

5ª edición

Fundamentos de Sistemas de Bases de Datos

www.librosite.net/elmasri

Ramez Elmasri
Shamkant B. Navathe

PEARSON
Addison
Wesley

Modelado de datos con el modelo Entidad-Relación (ER)

El modelado conceptual es una fase muy importante para diseñar correctamente una aplicación de base de datos. Por lo general, el término **aplicación de base de datos** se refiere a una base de datos concreta y a los programas asociados encargados de implementar las consultas y las actualizaciones de la base de datos. Por ejemplo, una aplicación de base de datos **BANCO** que realiza el seguimiento de las cuentas de los clientes debe incluir programas encargados de implementar las actualizaciones de la base de datos correspondientes a los depósitos y las retiradas de fondos de los clientes. Estos programas ofrecen a los usuarios finales de la aplicación interfaces gráficas de usuario (GUIs) compuestas por formularios y menús. Por tanto, parte de la aplicación de base de datos requerirá el diseño, la implementación y la comprobación de esos programas de aplicación. Tradicionalmente, el diseño y la comprobación de los **programas de aplicación** se han considerado más como parte del dominio de la ingeniería de software que del dominio de las bases de datos. Como las metodologías de diseño de bases de datos incluyen cada vez más conceptos que sirven para especificar las operaciones con los objetos de bases de datos, y como las metodologías de ingeniería de software especifican la estructura de las bases de datos que los programas utilizarán y a las que accederán, es evidente que estas actividades están estrechamente relacionadas. En la Sección 3.8 explicaremos brevemente algunos de los conceptos que sirven para especificar las operaciones con las bases de datos, y de nuevo los veremos cuando expliquemos la metodología de diseño de una base de datos con las aplicaciones de ejemplo del Capítulo 12.

En este capítulo seguiremos la metodología tradicional de concentrarse en las estructuras y las restricciones de la base de datos durante el diseño de esta última. Presentaremos los conceptos de modelado del **modelo Entidad-Relación (ER)**, que es un modelo de datos conceptual de alto nivel. Este modelo y sus variaciones se utilizan con frecuencia para el diseño conceptual de las aplicaciones de base de datos, y muchas herramientas de diseño emplean estos conceptos. Describimos los conceptos básicos de la estructura de datos y las restricciones del modelo ER, así como su uso en el diseño de esquemas conceptuales para las aplicaciones de base de datos. También presentamos la notación esquemática asociada con el modelo ER, conocida como **diagramas ER**.

Las metodologías de modelado de objetos, como el **Lenguaje de modelado universal (UML, Universal Modeling Language)**, son cada vez más populares en el diseño y la ingeniería del software. Estas metodologías van más allá del diseño de bases de datos a fin de especificar un diseño específico de los módulos software y sus interacciones mediante varios tipos de diagramas. Una parte importante de estas metodologías

(antiguamente conocidas como *diagramas de clase*¹) son parecidas a los diagramas ER. En los diagramas de clase se especifican *operaciones* sobre los objetos, además de especificar la estructura del esquema de la base de datos. Las operaciones se pueden utilizar para especificar los *requisitos funcionales* durante el diseño de la base de datos, como se explica en la Sección 3.1. La Sección 3.8 presenta algunos de los conceptos y notaciones UML para los diagramas de clase que están relacionados con el diseño de la base de datos, y los compara brevemente con la notación y los conceptos ER. En la Sección 4.6 y el Capítulo 12 se explican otros conceptos y notaciones UML.

Este capítulo está organizado de este modo: la Sección 3.1 explica el papel de los modelos de datos conceptuales de alto nivel en el diseño de las bases de datos. En la Sección 3.2 se exponen los requisitos de una aplicación de base de datos de ejemplo para ilustrar el uso de los conceptos del modelo ER. Esta base de datos de ejemplo también se utiliza en capítulos posteriores. En la Sección 3.3 se presentan los conceptos de entidades y atributos, y se va introduciendo gradualmente la técnica diagramática para visualizar un esquema ER. En la Sección 3.4 se introducen los conceptos de las relaciones binarias y sus roles y restricciones estructurales. La Sección 3.5 introduce los tipos de entidad débiles. La Sección 3.6 muestra cómo se refina el diseño de un esquema para incluir las relaciones. La Sección 3.7 repasa la notación de los diagramas ER, resume los problemas que pueden surgir en el diseño del esquema, y explica cómo elegir los nombres de las estructuras del esquema de la base de datos. La Sección 3.8 introduce algunos conceptos diagramáticos de clase UML, los compara con los conceptos de modelo ER y los aplica a la misma base de datos de ejemplo. La Sección 3.9 explica los tipos más complejos de relaciones, mientras que la Sección 3.10 resume el capítulo.

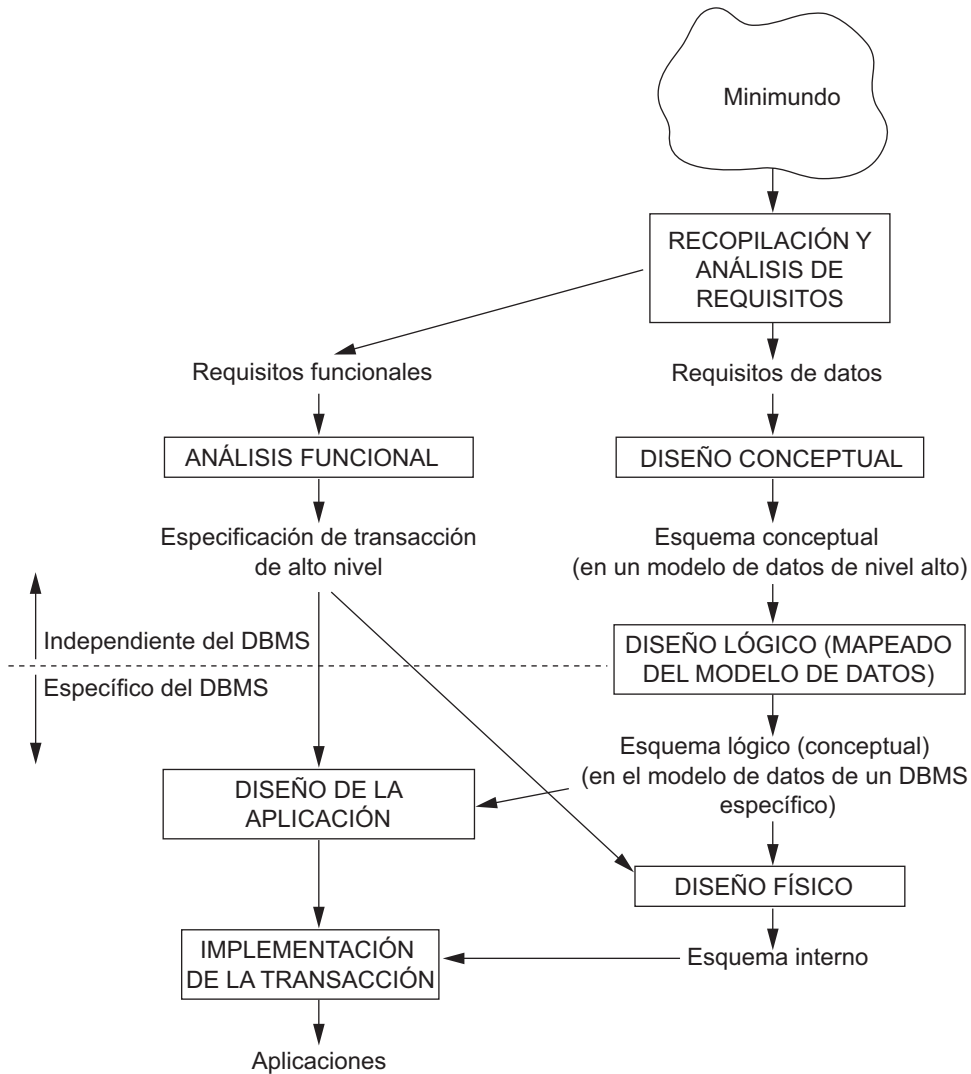
El material de las Secciones 3.8 y 3.9 se puede excluir de un curso introductorio; si desea una explicación más detallada de los conceptos de modelado de datos y del diseño de bases de datos conceptual, debe pasar de la Sección 3.7 al Capítulo 4, donde se describen las extensiones del modelo ER que conducen al modelo ER mejorado (EER), el cual incluye conceptos como la especialización, la generalización, la herencia y los tipos de unión (categorías). En el Capítulo 4 también se incluyen algunos conceptos UML y notaciones adicionales.

3.1 Uso de modelos de datos conceptuales de alto nivel para el diseño de bases de datos

La Figura 3.1 muestra una descripción simplificada del proceso de diseño de una base de datos. El primer paso es la **recopilación de requisitos y el análisis**. Durante este paso, los diseñadores de bases de datos entrevistan a los potenciales usuarios de la base de datos para comprender y documentar sus **requisitos en cuanto a datos**. El resultado de este paso es un conjunto por escrito de los requisitos del usuario. Estos requisitos se deben plasmar en un formulario lo más detallado y completo posible. En paralelo al estudio de estos requisitos, resulta útil especificar los **requisitos funcionales** de la aplicación, que consisten en las **operaciones** (o **transacciones**) definidas por el usuario que se aplicarán a la base de datos, incluyendo las recuperaciones y las actualizaciones. En el diseño de software, es frecuente utilizar los *diagramas de flujo de datos*, *diagramas de secuencia*, *escenarios* y otras técnicas para especificar los requisitos funcionales. No vamos a explicar ninguna de estas técnicas; normalmente se explican en profundidad en los textos de ingeniería de software. En el Capítulo 12 ofreceremos una visión general de algunas de estas técnicas.

Una vez recopilados y analizados todos los requisitos, el siguiente paso es crear un **esquema conceptual** para la base de datos, mediante un modelo de datos conceptual de alto nivel. Este paso se denomina **diseño conceptual**. El esquema conceptual es una descripción concisa de los requisitos de datos por parte de los usuarios e incluye descripciones detalladas de los tipos de entidades, relaciones y restricciones; se expresan utilizando los conceptos proporcionados por el modelo de datos de alto nivel. Como estos conceptos no incluyen detalles de implementación, normalmente son más fáciles de entender y se pueden utilizar para comuni-

¹ Una **clase** se parece en muchos casos a un tipo de entidad.

Figura 3.1. Diagrama simplificado para ilustrar las fases principales del diseño de una base de datos.

car con usuarios no técnicos. El esquema conceptual de alto nivel también se puede utilizar como referencia para garantizar que se han reunido todos los requisitos de datos de los usuarios y que esos requisitos no entran en conflicto. Esta metodología permite a los diseñadores de bases de datos concentrarse en especificar las propiedades de los datos, sin tener que preocuparse por los detalles del almacenamiento. En consecuencia, es más fácil para ellos crear un buen diseño conceptual de bases de datos.

Durante o después del diseño del esquema conceptual, se pueden utilizar las operaciones básicas del modelo de datos para especificar las operaciones de usuario de alto nivel identificadas durante el análisis funcional. Esto también sirve para confirmar que el esquema conceptual satisface todos los requisitos funcionales identificados. Es posible realizar modificaciones en el esquema conceptual si con el esquema inicial no se pueden especificar algunos de los requisitos funcionales.

El siguiente paso del diseño de una base de datos es la implementación real de la misma mediante un DBMS comercial. La mayoría de los DBMSs comerciales actuales utilizan un modelo de datos de implementación

(como el modelo de base de datos relacional u objeto-relación), de modo que el esquema conceptual se transforma de modelo de datos de alto nivel en modelo de datos de implementación. Este paso se conoce como **diseño lógico** o **asignación de modelo de datos**; su resultado es un esquema de base de datos en el modelo de datos de implementación del DBMS.

El último paso es la fase de **diseño físico**, durante la cual se especifican las estructuras de almacenamiento interno, los índices, las rutas de acceso y la organización de los archivos para la base de datos. En paralelo a estas actividades, se diseñan e implementan los programas de aplicación como transacciones de bases de datos correspondientes a las especificaciones de transacción de alto nivel. En el Capítulo 12 explicaremos más en profundidad el proceso de diseño de una base de datos.

En este capítulo sólo presentamos los conceptos básicos del modelo ER para el diseño del esquema conceptual. Los conceptos adicionales sobre modelado se explican en el Capítulo 4, donde se introduce el modelo EER.

3.2 Un ejemplo de aplicación de base de datos

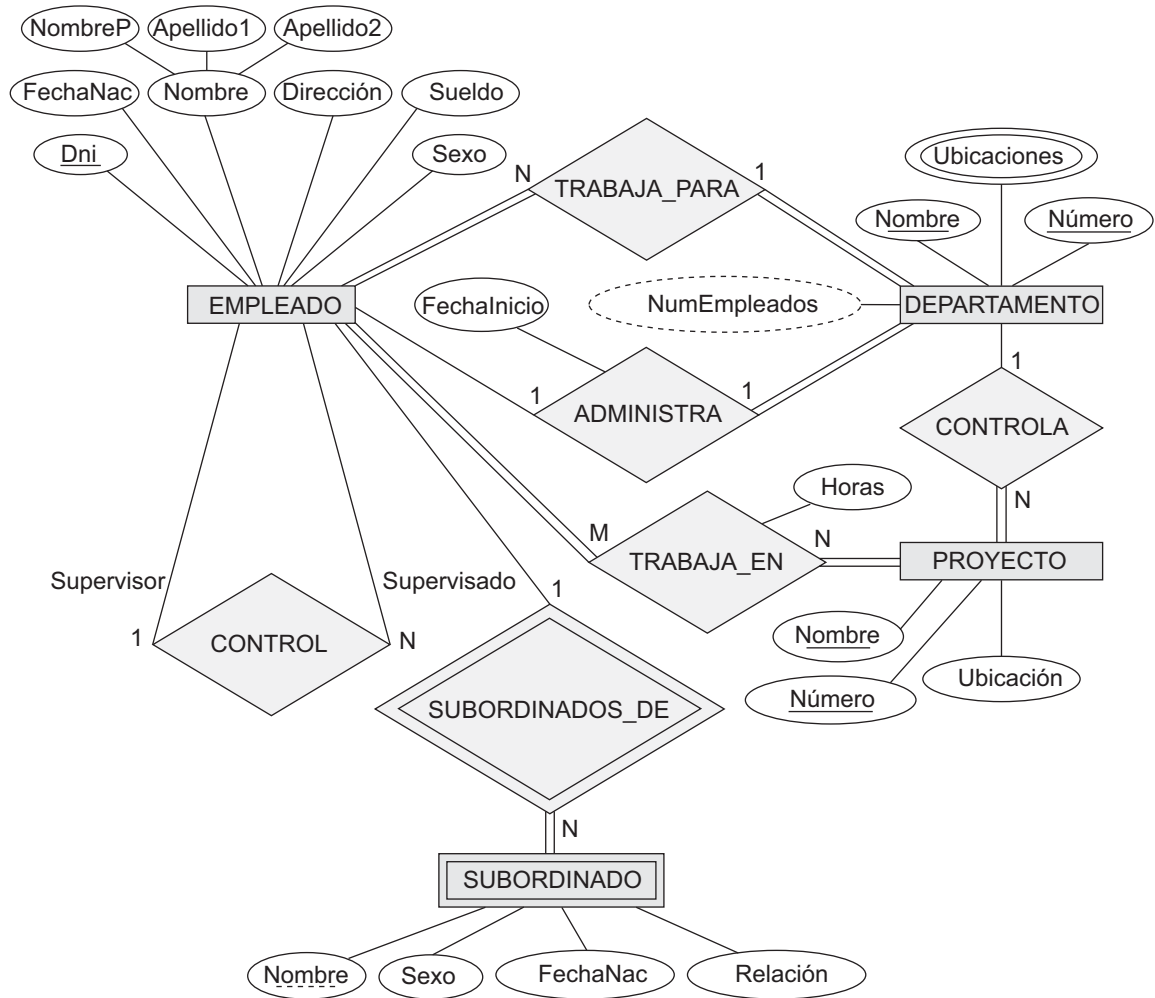
En esta sección se describe un ejemplo de aplicación de base de datos, denominada EMPRESA, que sirve para ilustrar los conceptos del modelo ER básico y su uso en el diseño del esquema. En primer lugar se enumeran los requisitos de datos para la base de datos, y después se crea su esquema conceptual paso a paso tras introducir los conceptos de modelado del modelo ER. La base de datos EMPRESA sirve como seguimiento de los empleados, los departamentos y los proyectos de una empresa. Suponga que después de la fase de recopilación de requisitos y análisis, los diseñadores de la base de datos proporcionan la siguiente descripción del *minimundo* (la parte de la empresa que se va a representar en la base de datos):

- La empresa está organizada en departamentos. Cada uno tiene un nombre único, un número único y un empleado concreto que lo administra. Se realizará un seguimiento de la fecha en que ese empleado empezó a administrar el departamento. Un departamento puede tener varias ubicaciones.
- Un departamento controla una cierta cantidad de proyectos, cada uno de los cuales tiene un nombre único, un número único y una sola ubicación.
- Almacenaremos el nombre, el documento nacional de identidad,² la dirección, el sueldo, el sexo y la fecha de nacimiento de cada empleado. Un empleado está asignado a un departamento, pero puede trabajar en varios proyectos, que no están controlados necesariamente por el mismo departamento. Se hará un seguimiento del número de horas por semana que un empleado trabaja en cada proyecto. También se realizará el seguimiento del supervisor directo de cada empleado.
- También se desea realizar un seguimiento de las personas a cargo de cada empleado por el tema de los seguros. Por cada persona a cargo o subordinado, se registrará su nombre de pila, sexo, fecha de nacimiento y relación con el empleado.

La Figura 3.2 muestra cómo se puede visualizar el esquema de esta aplicación de base de datos mediante la notación gráfica conocida como **diagramas ER**. Esta figura se explicará gradualmente a medida que se vayan presentando los conceptos del modelo ER. Describiremos el proceso por pasos para deducir este esquema a partir de los requisitos indicados (y de la explicación de la notación diagramática ER) a medida que vayamos introduciendo los conceptos del modelo ER.

² En Estados Unidos se utiliza el número de la seguridad social, que es un identificador de nueve dígitos único asignado a cada persona, para hacer un seguimiento de su empleo, sus beneficios y sus impuestos. En el resto de países hay esquemas de identificación parecidos, como, por ejemplo, el número del DNI (Documento Nacional de Identidad) español.

Figura 3.2. Diagrama de un esquema ER para la base de datos EMPRESA. La notación diagramática se introduce gradualmente a lo largo de este capítulo.



3.3 Tipos de entidad, conjuntos de entidades, atributos y claves

El modelo ER describe los datos como entidades, relaciones y atributos. En la Sección 3.3.1 introducimos los conceptos de entidades y sus atributos. En la Sección 3.3.2 explicamos los tipos de entidad y los atributos clave. Después, ya en la Sección 3.3.3, concretamos el diseño conceptual inicial de los tipos de entidad para la base de datos EMPRESA. Las relaciones se describen en la Sección 3.4.

3.3.1 Entidades y atributos

Entidades y sus atributos. El objeto básico representado por el modelo ER es una **entidad**, que es una *cosa* del mundo real con una existencia independiente. Una entidad puede ser un objeto con una existencia física (por ejemplo, una persona en particular, un coche, una casa o un empleado) o puede ser un objeto con

una existencia conceptual (por ejemplo, una empresa, un trabajo o un curso universitario). Cada entidad tiene **atributos** (propiedades particulares que la describen). Por ejemplo, una entidad EMPLEADO se puede describir mediante el nombre, la edad, la dirección, el sueldo y el trabajo que desempeña. Una entidad en particular tendrá un valor para cada uno de sus atributos. Los valores de los atributos que describen cada entidad se convierten en la parte principal de los datos almacenados en la base de datos.

La Figura 3.3 muestra dos entidades y los valores de sus atributos. La entidad EMPLEADO *e1* tiene cuatro atributos: Nombre, Dirección, Edad y TlfCasa; sus valores son ‘José Pérez’, ‘Ribera del Sena, 915. Getafe, Madrid 28903’, ‘55’ y ‘91-123-4567’, respectivamente. La entidad EMPRESA *c1* tiene tres atributos: Nombre, SedeCentral y Presidente; sus valores son ‘Sunco Oil’, ‘Madrid’ y ‘José Pérez’, respectivamente.

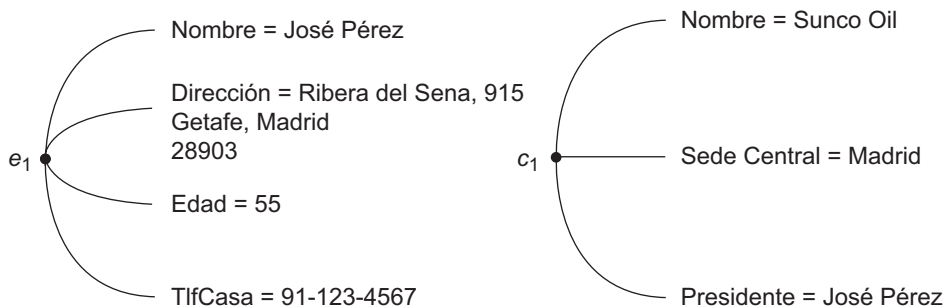
En el modelo ER se dan varios tipos de atributos: *simple* frente a *compuesto*, *monovalor* frente a *multivalor*, y *almacenado* frente a *derivado*. En primer lugar, definimos estos tipos de atributos e ilustramos su uso mediante ejemplos. Después, introducimos el concepto de *valor NULL* (nulo) para un atributo.

Atributos compuestos frente a atributos simples (atómicos). Los **atributos compuestos** se pueden dividir en subpartes más pequeñas, que representan atributos más básicos con significados independientes. Por ejemplo, el atributo Dirección de la entidad EMPLEADO de la Figura 3.3 se puede subdividir en DirCalle, Ciudad, Provincia y CP,³ con los valores ‘Ribera del Sena, 915’, ‘Getafe’, ‘Madrid’ y ‘28903’. Los atributos que no son divisibles se denominan **atributos simples** o **atómicos**. Los atributos compuestos pueden formar una jerarquía. Por ejemplo, DirCalle se puede subdividir en tres atributos simples: Número, Calle y NumApto, como se muestra en la Figura 3.4. El valor de un atributo compuesto es la concatenación de los valores de sus atributos simples.

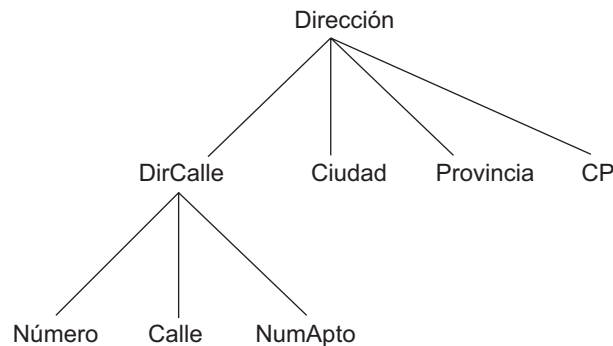
Los atributos son útiles para modelar situaciones en las que un usuario se refiere a veces al atributo compuesto como una unidad, pero otras veces se refiere específicamente a sus componentes. Si se hace referencia al atributo compuesto como un todo, no hay necesidad de subdividirlo en atributos componentes. Por ejemplo, si no hay necesidad de referirse a los componentes individuales de una dirección (código postal, calle, etcétera), entonces la dirección entera se puede designar como un atributo simple.

Atributos monovalor y multivalor. La mayoría de los atributos tienen un solo valor para una entidad en particular; dichos atributos reciben el nombre de **monovalor** o **de un solo valor**. Por ejemplo, Edad es un atributo monovalor de una persona. En algunos casos, un atributo puede tener un conjunto de valores para la misma entidad (por ejemplo, un atributo Colores para un coche, o un atributo Licenciaturas para una persona). Los coches con un solo color tienen un solo valor, mientras que los coches de dos tonos tienen dos valores de color. De forma parecida, puede que una persona no tenga ninguna licenciatura, otra puede que tenga una, y una tercera persona puede que tenga dos o más; por consiguiente, diferentes personas pueden

Figura 3.3. Dos entidades, EMPLEADO *e1* y EMPRESA *c1*, y sus atributos.



³ CP es la abreviatura que se utiliza en España para el código postal de cinco dígitos.

Figura 3.4. Una jerarquía de atributos compuestos.

tener una *cantidad de valores* diferente para el atributo *Licenciaturas*. Dichos atributos se denominan **multi-valor**. Un atributo multivalor puede tener límites superior e inferior para restringir el número de valores permitidos para cada entidad individual. Por ejemplo, el atributo *Colores* de un coche puede tener entre uno y tres valores, si asumimos que un coche puede tener tres colores a lo sumo.

Atributos almacenados y derivados. En algunos casos, dos (o más) valores de atributo están relacionados (por ejemplo, los atributos *Edad* y *FechaNac* de una persona). Para una entidad de persona en particular, el valor de *Edad* puede determinarse a partir de la fecha actual (el día de hoy) y el valor de *FechaNac* de esa persona. El atributo *Edad* se denomina entonces **atributo derivado** y se dice que se ha **derivado** del atributo *FechaNac*, que es el denominado **atributo almacenado**. Algunos valores de atributo se pueden derivar de *entidades relacionadas*; por ejemplo, un atributo *NumEmpleados* de una entidad *DEPARTAMENTO* puede derivarse contando el número de empleados relacionados con (o que trabajan para) ese departamento.

Valores NULL (nulos). En algunos casos, es posible que una entidad en particular no tenga un valor aplicable para un atributo. Por ejemplo, el atributo *NumApto* de una dirección sólo se aplica a las direcciones correspondientes a edificios de apartamentos, y no a otros tipos de residencias, como las casas unifamiliares. De forma parecida, un atributo *Licenciaturas* sólo se aplica a las personas con carrera universitaria. Para estas situaciones se ha creado un valor especial denominado NULL (nulo). La dicción de una casa unifamiliar tendría el valor NULL para su atributo *NumApto*, y una persona sin carrera universitaria tendría NULL para *Licenciaturas*. NULL también se puede utilizar cuando no se conoce el valor de un atributo para una entidad en particular (por ejemplo, si no conocemos el número de teléfono de la casa de ‘José Pérez’ en la Figura 3.3). El significado del tipo anterior de NULL no es aplicable, mientras que el significado del último es *desconocido*. La categoría *desconocido* se puede clasificar en dos casos. El primero se da cuando se sabe que existe el valor del atributo pero *no se encuentra*: por ejemplo, si el atributo *Altura* de una persona aparece como NULL. El segundo caso se da cuando es *no conocido* si existe el valor del atributo: por ejemplo, si el atributo *TlfCasa* de una persona es NULL.

Atributos complejos. Los atributos compuestos y multivalor se pueden anidar arbitrariamente. Podemos representar el anidamiento arbitrario agrupando componentes de un atributo compuesto entre paréntesis () y separando los componentes con comas, y mostrando los atributos multivalor entre llaves {}. Dichos atributos se denominan **atributos complejos**. Por ejemplo, si una persona puede tener más de una residencia y cada residencia puede tener una sola dirección y varios teléfonos, el atributo *TlfDir* de una persona se puede especificar como en la Figura 3.5.⁴ Los dos atributos, *Tlf* y *Dir*, son compuestos.

⁴ Los que están familiarizados con XML verán que los atributos complejos son parecidos a los elementos complejos de XML (consulte el Capítulo 27).

Figura 3.5. Un atributo complejo: TlfDir.

{TlfDir({Tlf(CodÁrea,NumTlf)},Dir(DirCalle(Número,Calle,NumApto),Ciudad,Provincia,CP)))}

3.3.2 Tipos de entidades, conjuntos de entidades, claves y conjuntos de valores

Tipos de entidades y conjuntos de entidades. Una base de datos normalmente contiene grupos de entidades que son parecidas. Por ejemplo, una compañía que da trabajo a cientos de empleados puede querer almacenar información parecida relacionada con cada uno de ellos. Estas entidades de empleado comparten los mismos atributos, pero cada entidad tiene su(s) *propio(s) valor(es)* para cada atributo. Un **tipo de entidad** define una *colección* (o *conjunto*) de entidades que tienen los mismos atributos. La Figura 3.6 muestra dos tipos de entidades: EMPLEADO y EMPRESA, y una lista de atributos de cada una. También se ilustran unas cuantas entidades individuales de cada tipo, junto con los valores de sus atributos. La colección de todas las entidades de un tipo de entidad en particular de la base de datos en cualquier momento del tiempo se denomina conjunto de entidades; al conjunto de entidades normalmente se hace referencia utilizando el mismo nombre que para el tipo de entidad. Por ejemplo, EMPLEADO se refiere tanto al *tipo de entidad* como al conjunto actual de *todas las entidades de empleado* de la base de datos.

Un tipo de entidad se representa en los diagramas ER⁵ (véase la Figura 3.2) como un rectángulo con el nombre del tipo de entidad en su interior. Los nombres de los atributos se encierran en óvalos y están unidos a su tipo de entidad mediante líneas rectas. Los atributos compuestos están unidos a sus atributos componente mediante líneas rectas. Los atributos multivalor se muestran en óvalos dobles. La Figura 3.7(a) muestra un tipo de entidad COCHE en esta notación.

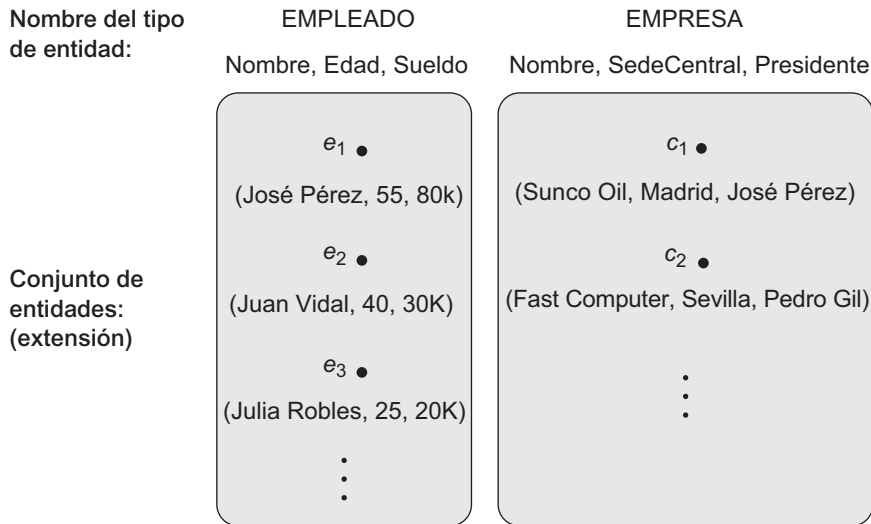
Un tipo de entidad describe el **esquema** o la **intención** de un *conjunto de entidades* que comparten la misma estructura. La colección de entidades de un tipo de entidad en particular está agrupada en un conjunto de entidades, que también se denomina **extensión** del tipo de entidad.

Atributos clave de un tipo de entidad. Una restricción importante de las entidades de un tipo de entidad es la **clave** o **restricción de unicidad** de los atributos. Un tipo de entidad normalmente tiene un atributo cuyos valores son distintos para cada entidad individual del conjunto de entidades. Dicho atributo se denomina **atributo clave**, y sus valores se pueden utilizar para identificar cada entidad sin lugar a dudas. Por ejemplo, en la Figura 3.6 el atributo Nombre es una clave del tipo de entidad EMPRESA porque no está permitido que dos empresas tengan el mismo nombre. Para el tipo de entidad PERSONA, un atributo clave típico es DNI. En ocasiones, una clave está formada por varios atributos juntos, lo que da a entender que la *combinación* de los valores de atributo debe ser distinta para cada entidad. Si un conjunto de atributos posee esta propiedad, la forma correcta de representar esto en el modelo ER que aquí describimos es definir un *atributo compuesto* y designarlo como atributo clave del tipo de entidad. Una clave compuesta debe ser mínima; es decir, en el atributo compuesto se deben incluir todos los atributos componente para tener una propiedad de unicidad. En una clave no se deben incluir atributos superfluos. En la notación diagramática ER, cada atributo clave tiene su nombre **subrayado** dentro del óvalo, como muestra la Figura 3.7(a).

Especificar que un atributo es una clave de un tipo de entidad significa que debe mantenerse la propiedad de unicidad anterior para *cada conjunto de entidades* del tipo de entidad. Por tanto, es una restricción que prohíbe que dos entidades tengan el mismo valor para el atributo clave al mismo tiempo. No es la propiedad de una extensión en particular; en cambio, es una restricción de *todas las extensiones* del tipo de entidad. Esta restricción clave (y otras restricciones que veremos más tarde) se derivan de las restricciones del minimundo representado por la base de datos.

