

# Diseño lógico relacional

## Contenidos

- ☛ Representación del problema
- ☛ Modelo de datos
- ☛ Diagramas E/R
- ☛ El modelo E/R ampliado
- ☛ El modelo relacional
- ☛ Transformación E/R al modelo relacional
- ☛ Normalización

## Objetivos

- ☛ Identificar el significado de la simbología de los diagramas E/R
- ☛ Identificar las tablas del diseño lógico
- ☛ Identificar los campos que forman parte de las tablas
- ☛ Identificar las relaciones entre tablas del diseño lógico
- ☛ Identificar los campos clave
- ☛ Aplicar reglas de integridad
- ☛ Identificar reglas de normalización
- ☛ Identificar y documentar reglas que no se pueden plasmar en el diseño lógico

*En este tema se trata a fondo el diseño de una base de datos, desde la interpretación y análisis de un problema hasta el diseño y propuesta de un modelo que dé solución al problema planteado.*

## 2.1. Representación del problema

Una base de datos representa la información contenida en algún dominio del mundo real. El diseño de base de datos consiste en extraer todos los datos relevantes de un problema, por ejemplo, saber qué datos están implicados en el proceso de facturación de una empresa que vende vehículos agrícolas, o, qué datos son necesarios para llevar el control veterinario de los animales de un zoológico.

Para extraer estos datos, se debe realizar un análisis en profundidad del dominio del problema, y saber, de esta forma, qué datos son esenciales para la base de datos y descartar los que no son de utilidad. Una vez extraídos los datos esenciales comienza el proceso de *modelización*, esto es, construir, mediante una herramienta de diseño de base de datos, un esquema que exprese con total exactitud todos los datos que el problema requiere almacenar.

Típicamente, los informáticos analizan un problema a través de diversas reuniones con los futuros usuarios del sistema. Nótese, que generalmente, el problema no solo se resuelve poniendo una base de datos a disposición de un usuario, sino también un conjunto de aplicaciones de software que automaticen el acceso a los datos y su gestión. De estas reuniones, se extrae el documento más importante del análisis de un sistema informático, el documento de *Especificación de Requisitos Software* o *E.R.S.* A partir de esta E.R.S. se extrae toda la información necesaria para la modelización de los datos.

- 
- ◊ **Actividad 2.1:** Busca en Internet la estructura del documento estándar IEEE 830 SRS o “Software Requirements Specification”. Descarga de internet algún ejemplo de SRS y examina cómo los analistas de software organizan los requisitos de una aplicación extraídos de las conversaciones con usuarios.
- 

## 2.2. El modelo de datos

La modelización consiste en representar el problema realizando múltiples abstracciones<sup>1</sup> para asimilar toda la información de un problema, y de esta manera, generar un mapa donde estén identificados todos los objetos de la base de datos.

---

<sup>1</sup>Una de las acepciones de la RAE para **abstraer** es: “Separar por medio de una operación intelectual las cualidades de un objeto para considerarlas aisladamente o para considerar el mismo objeto en su pura esencia o noción”

Para modelar un problema de base de datos es necesario tener en cuenta las siguientes consideraciones:

- Casi con toda probabilidad, la persona que realiza la modelización es un analista informático, por lo que puede no ser un experto en el dominio del problema que debe resolver (Contabilidad, Medicina, Economía, etc.). Se ha de contar con la experiencia de un futuro usuario de la base de datos que conozca a fondo todos los pormenores del negocio, y que, a su vez, puede no tener conocimientos de informática.
- Hay que modelar siguiendo unas directrices o estándares, es decir, usando una filosofía estándar para que el resto de la comunidad informática pueda entender y comprender el modelo realizado. De esta manera, será posible aprovechar las herramientas informáticas software del mercado para realizar diseños.
- La base de datos estará gestionada por un SGBD que tendrá unas características técnicas, de esta manera, no se tratará igual la implantación de la base de datos en un sistema MySQL que en uno DB2.

Para satisfacer estas necesidades, se suele recurrir a tres modelados:

1. El modelo conceptual. Es un modelo que tiene un gran poder expresivo para poder comunicarse con un usuario que no es experto en informática. Tiene una gran potencia para representar el dominio del problema tal y como el usuario lo concibe. El modelo que se usará en este libro será el modelo Entidad/Relación.
2. El modelo lógico. Este modelo es más técnico que el anterior. Los conceptos expresados por este modelo, suelen ser difíciles de entender por los usuarios y generalmente tienen traducción directa al modelo físico que entiende el SGBD. El modelo lógico elegido dependerá de la implementación de la base de datos, así, no es lo mismo modelizar una base de datos orientada a objetos, que modelizar una base de datos relacional. El modelo que se usará en este libro será el Modelo Relacional.
3. El modelo físico. Es el resultado de aplicar el modelo lógico a un SGBD concreto. Generalmente está expresado en un lenguaje de programación de BBDD tipo SQL. Este libro transformará el Modelo Relacional en modelo físico a través del sublenguaje DDL de SQL.

La interacción entre estos tres modelos es fundamental para un diseño de calidad:

1. Primero, se negocia con el usuario el modelo conceptual.

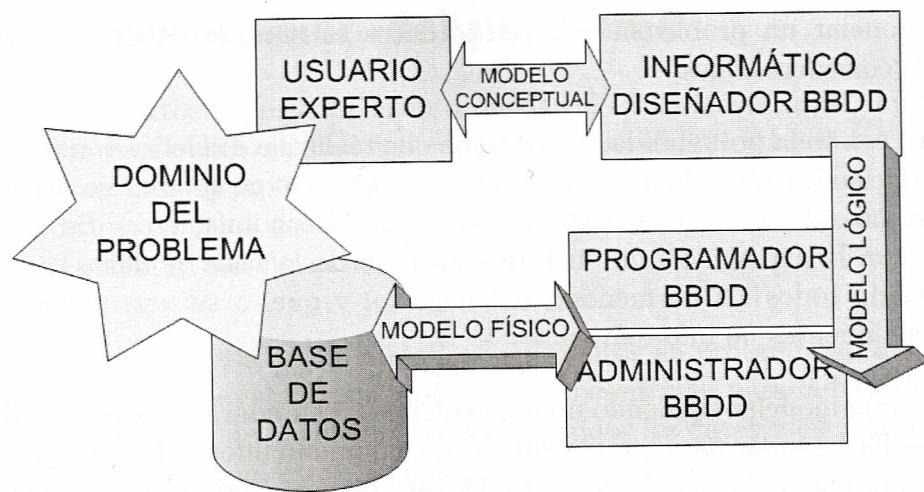


Figura 2.1: Interacción entre modelos.

2. Segundo, se pasa el modelo conceptual al modelo lógico, realizando una serie de transformaciones necesarias para adaptar el lenguaje del usuario al del gestor de base de datos.
3. Finalmente, se transforma el modelo lógico en físico, obteniendo de esta forma la base de datos final.

#### El consejo del buen administrador...

*En ocasiones, los diseñadores experimentados, realizan el diseño de la base de datos directamente en el modelo relacional. Esto puede representar un ahorro de tiempo si el problema a resolver es relativamente sencillo. Pero, generalmente, en problemas más complejos, saltarse el diseño conceptual y la opinión del usuario, da como resultado diseños incompletos e incoherentes.*

### 2.3. Diagramas E/R

Para representar el modelo conceptual se usará el modelo Entidad/Relación. Este modelo consiste en plasmar el resultado del análisis del problema mediante diagramas entidad-relación.<sup>2</sup>

<sup>2</sup>También llamados en otras fuentes diagramas Entidad-Interrelación, o en inglés Entity-relationship

Estos diagramas fueron propuestos por Peter P. Chen a mediados de los años 70 para la representación conceptual de los datos y establecer qué relaciones existían entre ellos.

La notación es muy sencilla, y, precisamente, esta sencillez, permite representar el mundo real de forma que el usuario pueda validar si el modelo propuesto se ajusta perfectamente a la resolución del problema.

A continuación, se presentan las definiciones necesarias para comprender el modelo entidad relación.

### 2.3.1. Entidad

Cualquier tipo de objeto o concepto sobre el que se recoge información: cosa, persona, concepto abstracto o suceso. Se representan mediante un cuadrado. Por ejemplo: coche, casa, empleado, cliente, etc. Las entidades se representan gráficamente mediante rectángulos y su nombre aparece en el interior (generalmente en singular). Un nombre de entidad solo puede aparecer una vez en el diagrama.

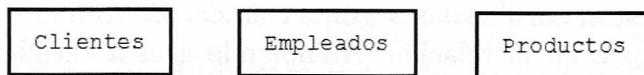


Figura 2.2: Ejemplos de entidades.

Hay dos tipos de entidades: fuertes (o regulares) y débiles. Las entidades débiles se representan mediante un cuadro doble. Una entidad débil es una entidad cuya existencia depende de la existencia de otra entidad. Una entidad fuerte es una entidad que no es débil, es decir, existe por méritos propios. Un ejemplo típico es la existencia de dos entidades para la representación de un pedido. Por un lado, la entidad pedido representa información genérica sobre el pedido como la fecha del pedido, fecha de envío, el estado, etc. Por otro lado, la entidad "Detalle de Pedido" recopila las líneas de información específica sobre los artículos y unidades pedidas. En este caso, "Detalle de Pedido" es una entidad débil, puesto que la eliminación del pedido implica la eliminación de las líneas de detalle asociadas al pedido, es decir, no tiene sentido almacenar información específica del pedido si se ha eliminado ese pedido.

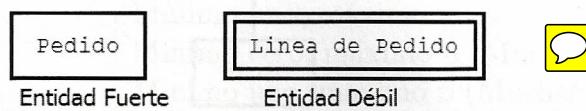


Figura 2.3: Entidad fuerte y débil.

### 2.3.2. Ocurrencia de una entidad

Es una instancia de una determinada entidad, esto es, una unidad del conjunto que representa la entidad. Ejemplo: La entidad “coche” tiene varias instancias, una de ellas es el vehículo “seat ibiza con matrícula 1222FHD de color negro y con 5 puertas”.

### 2.3.3. Relación

Una relación (o interrelación), es una correspondencia o asociación entre dos o más entidades.

Cada relación tiene un nombre que describe su función. Normalmente debe utilizarse un nombre que exprese con totalidad la finalidad de la relación, evitando poner un nombre que pueda significar muchas cosas, por ejemplo, tener, hacer, poseer.

Las relaciones se representan gráficamente mediante rombos y su nombre aparece en el interior. Generalmente este nombre de relación corresponde a un verbo, pues las relaciones suelen describir las acciones entre dos o más entidades.

Las relaciones están clasificadas según su *grado*. El grado es el número de entidades que participan en la relación. Atendiendo a esta clasificación, existen los siguientes tipos de relaciones:

- Relaciones binarias: (grado 2), son aquellas que se dan entre dos entidades.



Figura 2.4: Ejemplo de relación binaria.

- Relaciones ternarias: (grado 3), son aquellas que se dan entre tres entidades.

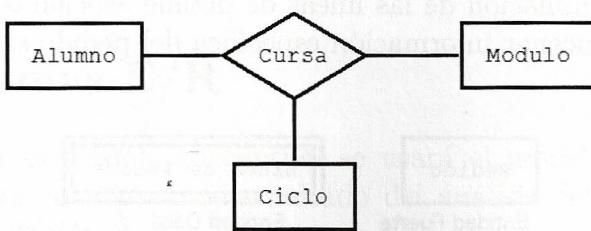


Figura 2.5: Ejemplo de relación ternaria.

- Relaciones unarias o reflexivas: (grado 1), Es una relación donde la misma entidad participa más de una vez en la relación con distintos papeles. El nombre de estos papeles es importante para determinar la función de cada participación.

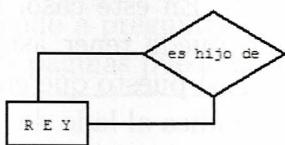


Figura 2.6: Ejemplo de relación reflexiva.

- Relaciones n-arias: (grado  $>3$ ) Son aquellas donde participan más de 3 entidades. Aparecen en muy raras ocasiones, puesto que generalmente se pueden descomponer en varias de grado 2 o de grado 3.

#### El consejo del buen administrador...

*Si en tu diagrama entidad relación aparecen relaciones de grado  $>3$ , es posible que la interpretación del problema sea incorrecta. Incluso si aparecen relaciones de grado 3, intenta descomponerlas en varias de grado 2 para simplificar tu modelo. Eso sí, asegúrate de que no es posible descomponerlas en varias de grado 2 sin perder semántica.*

#### 2.3.4. Participación

La participación de una ocurrencia de una entidad, indica, mediante una pareja de números, el mínimo y máximo número de veces que puede aparecer en la relación asociada a otra ocurrencia de entidad. Las posibles participaciones son:

Participación	Significado
(0,1)	Mínimo cero, máximo uno
(1,1)	Mínimo uno, máximo uno
(0,n)	Mínimo cero, máximo n (Muchos)
(1,n)	Mínimo uno, máximo n (Muchos)

Las reglas que definen la participación de una ocurrencia en una relación son las *reglas de negocio*, es decir, se reconocen a través de los requisitos del problema.

La notación que se utiliza para expresar las participaciones en el diagrama entidad relación es poner al lado de la entidad correspondiente, la pareja de números máximo y mínimo de participaciones. Por ejemplo, los empleados pueden trabajar para varios proyectos, o pueden estar de vacaciones (sin proyecto). Por otro lado, en un proyecto trabajan de 1 a varios trabajadores. En este caso, la participación de *proyecto* es de (0,n), puesto que un empleado puede tener asignados de 0 a n proyectos. La participación del *empleado* es de (1,n) puesto que en un proyecto puede haber de 1 a n empleados. De esta manera, se indica al lado de la entidad proyecto, el par (0,n) y al lado de la entidad empleado el par (1,n).

