

BASE DE DATOS

# Sistemas de representación de la información. Ficheros

---

# ÍNDICE

/ 1. Introducción y contextualización práctica	3
/ 2. Sistemas de representación de la información	4
<b>/ 3. Ficheros</b>	<b>5</b>
3.1. Concepto de fichero	5
3.2. Tipos de ficheros según su uso	5
3.3. Tipos de ficheros según su organización	7
3.4. Tipos de ficheros según su contenido	8
<b>/ 4. Sistemas de almacenamiento de la información</b>	<b>9</b>
4.1. Concepto	9
4.2. Sistemas lógicos de almacenamiento	9
4.3. Tipos de sistemas de almacenamiento	10
4.4. Gestión de los sistemas de almacenamiento	11
<b>/ 5. Caso práctico 2: “Gestión de tienda y almacén”</b>	<b>11</b>
<b>/ 6. Resumen y resolución del caso práctico de la unidad</b>	<b>12</b>
<b>/ 7. Bibliografía</b>	<b>13</b>

# OBJETIVOS



*Comprender la importancia de la información y de su correcta gestión.*

*Conocer el concepto de «fichero» e identificar los diferentes tipos.*

*Entender la estructura organizativa de cualquier sistema de información.*

*Identificar los distintos tipos de sistemas de almacenamiento de la información.*

*En su conjunto, interpretar los fundamentos sobre las que se construyen las bases de datos en la actualidad.*



## / 1. Introducción y contextualización práctica

Hoy en día, gran parte de lo que hacemos en nuestra vida cotidiana está intrínsecamente ligado con el uso de sistemas de almacenamiento y gestión de la información. Cuando solicitamos cita en el médico, accedemos a nuestras redes sociales, o incluso cuando hacemos una llamada telefónica con nuestro smartphone estamos utilizando este tipo de sistemas.

La información (datos) probablemente sea el elemento más valioso que puede tener una organización actualmente. Nuestro mundo actual gira en torno al dato: comunicación con otras personas, interesarnos por viajes a otros lugares, buscar restaurantes en los que comer, movernos por nuestra ciudad a través del transporte público, leer el periódico... continuamente estamos generando y demandando información, que habitualmente está contenida en bases de datos de todo tipo.

Es por tanto fundamental entender, por un lado, la importancia de una correcta gestión y almacenamiento de esta información, y, por otro lado, los pilares principales sobre los que se construyen las bases de datos.

En este tema sentaremos las bases de cualquier sistema de información, analizando diversos conceptos básicos que es necesario entender y asimilar previamente.

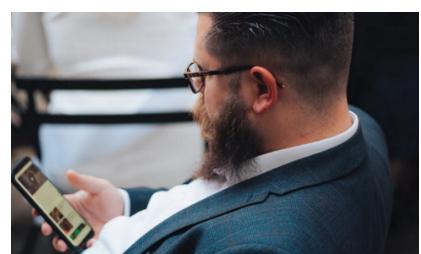


Fig. 1 El smartphone es una herramienta cotidiana que nos comunica con el mundo, permitiéndonos consultar y compartir información.



Audio intro. Manejar y gestionar la información correctamente es fundamental.

<https://bit.ly/2Uuzu4Y>





## / 2. Sistemas de representación de la información

Todos sabemos que un ordenador puede almacenar y procesar muchos tipos de datos distintos: textos, imágenes, vídeos, números, etc. Pero ¿de qué manera se manejan tantos tipos distintos? la respuesta es, a priori, sencilla: cualquier dato que se introduce en un sistema informático debe traducirse a un tipo concreto que pueda entender y manejar la máquina a bajo nivel. Este tipo concreto que es la base de cualquier sistema informático es el *bit*.

El *bit* sería la información de menor tamaño que puede manejar cualquier equipo digital, constituyendo la base sobre la que se construye el sistema.

Sin embargo, no se pueden representar todos los datos sólo con el bit 0 y 1, es evidente que son necesarias estructuras más grandes que permitan representar diferentes tipos de datos. Para ello, se utiliza un “patrón de bits”, que no es más que un grupo de *bits* consecutivos. Este grupo de bits será interpretado por los dispositivos o programas para representar la información adecuada.

Por tanto, para que un sistema informático pueda realizar operaciones, debe traducir (codificar) cualquier instrucción que reciba al sistema binario (formado por 0 y 1), de manera que puedan ser interpretadas por la máquina.

Para que la información pueda ser interpretada por el ser humano, existen dispositivos que “traducen” (decodifican) estas secuencias en información que podamos comprender: un texto, un sonido, vídeos, etc.

De esta forma, podemos distinguir diferentes múltiplos de bits:

UNIDAD	SÍMBOLO	EQUIVALENTE
Bit	Bit	0 ó 1
Byte	B	8 bits
Kilobyte	kB	1024 bytes
Megabyte	MB	1024 Kb
Gigabyte	GB	1024 Mb
Terabyte	TB	1024 Gb
Petabyte	PB	1024 Tb
Exabyte	EB	1024 Pb

Tabla 1. Múltiplos de bit.

Los *bits* se agrupan para formar números, caracteres, datos... o ficheros. De esta forma, los archivos de imagen (jpg, bmp, png, etc), vídeo (mp4, mpg, mov, etc), ejecutables (exe, com, etc.) o de muchos otros tipos se conocen como ficheros binarios, es decir, son ficheros que requieren de un formato determinado para poder ser interpretados por el sistema.

De manera general, las bases de datos están formadas por ficheros de este tipo, que cuentan con un formato determinado en función del motor que utilicen en cada caso (Oracle, MySQL o Access).



## / 3. Ficheros

### 3.1. Concepto de fichero

En un sistema informático se pueden almacenar muchas clases de información (datos, música, vídeos, etc.), que estará contenida en los diferentes dispositivos de almacenamiento que existen. Los ficheros se utilizan para poder gestionar toda esa información, siendo estructuras que permiten almacenar datos de distintos tipos.

**Un fichero se puede definir como una secuencia de bits que contiene información relacionada con un asunto concreto**, de forma estructurada y organizada, y que están ubicados en dispositivos de almacenamiento. También se conocen habitualmente como “archivos”, debiendo contar con un nombre y una extensión.

Los datos que contienen los ficheros están pensados para que se pueda actuar sobre ellos de alguna manera: consultar, modificar, suprimir, etc.

Un fichero también puede entenderse como un conjunto de registros lógicos similares. Estos registros lógicos, a su vez, están formados por atributos, como se puede comprobar en la siguiente imagen:

The diagram illustrates a file structure. At the top, the word "Atributos" (Attributes) has two blue arrows pointing down to the first three columns of a table. To the right of the table, another blue arrow points right to the word "Registros" (Records). The table has a header row with columns labeled "Nº", "TIPO", and "COLOR". Below the header are three data rows:

Nº	TIPO	COLOR
1	Coche	Rojo
2	Moto	Negro
3	Camión	Blanco

Fig. 2. Ejemplo de Fichero y campos asociados.

Podemos distinguir entre diferentes tipos de ficheros. En nuestro caso los clasificaremos según su uso, según cómo se organicen y según su contenido.

### 3.2. Tipos de ficheros según su uso

Si clasificásemos los ficheros en función de para qué vayan a ser utilizados, podemos distinguir dos opciones principales:



Fig. 3. Clasificación de ficheros según su uso.



### 3.2.1. Ficheros permanentes.

Como su propio nombre indica, están destinados a mantenerse en el sistema ya que contienen información de vital importancia para la aplicación. Dentro de ellos podemos distinguir:

- **Ficheros maestros:** Contienen información relativa al estado actual de datos concretos, que pueden ser modificados. Es decir, indican el estado de una organización o empresa en un instante determinado, en relación con algún asunto de importancia para la organización.

Por ejemplo, podría ser el fichero con los usuarios de una determinada plataforma, que sufrirá modificaciones debido a las altas y bajas de usuarios.

- **Ficheros históricos:** Suelen construirse a partir de los maestros, ya que contienen información antigua, con todos los movimientos que han podido ocurrir sobre el asunto concreto para el que se utilice el fichero.

Un ejemplo de fichero histórico podría ser los libros de contabilidad de una compañía, que contendrán información sobre su facturación desde la creación de la misma.

- **Fichero de constantes:** Como su propio nombre indica, la información que contienen varía muy poco a lo largo del tiempo.

Por ejemplo, un fichero de constantes podría ser el que contiene los datos personales de los trabajadores de una empresa determinada.

Provincia	Población	Código Postal	lat	lon
Ciudad Real	Arroba de los Montes	13193	-4.543.405.160	39.153.483.150
Ciudad Real	Ballesteros de Calatrava	13432	-3.944.291.080	38.832.773.180
Ciudad Real	Bolaños de Calatrava	13260	-3.666.163.430	38.905.878.650
Ciudad Real	Brazatorras	13449	-4.292.527.750	38.658.227.580
Ciudad Real	Brazatorras	13450	-4.292.527.750	38.658.227.580
Ciudad Real	Cabezarados	13192	-4.297.622.270	38.844.702.910
Ciudad Real	Cabezarrubias del Puerto	13591	-4.183.781.480	38.614.771.350
Ciudad Real	Calzada de Calatrava	13370	-3.776.998.700	38.704.690.550
Ciudad Real	Calzada de Calatrava	13739	-3.776.998.700	38.704.690.550
Ciudad Real	Calzada de Calatrava	13779	-3.776.998.700	38.704.690.550
Ciudad Real	Campo de Criptana	13610	-3.126.142.790	39.405.568.800

Fig. 4. Ejemplo de fichero de constantes.

### 3.2.2. Ficheros temporales.

Se generan a partir de los permanentes, y su durabilidad es más reducida, ya que guardan información necesaria durante un periodo de tiempo, que posteriormente puede ser desecharada.

- **Ficheros de movimientos:** De manera habitual contienen información sobre las modificaciones realizadas en el fichero maestro, ya sea modificar registros existentes, crear nuevos o eliminar alguno.

Por tanto, cuando se actualiza el fichero maestro con la información que contiene el fichero de movimientos, este último ya puede desecharse.

- **Ficheros de trabajo o de maniobra:** Estos archivos contienen información sobre datos que no pueden almacenarse en memoria por falta de espacio. Es decir, se utilizan como ficheros intermedios, para guardar datos de manera temporal en sistemas o aplicaciones que disponen de poca memoria.

### 3.2.3. Caso práctico 1: “Gestión de tienda y almacén”.

**Planteamiento:** Un amigo de la infancia ha abierto una tienda de productos de informática con la que está muy ilusionado. Se le da muy bien la parte comercial, pero no tiene mucho conocimiento sobre aplicaciones informáticas. Sabes que somos unos grandes profesionales en este campo, por lo que nos ha pedido ayuda para la gestión de su negocio con el fin de controlar el stock del que dispone en su almacén.



**Nudo:** Una solución rápida (mientras construimos una base de datos potente y robusta) sería construir un fichero maestro que contenga todo el stock del almacén. Además, generaremos dos ficheros más de movimientos: uno para el personal de tienda (que contendrá las ventas del día) y otro para el de almacén (que contendrá las entradas de nuevo material). Cada tipo de producto tendrá un ID distinto para poder distinguirlo del resto. Debemos exemplificarlo con un caso para poder presentárselo a nuestro amigo y que conozca cómo debe utilizar estos ficheros.

**Desenlace:** Una posible propuesta de diseño es la siguiente. En ella, se ha generado un fichero de stock existente en el almacén. Al final del día, se ha obtenido el fichero de movimientos con las ventas realizadas y el de almacén con las nuevas entradas. Como resultado, se obtiene el fichero maestro de stock modificado.

ID PRODUCTO	DESCRIPCIÓN	UDs
1	Cable CAT6A	5000
2	Conector RJ45 CAT6A	100
3	Panel de 24p RJ45	10

Tabla 2. Archivo maestro de Stock existente.

ID PRODUCTO	DESCRIPCIÓN	UDs
4	Fibra óptica	1000

Tabla 3. Archivo de movimientos del día (entradas a almacén).

ID PRODUCTO	DESCRIPCIÓN	UDs
1	Cable CAT6A	1000
2	Conector RJ45 CAT6A	30
3	Panel de 24p RJ45	2

Tabla 4. Archivo de movimientos del día (ventas realizadas).

ID PRODUCTO	DESCRIPCIÓN	UDs
1	Cable CAT6A	4000
2	Conector RJ45 CAT6A	70
3	Panel de 24p RJ45	8
4	Fibra óptica	1000

Tabla 5. Archivo maestro de Stock resultante.

### 3.3. Tipos de ficheros según su organización

En este caso, distinguimos diferentes tipos de ficheros según la forma en la que se accede a su información.

#### 3.3.1. Ficheros secuenciales.

La información en estos ficheros está dispuesta de manera ordenada y consecutiva. Por tanto, para poder acceder a unos datos concretos es necesario pasar por los todos los anteriores. Cuentan con una marca o hito al final del mismo para indicar que no hay más contenido. Un ejemplo de fichero de este tipo se muestra a continuación:

```
1111AJT // BMW320D // Madrid // Antonio // Garcia // Jiménez //
2222BJT // RENAULT MEGANE //Ciudad Real // Manuel // Jiménez //
Del Rio // 3333CZD // SMART FOR FOUR // Toledo // Laura //
Martínez // Aparicio // 3872DAD // Zaragoza // Ana // López //
Fernández//
```

Fig.5. Ejemplo de fichero secuencial



### 3.3.2. Ficheros de acceso directo.

En este tipo de ficheros los datos están organizados con unos tamaños fijos, de forma que se puede acceder directamente. Al fijarse un tamaño determinado, a veces no se ocupa de manera completa, con lo que se desperdicia espacio en disco. A continuación, se muestra un ejemplo de este tipo de fichero:

1111AJT BMW320D	Madrid	Antonio García	Jiménez
2222BJT RENAULT MEGANE	Ciudad Real	Manuela Jiménez	Del Río
3333CZD SMART FOR FOUR	Toledo	Laura Martínez	Aparicio
3872DAD VOLVO V49	Zaragoza	Ana López	Fernández

Fig. 6. Ejemplo de fichero de acceso directo

### 3.3.3. Ficheros indexados.

En este caso, los datos que contienen los ficheros están organizados en base a un índice, que se utiliza para poder acceder a los mismos. Obviamente, el acceso a la información en este tipo de ficheros es más rápida, pero también más compleja de implementar.

1111AJT → BMW320D	Madrid	Antonio García	Jiménez
2222BJT → RENAULT MEGANE	Ciudad Real	Manuela Jiménez	Del Río
3872DAD → SMART FOR FOUR	Toledo	Laura Martínez	Aparicio
4333CZD → VOLVO V49	Zaragoza	Ana López	Fernández

Fig. 7. Ejemplo de fichero indexado

## 3.4. Tipos de ficheros según su contenido

Desde el punto de vista de las bases de datos, y debido a las características de las de nueva generación, cada vez se atiende más a clasificar los ficheros según su uso o según el tipo del que sean. En este caso, podemos distinguir entre ficheros planos y ficheros binarios:

### 3.4.1. Ficheros planos o de texto.

Se caracterizan por ser entendibles directamente por nosotros, sin necesidad de ser interpretados por parte de ningún software.

Suelen estar construidos mediante código ASCII, que es un estándar de representación basado en asociar un número a cada carácter que se quiera representar, según una tabla de códigos normalizada.

El hecho de que no necesiten ser procesados no implica que no tengan una extensión determinada, con el fin de que el usuario sepa de antemano qué tipo de texto contiene el fichero. De esta forma, existen archivos de texto de diferentes tipos:

- **Ficheros web:** html, php, xml
- **Ficheros de código:** java, js, sql
- **Ficheros de configuración:** ini, conf

```
<!DOCTYPE html>
]<html>
]<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Esta es nuestra ayuda JavaHelp de prueba</title>
-</head>
]<body>

<h1>Esta es una ayuda creada con JavaHelp</h1>
<h2>Esta ayuda la hemos creado a modo de prueba</h2>

]<p>Para ello podemos utilizar un generador de código HTML cualquiera<h2></h2>
]<p>Podemos también añadir imágenes o incluso referenciar
entre si las diferentes páginas de la ayuda<h2></h2>



<p></p>

Este es el enlace al
<a href="tema1.html">Tema 1</a> de ayuda
]<p><a href="tema2.html">Este es el enlace al</a>
| <a href="tema2.html">Tema 2</a> de ayuda
]</p>
-</body>
```

Fig.8. Ejemplo de fichero de texto en HTML



Video 1. "Diferencias entre ficheros planos de texto y ficheros binarios"  
<https://bit.ly/3cSv7qC>





### 3.4.2. Ficheros binarios.

Según se indicó en el punto 2, los datos que contienen típicamente las bases de datos están basados en este tipo de ficheros.

Se caracterizan por estar codificados en binario, por lo que deben ser procesados por alguna aplicación concreta, según su extensión.

Algunos ejemplos de este tipo de ficheros son los siguientes:

- **Ficheros de vídeo:** mp4, avi
- **Ficheros ejecutables:** exe
- **Ficheros de imágenes:** jpeg, png, gif
- **Ficheros de aplicaciones concretas:** xls, pdf

## / 4. Sistemas de almacenamiento de la información

### 4.1. Concepto

Los sistemas de almacenamiento de la información consisten en dispositivos de diferentes características y construcciones que permiten el almacenamiento y gestión de datos de diferentes tipos.

Algunos ejemplos de dispositivos de almacenamiento se pueden observar en la siguiente imagen: disco duro, dispositivos de memoria flash, DVD, NAS, o incluso el almacenamiento en la nube, que cada día cobra mayor importancia.

En el aspecto concreto de las bases de datos, todos los ficheros que las forman deben encontrarse almacenados de alguna manera a través de uno de estos sistemas, mediante los cuales, se podrá acceder a la información que se necesite en un momento determinado.



Fig.9. Diferentes sistemas de almacenamiento de información.

<https://comofriki.com/que-es-un-dispositivo-de-almacenamiento/>



Audio 1. "Operaciones básicas de cualquier sistema de almacenamiento de la información".  
<https://bit.ly/37jH1ZK>



### 4.2. Sistemas lógicos de almacenamiento

Desde el punto de vista de las bases de datos, es importante tener claro el concepto de sistema de almacenamiento lógico.

Como se puede comprobar en la imagen anterior, un sistema de almacenamiento lógico abarca uno o varios medios físicos, ocupando particiones de los mismos. Desde el punto de vista lógico, tendremos un único sistema de almacenamiento, que físicamente se encuentra distribuido.

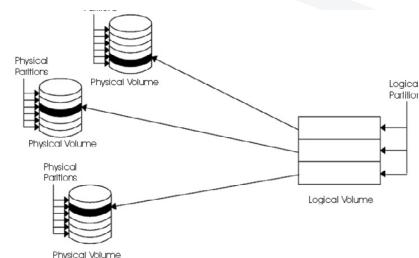


Fig. 10. Almacenamiento lógico



### 4.3. Tipos de sistemas de almacenamiento

Los sistemas de almacenamiento pueden clasificarse atendiendo a diferentes criterios:

- Según su construcción (ópticos y magnéticos)
- Según su capacidad
- En base a su velocidad
- Según la organización de los datos (secuenciales o direccionables)
- Según el tipo de datos que se dese almacenar y en función de la cantidad de los mismos, se podrá determinar la elección de un sistema u otro.

A continuación, se presentan las características de algunos de los sistemas de almacenamiento más conocidos y utilizados.

#### 4.3.1. Discos duros.

Los discos duros pueden ser magnéticos o de estado sólido (SSD), que aportan una mayor velocidad y fiabilidad, aunque también son más caros.

La capacidad de estos medios se incrementa conforme avanza la tecnología. Hoy en día, permiten almacenar cientos de TB de datos para almacenamiento.

#### 4.3.2. NAS (Network Attached Storage) y SAN (Storage Area Network).

Se trata de sistemas de almacenamiento a través de la red, con diferencias importantes entre sí. Mientras que un NAS es un solo dispositivo de almacenamiento centralizado (por ejemplo, un disco duro conectado a una red LAN), SAN es en sí mismo una red de almacenamiento dedicada, que permite intercambiar datos mediante la utilización de un software especial.

¿Cuál es preferible utilizar? Como en muchos casos, esto dependerá del caso concreto. Un NAS es un sistema más sencillo y fácil de mantener, además de eficiente.

Sin embargo, una SAN es una opción más potente, que ofrece mayor rendimiento, pero también más cara y compleja.

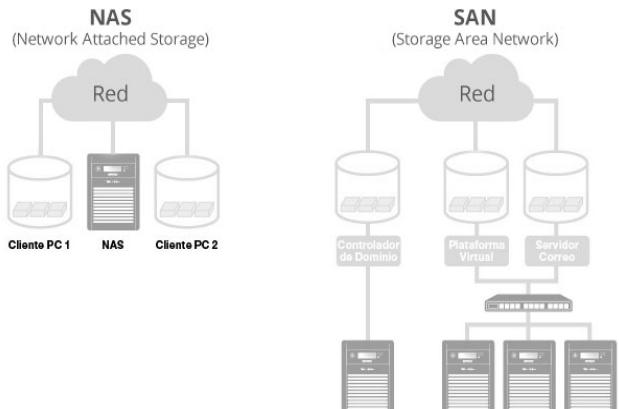


Fig. 11. NAS y SAN

<https://www.infordisa.com/es/comparativa-san-vs-nas/>

#### 4.3.3. Almacenamiento RAID.

RAID (Redundant Array of Independent Disks) es una configuración basada en la utilización de varios discos independientes entre sí, de forma que estos se comporten como si fuesen un único dispositivo.

Este hecho permite reducir el tiempo necesario para acceder a la información, mientras que posibilita utilizar redundancia en los discos, con el fin de reducir las posibilidades de pérdida de información.



Existen varias construcciones RAID, con diferentes características según su configuración. De esta manera, se habla de niveles RAID, algunos de los más comúnmente utilizados son los siguientes:

Nivel	Confiabilidad	Rendimiento	Disponibilidad
<b>RAID 0</b>	• No proporciona tolerancia a fallos.	• Mejora la tasa de transferencia y el tiempo de acceso a los datos.	• El sistema deja de funcionar si hoy una unidad de disco en falla.
<b>RAID 1</b>	• Protege la información en caso de fallo.	• Mejora la lectura de los datos.	• Evita interrupciones por fallos en las unidades.
<b>RAID 2</b>	• El uso del código Hamming permite detectar y corregir errores.	• Mejora la operación de aplicaciones con alta tasa de transferencia.	• Usa múltiples discos dedicados que permiten redundancia de datos.
<b>RAID 3</b>	• El disco de paridad permite reconstruir la información.	• Elevada tasa de transferencias secuenciales.	• Si falla un disco el sistema puede seguir en funcionamiento.
<b>RAID 4</b>	• Es ideal para almacenar ficheros de gran tamaño.	• Durante las operaciones de lectura-escritura las unidades de disco son accedidas de forma individual.	• Es tolerante a fallos ya que se puede recuperar los datos de un disco averiado en tiempo real.
<b>RAID 5</b>	• Distribuye los datos de paridad entre todas las unidades de disco.	• La velocidad de transferencia de datos es alta.	• Es tolerante a fallos con una unidad de disco averiada.
<b>RAID 6</b>	• Cada dato de paridad es redundante y distribuido en dos unidades de disco diferentes.	• Las operaciones de escritura resultan más lentas que las de lectura de datos.	• Es tolerante a fallos con dos unidades de discos averiadas.

Fig. 12. Diferentes tipos de RAID

<https://tecnovirtualization.wordpress.com/2015/10/11/raid/>



Video 1. "Sistemas de almacenamiento RAID anidados"  
<https://bit.ly/37gz2wp>



## 4.4. Gestión de los sistemas de almacenamiento

En cualquier organización es de vital importancia la correcta gestión de la información, ya que como se indicó anteriormente, es el elemento que más valor tiene y que es más importante mantener.

Para ello, es muy importante, entre otras cosas, realizar una gestión y un mantenimiento adecuados de los sistemas de almacenamiento de manera que:

- Se minimice la pérdida de datos en caso de fallo crítico.
- Se alargue la vida útil de los dispositivos utilizados.
- Se optimice la utilización de los mismo.
- Se garantice en todo momento la seguridad e integridad de la información existente.

De esta manera, hay cuatro aspectos fundamentales en la gestión de cualquier sistema de almacenamiento que se deben tener siempre presentes en la gestión del mismo: **capacidad, recuperabilidad, rendimiento y fiabilidad**.

## / 5. Caso práctico 2: “Gestión de tienda y almacén”

**Planteamiento:** Un amigo que ha abierto una tienda de productos de informática nos ha pedido ayudar para la gestión de su negocio, ya que no tiene claro cómo puede controlar el stock del que dispone en su almacén. En una de sus comprobaciones, necesita que, en cada test unitario, se realicen pruebas matemáticas con 2 números. Para ello, al ejecutar cada test, Miguel deberá cargar automáticamente en memoria una lista con 2 números antes de empezar el código de cada prueba unitaria.

**Nudo:** Ya le hemos presentado varios ficheros que le podrán ayudar de momento a gestionar su tienda, mientras que pensamos en una solución más robusta y versátil.