5. Para los diagramas ER utilizamos una notación cercana a la notación original (Chen 1976). Se utilizan muchas otras notaciones; cuando hablemos de los diagramas de clase UML en este capítulo ofreceremos algunas de ellas, así como en el Apéndice A.

Figura 3.6. Dos tipos de entidades, EMPLEADO y EMPRESA, y algunas entidades miembro de cada una de ellas.



Algunos tipos de entidad tienen *más de un atributo clave*. Por ejemplo, cada uno de los atributos IDVehículo y Matrícula del tipo de entidad COCHE (véase la Figura 3.7) es una clave por propio derecho. El atributo Matrícula es un ejemplo de clave compuesta formada en España por dos atributos componente simples, un Número y unas Letras, ninguno de los cuales es una clave por sí mismo. Un tipo de entidad también puede *carecer de clave*, en cuyo caso se denomina *tipo de entidad débil* (consulte la Sección 3.5).

Conjuntos de valores (dominios) de atributos. Cada atributo simple de un tipo de entidad está asociado con un **conjunto de valor** (o **dominio** de valores), que especifica el conjunto de los valores que se pueden asignar a ese atributo por cada entidad individual. En la Figura 3.6, si el rango de edades permitido para los empleados está entre 16 y 70, podemos especificar el conjunto de valores del atributo Edad de EMPLEADO como un conjunto de números enteros entre 16 y 70. De forma parecida, podemos especificar el conjunto de valores para el atributo Nombre como un conjunto de cadenas de caracteres alfabéticos separados por espacios en blanco, etcétera. Los conjuntos de valores no se muestran en los diagramas ER; normalmente se especifican mediante los **tipos de datos** básicos disponibles en la mayoría de los lenguajes de programación, como entero, cadena, booleano, flotante, tipo enumerado, subrango, etcétera. También se emplean otros tipos de datos adicionales para representar la fecha, la hora y otros conceptos.

Matemáticamente, un atributo A de un tipo de entidad E cuyo conjunto de valores es V se puede definir como una **función** de E al conjunto potencia⁶ $P(V)$ de V :

$$A : E \rightarrow P(V)$$

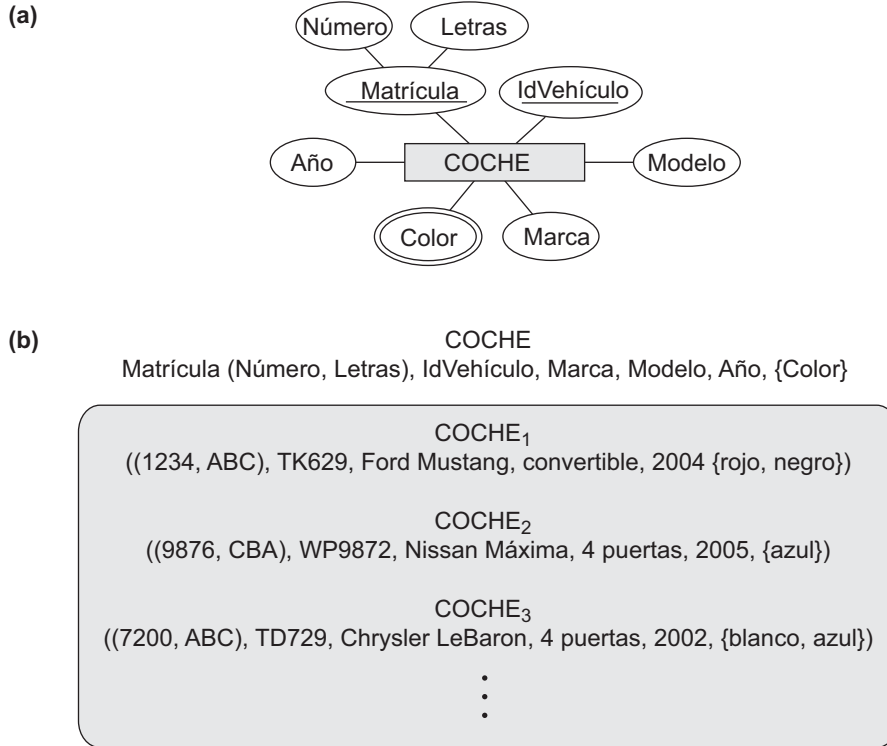
Al valor del atributo A para la entidad e nos referimos como $A(e)$. La definición anterior abarca tanto los atributos de un solo valor como los multivalor, así como los nulos. Un valor NULL queda representado por el *conjunto vacío*. Para los atributos de un solo valor, $A(e)$ está restringido a ser un conjunto sencillo para cada entidad e de E , mientras no haya una restricción en los atributos multivalor.⁷ Para un atributo compuesto A , el conjunto de valores V es el producto cartesiano de $P(V_1), P(V_2), \dots, P(V_n)$, donde V_1, V_2, \dots, V_n son los conjuntos de valores de los atributos simples que forman A :

$$V = P(V_1) \times P(V_2) \times \dots \times P(V_n)$$

6. El **conjunto potencia** $P(V)$ de un conjunto V es el conjunto de todos los subconjuntos de V .

7. Un conjunto **sencillo** es un conjunto con un solo elemento (valor).

Figura 3.7. El tipo de entidad COCHE con dos atributos clave, Matrícula e IDVehículo. (a) Notación de diagrama ER. (b) Conjunto de entidades con tres entidades.



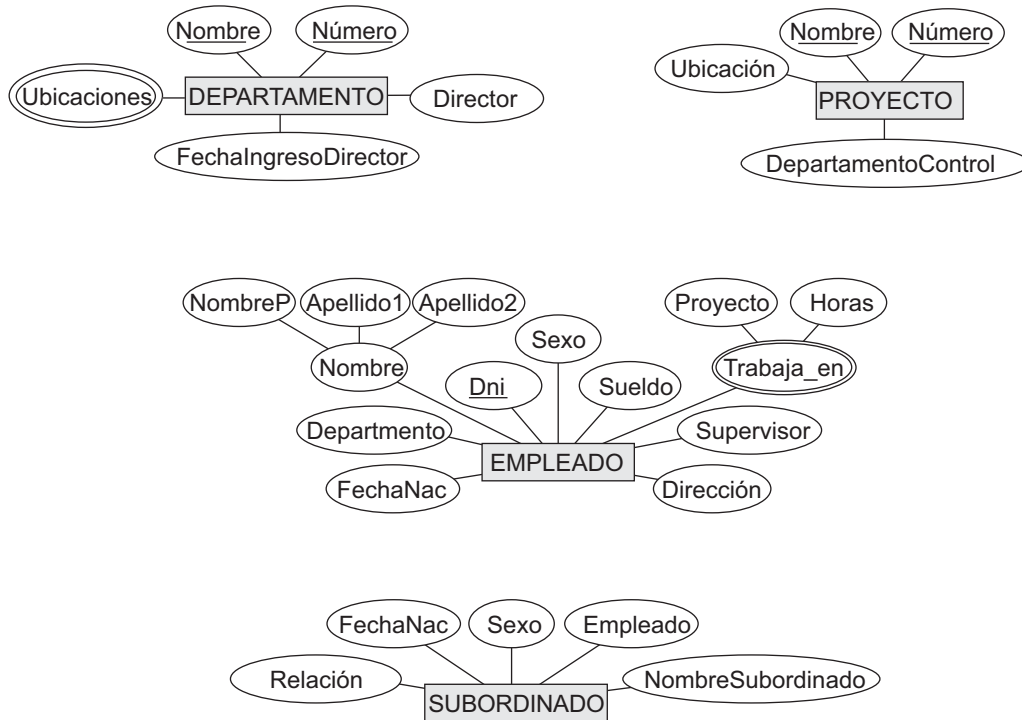
El conjunto de valores proporciona todos los valores posibles. Normalmente, en la base de datos sólo existen unos cuantos de estos valores. Estos valores representan los datos del estado del minimundo. Corresponden a los datos tal y como existen en el minimundo.

3.3.3 Diseño conceptual inicial de la base de datos EMPRESA

Ahora podemos definir los tipos de entidad para la base de datos EMPRESA, en base a los requisitos descritos en la Sección 3.2. Después de definir aquí varios tipos de entidad y sus atributos, refinaremos nuestro diseño en la Sección 3.4 después de introducir el concepto de una relación. De acuerdo con los requisitos enumerados en la Sección 3.2, podemos identificar cuatro tipos de entidades (uno por cada uno de los cuatro elementos de la especificación [véase la Figura 3.8]):

1. Tipo de entidad DEPARTAMENTO con los atributos NombreDpto, NúmeroDpto, Ubicaciones, Director y FechaIngresoDirector. Ubicaciones es el único atributo multivalor. Podemos especificar que Nombre y NúmeroDpto son atributos clave (separados) porque cada uno se especificó como único.
2. Tipo de entidad PROYECTO con los atributos Nombre, Número, Ubicación y DepartamentoControl. Tanto Nombre como Número son atributos clave (separados).
3. Tipo de entidad EMPLEADO con los atributos Nombre, Dni, Sexo, Dirección, Sueldo, FechaNac, Departamento y Supervisor. Nombre y Dirección pueden ser atributos compuestos; no obstante, esto no se especificó en los requisitos. Debemos volver a los usuarios para ver si alguno de ellos se referirá a los componentes individuales de Nombre (NombrePila, PrimerApellido, SegundoApellido) o de Dirección.

Figura 3.8. Diseño preliminar de los tipos de entidad para la base de datos EMPRESA. Algunos de los atributos mostrados se redefinirán como relaciones.



4. Tipo de entidad SUBORDINADO con los atributos Empleado, NombreSubordinado, Sexo, FechaNac y Relación (con el empleado).

Hasta ahora, no hemos representado el hecho de que un empleado pueda trabajar en varios proyectos, ni tampoco el número de horas por semana trabajadas por un empleado en cada proyecto. Esta característica se muestra como parte del tercer requisito en la Sección 3.2, y se puede representar mediante un atributo compuesto multivalor de EMPLEADO denominado TrabajaEn con los componentes Proyecto y Horas. De forma alternativa, puede representarse como un atributo compuesto multivalor de PROYECTO denominado Trabajadores con los componentes Empleado y Horas. En la Figura 3.8 elegimos la primera alternativa, que muestra cada uno de los tipos de entidad que acabamos de describir. El atributo Nombre de EMPLEADO se muestra como un atributo compuesto, probablemente después de consultar con los usuarios.

3.4 Tipos de relaciones, conjuntos de relaciones, roles y restricciones estructurales

En la Figura 3.8 hay varias *relaciones implícitas* entre los distintos tipos de entidades. De hecho, en cuanto un atributo de un tipo de entidad se refiere a otro tipo de entidad, decimos que existen algunas relaciones. Por ejemplo, el atributo Director de DEPARTAMENTO se refiere a un empleado que dirige el departamento; el atributo DepartamentoControl de PROYECTO se refiere al departamento que controla el proyecto; el atributo Supervisor de EMPLEADO se refiere a otro empleado (el que supervisa a este empleado); el atributo

Departamento de EMPLEADO se refiere al departamento para el que trabaja el empleado; etcétera. En el modelo ER, estas referencias no deben representarse como atributos, sino como **relaciones**, que explicaremos en esta sección. El esquema de la base de datos EMPRESA se refinará en la Sección 3.6 para representar explícitamente las relaciones. En el diseño inicial de los tipos de entidades, las relaciones se capturan normalmente en forma de atributos. Al depurar el diseño, estos atributos se convierten en relaciones entre los tipos de entidades.

Esta sección está organizada de este modo: la Sección 3.4.1 introduce los conceptos de tipos de relaciones, conjuntos de relaciones e instancias (también conocidas como ejemplares u ocurrencias) de relaciones. Además, en la Sección 3.4.2 definimos los conceptos de grado de relación, nombres de rol y relaciones recursivas. Después, ya en la Sección 3.4.3, explicamos las restricciones estructurales en las relaciones (por ejemplo, como las razones de cardinalidad y las dependencias existentes). La Sección 3.4.4 muestra cómo los tipos de relaciones también pueden tener atributos.

3.4.1 Tipos, conjuntos e instancias de relaciones

Un **tipo de relación** R entre n tipos de entidades E_1, E_2, \dots, E_n define un conjunto de asociaciones (o un **conjunto de relaciones**) entre las entidades de esos tipos de entidades. Como en el caso de los tipos de entidades y los conjuntos de entidades, normalmente se hace referencia a un tipo de relación y su correspondiente conjunto de relaciones con el *mismo nombre*, R . Matemáticamente, el conjunto de relaciones R es un conjunto de instancias de relación r_i , donde cada r_i asocia n entidades individuales (e_1, e_2, \dots, e_n) , y cada entidad e_j de r_i es un miembro del tipo de entidad E_j , $1 \leq j \leq n$. Por tanto, un tipo de relación es una relación matemática en E_1, E_2, \dots, E_n ; de forma alternativa, se puede definir como un subconjunto del producto cartesiano $E_1 \times E_2 \times \dots \times E_n$. Se dice que cada uno de los tipos de entidad E_1, E_2, \dots, E_n participa en el tipo de relación R ; de forma parecida, cada una de las entidades individuales e_1, e_2, \dots, e_n se dice que **participa** en la instancia de relación $r_i = (e_1, e_2, \dots, e_n)$.

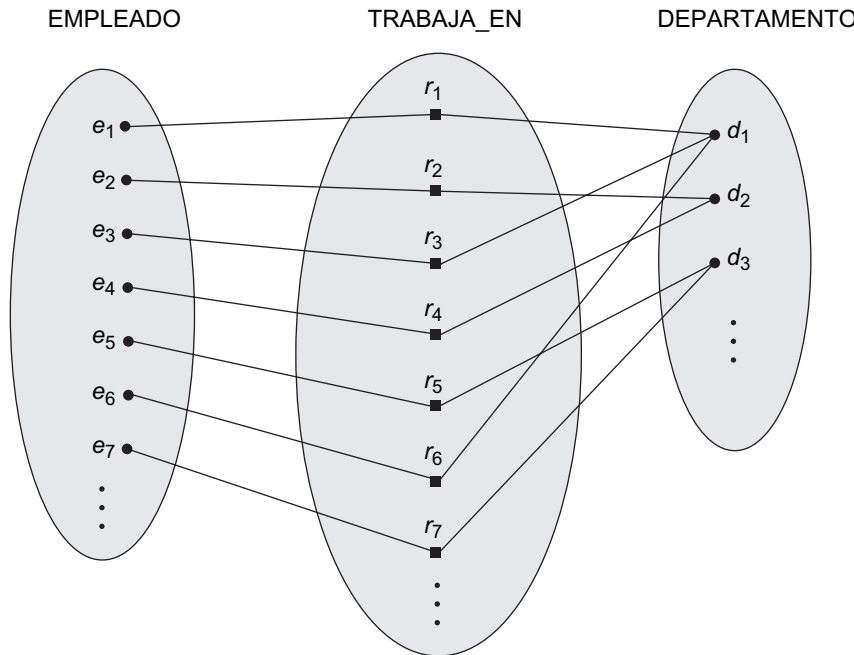
Informalmente, cada instancia de relación r_i en R es una asociación de entidades, donde la asociación incluye exactamente una entidad de cada tipo de entidad participante. Cada una de dichas instancias de relación r_i representa el hecho de que las entidades que participan en r_i están relacionadas de alguna forma en la situación correspondiente del minimundo. Por ejemplo, considere un tipo de relación TRABAJA_PARA asociado con una entidad EMPLEADO y una entidad DEPARTAMENTO. La Figura 3.9 ilustra este ejemplo, donde cada instancia de relación r_i se muestra conectada a las entidades EMPLEADO y DEPARTAMENTO que participan en r_i . En el minimundo representado por la Figura 3.9, los empleados e_1 , e_3 y e_6 trabajan para el departamento d_1 ; los empleados e_2 y e_4 trabajan para el departamento d_2 ; y los empleados e_5 y e_7 trabajan para el departamento d_3 .

En los diagramas ER, los tipos de relaciones se muestran mediante rombos, conectados a su vez mediante líneas a los rectángulos que representan los tipos de entidad participantes. El nombre de la relación se muestra en el rombo (véase la Figura 3.2).

3.4.2 Grado de relación, nombres de rol y relaciones recursivas

Grado de un tipo de relación. El **grado** de un tipo de relación es el número de tipos de entidades participantes. Por tanto, la relación TRABAJA_PARA es de grado dos. Un tipo de relación de grado dos se denomina **binario**, y uno de grado tres, **ternario**. Un ejemplo de relación ternaria es SUMINISTRO, en la Figura 3.10, donde cada instancia de relación r_i asocia tres entidades (un proveedor s , un repuesto p y un proyecto j), siempre que s suministre un repuesto p al proyecto j . Las relaciones pueden ser generalmente de cualquier grado, pero las más comunes son las relaciones binarias. Las relaciones de grado más alto son, por lo general, más complejas que las relaciones binarias; las explicaremos más tarde, en la Sección 3.9.

Figura 3.9. Algunas instancias del conjunto de relación TRABAJA_PARA, que representa un tipo de relación TRABAJA_PARA entre EMPLEADO y DEPARTAMENTO.

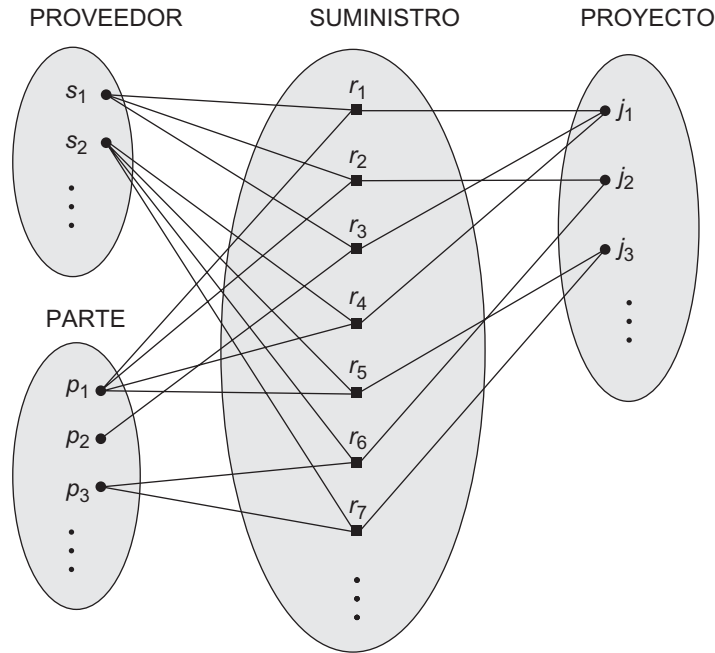
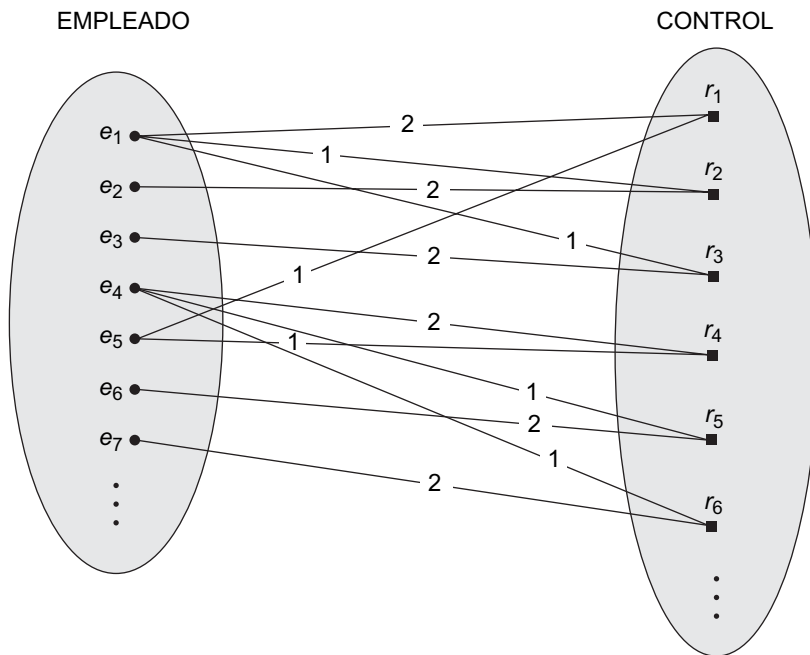


Relaciones y atributos. A veces es conveniente imaginar un tipo de relación en términos de atributos, como explicamos en la Sección 3.3.3. Considere el tipo de relación TRABAJA_PARA de la Figura 3.9. Uno puede pensar en un atributo denominado Departamento del tipo de entidad EMPLEADO donde el valor de Departamento por cada entidad EMPLEADO es (una referencia a) la entidad DEPARTAMENTO para la que ese empleado trabaja. Por tanto, el conjunto de valores para este atributo Departamento es el conjunto de *todas* las entidades DEPARTAMENTO, que es el conjunto de entidades DEPARTAMENTO. Es lo que hacíamos en la Figura 3.8 cuando especificábamos el diseño inicial del tipo de entidad EMPLEADO para la base de datos EMPRESA. No obstante, cuando pensamos en una relación binaria como si fuera un atributo, siempre tenemos dos opciones. En este ejemplo, la alternativa es pensar en un atributo Empleado multivalor del tipo de entidad DEPARTAMENTO cuyos valores por cada entidad DEPARTAMENTO es el conjunto de entidades EMPLEADO que trabajan para ese departamento. El conjunto de valores de este atributo Empleado es el conjunto potencia del conjunto de entidades EMPLEADO. Cualquiera de estos dos atributos (Departamento de EMPLEADO o Empleado de DEPARTAMENTO) puede representar el tipo de relación TRABAJA_PARA. Si se representan ambas, se restringen para ser mutuamente inversas.⁸

Nombres de rol y relaciones recursivas. Cada tipo de entidad que participa en un tipo de relación juega un papel o rol particular en la relación. El **nombre de rol** hace referencia al papel que una entidad participante del tipo de entidad juega en cada instancia de relación, y ayuda a explicar el significado de la relación. Por ejemplo, en el tipo de relación TRABAJA_PARA, EMPLEADO juega el papel de *empleado* o *trabajador* y DEPARTAMENTO juega el papel de *departamento* o *empleador*.

Los nombres de rol no son técnicamente necesarios en los tipos de relación donde todos los tipos de entidad participantes son distintos, puesto que cada nombre de tipo de entidad participante se puede utilizar como

⁸ Este concepto de representar los tipos de relación como atributos se utiliza en una clase de modelos de datos denominada **modelos de datos funcionales**. En las bases de datos de objetos (consulte el Capítulo 20), las relaciones se pueden representar mediante atributos de referencia, en una dirección o en ambas direcciones. En las bases de datos relacionales (consulte el Capítulo 5), las claves extrañas son un tipo de atributo de referencia que se utiliza para representar las relaciones.

Figura 3.10. Algunas instancias de relación en el conjunto de relaciones ternarias SUMINISTRO.**Figura 3.11.** Una relación recursiva **CONTROL** entre **EMPLEADO** en el papel de *supervisor* (1) y **EMPLEADO** en el papel de *subordinado* (2).

nombre de rol. No obstante, en algunos casos el *mismo* tipo de entidad participa más de una vez en un tipo de relación con *diferentes roles*. En esos casos, el nombre de rol es esencial para distinguir el significado de cada

participación. Dichos tipos de relaciones se denominan **relaciones recursivas**. La Figura 3.11 muestra un ejemplo. El tipo de relación CONTROL relaciona un empleado con un supervisor, donde las entidades empleado y supervisor son miembros del mismo tipo de entidad EMPLEADO. Por tanto, el tipo de entidad EMPLEADO *participa* dos veces en CONTROL: una en el papel de *supervisor* (o *jefe*), y otra en el papel de *supervisado* (o *subordinado*). Cada instancia de relación r_i en CONTROL asocia dos entidades de empleado, e_j y e_k , una de las cuales desempeña el papel de supervisor y la otra el papel de supervisado.

En la Figura 3.11, las líneas marcadas como '1' representan el papel de supervisor, y las marcadas con el '2' representan el papel de supervisado; por tanto, e_1 supervisa a e_2 y e_3 ; e_4 supervisa a e_6 y e_7 ; y e_5 supervisa a e_1 y e_4 . En este ejemplo, cada instancia de relación debe tener dos líneas, una marcada con el '1' (supervisión) y otra con el '2' (supervisado).

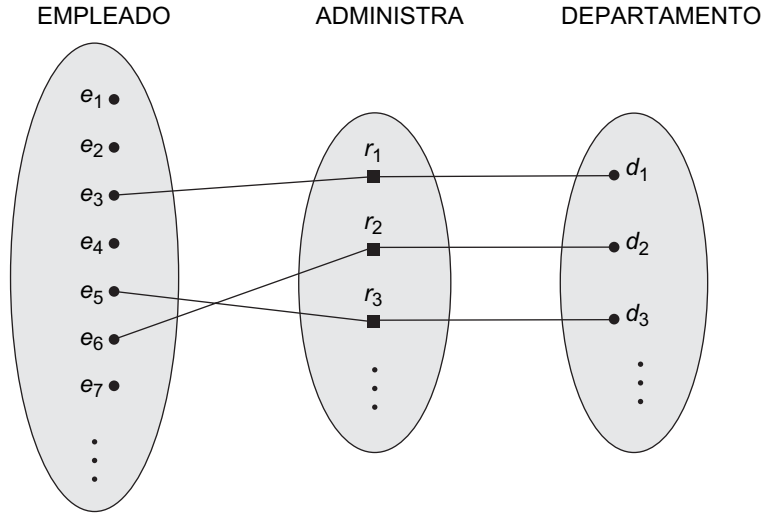
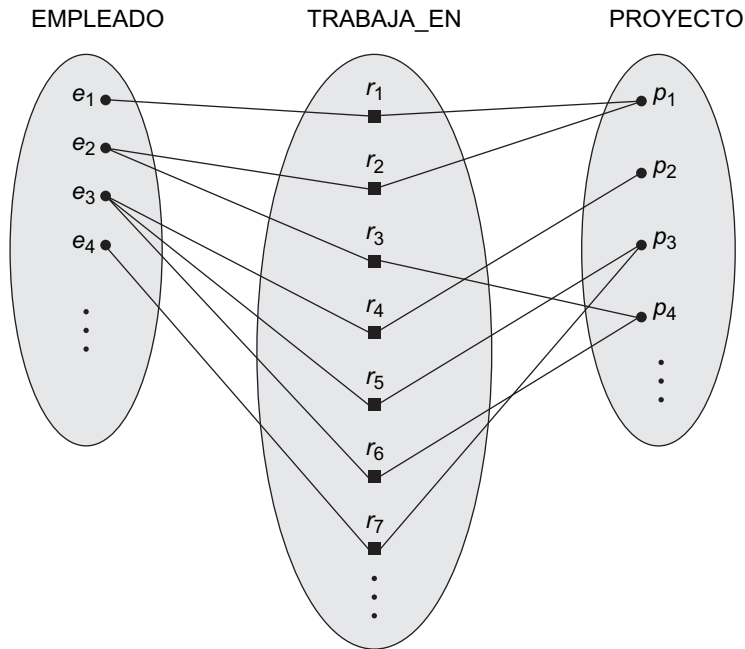
3.4.3 Restricciones en los tipos de relaciones

Los tipos de relaciones normalmente tienen ciertas restricciones que limitan las posibles combinaciones entre las entidades que pueden participar en el conjunto de relaciones correspondiente. Estas restricciones están determinadas por la situación del minimundo representado por las relaciones. Por ejemplo, en la Figura 3.9, si la empresa tiene por norma que cada empleado debe trabajar únicamente para un departamento, entonces tendríamos que describir esta restricción en el esquema. Podemos distinguir dos tipos principales de restricciones de relación: *razón de cardinalidad* y *participación*.

Razones de cardinalidad para las relaciones binarias. La **razón de cardinalidad** de una relación binaria especifica el número *máximo* de instancias de relación en las que una entidad puede participar. Por ejemplo, en el tipo de relación binaria TRABAJA_PARA, DEPARTAMENTO:EMPLEADO tiene una razón de cardinalidad de 1:N, que significa que cada departamento puede estar relacionado con (es decir, emplea a) cualquier cantidad de empleados,⁹ pero un empleado puede estar relacionado con (trabajar para) un solo departamento. Las posibles razones de cardinalidad para los tipos de relación binaria son 1:1, 1:N, N:1 y M:N. Un ejemplo de relación binaria 1:1 es ADMINISTRA (véase la Figura 3.12), que relaciona una entidad departamento con el empleado que dirige ese departamento. Esto representa las restricciones del minimundo, según las cuales, en cualquier momento del tiempo, un empleado puede dirigir un solo departamento y un departamento sólo puede tener un director. El tipo de relación TRABAJA_EN (véase la Figura 3.13) tiene una razón de cardinalidad de M:N, porque la norma del minimundo es que un empleado puede trabajar en varios proyectos, y un proyecto puede tener varios empleados. Las razones de cardinalidad de las relaciones binarias se representan en los diagramas ER mediante 1, M y N en los rombos (véase la Figura 3.2).

Restricciones de participación y dependencias de existencia. La **restricción de participación** especifica si la existencia de una entidad depende de si está relacionada con otra entidad a través de un tipo de relación. Esta restricción especifica el número *mínimo* de instancias de relación en las que puede participar cada entidad, y en ocasiones recibe el nombre de **restricción de cardinalidad mínima**. Hay dos tipos de restricciones de participación, total y parcial, que ilustramos con un ejemplo. Si una política de la empresa dice que *cada* empleado debe trabajar para un departamento, entonces una entidad de empleado sólo puede existir si participa en al menos una instancia de relación TRABAJA_PARA (véase la Figura 3.9). De este modo, la participación de EMPLEADO en TRABAJA_PARA se denomina **participación total**, es decir, cada entidad del *conjunto total* de entidades empleado debe estar relacionada con una entidad departamento a través de TRABAJA_PARA. La participación total también se conoce como **dependencia de existencia**. En la Figura 3.12 no esperamos que cada empleado dirija un departamento, de modo que la participación de EMPLEADO en el tipo de relación ADMINISTRA es parcial; esto significa que *algo* o *parte del conjunto* de entidades empleado está relacionado con alguna entidad departamento a través de ADMINISTRA, pero no necesariamente con todas. Nos referiremos a la razón de cardinalidad y a las restricciones de participación, en conjunto, como **restricciones estructurales** de un tipo de relación.

⁹ N significa cualquier número de entidades relacionadas (cero o más).

Figura 3.12. Una relación 1:1, ADMINISTRA.**Figura 3.13.** Una relación M:N, TRABAJA_EN.

En los diagramas ER, la participación total (o dependencia existente) se muestra como una *línea doble* que conecta el tipo de entidad participante con la relación, mientras que las participaciones parciales se representan mediante una *línea sencilla* (véase la Figura 3.2).

3.4.4 Atributos de los tipos de relación

Los tipos de relación también pueden tener atributos, parecidos a los de los tipos de entidad. Por ejemplo, para registrar el número de horas por semana que un empleado trabaja en un proyecto en particular, podemos

incluir un atributo Horas para el tipo de relación TRABAJA_EN de la Figura 3.13. Otro ejemplo es incluir la fecha en que el director empezó a dirigir un departamento, mediante un atributo Fechalnicio para el tipo de relación ADMINISTRA de la Figura 3.1.

Los atributos de los tipos de relación 1:1 o 1:N se pueden trasladar a uno de los tipos de entidad participantes. Por ejemplo, el atributo Fechalnicio para la relación ADMINISTRA puede ser un atributo de EMPLEADO o DEPARTAMENTO, aunque conceptualmente pertenece a ADMINISTRA. Esto se debe a que ADMINISTRA es una relación 1:1, por lo que cada entidad departamento o empleado participa *a lo sumo en una* instancia de relación. Por tanto, el valor del atributo Fechalnicio se puede determinar por separado, bien mediante la entidad departamento participante, bien mediante la entidad empleado (director) participante.

En el caso de un tipo de relación 1:N, un atributo de relación *sólo* se puede migrar al tipo de entidad que se encuentra en el lado N de la relación. En la Figura 3.9, por ejemplo, si la relación TRABAJA_PARA también tiene un atributo Fechalnicio que indica la fecha en que un empleado empezó a trabajar para un departamento, este atributo se puede incluir como un atributo de EMPLEADO. Esto se debe a que cada empleado trabaja sólo para un departamento y, por tanto, participa en un máximo de una instancia de relación en TRABAJA_PARA. En los tipos de relación 1:1 y 1:N, la decisión sobre dónde debe colocarse un atributo de relación (como un atributo de tipo de relación o como un atributo de un tipo de entidad participante) la determina subjetivamente el diseñador del esquema.

