Parte 1
Bases de datos relacionales

Un modelo de datos es un conjunto de herramientas conceptuales para la descripción de los datos, las relaciones entre ellos, su semántica y las restricciones de consistencia. Esta parte centra la atención en el modelo relacional.

El modelo de datos relacional, que se trata en el Capítulo 2, utiliza una colección de tablas para representar tanto los datos como las relaciones entre ellos. Su simplicidad conceptual le ha permitido una amplia adopción; en la actualidad la inmensa mayoría de los productos de bases de datos utilizan el modelo relacional. El modelo relacional describe los datos en los niveles lógico y de vista, abstrayendo los detalles de bajo nivel sobre el almacenamiento de los datos. El modelo entidad relación que se trata en el Capítulo 7 (Parte 2), es un modelo de datos de alto nivel que se utiliza ampliamente para el diseño de bases de datos.

Para poner a disposición de los usuarios los datos de una base de datos relacional hay que abordar varios aspectos. El aspecto más importante es determinar cómo los usuarios pueden recuperar y actualizar los datos; se han desarrollado varios lenguajes de consulta para ello. Un segundo aspecto, pero también importante, trata sobre la integridad y la protección; hay que proteger las bases de datos del posible daño por acciones del usuario, sean intencionadas o no.

Los Capítulos 3, 4 y 5 tratan sobre el lenguaje SQL, que es el lenguaje de consultas más utilizado hoy en día. En los Capítulos 3 y 4 también se realiza una descripción introductoria e intermedia de SQL. El Capítulo 4 trata así mismo de las restricciones de integridad que fuerza la base de datos y de los mecanismos de autorización, que controlan el acceso y las acciones de actualización que puede llevar a cabo un usuario. El Capítulo 5 trata temas más avanzados, incluyendo el acceso a SQL desde lenguajes de programación y el uso de SQL para el análisis de datos.

El Capítulo 6 trata sobre tres lenguajes de consulta formales: el álgebra relacional, el cálculo relacional de tuplas y el cálculo relacional de dominios, que son leguajes de consulta declarativos basados en lógica matemática. Estos lenguajes formales son la base de SQL y de otros lenguajes sencillos para el usuario, QBI y Datalog, que se describen en el Apéndice B (disponible en línea en db-book.com).

Introducción al modelo relacional

02

El modelo relacional es hoy en día el principal modelo de datos para las aplicaciones comerciales de procesamiento de datos. Ha conseguido esa posición destacada debido a su simplicidad, lo cual facilita el trabajo del programador en comparación con los modelos anteriores, como el de red y el jerárquico.

En este capítulo se estudian en primer lugar los fundamentos del modelo relacional. Existe una amplia base teórica para las bases de datos relacionales. En el Capítulo 6 se estudia la parte de esa base teórica referida a las consultas. En los Capítulos 7 y 8 se examinarán aspectos de la teoría de las bases de datos relacionales que ayudan en el diseño de esquemas de bases de datos relacionales, mientras que en los Capítulos 12 y 13 se estudian aspectos de la teoría que se refieren al procesamiento eficiente de consultas.

2.1. La estructura de las bases de datos relacionales

Una base de datos relacional consiste en un conjunto de **tablas,** a cada una de las cuales se le asigna un nombre único. Por ejemplo, suponga la tabla *profesor* de la Figura 2.1, que guarda información de todos los profesores. La tabla tiene cuatro columnas, *ID*, *nombre*, *nombre_dept* y *sueldo*. Cada fila de la tabla registra información de un profesor que consiste en el ID del profesor, su nombre, su nombre de departamento y su sueldo. De forma análoga, la tabla *asignatura*, de la Figura 2.2, guarda información de las asignaturas, que consisten en los valores *asignatura_id*, *nombre*, *nombre_dept* y *créditos* de cada asignatura. Fíjese que a cada profesor se le identifica por el valor de la columna *ID*, mientras que cada asignatura se identifica con el valor de su columna *asignatura_id*.

ID	*nombre*	*nombre_dept*	*sueldo*
10101	Srinivasan	Informática	65000
12121	Wu	Finanzas	90000
15151	Mozart	Música	40000
22222	Einstein	Física	95000
32343	El Said	Historia	60000
33456	Gold	Física	87000
45565	Katz	Informática	75000
58583	Califieri	Historia	62000
76543	Singh	Finanzas	80000
76766	Crick	Biología	72000
83821	Brandt	Informática	92000
98345	Kim	Electrónica	80000

Figura 2.1. La relación *profesor*.

La Figura 2.3 muestra una tercera tabla, *prerreq*, que guarda las asignaturas prerrequisito de otra asignatura. La tabla tiene dos columnas, *asignatura_id* y *prerreq_id*. Cada fila consta de un par de identificadores de asignatura, de forma que la segunda es un prerrequisito para la primera.

Por tanto, en una fila de la tabla *prerreq* se indica que dos asignaturas están *relacionadas* en el sentido de que una de ellas es un prerrequisito para la otra. Como otro ejemplo, considere la tabla *profesor*, en la que se puede pensar que una fila de la tabla representa la relación entre un *ID* específico y los correspondientes valores para el *nombre*, el nombre de departamento (*nombre_dept*) y el *sueldo*.

asignatura_id	*nombre*	*nombre_dept*	*créditos*
BIO-101	Introducción a la Biología	Biología	4
BIO-301	Genética	Biología	4
BIO-399	Biología computacional	Biología	3
CS-101	Introducción a la Informática	Informática	4
CS-190	Diseño de juegos	Informática	4
CS-315	Robótica	Informática	3
CS-319	Procesado de imágenes	Informática	3
CS-347	Fundamentos de bases de datos	Informática	3
EE-181	Intro. a los sistemas digitales	Electrónica	3
FIN-201	Banca de inversión	Finanzas	3
HIS-351	Historia mundial	Historia	3
MU-199	Producción de música y vídeo	Música	3
PHY-101	Fundamentos de Física	Física	4

Figura 2.2. La relación *asignatura*.

asignatura_id	*prerreq_id*
BIO-301	BIO-101
BIO-399	BIO-101
CS-190	CS-101
CS-315	CS-101
CS-319	CS-101
CS-347	CS-101
EE-181	PHY-101

Figura 2.3. La relación *prerreq*.

