la vez.

La sentencia JOIN

Es una instrucción de SQL que permite combinar (concatenar) registros (filas) de dos o más tablas en una BD relacional, si cumplen una condición de emparejamiento. Es un componente de la instrucción Select. Matemáticamente, se trata de una función de composición relacional, la operación fundamental en el álgebra relacional.

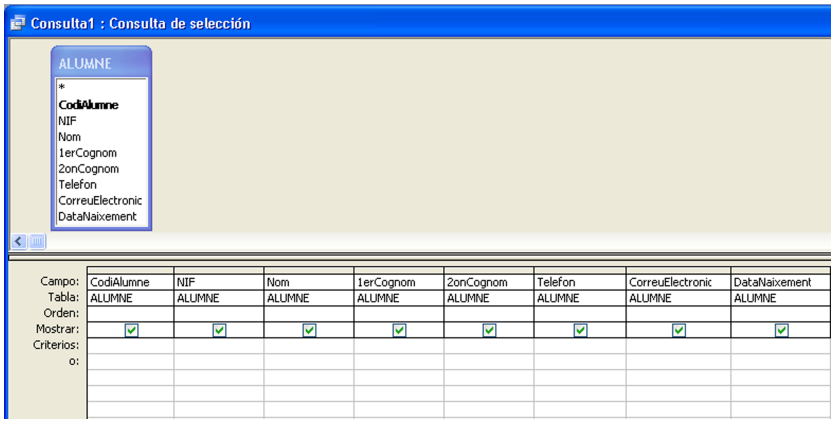
Ejemplos de diseño de consultas

A pesar de no ser objeto de esta asignatura, a continuación se ejemplifica el diseño de consultas (con Microsoft Access).

Se consultan los datos de los alumnos de un centro de estudios y las asignaturas en las que se matriculan.

1) Diseño de una consulta para obtener todos los datos de los alumnos del centro

En el diseño de una consulta en Access, hay que distinguir dos áreas diferenciadas. En el panel superior, se sitúan las tablas (con los campos) que intervienen en la consulta y se muestran las relaciones que hay entre ellas. En el panel inferior, se detallan los campos específicos que tienen que aparecer en la consulta y la tabla a la que pertenecen; además, otros aspectos (cómo se ordenarán los datos, si se tiene que mostrar o no la columna, los criterios de filtrado, etc.).



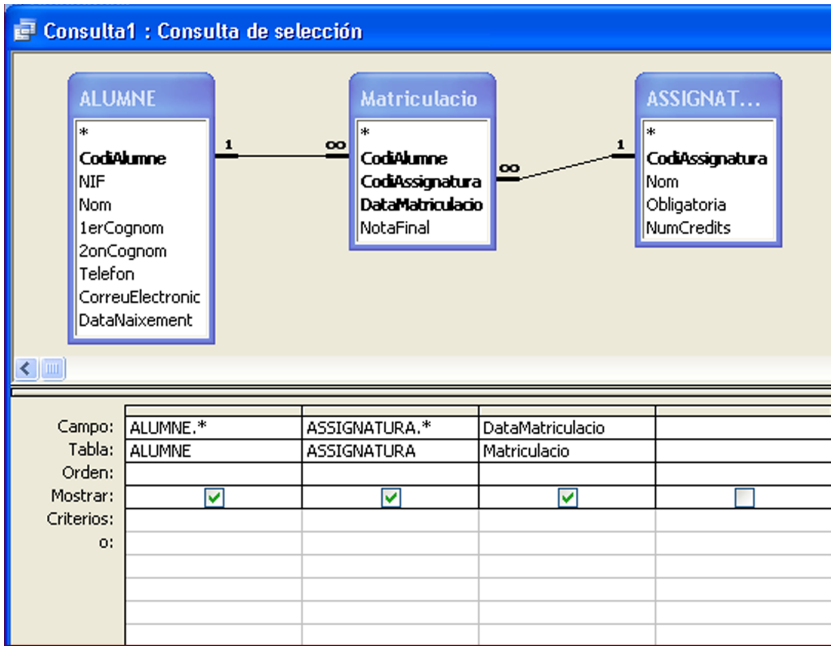
Para saber más

En el subapartado "Tablas intermedias", podéis ver el ejemplo de referencia de la BD de un centro de estudios donde se muestra cómo se relacionan las tablas Alumno, Asignatura y Matriculación.

2) Diseño de una consulta para obtener todos los datos de las matriculaciones del centro

En la fila de campos, en el panel inferior de esta consulta, aparecen los nombres de dos tablas acompañados de un asterisco. Eso quiere decir que se mostrarán todos sus campos.

Por lo tanto, el resultado de la consulta mostrará todos los datos de los alumnos y de las asignaturas a las que se han inscrito, además de la fecha de matriculación.



2.6. Lenguajes de bases de datos relacionales

Cualquier SGBD dispone de un lenguaje de BD que posee dos módulos para realizar funciones diferentes:

- El **lenguaje de definición de datos** (DDL, *data definition language*) especializado en la creación de la BD; es decir, en la definición del esquema de la BD.

- El **lenguaje de manipulación de datos** (DML, *data management language*) especializado en la utilización y la gestión de la BD; es decir, en el mantenimiento de la información y la realización de consultas.

Para comunicarse con el SGBD, el usuario, ya sea un programa de aplicación o un usuario final, se vale igual de este lenguaje de BD. Hay muchos lenguajes diferentes según el **tipo de usuario** para el que están pensados y lo que éstos tienen que poder expresar:

- Los **usuarios informáticos** muy expertos (programadores) que quieren escribir procesos complejos necesitarán lenguajes complejos.

- Los **usuarios finales esporádicos** u ocasionales que sólo hacen consultas necesitarán un lenguaje muy sencillo, aunque dé un rendimiento bajo en tiempo de respuesta.

- Los **usuarios finales cotidianos**, especializados o incluso dedicados, exclusivamente, a trabajar con la BD necesitarán lenguajes muy eficientes y compactos especializados en tipos concretos de tareas, aunque no sean fáciles de aprender.

El lenguaje de consulta estructurado **SQL** (*structured query language*) se considera el lenguaje estándar de BD relacionales. Por esto, se estudiará el DDL y el DML que utiliza el SQL.

Las características principales de SQL son la sencillez, el carácter estándar y ser un lenguaje declarativo o no procedimental; o sea, para que SQL realice una acción concreta no se especifica cómo lo tiene que hacer, sino qué es lo que quiere obtener.

El lenguaje SQL tiene **instrucciones** de tres tipos:

- Instrucciones de **definición de datos** (tipo DDL). Por ejemplo, Create Table para definir las tablas, sus campos y las restricciones.

- Instrucciones de **gestión de datos** (tipo DML). Por ejemplo, SELECT para hacer consultas e, INSERT, UPDATE y DELETE para el mantenimiento de los datos (insertar, actualizar y borrar registros, respectivamente).

- Instrucciones de **control del entorno**. Por ejemplo, COMMIT y ROLLBACK para delimitar transacciones.

A continuación, se exponen las instrucciones más importantes de definición y de manipulación que utiliza el SQL.

2.6.1. Lenguaje de definición de datos (DDL)

El lenguaje de definición de datos proporciona órdenes para definir, eliminar y modificar tablas, y también para crear índices y vistas.

Las órdenes o instrucciones SQL más importantes del DDL son:

- **CREATE TABLE**: crea una tabla con los campos y sus tipos de datos. Su sintaxis es la siguiente:
  CREATE TABLE n_tabla (n_camp1 t_camp1 ... n_campN t_campN)
  Donde n_tabla es el nombre de la tabla que se creará, n_camp1 es el nombre del primer campo, t_camp1 es el tipo de datos del camp1.
  Cada campo tiene asociado un tipo de datos, siendo los más comunes: INTEGER para números enteros; DECIMAL (x,y) para números reales, donde *x* representa la longitud total del campo e *y* el número de decimales; CHAR (n) para cadenas de caracteres de longitud *n* (entre 1 y 255); DATE para fechas, etc.

- **ALTER TABLE**: añade nuevos campos a una tabla existente, los modifica o los borra. No es una orden estándar. Su sintaxis es la siguiente:
  ALTER TABLE n_tabla ADD (n_camp t_camp) para añadir campos a la tabla.
  ALTER TABLE n_tabla MODIFY (n_camp t_camp) para modificar campos de la tabla.
  ALTER TABLE n_tabla DROP n_camp para borrar campos de la tabla.

- **DROP TABLE**: borra el esquema de la BD, es decir, destruye tanto los datos contenidos en la tabla como la estructura de la misma. Su sintaxis es la siguiente:
  DROP TABLE n_tabla.

2.6.2. Lenguaje de manipulación de datos (DML)

El lenguaje de manipulación de datos está basado en el álgebra relacional e incluye órdenes para insertar, suprimir y modificar registros (filas) de la BD.

Las órdenes SQL más importantes del DML son:

- **INSERT**: inserta nuevos registros (filas) en una tabla. Su sintaxis es la siguiente:

INSERT INTO n_tabla VALUES (valor_camp1 ... valor_campN).

- **UPDATE**: modifica valores de determinados campos (columnas). Su sintaxis es la siguiente:
  UPDATE n_tabla SET camp=valor_camp.
  UPDATE n_tabla SET camp=valor_camp (WHERE condición)
  Si se especifica una condición, únicamente se actualizan los valores de los campos de los registros que cumplen la condición.

- **DELETE**: borra registros (filas) de la tabla especificada. Su sintaxis es la siguiente:
  DELETE FROM n_tabla
  Si no se especifica ninguna condición, se eliminarán todos los registros de la tabla. Es decir, deja la tabla vacía, como si se acabara de crear, pero no destruye la estructura. No se tiene que confundir con orden DROP que, además de borrar la información, también elimina la estructura.
  DELETE FROM n_tabla (WHERE condición)
  Si se especifica una condición, únicamente se borran los registros que satisfacen la mencionada condición. Esta es su forma más usual.

- **SELECT**: es el orden principal de SQL y se utiliza para consultar tablas. Es muy flexible y admite muchas variaciones. Selecciona un conjunto de registros (filas) de una o más tablas que cumplan una condición. Si no se especifica la condición, se seleccionan todos los registros de la tabla. También permite crear un filtro seleccionando sólo algunos campos de la tabla. Su sintaxis es la siguiente:
  SELECT FROM n_tabla (WHERE condición) (ORDER BY campX) para seleccionar todos los campos de la tabla. Esta es su forma más usual.
  SELECT camp1, campN FROM n_tabla (WHERE condición) (ORDER BY campX) para consultar únicamente un conjunto de campos.
  SELECT camp1, campN FROM n_tabla1, n_tablaM (WHERE condición) (ORDER BY campX) para consultar campos de más de una tabla.

2.6.3. Herramientas de interfaz

Aunque casi todos los SGBD del mercado tienen el **SQL como lenguaje nativo**, ofrecen otras posibilidades:

- **Lenguajes de cuarta generación** (4GL, de *4th generation languages*) de muy alto nivel, que suelen combinar elementos procedimentales con elementos declarativos. Pretenden hacer muy fácil no tan sólo el tratamiento de la BD, sino también la definición de menús, pantallas y diálogos.

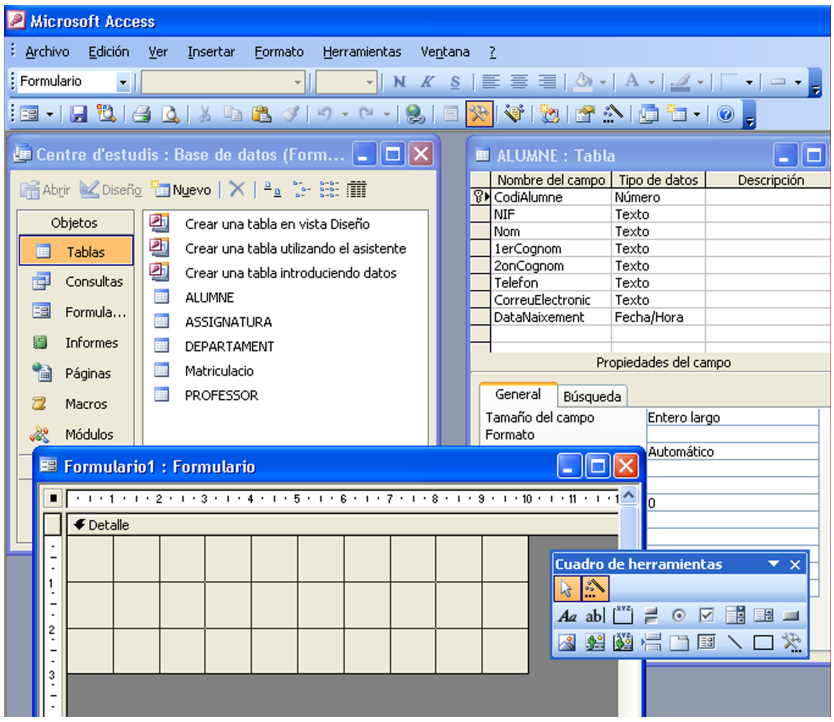
- **Interfaces visuales** muy fáciles de utilizar, que permiten utilizar las BD siguiendo el estilo de cuadros de diálogo con ventanas, iconos y ratón, típico de las aplicaciones en entorno Windows. No sólo son útiles a los

usuarios no informáticos, sino que facilitan mucho el trabajo a los usuarios informáticos: permiten consultar y actualizar la BD, y también definirla y actualizar la definición con mucha facilidad y claridad. Aparecieron a finales de los años ochenta y, desde entonces, han proliferado.

Ejemplo de interfaz visual

Un claro ejemplo de herramienta con interfaz visual es Microsoft Access, que permite crear los diferentes elementos de una BD (que aquí se denominan objetos: tablas, consultas, formularios, informes...) para gestionar y extraer la información mediante ventanas (*windows*) sin tener que utilizar, si no se quiere, el lenguaje SQL.

