CAPÍTULO 6

El álgebra relacional y los cálculos relacionales

En este capítulo vamos a tratar los dos lenguajes formales del modelo relacional: el álgebra relacional y los cálculos relacionales. Como ya comentamos en el Capítulo 2, un modelo de datos debe incluir un conjunto de operaciones para manipular la base de datos junto con los conceptos necesarios para la definición de su estructura y restricciones. El conjunto de operaciones básicas del modelo relacional es el **álgebra relacional**, el cual permite al usuario especificar las peticiones fundamentales de recuperación. El resultado de una recuperación es una nueva relación, la cual puede estar constituida por una o más relaciones. Por consiguiente, las operaciones de álgebra producen nuevas relaciones que pueden ser manipuladas más adelante usando operaciones del mismo álgebra. Una secuencia de operaciones de álgebra relacional conforma una expresión de álgebra relacional, cuyo resultado será también una nueva relación que representa el resultado de una consulta a la base de datos (o una petición de recuperación).

El álgebra relacional es muy importante por varias razones. La primera, porque proporciona un fundamento formal para las operaciones del modelo relacional. La segunda razón, y quizá la más importante, es que se utiliza como base para la implementación y optimización de consultas en los RDBMS (Sistemas de administración de bases de datos relacionales, *Relational DataBase Management Systems*), tal y como se comentará en la Parte 4. Tercera, porque algunos de sus conceptos se han incorporado al lenguaje estándar de consultas SQL para los RDBMS.

Aunque ninguno de los RDBMS comerciales actuales proporcionan una interfaz para las consultas de álgebra relacional, las funciones y operaciones centrales de cualquier sistema relacional están basadas en estas operaciones, que explicamos con detalle en las siguientes secciones.

Mientras que el álgebra define un conjunto de operaciones del modelo relacional, los **cálculos relacionales** ofrecen una notación declarativa de alto nivel para especificar las consultas relacionales. Una expresión de cálculo relacional crea una nueva relación, la cual está especificada en términos de variables que engloban filas de las relaciones almacenadas en la base de datos (en cálculos de tupla) o columnas de las relaciones almacenadas (para los cálculos de dominio). En una expresión de cálculo, *no existe un orden de operaciones* para recuperar los resultados de la consulta: la expresión sólo especifica la información que el resultado debería contener. Ésta es la diferencia principal entre el álgebra relacional y los cálculos relacionales. Éste último es importante porque tiene una base firme en la lógica matemática y porque el SQL (Lenguaje de consulta estándar, *Standard Query Language*) para los RDBMS tiene alguno de sus fundamentos en los cálculos de tupla relacional.[1]

[1] SQL está basado en los cálculos relacionales de tupla, aunque también incorpora algunas de las operaciones del álgebra relacional y sus extensiones, como veremos en los Capítulos 8 y 9.

El álgebra relacional tiende a ser considerado como una parte integral del modelo de datos relacional. Sus operaciones pueden dividirse en dos grupos. Uno de ellos incluye el conjunto de operaciones de la teoría matemática de conjuntos, los cuales son aplicables porque cada relación está definida de modo que sea un conjunto de tuplas en el modelo relacional formal. Estas operaciones incluyen UNIÓN (UNION), INTERSECCIÓN (*INTERSECTION*), DIFERENCIA DE CONJUNTOS (*SET DIFFERENCE*) y PRODUCTO CARTESIANO (*CARTESIAN PRODUCT*). El otro grupo está constituido por las operaciones desarrolladas específicamente para las bases de datos relacionales, como la SELECCIÓN (SELECT), la PROYECCIÓN (*PROJECT*), la CONCATENACIÓN o COMBINACIÓN (JOIN) y otras. La Sección 6.1 empieza tratando las operaciones SELECCIÓN y PROYECCIÓN porque son **operaciones unarias** que operan en relaciones individuales. La Sección 6.2 se encarga del conjunto de operaciones, mientras que la 6.3 se centra en la CONCATENACIÓN y otras **operaciones binarias** complejas que operan sobre dos tablas. En los ejemplos utilizaremos la base de datos EMPRESA de la Figura 5.6.

Algunas de las peticiones de base de datos más comunes no pueden llevarse a cabo con las operaciones del álgebra relacional originales, por lo que se desarrollaron otras nuevas para lograrlo. Entre estas operaciones se incluyen las **funciones agregadas**, que son operaciones que pueden *resumir* datos a partir de tablas, así como operaciones CONCATENACIÓN y UNIÓN adicionales. Estas operaciones fueron añadidas al álgebra relacional original debido a su importancia para muchas aplicaciones de bases de datos, y se describen en la Sección 6.4. En la Sección 6.5 ofreceremos ejemplos de consultas que usan operaciones relacionales, y algunas de ellas se utilizan posteriormente en otros capítulos para ilustrar varios lenguajes.

En las Secciones 6.6 y 6.7 se describe el otro tipo de lenguaje formal para las bases de datos relacionales, los **cálculos relacionales**. Existen dos variantes del mismo. El cálculo relacional de tupla se explica en la Sección 6.6, mientras que el de dominio se detalla en la Sección 6.7. Algunas de las construcciones SQL tratadas en el Capítulo 8 están basadas en la primera de estas variantes. El cálculo relacional es un lenguaje formal basado en una rama de la lógica matemática llamada cálculos de predicado[2]. En los cálculos relacionales de tupla, las variables alcanzan a las tuplas, mientras que en los de dominio "atacan" a los valores de los atributos. En el Apéndice D podrá encontrar una descripción del QBE (Consulta mediante ejemplo, *Query-By-Example*), un lenguaje relacional gráfico y agradable para el usuario basado en los cálculos relacionales de dominio. La Sección 6.8 resume todo este capítulo.

Los lectores interesados en una introducción menos detallada de los lenguajes relacionales formales pueden saltarse las Secciones 6.4, 6.6 y 6.7.

6.1 Operaciones relacionales unarias: SELECCIÓN (*SELECT*) y PROYECCIÓN (*PROJECT*)

6.1.1 La operación SELECCIÓN

SELECCIÓN se emplea para seleccionar un *subconjunto* de las tuplas de una relación que satisfacen una **condición de selección**. Se puede considerar esta operación como un *filtro* que mantiene sólo las tuplas que satisfacen una determinada condición. SELECCIÓN puede visualizarse también como una *partición horizontal* de la relación en dos conjuntos de tuplas: las que satisfacen la condición son seleccionadas y las que no, descartadas. Por ejemplo, para seleccionar las tuplas de EMPLEADO cuyo departamento sea 4, o cuyo salario sea mayor de 30.000 euros, podemos especificar individualmente cada una de estas condiciones con una operación SELECCIÓN como ésta:

$\sigma_{Dno=4}$(EMPLEADO)

$\sigma_{Sueldo>30000}$(EMPLEADO)

[2] En este capítulo se asume que no tiene ningún conocimiento sobre los cálculos de predicado de primer orden (los que tratan con variables y valores cuantificados).

En general, SELECCIÓN está designada como:

$$\sigma_{\text{<condición de selección>}}(R)$$

donde el símbolo σ (sigma) se utiliza para especificar el operador de SELECCIÓN, mientras que la condición de selección es una expresión lógica (o booleana) especificada sobre los atributos de la relación *R*. Observe que *R* es, generalmente, una *expresión de álgebra relacional* cuyo resultado es una relación: la más sencilla de estas expresiones es sólo el nombre de una relación de base de datos. El resultado de SELECCIÓN tiene los *mismos atributos* que *R*.

La expresión lógica especificada en <condición de selección> está constituida por un número de **cláusulas** de la forma:

<nombre de atributo> <operador de comparación> <valor constante>,

o bien:

<nombre de atributo> <operador de comparación> <nombre de atributo>

donde <nombre de atributo> es el nombre de un atributo de *R*, <operador de comparación> suele ser uno de los operadores {=, <, ≤, >, ≥, ≠ } y <valor constante> es un valor del dominio del atributo. Las cláusulas pueden estar conectadas arbitrariamente por operadores lógicos *and*, *or* y *not* para formar una condición de selección general. Por ejemplo, para seleccionar las tuplas de todos los empleados que trabajan en el departamento 4 y ganan sobre 25.000 euros al año, o los que trabajan en el 5 y ganan alrededor de 30.000, podemos especificar la siguiente operación de SELECCIÓN:

$$\sigma_{(\text{Dno=4 AND Sueldo>25000}) \text{ OR } (\text{Dno=5 AND Sueldo>30000})}(\text{EMPLEADO})$$

El resultado aparece en la Figura 6.1(a).

Figura 6.1. Resultado de las operaciones SELECCIÓN y PROYECCIÓN. (a) s (Dno=4 AND Sueldo>25000) OR (Dno=5 AND Sueldo > 30000) (EMPLEADO). (b) pApellido1, Nombre, Sueldo(EMPLEADO). (c) pSexo, Sueldo(EMPLEADO).

(a)

Nombre	Apellido1	Apellido2	Dni	FechaNac	Dirección	Sexo	Sueldo	SuperDni	Dno
Alberto	Campos	Sastre	333445555	08-12-1955	Avda. Ríos, 9	H	40000	888665555	5
Juana	Sainz	Oreja	987654321	20-06-1941	Cerquillas, 67	M	43000	888665555	4
Fernando	Ojeda	Ordóñez	666884444	15-09-1962	Portillo, s/n	H	38000	333445555	5

(b)

Apellido1	Nombre	Sueldo
Pérez	José	30000
Campos	Alberto	40000
Jiménez	Alicia	25000
Sainz	Juana	43000
Ojeda	Fernando	38000
Oliva	Aurora	25000
Pajares	Luis	25000
Ochoa	Eduardo	55000

(c)

Sexo	Sueldo
H	30000
H	40000
M	25000
M	43000
H	38000
H	25000
H	55000

Observe que los operadores de comparación del conjunto $\{=, <, \leq, >, \geq, \neq\}$ se aplican a los atributos cuyos dominios son *valores ordenados*, como los de tipo numérico o de fecha. Los de cadenas de caracteres también se consideran del mismo tipo debido a la secuencia en los códigos. Si el dominio de un atributo es un conjunto de *valores desordenados*, sólo pueden usarse los operadores $\{=, \neq\}$. Color = {'rojo', 'azul', 'verde', 'blanco', 'amarillo', . . .} es un ejemplo de dominio desordenado. Algunos de ellos permiten operadores de comparación adicionales; por ejemplo, un dominio de cadenas de caracteres permite el operador SUBCADENA_DE.

En general, el resultado de una operación SELECCIÓN puede determinarse como sigue. La <condición de selección> se aplica independientemente a cada tupla t de R. Esto se realiza sustituyendo cada ocurrencia de un atributo A_i en la condición de selección por su valor en la tupla $t[A_i]$. Si la condición se evalúa como VERDADERO (*TRUE*), la tupla t se **selecciona**.

Todas las tuplas seleccionadas aparecen en el resultado de SELECCIÓN. Las condiciones lógicas AND, OR y NOT tienen la siguiente interpretación:

- (condición1 **AND** condición2) es VERDADERO si las dos condiciones (condición1 y condición2) lo son; en cualquier otro caso, es FALSO (FALSE).
- (condición1 **OR** condición2) es VERDADERO si (condición1) o (condición2) o ambas son verdaderas; en cualquier otro caso, es FALSO.
- (**NOT** condición) es VERDADERO si condición es FALSO; en cualquier otro caso es FALSO.

SELECCIÓN es **unaria**, es decir, se aplica a una sola relación. Además, esta operación se aplica a *cada tupla individualmente*; por consiguiente, las condiciones de selección no pueden implicar a más de una tupla. El **grado** de la relación resultante de una operación SELECCIÓN (su número de atributos) es el mismo que el de R. El número de tuplas en la relación resultante es siempre *menor que o igual que* el número de tuplas en R. Esto es, $\sigma_c(R)| \leq |R|$ para cualquier condición C. La fracción de tuplas seleccionadas por una condición de relación está referida como la **selectividad** de la condición.

Observe que la operación SELECCIÓN es **conmutativa**, es decir:

$$\sigma_{\text{<condición1>}}(\sigma_{\text{<condición2>}}(R)) = \sigma_{\text{<condición2>}}(\sigma_{\text{<condición1>}}(R))$$

Por tanto, puede aplicarse una secuencia de SELECCIONES en cualquier orden. Además, siempre podemos combinar una **cascada** de operaciones SELECCIÓN en una sola a través de un (AND):

$$\sigma_{\text{<condición1>}}(\sigma_{\text{<condición2>}}(\ldots(\sigma_{\text{<condición>}}(R))\ldots)) = \sigma_{\text{<condición1>}}\ \textbf{AND}\ _{\text{<condición2>}}\ \textbf{AND}\ \ldots\ \textbf{AND}\ _{\text{<condición>}}(R)$$

6.1.2 La operación PROYECCIÓN

Si pensamos en una relación como en una tabla, la operación SELECCIÓN elige algunas de las *filas* de la tabla a la vez que descarta otras. Por otro lado, PROYECCIÓN selecciona ciertas *columnas* de la tabla y descarta otras. Si sólo estamos interesados en algunos atributos de una relación, usamos la operación PROYECCIÓN para *planear* la relación sólo sobre esos atributos. Por consiguiente, el resultado de esta operación puede visualizarse como una *partición vertical* de la relación en otras dos: una contiene las columnas (atributos) necesarias y otra las descartadas. Por ejemplo, para listar el nombre, el primer apellido y el sueldo de cada empleado, podemos usar PROYECCIÓN de la siguiente forma:

$$\pi_{\text{Apellido1, Nombre, Sueldo}}(\text{EMPLEADO})$$

La relación resultante se muestra en la Figura 6.1(b). La forma general de la operación PROYECCIÓN es:

$$\pi_{\text{<lista de atributos>}}(R)$$

donde π (pi) es el símbolo usado para representar la operación PROYECCIÓN, mientras que <lista de atributos> contiene la lista de campos de la relación R que queremos. De nuevo, observe que R es, en general, una *expresión de álgebra relacional* cuyo resultado es una relación, cuyo caso más simple es obtener sólo el nombre de una relación de base de datos. El resultado de la operación PROYECCIÓN sólo tiene los atributos especificados en <lista de atributos> *en el mismo orden a como aparecen en la lista*. Por tanto, su **grado** es igual al número de atributos contenidos en <lista de atributos>.

Si la lista de atributos sólo incluye atributos no clave de R, es posible que se dupliquen tuplas. La operación PROYECCIÓN *elimina cualquier tupla duplicada*, por lo que el resultado de la misma es un conjunto de tuplas y, por consiguiente, una relación válida. Esto se conoce como **eliminación de duplicados.** Por ejemplo, considere la siguiente operación PROYECCIÓN:

$$\pi_{\text{Sexo, Sueldo}}(\text{EMPLEADO})$$

El resultado aparece en la Figura 6.1(c). Observe que la tupla <'M', 25000> sólo aparece una vez en dicha figura, aun cuando su combinación de valores aparezca dos veces en EMPLEADO. La eliminación de duplicados lleva implícito un proceso de ordenación para detectar esos registros repetidos y, por tanto, añadir más capacidad de procesamiento. Si no se llevara a cabo este proceso, el resultado sería un **multiconjunto** o **bolsa** de tuplas en lugar de un conjunto. Esto no estaba permitido en el modelo relacional formal, aunque sí lo estaba en la práctica. En el Capítulo 8 mostraremos que el usuario puede optar por eliminar o no los registros duplicados.

El número de tuplas de una relación resultante de una operación PROYECCIÓN es siempre menor o igual que el de las contenidas en R. Si la lista de proyección es una superclave de R (esto es, incluye alguna clave de R) la relación resultante tiene el mismo número de tuplas que R. Además,

$$\pi_{\text{<lista1>}}(\pi_{\text{<lista2>}}(R)) = \pi_{\text{<lista1>}}(R)$$

con tal de que <lista2> contenga los atributos de <lista1>; de otro modo, la parte de la izquierda es una expresión incorrecta. También resulta digno de mención el hecho de que la conmutatividad *no* se almacena en una PROYECCIÓN.