En general, una fila de una tabla representa una *relación* entre un conjunto de valores. Como una tabla es una colección de estas relaciones, existe una fuerte correspondencia entre el concepto de *tabla* y el concepto matemático de *relación*, de donde toma su nombre el modelo de datos relacional. En terminología matemática, una *tupla* es sencillamente una secuencia (o lista) de valores. Una relación entre n valores se representa matemáticamente como una ***n-tupla*** de valores, es decir, como una tupla con n valores, que se corresponde con una fila de una tabla.

En el modelo relacional, el término **relación** se utiliza para referirse a una tabla, mientras el término **tupla** se utiliza para referirse a una fila. De forma similar, el término **atributo** se refiere a una columna de una tabla.

En la Figura 2.1 se puede ver que la relación *profesor* tiene cuatro atributos: *ID*, *nombre*, *nombre_dept* y *sueldo*.

Se utiliza el término **ejemplar de relación** para referirse a una instancia específica de una relación, es decir, que contiene un determinado conjunto de filas. La instancia de *profesor* que se muestra en la Figura 2.1 tiene 12 tuplas, correspondientes a 12 profesores.

En este capítulo se verán distintas relaciones para tratar los distintos conceptos sobre el modelo de datos relacional. Estas relaciones representan parte de una universidad. No se trata de toda la información que tendría una base de datos de una universidad real, para simplificar la representación. En los Capítulos 7 y 8 se tratarán con mayor detalle los criterios para definir estructuras relacionales.

El orden en que las tuplas aparecen en una relación es irrelevante, ya que la relación es un *conjunto* de tuplas. Por tanto, si las tuplas de una relación se listan en un determinado orden, como en la Figura 2.1, o están desordenadas, como en la Figura 2.4, no tiene ninguna importancia; la relación de las dos figuras es la misma, ya que ambas contienen las mismas tuplas. Para facilitar la presentación, normalmente se mostrarán las relaciones ordenadas por su primer atributo.

ID	*nombre*	*nombre_dept*	*sueldo*
22222	Einstein	Física	95000
12121	Wu	Finanzas	90000
32343	El Said	Historia	60000
45565	Katz	Informática	75000
98345	Kim	Electrónica	80000
76766	Crick	Biología	72000
10101	Srinivasan	Informática	65000
58583	Califieri	Historia	62000
83821	Brandt	Informática	92000
15151	Mozart	Música	40000
33456	Gold	Física	87000
76543	Singh	Finanzas	80000

Figura 2.4. Lista no ordenada de la relación *profesor*.

Para cada atributo de una relación existe un conjunto de valores permitidos, llamado **dominio** del atributo. Por tanto, el dominio del atributo *sueldo* de la relación *profesor* es el conjunto de todos los posibles valores de sueldo, mientras que el dominio del atributo *nombre* es el conjunto de todos los posibles nombres de profesor.

Es necesario que, para toda relación r, los dominios de todos los atributos de r sean atómicos. Un dominio es **atómico** si los elementos del dominio se consideran unidades indivisibles. Por ejemplo, suponga que la tabla *profesor* tiene un atributo *número_teléfono*, que puede almacenar un conjunto de números de teléfono correspondientes al profesor. Entonces, el dominio de *número_teléfono* no sería atómico, ya que un elemento del dominio es un conjunto de números de teléfono, y tiene subpartes, que son cada uno de los números de teléfono individuales del conjunto.

Lo importante no es cuál es el dominio sino cómo utilizar los elementos del dominio en la base de datos. Suponga que el atributo *número_teléfono* almacena un único número de teléfono. Incluso en este caso, si se divide el valor del teléfono en un código de país, un código de área y un número local, se trataría como un valor no atómico. Si se trata cada número de teléfono como un valor único e indivisible, entonces el atributo *número_teléfono* sería un dominio atómico.

En este capítulo, así como en los Capítulos 3 al 6, se supone que todos los atributos tienen dominios atómicos. En el Capítulo 22 se tratarán las extensiones al modelo de datos relacional para permitir dominios no atómicos.

El valor ***null* (nulo)** es un valor especial que significa que el valor es desconocido o no existe. Por ejemplo, suponga como antes que se incluye el atributo *número_teléfono* en la relación *profesor*. Puede que un profesor no tenga número de teléfono o que no se conozca. Entonces se debería utilizar el valor *null* para indicar que el valor es desconocido o no existe. Más adelante se verá que el valor *null* genera distintas dificultades cuando se accede o actualiza la base de datos y, por tanto, si fuese posible se debería eliminar. Se supondrá inicialmente que no existen valores *null*, y en la Sección 3.6 se describe el efecto de los valores nulos sobre las distintas operaciones.

2.2. Esquema de la base de datos

Cuando se habla de bases de datos se debe diferenciar entre el **esquema de la base de datos,** que es el diseño lógico de la misma, y el **ejemplar de la base de datos,** que es una instantánea de los datos de la misma en un momento dado.

El concepto de relación se corresponde con el concepto de variable de los lenguajes de programación. El concepto **esquema de la relación** se corresponde con el concepto de definición de tipos de los lenguajes de programación.

En general, los esquemas de las relaciones consisten en una lista de los atributos y de sus dominios correspondientes. La definición exacta del dominio de cada atributo no será relevante hasta que se estudie el lenguaje SQL en el Capítulo 3.

El concepto de ejemplar de la relación se corresponde con el concepto de valor de una variable en los lenguajes de programación. El valor de una variable dada puede cambiar con el tiempo; de manera parecida, el contenido del ejemplar de una relación puede cambiar con el tiempo cuando la relación se actualiza. Sin embargo, el esquema de una relación normalmente no cambia.

