12 DIAGRAMAS DE ENTIDAD-RELACION

Obviamente, el juicio de un hombre no puede superar la información en la que él lo basó. Désele la verdad y aun así pudiera errar teniendo la oportunidad de acertar; pero no le den noticias, o dénsele sólo datos incompletos y distorsionados, mostrados de manera ignorante, descuidada o prejuiciosa, con propaganda y falsedades, y se destruirán todos sus procesos de razonamiento, y él se convertirá en algo menos que un hombre.

Arthur Hays Sulzberger
Discurso, Asociación de Editores del Estado de Nueva York, 1948

En este capítulo se aprenderá:

1. Por qué son útiles los modelos de datos en el análisis de sistemas.
2. Los componentes de un diagrama de entidad-relación.
3. Cómo escribir un diagrama de entidad-relación.
4. Cómo refinar un diagrama inicial de entidad-relación.

En este capítulo se explora una notación gráfica para modelar datos. El *diagrama de entidad-relación* (también conocido como DER, o diagrama E-R) es un modelo de red que describe con un alto nivel de abstracción la distribución de datos almacenados en un sistema. Es muy diferente del DFD, que modela las *funciones* que lleva a cabo un sistema; y es diferente del *diagrama de transición de estados*, que modela el comportamiento dependiente del tiempo de un sistema.

¿Por qué podríamos estar interesados en modelar los datos de un sistema? Primariamente, porque las estructuras de datos y las relaciones pueden ser tan complejas que se deseara enfatizarlas y examinarlas *independientemente* del proceso que se llevará a cabo. De hecho, esto se da sobre todo si mostramos el modelo del sistema a los usuarios ejecutivos de mayor nivel de una organización (por ejemplo el vicepresidente o los gerentes de departamento, quienes podrían no estar interesados en los detalles funcionales cotidianos del sistema). Tales usuarios a menudo se preocupan más por los datos: ¿qué datos requerimos para manejar nuestro negocio? ¿Quién los tiene? ¿Quién tiene acceso a ellos?

Algunas de estas preguntas, por ejemplo, la tenencia o acceso a los datos, pudieran ser responsabilidad de un grupo dedicado dentro de la organización. A menudo, el grupo de *administración de datos* (grupo AD) es responsable de administrar y controlar la informacion esencial de un negocio; cuando se comience a construir un nuevo sistema de información se requerirá hablar con estas personas para poder coordinar la información del sistema con el modelo global, corporativo, de información. *El diagrama de entidad-relación es una herramienta útil para llevar a cabo esta conversación.*

A menudo existe otro grupo dentro de la administración con un nombre similar: el grupo de *administración de bases de datos* (a veces conocido como grupo de ABD). Se suele localizar dentro del departamento de proceso de datos (mientras que el de administración de datos no necesariamente está ahí), y su labor es asegurar que las bases de datos computarizadas se organicen, administren y controlen de manera eficiente. Así que suele ser el equipo de implantación responsable de tomar el modelo esencial (independiente de alguna tecnología específica) y traducirlo a un diseño de base de datos eficaz para IMS, ADABAS, IDMS, TOTAL o algún otro sistema de base de datos. *El diagrama de entidad-relación es una herramienta efectiva de modelado para comunicarse con el grupo de administración de bases de datos.* Basándose en la información presentada por el DER, el grupo de administración de base de datos puede ver el tipo de claves o índices o apuntadores que se necesitarán para llegar de manera eficiente a los registros de las bases de datos.

Para el analista, el DER representa un gran beneficio también: enfatiza las relaciones entre almacenes de datos en el DFD que de otra forma se hubieran visto sólo en la especificación de proceso. Por ejemplo, un DER típico se muestra en la figura 12.1. Cada una de las cajas rectangulares corresponde a un almacén de datos en un DFD (correspondencia que se explorará más a fondo en el Capítulo 14), y puede verse que hay relaciones (conexiones) que normalmente no se aprecian en un DFD. Esto se debe a que el DFD enfoca la atención del lector a las *funciones* que el sistema efectúa, no a los datos que ocupa.

Considere un caso extremo: ¿Qué tal si no se están realizando funciones? ¿Qué tal si el propósito del sistema que está construyendo no es *hacer* algo, sino meramente ser el recipiente de una gran cantidad de información interesante? Tal

sistema pudiera llamarse sistema de consultas ad hoc, o sistema de apoyo a decisiones. En tal tipo de sistema, pudiéramos concentrarnos por completo en el modelo de datos, y ni siquiera preocuparnos por construir el modelo de DFD, que está orientado a las funciones. Desde luego, esto es claramente una situación rara: la mayoría de los sistemas sí efectúan funciones; a menudo, encontramos que construir primeramente el modelo de datos hace más fácil descubrir cuáles son las funciones requeridas.

Desde luego, la notación del DER de la figura 12.1 es bastante misteriosa hasta el momento. En las siguientes secciones se examinará la estructura y los componentes de un DER para luego discutir las reglas para dibujar un DER bien estructurado. La notación que se presenta en este capítulo se deriva de [Flavin, 1981] y es similar a la que desarrollaron [Chen, 1976], [Martin, 1982], [Date, 1986] y otros.

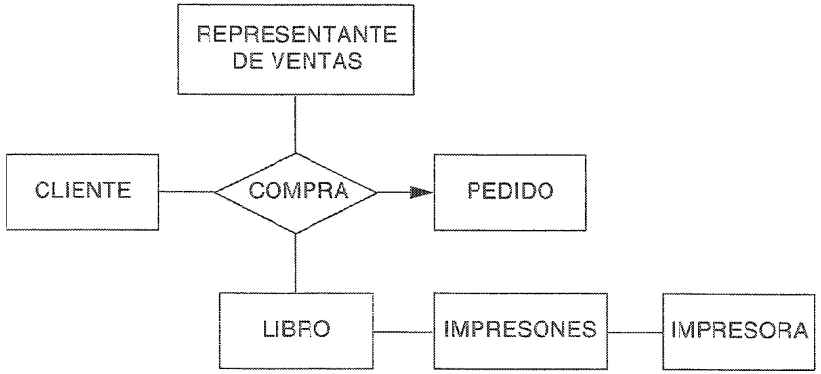


Figura 12.1: **Diagrama de entidad-relación típico**

12.1 LOS COMPONENTES DE UN DER

Hay cuatro componentes principales en un diagrama de entidad-relación:

1. Tipos de objetos
2. Relaciones
3. Indicadores asociativos de tipo de objeto
4. Indicadores de supertipo/subtipo

12.1.1 Tipos de objetos

El tipo de objeto se representa en un diagrama de entidad-relación por medio de una caja rectangular; en la figura 12.2 se muestra un ejemplo. Representa una

colección o conjunto de *objetos* (cosas) del mundo real cuyos miembros individuales (o instancias) tienen las siguientes características:

- *Cada una puede identificarse de manera única por algún medio.* Existe alguna forma de diferenciar entre instancias individuales del tipo de objeto. Por ejemplo, si se tiene un tipo de objeto conocido como **CLIENTE,** debemos ser capaces de distinguir uno de otro (tal vez por un número de cuenta, por su apellido, o por su número de Seguro Social). Si todos los clientes son iguales (si hay un negocio en el que son sólo entes sin cara y sin nombre que entran a la tienda a comprar cosas), entonces **CLIENTE** no sería un tipo de objeto con significado.

Figura 12.2: **Un tipo de objeto**

- *Cada uno juega un papel necesario en el sistema que se construye.* Es decir, para que el tipo de objeto sea legítimo, debe poder decirse que *el sistema no puede operar sin tener acceso a esos miembros.* Si se está construyendo un sistema de ingreso de pedidos para la tienda, por ejemplo, se pudiera pensar que, además de los clientes, la tienda tiene mozos, cada uno de los cuales se identifica de manera individual por su nombre. A pesar de que los mozos juegan un papel útil en la tienda, el sistema de ingreso de pedidos puede funcionar felizmente sin ellos; por tanto, no merecen un papel como tipos de objeto en el modelo del sistema. Obviamente, esto es algo que debe verificarse con los usuarios al construir el modelo.