En este punto, creemos necesario que nuestro amigo haga una pequeña inversión para adquirir un sistema de almacenamiento de la información más potente, ya que actualmente utiliza el mismo disco duro externo en el que guarda las películas.

Dado que nuestra idea es ayudarle en un proceso de renovación de su gestión de información, estudiamos varias posibilidades para verificar qué opción de las vistas en este tema podría resultarle más atractiva, y se adapta mejor a sus necesidades.

Hay que tener presente que, en principio, nuestro amigo no piensa ampliar su negocio, que consiste en una pequeña tienda de informática con un pequeño almacén anexo, en la que trabajan él y su pareja. Aunque se trata de un negocio pequeño en el que no se manejarán grandes cantidades de información (al menos de momento), sí le gustaría tener cierta tranquilidad de que no perderá los datos ante cualquier fallo del equipo.

**Desenlace:** Básicamente, nuestro amigo requiere para su tienda seguridad en la información y asegurar su disponibilidad (redundancia).

Das las características de la tienda, y en base a lo visto a lo largo el tema, no parece que la opción del disco duro en un PC vaya a resultar la más óptima, aunque podría ser viable siempre y cuando se aplique redundancia.

Tampoco parece que implementar un sistema SAN sea lo mejor, dado que son caros y complejos.

Por tanto, para este entorno quizás la mejor opción sería un sistema NAS que aplique RAID, de forma que se asegure la redundancia del sistema. Una posible opción sería utilizar el configurador de Synology para construir el RAID que queramos, y poder comparar diferentes equipos que lo implementen. [Pruébalo aquí.](#)



Fig. 13. Ejemplo de configuración RAID de Synology.

## / 6. Resumen y resolución del caso práctico de la unidad

A lo largo del tema se han desarrollado varios conceptos básicos pero de gran importancia para poder iniciarse en el mundo de las bases de datos.

De esta manera, comenzamos haciendo énfasis en la **importancia que tiene para cualquier organización una buena gestión de la información**, lo que implica la necesidad de una correcta infraestructura que lo permita, tanto física como lógica. Ahí es dónde el mundo de las bases de datos demuestra su gran importancia y utilidad, especialmente en la actualidad, dónde cada día hay más aplicaciones que dependen del uso del dato.

Posteriormente se presentó el **concepto de fichero, haciendo hincapié en los diferentes tipos de ficheros que hay**. Al fin y al cabo, las primeras formas de tratamiento y gestión de la información se llevaron a cabo gracias a los ficheros, antes de la llegada de las bases de datos tal y como las conocemos hoy en día.

A continuación, se detallaron las **principales características de los sistemas de almacenamiento de la información** más extendidos y conocidos en la actualidad, haciendo énfasis en las fortalezas y debilidades de cada uno de ellos.

Este asunto es importante, dado que en función del caso que nos ocupe será preferible la utilización de uno u otro, según los objetivos que queramos cumplir. Cerramos el tema recordando la **importancia de la redundancia en estos sistemas**, y de una correcta gestión de los mismos.



## Resolución del caso práctico de la unidad

En el caso inicial planteábamos algunas cuestiones para ayudar a nuestro amigo con su tienda. Todas ellas pueden resolverse con el contenido que hemos visto a lo largo del tema, e incluso los casos prácticos 1 y 2 nos ayudan tanto a ello, como a tener una mayor perspectiva de lo que necesitamos hacer para poder echar una mano a nuestro amigo. Por tanto, una posible respuesta a las preguntas planteadas al principio puede ser la siguiente:

- **¿En qué manera le ayudará en su gestión diaria informatizar sus procesos?**

Se ha comprobado la importancia de contar con un sistema de gestión de la información robusto. El principal activo de una compañía es la información que maneja.

- **¿Cómo podemos ayudarle?**

Podemos ayudarle a implementar un sistema de base de datos que le ayude en su día a día.

- **¿Qué información de partida necesitaremos para ello?**

Básicamente, necesitamos conocer la problemática de su día a día y sus necesidades, para poder diseñar algo a su medida: ni demasiado sencillo ni demasiado potente (y por tanto costoso).

Es importante saber para qué lo quiere y qué quiere hacer con él.

- **¿Habrá alguna solución rápida que podamos aplicar?**

Mientras diseñamos un sistema más potente, podemos realizar la gestión mediante ficheros informáticos, que siempre serán más potentes que el papel. Al fin y al cabo, así es cómo se hacía antes de la aparición de las bases de datos modernas.

## / 7. Bibliografía

- Oppel, A. (2009): *Databases: A Beginner's Guide*. Madrid, España: McGraw-Hill.  
Elmasri, R.; Navathe, S. (2007): *Fundamentos de Bases de Datos* (5.a. ed.). Madrid, España: Pearson Addison-Wesley  
López, I.; Castellano, M.J. y Ospino, J. (2011): *Bases de datos*. Madrid, España: Garceta  
Cabrera, G. (2011): *Sistemas gestores de bases de datos*. Madrid, España: Paraninfo.



BASE DE DATOS

## Las bases de datos

2

# ÍNDICE

/ 1. Introducción y contextualización práctica	4
/ 2. Bases de datos. Definición y elementos	5
2.1. Concepto de base de datos	5
2.2. Elementos que componen una base de datos	5
2.3. Perfiles involucrados en las bases de datos	6
2.4. Tipos de bases de datos y usos principales	6
/ 3. Caso práctico 1: “Base de datos para tienda y almacén”	8
/ 4. Modelos de bases de datos	9
4.1. Modelo jerárquico	9
4.2. Modelo en red	10
4.3. Modelo relacional	10
4.4. Modelo orientado a objetos	10
4.5. Modelo objeto-relacional	11
4.6. Modelo multidimensional	11
4.7. Modelo deductivo	11
/ 5. Bases de datos centralizadas y distribuidas.	
Fragmentación de la información	12
5.1. Bases de datos centralizadas	12
5.2. Bases de datos distribuidas	12
5.3. Fragmentación de la Información	13

# ÍNDICE

/ 6. Caso práctico 2. “¿Qué modelo usar?”	14
/ 7. Resumen y resolución del caso práctico de la unidad	15
/ 8. Bibliografía	16

# OBJETIVOS

*Comprender el concepto de base de datos y asimilar sus posibles usos, como base para ampliar conocimientos.*

*Identificar los elementos principales de una base de datos, profundizando en sus utilidades.*

*Conocer los diferentes tipos de bases de datos existentes y las diferencias entre los mismos.*

*Entender las diferencias entre las bases de datos centralizadas y bases de datos distribuidas, en cuanto a concepto y arquitectura.*

*Estudiar la fragmentación de información y los criterios más adecuados para llevarlo a cabo.*

## / 1. Introducción y contextualización práctica

Como vimos en el primer tema, una buena gestión y almacenamiento de los datos es algo fundamental en nuestros días para gran parte de los procesos que llevamos a cabo en nuestra vida cotidiana, y precisamente para ello se crearon las bases de datos.

Esta tecnología nació en la década de los sesenta, cuando varias compañías empezaban a utilizar discos magnéticos para almacenar datos. A continuación, se crearon los ficheros, que sólo podían ser utilizados de manera secuencial. Además, los programas utilizados tenían aplicaciones muy concretas y prácticamente no interactuaban entre sí. Por ejemplo, era habitual que en una empresa se utilizase un programa para gestionar las ventas de producto, y otro distinto para manejar los stocks de los almacenes, lo que producía incoherencias y errores. Este problema se resuelve con la utilización de bases de datos, que permiten trabajar con ficheros de varios tipos que puedan interrelacionarse entre sí. Evidentemente, la evolución de las TIC durante estos últimos años ha sido exponencial, lo que a su vez ha permitido un acceso general a esta tecnología, no sólo en el ámbito empresarial.



En este tema sentaremos los conceptos básicos sobre las bases de datos, algo fundamental para poder ampliar y profundizar más en próximos temas.

*Fig. 1. El uso de BBDD está presente en nuestra vida cotidiana.*



Audio intro. Pensemos en la base de datos que vamos a crear

<https://bit.ly/2BNA1Zo>





## / 2. Bases de datos. Definición y elementos

### 2.1. Concepto de base de datos

Como se indicó anteriormente, de manera previa a la aparición de las bases de datos, la información se procesaba mediante ficheros, que se construían por separado para cada sistema. Esto ocasionaba varios problemas:

- Ausencia de flexibilidad en el tratamiento de los datos.
- Duplicidad de datos entre diferentes sistemas.
- Dificultades para la administración de la información
- Incoherencias entre datos ubicados en distintos sistemas de información.
- Problemas al tratar la información desde el exterior del sistema, por incompatibilidades.
- Aumento de las necesidades del espacio de almacenamiento.

Todos estos puntos se corrigen mediante la utilización de bases de datos. **El objetivo de éstas es realizar no sólo una estructura de almacenamiento de datos, sino también facilitar su gestión, manipulación y consulta, garantizando en todo momento su integridad.**

### 2.2. Elementos que componen una base de datos

Aunque estos conceptos se ampliarán a lo largo del curso, es necesario tener una perspectiva de cuáles son los elementos principales que componen una base de datos:

- **Entidades:** Son los objetos para los que se almacena información, con características distintas entre ellos. En el caso de una tienda, podrían ser los clientes, el stock de almacén, etc.
- **Tablas:** Las tablas dan forma a la información que queremos almacenar para cada entidad. Por ejemplo, tabla clientes contiene los datos de todos ellos: DNI, nombre, apellidos, dirección, etc.
- **Campo:** Identifica cada tipo de datos distinto, caracterizándose como una columna de la tabla. Por ejemplo, el campo "dni" en la tabla "clientes". Se verá más adelante que hay campos clave, es decir, que son únicos para cada registro. Un ejemplo puede ser el DNI. Los campos pueden contener datos numéricos, alfanuméricos, fechas, etc.
- **Registro:** El registro vendría a ser la fila de la tabla. Es decir, en la tabla de clientes, cada cliente sería un registro distinto.

ID	DNI	NOMBRE	APELLIDOS	LOCALIDAD
1	8321872A	Maria	Fernandez Sanz	Cartagena
2	2319129B	Manuel	Lopez Ayala	Murcia
3	1289381C	Lucía	García Sanchez	Albacete

Tabla 1. Tabla de clientes ejemplo, con campos y registros.



## 2.3. Perfiles involucrados en las bases de datos

En cualquier organización que utilice y administre bases de datos existen al menos cuatro perfiles de usuarios básicos que se deben conocer, ya que cada uno tendrá unas funciones distintas:

PERFIL	CARACTERÍSTICAS
Usuario	Es el perfil más importante, ya que el resto tienen como objetivo satisfacer sus necesidades.
Diseñador	Deciden la arquitectura de la base de datos, cómo se almacenarán los datos y cómo se relacionarán entre sí.
Programador	Es fundamental para el éxito de una base de datos que esté correctamente diseñada, para lo que es importante contar con los usuarios que vayan a utilizarla, entender sus problemas y objetivos. Construyen el software que permitirá a los usuarios utilizar los datos, realizando consultas, inserciones de nuevos datos, eliminación de los anteriores, etc.
Administrador	Una vez construida la base de datos, será el encargado de mantener su funcionamiento y garantizar su integridad y seguridad.

Tabla 2. Distintos perfiles que participan en la creación o utilización de bases de datos.

## 2.4. Tipos de bases de datos y usos principales

Las bases de datos actualmente pueden clasificarse en base a diferentes criterios, algunos de los cuales se describen a continuación.

### 2.4.1. Bases de datos según su utilización

- **Individual:** como su nombre indica, se usa sólo por una persona, que hace de administrador, usuario, diseñador y programador al mismo tiempo.

Por ejemplo, una base de datos sobre una colección de CDs.

- **Compartida:** cuenta con varios usuarios que pueden acceder a la misma información. Los requerimientos del equipo en el que se sitúe serán mayores, al permitir accesos concurrentes. Por ejemplo, podría ser una base de datos de una pyme.



- **Bancos de datos:** caracterizadas por contener un gran volumen de datos, que normalmente están referidos a un tema en concreto (publicidad, datos técnicos o científicos, etc). Suelen estar desarrolladas por alguna organización o empresa determinada, para explotar el servicio de consulta. Un ejemplo de este tipo sería BASE: <https://www.base-search.net/>
- **Pública:** permiten el acceso a cualquier usuario que cuente con los medios técnicos para hacerlo. Por ejemplo, la base de datos con los datos del catastro.

#### 2.4.2. Bases de datos según su temática

- **Empresariales:** almacenan datos sobre una empresa determinada, cuentas, transacciones, etc.
- **Científicas:** este tipo se construyen con datos útiles para investigadores de diversos campos científicos: Historia, física, biología, etc.
- **Políticas y jurídicas:** contiene información relacionada sobre la legislación en vigor.
- **Sanitarias:** son utilizadas por personal médico, y que contienen, por ejemplo, los historiales médicos.

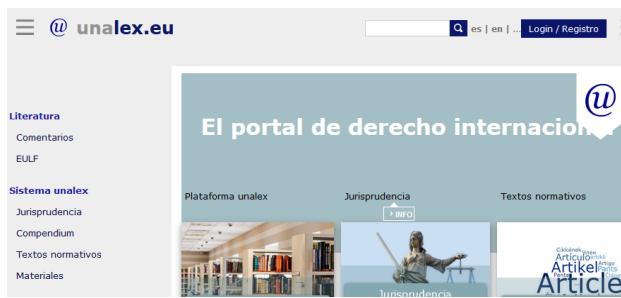
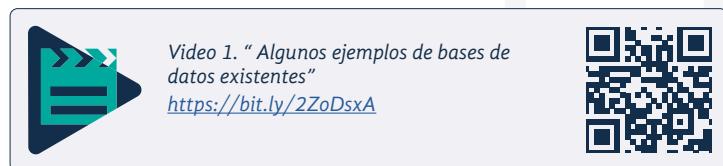


Fig. 3. Base de datos con contenido jurídico



#### 2.4.3. Usos y aplicaciones de las bases de datos

Las aplicaciones de las bases de datos son muy amplias, prácticamente inabarcables hoy en día. A continuación, se describen algunas de ellas:

- **Administrativas:** existen prácticamente en cualquier empresa u organización. Se utilizan para almacenar información sobre clientes, productos, facturas, etc.
- **Médicas:** aportan información sobre pacientes. Importante la seguridad de la información y el acceso a la misma desde diferentes sitios (Centro de Salud y Hospital).
- **Sistemas de posición geográfico**, para por ejemplo poder utilizar sistemas de navegación GPS.
- **Organismos Públicos:** contienen información sobre los ciudadanos para poder realizar trámites: certificados, declaración de Hacienda, citaciones, etc.



- **Comunicaciones:** para establecer cualquier llamada a través de un smartphone se consultan varias bases de datos, en las que están almacenados datos del origen y del destino, para poder establecer la comunicación.
- **Turismo:** reservas de hoteles, compra de billetes de tren o de avión, consulta de aplicaciones turísticas sobre atracciones de un lugar, etc.
- **Motores de búsqueda web:** se basan en bases de datos con mucha información sobre páginas web.

## / 3. Caso práctico 1: “Base de datos para tienda y almacén”

**Planteamiento:** Siguiendo con el planteamiento del tema 1, vamos a intentar ayudar a nuestro amigo a construir una base de datos sencilla que le ayude con la gestión de su tienda. Necesitamos diseñar la base de datos, antes de su implementación.

**Nudo:** Como diseñadores de la base de datos, el primer paso que tenemos que dar será el de definir la arquitectura que tendrá nuestra base de datos.

Para ello, debemos decidir qué entidades, tablas, registros y campos consideraremos en este primer diseño.

Recordemos que el objetivo principal es la creación de un sistema de base de datos que le ayude en la gestión de su almacén, así como en el proceso de ventas y compras.

**Desenlace:** Una posible propuesta básica de diseño (posteriormente, podrá ampliarse y/o potenciarse para cubrir otras necesidades) puede ser la que se presenta a continuación.

- **Entidades:** Almacén, Cliente y Proveedor
- **Tablas:**
  - Almacén: Stock y Movimientos
  - Clientes: Ventas y Clientes
  - Proveedor: Compras y Proveedores
- **Campos:**
  - Stock: ID producto, concepto, unidades, proveedor
  - Movimientos: ID producto, concepto, unidades,
  - Ventas: Id producto, concepto, unidades vendidas, precio, fecha venta, cliente
  - Clientes: Id cliente, nombre, apellidos, dirección, teléfono
  - Compras: Id producto, concepto, unidades compradas, precio, fecha compra, proveedor
  - Proveedores: Id proveedor, nombre, apellidos, dirección, teléfono



STOCK	MOVIMIENTOS	VENTAS
Id producto Concepto Unidades Proveedor	Id producto Concepto Unidades	Id producto Concepto Unidades Precio Fecha Cliente
CLIENTES	COMPRAS	PROVEEDORES
Id cliente Nombre Apellido Dirección Teléfono	Id producto Concepto Unidades Precio Fecha Proveedor	Id proveedor Nombre Apellido Dirección Teléfono

Tabla 3. Diseño base de datos para tienda informática

## / 4. Modelos de bases de datos

Entendemos como “modelo de base de datos” a la arquitectura de la misma, es decir, a cómo se almacena la información que contiene y de qué manera se interrelaciona.

A continuación, se detallan las principales características de los modelos de bases de datos más conocidos, que dan lugar a nuevos tipos de bases de datos.

### 4.1. Modelo jerárquico

Es el primero que se creó. Se basa en crear una jerarquía entre los datos o ficheros, en forma de árbol. Cada nodo del árbol puede tener varios hijos, pero cada hijo sólo tiene un padre. Por lo tanto, la relación entre las entidades existentes es de padre a hijo.

El sistema IMS de IBM es el ejemplo quizás más extendido, que se utiliza mucho en sistemas mainframe.

Actualmente se utiliza poco, ya que existen modelos más robustos. Sin embargo, este modelo constituyó la base sobre la que se crearon otros.

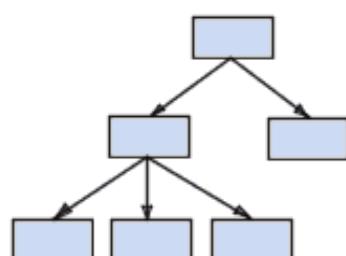


Fig. 4. Representación del modelo jerárquico.



## 4.2. Modelo en red

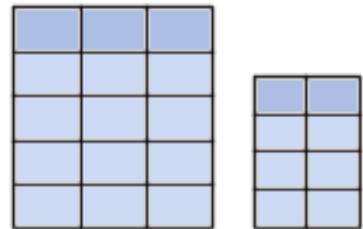
Constituye en sí una **evolución del jerárquico**, diferenciándose en que es más flexible (no tiene por qué seguirse un modelo de árbol), aunque también más complejo. Proporciona buen rendimiento, pero comercialmente no ha tenido mucho desarrollo.

Un ejemplo es el sistema CODASYL de CA Technologies.

## 4.3. Modelo relacional

En este caso, se introduce el concepto de bidimensionalidad, ya que los datos se almacenan en tablas con filas y columnas relacionadas entre sí. Una vez surgió este modelo fue el más utilizado, ya que proporciona una gran flexibilidad y sencillez en su utilización. A ello contribuyeron empresas como Oracle o Microsoft.

Con su aparición se crearon también los conceptos de entidad, campo y registro.



También a raíz de su aparición se creó el lenguaje SQL, para realizar consultas *Fig. 5. Representación del modelo relacional.* a estos.

## 4.4. Modelo orientado a objetos

Este modelo surgió en los noventa, junto al auge de los lenguajes de programación orientados a objetos (Java, C++). Se basa en aplicar a las bases de datos el paradigma de estos lenguajes de programación.

De esta manera, se trata de construir una base de datos en base a objetos (datos), cada uno con sus propiedades y atributos. Además, las operaciones o procedimientos sobre los mismos se les conoce como métodos, de manera similar a los lenguajes de programación.

Para ayudar a entender este concepto, se muestra el siguiente ejemplo, sobre una base de datos de empleados:

CLIENTES	COMPRAS	PROVEEDORES
Empleados	Luis Herance Martín	Jefe de Proyecto
		Salario: 3000
		Madrid
	Lucía Valverde Fdez.	Consultor de Ventas
		Salario: 4000
		Murcia

Tabla 3. Diseño base de datos para tienda informática



Este modelo proporciona ventajas frente al relacional, especialmente para gestión de imágenes y documentos o sistemas multimedia, ya que se trata de modelos de datos más complejos que son difíciles de representar mediante tablas.

Algunos ejemplos de bases de datos de este tipo son InterSystems u Objectivity.

## 4.5. Modelo objeto-relacional

Se trata de un modelo híbrido entre los dos anteriores, y surge de la incorporación a base de datos relaciones ya existentes, de ciertas capacidades de las orientadas a objetos, con el fin de mejorarlas, manteniendo sus bondades características. El motivo es claro: se reducen costes respecto a construir una base de datos relacional en una base de datos orientada a objetos. Por ello, los principales fabricantes han apostado por este tipo, con el fin de optimizar costes y mejorar sus productos: éstos son Oracle, Microsoft o IBM.

En concreto, permiten añadir a las bases de datos relacionales tipos de datos o procedimientos definidos por el usuario, realizar consultas recursivas, etc.

A través del siguiente ejemplo, creamos un objeto nuevo en una base de datos Oracle, utilizando el ejemplo anterior de la Tabla 4:

```
CREATE TYPE ANTONIO MARTÍN AS OBJECT
(
  CARGO VARCHAR(20),
  SALARIO NUMBER(4),
  CIUDAD VARCHAR(20)
);
```

Figura 6. Ejemplo creación objeto en Oracle.

## 4.6. Modelo multidimensional

El modelo de base de datos multidimensional está orientado al mundo de la inteligencia artificial, con el fin de realizar aplicaciones muy dirigidas a algún tema concreto.

Como su propio nombre indica, los datos se distribuyen en matrices de varias dimensiones formando cubos. De esta forma, un dato puede tener varios valores en función del eje que se recorra.

Su gran ventaja es la facilidad para gestionar grandes volúmenes de datos.

En la siguiente imagen se muestra un ejemplo de base de datos de este tipo, que podría ser referente a las ventas realizadas de varios productos (eje producto), en diversas regiones (eje región) durante un año (eje tiempo).

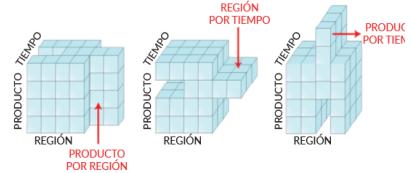


Fig. 7. Ejemplo de modelo de base de datos multidimensional

## 4.7. Modelo deductivo

La característica principal de estas bases de datos es que permiten obtener nuevos datos a partir de los que se han introducido. Es decir, mediante la aplicación de inferencias (procedimiento por el que se generan conclusiones a partir de premisas) se generan deducciones a partir de los datos existentes.

A modo de resumen, se recogen en la siguiente tabla las principales características de las bases de datos vistas hasta ahora:



Modelo	Datos almacenados	Ubicación
Jerárquicas	Listas y árboles	Varios ficheros
En red	Árboles y grafos	
Relacionales	Conjuntos y relaciones	
Orientadas a obj.	Objetos complejos	
Multidimensional	Cubos	Una o varias BBDD
Deductivo	Hechos y reglas	

Tabla 5. Resumen sobre los principales tipos de bases de datos

## / 5. Bases de datos centralizadas y distribuidas. Fragmentación de la información

En primer lugar, debemos considerar que la distinción entre bases de datos centralizadas y distribuidas es en sí misma, otra manera de clasificar las bases de datos, en esta ocasión, en función de dónde esté ubicada la información.

### 5.1. Bases de datos centralizadas

Como su propio nombre indica, las bases de datos centralizadas se caracterizan principalmente porque se ubican en un único dispositivo o máquina, que puede ser de varios tipos.

Su aplicación puede ser variada, pero suelen darse en empresas o incluso en sistemas domésticos de un único usuario. Es decir, el hecho de ser centralizadas no significa que no puedan soportar grandes cantidades de datos, con un alto grado de rendimiento.

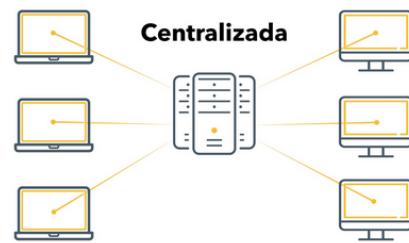


Fig. 8. Arquitectura de base de datos centralizada

### 5.2. Bases de datos distribuidas

Al contrario que las bases de datos centralizadas, estas se caracterizan porque la información no se concentra en un único dispositivo, sino que es encuentran repartidos en varios nodos.

Podría decirse que las bases de datos distribuidas son bases de datos unidas entre sí mediante redes de comunicaciones, a través de las cuales pueden acceder a la información los diferentes usuarios del sistema.

Consiste en multiplicar las máquinas o nodos que controlan la base de datos, ubicándolos en diferentes lugares y permitiendo el intercambio y acceso de información entre ellos a través de la red.