Aunque es importante conocer la diferencia entre un esquema de relación y un ejemplar de relación, normalmente se utiliza el mismo nombre, como por ejemplo *profesor*, para referirse tanto a uno como a otro. Cuando es necesario, explícitamente se indica esquema o ejemplar, por ejemplo «el esquema *profesor*», o «un ejemplar de la relación *profesor*». Sin embargo, donde quede claro si significa esquema o ejemplar, simplemente se utiliza el nombre de la relación.

Suponga la relación *departamento* de la Figura 2.5. El esquema de dicha relación es:

departamento (*nombre_dept*, *edificio*, *presupuesto*)

nombre_dept	*edificio*	*sueldo*
Biología	Watson	90000
Informática	Taylor	100000
Electrónica	Taylor	85000
Finanzas	Painter	120000
Historia	Painter	50000
Música	Packard	80000
Física	Watson	70000

Figura 2.5. La relación *departamento*.

Tenga en cuenta que el atributo *nombre_dept* aparece tanto en el esquema *profesor* como en el esquema *departamento*. Esta duplicación no es una coincidencia, sino que utilizar atributos comunes en esquemas de relación es una forma de relacionar tuplas de relaciones diferentes. Por ejemplo, suponga que se desea encontrar información sobre todos los profesores que trabajan en el edificio Watson. En primer lugar se busca en la relación *departamento* los *nombre_dept* de todos los departamentos que se encuentran en el edificio Watson. A continuación, para cada relación *departamento*, se busca en la relación *profesor* para ver la información de los profesores asociados con el correspondiente *nombre_dept*.

Sigamos con el ejemplo de base de datos de una universidad.

Las asignaturas de una universidad se pueden ofertar en diversos momentos, durante distintos semestres, o incluso en un único semestre. Se necesita una relación que describa cada una de estas ofertas, o sección, de una asignatura. El esquema es:

sección (*asignatura_id*, *secc_id*, *semestre*, *año*, *edificio*, *aula*, *franja_horaria*)

La Figura 2.6 muestra un ejemplar de la relación *sección*.

Se necesita una relación que describa la asociación entre profesor y las secciones de asignaturas que imparte. El esquema de relación que describe esta asociación es:

enseña (ID, *asignatura_id*, *secc_id*, *semestre*, *año*)

La Figura 2.7 muestra un ejemplar de la relación *enseña*.

Como cabe imaginar, existen muchas más relaciones en una base de datos real de una universidad. Además de estas relaciones que ya se han visto, *profesor*, *departamento*, *asignatura*, *sección*, *prerreq* y *enseña*, se utilizarán las siguientes relaciones en el texto:

- *estudiante (ID*, *nombre*, *nombre_dept*, *total_créditos)*
- *tutor (e_id*, *p_id)*
- *matricula (ID*, *asignatura_id*, *secc_id*, *semestre*, *año*, *nota)*
- *aula (edificio*, *número_aula*, *capacidad)*
- *franja_horaria (franja_horaria_id*, *día*, *hora_inicio*, *hora_fin)*

asignatura_id	*secc_id*	*semestre*	*año*	*edificio*	*aula*	*franja_horaria*
BIO-101	1	Verano	2009	Painter	514	B
BIO-301	1	Verano	2010	Painter	514	A
CS-101	1	Otoño	2009	Packard	101	H
CS-101	1	Primavera	2010	Packard	101	F
CS-190	1	Primavera	2009	Taylor	3128	E
CS-190	2	Primavera	2009	Taylor	3128	A
CS-315	1	Primavera	2010	Watson	120	D
CS-319	1	Primavera	2010	Watson	100	B
CS-319	2	Primavera	2010	Taylor	3128	C
CS-347	1	Otoño	2009	Taylor	3128	A
EE-181	1	Primavera	2009	Taylor	3128	C
FIN-201	1	Primavera	2010	Packard	101	B
HIS-351	1	Primavera	2010	Painter	514	C
MU-199	1	Primavera	2010	Packard	101	D
PHY-101	1	Otoño	2009	Watson	100	A

Figura 2.6. La relación *sección*.

ID	*asignatura_id*	*secc_id*	*semestre*	*año*
10101	CS-101	1	Otoño	2009
10101	CS-315	1	Primavera	2010
10101	CS-347	1	Otoño	2009
12121	FIN-201	1	Primavera	2010
15151	MU-199	1	Primavera	2010
22222	PHY-101	1	Otoño	2009
32343	HIS-351	1	Primavera	2010
45565	CS-101	1	Primavera	2010
45565	CS-319	1	Primavera	2010
76766	BIO-101	1	Verano	2009
76766	BIO-301	1	Verano	2010
83821	CS-190	1	Primavera	2009
83821	CS-190	2	Primavera	2009
83821	CS-319	2	Primavera	2010
98345	EE-181	1	Primavera	2009

Figura 2.7. La relación *enseña*.

2.3. Claves

Es necesario disponer de un modo de especificar la manera en que las tuplas de una relación dada se distingan entre sí. Esto se expresa en términos de sus atributos. Es decir, los valores de los atributos de una tupla deben ser tales que puedan *identificarla unívocamente*. En otras palabras, no se permite que dos tuplas de una misma relación tengan exactamente los mismos valores en todos sus atributos.

Una **superclave** es un conjunto de uno o varios atributos que, considerados conjuntamente, permiten identificar de manera unívoca una tupla de la relación. Por ejemplo, el atributo *ID* de la relación *profesor* es suficiente para distinguir una tupla instructor de otra. Por tanto, *ID* es una superclave. El atributo *nombre* de un *profesor*, por otra parte, no es una superclave porque podría haber varios profesores con el mismo nombre.

Formalmente, sea R un conjunto de atributos en un esquema de relación r. Si se dice que un subconjunto K de R es una *superclave* de r, se restringe a ejemplares de la relación r en las que no existen dos tuplas distintas con los mismos valores en todos los atributos de K. Es decir, si existen t_1 y t_2 en r y $t_1 \neq t_2$, entonces $t_1.K \neq t_2.K$.