La siguiente imagen muestra algunos elementos de la interfaz de usuario de este SGBD (el panel de de exploración, una tabla y un formulario).



2.6.4. Programación de bases de datos

Si se quiere escribir un programa de aplicación que trabaje con la BD, seguramente se utilizará el mismo lenguaje de programación utilizado para desarrollar la aplicación informática (Pascal, Cobol, Basic, C, etc.). Sin embargo, generalmente, estos lenguajes no tienen instrucciones para las BD. Entonces, se puede recorrer en una de las siguientes **dos opciones**:

- **Llamamientos a funciones**: en el mercado, hay librerías de funciones especializadas en BD; por ejemplo, las librerías ODBC. Sólo hay que incluir llamamientos a las funciones deseadas dentro del programa escrito con el lenguaje de programación habitual. Las funciones se encargarán de enviar las instrucciones (generalmente, en SQL) en tiempo de ejecución en el SGBD.

- **Lenguaje hospedado**: otra posibilidad consiste en incluir directamente las instrucciones del lenguaje de BD en el programa. Pero eso exige utilizar

un precompilador especializado que acepte, en el lenguaje de programación habitual, las instrucciones del lenguaje de BD. Entonces, se dice que este lenguaje (casi siempre el SQL) es el lenguaje hospedado o incorporado (*embedded*), y el lenguaje de programación (Pascal, C, Cobol, etc.), el lenguaje anfitrión (*host*).

2.7. Resumen

Una **base de datos** es un conjunto de datos organizado que permite acceder, administrar y actualizar sus contenidos con facilidad.

Las BD evitan los datos redundantes, garantizan la independencia de los datos con respecto a los programas que las utilizan, almacenan los datos junto con sus relaciones y se puede acceder a ellos de diferentes maneras. Es decir, resuelven la mayoría de problemas que presentan los ficheros.

Las principales **características** que tiene que cumplir un sistema de BD son:

- Es accesible simultáneamente para diferentes usuarios, cada uno a una determinada información.
- Se controla el acceso de diferentes usuarios a los datos y se garantiza la confidencialidad y la seguridad.
- Los datos se pueden almacenar sin redundancia.
- Permite utilizar diferentes métodos de acceso y asegurar flexibilidad en las buscas.
- Dispone de mecanismos de recuperación de información en caso de que falle el hardware.
- El soporte físico donde se almacenan los datos se puede cambiar sin que se resientan éstos ni los programas que los utilizan.
- La modificación de su contenido y de las relaciones entre los datos no afecta a los programas que las utilizan.
- Dispone de una interfaz de usuario que permite utilizarla de manera cómoda y flexible.

El **SGBD** (sistema gestor de bases de datos) es el conjunto de software destinado a la creación, gestión, control y manipulación de la información almacenada en una BD.

Entre las principales **funciones de un SGBD**, destacan:

- Definición del esquema de la BD, por medio de un **lenguaje de definición de datos** (DDL).
- Acceso a la información desde un lenguaje de alto nivel denominado **lenguaje de manipulación de datos** (DML).
- Acceso a la información en modo conversacional por medio de una **interfaz de usuario**.
- Interacción con el sistema operativo para trabajar con los ficheros de datos que gestiona, mediante un módulo denominado **gestor de datos**.
- Los **objetivos de un SGBD** son los siguientes:
- Permitir realizar **consultas no predefinidas y complejas**.
- Dar **flexibilidad** a los cambios y obtener **independencia** entre los datos y los programas.
- Facilitar la **eliminación de la redundancia** para evitar inconsistencia e incoherencia de datos.
- Permitir el **acceso concurrente de los usuarios** a la misma BD.
- Permitir definir **autorizaciones** y **derechos de acceso** a diferentes niveles para garantizar la **confidencialidad** y la **seguridad** de los datos.

Históricamente, se han diferenciado tres tipos de BD:

- **BD jerárquicas**. Sistemas, aparecidos en los años sesenta, donde la información se representaba en forma de árbol. Su principal inconveniente era que no todas las BD se adaptaban a esta estructura.
- **BD en red**. Aparecieron para resolver el problema de las anteriores. Permitían cualquier tipo de relación y, por lo tanto, representar cualquier conjunto deinformación. Su principal inconveniente era su falta de flexibilidad.

- **BD relacionales**. Aparecieron, en los años setenta, para obtener una mayor flexibilidad en el tratamiento de los datos y son las más utilizadas desde los años ochenta.

Otros tipos de BD son los siguientes:

- **BD de objetos**. Almacenan la información en clases y subclases de objetos completos (estado y comportamiento) e incorporan conceptos de la programación orientada a objetos.

- **BD relacionales extendidas**. Son evoluciones del modelo relacional para obtener una mayor capacidad expresiva, incorporando funcionalidades de otros modelos de datos. Se distinguen:
  - **BD relacionales con frontal OO**: añaden una capa de OO sobre un SGBD relacional.

  - **BD objeto-relacionales**: integran los dos modelos, incorporando nuevos tipos de datos, operaciones para gestionar el comportamiento, nuevas capacidades consultiva y expresiva.

  - **BD activas**: permiten definir reglas que se activan cuando suceden determinados acontecimientos, que inician la ejecución de una acción si se dan unas condiciones.

  - **BD deductivas**: incluyen mecanismos de inferencia basados en reglas deductivas que permiten generar información adicional a partir de los hechos almacenados en la BD.

- **BD temporales**. Soportan la variable tiempo y otros conceptos temporales. Permiten almacenar un historial de cambios y consultar el estado (actual o pasado) de la BD.La información temporal se puede referir a hechos puntuales o a hechos de duración.

- **BD espaciales**. Proporcionan conceptos para interpretar y especificar las características espaciales de objetos que se encuentran en un espacio multidimensional. Pueden ser cartográficas (incluyen conceptos geométricos bidimensionales) y meteorológicas (incluyen información de puntos espaciales tridimensionales).

- **Sistemas de información geográfica**. Permiten almacenar, analizar y representar gráficamente información, que describe propiedades geográficas, gestionada por una BD que contiene datos espaciales (físicas, políticas, administrativas) y datos no espaciales (demográficos, económicos, comerciales).

- **BD multidimensionales**. Organizan los datos en estructuras matriciales de varias dimensiones (por ejemplo, productos, zonas comerciales, perio-

dos impositivos) y disponen de lenguajes especiales para realizar consultas complejas. Facilitan la visualización de datos, en cualquier combinación de dimensiones, mediante diferentes criterios, jerarquías (niveles de agrupamiento en categorías más generales o de disgregación en categorías más concretas) y operaciones de cálculo intensivo (agregación, suma, media).

- **BD paralelas**. Aprovechan el paralelismo de unidades de proceso para el procesamiento paralelo de gran número de tareas de consulta y actualización mediante la utilización concurrente de dos o más procesadores y/o dispositivos de almacenamiento. Eso mejora significativamente el tiempo de transacción y de respuesta en BD de grandes dimensiones.

- **BD distribuidas**. Reparten los datos entre diferentes ordenadores conectados en red y se basan en una arquitectura cliente-servidor. Tienen la ventaja que descentralizan la información y la hacen más disponible localmente, pero el inconveniente de que la gestión es más complicada.

- **BD accesibles por Internet**. Permiten el acceso flexible, homogéneo, eficiente y seguro, desde cualquier plataforma, en BD distribuidas y heterogéneas (que contienen tipo de datos muy distintos como, por ejemplo, datos multimedia) actualizadas permanentemente y almacenadas en servidores web, evitando problemas de compatibilidad de formatos y sistemas.

- **BD XML**. Almacenan documentos XML (que son estructuras semiestructuradas) con la eficiencia de las BD convencionales. Los SGBD XML pueden ser extensiones del modelo relacional o de objetos (que desglosan el documento XML) o nativos (que respetan la estructura del documento y lo recuperan tal como fue guardado originalmente).

- **BD multimedia.** Ofrece características que permiten almacenar y consultar información multimedia que incluye imágenes, vídeo, audio y documentos de texto. Su aplicación abarca disciplinas como la gestión documental, la difusión de conocimientos, la comunicación, el entretenimiento, el trabajo colaborativo y el control de actividades en tiempo real.

- **BD documentales o textuales**. Almacenan las referencias y/o el texto completo de documentos de propósito general que pueden contener texto extenso y datos multimedia. Se consultan por medio de palabras clave o de términos de un *thesaurus* de estructura jerarquizada.

Se pueden clasificar en BD referenciales (que contienen referencias para poder localizar los documentos originales), BD de texto completo (que almacenan el contenido completo de los documentos originales) o archivos electrónicos de imágenes (constituidos por referencias que tienen un enlace a la imagen del documento original).

Una **BD relacional** está formada por **tablas, que son estructuras en filas** (los registros de información) y **columnas** (los campos de información de los registros). Cada tabla tiene una **clave**, que es un campo o conjunto de campos (columnas) que identifica de manera única cada registro (fila), y permite relacionar la tabla con otras tablas de la BD.

Una **tabla** de una BD relacional cumple las siguientes **condiciones**:

- Todas sus filas son registros del mismo tipo. Para almacenar registros de diferentes tipos se utilizan diferentes tablas.
- Cada columna se identifica por un nombre de campo.
- No acepta nombres de campos (columnas) repetidos.
- No acepta registros (filas) duplicados.
- El orden de los registros es indiferente.
- Su contenido es independiente del soporte de almacenaje físico de los datos.
- Se relaciona con otras tablas haciendo coincidir valores iguales de los registros de ambas, mediante claves.

Ejemplos destacados de SGBD relacionales son: Microsoft Access, el servidor de BD Oracle, Microsoft SQL Server o MySQL.

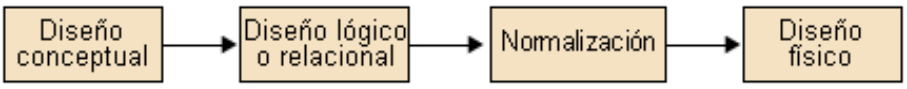
3. Diseño de bases de datos

Un error muy frecuente a la hora de diseñar una base de datos relacional es generar el modelo relacional (tablas, campos, etc.) directamente a partir de la identificación de las necesidades que tiene el usuario u organización, utilizando el sistema gestor de bases de datos.

Este método puede ser válido cuando se trata de crear bases de datos muy sencillas y con poca complejidad (pocas tablas y con pocas relaciones).

En cambio, cuando la complejidad aumenta, se debe recurrir a unas metodologías que garanticen la definición correcta de nuestro sistema de información y a unas estructuras que también faciliten que no haya errores y que simplifiquen el trabajo posterior, el tratamiento de los datos y la evolución del sistema.

La metodología o las etapas recomendadas son las siguientes:



a) Diseño conceptual

Durante el diseño conceptual, se construye un esquema de la información que se utiliza en la organización (empresa, asociación, departamento, etc.) independientemente de cualquier consideración física. Es decir, independientemente de qué modelo de bases de datos se utilizará posteriormente y en qué sistema gestor de base de datos se implementará.

El esquema resultante lo denominamos **esquema conceptual**.

En este paso, se describen las entidades, atributos y relaciones de la información que se debe almacenar posteriormente y que han de gestionar los sistemas informáticos.

La notación más popular para crear el esquema conceptual es el modelo entidad-relación, que se describe en el apartado siguiente.

Una vez se tiene el modelo conceptual, se puede generar el diseño lógico.

b) Diseño lógico

El diseño lógico es el proceso de construir un esquema de la información basándose en un modelo de base de datos (en red, jerárquico, relacional, etc.) e independientemente, sin embargo, del SGBD que después se utilizará para gestionar la información.

En el marco de este módulo, entenderemos por modelo lógico la creación del modelo relacional.

En esta etapa, se transformará el esquema conceptual en el conjunto de tablas, campos e interrelaciones entre las tablas con el objetivo de poder crear más tarde las bases de datos.

c) Normalización

Después de este paso, y antes de crear el esquema físico, es conveniente normalizar la base de datos, una técnica que se utiliza para verificar que las tablas obtenidas no tienen datos redundantes y están "optimizadas". En los próximos apartados se comentarán estos aspectos.

d) Diseño físico

El diseño físico es el proceso de generar la base de datos utilizando un sistema gestor de base de datos correspondiente.

En este paso, se llevan a cabo todas las actividades necesarias para crear las tablas, definir los campos, las interrelaciones entre tablas, índices, etc.

Creación de índices

Un índice permite ordenar y encontrar registros con mayor rapidez. Se puede indexar un único campo o una combinación de campos. La clave primaria de una tabla se indexa de forma automática. Determinados campos no se pueden indexar debido al tipo de datos que contienen.

En el marco de este material, sólo se tratará, a modo de introducción, el diseño conceptual y logicorrelacional y se introducirán conceptos sobre la normalización.

3.1. Modelos de datos: conceptos básicos

El objetivo básico de un sistema de BD es obtener, mediante la abstracción del mundo real, un conjunto estructurado de datos y un conjunto de operaciones definidas sobre ellas que permitan satisfacer, de forma eficiente, las necesidades de información de una organización.

La representación de una parcela de la realidad mediante un modelo de datos da lugar a un esquema. Sin embargo, el mundo real también comprende aspectos dinámicos, ya que los datos varían en el tiempo. Por eso, los modelos de datos se componen de dos submodelos que proporcionan elementos (o conceptos básicos) para representar la naturaleza estática y dinámica del sistema:

- **Naturaleza estática**: características inalterables que identifican al sistema y a sus objetos. Corresponden a lo que se entiende por estructura.

- **Naturaleza dinámica**: acciones que admite el sistema haciéndolo evolucionar y cambiar de estado a lo largo del tiempo. Describen el comportamiento de la estructura.

Un **modelo de datos** es el conjunto de conceptos, reglas y convenciones que permiten describir una parcela del mundo real que interviene en un problema determinado (de universo de discurso).