Por ejemplo, este podría ser el caso de una empresa que cuenta con varias delegaciones a nivel nacional. Puede construirse de diferentes maneras:

- **Modo Réplica:** se duplica la información de un nodo en otro.
- **Nodo Particionado:** no existe copia de la información, pero ésta se encuentra distribuida por todos los nodos.
- **Nodo Híbrido:** sería una mezcla de las anteriores, siendo la más usada.



Video 2. "Tecnología Blockchain aplicada a Bases de Datos"  
<https://bit.ly/2VtViy2>



### 5.3. Fragmentación de la Información

Según vimos en el punto anterior, las bases de datos distribuidas cuentan con la información dividida en diversos sitios, es decir, la información suele encontrarse fragmentada.

Por tanto, es conveniente particionar adecuadamente la información para evitar que haya problemas a la hora de recuperarla.

En esta línea, es importante tener en cuenta el concepto de “grado de fragmentación”, que básicamente se refiere al número de veces que troceamos los datos. Este es un concepto importante a la hora de poder ejecutar las consultas a la base de datos adecuadamente, no siendo permitido un grado de fragmentación cero, o bien un valor de fragmentación demasiado elevado, dado que no sería viable poder gestionarlo. Este debe ser un valor intermedio, que dependerá básicamente de lo que se pretenda hacer con la base de datos, así como de la arquitectura de la misma y de las aplicaciones que se utilicen.

La fragmentación puede llevarse a cabo a nivel de registro (fila) o bien a nivel de campo (columna).



Audio 1. "Reglas a cumplir al realizar la fragmentación"  
<https://bit.ly/2YHYwA7>



Podemos diferenciar entre varios tipos de fragmentación de la información:

- **Fragmentación vertical:** Se caracteriza porque cada fragmento obtenido de dividir la información cuenta con una serie de atributos de la información de partida, que se repiten en todos ellos.

Lo que se pretende con ella es que haya aplicaciones de usuario que se puedan ejecutar sobre un solo fragmento, para optimizar el tiempo de respuesta ante consultas.



<b>Id empleado</b>	<b>Nombre</b>	<b>Salario</b>	<b>Ubicación</b>	<b>Departamento</b>
1	Francisco	5000	Toledo	RRHH
2	Ana María	10000	Madrid	Consultoría
3	Luisa	4000	Cuenca	Calidad
<b>Id empleado</b>	<b>Nombre</b>	<b>Salario</b>		
1	Francisco	5000		
2	Ana María	10000		
3	Luisa	4000		
<b>Id empleado</b>	<b>Ubicación</b> <b>Departamento</b>			
1	Toledo RRHH			
2	Madrid Consultoría			
3	Cuenca Calidad			

Tabla 9. Ejemplo de fragmentación vertical

- **Fragmentación horizontal:** se realiza directamente sobre los registros o filas de las tablas, dividiendo la información en fragmentos que contienen varias de ellas. Es más sencilla de realizar que la vertical, ya que es menos complicada de llevar a cabo.
- **Fragmentación híbrida:** como su nombre indica, es una mezcla de las anteriores. Puede ser VH (vertical + horizontal) y HV (horizontal + vertical).

## / 6. Caso práctico 2. “¿Qué modelo usar?”

Modelo	Resultado	Observaciones
Jerárquica	X	Obsoleto. Modelo demasiado rígido.
En red	X	Prácticamente en desuso.
Relacional	✓	Amplio soporte y posibilidades, muy extendidas y totalmente válidas para el objetivo que se persigue.
Orientada a objetos	X	Parece demasiado compleja para lo que se quiere hacer, aunque sería viable.
Objeto relacional	✓	Sería una opción válida si se quiere hacer algo más sofisticado y preparado para otras posibles funcionalidades a futuro.
Multidimensional	X	No tiene sentido aplicarlo en este caso por complejidad.
Deductiva	X	No tiene sentido aplicarlo en este caso.

Tabla 7. Análisis del modelo a emplear para la base de datos de la tienda de informática



**Planteamiento:** Dado que seguimos involucrados en ayudar a nuestro amigo con su tienda, tras plantear en el caso práctico 1 cómo podemos construir la base de datos, tenemos ciertas dudas sobre en qué modelo debemos basarnos.

**Nudo:** Tras analizar las características principales de todos los modelos existentes, necesitamos decidirnos por aquel que se adapte mejor a nuestro caso concreto y exponer los motivos que nos llevan a su elección.

**Desenlace:** La elección del modelo de datos que queremos aplicar dependerá fundamentalmente de las aplicaciones que contenga la base de datos, así como de las funcionalidades que requieran los usuarios de la misma.

Dado que nuestro amigo solo nos traslada su necesidad de organizar su negocio, lo más lógico parece establecer un modelo relacional tradicional de partida, que puede ser rápido y sencillo de construir e implementar. Con el tiempo, además, este modelo se podría llegar a adaptar a un modelo de objeto-relación, en caso de que se requiera potenciar más funcionalidades. Por otro lado, al únicamente disponer de una tienda, parece que será más sencillo pensar en una base de datos centralizada, aunque debería ser accesible de manera remota. Podemos llegar, por lo tanto, a las siguientes conclusiones:

## / 7. Resumen y resolución del caso práctico de la unidad

Como vimos en el primer tema, **una buena gestión y almacenamiento de los datos es algo fundamental en nuestros días**, ya que se aplica a prácticamente cualquier ámbito cotidiano de nuestra vida: cuando nos registramos en una web, accedemos a un espectáculo, reproducimos contenidos en Spotify o Netflix, etc.

En este tema se han fijado las bases para entender el concepto de base de datos, detallándose su definición y los principales elementos que las componen. Además, se han descrito los diferentes tipos de usuarios que suelen intervenir en una base de datos. Todos estos son conceptos en los que se profundizará a lo largo de la asignatura.

También se han presentado los principales modelos de datos que se utilizan en bases de datos, sobre algunos de los cuales también se ampliará información en temas posteriores, al ser los más extendidos y utilizados (por ejemplo, el modelo relacional). Por último, se han planteado las diferencias existentes entre las bases de datos centralizadas y distribuidas. Este es un concepto muy importante, ya que en función de los objetivos que queramos cubrir con la base de datos convendrá utilizar uno u otro.

### Resolución del caso práctico de la unidad.

En el caso práctico inicial se planteaban diversas cuestiones para dar respuesta, en relación al proyecto en el que estamos ayudando a nuestro amigo con la base de datos de su tienda de informática.

Algunas posibles respuestas a las preguntas planteadas podrían ser las siguientes:

- **¿Necesitaremos más información de la que ya tenemos? Si es así, ¿cuál concretamente?** En cualquier diseño, especialmente de base de datos, es fundamental partir de la información adecuada, para lo que es importante tratar con varios perfiles de usuarios que tenga la aplicación, con el fin de conocer las necesidades de cada uno. Es muy importante definir el tipo de usuarios que tendrá el sistema, cuál será el medio de alojamiento, qué tipo de modelo de datos se aplicará, cuáles son los datos que necesitaremos para construir la base de datos (ver caso práctico 1), etc.
- **¿Cómo vamos a construir la base de datos?** A esta pregunta le hemos dado respuesta entre los casos prácticos 1 y 2, en los que hemos definido en gran medida la arquitectura que tendrá nuestra base de datos y la información que vamos a recoger en la misma. En pasos anteriores hemos definido el medio físico y cómo se accederá a la información.
- **¿Vamos a aplicar algún modelo en concreto?** Dadas las características y necesidades a cubrir, parece lo lógico plantear la base de datos como una base de datos relacional.



## / 8. Bibliografía

- Oppel, A. (2009): Databases: A Beginner's Guide. Madrid, España: McGraw-Hill.
- Elmasri, R.; Navathe, S. (2007): Fundamentos de Bases de Datos (5.a. ed.). Madrid, España: Pearson Addison-Wesley
- López, I.; Castellano, M.J. y Ospino, J. (2011) : Bases de datos. Madrid, España: Garceta
- Cabrera, G. (2011): Sistemas gestores de bases de datos. Madrid, España: Paraninfo.



BASE DE DATOS

## **Sistemas gestores de bases de datos (SGDB)**

---

# ÍNDICE

/ 1. Introducción y contextualización práctica	4
/ 2. Definición y funciones	5
2.1. Concepto de SGBD	5
2.2. Funciones de un SGBD	5
/ 3. Estructura y componentes de un SGBD	6
3.1. Estructura de un SGBD	6
3.2. Independencia físico-lógica	7
3.3. Componentes de un SGBD	7
/ 4. Caso práctico 1: “Base de datos para tienda y almacén”	8
/ 5. Tipos de SGBD	9
5.1. Segundo modelo lógico	9
5.2. Segundo tamaño	9
5.3. Segundo su forma de ejecución	9
5.4. Segundo la ubicación de la base de datos	9
/ 6. Funcionamiento de un SGBD paso a paso	10
/ 7. SGBD vs. ficheros clásicos	10
/ 8. SGBD libres y comerciales	11
8.1. SGBD libres o de código abierto	11
8.2. SGBD comerciales	12

# ÍNDICE

/ 9. Caso práctico 2: “Comparativa SGBD”	12
/ 10. Resumen y resolución del caso práctico de la unidad	14
/ 11. Bibliografía	15

# OBJETIVOS

*Comprender en qué consiste un Sistema Gestor de Bases de Datos (SGDB) y reconocer su utilidad.*

*Identificar las partes de las que se compone un SGDB.*

*Describir la función de los diferentes elementos de los que se compone un SGDB.*

*Conocer los diferentes tipos y tecnologías de SGDB existentes en la actualidad.*

## / 1. Introducción y contextualización práctica

Los SGDB surgieron de manera conjunta con las bases de datos en la década de los sesenta del pasado siglo, aunque entonces no se les conocía por este nombre. Inicialmente, estaban concebidos como un sistema para trabajar con volúmenes grandes de datos en sistemas propietarios, siendo empleados sobre todo en la industria aeroespacial o automovilística.

Estos sistemas estaban muy ligados a la parte física del sistema, por lo que no eran interoperables, y su programación era compleja. Eso sí, supusieron un gran salto en cuanto a rendimiento y usabilidad.

Durante los años ochenta los equipos informáticos comenzaron a extenderse por empresas y organizaciones, lo que potenció el desarrollo de aplicaciones informáticas más flexibles y accesibles por los usuarios. Es entonces cuando nace Oracle (1980) y se crea el lenguaje SQL (1986), que conllevó una revolución, apareciendo los llamados SGDB relacionales, los cuales lograban por fin separar la programación de bases de datos de la parte física de la máquina.



Durante los noventa aparece el concepto de base de datos distribuida, que vimos en el tema 2, con el fin de aglutinar las diferentes BBDD que se habían ido creando en varios departamentos de la misma empresa. La explosión definitiva de la informática hace que las bases de datos se extiendan exponencialmente, al mismo ritmo que lo hacen los servicios informáticos, que ya llegan al usuario doméstico, y no sólo al empresarial o institucional.

Fig. 1. Oracle es uno de los SGDB más conocidos y extendidos.

Actualmente, los SGDB continúan en pleno desarrollo para poder hacer frente a varios retos, como puede ser el mundo de la nube, garantizar la máxima disponibilidad, o la arquitectura multitenant. Algunas de las características de Oracle 19c se pueden encontrar haciendo click [aquí](#).



Audio intro. SGBD en la actualidad

<https://bit.ly/2ZsQTN8>





## / 2. Definición y funciones

### 2.1. Concepto de SGBD

Un sistema gestión de bases de datos (SGBD) se puede definir como la capa de aplicación (software) que utiliza el usuario (analista, programador, usuario, etc) para poder gestionar y manipular las bases de datos, garantizando en todo momento la seguridad.

Según el cuadrante de Gartner, esta es la visión actual del mercado de los SGBD, aunque no están todos los que existen:

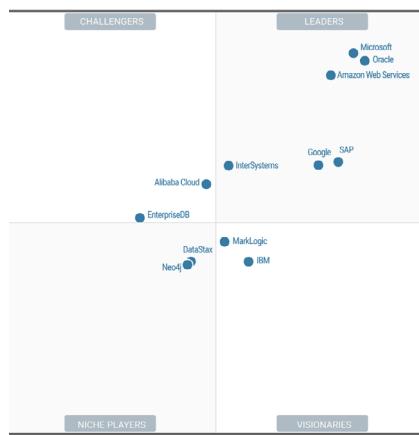


Figura 2. Estado actual de los SGBD según Gartner.

### 2.2. Funciones de un SGBD

Algunas de las principales funciones serían las siguientes:

- Permiten almacenar, consultar y manipular datos de manera sencilla y transparente para el usuario.
- Facilitan la migración del sistema a otras máquinas independientemente de su sistema operativo, facilitando la distribución de la base de datos.
- Garantizar el control del sistema, por ejemplo, dando acceso sólo a quien debe tenerlo, proporcionando además seguridad y asegurando la integridad de los datos.
- Integran mecanismos para realizar copias de seguridad.
- Proporcionan una solución válida a posibles problemas producidos por la concurrencia de usuarios a los datos (varios usuarios manipulando los mismos datos).
- Permiten consultar estadísticas sobre su uso, así como monitorizar la utilización de la base de datos.
- Permiten visualizar la arquitectura de la base de datos, es decir, la estructura de los datos en tablas, registros, campos, relaciones entre ellos, etc.
- Posibilitan el uso de lenguajes de programación para acceder a la información.



## / 3. Estructura y componentes de un SGBD

### 3.1. Estructura de un SGBD

Antes de conocer en profundidad los tipos de SGBD y sus características, es importante conocer los niveles de abstracción que existen en una base de datos, ya que son importantes a la hora de identificar las funciones que se deben realizar. Son tres:

NIVEL	CARACTERÍSTICAS
Interno o físico	Se refiere a la gestión de los datos, es decir: dónde se guardan, el nombre los archivos, cómo se accede a los registros, qué tipo de registros hay, los campos que los forman, etc.
Conceptual o lógico	En el que se describe la estructura de los datos (normalmente por parte del analista), así como las relaciones entre los mismos. Contiene: <ul style="list-style-type: none"><li>• Entidades (p.e. clientes, pedidos, etc)</li><li>• Atributos (p.e. nombre, DNI, etc)</li><li>• Asociaciones entre ellas (p.e. compra)</li><li>• Normas que deben cumplir los datos</li></ul>
Externo o usuario	Representa cómo percibe el usuario la base de datos, siendo el nivel más cercano al mismo

Tabla 1. Nivel y características estructura de un SGBD.

En una base de datos, suele ocurrir que existe un único esquema físico, un solo esquema lógico, pero varios esquemas externos. Esto es así porque en función del usuario, se le mostrarán unos datos y se omitirán otros.

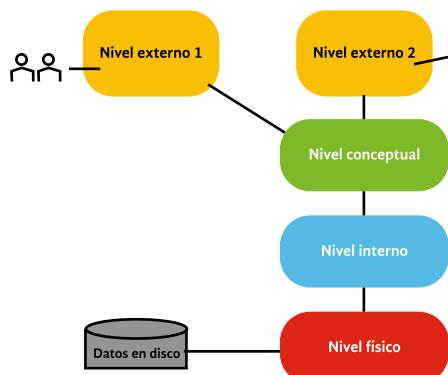
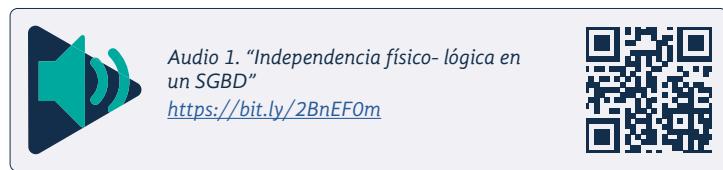


Fig. 3. Esquema de la estructura de un SGBD.



## 3.2. Independencia físico-lógica

Escucha este audio para poder ampliar la información sobre la independencia físico – lógica en un SGBD:



## 3.3. Componentes de un SGBD

Un SGBD habitualmente cuenta con los siguientes componentes:

- **Datos almacenados:** en forma de ficheros organizados adecuadamente y de manera eficiente.
- **Lenguaje de programación:** a través del cual los usuarios podrán acceder a los datos, con el fin de añadir, consultar o modificar información.
- **Diccionario de datos:** contiene los llamados «metadatos», que es información relativa a los datos contenidos en el sistema y a su estructura.
- **Utilidades y herramientas de la base de datos:** Se consideran en este grupo las herramientas que ayudan a una correcta gestión y administración de la base de datos. Facilitan a los administradores tareas relativas a la realización de copias de seguridad, gestión de usuarios y grupos, realización de estadísticas e informes, permisos de acceso, etc. Es una parte muy importante en el SGBD, para su correcto funcionamiento y mantenimiento..
- **Entorno gráfico:** ayuda a facilitar el acceso al sistema por parte de usuarios menos avanzados, haciendo más intuitivo su uso.
- **Usuarios:** junto a los datos, son el principal activo de la base de datos, ya que serán los que se encarguen de su explotación, gestión y mantenimiento. Existirán diferentes perfiles:
- **Administradores:** gestionan su funcionamiento y velan por su buen funcionamiento.
- **Analistas:** se encargan de diseñar la base de datos y de controlar su desarrollo e implantación.
- **Programadores:** desarrollan las aplicaciones de usuario en base a las directrices de los analistas.
- **Operadores de mantenimiento:** proporcionan soporte a los usuarios en tareas cotidianas.
- **Usuarios finales:** encargados de explotar la base de datos. Tienen diferentes permisos y perfiles.



Fig. 4. Perfiles de usuarios relacionados con los esquemas.



## / 4. Caso práctico 1: “Base de datos para tienda y almacén”

**Planteamiento:** Siguiendo con el planteamiento del tema 1, continuamos ayudando a nuestro amigo a implementar la base de datos que necesita para su tienda.

**Nudo:** En el tema 2 ya tratamos sobre las entidades, tablas, registros y campos consideraremos en este primer diseño.

Recordemos que el objetivo principal es la creación de un sistema de base de datos que le ayude en la gestión de su almacén, así como en el proceso de ventas y compras.

Dado que ya tenemos más información sobre la estructura y componentes de un SGBD, vamos a plantear cómo será esa estructura y cuáles serán los componentes en nuestro caso.

**Desenlace:** Una propuesta básica puede ser la que se presenta a continuación:

En relación al esquema interno, se definió en parte en el Caso práctico del tema 2, quedando pendiente de resolver la ubicación de los ficheros y el nombre de estos. Nuestra propuesta es que el nombre coincida con el de la tabla por sencillez, y que se almacenen en un dispositivo tipo NAS que implemente un sistema RAID redundante para garantizar la integridad de los mismos.

En cuanto al esquema lógico o conceptual, en el tema 2 se definieron las siguientes tablas y atributos para cada una de ellas, quedando pendiente cómo se relacionarán entre sí. Una posible propuesta puede ser la siguiente, aunque puede haber otras:

STOCK	MOVIMIENTOS	VENTAS
Id producto Concepto Unidades Proveedor	Id producto Concepto Unidades	Id producto Concepto Unidades Precio Fecha Cliente
CLIENTES	COMPRAS	PROVEEDORES
Id cliente Nombre Apellido Dirección Teléfono	Id producto Concepto Unidades Precio Fecha Proveedor	Id proveedor Nombre Apellido Dirección Teléfono

Tabla 2. Esquema lógico o conceptual.

Por último, en cuanto al esquema externo, dependerá en gran medida del SGBD que decidimos utilizar finalmente.

Habrá que definir también los perfiles de usuario o grupos de usuarios que implementaremos. En nuestro caso, habrá un perfil de administrador con permisos completos, otro perfil de usuario avanzado, y, por último, otro perfil de consulta con acceso limitado a ciertas funciones.



## / 5. Tipos de SGBD

De igual manera que ocurre con las bases de datos, los SGBD se pueden clasificar en base a múltiples criterios.

### 5.1. Segundo modelo lógico

Esta clasificación está muy extendida, y se basa en los modelos lógicos que vimos en el tema 2: jerárquico, de red, relacional, orientado a objetos, etc.

### 5.2. Segundo tamaño

Según su tamaño, pueden ser:

- **Ligero:** tamaño muy reducido, se utilizan en entornos donde no se dispone de equipos muy potentes y donde no se necesita un gran rendimiento. Un ejemplo es el SQLite, MS Access o LibreOffice Base.
- **Alto rendimiento:** se utilizan para manejar grandes cantidades de datos, ya que ofrecen funcionalidades más complejas: gestión de usuarios, control de concurrencia, estadísticas e informes, monitorización, etc. A este grupo pertenecen Oracle, Microsoft SQL Server, MySQL y PostgreSQL, entre otros.

### 5.3. Segundo su forma de ejecución

- **SGBD Monocapa:** el SGBD se encuentra alojado en el mismo equipo de la base de datos y que el cliente. Se utiliza con pequeñas cantidades de datos y pocos usuarios.

Un ejemplo podría ser el LibreOffice Base.

- **SGBD Bicapa:** esquema cliente / servidor, ya que el SGBD y la base de datos se encuentran en la misma máquina o máquinas (servidores), que reciben conexiones de los clientes a través de la red.
- **SGBD Multicapa:** en este caso existe uno o varios servidores intermedios entre los usuarios y la base de datos. Es el esquema que se da en la web habitualmente.

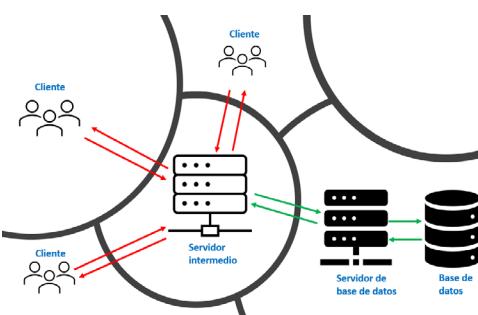


Fig. 5. Esquema SGBD multicapa

### 5.4. Segundo la ubicación de la base de datos

En este caso, podemos hablar de SGBD centralizados o distribuidos. Significa que el SGBD y la base de datos pueden estar distribuidos en varios equipos, conectados entre sí a través de la red. Es la misma idea que vimos en el tema 2 en relación a las bases de datos, aplicada en este caso a los SGBD.



## / 6. Funcionamiento de un SGBD paso a paso

En la siguiente imagen se muestran los procesos que tienen lugar entre una **solicitud de datos por parte del usuario y el SGBD**, en un entorno de acceso a través de la red:

El usuario realiza una **petición de acceso** a la información contenida en la base de datos (1), a través de la aplicación (nivel externo), que traduce la misma para que sea comprensible por parte del sistema.

A continuación, se activa el **proceso de cliente** (2), que se encarga de gestionar la petición y enviarla al proceso servidor de la **base de datos** (3). La petición la recoge el proceso del servidor (equivalente al proceso de cliente en el lado servidor), que a su vez se la pasa al **SGBD** (4) después de haberla analizado y traducido para facilitar su comprensión.

Una vez que la petición ha llegado al SGBD, este verifica si es correcta a través del diccionario de datos, comprobando además en qué parte concreta de los ficheros se encuentra la **información demandada** (5). Cuando tiene este dato, traduce y envía la **solicitud** (6) al Sistema Operativo, que será el encargado de acceder directamente al **soporte físico** de los datos (7).

La **base de datos** recibe la petición de información por parte del Sistema Operativo y le contesta con ella, siempre y cuando no haya ocurrido ningún error en el proceso.

A su vez, el Sistema Operativo contesta al SGBD, que traduce los datos y se los entrega de vuelta al proceso servidor.

El proceso servidor a su vez le envía la información procesada al proceso cliente a través de la red como vemos en el ejemplo. Éste los comprueba y se los hace llegar a la aplicación de usuario, que en última instancia se encarga de traducir la información recibida en el formato gráfico determinado, para que el usuario pueda visualizarla de la mejor manera posible, en función del sistema empleado.

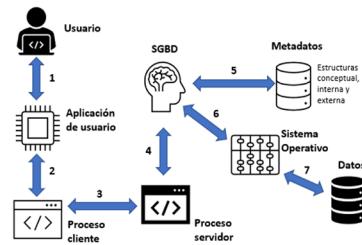


Fig. 6. Esquema funcionamiento SGBD

## / 7. SGBD vs. ficheros clásicos

Como se introdujo en el tema 2, antes de la aparición de las bases de datos y SGBD, la gestión de la información se hacía en gran medida a través de ficheros. A grandes rasgos, están formados por **varios registros** que están contenidos en una **estructura única**, siendo ésta accesible mediante una aplicación informática, en función del tipo de fichero.

En el caso de las bases de datos, estamos hablando de una **entidad formada por un conjunto de ficheros de distinto tipo**, que además están relacionados entre sí, y que suelen tener un SGBD que facilita y potencia su gestión, a la vez que regula el acceso a la información.

Por tanto, el usuario final de la base de datos no conoce cómo está repartida la información en la base de datos, ni tampoco cómo está organizada. Es una información irrelevante para él, ya que la **función de acceso a la información la resuelve el propio sistema**, que al fin y al cabo es lo que el usuario busca: eficacia, comodidad y ausencia de problemas que impliquen pérdida de tiempo.