Una superclave puede contener atributos externos. Por ejemplo, la combinación de *ID* y *nombre* es una superclave para la relación *profesor*. Si K es una superclave, también lo es cualquier superconjunto de K. Normalmente nos interesan las superclaves para las que no hay subconjuntos que sean superclaves. Estas superclaves mínimas se denominan **claves candidatas.**

Es posible que varios conjuntos diferentes de atributos puedan ejercer como claves candidatas. Suponga que una combinación de *nombre* y *nombre_dept* es suficiente para distinguir entre los miembros de la relación *profesor*. Entonces, tanto {*ID*} como {*nombre*, *nombre_dept*} son claves candidatas. Aunque los atributos *ID* y *nombre* juntos pueden distinguir las distintas tuplas *profesor*, su combinación {*ID*, *nombre*} no forma una clave candidata, ya que el atributo *ID* por sí solo ya lo es.

Se usará el término **clave primaria** para denotar una clave candidata que ha elegido el diseñador de la base de datos como medio principal para la identificación de las tuplas de una relación. Las claves (sean primarias, candidatas o superclaves) son propiedades de toda la relación, no de cada una de las tuplas. Ninguna pareja de tuplas de la relación puede tener simultáneamente el mismo valor de los atributos de la clave. La selección de una clave representa una restricción de la empresa del mundo real que se está modelando.

Las claves candidatas deben escogerse con cuidado. Como se ha indicado, el nombre de una persona evidentemente no es suficiente, ya que puede haber muchas personas con el mismo nombre. En Estados Unidos, el atributo número de la Seguridad Social de cada persona sería una clave candidata. Dado que los residentes extranjeros no suelen tener número de la seguridad social, las empresas internacionales deben generar sus propios identificadores unívocos. Una alternativa es usar como clave alguna combinación exclusiva de otros atributos.

La clave primaria debe escogerse de manera que los valores de sus atributos no se modifiquen nunca, o muy rara vez. Por ejemplo, el campo domicilio de una persona no debe formar parte de la clave primaria, ya que es probable que se modifique. Por otra parte, está garantizado que los números de la Seguridad Social no cambian nunca. Los identificadores exclusivos generados por las empresas no suelen cambiar, salvo si se produce una fusión entre dos de ellas; en ese caso, puede que el mismo identificador haya sido emitido por ambas empresas, y puede ser necesaria una reasignación de identificadores para garantizar que sean únicos.

Los atributos de clave primaria de un esquema de relación se indican antes que el resto de los atributos; por ejemplo, el atributo *nombre_dept* del departamento aparece el primero, ya que es la clave primaria. Los atributos de la clave primaria se indican subrayados.

El esquema de una relación, por ejemplo r_1, puede incluir entre sus atributos la clave primaria de otro esquema de relación, por ejemplo r_2. Este atributo se denomina **clave externa** de r_1, que hace referencia a r_2. La relación r_1 también se denomina **relación referenciante** de la dependencia de clave externa, y r_2 se denomina **relación referenciada** de la clave externa. Por ejemplo, el atributo *nombre_dept* de *profesor* es una clave externa de *profesor* que hace referencia a *departamento*, ya que *nombre_dept* es la clave primaria de *departamento*. En cualquier ejemplar de la base de datos, dada cualquier tupla, por ejemplo t_a, de la relación *profesor*, debe haber alguna tupla, por ejemplo t_b, en la relación *departamento* tal que el valor del atributo *nombre_dept* de t_a sea el mismo que el valor de la clave primaria de t_b *nombre_dept*.

Suponga ahora las relaciones *sección* y *enseña*. Sería razonable exigir que si existe una sección para una asignatura, deba enseñarla al menos un profesor; sin embargo, podría ser que la enseñase más de un profesor. Para forzar esta restricción, se debería requerir que si una determinada combinación (*asignatura_id*, *secc_id*, *semestre*, *año*) aparece en una sección, entonces esa misma combinación aparezca en la relación *enseña*. Sin embargo, este conjunto de valores no es una clave primaria de *enseña*, ya que puede haber más de un *profesor* que enseñe dicha *sección*. Por tanto, no se puede declarar una restricción de clave externa de *sección* a *enseña* (aunque se puede definir una restricción de clave externa en la otra dirección, de *enseña* a *sección*).

La restricción de *sección* a *enseña* es un ejemplo de **restricción de integridad referencial;** una restricción de integridad referencial requiere que los valores que aparecen en determinados atributos de una tupla en la relación referenciante también aparezcan en otros atributos de al menos una tupla de la relación referenciada.