6.1.3 Secuencias de operaciones y la operación RENOMBRAR (*RENAME*)

Las relaciones mostradas en la Figura 6.1 no tienen ningún nombre. En general, podemos querer aplicar varias operaciones de álgebra relacional una tras otra. De cualquier forma, podemos escribir las operaciones como una única **expresión de álgebra relacional** anidando dichas operaciones, o aplicar una sola expresión una única vez y crear relaciones intermedias. En el último caso, puede que queramos asignar nombres a dichas relaciones intermedias. Por ejemplo, para recuperar el nombre, el primer apellido y el sueldo de todos los empleados que trabajan en el departamento 5, debemos aplicar una SELECCIÓN y una PROYECCIÓN. Podemos escribir una expresión de álgebra relacional sencilla de la siguiente forma:

$$\pi_{\text{Nombre, Apellido1, Sueldo}}(\sigma_{\text{Dno=5}}(\text{EMPLEADO}))$$

La Figura 6.2(a) muestra el resultado de esta operación. Alternativamente, podemos mostrar la secuencia de operaciones, dando un nombre a cada relación intermedia:

$$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{Dno=5}}(\text{EMPLEADO})$$
$$\text{RESULTADO} \leftarrow \pi_{\text{Nombre, Apellido1, Sueldo}}(\text{DEP5\_EMPS})$$

Con frecuencia, resulta más simple partir una secuencia compleja de operaciones en relaciones intermedias que escribir una única expresión de álgebra relacional. Podemos usar también esta técnica para **renombrar**

Figura 6.2. Resultado de una secuencia de operaciones. (a) $\pi_{\text{Nombre, Apellido1, Sueldo}}(\sigma_{\text{Dno=5}}(\text{EMPLEADO}))$. (b) Usando relaciones intermedias y renombrando los atributos.

(a)

Nombre	Apellido1	Sueldo
José	Pérez	30000
Alberto	Campos	40000
Fernando	Ojeda	38000
Aurora	Oliva	25000

(b)

TEMP

Nombre	Apellido1	Apellido2	Dni	FechaNac	Dirección	Sexo	Sueldo	SuperDni	Dno
José	Pérez	Pérez	123456789	01-09-1965	Eloy I, 98	H	30000	333445555	5
Alberto	Campos	Sastre	333445555	08-12-1955	Avda. Ríos, 9	H	40000	888665555	5
Fernando	Ojeda	Ordóñez	666884444	15-09-1962	Portillo, s/n	H	38000	333445555	5
Aurora	Oliva	Avezuela	453453453	31-07-1972	Antón, 6	M	25000	333445555	5

R

NuevoNombre	NuevoApellido	NuevoSueldo
José	Pérez	30000
Alberto	Campos	40000
Fernando	Ojeda	38000
Aurora	Oliva	25000

los atributos en las relaciones intermedias y resultantes, lo que puede resultar útil cuando se emplea junto con operaciones más complejas como UNIÓN y CONCATENACIÓN. Para renombrar los atributos de una relación, simplemente enumeramos los nuevos nombres de atributos dentro de los paréntesis, como puede verse en el siguiente ejemplo:

$$\text{TEMP} \leftarrow \sigma_{\text{Dno=5}}(\text{EMPLEADO})$$

$$R(\text{NuevoNombre, NuevosApellido, NuevoSueldo}) \leftarrow \pi_{\text{Nombre, Apellido1, Sueldo}}(\text{TEMP})$$

Ambas operaciones se ilustran en la Figura 6.2(b).

Si no se realiza un renombrado, los nombres de atributo de la relación resultante de una SELECCIÓN son los mismos que los de la relación original y aparecen en el mismo orden. Para el caso de una operación PROYECCIÓN, la relación resultante tiene los mismos nombres de atributo que los indicados en la lista de proyección y están en el mismo orden en que aparecen en dicha lista.

Podemos definir una operación **RENOMBRAR** como un operador unario. Una operación RENOMBRAR aplicada a una relación R de grado n aparece denotada de cualquiera de estas tres formas:

$$\rho_{S(B1, B2, ..., Bn)}(R) \quad \text{o} \quad \rho_S(R) \text{ o } \rho_{(B1, B2, ..., Bn)}(R)$$

donde el símbolo ρ (rho) se utiliza para especificar el operador RENOMBRAR, S es el nombre de la nueva relación y $B_1, B_2, \ldots, B_n$ son los de los nuevos atributos. La primera expresión renombra tanto la relación

como sus atributos, la segunda sólo lo hace con la relación y la tercera sólo con los atributos. Si los atributos de R son $(A_1, A_2, \ldots, A_n)$ por este orden, entonces cada A_i es renombrado como B_i.

6.2 Operaciones de álgebra relacional de la teoría de conjuntos

6.2.1 Las operaciones UNIÓN (*UNION*), INTERSECCIÓN (*INTERSECTION*) y MENOS (*MINUS*)

El siguiente grupo de operaciones de álgebra relacional son las correspondientes a la operativa matemática sobre conjuntos. Por ejemplo, para recuperar los Documentos Nacionales de Identidad de todos los empleados que, o bien trabajan en el departamento 5 o supervisan a éstos, podemos usar la operación UNIÓN del siguiente modo:[3]

DEP5_EMPS $\leftarrow \sigma_{Dno=5}$(EMPLEADO)
RESULTADO1$\leftarrow \pi_{Dni}$(DEP5_EMPS)
RESULTADO2(Dni). $\leftarrow \pi_{SuperDni}$(DEP5_EMPS)
RESULTADO $\leftarrow$ RESULTADO1 $\cup$ RESULTADO2

La relación RESULTADO1 tiene el Dni de todos los empleados del departamento 5, mientras que RESULTADO2 contiene el de aquellos empleados que supervisan directamente a los del primer grupo. La UNIÓN produce las tuplas que están en RESULTADO1 o RESULTADO2, o en ambas (consulte la Figura 6.3). De este modo, el Dni '333445555' sólo aparece una vez en el resultado.

Para combinar los elementos de dos conjuntos se utilizan varias operaciones de la teoría de conjuntos, como la **UNIÓN**, la **INTERSECCIÓN** y la **DIFERENCIA DE CONJUNTOS** (llamada también a veces **MENOS**, o ***MINUS***). Todas ellas son operaciones **binarias**, es decir, se aplican a dos conjuntos (de tuplas).

Cuando se refieren a las bases de datos relacionales, las relaciones sobre las que se aplican estas tres operaciones deben tener el mismo **tipo de tuplas**; esta condición recibe el nombre de *compatibilidad de unión*. Dos relaciones $R(A_1, A_2, \ldots, A_n)$ y $S(B_1, B_2, \ldots, B_n)$ se dice que son de **unión compatible** si tienen el mismo grado n y si el $\text{dom}(A_i) = \text{dom}(B_i)$ para $1 \leq i \leq n$. Esto significa que ambas relaciones tienen el mismo número de atributos y que cada par correspondiente cuenta con el mismo dominio.

Figura 6.3. Resultado de la operación de UNIÓN RESULTADO $\leftarrow$ RESULTADO1 $\cup$ RESULTADO2.

RESULTADO1

Dni
123456789
333445555
666884444
453453453

RESULTADO2

Dni
333445555
888665555

RESULTADO

Dni
123456789
333445555
666884444
453453453
888665555

[3] Denotado como una expresión de álgebra relacional sencilla, esto se convierte en: Resultado $\leftarrow \pi_{Dni}$ ($\sigma_{Dno=5}$ (EMPLEADO)) $\cup$. $\pi_{SuperDni}$ ($\sigma_{Dno=5}$ (EMPLEADO)).

Podemos definir las tres operaciones UNIÓN, INTERSECCIÓN y DIFERENCIA DE CONJUNTO en dos relaciones de unión compatible R y S del siguiente modo:

- UNIÓN. El resultado de esta operación, especificada como $R \cup S$, es una relación que incluye todas las tuplas que están tanto en R como en S o en ambas, R y S. Las tuplas duplicadas se eliminan.
- INTERSECCIÓN. El resultado de esta operación, especificada como $R \cap S$, es una relación que incluye todas las tuplas que están en R y en S.
- DIFERENCIA DE CONJUNTO (o MENOS). El resultado de esta operación, especificada como $R - S$, es una relación que incluye todas las tuplas que están en R pero no en S.

Adoptaremos por convenio que los nombres de atributo de la relación resultante son los mismos que los de R. Siempre es posible cambiar el nombre de estos valores a través del operador renombrar.

La Figura 6.4 ilustra las tres operaciones. Las relaciones ESTUDIANTE y PROFESOR de la Figura 6.4(a) son una unión compatible y sus tuplas representan los nombres de los estudiantes y los profesores respectivamente. El resultado de la UNIÓN de la Figura 6.4(b) muestra los nombres de todos los estudiantes y profesores. Indicar que las tuplas duplicadas aparecen una sola vez en el resultado. La INTERSECCIÓN de la Figura 6.4(c) incluye sólo aquéllos que son estudiantes y profesores a la vez.

Observe que tanto la UNIÓN como la INTERSECCIÓN son *operaciones conmutativas*, esto es,

$$R \cup S = S \cup R \text{ y } R \cap S = S \cap R$$

La UNIÓN y la INTERSECCIÓN pueden tratarse como operaciones n-ary aplicables a cualquier número de relaciones porque son *estructuras asociativas*, es decir,

$$R \cup (S \cup T) = (R \cup S) \cup T \text{ y } (R \cap S) \cap T = R \cap (S \cap T)$$

La operación MENOS *no es conmutativa;* por tanto, en general,

$$R - S \neq S - R$$

La Figura 6.4(d) muestra los nombres de los estudiantes que no son profesores, mientras que la 6.4(e) contiene los profesores que no son estudiantes.

Observe que la INTERSECCIÓN puede expresarse en términos de unión y diferencia de conjuntos del siguiente modo:

$$R \cap S = R \cup S - (R - S) - (S - R)$$

6.2.2 El producto cartesiano (producto cruzado)

Ahora vamos a tratar la operación **PRODUCTO CARTESIANO (*CARTESIAN PRODUCT*)**, conocida también como **PRODUCTO CRUZADO (*CROSS PRODUCT*)** o **CONCATENACIÓN CRUZADA (*CROSS JOIN*)**, que se identifica por $\times$. Se trata también de una operación de conjuntos binarios, aunque no es necesario que las relaciones en las que se aplica sean una unión compatible. En su forma binaria produce un nuevo elemento combinando cada miembro (tupla) de una relación (conjunto) con los de la otra. En general, el resultado de $R(A_1, A_2, \ldots, A_n) \times S(B_1, B_2, \ldots, B_m)$ es una relación Q con un grado de n + m atributos $Q(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$, en este orden. La relación resultante Q tiene una tupla por cada combinación de éstas (una para R y otra para S). Por tanto, si R tiene n_R tuplas (indicado como $|R| = n_R$), y S cuenta con n_S tuplas, $R \times S$ tendrá $n_R * n_S$ tuplas.

La operación PRODUCTO CARTESIANO n-ary es una extensión del concepto indicado más arriba que produce nuevas tuplas concatenando todas las posibles combinaciones de tuplas desde n relaciones subyacentes. La operación aplicada es, por sí misma, absurda. Es útil cuando va seguida por una selección que correlacione los valores de los atributos procedentes de las relaciones componentes. Por ejemplo, suponga que

Figura 6.4. Las operaciones de conjunto UNIÓN, INTERSECCIÓN y MENOS. (a) Dos relaciones de unión compatible. (b) ESTUDIANTE ∪ PROFESOR. (c) ESTUDIANTE ∩ PROFESOR. (d) ESTUDIANTE − PROFESOR. (e) PROFESOR − ESTUDIANTE.

(a) **ESTUDIANTE**

Nombre	Apellido
Susana	Gómez
Luis	Campos
Juan	Garrido
Bárbara	Durán
Amanda	González
Joaquín	Martín
Ernesto	Flores

PROFESOR

Nom	Apell
Antonio	Fernández
Ricardo	Adriano
Susana	Gómez
Francisco	Peláez
Luis	Campos

(b)

Nombre	Apellido
Susana	Gómez
Luis	Campos
Juan	Garrido
Bárbara	Durán
Amanda	González
Joaquín	Martín
Ernesto	Flores
Antonio	Fernández
Ricardo	Adriano
Francisco	Peláez

(c)

Nombre	Apellido
Susana	Gómez
Luis	Campos

(d)

Nombre	Apellido
Juan	Garrido
Bárbara	Durán
Amanda	González
Joaquín	Martín
Ernesto	Flores

(e)

Nom	Apell
Antonio	Fernández
Ricardo	Adriano
Francisco	Peláez

queremos recuperar una lista de nombres de cada subordinado de una empleada femenina. Podemos realizar esta operación como sigue:

$$\text{EMPLEADAS\_FEMENINAS} \leftarrow \sigma_{\text{Sexo}='M'}(\text{EMPLEADO})$$
$$\text{NOMBRES\_EMPLEADOS} \leftarrow \pi_{\text{Nombre, Apellido1, Dni}}(\text{EMPLEADAS\_FEMENINAS})$$
$$\text{EMPLEADOS\_SUBORDINADOS} \leftarrow \text{NOMBRES\_EMPLEADOS} = \text{SUBORDINADO}$$
$$\text{SUBORDINADOS\_ACTUALES} \leftarrow \sigma_{\text{Dni=DniEmpleado}}(\text{EMPLEADOS\_SUBORDINADOS})$$
$$\text{RESULTADO} \leftarrow \pi_{\text{Nombre, Apellido1, NombreSubordinado}}(\text{SUBORDINADOS\_ACTUALES})$$

Las relaciones resultantes de esta secuencia de operaciones se muestran en la Figura 6.5. EMPLEADOS_SUBORDINADOS es el resultado de aplicar el PRODUCTO CARTESIANO a los NOMBRES_EMPLEADOS de la Figura 6.5 con los SUBORDINADOs de la Figura 5.6. En EMPLEADOS_SUBORDINADOS cada tupla de NOMBRES_EMPLEADOS se combina con las de SUBORDINADO, obteniéndose un resultado que no tiene sentido. Queremos combinar una tupla de empleada femenina sólo con las de sus subordinados particulares; digamos, las tuplas de SUBORDINADO cuyos valores de DniEmpleado coincidan con el Dni de EMPLEADO. La relación SUBORDINADOS_ACTUALES consigue esto. EMPLEADOS_SUBORDINADOS es un buen ejemplo de cómo puede aplicarse correctamente el álgebra relacional para producir resultados que no tengan

Figura 6.5. El PRODUCTO CARTESIANO (PRODUCTO CRUZADO).