Es un hecho evidente que sea necesario el uso de una base de datos (y SGBD) para la **correcta gestión y tratamiento de grandes volúmenes de datos**, aunque esto no quiere decir que sea la mejor solución en todos los casos. Es decir, en caso de que las necesidades de almacenamiento y acceso a la información sean muy reducidas (caso que puede ser válido en un entorno doméstico, por ejemplo) quizás no sea la mejor opción el utilizar una base de datos muy potente con un SGBD avanzado, y por tanto complicado. Sin embargo, es cierto que hoy en día, como se indicó

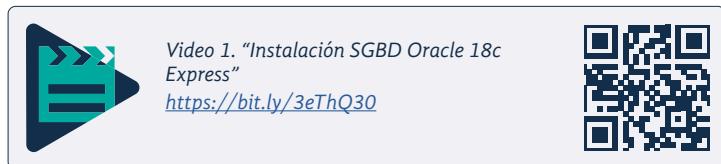


anteriormente, para estos casos existen SGBD muy livianos que se adaptan perfectamente a estas condiciones, como puede ser SQLite, que además tiene soporte para Android.

Por otro lado, un SGBD ofrece posibilidades muy amplias y potentes, al permitir desarrollar aplicaciones específicas para los diferentes casos que nos pudieran surgir, siendo mucho más versátil. Además, los SGBD suelen contar con un interfaz intuitivo, y se adapta a la gran mayoría de los entornos de manera automática.

Quizá el gran aporte de un SGBD es la **posibilidad de poder utilizar lenguajes estándar** de consulta, especialmente SQL, que además puede ser combinado con otros lenguajes (JAVA o PHP) para construir funcionalidades más complejas y potentes.

Además, como se vio anteriormente, hoy en día **existen gran variedad de SGBD** adaptados a todos los mercados y casuísticas.



## / 8. SGBD libres y comerciales

Como ya se ha visto previamente, existe en el mercado una amplia cantidad de SGBD, de diferentes tipos y características. En este punto vamos a describir varios de ellos, los más conocidos y extendidos actualmente.

### 8.1. SGBD libres o de código abierto

Los principales son los siguientes:

- **MySQL:** es un SGBD relacional, considerado uno de los más conocidos del mundo. Soporta varios lenguajes de programación (Java, C++, Python, PHP, etc) y se caracteriza por estar muy relacionado con aplicaciones web. Tiene licencia libre pero también comercial (Oracle), así que está a caballo entre ambos, aunque nació como sistema de código libre. Fue desarrollado por MySQL AB, que fue adquirida por Oracle. El SGBD MariaDB nace a partir de MySQL, a raíz de una escisión de varios desarrolladores.
- **PostgreSQL:** está considerado un SGBD muy avanzado, caracterizándose por ser un sistema relacional orientado a objetos multiplataforma. Por lo tanto, soporta varios lenguajes de programación (Perl, Python, PHP, Java, etc). Ha sido desarrollado desinteresadamente por una comunidad de programadores.
- **SQLite:** se trata de un gestor muy liviano pero potente, con tiempos de acceso muy reducidos. Soporta diversos lenguajes de programación y es multiplataforma. Está muy extendido en aplicaciones móviles, y fue desarrollado en C.
- **MongoDB:** es una base de datos no relacional (NoSQL) distribuida, diseñada especialmente para aplicaciones en la nube. También es multiplataforma.
- **Firebird:** es reconocido por su buena gestión de la concurrencia y por un consumo de recursos muy bajo. Aún así, es potente, multiplataforma y tiene soporte para varios lenguajes de programación. Es un SGBD del tipo relacional.



Fig. 7. Logo MySQL



## 8.2. SGBD comerciales

Existe gran variedad de SGBD comerciales, de varias clases y tipos. En este punto se describen las principales características de los más extendidos:

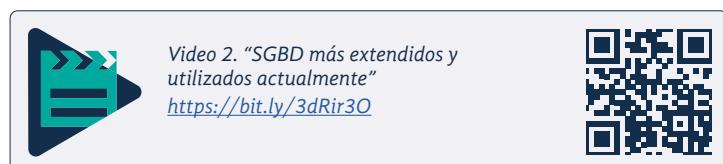
- **Oracle:** probablemente sea el más reconocido y utilizado a nivel mundial. Es un SGBD objeto - relacional potente, robusto, seguro y flexible, basado en arquitectura cliente/servidor. Está muy extendido en grandes empresas y corporaciones importantes por su fiabilidad, robustez y soporte principalmente. Oracle, además, es propietaria de Sun Microsystems (Java) y MySQL.
- **Microsoft SQL Server:** creado por Microsoft, está también muy extendido. Es un SGBD relacional que se caracteriza por la alta disponibilidad y rapidez en la comutación. Además, se integra muy bien con Microsoft Server, lo que le aporta robustez. Además, es escalable, y cuenta con distintas versiones adaptadas al uso que se pretenda dar: DataCenter, empresa, Web, Business Intelligence, etc.



Fig. 8. Logo ORACLE

Permite además su utilización en máquinas virtuales de Azure, aportando seguridad y facilidad de gestión.

- **Sybase – SAP:** se caracteriza por ser también un SGBD relacional, escalable y que ofrece buen soporte a bases de datos con gran cantidad de datos. Cuenta además con opciones para implementar funcionalidades en la nube, y existen diferentes paquetes en función de las necesidades que se deseen cubrir. Soporta diferentes plataformas: Windows, Linux, Unix, Solaris, etc.
- **DB2:** desarrollado por IBM en los setenta, es un SGBD multiplataforma con un motor potente desarrollado por la propia IBM. Destaca por la capacidad de automatización de tareas, lo que redunda en una mayor rapidez de respuesta. Cuenta con diferentes versiones, incluso con una versión gratuita con funcionalidades más limitadas.



## / 9. Caso práctico 2: “Comparativa SGBD”

**Planteamiento:** Necesitamos ampliar nuestros conocimientos sobre las diferentes alternativas de SGBD que existen actualmente en el mercado, para poder seleccionar uno de cara a implementar la base de datos que estamos construyendo para nuestro amigo.

**Nudo:** Como diseñadores de la misma, necesitaremos recabar información acerca de algunas características de los SGBD de nuestra elección, en total 7 SGBD distintos a elegir entre todos los existentes.

La información que necesitamos de cada uno, a fin de establecer una comparativa, es la siguiente:

- **Características principales**
- **Ventajas**
- **Inconvenientes**

**Desenlace:** Una posible solución podría ser la tabla que se muestra a continuación, en la que se recogen los SGBD más conocidos, aunque existen muchos otros en el mercado que también pueden resultar interesantes.



SGBD	Características	Ventajas	Inconvenientes
<b>ACCESS</b>	Pertenece a microsoft. Es muy gráfico. Métodos simples y directos con formularios para trabajar con la información.	Asequible para personas con poco manejo con las bases de datos. Crea varias vistas para una misma información.	No es multiplataforma. No funciona con bases de datos grandes. Tampoco para registros como para usuarios.
<b>SQLite</b>	Los tipos de datos se asignaban a valores individuales y no a la columna como la mayoría de los SGBD.	Multiplataforma. No requiere configuración. Acceso muy rápido. No require servidor.	El dinamismo de los datos hace que no se portable a otras bases de datos. Falta de clave foráneas.
<b>SQL Server</b>	Software propietario. El lenguaje es TSQL.	Multiplataforma aunque pertenezca a Microsoft. Transacciones.	Utiliza mucha ram. Tamaño de página fijo y pequeño. Relación calidad/precio inferior a Oracle.
<b>MYSQL</b>	Pertenece a Oracle. Licencia GPL/ licencia comercial.	Agrupación de transacciones. Distintos motores de almacenamiento. Instalación sencilla.	No tiene soporte. Capacidad limitada.
<b>POSTGRESQL</b>	Tiene la extensión POSTGIS para bases de datos especiales.	Código abierto y gratuito, multiplataforma. Gran volumen de datos. Transacciones, disparadores y afirmaciones.	Respuesta lenta. Requiere hardware. No es intuitivo.
<b>ORACLE</b>	Dispone de su propio lenguaje PL/SQL. Soporta bases de datos de gran tamaño.	Es el más usado a nivel mundial. Multiplataforma. Es intuitiva y fácil de usar.	Precio muy elevado. Elevado coste de la información, tratado por trabajadores formados por oracle.

*Tabla 1. Nivel y características estructura de un SGBD.*



## / 10. Resumen y resolución del caso práctico de la unidad

Durante este tema, se ha estudiado el **concepto de los SGBD**, comenzando por el origen de los mismos para poder comprender su evolución y la motivación de su desarrollo y expansión.

Por otro lado, también hemos analizado la **estructura** (nivel físico, conceptual y externo) y **componentes** principales de los SGBD. Se han repasado de manera pormenorizada los diferentes **tipos de SGBD** en función de diferentes criterios: según su modelo, su tamaño, su forma de ejecución o en función de la ubicación de la base de datos.

Posteriormente se ha explicado en detalle el **funcionamiento** paso por paso, desde que el usuario genera una petición hasta que el sistema le contesta, indicando los procedimientos que ocurren en cada caso.

A continuación, se han analizado las diferencias entre los **SGBD y los ficheros clásicos** y, por último, se han descrito las **características principales de los SGBD más importantes y extendidos actualmente**. Sin embargo, la lista de SGBD existentes es muy extensa, e incluso dentro de cada SGBD descrito existen múltiples versiones y posibilidades (por ejemplo, Oracle), cuyo contenido es prácticamente inabarcable. Por lo tanto, te animamos a que investigues por tu cuenta para poder ampliar la información relacionada con este tema, si lo consideras de tu interés.

### Resolución del caso práctico de la unidad.

En el caso práctico inicial se planteaban diferentes cuestiones relacionadas con los SGBD, algunas de las cuales han sido tratadas directamente en el tema, y otras se pueden deducir tras estudiarlo.

A continuación, se contesta una por una a las cuestiones indicadas, aunque en algún caso la respuesta puede depender del criterio subjetivo de cada uno:

- **¿Cuál sería el SGBD más recomendable para utilizar?**

En este caso, la respuesta debería ser “depende”, ya que hay ciertos SGBD que se adaptan mejor que otros a unas casuísticas determinadas. También dependerá mucho de los objetivos que se persigan y de cómo se diseñe y administre la base de datos.

- **¿Cuál de ellos ofrecerá mejores prestaciones y tecnología?**

De igual forma, después de todo lo estudiando en el tema, deberíamos contestar con un “depende” a esta pregunta, ya que es cierto que dependerá del caso y la problemática concreta. En un entorno empresarial más profesionalizado, quizás lo más lógico sea apostar por tecnologías consolidadas como Oracle, MySQL o SQL Server. En otros entornos, puede ser preferible utilizar otro modelo.

- **¿Realmente nos interesa contar con un SGBD para gestionar nuestra base de datos?**

En este caso, la respuesta debe ser afirmativa. Un SGBD es básico hoy en día para poder sacarle todo el rendimiento que puede ofrecer una base de datos, por muy simple que sea su uso. La incorporación de herramientas gráficas y visuales potentes hace que cada vez sean más sencillos de utilizar.



## / 11. Bibliografía

- Oppel, A. (2009): Databases: A Beginner's Guide. Madrid, España: McGraw-Hill.
- Elmasri, R.; Navathe, S. (2007): Fundamentos de Bases de Datos (5.a. ed.). Madrid, España: Pearson Addison-Wesley
- López, I.; Castellano, M.J. y Ospino, J. (2011) : Bases de datos. Madrid, España: Garceta
- Cabrera, G. (2011): Sistemas gestores de bases de datos. Madrid, España: Paraninfo.
- García-Molina, H., Ullman, J. y Widom, J. (2002): Database Systems. New Jersey: Prentice Hall.

## BASE DE DATOS

# El modelo de datos. Fases y modelo E/R

---

# ÍNDICE

/ 1. Introducción y contextualización práctica	3
/ 2. El modelo de datos. Conceptos y tipos	4
2.1. Concepto modelo de datos vs. esquema	4
2.2. Tipos de modelos	5
/ 3. Diseño de una base de datos	5
/ 4. Modelo entidad-relación (E/R)	6
/ 5. Entidades	7
/ 6. Atributos	8
6.1. Tipos de atributos	9
/ 7. Relaciones	10
7.1. Cardinalidad de una relación	10
7.2. Cardinalidad de entidades y roles	11
/ 8. Caso práctico 1: “Base de datos para tienda y almacén”	12
/ 9. Caso práctico 2: “Identificación de entidades, atributos y relaciones en la base de datos de la tienda informática”	13
/ 10. Resumen y resolución del caso práctico de la unidad	14
/ 11. Bibliografía	14

# OBJETIVOS

**Comprender el modelo de datos, su concepto y tipos.**

**Entender el proceso de diseño de una base de datos.**

**Conocer las características del modelo entidad–relación.**

**Comprender el concepto de entidad y su aplicación en modelos entidad–relación.**

**Estudiar e identificar los atributos en el ámbito del modelo entidad–relación y las bases de datos.**

**Conocer y manejar el concepto de relaciones en el esquema de entidad–relación.**

## / 1. Introducción y contextualización práctica

La utilización de símbolos para expresarse es una constante a lo largo de la vida de una persona: el lenguaje, la pintura, la música..., todas ellas son vías de expresión de sentimientos o ideas. En campos como la ingeniería o la arquitectura se utilizan, habitualmente, diferentes tipos de esquemas o representaciones de la realidad: planos, esquemas mecánicos, etc. Es decir, es necesario utilizar símbolos y esquemas que nos ayuden a entender su funcionamiento o cómo se construyen, o que nos ayuden a entender determinados procesos.

Es evidente que en el campo de la informática, el diseño de una base de datos no escapa a estas consideraciones. Sea lo que sea lo que se pretende desarrollar, es necesario establecer un proceso de creación que incluirá diversas etapas: el planteamiento de una necesidad, el objetivo que se pretende conseguir, el diseño de la herramienta, la puesta en marcha, etc. Para cada uno de estos procesos, se utilizarán símbolos o esquemas determinados para ayudarnos a entender mejor lo que se pretende hacer.

En el caso concreto de las bases de datos, es muy importante un correcto diseño previo. Habría que conocer las diferentes fases que intervienen en el propio diseño, así como los modelos de datos que utilizamos en cada una para poder representar la información del mundo real que queremos almacenar en la base de datos. El **modelo de datos** son unas reglas que ayudan a comprender y expresar la realidad, es decir, supone una abstracción de la misma expresada en forma de esquema o a través de símbolos. En este tema, vamos a tratar el modelo de datos para entender el proceso de diseño, e introducir los distintos modelos centrándonos en los que generan bases de datos relacionales, que son las más extendidas.

Escucha el siguiente audio en el que planteamos la contextualización práctica del tema. Encontrarás su resolución en el apartado final.

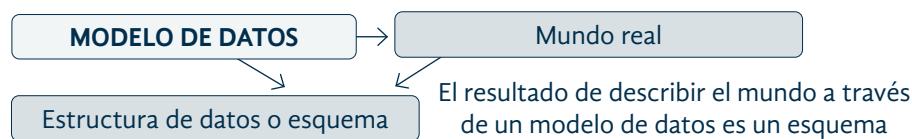


Fig.1. Relación entre modelo de datos, esquema y mundo real.

Audio intro. "Modificación de la información contenida en una base de datos"  
<https://bit.ly/3fX03sU>

This block contains a speaker icon, indicating an audio file is available. The text describes the audio content as "Modificación de la información contenida en una base de datos" and provides a URL: <https://bit.ly/3fX03sU>. A QR code is also present in the bottom right corner of the block.



## / 2. El modelo de datos. Conceptos y tipos

### 2.1. Concepto modelo de datos vs. esquema

En cualquier base de datos que diseñemos, debemos definir **qué es lo que forma parte del problema y lo que no**. Esto es lo que se conoce como **universo del discurso** o **mini-mundo**. Para definirlo, necesitamos abstraernos, simplificarlo de modo que solo representemos los aspectos (datos y restricciones) que se tienen que almacenar.

Es aquí donde entran en juego los modelos de datos que nos ayudan a simplificar la realidad, para expresarla mediante esquemas y comprenderla mejor.

Pero ¿qué entendemos por **modelo**?

Según Dittrich (1994), es “**un conjunto de herramientas conceptuales que se utilizan para describir la representación de la información en términos de datos. Los modelos de datos comprenden aspectos relacionados con la estructura, tipos de datos, operaciones y restricciones**”.

Este concepto no debe confundirse con el concepto de **esquema** que según Dittrich (1994), “**es la descripción de un determinado mini-mundo en términos de un modelo de datos. La colección de datos representados por el esquema es lo que constituye la base de datos**”.

Se utilizan modelos para resolver cualquier problema, pero centrándonos en nuestra misión, que es almacenar datos, estaríamos hablando del **modelo de datos**, que es un **lenguaje utilizado para describir una base de datos**, es decir, describir:

- Estructuras de datos (tipos de datos y relaciones entre ellos).
- Restricciones de integridad (condiciones que deben cumplir los datos basados en la realidad).
- Operaciones de manipulación de los datos.

Todo esto se llevará a cabo con dos “sublenguajes”:

- **Lenguaje de Definición de Datos (DDL)**: que describe las estructuras de datos y las restricciones de integridad. Define cuáles son los elementos que está permitido utilizar y cómo se debe hacer.
- **Lenguaje de Manipulación de Datos (DML)**: sirve para describir las operaciones de manipulación de los datos a través de expresiones y operadores.

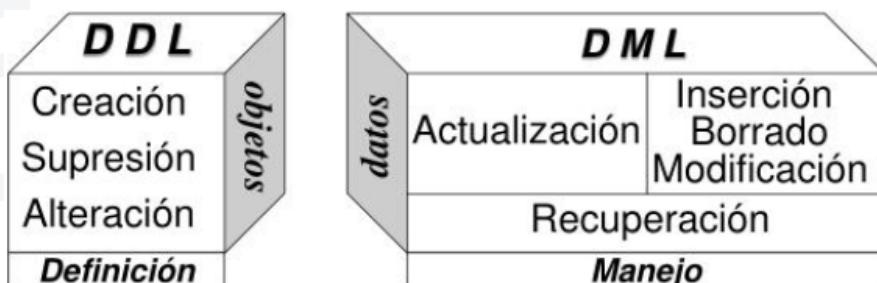


Fig.2. Diferencias entre DDL y DML.



## 2.2. Tipos de modelos

Los diferentes modelos de datos se pueden clasificar según su nivel de abstracción, existiendo, en un primer nivel, los siguientes:

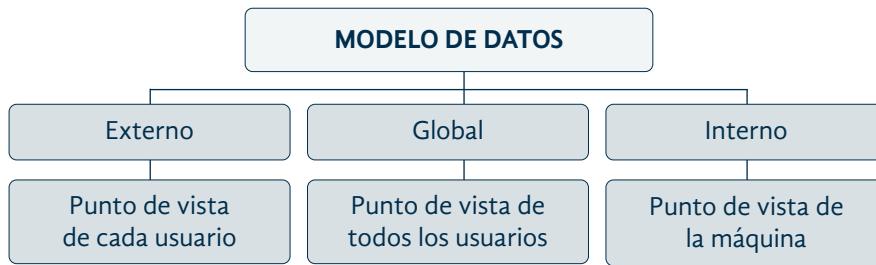


Fig.3. Clasificación de los modelos de datos.

Los **modelos de datos externos y globales** se conocen como **modelos lógicos**, mientras que **los internos** se conocen como **físicos**.

A su vez, los **modelos lógicos** (externos y globales) pueden dividirse en:

- **Conceptuales:** describen el mundo real sin tener en cuenta la tecnología: redes, sistemas operativos, etc.  
Ejemplos: modelo entidad-relación o UML para objetos.
- **Convencionales:** orientados a la implementación del mundo real en un SGBD. Ejemplo, jerárquico, red, relacional.

De manera concreta, **las principales características** de uno y otro se resumen en la siguiente tabla:

MODELO CONCEPTUAL	MODELO CONVENCIONAL
No tienen por qué implementarse en SGBD	Están implementados, típicamente, en SGBD
Son independientes del SGBD	Dependen en gran medida del SGBD
Cuentan con un nivel de abstracción más elevado	Están más cercanos a la máquina
Tienen más capacidad semántica	Tienen menor capacidad semántica
Están más enfocados al diseño de alto nivel	Están más enfocados a la implementación
Interfaz usuario/informático	Interfaz informático/sistema
-	Nivel intermedio entre nivel externo e interno

Tabla 1. Diferencias entre modelo conceptual y modelo convencional.

## / 3. Diseño de una base de datos

El proceso de diseño de una base de datos **consta de varias fases secuenciales**. Este diseño parte de la realización de un análisis de todos los requisitos del problema para definir una **estructura física y lógica**. También, es necesario definir la **información que se quiere almacenar, cómo se almacenará y cómo se relacionará entre sí**.

Por último, se deberá **considerar el formato de almacenamiento** de la base de datos.



Este proceso lo podemos resumir en la siguiente imagen donde se ponen de manifiesto no solo las fases, sino también los esquemas que vamos a aprender a lo largo del curso.

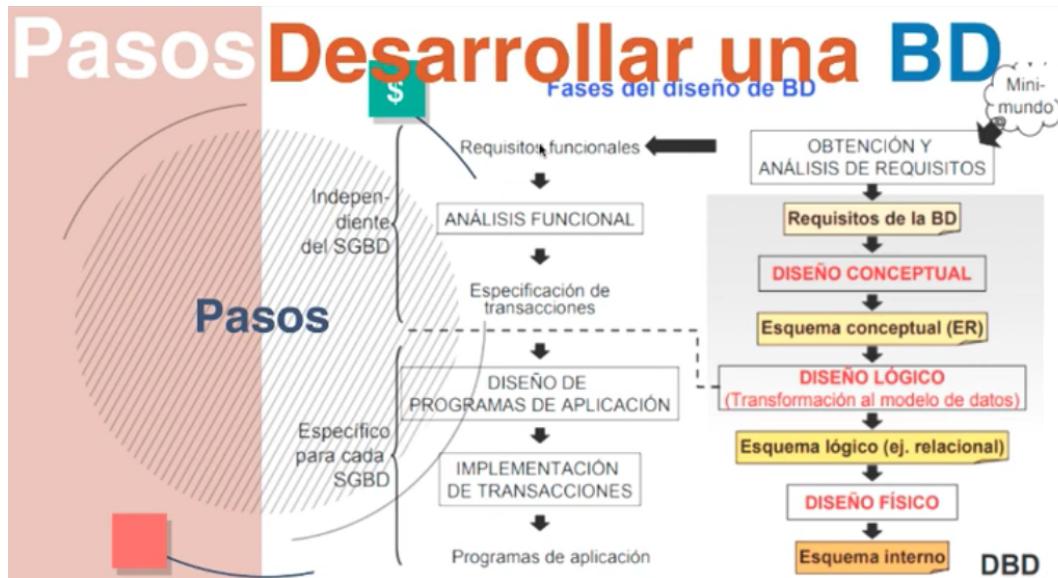


Fig.4. Fases del diseño de una base de datos. <https://youtu.be/aOyKHka8rRY>

Con la primera etapa se consigue establecer qué requisitos y restricciones tendrán que cumplir los datos.

Es fundamental **valorar adecuadamente los usos que tendrá el sistema**, teniendo en cuenta los diferentes tipos de usuarios y el tipo de información que demandarán, para establecerlos cuando hagamos el diseño físico.

En la **fase de diseño conceptual** se realiza el **esquema conceptual** (que en nuestro caso es el esquema entidad-relación) a partir de los datos obtenidos en la primera fase, aplicándose técnicas de modelos de datos.

En la **fase de diseño lógico**, ya será necesario **elegir el SGBD y el tipo de base de datos** (relacional, orientada a objetos, etc.) que se utilizará. Se obtiene del diseño conceptual.

El **diseño físico** es la fase donde **se implementa la base de datos diseñada en el SGBD** y, por tanto, es donde se tiene en cuenta el sistema de almacenamiento necesario para que lo diseñado pueda funcionar, y la arquitectura *hardware* que se utilizará.



## / 4. Modelo entidad-relación (E/R)

Este modelo fue creado en la década de los 70 por Peter P. Chen, con el fin de **establecer un modelo que ayudase a la elaboración de esquemas que unifiquen y simplifiquen la representación de datos del mundo real**.

Sirve para crear **esquemas conceptuales de bases de datos**, para lo que utilizamos el **diagrama entidad-relación**. Corresponde a la **segunda fase** de creación de una base de datos y se realiza una vez hemos recopilado toda la información a gestionar.

Inicialmente, solo se incluían los conceptos de entidad, relación y atributos, pero no eran suficientes para reflejar en el modelo todos los datos y semántica del problema del mundo real. Por esto, se añadieron otras propuestas que forman el llamado **modelo entidad-relación extendido o ampliado** y que estudiaremos en el siguiente tema.



Como hemos adelantado, entre sus características se encuentra la de **representar de forma gráfica el modelo de la base de datos**, para lo que se utilizan diferentes **símbolos y reglas**. A pesar de que la representación de diagramas en el modelo de E/R se estudiará con mayor profundidad más adelante, introducimos en este apartado los símbolos que son más utilizados, ya que nos ayudarán a entender los conceptos teóricos que estudiaremos a continuación.