Para los tipos de relación M:N, algunos atributos pueden determinarse mediante la *combinación de entidades participantes* en una instancia de relación, no mediante una sola relación. Dichos atributos *deben especificarse como atributos de relación*. Un ejemplo de esto es el atributo Horas de la relación M:N TRABAJA_EN (véase la Figura 3.13); el número de horas que un empleado trabaja en un proyecto viene determinado por una combinación empleado-proyecto, y no separadamente por cualquiera de estas entidades.

3.5 Tipos de entidades débiles

Los tipos de entidad que no tienen atributos clave propios se denominan **tipos de entidad débiles**. En contraposición, los **tipos de entidad regulares** que tienen un atributo clave (que incluye todos los ejemplos que hemos explicado hasta ahora) se denominan **tipos de entidad fuertes**. Las entidades que pertenecen a un tipo de entidad débil se identifican como relacionadas con entidades específicas de otro tipo de entidad en combinación con uno de sus valores de atributo. Podemos llamar a este otro tipo de entidad **tipo de entidad identificado** o **propietario**¹⁰, y al tipo de relación que relaciona un tipo de entidad débil con su propietario lo podemos llamar **relación identificativa** del tipo de entidad débil.¹¹ Un tipo de entidad débil siempre tiene una *restricción de participación total* (dependencia de existencia) respecto a su relación identificativa, porque una entidad débil no puede identificarse sin una entidad propietaria. No obstante, no toda dependencia de existencia produce un tipo de entidad débil. Por ejemplo, la entidad PERMISO_CONducir no puede existir a menos que esté relacionada con una entidad PERSONA, aunque tiene su propia clave (NumPermiso) y, por tanto, no es una entidad débil.

Considere el tipo de entidad SUBORDINADO, relacionada con EMPLEADO, que sirve para llevar el control de las personas a cargo de cada empleado a través de una relación 1:N (véase la Figura 3.2). Los atributos de SUBORDINADO son Nombre (el nombre de pila del dependiente), FechaNac, Sexo y Relación (con el empleador). Dos personas a cargo de *dos empleados diferentes* pueden, por casualidad, tener los mismos valores para Nombre, FechaNac, Sexo y Relación, pero todavía seguirán siendo entidades distintas. Se identifican como entidades diferentes únicamente después de determinar la *entidad de empleado en particular* con la que

¹⁰ El tipo de entidad identificativa también se denomina a veces **tipo de entidad padre** o **tipo de entidad dominante**.

¹¹ El tipo de entidad débil también recibe a veces el nombre de **tipo de entidad hija** o **tipo de entidad subordinada**.

cada persona a cargo está relacionada. Cada entidad empleado posee entidades dependientes que están relacionadas con ella.

Un tipo de entidad débil normalmente tiene una **clave parcial**, que es el conjunto de atributos que pueden identificar sin lugar a dudas las entidades débiles que están *relacionadas con la misma entidad propietaria*.¹² En nuestro ejemplo, si asumimos que no puede haber dos dependientes del mismo empleado con el mismo nombre, el atributo Nombre de SUBORDINADO es la clave parcial. En el peor de los casos, la clave parcial será un atributo compuesto por todos los *atributos de la entidad débil*.

En los diagramas ER, tanto el tipo de la entidad débil como la relación identificativa, se distinguen rodeando sus cuadros y rombos mediante unas líneas dobles (véase la Figura 3.2). El atributo de clave parcial aparece subrayado con una línea discontinua o punteada.

Los tipos de entidades débiles se puede representar a veces como atributos complejos (compuestos, multivalor). En el ejemplo anterior, podríamos especificar un atributo multivalor Subordinados para EMPLEADO, que es un atributo compuesto por los atributos simples Nombre, FechaNac, Sexo y Relación. El diseñador de la base de datos toma la decisión del tipo de representación que hay que usar. Uno de los criterios que puede utilizar es elegir la representación del tipo de entidad débil si hay muchos atributos. Si la entidad débil participa independientemente en los tipos de relación de otra forma que su tipo de relación identificativa, entonces no debe modelarse como un atributo complejo.

En general, se puede definir cualquier cantidad de niveles de tipos de entidad débil; un tipo de entidad propietaria puede ser ella misma un tipo de entidad débil. Además, un tipo de entidad débil puede tener más de un tipo de entidad identificativa y un tipo de relación identificativa de grado superior a dos, como se ilustra en la Sección 3.9.

3.6 Perfeccionamiento del diseño ER para la base de datos EMPRESA

Ahora podemos refinar el diseño de la base de datos EMPRESA de la Figura 3.8 convirtiendo los atributos que representan relaciones en tipos de relaciones. La razón de cardinalidad y la restricción de participación de cada tipo de relación vienen determinadas por los requisitos enumerados en la Sección 3.2. Si no es posible determinar alguna razón de cardinalidad o dependencia a partir de los requisitos, habrá que consultar con los usuarios para determinar esas restricciones estructurales.

En nuestro ejemplo, especificaremos los siguientes tipos de relaciones:

- **ADMINISTRA**, un tipo de relación 1:1 entre EMPLEADO y DEPARTAMENTO. La participación de EMPLEADO es parcial, pero la de DEPARTAMENTO no queda clara a partir de los requisitos. Consultamos con los usuarios, que nos dicen que un departamento siempre debe tener un gerente, lo que implica una participación total.¹³ El atributo FechaInicio se asigna a este tipo de relación.
- **TRABAJA_PARA**, un tipo de relación 1:N entre DEPARTAMENTO y EMPLEADO. Ambas participaciones son totales.
- **CONTROLA**, un tipo de relación 1:N entre DEPARTAMENTO y PROYECTO. La participación de PROYECTO es total, mientras que la de DEPARTAMENTO se ha determinado como parcial, después de haber consultado con los usuarios, que indicaron que es posible que algunos departamentos no controlen proyecto alguno.

¹² La clave parcial a veces recibe el nombre de **discriminador**.

¹³ Las reglas del minimundo que determinan las restricciones se denominan a veces reglas empresariales, pues están determinadas por la *empresa* u organización que utilizará la base de datos.

- **CONTROL**, un tipo de relación 1:N entre EMPLEADO (en el papel de supervisor) y EMPLEADO (en el papel de supervisado). Como los usuarios han indicado que no todo empleado es un supervisor y no todo empleado tiene un supervisor, se determina que las dos participaciones son parciales.
- **TRABAJA_EN** que, después de que los usuarios hayan indicado que en un proyecto pueden trabajar varios empleados, se determina que es un tipo de relación M:N con el atributo Horas. Se determina que ambas participaciones son totales
- **SUBORDINADOS_DE**, un tipo de relación 1:N entre EMPLEADO y SUBORDINADO, que también es la relación identificativa del tipo de entidad débil SUBORDINADO. La participación de EMPLEADO es parcial, en tanto que la de SUBORDINADO es total.

Después de especificar los seis tipos de relación anteriores, eliminamos de los tipos de entidad de la Figura 3.8 todos los atributos que se han convertido en relaciones, entre los que se encuentran Director y FechaIngresoDirector de DEPARTAMENTO; DepartamentoControl de PROYECTO; Departamento, Supervisor y Trabaja_en de EMPLEADO; y Empleado de SUBORDINADO. Es importante tener el mínimo de redundancia posible cuando diseñemos el esquema conceptual de una base de datos. Si es deseable algo de redundancia a nivel de almacenamiento o a nivel de la vista de usuario, se puede introducir más tarde, como explicamos en la Sección 1.6.1.

3.7 Diagramas ER, convenciones de denominación y problemas de diseño

3.7.1 Resumen de la notación para los diagramas ER

Las Figuras 3.9 a 3.13 ilustran ejemplos de la participación de los tipos de entidad en los tipos de relación mediante sus extensiones: las instancias de entidad individuales y las instancias de relación en los conjuntos de entidades y los conjuntos de relaciones. En los diagramas ER se hace hincapié en la representación de los esquemas, más que de las instancias. Esto es más útil en el diseño de bases de datos porque el esquema de una base de datos rara vez cambia, mientras que el contenido de los conjuntos de entidades cambia con frecuencia. Además, normalmente es más fácil visualizar el esquema que la extensión de una base de datos, porque es mucho más pequeño.

La Figura 3.2 muestra el **esquema ER de la base de datos EMPRESA** como un **diagrama ER**. Vamos a repasar la notación completa de un diagrama ER. Los tipos de entidad como EMPLEADO, DEPARTAMENTO y PROYECTO aparecen en rectángulos. Los tipos de entidad como TRABAJA_PARA, ADMINISTRA, CONTROLA y TRABAJA_EN se muestran en rombos conectados a los tipos de entidades participantes mediante líneas rectas.

Los atributos se muestran en óvalos, y cada uno está conectado a su tipo de entidad o tipo de relación mediante una línea recta. Los atributos que componen un atributo compuesto se conectan con el óvalo que representa el atributo compuesto, como se muestra con el atributo Nombre de EMPLEADO. Los atributos multivalor se muestran con óvalos dobles, como el atributo Ubicaciones de DEPARTAMENTO. Los nombres de los atributos clave aparecen subrayados. Los atributos derivados se muestran con óvalos de línea punteada, como el atributo NumEmpleados de DEPARTAMENTO.

Los tipos de entidades débiles se distinguen porque se colocan en rectángulos de borde doble y porque las relaciones que los identifican aparecen en rombos dobles, como se ilustra con la entidad SUBORDINADO y el tipo de relación identificativa SUBORDINADOS_DE. La clave parcial del tipo de entidad débil se subraya con una línea punteada.

En la Figura 3.2 la razón de cardinalidad de cada tipo de relación *binaria* se especifica adjuntando 1, M o N a cada borde participante. La razón de cardinalidad de DEPARTAMENTO:EMPLEADO en ADMINISTRA es

1:1, mientras que es 1:N para DEPARTAMENTO:EMPLEADO en TRABAJA_PARA, y M:N para TRABAJA_EN. La restricción de participación se especifica mediante una línea sencilla para la participación parcial y con líneas dobles para la participación total (dependencia de existencia).

En la Figura 3.2 mostramos los nombres del papel para el tipo de relación CONTROL porque el tipo de entidad EMPLEADO puede desempeñar los dos papeles en esa relación. La cardinalidad es 1:N de supervisor a supervisado porque cada empleado en el papel de supervisado tiene como máximo un supervisor directo, mientras que un empleado en el papel de supervisor puede supervisar a ninguno o más empleados.

La Figura 3.14 resume las convenciones de los diagramas ER.

3.7.2 Asignación correcta de nombre a las construcciones del esquema

Al diseñar el esquema de una base de datos, la elección de nombre para los tipos de entidad, atributos, tipos de relaciones y (especialmente) los papeles desempeñados no siempre es directo. Hay que elegir nombres que transmitan, lo mejor posible, los significados de las distintas estructuras del esquema. Hemos optado por *nombres en singular* para los tipos de entidad, en lugar de nombres plurales, porque el nombre de un tipo de entidad se aplica a cada entidad individual que pertenece a ese tipo de entidad. En nuestros diagramas ER utilizaremos la convención de que los nombres de los tipos de entidades y de los tipos de relación se escriben en mayúsculas, los nombres de los atributos se escriben con la primera letra en mayúscula, y los nombres de los papeles en minúsculas. En la Figura 3.2 hemos utilizado esta convención.

Como práctica general, dada una descripción narrativa de los requisitos de la base de datos, los *nombres* que aparecen en la narrativa tienden a ser nombres de tipos de entidades, y los *verbos* tienden a indicar nombres de tipos de relación. Los nombres de los atributos normalmente proceden de los nombres originales que describen los nombres correspondientes a los tipos de entidades.

Otra consideración en cuanto a la denominación implica la elección de nombres de relación binaria para que el diagrama ER del esquema se pueda leer de izquierda a derecha y de arriba abajo. Por regla general, hemos seguido esta norma en la Figura 3.2. Para completar la explicación de esta convención, tenemos que hacer una excepción a lo mostrado en la Figura 3.2: el tipo de relación SUBORDINADOS_DE se lee de abajo hacia arriba. Al describir esta relación, podemos decir que las entidades SUBORDINADO (tipo de entidad inferior) son SUBORDINADOS_DE (nombre de relación) un EMPLEADO (tipo de entidad superior). Para que esto se pueda leer de arriba hacia abajo, tenemos que renombrar el tipo de relación a TIENE_SUBORDINADOS, que se puede leer del siguiente modo: una entidad EMPLEADO (tipo de entidad superior) TIENE_SUBORDINADOS (nombre de relación) del tipo SUBORDINADO (tipo de entidad inferior). Este asunto surge porque cada relación binaria puede describirse como que empieza en cualquiera de los dos tipos de entidad participantes, como se explica al principio de la Sección 3.4.


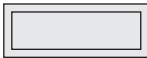

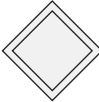



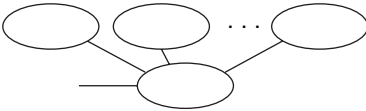

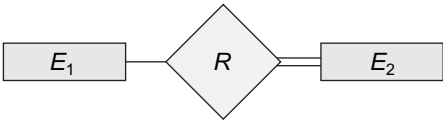
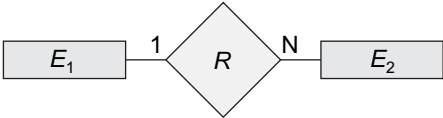
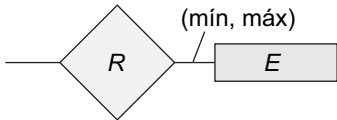
3.7.3 Opciones de diseño para el diseño conceptual ER

A veces puede resultar complejo decidir si un concepto en particular del minimundo debe modelarse como un tipo de entidad, un atributo o un tipo de relación. En esta sección ofrecemos algunos consejos breves sobre la construcción que debe elegirse en situaciones particulares.

En general, el proceso de diseño del esquema debe considerarse como un proceso de refinamiento iterativo: primero se crea un diseño inicial y después se va refinando paulatinamente hasta alcanzar el diseño más adecuado. Algunos de los refinamientos que a menudo se utilizan son los siguientes:

- Un concepto se puede interpretar primero como un atributo y, después, acabar como una relación porque se haya determinado que el atributo es una referencia a otro tipo de entidad. A menudo sucede que un par de estos atributos son inversos entre sí y se refinan como una relación binaria. Explicamos en profundidad este tipo de refinamiento en la Sección 3.6.

Figura 3.14. Resumen de la notación para los diagramas ER.

Símbolo	Significado
	Entidad
	Entidad débil
	Relación
	Relación de identificación
	Atributo
	Atributo clave
	Atributo multivalor
	Atributo compuesto
	Atributo derivado
	Participación total de E_2 en R
	Razón de cardinalidad 1: N para $E_1:E_2$ en R
	Restricción estructural (mín, máx) en la participación de E en R

- De forma parecida, un atributo que existe en varios tipos de entidad puede elevarse o promocionarse a un tipo de entidad independiente. Por ejemplo, suponga que varios tipos de entidades de la base de datos UNIVERSIDAD, como ESTUDIANTE, PROFESOR y CURSO, disponen en el diseño inicial de

un atributo Departamento; el diseñador puede optar entonces por crear un tipo de entidad DEPARTAMENTO con un solo atributo NombreDpto y relacionarlo con los tres tipos de entidad (ESTUDIANTE, PROFESOR y CURSO) a través de las relaciones apropiadas. Más tarde pueden descubrirse otros atributos/relaciones de DEPARTAMENTO.

- Es posible aplicar un refinamiento inverso al caso anterior: por ejemplo, si en el diseño inicial existe un tipo de entidad DEPARTAMENTO con un solo atributo NombreDpto que está únicamente relacionado con otro tipo de entidad, ESTUDIANTE. En este caso, DEPARTAMENTO se puede reducir o degradar a un atributo de ESTUDIANTE.
- La Sección 3.9 explica las opciones concernientes al grado de una relación. En el Capítulo 4 explicamos otros refinamientos relacionados con la especialización/generalización. El Capítulo 12 explica los refinamientos *top-down* (descendente) y *bottom-up* (ascendente) adicionales que son comunes en el diseño del esquema conceptual.

3.7.4 Notaciones alternativas para los diagramas ER

Hay muchas notaciones diagramáticas alternativas para la visualización de los diagramas ER. El Apéndice A ofrece algunas de las notaciones más populares. En la Sección 3.8 introducimos la notación UML (Lenguaje de modelado universal, *Universal Modeling Language*) para los diagramas de clase, que se han propuesto como estándar para el modelado conceptual de objetos.

En esta sección describimos una notación ER alternativa para especificar las restricciones estructurales en las relaciones. Esta notación implica asociar un par de números enteros (mín, máx) a cada participación de un tipo de entidad E en un tipo de relación R , donde $0 \leq \text{mín} \leq \text{máx}$ y $\text{máx} \geq 1$. Los números significan que para cada entidad e de E , e debe participar en al menos mín y a lo sumo en máx instancias de relación de R en cualquier momento. En este método, $\text{mín} = 0$ significa una participación parcial, mientras que $\text{mín} > 0$ implica una participación total.

La Figura 3.15 muestra el esquema de la base de datos EMPRESA utilizando la notación (mín, máx).¹⁴ Normalmente, se utiliza la notación de cardinalidad razón/línea-sencilla/línea-doble o la notación (mín, máx). Esta última es más precisa, y la podemos utilizar para especificar las restricciones estructurales de los tipos de relación de *cualquier grado*. Sin embargo, no es suficiente para especificar algunas restricciones clave o relaciones de grado superior, como explicamos en la Sección 3.9.

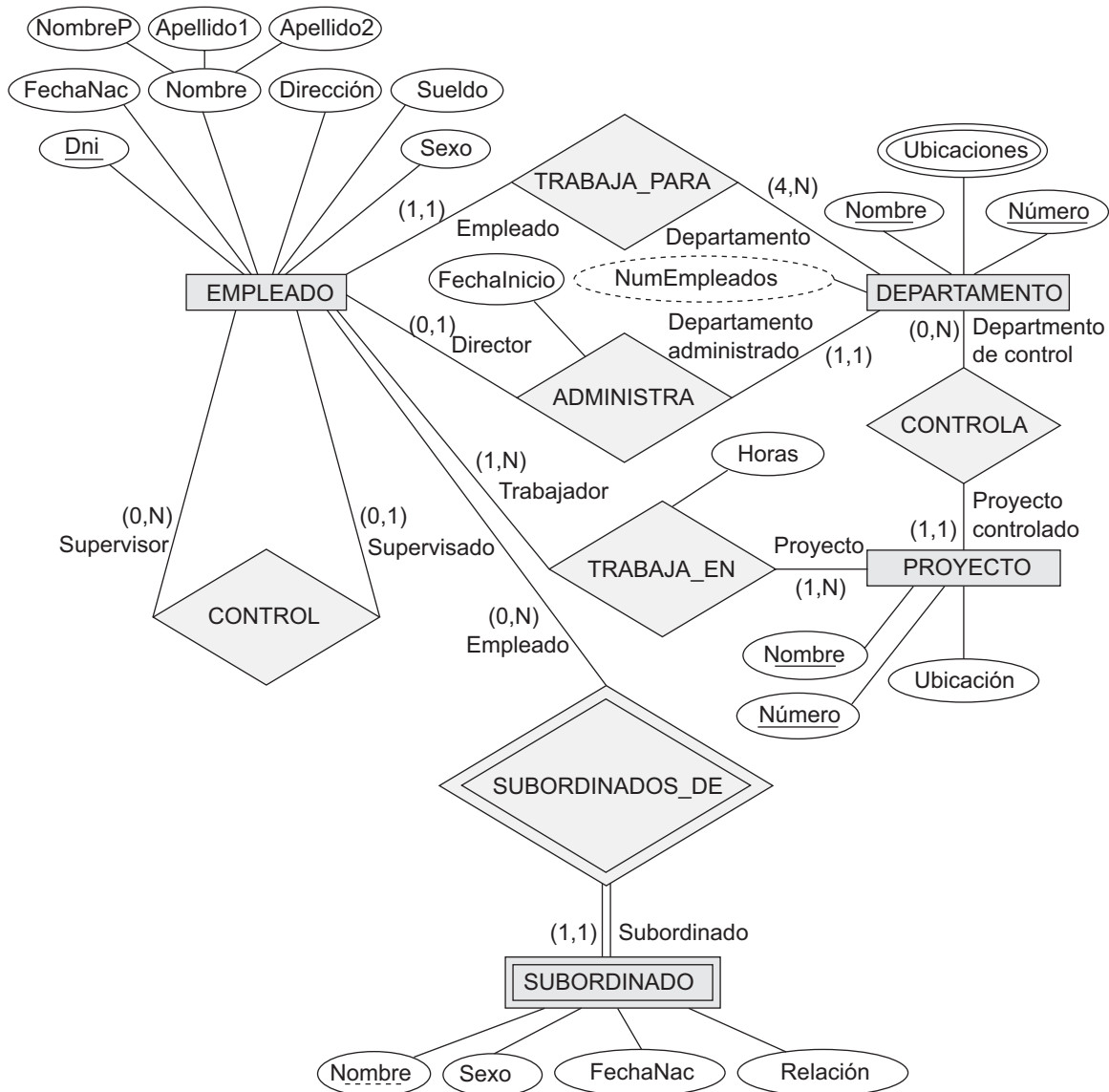
La Figura 3.15 también muestra los nombres de todos los papeles para el esquema de la base de datos EMPRESA.

3.8 Ejemplo de otra notación: diagramas de clase UML

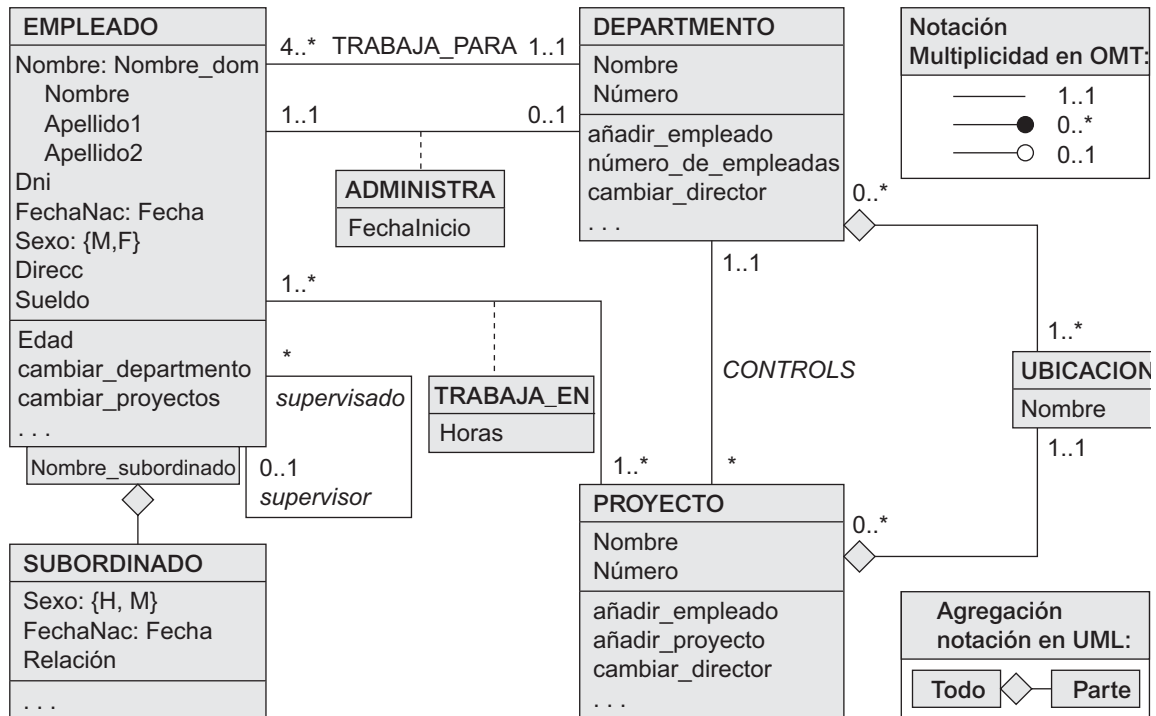
La metodología UML se está utilizando extensamente en el diseño de software y tiene muchos tipos de diagramas para los distintos fines del diseño de software. Sólo explicaremos brevemente los fundamentos de los **diagramas de clase UML** y los compararemos con los diagramas ER. En algunos casos, los diagramas de clase se pueden considerar como una notación alternativa a los diagramas ER. La notación UML adicional y sus conceptos se presentan en la Sección 4.6 y el Capítulo 12. La Figura 3.16 muestra cómo el esquema de la base de datos EMPRESA (véase la Figura 3.15) se puede visualizar utilizando la notación de los diagramas de clase UML. Los *tipos de entidades* de la Figura 3.15 se modelan como *clases* en la Figura 3.16. Una *entidad* en ER se corresponde con un *objeto* en UML.

¹⁴ En algunas notaciones, en concreto las que se utilizan en las metodologías de modelado de objetos, como UML, (mín, máx) se coloca en los lados opuestos a los que mostramos. Por ejemplo, para la relación TRABAJA_PARA de la Figura 3.15, el (1,1) estaría en el lado DEPARTAMENTO, y (4,N) estaría en el lado EMPLEADO. Aquí hemos utilizado la notación original de Abrial (1974).

Figura 3.15. Los diagramas ER para el esquema de la empresa. Hemos utilizado la notación (mín, máx) y los nombres de papel para especificar las restricciones estructurales.



En los diagramas de clase UML, una **clase** (equivalente a un tipo de entidad en ER) se muestra como un cuadro (véase la Figura 3.16) que incluye tres secciones: la sección superior ofrece el **nombre de la clase**; la sección intermedia incluye los **atributos** de los objetos individuales de la clase; y la última sección incluye las **operaciones** que se pueden aplicar a esos objetos. En los diagramas ER *no* se especifican las operaciones. Tomemos como ejemplo la clase EMPLEADO de la Figura 3.16. Sus atributos son Nombre, Dni, FechaNac, Sexo, Dirección y Sueldo. El diseñador puede especificar opcionalmente el **dominio** de un atributo, si lo desea, colocando el símbolo de dos puntos (:) seguido por el nombre de dominio o descripción, como se ilustra para los atributos Nombre, Sexo y FechaNac de EMPLEADO en la Figura 3.16. Un atributo compuesto se modela como un **dominio estructurado**, como en el caso del atributo Nombre de EMPLEADO. Un atributo multivalor generalmente se modelará como una clase separada, como UBICACIÓN en la Figura 3.16.

Figura 3.16. Esquema conceptual de EMPRESA en la notación de diagrama de clase UML.

En la tecnología UML, los tipos de relación se denominan **asociaciones** y las instancias de relación, **vínculos**. Una **asociación binaria** (tipo de relación binaria) se representa como una línea que conecta las clases participantes (tipos de entidad) y, opcionalmente, puede tener un nombre. Un atributo de relación, denominado **atributo de vínculo**, se coloca en un recuadro conectado con la línea de la asociación mediante una línea discontinua. La notación (mín, máx) descrita en la Sección 3.7.4 se utiliza para especificar las restricciones de relación, que en terminología UML se denominan **multiplicidades**. Las multiplicidades se especifican como *mín..máx*, y un asterisco (*) indica que no hay un límite máximo en la participación. No obstante, las multiplicidades se colocan en los *extremos opuestos de la relación* en comparación con la notación explicada en la Sección 3.7.4 (compare las Figuras 3.15 y 3.16). En UML, un asterisco indica una multiplicidad de 0..*, y un 1 indica una multiplicidad de 1..1. Una relación recursiva (consulte la Sección 3.4.2) se denomina **asociación reflexiva** en UML, y los nombres de papeles (como las multiplicidades) se colocan en los extremos opuestos de una asociación en comparación con la colocación de los nombres de papel en la Figura 3.15. En UML, hay dos tipos de relaciones: asociación y agregación. La **agregación** está pensada para representar una relación entre un objeto completo y sus partes constitutivas, y tiene una notación diagramática distinta. En la Figura 3.16 modelamos las ubicaciones de un departamento y la ubicación sencilla de un proyecto como agregaciones. No obstante la agregación y la asociación no tienen propiedades estructurales diferentes y la elección del tipo de relación que hay que utilizar es algo subjetivo. En el modelo ER, las dos se representan como relaciones.

UML también distingue entre asociaciones (o agregaciones) **unidireccionales** y **bidireccionales**. En el caso unidireccional, la línea que conecta las clases se muestra con una flecha para indicar que sólo se necesita una dirección para acceder a los objetos relacionados. Si no aparece una flecha, se asume la cualidad bidireccional, que es lo predeterminado. Por ejemplo, si siempre esperamos acceder al director de un departamento a partir de un objeto DEPARTAMENTO, dibujaremos la línea de asociación que representa la asociación ADMINI-

NISTRA con una flecha desde DEPARTAMENTO hasta EMPLEADO. Además, es posible especificar que las instancias de relación se **ordenen**. Por ejemplo, podríamos especificar que los objetos de empleado relacionados con cada departamento a través de la asociación (relación) TRABAJA_PARA se ordenen por el valor de su atributo FechaNac. Los nombres de asociación (relación) son *opcionales* en UML, y los atributos de relación se muestran en un cuadro conectado con una línea discontinua a la línea que representa la asociación/agregación (consulte FechaInicio y Horas en la Figura 3.16).

Las operaciones dadas en cada clase se derivan de los requisitos funcionales de la aplicación, como se explicó en la Sección 3.1. Normalmente, es suficiente especificar los nombres de operación al principio para las operaciones lógicas que se espera aplicar a los objetos individuales de una clase (véase la Figura 3.16). Al refinar un diseño, se añaden más detalles, como los tipos de argumento exactos (parámetros) para cada operación, más una descripción funcional de cada operación. UML tiene *descripciones de función y diagramas de secuencia* para especificar parte de los detalles de operación, pero esto queda fuera del ámbito de nuestra explicación. El Capítulo 12 introducirá algunos de estos diagramas.

Las entidades débiles se pueden modelar utilizando la construcción denominada **asociación cualificada** (o **agregación cualificada**) en UML; esto puede representar tanto la relación identificativa como la clave parcial, que se coloca en un cuadro conectado con la clase propietaria. Es lo que se ilustra en la Figura 3.16 con la clase SUBORDINADO y su agregación cualificada a EMPLEADO. La clave parcial NombreSubordinado se denomina **discriminador** en terminología UML, puesto que su valor distingue los objetos asociados con (relacionados con) el mismo EMPLEADO. Las asociaciones cualificadas no están restringidas al modelado de entidades débiles, y se pueden utilizar para modelar otras situaciones en UML.

3.9 Tipos de relación con grado mayor que dos

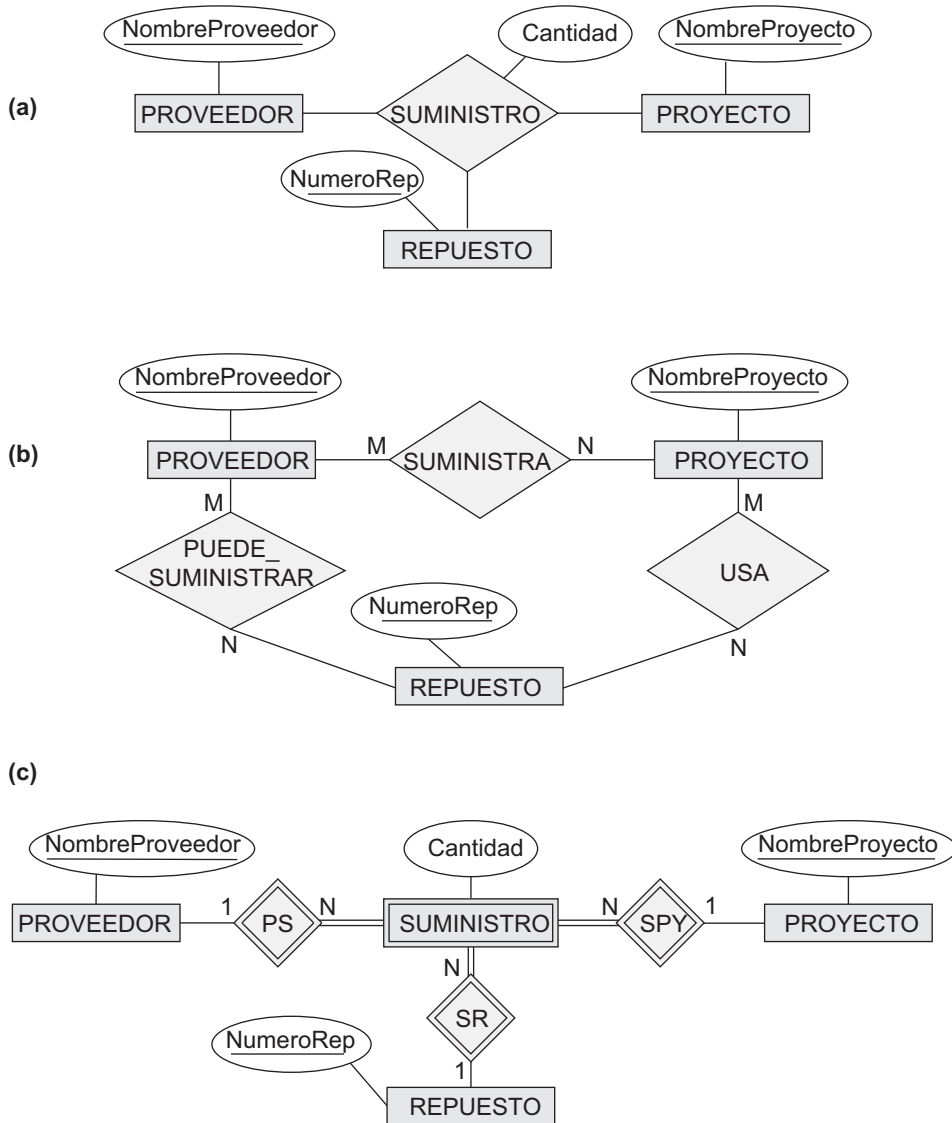
En la Sección 3.4.2 definimos el **grado** de un tipo de relación como el número de tipos de entidades participantes. Un tipo de relación de grado dos es *binario* y un tipo de relación de grado tres, *ternario*. En esta sección explicamos más en detalle las diferencias entre las relaciones binarias y de grado superior, cuándo elegir relaciones de grado superior o binarias, y las restricciones en las relaciones de grado superior.