- *Cada uno puede describirse por uno o más datos.* Es decir, un **CLIENTE** puede describirse por medio de datos tales como nombre, domicilio, límite de crédito y número telefónico. Muchos textos sobre bases de datos describen esto como "asignar datos a un tipo de objeto". Nótese que los atributos deben aplicarse a *cada instancia* del tipo de objeto; por ejemplo, cada cliente debe tener nombre, domicilio, límite de crédito, número telefónico, etc.

En muchos de los sistemas que desarrolle, los tipos de objetos serán la representación en el sistema de *algo material* del mundo real. Esto significa que los clientes, artículos de inventario, empleados, partes manufacturadas, etc, son objetos

típicos. El *objeto* es el algo material del mundo real, y el *tipo de objeto* es su representación en el sistema. Sin embargo, un objeto también pudiera ser algo no material: por ejemplo, horarios, planes, estándares, estrategias y mapas.

Dado que a menudo las personas son tipos de objetos en un sistema, debe tenerse otra cosa en mente: una persona (o para el caso, cualquier cosa material) pudiera ser diversos tipos de objetos distintos en distintos modelos de datos, *o incluso en un mismo modelo.* Juan Pérez, por ejemplo, puede ser **EMPLEADO** en un modelo de datos y **CLIENTE** en otro. También pudiera ser **EMPLEADO** y **CLIENTE** dentro del mismo modelo.

Nótese que en todos los ejemplos de un objeto se ha usado un sustantivo singular (por ejemplo, empleado, cliente). Esto no es necesario, pero es un convenio útil; como veremos en el Capítulo 14, existe una correspondencia entre objetos en el DER y almacenes en el DFD; así, si existe un objeto **CLIENTE** en el DER, debe haber un almacén de **CLIENTES** en el DFD.

12.1.2 Relaciones

Los objetos se conectan entre sí mediante *relaciones.* Una relación representa un conjunto de conexiones entre objetos, y se representa por medio de un rombo. La figura 12.3 muestra una relación sencilla que pudiera existir entre dos o más objetos.

Figura 12.3: **Una relación**

Es importante reconocer que la relación representa un *conjunto de conexiones.* Cada instancia de la relación representa una asociación entre cero o más ocurrencias de un objeto y cero o más ocurrencias del otro. Así, en la figura 12.3, la relación etiquetada como **COMPRAS** puede contener las siguientes instancias individuales:

- instancia 1: el cliente 1 compra el artículo 1
- instancia 2: el cliente 2 compra los artículos 2 y 3
- instancia 3: el cliente 3 compra el artículo 4
- instancia 4: el cliente 4 compra los artículos 5, 6 y 7

- instancia 5: el cliente 5 *no* compra ningún artículo
- instancia 6: los clientes 6 y 7 compran el artículo 8
- instancia 7: los clientes 8, 9 y 10 compran los artículos 9, 10 y 11
- etc.

Como puede verse, entonces, una relación puede conectar dos o más instancias del mismo objeto.

Nótese que la relación representa algo que debe ser *recordado* por el sistema: algo que no pudo haberse calculado ni derivado mecánicamente. Así, el modelo de datos de la figura 12.3 indica que existe alguna razón relacionada con el usuario para recordar el hecho de que el cliente 1 compra el artículo 1, etc. Y la relación también indica que no existe nada *a priori* que hubiera permitido determinar que el cliente 1 compró el artículo 1 y nada más. *La relación representa la memoria del sistema.*[1] (Un objeto representa la memoria del sistema también, desde luego.)

Nótese también que puede existir más de una relación entre dos objetos. La figura 12.4, por ejemplo, muestra dos relaciones distintas entre un **PACIENTE** y un **MEDICO.** A primera vista, pudiera pensarse que esto es rebuscar lo obvio: cada vez que el médico trata a un paciente, le cobra mediante un recibo. Pero la figura 12.4 sugiere que la situación puede ser distinta; pudiera resultar, por ejemplo, que existan diversas instancias de "tratamiento" entre un médico y el mismo paciente (es decir, un tratamiento inicial, tratamientos de seguimiento, etc.). La figura 12.4 implica que la relación de recibos está completamente separada de la relación de tratamiento: tal vez a algunos pacientes se les dan recibos sólo por su primer tratamiento, mientras que a otros se les dan para cada tratamiento y a otros jamás se les dan.

Una situación más común es ver múltiples relaciones entre múltiples objetos. La figura 12.5 muestra la relación que existe típicamente entre un cliente, un vendedor, un agente de bienes raíces, el abogado del cliente y el abogado del vendedor, para la compra-venta de una casa.

Con un diagrama complejo como el de la figura 12.5 (que es típico, y tal vez más simple que los DER que es probable encontrar en un proyecto real), la relación y sus tipos de objetos deben leerse como una unidad. La relación se puede describir desde la perspectiva de *cualquiera* de los tipos de objetos participantes, y *todas*

1 Entre los objetos, pueden existir otras relaciones que no aparezcan en el DER. Dado que el DER es un *modelo de datos almacenados,* no se muestran las relaciones que se pueden calcular o derivar. Por ejemplo, considere el objeto **CONDUCTOR** y el objeto **LICENCIA.** Puede existir entre los dos una relación de fecha de renovación que se ha calculado con base en la fecha de nacimiento del conductor (por ejemplo, un conductor debe renovar su licencia cada año en su cumpleaños). Tal relación no se mostraría en el DER pues no es una relación de datos almacenados. Sin embargo, si la fecha de renovación se escogiera al azar, probablemente tendría que ser recordada por el sistema.

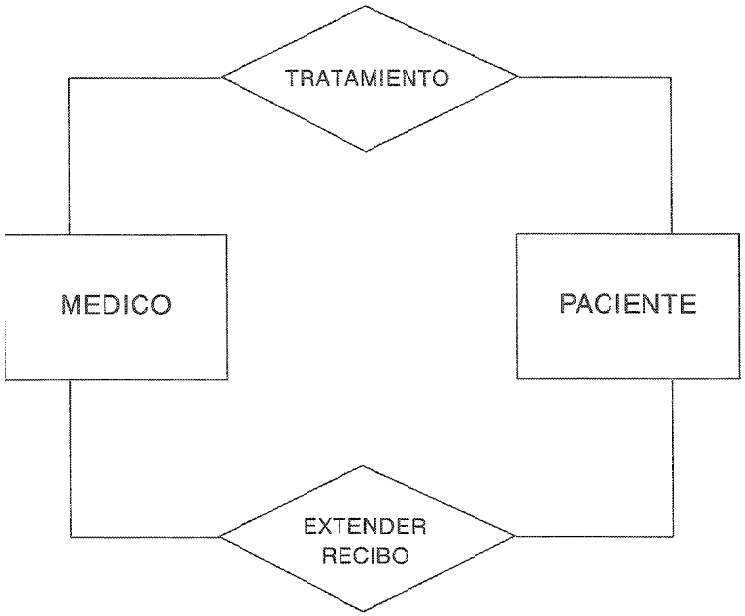


Figura 12.4: **Relaciones múltiples entre objetos**

esas perspectivas son válidas. De hecho, el conjunto de todos estos puntos de vista es el que describe completamente la relación. Por ejemplo, en la figura 12.5 puede verse la relación de negociación de precios en cualquiera de las siguientes tres formas:

1. El agente de bienes raíces negocia el precio entre el cliente y el vendedor.
2. El cliente negocia el precio con el vendedor, mediante el agente de bienes raíces.
3. El vendedor negocia el precio con el cliente, mediante el agente de bienes raíces.