Los elementos básicos del modelo E/R son **entidades, atributos y relaciones**, los cuales veremos en los siguientes apartados y que tienen la siguiente simbología:

DESCRIPCIÓN	SÍMBOLO
<b>Rectángulos:</b> Representan conjuntos de entidades	 Entidad
<b>Elipses:</b> Representan atributos	 Atributo
<b>Rombos:</b> Representan conjuntos de relaciones	 Relación
<b>Líneas:</b> Conectan los atributos a las entidades y relaciones	 Conexión

Tabla 2. Símbolos del modelo entidad–relación.

## / 5. Entidades

### A. Concepto

Una **entidad** es un **objeto (real o abstracto)** del que queremos almacenar información en la base de datos. Hablamos de **tipo de entidad** para referirnos al **conjunto de entidades, y de entidad, a las ocurrencias o instancias de este tipo de entidad**.

Si lo comparamos con la programación orientada a objetos, son equivalentes a los conceptos de clase y de objeto, respectivamente.

### B. Tipos de entidades

Podemos destacar tipos de entidades **fuertes o regulares**, y **débiles**, como veremos a continuación:

TIPO DE ENTIDAD	DESCRIPCIÓN
<b>Entidades fuertes o regulares</b>	Existen por sí mismas. Ejemplo, entidad PACIENTE en la base de datos de un hospital, cuya existencia no depende de que existan otras entidades.
<b>Entidades débiles</b>	Necesitan de otra entidad para existir, o lo que es lo mismo, su existencia depende de otra entidad. Ejemplo, la entidad TAREA solo puede existir si existe otra llamada TRABAJO.

Tabla 3. Tipos de entidad.



En el caso particular de las **entidades débiles**, se distinguen dos tipos de dependencia de otras entidades:

- » **Dependencia en existencia:** Si la entidad fuerte de la que dependen se elimina, también lo hacen las débiles.
- » **Dependencia en identificación:** En este caso, además de una dependencia en existencia con una entidad fuerte, ocurre que la entidad débil no tiene manera de identificarse por sí misma, siendo necesario hacerlo mediante la clave de la entidad fuerte a la que está asociada.

### C. Representación gráfica

Con respecto a su **representación gráfica**, una entidad es un **rectángulo etiquetado con nombre del tipo de entidad**, y las **entidades débiles** se representan con **dos rectángulos concéntricos con su nombre en el interior**:



Fig.5. Representación de entidades fuertes y débiles.

## / 6. Atributos

### A. Concepto

Los atributos son las **características o propiedades de cada entidad o relación**, cuyo valor distingue a cada una del resto de entidades similares. También son conocidos como **campos**. Ejemplo, una entidad ALUMNO puede tener los atributos nombre, apellidos, teléfono, edad, DNI y dirección. En el siguiente ejemplo se muestra la relación ASISTEN que contiene el atributo NOTA:

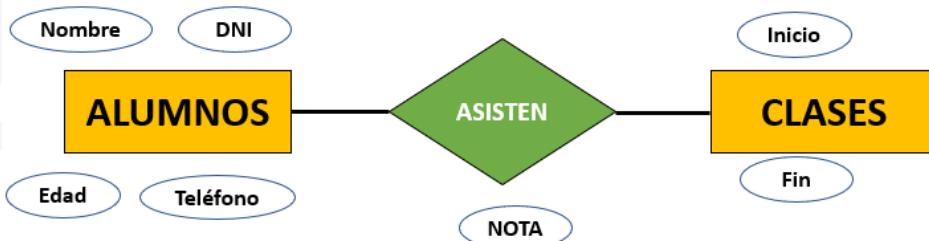


Fig.6. Ejemplo de atributos en entidades y relaciones.

Los valores que pueden tomar los atributos estarán dentro de un rango de valores que se llama **dominio** y que tenemos que definir cuando estamos haciendo el diseño.

### B. Representación de los atributos

En el modelo entidad-relación, los atributos se representan mediante una **elipse**, que se une a su entidad o relación correspondiente mediante un **conector en forma de línea**. La elipse puede tomar **distintos colores y formas** en función del tipo de atributo que se represente, tal y como se verá en el próximo punto. A continuación, se muestran dos ejemplos de cómo se representarían los atributos de dos entidades distintas.

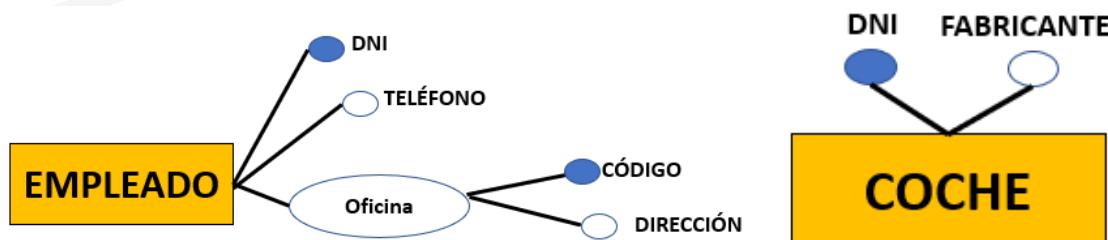


Fig.7. Ejemplo de representación de atributos.



## 6.1. Tipos de atributos

Son dos: obligatorios u opcionales. Los **atributos obligatorios** son los que **deben estar siempre definidos para una entidad o relación y tener valor**, como, por ejemplo, el DNI de un empleado, mientras que **un atributo opcional** puede **tener o no valores**.

De entre todos los atributos existen **uno o varios que identifican de forma única a cada entidad** y se conoce como **clave**. Escucha el siguiente audio para ampliar la información sobre este asunto.



Audio 1. "Claves"  
<https://bit.ly/2KZux2d>



En la siguiente tabla, se resumen todos los tipos de atributos existentes y sus principales características:

TIPO	CARACTERÍSTICAS	
<b>Identificadores o identificativos</b>	Identifican de manera única una entidad o relación. Puede ser el DNI de una persona o el número de cuenta.	
<b>Descriptivos</b>	Son los más habituales. Describen propiedades de una entidad o relación, no necesariamente únicas.	
<b>Derivados</b>	Tienen la particularidad de que su valor se calcula a partir de otros atributos. Por ejemplo, el precio calculado a partir del IVA.	
<b>Multivaluados</b>	Toma distintos valores para cada entidad. Un ejemplo puede ser el e-mail de una persona, que puede tener varias cuentas. En este caso, se debe tener en cuenta el concepto de cardinalidad.	
<b>Compuestos</b>	Pueden ser divididos en otros atributos. Por ejemplo, una dirección puede ser dividida en calle, número y población.	 <pre> graph TD     tipo --- dirección     nombre --- dirección     num --- dirección     </pre>

Tabla 4. Tipos de atributos existentes.

En relación con los multivaluados, la **cardinalidad** de un atributo hace referencia al número mínimo y máximo de valores que puede tomar cada entidad o relación.



## / 7. Relaciones

### A. Concepto

Son las diferentes asociaciones que pueden existir entre las entidades del modelo y, como su propio nombre indica, **relacionan los datos de una y otra**. Define la **manera en la que las entidades interactúan entre sí**. Habitualmente, se nombran según la función que representan, y **se simbolizan con un rombo**, tal y como se muestra en el siguiente ejemplo:



Fig.8. Ejemplo de relación.

Los conceptos grado y cardinalidad de la relación, y cardinalidad de la entidad son importantes para describir adecuadamente las relaciones existentes entre las entidades. A continuación, estudiaremos en qué consisten estos conceptos.

### B. Grado de una relación

Se llaman **entidades participantes** a las involucradas en una relación. Del mismo modo, el **grado** de una relación corresponde al **número** de entidades participantes. En el ejemplo anterior, la relación PUBLICA sería de grado 2. Si le añadiésemos, por ejemplo, la entidad EDITOR, entonces sería una relación de grado 3.

De esta manera, se puede hablar de relaciones **binarias** (asocian dos entidades), **ternarias** (tres entidades), **n-arias** ( $n^o$  de entidades), **dobles** (se utilizan para relacionar las mismas entidades) y **reflexivas** (se utilizan para relacionar casos concretos de una misma entidad).

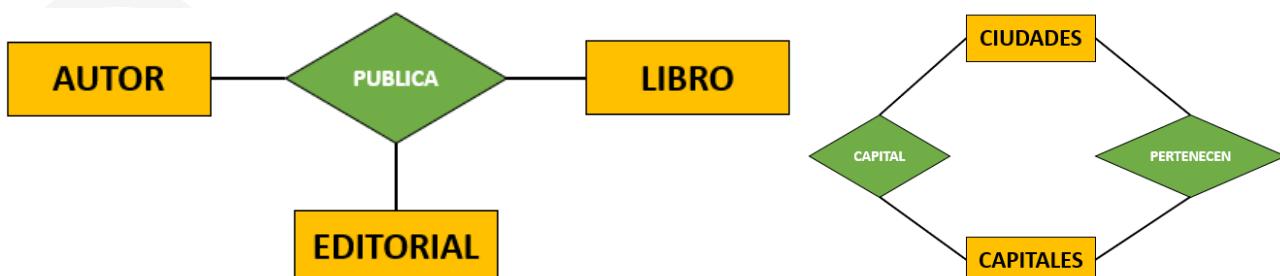


Fig.9. Representación de relación ternaria y doble, respectivamente



Vídeo 1. "Identificación de entidades, atributos y relaciones"  
<https://bit.ly/3qnnfM>



### 7.1. Cardinalidad de una relación

Es la cantidad de veces que una entidad puede participar en una relación. Se podría definir de otra forma, como el número de ocurrencias de una entidad que están asociadas a las ocurrencias de otra entidad. Por ejemplo, en una relación de pertenencia entre un jugador de baloncesto y un equipo, un jugador solo puede pertenecer a un único club de baloncesto, pero el equipo contará en su conjunto con **N jugadores**.



De esta manera, se pueden definir varios tipos de cardinalidades en función de esa relación de ocurrencias entre dos entidades:

TIPO	CARACTERÍSTICAS
<b>Relación 1:1</b>	Es la relación más sencilla. En este caso, una ocurrencia de la primera entidad solo está relacionada con otra ocurrencia de la segunda entidad.  Por ejemplo, un coche será conducido por un único conductor. De igual forma, un alumno tendrá un único expediente asociado, y viceversa.
<b>Relación uno a muchos (1:N)</b>	En este caso, una ocurrencia de la primera entidad está relacionada con varias ocurrencias de la segunda, mientras que una ocurrencia de la segunda solo está relacionada con una de la primera.  Por ejemplo, un profesor puede impartir varias asignaturas, pero una asignatura solo puede ser impartida por un profesor.
<b>Relación muchos a uno (N:1)</b>	Este caso es el contrario del anterior. La primera entidad está asociada con una ocurrencia de la segunda, mientras que una ocurrencia de la segunda se relaciona con muchas de la primera.  Por ejemplo, un profesor pertenece a un único departamento, mientras que un departamento está formado por varios profesores.
<b>Relación muchos a muchos (M:N)</b>	En esta relación, una ocurrencia de la primera entidad está relacionada con varias ocurrencias de la segunda entidad, y viceversa.  Por ejemplo, un alumno puede cursar varias asignaturas, que a su vez son cursadas por varios alumnos distintos.

Tabla 5. Tipos de cardinalidad de una relación.

El tipo de cardinalidad de una relación suele reflejarse en los diagramas de entidad-relación mediante la notación que corresponda entre paréntesis situada sobre el rombo de la relación.

## 7.2. Cardinalidad de entidades y roles

### A. Cardinalidad de entidades

Al igual que hablamos de cardinalidad entre relaciones, también podemos hablar de cardinalidad entre entidades, siendo un concepto similar en ambos casos.

En este caso, podemos **distinguir entre cardinalidad máxima y cardinalidad mínima**:

- **Cardinalidad máxima:** Se refiere al máximo número de relaciones en las que puede existir alguna ocurrencia de la entidad.

Su valor puede ser **uno o un número mayor**. En caso de que se trate de un número mayor a uno, se representa con una N.

- **Cardinalidad mínima:** En este caso, se refiere al número mínimo de relaciones en las que existirá alguna ocurrencia de la entidad.

Su valor puede ser **cero o uno**. El valor cero, normalmente, se indica cuando su aparición en la entidad sea de carácter opcional.



La cardinalidad **se representa entre paréntesis**, indicando la cardinalidad mínima y la cardinalidad máxima. En los diagramas de entidad–relación, se coloca **junto a la entidad que estamos analizando**.

Por ejemplo, un jugador debe pertenecer como máximo a un equipo, pero puede estar sin equipo, por lo que sería una cardinalidad (0,1). En el caso contrario, un equipo debe estar formado por un mínimo de un jugador, mientras que el máximo será de varios jugadores. En este caso, se representará como (1,N).

Un libro puede estar escrito por ninguno, uno o varios autores y un autor, escribir al menos un libro o puede escribir varios.

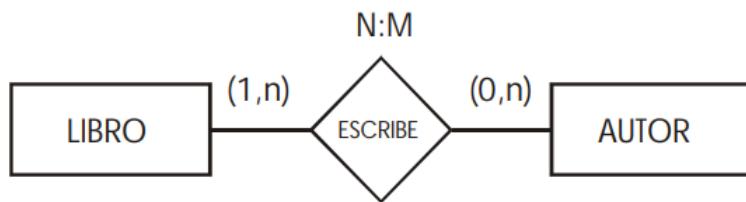


Fig.10. Ejemplo de cardinalidad.

## B. Roles

Es la función que tiene una entidad respecto a una relación determinada. Por ejemplo, un trabajador puede ser empleado y puede ser jefe, en ese caso, se podría obtener esta representación:

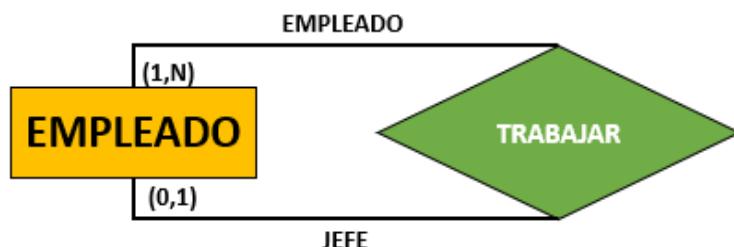
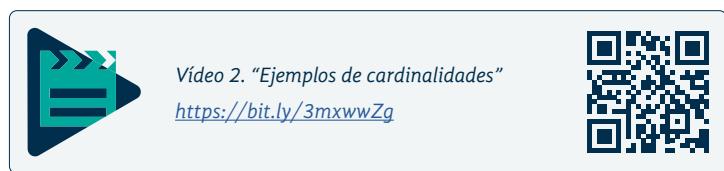


Fig.11. Ejemplo de roles.

Aunque no es obligatorio, en ocasiones, se indican para aclarar las relaciones que dan. Se representa con el nombre sobre las líneas de enlace de una relación.



## / 8. Caso práctico 1: “Base de datos para tienda y almacén”

**Planteamiento:** Siguiendo con el planteamiento de los últimos temas, y en base a lo visto anteriormente, en este, vamos a analizar las actividades que hemos hecho hasta ahora para ayudar a nuestro amigo con su tienda de informática.

**Nudo:** A lo largo de los últimos temas, hemos definido varias estructuras de la base de datos que estamos generando para la tienda.

Ahora que conocemos en mayor detalle las fases de un proceso de diseño de una base de datos, vamos a analizar la información que hemos generado hasta ahora, a través de los casos prácticos realizados en temas anteriores, así podremos revisar que el proceso lo estamos realizando adecuadamente, y verificar qué nos falta. Debemos echar la vista atrás para revisar que estamos completando todos los pasos del proceso, identificando los mismos.



**Desenlace:** En la siguiente tabla, indicamos a modo de resumen las entradas y salidas que hemos generado, relacionadas con las fases de diseño, pero aplicadas a los ficheros como sistemas de almacenamiento para, ahora, solucionar el mismo problema aplicando las fases de diseño y modelado de bases de datos como sistema de almacenamiento.

FASE	SOLUCIONES PLANTEADAS	TEMA
Datos iniciales	Recabar datos sobre la información a almacenar, usos de la base de datos, qué usuarios la utilizarán y para qué, qué necesidades se necesitan cubrir, etc.	Tema 1: Caso práctico nº 1
Diseño conceptual	Tipo de base de datos, modelo de datos a utilizar, arquitectura de la misma (se verá en este tema cómo hacerlo aplicando los conceptos de este diseño que se aplicaron de forma genérica en el tema 2).	Tema 2: Casos prácticos 1 y 2
Diseño lógico	Modelo relacional (se verá en detalle en el tema 6). SGBD basado en Oracle.	Tema 3: Casos prácticos 1 y 2
Diseño físico	Entre las opciones disponibles, seleccionamos un sistema de almacenamiento RAID.	Tema 1: Caso práctico nº 2

Tabla 6. Resumen de entradas y salidas generadas.

Como se puede comprobar, ya hemos trabajado en gran parte de las fases de diseño de una base de datos. Nos queda pendiente aplicarlas de forma específica y, por tanto, la implementación de la misma.

## / 9. Caso práctico 2: “Identificación de entidades, atributos y relaciones en la base de datos de la tienda informática”

**Planteamiento:** En este caso, vamos a poner en práctica la teoría que hemos estudiado a lo largo del tema, utilizando la base de datos de la tienda de informática que hemos implementado a lo largo del curso para nuestro amigo.

**Nudo:** Para ello, vamos a identificar las entidades, relaciones y atributos que contendrá el modelo entidad–relación para nuestra base de datos.

**Desenlace:** Aunque no se pide representarlo (la realización de diagramas E/R se estudiarán en temas sucesivos), una posible solución sería la siguiente:

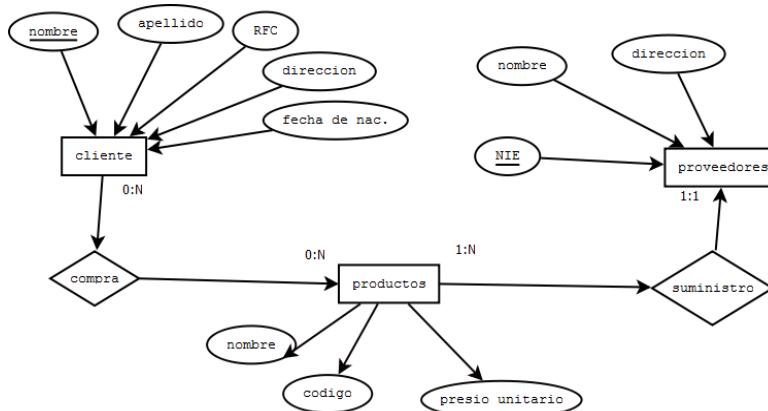


Fig.12. Ejemplo de entidades, relaciones y atributos de la tienda informática.

<http://bdalfonso.blogspot.com/2013/06/modelo-err-ejercicios-resueltos.html>

Además de las entidades identificadas anteriormente, convendría reflejar otras, como empleados, almacén, etc., en función del diseño que cada uno haya realizado para su base de datos.



## / 10. Resumen y resolución del caso práctico de la unidad

En este tema, hemos estudiado las distintas etapas del proceso **de diseño** de una base de datos, así como los **modelos de datos** empleados en las distintas fases, viendo la diferencia entre modelo de datos y esquema. También hemos analizado las partes que componen todo modelo de datos: DDL y DML.

Hemos comenzado con el estudio de la primera fase que corresponde al **modelo entidad–relación**, cuyo uso está muy extendido para el modelado de las bases de datos de distinta tipología, debido a su versatilidad.

En primer lugar, hemos estudiado **en qué consiste** este modelo y cuáles son las principales **partes**. Además, hemos estudiado la **simbología** que suele utilizarse para su representación.

Posteriormente, hemos analizado el concepto de **entidad**, estudiándose, además, los tipos de entidades existentes y cómo se suelen representar gráficamente. De manera análoga, hemos estudiado el concepto de **atributo**, junto con los tipos de atributos existentes y la representación de estos.

Como se ha comentado, las **relaciones** son una de las partes principales de este modelo, junto a las entidades y a los atributos. Se finalizó el tema con el estudio de las mismas, introduciendo los conceptos de grado, cardinalidad de las relaciones, cardinalidad de las entidades y roles.

En posteriores temas, continuaremos con el estudio de este modelo, ya que es un aspecto fundamental en el campo de las bases de datos.

## Resolución del caso práctico inicial

Recordando lo que adelantamos en el audio inicial, exponíamos que, en temas anteriores, habíamos estudiado la importancia de un correcto modelado y diseño de la base de datos, para lo que estudiamos diferentes modelos y opciones existentes.

En el tema que nos ocupa, para el caso del modelo entidad–relación, trasladábamos dos cuestiones para su reflexión: **¿qué características puede aportar para facilitar el modelado de una base de datos?, ¿en qué se basa el modelo?**

Una de las principales características del modelo de entidad–relación es que se adapta a prácticamente cualquier entorno y características de una base de datos, permitiendo modelar casi cualquier sistema que se pretenda diseñar.

Como hemos estudiado a lo largo del tema, el modelo está basado en el uso de entidades, atributos y relaciones que constituyen la base del modelado de un sistema, ya que ayudan a plasmar la realidad desde el punto de vista de una base de datos.

## / 11. Bibliografía

Oppel, A. (2009). *Databases: A Beginner's Guide*. Madrid, España: McGraw-Hill.

Elmasri, R. y Navathe, S. (2007). *Fundamentos de Bases de Datos* (5.a. ed.). Madrid, España: Pearson Addison-Wesley

López, I.; Castellano, M.J., y Ospino, J. (2011). *Bases de datos*. Madrid, España: Garceta

Cabrera, G. (2011). *Sistemas gestores de bases de datos*. Madrid, España: Paraninfo.

Pérez Marqués, M. (2016). *Administración básica de bases de datos con Oracle 12c SQL*. Madrid, España: Alfaomega

**BASES DE DATOS**

**Modelo  
entidad – relación  
ampliado**

---

# ÍNDICE

/ 1. Introducción y contextualización práctica	3
/ 2. Restricciones sobre las relaciones: Exclusividad	4
/ 3. Restricciones sobre las relaciones: Exclusión e inclusión	5
/ 4. Restricciones sobre las relaciones: Restricción de inclusividad	6
/ 5. Caso práctico 1: “Modelado de diferentes restricciones a partir de su descripción”	7
/ 6. Jerarquías: Generalización	7
/ 7. Jerarquías: Especialización	8
/ 8. Jerarquías: Herencia	9
/ 9. Agregación	10
9.1. Tipos de agregaciones	11
/ 10. Caso práctico 2: “Jerarquías”	12
/ 11. Resumen y resolución del caso práctico inicial	13
/ 12. Bibliografía	13

# OBJETIVOS

*Conocer las características principales del modelo entidad–relación.*

*Comprender el concepto de entidad y su aplicación en modelos entidad–relación.*

*Estudiar e identificar los atributos en el ámbito del modelo entidad–relación y las bases de datos.*

*Conocer y manejar el concepto de relaciones en el esquema de entidad–relación.*

## / 1. Introducción y contextualización práctica

En el tema anterior se estudiaron las principales características del modelo entidad – relación que, como se vio, es básico en el diseño y modelo de bases de datos de cualquier tipo. El tema anterior sirve por tanto para sentar las bases para el conocimiento de este modelo, que se ampliarán a lo largo de este tema y del próximo, en el que se estudiará en profundidad la representación gráfica del modelo E/R.

Debido fundamentalmente a la existencia de ciertas limitaciones en la tecnología en los años 70 (cuando fue creado), el modelo entidad – relación tuvo al principio un alcance acotado, que fue ampliado a medida que la tecnología se iba desarrollando. El modelo entidad – relación extendido o ampliado cuenta con todos los elementos del modelo entidad – relación que vimos en el tema anterior, incorporando además otros conceptos, como el de especialización, generalización y jerarquía.

Este modelo ampliado supone un complemento para el existente, y se utiliza para dar respuesta a problemas que exigen mayor dificultad para su representación o modelado. A lo largo de este tema estudiaremos estas mejoras sobre el modelo entidad – relación que estudiamos en temas anteriores, a través de teoría y ejemplos prácticos.

Escucha el siguiente audio en el que planteamos la contextualización práctica del tema. Encontrarás su resolución en el apartado final.



Fig. 1. El modelo E/R ampliado amplía las posibilidades de modelado.



Audio 1 Intro. "Modelo entidad-relación ampliado: casos de uso"  
<http://bit.ly/3bEbIRq>





## / 2. Restricciones sobre las relaciones: Exclusividad

A pesar de que mediante el modelo E/R es posible realizar el modelado de prácticamente cualquier base de datos, hay casos más complejos en los que se necesitan extensiones al mismo, que vienen dadas por el **modelo E/R ampliado**.

Un ejemplo de estas extensiones (el resto se estudiarán en puntos posteriores del tema) son los tipos de restricciones existentes sobre las relaciones. Entre las mismas, se distinguen diferentes tipos:

- **Restricción de exclusividad.**
- **Restricción de exclusión.**
- **Restricción de inclusividad.**
- **Restricción de inclusión.**

Cada uno de estos tipos se analiza con mayor profundidad a continuación.