3.9.1 Elección entre relaciones binarias y ternarias (o de grado superior)

En la Figura 3.17(a) se muestra la notación de diagrama ER para un tipo de relación ternario: el esquema para el tipo de relación SUMINISTRO que se mostraba a nivel de instancia en la Figura 3.10. Recuerde que el conjunto de relación de SUMINISTRO es un conjunto de instancias de relación (s, j, p) , donde s es un PROVEEDOR que actualmente está suministrando un REPUESTO p a un PROYECTO j . En general, un tipo de relación R de grado n tendrá n bordes en un diagrama ER, uno conectando R con cada tipo de entidad participante.

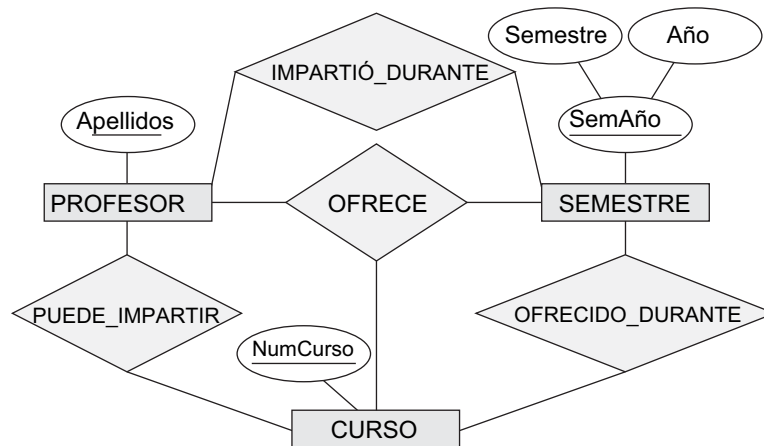
La Figura 3.17(b) muestra un diagrama ER para los tres tipos de relación binaria PUEDE_SUMINISTRAR, USA y SUMINISTRA. En general, un tipo de relación ternaria representa información diferente que tres tipos de relación binaria. Considere los tres tipos de relación binaria PUEDE_SUMINISTRAR, USA y SUMINISTRA. Suponga que PUEDE_SUMINISTRAR, entre PROVEEDOR y REPUESTO, incluye una instancia (s, p) cuando un proveedor s puede suministrar el repuesto p (a cualquier proyecto); USA, entre PROYECTO y REPUESTO, incluye una instancia (j, p) cuando el proyecto j utiliza el repuesto p ; y SUMINISTRA, entre PROVEEDOR y PROYECTO, incluye una instancia (s, j) cuando el proveedor s suministra algún repuesto al proyecto j . La existencia de tres instancias de relación (s, p) , (j, p) y (s, j) en PUEDE_SUMINISTRAR, USA y SUMINISTRA, respectivamente, no implica necesariamente que exista una instancia (s, j, p) en la relación ternaria SUMINISTRO, porque el *significado es diferente*. A menudo es complejo decidir si una relación en particular debe representarse como un tipo de relación de grado n o si debe dividirse en varios tipos de relación de

Figura 3.17. Tipos de relaciones ternarias . (a) La relación SUMINISTRO. (b) Tres relaciones binarias no son equivalentes a SUMINISTRO. (c) SUMINISTRO representada como un tipo de entidad débil.



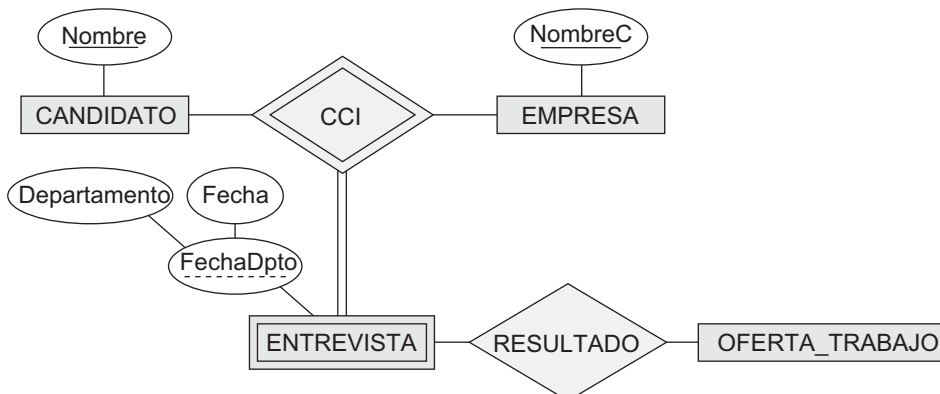
menor grado. El diseñador debe basar esta decisión en la semántica o significado de la situación particular que se está representando. La solución típica es incluir la relación ternaria *más* una o más relaciones binarias, si representan varios significados y todas son necesarias para la aplicación.

Algunas herramientas de diseño de bases de datos están basadas en variaciones del modelo ER que sólo permiten las relaciones binarias. En este caso, una relación ternaria como **SUMINISTRO** debe representarse como un tipo de entidad débil, sin clave parcial y con tres relaciones identificativas. Los tres tipos de entidad participantes, **PROVEEDOR**, **REPUESTO** y **PROYECTO**, son conjuntamente los tipos de entidad propietaria (véase la Figura 3.17[c]). Por tanto, una entidad en el tipo de entidad débil **SUMINISTRO** de la Figura 3.17(c) queda identificada por la combinación de sus tres entidades propietarias de **PROVEEDOR**, **REPUESTO** y **PROYECTO**.

Figura 3.18. Otro ejemplo de relación ternaria frente a relación binaria.

También es posible representar la relación ternaria como un tipo de entidad regular introduciendo una clave artificial o sustituta. En este ejemplo, podría utilizarse un atributo clave *IdSuministro* para el tipo de entidad suministro, convirtiéndolo en un tipo de entidad regular. Tres relaciones 1:N binarias relacionan SUMINISTRO con los tres tipos de entidad participantes.

En la Figura 3.18 se muestra otro ejemplo. El tipo de relación ternaria *OFRECE* representa información de los profesores que ofrecen cursos durante los semestres; por tanto, incluye una instancia de relación (p, s, c) siempre que el PROFESOR p ofrece un CURSO c durante el SEMESTRE s . Los tres tipos de relación binaria de la Figura 4.12 significan lo siguiente: *PUEDE_IMPARTIR* relaciona un curso con los profesores que pueden impartirlo, *IMPARTIÓ_DURANTE* relaciona un semestre con los profesores que *impartieron algún curso* durante ese semestre, y *OFRECIDO_DURANTE* relaciona un semestre con los cursos ofrecidos durante ese semestre *por cualquier profesor*. Estas relaciones ternarias y binarias representan diferente información, pero debe haber algunas restricciones entre las relaciones. Por ejemplo, en *OFRECE* no debería existir una instancia de relación (p, s, c) a menos que exista una instancia (p, s) en *IMPARTIÓ_DURANTE*, que exista una instancia (s, c) en *OFRECIDO_DURANTE*, y que exista una instancia (p, c) en *PUEDE_IMPARTIR*. No obstante, lo opuesto no siempre se cumple; podemos tener instancias (p, s) , (s, c) y (p, c) en las tres relaciones binarias sin que haya una instancia (p, s, c) en *OFRECE*. En este ejemplo, y basándose en los significados de las relaciones, podemos inferir las instancias de *IMPARTIÓ_DURANTE* y *OFRECIDO_DURANTE* de las instancias en

Figura 3.19. Un tipo de entidad débil ENTREVISTA con un tipo de relación identificativa ternaria.

OFRECE, pero no podemos inferir las instancias de PUEDE_IMPARTIR; por consiguiente, IMPARTIÓ_DURANTE y OFRECIDO_DURANTE son redundantes y se pueden omitir.

Aunque por lo general tres relaciones binarias *no pueden* reemplazar a una relación ternaria, esto puede ser válido bajo ciertas *restricciones adicionales*. En nuestro ejemplo, si la relación PUEDE_IMPARTIR es 1:1 (un profesor puede impartir un curso, y un curso puede ser impartido por un solo profesor), entonces la relación ternaria OFRECE se puede omitir porque puede inferirse de las tres relaciones binarias PUEDE_IMPARTIR, IMPARTIÓ_DURANTE y OFRECIDO_DURANTE. El diseñador del esquema debe analizar el significado de cada situación específica para decidir los tipos de relaciones binarias y ternarias que son necesarias.

Es posible tener un tipo de entidad débil con un tipo de relación identificativa ternaria (o n -ary). En este caso, el tipo de entidad débil puede tener *varios* tipos de entidad propietarias. En la Figura 3.19 se ofrece un ejemplo.

3.9.2 Restricciones en las relaciones ternarias (o de grado superior)

Hay dos notaciones para especificar las restricciones estructurales en las relaciones n -ary. *Deben utilizarse ambos* si es importante especificar completamente las restricciones estructurales de una relación ternaria o de grado superior. La primera notación está basada en la notación de la razón de cardinalidad de las relaciones binarias de la Figura 3.2, donde se utiliza 1, M o N en cada arco de participación (los símbolos M y N significan *muchos* o *cualquier cantidad*).¹⁵ Permítanos ilustrar esta restricción con la relación SUMINISTRO de la Figura 3.17.

Recuerde que el conjunto de relación de SUMINISTRO es un conjunto de instancias de relación (s, j, p) , donde s es un PROVEEDOR, j es un PROYECTO y p es un REPUESTO. Suponga que existe una restricción según la cual sólo se puede utilizar un proveedor para una combinación proyecto-repuesto particular (sólo un proveedor suministra un repuesto particular a un proyecto concreto). En este caso, colocamos un 1 en la participación PROVEEDOR, y M, N en las participaciones PROYECTO, REPUESTO de la Figura 3.17. Esto especifica la restricción de que una combinación (j, p) en particular puede aparecer a lo sumo una vez en el conjunto de relación porque cada combinación (PROYECTO, REPUESTO) determina sin lugar a dudas un único proveedor. Por tanto, cualquier instancia de relación (s, j, p) es identificada excepcionalmente en el conjunto de relación por su combinación (j, p) , lo que convierte a (j, p) en una clave para el conjunto de relación. En esta notación, no es necesario que las participaciones que tienen una especificada sean parte de la clave de identificación del conjunto de relación.¹⁶

La segunda notación está basada en la notación (mín, máx) de la Figura 3.15 para las notaciones binarias. Una pareja (mín, máx) en una participación específica aquí que cada entidad está relacionada con al menos *mín* relaciones y con a lo sumo *máx* instancias de relación en el conjunto de relación. Estas restricciones no conllevan determinar la clave de una relación n -ary, donde $n > 2$,¹⁷ pero especifica un tipo de restricción diferente que restringe la cantidad de instancias de relación en las que cada entidad puede participar.

3.10 Resumen

En este capítulo hemos presentado los conceptos de modelado de un modelo de datos conceptual de nivel alto, el modelo Entidad-Relación (ER). Hemos empezado explicando el papel que un modelo de este tipo juega en el proceso de diseño de una base de datos, y luego presentamos un conjunto de requisitos para la base de datos EMPRESA, que es uno de los ejemplos que utilizamos a lo largo del libro. Definimos los conceptos básicos

¹⁵ Esta notación permite determinar la clave de la relación, como se explica en el Capítulo 7.

¹⁶ Esto también es cierto para las razones de cardinalidad de las relaciones binarias.

¹⁷ Las restricciones (mín, máx) pueden determinar las claves para las relaciones binarias.

de entidad y atributo del modelo ER. Después hablamos de los valores NULL y presentamos los distintos tipos de atributos, que se pueden anidar arbitrariamente para producir atributos complejos:

- Simple o atómico.
- Compuesto.
- Multivalor.

También explicamos brevemente los atributos almacenados frente a los derivados, para después adentrarnos en los conceptos del modelo ER relativos al nivel de esquema o “intención”:

- Tipos de entidad y sus conjuntos de entidades correspondientes.
- Atributos clave de los tipos de entidad.
- Conjuntos de valores (dominios) de atributos.
- Tipos de relaciones y sus conjuntos de relaciones correspondientes.
- Participaciones de los tipos de entidades en los tipos de relaciones.

Hemos presentado dos métodos para especificar las restricciones estructurales en los tipos de relaciones. El primer método distingue dos tipos de restricciones estructurales:

- Razones de cardinalidad (1:1, 1:N, M:N en las relaciones binarias).
- Restricciones de participación (total, parcial).

Otro método alternativo para especificar las restricciones estructurales consiste en especificar una cantidad mínima y máxima (mín, máx) de participación de cada tipo de entidad en un tipo de relación. Asimismo, explicamos los tipos de entidad débiles y los conceptos relacionados de tipos de entidad propietarios, identificación de tipos de relación y atributos de clave parcial.

Los esquemas de Entidad-Relación se pueden representar diagramáticamente como diagramas ER. Hemos visto cómo diseñar un esquema ER para la base de datos EMPRESA, definiendo en primer lugar los tipos de entidades y sus atributos, para después refinar el diseño a fin de incluir los tipos de relaciones. Mostramos el diagrama ER correspondiente al esquema de la base de datos EMPRESA y explicamos algunos de los conceptos básicos de los diagramas de clase UML y de cómo se relacionan con los conceptos del modelo ER. También hemos descrito más en detalle los tipos de relación ternaria de alto nivel, así como las circunstancias que los diferencian de las relaciones binarias.

Los conceptos de modelado ER que hemos presentado hasta ahora (tipos de entidades, tipos de relaciones, atributos, claves y restricciones estructurales) pueden modelar las típicas aplicaciones de bases de datos de procesamiento de datos empresariales. No obstante, las aplicaciones más modernas y complejas (por ejemplo, diseño en ingeniería, sistemas de información médica o telecomunicaciones) requieren conceptos adicionales si queremos modelarlas con mayor precisión. En el Capítulo 4 explicamos algunos conceptos avanzados sobre modelado, mientras que en el Capítulo 24 volveremos a ver las técnicas de modelado de datos.

Preguntas de repaso

- 3.1. Explique el papel de un modelo de datos de alto nivel en el proceso de diseño de una base de datos.
- 3.2. Enumere los distintos casos donde podría resultar apropiado utilizar un valor NULL.
- 3.3. Defina los siguientes términos: *entidad*, *atributo*, *valor de atributo*, *instancia de relación*, *atributo compuesto*, *atributo multivalor*, *atributo derivado*, *atributo complejo*, *atributo clave* y *conjunto de valores (dominio)*.
- 3.4. ¿Qué es un tipo de entidad? ¿Qué es un conjunto de entidades? Explique las diferencias entre una entidad, un tipo de entidad y un conjunto de entidades.

- 3.5. Explique la diferencia entre un atributo y un conjunto de valores.
- 3.6. ¿Qué es un tipo de relación? Explique las diferencias entre una instancia de relación, un tipo de relación y un conjunto de relaciones.
- 3.7. ¿Qué es un rol de participación? ¿Cuándo es necesario utilizar nombres de rol en la descripción de los tipos de relaciones?
- 3.8. Describa las dos alternativas que hay para especificar las restricciones estructurales en los tipos de relaciones. ¿Cuáles son las ventajas y los inconvenientes de cada una?
- 3.9. ¿Bajo qué condiciones un atributo de un tipo de relación binaria puede migrarse para convertirse en un atributo de uno de los tipos de entidad participantes?
- 3.10. Cuando pensamos en las relaciones como en atributos, ¿cuáles son los conjuntos de valores de esos atributos? ¿Qué clase de modelos de datos está basada en este concepto?
- 3.11. ¿Qué se entiende por tipo de relación recursivo? Ponga algunos ejemplos.
- 3.12. ¿Cuándo se utiliza el concepto de tipo de entidad débil en el modelado de datos? Defina los términos *tipo de entidad propietaria*, *tipo de entidad débil*, *identificación del tipo de entidad* y *clave parcial*.
- 3.13. ¿Una relación de identificación de un tipo de entidad débil puede ser de un grado mayor que dos? Ofrezca algunos ejemplos para ilustrar su respuesta.
- 3.14. Explique las convenciones para visualizar un esquema ER como un diagrama ER.
- 3.15. Explique las convenciones de denominación que se utilizan para los diagramas de esquema ER.

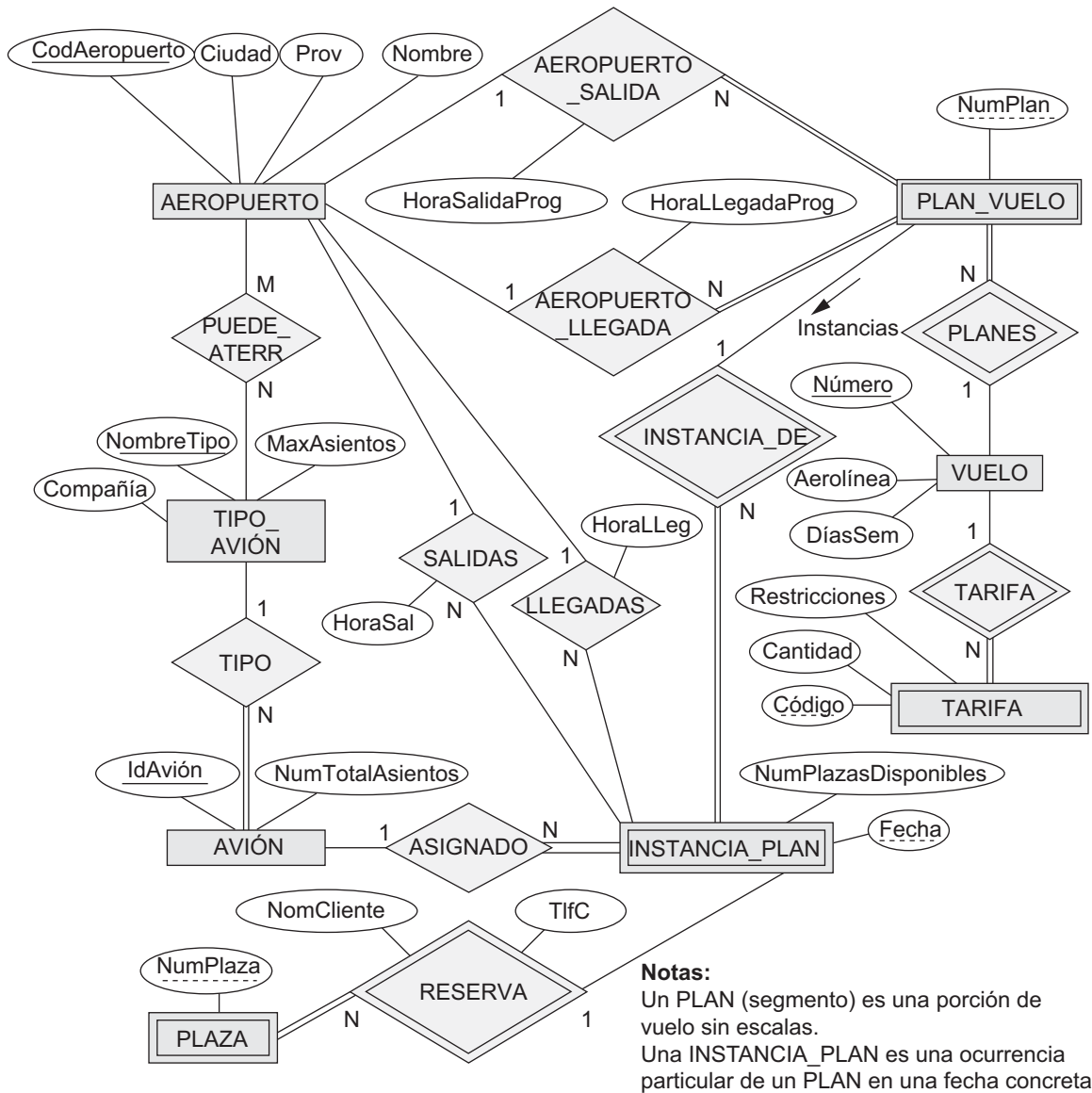
Ejercicios

- 3.16. Considere el siguiente conjunto de requisitos para una base de datos UNIVERSIDAD que se utiliza para hacer un seguimiento del certificado de estudios de los estudiantes. Es parecido pero no idéntico a la base de datos de la Figura 1.2:
 - a. La universidad registra el nombre, el número de estudiante, el dni, la dirección y el teléfono actuales, la dirección y el teléfono permanentes, la fecha de nacimiento, el sexo, la clase (estudiante de primer año, de segundo año,..., diplomado), departamento principal, departamento menor (si lo hay) y programa de grado (B.A., B.S., . . . , Ph.D.). Algunas aplicaciones de usuario necesitan referirse a la ciudad, la provincia y el código postal de la dirección permanente del estudiante, así como a los apellidos. Tanto el DNI como el número de estudiante tienen valores únicos para cada estudiante.
 - b. Cada departamento está descrito por un nombre, un código de departamento, un número de oficina, un teléfono de la oficina y la universidad. El nombre y el código tienen valores únicos para cada departamento.
 - c. Cada curso tiene un nombre de curso, una descripción, un número de curso, un número de horas por semestre, un nivel y el departamento que lo ofrece. El valor del número de curso es único para cada curso.
 - d. Cada sección tiene un profesor, un semestre, un año, un curso y un número de sección. Este último distingue las secciones del mismo curso que se imparten durante el mismo semestre/año; sus valores son 1, 2, 3, . . . , hasta el número de secciones impartidas durante cada semestre.
 - e. Un informe de calificaciones consta del estudiante, la sección, la letra de la calificación y un grado numérico (0, 1, 2, 3, o 4).

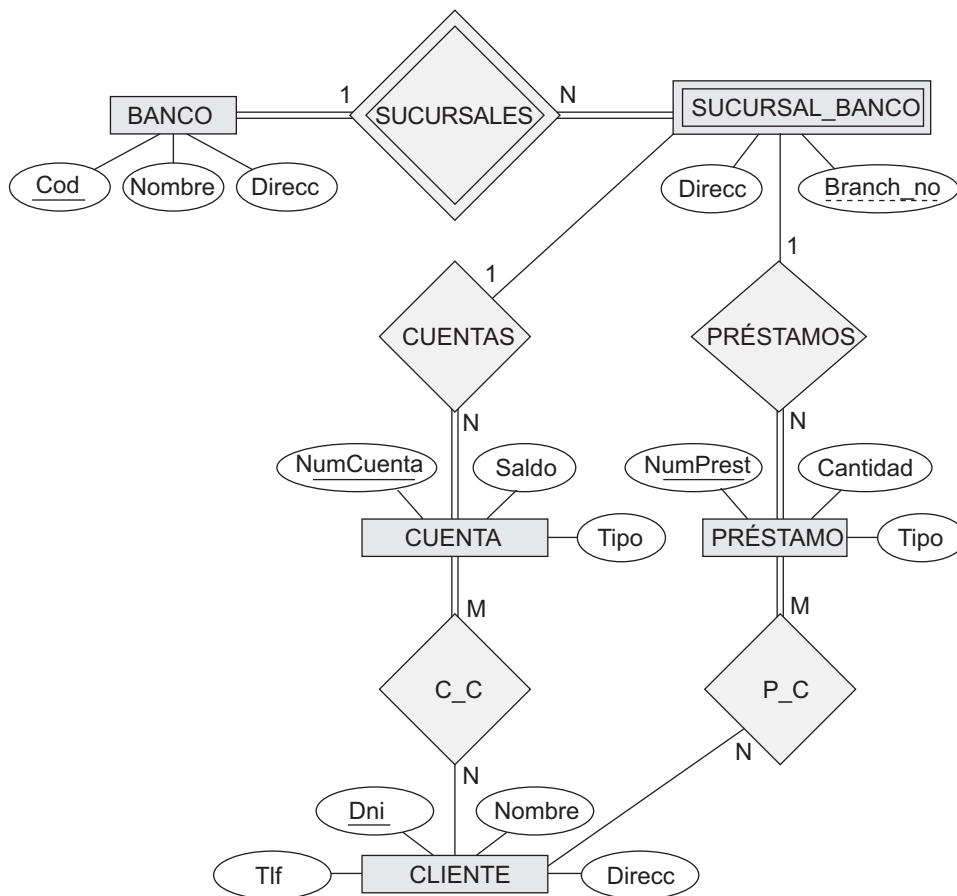
Diseñe un esquema ER para esta aplicación y dibuje un diagrama ER para el esquema. Especifique los atributos clave de cada tipo de entidad y las restricciones estructurales de cada tipo de relación.

Anote cualquier requisito no especificado y haga las suposiciones adecuadas para realizar una especificación completa.

- 3.17. Los atributos compuestos y multivalor se pueden anidar hasta cualquier nivel. Suponga que queremos diseñar un atributo para un tipo de entidad ESTUDIANTE para hacer un seguimiento de su formación universitaria. Un atributo así tendrá una entrada por cada universidad en la que haya estudiado, y cada una de estas entradas estará compuesta por el nombre de la universidad, las fechas de inicio y fin, las calificaciones (los grados otorgados por esa universidad, si los hubiera) y las entradas con los certificados de estudios (cursos completados en esa universidad, si fuera aplicable). Cada entrada de grado contiene el nombre del grado y el mes y el año en que se consiguió ese grado, y cada entrada de certificado de estudios contiene un nombre del curso, el semestre, el año y la calificación. Diseñe un atributo para almacenar esta información. Utilice las convenciones de la Figura 3.5.
- 3.18. Muestre un diseño alternativo para el atributo descrito en el Ejercicio 3.17 que utilice únicamente tipos de entidad (incluyendo tipos de entidad débiles, si es necesario) y tipos de relación.
- 3.19. Considere el diagrama ER de la Figura 3.20, que muestra un esquema simplificado para un sistema de reservas en aerolíneas. Extraiga del diagrama ER los requisitos y las restricciones que produjeron este esquema. Intente ser tan preciso como sea posible en su especificación de requisitos y restricciones.
- 3.20. En los Capítulos 1 y 2 explicamos el entorno de una base de datos y los usuarios de las bases de datos. Podemos considerar muchos tipos de entidad para describir un entorno semejante, como un DMBS, una base de datos almacenada, un DBA y un diccionario catálogo/datos. Intente especificar todos los tipos de entidad que pueden describir completamente un sistema de bases de datos y su entorno; después, especifique los tipos de relación entre ellos y dibuje un diagrama ER para describir un entorno de bases de datos general semejante.
- 3.21. Diseñe un esquema ER para seguir la información sobre las votaciones llevadas a cabo en la Cámara de Diputados de Estados Unidos durante la sesión congresional actual de dos años. La base de datos tiene que registrar el nombre de todos los estados (por ejemplo, 'Texas', 'Nueva York', 'California') e incluir la región del estado (cuyo dominio es {'Noreste', 'Medio oeste', 'Sureste', 'Suroeste', 'Oeste'}). Cada PERSONA_CONGRESO de la Cámara de Diputados aparece descrita por su Nombre más el Distrito representado, la FechaInicio de cuando ese diputado fue elegido por primera vez, y el Partido político al que esa persona pertenece (cuyo dominio es {'Republicano', 'Demócrata', 'Independiente', 'Otro'}). La base de datos hace un seguimiento de cada PROYECTOLEY (por ejemplo, propuesta de ley), incluyendo el NombrePropuesta, la FechaDeVoto de la propuesta, si la propuesta fue o no aprobada (AprobadaONo) (cuyo dominio es {'Sí', 'No'}), y el Promotor (el o los diputados que promovieron la propuesta de ley). La base de datos también registra la votación de los diputados (el dominio del atributo Voto es {'Sí', 'No', 'Abstención', 'Ausencia'})). Dibuje el diagrama del esquema ER para esta aplicación. Declare explícitamente todas las suposiciones que haga.
- 3.22. Se está construyendo una base de datos para hacer un seguimiento de los equipos y los partidos de una liga deportiva. Un equipo tiene un determinado número de jugadores, y no todos juegan en cada partido. Es deseable hacer un seguimiento de los jugadores que disputan cada partido y por cada equipo, las posiciones en las que jugaron en ese partido y el resultado del mismo. Diseñe un diagrama de esquema ER para esta aplicación, describiendo las suposiciones que haga. Elija su deporte favorito (por ejemplo, fútbol, baloncesto, béisbol).
- 3.23. Considere el diagrama ER de la Figura 3.21 para parte de la base de datos de un BANCO. Cada banco puede tener varias sucursales, y cada sucursal puede tener varias cuentas y préstamos.
 - a. Liste los tipos de entidad (no débiles) del diagrama ER.

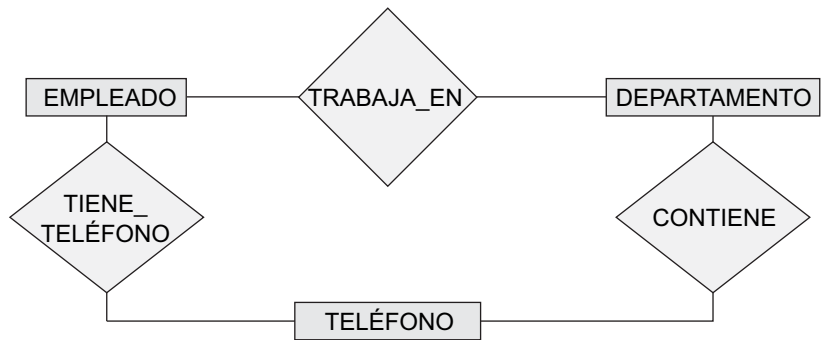
Figura 3.20. Diagrama ER para el esquema de una base de datos AEROLÍNEA.

- ¿Hay algún tipo de entidad débil? En ese caso, proporcione su nombre, la clave parcial y la relación de identificación.
- ¿Qué restricciones especifican en este diagrama la clave parcial y la relación de identificación del tipo de entidad débil?
- Liste los nombres de todos los tipos de relación y especifique la restricción (mín, máx) de cada participación de un tipo de entidad en un tipo de relación. Justifique sus opciones.
- Enumere brevemente los requisitos de usuario que conducen a este diseño de esquema ER.
- Suponga que cada cliente debe tener al menos una cuenta pero está restringido a tener un máximo de dos préstamos simultáneos, y que una sucursal de un banco no puede tener más de 1.000 préstamos. ¿Cómo se muestra esto en las restricciones (mín, máx)?

Figura 3.21. Diagrama ER para el esquema de la base de datos de un BANCO.

- 3.24.** Considere el diagrama ER de la Figura 3.22. Asuma que un empleado puede trabajar en hasta dos departamentos o que puede no ser asignado a cualquier departamento. Suponga que cada departamento debe tener un número de teléfono y que puede tener hasta tres. Proporcione las restricciones (mín, máx) en este diagrama. *Explique claramente las suposiciones adicionales que haga.* En este ejemplo, ¿bajo qué condiciones sería redundante la relación TIENE_TELÉFONO?
- 3.25.** Considere el diagrama ER de la Figura 3.23. Un curso puede o no utilizar un libro de texto, pero un texto, por definición, es un libro que se utiliza en algún curso. Un curso no puede utilizar más de cinco libros. Los profesores imparten de dos a cuatro cursos. Proporcione las restricciones (mín, máx) de este diagrama. *Explique claramente las suposiciones adicionales que haga.* Si añadimos la relación ADOPTA entre PROFESOR y TEXTO, ¿qué restricciones (mín, máx) especificaría? ¿Por qué?
- 3.26.** Considere un tipo de entidad SECCIÓN en la base de datos UNIVERSIDAD que describe la sección que ofrece los cursos. Los atributos de SECCIÓN son NumSección, Semestre, Año, NumCurso, Profesor, NumSala (donde se imparte la sección), Edificio (donde se imparte la sección), DíasSemana (dominio de las posibles combinaciones de días laborables en las que puede ofrecerse una sección, {'LXV', 'XV', 'MJ', etcétera}) y Horas (dominio de todos los periodos de tiempo posibles durante los que se ofrecen las secciones {'9–9:50 A.M.', '10–10:50 A.M.', ..., '3:30–4:50 P.M.', '5:30–6:20 P.M.', etcétera}). Suponga que NumSección es un valor único por cada curso

Figura 3.22. Parte de un diagrama ER para una base de datos EMPRESA.