Nótese que, en algunos casos, puede haber relaciones entre diferentes instancias de un mismo tipo de objeto. Por ejemplo, imagine un sistema que se esté desarrollando para una universidad, en el cual **CURSO, ESTUDIANTE** y **PROFESOR** son tipos de objetos. La mayoría de las relaciones de interés serán entre instancias de diferentes tipos de objetos (por ejemplo, las relaciones "se inscribe en", "imparte", etc.). Sin embargo, pudiera requerirse modelar la relación "es prerrequisito para" entre una instancia de **CURSO** y otra.

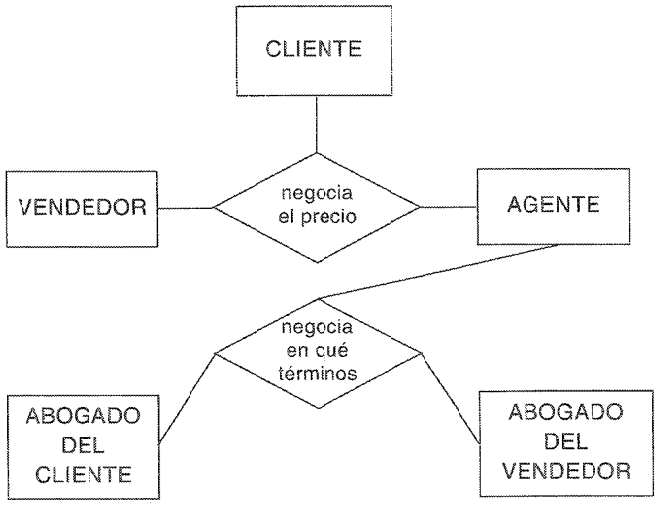


Figura 12.5: **Relaciones múltiples entre múltiples objetos**

12.1.3 Notación alternativa para relaciones

Como se vio en la Sección 12.1.2, las relaciones en el diagrama E-R son *multidireccionales;* pueden leerse siguiendo cualquier dirección. Además, vimos que los diagramas E-R no muestran *cardinalidad;* es decir, no muestran el número de objetos que participan en la relación. Esto es consciente y deliberado: se prefiere dejar tales detalles en el diccionario de datos. Esto se discutirá más a fondo en la Sección 12.3.

Una notación alternativa utilizada por algunos analistas muestra tanto la *cardinalidad* como la *ordinalidad.* Por ejemplo, la figura 12.6(a) muestra una relación entre **CLIENTE** y **ARTICULO** en la cual la notación acicional indica que:

(1) El **CLIENTE** es el punto de ancla, es decir, el objeto primario desde cuyo punto de vista debe leerse la relación.[2]

(2) La relación consiste en un cliente conectado con N artículos. Es decir, un cliente individual puede adquirir 0, 1, 2, ... o N artículos. Sin embargo, la relación incica que sólo puede haber *un* cliente involucrado en cada instancia de la relación. Esto excluye, por ejemplo, la posibilidad de que múltiples clientes estuvieran involucrados en la compra de un solo artículo

2 El término punto de ancla se introdujo en [Flavin, 1981].

Figura 12.6(a): **Notación de punto ancla para diagramas E-R**

Otra notación común aparece en la figura 12.6(b), en donde la flecha de dos puntas seguidas muestra la relación de uno a muchos, mientras que se emplea una flecha sencilla para mostrar relaciones de uno a uno entre objetos.

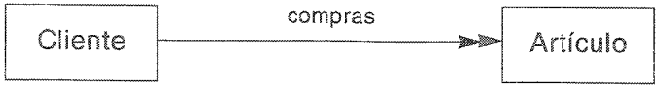


Figura 12.6(b): **Notación alternativa para relaciones de uno a muchos**

Tales diagramas de E-R anotados se discuten con detalle en [Chen, 1976], [Martin, 1982], y [Date, 1986]. Sin embargo, prefiero *no* agregar tales detalles, porque se pueden incluir fácilmente en el diccionario de datos (como se discutirá en la Sección 12.4), y porque tienden a distraer la atención del propósito principal del diagrama E-R, que es dar una visión *global* de los componentes e interfaces entre datos en un sistema. Aunque no hay nada intrínsecamente malo en tener anotaciones de tipo procedimiento en el diagrama, mi experiencia indica que los analistas a menudo llevan más allá una buena idea y atiborran el diagrama con demasiada información.

12.1.4 Indicadores asociativos de tipo de objeto

Una notación especial en el diagrama de E-R es el *indicador asociativo de tipo de objeto;* representa algo que funciona como objeto y como relación. Otra manera de ver esto es considerar que el tipo asociativo de objeto representa una *relación acerca de la cual se desea mantener alguna información.*[3]

Considere, por ejemplo, el caso sencillo de un cliente que adquiere un artículo (o artículos), que ilustra la figura 12.6. Sin tener en cuenta si se incluye o no la anotación de tipo procedimiento, el punto principal es que la *relación de* **COMPRA** *no hace más que asociar un* **CLIENTE** *con uno o más* **ARTICULOS.** Pero suponga que existen datos que deseamos recordar acerca de cada instancia de una compra (por ejemplo, a qué hora del día se hizo). ¿Dónde se podría almacenar dicha información? "Hora del día" definitivamente no es un atributo de **CLIENTE,** ni de **ARTICULO.** Más bien, se asocia "hora del día" con la compra misma, y esto se muestra en un diagrama como el que ilustra la figura 12.7.

3 En diversos libros de bases de datos esto se conoce como datos de intersección.

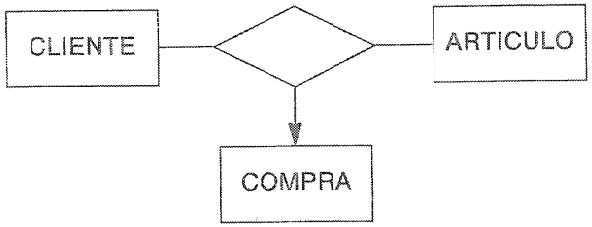


Figura 12.7: **Indicador asociativo de tipo de objeto**

Nótese que **COMPRA** ahora se escribe dentro de una caja rectangular conectada, por medio de líneas dirigidas, a un rombo de relación sin nombre. Esto pretende indicar que **COMPRA** funciona como:

- Un *tipo de objeto,* algo acerca de lo cual se desea almacenar información. En este caso, se desea recordar la hora en la cual se realizó la compra y el descuento que se dio al cliente. (Nuevamente, esto supone que tal información no la puede derivar, después del hecho, el sistema).
- Una *relación* que conecta los dos tipos de objeto **CLIENTE** y **ARTICULO.** Lo significativo aquí es que **CLIENTE** y **ARTICULO** se mantienen solos. *Existirían con o sin la compra.*[4]

Una **COMPRA,** por otro lado, obviamente depende para su misma existencia del **CLIENTE** y del **ARTICULO.** Aparece *sólo como resultado de una relación entre los otros objetos con los cuales está conectada.*

La relación de la figura 12.7 no tiene nombre a propósito. Esto se debe a que el indicador asociativo de objeto **(COMPRA)** *también* es el nombre de la relación.

12.1.5 Indicadores de subtipo/supertipo

Los tipos de objeto de subtipo/supertipo consisten en tipos de objeto de una o más subcategorías, conectados por una relación. La figura 12.8 muestra un subtipo/supertipo típico: la categoría general es **EMPLEADO** y las subcategorías son **EMPLEADO ASALARIADO** y **EMPLEADO POR HORAS.** Nótese que los subtipos se conectan al supertipo por medio de una relación sin nombre; note también que el supertipo se conecta a la relación con una línea que contiene una barra.