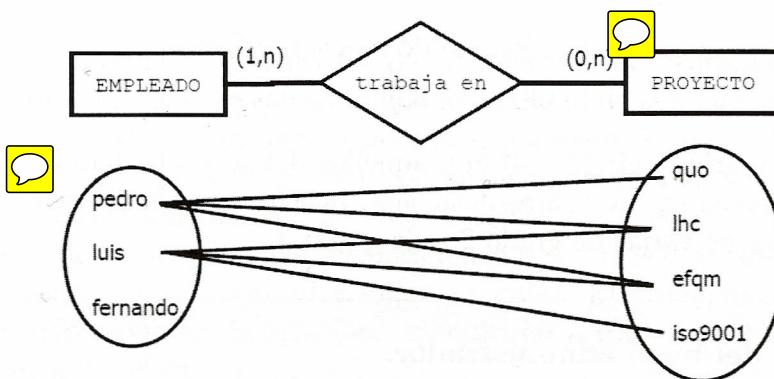


Figura 2.7: Participaciones de ocurrencias en una relación.

◇ **Actividad 2.2:** En un supermercado hay productos organizados en categorías (frutas, ultramarinos, carnes, pescados, etc.). Cada producto pertenece a una única categoría, y puede haber categorías que todavía no tengan ningún producto asignado, sin embargo, no puede haber productos sin categoría. Calcula las participaciones de cada entidad en la relación *Producto Pertenecer a Categoría*

Un producto puede y debe pertenecer a una única categoría (mínimo 1 y máximo 1), por tanto la participación de categoría en la relación con el producto es de (1,1). A una categoría pueden pertenecer muchos productos (máximo n), y puede no tener productos (mínimo 0), por tanto, los productos participan en las categorías con cardinalidad (0,n).



◊ **Actividad 2.3:** Las páginas web contienen controles de muchos tipos (campos de texto, listas desplegables, etc.). Si se quiere almacenar en una base de datos, cada página web, qué tipos de controles tiene, ¿qué participaciones habría que asignar? Justifica tu respuesta respondiendo a preguntas del tipo ¿un control, (por ejemplo, un cuadro de texto), en cuántas páginas puede estar como máximo y mínimo?

◊ **Actividad 2.4:** Los clientes pueden realizar pedidos a través de sus representantes de ventas. Indica las entidades que hay, relaciones y sus respectivas participaciones.

### 2.3.5. Cardinalidad

La cardinalidad de una relación se calcula a través de las participaciones de sus ocurrencias en ella. Se toman el número máximo de participaciones de cada una de las entidades en la relación. Por ejemplo, la relación *organiza* de la actividad 2.2, tendría una cardinalidad de 1:N, puesto que por el lado de las categorías, el máximo de (1,1) es 1, y por el lado de los productos, el máximo de (0,n) es N.

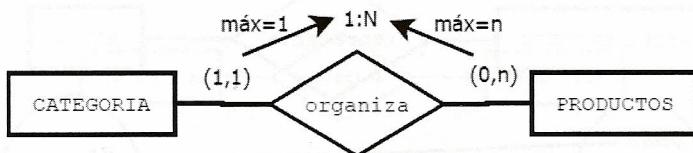


Figura 2.8: Cálculo de la cardinalidad de una relación.

De esta manera, se clasifican las siguientes cardinalidades:

- **Cardinalidad 1:1** Esta cardinalidad especifica que una entidad A puede estar vinculada mediante una relación a una y solo una ocurrencia de otra entidad B. A su vez una ocurrencia de la entidad B solo puede estar vinculada a una ocurrencia de la entidad A. Por ejemplo, se puede limitar el número de directores de departamento mediante una relación 1:1. Así, un empleado solo puede ser jefe de un departamento, y un departamento solo puede tener un jefe.

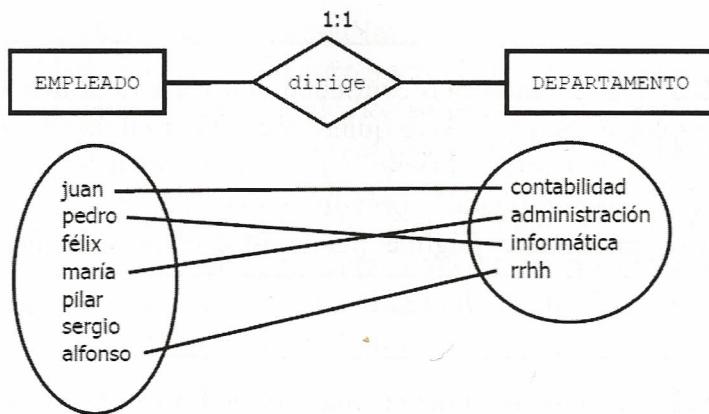


Figura 2.9: Ejemplo de cardinalidad 1-1.

- **Cardinalidad 1:N (o 1:Muchos)** Esta relación especifica que una entidad A puede estar vinculada mediante una relación a varias ocurrencias de otra entidad B. Sin embargo, una de las ocurrencias de la entidad B solo puede estar vinculada a una ocurrencia de la entidad A. Por ejemplo, un representante gestiona las carreras de varios actores, y un actor solo puede tener un representante .

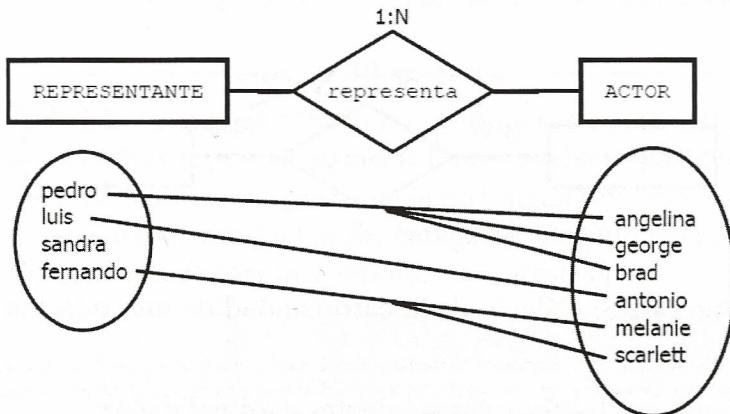


Figura 2.10: Ejemplo de cardinalidad 1-N.

- **Cardinalidad M:N (o Muchos:Muchos)** O también N:M, esta cardinalidad especifica que una entidad A puede estar vinculada mediante una relación a varias ocurrencias de la entidad B, y a su vez, una ocurrencia de la entidad B puede estar vinculada a varias de la entidad A. Por ejemplo, un empleado

puede trabajar para varios proyectos; al mismo tiempo, en un mismo proyecto, pueden trabajar varios empleados.

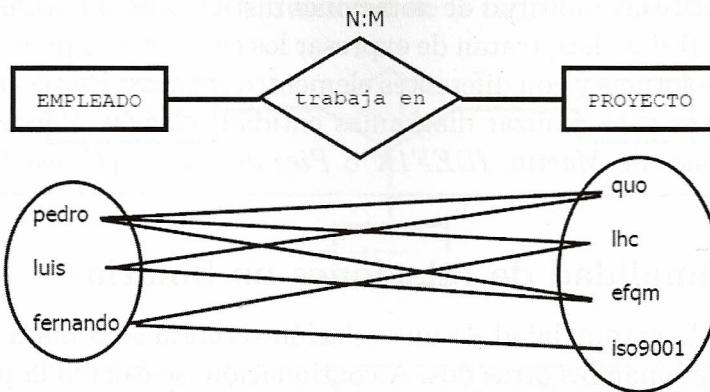


Figura 2.11: Ejemplo de cardinalidad N-M.

Como puede observarse en las figuras anteriores, la notación para representar el tipo de relación según su cardinalidad, consiste en escribir el tipo de cardinalidad justo encima del rombo. Existen numerosas alternativas a esta nomenclatura, siendo muy típicas las dos siguientes:

- Puntas de flecha: En esta notación, la línea de la relación que termina en flecha, indica la rama N de la cardinalidad de la relación.



Figura 2.12: Ejemplo de notación “puntas de flecha“.

- Notación *classic* de MySQL Workbench: En esta notación, las relaciones se expresan con un pequeño rombo, rellenando en negro la mitad de la figura, en el lado de la entidad cuya cardinalidad es N.

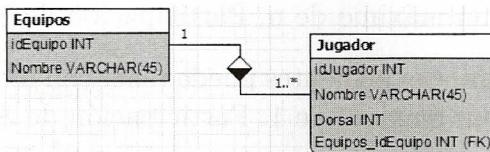


Figura 2.13: Ejemplo de notación *classic* de “MySQL Workbench“.

◊ **Actividad 2.5:** Hay multitud de notaciones distintas para realizar los diagramas entidad relación. Todas ellas, tratan de expresar los conceptos expuestos en este libro, pero de diferentes formas y con diferentes elementos gráficos. Busca en Internet otros tipos de notaciones para realizar diagramas entidad relación. Puedes buscar, entre otros, las notaciones de *Martin*, *IDEF1X* o *Pies de cuervo (Crows foot)*.

### 2.3.6. Cardinalidad de relaciones no binarias

Para calcular la cardinalidad de una relación ternaria se tomará una de las tres entidades y se combinan las otras dos. A continuación, se calcula la participación de la entidad en la combinación de las otras dos. Posteriormente, se hará lo mismo con las otras dos entidades. Finalmente, tomando los máximos de las participaciones se generan las cardinalidades.

$$\begin{aligned} \text{Cardinalidad} &= \max(0,1) : \max(1,1) : \max(0,n) \\ &= 1:1:N \end{aligned}$$

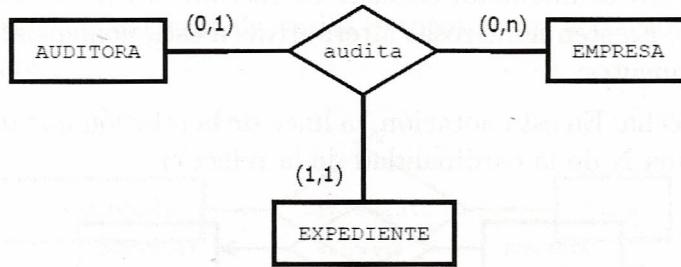
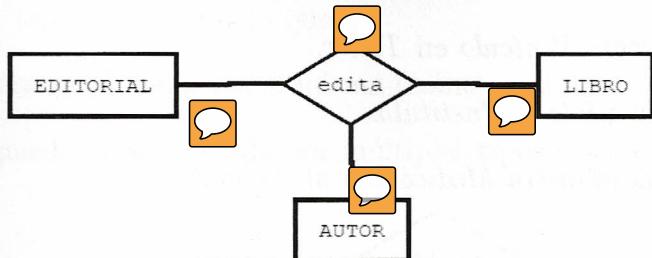


Figura 2.14: Cardinalidad de una relación ternaria.

Por ejemplo, en la figura 2.14 se distinguen tres participaciones, la que se produce entre empresa y auditora-expediente, la que se distingue entre auditora y empresa-expediente, y por último la de expediente con auditora-empresa:

- Una empresa ¿Cuántos expedientes puede tener con una auditora? Puede tener un mínimo de 0 y un máximo de n. Participación de Empresa (0,n).
- Una auditora ¿Cuántos expedientes puede tener con una empresa? Puede tener un mínimo de 0 y un máximo de 1. Participación de Auditora (0,1).
- Un expediente ¿A cuántas empresas auditadas por la auditora puede pertenecer? Un expediente solo puede pertenecer a una empresa auditada (1,1), por tanto Participación de Expediente (1,1).

- **Actividad 2.6:** Calcula la cardinalidad de la siguiente relación ternaria:



Hay que contestar a las siguientes preguntas: ¿Cuántos autores puede tener un determinado libro publicado en una determinada editorial?

Mínimo 1, máximo n, participación de Autor (1,n).

¿Cuántos libros puede tener un determinado autor publicados en una determinada editorial?

Mínimo 0, máximo n, participación de Libro (0,n).

¿En cuántas editoriales puede un determinado autor publicar un mismo libro?

Mínimo 1, máximo 1. Participación de Editorial (1,1).

Tomando los máximos de cada participación se obtiene que la cardinalidad de la relación es de 1:N:N

- **Actividad 2.7:** Calcula la cardinalidad de las siguientes relaciones binarias:

- *Hombre* está casado con *Mujer*, en una sociedad monogámica.
- *Hombre* está casado con *Mujer*, en una sociedad machista poligámica.
- *Hombre* está casado con *Mujer*, en una sociedad poligámica liberal.
- *Pescador* pesca *Pez*.
- *Arquitecto* diseña *Casa*.
- *Piezas* forman *Producto*.
- *Turista* viaja *Hotel*.
- *Jugador* juega en *Equipo*.
- *Político* gobierna en *País*.

◊ **Actividad 2.8:** Calcula la cardinalidad de las siguientes relaciones ternarias:

- *Mecánico* arregla *Vehículo* en *Taller*.
  - *Alumno* cursa *Ciclo* en *Instituto*.
  - *Veterinario* administra *Medicación* al *Animal*.
- 

### 2.3.7. Cardinalidad de las relaciones reflexivas

En las relaciones reflexivas, la misma entidad juega dos papeles distintos en la relación. Para calcular su cardinalidad hay que extraer las participaciones según los dos roles existentes. Por ejemplo, en la relación reflexiva “Es jefe”, la entidad Empleado aparece con dos Roles. El primer rol es el empleado como jefe, y el segundo rol el empleado como subordinado. Así, se puede calcular las participaciones preguntando:

- ¿Cuántos subordinados puede tener un jefe? Un jefe puede tener un mínimo de 1 y un máximo de n: (1,n)
- ¿Cuántos jefes puede tener un subordinado? Un mínimo de 0 (un empleado sin jefes sería el responsable de la empresa) y un máximo de 1 (suponiendo una estructura, típicamente piramidal): (0,1).

Por tanto, la relación sería de cardinalidad 1:N

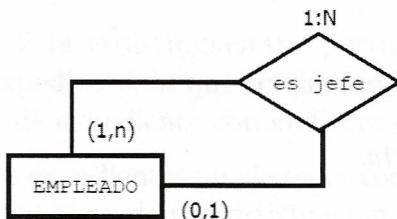
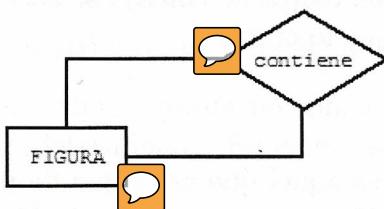


Figura 2.15: Cardinalidad de una relación reflexiva.