EMPLEADAS_FEMENINAS

Nombre	Apellido1	Apellido2	Dni	FechaNac	Dirección	Sexo	Sueldo	SuperDni	Dno
Alicia	Jiménez	Celaya	999887777	12-05-1968	Gran Vía, 38	M	25000	987654321	4
Juana	Sainz	Oreja	987654321	20-06-1941	Cerquillas, 67	M	43000	888665555	4
Aurora	Oliva	Avezuela	453453453	31-07-1972	Antón, 6	M	25000	333445555	5

NOMBRES_EMPLEADOS

Nombre	Apellido1	Dni
Alicia	Jiménez	999887777
Juana	Sainz	987654321
Aurora	Oliva	453453453

EMPLEADOS_SUBORDINADOS

Nombre	Apellido1	Dni	DniEmpleado	NombreSubordinado	Sexo	FechaNac	...
Alicia	Jiménez	999887777	333445555	Ana	M	05-04-1986	...
Alicia	Jiménez	999887777	333445555	Teodoro	H	25-10-1983	...
Alicia	Jiménez	999887777	333445555	Ruth	M	03-05-1958	...
Alicia	Jiménez	999887777	987654321	Augusto	H	28-02-1942	...
Alicia	Jiménez	999887777	123456789	Miguel	H	01-04-1988	...
Alicia	Jiménez	999887777	123456789	Ana	M	30-12-1988	...
Alicia	Jiménez	999887777	123456789	Elisa	M	05-05-1967	...
Juana	Sainz	987654321	333445555	Ana	M	05-04-1986	...
Juana	Sainz	987654321	333445555	Teodoro	H	25-10-1983	...
Juana	Sainz	987654321	333445555	Ruth	M	03-05-1958	...
Juana	Sainz	987654321	987654321	Augusto	H	28-02-1942	...
Juana	Sainz	987654321	123456789	Miguel	H	04-01-1988	...
Juana	Sainz	987654321	123456789	Ana	M	30-12-1988	...
Juana	Sainz	987654321	123456789	Elisa	M	05-05-1967	...
Aurora	Oliva	453453453	333445555	Ana	M	05-04-1986	...
Aurora	Oliva	453453453	333445555	Teodoro	H	25-10-1983	...
Aurora	Oliva	453453453	333445555	Ruth	M	03-05-1958	...
Aurora	Oliva	453453453	987654321	Augusto	H	28-02-1942	...
Aurora	Oliva	453453453	123456789	Miguel	H	04-01-1988	...
Aurora	Oliva	453453453	123456789	Ana	M	30-12-1988	...
Aurora	Oliva	453453453	123456789	Elisa	M	05-05-1967	...

Figura 6.5. (*Continuación*).

SUBORDINADOS_ACTUALES

Nombre	Apellido1	Dni	DniEmpleado	NombreSubordinado	Sexo	FechaNac	...
Juana	Sainz	987654321	987654321	Augusto	H	28-02-1942	...

RESULTADO

Nombre	Apellido1	NombreSubordinado
Juana	Sainz	Augusto

ningún sentido. Por consiguiente, es responsabilidad del usuario aplicar a las relaciones sólo operaciones que sean coherentes.

El PRODUCTO CARTESIANO crea tuplas con los atributos combinados de ambas relaciones. Sólo podemos hacer una SELECCIÓN de tuplas de las dos relaciones especificando una condición de selección apropiada (tal y como se comentó en el ejemplo precedente). Ya que esta secuencia de PRODUCTO CARTESIANO seguido de una SELECCIÓN se emplea con mucha frecuencia para identificar y elegir tuplas de dos relaciones, existe una operación especial llamada CONCATENACIÓN que permite especificar esta secuencia como una operación única. Vamos a estudiar esta operación a continuación.

6.3 Operaciones relacionales binarias: CONCATENACIÓN (*JOIN*) y DIVISIÓN (*DIVISION*)

6.3.1 La operación CONCATENACIÓN

CONCATENACIÓN, especificada mediante $\bowtie$, se emplea para combinar *tuplas relacionadas* de dos relaciones en una sola. Esta operación es muy importante para cualquier base de datos relacional que cuente con más de una relación, ya que nos permite procesar relaciones entre relaciones. Para ilustrar CONCATENACIÓN, supongamos que queremos recuperar el nombre del director de cada departamento. Para ello, necesitamos combinar las tuplas de departamento y empleado cuyos valores de DniDirector y Dni, respectivamente, sean iguales. Esto se consigue mediante la operación CONCATENACIÓN y extrapolando después el resultado sobre los atributos necesarios de la siguiente forma:

$$\text{DIRECTOR\_DPTO} \leftarrow \text{DEPARTAMENTO} \bowtie_{\text{DniDirector=Dni}} \text{EMPLEADO}$$
$$\text{RESULTADO} \leftarrow \pi_{\text{NombreDpto, Apellido1, Nombre}}(\text{DIRECTOR\_DPTO})$$

La primera operación se ilustra en la Figura 6.6. Observe que DniDirector es una *foreign key*, y que las restricciones de integridad referencial juegan un papel a la hora de hacer la correspondencia de tuplas en la relación EMPLEADO.

La CONCATENACIÓN puede ser enunciada en términos de un PRODUCTO CARTESIANO seguido de una SELECCIÓN. Sin embargo, CONCATENACIÓN es muy importante porque se utiliza con mucha frecuencia cuando se definen consultas a la base de datos. Considere el ejemplo que mostramos anteriormente para explicar el PRODUCTO CARTESIANO, el cual incluía la siguiente secuencia de operaciones:

$$\text{EMPLEADOS\_SUBORDINADOS} \leftarrow \text{NOMBRES\_EMPLEADOS} \times \text{SUBORDINADO}$$
$$\text{SUBORDINADOS\_ACTUALES} \leftarrow \sigma_{\text{Dni=DniEmpleado}}(\text{EMPLEADOS\_SUBORDINADOS})$$

Figura 6.6. Resultado de la operación CONCATENACIÓN DIRECTOR_DPTO ← DEPARTAMENTO ⋈ DniDirector=DniEMPLEADO.

DIRECTOR_DPTO

NombreDpto	NúmeroDpto	DniDirector	...	Nombre	Apellido1	Apellido2	Dni	...
Investigación	5	333445555	...	Alberto	Campos	Sastre	333445555	...
Administración	4	987654321	...	Juana	Sainz	Oreja	987654321	...
Sede Central	1	888665555	...	Eduardo	Ochoa	Paredes	888665555	...

Esas dos operaciones pueden sustituirse por una única operación CONCATENACIÓN de la siguiente forma:

SUBORDINADOS_ACTUALES ← NOMBRES_EMPLEADOS $\bowtie_{\text{Dni=DniEmpleado}}$ SUBORDINADO

La forma general de una CONCATENACIÓN en dos relaciones[4] $R(A_1, A_2, \ldots, A_n)$ y $S(B_1, B_2, \ldots, B_m)$ es:

$$R \bowtie_{\text{<condición de conexión>}} S$$

El resultado de la CONCATENACIÓN es una relación Q de $n + m$ atributos $Q(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$ por este orden; Q tiene una tupla por cada combinación de éstas (una para R y otra para S) *siempre que dicha combinación satisfaga la condición de conexión*. Ésta es la principal diferencia existente entre el PRODUCTO CARTESIANO y la CONCATENACIÓN. En la CONCATENACIÓN sólo aparecen en el resultado las combinaciones de tuplas que *satisfacen la condición de conexión*, mientras que en el PRODUCTO CARTESIANO se incluyen *todas* las combinaciones de tuplas. La condición de conexión está especificada sobre los atributos de las dos relaciones R y S y es evaluada para cada combinación de tuplas, incluyéndose en la relación Q resultante en forma de una *única tupla combinada* sólo aquéllas cuya condición de conexión se evalúe como VERDADERO. Una condición general de conexión tiene la forma:

<condición> **AND** <condición> **AND** . . . **AND** <condición>

donde cada condición es de la forma $A_i\ \theta\ B_j$, A_i es un atributo de R, B_j lo es de S, A_i y B_j tienen el mismo dominio y θ (*theta*) es uno de los operadores de comparación $\{=, <, \leq, >, \geq, \neq\}$. Una CONCATENACIÓN con una condición de conexión de este tipo recibe el nombre de **ASOCIACIÓN (*THETA JOIN*)**. Las tuplas cuyos atributos de conexión son NULL, o aquéllas cuya condición de conexión es FALSA, *no* aparecen en el resultado. En este caso, CONCATENACIÓN *no* preserva necesariamente toda la información de las relaciones participantes.

6.3.2 Variaciones de CONCATENACIÓN: EQUIJOIN y CONCATENACIÓN NATURAL (*NATURAL JOIN*)

El uso más habitual de CONCATENACIÓN supone el uso de condiciones de conexión sólo con comparaciones de igualdad, en cuyo caso recibe el nombre de **EQUIJOIN**. Observe que en el resultado de una EQUIJOIN siempre tenemos uno o más pares de atributos que cuentan con *valores idénticos* en cada tupla. Por ejemplo, en la Figura 6.6, DniDirector y Dni tienen valores idénticos en cada tupla de DIRECTOR_DPTO debido a la condición de conexión especificada ente estos atributos. Ya que uno de estos valores idénticos es innecesario, se creó una nueva operación llamada **CONCATENACIÓN NATURAL** (identificada por *) para

[4] De nuevo, observe que tanto R como S pueden ser cualquier relación resultante de expresiones generales de *álgebra relacional*.

deshacerse del segundo atributo superfluo en una condición EQUIJOIN.[5] La definición estándar de esta operación precisa que los dos atributos de conexión tengan el mismo nombre en ambas relaciones. Si éste no es el caso, se aplica en primer lugar una operación de renombrado.

En el siguiente ejemplo, primero renombramos el atributo NúmeroDpto de DEPARTAMENTO como NumDptoProyecto, con lo que éste y el de PROYECTO tendrán el mismo nombre, y después se aplica CONCATENACIÓN NATURAL:

$$\text{PROYECTO\_DPTO} \leftarrow .\text{PROYECTO } \rho_{(\text{NombreDpto, NumDptoProyecto, DniDirector,FechaIngresoDirector})}(\text{DEPARTAMENTO})$$

Se puede realizar la misma consulta en dos pasos creando una tabla intermedia DEPT de la siguiente forma:

$$\text{DEPT} \leftarrow \rho_{(\text{NombreDpto, NumDptoProyecto, DniDirector, FechaIngresoDirector}}(\text{DEPARTAMENTO})$$
$$\text{PROYECTO\_DPTO} \leftarrow \text{PROYECTO} * \text{DEPT}$$

El atributo NumDptoProyecto recibe el nombre de **atributo de conexión**. La Figura 6.7(a) muestra la relación resultante. En la relación PROYECTO_DPTO, cada tupla combina una de tipo PROYECTO con otra de DEPARTAMENTO para el departamento que controla el proyecto, aunque *sólo se mantiene un atributo de conexión*.

Si los atributos en los que se aplicará la concatenación natural ya *tienen el mismo nombre en ambas relaciones*, el renombrado es innecesario. Por ejemplo, para aplicar esta operación en los atributos NúmeroDpto de DEPARTAMENTO y LOCALIZACIONES_DPTO, es suficiente con escribir:

$$\text{LOC\_DPTO} \leftarrow . \text{ DEPARTAMENTO} * \text{LOCALIZACIONES\_DPTO}$$

La relación resultante aparece en la Figura 6.7(b), y en ella se combina cada departamento con sus localizaciones, exigiendo una tupla por cada una de estas localizaciones. En general, una CONCATENACIÓN NATURAL se lleva a cabo equiparando *todos* los pares de atributos que tengan el mismo nombre en las dos relaciones. Puede existir una lista de atributos de conexión para cada relación, y los pares correspondientes deben tener el mismo nombre.

De forma general, *aunque no estandarizada*, ésta es una definición de CONCATENACIÓN NATURAL:

$$Q \leftarrow R *_{(\text{<lista1>}),(\text{<lista2>})} S$$

En este caso, <lista1> especifica una lista de i atributos de R, mientras que <lista2> especifica una lista de i atributos de S. Estas listas se emplean para formar condiciones de comparación coherentes entre los atributos correspondientes para, a continuación, evaluarlas juntas mediante un operador AND. Sólo se mantiene en el resultado Q la lista de atributos correspondiente a la primera relación R (<lista1>).

Observe que si ninguna combinación de tuplas satisface la condición de conexión, el resultado de una CONCATENACIÓN es una relación vacía. En general, si R tiene n_R tuplas y S n_S, el resultado de una operación de CONCATENACIÓN $R \bowtie_{\text{<condición de conexión>}} S$ tendrá entre cero y $n_R * n_S$ tuplas. El tamaño estimado del resultado dividido entre el valor $n_R * n_S$ máximo da como resultado un cociente llamado **selectividad de concatenación (*join selectivity*)**, que es una propiedad de cada condición de conexión. Si no existe ninguna de ellas, todas las combinaciones de tuplas cualificadas y la CONCATENACIÓN degenera en un PRODUCTO CARTESIANO, llamado también PRODUCTO CRUZADO o CONCATENACIÓN CRUZADA.

Como podemos ver, la CONCATENACIÓN se emplea para combinar datos procedentes de múltiples relaciones, de forma que la información pueda presentarse en una única tabla. Estas operaciones se conocen también como **concatenaciones internas** (***inner joins***) para distinguirlas de una variación llamada *concatenaciones externas* (*outer joins)* (consulte la Sección 6.4.4). Informalmente, una *concatenación interna* es un tipo de operación de correspondencia y asociación definida formalmente como una combinación de un PRODUCTO

[5] La CONCATENACIÓN NATURAL es, básicamente, una EQUIJOIN seguida de la eliminación de los atributos superfluos.

Figura 6.7. Resultado de dos operaciones CONCATENACIÓN NATURAL. (a) PROYECTO_DPTO ← PROYECTO * DEPT. (b) LOC_DPTO ← DEPARTAMENTO * LOCALIZACIONES_DPTO.

(a)

PROYECTO_DPTO

NombreProyecto	NumProyecto	UbicacionProyecto	NumDpto-Proyecto	NombreDpto	DniDirector	FechaIngresoDirector
ProductoX	1	Valencia	5	Investigación	333445555	22-05-1988
ProductoY	2	Sevilla	5	Investigación	333445555	22-05-1988
ProductoZ	3	Madrid	5	Investigación	333445555	22-05-1988
Computación	10	Gijón	4	Administración	987654321	01-01-1995
Reorganización	20	Madrid	1	Sede Central	888665555	19-06-1981
Comunicaciones	30	Gijón	4	Administración	987654321	01-01-1995

(b)

LOC_DPTO

NombreDpto	NúmeroDpto	DniDirector	FechaIngresoDirector	Lugar
Sede Central	1	888665555	19-06-1981	Madrid
Administración	4	987654321	01-01-1995	Gijón
Investigación	5	333445555	22-05-1988	Valencia
Investigación	5	333445555	22-05-1988	Sevilla
Investigación	5	333445555	22-05-1988	Madrid

CARTESIANO y una SELECCIÓN. Una *concatenación externa* es otra versión más permisiva de la otra. Observe que puede especificarse una concatenación entre una relación y ella misma (consulte la Sección 6.4.3). La CONCATENACIÓN NATURAL o la EQUIJOIN pueden establecerse también entre múltiples tablas, lo que lleva a una *concatenación de n-vías*. Por ejemplo, considere la siguiente concatenación de tres vías:

$$((\text{PROYECTO} \bowtie_{\text{NúmeroDptoProyecto=NúmeroDpto}} \text{DEPARTAMENTO}) \bowtie_{\text{DniDirector=Dni}} \text{EMPLEADO})$$

Esta operación enlaza cada proyecto con el departamento al que pertenece para, a continuación, relacionarlo con su director. La malla resultante es una relación consolidada en la que cada tupla contiene esta información proyecto-departamento-director.

6.3.3 Un conjunto completo de operaciones de álgebra relacional

Ya hemos visto que todas las operaciones de álgebra relacional $\{\sigma, \pi, \cup, -, \times\}$ conforman un conjunto **completo**, es decir, que cualquiera de las operaciones originales puede expresarse como una *secuencia de operaciones de este conjunto*. Por ejemplo, la INTERSECCIÓN puede expresarse usando UNIÓN y MENOS del siguiente modo:

$$R \cap S \equiv (R \cup S) - ((R - S) \cup (S - R))$$

Aunque, estrictamente hablando, la INTERSECCIÓN no es necesaria, no es conveniente especificar esta compleja operación cada vez que queramos llevar a cabo una intersección. Por otro lado, y como ya se comentó, una CONCATENACIÓN puede especificarse como un PRODUCTO CARTESIANO seguido de una SELECCIÓN:

$$R \bowtie_{\text{<condición>}} S \equiv \sigma_{\text{<condición>}}(R \times S)$$

De forma análoga, una CONCATENACIÓN NATURAL es un PRODUCTO CARTESIANO precedido por una operación RENOMBRAR y seguido de una SELECCIÓN y una PROYECCIÓN. Así pues, las distintas operaciones CONCATENACIÓN no son *estrictamente necesarias* desde el elocuente poder del álgebra relacional. Sin embargo, es importante considerarlas como operaciones separadas porque es conveniente usarlas y se utilizan mucho en las aplicaciones de bases de datos más comunes. Alguna puede considerar RENOMBRAR como una operación esencial en el caso de que la necesidad de cambiar el nombre de una expresión de álgebra relacional sea imprescindible. Otras operaciones se han incluido en el álgebra relacional por conveniencia, más que por necesidad. Trataremos una de ellas en la sección siguiente.