4 Un purista pudiera argumentar que esto no se cumple a la larga. Si no hubiera **ARTICULOS** durante varios días seguidos, los **COMPRADORES** desaparecerían de la escena y comprarían en otra parte. Y si no hubiera **COMPRADORES,** la tienda tendría que cerrar y los **ARTICULOS** desaparecerían. Pero en la situación de estado estable a corto plazo, es obvio que compradores y artículos pueden coexistir felizmente sin tener algo que ver necesariamente unos con otros.

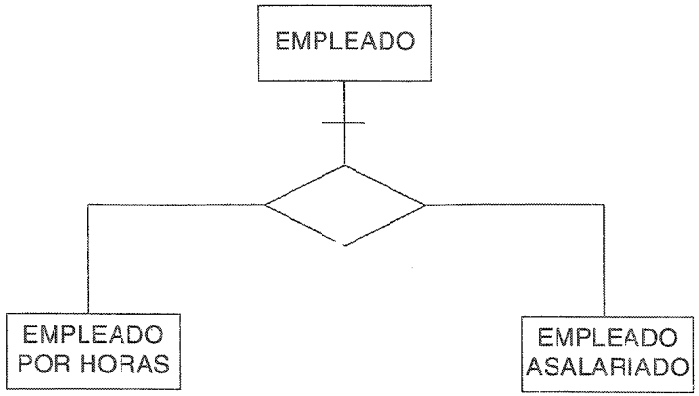


Figura 12.8: **Indicador de subtipo/supertipo**

En esta notación el supertipo se describe por datos que se aplican a *todos* los subtipos. Por ejemplo, en la figura 12.8, se podría imaginar que todos los empleados se describen por hechos tales como:

- Nombre
- Años de servicio
- Domicilio particular
- Nombre del supervisor

Sin embargo, cada subtipo se describe por medio de datos *diferentes;* de otro modo, no tendría caso hacer distinción entre ellos. Por ejemplo, se podría imaginar que un **EMPLEADO ASALARIADO** se describe por cosas tales como:

- Salario mensual
- Porcentaje anual adicional
- Aportación para coche de la empresa

Y el **EMPLEADO POR HORAS** por medio de cosas como:

- Paga por hora
- Cantidad por tiempo extra
- Hora de comienzo

12.2 REGLAS PARA LA CONSTRUCCION DE DIAGRAMAS DE ENTIDAD-RELACION

La notación que se muestra en la sección anterior es suficiente para construir diagramas E-R arbitrariamente complejos. Sin embargo, podría estar pensando en este momento: "¿Cómo descubrir qué son, para comenzar, los objetos y las relaciones?". El modelo inicial de objetos y relaciones usualmente se derivará de 1) su comprensión de la aplicación del usuario, 2) entrevistas con el usuario y 3) cualquier otro tipo de investigación y recolección de información que pueda usar. (En el Apéndice E se discuten técnicas de entrevista y de recolección de datos.)

No espere que el primer diagrama E-R que haga sea el final, que revisará con la comunidad usuaria o que entregará a los diseñadores del sistema. Como los diagramas de flujo de datos y todas las demás herramientas de modelado, los diagramas E-R deben revisarse y mejorarse muchas veces; la primera versión típicamente no será más que un borrador, y las versiones subsecuentes se producirán utilizando una serie de reglas de refinamiento que se presentan en esta sección. Algunas de las reglas de refinamiento llevan a la creación de tipos adicionales de objeto, mientras que otras llevarán a la eliminación de objetos y/o relaciones.

12.2.1 Añadir tipos de objetos adicionales

Como se indicó anteriormente, el primer DER típicamente se creará a partir de entrevistas iniciales con el usuario, y de su conocimiento de la materia en cuanto al negocio del usuario. Esto, desde luego, le dará una buena pista respecto a la identidad de los principales objetos y relaciones.[5]

Después de haber desarrollado el primer DER, el siguiente paso es asignar los datos del sistema a los diversos tipos de objetos. Se supone, desde luego, que sabe cuáles son los datos. Esto puede suceder en cualquiera de tres maneras:

1. Si el modelo del proceso (el DFD) ya se ha desarrollado o se está desarrollando paralelamente al modelo de datos, entonces el diccionario de datos ya existirá. Pudiera no estar completo aún, pero lo que haya será suficiente para comenzar el proceso de asignación.

5 Sin embargo, probablemente no identificará *todas* las relaciones entre objetos. Dado un sistema con N objetos, ¡el número de relaciones posibles es proporcional a N!, lo cual típicamente es un número *enorme*. Muchas de las relaciones (teóricamente) posibles resultarán en 1) no tener un significado legítimo dentro del sistema, o 2) no tener relevancia dentro del contexto en el que se está modelando. Se puede uno imaginar, por ejemplo, un sistema en donde la relación primaria entre compradores y vendedores es **VENDER.** Pudiera también resultar que comprador y vendedora estén casados el uno con la otra; o que la vendedora sea hija del comprador; o que el comprador y el vendedor sean condiscípulos en la escuela. Todo esto pudiera ser muy interesante, pero no se va a representar en el modelo a menos que le sea relevante.

2. Si el modelo del proceso no se ha desarrollado (o, en el caso extremo, si no tiene intención de desarrollar uno), entonces pudiera tener que empezar por entrevistar a todos los usuarios apropiados para construir una lista exhaustiva de datos (y sus definiciones). Al hacer esto, puede asignar los datos a los objetos en el diagrama de E-R. (Sin embargo, note que esto es un proceso ascendente que consume tiempo, y que pudiera ocasionar retrasos y frustración.)

3. Si está trabajando con un grupo activo de administración de datos, hay una buena probabilidad de que ya exista un diccionario de datos, que podría obtenerse pronto durante el proyecto, de manera que en ese momento ya pudiera comenzar el proceso de asignación.

El proceso de asignación puede ofrecer una de tres razones para crear nuevos tipos de objetos:

1. Es posible descubrir datos que se pueden asignar a algunas instancias de un tipo de objeto pero no a otras.

2. Pudieran descubrirse datos aplicables a todas las instancias de dos objetos distintos.

3. Podría descubrirse que algunos datos describen relaciones entre otros tipos de objetos.

Si durante el proceso de asignar datos a tipos de objetos encuentra que algunos datos *no se pueden* aplicar a todas las instancias de algún tipo de objeto dado, necesitará crear un conjunto de subtipos abajo del tipo de objeto con el que ha estado trabajando, y asignar los datos específicos a los subtipos apropiados.

Suponga que, por ejemplo, se está desarrollando un sistema de personal, y se ha identificado (con gran perspicacia y creatividad) un tipo de objeto llamado **EMPLEADO.** Al revisar los datos disponibles se encuentra que muchos de ellos **(edad, estatura, fecha de contratación,** etc.) se aplican a todas las instancias de un empleado. Sin embargo, luego se descubre un dato llamado **número-de-embarazos;** se trata obviamente de un dato relevante para las empleadas, pero no para los empleados varones. Esto llevaría a crear **EMPLEADOS-VARONES** y **EMPLEADAS** como subtipos de la categoría general de empleado.

Obviamente, no estoy sugiriendo que todos los sistemas de personal deban hacer seguimiento del número de embarazos que cada empleada ha tenido; el ejemplo se escogió meramente porque existe un consenso general de que los empleados varones *no se pueden* embarazar. Compare esto, sin embargo, con el dato **nombre-del-cónyuge:** hay varias instancias de **EMPLEADO** para quienes no se aplicaría esto (porque son solteros), pero es una situación muy distinta a la de un dato que *no se puede* aplicar definitivamente.[6]

En la mayoría de los casos el proceso de crear nuevos subtipos y asignarles datos de manera apropiada es bastante directo. Sin embargo, debe tenerse siempre en mente una situación excepcional: pudiera suceder que *todos* los datos relevantes se atribuyan a uno de los subtipos, y que *ninguno* de los datos se pueda asignar al objeto supertipo; es decir, puede suceder que los datos sean mutuamente excluyentes, perteneciendo a un subtipo o a otro pero no a ambos. Suponga, por ejemplo, que los únicos datos que se puede asignar a los empleados son **número-de-embarazos** y **años-de-experiencia-jugando-con-el-equipo-Knicks-de-basquetbol.** Podría decidirse (tras preguntarnos a qué lunático usuario pudiera habérsele ocurrido tal sistema) que el supertipo general **EMPLEADO** no se aplica.