- **Actividad 2.9:** Justifica cuál serían las participaciones y la cardinalidad de la siguiente relación, teniendo en cuenta que:

- Una figura puede contenerse a sí misma (como en el caso de los fractales).
- Una figura puede estar formada por múltiples tipos distintos de figuras.



### 2.3.8. Atributos y Dominios

Los atributos de una entidad son las características o propiedades que la definen como entidad. Se representan mediante elipses conectadas directamente a la entidad.

Por ejemplo, para representar la entidad HOTEL, son necesarias sus características, esto es, el número de plazas disponibles, su dirección, la ciudad donde se encuentra, etc.

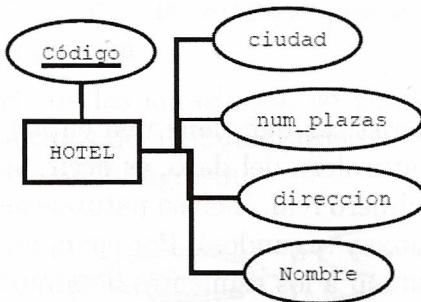


Figura 2.16: Ejemplo de atributos.

#### Atributo Clave

En la figura anterior, aparece el atributo *código*, subrayado. Este atributo se denomina clave, y designa un campo que no puede repetir ninguna ocurrencia de entidad. Se dice, que este campo identifica únicamente a una entidad, es decir,

que con la sola referencia a un campo clave se tiene acceso al resto de atributos de forma directa. Ejemplo: El DNI es el campo clave de una persona, pues ninguna persona tiene el mismo DNI. Por tanto, si se especifica el DNI de esa persona se sabe exactamente a qué ocurrencia de persona se refiere. Todas las entidades fuertes deberían tener, al menos, un atributo clave. Nótese que una entidad puede formar la clave mediante varios atributos, en este caso, se dice que la clave de la entidad es la suma de esos atributos y que la entidad tiene una clave *compuesta*. Si la clave está formada por un único atributo se dice que es *atómica*. Por ejemplo, para identificar de forma única una oferta de trabajo se necesitaría el nombre del puesto y el nombre de la empresa que lo oferta.

### Atributo de relación

Un atributo de relación es aquel que es propio de una relación y que no puede ser cedido a las entidades que intervienen en la relación. Por ejemplo, un mecánico repara un vehículo, la reparación se realiza en una determinada fecha.

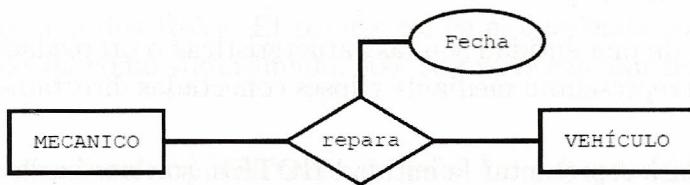


Figura 2.17: Ejemplo de atributo de relación.

### Dominios

Cada una de las características que tiene una entidad pertenece a un dominio. El dominio representa la naturaleza del dato, es decir, si es un número entero, una cadena de caracteres o un número real. Incluso naturalezas más complejas, como una fecha o una hora (con minutos y segundos). Por ejemplo, los siguientes atributos de la entidad empleado pertenecen a los siguientes dominios:

Atributo	Dominio
DNI	Cadena de Caracteres de longitud 10
Nombre	Cadena de Caracteres de longitud 50
Fecha_Nacimiento	Fecha
Dirección	Cadena de Caracteres de longitud 100
Sueldo	Números reales
Número de hijos	Números enteros
Departamento	Departamentos

Si un dominio se especifica mediante el tipo de datos, como en el caso de DNI, Nombre o Fecha\_Nacimiento se dice que se define por *intensión*. Si se especifica mediante un conjunto de valores, como en el dominio Departamentos, que puede tener los valores (RRHH, Informática, Administración o Contabilidad), la definición del dominio es por *extensión*.

### 2.3.9. Tipos de atributos

Se pueden clasificar los atributos según las siguientes restricciones:

**Atributos obligatorios:** Un atributo debe tomar un valor obligatoriamente.

**Atributos opcionales:** Un atributo puede no tomar un valor porque sea desconocido en un momento determinado. En este caso, el atributo tiene un valor *nulo*.

**Atributos compuestos:** Un atributo compuesto es aquel que se puede descomponer en atributos más sencillos, por ejemplo, el atributo hora\_de\_salida se puede descomponer en dos (hora y minutos).

**Atributos univaluados:** Un atributo que toma un único valor.

**Atributos multivaluados:** Estos atributos pueden tomar varios valores, por ejemplo el atributo teléfono puede tomar los valores de un teléfono móvil y un teléfono fijo.

**Atributo derivado:** Son aquellos cuyo valor se puede calcular a través de otros atributos. Por ejemplo, el atributo Edad, se puede calcular a partir de la fecha de nacimiento de una persona.

Al igual que con la mayoría de las notaciones, no existe unanimidad a la hora de dibujar en un diagrama los tipos de atributos. Una de las más extendidas entre los diseñadores de bases de datos es la siguiente:

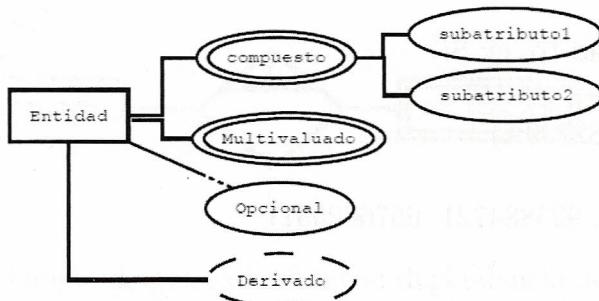


Figura 2.18: Notación para los distintos tipos de atributos.

### 2.3.10. Otras notaciones para los atributos

Al igual que para las entidades, los atributos tienen multitud de notaciones, y, aunque la original adoptada por Peter Chen es la más usada hasta ahora, por simplificar la construcción de mapas a través de herramientas software, se opta por usar otras notaciones que producen mapas más manejables. Por ejemplo, la herramienta MySQL Workbench utiliza una sintaxis muy similar a la que usa la notación *UML* para representar las características de un objeto:

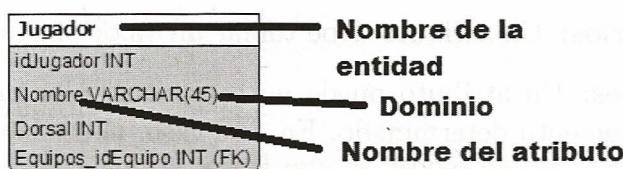


Figura 2.19: Otras formas de representar atributos

◊ **Actividad 2.10:** Justifica qué tipo de atributos son los siguientes atributos de la entidad Persona:

- Fecha de Nacimiento (p.ej. 24/11/1976)
- Lugar de Nacimiento (p.ej. Zaragoza)
- Edad (p.ej. 36 años)
- EsMayorDeEdad (p. ej: Sí)
- DNI (p.ej. 55582739A)
- Teléfonos (p.ej. 925884721, 657662531)
- Apellidos

### 2.3.11. Las entidades débiles

Como se ha expuesto anteriormente, las entidades débiles dependen de una entidad fuerte mediante una relación. La relación que une ambas entidades también es débil, puesto que también desaparece si desaparece la entidad fuerte. En estos casos, la relación tiene una dependencia que puede ser de dos tipos:

- **Dependencia de existencia:** Este tipo de dependencia expresa que, las ocurrencias de una entidad débil, no tienen ningún sentido en la base de datos sin la presencia de las ocurrencias de la entidad fuerte con la que están relacionadas. Por ejemplo, las transacciones que se dan en una cuenta bancaria, no tienen sentido si no existe la cuenta bancaria a la que están asociadas.

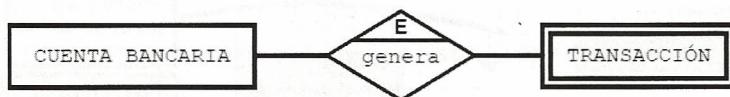


Figura 2.20: Ejemplo de entidad débil con dependencia de existencia.

- **Dependencia de identificación:** Este tipo se produce cuando, además de la dependencia de existencia, la entidad débil necesita a la fuerte para poder crear una clave, de tal manera que pueda completar la identificación de sus ocurrencias. Por ejemplo, una empresa fabricante de software crea aplicaciones:

1. La compañía se identifica por su nombre (por ejemplo, Microsoft).
2. Las aplicaciones se identifican por su nombre comercial, por ejemplo (Office).
3. Cada compañía de software pone un nombre a cada una de sus aplicaciones.

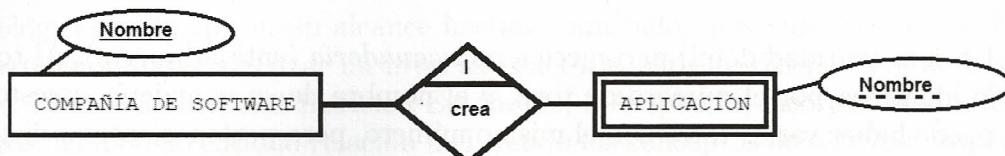


Figura 2.21: Ejemplo de entidad débil con dependencia de identificación.

De esta forma puede ocurrir que haya dos aplicaciones con el mismo nombre y que pertenezcan a dos compañías distintas (Office de Microsoft y Office de Sun). En

este caso para identificar a cada aplicación de forma única, hace falta el nombre de la aplicación y además, el nombre de la compañía. Así, Aplicación depende en identificación de la Compañía y el nombre de la aplicación es una clave débil. Se expresa de la siguiente forma:

Una vez más, para representar las dependencias, cada herramienta usa su propia notación. Por ejemplo, en el caso de MySQL workbench, no diferencia entre entidades fuertes o débiles (las llama a todas *tablas*), y crea las relaciones con líneas discontinuas en caso de no tener dependencia de identificación (non identifying relationship), y con línea continua en caso de tener dependencia de identificación (identifying relationship).

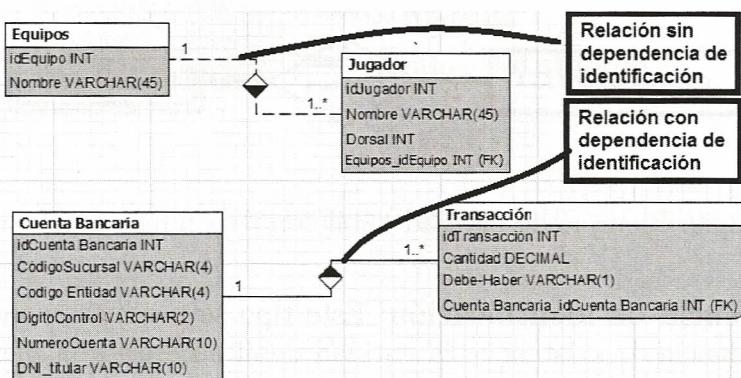


Figura 2.22: Ejemplo de notación. Dependencias en MySQL Workbench.

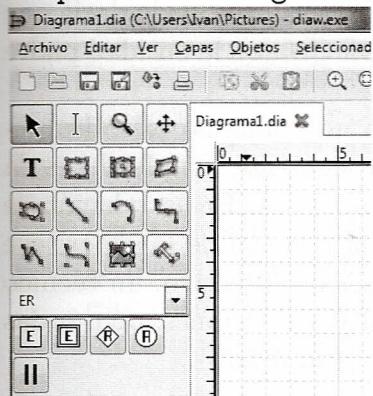
◊ **Actividad 2.11:** ¿Qué tipo de relación de dependencia tienen las siguientes entidades?

- Un *toro* (entidad débil) pertenece a una *ganadería* (entidad fuerte). Al toro se le identifica por el número de toro, y el nombre de su ganadería, puesto que puede haber varios toros con el mismo número, pero pertenecientes a distintas ganaderías.
- En el acceso al parking de una empresa un *empleado* (entidad fuerte) tiene un *vehículo* (entidad débil).

**El consejo del buen administrador...**

Muchos diseñadores se abstraen del hecho de tener entidades fuertes y débiles y, tal y como hace MySQL Workbench, no distinguen entre entidades fuertes y débiles. En general, es buena idea simplificar el diseño (filosofía KISS - “Keep it simple, sir”), pero se ha de ser consciente de la pérdida de semántica que implica.

- ◊ **Actividad 2.12:** Descarga el software DIA para la creación de diagramas de diversos tipos. Puedes encontrarlo de forma gratuita en la página web <http://sourceforge.net/projects/dia-installer/>.



Crea un diagrama nuevo y con los símbolos de los diagramas entidad relación de la notación Chen (ER) modela la relación Cliente-adquiere-Producto con los atributos que se te ocurran. Identifica el ícono de cada figura con los símbolos Entidad, Entidad Débil, Atributo (con sus múltiples tipos) y Relación.

## 2.4. El modelo E/R ampliado

La primera concepción del modelo entidad relación tuvo, por las limitaciones tecnológicas de la época, un alcance bastante limitado, que, con los años, se ha ido desarrollando hasta alcanzar un nivel satisfactorio para los diseñadores de bases de datos. El modelo Entidad-Relación Extendido, o Ampliado, incorpora todos los elementos del modelo entidad relación incluyendo los conceptos de subclase, superclase junto a los conceptos de especialización y generalización.

### 2.4.1. Generalización y Especialización

Una entidad E es una generalización de un grupo de entidades  $E_1, E_2, \dots, E_n$ , si cada ocurrencia de cada una de esas entidades es también una ocurrencia de E.

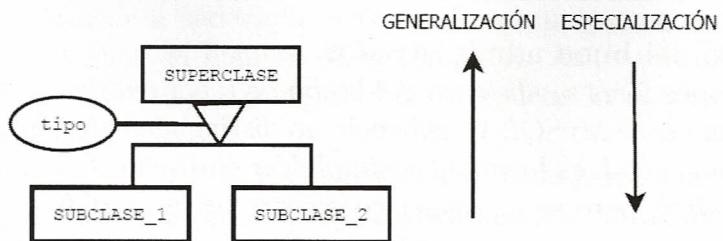


Figura 2.23: Generalización y Especialización.