6.3.4 La operación DIVISIÓN (*DIVISION*)

La DIVISIÓN, especificada mediante ÷, es útil para cierto tipo de consultas que a veces se realizan en aplicaciones de bases de datos. Un ejemplo es, *Recuperar los nombre de los empleados que trabajan en todos los proyectos en los que también lo haga 'José Pérez'*. Para expresar esta consulta usando una DIVISIÓN, proceda del siguiente modo. Primero, recupere en la relación intermedia PEREZ_PNOS la lista de números de proyecto en los que trabaja 'José Pérez':

$$\text{PEREZ} \leftarrow \sigma_{\text{Nombre='Jo´se' AND Apellido1='Pérez'}}(\text{EMPLEADO})$$

$$\text{PEREZ\_PNOS} \leftarrow \pi_{\text{NumProy}}(\text{TRABAJA\_EN} \bowtie_{\text{DniEmpleado=Dni}} \text{PEREZ})$$

A continuación, cree una relación que incluya una tupla <NumProy, DniEmpleado> siempre que el empleado cuyo Dni es DniEmpleado trabaje en el proyecto cuyo número es NumProy en la relación intermedia DNI_PNOS:

$$\text{DNI\_PNOS} \leftarrow \pi_{\text{DniEmpleado, NumProy}}(\text{TRABAJA\_EN})$$

Por último, aplique la DIVISIÓN a ambas relaciones, lo que nos facilita los Números Nacionales de Identidad de los empleados que queremos:

$$\text{DNIS(Dni)} \leftarrow \text{DNI\_PNOS} \div \text{PEREZ\_PNOS}$$

$$\text{RESULTADO} \leftarrow \pi_{\text{Nombre, Apellido1}}(\text{DNIS} * \text{EMPLEADO})$$

Las operaciones precedentes aparecen en la Figura 6.8(a).

En general, la operación DIVISIÓN se aplica a dos relaciones $R(Z) \div S(X)$, donde $X \subseteq Z$. Permite $Y = Z - X$ (y, por tanto, $Z = X \cup Y$); es decir, consiente que Y sea el conjunto de atributos de R que no lo son de S. El resultado de una DIVISIÓN es una relación $T(Y)$ que incluye una tupla t si las tuplas t_R aparecen en R con $t_R[Y] = t$, y con $t_R[X] = t_S$ para *cada* tupla t_S en S. Esto significa que, para que una tupla t aparezca en el resultado T de la DIVISIÓN, los valores de aquélla deben aparecer en R en combinación con cada tupla en S. Observe que en la formulación de la operación DIVISIÓN, las tuplas de la relación denominador restringen la relación numerador seleccionando aquellas tuplas del resultado que sean iguales a todos los valores presentes en el denominador. No es necesario saber qué valores son los que están presentes.

La Figura 6.8(b) ilustra una operación DIVISIÓN en la que $X = \{A\}$, $Y = \{B\}$ y $Z = \{A, B\}$. Observe que la tuplas (valores) b_1 y b_4 aparecen en R en combinación con las tres de S; este es el motivo por el que aparecen en la relación resultante T. El resto de valores de B en R no están en todas las tuplas de S, por lo que no se seleccionan: b_2 no aparece con a_2 ni b_3 con a_1.

La DIVISIÓN puede expresarse como una secuencia de operaciones π, $\times$ y $-$ del siguiente modo:

$$T1 \leftarrow \pi_Y(R)$$

$$T2 \leftarrow \pi_Y((S \times T1) - R)$$

$$T \leftarrow T1 - T2$$

Figura 6.8. La operación DIVISIÓN. (a) Dividiendo DNI_PNOS entre PEREZ_PNOS. (b) $T \leftarrow R \div S$.

(a)

DNI_PNOS

DniEmpleado	NumProy
123456789	1
123456789	2
666884444	3
453453453	1
453453453	2
333445555	2
333445555	3
333445555	10
333445555	20
999887777	30
999887777	10
987987987	10
987987987	30
987654321	30
987654321	20
888665555	20

PEREZ_PNOS

NumProy
1
2

DNIS

Dni
123456789
453453453

(b)

R

A	B
a1	b1
a2	b1
a3	b1
a4	b1
a1	b2
a3	b2T
a2	b3
a3	b3
a4	b3
a1	b4

S

A
a1
a2
a3

T

B
b1
b4

La operación DIVISIÓN está definida por conveniencia para gestionar las consultas que implican una *cuantificación universal* (consulte la Sección 6.6.7) o la condición *todo*. La mayoría de implementaciones RDBMS que cuentan con SQL como lenguaje de consulta primario no implementan directamente esta operación. SQL dispone de un camino alternativo para tratar el tipo de consulta mostrado más arriba (consulte la Sección 8.5.4). La Tabla 6.1 enumera las distintas operaciones de álgebra relacional básicas que hemos visto.

6.3.5 Notación para los árboles de consultas

En esta sección vamos a tratar una notación usada habitualmente en sistemas relacionales para representar consultas internamente. Dicha notación recibe el nombre de árbol de consulta, o también árbol de evaluación de consulta o árbol de ejecución de consulta. Permite la ejecución de las operaciones del álgebra relacional y se utiliza como una posible estructura de datos para la representación interna de la consulta en un RDBMS.

Un árbol de consulta es una estructura de datos en árbol que se corresponde con una expresión de álgebra relacional. Representa las relaciones de entrada de la consulta como los *nodos hoja* del árbol y las operaciones como nodos internos. La ejecución de uno de estos árboles supone la ejecución de la operación de un nodo interno, siempre que estén disponibles sus operandos, para, a continuación, reemplazar ese nodo interno por la relación que resulta de la ejecución de la operación. El proceso concluye cuando se ejecuta el nodo raíz y se obtiene la relación resultante de la consulta.

La Figura 6.9 muestra un árbol de consulta para la consulta Q2: *Para cada proyecto localizado en 'Gijón', recuperar el número del mismo, el número del departamento que lo controla y la fecha de nacimiento, direc-*

Tabla 6.1. Operaciones del álgebra relacional.

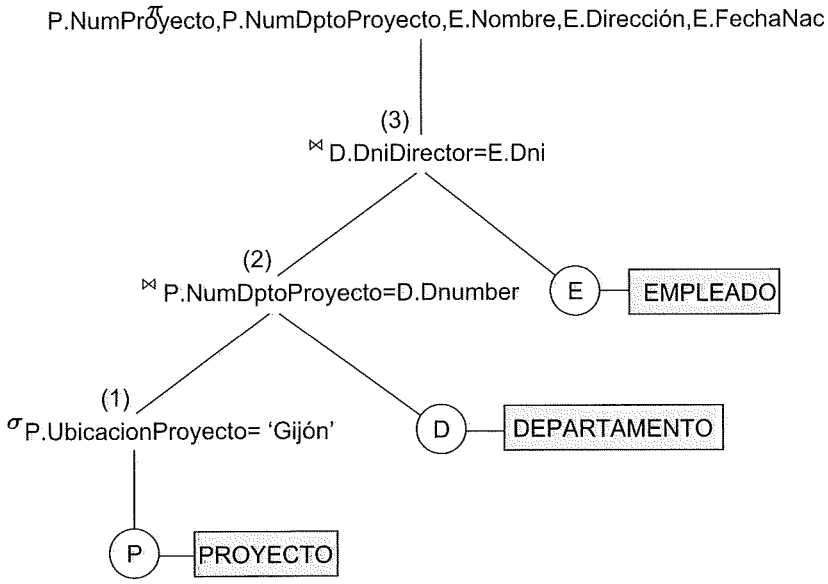
Operación	Objetivo	Notación
SELECCIÓN	Selecciona todas las tuplas de una relación R que satisfacen la condición de selección.	$\sigma_{<\text{condición de selección}>}(R)$
PROYECCIÓN	Produce una nueva relación en la que sólo existen algunos de los atributos de R, y elimina las tuplas duplicadas.	$\pi_{<\text{lista de atributos}>}(R)$
ASOCIACIÓN (*THETA JOIN*)	Genera todas las combinaciones de tuplas de R_1 y R_2 que satisfacen la condición de conexión.	$R_1 \bowtie_{<\text{condición de conexión}>} R_2$
EQUIJOIN	Genera todas las combinaciones de tuplas de R_1 y R_2 que satisfacen una condición de conexión sólo con comparaciones de igualdad.	$R_1 \bowtie_{<\text{condición de conexión}>} R_2$ OR $R_1 \bowtie_{(<\text{atributos de conexión 1}>), (<\text{atributos de conexión 2}>)} R_2$
CONCATENACIÓN NATURAL	Es lo mismo que EQUIJOIN excepto por el hecho de que los atributos de conexión de R_2 no están incluidos en la relación resultante; si estos atributos tienen los mismos nombres, no tienen que especificarse.	$R_1 *_{<\text{condición de conexión}>} R_2$ OR $R_1 *_{(<\text{atributos de conexión 1}>), (<\text{atributos de conexión 2}>)} R_2$ OR $R_1 * R_2$
UNIÓN	Produce una relación que incluye todas las tuplas de R_1 o R_2 o de ambas; R_1 y R_2 deben ser de unión compatible.	$R_1 \cup R_2$
INTERSECCIÓN	Produce una relación que incluye todas las tuplas que están en R_1 y R_2; R_1 y R_2 deben ser compatibles con la unión.	$R_1 \cap R_2$
DIFERENCIA	Produce una relación que incluye todas las tuplas de R_1 que no están en R_2; R_1 y R_2 deben ser compatibles con la unión.	$R_1 - R_2$
PRODUCTO CARTESIANO	Produce una relación que tiene los atributos de R_1 y R_2 e incluye tantas tuplas como posibles combinaciones de tuplas de R_1 y R_2.	$R_1 \times R_2$
DIVISIÓN	Produce una relación $R(X)$ que incluye todas las tuplas $t[X]$ en $R_1(Z)$ que aparecen en R_1 en combinación con cada tupla de $R_2(Y)$, donde $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

ción, nombre y apellidos de su director. Esta consulta está especificada en el esquema relacional de la Figura 5.5 y se corresponde con la siguiente expresión de álgebra relacional:

$$\pi_{\text{NumProyecto, NumDptoProyecto, Apellido1, Dirección, FechaNac}}(((\sigma_{\text{UbicacionProyecto='Gijón'}}(\text{PROYECTO})) \bowtie_{\text{NumDptoProyecto=NúmeroDpto}}(\text{DEPARTAMENTO})) \bowtie_{\text{DniDirector=Dni}}(\text{EMPLEADO}))$$

En la Figura 6.9, las tres relaciones PROYECTO, DEPARTAMENTO y EMPLEADO están representadas por los nodos hoja P, D y E, mientras que las operaciones de álgebra relacional de la expresión lo están por tres nodos árbol internos. Esto supone el siguiente orden de ejecución de Q2. El nodo marcado como (1) en la Figura 6.9 debe ejecutarse antes que el (2) porque algunas de las tuplas resultantes de la operación (1) deben estar disponibles antes de ejecutar la operación (2). De forma análoga, el nodo (2) debe ejecutarse y producir

Figura 6.9. Árbol de consulta correspondiente a la expresión de álgebra de Q2.



resultados antes que lo haga el (3), y así sucesivamente. En general, un árbol de consulta ofrece una correcta representación visual y comprensión de la consulta en términos de las operaciones relacionales que usa, y está recomendado como medio adicional de expresar consultas en el álgebra relacional. Volveremos a los árboles de consulta cuando tratemos el procesamiento y la optimización de consultas en el Capítulo 15.

6.4 Operaciones relacionales adicionales

Existen algunas peticiones habituales a bases de datos, las cuales son necesarias en aplicaciones comerciales para los RDBMS, que no pueden llevarse a cabo con las operaciones de álgebra relacional descritas en las Secciones de la 6.1 a la 6.3. A continuación vamos a ver esas expresiones, que mejoran considerablemente la potencia del álgebra relacional original.

6.4.1 Proyección generalizada

La proyección generalizada es una operación que amplía las posibilidades de la proyección original permitiendo la inclusión de funciones de atributos en la lista de proyección. La forma generalizada puede expresarse del siguiente modo:

$$\pi_{F_1, F_2, \ldots, F_n}(R)$$

donde $F_1, F_2, ..F_n$ son funciones sobre los atributos de la relación R y pueden involucrar constantes. Esta operación está ideada como una ayuda a la hora de desarrollar informes en los que los valores calculados deben generarse en columnas.

Como ejemplo, considere la relación:

EMPLEADO (Dni, Sueldo, Deducción, Antiguedad)

Un informe podría necesitar mostrar:

SalarioNeto = Sueldo − Deducción,

Gratificaciones = 2000 * Antiguedad e

Impuestos = 0,25 * Sueldo.

De este modo, puede usarse una proyección generalizada combinada con una operación de renombrado de la siguiente forma:

INFORME. $\leftarrow \rho_{(\text{Dni, SalarioNeto, Gratificaciones, Impuestos})}$
$(\pi_{\text{Dni, Sueldo – Deducción, 2000 * Antiguedad, 0.25 * Sueldo}}(\text{EMPLEADO}))$.

6.4.2 Funciones de agregación y agrupamiento

Otro tipo de peticiones que no pueden expresarse a través del álgebra relacional básico son las que se utilizan para calcular **funciones matemáticas de agregación** en las colecciones de valores de la base de datos. Como ejemplos podemos citar funciones para recuperar la media o el sueldo total de todos los empleados o el número total de tuplas de empleados. Todas estas operaciones se utilizan en consultas estadísticas sencillas que resumen información procedente de las tuplas de la base de datos, y las más comunes son SUMA (SUM), MEDIA (*AVERAGE*), MÁXIMO (*MAXIMUM*) y MÍNIMO (*MINIMUM*). La función CONTAR (*COUNT*) se emplea para contar tuplas o valores.

Otro tipo común de petición supone realizar la agrupación de las tuplas de una relación por el valor de uno de sus atributos y la aplicación posterior de una función de agregación independiente a cada grupo. Un ejemplo podría ser clasificar tuplas por Dno, de modo que cada grupo incluya sólo los empleados que trabajan en el mismo departamento. A continuación queremos listar cada valor Dno junto con, por ejemplo, la media del sueldo de todos esos empleados, o el número de los que trabajan en ese departamento.

Podemos definir una operación FUNCIÓN AGREGADA (*AGGREGATE FUNCTION*) usando el símbolo $\Im$ (pronunciado *script F*)[6], para especificar este tipo de peticiones:

<atributos de agrupamiento> $\Im$ <lista de funciones> (R)

donde <atributos de agrupamiento> es una lista de los atributos de la relación especificados en R, y <lista de funciones> es una lista de parejas (<función> <atributo>). En cada una de estas parejas, <función> puede ser SUMA, MEDIA, MÁXIMO, MÍNIMO o CONTAR, mientras que <atributo> es un atributo de la relación especificada por R. La relación resultante cuenta con los atributos de agrupamiento además de otro por cada elemento de la lista de funciones. Por ejemplo, para recuperar cada número de departamento, el número de empleados del mismo y la media de sueldos, renombrando los atributos resultantes tal y como se especifica más adelante, podemos escribir:

$\rho_{R(\text{Dno, NumEmpleados, MediaSueldos})}$ (Dno $\Im$ COUNT Dni, AVERAGE Sueldo (EMPLEADO))

El resultado de esta operación de la relación EMPLEADO de la Figura 5.6 se muestra en la Figura 6.10(a).