También puede ocurrir la situación inversa: los datos pueden describir instancias de dos (o más) tipos distintos de objetos de la misma manera. Si esto ocurre, debe crearse un supertipo nuevo y asignarle los datos comunes al supertipo. Por ejemplo, tal vez se identificó **CLIENTE-DE-CONTADO** y **CLIENTE-A-CREDITO** como dos tipos de objetos distintos cuando se creó el DER para un sistema de pedidos (tal vez porque el usuario señaló que eran dos categorías distintas). Sin embargo, pronto se hace evidente que los datos **nombre-del-cliente** y **domicilio-del-cliente** describen ambos tipos de cliente de la misma forma, lo cual apoyaría la creación de un supertipo, como muestra la figura 12.9.

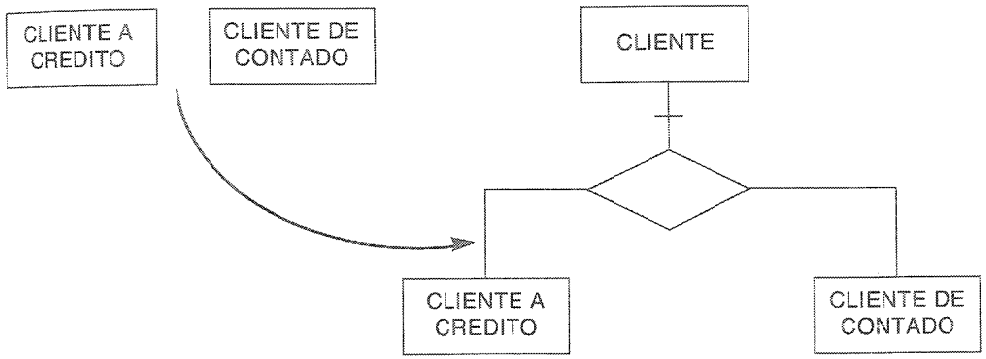


Figura 12.9: **Creación de un nuevo objeto subtipo/supertipo**

6 A lo largo de todo el ejemplo obviamente ignoramos una serie de excepciones obscuras. Ignoramos, por ejemplo, el caso de la empleada que tuvo tres embarazos y luego se hizo una operación de cambio de sexo. Para el caso de los nombres de los cónyuges suponemos que ninguno de los empleados es un niño, porque se supone que les sería imposible estar casados.

Similarmente, si un dato describe la *interacción* de dos o más tipos de objetos, entonces debería reemplazarse la relación "desnuda" entre los dos objetos con un tipo asociativo de objeto. Por ejemplo, en el primer borrador de DER, que se muestra en la figura 12.10(a), existe una relación de **COMPRA** entre **CLIENTE** y **ARTICULO.** Durante la asignación de datos pudiera encontrarse con que hay un dato llamado fecha-de-compra que 1) parece pertenecer a la relación **COMPRA** y 2) obviamente describe, *o proporciona datos acerca de,* la interacción de un **CLIENTE** con un **ARTICULO.** Esto sugiere que debe sustituirse la relación **COMPRA** por un tipo asociativo de objeto, como muestra la figura 12.10(b)

Figura 12.10 (a): **Diagrama E-R inicial**

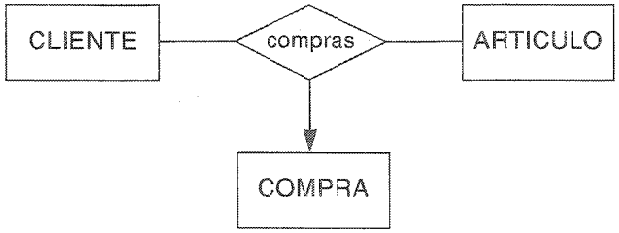


Figura 12.10 (b): **Reemplazo de una relación por un tipo asociativo de objeto**

A veces el diagrama E-R inicial tendrá un tipo de objeto que, visto de cerca, claramente amerita ser un tipo asociativo de objeto. Por ejemplo, la figura 12.11(a) muestra un diagrama E-R con tres objetos relacionados: **CLIENTE, PEDIDO** y **PRODUCTO.** Durante el proceso de asignar datos a los diversos objetos, se encuentra que **fecha-de-entrega** en realidad se aplica al objeto **PEDIDO** porque a los clientes no se les entrega en ningún lado, y los productos se entregan sólo como resultado de un pedido. De hecho, esto hace evidente que **PEDIDO** en sí es una relación entre **CLIENTE** y **PRODUCTO,** además de un objeto acerca del cual interesa recordar algunos hechos. Esto lleva a la figura 12.11(b).

Finalmente, tenemos el caso de grupos que se repiten. Considere, por ejemplo, el tipo de objeto **EMPLEADO,** con los datos obvios como nombre y domicilio. Suponga que hay datos adicionales como **nombre-del-hijo, edad-del-hijo** y **sexo-del-hijo.** Podria argumentarse obviamente que son formas de describir un objeto nuevo llamado **HIJO,** que inadvertidamente se había incluido anteriormente en **EMPLEADO.** Podría también argumentarse que existen (potencialmente) múltiples ins-

tancias de información relacionadas con hijos en cada instancia de un empleado, y que cada instancia de información relacionada con los hijos se define de manera única por el **nombre-del-hijo.** En este caso, el tipo de objeto que inicialmente se imaginó de la forma que muestra la figura 12.12(a) debe transformarse en dos objetos tipo, conectados por una nueva relación, como se muestra en la figura 12.12(b).

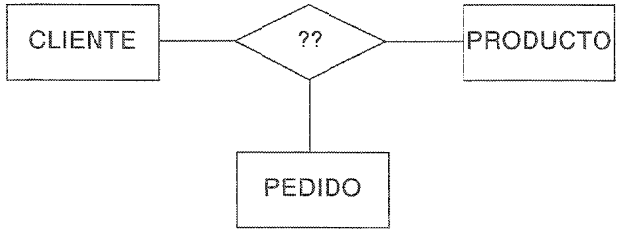


Figura 12.11(a): **DER inicial**

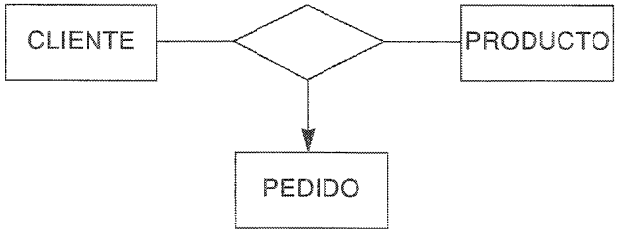


Figura 12.11(b): **Un objeto transformado en objeto asociativo**

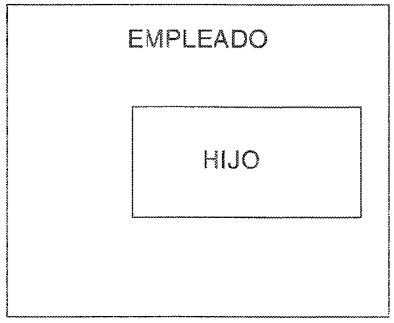


Figura 12.12(a): **Vista inicial de un objeto**

Este proceso de eliminar objetos incluidos en otros es parte de una actividad de refinamiento más general llamada *normalización*. El objetivo de la normalización es producir tipos de objetos, en los que cada instancia (o miembro) consiste en un valor llave primario que identifica a alguna entidad, junto con un conjunto de valores de atributo independientes que describen a la entidad de alguna manera. El proceso de normalización se describe con detalle en el Capítulo 14 de [Date, 1986] y en el capítulo 19 de [DeMarco, 1978].