Todas las propiedades de la entidad genérica E son heredadas por las subentidades. Además, cada subentidad tendrá sus propios atributos independientes de la generalización.

Las subentidades son especializaciones de la entidad general, se puede decir que las subentidades o *subclases* tienen una relación del tipo *ES UN* con la entidad padre o *superclase*.

La relación de generalización se representa mediante un triángulo isósceles pegado por la base a la entidad superclase. En la figura siguiente *Empleado* es la superclase y los directivos, comerciales y técnicos son subclases. En la relación se adjunta un atributo que indica cómo debe interpretarse la relación de la superclase con la subclase. La generalización *Empleado* que puede ser un directivo, un técnico o un comercial. Cada subentidad tiene sus propios atributos y relaciones, pero todas heredan los atributos nombre y DNI de la entidad *padre* (*Empleado*).

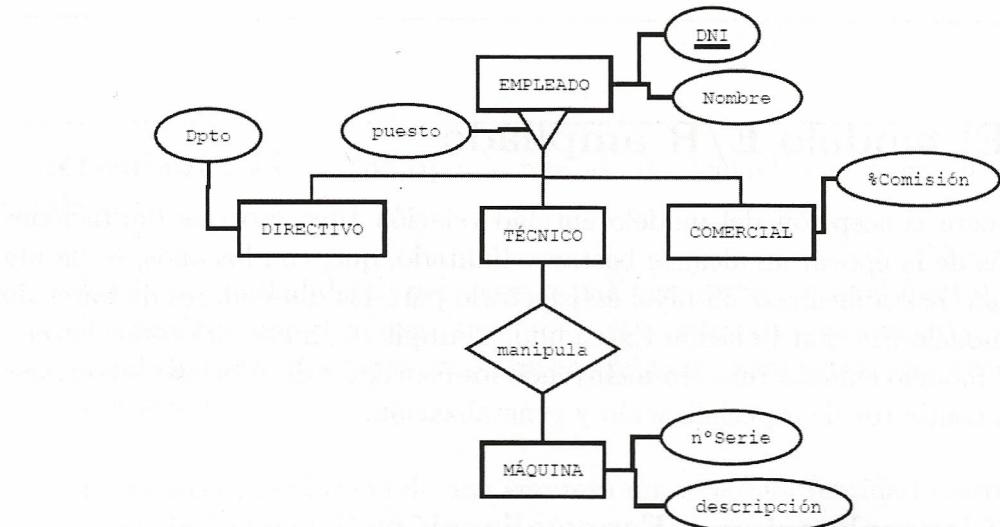


Figura 2.24: Ejemplo de generalización.

### Tipos de especialización

Se puede agregar más semántica al diagrama entidad relación extendido combinando los siguientes tipos de especialización:

- Especialización Exclusiva: En este caso, cada una de las ocurrencias de la superclase solo puede materializarse en una de las especializaciones. Por ejemplo, si un empleado es un directivo, no puede ser un técnico o un comercial. Para representar esta especialización exclusiva, el triángulo de la jerarquía lleva un arco.

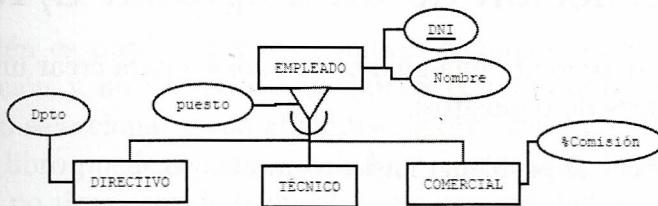


Figura 2.25: Especialización exclusiva.

- Especialización Inclusiva: Se produce cuando las ocurrencias de la superclase pueden materializarse a la vez en varias ocurrencias de las subclases. En este caso, el empleado directivo, podría ser también técnico y comercial. Se representa sin el arco, como en la figura 2.24.
- Especialización Total: Se produce cuando la entidad superclase tiene que materializarse obligatoriamente en una de las especializaciones. Se representan añadiendo un pequeño círculo al triángulo de la generalización:

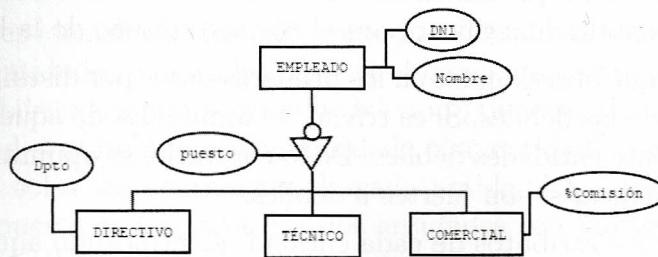


Figura 2.26: Especialización Total.

- Especialización Parcial: La entidad superclase no tiene por qué materializarse en una de las especializaciones (es opcional). Se representa sin el pequeño círculo, como en la figura 2.24.

◊ **Actividad 2.13:** Crea un E/R para almacenar datos de los distintos tipos de ordenadores que puede tener una organización. Clasifícalos en Sobremesa, Portátiles y Servidores, y asigna correctamente los atributos: N°Serie, Procesador, Memoria, CapacidadDisco, TipoBatería, DuraciónBatería, N°Procesadores y TipoProxy.

---

## 2.5. Construcción de un diagrama E/R

A continuación se presenta una guía metodológica para crear un entidad relación a partir de un análisis de requisitos:

1. Leer varias veces el problema hasta memorizarlo.
2. Obtener una lista inicial de candidatos a entidades, relaciones y atributos. Se realiza siguiendo los siguientes consejos:
  - Identificar las entidades. Suelen ser aquellos nombres comunes que son importantes para el desarrollo del problema. Por ejemplo, empleado, vehículo, agencia, etc. En principio, todos los conceptos deberían estar perfectamente especificados en el documento de requisitos, pero, de no existir el documento ERS, quizás solo se disponga de extractos de conversaciones con usuarios en las que se hacen referencias vagas a ciertos objetos, teniendo que hacer un importante ejercicio de abstracción para poder distinguir si son entidades, atributos, etc. Por ejemplo, un mecánico que tan solo habla de ciertos modelos de vehículos pertenecientes a determinadas personas, pero que nunca hace referencia a los vehículos *Diesel* que serán fundamentales identificar para el correcto diseño de la base de datos.
  - No hay que obsesionarse en los primeros pasos por distinguir las entidades fuertes de las débiles. Si es trivial, se toma nota de aquellas que parezcan claramente entidades débiles. De lo contrario, se apuntan como entidades sin especificar si son fuertes o débiles.
  - Extraer los atributos de cada entidad, identificando aquellos que pueden ser clave. Se suelen distinguir por ser adjetivos asociados a un nombre común seleccionadas como entidades. Por ejemplo, color, que es un adjetivo, puede ir asociado a la entidad vehículo. Además, se debe establecer el tipo de atributo, seleccionando si es opcional, obligatorio, multivalorado, compuesto o derivado. Si es compuesto se indica su composición, y si es derivado, cómo se calcula. Es bastante útil apuntar sinónimos utilizados para el atributo para eliminar redundancias.

- Es fácil identificar las generalizaciones si se obtiene un atributo que es aplicable a más de una entidad. En ese caso, se puede intentar aplicar una generalización/especialización, indicando cuál es la superclase y cuál las subclases. Además, se deben especificar los tipos de especialización (inclusiva, exclusiva, parcial, total).
  - Identificar los atributos de cada relación. Se suelen distinguir, al igual que los de entidad, por ser adjetivos, teniendo en cuenta que para que sean de relación, solo deben ser aplicables a la relación y no a ninguna de las de las entidades relacionadas.
  - También es posible que los nombres comunes contengan muy poca información y no sea posible incluirlas como entidades. En este caso, se pueden seleccionar como atributos de otra entidad, por ejemplo, el autor de un libro puede ser una entidad, pero si solo se dispone del nombre del autor, no tiene sentido incluirlo como una entidad con un único atributo. En este caso, se puede incluir como atributo de la entidad *Libro*.
  - Extraer los dominios de los atributos. Siempre es buena práctica, ir apuntando, aunque en el diagrama entidad relación no se exprese explícitamente, a qué dominio pertenece cada atributo. Por ejemplo, el Salario pertenece a los números reales (Salario: Reales), o el color de un objeto puede ser verde, amarillo o rojo (Color: Verde, Amarillo, Rojo).
  - Identificar las relaciones. Se pueden ver extrayendo los verbos del texto del problema. Las entidades relacionadas serán el sujeto y el predicado unidos por el verbo que hace de relación. Por ejemplo, agente inmobiliario vende edificio. En este caso el agente inmobiliario representaría una entidad el edificio la otra entidad y 'vende' sería la relación.
  - Una vez identificada las relaciones, hay que afinar cómo afecta la relación a las entidades implicadas. Este es el momento de distinguir las fuertes de las débiles haciendo preguntas del tipo ¿tiene sentido esta ocurrencia de entidad si quito una ocurrencia de la otra entidad? ¿se pueden identificar por sí solas las ocurrencias de cada entidad? Si a la primera pregunta la respuesta es negativa, las dos entidades son fuertes, si no, alguna de ellas es débil. Si la respuesta a la segunda es positiva, dependerán solo en existencia, si es negativa, alguna de las dos depende en identificación de la otra.
3. Averiguar las participaciones y cardinalidades. Generalmente se extraen del propio enunciado del problema. Si no vienen especificadas, se elegirá la que almacene mayor cantidad de información en la base de datos.

4. Poner todos los elementos listados en el paso 2 en un mapa y volver a considerar la pertenencia de cada uno de los elementos listados a su categoría. Así, se replanteará de nuevo si cierto atributo es una entidad, o si cierta entidad puede ser una relación, etc.
5. Refinar el diagrama hasta que se eliminen todas las incoherencias posibles, volviendo a los pasos anteriores en caso de encontrar algún atasco mental o conceptos dudosos que dificulten la continuación del análisis. Es bueno, en estas circunstancias discutir con compañeros u otros expertos sobre el diseño realizado.
6. Si hay dudas sobre el enunciado o sobre los requisitos, o se han quedado algunas cosas en el tintero, será necesario acudir al responsable del documento ERS o volver a concertar una entrevista con el usuario para aclarar conceptos. En este caso, se aclararán las dudas y se volverá al punto 2 para reiniciar el análisis.

**¿Sabías que . . . ?** En muchas ocasiones, cuando no se sabe cómo continuar, o no se encuentra la solución de un problema, basta con explicarle el problema a otra persona, aunque esa persona no tenga ni idea del tema o no sea experta en la materia. Automáticamente, la solución aparece. Se ha iniciado inconscientemente un proceso mental para ordenar las ideas que ha desembocado en la resolución del problema.

## 2.6. El modelo relacional

El objetivo principal del modelo relacional es proteger al usuario de la obligación de conocer las estructuras de datos físicas con las que se representa la información de una base de datos. Desvincular estas estructuras de datos, que son complejas por naturaleza, del diseño lógico (Modelo Relacional), permite que la base de datos se pueda implementar en cualquier gestor de bases de datos relacional (Oracle, MySQL, DB2, etc.) A continuación se enumeran las características fundamentales del modelo relacional:

- La relación es el elemento fundamental del modelo. Los usuarios ven la base de datos como una colección de relaciones. Estas relaciones se pueden operar mediante el *Álgebra Relacional*.
- El modelo relacional es independiente de la forma en que se almacenan los datos y de la forma de representarlos, por tanto, la base de datos se puede

implementar en cualquier SGBD y los datos se pueden gestionar utilizando cualquier aplicación gráfica. Por ejemplo, se pueden manejar las tablas de una base de datos MySQL u Oracle con Microsoft Access. Véase práctica 5.3.

- Al estar fundamentado en una fuerte base matemática, se puede demostrar la eficacia del modelo a la hora de *operar* conjuntos de datos.

### 2.6.1. Las relaciones en el modelo relacional

Se define una relación como un conjunto de atributos, cada uno de los cuales pertenece a un dominio, y que posee un nombre que identifica la relación. Se representa gráficamente por una tabla con columnas (atributos) y filas (tuplas). El conjunto de *tuplas* de una relación representa el *cuerpo* de la relación y el conjunto de atributos y el nombre representan el *esquema*.

Relación: ImpuestoVehículos					Nombre	Cabecera	Esquema
Vehículo	Dueño	TeléfonoDueño	Matrícula	Cuota			
Seat Ibiza TDI 1.9	Francisco	925884721	9918-FTV	92,24			
Volkswagen Polo TDI 1.0	Pedro	918773621	4231-FHD	61,98			
Renault Laguna Coupé	María	929883762	7416-GSJ	145,32			
Fiat Punto 1.0	Ernesto	646553421	9287-BHF	45,77			

Figura 2.27: Definición de Relación.

**¿Sabías que ...?** Actualmente, los modelos lógicos más extendidos con diferencia son el modelo relacional y los diagramas de clases que utiliza UML para modelar las bases de datos orientadas a objetos. El modelo relacional de Codd se ajusta a la perfección al modelo entidad/relación de Chen creado en la fase de modelización conceptual. No así el modelo de UML, que requiere técnicas más complejas y específicas como los casos de uso o los diagramas de transición de estados.

### 2.6.2. Otros conceptos del modelo relacional

A continuación se definen los conceptos necesarios para transformar el modelo conceptual (diagrama entidad-relación) en el modelo lógico (modelo relacional).

- **Atributo:** Características que describen a una entidad o relación.
- **Dominio:** Conjunto de valores permitidos para un atributo. Por ejemplo, cadenas de caracteres, números enteros, los valores Sí o No, etc.