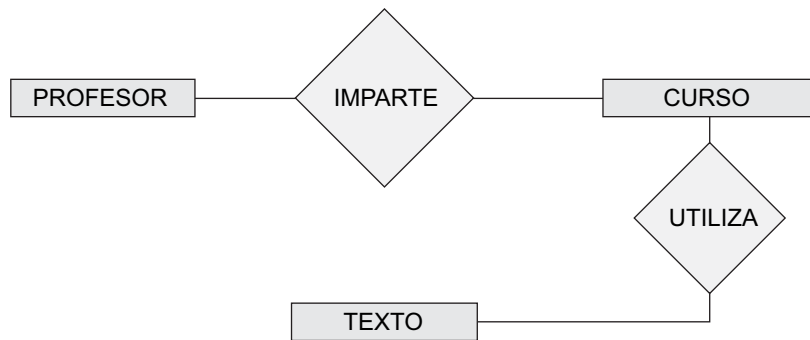
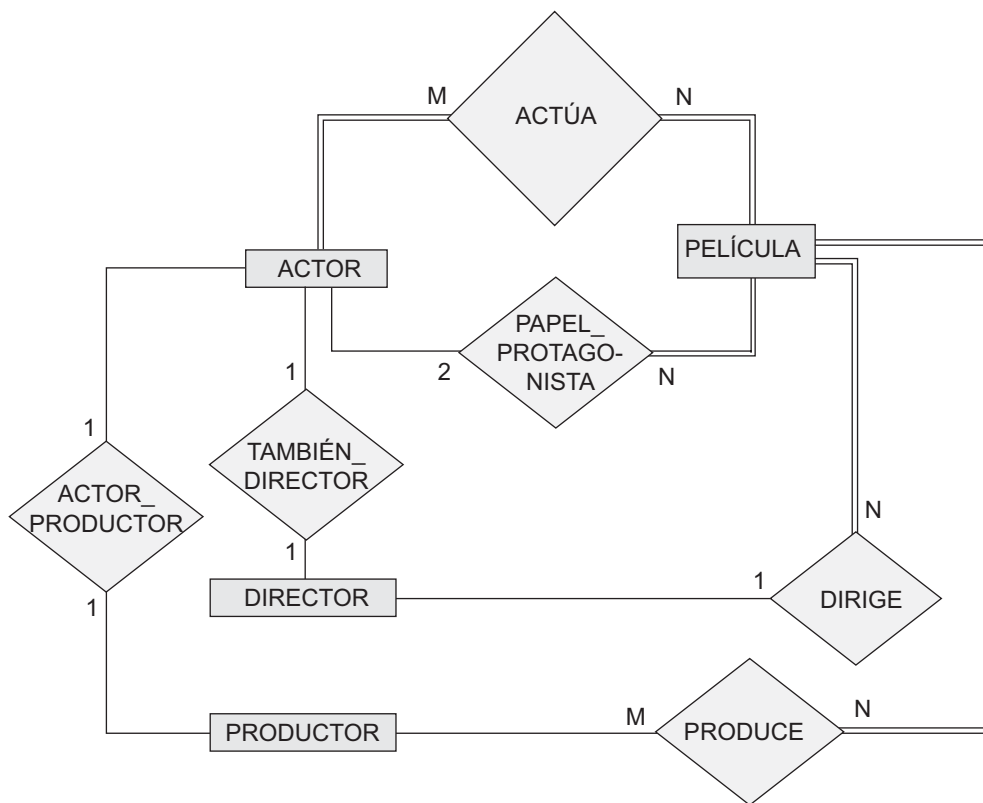


dentro de una combinación semestre/año particular (es decir, si un curso se ofrece varias veces durante un semestre en particular, sus ofertas de sección se numeran como 1, 2, 3, etcétera). Hay varias claves compuestas por sección, y algunos atributos están compuestos por más de una clave. Identifique tres claves compuestas y muestre cómo se pueden representar en un diagrama de esquema ER.

3.27. Las razones de cardinalidad a menudo dictaminan el diseño detallado de una base de datos. La razón de cardinalidad depende del significado real de los tipos de entidad implicados y queda definida por la aplicación específica. Para las siguientes relaciones binarias, sugiera las razones de cardinalidad basándose en el significado de sentido común de los tipos de entidad. Explique claramente las suposiciones que haga.

Entidad 1	Razón de cardinalidad	Entidad 2
1. ESTUDIANTE	_____	DNI
2. ESTUDIANTE	_____	PROFESOR
3. AULA	_____	PARED
4. PAÍS	_____	PRESIDENTE_ACTUAL
5. CURSO	_____	LIBROTEXTO
6. ELEMENTO (que se puede encontrar en un pedido)	_____	PEDIDO
7. ESTUDIANTE	_____	CLASE
8. CLASE	_____	PROFESOR
9. PROFESOR	_____	OFICINA
10. ARTÍCULO_SUBASTA_EBAY	_____	OFERTA_EBAY

3.28. Considere el esquema ER para la base de datos PELÍCULAS de la Figura 3.24. Asuma que PELÍCULAS es una base de datos rellena. Actor se utiliza como término genérico e incluye actrices. Dadas las restricciones mostradas en el esquema ER, responda a las siguientes afirmaciones con Verdadero, Falso o Quizás. Asigne esta última respuesta a las afirmaciones que, aun no pudiendo mostrarse explícitamente como Verdaderas, tampoco se puede probar que sean Falsas basándose en el esquema mostrado. Justifique sus respuestas.

Figura 3.23. Parte de un diagrama ER para una base de datos CURSOS.**Figura 3.24.** Diagrama ER para el esquema de una base de datos PELÍCULAS.

- a. En esta base de datos no hay ningún actor que no haya actuado en ninguna película.
- b. Hay algunos actores que han actuado en más de diez películas.
- c. Algunos actores han sido protagonistas en varias películas.
- d. Una película sólo puede tener un máximo de dos protagonistas.
- e. Cada director ha sido actor en alguna película.
- f. Ningún productor ha sido actor alguna vez.
- g. Un productor no puede ser actor en alguna otra película.

- h. Hay películas con más de una docena de actores.
 - i. Algunos productores también han sido directores.
 - j. La mayoría de las películas tienen un director y un productor.
 - k. Algunas películas tienen un director pero varios productores.
 - l. Hay algunos actores que han interpretado el papel de protagonista, dirigido una película y producido alguna película.
 - m. Ninguna película tiene un director que también haya actuado en ella.
- 3.29.** Dado el esquema ER para la base de datos PELÍCULAS de la Figura 3.24, dibuje un diagrama de instancia utilizando tres películas que se hayan proyectado recientemente. Dibuje las instancias de cada tipo de entidad: PELÍCULAS, ACTORES, PRODUCTORES, DIRECTORES implicados; cree las instancias de las relaciones para estas películas tal y como existen en la realidad.
- 3.30.** Ilustre el diagrama UML para el Ejercicio 3.16. Su diseño UML debe observar los siguientes requisitos:
- a. Un estudiante debe tener la posibilidad de calcular su nota media y añadir o descartar especializaciones principales y secundarias.
 - b. Cada departamento debe ser capaz de añadir o eliminar cursos, y de contratar o despedir profesores.
 - c. Cada profesor debe ser capaz de asignar o cambiar la nota de un estudiante en un curso.
- Nota:* Algunas de estas funciones pueden extenderse por varias clases.

Ejercicios de práctica

- 3.31.** Considere la base de datos UNIVERSIDAD descrita en el Ejercicio 3.16. Cree un esquema ER para ella utilizando una herramienta de modelado de datos como, por ejemplo, ERWin o Rational Rose.
- 3.32.** Considere una base de datos PEDIDOS_CORREO en la que los empleados registran los pedidos de piezas por parte de los clientes. Los requisitos en cuanto a datos son los siguientes:
- La empresa de venta por correo tiene empleados identificados por un número de empleado único, además del nombre, los apellidos y el código postal.
 - Cada cliente de la empresa está identificado mediante un número de cliente único, el nombre, los apellidos y el código postal.
 - Cada pieza o repuesto vendido por la empresa está identificado por un número de repuesto único, un nombre, un precio y la cantidad en stock.
 - Cada pedido efectuado por un cliente es registrado por un empleado y se le asigna un número de pedido que es único. Cada pedido contiene la cantidad especificada de uno o más repuestos, la fecha de recibo y la fecha de envío estimada. También se registra la fecha de envío real.
- Diseñe un diagrama Entidad-Relación para esta base de datos y construya un diseño utilizando una herramienta de modelado de datos como ERWin o Rational Rose.
- 3.33.** Considere una base de datos CINE en la que se registra información relativa a la industria cinematográfica. Los requisitos de datos se resumen a continuación:
- Cada película está identificada por su título y año de proyección. Además, la película tiene una duración en minutos, una productora y está clasificada según uno o más géneros (terror, acción, drama, etcétera). Cada película tiene uno o más directores, y uno o más actores, además de un resumen de la trama. Por último, cada película consta de ninguna o más citas reseñables por parte de los actores que aparecen en ella.
 - Los actores están identificados por su nombre y fecha de nacimiento, y aparecen en una o más películas. Cada actor tiene un papel en la película.

- Los directores también están identificados por su nombre y fecha de nacimiento, y dirigen una o más películas. Es posible que un director también actúe en alguna película (incluyendo alguna que él o ella también haya dirigido).
- Las productoras están identificadas por su nombre y una dirección. Una productora produce una o más películas.

Diseñe un diagrama Entidad-Relación para esta base de datos e introduzca el diseño utilizando una herramienta de modelado de datos como ERWin o Rational Rose.

- 3.34.** Considere el diagrama ER de la base de datos AEROLÍNEA de la Figura 3.20. Cree este diseño mediante una herramienta de modelado de datos como ERWin o Rational Rose.

Bibliografía seleccionada

El modelo Entidad-Relación fue introducido por Chen (1976), y en Schmidt and Swenson (1975), Wiederhold and Elmasri (1979) y Senko (1975) aparecen trabajos relacionados. Desde entonces, se han sugerido numerosas modificaciones para el modelo ER. En nuestra presentación hemos incorporado algunas de ellas. Las restricciones estructurales en las relaciones se explican en Abrial (1974), Elmasri y Wiederhold (1980) y Lenzerini and Santucci (1983). Los atributos multivalor y compuestos se incorporan al modelo ER en Elmasri y otros (1985). Aunque no explicamos lenguajes para el modelo ER y sus extensiones, ha habido varias propuestas para dichos lenguajes. Elmasri and Wiederhold (1981) propuso el lenguaje de consulta más amplio para el modelo ER. Markowitz y Raz (1983) propuso otro lenguaje de consulta ER. Senko (1980) presentó un lenguaje de consulta para el modelo DIAM de Senko. Parent y Spaccapietra (1985) presentó un conjunto formal de operaciones denominado álgebra de ER. Gogolla y Hohenstein (1991) presentó otro lenguaje formal para el modelo ER. En Campbell y otros (1985) se presentó un conjunto de operaciones ER y se mostró que estaban relacionamente completas. Desde 1979 viene celebrándose regularmente una conferencia para la difusión de los resultados de la investigación sobre el modelo ER. La conferencia, ahora conocida como International Conference on Conceptual Modeling, se ha celebrado en Los Ángeles (ER 1979, ER 1983, ER 1997), Washington, D.C. (ER 1981), Chicago (ER 1985), Dijon, Francia (ER 1986), Nueva York (ER 1987), Roma (ER 1988), Toronto (ER 1989), Lausanne, Suiza (ER 1990), San Mateo, California (ER 1991), Karlsruhe, Alemania (ER 1992), Arlington, Texas (ER 1993), Manchester, Inglaterra (ER 1994), Brisbane, Australia (ER 1995), Cottbus, Alemania (ER 1996), Singapur (ER 1998), Salt Lake City, Utah (ER 1999), Yokohama, Japón (ER 2001), Tampere, Finlandia (ER 2002), Chicago, Illinois (ER 2003), Shanghai, China (ER 2004) y Klagenfurt, Austria (ER 2005). La conferencia de 2006 tendrá lugar en Tuscon, Arizona.

CAPÍTULO 4

El modelo Entidad-Relación mejorado (EER)

Los conceptos de modelado ER tratados en el Capítulo 3 son suficientes para representar muchos de los esquemas de bases de datos de las aplicaciones *tradicionales*. Sin embargo, desde finales de los años 70, los diseñadores de aplicaciones de base de datos han intentado crear esquemas que reflejen de un modo más preciso las propiedades y restricciones de los datos. Esto fue algo especialmente importante en las nuevas aplicaciones de tecnología de bases de datos, como las orientadas al diseño de la ingeniería y la fabricación (CAD/CAM)¹, las telecomunicaciones, los sistemas complejos de software y los GIS (Sistemas de información geográfica, *Geographic Information Systems*). Este tipo de bases de datos tienen unos requisitos más complejos que los necesarios en las aplicaciones tradicionales, lo que llevó al desarrollo de nuevos conceptos en la *semántica de modelado de datos* que se incorporaron en modelos de datos conceptuales como el ER. Se han propuesto muchos modelos de semántica de datos. Muchos de estos conceptos fueron desarrollados de forma independiente en otras áreas de la computación, como la **representación del conocimiento** en la inteligencia artificial y el **modelado de objetos** en la ingeniería de software.

En este capítulo se describen algunos de los aspectos que se han propuesto para la semántica de los modelos de datos, y muestra la manera de mejorar el modelo ER para incluir esos conceptos y obtener el modelo **EER (ER mejorado, Enhanced ER)**.² Empezaremos en la Sección 4.1 incorporando los conceptos de relación clase/subclase y de tipo de herencia en el modelo ER.

A continuación, en la Sección 4.2, se incorporarán los conceptos de especialización y generalización.

La Sección 4.3 trata los distintos tipos de restricciones en la especialización/generalización, mientras que la Sección 4.4 muestra la forma de modificar la construcción UNION para incluir el concepto de categoría en el modelo EER. La Sección 4.5 muestra un ejemplo del esquema de bases de datos UNIVERSIDAD en el modelo EER y resume los conceptos de este modelo a través de definiciones formales.

En la Sección 4.6 presentamos la notación del diagrama de la clase UML para la representación de la especialización y la generalización, comparándose brevemente con la notación EER y sus conceptos. Todo esto sirve como ejemplo de notación alternativa, y representa una continuación de la Sección 3.8, en la que se presentó la notación del diagrama de la clase UML básica. En la Sección 4.7 explicamos las abstracciones

¹ CAD/CAM son las siglas de Diseño asistido por computador/Fabricación asistida por computador (*Computer-Aided Design/Computer-Aided Manufacturing*).

² EER también se ha utilizado como siglas para el modelo ER extendido, *Extended ER*.

fundamentales que se emplean como base de muchos de los modelos semánticos de datos. Por último, la Sección 4.8 resume todo el capítulo.

Para entrar en más detalle en el modelado conceptual, el Capítulo 4 debe considerarse como una continuación del 3. Sin embargo, si sólo necesita una breve introducción al modelado ER, puede saltarse este capítulo, aunque también puede optar por omitir algunas secciones del mismo (de la 4.4 a la 4.8).

4.1 Subclases, superclases y herencia

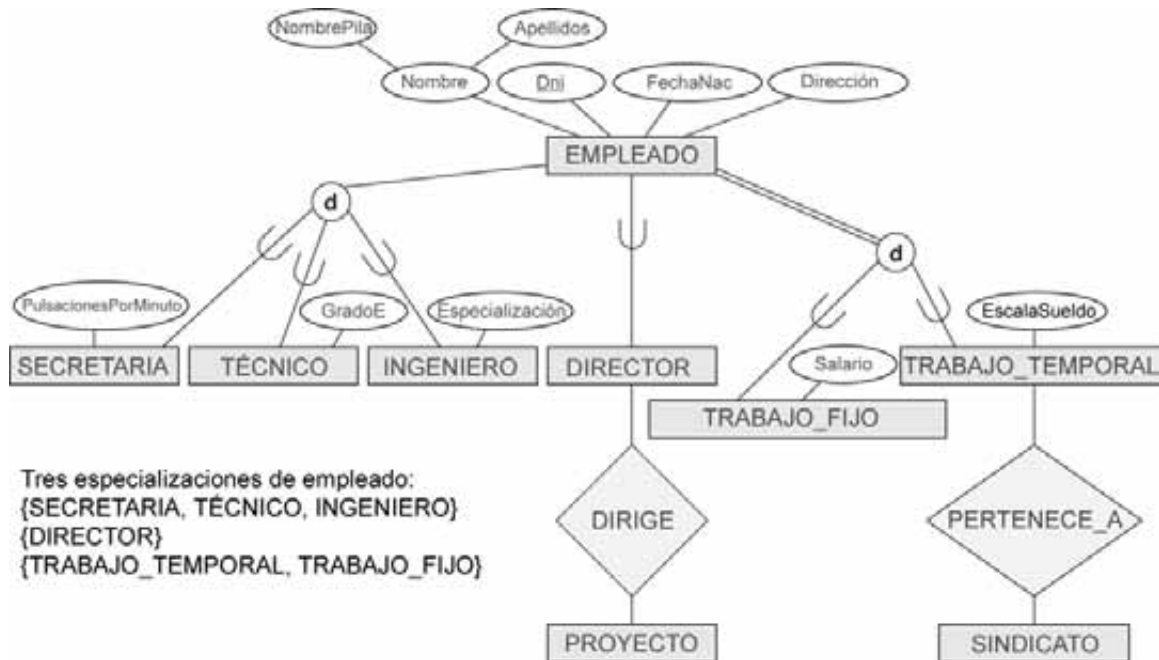
EER contiene todos los conceptos de modelado del modelo ER mostrados en el Capítulo 3, además de incluir la definición de **subclase** y **superclase** y los términos **especialización** y **generalización** (consulte las Secciones 4.2 y 4.3). Otro concepto incluido en el modelo EER es el de **categoría** o **tipo unión** (consulte la Sección 4.4), que se emplea para representar una colección de objetos que es la unión de los objetos de distintos tipos de entidades. Junto a todos estos términos se encuentra el importante mecanismo de **atributo y relación de herencia**. Por desgracia, no existe una terminología estándar para estos conceptos, por lo que utilizaremos la más común (las notas al pie contienen definiciones alternativas). También se describe una técnica de diagramación para mostrar todos estos conceptos cuando confluyen en un esquema EER. Estos esquemas reciben el nombre de **diagramas EER** o **ER mejorado**.

El primer concepto EER que vamos a tratar es el de una **subclase** de un tipo de entidad. Como ya se comentó en el Capítulo 3, un tipo de entidad se emplea para representar tanto a un *tipo de entidad* como al *conjunto de entidades o colección de entidades de ese tipo* que existen en la base de datos. Por ejemplo, el tipo de entidad EMPLEADO describe el tipo (es decir, los atributos y las relaciones) de cada empleado, y hace también referencia al conjunto actual de entidades EMPLEADO de la base de datos EMPRESA. En muchos casos, un tipo de entidad cuenta con varios subgrupos de entidades que son significativos y deben ser representados de forma explícita debido a su importancia en la aplicación de base de datos. Por ejemplo, las entidades que forman parte de EMPLEADO pueden agruparse en SECRETARIA, INGENIERO, DIRECTIVO, TÉCNICO, PERSONAL_FIJO, PERSONAL_TEMPORAL, etc. El conjunto de entidades de cada uno de estos grupos es un subconjunto que pertenece a EMPLEADO, lo que implica que cada una de estas entidades es también un empleado. Podemos decir que cada una de estas agrupaciones es una subclase de la entidad EMPLEADO, mientras que ésta última es la superclase de todas las demás. La Figura 4.1 muestra el modo de representar estos conceptos en forma de diagramas EER (el significado del círculo se explicará en la Sección 4.2).

La relación entre una superclase y una de sus subclases recibe el nombre de **superclase/subclase**, o simplemente **relación clase/subclase**.³ Volviendo a nuestro anterior ejemplo, EMPLEADO/SECRETARIA y EMPLEADO/TÉCNICO son dos relaciones clase/subclase. Observe que una entidad miembro de la subclase representa a la *misma entidad del mundo real* en la superclase; por ejemplo, la entidad SECRETARIA 'Laura Logano' es también el EMPLEADO 'Laura Logano'. Por tanto, el miembro de la subclase es el mismo que la entidad en la superclase, pero en un *papel específico* distinto. Sin embargo, cuando se implementa una relación superclase/subclase en el sistema de bases de datos, podemos representar a un miembro de la subclase como un objeto de base de datos distinto; digamos, un registro diferente que está relacionado a través del atributo clave con su entidad superclase. En la Sección 7.2 se ofrecen varias opciones para representar la relación superclase/subclase en bases de datos relaciones.

Una entidad no puede existir en una base de datos siendo sólo miembro de una subclase; también debe pertenecer a una superclase. Del mismo modo, una entidad puede estar incluida opcionalmente en varias subclases. Por ejemplo, un empleado fijo que también es un ingeniero pertenecerá a las subclases INGENIERO y PERSONAL_FIJO de la entidad EMPLEADO. Sin embargo, no es necesario que cada entidad de una superclase sea miembro de alguna subclase.

³ Una relación clase/subclase suele recibir el nombre de **relación ES-UN/ES-UNA** debido al modo de hacer referencia al concepto. Decimos que una SECRETARIA *es un* EMPLEADO, un TÉCNICO *es un* EMPLEADO, etc.

Figura 4.1. Diagrama EER que representa las subclases y la especialización.

Un concepto importante asociado a las subclases es el de **tipo de herencia**. Recuerde que el *tipo* de una entidad está definido por los atributos que posee y los tipos de relación en los que participa. Ya que una entidad en una subclase representa también a la misma persona en la superclase, debe disponer de valores para sus atributos específicos, *así como* otros como miembro de la superclase. Decimos, por tanto, que una entidad que es miembro de una subclase **hereda** todos los atributos de la entidad como miembro de la superclase y las relaciones en las que ésta participa. Observe que una subclase, con sus atributos y relaciones propias (o locales) junto con los que hereda de la superclase, puede ser considerada como un *tipo de entidad* por derecho propio.⁴

4.2 Especialización y generalización

4.2.1 Especialización

La **especialización** es el proceso de definir un *conjunto de subclases* de un tipo de entidad, la cual recibe el nombre de **superclase** de la especialización. El conjunto de subclases que forman una especialización se define basándose en algunas características distintivas de las entidades en la superclase. Por ejemplo, las subclases {SECRETARIA, INGENIERO, TÉCNICO} son una especialización de la superclase EMPLEADO que distingue a los trabajadores en función al *tipo de trabajo* que desempeñan. Podemos tener varias especializaciones del mismo tipo de entidad en función de varias características distintivas. Por ejemplo, otra especialización de EMPLEADO podría ser el conjunto de subclases {PERSONAL_FIJO, PERSONAL_TEMPORAL}, las cuales distinguen a los trabajadores por el *tipo de contrato que tienen*.

⁴ En algunos lenguajes de programación orientados a objetos, existe una restricción común que dice que una entidad (u objeto) *sólo tiene un tipo*. En general, este planteamiento es demasiado restrictivo para cualquier modelo de base de datos.

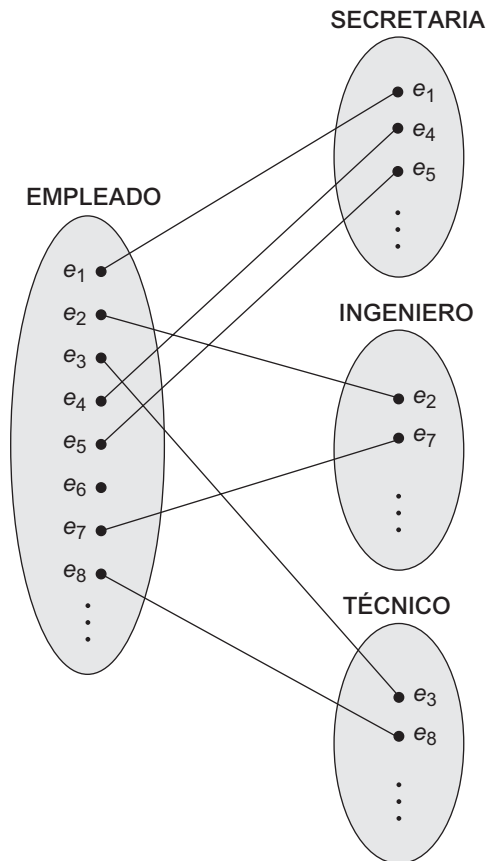
La Figura 4.1 muestra la forma de representar una especialización en un diagrama EER. Las subclases que la definen están unidas por líneas a un círculo que representa la especialización, la cual está a su vez unida a la superclase.

El *símbolo de subconjunto* de cada línea que conecta una subclase al círculo indica la dirección de la relación superclase/subclase.⁵ Los atributos que se aplican únicamente a las entidades de una subclase particular (como las PulsacionesPorMinuto de una SECRETARIA) están unidas al rectángulo que la representa, y reciben el nombre de **atributos específicos** (o **atributos locales**) de la subclase. Además, una subclase puede participar de **tipos de relación específicos**, como ocurre en la Figura 4.1 con la subclase PERSONAL_TEMPORAL que participa de la relación PERTENECE_A. El significado del símbolo **d** que aparece en los círculos de dicha figura, y en otros diagramas EER, se explicará a continuación.

La Figura 4.2 muestra algunas instancias que pertenecen a las subclases de la especialización {SECRETARIA, INGENIERO, TÉCNICO}. Observe de nuevo que una entidad que pertenece a una subclase representa a la misma persona que la entidad conectada a ella en la superclase EMPLEADO, aun cuando se muestre dos veces; por ejemplo, *e1* aparece en la Figura 4.2 como EMPLEADO y SECRETARIA.

Tal y como sugiere la figura, una relación superclase/subclase como EMPLEADO/SECRETARIA se asemeja en cierto modo a otra del tipo 1:1 a *nivel de instancia* (véase la Figura 3.12). La diferencia principal es que

Figura 4.2. Instancias de una especialización.



⁵ Existen notaciones alternativas para las especializaciones; mostraremos la UML en la Sección 4.6 y otros tipos en el Apéndice A.

en una relación 1:1, dos *entidades distintas* están relacionadas, mientras que en el caso de una superclase/subclase, la entidad de la subclase es la misma que la de la superclase pero desempeñando un *papel concreto* (por ejemplo, un EMPLEADO especializado en el papel de SECRETARIA o de TÉCNICO).

Existen dos razones principales para incluir relaciones clase/subclase y especializaciones en un modelo de datos. La primera es que pueden existir ciertos atributos que sólo deban aplicarse a algunas entidades, pero no a toda la superclase. Por tanto, la subclase se define para agrupar a todas esas entidades. Los integrantes de la subclase pueden seguir compartiendo la mayor parte de sus atributos con los otros miembros de la superclase. Por ejemplo, en la Figura 4.1, la subclase SECRETARIA cuenta con el atributo específico PulsacionesPorMinuto, mientras que INGENIERO dispone de otro llamado Especialización, aunque ambos comparten los atributos heredados de la entidad EMPLEADO.

El segundo motivo para usar subclases es que algunos tipos de relaciones sólo pueden establecerse entre miembros de esa subclase. Por ejemplo, si sólo el PERSONAL_TEMPORAL puede estar afiliado a un sindicato, podemos representar esta circunstancia creando la subclase PERSONAL_TEMPORAL de EMPLEADO y relacionarla con una entidad SINDICATO a través de la relación PERTENECE_A, tal y como puede verse en la Figura 4.1.

En resumen, el proceso de especialización no permite hacer lo siguiente:

- Definir un conjunto de subclases de un tipo de entidad.
- Establecer atributos específicos adicionales en cada subclase.
- Establecer relaciones específicas adicionales entre cada subclase y otras entidades u otras subclases.

4.2.2 Generalización

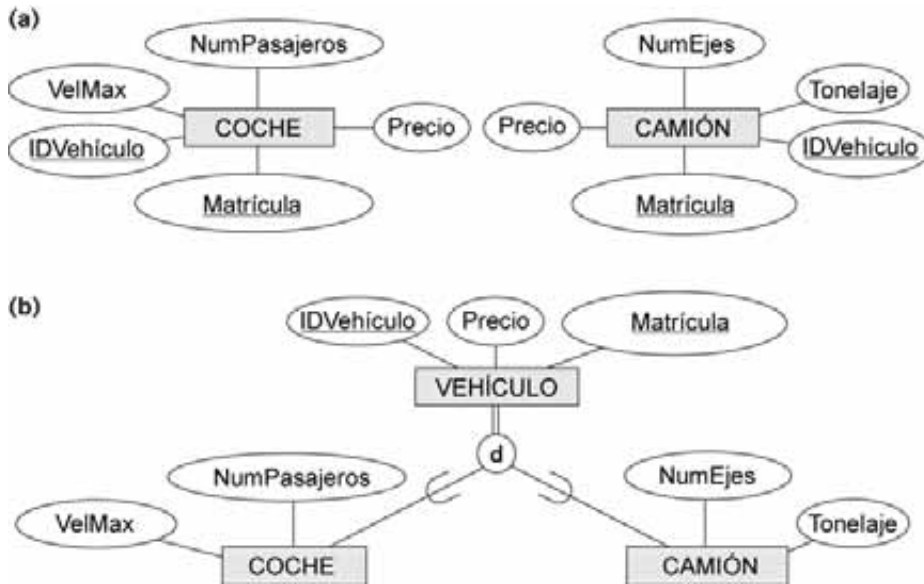
Podemos pensar en un *proceso inverso* de abstracción en el que eliminemos las diferencias existentes entre distintas entidades, identifiquemos las características comunes y las **generalicemos** en una única **superclase** de la que las entidades originales sean **subclases** especiales. Por ejemplo, consideremos las entidades COCHE y CAMIÓN mostradas en la Figura 4.3(a). Ya que ambas cuentan con características comunes, podrían generalizarse en la entidad VEHÍCULO (véase la Figura 4.3[b]). Tanto COCHE como CAMIÓN son ahora subclases de la **superclase generalizada** VEHÍCULO. Usamos el término **generalización** para referirnos al proceso por el cual se define una entidad generalizada a partir de entidades individuales.

Observe que la generalización puede considerarse como el proceso inverso de la especialización desde un punto de vista funcional. Por tanto, en la Figura 4.3 podemos decir que {COCHE, CAMIÓN} son una especialización VEHÍCULO, en lugar de considerar a VEHÍCULO como una generalización de COCHE y CAMIÓN. De forma análoga, en la Figura 4.1 podemos ver a EMPLEADO como una generalización de SECRETARIA, TÉCNICO e INGENIERO. En algunas metodologías de diagramación existen distintos elementos para distinguir una generalización de una especialización. Una flecha que apunta a la superclase generalizada representa una generalización, mientras que cuando lo hace hacia las subclases especializadas indica una especialización.

No utilizaremos esta notación ya que la decisión sobre qué proceso es más apropiado en una situación suele ser algo bastante subjetivo. El Apéndice A muestra algunas alternativas para la representación de diagramas de esquema y de diagramas de clase.

Hasta ahora hemos visto los conceptos de subclase y de relación superclase/subclase, así como los procesos de especialización y generalización. En general, una superclase o una subclase representan una colección de entidades del mismo tipo que, por consiguiente, también describe un *tipo de entidad*; ésta es la razón por la que ambos elementos aparecen como rectángulos en los diagramas EER. A continuación, vamos a tratar con más detalle las propiedades de las especializaciones y las generalizaciones.

Figura 4.3. Generalización. (a) Dos entidades, COCHE y CAMIÓN. (b) Generalizando COCHE y CAMIÓN en la superclase VEHÍCULO.