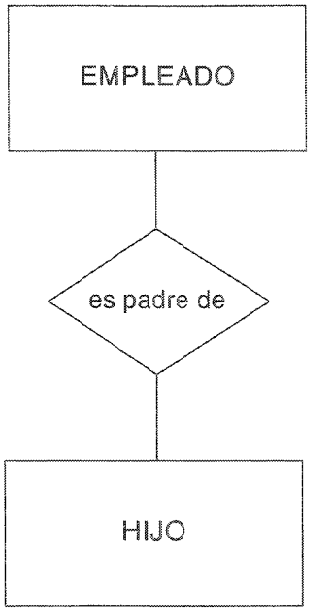


Figura 12.12(b): **Diagrama E-R revisado**

12.2.2 Eliminar tipos de objetos

La sección anterior trató los refinamientos del DER que crean objetos y/o relaciones adicionales. Sin embargo, existe un buen número de situaciones en las que los refinamientos del DER llevan a la eliminación de tipos de objetos y relaciones redundantes o erróneos. Examinaremos cuatro situaciones comunes:

1. Tipos de objetos que consisten sólo en un identificador
2. Tipos de objeto para los cuales existe una sola instancia
3. Tipos asociativos de objetos flotantes
4. Relaciones derivadas

Si se tiene un diagrama E-R en el cual uno de los tipos de objeto tiene sólo un *identificador* asignado como dato, existe la oportunidad de eliminar el tipo de objeto y asignar el identificador, como dato, a un tipo de objeto relacionado. Por ejemplo, imagine que se construyó un DER como muestra la figura 12.13(a) para un sistema de personal:

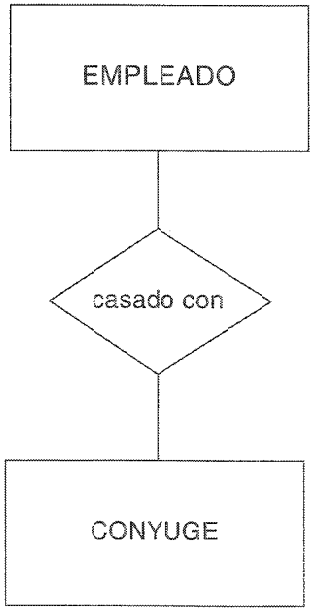


Figura 12.13(a): **Diagrama E-R inicial**

Durante el proceso de asignar datos a los diversos objetos, sin embargo, podría encontrarse que la *única* información que el sistema mantiene acerca del cónyuge es su nombre (es decir, el identificador que distingue a uno de cualquier otro en el sistema). En este caso, un refinamiento obvio sería eliminar **CONYUGE** como tipo de objeto e incluir **nombre-del-cónyuge** como dato dentro del objeto **EMPLEADO.**

Observe que este refinamiento sólo tiene sentido si existe una correspondencia uno a uno entre instancias del objeto que está a punto de ser eliminado e instancias del objeto relacionado. El ejemplo anterior tiene sentido porque la sociedad moderna supone que una persona tendrá cuando más un cónyuge. Esto lleva al diagrama E-R reducido que se muestra en la figura 12.13(b).

Figura 12.13(b): **Diagrama E-R reducido**

Se puede hacer una reducción aún mayor si encontramos que el diagrama E-R inicial contiene un objeto para el cual el único hecho es el identificador, *y* ése es un objeto de una sola instancia. Considere el diagrama E-R inicial que se muestra en la figura 12.14(a).

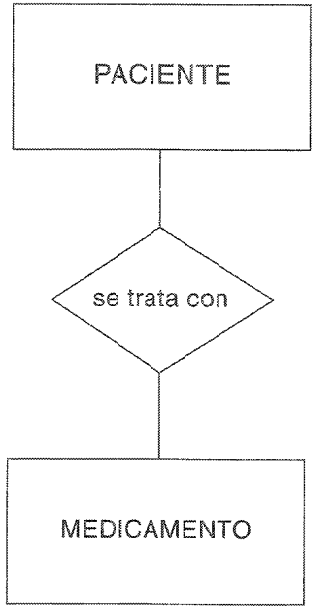


Figura 12.14(a): **Diagrama E-R inicial**

A primera vista parece ser una manera razonable de mostrar la relación entre pacientes y drogas (medicinales, claro) en un hospital. Pero suponga que la única información que se guarda acerca del medicamento es su nombre (identificador); y suponga que el hospital sólo administra un tipo de medicamento (por ejemplo, aspirina). En este caso, el medicamento es una constante y ni siquiera tiene que mostrarse en el diagrama. (Note que esto también significa que el sistema no tendría un almacén de datos llamado medicamentos.) El diagrama reducido se vería como la figura 12.14(b)

Figura 12.14(b): **Diagrama E-R reducido**

Debido a la situación anterior, es posible crear un tipo asociativo de objeto flotante. Considere la siguiente variante del ejemplo del hospital anterior, que muestra la figura 12.15(a). Si, como se sugirió anteriormente, resulta que **MEDICAMENTO** es un objeto de instancia única, sólo con identificador, entonces se eliminaría. Esto resultaría en el diagrama reducido de la figura 12.15(b); nótese que **TRATAMIENTO** todavía es un tipo de objeto asociativo, aunque se conecte sólo con un tipo de objeto. Esto se conoce como tipo de objeto asociativo flotante y es legal (aunque poco usual) en un DER.

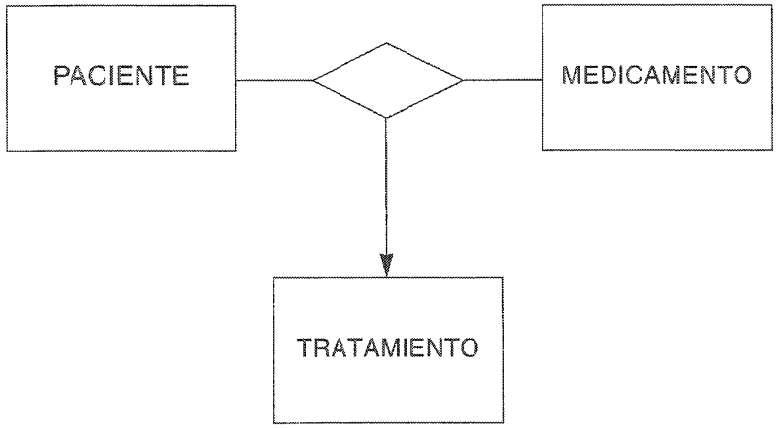


Figura 12.15(a): **Diagrama E-R inicial**

Finalmente, las relaciones que se pueden *derivar*, o calcular, deben eliminarse del diagrama E-R inicial. Como se mencionó anteriormente en este capítulo, el DER debe mostrar los requerimientos para los **datos almacenados.** Por ello, en la figura

12.16(a), si la relación **renovar** entre **CONDUCTOR** y **LICENCIA** se puede derivar (basándose en el cumpleaños del conductor, o en la primera letra de su apellido, o en algún otro esquema usado en la oficina de tránsito), entonces debe eliminarse. Esto lleva a la figura 12.16(b), en la cual los tipos de objeto *no* están conectados. Esto es legal en un DER; no es necesario que todos los tipos de objetos estén conectados entre sí.