- **Restricciones de semántica:** Condiciones que deben cumplir los datos para su correcto almacenamiento. Hay varios tipos:

- Restricciones de clave: Es el conjunto de atributos que identifican de forma única a una entidad.
- Restricciones de valor único (*UNIQUE*): Es una restricción que impide que un atributo tenga un valor repetido. Todos los atributos clave cumplen esta restricción. No obstante es posible que algún atributo no sea clave y requiera una restricción de valor único. Por ejemplo, el número de bastidor de un vehículo no es clave (lo es la matrícula) y sin embargo, no puede haber ningún número de bastidor repetido.
-  Restricciones de integridad referencial: Se da cuando una tabla tiene una referencia a algún valor de otra tabla. En este caso la restricción exige que exista el valor referenciado en la otra tabla. Por ejemplo, no se puede poner una nota a un alumno que no exista.
- Restricciones de dominio: Esta restricción exige que el valor que puede tomar un campo esté dentro del dominio definido. Por ejemplo, si se establece que un campo DNI pertenece al dominio de los números de 9 dígitos + 1 letra, no es posible insertar un DNI sin letra, puesto que la restricción obliga a poner al menos 1 letra.
- Restricciones de verificación (*CHECK*): Esta restricción permite comprobar si un valor de un atributo es válido conforme a una expresión.
- Restricción de valor NULO (NULL o NOT NULL): Un atributo puede ser obligatorio si no admite el valor NULO o *NULL*, es decir, el valor *falta de información o desconocimiento*. Si admite como valor el valor NULL, el atributo es opcional.
- Disparadores o *triggers*: Son procedimientos que se ejecutan para hacer una tarea concreta en el momento de insertar, modificar o eliminar información de una tabla.
- Restricciones genéricas adicionales o *aserciones (ASSERT)*: Permite validar cualquiera de los atributos de una o varias tablas.

- **Clave:** Una clave es un conjunto de atributos que identifican de forma única una ocurrencia de entidad. En este caso, las claves pueden ser simples (atómicas) o compuestas. Además, hay varios tipos de clave:

- Superclave: Identifican a una entidad (pueden ser no mínimas). Por ejemplo, para un empleado, las superclaves posibles son el DNI, o el DNI+Nombre, o el DNI+Nombre+Número de la Seguridad Social, etc.

- Clave Candidata: Es la mínima Superclave (en el caso anterior el DNI, o el Número de la seguridad social).
- Clave Primaria: Es la clave candidata elegida por el diseñador como clave definitiva (en el ejemplo anterior se elegiría el DNI por ser la más representativa para un empleado).
- Clave foránea: Es un atributo de una entidad, que es clave en otra entidad. Por ejemplo, la nota en un módulo de una asignatura corresponde a un DNI, que es clave de otra entidad. En este caso el DNI es una clave foránea en la tabla notas.

## 2.7. Transformación de un diagrama E/R al modelo relacional

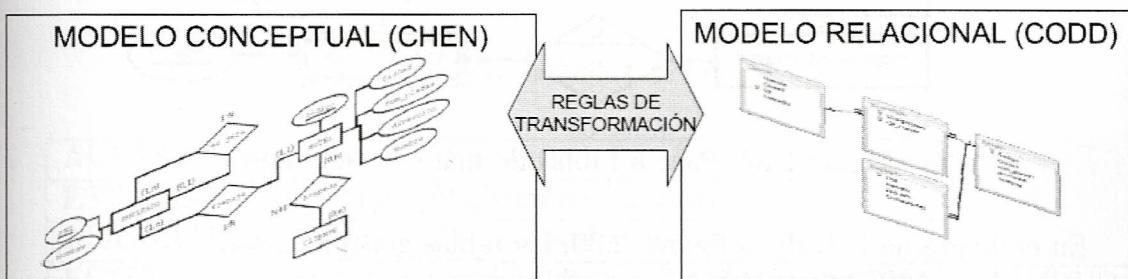


Figura 2.28: Transformación del modelo E/R en relacional.

Para realizar esta transformación se siguen las siguientes reglas:

### Transformación de entidades fuertes

Para cada entidad A, entidad fuerte, con atributos  $(a_1, a_2, \dots, a_n)$  se crea una tabla A (con el nombre en plural) con n columnas correspondientes a los atributos de A, donde cada fila de la tabla A corresponde a una ocurrencia de la entidad A. La clave primaria de la tabla A la forman los atributos clave de la entidad A.

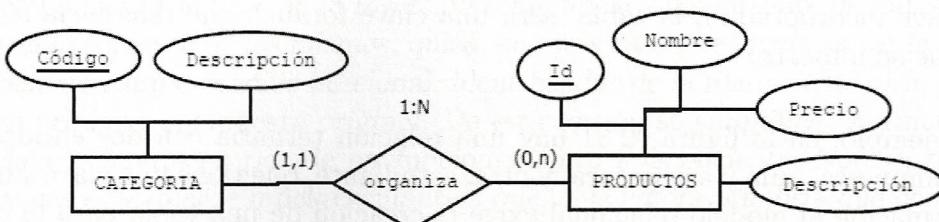


Figura 2.29: Paso a tablas de entidades fuertes.

En el diagrama E-R de la figura 2.29, las tablas generadas son:

CATEGORÍAS(Código, Descripción)

PRODUCTOS(Id,NOMBRE,Precio,Descripción)

## Transformación de entidades débiles

Para cada entidad débil D, con atributos  $cd_1, cd_2, \dots, cd_t, d_{t+1}, d_{t+2}, \dots, d_n$ , donde  $cd_1, cd_2, \dots, cd_t$  son los atributos clave de la entidad D, y una entidad fuerte F de la que depende D con atributos clave  $(cf_1, cf_2, \dots, cf_m)$ : se crea una tabla D con  $m+n$  columnas  $cd_1, cd_2, \dots, cd_n, d_{t+1}, d_{t+2}, \dots, d_n, cf_1, cf_2, \dots, cf_m$  correspondientes a los atributos de D y a los atributos clave de F. Si solo tiene dependencia de existencia, la clave primaria de la tabla D será la unión de los atributos clave de la entidad D. Si la entidad débil D, además, tiene una dependencia de identificación, la clave primaria de la tabla D será la unión de los atributos  $cd_1, cd_2, \dots, cd_t, cf_1, cf_2, \dots, cf_m$ , es decir, la unión de los atributos clave de la entidad débil D y de la entidad fuerte F.

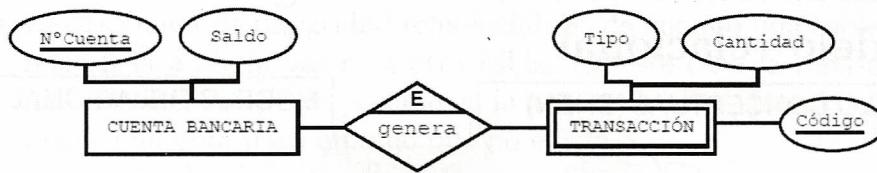


Figura 2.30: Paso a tablas de una entidad débil.

En el diagrama E-R de la figura 2.30, las tablas generadas son<sup>3</sup>:

CUENTAS_BANCARIAS(N°Cuenta, saldo)
TRANSACCIONES(Código, Tipo, Cantidad)

## Transformación de relaciones

Por cada relación R entre entidades  $E_1, E_2, \dots, E_N$ . La clave de  $E_i$  es  $C_i = a_{i1}, a_{i2}, \dots, a_{iN}$ .

Regla general para las relaciones: se crea una tabla con todos los campos claves de las entidades relacionadas y los atributos de la relación. La clave primaria de la tabla generada es la suma de los atributos claves de las entidades relacionadas, y cada clave incorporada a la tabla, será una clave foránea que referencia a la tabla de la que se importa.

Por ejemplo, en la figura 2.31 hay una relación ternaria con dos entidades con clave compuesta, aula y asignatura, y otra, estudiante, que tiene una clave simple. La transformación al modelo relacional exige la creación de una tabla para la relación. La tabla ESTUDIOS, tendrá como columnas los atributos clave del aula, los de asignatura y el atributo clave de estudiante, todos ellos formando la clave primaria y,

<sup>3</sup>Falta incorporar en la figura el atributo N°Cuenta a la tabla TRANSACCIONES, aquí se ha omitido para centrar la atención en los atributos de las entidades. Consultar la sección de excepciones en la transformación de relaciones.

al mismo tiempo, actuando como claves foráneas de sus respectivas tablas. Además, se incorpora el atributo de relación “hora”.

Cardinalidad N:M

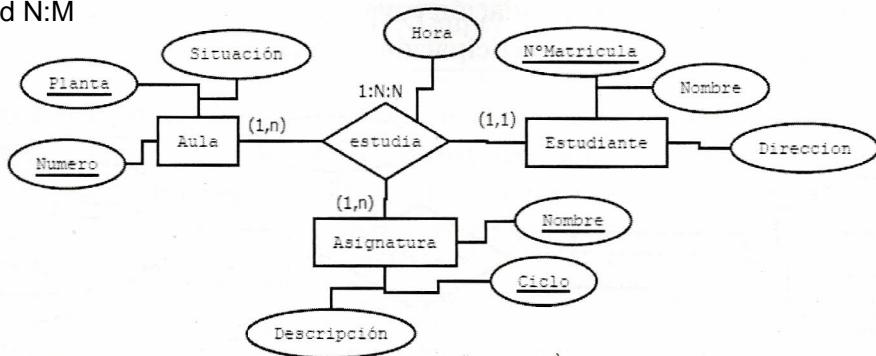


Figura 2.31: Paso a tablas de una relación.

AULAS(Número, Planta, Situación)
ESTUDIANTES(NºMatrícula,Nombre,Direccion)
ASIGNATURAS(Nombre, Ciclo,Descripción)
ESTUDIOS(Número, Planta, NºMatrícula, Nombre, Ciclo,Hora)

#### El consejo del buen administrador...

#### IMPORTANTE!!!!!!

Aunque en teoría, la tabla ESTUDIOS tiene como clave primaria la suma de las claves primarias de las tablas que relaciona, tener en una base de datos tablas con claves tan complejas, hace que el sistema pueda funcionar más lento de lo esperado debido a la multitud de comprobaciones que el gestor debe realizar cuando se inserta o modifica un dato. Si es un sistema cuyo funcionamiento se basa en la inserción o modificación constante de datos, más que en la consulta de los mismos, quizás, en estos casos, se pueda saltar la teoría y crear un campo sencillo adicional, identificador de la fila, y sustituirlo por la clave primaria compuesta original. De esta forma, se simplifica enormemente la clave primaria en pos de un funcionamiento más eficiente. Nótese, que en estos casos, se pierde mucha semántica que, o se ignora o habría que controlar de otros modos.

No siempre se aplica la regla general para crear una tabla por cada relación. Generalmente, se pueden encontrar las siguientes excepciones a la regla general

- Relaciones con cardinalidad 1:N.** En este caso, no se crea una tabla para la relación, sino que se añade a la tabla de la entidad que actúa con participación máxima N la clave de la entidad que actúa con participación máxima 1 (como clave foránea). Si además, la relación tuviera atributos se importarían también a la entidad que actúa con participación máxima N:

Se pasa la clave de la (1,1) a la (1,n):

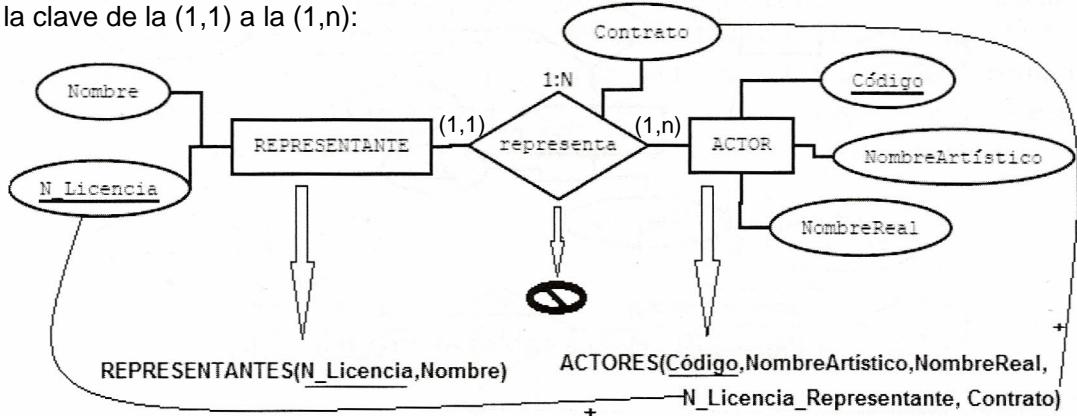


Figura 2.32: Excepción I. Relaciones 1-N.

La transformación quedaría como se ilustra en la Figura 2.32. Se puede observar que en este caso, no se ha creado una tabla para la relación, sino que se ha añadido a la tabla ACTORES la clave foránea N\_Licencia\_Representante que referencia al campo N\_Licencia de la tabla REPRESENTANTES. También se ha añadido el campo Contrato, atributo de la relación, a la tabla ACTORES.

ACTORES( <u>Código</u> , NombreArtístico, NombreReal, N_Licencia_Representante, Contrato)
REPRESENTANTES(N_Licencia,Nombre)

- Relaciones reflexivas con cardinalidad 1-N.** En este caso, tampoco se crea una tabla para la relación. Hay que crear una tabla con el nombre de la entidad, añadiendo otra vez la clave cambiada de nombre.

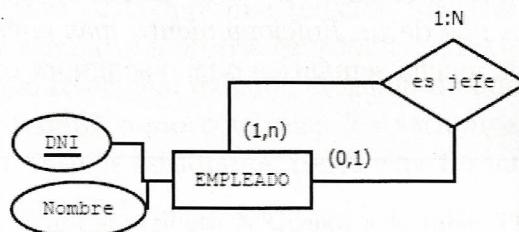


Figura 2.33: Excepción II. Relaciones reflexivas 1-N.

En el ejemplo de la Figura 2.33, el empleado solo puede tener un jefe, por tanto, se incorpora el DNI del jefe del empleado (DNISupervisor) como clave foránea.

EMPLEADOS(DNI,NOMBRE,DNISupervisor)
-------------------------------------

◊ **Actividad 2.14:** En las relaciones reflexivas con cardinalidad m-n, se aplica la regla general para la transformación de relaciones. Expresa cómo sería la regla para crear tablas con relaciones reflexivas con cardinalidad m-n. Despues, aplica esa regla para transformar la Figura 2.33 suponiendo que tuviera cardinalidad m-n.

3. **Relaciones 1-1.** Este tipo de relaciones tampoco generan tabla. El paso a tablas se realiza de forma muy parecida a las relaciones 1-N. En este caso, tampoco se genera tabla para la relación y se tiene la libertad de poder incorporar la clave de una de las dos entidades a la otra.