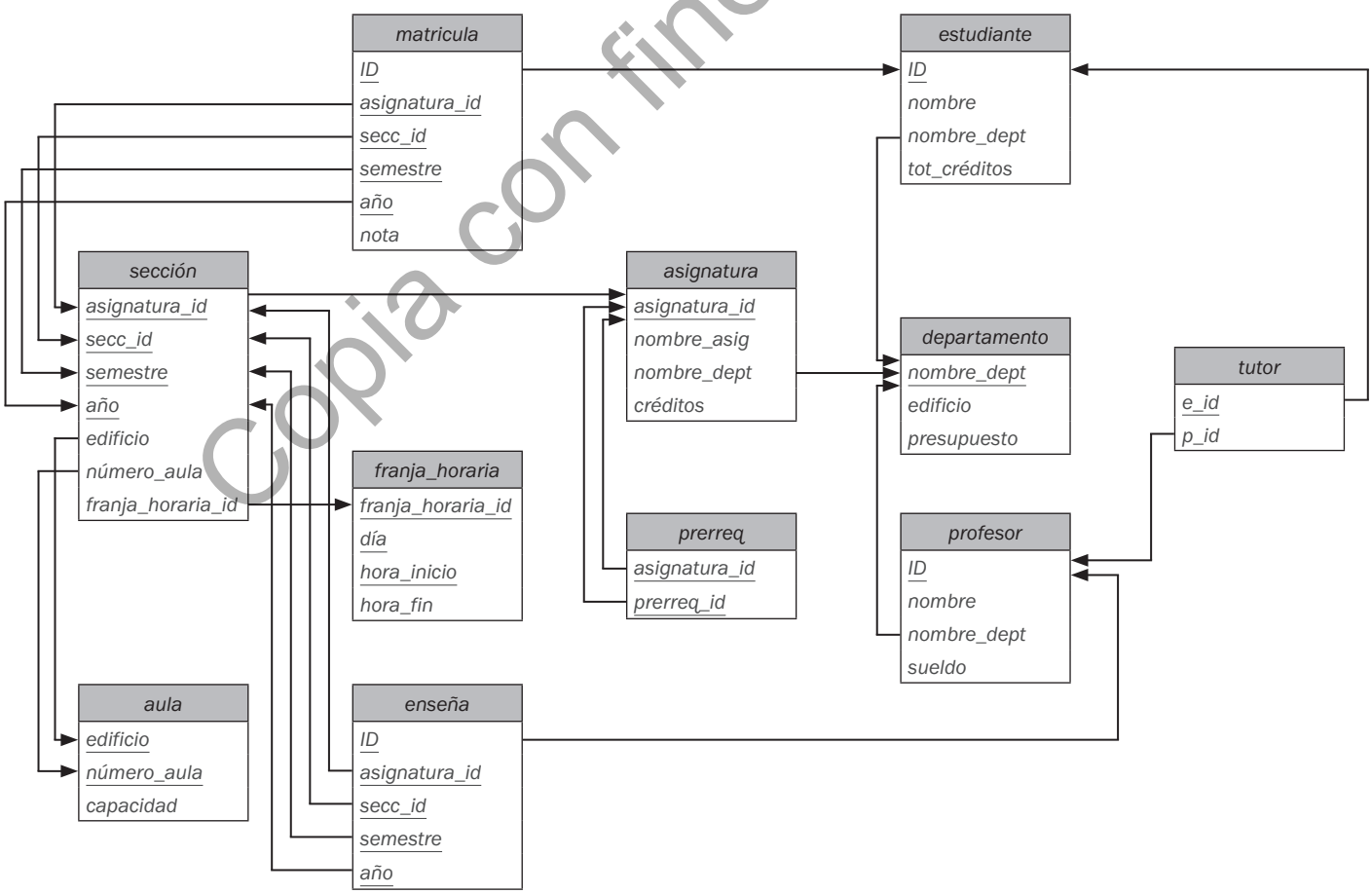

Figura 2.8. Diagrama de esquema para la base de datos de la universidad.

2.4. Diagramas de esquema

El esquema de la base de datos, junto con las dependencias de clave primaria y externa, se puede mostrar gráficamente mediante **diagramas de esquema.** La Figura 2.8 muestra el diagrama de esquema de nuestra universidad. Cada relación aparece como un cuadro con el nombre de la relación en la parte superior en un recuadro gris y los atributos en el interior del recuadro blanco. Los atributos que son clave primaria se muestran subrayados. Las dependencias de clave externa aparecen como flechas desde los atributos de clave externa de la relación referenciante a la clave primaria de la relación referenciada.

Las restricciones de integridad referencial distintas de las restricciones de clave externa no se muestran explícitamente en los diagramas de esquema. Se estudiará una representación gráfica diferente denominada diagrama entidad-relación más adelantes en el Capítulo 7. Los diagramas entidad-relación permiten representar distintos tipos de restricciones, incluyendo las restricciones de integridad referencial general.

Muchos sistemas de bases de datos proporcionan herramientas de diseño con una interfaz gráfica de usuario para la creación de los diagramas de esquema. En el Capítulo 7 se tratarán en profundidad las representaciones diagramáticas de los esquemas.

El ejemplo que su utilizará en los próximos capítulos es una universidad. La Figura 2.9 representa el esquema relacional que se utilizará en el ejemplo, en el que los atributos de clave primaria aparecen subrayados. Como se verá en el Capítulo 3, se corresponde con el enfoque de definir las relaciones en el lenguaje de definición de datos de SQL.

2.5. Lenguajes de consulta relacional

Un **lenguaje de consulta** es un lenguaje en el que los usuarios solicitan información de la base de datos. Estos lenguajes suelen ser de un nivel superior al de los lenguajes de programación habituales. Los lenguajes de consultas pueden clasificarse como procedimentales o no procedimentales. En los **lenguajes procedimentales,** el usuario indica al sistema que lleve a cabo una serie de operaciones en la base de datos para calcular el resultado deseado. En los **lenguajes no procedimentales,** el usuario describe la información deseada sin establecer un procedimiento concreto para obtener esa información.

aula(*edificio*, *número_aula*, *capacidad*)

departamento(*nombre_dept*, *edificio*, *presupuesto*)

asignatura(*asignatura_id*, *nombre_asig*, *nombre_dept*, *créditos*)

profesor(*ID*, *nombre*, *nombre_dept*, *sueldo*)

sección(*asignatura_id*, *secc_id*, *semestre*, *año*, *edificio*, *número_aula*, *franja_horaria_id*)

enseña(*ID*, *asignatura_id*, *secc_id*, *semestre*, *año*)

estudiante(*ID*, *nombre*, *nombre_dept*, *tot_créditos*)

matricula(*ID*, *asignatura_id*, *secc_id*, *semestre*, *año*, *nota*)

tutor(*e_ID*, *p_ID*)

franja_horaria(*franja_horaria_id*, *día*, *hora_inicio*, *hora_fin*)

prerreq(*asignatura_id*, *prerreq_id*)

Figura 2.9. Esquema de la base de datos de la universidad.

La mayor parte de los sistemas comerciales de bases de datos relacionales ofrecen un lenguaje de consultas que incluye elementos de los enfoques procedimental y no procedimental. Se estudiará ampliamente el lenguaje de consultas SQL en los Capítulos 3 al 5.