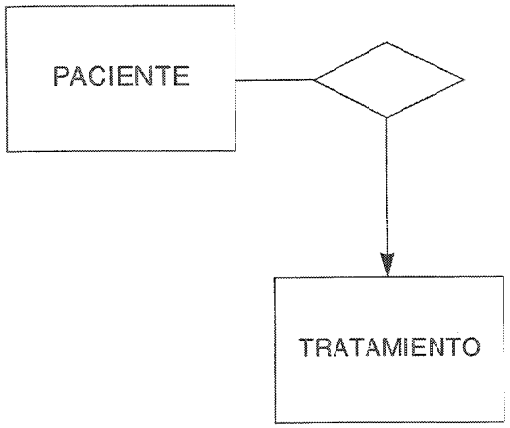


Figura 12.15(b): **Diagrama E-R reducido**

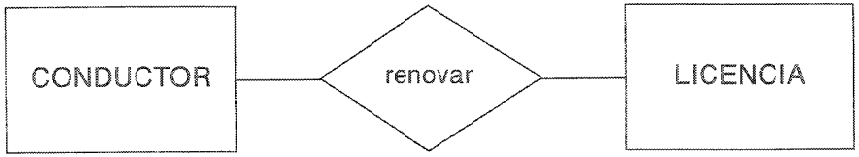


Figura 12.16(a): **DER inicial**

Figura 12.16(b): **DER reducido**

12.3 EXTENSIONES AL DICCIONARIO DE DATOS PARA DIAGRAMAS E-R

Finalmente, observamos que el diccionario de datos que se discutió en el capítulo 10 necesita extenderse para tomar en cuenta la notación de DER discutida en este capítulo. En general, los objetos del DER corresponden con *almacenes* del DFD, lo cual se analizará más a fondo en el capítulo 14. Esto significa que en la definición sacada del diccionario de datos que se da a continuación, **CLIENTE** es tanto definición del tipo de objeto como instancia del almacén **CLIENTES.**

CLIENTES = {CLIENTE}

CLIENTE = @nombre-del-cliente + domicilio + número- telefónico

Nótese también que la definición de un **CLIENTE** incluye la especificación del *campo llave,* que es el dato (o atributo) que diferencia una instancia de un cliente de cualquier otra. El signo de arriba (@) se utiliza para indicar el o los campos llave.[7]

Sin embargo, también hay que incluir en el diccionario de datos una definición de todas las *relaciones* que se muestran en el DER. La definición de relación debe incluir una descripción de su *significado* en el contexto de la aplicación; y debe indicar los objetos que forman la asociación. Los límites superiores e inferiores apropiados deben especificarse para indicar si la asociación es de uno a uno, de uno a muchos o de muchos a muchos. Por ejemplo, la relación **compras** que se muestra en la figura 12.10(a) puede definirse en el diccionario de datos de la siguiente forma:

compras = *la asociación de un cliente y uno o más artículos*
@identidad-del-cliente + 1{@identidad-del-artículo + cantidad-comprada}

12.4 RESUMEN

Para un sistema con múltiples almacenes (objetos) y relaciones complejas entre datos, el DER puede ser una herramienta valiosa. Como se vio en este capítulo, se enfoca totalmente a las relaciones entre datos, sin dar información acerca de las *funciones* que los crean o usan.

A pesar de que en este libro hemos usado el DER como una herramienta gráfica de modelado para mostrar relaciones entre datos, debe saber que se utilizan varias herramientas más para el mismo propósito; [Martin, 1982] y [Date, 1986] muestarn muchas alternativas de herramientas de modelado.

---

[7] Algunos textos usan el convenio de subrayar el o los campos clave, por lo cual se puede definir comprador como

COMPRADOR = nombre-del-comprador + domicilio + número-telefónico

Llegando hasta aquí, muchos estudiosos preguntan si debe desarrollarse primero el DFD y luego el DER, o a la inversa. Algunos incluso preguntan si es necesario realmente desarrollar *ambos* modelos, siendo que *cualquiera* de ellos provee tanta información interesante. La respuesta a la primera pregunta es sencilla: no importa qué modelo se desarrolle primero. Uno de los modelos pedirá a gritos ser desarrollado primero según sus propias preferencias, las del usuario, o la naturaleza del sistema mismo (es decir, si es rico en funciones o rico en datos). En otros casos, sin embargo, pudiera encontrarse que ambos modelos se desarrollan paralelamente; esto es particularmente común cuando el equipo del proyecto contiene un grupo bien definido de diseño de bases de datos, o cuando la organización de procesamiento electrónico de datos tiene un grupo administrador que desarrolla modelos de datos corporativos.

La segunda cuestión es más importante: ¿Realmente tiene importancia desarrollar dos modelos distintos de un sistema (y, como veremos en el capítulo 13, un tercer modelo del comportamiento dependiente del tiempo del sistema)? La respuesta es que depende del tipo de sistema que se esté desarrollando. Muchos sistemas clásicos de proceso de datos de negocios desarrollados en los años 60 y 70 parecían (por lo menos superficialmente) consistir en muchas funciones complejas, pero estructuras de datos relativamente triviales, por lo cual el modelo de DFD se enfatizaba y a menudo se ignoraba el de DER. De manera inversa, muchos de los sistemas de apoyo a decisiones y sistemas de bases de datos de investigación ad hoc que se crearon en los años 80 parecían (por lo menos superficialmente) consistir en relaciones complejas entre datos, pero casi nada de actividades funcionales; de aquí que se enfatizara el modelo de DER, y se redujera el de DFD. Las características de tiempo de los sistemas de tiempo real construidos en los años 60 y 70 parecían (por lo menos de manera superficial) dominar sobre cualquier consideración de funciones y relaciones entre datos; en tales sistemas, el modelo de transición de estados (que se discute en el capítulo 13) a menudo se enfatizaba, excluyendo los DFD y DER.

Sin embargo, los sistemas de fines de los años 80 y 90, tienden a ser bastante más complejos que los sistemas de propósito especial de hace una o dos décadas. De hecho, muchos de ellos son entre 100 y 1000 veces más grandes y complejos. Muchos de estos sistemas grandes y complejos tienen funciones, relaciones entre datos y comportamientos dependientes del tiempo increíblemente complejos; considere, por ejemplo, el sistema Guerra de las Galaxias que, según se estima, requiere de *100 millones* de instrucciones de computadora, y que tendrá un comportamiento de tiempo real extraordinariamente complejo. Para un sistema así de elaborado, es obvio que las tres herramientas que se discuten en este libro serán críticamente necesarias. Por otro lado, si Ud. se involucra en un sistema sencillo y unidimensional, puede concentrarse en la herramienta de modelado que enfatiza e ilumina el aspecto más importante de su sistema.

En el capítulo 14 veremos cómo el DER, el DFD, el diagrama de transición de estados, la especificación del proceso y el diccionario de datos pueden compararse entre sí para producir un modelo completo del sistema que sea internamente consistente.

REFERENCIAS

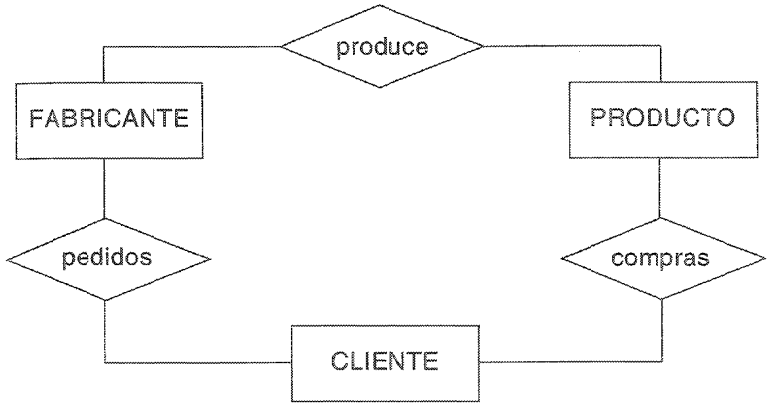
1. Matt Flavin, *Fundamental Concepts of Information Modeling,* Nueva York, YOURDON Press, 1981.
2. Peter Chen, "The Entity-Relationship Model: Toward A Unified View of Data", *ACM Transactions on Database Systems,* Vol. 1, número 1 (marzo de 1976), pp. 9-36.
3. Peter Chen, *The Entity-Relationship Approach to Logical Database Design.* Wellesley, Mass.: Q.E.D. Information Sciences, 1977.
4. D.C. Tsichritzis y F.H. Lochovsky, *Data Models,* Englewood Cliffs, N.J.: Prentice-Hall, 1982.
5. James Martin, *Computer Database Organization,* Englewood Cliffs, N.J.: Prentice-Hall, 1982.
6. *Proceedings of the International Conference on Data Engineering.* Washington, D.C.: IEEE Press, 1984.
7. C.J. Date, *An Introduction to Database Systems,* 4ª edición, Reading, Mass.: Addison-Wesley, 1986.
8. Sally Shlaer y Stephen Mellor, *Object-Oriented Systems Analysis: Modeling the World in Data.* Englewood Cliffs, N.J.: YOURDON Press, 1988.
9. R. Veryard, *Pragmatic Data Analysis.* Oxford, U.K.: Blackwell Scientific Publications, 1984.
10. Jeffrey Ullman, *Principles of Database Systems.* Potomac, Md.: Computer Science Press, 1982.
11. Tom DeMarco, *Structured Analysis and System Specification.* Nueva York, YOURDON Press, 1978.