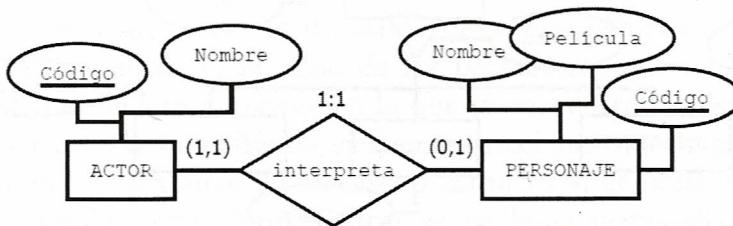


Figura 2.34: Excepción III. Relaciones 1-1.

En este caso existen las siguientes opciones:

- (0,1) - (1,1) ▪ Incorporar la clave de Personajes como clave foránea en la tabla actores:  

ACTORES(Codigo, Nombre, CódigoPersonaje)
PERSONAJES(Codigo,Nombre,Película)
- (1,1) - (0,1) ▪ Incorporar la clave de Actores como clave foránea en la tabla Personajes:  

ACTORES(Codigo, Nombre)
PERSONAJES(Codigo,Nombre,Película,CódigoActor)
- (1,1) - (1,1) ▪ Incorporar la clave de Actores como clave foránea en la tabla Personajes y la clave de Personajes a la tabla de Actores como clave foránea<sup>4</sup>:  

ACTORES(Codigo, Nombre,CódigoPersonaje)
PERSONAJES(Codigo,Nombre,Película,CódigoActor)



<sup>4</sup>Téngase en cuenta, que en este caso se está introduciendo una pequeña redundancia, pero que puede ser de mucha utilidad para simplificar futuras consultas.

## Participaciones 0,x

Normalmente las participaciones son importantes para calcular la cardinalidad de la relación, y transformar conforme a las reglas expuestas hasta ahora. Incluso en muchas ocasiones, las participaciones se omiten en los diagramas entidad-relación. No obstante, es necesario tener en cuenta cuándo la participación tiene un mínimo de 0, para adoptar un campo de una tabla como opcional *NULL*, u obligatorio *NOT NULL*.

## Generalizaciones y especializaciones

Para transformar las generalizaciones se puede optar por 4 opciones. Cada opción se adaptará mejor o peor a los diferentes tipos de especialización (Exclusiva, Inclusiva, Total, Parcial).

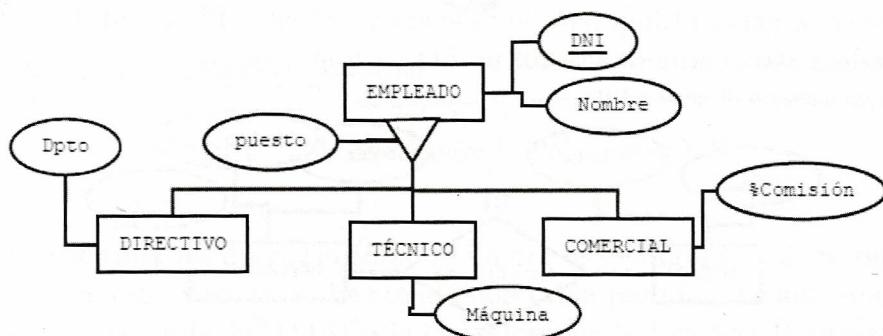


Figura 2.35: Paso a tablas de generalizaciones.

1. Se puede crear una tabla para la superclase y otras tantas para cada subclass, incorporando el campo clave de la superclase a las tablas de las subclasses.

EMPLEADOS(DNI, Nombre, Puesto)
DIRECTIVOS(DNI, Dpto)
TECNICOS(DNI, Máquinas)
COMERCIALES(DNI, Comisión)

2. Se puede crear una tabla para cada subclass incorporando todos los atributos de la clase padres, y no crear una tabla para la superclase.

DIRECTIVOS(DNI, Nombre, Puesto, Dpto)
TÉCNICOS(DNI, Nombre, Puesto, Máquinas)
COMERCIALES(DNI, Nombre, Puesto, Comisión)

3. Se puede crear una sola tabla para la superclase, incorporando los atributos de todas las subclasses y añadir, para distinguir el tipo de la superclase, un campo

llamado “tipo”, que contendrá el tipo de subclase al que representa cada tupla. Este tipo de opción se adapta muy bien a las especializaciones exclusivas.

**EMPLEADOS(DNI, Nombre,Puesto, Dpto, Máquinas, Comisión, Tipo)**

4. Se puede crear una sola tabla para la superclase como en la opción anterior, pero en lugar de añadir un solo campo “tipo”, se añaden varios campos que indiquen si cumple un perfil, de este modo se soportan las especializaciones inclusivas.

**EMPLEADOS(DNI, Nombre,Puesto, Dpto, Máquinas, Comisión, EsDirectivo, EsTécnico, EsComercial)**

## 2.8. Normalización

Habitualmente, el diseño de una base de datos termina en el paso del modelo entidad-relación al modelo relacional. No obstante, siempre que se diseña un sistema, no solo una base de datos, sino también cualquier tipo de solución informática, se ha de medir la calidad de la misma, y si no cumple determinados criterios de calidad, hay que realizar, de forma iterativa, sucesivos *refinamientos* en el diseño, para alcanzar la calidad deseada. Uno de los parámetros que mide la calidad de una base de datos es la *forma normal* en la que se encuentra su diseño. Esta forma normal puede alcanzarse cumpliendo ciertas restricciones que impone cada forma normal al conjunto de atributos de un diseño. El proceso de obligar a los atributos de un diseño a cumplir ciertas formas normales se llama **normalización**.

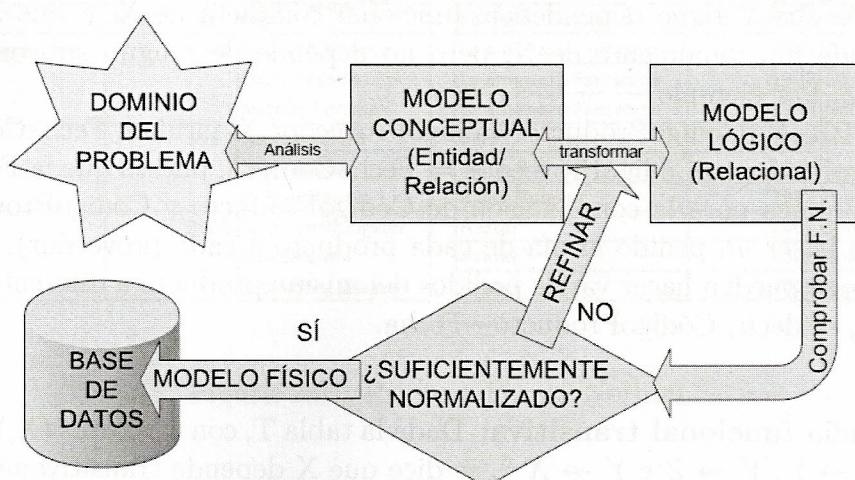


Figura 2.36: Refinamiento de un diseño de base de datos.

Las formas normales pretenden alcanzar dos objetivos:

1. Almacenar en la base de datos cada hecho solo una vez, es decir, evitar la redundancia de datos. De esta manera se reduce el espacio de almacenamiento.
2. Que los hechos distintos se almacenen en sitios distintos. Esto evita ciertas anomalías a la hora de operar con los datos.

En la medida que se alcanza una forma normal más avanzada, en mayor medida se cumplen estos objetivos. Hay definidas 6 formas normales, cada una agrupa a las anteriores, de forma que, por ejemplo, la forma normal 3 cumple la forma normal 2 y la forma normal 1.

Antes de abordar las distintas formas normales, es necesario definir los siguientes conceptos <sup>5</sup>:

**Dependencia funcional:** Se dice que un atributo Y depende funcionalmente de otro atributo X, o que  $X \rightarrow Y$ , si cada valor de X tiene asociado en todo momento un único valor de Y. También se dice que X implica Y, y por tanto, que X es el *implicante*. Por ejemplo:

PRODUCTOS (CódigoProducto, Nombre, Precio, Descripcion)

CódigoProducto → Nombre, puesto que un código de producto solo puede tener asociado un único nombre, dicho de otro modo, a través del código de producto se localiza un único nombre.

### clave compuesta

**Dependencia funcional completa:** Dado una combinación de atributos  $X(X_1, X_2, \dots)$ , se dice que Y tiene dependencia funcional completa de X, o que  $X \Rightarrow Y$ , si depende funcionalmente de X, pero no depende de ningún subconjunto del mismo. Por ejemplo:

COMPRAS (CódigoProducto, CódigoProveedor, Cantidad, FechaCompra)

CódigoProducto, CódigoProveedor ⇒ FechaCompra, puesto que la FechaCompra es única para la combinación de CódigoProducto y CódigoProveedor (se puede hacer un pedido al día de cada producto a cada proveedor), y sin embargo, se pueden hacer varios pedidos del mismo producto a diferentes proveedores, es decir, CódigoProducto → Fecha.

**Dependencia funcional transitiva:** Dada la tabla T, con atributos (X,Y,Z), donde  $X \rightarrow Y$ ,  $Y \rightarrow Z$  e  $Y \Rightarrow X$  <sup>6</sup>, se dice que X depende transitivamente de Z, o que,  $X \rightarrow Z$ .

---

<sup>5</sup>No es objetivo del libro entrar en los detalles matemáticos del proceso de normalización, y sí proporcionar al lector una idea intuitiva del mismo.

<sup>6</sup>Y no depende de X.

Ejemplo 1:

PRODUCTOS (CódigoProducto, Nombre, Fabricante, País)

CódigoProducto → Fabricante

Fabricante → País

CódigoProducto → País, es decir, CódigoProducto depende transitivamente de País.

Ejemplo 2:

PRODUCTOS (CódigoProducto, Nombre, Fabricante, País)

CódigoProducto → Nombre

Nombre → CódigoProducto

CódigoProducto → Nombre

A continuación, se describe de forma intuitiva las siguientes formas normales:

 FN 1: En esta forma normal se prohíbe que en una tabla haya atributos que puedan tomar más un valor. Esta forma normal es inherente al modelo relacional, puesto que las tablas gestionadas por un SGBD relacional, están construidas de esta forma. Por ejemplo, en la Figura 2.37 se puede ver la diferencia entre una tabla que cumple la fn1 y otra que no:

\*\*\*

Película	Año	Actor
La amenaza Fantasma	1999	Ewan McGregor Liam Neeson Natalie Portman
Blade Runner	1982	Harrison Ford Sean Young Rutger Hauer
Avatar	2009	Sam Worthington Zoe Saldana Sigourney Weaver

*NO CUMPLE FN1*

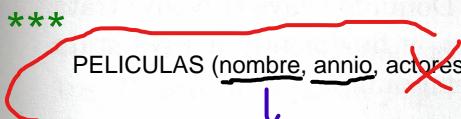
*CUMPLE FN1*

Película	Año	Actor
La amenaza Fantasma	1999	Ewan McGregor
La amenaza Fantasma	1999	Liam Neeson
La amenaza Fantasma	1999	Natalie Portman
Blade Runner	1982	Harrison Ford
Blade Runner	1982	Sean Young
Blade Runner	1982	Rutger Hauer
Avatar	2009	Sam Worthington
Avatar	2009	Zoe Saldana
Avatar	2009	Sigourney Weaver

Figura 2.37: Primera Forma Normal.

\*\*\*

PELICULAS (nombre, anno, actores)



ACTORES (nombre\_pelicula, nombre\_actor)

**FN 2:** Un diseño se encuentra en FN2 si está en FN1 y además, cada atributo que no forma parte de la clave tiene dependencia completa de la clave principal.

Ejemplo:

COMPRAS (CódigoProducto, CódigoProveedor, NombreProducto, Cantidad, FechaCompra).

CódigoProducto → NombreProducto, por tanto, al no ser dependencia funcional completa, no está en FN2. Se crea otra tabla: PRODUCTOS (codigo\_producto, nombre)

**FN 3:** Un diseño se encuentra en FN3 si está en FN2 y además, no hay ningún atributo no clave que depende de forma transitiva de la clave.

Ejemplo:

PRODUCTOS (CódigoProducto, Nombre, Fabricante, País).

CódigoProducto → Fabricante

Fabricante → País

CódigoProducto → País

País depende transitivamente de CódigoProducto, por tanto, no está en tercera forma normal.

**FNBC:** Esta forma normal, llamada Forma Normal de Boyce-Codd, exige que el modelo esté en FN3, y que además, todo implicante de la tabla, sea una clave candidata.

Ejemplo:

NOTAS (DNIAlumno, DNIProfesor, NombreProfesor, Nota).

DNIProfesor → NombreProfesor

NombreProfesor → DNIProfesor

DNIProfesor, DNIAlumno → Nota

En este caso, la tabla está en 3FN porque no hay dependencias funcionales transitivas, y sin embargo no está en FNBC porque NombreProfesor y DNI-Profesor son implicantes, y no son claves candidatas. Para obtener la tabla en FNBC habría que quitar de la tabla los atributos DNIProfesor y NombreProfesor.

**Otras formas normales:** Existen más formas normales (FN4, FN5, FNDK, FN6 cuyo alcance excede el de este libro y cuya aplicación en el mundo real es únicamente teórica). Las formas normales 4 y 5, se ocupan de las dependencias entre atributos multivaluados, la Forma Normal Dominio Clave (FNDK) trata las restricciones y los dominios de los atributos, y finalmente la FN6 trata ciertas consideraciones con las bases de datos temporales.