- **Restricción de exclusividad**

La restricción de exclusividad se da cuando **una misma entidad está involucrada en dos o más relaciones, y cada ocurrencia de la misma sólo puede pertenecer a una de ellas**. De esta manera, no podrá ser parte también de la otra entidad, siendo por tanto exclusiva de una u otra entidad.

De manera gráfica, este tipo de restricción **se representa mediante un arco** que engloba las relaciones y la entidad, tal y como se indica en la siguiente imagen:

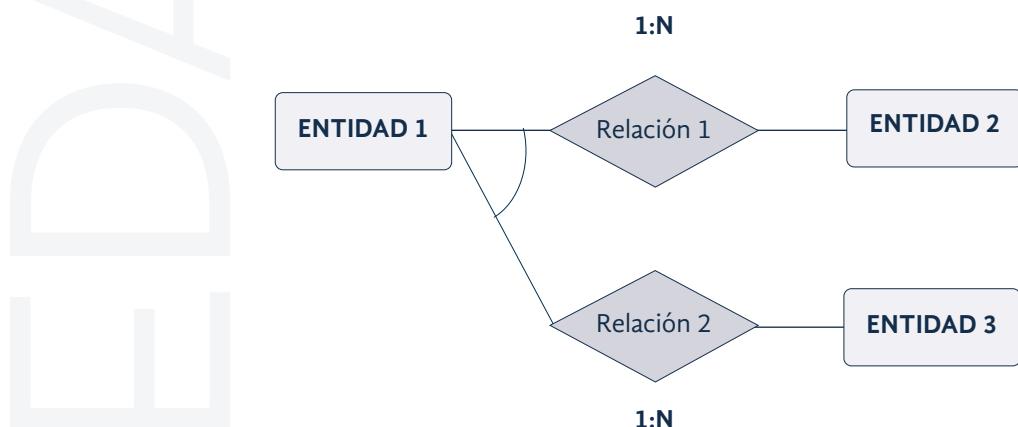


Fig. 2. Restricción de exclusividad.

<https://manuel.cillero.es/doc/metrica-3/tecnicas/modelo-entidad-relacion-extendido/>

**Un ejemplo** de este tipo de restricción podría ser el de una **entidad COCHE**, que está **relacionado con las entidades “GASOIL” y “GASOLINA”** a través de las relaciones “CONSUME” y “GASTA” respectivamente.

Otro posible ejemplo sería el de un entrenador de un equipo que, por un lado, lo dirige y, por otro, también puede jugar en él, pero no de manera simultánea. Por tanto, entre las entidades **ENTRENADOR** y **EQUIPO** existirán dos relaciones **DIRIGE** y **JUEGA** que representan una restricción de exclusividad.



## / 3. Restricciones sobre las relaciones: Exclusión e inclusión

- **Restricción de exclusión**

En este caso, la restricción de exclusión tiene lugar si **ambas ocurrencias sólo pueden usar una única relación para asociarse entre sí**.

**Un ejemplo** puede ser el de las **entidades PROFESOR y CURSO**, cuyas **ocurrencias están unidas entre sí a través de una relación llamada “IMPARTE”, no pudiendo estar relacionado al mismo curso a través de otra entidad, como puede ser “RECIBE”**.

Es lógico, dado que si un profesor está impartiendo un curso no puede estar recibiendo al mismo tiempo. La relación de exclusión **se representa a través de una línea discontinua trazada entre ambas relaciones**:

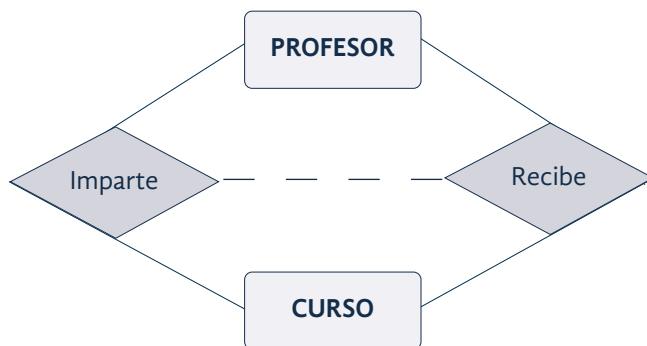


Fig. 3. Ejemplo de restricción de exclusión.

- **Restricción de inclusión**

La restricción de inclusión se utiliza cuando la de inclusividad (se desarrolla en el próximo punto) no es suficiente para modelar el caso concreto que se quiere representar, siendo necesaria una relación más fuerte.

Relacionándolo con el ejemplo anterior del profesor y del curso, se puede dar el caso de que el profesor que va a impartir el curso lo haya recibido de manera previa, por lo que entonces habría que aplicar la restricción de inclusión.

En este caso, se representa gráficamente mediante una flecha discontinua, tal y como se puede verificar en la siguiente imagen:

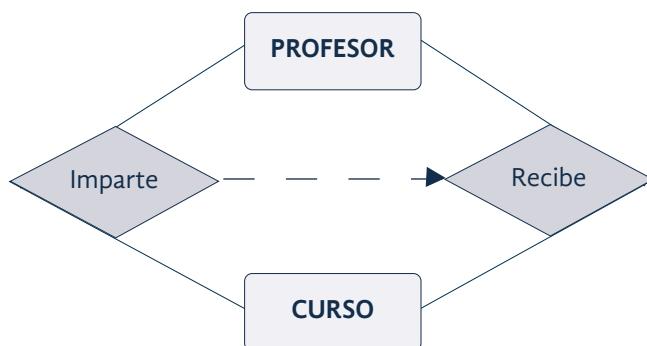


Fig. 4. Ejemplo de restricción de inclusión.



## / 4. Restricciones sobre las relaciones: Restricción de inclusividad

Las restricciones de inclusividad se utilizan en aquellas ocasiones en las que se hace necesario realizar el modelado de los casos concretos en los que, **para asociar dos ocurrencias entre sí a través de una relación, es necesario que hayan estado previamente asociadas a través de otra relación distinta.**

A priori se trata de un concepto un tanto complicado de interpretar, por lo que se podrá comprender de mejor manera mediante un **ejemplo** demostrativo.

Imaginemos que para que un profesor de Formación Profesional pueda impartir la asignatura de Bases de Datos es necesario que haya recibido previamente una formación determinada, concretamente debe haber cursado una Ingeniería Informática o de Telecomunicaciones.

Es decir, los cursos que imparte el profesor no son los mismos que previamente debe haber recibido.

Por tanto, en este caso se puede aplicar claramente una restricción de inclusividad, dado que para cualquier ocurrencia de la entidad PROFESOR que participa en la relación “IMPARTE” es necesario que participe de manera obligatoria en la relación “RECIBE”, de manera previa.

En la siguiente imagen se muestra un ejemplo representativo de este tipo de restricción.

Como se puede apreciar, la **representación gráfica de este tipo de restricción** se lleva a cabo mediante la inclusión de un **arco acabado en flecha**, que va desde la relación que debe cumplirse finalmente hasta la segunda relación.

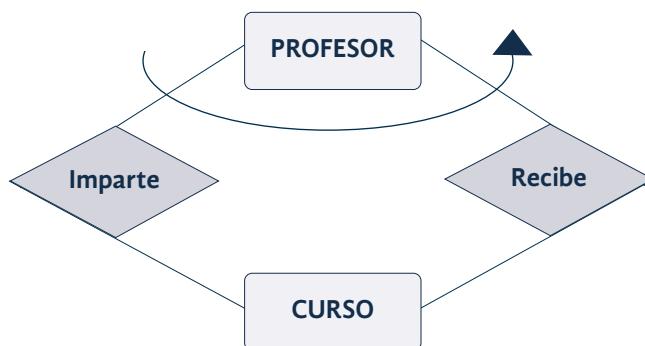


Fig. 5. Ejemplo de restricción de inclusividad.

Junto a la flecha que representa esta restricción de inclusividad, es habitual que se indique la **cardinalidad mínima y máxima de la restricción de inclusividad**.

En el siguiente video se muestran otros ejemplos prácticos de este tipo de restricciones, para reforzar la comprensión del concepto.

Video 1. “Casos prácticos y ejemplos sobre restricciones”  
<http://bit.ly/38H9Lxw>



## / 5. Caso práctico 1: “Modelado de diferentes restricciones a partir de su descripción”

**Planteamiento:** Vamos a practicar con casos prácticos basados en los conceptos estudiados anteriormente en cuanto a las restricciones del modelo E/R extendido. Se plantean varios casos de partida:

- Un profesor no puede impartir una asignatura al mismo tiempo que la recibe.
- Para que un hombre pueda divorciarse de una mujer es necesario que estén casados.
- Los empleados de una de moda pueden ser diseñadores de ropa o bien trabajar en su confección. No puede darse el caso en el que un empleado realice ambas tareas a la vez.

**Nudo:** Se debe identificar en primer lugar el tipo de restricción que se ajusta a cada caso, e implementarse el diagrama asociado.

**Desenlace:** Las restricciones que se adaptan a cada caso son las siguientes:

- Restricción de exclusión.
- Restricción de inclusión.
- Restricción de exclusividad.

La representación de estos diagramas es trivial, debiéndose tener en cuenta el tipo de restricción en cada caso. Para el caso número 1, es equivalente al del siguiente ejemplo:

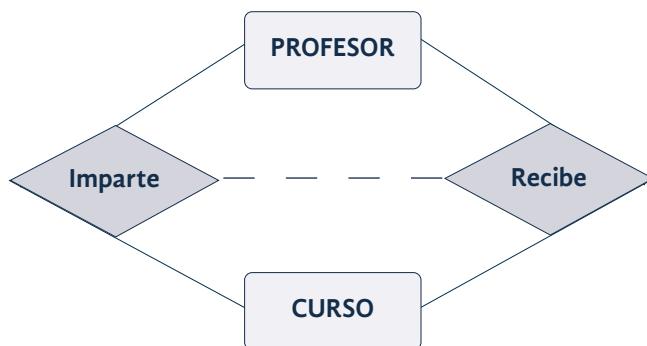


Fig. 6. Ejemplo de relación de exclusión.

## / 6. Jerarquías: Generalización

En este punto se introduce **otra de las extensiones con las que cuenta el modelo E/R extendido**. En este caso, aparecen tipos de relaciones nuevos llamados **jerarquías**, que **ofrecen mayores posibilidades para representar la realidad de mejor manera**. A su vez, estas jerarquías **se basan en tres conceptos básicos**, que están relacionados entre sí:

- Generalización.
- Especialización.
- Herencia.



La idea general es que:

- Las entidades se pueden agrupar en una unidad, **generalización**.
- Al mismo tiempo que una entidad general puede dividirse en entidades más pequeñas, **especificación**.
- Basándose en ambos casos en relaciones de **herencia**.

Se dice que una entidad es una **generalización** de un grupo si cada ocurrencia de cada una de las entidades es, a su vez, una ocurrencia de dicha entidad. La generalización de entidades permite comprobar que algunas entidades de nivel superior pueden contener conjuntos de entidades de nivel inferior. Este hecho está relacionado directamente con el concepto de superclase (o superentidad) y de **subclase (o subentidad)**. El conjunto de entidades de nivel superior es la superclase, y los conjuntos de nivel inferior las subclases, de manera similar a como se muestra en la siguiente imagen:



Fig. 7. Ejemplo de generalización.

Escucha el siguiente audio para conocer más características sobre la generalización:

Audio 1. "Generalización y conceptos relacionados"  
<http://bit.ly/2XEXgws>

Por último, como se puede apreciar en la imagen, la relación de generalización **se representa gráficamente a través de un triángulo**.

## / 7. Jerarquías: Especialización

Como se comentó anteriormente, el concepto de especialización está directamente relacionado con el de generalización. Las **subentidades** de las que hablábamos anteriormente se pueden definir como especializaciones de la entidad principal. Podemos distinguir diferentes tipos de especialización:

- **Especialización Exclusiva:** Cada ocurrencia de la superclase sólo puede materializarse en una de las especializaciones. Se representa con un **arco junto al triángulo**. Un **ejemplo** podría ser un empleado que es directivo, no pudiendo ser técnico ni comercial.

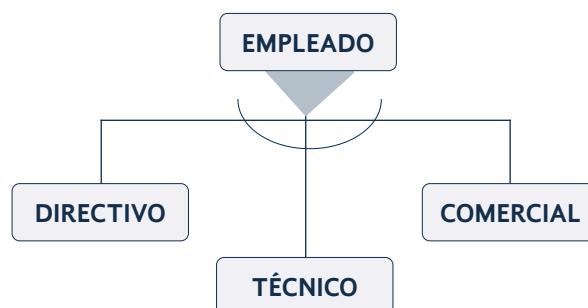


Fig. 8. Especialización exclusiva.



- **Especialización Inclusiva:** Las ocurrencias de la superclase pueden materializarse a la vez en varias ocurrencias de las subclases. Se representa **sin el arco**.

En referencia al **ejemplo** anterior, en este caso el empleado puede ser directivo, técnico y comercial.

- **Especialización Total:** En este caso, es obligatorio que la entidad superclase se materialice en una de las especializaciones.

En este caso se representa mediante un **círculo ubicado sobre el triángulo**.

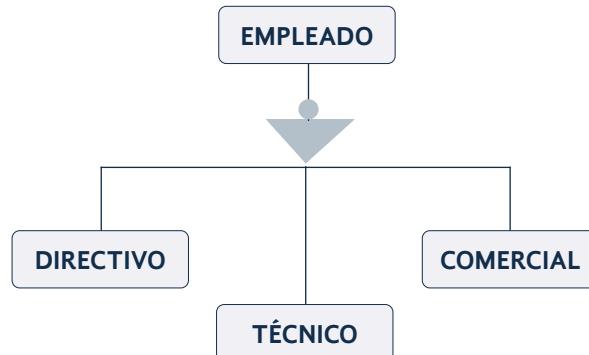


Fig. 9. Especialización total.

- **Especialización Parcial:** De manera opuesta al caso anterior, el hecho de que la superentidad se materialice en una de las especializaciones es opcional.

## / 8. Jerarquías: Herencia

La herencia es un concepto a través del cual se caracterizan las jerarquías que hemos estudiado en los puntos anteriores del tema.

A través de la herencia, **una subentidad puede heredar los atributos de una superentidad**. Además, si una superentidad interviene en una relación también lo harán las subclases asociadas.

**Por ejemplo**, en la siguiente jerarquía, los conserjes y profesores heredan el atributo *num\_personal* y el atributo nombre. El resto de atributos son atributos propios para cada una de las entidades:

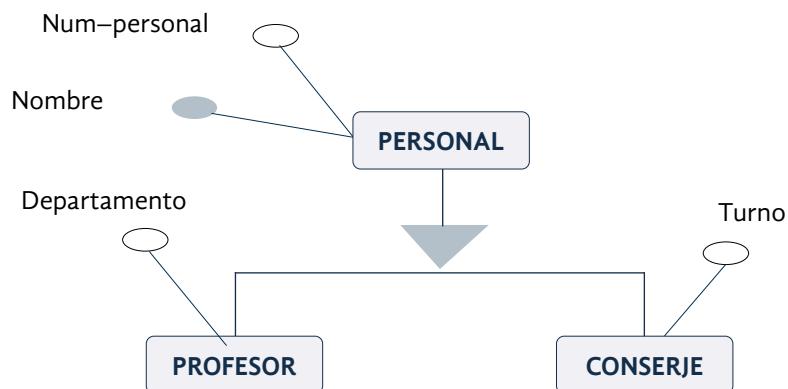


Fig.10. Ejemplo de herencia.



También se puede dar el caso de que cada subentidad tenga una clave de identificación distinta, sin que se llegue a heredar de la superentidad.

En el siguiente **ejemplo** se muestra este hecho. Cada disco y cada libro cuenta con una id distinta. Sin embargo, heredan el atributo precio de la superentidad:

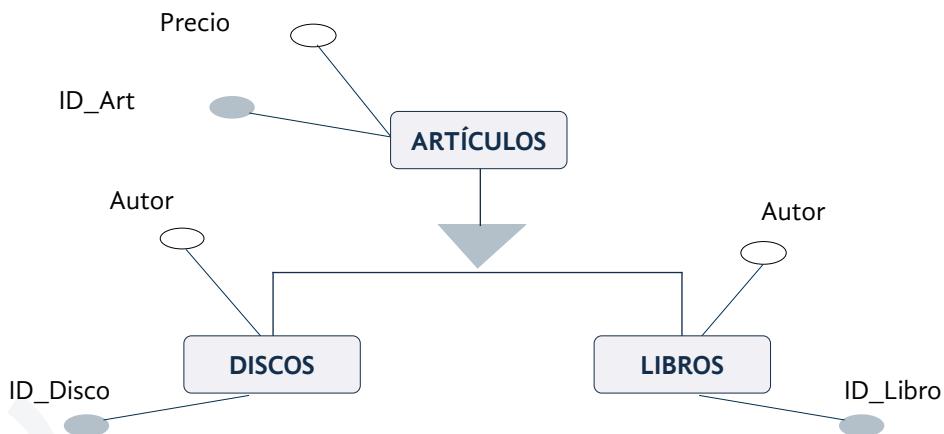


Fig. 11. Otro ejemplo de herencia.

Video 2 "Especialización y herencia"  
<http://bit.ly/39u14G8>

## / 9. Agregación

La agregación es la **tercera y última extensión que incluye el modelo de entidad – relación extendido**, además de las estudiadas en los puntos anteriores del tema.

La agregación en sí misma **es una abstracción**, que permite que se **puedan modelar relaciones entre relaciones**, algo que no es posible mediante el uso del **modelo entidad – relación**. Para poder lograr este hecho, mediante la agregación **las relaciones son consideradas como entidades de más alto nivel**, de manera que se puedan expresar relaciones entre relaciones.

Para comprenderlo mejor veamos un caso concreto de ello. En el siguiente **ejemplo** se muestra un diagrama E/R con relaciones redundantes:

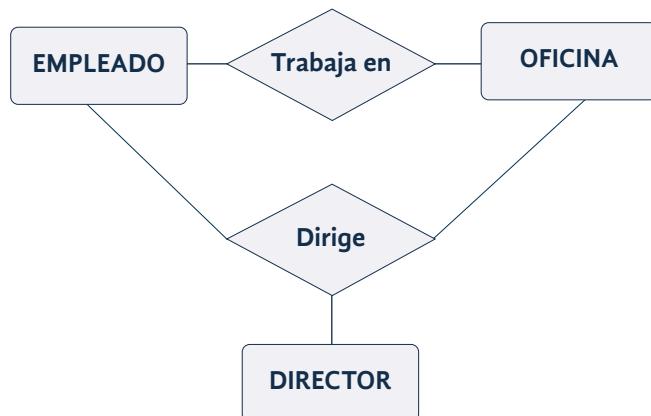


Fig. 12. Ejemplo de diagrama E/R con relaciones redundantes.



Observamos, que tenemos las entidades “EMPLEADO”, “OFICINA” y “DIRECTOR”, y las relaciones “TRABAJA EN” y “DIRIGE”. Existen relaciones redundantes como es evidente, por lo que se podría establecer entidad de más alto nivel formada por lo siguiente:



Fig. 13. Entidad de más alto nivel.

Por lo tanto, aplicando agregación podemos obtener un nuevo diagrama E/R, en el que la agregación se representa mediante un rectángulo que engloba las entidades y relaciones que incluye:

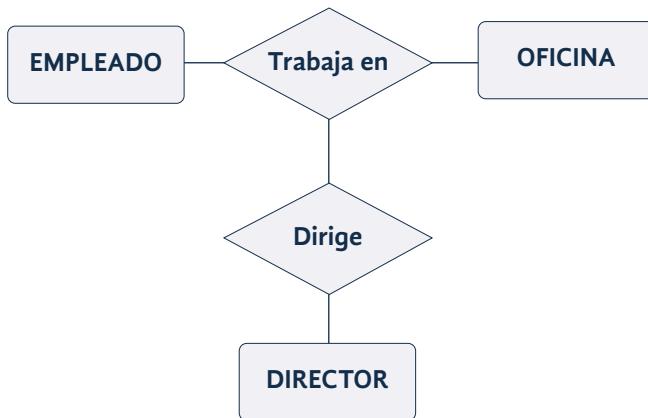


Fig. 14. Diagrama E/R aplicando agregación.

## 9.1. Tipos de agregaciones

Las agregaciones que se pueden considerar en el modelo de entidad – relación ampliado pueden ser de **dos tipos distintos**, que se definen a continuación:

- **Agregaciones Compuesto / Componente:** Se basa en que **permite la representación de un todo (agregado) mediante la unión de diversas partes o componentes**.

Cada una de estas partes o componentes pueden ser diferentes tipos de entidades diferentes, y que además tengan diferentes roles en la agregación. Es un concepto similar al de una clase en la programación orientada a objetos.

**Se representa mediante un rombo** como se puede comprobar en la Figura 15. En dicho **ejemplo** se muestra un todo (coche) que está formado por diversas partes (motor, chasis, ruedas, accesorios).

- **Agregaciones Miembro / Colección:** Es un tipo de agregación que **permite representar un todo o agregado como una colección de miembros, los cuales serán de un mismo tipo de entidad y que tendrán el mismo rol**.

Por tanto, el todo se obtiene de la unión de diversas partes de un mismo tipo y con el mismo rol en la relación.



En este caso, se representa también con **un rombo**, pero del que sólo partirá una única línea, como se puede apreciar en las Figuras 16 y 17.

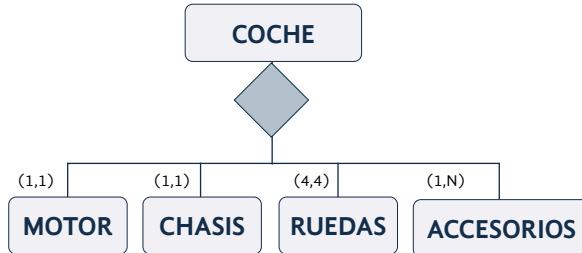


Fig. 15. Agregación Compuesto / Componente.

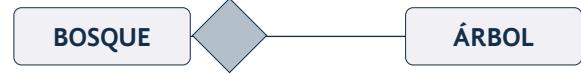


Fig. 16. Agregación Miembro / Colección 1.

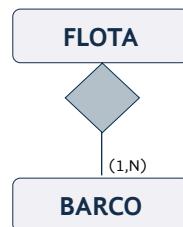


Fig. 17. Agregación Miembro / Colección 2.

## / 10. Caso práctico 2: “Jerarquías”

**Planteamiento:** En este caso vamos a poner en práctica la teoría que hemos estudiado en los últimos apartados del tema, identificando atributos comunes a dos entidades para establecer una jerarquía.

**Nudo:** Partiremos de las entidades “COCHE” y “CAMIÓN”. Cada una de ellas tiene los siguientes atributos:

- Coche: id\_coche, fecha\_fab, precio y peso
- Camión: id\_camion, fecha\_fab, precio y peso

Aplicando los conceptos de generalización, especialización y herencia, ¿cómo podría representarse este esquema?

**Desenlace:** Se podría realizar mediante la creación de una superclase “VEHÍCULO”, que engloba las subclases “COCHE” y “CAMIÓN”. Los atributos comunes (fecha fabricación, precio y peso) podrían asociarse a la superclase, mientras que el atributo id\_coche o id\_camion serían propios de la subclase.

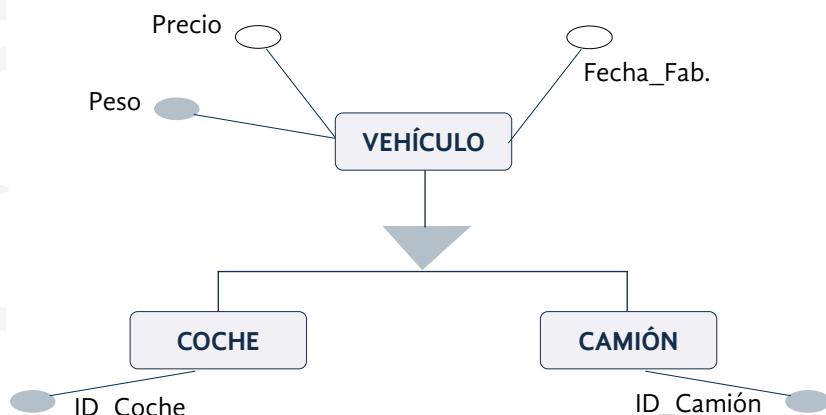


Fig.18. Posible solución del caso práctico.



## / 11. Resumen y resolución del caso práctico inicial

En este tema se ha completado el estudio del **modelo entidad – relación extendido**, que aporta extensiones y, por lo tanto, mayor facilidad de modelado al modelo E/R que estudiamos en el tema anterior.

En primer lugar, hemos estudiado **las restricciones sobre las relaciones: exclusión, exclusividad, inclusión e inclusividad**, analizando sus principales características mediante el uso de ejemplos prácticos. También se ha estudiado de qué manera se representan gráficamente.

Posteriormente, se han analizado los conceptos de **generalización y jerarquías**, verificando las características de la **generalización, especialización y herencia**, tres conceptos clave en el modelo entidad – relación ampliado.