Los elementos que proporcionan los modelos de datos, a pesar de estar definidos con terminología y formalismo diferentes, tienen significados equivalentes. Eso permite agruparlos, genéricamente, en:

1) Elementos permitidos:

- Describen las **estructuras de datos** (los objetos, sus propiedades) y la forma en la que se relacionan (asociaciones).

- Definen las **operaciones** (y transacciones) que permiten la manipulación de los datos (adición, eliminación, modificación y recuperación).

2) Elementos no permitidos:

Son las **restricciones** o limitaciones impuestas a la estructura del modelo o a los datos, que invalidan determinadas ocurrencias en la BD. Se distinguen dos tipos:

a) Restricciones inherentes: son limitaciones de utilización en la misma naturaleza del modelo de datos, que no admite determinadas estructuras. Pueden ser de dos tipos:

- **Propias del modelo de datos**: su definición corresponde al diseñador, pero su gestión es responsabilidad del modelo de datos, que las reconoce y las recoge en el esquema.

- **Ajenas al modelo de datos**: el modelo de datos no las reconoce ni proporciona instrumentos para manipularlas; son responsabilidad del diseñador.

Ejemplos de restricciones inherentes

a) Propias del modelo de datos:

- El modelo relacional no permite atributos multivalor.
- Utilizar una regla de validación (restricción CHECK) para comprobar que la edad de los votantes es mayor de 18 años.

b) Ajenas al modelo de datos: las restricciones de cardinalidad mínima.

b) Restricciones semánticas o de integridad: son limitaciones impuestas a los valores de los atributos o a las características de las interrelaciones para reflejar, fielmente, la realidad. Permiten captar la semántica del universo de discurso que se quiere modelar, y verificar la corrección de los datos almacenados en la BD.

Ejemplos de restricciones semánticas

- El estado civil de una persona no puede pasar directamente de soltero/a a viudo/a.
- Una persona no puede tener una profesión si es menor de 18 años.
- El salario de un trabajador no puede ser mayor que el de su jefe o supervisor.

Las restricciones garantizan la integridad de la BD y la validez semántica de su contenido.

Todo **modelo de datos** comporta un modelado o proceso de abstracción que es la labor intelectual mediante la que se representa la realidad, con el fin de obtener una estructura para almacenar los datos.

Los **tipos de abstracción** básicos son los siguientes: clasificación/instanciación, generalización/especialización, agregación/desagregación y asociación/disociación.

3.2. Definición conceptual de una BD

Una base de datos es cara y difícil de crear.

La inclusión de una o más bases de datos en nuestras organizaciones implicará la necesidad de tiempo de preparación del software, la necesidad de contar con un experto para que diseñe la base de datos y un coste de mantenimiento.

Estas tareas se pueden llevar a cabo internamente o por medio de la contratación de terceros, externos a la organización.

Con el objetivo de generar una base de datos, se establece un proceso o metodología que se inicia con la visión del mundo exterior, en concreto, de la parte que nos interesa representar en datos.

En este proceso se debe aprender, comprender y conceptualizar este mundo exterior y transformarlo en un conjunto de ideas y definiciones que representen una imagen fiel del comportamiento del mundo real.

Este proceso de abstracción genera lo que conocemos como el **modelo conceptual**.

La definición correcta del modelo conceptual de la base de datos es imprescindible para garantizar la generación correcta de la posterior base de datos.

A continuación, se enumeran los componentes y elementos principales de una metodología para definir un modelo conceptual de datos: **entidades**, **atributos**, **claves** y **relaciones**.

3.2.1. Modelo de datos: el modelo entidad-relación

El modelo entidad-relación se basa en el uso de un conjunto de símbolos gráficos que permiten representar "la realidad" de la información que se quiere gestionar.

De esta manera, se identifican diferentes elementos de información como son las **entidades** y sus **atributos**, las **relaciones entre entidades** y la **cardinalidad** y el **grado** de estas relaciones.

El correcto diseño conceptual de la base de datos, siguiendo las normas y reglas de este modelo, facilitará posteriormente la definición y creación de la base de datos relacional.

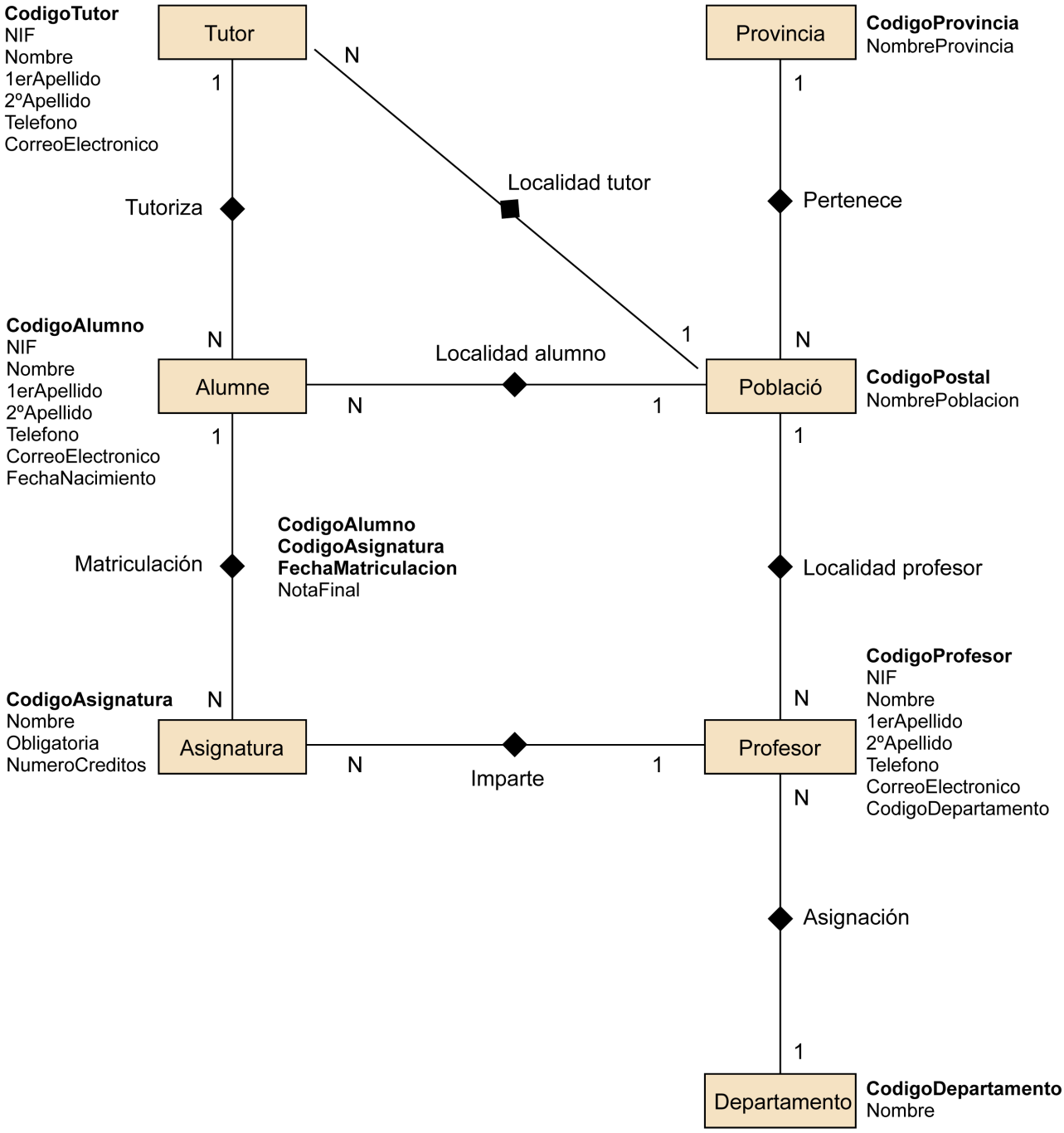
En este apartado, se definen los principales elementos de este modelo y en los casos prácticos del módulo se ejemplariza su uso.

Ejemplo de modelado

El siguiente es un posible modelado de la BD relacional de un centro de estudios (con departamentos, profesores, asignaturas, alumnos...) expuesta en los ejemplos de Bases de datos relacionales" del apartado anterior.

Para saber más

Otros ejemplos de modelación conceptual (generación del diagrama entidad-relación) se incluyen al final, en el subapartado "Casos prácticos de diseño y creación de bases de datos".



Para simplificar el modelado del ejemplo, se hacen los siguientes supuestos o restricciones semánticas:

- Una asignatura sólo puede ser impartida por un profesor.

- Los tutores no pueden ser profesores.

Equivalencia de los elementos del modelo relacional y del modelo entidad-relación

La tabla siguiente muestra la equivalencia entre los elementos básicos del modelo relacional y los del modelo entidad-relación.

	Modelo de BD		
	Relacional	**Entidad-relación**	Ejemplos
Elementos	Tabla	Entidad	Alumno, asignatura, profesor
	Registro	Ocurrencia	Joan Soler Pou, estadística
	Campo	Atributo	Nombre, apellidos, NIF, dirección
	Clave	Clave	NIF
	Relación	Relación	Matriculación (relación alumno-asignatura)

3.2.2. Entidades

Una entidad es un objeto real o abstracto con características diferenciadoras de otros objetos y cuya información se almacenará en la base de datos.

Para saber más

Los ejemplos presentados a continuación, para ilustrar los elementos del modelo entidad-relación, tienen un paralelismo directo con los expuestos para cada elemento del modelo relacional en el apartado "Bases de datos relacionales".

Toma como significado conceptos u objetos que tienen un papel importante en la compañía u organización.

Una entidad está formada por un conjunto de elementos de datos o atributos, cada uno de los cuales aporta alguna característica a la definición de la entidad.

Ejemplos de entidades

Estableciendo una analogía con los ejemplos vistos anteriormente, ejemplos de entidades serían Alumnos, Asignaturas y Profesores.

3.2.3. Ocurrencia

Por ocurrencia se entiende cada uno de los elementos que representa una entidad. Es decir, cada uno de los alumnos o profesores o cada una de las asignaturas sería una ocurrencia de las entidades Alumnos, Profesores y Asignaturas.

3.2.4. Atributo

Un atributo es una unidad básica e indivisible de información sobre una entidad que sirve para identificarla o describirla.

Ejemplo de atributos

Por ejemplo, el nombre, los apellidos, la dirección, el NIF, etc. serían atributos de la entidad Alumnos.

Normalmente, se utiliza la denominación **atributo** para especificar el nombre de atributo, que se tiene que diferenciar del valor de atributo o **valor.**

Ejemplos de valor

55.123.708-H, Joan, Soler Pou, 36 y Mayor, 51, 2.º-1.ª son, respectivamente, los valores de los atributos DNI, nombre, apellidos, edad y dirección.

Los atributos (y los campos) pueden ser de diferentes tipos:

- Atributo compuesto: dirección (se puede dividir en calle, número, piso y puerta) o fecha de nacimiento (día, mes y año).

- Atributo simple (o atómico): calle, año de nacimiento.
- Atributo derivado (o calculado): edad.
- Atributo almacenado: fecha de nacimiento.
- Atributo monoevaluado: edad.
- Atributo multievaluado: teléfono.

Un valor nulo puede tener varias interpretaciones:

- Valor no aplicable: el atributo no se aplica en esta ocurrencia de entidad (o registro). Por ejemplo, si una persona no tiene DNI.
- Valor desconocido: falta (por ejemplo, si no se conoce la estatura); no se sabe si existe (por ejemplo, correo electrónico); o se conoce, pero está ausente, todavía no se ha registrado.

3.2.5. Clave

Se denomina *clave de una entidad* al atributo o conjunto de atributos que permite identificar, de manera única, un elemento de una entidad.

Ejemplo de clave

El NIF podría ser, por ejemplo, la clave de las entidades Alumnos y Profesores.

3.2.6. Relaciones

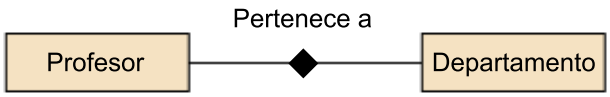
Las entidades, por sí solas, no describen la realidad de un sistema de información.

No es suficiente con identificar objetos; además, se han de definir las asociaciones que se dan entre los objetos o entidades.

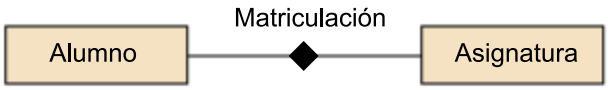
Estas asociaciones se denominan *relaciones*.

Ejemplos de relación

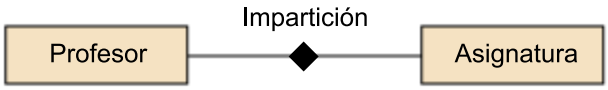
- La relación "Pertenece a" (o "Pertenencia a") que hay entre las entidades PROFESOR y DEPARTAMENTO.



- La relación "Matriculación" (o "Matriculado en") que se establece entre las entidades ALUMNO y ASIGNATURA.



- La relación "Impartición" existente entre las entidades PROFESOR y ASIGNATURA.



Toda relación presenta dos características que la definen: el grado (número de entidades que participan en la relación) y la cardinalidad de cada uno de ellas en la relación. Se describen a continuación.

Grado de una relación

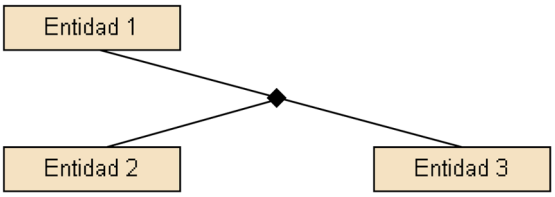
Por este concepto, se identifica el número de entidades que se interrelacionan.

La mayoría de las veces, el grado de una relación será binario (entre dos entidades, como las vistas anteriormente) o ternario (entre tres entidades, tal y como se verá en uno de los ejemplos posteriores del material).

Ejemplo de relación binaria y ternaria

Relación binaria

Relación ternaria



Se podría dar el caso de relaciones cuaternarias o superiores, pero son casos complejos y poco frecuentes.