Existen varios lenguajes de consultas «puros»: el álgebra relacional es procedimental, mientras que el cálculo relacional de tuplas y el cálculo relacional de dominios no lo son. Estos lenguajes de consultas son rígidos y formales, y carecen del «azúcar sintáctico» de los lenguajes comerciales, pero ilustran las técnicas fundamentales para la extracción de datos de las bases de datos. En el Capítulo 6 se examina con detalle el álgebra relacional y las dos versiones de cálculo relacional, el cálculo relacional de tuplas y el cálculo relacional de dominios. El álgebra relacional consta de un conjunto de operaciones que toma una o dos relaciones como entrada y genera una nueva relación como resultado. El cálculo relacional usa la lógica de predicados para definir el resultado deseado sin proporcionar ningún procedimiento concreto para obtener dicho resultado.

2.6. Operaciones relacionales

Todos los lenguajes procedimentales de consulta relacional proporcionan un conjunto de operaciones que se pueden aplicar a una única relación o a un par de relaciones. Estas operaciones tienen la propiedad, por otra parte deseable, de que el resultado es siempre una única relación. Esta propiedad permite combinar varias de las operaciones de forma modular. Concretamente, como el resultado de una consulta relacional es una relación, se pueden aplicar operaciones a los resultados de las consultas de la misma forma que a los conjuntos de relaciones dadas.

Las operaciones relacionales concretas se expresan de forma diferente dependiendo del lenguaje, pero caen en el marco general que se describe en esta sección. En el Capítulo 3 se muestra la forma concreta de expresar estas operaciones en SQL.

La operación más frecuente es la selección de tuplas concretas de una única relación (por ejemplo, *profesor*) que satisfaga algún predicado particular (por ejemplo, *sueldo* $>$ 85.000 €). El resultado es una nueva relación que sea un subconjunto de la relación original (*profesor*). Por ejemplo, si se seleccionan las tuplas de la relación *profesor* de la Figura 2.1, que satisfacen el predicado «*sueldo* es mayor que 85.000 €», se obtiene el resultado que se muestra en la Figura 2.10.

ID	*nombre*	*nombre_dept*	*sueldo*
12121	Wu	Finanzas	90000
22222	Einstein	Física	95000
33456	Gold	Física	87000
83821	Brandt	Informática	92000

Figura 2.10. Resultado de la consultas de seleccionar las tuplas de *profesor* con un sueldo superior a 85.000 €.

Otra operación habitual es seleccionar ciertos atributos (columnas) de una relación. El resultado es una nueva relación que solo tiene los atributos seleccionados. Por ejemplo, suponga que se desea una lista de los *ID* de los profesores y los sueldos sin listar los valores de *nombre* ni *nombre_dept* de la relación *profesor* de la Figura 2.1, entonces el resultado (Figura 2.11) posee los dos atributos, *ID* y *sueldo*. Las tuplas del resultado provienen de las tuplas de la relación *profesor* pero solo con los atributos que se muestran.

La operación ***join*** **(reunión)** permite combinar dos relaciones mezclando pares de tuplas, una de cada relación en una única tupla. Existen diferentes formas de unir relaciones (se verán en el Capítulo 3). En la Figura 2.12 se muestra un ejemplo de reunión de tuplas de las tablas *profesor* y *departamento* en la que las nuevas tuplas muestran la información de los profesores y los departamentos donde trabajan. El resultado se ha formado combinando las tuplas de la relación *profesor* con la tupla de la relación *departamento* por el departamento del profesor.

En la forma de reunión que se muestra en la Figura 2.12, que se denomina ***reunión natural,*** una tupla de la relación *profesor* coincide con una tupla de la relación *departamento* si los valores de sus atributos *nombre_dept* son iguales. Todas las tuplas así coincidentes son las que aparecen como resultado de la reunión. En general, la operación de reunión natural sobre dos relaciones hace casar las tuplas cuyos valores coinciden para el o los atributos que son comunes a ambas relaciones.

ID	*sueldo*
10101	65000
12121	90000
15151	40000
22222	95000
32343	60000
33456	87000
45565	75000
58583	62000
76543	80000
76766	72000
83821	92000
98345	80000

Figura 2.11. Resultado de la consulta de seleccionar los atributos *ID* y *sueldo* de la relación *profesor*.

ID	*nombre*	*sueldo*	*nombre_dept*	*edificio*	*presupuesto*
10101	Srinivasan	65000	Informática	Taylor	100000
12121	Wu	90000	Finanzas	Painter	120000
15151	Mozart	40000	Música	Packard	80000
22222	Einstein	95000	Física	Watson	70000
32343	El Said	60000	Historia	Painter	50000
33456	Gold	87000	Física	Watson	70000
45565	Katz	75000	Informática	Taylor	100000
58583	Califieri	62000	Historia	Painter	50000
76543	Singh	80000	Finanzas	Painter	120000
76766	Crick	72000	Biología	Watson	90000
83821	Brandt	92000	Informática	Taylor	100000
98345	Kim	80000	Electrónica	Taylor	85000

Figura 2.12. Resultado de la reunión natural de las relaciones *profesor* y *departamento*.

La operación ***producto cartesiano*** combina tuplas de dos relaciones, pero al contrario que la operación reunión, su resultado contiene *todos* los pares de tuplas de las dos relaciones independientemente de si sus atributos coinciden o no.

Como las relaciones son conjuntos, se pueden realizar operaciones de conjuntos sobre ellas. La operación ***unión*** realiza la unión de conjuntos de dos tablas «de estructura similar» (digamos una tabla de estudiantes graduados y una tabla de estudiantes aún sin graduar). Por ejemplo, se puede obtener el conjunto de todos los estudiantes de un departamento. También se pueden llevar a cabo otras operaciones sobre conjuntos, como la ***intersección*** y la ***diferencia de conjuntos.***

Como se ha indicado anteriormente, se pueden realizar operaciones sobre los resultados de las consultas. Por ejemplo, si se desea encontrar el *ID* y el *sueldo* de los profesores que tienen un sueldo de más de 85.000 €, se podrían utilizar las dos primeras operaciones del ejemplo anterior. En primer lugar se seleccionan las tuplas de la relación *profesor* en las que el valor *sueldo* es mayor de 85.000 € y, después del resultado, se seleccionan los dos atributos *ID* y *sueldo*, lo que genera como resultado la relación que se muestra en la Figura 2.13, que consta del *ID* y el *sueldo*. En este ejemplo, se podrían haber realizado las operaciones en cualquier orden, pero este no suele ser el caso en todas las situaciones, como se verá más adelante.