Se ha finalizado el tema con el estudio del concepto de **agregación**, la última de las extensiones que aporta este modelo, y para la que se han introducido los tipos existentes, a través del uso de ejemplos prácticos.

### Resolución del caso inicial

Como se va estudiando a lo largo del tema, existen varios casos prácticos en los que puede ser de aplicación este modelo entidad – relación ampliado. Se podría utilizar cualquiera de ellos.

Como norma general, resultará interesante su aplicación en casos en los que sea útil aplicar alguna de las extensiones que contiene: generalización, especialización, herencia, agregación, restricciones sobre relaciones, etc.

Su utilización dependerá del caso concreto, y sus posibles ámbitos de aplicación son múltiples. Algunos ejemplos de ello podrían ser los siguientes:

- Modelado de un club deportivo.
- Modelado de una empresa multinacional.
- Modelado de una entidad educativa.
- Modelado de una industria de fabricación de componentes y/o materiales.

## / 12. Bibliografía

Oppel, A. (2009): *Databases: A Beginner's Guide*. Madrid, España: McGraw-Hill.

Elmasri, R.; Navathe, S. (2007): *Fundamentos de Bases de Datos* (5.a. ed.). Madrid, España: Pearson Addison-Wesley.

López, I.; Castellano, M.J. y Ospino, J. (2011): *Bases de datos*. Madrid, España: Garceta.

Cabrera, G. (2011): *Sistemas gestores de bases de datos*. Madrid, España: Paraninfo.

Pérez Marqués, M. (2016): *Administración básica de bases de datos con Oracle 12c SQL*. Madrid, España: Alfaomega.

BASE DE DATOS

# **Elaboración de diagramas de entidad-relación. Conceptos del Modelo relacional**

---

# ÍNDICE

/ 1. Introducción y contextualización práctica	3
/ 2. Identificación de los elementos necesarios para la elaboración de los diagramas E/R	4
/ 3. Metodologías y propiedades para la elaboración de diagramas E/R	4
/ 4. Caso práctico 1: “Construcción de un diagrama E/R”	5
/ 5. Modelo relacional	6
5.1. Relaciones, atributos y tuplas	6
5.2. Características, tipos de relaciones y dominio de valores	7
5.3. Grado y cardinalidad	8
5.4. Sinónimos	9
/ 6. Tipos de datos. Juegos de caracteres	9
6.1. Tipos de datos	9
6.2. Juegos de caracteres. Criterios de comparación y ordenación	10
/ 7. Caso práctico 2: “Dominio de valores, grado y cardinalidad”	11
/ 8. Resumen y resolución del caso práctico de la unidad	12
/ 9. Bibliografía	13

# OBJETIVOS

**Conocer e identificar los elementos que se utilizan para elaborar los diagramas de entidad-relación.**

**Comprender el modelo relacional y su terminología.**

**Conocer los diferentes tipos de datos existentes.**

**Manejar con soltura la realización de diagramas E/R mediante el estudio de diversos casos prácticos.**

## / 1. Introducción y contextualización práctica

En los dos temas anteriores se han introducido los principales conceptos relacionados con el modelo entidad–relación, que nos ayudarán a modelar una situación real que queramos representar, de manera independiente al SGBD que se pretenda utilizar. En ambos temas, hemos estudiado conceptos básicos para elaborar estos diagramas, y se han introducido ciertos símbolos que los representan.

La representación gráfica y elaboración de diagramas de entidad–relación nos facilitará la comprensión y posterior modelado del caso que se quiere representar. Para ello se utiliza una simbología normalizada y basada en unos requisitos o condiciones, que estudiaremos a lo largo de este tema. Los diagramas de entidad–relación son una parte fundamental en el estudio de las bases de datos, porque a partir de ellos se realiza el diseño lógico utilizando el modelo relacional.

En este tema, estudiaremos la terminología del modelo relacional para, en el siguiente tema, transformar los diagramas al modelo relacional.

En el siguiente audio planteamos la contextualización práctica del tema. Encontrarás su resolución en el apartado final.

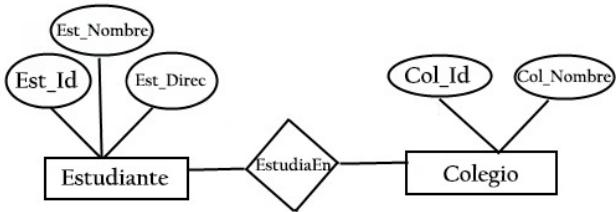


Fig.1. Ejemplo de diagrama de entidad–relación.  
<https://www.lifeder.com/modelo-entidad-relacion/>



Audio intro. "Construcción de diagrama entidad–relación"  
<http://bit.ly/2Lx8Jvp>





## / 2. Identificación de los elementos necesarios para la elaboración de los diagramas E/R

Los pasos previos necesarios para la elaboración de un diagrama E/R se basan en analizar los requisitos del problema, **identificando inequívocamente los diferentes elementos necesarios para su construcción: entidades, atributos, generalizaciones, etc.**

De manera previa a la identificación de estos elementos, es importante realizar una o varias lecturas del caso para interiorizar claramente el objetivo que se pretende alcanzar con el modelado.

ELEMENTO	CONSEJOS	EJEMPLOS
Entidades	Suelen ser sustantivos, que son fundamentales en el problema. Cada ejemplar debe ser distinto del resto de ejemplares, y deben tener las mismas propiedades. Características o propiedades no se convierten en entidades.	Empleado, cliente, vehículo, pieza.
Relaciones	Las formas verbales ayudan a establecer la relación entre entidades. Importante reflejar correctamente la cardinalidad de entidades y relaciones.	Trabaja, compra, contiene.
Atributos y claves	Se identifican los atributos de cada entidad, considerando aquellos que pueden ser claves. Suelen ser adjetivos, identificadores o propiedades referidos al sustantivo entidad.	Edad, domicilio, color.
Claves	En el proceso de identificación de atributos se pueden detectar aquellos que son claves candidatas, escogiéndose entre ellas la principal.	Num_empl, Num_client, matrícula
Jerarquías	Se pueden identificar fácilmente si algún atributo se puede aplicar a varias entidades. A partir de ahí se puede intentar definir una superclase o subclase.	-

Tabla 1. Identificación de elementos para el modelo diagrama E/R.

Una vez identificados los elementos, conviene realizar un listado de los mismos y repasar la pertenencia de cada uno a su categoría. De esta manera, se podrán detectar posibles incoherencias y se podrá depurar el diagrama resultante.

En este punto puede ser aconsejable, o necesario, el **escalado de dudas** que pudieran surgir sobre el problema que queremos modelar.

## / 3. Metodologías y propiedades para la elaboración de diagramas E/R

El siguiente paso una vez identificados los elementos es el de **construir el esquema**. En el siguiente audio se describen las principales **metodologías** existentes para elaborar los esquemas conceptuales.



Audio 1. "Metodologías para la elaboración de diagramas E/R"

<http://bit.ly/39uinXz>





Por otra parte, para poder construir diagramas E/R de la mayor calidad posible es importante que **cumplan** una serie de **propiedades**:

- **Sencillez:** Un diagrama E/R debe facilitar su comprensión, representando la información con la menor complejidad posible.
- **Corrección:** Se deben utilizar adecuadamente los diferentes elementos del modelo E/R, en base a las características del problema que se pretende modelar. Un atributo debe modelarse como tal, y no como una entidad. Del mismo modo, una entidad no debe confundirse con una relación.
- **Contenido:** Lógicamente, el diagrama E/R debe contener toda la información necesaria, es decir, debe ser completo.
- **Legibilidad:** Es una propiedad que mide la facilidad de interpretación del diagrama E/R. Este concepto está relacionado con la disposición y orden de los elementos y conexiones entre los mismos.
- **Escalabilidad:** Un diagrama E/R es escalable si permite introducir ciertos cambios, que pueden ser consecuencia de nuevos requerimientos.

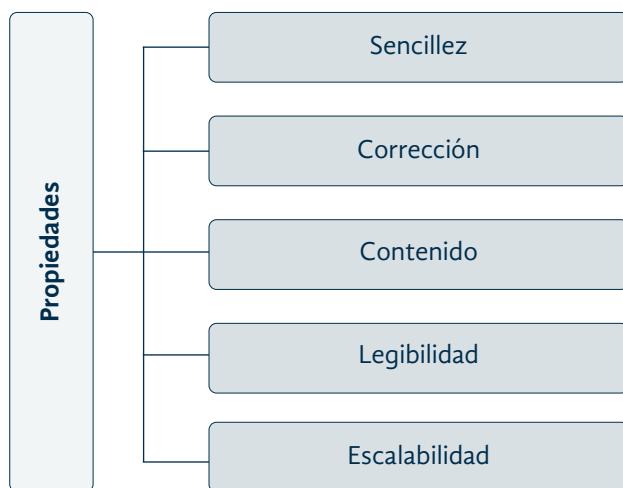


Fig.2. Propiedades de los diagramas E/R.

## / 4. Caso práctico 1: “Construcción de un diagrama E/R”

**Planteamiento:** Vamos a entrenarnos con un caso práctico para elaborar el diagrama entidad-relación completo. Estará basado en un sistema de ventas de una empresa. Se necesita llevar el control de: proveedores, clientes, productos y ventas de la empresa.

Cada proveedor tiene un número, y se desea almacenar su nombre, dirección, teléfono y web. Un cliente también tendrá un id, y además queremos guardar datos como nombre, dirección y varios teléfonos. En cuanto al producto, tendrá también un id único para cada uno, un nombre, un precio, un stock y un proveedor asociado. Se organizarán en categorías, y cada producto solo pertenecerá a una de ellas.

Cada categoría, asimismo, tendrá un id, nombre y descripción.

En cuanto a las ventas, se debe almacenar cada una de ellas con un id, una fecha, un cliente y un precio final. También es interesante almacenar la cantidad de unidades vendidas y el precio por producto.

**Nudo:** Debemos identificar las entidades, atributos, relaciones y cardinalidad según el caso planteado, y representar el diagrama de entidad-relación resultante.



**Desenlace:** Un posible ejemplo de solución para el caso planteado puede ser la mostrada en la siguiente imagen:

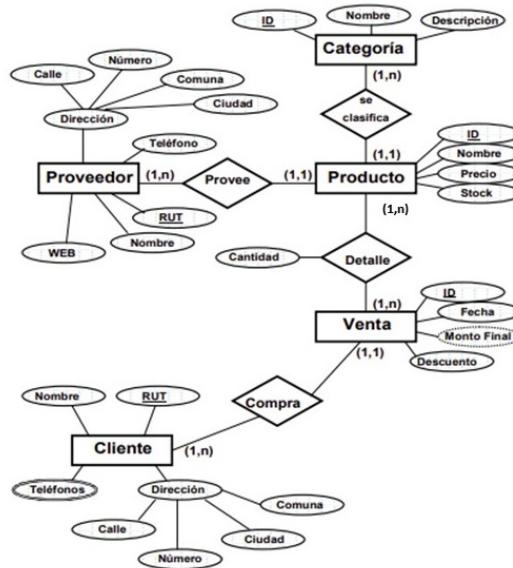


Fig.3. Posible resolución del caso práctico número 1.

## / 5. Modelo relacional

Como hemos visto anteriormente, el modelo relacional **nos permitirá esquematizar información real de manera sencilla**. Fue creado en la década de los 70 por Edgar Frank Codd en los laboratorios IBM, siendo aún, a día de hoy, un modelo muy utilizado. Se llamó relacional debido, en gran parte, al concepto de relación en matemáticas aplicado al producto cartesiano entre dos conjuntos.



### 5.1. Relaciones, atributos y tuplas

El objetivo de este modelo es ser **eficiente y fácil de comprender**. Se caracteriza por **organizar la información en tablas**, cada una de las cuales tiene datos que se refieren a un elemento del mundo real (cliente, proveedor, persona, empleado, factura, etc.). Cada tabla se puede considerar como una **relación**, en la que se puede distinguir entre atributos y tuplas:

- **Atributo:** Correspondría a la **columna** de la tabla. Representa el **nombre de la información almacenada**. Por ejemplo, en una tabla de clientes, los atributos serían el DNI, nombre, apellidos, etc.
- **Tuplas:** Corresponden a las **filas** de la tabla. Representa a **cada uno de los elementos de la tabla**, correspondiendo a la misma idea de Registro que vimos en el tema 2.

Por ejemplo, en el caso de la tabla de clientes, las tuplas serían el DNI, nombre y apellidos de cada uno de ellos, es decir, los registros de cada cliente.

Es evidente, por tanto, que cada tupla debe tener correspondencia con el elemento del mundo real que se pretende representar, no pudiendo existir dos tuplas iguales.



ID HABITACIÓN	NÚMERO DE CAMAS	TIPO DE CAMA	UBICACIÓN
1	2	Doble	Piso 1
2	2	Doble	Piso 1
3	2	Individual	Piso 2
4	1	Doble	Piso 2
5	1	Doble	Piso 3

Tabla 2. Ejemplo de tabla "Habitaciones".

En el ejemplo anterior, los atributos serían ID habitación, nº de camas, tipo de cama y ubicación. Por otro lado, las tuplas serían las diferentes habitaciones, de la 1 a la 5.

La intersección entre atributo y tupla (columna y fila) contiene un dato concreto. Con el fin de relacionar datos de distintas tablas, se utilizan claves.

## 5.2. Características, tipos de relaciones y dominio de valores

**A. Características:** En el modelo relacional existen ciertas **normas** a la hora de utilizar una relación, muchas de ellas son lógicas y evidentes:

- » Cada relación o tabla debe tener un **nombre diferente**.
- » Solo puede haber un **único valor para cada atributo**.
- » El **dominio de datos** debe ser **el mismo para un atributo**.
- » El **nombre de los atributos debe ser diferente en cada tabla**, aunque puede ser igual en relaciones distintas.
- » **No pueden repetirse las tuplas**.
- » El **orden** de las tuplas y de los atributos **no es relevante**.

ID HABITACIÓN	NÚMERO DE CAMAS
1	2
2	2

ID HABITACIÓN	NÚMERO DE CAMAS
2	2
1	2

ID HABITACIÓN	NÚMERO DE CAMAS
1	2
2	2

NÚMERO DE CAMAS	ID HABITACIÓN
2	1
2	2

Tabla 3. Ejemplos de tabla "Habitaciones".



**B. Tipos de relaciones:** En cuanto a los **tipos de relaciones**, podríamos distinguir entre:

- » **Temporales:** Las elimina directamente el sistema cuando no las necesita. Se utilizan como tablas auxiliares por el sistema.
- » **Persistentes:** Solo pueden ser borradas por los usuarios (administradores). Hay tres subtipos:
  - › **Bases:** Contienen datos y metadatos, siendo la base del modelo relacional. Son tablas independientes, que se crean en base a una estructura con datos.
  - › **Vistas:** Contienen consultas (típicamente en SQL) a tablas base, a partir de cuyo resultado se genera una tabla.
  - › **Instantáneas:** Son tablas vistas que almacenan los datos que muestran, y también la consulta a través de la cual se creó.

### C. Dominio de valores

El dominio de valores se refiere a que se debe **utilizar un valor determinado para cada atributo**, en función de cada caso concreto. Es decir, no sería válido un valor de tipo fecha o monetario para un atributo DNI, nombre o apellido.

Por tanto, el **dominio de valores** se puede entender como **el conjunto de los mismos que puede tomar un atributo**, tratándose de una regla que se debe definir en el proceso de diseño.

El dominio de valores es un concepto similar al tipo de dato, pero no es exactamente lo mismo, ya que el dominio de valores puede ser **más restrictivo** que el tipo de datos. Por ejemplo, para el atributo DNI, el dominio de valores será numérico, pero tendrá un rango de valores determinado, no siendo válido cualquier número (ejemplo: 11111111).

## 5.3. Grado y cardinalidad

El **grado define el tamaño de una tabla** (o relación) **y cuántas columnas** (es decir, atributos) **puede contener**. Es evidente que las relaciones con un número de columnas mayor, también, serán más complejas de tratar.

Además del número de columnas, es necesario definir el **número de tuplas** (o filas) que puede tener la relación (o tabla). A este dato se le llama **cardinalidad**. De igual manera que en el caso del grado, a mayor número de filas, más complicada de tratar será la tabla.

Existen bases de datos que contienen tablas con miles o incluso millones de registros, necesitándose un *software* y *hardware* muy potentes para poder tratarlas adecuadamente.

ID HABITACIÓN	NÚMERO DE CAMAS	TIPO DE CAMA	UBICACIÓN
1	2	Doble	Piso 1
2	2	Doble	Piso 1
3	2	Individual	Piso 2
4	1	Doble	Piso 2
5	1	Doble	Piso 3

Tabla 4. Relación de grado 4 y cardinalidad 5.



## 5.4. Sinónimos

Todos los términos que hemos visto hasta ahora durante el tema tienen diferentes sinónimos, según la nomenclatura que se haya utilizado en cada entorno. Este hecho nos va a ayudar a relacionar entre sí los diferentes conceptos vistos hasta ahora.

Los más utilizados son estos tres:

- **Modelo relacional:** Relación, tupla, atributo, grado y cardinalidad.
- **Tablas:** Tabla, fila, columna, número de columnas y número de filas.
- **Registros:** Ficheros, registros, campos, número de campos y número de registros.

MODELO RELACIONAL	NOMENCLATURA DE TABLA	NOMENCLATURA DE FICHEROS
Relación	Tabla	Fichero
Tupla	Fila	Registro
Atributo	Columna	Campo
Grado	Número de columnas	Número de campos
Cardinalidad	Número de filas	Número de registros

Tabla 5. Resumen de sinónimos entre modelo relacional, tabla y nomenclatura de ficheros.

## / 6. Tipos de datos. Juegos de caracteres

### 6.1. Tipos de datos

Como se ha indicado a lo largo del tema, el objetivo del **modelo relacional** es **representar información relacionada entre sí y relativa al mundo real**, la cual nos interesa almacenar.

Además, se definió el concepto de dominio de datos comparándolo con el de tipo de datos, ya que son conceptos que pueden tender a generar confusión.

Lo que es claro y evidente es que al realizar el diseño de una relación, debemos asignar **qué tipo de datos va a contener cada uno de los atributos de la misma**, aplicándoles un dominio de datos en caso de ser necesario.

En este punto conviene indicar que, **en función del lenguaje de programación que se utilice para manejar la base de datos, la manera de nombrar los tipos de datos puede variar**.

Por ejemplo, un dato de tipo entero se codifica como “int” en Java, mientras que en PHP se conoce como “integer”.



En la siguiente tabla se muestran los tipos de datos que se utilizan más comúnmente en el ámbito de las bases de datos, aunque no son los únicos:

TIPO DE DATO	CARACTERÍSTICAS	EJEMPLO
Numérico	Pueden ser enteros (int) o con decimales (float, double).	1545
Autonumérico	Es un valor secuencia que asigna automáticamente el SGBD al añadir una tupla.	1
Texto	Cadenas o juegos de caracteres. Pueden estar formadas por números, letras o símbolos. CHAR o VARCHAR.	Andrea
Booleano	Solo dos opciones: verdadero (true) o falso (false), sí o no.	Sí
Moneda	Es del tipo numérico, pero referente a cantidades de dinero (euros, dólares, yenes, etc.).	1.500€
Fecha /DATE	Almacena fechas en un formato determinado (DD/MM/AAAA, DD/MM/AA).	15/06/2001
DATETIME	Combinación de fecha y hora.	2020-04-15 15:30:15
TIME	Almacena una hora.	15:30:15
Memo	Almacena cadenas de texto de mayor longitud que Texto. No permite búsquedas ni ordenaciones.	Andrea López
Objeto OLE	Contiene gráficos, imágenes, textos, etc., que han sido creados por otras aplicaciones. No permite búsquedas ni ordenaciones.	

Tabla 6. Resumen de los principales tipos de datos que pueden utilizarse.

## 6.2. Juegos de caracteres. Criterios de comparación y ordenación

Para la extracción de información de una base de datos **es habitual la utilización de caracteres que ayuden a realizar consultas de comparación y/u ordenación de los datos**, como se ampliará en próximos temas.

Es importante, por tanto, tener en cuenta los diferentes **operadores de comparación**:

OPERADOR DE COMPARACIÓN	UTILIZACIÓN
=	Igual que
>	Mayor que
<	Menor que
>=	Mayor o igual que
<=	Menor o igual que

Tabla 7. Principales operadores de comparación.



También existen **operadores lógicos**, que se utilizan para comprobar si una condición es cierta o no, devolviendo TRUE o FALSE:

OPERADOR LÓGICO	UTILIZACIÓN
<b>ALL</b>	TRUE si todas las comparaciones son ciertas
<b>AND</b>	TRUE si ambas expresiones lo son
<b>ANY</b>	TRUE si cualquiera es TRUE
<b>BETWEEN</b>	TRUE si el operando está entre un intervalo
<b>EXISTS</b>	TRUE si contiene el elemento indicado
<b>IN</b>	TRUE si es igual a uno de la lista
<b>LIKE</b>	TRUE si coincide con el patrón indicado
<b>NOT</b>	Invierte el valor del operador
<b>OR</b>	TRUE si cualquiera de las dos es TRUE

Tabla 8. Principales operadores lógicos.

En el siguiente audio se detallan las diferentes operaciones que se pueden realizar en el modelo relacional.



Audio 2. "Operaciones en el modelo relacional"  
<https://bit.ly/3hrkoX4>



En cuanto a los **criterios de comparación**, tienen cierta dependencia con el tipo de dato que se requiere comparar, por lo que es importante tenerlo presente. De esta manera:

- Si el tipo de dato es **numérico**, lo habitual es realizar una ordenación natural.
- En cambio, si es **alfanumérico**, se pueden ordenar por orden alfabético.
- Si se trata de un tipo de dato **booleano**, se puede ordenar en función de si cumple o no una condición.

## / 7. Caso práctico 2: “Dominio de valores, grado y cardinalidad”

**Planteamiento:** Siguiendo con el planteamiento de los últimos temas, y en base a lo visto anteriormente, vamos a analizar las actividades que hemos hecho hasta ahora para ayudar a nuestro amigo con su tienda de informática.

**Nudo:** A lo largo de los últimos temas, hemos definido varias estructuras de la base de datos que estamos generando para la tienda de nuestro amigo. Ahora que conocemos en mayor detalle los conceptos de dominio de valores, grado y cardinalidad, vamos a aplicarlos a las estructuras que hemos realizado hasta el momento. Para ello, vamos a definir el dominio de valores de cada atributo, y definiremos el grado y cardinalidad de nuestras relaciones.



**Desenlace:** En este caso, a modo de ejemplo, analizamos una de las entidades definidas para nuestro ejemplo en el caso práctico nº 1 del tema 2 de la asignatura:

COMPRAS
<b>Id producto</b>
<b>Concepto</b>
<b>Unidades</b>
<b>Precio</b>
<b>Fecha</b>
<b>Proveedor</b>

Tabla 9. Entidad compras.

El dominio de valores podría ser el siguiente:

COMPRAS
<b>Id producto - entre 1 y 5000</b>
<b>Concepto - N/A</b>
<b>Unidades - entre 1 y 5000</b>
<b>Precio - entre 1 y 999999</b>
<b>Fecha - entre 01/01/2020 y 01/01/2060</b>
<b>Proveedor - N/A</b>

Tabla 10. Dominio de valores.

En este caso, el grado tomará un valor de seis, al ser seis los atributos definidos. Así mismo, la cardinalidad vendrá definida por el número de compras (filas) que se almacenén en la base de datos.

## / 8. Resumen y resolución del caso práctico de la unidad

En este tema se ha finalizado el estudio del **modelo entidad–relación**, completando los conceptos restantes para la elaboración de diagramas de entidad–relación.

En primer lugar, hemos identificado los **elementos necesarios para la elaboración de los diagramas de entidad–relación**, y las características principales de los mismos. Posteriormente, se han estudiado las principales **metodologías para la elaboración de diagramas de entidad–relación**, que nos ayudan a elaborar diagramas completos y útiles.

Además, hemos visto las principales **características del modelo relacional**, en qué consisten los conceptos de relaciones, atributos y tuplas, así como los tipos de relaciones existentes entre los elementos de una base de datos relacional.

Por último, hemos revisado los conceptos de **grado, cardinalidad y sinónimos** dentro de un modelo relacional. Estos conceptos se han reforzado mediante la realización del caso práctico número 2.



## Resolución del caso práctico inicial

La resolución del caso práctico inicial es muy sencilla, si hemos logrado realizar de manera correcta la construcción del diagrama propuesto en el caso práctico número 1.