En el ejemplo anterior, especificamos una lista de nombres de atributos (entre los paréntesis de la operación RENOMBRAR) para la relación resultante R. En caso de no aplicarse este cambio de nombre, los atributos correspondientes a la lista de funciones serán el resultado de la concatenación del nombre de la función con el del atributo en la forma <función>_<atributo>.[7] Por ejemplo, la Figura 6.10(b) muestra el resultado de la siguiente operación:

Dno $\Im$ COUNT Dni, AVERAGE Sueldo (EMPLEADO)

Si no se especifican atributos de agrupamiento, las funciones se aplican a *todas las tuplas* de la relación, por lo que obtendremos como resultado una única *tupla*. Por ejemplo, la Figura 6.10(c) muestra el resultado de la siguiente operación:

---

[6] No existe ninguna notación añadida para representar las funciones de agregación. En algunos casos se emplea un "script A".

[7] Tenga en cuenta que ésta es una notación arbitraria que hemos sugerido. No existe una notación estándar.

Figura 6.10. Operativa de una función de agregación.
(a) $\rho_{R(\text{Dno, NumEmpleados, MediaSueldos})}$ (Dno $\Im$ COUNT Dni, AVERAGE Sueldo (EMPLEADO)).
(b) Dno $\Im$ COUNT Dni, AVERAGE Sueldo (EMPLEADO).
(c) $\Im$ COUNT Dni, AVERAGE Sueldo (EMPLEADO).

(a) R

Dno	NumEmpleados	MediaSueldos
5	4	33250
4	3	31000
1	1	55000

(b)

Dno	ContarDni	PromedioSueldo
5	4	33250
4	3	31000
1	1	55000

(c)

ContarDni	PromedioSueldo
8	35125

$\Im$ COUNT Dni, AVERAGE Sueldo (EMPLEADO)

Es importante indicar que, en general, las duplicaciones *no se eliminan* cuando se aplica una función de agregación; de esta forma, la interpretación normal de funciones como SUMA y MEDIA es calculada.[8] Merece la pena enfatizar que el resultado de aplicar una función de agregación es una relación, y no un número escalar, aun cuando sólo tenga un valor. Esto hace del álgebra relacional un sistema cerrado.

6.4.3 Operaciones de cierre recursivo

Otro tipo de operación que, en general, no puede especificarse en el álgebra relacional básico, es el **cierre recursivo**. Esta operación se aplica a una **relación recursiva** entre las tuplas del mismo tipo, como la que se establece entre un empleado y un supervisor. Esta relación está descrita por la *foreign key* SuperDni de EMPLEADO de las Figuras 5.5 y 5.6, y relaciona cada tupla de empleado supervisado con otra que lo supervisa. Un ejemplo de operación recursiva es la recuperación de las supervisiones de un empleado *e* a todos sus niveles, es decir, todos los empleados *e'* supervisados directamente por *e*, todos los *e''* que lo son por *e'*, los *e'''* que los son por *e''*, y así sucesivamente.

Aunque en el álgebra relacional especificar todos los empleados supervisados por *e a un nivel específico* es directo, es complicado hacerlo a *todos* los niveles. Por ejemplo, para indicar los Dni de todos los empleados *e'* supervisados directamente (a *nivel uno*) por el empleado *e* cuyo nombre es 'Eduardo Ochoa' (consulte la Figura 5.6), podemos aplicar la siguiente operación:

DNI_OCHOA $\leftarrow \pi$Dni($\sigma_{\text{Nombre='Eduardo' AND Apellido1='Ochoa'}}$(EMPLEADO))
SUPERVISION(Dni1, Dni2) $\leftarrow \pi_{\text{Dni, SuperDni}}$(EMPLEADO)
RESULTADO1(DNI) $\leftarrow \pi$Dni1(SUPERVISION $\bowtie_{\text{Dni2=Dni}}$ DNI_OCHOA)

Para recuperar todos los empleados supervisados por Ochoa a nivel 2 (esto es, los empleados *e''* supervisados por algún *e'* que está supervisado directamente por Ochoa) podemos aplicar otra CONCATENACIÓN al resultado de la primera consulta de la siguiente forma:

RESULTADO2(Dni) $\leftarrow \pi_{\text{Dni1}}$(SUPERVISION$\bowtie_{\text{Dni2=Dni}}$ RESULTADO1)

[8] En SQL, existe la posibilidad de eliminar duplicados antes de aplicar la función de agregación incluyendo la palabra DISTINCT (consulte la Sección 8.4.4).

Figura 6.11. Una consulta recursiva a dos niveles.

SUPERVISION

(El DNI de Ochoa es 888665555)

(Dni) (SuperDni)

Dni1	Dni2
123456789	333445555
333445555	888665555
999887777	987654321
987654321	888665555
666444444	333445555
453453453	333445555
987987987	987654321
888665555	null

RESULTADO1

Dni
333445555
987654321

(Supervisado por Ochoa)

RESULTADO2

Dni
123456789
999887777
666884444
453453453
987987987

(Supervisado por subordinados de Ochoa)

RESULTADO

Dni
123456789
999887777
666884444
453453453
987987987
333445555
987654321

(RESULTADO1 ∪ RESULTADO2)

Para obtener los conjuntos de empleados supervisados a los niveles 1 y 2 por 'Eduardo Ochoa', podemos aplicar la operación UNIÓN a los dos resultados:

RESULTADO ← RESULTADO2 ∪ RESULTADO1

Los resultados de estas consultas aparecen en la Figura 6.11. Aunque es posible recuperar los empleados de cada nivel y después realizar su UNIÓN no podemos, en general, especificar una consulta del tipo "recuperar los empleados supervisados por 'Eduardo Ochoa' a todos los niveles" sin emplear un mecanismo de bucle.[9] Se ha propuesto una operación llamada *cierre transitivo* de relaciones para procesar la relación recursiva hasta donde procede la recursión.

6.4.4 Operaciones CONCATENACIÓN EXTERNA (OUTER JOIN)

A continuación vamos a tratar algunas extensiones de la operación CONCATENACIÓN necesarias para especificar ciertos tipos de consultas. Las operaciones CONCATENACIÓN descritas anteriormente emparejan

[9] El estándar SQL3 incluye sintaxis para el cierre recursivo.

tuplas que satisfacen la condición de conexión. Por ejemplo, para una CONCATENACIÓN NATURAL $R * S$, sólo aparecen en el resultado las tuplas de R que coinciden con las de S (y viceversa). Por consiguiente, aquellas tuplas sin *coincidencia* (o *relacionadas*) se eliminan del resultado. Las tuplas con valores NULL en los atributos de conexión también se eliminan. Esto equivale a una pérdida de información en el caso de que CONCATENACIÓN se utilice para generar un informe basado en todos los datos de las relaciones.

Existe un conjunto de operaciones, llamadas **concatenaciones externas**, que pueden usarse cuando queremos mantener en el resultado todas las tuplas de R, o de S, o de ambas, independientemente de si tienen correspondencias o no en la otra relación. Esto permite ejecutar consultas en las que tuplas procedentes de dos tablas se combinan emparejando las filas correspondientes, pero sin perder aquéllas que no tienen ese "compañero". Las operaciones de concatenación descritas en la Sección 6.3, que sólo mantenían las tuplas coincidentes, reciben el nombre de **concatenaciones internas**.

Por ejemplo, supongamos que queremos una lista de todos los nombres de empleados, junto con los de los departamentos que controlan, en el *caso de que dirijan un departamento*; en caso de no hacerlo, podemos indicarlo con un valor NULL. Podemos aplicar una operación **CONCATENACIÓN EXTERNA IZQUIERDA (*LEFT OUTER JOIN*)**, indicada por $\rtimes$, para recuperar el resultado:

$$\text{TEMP} \leftarrow (\textbf{EMPLEADO} \rtimes_{\text{Dni=DniDirector}} \text{DEPARTAMENTO})$$
$$\text{RESULTADO} \leftarrow \pi_{\text{Nombre, Apellido1, Apellido2, NombreDpto}}(\text{TEMP})$$

La operación CONCATENACIÓN EXTERNA IZQUIERDA mantiene cada tupla de la *primera* relación, o relación *izquierda*, R en $R \rtimes S$; si no se encuentra ninguna tupla en S, sus atributos en la concatenación resultante se rellenan con valores NULL. La Figura 6.12 muestra el resultado de esta operación.

Una operación parecida, la **CONCATENACIÓN EXTERNA DERECHA (*RIGHT OUTER JOIN*)**, especificada por $\ltimes$, es una operación similar que mantiene cada tupla de la *segunda* relación o relación *derecha*, S en el resultado de $R \ltimes S$. Una tercera operación, **CONCATENACIÓN EXTERNA COMPLETA (*FULL OUTER JOIN*)**, expresada por $\rtimes$, mantiene todas las tuplas de ambas relaciones (las de la derecha y las de la izquierda) cuando no existen tuplas coincidentes, rellenándolas con valores NULL cuando sea necesario. Las tres operaciones forman parte del estándar SQL2 (consulte el Capítulo 8), y fueron incorporadas más tarde como una extensión del álgebra relacional en respuesta a las necesidades de las aplicaciones empresariales relacionadas con la información procedente de múltiples tablas. A veces se hace necesario generar informes de datos procedentes de varias tablas independientemente de que existan o no valores coincidentes.

Figura 6.12. El resultado de una operación CONCATENACIÓN EXTERNA IZQUIERDA.

RESULTADO

Nombre	Apellido1	Apellido2	NombreDpto
José	Pérez	Pérez	NULL
Alberto	Campos	Sastre	Investigación
Alicia	Jiménez	Celaya	NULL
Juana	Sainz	Oreja	Administración
Fernando	Ojeda	Ordóñez	NULL
Aurora	Oliva	Avezuela	NULL
Luis	Pajares	Morera	NULL
Eduardo	Ochoa	Paredes	Sede Central

6.4.5 La operación UNIÓN EXTERNA (*OUTER UNION*)

La **UNIÓN EXTERNA** fue desarrollada para obtener la unión de tuplas de dos relaciones en el caso de que esas relaciones *no sean compatibles con la unión*. Esta operación tomará la UNIÓN de tuplas de dos relaciones $R(X, Y)$ y $S(X, Z)$ que son **parcialmente compatibles**, lo que significa que sólo algunos de sus atributos son compatibles con la unión. Estos atributos sólo aparecen una vez en el resultado, y los que no son compatibles con la unión también se mantienen en la relación resultante $T(X, Y, Z)$.

Dos tuplas, t_1 en R y t_2 en S, se dice que son **coincidentes** si $t_1[X]=t_2[X]$, y se considera que representan la misma entidad o instancia de relación. Se combinarán en una única tupla en T. Las tuplas de una relación que no coinciden con las de la otra relación se rellenan con valores NULL. Por ejemplo, puede aplicarse una UNIÓN EXTERNA a dos relaciones cuyos esquemas son ESTUDIANTE(Nombre, Dni, Departamento, Tutor) y PROFESOR(Nombre, Dni, Departamento, Cargo). Las tuplas de dos relaciones se emparejan en función a la misma combinación de valores de los atributos compartidos (Nombre, Dni, Departamento). La relación resultante, ESTUDIANTE_O_PROFESOR, tendrá los siguientes atributos:

ESTUDIANTE_O_PROFESOR(Nombre, Dni, Departamento, Tutor, Cargo)

Todas las tuplas de ambas relaciones están incluidas en el resultado, aunque las que tienen la combinación (Nombre, Dni, Departamento) aparecerán sólo una vez. Las contenidas solamente en la relación ESTUDIANTE tendrán NULL para el atributo Cargo, mientras que las de PROFESOR lo tendrán en Tutor. Una tupla existente en ambas relaciones, como un estudiante que también es un profesor, tendrá valor en todos sus atributos.[10]

Observe que la misma persona podría aparecer dos veces en el resultado. Por ejemplo, podríamos tener un estudiante graduado en el departamento de Matemáticas que fuera profesor del de Informática. Aunque las tuplas que representan a esa persona en ESTUDIANTE y PROFESOR tienen los mismos valores para (Nombre, Dni), no lo tienen para Departamento, lo que hará que no sean coincidentes. Esto se debe a que Departamento tiene dos significados distintos dependiendo de si se trata de un ESTUDIANTE (el departamento en el que esa persona estudia) o un PROFESOR (en el que trabaja como profesor). Si queremos unir personas en base a la misma combinación (Nombre, Dni), deberemos renombrar el atributo Departamento en cada tabla de modo que indiquen que tienen significados distintos y hacer que no formen parte de los atributos compatibles con la unión.

Otra capacidad disponible en la mayoría de lenguajes comerciales (aunque no en el álgebra relacional básica) es la posibilidad de especificar operaciones en los valores una vez extraídos de la base de datos. Por ejemplo, puede aplicarse operaciones aritméticas como +, − y * a los valores numéricos que aparecen en el resultado de una consulta, tal y como ya comentamos en la Sección 6.4.1.

6.5 Ejemplos de consultas del álgebra relacional

A continuación vamos a ver algunos ejemplos adicionales que ilustran el uso de las operaciones de álgebra relacional. Todos ellos están referidos a la base de datos de la Figura 5.6. En general, la misma consulta puede declararse de diversas formas usando distintas operaciones. Aquí utilizaremos una de esas nomenclaturas, y le dejaremos al lector el trabajo de obtener formulaciones equivalentes.

Consulta 1. Recupere el nombre y la dirección de todos los empleados que trabajan en el departamento 'Investigación'.

$$\text{DPTO\_INVESTIGACION} \leftarrow \sigma_{\text{NombreDpto='Investigación'}}(\text{DEPARTAMENTO})$$

$$\text{EMPS\_INVESTIGACION} \leftarrow (\text{DPTO\_INVESTIGACION} \bowtie_{\text{NúmeroDpto=Dno}} \text{EMPLEADO})$$

[10] Observe que UNIÓN EXTERNA es equivalente a una CONCATENACIÓN EXTERNA COMPLETA si los atributos de conexión son todos los atributos comunes de las dos relaciones.

$$\text{RESULTADO} \leftarrow \pi_{\text{Nombre, Apellido1, Dirección}}(\text{EMPS\_INVESTIGACION})$$

Como una expresión única, esta consulta se convierte en:

$$\pi_{\text{Nombre, Apellido1, Dirección}}(\sigma_{\text{NombreDpto= 'Investigación'}}(\text{DEPARTAMENTO} \bowtie_{\text{NúmeroDpto=Dno}} (\text{EMPLEADO}))$$

La consulta podría expresarse de otras formas; por ejemplo, podría invertirse el orden de las operaciones CONCATENACIÓN y SELECCIÓN, o CONCATENACIÓN podría sustituirse por una CONCATENACIÓN NATURAL después de cambiar el nombre a uno de los atributos de conexión.