4.3 Restricciones y características de las jerarquías de especialización y generalización

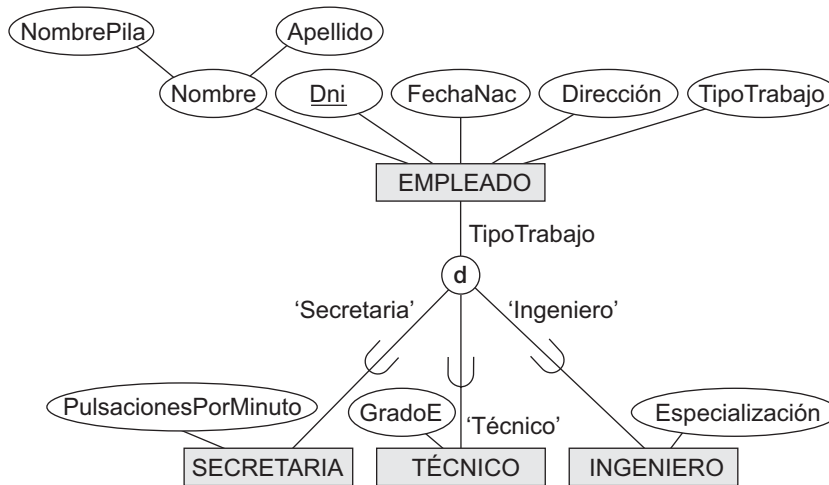
En primer lugar, trataremos las restricciones que se aplican a una única especialización o generalización. Por brevedad, la explicación sólo hará referencia a la *especialización* aun cuando ésta se aplique a ambos términos. A continuación, nos centraremos en las diferencias existentes entre los *entramados* (*lattices*) (*herencia múltiple*) y las *jerarquías* (*herencia sencilla*) de especialización/generalización, y elaboraremos las principales diferencias existentes entre ambos procesos durante el diseño de un esquema de base de datos conceptual.

4.3.1 Restricciones en la especialización y la generalización

En general, podemos contar con varias especializaciones definidas en la misma entidad (o superclase), como puede verse en la Figura 4.1. En este caso, las entidades pueden pertenecer a las subclases de cada una de las especializaciones. Sin embargo, una especialización también puede contar con una *única* subclase, como ocurre en el caso de {DIRECTOR} de la Figura 4.1; en esta situación, no utilizamos el círculo correspondiente.

En algunas especializaciones, podemos determinar con exactitud las entidades que se convertirán en miembros de cada subclase situando una condición en el valor de algunos atributos de la superclase. Estas subclases reciben el nombre de **subclases de predicado definido** (o de **condición definida**). Por ejemplo, si la entidad EMPLEADO cuenta con un atributo TipoTrabajo, tal y como puede verse en la Figura 4.4, podemos especificar la pertenencia a la subclase SECRETARIA mediante la condición (TipoTrabajo = 'Secretaria'), es decir, **definiendo el predicado** de la subclase. Esta condición es una restricción específica que nos permite decir que aquellas entidades de EMPLEADO cuyo valor para el atributo TipoTrabajo sea 'Secretaria' pertenecen a esa subclase. Identificamos una subclase de predicado definido escribiendo la condición a continuación de la línea que conecta la subclase al círculo de especialización.

Si *todas* las subclases de una especialización tienen su condición de pertenencia en el *mismo* atributo de la superclase, la propia especialización recibe el nombre de **especialización de atributo definido**, y el atributo

Figura 4.4. Indicación en un diagrama EER de una especialización de atributo definido para TipoTrabajo.

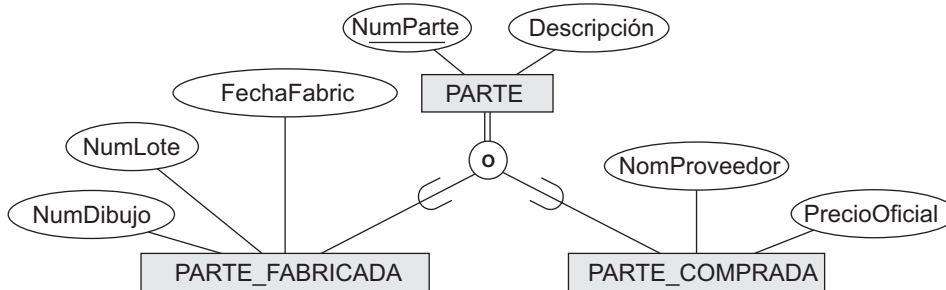
recibe el nombre de **atributo definitorio** de la especialización.⁶ Identificamos una especialización de atributo definido colocando el nombre del mismo al lado del arco que va desde el círculo a la superclase, tal y como puede verse en la Figura 4.4.

Cuando no tenemos una condición para determinar los miembros de una subclase, se dice que es de tipo **definido por usuario**. Los miembros de este tipo de subclase son determinados por los usuarios de la base de datos cuando aplican la operación para añadir una entidad a la subclase; así pues, los miembros son *especificados individualmente por el usuario para cada entidad*, y no por una condición que pueda evaluarse automáticamente.

Existen otras dos restricciones que pueden aplicarse a una especialización. La primera es la **restricción de disyunción** (*disjointness*), la cual especifica que las subclases de la especialización deben estar separadas. Esto significa que una entidad puede ser, como máximo, miembro de una de las subclases de la especialización. Una especialización de tipo atributo-definido implica la restricción de disyunción en el caso de que el atributo utilizado para definir el predicado de agrupación sea de un solo valor, o monovalor. La Figura 4.4 ilustra este caso, donde la **d** incluida en el círculo simboliza la *separación*. También se utiliza este símbolo para especificar que las subclases de una especialización definidas por el usuario deben ser del mismo tipo, tal y como puede verse en el ejemplo {PERSONAL_FIJO, PERSONAL_TEMPORAL} de la Figura 4.1. Si las subclases no están obligadas a estar separadas, su conjunto de entidades pueden **solaparse**, es decir, la misma entidad podría ser miembro de más de una subclase de la especialización. Este caso, que es el que se produce por defecto, se especifica colocando una **o** en el círculo, tal y como puede verse en la Figura 4.5.

La segunda restricción de una especialización se conoce como **restricción de integridad**, la cual puede ser total o parcial. Una **especialización total** especifica que *cada* entidad en la superclase debe ser miembro de, al menos, una subclase en la especialización. Por ejemplo, si cada EMPLEADO debe ser PERSONAL_FIJO o PERSONAL_TEMPORAL, entonces la especialización {PERSONAL_FIJO, PERSONAL_TEMPORAL} de la Figura 4.1 es una especialización total de EMPLEADO. Esto se muestra en los diagramas EER usando una línea doble que conecta la superclase al círculo. Para mostrar una **especialización parcial** se emplea una línea sencilla, lo que permite que una entidad no pertenezca a ninguna de las subclases. Por

⁶ En terminología UML, este tipo de atributo se conoce como *discriminador*.

Figura 4.5. Indicación de una especialización de solapamiento (*nondisjoint*) en un diagrama EER.

ejemplo, si alguna entidad EMPLEADO no está incluida en ninguna de las subclases {SECRETARIA, INGENIERO, TÉCNICO} de las Figuras 4.1 y 4.4, entonces esa especialización es parcial.⁷

Observe que las restricciones de disyunción y de integridad son *independientes*. Por consiguiente, son posibles las cuatro siguientes restricciones en una especialización:

- Disyunción, total.
- Disyunción, parcial.
- Solapamiento, total.
- Solapamiento, parcial.

Desde luego, la restricción correcta viene determinada por la aplicación real que se le quiera dar a cada especialización. Por lo general, una superclase identificada a través del proceso de *generalización* es **total**, ya que está *derivada a partir* de las subclases y, por consiguiente, sólo contiene las entidades incluidas en ellas.

Existen ciertas reglas de inserción y borrado que se aplican a una especialización (y una generalización) como consecuencia de las restricciones indicadas anteriormente. Éstas son algunas de esas reglas:

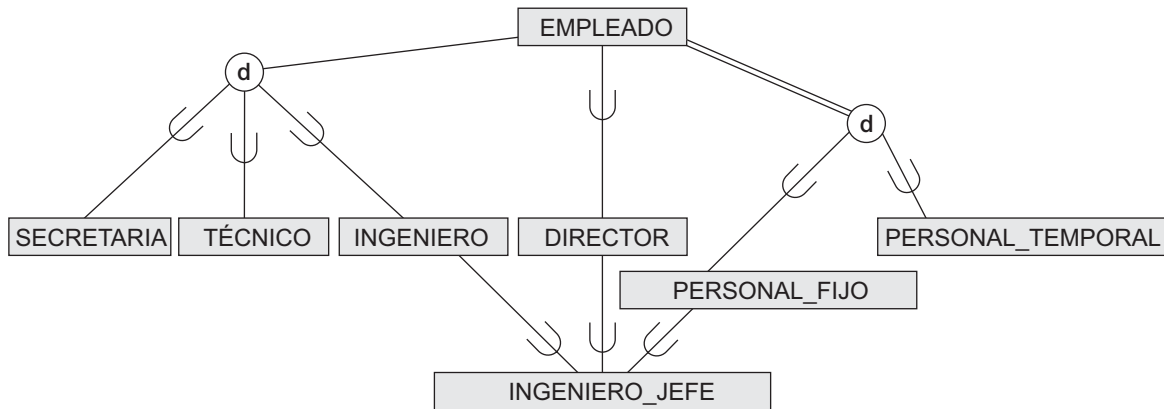
- El borrado de una entidad de una superclase implica su eliminación automática de todas las subclases a las que pertenece.
- La inserción de una entidad en una superclase supone que la misma debe insertarse en todas las subclases de *predicado definido* (o *atributo definido*) en las que esa entidad cumpla la regla.
- La inserción de una entidad en una superclase de *especialización total* conlleva que dicha entidad sea incluida obligatoriamente en, al menos, una de las subclases de la especialización.

Se exhorta al lector a realizar una lista completa de reglas de inserción y borrado para los distintos tipos de especialización.

4.3.2 Jerarquías y entramados de especialización y generalización

Una subclase, por si misma, puede tener más subclases definidas en ella formando una jerarquía (o entramado) de especializaciones. Por ejemplo, en la Figura 4.6, INGENIERO es una subclase de EMPLEADO y, a su vez, una superclase de INGENIERO_JEFE; esto representa una restricción real que dice que todo ingeniero jefe debe ser ingeniero. Una **especialización jerárquica** tiene una restricción que dice que cada subclase participa como tal en *una única* relación clase/subclase, es decir, cada subclase sólo tiene un padre, lo que deriva en la formación de una estructura en árbol. En contraposición, en una **especialización entramada**, una subclase puede serlo en *más de una* relación clase/subclase. Por tanto, la Figura 4.6 es un entramado.

⁷ El uso de líneas dobles o sencillas es similar a lo que sucede con la participación parcial o total de un tipo de entidad en un tipo de relación, tal y como se describió en el Capítulo 3.

Figura 4.6. Un entramado de especialización con una subclase INGENIERO_JEFE compartida.

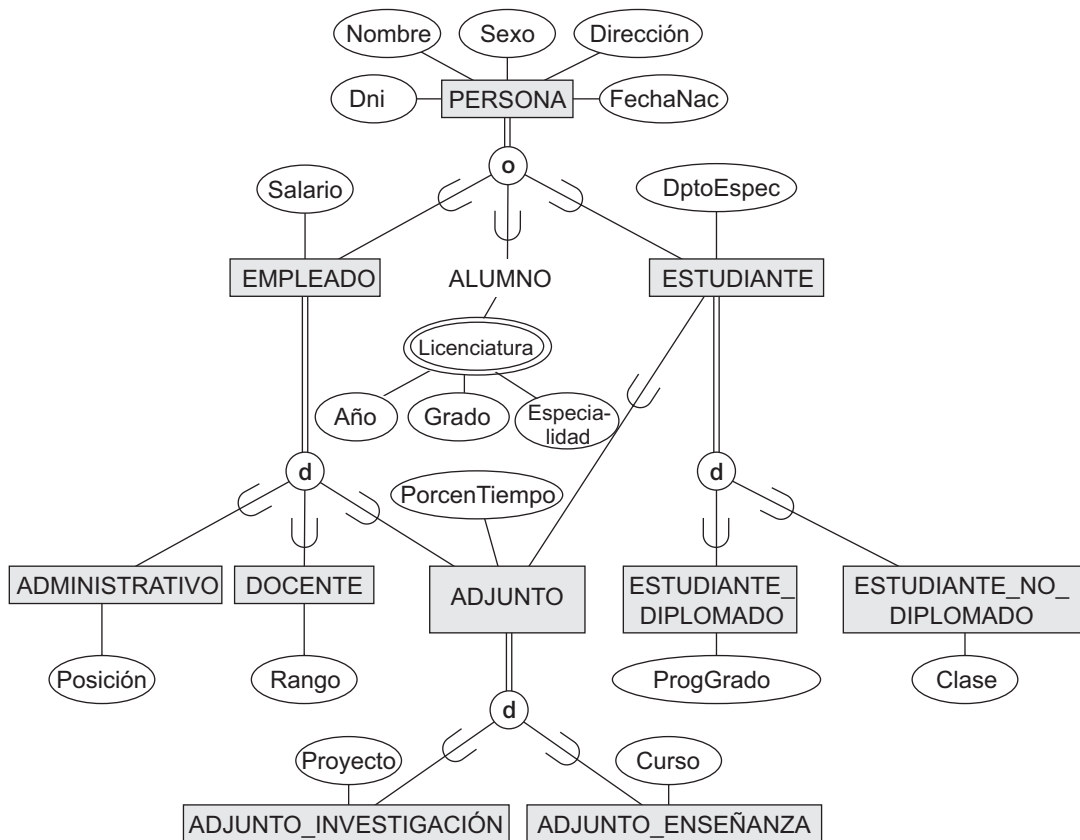
La Figura 4.7 muestra otro entramado de especialización de más de un nivel. Esto puede formar parte del esquema conceptual de una base de datos UNIVERSIDAD. Observe que esta disposición podría ser también una jerarquía excepto por la subclase ADJUNTO_ENSEÑANZA, la cual forma parte de dos relaciones clase/subclase diferentes.

Los requisitos de la base de datos UNIVERSIDAD de la Figura 4.7 son los siguientes:

1. La base de datos mantiene tres tipos de personas: empleados, ex alumnos y estudiantes. Una persona puede pertenecer a uno, dos o los tres tipos, y dispone de un nombre, un DNI, su sexo, su dirección y fecha de nacimiento.
2. Cada empleado dispone de un salario, y se agrupan en tres categorías distintas: personal docente, administrativos y adjuntos. Cada trabajador pertenece a uno de los tres grupos. Para los antiguos alumnos, se mantiene un registro con la titulación máxima conseguida, incluyendo el nombre de dicha titulación, el año de obtención y la especialidad.
3. Cada profesor tiene un rango, mientras que el personal administrativo cuenta con una posición. Los adjuntos están clasificados como asistentes de investigación o de enseñanza, registrándose en la base de datos el porcentaje de tiempo que trabajan, así como sus proyectos de investigación (los primeros) o el curso que imparten (los segundos).
4. Los estudiantes están clasificados como diplomados o como estudiantes propiamente dichos, con los atributos específicos del programa de grado (M.S., Ph.D., M.B.A., etc.) o clase (estudiante de primer año, estudiante de segundo año, etc.), respectivamente.

En la Figura 4.7, todas las personas representadas en la base de datos son miembros de la entidad PERSONA, la cual está especializada en las subclases {EMPLEADO, EX_ALUMNO, ESTUDIANTE}. Esta especialización está solapada; por ejemplo, un ex alumno puede ser también un empleado o un estudiante cursando un nivel más avanzado. La subclase ESTUDIANTE es la superclase de la especialización {ESTUDIANTE_DIPLOMADO, ESTUDIANTE_NO_DIPLOMADO}, mientras que EMPLEADO lo es de {ADJUNTO, DOCENTE, ADMINISTRATIVO}. Observe que ADJUNTO es también una subclase de ESTUDIANTE. Por último, ADJUNTO es la superclase de la especialización {ADJUNTO_INVESTIGACIÓN, ADJUNTO_ENSEÑANZA}.

Independientemente de si se trata de una jerarquía o un entramado de especialización, una subclase hereda los atributos no sólo de su superclase directa, sino también de todas sus superclases predecesoras hasta el tope de la jerarquía o el entramado. Por ejemplo, una entidad en ESTUDIANTE_DIPLOMADO hereda todos los atributos de ESTUDIANTE y PERSONA. Observe que una entidad puede existir en varios *nodos hoja* de la jerarquía, donde un **nodo hoja** es una clase que *no tiene subclases*. Por ejemplo, un miembro de ESTUDIANTE_DIPLOMADO puede serlo también de ADJUNTO_INVESTIGACIÓN.

Figura 4.7. Un entramado de especialización con herencia múltiple para una base de datos UNIVERSIDAD.

Una subclase que cuente con *más de una* superclase recibe el nombre de **subclase compartida**, como ocurre con **INGENIERO_JEFE** en la Figura 4.6. Esto nos lleva al concepto conocido como **herencia múltiple**, en donde la subclase compartida **INGENIERO_JEFE** hereda directamente atributos y relaciones de varias clases. Observe que la existencia de al menos una subclase compartida nos lleva a un entramado (y, por consiguiente, a una *herencia múltiple*); si no existieran subclases compartidas, tendríamos una jerarquía en lugar de un entramado. La subclase compartida **ADJUNTO_ENSEÑANZA** de la Figura 4.7, que hereda atributos tanto de **EMPLEADO** como de **ESTUDIANTE**, puede ilustrar una regla importante relacionada con la herencia múltiple. En este caso, **EMPLEADO** y **ESTUDIANTE** heredan los *mismos atributos* de **PERSONA**. La regla establece que si un atributo (o relación) que se origina en la *misma superclase* (**PERSONA**) es heredado más de una vez a través de caminos diferentes (**EMPLEADO** y **ESTUDIANTE**) en el entramado, sólo podrá incluirse una vez en la subclase compartida (**ADJUNTO_ENSEÑANZA**). Por tanto, los atributos de **PERSONA** sólo se heredan una vez en la subclase **ADJUNTO_ENSEÑANZA** de la Figura 4.7.

Es importante indicar que algunos modelos y lenguajes *no permiten* la herencia múltiple (subclases compartidas). En un modelo de este tipo es necesario crear subclases adicionales que cubran todas las posibles combinaciones de clases en las que una entidad pertenezca simultáneamente a todas ellas. Por tanto, cualquier especialización de solapamiento precisará de múltiples subclases adicionales. Por ejemplo, el solapamiento de **PERSONA** en {**EMPLEADO**, **EX_ALUMNO**, **ESTUDIANTE**} (o {**E**, **A**, **S**} para abreviar, según el inglés), precisaría de la creación de siete subclases de **PERSONA** que cubrieran todos los posibles tipos de entidades: **E**, **A**, **S**, **E_A**, **E_S**, **A_S**, y **E_A_S**. Obviamente, esto conlleva una complejidad añadida.

Es importante indicar también que algunos mecanismos de herencia que permiten la herencia múltiple no dejan que una entidad tenga varios tipos, lo que implica que sólo puede ser miembro de *una única clase*.⁸ En modelos de este tipo, es necesario crear también subclases compartidas adicionales como nodos hoja que abarquen todas las posibles combinaciones de clases.

Aunque hemos empleado la especialización para explicar el tema, se pueden aplicar los mismos conceptos en la generalización, por lo que podríamos hablar de **jerarquía de generalización** y de **entramado de generalización**.

4.3.3 Utilización de la especialización y la generalización en el refinamiento de los esquemas conceptuales

Ahora profundizaremos en las diferencias entre los procesos de especialización y de generalización, y en cómo deben usarse para refinar esquemas conceptuales durante el diseño de bases de datos conceptuales. En el proceso de especialización, partimos de una entidad para definir a continuación subclases de la misma a través de especializaciones sucesivas, esto es, definimos de forma repetida agrupaciones más específicas de la entidad principal. Por ejemplo, cuando se diseña el entramado de especialización de la Figura 4.7, podríamos empezar especificando una entidad PERSONA en una base de datos universitaria. A continuación, descubrimos que existirán tres tipos de personas representados en dicha base de datos: trabajadores, ex alumnos y estudiantes. Para ello, creamos la especialización {EMPLEADO, EX_ALUMNO, ESTUDIANTE} y elegimos la restricción de solapamiento porque una persona puede pertenecer a más de una de estas subclases. Más adelante, especializamos EMPLEADO en {ADMINISTRATIVO, DOCENTE, ADJUNTO} y ESTUDIANTE en {ESTUDIANTE_DIPLOMADO, ESTUDIANTE_NO_DIPLOMADO}. Por último, ADJUNTO se especializa en {ADJUNTO_INVESTIGACIÓN, ADJUNTO_ENSEÑANZA}. Esta diversificación progresiva se corresponde con un **proceso de refinamiento conceptual de arriba abajo**.

Hasta aquí, tenemos una jerarquía; ahora nos damos cuenta de que ADJUNTO_ENSEÑANZA es una subclase compartida, ya que también es una subclase de ESTUDIANTE, lo que hace que tengamos un entramado.

Es posible llegar a la misma jerarquía o entramado desde otra dirección. En un caso como éste, el proceso implica llevar a cabo una generalización en lugar de una especialización y se corresponde con una **síntesis conceptual de abajo arriba**. Esta situación obliga, en primer lugar, a que los diseñadores descubran entidades del tipo ADMINISTRATIVO, DOCENTE, EX_ALUMNO, ESTUDIANTE_DIPLOMADO, ESTUDIANTE_NO_DIPLOMADO, ADJUNTO_INVESTIGACIÓN, ADJUNTO_ENSEÑANZA, etc.; a continuación, deben generalizar {DIPLOMADO, ESTUDIANTE_NO_DIPLOMADO} a ESTUDIANTE; {ADJUNTO_INVESTIGACIÓN, ADJUNTO_ENSEÑANZA} a ADJUNTO; {ADMINISTRATIVO, DOCENTE, ADJUNTO} a EMPLEADO; y, por último, {EMPLEADO, EX_ALUMNO, ESTUDIANTE} a PERSONA.

En términos estructurales, las jerarquías o entramados resultantes de este proceso pueden ser idénticas; la única diferencia radica en la manera, o el orden, en el que se especifica el esquema de superclases y subclases. En la práctica, es muy probable que no siga de manera estricta ni el proceso de generalización ni el de especialización, sino que se emplee una combinación de ambos. En este caso, existe una continua incorporación de nuevas clases a la jerarquía o el entramado a medida que el proceso se hace más claro a los usuarios y los diseñadores. Tenga en cuenta que la noción de representación de datos y conocimiento usando jerarquías o entramados de superclase/subclase es muy común en sistemas del conocimiento y sistemas expertos, los cuales combinan tecnología de bases de datos con técnicas de inteligencia artificial. Por ejemplo, los esquemas de representación del conocimiento basados en *frames* tienen un gran parecido con las jerarquías de clase. La especialización es también muy común en las metodologías de diseño de ingeniería de software basadas en el paradigma de la orientación a objetos.

⁸ En algunos modelos, la clase está obligada a ser un *nodo hoja* en la jerarquía o el entramado.

4.4 Modelado de tipos UNION usando categorías

Todas las relaciones superclase/subclase que hemos visto hasta ahora tenían una *sola superclase*. Una subclase compartida como INGENIERO_JEFE en el entramado de la Figura 4.6 es la subclase en tres relaciones superclase/subclase *distintas*, donde cada una de estas tres relaciones cuenta con una *única* superclase. Sin embargo, no resulta extraño que la necesidad obligue a modelar una única relación superclase/subclase con *más de una* superclase, donde esas superclases representen diferentes tipos de entidades. En este caso, la subclase representará una colección de objetos que es un subconjunto de la UNION de distintos tipos de entidades; llamaremos a esta *subclase* un **tipo unión** o una **categoría**.⁹

Por ejemplo, supongamos que tenemos tres entidades: PERSONA, BANCO y EMPRESA. En una base de datos de registro de vehículos, el propietario de uno de ellos puede ser una persona física, un banco (que lo haya adquirido mediante un embargo) o una empresa. Para desempeñar el papel de *propietario de un vehículo*, necesitamos crear una clase (colección de entidades) que incluya estos tres tipos. Para ello, se crea una categoría PROPIETARIO que sea una *subclase de la UNION* de los tres conjuntos de entidades. En la Figura 4.8 puede verse el modo de mostrar las categorías en un diagrama EER. Las superclases EMPRESA, BANCO y PERSONA se conectan al círculo mediante un punto (.). Un arco con el símbolo de subconjunto conecta el círculo a la categoría (subclase) PROPIETARIO.

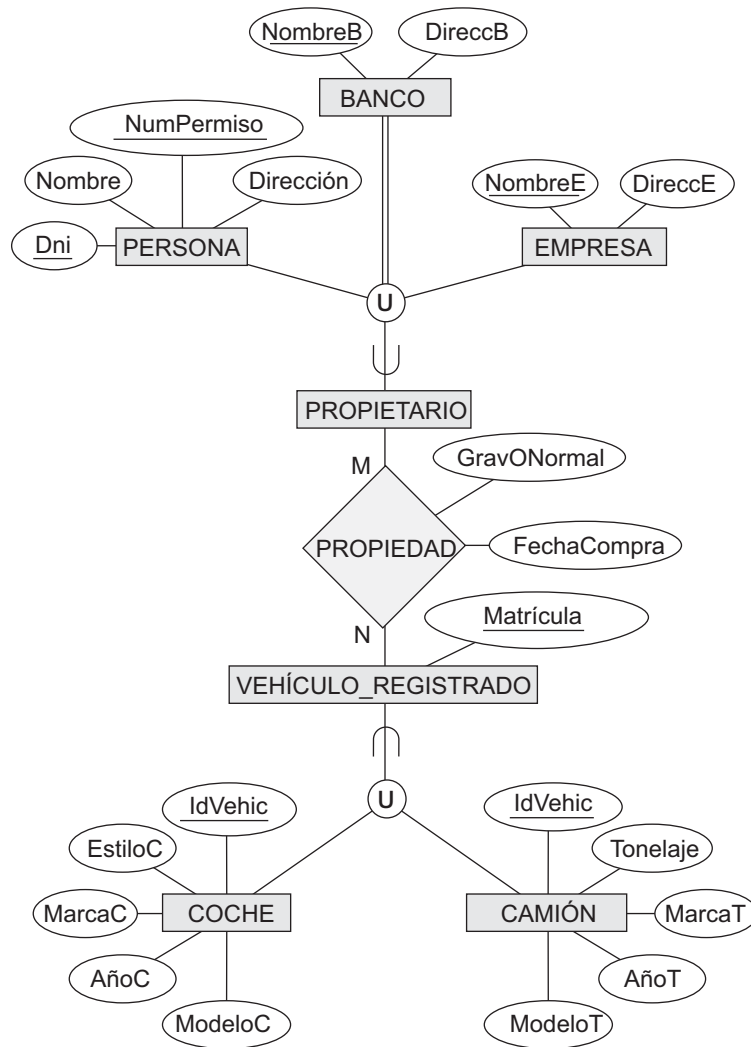
En caso de necesitarse un predicado, éste aparece a continuación de la línea que sale de la superclase a la cual debe aplicarse. En la Figura 4.8 tenemos dos categorías: PROPIETARIO, que es una subclase de la unión de PERSONA, BANCO y EMPRESA, y VEHÍCULO_REGISTRADO, que es una subclase de la unión de COCHE y CAMIÓN.

Una categoría tiene dos o más superclases que pueden representar *distintos tipos de entidades*, mientras que las otras relaciones superclase/subclase siempre tienen una sola superclase. Podemos comparar una categoría, como PROPIETARIO en la Figura 4.8, con la subclase compartida INGENIERO_JEFE de la Figura 4.6. El resultado es una subclase de *cada una* de las tres superclases INGENIERO, JEFE y PERSONAL_FIJO, por lo que una entidad que fuera miembro de INGENIERO_JEFE debería existir en las *tres*. Esto supone la aplicación de la restricción de que un ingeniero jefe debe ser un INGENIERO, un JEFE y PERSONAL_FIJO, es decir, INGENIERO_JEFE es un subconjunto de la *intersección* de las tres subclases (conjuntos de entidades). En el otro extremo, una categoría es un subconjunto de la *unión* de sus superclases. Por tanto, una entidad miembro de PROPIETARIO debe aparecer *sólo* en una de las superclases. Esto supone la aplicación de la restricción de que un PROPIETARIO puede ser una EMPRESA, un BANCO o una PERSONA en la Figura 4.8.

La herencia de atributo funciona de manera más selectiva en el caso de las categorías. Por ejemplo, en la Figura 4.8, cada entidad PROPIETARIO hereda los atributos de una EMPRESA, una PERSONA o un BANCO dependiendo de la superclase a la que pertenezca esa entidad. Por otro lado, una subclase compartida como INGENIERO_JEFE (Figura 4.6) hereda *todos* los atributos de sus superclases PERSONAL_FIJO, INGENIERO y JEFE.

Es interesante observar la diferencia existente entre la categoría VEHÍCULO_REGISTRADO (Figura 4.8) y la superclase generalizada VEHÍCULO (Figura 4.3[b]). En dicha figura, cada coche y cada camión es un VEHÍCULO; sin embargo, en la Figura 4.8, la categoría VEHÍCULO_REGISTRADO sólo incluye algunos coches y algunos camiones, pero no necesariamente todos ellos (por ejemplo, algunos de estos vehículos pueden no estar registrados). En general, una especialización o una generalización como la de la Figura 4.3(b), si fuera *parcial*, podrían no evitar que VEHÍCULO contuviera otros tipos de entidades, como motocicletas. Sin embargo, una categoría como VEHÍCULO_REGISTRADO de la Figura 4.8 implica que sólo coches y camiones, pero no otro tipo de entidad, pueden ser miembros de ella.

⁹ Nuestro uso del término *categoría* está basado en el modelo ECR (Relación Entidad-Categoría, *Entity-Category-Relationship*) (Elmasri y otros, 1985).

Figura 4.8. Dos categorías (tipo unión): PROPIETARIO y VEHÍCULO_REGISTRADO.

Una categoría puede ser **total** o **parcial**. Una categoría total contiene la *unión* de todas las entidades de sus superclases, mientras que una parcial puede almacenar un *subconjunto de la unión*. Una categoría total está representada por una línea doble que la conecta con el círculo, en tanto que una parcial está indicada por una línea sencilla.

Las superclases de una categoría pueden tener diferentes atributos clave, como ya ha quedado demostrado en la categoría **PROPIETARIO** de la Figura 4.8, o tener el mismo, como ocurre con **VEHÍCULO_REGISTRADO**. Observe que si una categoría es total (no parcial), puede estar representada alternativamente como una especialización total (o una generalización total). En este caso, la elección del tipo de representación a usar es algo subjetivo. Si las dos clases representan al mismo tipo de entidades y comparten numerosos atributos, incluyendo los clave, es preferible la especialización/generalización; en cualquier otro caso, es más apropiado decantarse por la categorización (tipo unión).

Es interesante hacer notar que en algunos modelos (consulte el Capítulo 20), todos los objetos son especializaciones de una única clase raíz. En un modelo de este tipo, es posible modelar los tipos de unión usando la especialización apropiada.

4.5 Ejemplo EER de un esquema UNIVERSIDAD, diseños y definiciones formales

En esta sección empezaremos por dar un ejemplo de un esquema de base de datos en el modelo EER para ilustrar el uso de los distintos conceptos mostrados en este capítulo y en el anterior. A continuación, trataremos el tema del diseño para los esquemas conceptuales y terminaremos resumiendo los conceptos del modelo EER y los definiremos formalmente de la misma manera que hicimos con los del modelo ER en el Capítulo 3.

4.5.1 La base de datos UNIVERSIDAD

Para nuestro ejemplo, consideremos la base de datos UNIVERSIDAD que contiene información sobre los estudiantes y sus especialidades, traslados y registros, así como los cursos ofrecidos por la misma. Esta base de datos también almacena los proyectos de investigación patrocinados por la universidad y los estudiantes graduados. Este esquema se muestra en la Figura 4.9.

Para cada persona, la base de datos mantiene su [Nombre], [DNI], [Dirección], [Sexo] y [FechaNac]. Hay definidas dos subclases de la entidad PERSONA: PROFESOR y ESTUDIANTE. Los atributos específicos de PROFESOR son [Rango] (asistente, asociado, adjunto, investigador, etc.), [Oficina], [TlfOficina] y [Salario]. Todos los profesores están relacionados con el/los departamento/s a los que pertenecen: [PERTENECE] (un profesor puede estar asociado a varios de ellos, por lo que la relación es M:N). Un atributo específico de ESTUDIANTE es [Clase] (estudiante de primer año=1, estudiante de segundo año=2, . . . , graduado=5). Cada ESTUDIANTE está también relacionado con su especialidad y su formación secundaria, ([PRINCIPAL] y [SECUNDARIA]), de los niveles del curso en el que actualmente están matriculados [REGISTRADO], y de los cursos completados, [CERTIFICADO]. Cada instancia CERTIFICADO incluye la [Nota] que el estudiante recibe al completar ese nivel.