## 2.9. Prácticas Resueltas

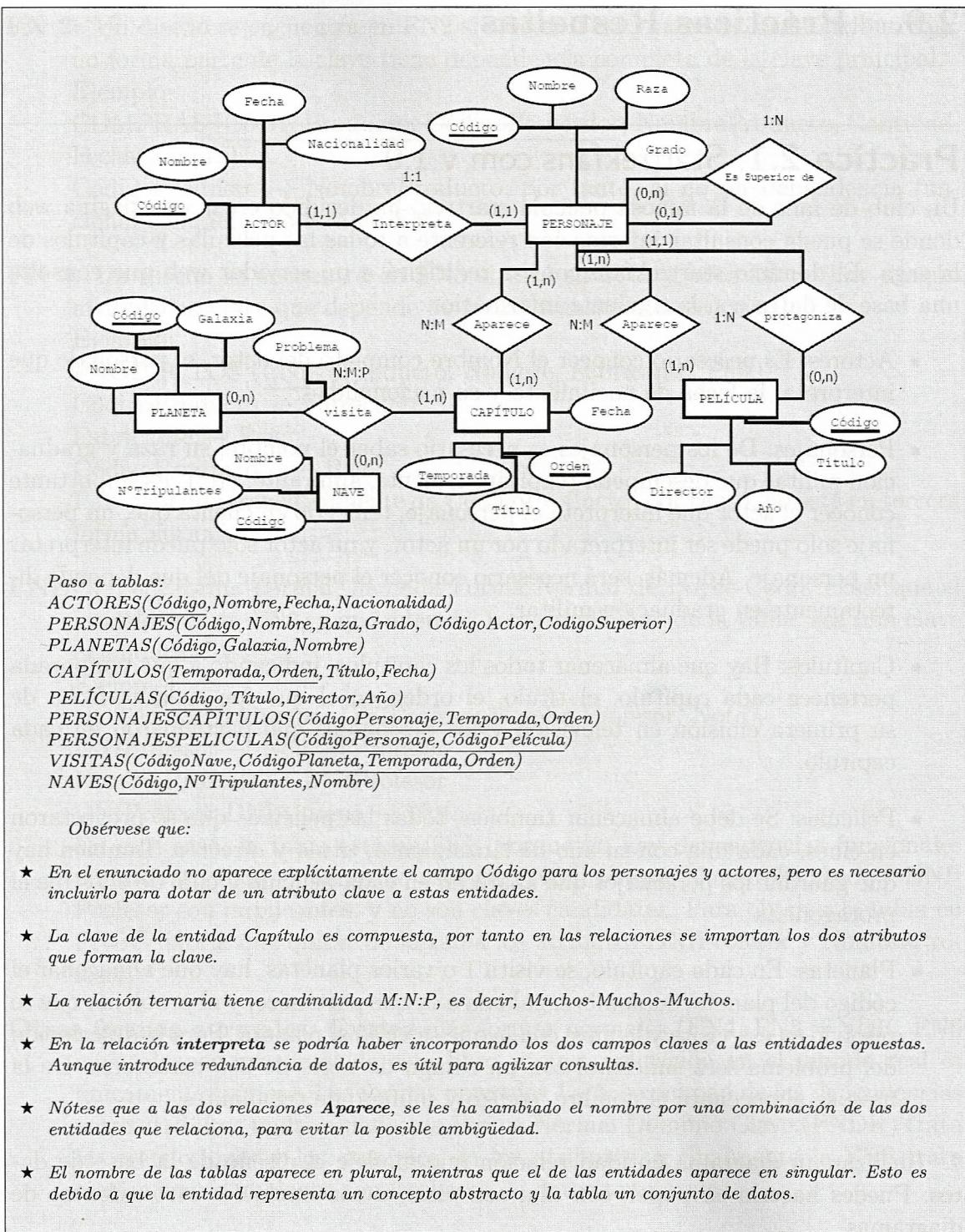
### Práctica 2.1: Startrekfans.com v.1.0

Un club de fans de la famosa película startrek, ha decidido crear una página web donde se pueda consultar información referente a todas las películas y capítulos de la saga. El dominio startrekfans.com se redirigirá a un servidor web que consulte una base de datos con la siguiente información:

- Actores: Es necesario conocer el Nombre completo del actor, el personaje que interpreta, la fecha de nacimiento y su nacionalidad.
- Personajes: De los personajes es necesario saber el nombre, su raza y graduación militar que desempeña (capitán, teniente, almirante, etc.). Es importante conocer el actor que interpreta el personaje, teniendo en cuenta que, un personaje solo puede ser interpretado por un actor, y un actor solo puede interpretar un personaje. Además, será necesario conocer el personaje del que depende directamente en graduación militar.
- Capítulos: Hay que almacenar todos los capítulos, indicando a qué temporada pertenece cada capítulo, el título, el orden en el que fue rodado, fecha de su primera emisión en televisión y los personajes que participaron en cada capítulo.
- Películas: Se debe almacenar también, todas las películas que se proyectaron en cines, cada una con su año de lanzamiento, título y director. También hay que guardar los personajes que aparecen en cada película y cuál de ellos fue el protagonista.
- Planetas: En cada capítulo, se visita 1 o varios planetas, hay que almacenar el código del planeta, su nombre, galaxia a la que pertenece, y el problema que se resolvió en esa visita y la nave con la que se viajó al planeta. Para la descripción del problema será suficiente con un campo de texto de 255 caracteres. De la nave se almacenará el nombre, código y número de tripulantes.

1. Realizar un diagrama entidad relación que modele el diseño de la base de datos. Puedes hacerlo en papel, con dia, con visio, o con cualquier otro software de diagramas.

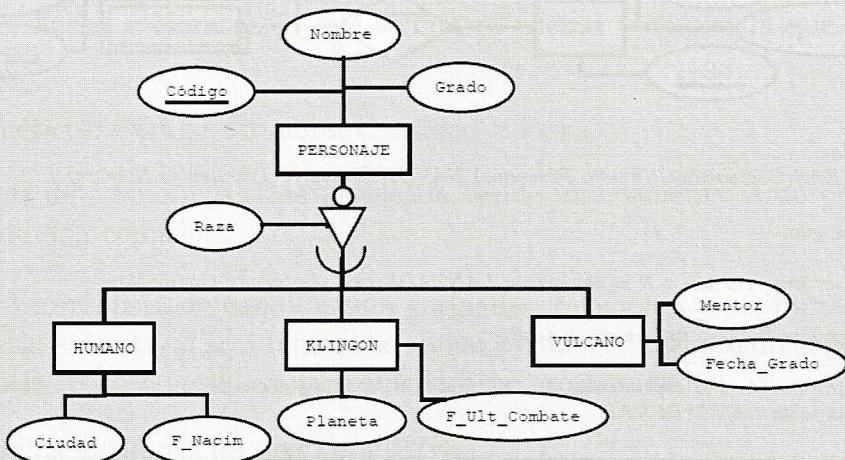
2. Realizar la conversión al modelo relacional del diagrama realizado en el primer Apartado, indicando qué claves primarias y foráneas se han de crear.



## Práctica 2.2: Startrekfans.com v.2.0

El club de fans de Star Trek ha pensado ampliar los requisitos de la página web para hacer una segunda versión. Esta segunda versión consiste en incluir información extra para los personajes. De esta manera, si el personaje es un humano, se indicará su fecha de nacimiento y ciudad terráquea donde nació. Si el personaje es de la raza Vulcano, se almacenará el nombre del mentor y la fecha de graduación, y si es de raza Klingon, se guardará su planeta natal y la fecha de su último combate.

1. Realizar una generalización de la entidad Personaje indicando las especializaciones necesarias.
2. Transforma al modelo relacional la generalización del apartado anterior.



Paso a tablas:

**PERSONAJES**(Código, Nombre, Grado, CódigoActor, CódigoSuperior, Raza, Ciudad, F\_Nacim, Planeta, F\_Ult\_Combate, Mentor, Fecha\_Grado)

Obsérvese que:

- ★ Se ha optado por una de las 4 opciones que hay. En este caso, se ha creado una única tabla con el campo **Raza** como discriminante de tipo, que indicará los campos a rellenar según su tipo. Por ejemplo, si la raza es **Klingon**, tan solo se rellenarán los campos **Planeta** y **F\_Ult\_Combate**. Esto generará 4 valores nulos por personaje, pero evitará complejidad en el modelo al no tener tablas extra.
- ★ Al tratarse de una especialización total y exclusiva, cada personaje solo puede pertenecer a una de las razas, y además tiene que pertenecer a una de manera obligatoria.

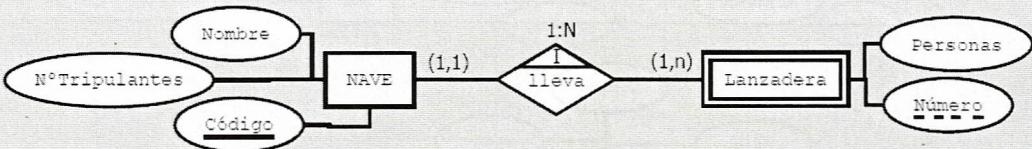


## Práctica 2.3: Startrekfans.com v.3.0

El club de fans de Startrek quiere la tercera versión de la base de datos de la siguiente forma:

En cada capítulo, la nave que viaja a un planeta, puede disponer de una nave pequeña llamada lanzadera con la que bajan a la superficie del planeta. La existencia de la lanzadera, solo tiene sentido si existe la nave a la que pertenece. Se identificará cada lanzadera mediante un número entero y el código de la nave. Es necesario conocer la capacidad en personas de la lanzadera.

1. Incorporar los cambios de la tercera versión al modelo conceptual y lógico de las prácticas anteriores.



*Paso a tablas:*

LANZADERAS(CódigoNave, Número, Personas) NAVES(Código, N° Tripulantes, Nombre)

*Obsérvese que:*

- ★ Al ser una relación 1-N se importa a LANZADERAS el atributo CódigoNave.
- ★ El campo Número es una clave débil.
- ★ Al tratarse de una dependencia de identificación, el CódigoNave forma parte de la clave primaria de la tabla LANZADERAS.

## 2.10. Prácticas Propuestas

---

### Práctica 2.4: Peluquería

Una peluquería desea llevar el control de sus empleadas y de sus clientes así como de los servicios que se prestan. Se desea almacenar la siguiente información:

- Empleadas: DNI, Nombre, Especialidad (Masaje, Corte, Color, Brushing, Manicuras, Rulos, etc.)
  - Clientes: Datos personales (DNI, Nombre, Profesión, Teléfono y Dirección) y los tratamientos médicos a los que está sometido el cliente.
  - Servicios prestados: Qué empleada atendió a qué cliente, y qué tipo de servicio le prestó en qué fecha y hora.
  - Citas: Fecha y Hora en la que se cita al cliente y empleada que realizará el servicio.
  - Cosméticos: Código, Nombre, Cantidad y Precio.
  - Ventas de cosméticos: Una empleada vende un cosmético a un cliente, obteniendo una comisión.
1. Realiza una lista de candidatos a entidades, relaciones y atributos, indicando en cada entrada, si se admite o se rechaza como tal razonando el porqué de tu decisión. Por cada relación, razona su tipo y cardinalidad.
  2. Modeliza mediante un diagrama E/R.
  3. Realiza el paso a tablas del modelo E/R.



## Práctica 2.5: Reyes Magos sin fronteras

La ONG *Reyes Magos sin fronteras* desea hacer una base de datos para que esta Navidad todos los niños pobres de España puedan recibir sus regalos la noche de los Reyes Magos. La ONG contacta con vecinos de distintos barrios para disfrazarlos de Reyes Magos y organizarlos en grupos lúdicos que realicen eventos para que los niños los visiten y puedan formular sus peticiones. Cada niño es recibido por un Rey Mago y puede hacer una única petición, la cual queda anotada en la base de datos para posteriormente, el día 6 de enero, entregar esa petición. La ONG comprará los regalos con el dinero que distintas organizaciones benéficas aportarán a la causa.

Los datos que interesa almacenar son los siguientes:

- De los vecinos: DNI, Nombre y apellidos, Rey Mago al que encarna y los vecinos a los que ha conseguido convencer para que se unan a la causa.
  - De los niños: Nombre, Dirección y el Regalo que pide al Rey Mago. (Los niños, no tienen DNI, y necesitarán un dato identificativo).
  - De los grupos de vecinos se necesita saber a qué Barrio pertenecen, número de integrantes del grupo y los Eventos que han organizado.
  - De los eventos interesa conocer la Ubicación física, la Fecha, la Hora y los niños asistentes.
1. Realiza una lista de candidatos a entidades, relaciones y atributos, indicando en cada entrada, si se admite o se rechaza como tal justificando el porqué de tu decisión. Por cada relación, razona su tipo y cardinalidad.
  2. Realiza el diagrama entidad-relación y el paso al modelo relacional.

## Práctica 2.6: Mundial de fútbol

Diseñar una base de datos para organizar el campeonato mundial de fútbol. Considerar los siguientes aspectos:

- Jugadores: un jugador puede pertenecer a un único equipo y puede actuar en varios puestos distintos, pero en un determinado partido solo puede jugar en un puesto.
  - Árbitros: En un partido intervienen 3 árbitros titulares, linier derecho, izquierdo, principal y un árbitro secundario. Un árbitro puede realizar una función en un partido y otra distinta en otro partido.
  - Estadísticas: Se desea saber los goles que ha marcado un jugador en qué partido y en qué minuto, también se desea poder describir cómo sucedió el gol.
  - Porteros: Se desea almacenar cuántas paradas ha realizado, en qué minuto, y en qué situación se han producido: penalti, tiro libre, corner o jugada de ataque.
  - Partidos: Todos generan un acta arbitral donde se pueden incluir todo los comentarios que el árbitro considere oportuno: lesiones, expulsiones, tarjetas, etc.
1. Realiza una lista de candidatos a entidades, relaciones y atributos, indicando en cada entrada, si se admite o se rechaza como tal, justificando el porqué de tu decisión. Por cada relación, razona su tipo y cardinalidad.
  2. Realiza el diagrama entidad-relación y el paso al modelo relacional.



## Práctica 2.7: Supermercado virtual

Se va a desarrollar una aplicación informática para [www.virtualmarket.com](http://www.virtualmarket.com) cuya interfaz de usuario estará basada en páginas web para que los clientes puedan realizar compras desde sus casas. La empresa dispone de una serie de repartidores que se encargan de distribuir los pedidos a los clientes. A continuación se muestra el informe de un analista tras una entrevista con el cliente:

La aplicación permitirá registrar nuevos clientes. Para usar la aplicación, un cliente deberá registrarse indicando sus datos personales (DNI, Nombre, Dirección, Código Postal, Teléfono de contacto, email y password) a través de un formulario de registro. Una vez registrado podrá acceder a la realización de pedidos con su email y su password.