Cardinalidad de una relación

La cardinalidad o el grado de una relación representa la participación en la relación de cada una de las entidades afectadas.

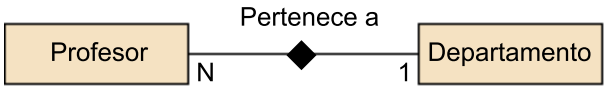
Eso, gráficamente, se indica poniendo sobre la línea que une las entidades el grado de participación (1, N o M) según corresponda a cada entidad.

En una relación binaria, hay tres tipos posibles:

- **Una a muchas** (*1:n*). A cada ocurrencia de la primera entidad le corresponde una o varias ocurrencias de la segunda.

Ejemplo de relación *1:n*

La relación "Pertenece a" tiene una cardinalidad "1 a n", ya que cada profesor pertenece a 1 único departamento y a un departamento pertenecen varios profesores (N).

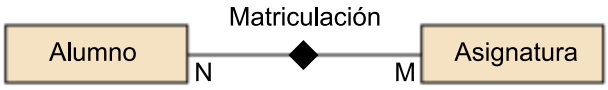


Eso se indica gráficamente poniendo, en la línea que une las entidades, una N junto a la primera entidad y un 1 junto a la segunda. En este caso, la primera entidad es DEPARTAMENTO (1) y la segunda PROFESOR (N).

- **Muchas a muchas** (*m:n*). A cada ocurrencia de la primera entidad le pueden corresponder diferentes ocurrencias de la segunda, y viceversa.

Ejemplo de relación *m:n*

La relación entre las entidades ALUMNO y ASIGNATURA, que se puede denominar Matriculación, tiene una cardinalidad "n a m", ya que cada alumno se puede matricular en más de una asignatura (M), y en cada asignatura se matricula más de un alumno (N).



- **Una a una** (*1:1*). A cada instancia u ocurrencia de una entidad le corresponde una y sólo una ocurrencia de la otra. Este caso se da pocas veces.

Ejemplo de relación *1:1*

Este caso se presenta al dividir en dos una entidad para separar/agrupar sus datos en dos tipos de entidad diferentes.

3.3. Del modelo conceptual al lógico o relacional

Una vez diseñado el modelo conceptual de una base de datos, se puede generar el modelo relacional a partir de este modelo.

A continuación, se incluye un resumen de las reglas de conversión principales:

- Toda entidad del modelo conceptual se convierte en una tabla en el modelo relacional. Por ejemplo, alumnos, profesores, asignaturas, etc.

- Todo atributo de una entidad se transforma en un campo de la tabla. Por ejemplo, de la entidad Alumnos se generan los campos Código, Nombre, Primer apellido, Segundo apellido, Teléfono, etc.

- Toda relación 1:*n* se convierte en una clave foránea en la tabla correspondiente a la entidad de grado *n*. Por ejemplo, la relación Adscripción entre profesores y departamentos.

- Toda relación *m*:*n* se transforma en una tabla intermedia. Por ejemplo, la tabla correspondiente a la relación Matriculación.

Para saber más

Podéis ver la conversión de la relación *1:n* ("adscripción", entre PROFESOR y DEPARTAMENTO) en clave foránea y de la relación *m:n* (matriculación, entre ALUMNO y ASIGNATURA) en tabla intermedia en los ejemplos expuestos en el ítem "Relaciones" del subapartado "Bases de datos relacionales".

Etapas de diseño conceptual y lógico de la base de datos

En esta asignatura sólo se abordarán, desde un nivel práctico, los pasos siguientes del diseño:

a) Diseño del modelo conceptual (modelo entidad-relación). Identificación de las entidades, sus atributos y las interrelaciones entre ellas, con los atributos de éstas, si fuera el caso.

b) Generación del modelo relacional (modelo lógico). Partiendo del modelo conceptual diseñado en el apartado anterior, éste se tiene que convertir en una BD relacional, indicando las tablas correspondientes, los campos de cada uno, el tipo de dato de cada campo, la clave primaria de cada tabla y las claves foráneas correspondientes.

Pautas para denominar entidades, atributos, tablas y campos

Los nombres de entidades y atributos (y, por extensión, de tablas y campos) pueden incluir cualquier combinación de letras, números, espacios en blanco y caracteres especiales excepto el punto (.), el signo de admiración (¡!), el acento grave (`) y los corchetes ([]), entre otros. Además, en Microsoft Access, las tablas y campos tampoco pueden empezar por un espacio en blanco ni incluir comillas dobles (") y, en total, pueden tener hasta 64 caracteres.

a) Consejos:

- A pesar de que se pueden usar espacios, es mejor evitarlos, ya que se pueden producir conflictos si se hace referencia a nombres con espacios en expresiones o código de Visual Basic para aplicaciones.

- Evitar el uso de nombres extremadamente largos porque es difícil recordarlos y referirse a ellos.

- Evitar que el nombre coincida con el nombre de una propiedad o elemento que utilice el SGBD (Microsoft Access), pues podría producir un comportamiento inesperado de la BD.

b) Convenios:

- Entidades (y tablas) en singular y mayúscula.
- Atributos (y campos) en singular, iniciales de palabra en mayúscula y resto en minúscula.
- Si tienen varias palabras, éstas se juntan sin dejar espacios. Otra opción, menos recomendable, es usar el guión bajo (_) para separarlas.

Para saber más

Al final del módulo, hay dos casos resueltos de diseño de BD donde se ejemplifican las etapas de diseño conceptual y lógico.

3.4. Normalización de bases de datos

La normalización de bases de datos es el proceso mediante el cual se transforman datos complejos en un conjunto de estructuras de datos más pequeños, que además de ser más simples y más estables, son más fáciles de mantener.

También se puede entender la normalización como una serie de reglas (formas normales) que sirven para ayudar a los diseñadores de bases de datos a desarrollar un modelo relacional que minimice los problemas de lógica.

Cada regla o forma normal está basada en la anterior.

La normalización se adoptó porque la antigua manera de poner todos los datos en un lugar, tanto en un archivo como en una tabla de la base de datos, era ineficiente y conducía a errores cuando se querían manipular los datos.

La normalización también hace las cosas más fáciles de entender entre los desarrolladores de la base de datos y los usuarios que la requieren y plantean los requisitos.

Otra ventaja de la normalización de una base de datos es que ayuda a optimizar espacio. Una base de datos normalizada ocupa menos espacio que una no normalizada, ya que evitamos repeticiones de valores innecesarios.

3.4.1. Grados de normalización

El proceso de normalización de las bases de datos tiene un conjunto de reglas para cada fase.

Básicamente, hay tres niveles de normalización:

- Primera forma normal (1FN).
- Segunda forma normal (2FN).
- Tercera forma normal (3FN).

Cada una de estas formas normales tiene sus reglas.

Cuando una base se conforma en un nivel, se considera normalizada según aquella forma normal.

Algunas veces, tener una base de datos normalizada al más alto nivel la puede hacer más compleja que si no lo está.

Normalización de una base de datos

Algunas veces, la normalización completa de una base de datos podría devenir en un gran número de tablas. Se podría dar el caso, entonces, de que para obtener informaciones relacionadas, hubiera que unir muchas de estas tablas, por medio de una consulta, moderando el proceso. En estos casos sería más eficiente disponer de tablas no normalizadas pero con la información ya relacionada.

Como introducción, en la tabla siguiente se introduce cada una de estas tres formas normales.

Regla (forma normal)	Descripción
Primera forma normal (1FN)	Consiste en la eliminación de todos los grupos repetidos
Segunda forma normal (2FN)	Asegura que todas las columnas (campos) que no son clave, dependan completamente de la clave primaria
Tercera forma normal (3FN)	Elimina las dependencias transitivas. Una dependencia transitiva es aquélla en la que las columnas (campos) que no son clave dependen de otras columnas (campos) que tampoco son clave.

3.4.2. Primera forma normal (1FN)

La regla de la primera forma normal establece que las columnas repetidas se deben eliminar y colocar en tablas separadas.

De esta manera se resuelven los problemas de cabeceras de columna múltiples. En lugar de tener una única tabla inmensa que tiene muchos aspectos, disponiendo de distintas tablas más pequeñas y gestionables, cada una con los datos de un aspecto concreto.

En este caso, la normalización ayuda a clarificar la base de datos y organizarla en partes más pequeñas y más fáciles de entender.

3.4.3. Segunda forma normal (2FN)

La regla de la segunda forma normal establece que todas las dependencias parciales se deben eliminar y separar de sus propias tablas. Una dependencia parcial es un término que describe los datos que no dependen de la clave primaria de la tabla para identificarlas.

3.4.4. Tercera forma normal (3FN)

Una tabla está en la tercera forma normal si todas las columnas que no son clave primaria son total y funcionalmente dependientes de la clave primaria y no hay dependencias transitivas. Una dependencia transitiva es aquélla en la que hay columnas que no son clave primaria y que dependen de otras columnas que tampoco son clave primaria.

Cuando las tablas están en la tercera forma normal se evitan errores al insertar o borrar registros.

Cada campo en una tabla está identificado de manera única por la clave primaria y no hay datos repetidos.

3.4.5. Ejemplos

El objetivo de este módulo es, simplemente, introducir el concepto de normalización, pero no se espera que los estudiantes dominen las técnicas de diseño de bases de datos y su normalización, por lo que, a continuación, se enumeran dos casos como ejemplo para hacer más fácil la comprensión de los conceptos.

Ejemplo de normalización en 1 FN

Los datos de los socios de un club de ocio y de las actividades que han llevado a cabo se podrían recoger en una tabla con la siguiente estructura:

Base de datos sin normalizar (una tabla)

NIF	Núm. socio	Nombre	Apellidos	...	Actividad	Fecha actividad	...
1					Globo	01-06-10	
1					Salida a Cardona	05-07-10	
2					Globo	01-06-10	
2					Salida a Cardona	05-07-10	
3					Salida a Cardona	05-07-10	
3					Baloncesto	30-06-10	
4					Baloncesto	30-06-10	
5					Baloncesto	30-06-10	
...							

En la tabla, se puede comprobar que los socios con NIF 1 y NIF 2 participaron en las actividades de globo y salida a Cardona, y que el socio con NIF 3 participó en las actividades salida a Cardona y baloncesto.

La tabla no está normalizada, dado que se presentan grupos de datos repetidos. Por una parte, en cada registro de actividad de un socio se repiten todos sus datos personales (núm. socio, nombre, apellidos, etc.) y, por la otra, para todos los socios que han parti-

cipado en una misma actividad se repiten los datos de la actividad (nombre y fecha de realización).

Base de datos normalizada (1FN) (dos tablas)

Socio				
NIF	**Núm. socio**	**Nombre**	**Apellidos**	**...**
1				
2				
3				
4				
5				
...				

Actividad		
Actividad	**Fecha actividad**	**...**
Globo	01-06-10	
Salida a Cardona	05-07-10	
Baloncesto	30-06-10	

Para obtener la BD normalizada en 1FN, se han colocado los grupos de columnas que repiten datos en tablas separadas. De esta manera, ya no se repiten los datos de los socios ni los datos de las actividades.

El siguiente paso en el proceso de normalización, dado que la relación entre las tablas resultantes es muchos a muchos (un socio puede participar en más de una actividad, y cada actividad puede tener varios socios participantes), sería crear una tabla intermedia.

Ejemplo de normalización en 2FN

Una situación muy habitual en BD, que contiene información de personas y/o de organizaciones, es la relación existente entre los códigos postales y la población.

Siguiendo con el ejemplo anterior, al definir los datos de la tabla SOCIO también se incluye el código postal y la población, tal como muestra la tabla.

Base de datos sin normalizar (una tabla)

NIF	Nombre	Apellidos	....	Código postal	Población	...
1				08700	Igualada	
2				08700	Igualada	
3				08202	Sabadell	
4				08206	Sabadell	
5				08225	Terrassa	
6				08760	Martorell	

NIF	Nombre	Apellidos	....	Código postal	Población	...
7				08700	Igualda (*)	
8				08202	Sabadell	
9				08760	Martorel (*)	
...						

Se puede comprobar que casi todos los campos dependen de la clave primaria (que puede ser el NIF). En cambio, el campo población depende del código postal. Eso significa que la BD no está normalizada según la segunda forma normal.

Para normalizarla, hay que crear una segunda tabla (POBLACIÓN) donde haya un registro para cada código postal y población, y en la tabla SOCIO dejar el campo código postal (será clave foránea) que se vinculará con la tabla POBLACIÓN.

En la tabla, se puede apreciar que se repiten valores (del código postal y la población) en los registros correspondientes al NIF 1 y NIF 2, o al NIF 3 y NIF 8.

Además, en una situación real, al no estar la BD normalizada según la 2FN, puede haber datos equivocados. En el ejemplo, eso se evidencia en los registros de los NIF 7 y NIF 9, en los que el nombre de la población es incorrecto ("Igualda" y "Martorel", respectivamente, que se han marcado con asterisco), aunque el código postal está bien. En tal caso, si se hace una consulta, por el campo población, de los socios que viven en Igualada no se encontrará el registro del NIF 7. Lo mismo pasará con Martorell y el registro del NIF 9.

Base de datos normalizada (2FN) (dos tablas)

Socio					
NIF	Nombre	Apellidos	....	Código postal	....
1				08700	
2				08700	
3				08202	
4				08206	
5				08225	
6				08760	
7				08700	
8				08202	
9				08760	
...					

Población	
Código postal	Población
08700	Igualada
08202	Sabadell

Población	
Código postal	**Población**
08206	Sabadell
08225	Terrassa
08760	Martorell
...	

Una vez normalizada la BD, queda resuelto el problema de los datos repetidos en la tabla SOCIO, y también posibles errores como los detectados en la situación anterior.

3.4.6. Otras formas normales

Después de la tercera forma normal, hay algunas más (forma normal Boyce-Codd, cuarta forma normal, quinta forma normal o forma normal de proyección-unión, etc.), pero si se garantiza tener las bases de datos en tercera forma normal, se resuelven la mayoría de los problemas y las dificultades.