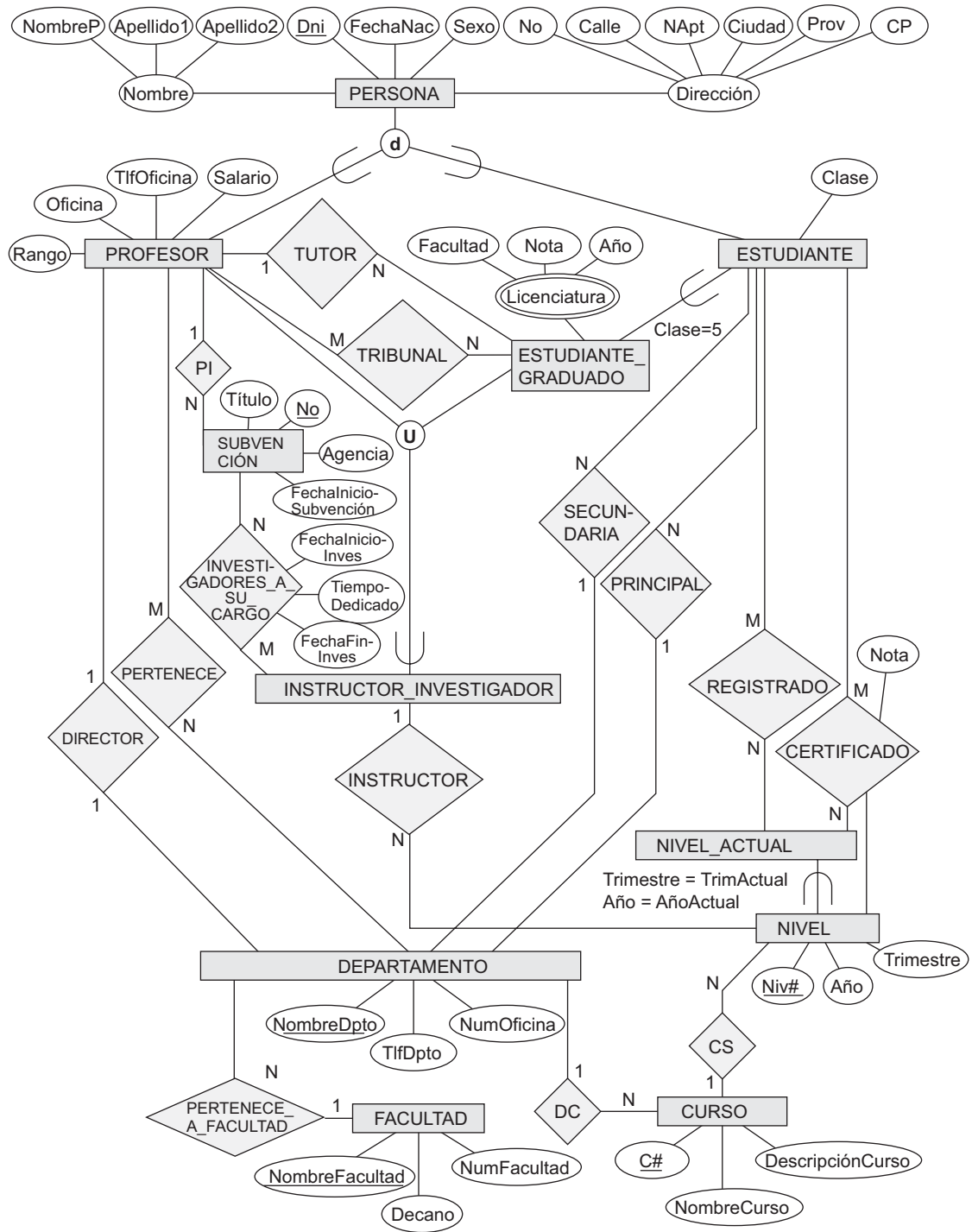
ESTUDIANTE_GRADUADO es una subclase de ESTUDIANTE, con el predicado definido Clase = 5. Por cada estudiante graduado se mantiene una lista de las calificaciones anteriores en un atributo multivalor compuesto llamado [Licenciatura]. También relacionamos al estudiante graduado con un [TUTOR] y su [TRIBUNAL], en caso de existir.

Un departamento académico cuenta con los atributos nombre [NombreDpto], teléfono [TlfDpto] y número de oficina [NumOficina] y está relacionado con la persona que lo dirige [DIRECTOR] y la facultad a la que pertenece [PERTENECE_A_FACULTAD]. Los atributos de cada facultad son su nombre [NombreFacultad], su número de oficina [NumFacultad] y su [Decano].

Cada curso cuenta con los atributos número de curso [C#], el nombre [NombreCurso] y la descripción del mismo [DescripcionCurso]. Cada curso ofrece varios niveles, por lo que cada uno de estos niveles cuenta con un número de nivel [Niv#] y el año y el trimestre en el que se ofrece ([Año] y [Trimestre]).¹⁰ Los números de nivel los identifican de forma única, y los que están siendo ofrecidos en el trimestre actual se encuentran en la subclase NIVEL_ACTUAL de NIVEL, la cual cuenta con un predicado del tipo Trimestre = TrimActual y Año = AñoActual. Cada nivel está relacionado con el profesor que lo imparte ([INSTRUCTOR]) en caso de que se encuentre en la base de datos.

La categoría INSTRUCTOR_INVESTIGADOR es un subconjunto de la unión de PROFESOR y ESTUDIANTE_GRADUADO que incluye a todos los profesores, así como a los estudiantes graduados que están dedicados a la investigación o la enseñanza. Por último, la entidad SUBVENCIÓN contiene las subvenciones y los contratos adjudicados a la universidad. Cada subvención tiene los atributos [Título], número de subvención [No], la entidad adjudicataria [Agencia] y la fecha de inicio [FechaInicioSubvención], y está relacionado con un investigador principal [PI] y todos los [INVESTIGADORES_A_SU_CARGO]. Cada instancia de esta última tiene como atributos la fecha en la que cada investigador empieza con su tarea [FechaInicioInvestigación], la

¹⁰ Asumimos que en esta diversidad se está usando el sistema *trimestral* en lugar del *semestral*.

Figura 4.9. Un esquema EER conceptual para la base de datos UNIVERSIDAD.

de finalización (en caso de conocerse) [FechaFinInvestigación] y el porcentaje de tiempo que dedica al proyecto [TiempoDedicado].

4.5.2 Consideraciones de diseño para la especialización/generalización

No siempre resulta sencillo elegir el diseño conceptual más apropiado para una base de datos. En la Sección 3.7.3, mostramos algunos de los problemas típicos a los que se suele enfrentar un diseñador de bases de datos cuando tiene que representar un tipo de entidad, una relación y los atributos en un esquema ER. En esta sección nos centraremos en las guías maestras para el diseño de la especialización/generalización y las categorías (tipos de unión) en un modelo EER.

Como ya se comentó en la Sección 3.7.3, el diseño conceptual de una base de datos debe considerarse como un proceso de refinamiento iterativo hasta llegar a lo que más se ajuste a nuestras necesidades. Las siguientes notas pueden ayudar a alcanzar este propósito:

- En general, se pueden definir muchas especializaciones y subclases para que el modelo conceptual sea fiel. Sin embargo, el inconveniente es que el diseño se vuelve algo confuso.
- Si una subclase tiene algunos atributos específicos (locales) y no cuenta con relaciones concretas, puede incluirse en la superclase. Los atributos específicos pueden contener valores NULL para aquellas entidades que no sean miembros de la subclase. Un atributo *tipo* podría especificar esta circunstancia.
- De forma análoga, si todas las subclases de una especialización/generalización cuentan con atributos específicos pero no con relaciones, pueden incluirse en la superclase y sustituirse con uno o más atributos *tipo* que especifiquen la subclase o subclases a la que cada entidad pertenece.
- Deben evitarse los tipos y las categorías a menos que la situación garantice este tipo de construcción, lo cual ocurre en ciertas situaciones prácticas. En caso de ser posible, intentaremos modelar usando especialización/generalización tal y como se ha comentado al final de la Sección 4.4.
- La elección de una restricción disyunción/solapamiento y total/parcial en una especialización/generalización está condicionada por las reglas en las que se está llevando a cabo el modelado. Si los requisitos no indican ningún tipo de restricción particular, la elección predeterminada debería ser el solapamiento parcial, ya que esto no especifica ninguna restricción en los miembros de la subclase.

Como ejemplo para la aplicación de esta indicación, considere el ejemplo mostrado en la Figura 4.6, donde no se muestran atributos específicos (locales). Podemos fundir todas las subclases en la entidad EMPLEADO y añadirle los siguientes atributos:

- TipoTrabajo, cuyo conjunto de valores {'Secretaria', 'Ingeniero', 'Técnico'} indicaría la subclase de la primera especialización a la que cada empleado pertenece.
- TipoContrato, cuyo conjunto de valores {'Fijo', 'Temporal'} indicaría la subclase de la segunda especialización a la que cada empleado pertenece.
- EsJefe, cuyo conjunto de valores {'Si', 'No'} indicaría si un empleado es jefe o no.

4.5.3 Definiciones formales para los conceptos del modelo EER*

Vamos a resumir ahora los conceptos del modelo EER y dar definiciones formales. Una **clase**¹¹ es un conjunto o colección de entidades; aquí se incluye cualquier construcción del esquema EER que agrupe entidades, como tipos de entidad, subclases, superclases y categorías. Una **subclase** *S* es una clase cuyas entidades deben ser siempre un subconjunto de las entidades de otra clase llamada la **superclase** *C* de la **relación superclase/**

¹¹ Aquí, el uso de la palabra *clase* difiere del uso más habitual que se le da en los lenguajes de programación orientados a objetos como C++. En él, una clase es una definición de tipo estructurada junto con sus funciones (operaciones).

subclase (o **IS-A**, o **ES-UN**). Indicamos una relación de este tipo como C/S . Para una relación superclase/subclase, siempre debemos tener:

$$S \subseteq C$$

Una **especialización** $Z = \{S_1, S_2, \dots, S_n\}$ es un conjunto de subclases que tienen la misma superclase G , es decir, G/S_i es una relación superclase/subclase para $i = 1, 2, \dots, n$. G recibe el nombre de **tipo entidad-generalizada** (o la **superclase** de la especialización, o una **generalización** de las subclases $\{S_1, S_2, \dots, S_n\}$). Z se dice que es **total** si siempre (y en cualquier momento) tenemos:

$$\bigcup_{i=1}^n S_i = G$$

En cualquier otro caso, Z se dice que es **parcial**. Z es una disyunción si siempre tenemos:

$$S_i \cap S_j = \emptyset \text{ (conjunto vacío) para } i \neq j$$

En cualquier otro caso, Z se dice que es **solapada**.

Se dice que una subclase S de C es de **predicado definido** si un predicado p de los atributos de C se utiliza para especificar las entidades de C que son miembros de S , esto es, $S = C[p]$, donde $C[p]$ es el conjunto de entidades de C que satisfacen p . Una subclase que no está definida por un predicado se dice que es de **usuario definido**.

Una especialización Z (o generalización G) se dice que es de **atributo definido** si un predicado $(A = c_i)$, donde A es un atributo de G y c_i es una constante del dominio de A , se utiliza para especificar los miembros de cada subclase S_i en Z . Observe que si $c_i \neq c_j$ para $i \neq j$, y A es un atributo de un solo valor, entonces la especialización será una disyunción.

Una **categoría** T es una clase que es un subconjunto de la unión de n superclases D_1, D_2, \dots, D_n , $n > 1$, y está formalmente especificada como:

$$T \subseteq (D_1 \cup D_2 \dots \cup D_n)$$

Puede usarse un predicado p_i en los atributos de D_i para especificar los miembros de cada D_i que lo son también de T . Si se especifica un predicado en cada D_i tenemos

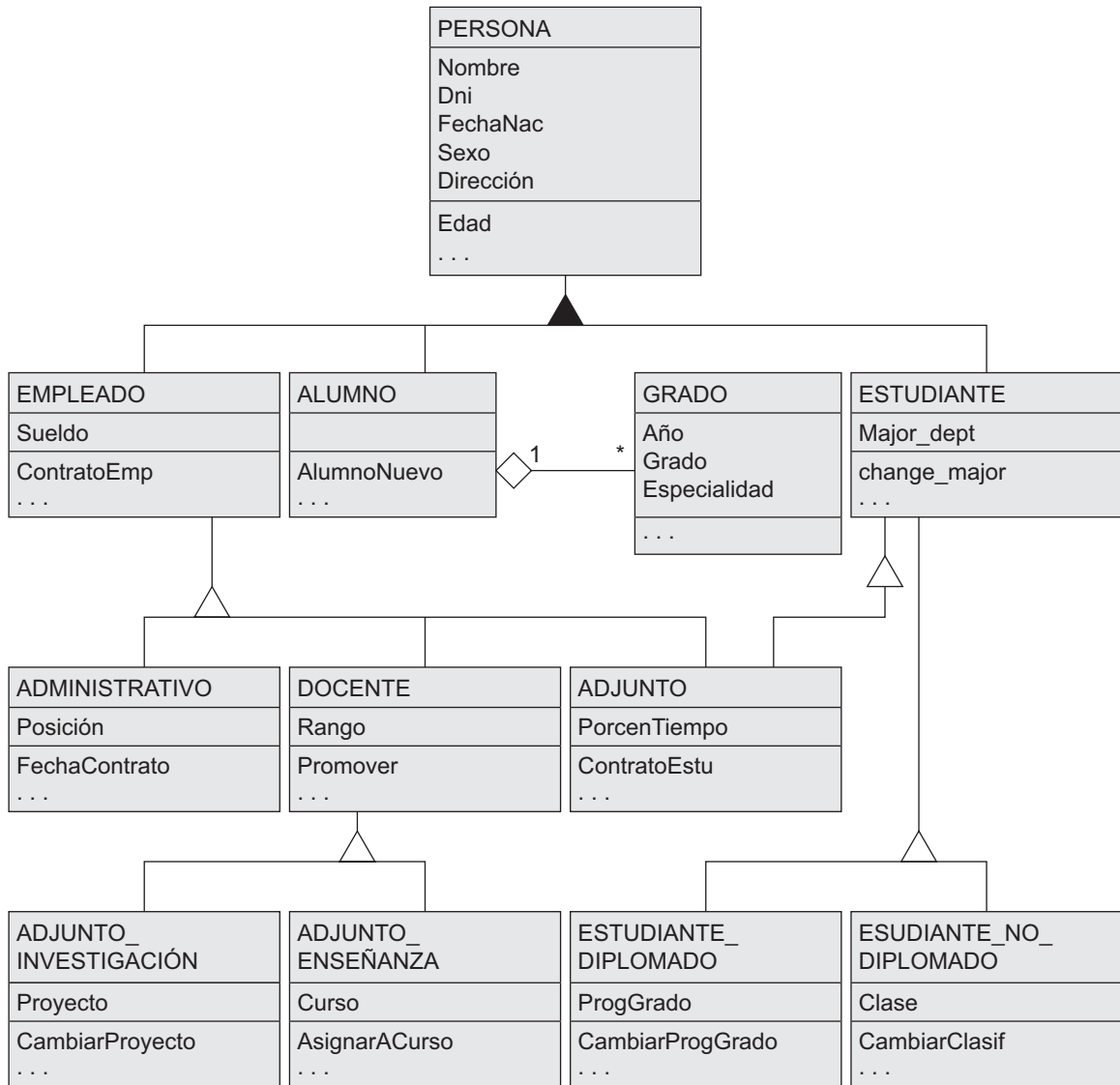
$$T = (D_1[p_1] \cup D_2[p_2] \dots \cup D_n[p_n])$$

Ahora, es necesario ampliar la definición de **tipo de relación** mostrada en el Capítulo 3 permitiendo que cualquier clase (y no sólo cualquier tipo de entidad) participe en la relación. Por tanto, en esta definición debemos cambiar las palabras *tipo de entidad* por *clase*. La indicación gráfica de EER es consecuente con la de ER porque todas las clases están representadas por rectángulos.

4.6 Ejemplo de otra notación: representación de la especialización y la generalización en diagramas de clase UML

Vamos a tratar ahora la notación UML para la generalización/especialización y la herencia. En la Sección 3.8 ya se presentó la terminología y la notación básica del diagrama de clase UML. La Figura 4.10 muestra una posible diagramación UML que coincide con el diagrama EER mostrado en la Figura 4.7. La notación básica para la especialización/generalización (véase la Figura 4.10) es conectar las subclases por líneas verticales a otra horizontal, la cual cuenta con un triángulo que conecta la línea horizontal a la superclase a través de otra línea vertical. Un triángulo en blanco indica una especialización/generalización con la restricción de

Figura 4.10. Diagrama de clase UML correspondiente al diagrama EER de la Figura 4.7, el cual ilustra la notación UML para una especialización/generalización.



disyunción, mientras que otro relleno especifica otra de tipo *solapamiento*. La superclase raíz recibe el nombre de **clase base**, mientras que los nodos hoja se conocen como **clases hoja**. En ambos casos está permitida la herencia simple o múltiple.

El comentario y el ejemplo anteriores (junto con la Sección 3.8) ofrecen una breve panorámica de los diagramas de clase UML y su terminología. En UML existen muchos otros detalles sobre los que no hemos hablado ya que están fuera del ámbito de este libro, y son poco relevantes para la ingeniería de software. Por ejemplo, las clases pueden ser de varios tipos:

- Las clases abstractas definen atributos y operaciones, pero no cuentan con objetos que se correspondan con esas clases. Se utilizan principalmente para especificar un conjunto de atributos y operaciones que pueden ser heredados.

- Las clases concretas pueden tener objetos (entidades) instanciados que pertenezcan a la clase.
- Las clases plantilla especifican un patrón que puede usarse más adelante para definir otras clases.

En el diseño de bases de datos, nos centramos principalmente en la especificación de clases concretas cuyas colecciones de objetos están permanentemente (o persistentemente) almacenadas en la base de datos. Las notas bibliográficas del final del capítulo muestran algunas referencias a libros que describen con detalle la operativa UML. El Capítulo 12 contiene material adicional sobre UML, y el modelado de objetos en general se trata en el Capítulo 20.

4.7 Abstracción de datos, representación del conocimiento y conceptos de ontología

En esta sección comentaremos, en términos abstractos, algunos de los conceptos de modelado que describimos algo más concretamente en nuestra representación de los modelos ER y EER del Capítulo 3 y, anteriormente, en este mismo capítulo. Esta terminología no sólo se emplea en el modelado de datos conceptual, sino también en la literatura sobre inteligencia artificial cuando se trata de la **KR (Representación del conocimiento, Knowledge Representation)**. Esta sección trata sobre las similitudes y las diferencias existentes entre el modelado conceptual y la representación del conocimiento, e introduce algo de terminología alternativa y algunos conceptos adicionales.

El objetivo de las técnicas KR es desarrollar conceptos para el modelado acertado de ciertos **dominios de conocimiento** creando una **ontología**¹² que describe los conceptos del área. Esto se utiliza para almacenar y manipular conocimiento para el dibujo de inferencias, la toma de decisiones o la respuesta a preguntas. Los objetivos de la KR son similares a los de los modelos de datos semánticos, aunque existen importantes similitudes y diferencias entre ambas:

- Las dos disciplinas usan un proceso de abstracción para identificar propiedades comunes y aspectos importantes de los objetos del minimundo (conocido también como dominio de discurso en el KR), a la vez que elimina diferencias insignificantes y detalles sin importancia.
- Ambas disciplinas ofrecen conceptos, restricciones, operaciones y lenguajes para la definición de datos y la representación del conocimiento.
- La KR es, generalmente, más extensa en el ámbito que en los modelos de datos semánticos. Diferentes formas de conocimiento, como reglas (usadas en la inferencia, la deducción y la búsqueda), conocimiento predeterminado e incompleto, y conocimiento espacial y temporal, son representados en esquemas KR. Los modelos de bases de datos están empezando a expandirse para incluir algunos de estos conceptos (consulte el Capítulo 24).
- Los esquemas KR incluyen **mecanismos de razonamiento** que deducen hechos adicionales a partir de otros almacenados en la base de datos. Así pues, mientras la mayor parte de los sistemas de bases de datos están limitados a realizar consultas directas, los sistemas basados en conocimiento que utilizan esquemas KR pueden efectuar preguntas que impliquen **inferencias** sobre los datos almacenados. Las bases de datos actuales están empezando a incluir mecanismos de inferencia (consulte la Sección 24.4).
- Mientras la mayor parte de los modelos de bases de datos se concentran en la representación de sus esquemas, o meta-conocimiento, los esquemas KR suelen mezclar los esquemas con las propias instancias para proporcionar mayor flexibilidad a la hora de representar excepciones. Esto, con frecuencia, suele desembocar en deficiencias cuando estos esquemas KR son implementados, especialmente

¹² Una *ontología* es algo similar a un esquema conceptual, aunque con más conocimiento, reglas y excepciones.

cuando se comparan con bases de datos y cuando es preciso almacenar una gran cantidad de datos (hechos).

En esta sección, trataremos cuatro **conceptos de abstracción** utilizados en modelos de datos semánticos, como el EER y en esquemas KR: (1) clasificación e instanciación, (2) identificación, (3) especialización y generalización y (4) agregación y asociación. Los conceptos pareados de clasificación e instanciación son inversos a los de generalización y especialización. La agregación y la asociación también están relacionadas. Para clarificar el proceso de abstracción de datos y mejorar nuestra comprensión de los procesos relacionados del diseño de esquema conceptual, hablaremos de estos conceptos abstractos y de su relación con las representaciones concretas usadas en el modelo EER. Terminaremos la sección con un breve comentario acerca de la *ontología*, la cual está siendo usada ampliamente en recientes investigaciones de la representación del conocimiento.

4.7.1 Clasificación e instanciación

El proceso de **clasificación** supone la asignación sistemática de objetos/entidades similares a objetos de tipo clase/entidad. Ahora podemos describir (hablando de bases de datos) o razonar (en KR) las clases en lugar de los objetos individuales. Las colecciones de objetos comparten los mismos tipos de atributos, relaciones y restricciones, y al clasificar los objetos simplificamos el descubrimiento de sus propiedades. La **instanciación** es la operación inversa a la clasificación y se refiere al proceso de generación y examen de los distintos objetos de una clase. Por tanto, una instancia de un objeto está relacionada con su clase objeto por la relación **ES-UNA-INSTANCIA-DE** o **ES-UN-MIEMBRO-DE**. Aunque los diagramas EER no muestran las instancias, los UML disponen de una forma de instanciación que permite la visualización de objetos individuales. En nuestra introducción a UML no describimos esta característica.

En general, los objetos de una clase deben tener una estructura similar. Sin embargo, algunos objetos pueden mostrar propiedades que difieran en parte de las de otros objetos de la clase; estos **objetos excepción** también tienen que modelarse, y los esquemas KR permiten más excepciones que los de base de datos. Además, ciertas propiedades se aplican a toda la clase y no a objetos individuales; los esquemas KR permiten este tipo de **propiedades de clase**, al igual que los UML.

En el modelo EER, las entidades están clasificadas en tipos de entidad según sus relaciones y atributos básicos. Las entidades, a su vez, están divididas en subclases y categorías en base a las similitudes y diferencias (excepciones) existentes entre ellas. Las instancias relación están clasificadas en tipos de relación. Por tanto, los tipos de entidad y de relación, las subclases y las categorías son diferentes tipos de clases en el modelo EER. Además, no ofrece explícitamente las propiedades de clase, aunque puede desarrollarse para hacerlo. En UML, los objetos están clasificados en clases, y es posible mostrar tanto las propiedades de la clase como la de los objetos individuales.

Los modelos de representación de conocimiento permiten múltiples esquemas de clasificación en los que una clase es una *instancia* de otra clase (llamada **meta-clase**). Tenga en cuenta que esto no puede mostrarse directamente en un modelo EER, ya que sólo disponemos de dos niveles: clases e instancias. Por tanto, la única relación posible entre las clases es la de tipo superclase/subclase, mientras que en algunos esquemas KR es posible representar directamente una relación adicional clase/instancia en una jerarquía de clase. Una instancia puede ser, por sí misma, otra clase, permitiendo esquemas de clasificación de múltiples niveles.

4.7.2 Identificación

La **identificación** es el proceso de abstracción por el que las clases y los objetos son identificables de forma única por medio de algún **identificador**. Por ejemplo, un nombre de clase identifica inequívocamente a toda esa clase. Es necesario un mecanismo adicional para mantener separadas distintas instancias de objetos mediante identificadores de objeto. Además, es preciso identificar múltiples manifestaciones del mismo obje-

to del mundo real en la base de datos. Por ejemplo, podemos tener una tupla <‘Matías Flis’, ‘610618’, ‘376-9821’> en una relación PERSONA y otra tupla <‘301-54-0836’, ‘CC’, 3.8> en ESTUDIANTE que pareciera que representasen a la misma persona. No existe manera de identificar que estos dos objetos de base de datos (tuplas) representan a la misma persona a menos que, durante la *fase de diseño*, preparemos los mecanismos adecuados para establecer esta relación cruzada. Por consiguiente, la identificación es necesaria a dos niveles:

- Para distinguir entre objetos y clases de la base de datos.
- Para identificar objetos de base de datos y relacionarlos con sus homólogos del mundo real.

En el modelo EER, la construcción de un esquema de identificación se basa en un sistema de nombres únicos para los constructores. Por ejemplo, cada clase (ya sea un tipo de entidad, una subclase, una categoría o un tipo de relación) debe contar con un nombre distinto. Los nombres de atributo de una clase específica también deben ser diferentes. También son necesarias reglas para identificar con claridad las referencias a los nombres de atributo en una jerarquía o entramado de especialización o generalización.

A nivel de objeto, se emplean los valores de los atributos clave para distinguir entre entidades de un tipo de entidad particular. Para los tipos débiles, las entidades se identifican por una combinación de sus propios valores clave parciales y las entidades que están relacionadas en el tipo (o tipos) de entidad propietaria. Las instancias de relación están identificadas por alguna combinación de las entidades que relacionan, en base al índice de cardinalidad especificado.

4.7.3 Especialización y generalización

La especialización es el proceso para clasificar una clase en subclases más especializadas. La generalización es el proceso inverso de generalizar varias clases en una clase abstracta de nivel superior que incluya los objetos de todas esas clases. La especialización es un refinamiento conceptual, mientras que la generalización es una síntesis conceptual. Las subclases se emplean en el modelo EER para representar la especialización y la generalización. La relación entre una subclase y su superclase recibe el nombre de relación **ES-UNA-SUBCLASE-DE** o, abreviando, una relación **ES-UN/ES-UNA**.

4.7.4 Agregación y asociación

La agregación es un concepto abstracto para la construcción de objetos complejos a partir de sus objetos componente. Existen tres situaciones en las que este concepto puede estar relacionado con el modelo EER. El primero se produce cuando añadimos atributos de un objeto para formar el objeto completo. El segundo se da cuando representamos una relación de agregación como una relación común. El tercer caso, para el cual no se proporciona explícitamente el modelo EER, supone la posibilidad de combinar objetos que están relacionados con una instancia relación particular en un *objeto agregado de nivel superior*. Esto suele ser útil a veces cuando este objeto, por sí mismo, está relacionado con otro. Llamamos a la relación existente entre los objetos primitivos y sus objetos agregados **ES-UNA-PARTE-DE**; la situación inversa recibe el nombre de **ES-UN-COMPONENTE-DE**. UML proporciona soporte para los tres tipos de agregación.

La abstracción de **asociación** se utiliza para asociar objetos procedentes de varias *clases independientes*. Por tanto, es muy parecido al segundo uso de la agregación. En el modelo EER está representado por los tipos de relación, mientras que en UML lo está por las asociaciones. Esta relación abstracta se llama **ESTÁ-ASOCIADA-CON**.

Para entender mejor los distintos usos de la agregación, considere el esquema ER de la Figura 4.11(a), el cual almacena información sobre las entrevistas realizadas por los aspirantes a obtener un empleo en distintas empresas. La clase EMPRESA es una agregación de los atributos (u objetos componente) NombreEmpresa (nombre de la empresa) y DirEmpresa (dirección de la empresa), mientras que ASPIRANTE_TRABAJO es una agregación de Dni, Nombre, Dirección y Teléfono. Los atributos de relación NombreContacto y TlfContacto

representan el nombre y el teléfono de la persona de la empresa responsable de la entrevista. Supongamos que alguna de las entrevistas tiene como consecuencia una oferta de trabajo, mientras que otras no. Podríamos querer tratar ENTREVISTA como una clase para asociarla con OFERTA_TRABAJO. El esquema de la Figura 4.11(b) es *incorrecto* porque supone que cada instancia de la relación entrevista tiene una oferta de trabajo. El esquema de la Figura 4.11(c) *no está permitido*, ya que el modelo ER no permite establecer relaciones entre relaciones.

Una forma de representar esta situación es crear una clase agregada de nivel superior compuesta por EMPRESA, ASPIRANTE_TRABAJO y ENTREVISTA, y relacionarla con OFERTA_TRABAJO (véase la Figura 4.11[d]). Aunque el modelo EER, como ya hemos mencionado, no permite esta situación, algunos modelos de datos semánticos sí que lo hacen y nombran al objeto resultante un **compuesto** u **objeto molecular**. Otros modelos tratan a los objetos entidad y relación uniformemente y, por ello, permiten las relaciones entre relaciones (véase la Figura 4.11[c]).

Para representar correctamente esta situación en el modelo ER, necesitamos crear un nuevo tipo de entidad débil ENTREVISTA (véase la Figura 4.11[e]), y relacionarlo con OFERTA_TRABAJO. De este modo, siempre es posible representar correctamente estas situaciones en el modelo ER creando tipos de entidad adicionales aunque, conceptualmente, puede ser más deseable permitir la representación directa de la agregación, como puede verse en la Figura 4.11(d), o consentir las relaciones entre relaciones (véase la Figura 4.11[c]).

La principal distinción estructural entre la agregación y la asociación es que cuando una instancia de asociación se borra, los objetos que participan de ella pueden seguir existiendo. Sin embargo, si abogamos por la noción de objeto agregado (por ejemplo, un COCHE compuesto de objetos MOTOR, CHASIS y RUEDAS), el borrado del objeto COCHE implica la eliminación de todos los demás.

4.7.5 Ontologías y la semántica Web

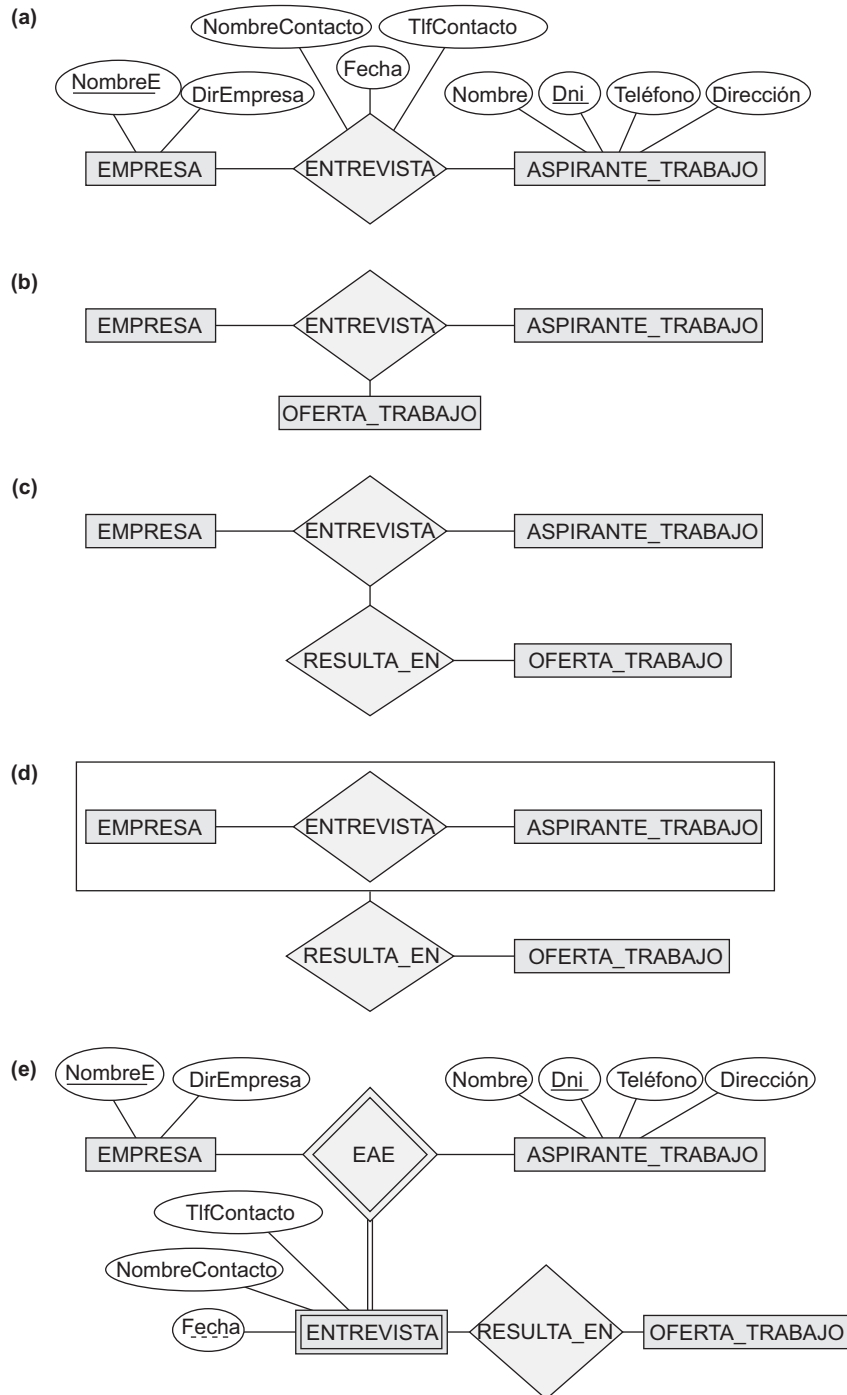
Últimamente, la cantidad de datos informatizados y de información disponible en la Web está fuera de control. Para ello, se utilizan muchos modelos y formatos diferentes. Además de los modelos de bases de datos mostrados en este libro, una gran cantidad de información se almacena en forma de **documentos**, los cuales precisan de una estructura mucho menor de la necesaria en la información de una base de datos. **Semantic Web** es un proyecto de investigación que está intentando permitir el intercambio de información entre computadores de la Web, además de intentar crear modelos de representación de conocimiento que sean lo más generales posible para permitir el intercambio y la búsqueda de información significativa entre máquinas. Se está intentando que la *ontología* sea la piedra angular sobre la que se asiente Semantic Web, y está íntimamente relacionado con la representación del conocimiento. En esta sección, se ofrecerá una breve introducción sobre qué es la ontología y cómo puede usarse para automatizar la comprensión, búsqueda e intercambio de información.