Consulta 2. Por cada proyecto ubicado en 'Gijón', enumere su número, el número de departamento que lo gestiona y los apellidos, dirección y fecha de nacimiento del director del departamento.

$$\text{PROYECTOS\_GIJON} \leftarrow \sigma_{\text{UbicacionProyecto='Gijón'}}(\text{PROYECTO})$$

$$\text{DEPT\_CONTROL} \leftarrow (\text{PROYECTOS\_GIJON} \bowtie_{\text{NumDptoProyecto=NúmeroDpto}} \text{DEPARTAMENTO})$$

$$\text{DIRECTOR\_DPTO\_PROYECTO?(DEPT\_CONTROL} \bowtie_{\text{DniDirector=Dni}} \text{EMPLEADO})$$

$$\text{RESULTADO} \leftarrow \pi_{\text{NumProyecto, NumDptoProyecto, Apellido1, Dirección, FechaNac}}(\text{DIRECTOR\_DPTO\_PROYECTO})$$

Consulta 3. Localice los nombres de los empleados que trabajan en *todos* los proyectos gestionados por el departamento número 5.

$$\text{PROYECTOS\_DEPT5(NumProy)} \leftarrow \pi_{\text{NumProyecto}}(\sigma_{\text{NumDptoProyecto=5}}(\text{PROYECTO}))$$

$$\text{EMP\_PROYECTO(Dni, NumProy)} \leftarrow \pi_{\text{DniEmpleado, NumProy}}(\text{TRABAJA\_EN})$$

$$\text{DNI\_EMPS\_RESULTADO} \leftarrow \text{EMP\_PROYECTO} \div \text{PROYECTOS\_DEPT5}$$

$$\text{RESULTADO} \leftarrow \pi_{\text{Apellido1, Nombre}}(\text{DNI\_EMPS\_RESULTADO} * \text{EMPLEADO})$$

Consulta 4. Haga una lista de los números de proyecto en los que esté involucrado cualquier empleado cuyo primer apellido sea 'Pérez', ya sean trabajadores o directores del departamento que gestiona ese proyecto.

$$\text{PEREZ(DniEmpleado)} \leftarrow \pi_{\text{Dni}}(\sigma_{\text{Apellido1='Pérez'}}(\text{EMPLEADO}))$$

$$\text{PROYS\_PEREZ} \leftarrow \pi_{\text{NumProy}}(\text{TRABAJA\_EN} * \text{PEREZ})$$

$$\text{DIRECTORES} \leftarrow \pi_{\text{Apellido1, NúmeroDpto}}(\text{EMPLEADO}_{\text{Dni=DniDirector}}\text{DEPARTAMENTO})$$

$$\text{DEPTS\_ADMINISTRADOS\_PEREZ(NumDptoProyecto)} \leftarrow \pi_{\text{NúmeroDpto}}(\sigma_{\text{Apellido1='Pérez'}}(\text{DIRECTORES}))$$

$$\text{DEPTS\_DIRECTOR\_PEREZ(NumProy)} \leftarrow \pi_{\text{NumProyecto}}(\text{DEPTS\_ADMINISTRADOS\_PEREZ} * \text{PROYECTO})$$

$$\text{RESULTADO} \leftarrow (\text{PROJS\_PEREZ} \cup \text{DEPTS\_DIRECTOR\_PEREZ})$$

Como una única expresión, esta consulta se transforma en

$$\pi_{\text{NumProy}}(\text{TRABAJA\_EN} \bowtie_{\text{DniEmpleado=Dni}} (\pi_{\text{Dni}}(\sigma_{\text{Apellido1='Pérez'}}(\text{EMPLEADO}))) \cup \pi_{\text{NumProy}}((\pi_{\text{NúmeroDpto}}(\sigma_{\text{Apellido1='Pérez'}}(\pi_{\text{Apellido1, NúmeroDpto}}(\text{EMPLEADO}))) \bowtie_{\text{Dni=DniDirector}} \text{DEPARTAMENTO})) \bowtie_{\text{NúmeroDpto=NumDptoProyecto}} \text{PROYECTO})$$

Consulta 5. Liste los nombres de todos los empleados con dos o más subordinados.

Estrictamente hablando, esta consulta no puede llevarse a cabo con el *álgebra relacional básica* (*original*). Tenemos que usar la operación FUNCIÓN AGREGADA con la función CONTAR. Asumimos que los asalariados del *mismo* empleado tienen *distinto* valor NOMBRE_SUBORDINADO.

$T1(\text{Dni, NumSubordinados}) \leftarrow_{\text{DniEmpleado}} \Im_{\text{COUNT NombreSubordinado}}(\text{SUBORDINADO})$

$T2 \leftarrow \sigma_{\text{NumSubordinados} \geq 2}(T1)$

$\text{RESULTADO} \leftarrow \pi_{\text{Apellido1, Nombre}}(T2 * \text{EMPLEADO})$

Consulta 6. Recuperar los nombres de los empleados que no tienen subordinados.

Se trata de un ejemplo del tipo de consulta que utiliza la operación MENOS (DIFERENCIA DE CONJUNTOS).

$\text{TODOS\_EMPLEADOS} \leftarrow \pi_{\text{Dni}}(\text{EMPLEADO})$

$\text{EMPS\_CON\_SUBORDINADOS(Dni)} \leftarrow \pi_{\text{DniEmpleado}}(\text{SUBORDINADO})$

$\text{EMPS\_SIN\_SUBORDINADOS} \leftarrow (\text{TODOS\_EMPLEADOS} - \text{EMPS\_CON\_SUBORDINADOS})$

$\text{RESULTADO} \leftarrow \pi_{\text{Apellido1, Nombre}}(\text{EMPS\_SIN\_SUBORDINADOS} * \text{EMPLEADO})$

Como una única expresión, esta consulta queda del siguiente modo:

$$\pi_{\text{Apellido1, Nombre}}((\pi_{\text{Dni}}(\text{EMPLEADO}) - \rho_{\text{Dni}}(\pi_{\text{DniEmpleado}}(\text{SUBORDINADO}))) * \text{EMPLEADO})$$

Consulta 7. Enumerar los nombres de los directivos que tienen, al menos, un subordinado.

$\text{DIRECTORES(Dni)} \leftarrow \pi_{\text{DniDirector}}(\text{DEPARTAMENTO})$

$\text{EMPS\_CON\_SUBORDINADOS(Dni)} \leftarrow \pi_{\text{DniEmpleado}}(\text{SUBORDINADO})$

$\text{DIRECTORES\_CON\_SUBORDINADOS} \leftarrow (\text{DIRECTORES} \cap \text{EMPS\_CON\_SUBORDINADOS})$

$\text{RESULTADO} \leftarrow \pi_{\text{Apellido1, Nombre}}(\text{DIRECTORES\_CON\_SUBORDINADOS} * \text{EMPLEADO})$

Como ya se comentó anteriormente, en general, la misma consulta puede especificarse de diferentes formas. Por ejemplo, las operaciones pueden aplicarse a menudo en órdenes diferentes. Además, algunas de ellas pueden sustituirse por otras; por ejemplo, la INTERSECCIÓN en Q7 puede sustituirse por una CONCATENACIÓN NATURAL. Como ejercicio, intente redefinir los ejemplos anteriores con operaciones diferentes.[11] Le mostramos cómo escribir consultas como expresiones simples de álgebra relacional para las consultas Q1, Q4 y Q6. Intente desarrollar las restantes como expresiones sencillas. En el Capítulo 8 y en las Secciones 6.6 y 6.7 mostramos el modo de reescribir estas consultas en otros lenguajes relacionales.

6.6 Cálculos relacionales de tupla

En éste y en el siguiente ejercicio vamos a estudiar otro lenguaje de consulta formal para el modelo relacional llamado **cálculo relacional**. En él escribimos una expresión **declarativa** para especificar una consulta de recuperación; por tanto, no hay una descripción del modo de evaluar una consulta. Una expresión de cálculo especifica *qué* se quiere recuperar en lugar de *cómo* hacerlo, por lo que se le considera un lenguaje **no procedural**. Esto le hace diferente del álgebra relacional, en donde debemos escribir una *secuencia de operaciones* para especificar la consulta de recuperación, razón por la que se le puede considerar como una forma **procedural** de declarar la misma. Es posible anidar operaciones de álgebra para formar una única expresión; sin embargo, siempre es necesario indicar explícitamente un cierto orden en una expresión de álgebra relacional. Este orden influye también en la estrategia de evaluación de la consulta. Una expresión de cálculo puede escribirse de diferentes formas, aunque esto no influye en el modo en que dicha consulta será evaluada.

Hemos visto también que cualquier recuperación que pueda especificarse mediante el álgebra relacional básico puede hacerse del mismo modo a través de cálculos relacionales, y viceversa; en otras palabras, la

[11] Cuando las consultas están optimizadas (consulte el Capítulo 15), el sistema elegirá una secuencia particular de operaciones que se corresponde con la mejor estrategia de ejecución.

potencia expresiva de ambos lenguajes es *idéntica*. Esto conduce a la definición del concepto de un lenguaje relacionalmente completo. Se considera que un lenguaje de consulta relacional L es **relacionalmente completo** si podemos expresar en él cualquier consulta que pueda realizarse mediante un cálculo relacional. La integridad relacional se ha convertido en una base importante para la comparación de la potencia expresiva de los lenguajes de consulta de alto nivel. Sin embargo, como ya vimos en la Sección 6.4, ciertas consultas muy frecuentes en las aplicaciones de bases de datos no pueden expresarse con ninguno de estos dos métodos. La mayoría de los lenguajes de consulta relacionales son relacionalmente completos, aunque tienen *más potencia expresiva* que el álgebra o los cálculos relacionales debido a ciertas operaciones añadidas, como las funciones agregadas, la agrupación y la ordenación.

En esta sección y en la siguiente, todos los ejemplos se refieren a la base de datos mostrada en las Figuras 5.6 y 5.7. Usaremos las mismas consultas de la Sección 6.5. Las Secciones 6.6.6, 6.6.7 y 6.6.8 tratan de los cuantificadores universales y los problemas de seguridad de una expresión. Estas secciones pueden saltárselas aquellos estudiantes interesados en una introducción general a los cálculos de tupla.

6.6.1 Variables de tupla y relaciones de rango

Los cálculos relacionales de tupla están basados en la especificación de un número de **variables de tupla**. Cada una de ellas suele *aplicarse sobre* una relación de base de datos particular, lo que significa que la variable podría tomar su valor de cualquier tupla individual de esa relación. Una consulta de cálculo relacional sencilla tiene la siguiente forma:

$\{t \mid \text{COND}(t)\}$

donde t es una variable de tupla y COND(t) es una expresión condicional que implica a t. El resultado es el conjunto de todas las tuplas t que satisfacen COND(t). Por ejemplo, para localizar todos los empleados cuyo salario es superior a 50.000 euros podemos escribir la siguiente expresión:

$\{t \mid \text{EMPLEADO}(t)$ **AND** $t.\text{Sueldo}>50000\}$

La condición EMPLEADO(t) especifica que la **relación de rango** de la variable de tupla t es EMPLEADO. Cada EMPLEADO t que satisface la condición t.Sueldo>50000 será recuperada. Observe que t.Sueldo hace referencia al atributo Sueldo de la variable de tupla t; esta notación se asemeja a cómo se cualifican los nombres de atributo con los de relación, o alias, en SQL, tal y como veremos en el Capítulo 8. En la notación del Capítulo 5, t.Sueldo es lo mismo que decir t [Sueldo].

La consulta anterior recupera todos los valores de atributo de cada tupla t EMPLEADO seleccionada. Para obtener sólo *algunos* de los atributos (por ejemplo, el nombre y el primer apellido) escribimos:

$\{t.\text{Nombre}, t.\text{Apellido1} \mid \text{EMPLEADO}(t)$ **AND** $t.\text{Sueldo}>50000\}$

Informalmente, tenemos que especificar la siguiente información en una expresión de cálculo de tupla:

- Para cada variable de tupla t, la **relación de rango** R de t. Este valor se indica con una condición de la forma $R(t)$.
- Una condición para seleccionar combinaciones de tuplas particulares. Como las variables de tupla alcanzan a sus respectivas relaciones de rango, la condición se evalúa por cada combinación posible de tuplas para identificar las **combinaciones seleccionadas** que la condición evalúa como VERDADERO.
- El conjunto de los atributos a recuperar, los **atributos solicitados**. Los valores de estos atributos se recuperan por cada combinación de tuplas seleccionada.

Antes de entrar a comentar la sintaxis formal de los cálculos relacionales de tupla, consideremos otra consulta.

Consulta 0. Recuperar la fecha de nacimiento y la dirección del empleado (o empleados) cuyo nombre sea José Pérez Pérez.

C0: {*t*.FechaNac, *t*.Dirección | EMPLEADO(*t*) **AND** *t*.Nombre='José'
AND *t*.Apellido1='Pérez' **AND** *t*.Apellido2='Pérez'}

En los cálculos relacionales de tupla, primero se especifican los atributos que queremos (*t*.FechaNac y *t*.Dirección) de cada tupla *t* seleccionada. A continuación, especificamos la condición de selección seguida de una barra vertical (|). En resumen, esto significa que *t* es una tupla de la relación EMPLEADO cuyos valores de los atributos Nombre, Apellido1 y Apellido2 son, respectivamente, 'José', 'Pérez' y 'Pérez'.

6.6.2 Expresiones y fórmulas en los cálculos relacionales de tupla

Una expresión genérica de cálculo relacional de tupla tiene esta forma:

$$\{t_1.A_j, t_2.A_k, \ldots, t_n.A_m \mid \text{COND}(t_1, t_2, \ldots, t_n, t_{n+1}, t_{n+2}, \ldots, t_{n+m})\}$$

donde $t_1, t_2, \ldots, t_n, t_{n+1}, \ldots, t_{n+m}$ son variables de tupla, cada A_i es un atributo de la relación a la que t_i engloba y COND es una condición o fórmula[12] de cálculo relacional de tupla. Una fórmula está compuesta por alguno de los siguientes elementos de cálculo más pequeños (llamados átomos):

1. Un átomo de la forma $R(t_i)$, donde R es un nombre de relación y t_i es una variable de tupla. Este átomo identifica el ámbito de la variable de tupla t_i como la relación cuyo nombre es R.
2. Un átomo de la forma $t_i.A$ **op** $t_j.B$, donde **op** es uno de los operadores de comparación del conjunto $\{=, <, \leq, >, \geq, \neq\}$, t_i y t_j son variables de tupla y A y B son atributos de las relaciones t_i y t_j a las que, respectivamente, engloban.
3. Un átomo de la forma $t_i.A$ **op** c o c **op** $t_j.B$, donde **op** es uno de los operadores de comparación del conjunto $\{=, <, \leq, >, \geq, \neq\}$, t_i y t_j son variables de tupla, A y B son atributos de las relaciones t_i y t_j a las que, respectivamente, engloban y c es una constante.

Cada uno de los átomos anteriores se evalúa como VERDADERO o FALSO para una combinación específica de tuplas; esto recibe el nombre de **valor de verdad (o de veracidad)** de un átomo. En general, una variable de tupla *t* abarca a todas las posibles tuplas *del universo*. Para los átomos de la forma $R(t)$, si *t* está asignada a una tupla que es *miembro de la relación R especificada*, ese átomo es VERDADERO; en cualquier otro caso es FALSO. En los átomos de tipo 2 y 3, si las variables de tupla están asignadas a tuplas en las que los valores de los atributos especificados de las mismas satisfacen la condición, entonces el átomo es VERDADERO.

Una **fórmula** (condición) está compuesta por uno o varios átomos conectados mediante los operadores lógicos **AND**, **OR** y **NOT** y definidas recursivamente del siguiente modo por las Reglas 1 y 2:

- *Regla 1*: Cada átomo es una fórmula.
- *Regla 2*: Si F_1 y F_2 son fórmulas, entonces también lo son (F_1 **AND** F_2), (F_1 **OR** F_2), **NOT** (F_1) y **NOT** (F_2). Los valores de veracidad de estas fórmulas se derivan de los obtenidos para F_1 y F_2 de la siguiente forma:
  - **a.** (F_1 **AND** F_2) es VERDADERO si F_1 y F_2 lo son; en cualquier otro caso, es FALSO.
  - **b.** (F_1 **OR** F_2) es FALSO si F_1 y F_2 lo son; en cualquier otro caso, es VERDADERO.
  - **c.** **NOT** (F_1) es VERDADERO si F_1 es FALSO; es FALSO si F_1 es VERDADERO.
  - **d.** **NOT** (F_2) es VERDADERO si F_2 es FALSO; es FALSO si F_2 es VERDADERO.