Un posible ejemplo de su resolución puede ser el siguiente diagrama entidad–relación:

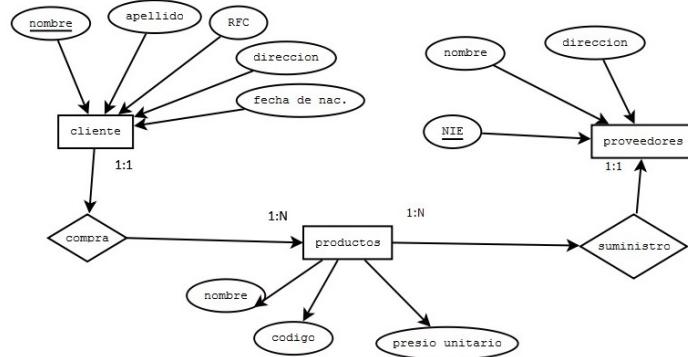


Fig.4. Ejemplo de entidades, relaciones y atributos de la tienda informática.  
<http://bdalfonso.blogspot.com/2013/06/modelo-err-ejercicios-resueltos.html>

## / 9. Bibliografía

- Oppel, A. (2009). *Databases: A Beginner's Guide*. Madrid, España: McGraw-Hill.
- Elmasri, R. y Navathe, S. (2007). *Fundamentos de Bases de Datos* (5<sup>a</sup> ed.). Madrid, España: Pearson Addison-Wesley.
- López, I.; Castellano, M.J., y Ospino, J. (2011). *Bases de datos*. Madrid, España: Garceta.
- Cabrera, G. (2011). *Sistemas gestores de bases de datos*. Madrid, España: Paraninfo.
- Pérez Marqués, M. (2016). *Administración básica de bases de datos con Oracle 12c SQL*. Madrid, España: Alfaomega.
- Connolly, T. y Begg, C. (2005). *Sistemas de Bases de Datos* (5<sup>a</sup> ed.). Madrid, España. Addison – Wesley.
- Pons, O.; Marín, N.; Medina, J.M.; Acid, S., y Vila, M.A. (2005). *Introducción a las Bases de Datos: el modelo relacional*. Madrid, España. Paraninfo.

BASE DE DATOS

# Transformación de diagramas E/R y normalización

---

# ÍNDICE

/ 1. Introducción y contextualización práctica	3
/ 2. Transformación al modelo relacional: Entidades fuertes	4
/ 3. Transformación al modelo relacional: Entidades débiles y de relaciones 1:1	5
/ 4. Transformación al modelo relacional: Relaciones 1:N y N:M	6
/ 5. Caso práctico 1: “Diferencias entre el modelo entidad-relación y el modelo relacional”	7
/ 6. Transformación de generalizaciones, especializaciones y de relaciones reflexivas	7
/ 7. Normalización	8
7.1. Primera, segunda y tercera forma normal	9
7.2. Forma normal de Boyce-Codd	10
7.3. Formas normales restantes: FN4, FN5 y Dominio clave	10
/ 8. Caso práctico 2: “Paso a modelo relacional”	10
/ 9. Resumen y resolución del caso práctico de la unidad	11
/ 10. Bibliografía	12

# OBJETIVOS

**Comprender el proceso de transformación del modelo E/R al modelo relacional.**

**Comprender y conocer el concepto de normalización aplicado al modelo de entidad–relación.**

**Manejar con soltura la realización de diagramas E/R mediante el estudio de diversos casos prácticos.**

## / 1. Introducción y contextualización práctica

En los tres temas anteriores, hemos estudiado cómo tenemos que analizar la información de un problema dado para empezar a diseñar una base de datos en la que tenemos que almacenar información. Una vez analizada, hemos visto las distintas fases implicadas en el diseño y estudiado la primera de ellas, que hace referencia al modelo conceptual que se representa utilizando los diagramas E/R.

Hemos analizado las ampliaciones necesarias de dicho diagrama para poder abarcar todas las casuísticas de un problema a resolver.

Una vez realizado el primer diseño, tenemos que pasar al diseño lógico, ya más cercano al SGBD, para lo que tuvimos que estudiar los conceptos del modelo relacional.

El diseño lógico de la base de datos se construye transformando el diagrama entidad–relación al modelo relacional aplicando una serie de reglas, que son las que vamos a estudiar en este tema y que se aplican tanto en entidades como atributos o en relaciones, para lo que tendremos que analizar las cardinalidades que hemos puesto al hacer el diseño conceptual.

No basta con obtener el modelo relacional, sino que después tendremos que ver si está normalizado, es decir, si se cumplen unas reglas de normalización y de dependencia que hacen que las bases de datos que vamos a construir sean eficientes y, sobre todo, no tengan redundancias.

Como siempre, es muy importante practicar para poder dominar las distintas transformaciones a realizar, y por eso trabajaremos con diversos casos prácticos que nos ayudarán a ello.

Escucha el siguiente audio en el que planteamos la contextualización práctica del tema. Encontrarás su resolución en el apartado final.



Fig.1. Imagen representativa de la unidad.



Audio Intro. "Transformación al modelo relacional"

<https://bit.ly/3jqtiql>





## / 2. Transformación al modelo relacional: Entidades fuertes

Para realizar el paso **del modelo entidad–relación** (modelo de Chen) **al modelo relacional** (modelo de Codd) es necesario llevar a cabo **una serie de pasos**, entre los que se encuentran la transformación de entidades, de relaciones y la normalización. Estudiaremos con mayor detalle cada una de estas transformaciones y sus pasos necesarios.

En primer lugar, vamos a conocer la **transformación al modelo relacional de entidades fuertes**.

- En este caso, para cada una de las entidades fuertes se crea una tabla asociada, con tantas columnas como atributos tenga la entidad en cuestión.
- Cada fila de la tabla creada se corresponderá con una ocurrencia de la entidad.
- Así mismo, la clave primaria de la tabla corresponderá con el atributo clave de la entidad.

El proceso se puede resumir en la siguiente tabla:

ELEMENTO EN EL MODELO E/R	ELEMENTO EQUIVALENTE EN EL MODELO RELACIONAL
Entidades	Tabla
Atributos	Columnas
Identificador principal	Clave primaria
Identificador secundario	Clave candidata

Tabla 1. Transformación de entidades al modelo relacional.

A modo de **ejemplo**, tenemos una entidad “EMPLEADO” que cuenta con una serie de atributos representativos de cada ocurrencia, como se puede apreciar en la siguiente imagen:

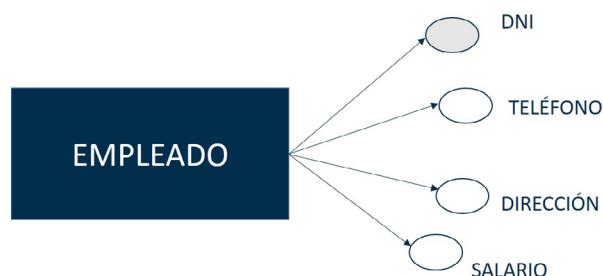


Fig.2. Entidad EMPLEADO y atributos.

Hemos definido como identificador principal el DNI, ya que identifica únicamente a cada uno de ellos. La transformación de esta entidad en el modelo relacional daría lugar a la siguiente tabla:

» **EMPLEADOS (DNI, Teléfono, Dirección, Salario)**

La clave primaria en este caso sería DNI.



## / 3. Transformación al modelo relacional: Entidades débiles y de relaciones 1:1

### A. Transformación de entidades débiles

Como hemos estudiado en temas anteriores, una entidad débil lleva implícita una relación con otra entidad fuerte, a través de un **campo de identificación**. Por tanto, para pasarlal al modelo relacional como tabla, sería necesario, únicamente, añadir ese mismo identificador de la entidad fuerte.

En el siguiente **ejemplo** de relación entre entidad fuerte y débil:

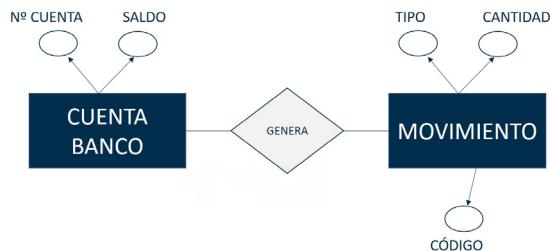


Fig.3. Ejemplo de entidad fuerte y entidad débil.

Las tablas que corresponden al modelo relacional serían las siguientes:

- » CUENTA\_BANCO (**Nº Cuenta**, Saldo).
- » MOVIMIENTO (**Nº Cuenta**, Código, Tipo, Cantidad).

### B. Transformación de relaciones 1:1

Las relaciones 1:1 ofrecen varias opciones para su paso al modelo relacional, aunque no se genera tabla para la relación. Veamos el proceso con el siguiente **ejemplo**:

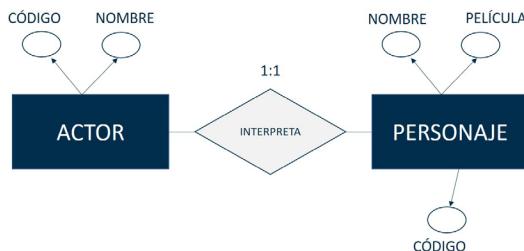


Fig.4. Ejemplo de relación 1:1

Las opciones serían las siguientes:

- » Incorporar la clave de PERSONAJES como clave externa de ACTORES (Código\_Personaje).
- » Incorporar la clave de ACTORES como clave externa de PERSONAJES (Código\_Actor).
- » Incorporar la clave de ACTORES como clave externa en la tabla PERSONAJE, y la clave de PERSONAJE, en la de ACTORES. En este caso, las tablas quedarían de la siguiente manera:
  - > ACTORES (Código, Nombre, Código\_Personaje).
  - > PERSONAJE (Código, Nombre, Película, Código\_Actor).



## / 4. Transformación al modelo relacional: Relaciones 1:N y N:M

### A. Transformación de relaciones 1:N

En este caso, a la tabla de la entidad que cuenta con la cardinalidad “N” se le añade la clave de la entidad que tiene cardinalidad “1” como clave externa. Además, si la relación tuviese atributos, también se incluirían en dicha entidad. Tampoco se genera tabla para la relación. Analicémoslo a través del siguiente **ejemplo**:

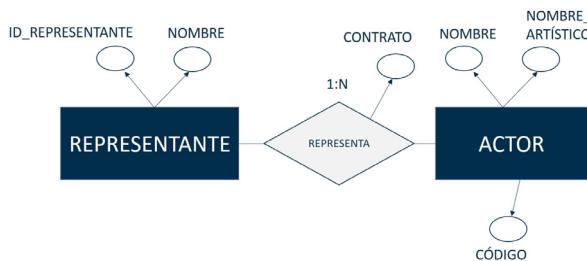


Fig.5. Ejemplo de relación 1:N.

La transformación daría resultado a dos tablas:

- » ACTORES (Código, NombreArtístico, Nombre, Id\_Representante, Contrato).
- » REPRESENTANTES (Id\_Representante, Nombre).

### B. Transformación de relaciones N:M

En las relaciones muchos a muchos se añade una tabla cuya clave primaria corresponderá a la unión de las claves primarias de las entidades asociadas. Además, si la relación tuviese atributos asociados, se incluyen también en la nueva tabla.

A continuación, se muestra un **ejemplo** de esta transformación, partiendo de la siguiente relación entre entidades:

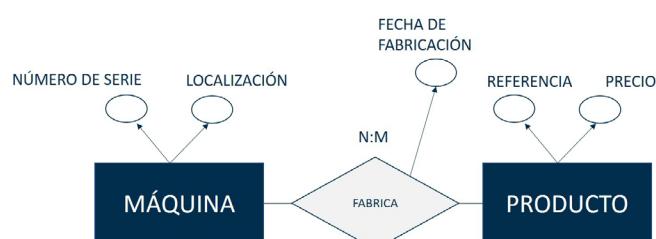


Fig.6. Ejemplo de relación N:M.

Para realizar la transformación al modelo relacional, se crearán las siguientes tablas:

- » PRODUCTOS (Referencia, Precio).
- » MÁQUINAS (Número\_serie, Localización).
- » FABRICA (Referencia, Número\_serie, Fecha\_fabricación).

Dado que las claves primarias de “PRODUCTO” y “MÁQUINA” son “Referencia” y “Número\_serie”, respectivamente.



## / 5. Caso práctico 1: “Diferencias entre el modelo entidad-relación y el modelo relacional”

**Planteamiento:** Hemos estudiado los distintos modelos utilizados para modelar datos en un sistema de base de datos. A partir de ahora lo que toca es implementar la base de datos, para lo que vamos a estudiar la siguiente fase de diseño, que es el diseño físico.

**Nudo:** A modo de resumen, y para tener una idea más clara de las características de los modelos estudiados, ¿podrías elaborar una tabla comparativa sobre las diferencias existentes entre el modelo relacional y el modelo E/R?

**Desenlace:** A continuación se muestra un ejemplo de una tabla que podría ser válida para la resolución de este caso práctico:

	MODELO ER	MODELO RELACIONAL
<b>Labor</b>	Representa la colección de objetos llamada entidades y la relación entre esas entidades.	Representa la colección de tablas y la relación entre esas tablas.
<b>Describir</b>	El Modelo de relación de entidad describe los datos como conjunto de entidades, conjunto de relaciones y atributo.	El Modelo Relacional describe los datos en una tabla como Dominio, atributos, Tuplas.
<b>Relación</b>	ER Model es más fácil de entender la relación entre las entidades.	Comparativamente, es menos fácil derivar una relación entre tablas en el Modelo Relacional.
<b>Cartografía</b>	El modelo de ER describe la asignación de cardinalidades.	El Modelo Relacional no describe las cardinalidades de mapeo.

Fig.7. Tabla comparativa entre modelo relacional y modelo E/R.

## / 6. Transformación de generalizaciones, especializaciones y de relaciones reflexivas

Las **generalizaciones y especializaciones** pueden transformarse al **modelo relacional** a través de cuatro vías **distintas**. Cada una de estas vías será más adecuada según el tipo de especialización que se esté dando en cada caso concreto (**Exclusiva, Inclusiva, Total o Parcial**).

Lo analizaremos igualmente a través de un ejemplo, partiendo del siguiente **diagrama entidad–relación**:

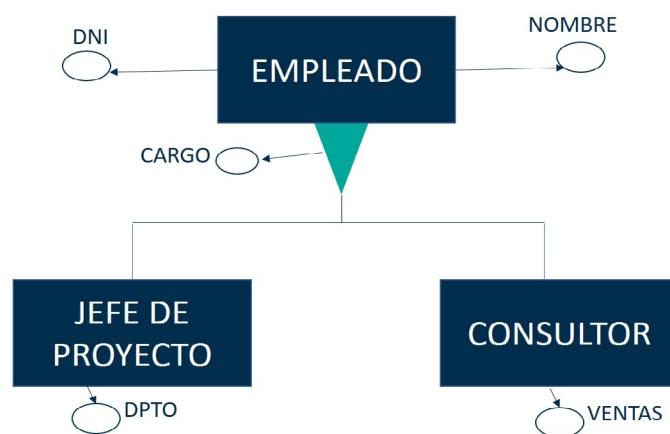


Fig.8. Ejemplo de jerarquía.



Las opciones que se pueden dar son las siguientes:

1. **Crear una tabla para la superclase y las que sean necesarias para cada subclase**, con el campo clave de la superclase. En el ejemplo anterior tendríamos:
  - a. EMPLEADOS (DNI, Nombre, Cargo).
  - b. JEFES DE PROYECTO (DNI, Dpto.).
  - c. CONSULTORES (DNI, Ventas).
2. **Crear una tabla para cada subclase incluyendo los atributos de la superclase**:
  - a. JEFES DE PROYECTO (DNI, Nombre, Cargo, Dpto.).
  - b. CONSULTORES (DNI, Nombre, Cargo, Ventas).
3. **Una única tabla para la superclase, incluyendo los atributos de las subclases y un campo “Tipo”**, que indica el tipo de subclase. Esta opción es recomendable para las Exclusivas:
  - a. EMPLEADOS (DNI, Nombre, Cargo, Dpto., Ventas, Tipo).
4. **Una única tabla para la superclase**, pero, en lugar del tipo, se añaden varios campos. Esta opción se adapta especialmente bien a las especializaciones inclusivas:
  - a. EMPLEADOS (DNI, Nombre, Cargo, Dpto., Ventas, EsJefeDeProyecto, EsConsultor).

Para las transformaciones de **relaciones reflexivas**, observa el siguiente vídeo para aprender el método de transformación que se utiliza en este caso:



Vídeo 1. "Transformación de relaciones reflexivas"  
<https://bit.ly/2LvZnQA>



## / 7. Normalización

Es habitual que el **diseño de una base de datos finalice en la transformación del modelo entidad–relación al relacional**. Sin embargo, es importante tener en cuenta que el **diseño debe cumplir una serie de parámetros de diseño o estándares**, lo que se conoce como **normalización**.

En sí mismo es un proceso que se utiliza para garantizar que las tablas del modelo relacional resultante cuenten con los atributos necesarios para describir la realidad.

**Los objetivos de la normalización** son dos, principalmente:

1. **Evitar la redundancia de datos**, para optimizar el sistema y reducir el espacio necesario de almacenamiento.
2. **Evitar dependencias erróneas entre elementos**.



El proceso de normalización consta de **varias etapas secuenciales**, que estudiaremos a continuación, llamadas **formas normales (Primera, Segunda, Tercera, Boyce–Codd, Cuarta, Quinta y Dominio–Clave)**.

Fig.9. La normalización hace que se cumplan una serie de parámetros o estándares.



No obstante, antes de ello es necesario conocer tres conceptos clave que se explican en el siguiente vídeo:



Vídeo 2. "Dependencia funcional, funcional completa y transitiva"  
<https://bit.ly/2Lwopiy>



## 7.1. Primera, segunda y tercera forma normal

### A. Primera forma normal

En la primera forma normal **no está permitida la existencia de atributos que puedan tomar más de un valor.** En el siguiente **ejemplo** se muestra un caso de tabla que no cumple con esta forma normal:

EMPLEADOS		
DNI	Nombre	Departamento
11111111X	Antonio	Contabilidad
22222222H	María	RRHH Administración

Tabla 2. Tabla que no cumple con Primera Forma Normal.

Para que la tabla pueda cumplir con la primera forma normal, debe representarse una fila más para María, en el dpto. de RRHH.

### B. Segunda forma normal

**Para que una tabla cumpla con la misma, debe cumplir, además, con la primera.** Además, **cada atributo que no sea clave debe tener dependencia completa de la clave principal.**

Por ejemplo, la tabla COMPRAS (CódProd, CódProveedor, NombreProducto, Cantidad).

El nombre del producto no tiene dependencia completa **de la clave**, por lo que no cumple FN2 (segunda forma normal).

### C. Tercera forma normal

**La tabla está en FN2 y, además, no puede haber ningún atributo que dependa de los atributos que no son clave.** En el siguiente ejemplo, el campo provincia depende del código, por lo que no está en FN3 (tercera forma normal):

EMPLEADOS		
DNI	Cód. Provincia	Provincia
11111111X	28	Madrid
22222222H	13	Ciudad Real

Fig.3. Tabla que no cumple con la Tercera Forma Normal.



Audio 1. "Ventajas del uso de la normalización"  
<https://bit.ly/3ipkO3k>





## 7.2. Forma normal de Boyce-Codd

Para su cumplimiento, debe cumplir FN3 y, además, cumplir que todo atributo (simple o compuesto) sea clave candidata. Una tabla en la que todos sus atributos forman parte de la clave cumplirá esta forma. Por ejemplo, en la siguiente tabla habría una dependencia funcional entre Responsable y Departamento, por lo que no cumpliría Boyce-Codd:

ORGANIZACIÓN		
Empleado	Departamento	Responsable
Antonio	RRHH	Ana
María	RRHH	Ana

Tabla 4. Tabla que no cumple Boyce-Codd.

Para hacer que se cumpla Boyce-Codd, se puede dividir en dos tablas:

PERSONAL	
Empleado	Responsable
Antonio	Ana
María	Ana

Tabla 5. Ejemplo representativo de Boyce-Codd.

RESPONSABLES	
Departamento	Departamento
RRHH	Ana
Organización	María

Tabla 6. Ejemplo representativo de Boyce-Codd (2).

## 7.3. Formas normales restantes: FN4, FN5 y Dominio clave

Estas formas normales tienen una aplicación en el mundo real que es puramente teórica, por lo que no se estudiarán en profundidad. Se caracterizan por lo siguiente:

- **FN4:** Se basa en el concepto de dependencias multivaluadas, y se debe cumplir previamente Boyce-Codd.
- **FN5:** Se basa en las dependencias de JOIN, y debe cumplir previamente FN4.
- **Dominio-clave:** Debe cumplirse FN5. Trata sobre las restricciones y los dominios de los atributos.

En las aplicaciones reales se recomienda normalizar como máximo hasta FN3 o Boyce-Codd.

## / 8. Caso práctico 2: “Paso a modelo relacional”

**Planteamiento:** En este caso, vamos a poner en práctica la teoría que hemos estudiado en los últimos apartados del tema, realizando el paso a modelo relacional (tablas) de una relación entre dos entidades dada por un diagrama entidad-relación.



**Nudo:** El caso en concreto se representa en la siguiente imagen, en la que se muestra la relación entre dos entidades y los atributos de las mismas, que se pretenden pasar al modelo relacional:

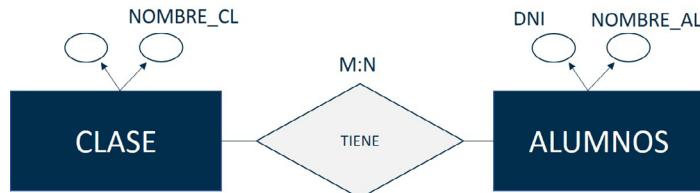


Fig.10. Caso de relación a transformar.

**Desenlace:** En el ejemplo se muestra una relación M:N entre entidades que se pretende pasar al modelo relacional. En este caso, como hemos estudiado en la teoría del tema, es necesario crear una tabla que represente la relación entre entidades. Esta tabla contendrá como claves externas las claves primarias de las tablas de las entidades. La unión de ambas claves externas formará la clave principal de la tabla resultante de la relación.

De esta forma, como paso al modelo relacional, obtendríamos tres tablas diferenciadas con los siguientes atributos:

- » ALUMNOS (DNI, Nombre\_AL).
- » CLASES (Num\_Clase, Nombre\_cl).
- » TIENE (DNI, Num\_Clase).

## / 9. Resumen y resolución del caso práctico de la unidad

Una vez que ya sabemos elaborar diagramas entidad-relación completos, hemos pasado a la siguiente fase de diseño analizando la diferente problemática existente para la **transformación de diagramas de entidad–relación al modelo relacional**, paso fundamental en el diseño y desarrollo de bases de datos.

Se ha finalizado el tema con el estudio del concepto de **normalización**, necesario para que las tablas resultantes del proceso de transformación cumplan con una serie de propiedades y normas de construcción y diseño.

### Resolución del caso práctico inicial

Las preguntas planteadas en el audio de inicio del tema, respecto de la tienda informática de nuestro amigo, se lleva a cabo aplicando las reglas de transformación estudiadas a lo largo de la unidad, tomando como referencia siempre el diagrama E/R, que recoge toda la semántica y restricciones para crear de forma eficiente la base de datos.

Si recordamos, el diagrama entidad-relación que obtuvimos en el tema anterior es el siguiente:

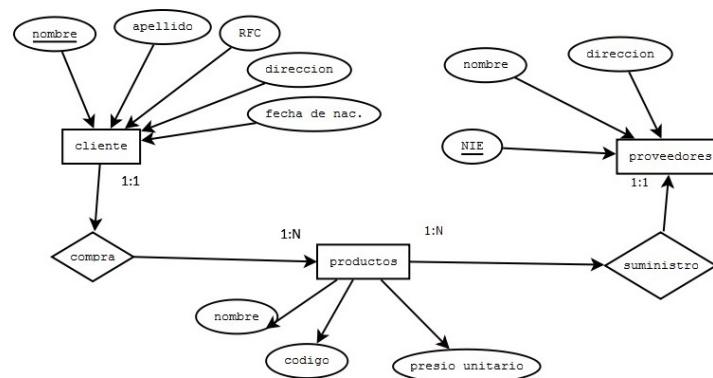


Fig.11. Ejemplo de entidades, relaciones y atributos de la tienda informática. [Fuente](#).



El modelo relacional resultante de su transformación sería:

- CLIENTE (nombre, apellido, RFC, dirección, fecha\_nacimiento).
- COMPRA (nombre, código).
- PRODUCTOS (código, nombre, precio\_unitario, NIE).
- PROVEEDORES (NIE , nombre, dirección).

A partir de este momento, ya estaremos en disposición de pasar al diseño físico, es decir, a su implementación en el SGBD de Oracle utilizando el lenguaje más extendido, que es SQL, y que estudiaremos en los próximos temas.

## / 10. Bibliografía

Oppel, A. (2009). *Databases: A Beginner's Guide*. Madrid, España: McGraw-Hill.

Elmasri, R. y Navathe, S. (2007). *Fundamentos de Bases de Datos* (5<sup>a</sup> ed.). Madrid, España: Pearson Addison-Wesley.

López, I.; Castellano, M.J., y Ospino, J. (2011). *Bases de datos*. Madrid, España: Garceta.

Cabrera, G. (2011). *Sistemas gestores de bases de datos*. Madrid, España: Paraninfo.

Pérez Marqués, M. (2016). *Administración básica de bases de datos con Oracle 12c SQL*. Madrid, España: Alfaomega.