ID	*sueldo*
12121	90000
22222	95000
33456	87000
83821	92000

Figura 2.13. Resultado de seleccionar los atributos *ID* y *sueldo* de los profesores con un sueldo superior a 85.000 €.

A veces, el resultado de una consulta contiene tuplas duplicadas. Por ejemplo, si se selecciona el atributo *nombre_dept* de la relación *profesor*, existen varios casos de duplicados, incluyendo «Informática», que aparece tres veces. Ciertos lenguajes relacionales cumplen estrictamente con la definición matemática de conjunto y eliminan los duplicados. Otros, en consideración a la relativa sobrecarga de procesamiento necesario para eliminar los duplicados de relaciones muy grandes, mantienen los duplicados. En este último caso, las relaciones no son relaciones reales en el sentido matemático del término.

Por supuesto, los datos de la base de datos cambian con el tiempo. Una relación puede actualizarse insertando nuevas tuplas, borrando tuplas existentes o modificando tuplas cambiando valores de ciertos atributos. Se pueden borrar relaciones completas y crear otras nuevas.

Se tratarán las consultas y las actualizaciones relacionales usando el lenguaje SQL en los Capítulos 3 al 5.

ÁLGEBRA RELACIONAL

El álgebra relacional define un conjunto de operaciones sobre relaciones, de manera similar a las operaciones algebraicas habituales como la suma, la resta o la multiplicación que operan sobre números. Al igual que las operaciones algebraicas operan con uno o más números como entrada y devuelven un número como salida, el álgebra relacional normalmente opera con una o dos relaciones como entrada y devuelve una relación como salida.

El álgebra relacional se trata con detalle en el Capítulo 6, pero a continuación se describen algunas de sus operaciones:

Símbolo (Nombre)	Ejemplo de uso
σ (Selección)	$\sigma_{sueldo>=85000}(profesor)$
	Devuelve las filas de la relación de entrada que satisfacen el predicado.
Π (Proyección)	$\Pi_{ID,sueldo}(profesor)$
	Genera los atributos indicados de las filas de la relación de entrada. Elimina de la salida las tuplas duplicadas.
$\bowtie$ (Reunión natural)	*profesor* $\bowtie$ *departamento*
	Genera pares de filas de las dos relaciones de entrada que tienen los mismos valores en todos los atributos con el mismo nombre.
$\times$ (Producto cartesiano)	*profesor* $\times$ *departamento*
	Genera todos los pares de filas de las dos relaciones de entrada (independientemente de que tengan o no los mismos valores en el o los atributos comunes).
$\cup$ (Unión)	$\Pi_{nombre}(profesor) \cup \Pi_{nombre}(estudiante)$
	Genera la unión de las tuplas de las dos relaciones de entrada.

2.7. Resumen

- El **modelo de datos relacional** se basa en una colección de tablas. El usuario del sistema de bases de datos puede consultar estas tablas, insertar nuevas tuplas, borrar tuplas y actualizar (modificar) tuplas. Existen varios lenguajes para expresar estas operaciones.
- El **esquema** de una relación se refiere a su diseño lógico, mientras que un **ejemplar** de una relación se refiere a su contenido en un momento dado en el tiempo. El esquema de una base de datos y una instancia de una base de datos se definen de forma parecida. El esquema de una relación incluye sus atributos y, opcionalmente, los tipos de los atributos y las restricciones sobre las relaciones, como las restricciones de clave primaria y de clave externa.
- Una **superclave** de una relación es un conjunto de uno o más atributos cuyos valores se garantiza que identifican las tuplas de una relación de forma única. Una clave candidata es una superclave mínima, es decir, un conjunto de atributos que forman una superclave, pero que ninguno de sus subconjuntos es una superclave. Una de las claves candidatas de una relación se elige como su **clave primaria.**
- Una **clave externa** es un conjunto de atributos en una relación referenciante, de manera que para las tuplas de la relación referenciada los valores de los atributos de la clave externa se garantiza que existen como valores de la clave primaria en la relación referenciada.
- Un **diagrama de esquema** es una representación gráfica del esquema de una base de datos que muestra las relaciones de la base de datos, sus atributos, sus claves primarias y sus claves externas.
- Los **lenguajes de consulta relacional** definen un conjunto de operaciones que operan sobre tablas y generan tablas como resultado. Estas operaciones se pueden combinar para obtener expresiones que reflejen las consultas deseadas.
- El **álgebra relacional** proporciona un conjunto de operaciones que toman una o más relaciones como entrada y devuelven una relación como salida. Los lenguajes de consulta habituales, como SQL, se basan en el álgebra relacional, pero añaden un cierto número de características sintácticas de utilidad.

Términos de repaso

- Tabla.
- Relación.
- Tupla.
- Atributo.
- Dominio.
- Dominio atómico.
- Valor nulo.
- Esquema de la base de datos.
- Ejemplar de la base de datos.
- Esquema de relación.
- Ejemplar de relación.
- Claves.
  - Superclave.
  - Clave candidata.
  - Clave primaria.
- Clave externa.
  - Relación referenciante.
  - Relación referenciada.
- Restricción de integridad referencial.
- Diagrama de esquema.
- Lenguaje de consulta.
  - Lenguaje procedimental.
  - Lenguaje no procedimental.
- Operaciones sobre relaciones.
  - Selección de tuplas.
  - Selección de atributos.
  - Reunión natural.
  - Producto cartesiano.
  - Operaciones de conjuntos.
- Álgebra relacional.

Ejercicios prácticos

2.1. Considere la base de datos relacional de la Figura 2.14.
Indique las claves primarias apropiadas.

2.2. Considere las restricciones de clave externa del atributo *nombre_dept* de la relación *profesor* a la relación *departamento*.
Indique un ejemplo de inserciones y borrados de estas relaciones que genere un no cumplimiento de las restricciones de clave externa.