6.6.3 Los cuantificadores Existencial y Universal

Además de los ya comentados, existen otros dos símbolos especiales llamados **cuantificadores** que pueden aparecer en las fórmulas: el **universal** (∀) y el **existencial** (∃). Los valores de comprobación de las fórmulas

[12] Llamada también WFF (Fórmula bien formada, *Well-Formed Formula*) en lógica matemática.

con estos cuantificadores están descritos en las Reglas 3 y 4; sin embargo, primero tenemos que definir los conceptos de variables de tupla libre y de tupla acotada en una fórmula. De manera informal, una variable de tupla t es ligada si está cuantificada, lo que significa que aparece en una cláusula $(\exists t)$ o $(\forall t)$; en cualquier otro caso, es libre. Formalmente, definimos en una fórmula una variable de tupla como **libre** o **acotada** según las siguientes reglas:

- Una variable de tupla en una fórmula F que *es un átomo* es libre en F.
- Una variable de tupla t es libre o acotada en una fórmula construida mediante conexiones lógicas [(F_1 **AND** F_2), (F_1 **OR** F_2), **NOT**(F_1) y **NOT**(F_2)] dependiendo de su estado en F_1 o F_2. Observe que en una fórmula de la forma $F = (F_1$ **AND** $F_2)$ o $F = (F_1$ **OR** $F_2)$, una variable de tupla puede ser libre en F_1 y acotada en F_2, o viceversa; en este caso, una de ellas será acotada y la otra libre en F.
- Todas las ocurrencias *libres* de una variable de tupla t en F son **acotadas** en una fórmula F' de la forma $F' = (\exists\ t)(F)$ o $F' = (\forall\ t)(F)$. La variable de tupla es acotada al cuantificador especificado en F'. Por ejemplo, considere las siguientes fórmulas:

  F_1 : d.NombreDpto='Investigación'

  F_2 : $(\exists t)(d$.NúmeroDpto=t.Dno)

  F_3 : $(\forall d)(d$.DniDirector='333445555')

La variable de tupla d es libre tanto en F_1 como en F_2, mientras que es acotada respecto al cuantificador $(\forall)$ en F_3. t es acotada respecto al cuantificador $(\exists)$ en F_2.

Ahora estamos en condiciones de ofrecer las Reglas 3 y 4 para la definición de la fórmula que empezamos anteriormente:

- *Regla 3*: Si F es una fórmula, entonces $(\exists t)(F)$ también lo es, donde t es una variable de tupla. La fórmula $(\exists t)(F)$ es VERDADERO si F se evalúa como tal en *alguna* (al menos una) tupla asignada a las ocurrencias libres de t en F; en cualquier otro caso, $(\exists t)(F)$ es FALSO.
- *Regla 4*: Si F es una fórmula, entonces $(\forall t)(F)$ lo es también, donde t es una variable de tupla. La fórmula $(\forall t)(F)$ es VERDADERO si F se evalúa como tal para *cada tupla* asignada a las ocurrencias libres de t en F; en cualquier otro caso, $(\forall t)(F)$ es FALSO.

$\exists$ se dice que es un cuantificador existencial porque una fórmula $(\exists t)(F)$ es VERDADERO si *existe* alguna tupla que haga que F sea VERDADERO. Para el cuantificador universal, $(\forall t)(F)$ es VERDADERO si cada posible tupla que puede asignarse a las ocurrencias libres de t en F es sustituida por t, y F es VERDADERO para *cada una de estas sustituciones*. Recibe el nombre de universal o *para todos los cuantificadores* porque cada tupla *del universo de* tuplas debe hacer que F sea VERDADERO para que la fórmula cuantificada también lo sea.

6.6.4 Ejemplos de consultas utilizando el cuantificador existencial

Vamos a utilizar algunas de las consultas de la Sección 6.5 para mostrar la forma de especificarlas tanto a través del álgebra relacional como mediante el cálculo relacional. Tenga en cuenta que será más sencillo especificar alguna de ellas mediante álgebra relacional, mientras que otras lo serán a través del cálculo relacional, y viceversa.

Consulta 1. Liste el nombre y la dirección de todos los empleados que trabajan para el departamento 'Investigación'.

C1: {t.Nombre, t.Apellido1, t.Dirección | EMPLEADO(t) **AND** $(\exists d)$
(DEPARTAMENTO(d) **AND** d.NombreDpto='Investigación' **AND** d.NúmeroDpto=t.Dno)}

Las *únicas variables de tupla libres* en una expresión de cálculo relacional deben ser aquéllas que aparecen a la izquierda de la barra (|). En C1, t es la única variable de tupla libre; entonces es *acotada sucesivamente* para

cada tupla. Si una tupla *satisface las condiciones* especificadas en C1, se recuperan los atributos Nombre, Apellido1 y Dirección. Las condiciones EMPLEADO(t) y DEPARTAMENTO(d) especifican las relaciones de rango para t y d. La condición d.NombreDpto='Investigación' es una **condición de selección** y se corresponde con una operación SELECCIÓN del álgebra relacional, mientras que d.NúmeroDpto = t.Dno es una **condición de concatenación** y sirve para un propósito similar a CONCATENACIÓN (consulte la Sección 6.3).

Consulta 2. Por cada proyecto ubicado en 'Gijón', obtenga su número, el número del departamento que lo gestiona y los apellidos, la fecha de nacimiento y la dirección del director del mismo.

C2: { p.NumProyecto, p.NumDptoProyecto, m.Apellido1, m.FechaNac, m.Dirección | PROYECTO(p)
AND EMPLEADO(m) **AND** p.UbicacionProyecto='Gijón'
AND ((∃d)(DEPARTAMENTO(d)
AND p.NumDptoProyecto=d.NúmeroDpto **AND** d.DniDirector=m.Dni))}

En C2 existen dos variables de tupla libres, p y m. d es de tipo acotada al cuantificador existencial. La condición de la consulta se evalúa por cada combinación de tuplas asignada a p y m; y como expulsa todas las posibles combinaciones de tuplas en las que p y m son acotadas, sólo se seleccionan las combinaciones que satisfacen la condición. Distintas variables de tupla de una consulta pueden alcanzar la misma relación. Por ejemplo, para especificar C8 (por cada empleado, recuperar su nombre y primer apellido y los de su supervisor inmediato), indicamos dos variables de tupla, e y s, que trabajan sobre la relación EMPLEADO:

C8: {e.Nombre, e.Apellido1, s.Nombre, s.Apellido1 | EMPLEADO(e) **AND** EMPLEADO(s)
AND e.SuperDni=s.Dni}

Consulta 3. Enumere el nombre de todos los empleados que trabajan en *algún* proyecto controlado por el departamento 5. Esto es una variación de la consulta 3 de la Sección 6.5, donde hablábamos de *todos* los proyectos, y no de *alguno*. En este caso necesitamos dos condiciones de conexión y dos cuantificadores existenciales.

C3: {e.Apellido1, e.Nombre | EMPLEADO(e)
AND ((∃ x)(∃ w)(PROYECTO(x) **AND** TRABAJA_EN(w) **AND** x.NumDptoProyecto=5
AND w.DniEmpleado=e.Dni **AND** x.NumProyecto=w.NumProy))}

Consulta 4. Obtenga una lista de los números de proyecto que impliquen a cualquier empleado cuyo primer apellido sea 'Pérez', independientemente de que sean trabajadores o directores del departamento que gestiona dicho proyecto.

C4: {p.NumProyecto | PROYECTO(p) **AND** (((∃ e)(∃ w)(EMPLEADO(e)
AND TRABAJA_EN(w) **AND** w.NumProy=p.NumProyecto
AND e.Apellido1='Pérez' **AND** e.Dni=w.DniEmpleado))
OR
((∃ m)(∃ d)(EMPLEADO(m) **AND** DEPARTAMENTO(d)
AND p.NumDptoProyecto=d.NúmeroDpto **AND** d.DniDirector=m.Dni
AND m.Apellido1='Pérez'))))}

Compare esto con la consulta en versión álgebra relacional de la Sección 6.5. En este caso, la operación UNIÓN puede sustituirse por un OR en los cálculos relacionales. En la sección siguiente trataremos las relaciones existentes entre los cuantificadores universal y existencial y la forma de convertir uno en otro.

6.6.5 Notación para consultas gráficas

En esta sección vamos a describir una notación que se ha propuesto para representar internamente consultas de cálculos relaciones. La representación más neutral de una consulta recibe el nombre de **gráfico de consulta**. La Figura 6.13 muestra el gráfico de consulta para C2. Las relaciones en la consulta están representadas por **nodos de relación**, los cuales aparecen como círculos sencillos. Las constantes, que se encuentran habitualmente en las condiciones de selección de la consulta, están representadas por **nodos constantes** que tienen la forma de círculos dobles u óvalos. Las condiciones de selección y conexión están representados por los **bordes** del gráfico (véase la Figura 6.13). Por último, los atributos a recuperar de cada relación aparecen entre corchetes sobre cada una de ellas.

La representación gráfica de una consulta no incluye el orden en el que se deben ejecutar las operaciones. Sólo existe una única correspondencia gráfica con cada consulta. Aunque algunas técnicas de optimización estaban basadas en este método, es preferible el uso de árboles de consulta porque, en la práctica, el optimizador de consultas necesita ver el orden de las operaciones para ejecutar la consulta, lo que no es posible en los gráficos de consulta.

6.6.6 Transformación de los cuantificadores universal y existencial

Ahora vamos a ver algunas transformaciones muy conocidas de la lógica matemática que están relacionadas con los cuantificadores universal y existencial. Es posible transformar uno en otro para obtener una expresión equivalente. De manera informal, una transformación general puede describirse como sigue: convertir un tipo de cuantificador en el otro con la negación (precedido de **NOT**); **AND** y **OR** se sustituyen uno por el otro; una fórmula negada se transforma en no-negada, y una fórmula no-negada se transforma en negada. Algunos casos especiales de esta transformación pueden declararse del siguiente modo, donde el símbolo ≡ significa "**equivalente a**":

$(\forall x)\ (P(x)) \equiv$ **NOT** $(\exists x)$ **(NOT** $(P(x)))$

$(\exists x)\ (P(x)) \equiv$ **NOT** $(\forall x)$ **(NOT** $(P(x)))$

$(\forall x)\ (P(x))$ **AND** $Q(x)) \equiv$ **NOT** $(\exists x)$ **(NOT** $(P(x))$ **OR NOT** $(Q(x)))$

$(\forall x)\ (P(x))$ **OR** $Q(x)) \equiv$ **NOT** $(\exists x)$ **(NOT** $(P(x))$ **AND NOT** $(Q(x)))$

$(\exists x)\ (P(x))$ **OR** $Q(x)) \equiv$ **NOT** $(\forall x)$ **(NOT** $(P(x))$ **AND NOT** $(Q(x)))$

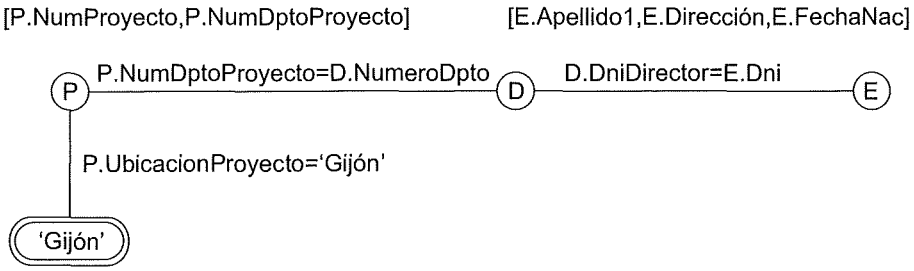
$(\exists x)\ (P(x))$ **AND** $Q(x)) \equiv$ **NOT** $(\forall x)$ **(NOT** $(P(x))$ **OR NOT** $(Q(x)))$

Observe también que lo siguiente es VERDADERO, donde el símbolo ⇒ significa "**implica**":

$(\forall x)\ (P(x)) \Rightarrow (\exists x)\ (P(x))$

NOT $(\exists X)\ (P(x)) \Rightarrow$ **NOT** $(\forall x)\ (P(x))$

Figura 6.13. Gráfico de consulta para C2.



6.6.7 Uso del cuantificador universal

Siempre que usemos un cuantificador universal, es bastante juicioso seguir ciertas reglas que garanticen que nuestra expresión tenga sentido. Trataremos estas reglas respecto a la C3.

> **Consulta 3.** Enumere los nombres de los empleados que trabajan en *todos* los proyectos controlados por el departamento 5. Una manera de especificar esta consulta es mediante el cuantificador universal:
>
> **C3:** {*e*.Apellido1, *e*.Nombre | EMPLEADO(*e*) **AND** ((**∀** *x*)(**NOT**(PROYECTO(*x*))
> **OR NOT** (*x*.NumDptoProyecto=5) **OR** ((**∃** *w*)(TRABAJA_EN(*w*) **AND** *w*.DniEmpleado= *e*.Dni
> **AND** *x*.NumProyecto=*w*.NumProy)))))}

Podemos dividir la C3 anterior en sus componentes básicos:

> **C3:** {*e*.Apellido1, *e*.Nombre | EMPLEADO(*e*) **AND** *F* '}
> *F* '= ((**∀** *x*)(**NOT**(PROYECTO(*x*)) **OR** F_1))
> F_1 = **NOT**(*x*.NumDptoProyecto=5) **OR** F_2
> F_2 = ((**∃** *w*)(TRABAJA_EN(*w*) **AND** *w*.DniEmpleado= *e*.Dni
> **AND** *x*.NumProyecto= *w*.NumProy))

Queremos asegurarnos de que un empleado *e* seleccionado trabaja en *todos los proyectos* controlados por el departamento 5, pero la definición de cuantificador universal dice que para que la fórmula cuantificada sea VERDADERA, la fórmula interna debe serlo también para *todas las tuplas del universo*. El truco consiste en excluir de la cuantificación universal aquellas tuplas en las que no estamos interesados, haciendo que la condición sea VERDADERA *para todas esas tuplas*. Esto es necesario porque una variable de tupla cuantificada universalmente, como lo es *x* en C3, debe evaluarse como VERDADERO *para cada posible tupla* asignada a ella de modo que la fórmula cuantificada también lo sea. Las primeras tuplas a excluir (haciendo que se evalúen automáticamente como VERDADERO) son aquéllas que no están en la relación *R*. En C3, el uso de la expresión **NOT**(PROYECTO(*x*)) dentro de la fórmula cuantificada universalmente evalúa como VERDADERO todas las tuplas *x* que no están en la relación PROYECTO. A continuación, excluimos de la propia *R* las tuplas que no nos interesan. En C3, la expresión **NOT**(*x*.NumDptoProyecto=5) evalúa como VERDADERO todas las tuplas *x* que están en la relación PROYECTO pero que no están controladas por el departamento 5. Para terminar, especificamos una condición F_2 que debe abarcar el resto de tuplas de *R*. Por consiguiente, podemos explicar C3 del siguiente modo:

1. Para que la fórmula *F*' = (**∀***x*)(*F*) sea VERDADERA, la fórmula *F* debe serlo también *para todas las tuplas del universo que puedan asignarse a x*. Sin embargo, en C3 sólo nos interesa que *F* sea VERDADERO para todas las tuplas de la relación PROYECTO que están controladas por el departamento 5. Por tanto, la fórmula *F* tiene la forma (**NOT**(PROYECTO(*x*)) **OR** F_1). La condición '**NOT**(PROYECTO(*x*)) **OR** . . .' es VERDADERA para todas las tuplas *que no estén en la relación PROYECTO* y tiene el efecto de eliminar esas tuplas del valor de veracidad de F_1. Para cada tupla de la relación PROYECTO, F_1 debe ser VERDADERO si *F*' lo es.

2. Usando la misma línea de razonamiento, no queremos considerar las tuplas de PROYECTO que no están controladas por el departamento número 5, ya que sólo nos interesan aquéllas cuyo NumDptoProyecto=5. Por consiguiente, podemos escribir:

   IF (*x*.NumDptoProyecto=5) **THEN** F_2

   lo que es equivalente a:

   (**NOT** (*x*.NumDptoProyecto=5) **OR** F_2)

3. La fórmula F_1, por tanto, tiene la forma **NOT**(x.NumDptoProyecto=5) **OR** F_2. En el contexto de C3, esto significa que, para una tupla x en la relación PROYECTO, o su NumDptoProyecto $\neq$ 5 o debe satisfacer F_2.
4. Por último, F_2 aporta la condición que queremos mantener para una tupla EMPLEADO seleccionada: que el empleado trabaje en *cada tupla* PROYECTO *que aún no se haya excluido.* Este tipo de tuplas de empleado son las que la consulta selecciona.