El estudio de las ontologías intenta describir las estructuras y las relaciones que son posibles en la realidad a través de vocabulario común; por consiguiente, puede considerarse como una forma de describir el conocimiento de la realidad de una cierta comunidad. La ontología tuvo su origen en la filosofía y la metafísica. Una definición de **ontología** comúnmente usada es la de la *especificación* de una *conceptualización*.¹³

En esta definición, una **conceptualización** es el conjunto de conceptos usados para representar la parte de realidad o conocimiento que son de interés a una comunidad de usuarios. La **especificación** se refiere al lenguaje y el vocabulario empleados para especificar la conceptualización. La ontología incluye tanto la *especificación* como la *conceptualización*. Por ejemplo, a través de dos ontologías diferentes puede especificarse la misma conceptualización. Aun basándonos en esta definición general, no existe consenso acerca de lo que es exactamente la ontología. Éstas son algunas formas de describirla:

¹³ Esta definición la propuso Gruber (1995).

Figura 4.11. Agregación. (a) Tipo de relación ENTREVISTA. (b) Incluyendo OFERTA_TRABAJO en un tipo de relación ternario (incorrecto). (c) Con la relación RESULTA_EN participando en otras relaciones (no está permitido en ER). (d) Usando una agregación y un objeto compuesto (molecular) (normalmente, no está permitido en ER, aunque sí en algunas herramientas de modelado). (e) Representación correcta en ER.



- Un **diccionario de sinónimos** (e incluso un **diccionario** o un **glosario** de términos) describen las relaciones existentes entre las palabras (vocabulario) que representan diferentes conceptos.
- Una **taxonomía** describe el modo en que están relacionados los conceptos de un dominio particular de conocimiento usando estructuras similares a las empleadas en una especialización o una generalización.
- Un **esquema de base de datos** detallado está considerado por algunos como una ontología que describe los conceptos (entidades y atributos) y las relaciones de un minimundo real.
- Una **teoría lógica** usa los conceptos de la lógica matemática para intentar definir los conceptos y sus interrelaciones.

Habitualmente, los conceptos utilizados para describir ontologías son muy similares a los que empleamos en el modelado conceptual, como entidades, atributos, relaciones, especializaciones, etc. La diferencia principal entre una ontología y, digamos, un esquema de base de datos es que el esquema suele limitarse a describir un pequeño subconjunto de un minimundo real con el objetivo de almacenar y administrar datos. Una ontología suele considerarse algo más general, ya que intenta describir una parte de la realidad o un área de interés (por ejemplo, términos médicos, aplicaciones de comercio electrónico) de la forma más completa posible.

4.8 Resumen

En este capítulo hemos estudiado las extensiones del modelo ER que mejoran sus capacidades de representación. Llamamos al modelo resultante ER mejorado o modelo EER. Presentamos el concepto de una subclase y su superclase y el mecanismo relacionado de herencia atributo/relación. Mostramos cómo, a veces, es necesario crear clases de entidades adicionales, ya fuera debido a atributos específicos adicionales o debido a tipos de relación concretos. Abordamos los dos procesos principales para la definición de jerarquías y entramados superclase/subclase: la especialización y la generalización.

A continuación, mostramos la forma de representar estas nuevas construcciones en un diagrama EER. También debatimos los diferentes tipos de restricciones que pueden aplicarse a la especialización o la generalización: total/parcial y disyunción/solapamiento. Además, puede definirse un predicado para una subclase o un atributo para una especialización. Explicamos las diferencias existentes entre subclases definidas por usuario y de predicado definido y entre especializaciones del mismo tipo. Para terminar, planteamos el concepto de una categoría o tipo unión, la cual se define como un subconjunto de la unión de dos o más clases, y ofrecemos definiciones formales de todos los conceptos presentados.

Mostramos parte de la notación y la terminología UML para representar la especialización y la generalización. En la Sección 4.7 abordamos brevemente la disciplina de la representación del conocimiento y el modo que está relacionado con el modelado de datos semántico. También ofrecemos una panorámica y un resumen de los tipos de conceptos de la representación abstracta de datos: la clasificación y la instanciación, la identificación, la especialización y la generalización, y la agregación y la asociación, sin olvidarnos de la forma en que los conceptos EER y UML están relacionados con todos ellos.

Preguntas de repaso

- 4.1. ¿Qué es una subclase? ¿Cuándo es necesaria una subclase en el modelado de datos?
- 4.2. Defina los siguientes términos: superclase de una subclase, relación superclase/subclase, relación es-una, especialización, generalización, categoría, atributos específicos (locales) y relaciones específicas.
- 4.3. Aborde el mecanismo de herencia atributo/relación. ¿Por qué es útil?
- 4.4. Comente las subclases definidas por usuario y de predicado definido, e identifique las diferencias existentes entre ellas.

- 4.5. Plantee las especializaciones definidas por usuario y de predicado definido, e identifique las diferencias existentes entre ellas.
- 4.6. Explique los dos tipos principales de restricciones en las especializaciones y las generalizaciones.
- 4.7. ¿Cuál es la diferencia entre una jerarquía y un entramado de especialización?
- 4.8. ¿Cuál es la diferencia entre una especialización y una generalización? ¿Por qué no podemos mostrar esta diferencia en los diagramas de esquema?
- 4.9. ¿En qué difiere una categoría de una subclase compartida corriente? ¿Para qué se usa una categoría? Argumente su respuesta con ejemplos.
- 4.10. Por cada uno de los siguientes términos UML (consulte las Secciones 3.8 y 4.6), indique el correspondiente en el modelo EER, en caso de existir: objeto, clase, asociación, agregación, generalización, multiplicidad, atributos, discriminador, enlace, atributo de enlace, asociación reflexiva y asociación cualificada.
- 4.11. Comente las diferencias principales existentes entre la notación en los diagramas de esquema EER y los de clase UML comparando el modo en que se representan los conceptos comunes.
- 4.12. Enumere los distintos conceptos de abstracción de datos y los conceptos de modelado correspondientes en el modelo EER.
- 4.13. ¿Qué característica de agregación no existe en el modelo EER? ¿Cómo podría mejorarse para soportarla?
- 4.14. ¿Cuáles son las principales similitudes y diferencias existentes entre las técnicas de modelado de bases de datos conceptuales y las de representación del conocimiento?
- 4.15. Comente las similitudes y diferencias existentes entre una ontología y un esquema de base de datos.

Ejercicios

- 4.16. Diseñe un esquema EER para una aplicación de bases de datos en la que esté interesado. Especifique todas las restricciones que necesite. Asegúrese de que el esquema dispone de, al menos, cinco tipos de entidad, cuatro tipos de relación, un tipo de entidad débil, una relación superclase/subclase, una categoría y un tipo de relación n-cualquiera ($n > 2$).
- 4.17. Considere el esquema ER BANCO de la Figura 3.21, y que es necesario controlar los diferentes tipos de CUENTA (AHORRO, ARQUEO, etc.) y PRÉSTAMO (COCHE, HIPOTECARIO, etc.). Suponga también que es aconsejable gestionar cada TRANSACCIÓN de una CUENTA (depósitos, retiradas, cheques, etc.) y cada PAGO del PRÉSTAMO; ambas situaciones incluyen la cantidad, la fecha y la hora. Modifique el esquema BANCO usando los conceptos ER y EER de especialización y generalización. Haga constar cualquier supuesto que haga sobre requerimientos adicionales.
- 4.18. La siguiente historia narra una versión simplificada de la organización de las instalaciones para unas Olimpiadas de verano. Dibuje un diagrama EER que muestre los tipos de entidad, los atributos, las relaciones y las especializaciones para esta aplicación. Haga constar cualquier supuesto que haga. Las instalaciones olímpicas están divididas en complejos deportivos, los cuales, a su vez, son de tipo *monodeportivo* y *multideportivo*. Los complejos multideportivos tienen áreas diseñadas para cada una de las especialidades y cuentan con un indicador (por ejemplo, centro, esquina NE, etc.). Un complejo dispone de una localización, su jefe de organización, el área ocupada, etc. Cada complejo alberga una serie de eventos (por ejemplo, en la pista de carreras se pueden disputar varios tipos de ellas). Cada evento está planificado para una fecha, tendrá una duración, un número de participantes y jueces, etc. Será preciso mantener una lista de todos los jueces junto con las pruebas en las que estarán presentes. Para cada prueba será necesario un equipamiento distinto (por ejemplo, las porterías, las pértigas, las barras paralelas) y su mantenimiento. Ambos tipos de complejos (monodeportivo y multideportivo) contarán con distintos tipos de información. Para cada uno, es preciso mantener el número de instalaciones necesarias, junto con un presupuesto aproximado.

- 4.19.** Identifique los conceptos más importantes representados en el estudio de la base de datos de una biblioteca mostrado más adelante. En particular, preste atención a las abstracciones de la clasificación (tipos de entidad y de relación), la agregación, la identificación y la especialización/generalización. Especifique las restricciones de cardinalidad (mínima, máxima) siempre que sea posible. Enumere los detalles que afectarán al diseño eventual, pero que no tengan interrelación con el conceptual. Identifique de forma separada las restricciones semánticas. Dibuje un diagrama EER de esta base de datos.

Caso a estudiar. La Georgia Tech Library (GTL) cuenta aproximadamente con unos 16.000 miembros, 100.000 títulos y 250.000 volúmenes (una media de 2,5 copias por libro). Alrededor del 10% se encuentra permanentemente fuera en modo de préstamo. Los bibliotecarios aseguran que los libros que se deseen pedir prestados lo estarán en el momento en que los miembros así lo deseen. Además, es preciso que conozcan cuántas copias están prestadas en cada momento. Existe un catálogo de libros *online* disponible que enumera las obras por autor, título y área. Para cada una de ellas, el catálogo mantiene un descriptor de libro que oscila desde una frase a varias páginas. Se quiere que estas referencias estén accesibles para los bibliotecarios cuando un miembro solicita información acerca de un libro. La plantilla de la biblioteca incluye un bibliotecario jefe, los bibliotecarios departamentales asociados, los de referencia, el plantel de verificación y los bibliotecarios asistentes.

Los libros pueden retenerse durante 21 días, y los miembros sólo pueden tener 5 ejemplares a la vez. Por lo general, los usuarios devuelven los libros a las tres o cuatro semanas, y la mayoría sabe que disponen de una semana de gracia antes de que se les notifique esta situación, por lo que todos intentan hacerlo antes de que expire dicho periodo. Es necesario hacer un recordatorio de la devolución a alrededor del 5% de los miembros, y la mayor parte de los libros se devuelven un mes después de vencer la fecha tope. Incluso, existe un 5% de obras que se pierden definitivamente. Los miembros más activos de la biblioteca son aquéllos que solicitan libros, al menos, diez veces al año. El 1% de los miembros más activos realiza el 15% de las peticiones, y el 10% de los miembros más activos el 40%. Cerca del 20% nunca realiza una petición.

Para pertenecer a la biblioteca, los solicitantes rellenan un formulario en el que se incluye su DNI, su dirección postal personal y la de su centro de estudios y los números de teléfono. Los bibliotecarios expiden entonces una tarjeta magnética numerada con su foto y válida para cuatro años. Un mes antes de que expire, se le envía un mensaje de renovación. Los profesores de los centros de enseñanza son considerados miembros de forma automática. Cuando un nuevo profesor entra en un colegio, se obtiene la información necesaria de su registro de empleado y se envía por correo una tarjeta a su dirección profesional. Los profesores pueden revisar libros en intervalos de tres meses, y su periodo de gracia es de dos semanas. Las renovaciones se remiten a su dirección profesional.

La biblioteca no presta ciertos libros, como obras de referencia, volúmenes raros y mapas, por lo que los bibliotecarios deben ser capaces de distinguir qué obras pueden prestar y cuáles no. Además, cuentan con una lista de algunos libros que resultaría interesante adquirir pero que no pueden, como obras raras o descatalogadas y otras que se perdieron o se destruyeron y que no han sido reemplazadas. Algunos libros pueden tener el mismo título; por consiguiente, este dato no puede usarse como campo clave. Cada uno de ellos está identificado por su ISBN (*International Standard Book Number*), un código internacional único asignado a todos los libros. Dos obras con el mismo título pueden tener ISBN diferentes si están escritos en idiomas diferentes o tienen distintas encuadernaciones (tapa dura, tapa blanda). Las ediciones de la misma obra tienen distintos ISBN.

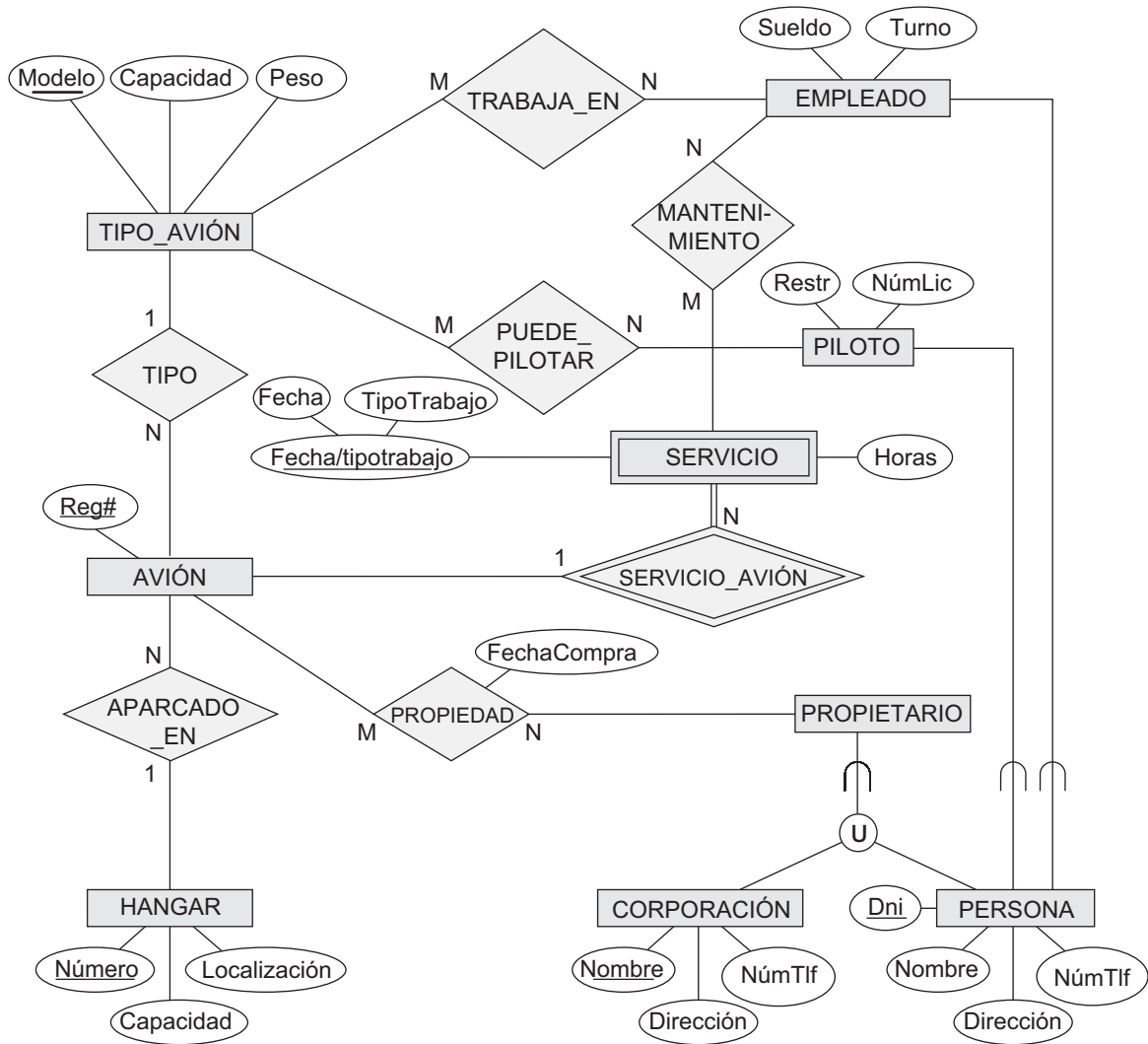
La base de datos propuesta debe diseñarse de forma que controle los miembros, los libros, el catálogo y los préstamos.

- 4.20.** Diseñe una base de datos para gestionar la información de un museo de arte. Asumimos que los siguientes datos fueron recopilados:

- El museo dispone de una colección de OBJETOS_DE_ARTE. Cada uno de ellos cuenta con un identificador único (Id), un Artista (en caso de conocerse), el Año de creación (si se conoce), un Título y una Descripción. Los objetos están clasificados de varias formas, tal y como se comentará más adelante.
- OBJETOS_DE_ARTE está categorizada en función a sus tipos, de los cuales hay tres principales: PINTURA, ESCULTURA y MONUMENTO más un cuarto llamado OTRO para acomodar a aquéllos que no se ajustan a ninguno de los otros tres.
- Una PINTURA tiene un TipoPintura (aceite, al agua, etc.), el material sobre el que está Dibujado (papel, lienzo, madera, etc.) y un EstiloPintura (moderno, abstracto, etc.).
- Una ESCULTURA o un MONUMENTO tiene el Material sobre el que fue creado (madera, piedra, etc.), una Altura, una Anchura y un EstiloEscultura.
- Un objeto de arte encuadrado en la categoría OTRO tiene un TipoObra (impresión, fotografía, etc.) y un EstiloOtro.
- Los OBJETOS_DE_ARTE están clasificados como una COLECCIÓN_PERMANENTE (aquéllos que son propiedad del museo) y como PRESTADO. Los datos con los que contamos del primer tipo son la FechaAdquisición, su Estado (en exhibición, en préstamo o almacenada) y su Coste. La información tomada sobre los objetos de tipo PRESTADO incluye la Colección propietaria de la misma, su FechaPréstamo y la FechaDevolución.
- La información acerca del país o la cultura de Origen (Italia, Egipto, América, India, etc.) y su Época (Renacimiento, Moderna, Antigua, etc.) se almacena en cada OBJETO_DE_ARTE.
- El museo conserva información sobre el ARTISTA, en caso de conocerse: Nombre, FechaNac (si se sabe), FechaFallecimiento (si corresponde), PaísOrigen, Época, EstiloPrincipal y Descripción. Se asume que el Nombre es un dato único.
- Se celebran distintas EXHIBICIONES, cada una con su Nombre, su FechaInicio y su FechaFinalización. Las EXHIBICIONES están relacionadas con los objetos de arte que están en estado de exhibición durante la misma.
- También se mantiene información sobre otras COLECCIONES con las que el museo interactúa, incluyendo su Nombre (único), el Tipo (museo, personal, etc.), Descripción, Dirección, Teléfono y PersonaContacto actual.

Diseñe un diagrama EER para esta aplicación. Haga constar cualquier supuesto que haga, y justifíquelo.

- 4.21.** La Figura 4.12 muestra un diagrama EER para la base de datos de un pequeño aeropuerto que se utiliza para mantener la información de las aeronaves, sus propietarios, los empleados del aeropuerto y los pilotos. Éstos son los datos recopilados. Cada AVIÓN dispone de un número de registro [Reg#], es de un tipo particular [TIPO] y está aparcado en un hangar concreto [APARCADO_EN]. Cada TIPO_AVIÓN tiene un número de [Modelo], una [Capacidad] y un [Peso]. Cada HANGAR cuenta con un [Número], una [Capacidad] y una [Localización]. La base de datos también controla los propietarios de cada aeronave [PROPIETARIO] y los empleados encargados del [MANTENIMIENTO]. Cada PROPIETARIO está relacionado con un AVIÓN e incluye la fecha de adquisición [FechaCompra]. Cada relación en MANTENIMIENTO asocia a un empleado con un [SERVICIO]. Cada avión se somete a revisión cada cierto tiempo; por tanto, está relacionado con [SERVICIO_AVIÓN] por un número de registro SERVICIO, cada uno de los cuales incluye como atributos la fecha de mantenimiento [Fecha], el número de horas empleadas en el trabajo [Horas] y el tipo de servicio efectuado [TipoTrabajo]. Utilizamos una entidad débil [SERVICIO] para representar el servicio del avión, ya que su número de registro se utiliza para identificar un registro de servicio. Un PROPIETARIO puede ser tanto una persona como una corporación. Por tanto, utilizamos

Figura 4.12. Esquema EER para la base de datos PEQUEÑO_AEROPUERTO.

un tipo de unión (categoría) [PROPIETARIO] que es un subconjunto de los tipos de entidad [CORPORACIÓN] y [PERSONA]. Tanto los pilotos [PILOTO] como los empleados [EMPLEADO] son subclases de PERSONA. Cada PILOTO tiene, como atributos específicos, su número de licencia de vuelo [NúmeroLicencia] y sus restricciones [Restricción], mientras que los de cada EMPLEADO son el [Salario] y el [Turno]. Toda entidad PERSONA de la base de datos tiene su número del Documento nacional de identidad [Dni], un [Nombre], una [Dirección] y un [NúmeroTeléfono]. Para cada CORPORACIÓN existe un [Nombre], una [Dirección] y un [NúmeroTeléfono]. La base de datos también mantiene los tipos de aviones que cada piloto está autorizado a pilotar [PUEDE_PILOTAR] y aquéllos en los que cada empleado puede realizar tareas de mantenimiento [TRABAJA_EN]. Observe cómo el esquema EER de PEQUEÑO_AEROPUERTO de la Figura 4.12 puede representarse en notación UML. (Nota. No hemos comentado la forma de representar las categorías [tipos de unión] en UML, por lo que no tiene que indicarlo ni en ésta ni en la siguiente pregunta.)

4.22. Desarrolle el modo de representar en notación UML el esquema EER UNIVERSIDAD de la Figura 4.9.

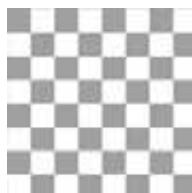
4.23. Considere los conjuntos de entidades y atributos mostrados en la siguiente tabla. Coloque una marca en una de las columnas de cada fila para indicar la relación entre las columnas derecha y la situada más a la izquierda.

- (a) El lado izquierdo tiene una relación con el derecho.
- (b) El lado derecho es un atributo del lado izquierdo.
- (c) El lado izquierdo es una especialización del derecho.
- (d) El lado izquierdo es una generalización del derecho.

Entidad	(a) Tiene una relación con	(b) Tiene un atributo que es	(c) Es una especialización de	(d) Es una generalización de	Entidad o atributo
1. MADRE					PERSONA
2. HIJA					MADRE
3. ESTUDIANTE					PERSONA
4. ESTUDIANTE					IdEstudiante
5. COLEGIO					ESTUDIANTE
6. COLEGIO					AULA
7. ANIMAL					CABALLO
8. CABALLO					Raza
9. CABALLO					Edad
10. EMPLEADO					DNI
11. MUEBLE					SILLA
12. SILLA					Peso
13. HUMANO					MUJER
14. SOLDADO					PERSONA
15. SOLDADO_ENEMIGO					PERSONA

4.24. Dibuje un diagrama UML para almacenar en una base de datos una partida de ajedrez. Puede consultar la dirección <http://www.chessgames.com> para obtener información acerca de una aplicación similar a la que tiene que desarrollar. Documente claramente cualquier decisión que tome en su diagrama UML. Las siguientes pueden ser algunas de esas decisiones:

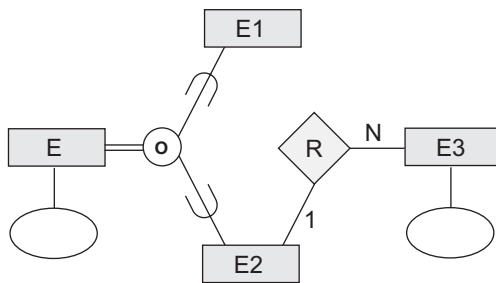
- 1. La partida se realiza entre dos jugadores.
- 2. Se juega en un tablero de 8×8 similar al mostrado a continuación:



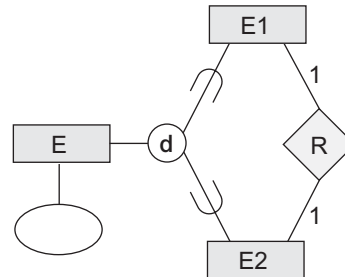
3. Los jugadores asumen un color (negro o blanco) al inicio de la partida.
4. Cada jugador empieza con las siguientes piezas:
 - a. 1 rey d. 2 alfiles
 - b. 1 reina e. 2 caballos
 - c. 2 torres f. 8 peones
5. Cada pieza se encuentra en su posición inicial.
6. Cada pieza cuenta con su propio conjunto de movimientos permitidos en función al estado de la partida. No es necesario preocuparse de qué movimientos son legales y cuáles no, excepto en los siguientes casos:
 - a. Una pieza puede moverse a un cuadro vacío o capturar una pieza del contrario.
 - b. Si una pieza es capturada, se elimina del tablero.
 - c. Si un peón alcanza la última fila es “promocionado”, convirtiéndose en otra pieza (reina, torre, alfil o rey).

Nota: Algunas de estas funciones pueden aplicarse a múltiples clases.

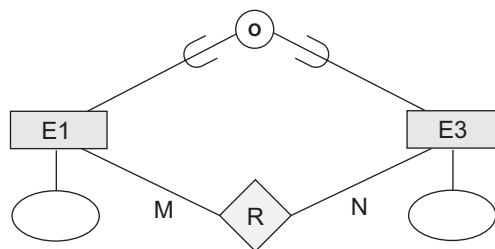
- 4.25. Dibuje un diagrama EER para la partida de ajedrez descrita en el Ejercicio 4.24. Concéntrese en los aspectos de almacenamiento persistente del sistema. Por ejemplo, puede que sea necesario recuperar todos los movimientos de cada partida en orden secuencial.
- 4.26. ¿Cuáles de los siguientes diagramas EER son incorrectos y por qué? Documente claramente cualquier decisión que tome.



(a)

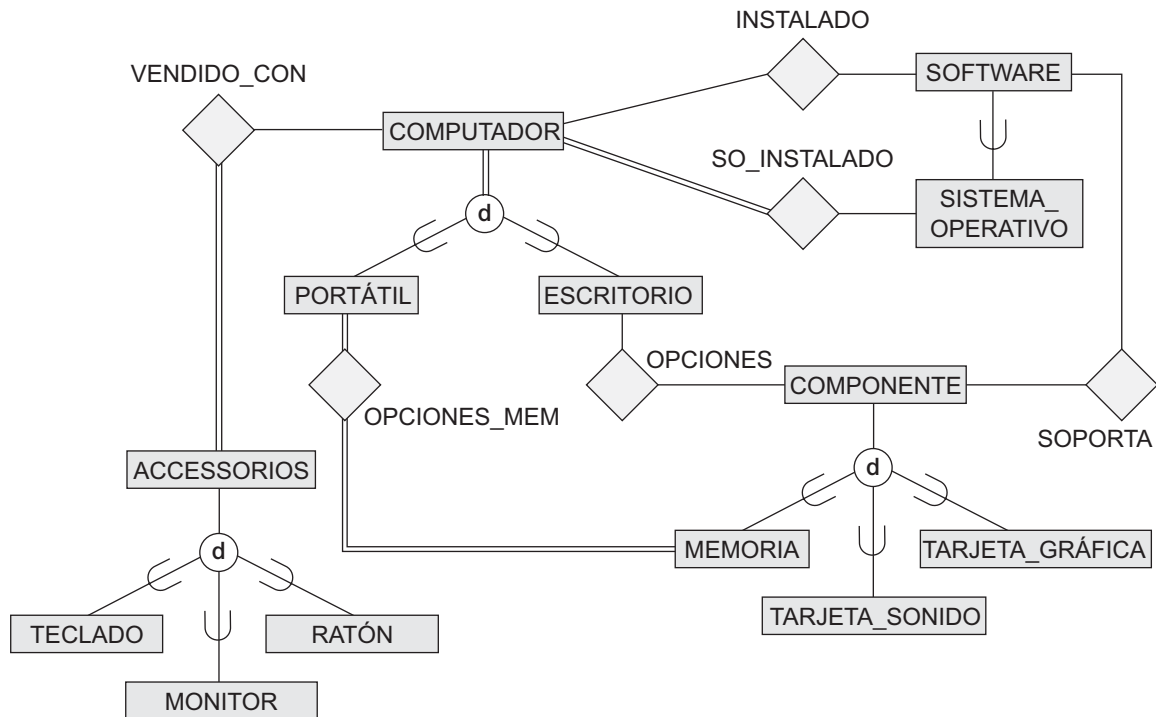


(b)



(c)

- 4.27. Considere un diagrama EER en el que se describe los sistemas informáticos de una empresa. Proporcione sus propios atributos y claves para cada tipo de entidad. Facilite las restricciones de cardinalidad máximas justificando su elección. Escriba una descripción completa de lo que representa este diagrama EER.



Ejercicios de práctica

- 4.28. Considere el diagrama EER de la base de datos UNIVERSIDAD mostrada en la Figura 4.9. Realice este diseño usando alguna herramienta de modelado de datos como ERWin o Rational Rose. Elabore una lista de las diferencias de notación existente entre el diagrama en modo texto y el equivalente construido con la herramienta.
- 4.29. Considere el diagrama EER de la base de datos del PEQUEÑO_AEROPUERTO mostrada en la Figura 4.12. Realice este diseño usando alguna herramienta de modelado de datos como ERWin o Rational Rose. Preste especial atención a la forma de modelar la categoría PROPIETARIO en este diagrama. (*Consejo:* Considere usar PROPIEDAD_DE_EMPRESA y una PROPIEDAD_DE_PERSONA como dos tipos de relación diferentes).
- 4.30. Considere la base de datos UNIVERSIDAD descrita en el Ejercicio 3.16. En la Práctica 3.31 realizó un esquema ER para la misma mediante herramientas de modelado como ERWin o Rational Rose. Modifique este diagrama clasificando un CURSO como CURSO_BÁSICO o CURSO_SUPERIOR y un INSTRUCTOR como PROFESOR_AGREGADO o PROFESOR_TITULAR. Incluya los atributos apropiados para estos nuevos tipos de entidades. A continuación, establezca las relaciones necesarias para que el profesor agregado sea el encargado de impartir los cursos básicos, mientras que los titulares se encarguen de los superiores.

Bibliografía seleccionada

Son muchos los artículos que han propuesto modelos de datos semánticos o conceptuales. A continuación le ofrecemos una lista representativa de los mismos. Un grupo de ellos, entre los que se incluyen Abrial (1974), el modelo DIAM de Senko (1975), el método NIAM (Verheijen y VanBekum 1982), y Bracchi y otros (1976), presentan los modelos semánticos que están basados en el concepto de relaciones binarias. Un segundo grupo más moderno aborda los métodos para extender el modelo relacional y mejorar sus posibilidades.