2.3. Considere la relación *franja_horaria*. Dado que una franja horaria concreta puede darse más de una vez a la semana, explique por qué *día* y *hora_inicio* forman parte de la clave primaria de esta relación, mientras que *hora_fin* no.

2.4. En el ejemplar de *profesor* que se muestra en la Figura 2.1 no existen dos profesores con el mismo nombre. De esto, ¿se puede concluir que el nombre se puede utilizar como superclave (o clave primaria) de profesor?

2.5. Indique el resultado de realizar primero el producto cartesiano de *estudiante* y *tutor* y después realizar la operación de selección sobre el resultado con el predicado $e\_id = ID$. Usando la notación simbólica del álgebra relacional, esta consulta se escribe como $\sigma_{e\_id=ID}(estudiante \times tutor)$.

empleado (nombre_persona, calle, ciudad)
trabaja (nombre_persona, nombre_empresa, sueldo)
empresa (nombre_empresa, ciudad)

Figura 2.14. Base de datos relacional para los Ejercicios 2.1, 2.7 y 2.12.

sucursal (nombre_sucursal, ciudad_sucursal, activos)
cliente (nombre_cliente, calle_cliente, ciudad_cliente)
préstamo (número_préstamo, nombre_sucursal, cantidad)
prestatario (nombre_cliente, número_préstamo)
cuenta (número_cuenta, nombre_sucursal, saldo)
impositor (nombre_cliente, número_cuenta)

Figura 2.15. Base de datos del banco para los Ejercicios 2.8, 2.9 y 2.13.

Ejercicios

2.9. Considere la base de datos del banco de la Figura 2.15.
- **a.** Indique cuáles son las claves primarias apropiadas.
- **b.** Dada su elección de claves primarias, identifique las claves externas apropiadas.

2.10. Considere la relación *tutor* que se muestra en la Figura 2.8, con *e_id* como clave primaria de *tutor*. Suponga que un estudiante puede tener más de un tutor. Entonces, ¿seguiría sirviendo *e_id* como clave primaria de la relación *tutor*? Si su respuesta es no, ¿cuál debería ser la cave primaria de *tutor*?

2.11. Describa la diferencia de significado entre los términos *relación* y *esquema de relación*.

2.12. Considere la base de datos de la Figura 2.14. Indique una expresión del álgebra relacional para expresar las siguientes consultas:
- **a.** Encontrar los nombre de todos los empleados que trabajan para «First Bank Corporation».
- **b.** Encontrar los nombres y las ciudades de residencia de todos los empleados que trabajan para «First Bank Corporation».

2.6. Considere las siguientes expresiones, que utilizan el resultado de una operación del álgebra relacional como entrada de otra operación. Para cada expresión, describa con palabras qué operación realiza.
- **a.** $\sigma_{año \geq 2009}(matricula) \bowtie estudiante$
- **b.** $\sigma_{año \geq 2009}(matricula \bowtie estudiante)$
- **c.** $\Pi_{ID,nombre,asignatura\_id}(estudiante \bowtie matricula)$

2.7. Considere la base de datos relacional de la Figura 2.14. Indique una expresión del álgebra relacional que exprese cada una de las siguientes consultas:
- **a.** Encontrar los nombres de todos los empleados que viven en la ciudad de «Miami».
- **b.** Encontrar los nombres de todos los empleados cuyos sueldos sean superiores a 100.000 €.
- **c.** Encontrar los nombre de todos los empleados que vivan en «Miami» y cuyos sueldos sean superiores a 100.000 €.

2.8. Considere la base de datos del banco de la Figura 2.15. Indique una expresión del álgebra relacional para cada una de las siguientes consultas:
- **a.** Encontrar los nombres de todas las sucursales ubicadas en «Chicago».
- **b.** Encontrar los nombres de todos los prestatarios que tienen un préstamo en la sucursal «Downtown».
- **c.** Encontrar los nombres, la calle y la ciudad de residencia de todos los empleados que trabajan para «First Bank Corporation» y ganan más de 10.000 €.

2.13. Considere la base de datos del banco de la Figura 2.15. Escriba una expresión del álgebra relacional para cada una de las siguientes consultas:
- **a.** Encontrar todos los números de préstamos con un valor de préstamo superior a 10.000 €.
- **b.** Encontrar los nombre de todos los impositores que tienen una cuenta con un valor superior a 6.000 €.
- **c.** Encontrar los nombre de todos los impositores que tienen una cuenta con un valor superior a 6.000 € en la sucursal «Uptown».

2.14. Indique dos razones por las que se deben introducir los valores nulos en la base de datos.

2.15. Discuta los méritos relativos de los lenguajes procedimentales y los no procedimentales.

Notas bibliográficas

E. F. Codd, del Laboratorio San Jose Research de IBM, propuso el modelo relacional a finales de los años sesenta (Codd [1970]). Este trabajo le permitió a Codd obtener el prestigioso premio ACM Turing Award en 1981 (Codd [1982]).

Tras la publicación por Codd de su original artículo, se formaron varios proyectos de investigación para construir sistemas de bases de datos relacionales prácticas, incluyendo el Sistema R en el Laboratorio San Jose Research de IBM, Ingres en la Universidad de California, en Berkeley, y Query-by-Example en el Centro de investigación T. J. Watson de IBM.

Ahora existen en el mercado muchos productos de bases de datos relacionales. Entre ellos se encuentran DB2 de IBM e Informx, Oracle, Sybase y SQL Server de Microsoft.

Entre los sistemas de bases de datos relaciones de fuente abierta están MySQL y PostgreSQL. Microsoft Access es un producto de bases de datos de un solo usuario que forma parte de Microsoft Office.

Atzeni y Antonellis [1993], Maier [1983] y Abiteboul et ál. [1995] son textos dedicados exclusivamente a la teoría de los modelos de datos relacionales.