3.5. Casos prácticos "Diseño y creación de bases de datos"

A continuación, se plantean y se resuelven dos casos prácticos que ayudan a ejemplificar los conceptos expuestos en el presente apartado de diseño de BD. Uno corresponde a un centro de formación, y el otro a una empresa de alquiler de coches. En el segundo, hay un ejemplo de relación ternaria.

El diseño de una base de datos no es único, ya que depende de los supuestos y/o restricciones que haya que tener en cuenta.

En una situación real, es habitual que el diseñador de la BD solicite diferentes requerimientos al cliente (responsable y/o usuarios de la BD) para ir aclarando posibles dudas que se presenten durante el proceso de diseño.

Por lo tanto, los ejercicios propuestos no tienen una única solución, ya que en función del diseño del modelo conceptual que se haga, la BD relacional obtenida será una u otra.

3.5.1. Caso práctico 1: Base de datos "Centro de formación"

FormProf, S. A., Formación Profesional en Informática, es un centro de formación de las tecnologías de información, de creación reciente.

Sus servicios se basan en la planificación y la ejecución de programas de formación en el área de informática, tanto en el ámbito del usuario (introducción a Internet, uso de MSFT Outlook, trabajo con procesadores de textos, creación

de animaciones con Flash, etc.), como en el de "programadores especialistas" (desarrollo de aplicaciones con Visual Studio .NET, instalación y configuración de redes con Unix, etc.).

La empresa se acaba de crear y a vosotros os han contratado como responsables del área de informática. No como profesores, sino como responsables de los servicios internos de TI.

El equipo humano del centro y su organización es la siguiente:

- Un director, responsable máximo del centro y de sus operaciones.
- Departamento Comercial y de Marketing. Dependiente del director del centro, habrá un equipo de fuerza de ventas, compuesto por dos comerciales asesores que cubrirán todo el territorio, junto con una persona responsable de las campañas de marketing y promoción.
- Un responsable del área de RRHH y Finanzas.
- Un auxiliar administrativo que dará apoyo a las tareas de RRHH y Finanzas y Contabilidad.
- Un responsable tanto de la recepción y atención al cliente, como de dar apoyo al director.
- Finalmente, hay un responsable de tecnologías, que sois vosotros.

Como equipo docente, tenemos la organización siguiente:

- Un jefe de estudios que también ejerce como profesor puntualmente.
- Dos profesores, pertenecientes a la plantilla del centro, responsables tanto de la dirección de programas de formación, como de impartir parte de sus módulos.
- Un conjunto de profesores, ajenos al centro, contratados ocasionalmente para impartir algunos de los módulos y cursos ofrecidos por FormProf.

A continuación, se especifican los requisitos de una base de datos para almacenar y gestionar toda la información de las acciones formativas que se llevan a cabo en este centro, y también el seguimiento de los clientes, los alumnos, las sesiones de formación, etc.

- Por una parte, se debe almacenar la información de todos los **cursos** que se planifican y ejecutan en nuestra compañía. Para cada curso, se ha de guardar la información de su identificador (será un código de cinco letras), el título del curso, la fecha de inicio y la fecha de finalización y un valor,

del tipo Sí/No, que indicará si el curso es de calendario (valor Sí) o a medida (valor No) para una compañía.

- Así, se debe guardar la información de las **compañías** clientes, indicando su CIF, razón social, dirección, código postal, población, teléfono, correo electrónico y nombre de la persona de contacto.

- Un curso lo puede organizar una compañía (en el caso de que sea a medida) o de calendario. En este último caso, no tiene ninguna compañía asociada. Una compañía nos puede contratar más de un curso.

- Para cada curso, también se almacenará la información de los **asistentes** o matriculados. Es necesario disponer de la información siguiente: NIF, nombre, apellidos, dirección, teléfono y correo electrónico. Se ha de tener en cuenta que un alumno puede participar en más de un curso.

- Por otra parte, también debemos poder guardar la información de las **sesiones** de un curso. Todo curso tendrá un mínimo de una sesión (en caso de que sea de un solo día). Para cada sesión debemos guardar la información siguiente: código de la sesión (identificador numérico), hora de inicio, hora de fin y fecha de realización.

- Para cada sesión se debe guardar la información del **profesor** responsable. De cada profesor guardaremos el NIF, el nombre y los apellidos. Sólo habrá un profesor por sesión, pero, evidentemente, un profesor participará en distintas sesiones de formación.

- Y, finalmente, es necesario guardar **el aula** donde se llevará a cabo la sesión (Aula 1, Aula 2 o "In Company").

El correcto diseño de la BD tiene que permitir realizar consultas para obtener, a partir de la información almacenada, los siguientes resultados:

- Determinar qué cursos son de calendario.

- Especificar la empresa para la que se ha organizado cada uno de los cursos hechos a medida.

- Obtener la lista de todos los alumnos asistentes a un curso.

- Conocer qué profesores impartirán alguna sesión de un curso.

- Saber qué sesiones tiene que impartir un profesor entre unas fechas concretas.

- Identificar las sesiones que se tienen que llevar a cabo o se han de realizar en una fecha concreta.

Se pide realizar las siguientes actividades:

1) Diseño del modelo conceptual de la BD. Identificación de las entidades, sus atributos y las interrelaciones entre ellas, con los atributos de éstas (si fuera el caso) en un esquema entidad-relación.

2) Generación del modelo relacional de la BD. Partiendo del esquema conceptual diseñado en el apartado anterior, convertirlo a una BD relacional, indicando las tablas correspondientes, los campos de cada una, el tipo de dato de cada campo, la clave primaria de cada tabla y las claves foráneas correspondientes.

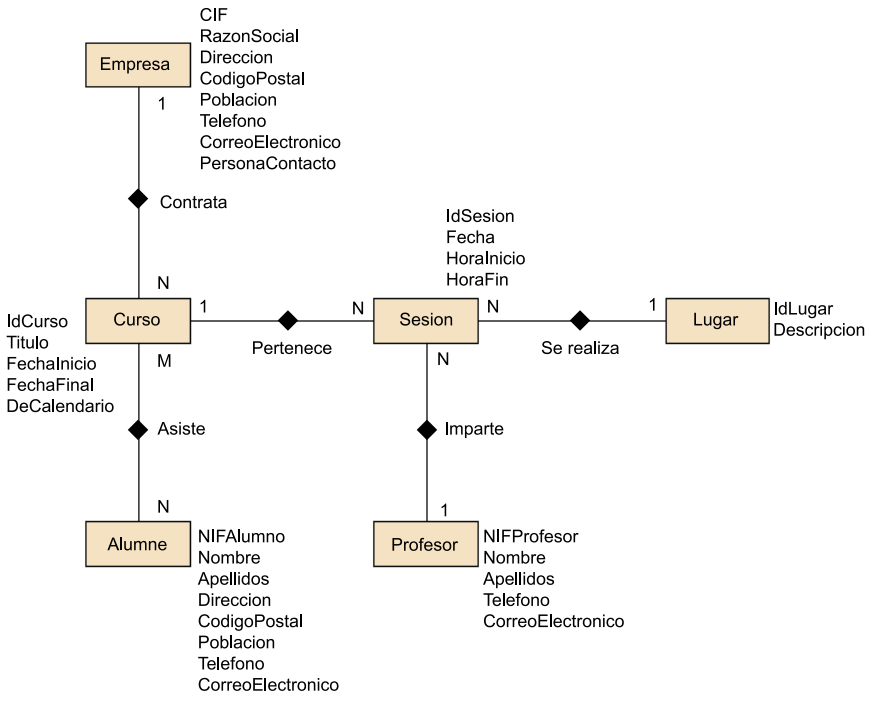
Caso práctico 1: Base de datos "Centro de formación". Resolución

Este ejercicio no tiene una solución única, ya que en función del diseño del modelo conceptual, la base de datos relacional final será una u otra.

En una situación real, durante el diseño de la base de datos, el responsable de ésta solicita diferentes requerimientos al "cliente" para ir determinando ambigüedades y dudas posibles.

1) Modelo conceptual de la base de datos

El modelo conceptual se representa, gráficamente, mediante el siguiente diagrama entidad-relación (DER), donde se identifican las entidades, sus atributos y las relaciones entre entidades:



Notas:

- Hay que destacar la relación *m:n* entre CURSO y ALUMNO, ya que normalmente un curso tiene diferentes alumnos y un alumno puede estar matriculado en varios cursos.

- En un modelo real, sería conveniente considerar la población como una entidad identificada por el código postal. De esta manera, no habría que guardar el nombre de la población en las entidades EMPRESA y ALUMNO, sino que la existencia de dos interrelaciones (empresa-población y alumno-población) permitiría guardar sólo el código postal. Este aspecto, sin embargo, no se ha considerado en el caso pronunciado con el fin de simplificar el diseño.

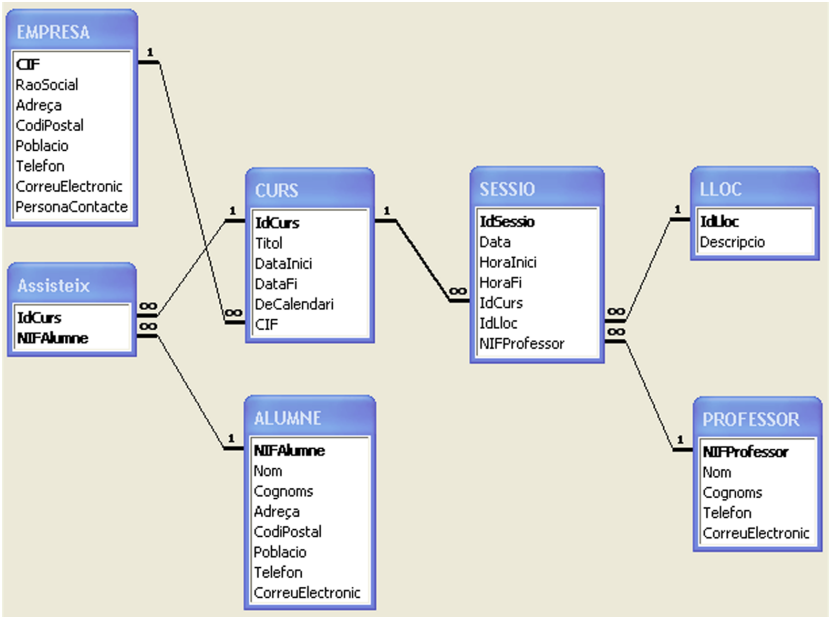
2) Generación modelo de base de datos relacional

Para generar este modelo, se han aplicado las siguientes reglas de conversión:

- La relación original *m:n* entre las entidades CURSO y ALUMNO se transforma en tres tablas. Una para CURSO, otra para ALUMNO y una intermedia que permite almacenar la información de la relación asiste (o alumno-curso) existente entre ellas.

- El resto de entidades se transforman en tablas; los atributos de cada entidad, en campos de cada tabla; y las relaciones *1:n* se establecen entre la clave primaria de una tabla (extremo 1) y la clave foránea de otra (extremo *n*).

La siguiente imagen muestra las tablas, sus campos y las relaciones entre las tablas del esquema relacional de la BD creado con MS Access. Los campos que forman clave primaria aparecen en negrita.



3.5.2. Caso práctico 2: Agencia de alquiler de coches

Se quiere diseñar una base de datos para almacenar y gestionar la información utilizada en una empresa dedicada al alquiler de vehículos, teniendo en cuenta los requisitos siguientes:

- La empresa dispone de un conjunto de vehículos para su alquiler. Se necesita conocer la matrícula, marca, modelo, color, tipo de combustible y tarifa diaria de alquiler de cada coche.

- Cada vehículo está asignado a un determinado garaje, que no puede cambiar, del que interesa conocer su código, la denominación, los datos de ubicación (dirección, población, código postal) y el teléfono.

- Los datos para almacenar de cada cliente son el NIF, el nombre, los datos de ubicación y los de contacto (teléfono y correo electrónico).

- Cuando un cliente desea alquilar un coche, se pone en contacto con alguna de las agencias o delegaciones de la empresa de alquiler para solicitar una reserva.

- De cada agencia se guardará su código, la denominación, los datos de ubicación y las de contacto.

- Un mismo cliente puede hacer o haber hecho varios reservas a lo largo del tiempo, que se pueden solapar en el tiempo.

- Un cliente puede hacer reservas en diferentes agencias. En cada reserva, participa una única agencia.

- Una reserva puede incluir más de un vehículo. De cada reserva se registrará la fecha de inicio (fecha de entrega del/de los vehículo/s al cliente), la fecha de finalización (fecha prevista para el retorno del/de los vehículo/s), el precio total de la reserva y un indicador de si el/los vehículo/s se ha/n devuelto.

- Para cada coche, interesa recoger el nivel de combustible que contiene el depósito en el momento de su entrega al cliente.

- El precio final de la reserva de cada coche se obtiene multiplicando la tarifa diaria de alquiler por el número de días que el cliente lo quiere reservar.

- El precio total de una reserva se obtiene sumando los precios finales de alquiler de todos los coches incluidos en la reserva.

El correcto diseño de la base de datos deberá permitir que la empresa pueda realizar consultas del tipo siguiente:

- Lista de todas las agencias, coches y clientes.

- Historial de todas las reservas realizadas indicando el cliente, las fechas y la agencia que lo ha tramitado.

- Número de reservas realizadas por los clientes de una población concreta.

- Resumen de las reservas (número, fecha inicio, fecha final y precio total) pendientes de devolución, ordenadas por fecha de finalización.

- Lista de los coches guardados en cada garaje.

- Agencias (código y nombre) que participan en reservas todavía no devueltas.

- Lista de todas las reservas que ha tenido un coche determinado.

- Cliente con más de tres reservas, alguna de las cuales la ha realizado la agencia con código "4".