Los productos que oferta el supermercado están divididos en diversas categorías. Los datos necesarios para cada categoría son: nombre de la categoría, condiciones de almacenamiento (frío, congelado, seco) y observaciones. Los datos de los productos son: nombre, marca, origen, dimensiones (volumen y peso), una fotografía, la categoría y unidades disponibles <sup>7</sup>.

La aplicación permitirá visualizar un listado de productos ordenado por categoría, permitiendo seleccionar los productos que deseé comprar mediante una caja de texto donde se indicará el número de unidades seleccionadas. La aplicación llevará la cuenta (cesta de la compra) de los productos que el cliente ha ido seleccionando.

La aplicación permitirá también efectuar un pedido con todos los productos que lleve almacenados en su cesta de la compra. Los datos del pedido son: código del pedido, fecha del pedido, cliente, dirección de entrega, productos pedidos, importe total del pedido y datos de pago (número de tarjeta y fecha de caducidad)<sup>8</sup>.

Para poder generar un pedido se deberán dar dos situaciones:

- El cliente deberá pertenecer a una zona (Código Postal) donde existan repartidores. Un repartidor se identifica mediante un nombre, número de matrícula de la furgoneta y zona donde reparte.
- Debe haber unidades suficientes por cada producto para satisfacer las demandas de cada pedido.

Una vez generado el pedido se mostrará al usuario una página con los datos de su pedido, se restarán del stock las unidades pedidas y se emitirá una nota de entrega

---

<sup>7</sup>El control del stock está supervisado por otra aplicación subcontratada, y por tanto no es necesario preocuparse por él.

<sup>8</sup>El pago del pedido está automatizado mediante otro software que proporciona el banco.

(albarán) a los responsables de almacén para que sirvan ese pedido.

Se pide:

1. Diseño Conceptual. Realizar el diagrama entidad relación de la aplicación. Se ha de tener en cuenta que el entrevistado narra todo el proceso que necesita la lógica de su negocio. Se deberán separar los procedimientos de los datos.
  2. Diseño Lógico. Realizar el paso al modelo relacional.
  3. ¿En qué forma normal está la tabla Clientes?
- 
- ◊

---

### Práctica 2.8: Requisitos de una aplicación

Busca alguna persona que tenga un negocio. Puedes trabajar haciendo equipo con otros compañeros. Tenga informatizado el negocio o nó, pídele que sea tu cliente y realiza las siguientes tareas de análisis:

- Entrevéstale, si es necesario graba la entrevista e intenta extraer una lista de requisitos.
- Extrae de esta lista de requisitos los que versen sobre almacenamiento de la información en bases de datos y realiza un diagrama entidad relación que satisfaga todos esos requisitos.
- Reúnete de nuevo con el cliente y pídele que verifique tu modelo E-R.
- Repite este proceso hasta que el cliente de el visto bueno.

Cuando el cliente haya validado tu modelo, realiza el diseño de base de datos, transformando el modelo E-R al modelo relacional y después, implementando ese modelo en Access.

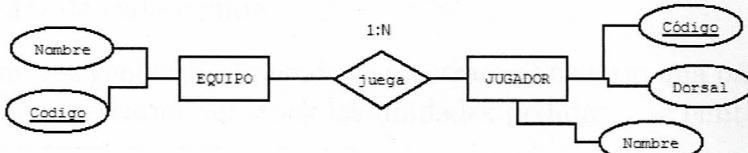
---

◊

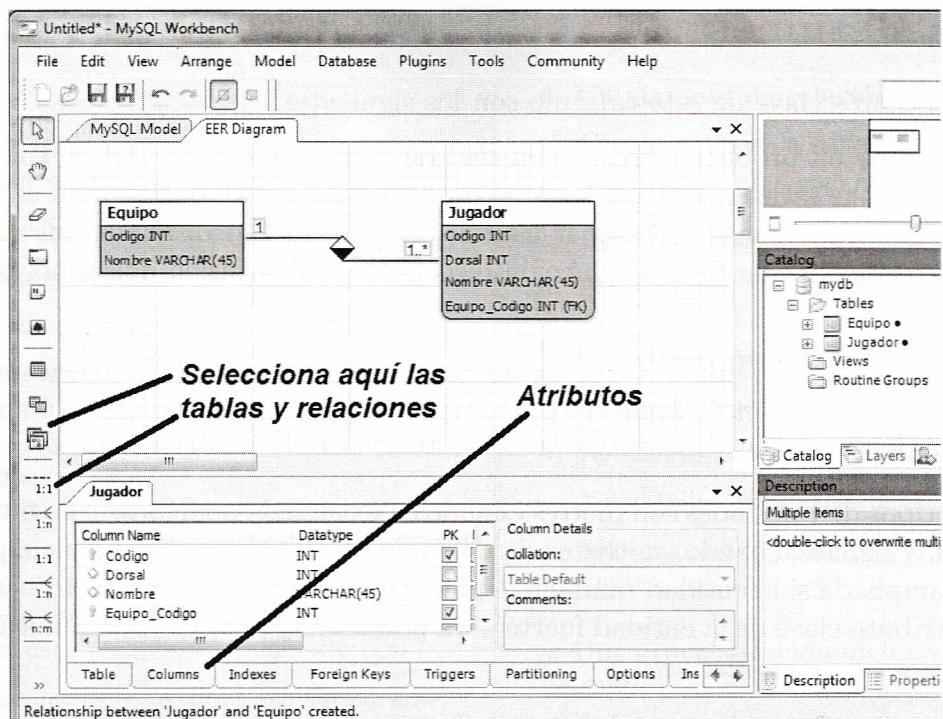
## Práctica 2.9: MySQL Workbench

El objetivo de esta práctica es conocer en profundidad un software de diseño de base de datos de licencia GNU. Debes tener en cuenta que los diagramas de MySQL Workbench son una mezcla del modelo conceptual de Chen y del modelo relacional de Codd, incluso más orientado al modelo relacional. Por ejemplo, no existen las relaciones ternarias, y a las entidades las llama tablas, como en el modelo relacional. Toda su nomenclatura está orientada a comunicarse con un SGBD sin necesidad de transformaciones previas. Además, aunque tiene el apellido MySQL, este software está basado en DB Designer, programa genérico de diseño de base de datos, y por tanto se puede conectar a cualquier SGBD (Oracle, DB2, Access...) y no solo a MySQL.

1. Conectarse a la página web de MySQL Workbench. <http://wb.mysql.com/>
2. Seleccionar la opción “Download Workbench” y mientras se descarga, selecciona el enlace “About - Screenshots” y comenta para qué sirve el software.
3. Buscar en google “MySQL workbench manual” y descárgate el manual de instrucciones de MySQL Workbench.
4. Instalar el software. El proceso de instalación es muy sencillo.
5. Iniciar la aplicación y seleccionar la opción *Add Diagram* para comenzar a dibujar un diagrama entidad relación (EER (Extended Entity Relationship - Entidad Relación Extendido). En la opción de menú *Model*, seleccionar la notación para los objetos y para las relaciones. Seleccionar la opción “Classic” y compararlas con las otras notaciones.
6. Insertar los objetos necesarios para representar el siguiente diagrama E-R:



Se puede usar el panel inferior para incorporar los atributos y así, del panel lateral izquierdo, poder seleccionar los objetos. Debe quedar así:



7. Observar que al generar la relación, automáticamente se crea una clave foránea en la tabla jugador que representa el equipo en el que juega.
8. Experimentar con el diseño para crear la relación 'juega' como n-m en lugar de 1 – n. ¿Qué sucede? Observar que automáticamente crea una nueva tabla.
9. Descargar el fichero sakila-db.zip de la página web de MySQL. Se encuentra muy fácil escribiendo *sakila-db.zip* en google. Descomprimir y abrir el archivo *sakila.mwb*. Observar cómo están organizadas en regiones las tablas y las relaciones del modelo.
10. Realizar los diagramas de las prácticas anteriores con MySQL Workbench y mediante la opción de menú *Database, Forward Engineer*, generar automáticamente los scripts de creación de base de datos.

◊

## 2.11. Resumen

Los conceptos clave de este capítulo son los siguientes:

- Modelizar un problema consiste en realizar múltiples abstracciones. En bases de datos, se utilizan tres modelos: el modelo conceptual, o diagrama entidad-relación, que es más cercano al usuario, el modelo lógico, que es un modelo más técnico y que tiene traducción directa al modelo físico que soportan los SGBD.
- Los componentes que hay que detallar en un diagrama entidad relación son: Entidades, Atributos, Relaciones, Participaciones, Cardinalidades y Generalizaciones.
- Los tipos de Entidades son fuertes cuando su existencia no depende de ninguna otra, y débiles, cuando su existencia depende de otra. Esta dependencia puede ser ampliada si la entidad también depende en Identificación, es decir, necesita el atributo clave de la entidad fuerte para poder identificar de forma única cada ocurrencia de entidad.
- Los atributos pueden ser clave o no clave, univaluados, multivaluados, compuestos, derivados, obligatorios u opcionales.
- Las relaciones pueden ser, según su grado, binarias, ternarias, reflexivas o n-arias.
- La cardinalidad de una relación se calcula tomando las participaciones máximas y mínimas de las ocurrencias de una entidad en la relación. Estas pueden ser 1-1, 1-N o M-N.
- Las generalizaciones pueden dar lugar a cuatro tipos de especializaciones, exclusivas, inclusivas, totales o parciales.
- El modelo relacional expresa, mediante relaciones, todos los conceptos detallados en el modelo conceptual.
- Se puede transformar el modelo entidad relación en un modelo relacional generando una tabla para las relaciones de tipo N:M y las n-arias. Para las relaciones de tipo 1-N se importan los atributos clave de la entidad cuya cardinalidad es 1 a la que tiene como cardinalidad N. Las relaciones 1:1 y las generalizaciones tienen varias opciones de transformación.
- La normalización es un proceso que sirve para medir la calidad de un diseño, la forma normal que cumple cada tabla es un indicador que se usa para evitar redundancias de datos.

## 2.12. Test de repaso

**1. ¿Cuál es el modelo que más se aproxima a la visión del usuario?**

- a) El modelo conceptual
- b) El modelo lógico
- c) El modelo físico
- d) El lenguaje SQL

**2. Una relación reflexiva es una entidad de grado**

- a) 0 b) 1 c) 2 d) 3

**3. La participación de una entidad en la relación es:**

- a) El máximo de ocurrencias que pueden aparecer en la relación
- b) El mínimo de ocurrencias que pueden aparecer en la relación
- c) El máximo y mínimo de ocurrencias que pueden aparecer en la relación
- d) El máximo y mínimo de ocurrencias de la entidad

**4. A qué participaciones corresponde una cardinalidad 1:N**

- a) (0,1) y (1,1)
- b) (1,n) y (0,n)
- c) (1,1) y (1,n)
- d) (0,1) y (n,n)

**5. Si un empleado puede trabajar en múltiples proyectos**

- a) Trabajar es 1:1
- b) Trabajar es N:N
- c) Trabajar es N:M
- d) Trabajar es 1:N

**6. Un atributo de relación es**

- a) Consecuencia de la relación
- b) Consecuencia de una de las entidades
- c) De las dos entidades
- d) Son atributos compuestos

**7. La dependencia de identificación**

- a) Incluye la dependencia de existencia
- b) No incluye la dependencia de existencia
- c) No se aplica a entidades débiles
- d) Sólo es posible cuando hay dos entidades fuertes

**8. Una especialización inclusiva es aquella que**

- a) Puede materializarse en más de una subclase
- b) Puede materializarse en solo una clase
- c) Puede no materializarse en alguna clase
- d) Tiene que materializarse en una clase

**9. La transformación de una relación con cardinalidad 1-N al modelo relacional**

- a) Genera una tabla para la relación
- b) Se incorpora una clave a la entidad 1
- c) Se incorpora una clave a la entidad N
- d) No se incorpora clave

**10. Con la normalización:**

- a) Se refina el modelo conceptual
- b) Se refina el modelo lógico
- c) Se refina el modelo físico
- d) No sirve para nada

Soluciones: 1.a,2.b,3.c,4.c,5.c,6.a,7.a,8.a,9.c,10.b

## 2.13. Comprueba tu aprendizaje

1. Nombra los distintos tipos de relaciones que puede haber atendiendo a su grado.
  2. Explica para qué sirve cada uno de los modelos expuestos en el tema para el diseño de una base de datos.
  3. Pon un ejemplo de cada uno de los tipos de cardinalidades.
  4. ¿Qué son las relaciones reflexivas?
  5. ¿Qué diferencia hay entre ocurrencia de entidad y entidad?
  6. ¿Cuándo una entidad es débil? ¿Y cuándo lo es una relación?
  7. ¿Qué significa que una entidad fuerte tenga una relación dependiente en existencia de otra entidad débil? Pon un ejemplo.
  8. ¿Cuándo dos entidades tienen dependencia de identificación?
  9. ¿Qué elementos incorpora el modelo entidad-relación extendido?
  10. Define los siguientes conceptos:
    - Atributo Clave
    - Atributo Derivado
    - Superclave
    - Clave Candidata
    - Dependencia Funcional
- Dependencia Funcional Completa
  - Dependencia Funcional Transitiva
11. ¿Qué diferencia hay entre una especialización total y otra parcial?
  12. ¿Qué diferencia hay entre una especialización exclusiva y otra inclusiva?
  13. Crea una lista con los pasos que hay que dar para pasar un diagrama entidad relación al modelo relacional.
  14. ¿Qué es el Álgebra Relacional? ¿Para qué sirve?
  15. Comenta cual es la utilidad de las restricciones de integridad referencial en una base de datos.
  16. ¿Qué es el valor NULO?
  17. ¿Qué es una clave foránea?
  18. Pon un ejemplo de especialización y comenta 4 formas distintas de generar las tablas.
  19. ¿Cómo representa MySQL Workbench las relaciones? ¿Y las dependencias?
  20. Haz un cuadro resumen con cada uno de los elementos gráficos que puede haber en un diagrama entidad-relación.