PREGUNTAS Y EJERCICIOS

1. ¿Qué es un diagrama de entidad-relación? ¿Cuál es su propósito?
2. ¿En qué difiere un DER de un DFD?
3. ¿Por qué se interesa tanto la gente en los modelos de datos?

4. ¿Aparte de los analistas, qué otro grupo dentro de una organización pudiera crear modelos de datos?
5. ¿Por qué se interesa normalmente el grupo DBA en una organización por el modelo de datos?
6. ¿Cuáles son los cuatro principales componentes de un diagrama de entidad-relación?
7. ¿Cuál es la definición de tipo de objeto?
8. ¿Cuál es la diferencia entre un objeto y un tipo de objeto?
9. ¿Cuáles son los tres criterios que debe satisfacer un tipo de objeto?
10. ¿Cuál de los siguientes es probable que sea un tipo de objeto razonable dentro de un sistema de negocios típico? Para aquellos que no considere tipos de objetos razonables, indique por qué:
    (a) "cliente"
    (b) "calcular impuestos de ventas"
    (c) "estatura"
    (d) "producto"
    (e) "jitomate"
    (f) "religión
    (g) "temperatura"
    (h) "editar la transacción"
    (i) "parte manufacturada"
    (j) "mapa"
    (k) "carácter ASCII"
11. ¿Cuál es la definición de relación?
12. ¿Cuántos tipos de objetos pueden conectarse por una relación?
13. ¿Cuáles de las siguientes son relaciones probables en un DER y cuáles no? ¿Por qué sí y por qué no?
    (a) "compras"
    (b) "cliente"
    (c) "pertenece a"

    (d) "peso"
    (e) "produce"
    (f) "cálculo de impuestos de ventas"
14. ¿Cuál es la diferencia entre relación *derivada* y relación *recordada?* ¿Cuál se muestra en un DER?
15. Dé dos ejemplos de una relación derivada entre dos objetos.
16. ¿Cuántas relaciones pueden existir entre dos objetos en un DER?
17. Considere el DER que se muestra.



    (a) Escriba una descripción narrativa de objetos y relaciones.
    (b) ¿Cuántos pedidos pueden existir en una instancia de **fabricante** y una de **cliente?**
    (c) ¿Cuántos **productos** puede comprar un **cliente** en una instancia de la relación **compras?**
18. ¿Muestran cardinalidad los DER?
19. Use la notación de la figura 12.6 para mostrar una versión razonable del diagrama de la figura 12.5.
20. ¿Qué argumentos existen *contra* la ordinalidad y la cardinalidad en un DER?
21. ¿Cuál es la notación alternativa para los DER que muestra tanto la cardinalidad como la ordinalidad?

22. Dibuje un diagrama DER para representar la siguiente situación en una aerolínea:

"La aerolínea XYZ tiene tres recursos principales: aviones, pilotos y miembros de la tripulación. Los pilotos y miembros de la tripulación tienen sus respectivas bases, a las cuales regresan al final de un vuelo. Un vuelo debe tener por lo menos un piloto y uno o más miembros de la tripulación en un avión. Cada avión tiene una base de mantenimiento."

23. Dibuje un DER para describir la siguiente situación de una editorial:

"La editorial ABC trabaja con varios autores diferentes que escriben los libros que publica. Algunos autores han escrito sólo un libro, mientras que otros han escrito varios; además, algunos libros tienen coautoría. ABC también trabaja con múltiples imprentas: sin embargo, un libro dado lo imprime una sola imprenta. Un editor de ABC trabaja con diversos autores al mismo tiempo, editando y produciendo sus libros; es labor del editor dar a la imprenta la copia final lista para la cámara cuando se ha revisado y formado el manuscrito."

24. Dibuje un DER de la siguiente situación para una organización de consultoría de administración:

"Cada representante de ventas trabaja con diversos tipos de clientes y tiene acceso a varios consultores distintos en la organización. Una sesión de consultoría con un cliente puede requerir varios consultores. Durante la sesión, el vendedor no se involucra y los consultores rinden sus informes directamente al cliente."

25. Dibuje un DER de la siguiente situación:

"Un profesor puede impartir varias clases diferentes, siempre que esté calificado para hacerlo. Cada clase debe tener un profesor, pero pueden asistir a ella varios alumnos. Al comienzo de cada semestre, los grupos se asignan a distintos salones, donde se reúnen regularmente."

26. ¿Qué es un tipo asociativo de objeto? ¿Cuál es la diferencia entre un tipo asociativo de objeto y una relación?

27. ¿Qué es un punto de ancla?

28. Dé tres ejemplos de tipo asociativo de objeto.

29. Vea la figura 12.7. Suponga que *no* existen datos sobre la compra que el sistema deba recordar. ¿Cómo cambia esto el diagrama?

30. ¿Qué es un subtipo/supertipo en un DER?

31. Dé tres ejemplos de subtipo/supertipo.

32. ¿Por qué molestarse en tener subtipos/supertipos en un DER? ¿Por qué no se tienen simplemente tipos de objetos "ordinarios"?

33. ¿Qué refinamientos puede esperar hacer el analista después de dibujar el primer borrador del DER?

34. ¿Cuáles son las tres formas probables que el analista usaría para descubrir los datos de un modelo de datos?

35. ¿Qué significa el término asignación en el contexto de este capítulo?

36. ¿Cómo debe el analista proceder con un DER si ya está desarrollado el DFD?

37. ¿Cuáles son las tres razones para crear tipos de objetos adicionales en un DER después de terminar el primer borrador del modelo?

38. ¿Qué debiera hacer el analista si descubre datos que se pueden asignar a algunas instancias de un tipo de objeto pero no a otras?

39. ¿Qué debe hacer el analista si descubre datos que se aplican a todas las instancias de dos tipos de objeto distintos?

40. ¿Qué debe hacer el analista si descubre datos que describen relaciones entre otros tipos de objetos?

41. ¿Qué debe hacer el analista si descubre conjuntos repetidos en un tipo de objeto?

42. Describa el significado de tener conjuntos repetidos en un tipo de objeto. Dé un ejemplo.

43. ¿Cuáles son las cuatro razones comunes para eliminar un tipo de objeto en un borrador de DER?

44. ¿Qué es un tipo asociativo de objeto flotante?

45. ¿Qué debe hacer el analista si descubre un tipo de objeto que consiste sólo en un identificador en un DER?

46. ¿Qué debe hacer el analista si descubre un tipo de objeto para el cual existe una sola instancia en el borrador del DER?

47. ¿Qué debe hacer el analista si descubre una relación derivada en un borrador de DER?

48. ¿Qué extensiones deben hacérsele al diccionario de datos para manejar el DER?

49. ¿Qué significa la notación @ en un diccionario de datos?