- Para cada cliente de la compañía, lista de todas las reservas realizadas, así como el precio medio de todas ellas.

- Las agencias que no participan en ninguna reserva con fecha de inicio posterior a una fecha concreta.

- Todos los coches (matrícula, marca y modelo) reservados más de tres veces.

- La marca y el modelo del coche más reservado.

- Coches que no se han reservado nunca.

- Agencias que participan en más de tres reservas de una semana o más de duración.

- Precio de la reserva más cara realizada para cada marca y modelo de coche.

Caso práctico 2: Agencia de alquiler de coches. Resolución

En este apartado se presenta una solución posible para este enunciado, dando por hecho que puede haber otras.

1) Diseño del modelo conceptual

A continuación se indican las entidades y los atributos identificados en el modelo conceptual (se ha subrayado el atributo que identifica de manera única las instancias de las entidades):

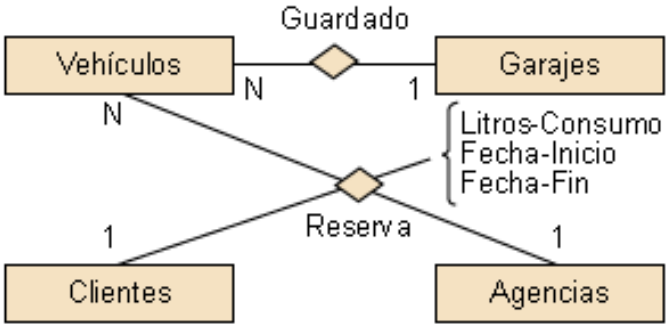
- Vehículos: matrícula, marca, modelo, color, tipo combustible y tarifa diaria.
- Garajes: código, denominación, dirección, código postal, población, teléfono.
- Clientes: NIF, nombre, apellidos, dirección, código postal, población, teléfono, correo electrónico.
- Agencias: código, nombre, dirección, código postal, población, teléfono, correo electrónico.

Por otra parte, se han identificado las relaciones siguientes entre las entidades:

- Guardado: Vehículos-Garajes
- Reserva: Vehículos-Cliente-Agencia

En el caso de la relación reserva, también se debe guardar una serie de atributos: litros-consumo, fecha-inicio, fecha-fin.

De manera gráfica, este modelo se representará del modo siguiente:



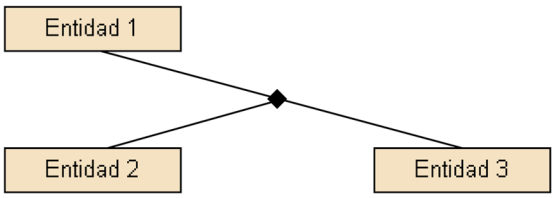
Modelo relacional

Al crear el modelo relacional, debemos tener en cuenta las consideraciones siguientes:

- Cada entidad será una tabla.
- El atributo de cada entidad es un campo de la tabla.
- El atributo que identifica de manera única las instancias de la entidad es la clave primaria de la tabla.
- La relación binaria Guardado, entre vehículos y garajes (1:n) se implementa por medio de una clave foránea en la tabla vehículos que "apunta" al código de garaje donde se encuentra aquel coche.
- La relación "ternaria" Reservas entre vehículos, cliente y agencia, se convierte en una tabla intermedia que contendrá las claves primarias de estas tres tablas, así como los litros consumidos por el vehículo concreto, la fecha de inicio de la reserva y la fecha de fin.
- La clave primaria de esta tabla será la combinación de las tres claves (vehículo, cliente y agencia) junto con la fecha de inicio de la reserva.

A continuación se presentan las diferentes tablas (creadas utilizando el MS Access):

Y en la imagen siguiente, se identifican las relaciones entre las diferentes tablas de la base de datos en MSFT Access:



Notas:

- También se habría podido realizar el diseño de la base de datos considerando "Reservas" como una entidad. Se ha preferido hacerlo de la otra manera para introducir un caso con una relación ternaria.
- Si se quisiera normalizar la base de datos deberíamos crear una tabla de Poblaciones, donde estaría almacenado el código postal y la población a la que corresponde. En las tablas Clientes, Agencias y Garajes no estaría el campo correspondiente a la po-

blación, sino el campo correspondiente al Código postal que "apuntaría" (sería clave foránea) a la tabla de Poblaciones.

Para saber más

Para tener más información sobre la normalización de las bases de datos, podéis consultar el apartado "Normalización de base de datos" de este mismo módulo.

3.6. Resumen

Un **modelo de datos** es el conjunto de conceptos, reglas y convenciones que permiten describir una parcela del mundo real que interviene en un problema determinado (de universo de discurso).

El diseño y la creación de una base de datos se caracterizan por lo siguiente:

- El diseño de una BD consiste en definir el modelo lógico; habitualmente, el **modelo relacional**. Es decir, construir un esquema de la información (crear las tablas, sus campos y las relaciones entre tablas), con independencia del SGBD que se utilizará.

- Previo al diseño de este modelo relacional, se aconseja hacer el **modelo conceptual**, que permite plasmar, en un modelo entidad-relación, las diferentes informaciones que se quieren almacenar (entidades, atributos e interrelaciones) con independencia de consideraciones físicas.

- Antes de crear el esquema físico, es conveniente aplicar una técnica de **normalización** de las tablas para verificar que éstas no tienen datos redundantes y están "optimizadas".

- Finalmente, en el **diseño físico** se implementan todos los elementos de la BD (tablas, campos, interrelaciones, índices, etc.) mediante un SGBD concreto.

Esquema de la conversión de elementos del modelo conceptual al modelo relacional

Modelo conceptual		**Modelo relacional**
Entidad	**se convierte en**	Tabla
Atributo	Campo	
Relación *1:n*	Clave foránea en la tabla correspondiente a la entidad de grado *n*	
Relación *m:n*	Tabla intermedia	

4. Bases de datos documentales

Tal como ya hemos visto en los apartados anteriores, una base de datos se compone de diferentes registros con su correspondiente número de identificación.

> Las bases de datos documentales **(BDD)** se caracterizan por que cada registro se corresponde con un documento, de cualquier tipo, publicación, documento gráfico o sonoro, etc., o con su referencia.

La información contenida en una base de datos documental se estructura en diferentes campos con el fin de facilitar el control a ella y el acceso individualizado.

Algunos campos se referirán a la descripción formal del documento y, en otros, se tratará el contenido temático. Incluso, en algunos casos, podrán registrar el mismo documento.

Para facilitar la rapidez en la recuperación de la información, las BDD facilitan la elaboración de diccionarios o índices alfabéticos. Algunos de estos sistemas trabajan con un índice único formado por palabras procedentes de diferentes campos de cada registro, mientras que en otros sistemas se gestionan índices para cada campo.

Las BDD se pueden clasificar en diferentes categorías según la información que contienen y la referencia al documento correspondiente: BD de texto completo (que contienen los propios documentos), archivos electrónicos de imágenes (que contienen imágenes de los documentos originales) y BD referenciales (que contienen referencias para localizar los documentos originales).

También se pueden clasificar por otras tipologías: según el **modo de acceso a la información**, la **cobertura documental** o según el **modelo de tratamiento documental**.

Por lo tanto, para utilizar eficazmente una BDD se recomienda adaptarse tanto como se pueda a sus características particulares.

Se debe estar informado del contenido y de cómo se han de efectuar las buscas, si cubre una o varias bases de datos, cuál es la cobertura temática, si es un catálogo o dispone de índices y resúmenes, etc.

4.1. Búsqueda de información

Todos los sistemas de recuperación de información permiten realizar diferentes modalidades de busca:

- **Directa**. Se teclean directamente una o varias palabras y se puede interrogar en texto libre o sobre campos individuales.

- **Por medio de índices**. En estos casos, el usuario visualiza un diccionario o índice alfabético de las entradas de todos los campos y selecciona los más adecuados. Se puede buscar por índices de palabras o por frase.

- **Jerarquizada**. La consulta se realiza mediante una estructura jerárquica. A partir de un concepto jerárquico, se pueden localizar no sólo los registros en los que aparece aquel término, sino todos aquéllos en los que figure algún concepto más específico de su campo semántico.

- **A través de código**. Numéricos o alfanuméricos que codifican la clasificación, el idioma o la tipología documental, entre otros.

Así pues, para realizar una busca hay que utilizar un número elevado de conceptos, para lo que hay distintas **herramientas** que construyen una estrategia y relacionan, de manera clara, los diferentes términos utilizados en la busca:

- **Operadores lógicos o booleanos**. Permiten la combinación de conceptos en una misma busca, tanto si es la unión (OR) como el resto (AND NOT), la intersección (AND) o la operación contraria (XOR).

- **Operadores sintácticos de proximidad**. Presencia en la misma frase, el mismo párrafo, aparición en un orden determinado o con una separación mínima, etc. Por ejemplo, NEAR.

- **Refinamiento**. Incluso, en algunos casos, el sistema permite refinar las buscas, aplicando una nueva al conjunto de registros encontrados en la busca anterior.

4.2. Categorías de BD según la información que contienen

Los registros de estas bases de datos pueden incluir o no el contenido completo de los documentos que describen, según el cual se describen tres categorías:

- **Bases de datos de texto completo**. Constituidas por los mismos documentos en formato electrónico (digital). Pueden incorporar, además, campos con información complementaria para facilitar su descripción y acceso. Permiten localizar términos presentes en el texto del documento.

- **Archivos electrónicos de imágenes**. Constituidos por referencias que permiten un enlace directo con la imagen del documento original. En estos casos, la busca está limitada a los campos de la referencia bibliográfica, pero no se pueden localizar términos presentes en el texto del documento original.

- **Bases de datos referenciales**. Son aquéllas en las que los registros no contienen el texto original sino sólo la información fundamental con el fin de describir y permitir la localización de documentos imprimidos, sonoros, iconográficos, etc. En estos casos sólo se pueden obtener referencias sobre documentos, que debemos localizar después en otro servicio (archivo, biblioteca, etc.). También pueden incluir campos que faciliten la localización del documento y/o enlaces directos para obtener directamente el original por medio de otro programa.

4.3. Tipología de BD según el modo de acceso

En función del tipo de acceso permitido a la base de datos, también se pueden clasificar en tres categorías: de acceso local, en dispositivo físico (CD-ROM, DVD, etc.) o en línea.

- **Bases de datos de acceso local**. Con el fin de consultarlas, hay que ir al centro productor, tanto a su biblioteca como al centro de documentación donde están almacenadas.

- **Bases de datos en dispositivos de almacenamiento**. Se adquieren, habitualmente, por compra o suscripción y están almacenadas, físicamente, en dispositivos como los CD-ROM o DVD.

- **Bases de datos "en línea"**. Se pueden consultar desde cualquier ordenador, ya que son accesibles "por Internet".

Para saber más

En el ítem sobre BD distribuidas del subapartado "Tipo de bases de datos", podéis ver las ventajas del acceso a BD en línea por medio de Internet y una descripción de la tecnología que se utiliza.

Una misma base de datos puede tener diferentes tipos de acceso.

Sin embargo, es posible que la actualización de ésta sea diferente en cada caso. Por ejemplo, una base de datos en CD-ROM se actualizará cada x meses o años, para volver a ser distribuida, mientras que una base de datos "en línea" puede estar actualizada en tiempo real.

4.4. Tipología de BD según la cobertura documental

Otra clasificación de las bases de datos documentales es la que se establece en función de los tipos de documentos que almacenan, centradas tanto en un único tipo de documento como en varios tipos.

- **Centradas en un único tipo de documento**. Recopilan y permiten la localización de un tipo de documento en concreto como, por ejemplo, patentes, tesis doctorales, informes, artículos de revista, etc.

- **En varios tipos de documentos**. Su objetivo es dar información sobre una disciplina e incorpora, por lo tanto, diferentes tipologías documentales. Se centran en una temática específica.

4.5. Tipología de BD según el modelo de tratamiento documental

Finalmente, una última clasificación de las BDD se realiza según el modelo de tratamiento documental, es decir, según las "operaciones" sobre la información del documento que el productor ha llevado a cabo.

- **Bases de datos de sumarios** o sin análisis de contenido. Se componen de referencias bibliográficas sencillas en las que el productor se limita a grabar los datos de la misma fuente y no realiza ningún análisis del documento. Sólo incorporan los datos descriptivos con el fin de localizar el documento, como, por ejemplo, el autor, el título y los datos de la fuente. En estas bases de datos, la busca por materias sólo se puede llevar a cabo por medio de las palabras del título del documento (normalmente artículos).

- **Catálogos de bibliotecas**. Estas bases de datos corresponden a fondos contenidos en una biblioteca o en una red de bibliotecas. Tienen una gran homogeneidad.

- **BD con análisis documental completo**. Sistemas de información que incorporan un número mayor de puntos de acceso con el fin de facilitar la localización por materias. Cada registro bibliográfico incluye un resumen del contenido del documento original y/o un conjunto de conceptos o términos representativos de los temas tratados. En este grupo de bases de datos se puede distinguir entre las que contienen clasificación y resúmenes, las que tienen una clasificación e indexación por descriptores o palabras clave, y las que disponen de una clasificación, una indexación y resúmenes.

- **Índices de citas**. Sistemas de información en los que además de extraer datos de descripción de los documentos, también se tratan las referencias bibliográficas citadas en los artículos de las revistas científicas.

Descriptor

Cada uno de los términos (o expresiones) escogidos entre un conjunto para representar o etiquetar (en calidad de término preferido) un concepto susceptible de aparecer con cierta frecuencia en los documentos indexables, y de ser utilizado en las consultas. Es la unidad mínima de significación que integra un *thesaurus*; el término por el que serán indexados y recuperados los documentos referidos a su temática.

Tesaurus (del latín *thesaurus*)

Lista alfabética de términos (palabras y expresiones) utilizados para representar los conceptos o temas de los contenidos de los documentos, con objeto de efectuar una normalización terminológica que permita un mejor acceso a los mismos.

Es un intermediario entre el lenguaje natural de los documentos y el lenguaje controlado usado por los especialistas de un determinado campo de conocimiento.

Los términos que conforman un *thesaurus* se interrelacionan entre sí bajo tres modalidades de relaciones:

- Jerárquicas: establecen subdivisiones que reflejan estructuras del tipo todo/parte.
- De equivalencia: controlan la sinonimia, homonimia, antonimia y polinimia entre los términos.
- Asociativas: mejoran las estrategias de recuperación y ayudan a reducir la multiplicidad de jerarquías entre términos.

4.6. Resumen

Las **bases de datos documentales** (BDD) se caracterizan por que cada registro se corresponde con un documento de cualquier tipo: publicación, documento gráfico o sonoro o a su referencia.

Almacena las referencias y/o el texto completo de documentos de propósito general que pueden contener texto extenso y datos multimedia que se consultan por medio de palabras clave (que aparecen en el texto) o de un *thesaurus* de estructura jerarquizada.

La información contenida en una BDD se estructura en diferentes campos para facilitar su control y acceso individualizado. Algunos campos se refieren a la descripción formal del documento y otros tratan sobre su contenido temático; incluso, pueden guardar el propio documento.

Se pueden clasificar en diferentes categorías:

- **Según la información que contienen**: BD referenciales (contienen información descriptiva y referencias que permiten localizar los documentos originales), BD de texto completo (guardan el contenido completo de los documentos originales) o archivos electrónicos de imágenes (contienen referencias que tienen un enlace en la imagen del documento original).
- **Según el modo de acceso a la información**: BD de acceso local, BD en dispositivos físicos de almacenaje y BD en línea.
- **Según la cobertura documental**: BD centradas en un único tipo de documento y BDcon diferentes tipos de documentos (centrado en una disciplina).

- **Según el modelo de tratamiento documental**: BD de sumarios (o sin análisis de contenido), BD con análisis documental completo, catálogos de bibliotecas e índices de citas.

Ejercicios de autoevaluación

Seleccionad para cada pregunta la respuesta adecuada.

1. Si existen las memorias (RAM y ROM), ¿por qué son necesarios los dispositivos para almacenar datos?

a) Porque la memoria RAM es volátil.
b) Porque la memoria ROM es permanente, no se puede cambiar (al menos no se puede cambiar de una manera sencilla y rápida).
c) Porque la RAM es muy cara y de capacidad limitada. Debemos contar con dispositivos con gran capacidad de almacenar datos, aunque paguemos el precio de una velocidad menor.
d) Todas son ciertas.

2. Señalad la única de las cuatro afirmaciones que es cierta:

a) La memoria ROM sólo se borra cuando se desenchufa el ordenador de la corriente eléctrica.
b) En la memoria RAM se puede leer y escribir.
c) Los dispositivos de almacenamiento de datos no son necesarios si el ordenador cuenta con memoria RAM.
d) Todas las afirmaciones anteriores son falsas.

3. Indicad cuál de las afirmaciones siguientes es verdadera:

a) La memoria ROM es sólo de lectura y de escritura bipolar.
b) La información almacenada en la memoria ROM se pierde si después de apagar el ordenador se desenchufa de la corriente eléctrica.
c) En la memoria RAM se puede leer y escribir información y perdura incluso si apagamos el ordenador.
d) Ninguna de las anteriores es cierta.

4. Indicad cuál de las afirmaciones siguientes sobre los dispositivos de almacenamiento de datos es cierta:

a) Permiten un volumen mayor de almacenaje de datos y con más rapidez que el ordenador.
b) Los diferentes dispositivos de almacenamiento existentes ofrecen diferentes prestaciones y capacidades de almacenaje.
c) Pierden su información cuando se desenchufan de la corriente eléctrica.
d) Permiten únicamente la escritura de datos, ya que la lectura se debe efectuar por medio de la CPU.

5. Indicad cuáles de las afirmaciones siguientes son falsas.

a) En los soportes secuenciales para acceder al registro **n** se deben leer necesariamente los **n – 1** anteriores.
b) A los soportes de acceso directo no se puede acceder directamente a menos que se conozca la dirección física.
c) Dentro de un fichero los registros se identifican por un único campo que es la clave.
d) La memoria intermedia es un espacio de la memoria principal que se utiliza en las operaciones de lectura y escritura en los ficheros.

6. Indicad cuál de las afirmaciones siguientes es cierta:

a) Los ficheros permanentes son los que, independientemente del tipo de soporte en el que estén, no desaparecen al apagar el ordenador.
b) Los ficheros permanentes son los que no sufren modificaciones en su contenido a lo largo del tiempo.
c) Los ficheros temporales sólo son necesarios para operar con la memoria principal.
d) Los ficheros de maniobras permiten ejecutar aplicaciones en las que los requerimientos de datos superan la capacidad de la memoria principal.

7. Indicad cuál es la única afirmación verdadera entre las cuatro siguientes:

a) En la operación de fusión de ficheros se unen dos o más ficheros para integrarlos en uno solo.
b) La operación de busca sobre un fichero consiste en llevar a cabo los pasos necesarios para encontrar el archivo dentro del disco duro del PC.
c) La eliminación de un registro es una operación de destrucción de ficheros.

d) Todas las afirmaciones anteriores son falsas.

8. Seleccionad la única afirmación verdadera entre las cuatro propuestas siguientes:

a) Hay dos tipos de acceso a los ficheros: secuencial directo y directo automático.
b) Exclusivamente, hay dos tipos de organización de ficheros: relativo y secuencial.
c) Hay cuatro tipos de organización de ficheros: secuencial, secuencial indexada, secuencial encadenada y rotativa.
d) Todas las afirmaciones anteriores son falsas.

9. Indicad cuál de las afirmaciones siguientes sobre ficheros es verdadera:

a) Un fichero se compone de campos que a su vez están compuestos de registros.
b) La información contenida en un fichero está desestructurada (de aquí la necesidad de las bases de datos).
c) El registro físico se corresponde con la cantidad de información que se puede leer y escribir al mismo tiempo en soporte físico.
d) Todas las anteriores son ciertas.

10. Indicad qué afirmación de las cuatro siguientes es verdadera:

a) Las bases de datos son una colección de información de tipo exclusivamente informático.
b) Las bases de datos evitan sin excepciones las redundancias de información.
c) Un sistema gestor de bases de datos (SGBD) accede a la información desde un lenguaje de alto nivel denominado "Lenguaje de manipulación de datos" (DML).
d) Todas las afirmaciones anteriores son falsas.

11. Indicad qué afirmación de las cuatro siguientes es verdadera:

a) Las bases de datos aseguran que no habrá redundancia en los datos que almacenan en ningún caso.
b) En algunos casos las bases de datos permiten redundancias con el fin de mejorar el espacio que necesitan para almacenarse.
c) Algunas veces las bases de datos presentan redundancias con el fin de mejorar los accesos y el rendimiento.
d) Todas las afirmaciones anteriores son falsas.

12. Seleccionad la afirmación falsa entre las cuatro siguientes:

a) Una base de datos es información almacenada de manera organizada en un ordenador.
b) La información contenida en una base de datos puede ser de cualquier tipo.
c) La utilización de una base de datos asegura la eliminación de las inconsistencias de datos.
d) La utilización de una base de datos presenta más ventajas que la utilización de un sistema de ficheros.

13. Indicad cuál de las afirmaciones siguientes es verdadera:

a) Los datos almacenados en una base de datos no presentan redundancias.
b) Un buen sistema de base de datos debe permitir realizar cambios sobre el soporte físico de la base de datos sin que por ello se vean afectados los programas que la utilizan.
c) Las bases de datos textuales son más modernas y más utilizadas que las jerárquicas y que las relacionales.
d) Todas las anteriores son falsas.

14. Indicad cuál de las afirmaciones siguientes es cierta:

a) Las relaciones entre diferentes tablas de una base de datos relacional se implementan mediante las claves internacionales.
b) Una tabla almacena toda la información de cualquier tipo que ha de contener la base de datos.
c) Para cada tabla debe haber una clave primaria que es uno y sólo uno de sus campos que no repite un valor en la tabla.
d) Todas las anteriores son falsas.

15. ¿Cuál de las afirmaciones siguientes es cierta?

a) Cuando la relación entre dos tablas es de uno a muchos es necesario crear una tercera tabla con el fin de poder representar el modelo físico.
b) En el modelo conceptual una entidad está formada por una serie de atributos.
c) En la generación del modelo físico a partir del conceptual toda relación 1:n se convierte en una clave foránea en la tabla correspondiente de grado 1.

d) Todas las anteriores son falsas.

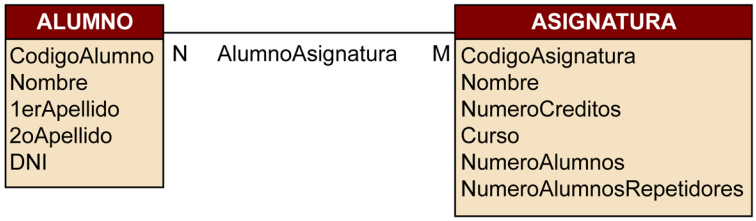
16. Indicad la única afirmación que es verdadera entre las cuatro siguientes:

a) El grado de una relación puede ser de una a una (1:1) o de una a dos (1:2).
b) El grado de una relación puede ser de una a muchas (1:n) o de dos a muchas (2:n).
c) El grado de una relación ternaria es siempre de una a tres (1:3).
d) Todas las afirmaciones anteriores son falsas.

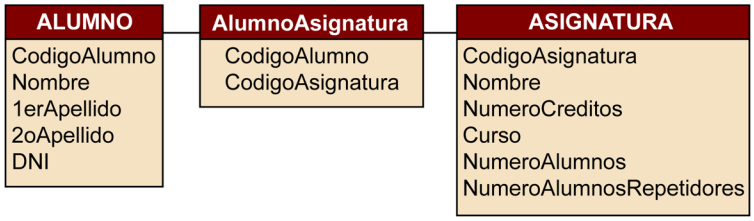
17. ¿Qué afirmación es verdadera?

a) Un atributo de una entidad en el modelo conceptual se convierte en un campo de una tabla en el modelo físico.
b) La relación n:1 indica que para un determinado registro de la primera tabla puede haber *n* registros en la segunda entidad que contengan el mismo atributo que es clave primaria en la primera tabla.
c) Las respuestas *a* y *b* son ciertas.
d) Todas son falsas.

18. En el modelo conceptual de una base de datos, se observan dos entidades con una relación *n:m*. Según la figura, cada alumno puede cursar unas o más asignaturas, y en cada asignatura, cursan uno o más alumnos. Los atributos que forman claves primarias aparecen subrayados.



Si se pasa al modelo relacional...



... indicad cuál de las siguientes afirmaciones referida a la tabla AlumnoAsignatura es falsa.

a) Se puede evitar que aparezca la tabla de relación AlumnoAsignatura que no existía en el modelo conceptual.
b) La relación entre ALUMNO y AlumnoAsignatura será *1:n*.
c) La clave primaria de la tabla ALUMNO, junto con la clave primaria de la tabla ASIGNATURA, forman la clave primaria de la tabla de relación AlumnoAsignatura.
d) Todas las afirmaciones anteriores son falsas.

19. Diseño de una base de datos

Os pedimos que diseñéis una pequeña base de datos que permita gestionar las actividades que se organizan en los diferentes espacios compartidos (sala de conferencias y aulas de formación) y las asociaciones que participan en un local de entidades.

El espacio dispone de tres espacios compartidos:

- la sala de conferencias,
- el aula 1, destinada a acciones de formación general,
- el aula 2, equipada con equipos informáticos, para poder llevar actuaciones de formación sobre las tecnologías de la información.

Aun así, estos espacios se pueden ampliar con el tiempo.

Se prevé disponer, por ejemplo, a medio plazo, de una tercera aula de formación y de una sala de exposiciones.

Estos espacios los pueden utilizar para las diferentes actividades que se organizan, tanto directamente por el local, como por asociaciones de la comarca.

Así pues, es necesario que la base de datos permita almacenar lo siguiente:

a) Lista de asociaciones, con sus principales datos (NIF, nombre entidad, responsable, dirección, teléfono de contacto, correo electrónico, web, etc.).
b) Relación de actividades organizadas (nombre actividad, responsable, fecha de inicio y fecha de finalización, asociación organizadora, etc.).
c) Espacios disponibles (identificador espacio, aforo máximo, etc.).
d) El uso de los espacios, por parte de las actividades, indicando los días y horarios.
e) Y la cesión, por otra parte, de los espacios a las asociaciones.

En el caso *d*, hay que considerar que una actividad puede utilizar más de un espacio, en diferentes días. Por ejemplo, un curso de gestión de entidades, que dura tres sábados y se lleva a cabo durante las mañanas, utiliza el primero y el segundo día el aula 1 y el tercer día, el aula 2, ya que se explican herramientas de apoyo a la gestión de entidades. En este caso, además, esta actividad no la organiza ninguna entidad, sino el mismo local.

Siempre hay que considerar el horario de uso, mañana o tarde.

Finalmente, en el caso *e* hay situaciones en las que una asociación pide el uso de un espacio para un día (horario de mañana o de tarde) con el fin de llevar a cabo alguna actuación interna o actividad que no se registra como tal.

En este caso, se habla de una cesión del espacio.

Por ejemplo, la Asociación Joven Cambra puede haber solicitado la cesión de la sala de conferencias para un viernes por la tarde, con el fin de hacer su asamblea general.

En estos casos, sólo nos interesa saber qué espacio ha sido cedido a la Asociación y para qué día y qué horario, pero no registramos la actividad.

Se pide, pues, que realicéis las actividades siguientes:

a) Diseño del **modelo conceptual de la base de datos** (modelo entidad-relación)
Identificación de las entidades, de sus atributos y de las interrelaciones entre ellas, con los atributos, si fuera el caso.
b) Generación del **modelo relacional de la base de datos**. Partiendo del modelo conceptual diseñado en el apartado 1, se debe convertir en una base de datos relacional, indicando las tablas correspondientes, los campos de cada una y su tipo, la clave primaria de cada tipo y las claves foráneas correspondientes.

Solucionario

Ejercicios de autoevaluación

1. La respuesta correcta es *d*. Todas las opciones son ciertas.

2. La respuesta correcta es *b*. En la memoria RAM se puede leer y escribir.

3. La respuesta correcta es *d*. Ninguna de las opciones anteriores es cierta.

4. La respuesta correcta es *b*. Los diferentes dispositivos de almacenamiento existentes ofrecen diferentes prestaciones y capacidades de almacenaje.

5. La respuesta correcta es la *c*. Dentro de un fichero, los registros se identifican por un único campo que es la clave.

6. La respuesta correcta es la *d*. Los ficheros de maniobras permiten ejecutar aplicaciones en las que los requerimientos de datos superan la capacidad de la memoria principal.

7. La respuesta correcta es la *a*. En la operación de fusión de ficheros se unen dos o más ficheros para integrarlos en uno solo.

8. La respuesta correcta es la *d*. Todas las afirmaciones anteriores son falsas.

9. La respuesta correcta es la *c*. El registro físico se corresponde con la cantidad de información que se puede leer y escribir al mismo tiempo en soporte físico.

10. La respuesta correcta es la *c*. Un sistema gestor de bases de datos (SGBD) accede a la información desde un lenguaje de alto nivel denominado *lenguaje de manipulación de datos* (DML).

11. La respuesta correcta es la *c*. Algunas veces las bases de datos presentan redundancias con el fin de mejorar los accesos y el rendimiento.

12. La respuesta correcta es la *c*. La utilización de una base de datos asegura la eliminación de las inconsistencias de datos.

13. La respuesta correcta es la *b*. Un buen sistema de base de datos debe permitir realizar cambios sobre el soporte físico de la base de datos sin que por ello se vean afectados los programas que la utilizan.

14. La respuesta correcta es la *d*. Todas las anteriores son falsas.

15. La respuesta correcta es la *b*. En el modelo conceptual, una entidad está formada por una serie de atributos.

16. La respuesta correcta es la *d*. Todas las afirmaciones anteriores son falsas.

17. La respuesta correcta es la *c*. Las opciones *a* y *b* son ciertas.

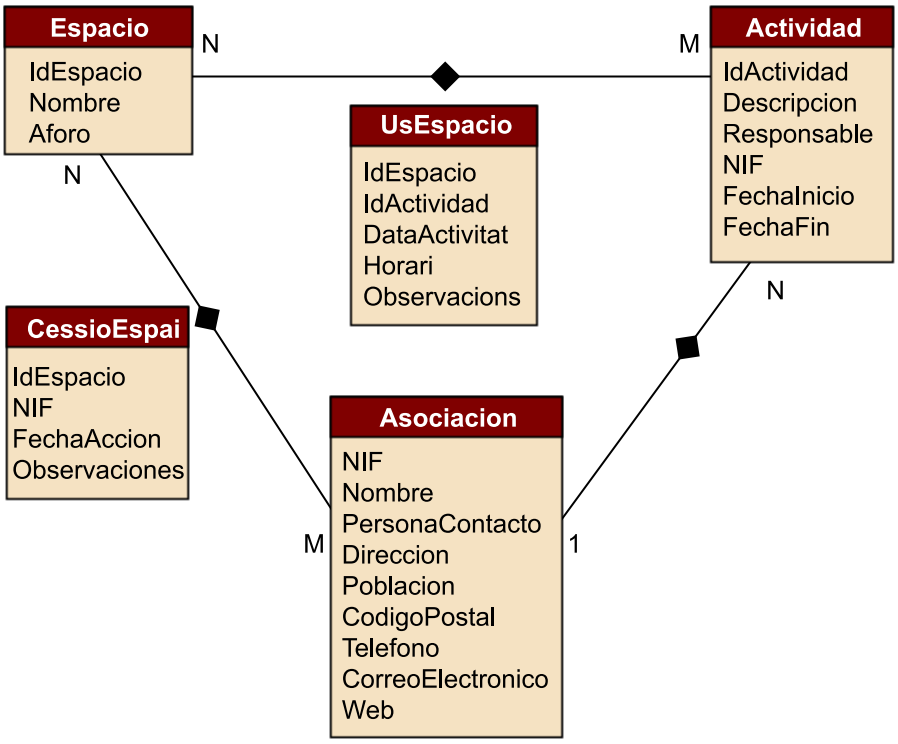
18. La respuesta correcta es la *a*. Se puede evitar que aparezca la tabla de relación "Alumno-Asignatura" que no teníamos en el modelo conceptual.

19. Diseño de una base de datos

Este ejercicio no tiene una única solución, ya que dependiendo del diseño del modelo conceptual, la base de datos relacional final será una u otra.

En una situación real, durante el diseño de la base de datos, el responsable solicita diferentes requerimientos al "cliente" para ir determinando posibles ambigüedades y dudas. En vuestro caso, el consultor de la asignatura actúa como cliente.

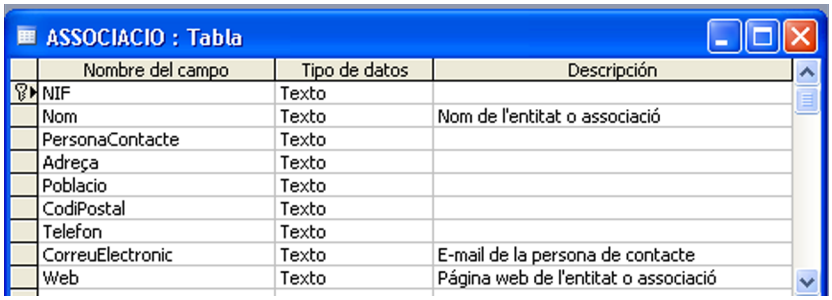
a) Diseño del modelo conceptual (entidades e interrelaciones, con sus atributos)



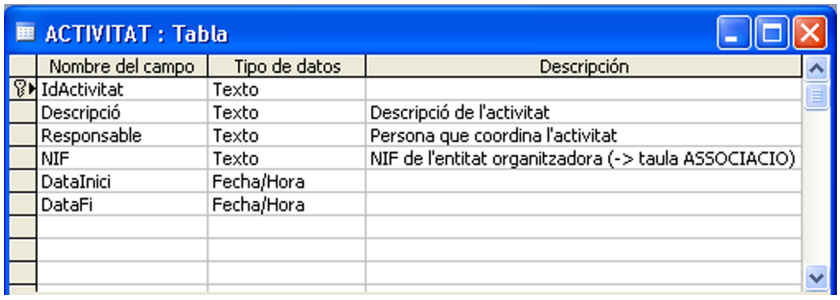
Las entidades se muestran en rectángulos con su nombre en la cabecera y los atributos a continuación. Las relaciones entre entidades se muestran con un rombo junto al que aparece su nombre, seguido de sus atributos. La cardinalidad de las relaciones se indica mediante 1, N o M al lado de las entidades que participan. Los atributos que forman clave primaria están subrayados.

b) Generación del modelo relacional (con MS Access)

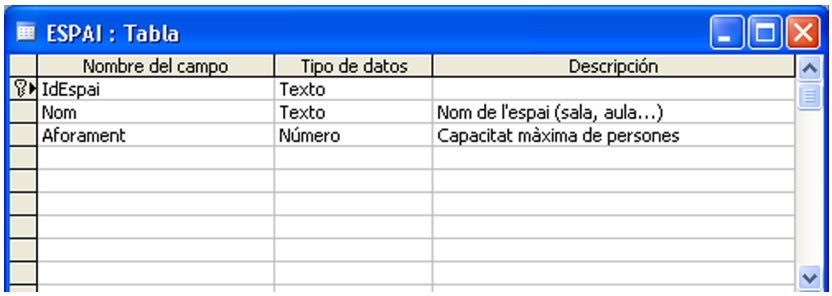
- Diseño de las tablas correspondientes a las entidades del modelo conceptual: ASOCIACIÓN, ACTIVIDAD y ESPACIO. Cada tabla incluye la definición de los campos, los tipos de datos de cada uno de ellos y la identificación de su clave primaria (en Access, los campos que la forman se distinguen con el símbolo de una clave).



Nombre del campo	Tipo de datos	Descripción
NIF	Texto	
Nom	Texto	Nom de l'entitat o associació
PersonaContacte	Texto	
Adreça	Texto	
Poblacio	Texto	
CodiPostal	Texto	
Telefon	Texto	
CorreuElectronic	Texto	E-mail de la persona de contacte
Web	Texto	Página web de l'entitat o associació



Nombre del campo	Tipo de datos	Descripción
IdActivitat	Texto	
Descripció	Texto	Descripció de l'activitat
Responsable	Texto	Persona que coordina l'activitat
NIF	Texto	NIF de l'entitat organitzadora (-> taula ASSOCIACIO)
DataInici	Fecha/Hora	
DataFi	Fecha/Hora	



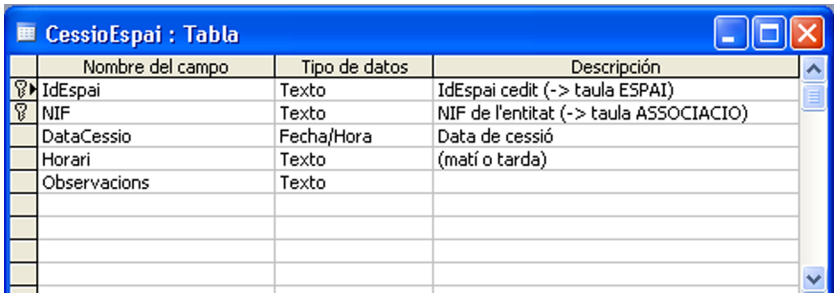
ESPAI : Tabla

Nombre del campo	Tipo de datos	Descripción
IdEspai	Texto	
Nom	Texto	Nom de l'espai (sala, aula...)
Aforament	Número	Capacitat màxima de persones

- Diseño de las tablas intermedias generadas a partir de las relaciones M:N del modelo conceptual: UsEspacio y CessioEspacio.

UsEspai : Tabla

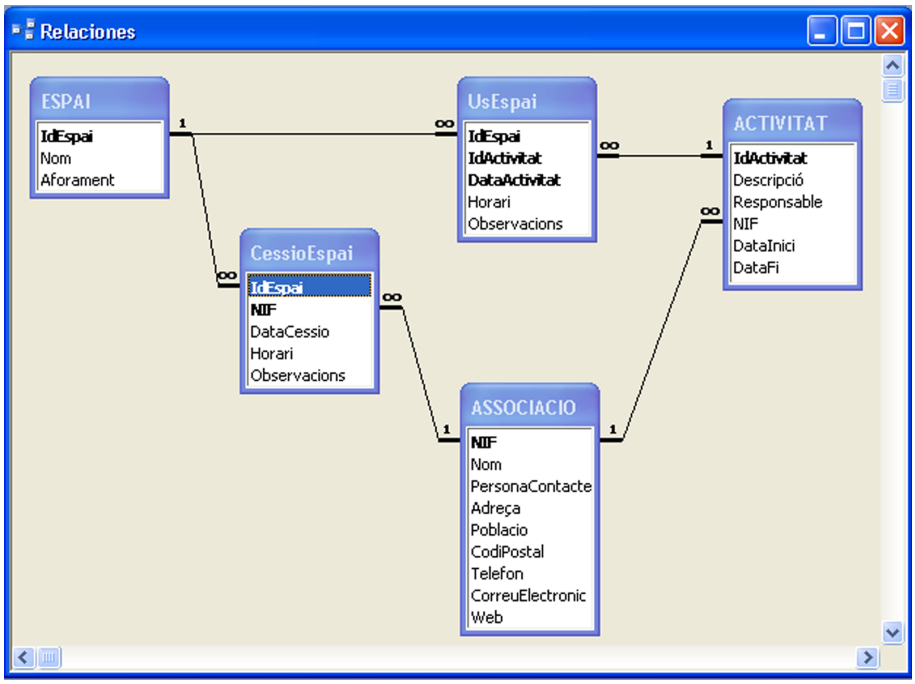
Nombre del campo	Tipo de datos	Descripción
IdEspai	Texto	IdEspai utilitzat (-> taula ESPAI)
IdActivitat	Texto	IdActivitat realitzada (-> taula ACTIVITAT)
DataActivitat	Fecha/Hora	Data d'utilització de l'espai
Horari	Texto	(matí o tarda)
Observacions	Texto	



CessioEspai : Tabla

Nombre del campo	Tipo de datos	Descripción
IdEspai	Texto	IdEspai cedit (-> taula ESPAI)
NIF	Texto	NIF de l'entitat (-> taula ASSOCIACIO)
DataCessio	Fecha/Hora	Data de cessió
Horari	Texto	(matí o tarda)
Observacions	Texto	

- La relación entre las entidades ASOCIACIÓN y ACTIVIDAD no requiere de una tabla intermedia, ya que es de tipo uno a muchos (*1:n*). Es decir, una asociación puede organizar distintas actividades, pero una actividad sólo la organiza una asociación.
- Implementación de las relaciones entre las tablas de la BD.



Los conjuntos de campos que forman las claves primarias se distinguen en negrita.

Los campos que son clave foránea aparecen junto al símbolo de infinito (¥).

Bibliografía

Silberschatz, A.; Korth, H.L.; Sudarsahn, S. (1998). *Fonaments de Bases de Dades (3.)*. Madrid: McGraw - Hill.

Ullman, J. (1991). *Principles of Database and knowledge - Base Systems*. Computer Science Press.

de Miguel, A.; Piattini, M. (1999). *Fonaments i models de bases de dades (2)*. Madrid : RA-DT.

Presuman, Roger S. (1997). *Ingeniería de Software. Un enfoque práctico*. Madrid: Mc Graw Hill