Dicho de forma sencilla, C3 podría enunciarse del siguiente modo a la hora de seleccionar una tupla EMPLEADO e: por cada tupla x en la relación PROYECTO cuyo x.NumDptoProyecto=5, debe existir otra tupla w en TRABAJA_EN en la que se cumpla que w.DniEmpleado=e.Dni y w.NumProy=x.NumProyecto. Esto es equivalente a decir que el EMPLEADO e trabaja en cada PROYECTO x del DEPARTAMENTO número 5 (¡GUAU!).

Usando la transformación general de cuantificadores de universal a existencial ofrecida en la Sección 6.6.6, podemos redefinir la consulta de C3 de la forma indicada en C3A:

C3A: {e.Apellido1, e.Nombre | EMPLEADO(e) **AND** (**NOT** (∃ x) (PROYECTO(x)
AND (x.NumDptoProyecto=5) **AND** (**NOT** (∃ w)(TRABAJA_EN(w)
AND w.DniEmpleado=e.Dni **AND** x.NumProyecto=w.NumProy)))))}

Ahora, mostraremos algunos ejemplos adicionales de consultas que usan cuantificadores.

Consulta 6. Liste los nombres de los empleados que no tienen subordinados.

C6: {e.Nombre, e.Apellido1 | EMPLEADO(e) **AND** (**NOT** (∃ d)(SUBORDINADO(d)
AND e.Dni=d.DniEmpleado))}

Usando la regla de transformación general, podemos redefinir C6 del siguiente modo:

C6A: {e.Nombre, e.Apellido1 | EMPLEADO(e) **AND** ((∀ d)(**NOT**(SUBORDINADO(d))
OR NOT(e.Dni=d.DniEmpleado)))}

Consulta 7. Liste los nombres de los directores que tienen un subordinado como mínimo.

C7: {e.Nombre, e.Apellido1 | EMPLEADO(e) **AND** ((∃ d)(∃ ρ)(DEPARTAMENTO(d)
AND SUBORDINADO(ρ) **AND** e.Dni=d.DniDirector **AND** ρ.DniEmpleado=e.Dni))}

Esta consulta se manipula interpretando *directores que tienen al menos un subordinado* como *directores para los que existe algún subordinado.*

6.6.8 Expresiones seguras

Siempre que usemos cuantificadores universales, existenciales o negaciones de predicado en una expresión de cálculo, debemos asegurarnos de que la expresión resultante tenga sentido. Una **expresión segura** en cálculo relacional es aquélla en la que está garantizada la recuperación de un *número finito de tuplas* como resultado; en cualquier otro caso, se dice que la expresión es **insegura**.

Por ejemplo, la expresión:

{t | **NOT** (EMPLEADO(t))}

es *insegura* porque recupera todas las tuplas del universo que *no* son tuplas EMPLEADO, las cuales son infinitamente numerosas. Si seguimos las reglas dadas anteriormente para C3, obtendremos una expresión segura usando cuantificadores universales. Podemos definir de un modo más preciso las expresiones seguras introduciendo el concepto de *dominio de una expresión de cálculo relacional de tupla:* es el conjunto de todos los

valores que podrían aparecer como constantes en la expresión o existir en cualquier tupla de las relaciones referenciadas en esa expresión. El dominio de {*t* | **NOT**(EMPLEADO(*t*))} es el conjunto de todos los valores de atributo que aparecen en alguna tupla de la relación EMPLEADO (para cualquier atributo). El dominio de la expresión C3A podría incluir todos los valores que aparecen en EMPLEADO, PROYECTO y TRABAJA_EN (unidas a través del valor 5 que aparece en la propia consulta).

Se dice que una expresión es **segura** si todos los valores de su resultado son del dominio de la misma. Observe que el resultado de {*t* | **NOT**(EMPLEADO(*t*))} es inseguro ya que, en general, incluirá tuplas (y, por tanto, valores) externas a la relación EMPLEADO; valores de este tipo no pertenecen al dominio de la expresión. El resto de nuestros ejemplos son expresiones seguras.

6.7 Los cálculos relacionales de dominio

Existe otro tipo de cálculo relacional llamado de dominio, o simplemente **cálculo de dominio**. Mientras que SQL (consulte el Capítulo 8), un lenguaje basado en los cálculos relacionales de tuplas, estaba en fase de desarrollo en los laboratorios de IBM Research en San José (California), QBE, otro lenguaje que estaba relacionado con los cálculos de dominio, se estaba preparando casi a la vez en el Centro de Investigación T. J. Watson de IBM en Yorktown Heights (Nueva York). Las especificaciones formales del cálculo de dominio fueron propuestas después del desarrollo del sistema QBE.

Los cálculos de dominio difieren de los de tupla en el *tipo de variables* usadas en las fórmulas: en lugar de que éstas operen sobre tuplas, lo hacen sobre valores individuales de los dominios de atributos. Para componer una relación de grado n para el resultado de una consulta, debe disponer de n de estas **variables de dominio**: una por cada atributo. Una expresión de cálculo de dominio tiene la siguiente forma:

$$\{x_1, x_2, \ldots, x_n \mid \text{CONDICIÓN}(x_1, x_2, \ldots, x_n, x_{n+1}, x_{n+2}, \ldots, x_{n+m})\}$$

donde $x_1, x_2, \ldots, x_n, x_{n+1}, x_{n+2}, \ldots, x_{n+m}$ son variables de dominio que operan sobre los dominios (de atributos), y CONDICIÓN es una **condición** o **fórmula del cálculo relacional de dominio**.

Una fórmula está compuesta por **átomos**, los cuales son algo diferentes de los del cálculo de tupla, y pueden ser uno de los siguientes:

1. Un átomo de la forma $R(x_1, x_2, \ldots, x_j)$, donde R es el nombre de una relación de grado j y cada x_i, $1 \leq i \leq j$, es una variable de dominio. Este átomo afirma que una lista de valores de $<x_1, x_2, \ldots, x_j>$ debe ser una tupla en la relación cuyo nombre es R, donde x_i es el $i$enésimo valor de atributo de esa tupla. Para hacer más concisa una expresión de cálculo de dominio, podemos *quitar las comas* de una lista de variables; de este modo, podemos escribir:

   $$\{x_1, x_2, \ldots, x_n \mid R(x_1\ x_2\ x_3)\ \textbf{AND}\ \ldots\}$$

   en lugar de:

   $$\{x_1, x_2, \ldots, x_n \mid R(x_1, x_2, x_3)\ \textbf{AND}\ \ldots\}$$

2. Un átomo de la forma x_i **op** x_j, donde **op** es uno de los operadores de comparación del conjunto $\{=, <, \leq, >, \geq, \neq\}$, y x_i y x_j son variables de dominio.
3. Un átomo de la forma x_i **op** c o c **op** x_j, donde **op** es uno de los operadores de comparación del conjunto $\{=, <, \leq, >, \geq, \neq\}$, x_i y x_j son variables de dominio y c es una constante.

Como ocurre en los cálculos de tupla, los átomos se evalúan como VERDADERO o FALSO para un conjunto específico de valores, llamados **valores de verdad** de esos átomos. En el caso 1, si las variables de dominio tienen asignados valores correspondientes a una tupla de la relación R especificada, entonces el átomo es VERDADERO. En los casos 2 y 3, si las variables de dominio están asignadas a valores que satisfacen la condición, entonces el átomo es VERDADERO.

De forma muy parecida a como ocurre con los cálculos relaciones de tupla, las fórmulas están compuestas de átomos, variables y cuantificadores, por lo que no vamos a repetir aquí de nuevo las especificaciones de las mismas.

A continuación se detallan algunos ejemplos de consultas especificadas mediante cálculos de dominio. Usaremos letras minúsculas (l, m, n, . . . , x, y, z) para especificar las variables de dominio.

Consulta 0. Enumere la fecha de nacimiento y la dirección de los empleados cuyo nombre sea 'José Pérez Pérez'.

C0: $\{uv \mid (\exists q)\ (\exists r)\ (\exists s)\ (\exists t)\ (\exists w)\ (\exists x)\ (\exists y)\ (\exists z)$
(EMPLEADO($qrstuvwxyz$) **AND** q='José' **AND** r='Pérez' **AND** s='Pérez')$\}$

Necesitamos diez variables para la relación EMPLEADO, a fin de ordenar el dominio de cada atributo. De estas diez variables q, r, s, . . . , z, sólo u y v son libres. Primero especificamos los *atributos solicitados*, FechaNac y Dirección, para las variables de dominio u (FECHANAC) y v (DIRECCIÓN). Después, indicamos la condición de selección de una tupla seguida por la barra(|): a saber, que la secuencia de valores asignados a las variables $qrstuvwxyz$ son una tupla de la relación EMPLEADO y que los valores para q (Nombre), r (Apellido1) y s (Apellido2) son 'José', 'Pérez' y 'Pérez', respectivamente. Por conveniencia, en el resto de nuestros ejemplos sólo cuantificaremos aquellas variables que *aparezcan en una condición* (q, r y s en C0).[13]

Una notación alternativa, usada en QBE, para escribir esta consulta es asignar las constantes 'José', 'Pérez' y 'Pérez' directamente como se muestra en C0A. Aquí, todas las variables que no aparecen a la izquierda de la barra son cuantificadas existencial e implícitamente:[14]

C0A: $\{uv \mid$ EMPLEADO('José','Pérez','Pérez',t,u,v,w,x,y,z) $\}$

Consulta 1. Recupere el nombre y la dirección de todos los empleados que trabajan en el departamento de 'Investigación'.

C1: $\{qsv \mid (\exists z)\ (\exists l)\ (\exists m)$ (EMPLEADO($qrstuvwxyz$) **AND**
DEPARTAMENTO($lmno$) **AND** l='Investigación' **AND** m=z)$\}$

Una condición que relaciona dos variables de dominio que trabajan sobre los atributos de dos relaciones, como ocurre con $m = z$ en C1, es una **condición de conexión**, mientras que la que relaciona una variable de dominio con una constante, como en el caso de l = 'Investigación', es una **condición de selección**.

Consulta 2. Por cada proyecto localizado en 'Gijón', liste su número, el número de departamento que lo gestiona y el nombre, los apellidos y la fecha de nacimiento de su director.

C2: $\{iksuv \mid (\exists j)(\ \exists m)(\ \exists n)(\ \exists t)$(PROYECTO($hijk$)
AND EMPLEADO($qrstuvwxyz$) **AND** DEPARTAMENTO($lmno$)
AND k=m **AND** n=t **AND** j='Gijón')$\}$

Consulta 6. Liste los nombres de los empleados que no tienen subordinados.

C6: $\{qs \mid (\exists t)$(EMPLEADO($qrstuvwxyz$)
AND (**NOT**($\exists l$)(SUBORDINADO($lmnop$) **AND** t=l)))$\}$

---

[13] Tenga en cuenta que la notación que usamos para cuantificar sólo las variables de dominio utilizadas actualmente en las condiciones, y para mostrar un predicado como EMPLEADO($qrstuvwxyz$) sin separar las variables de dominio con comas, es un método abreviado que usamos por conveniencia; no es la notación formal correcta.

[14] De nuevo, ésta no es una notación formalmente correcta.

C6 puede exponerse de otro modo usando cuantificadores universales en lugar de los existenciales, como puede verse en C6A:

C6A: {*qs* | (∃*t*)(EMPLEADO(*qrstuvwxyz*)
AND ((∀*l*)(**NOT**(SUBORDINADO(*lmnop*)) **OR NOT**(*t*=*l*))))}

Consulta 7. Obtenga los nombres de los directores que tengan, al menos, un subordinado.

Q7: {*sq* | (∃*t*)(∃*j*)(∃*l*)(EMPLEADO(*qrstuvwxyz*) **AND** DEPARTAMENTO(*hijk*)
AND SUBORDINADO(*lmnop*) **AND** *t*=*j* **AND** *l*=*t*)}

Como ya mencionamos anteriormente, es posible redefinir cualquier consulta expresada mediante álgebra relacional con cálculos relaciones de dominio o tupla. Además, cualquier *expresión segura* de estos cálculos puede expresarse también mediante álgebra relacional.

El lenguaje QBE se basaba en los cálculos relacionales de dominio, aunque esto se realizó más tarde, después de que los cálculos de dominio se hubieron formalizado. QBE fue uno de los primeros lenguajes de consulta gráficos con una sintaxis mínima desarrollados para sistemas de bases de datos. Fue elaborado por IBM Research y forma parte de la opción de interfaz QMF (*Query Management Facility*) de DB2. Ha sido imitado por otros productos comerciales. Debido a su importante posición en el campo de los lenguajes relacionales, hemos incluido una panorámica de QBE en el Apéndice D.

6.8 Resumen

En este Capítulo hemos presentado dos lenguajes formales del modelo relacional de datos. Se usan para manipular relaciones y producir nuevas relaciones como respuestas a consultas. Tratamos el álgebra relacional y sus operaciones, las cuales se utilizan para especificar la secuencia de operaciones para definir una consulta. A continuación, mostramos dos tipos de cálculos relaciones: los de tupla y los de dominio; son declarativos porque especifican el resultado de una consulta sin especificar el modo de producir dicho resultado.
En las Secciones de la 6.1 a la 6.3 presentamos las operaciones básicas del álgebra relacional e ilustramos los tipos de consultas para las que se usan cada una de ellas. En primer lugar, abordamos los operadores relacionales unarios SELECCIÓN y PROYECCIÓN, así como la operación RENOMBRAR. Después nos centramos en el conjunto de operaciones binarias en las que es necesario que las relaciones sean de tipo compatible con la unión: UNIÓN, INTERSECCIÓN y DIFERENCIA DE CONJUNTOS. El PRODUCTO CARTESIANO es una operación de conjuntos que puede usarse para combinar tuplas de dos relaciones que produce todas las combinaciones posibles.

Aunque en la práctica es muy raro usarlo, mostramos cómo puede usarse el PRODUCTO CARTESIANO seguido de una SELECCIÓN para definir tuplas coincidentes de dos relaciones e inducir una operación CONCATENACIÓN. Presentamos varios tipos de operaciones CONCATENACIÓN llamadas AGRUPAMIENTO, EQUIJOIN y CONCATENACIÓN NATURAL. Los árboles de consultas se definieron como una representación interna de las consultas de álgebra relacional.

Tratamos algunos tipos de consultas importantes que *no* pueden obtenerse con el álgebra relacional básico, pero que son importantes en ciertas situaciones prácticas. Presentamos la PROYECCIÓN GENERALIZADA para usar funciones de atributos en la lista de proyección y la operación FUNCIÓN AGREGADA para tratar con tipos de peticiones agregadas. Comentamos las consultas recursivas, para las que no existe soporte directo en el álgebra. A continuación nos centramos en la CONCATENACIÓN EXTERNA y la UNIÓN EXTERNA, las cuales amplían la CONCATENACIÓN y la UNIÓN y permiten preservar toda la información de las relaciones originales en el resultado.

Las dos últimas secciones tratan acerca de los conceptos básicos que se esconden tras los cálculos relacionales, los cuales están basados en la rama de la lógica matemática llamada cálculo de predicado. Existen dos tipos de cálculos relacionales: (1) el de tupla, que utiliza variables de tupla que engloban las tuplas (filas) de las relaciones, y (2) los de dominio, que emplean variables de dominio que se centran en los dominios (columnas de las relaciones). En los cálculos relacionales, una consulta se especifica en una única sentencia declarativa sin especificar ningún orden o método para la recuperación del resultado de la consulta. Por consiguiente, los cálculos relacionales se consideran a menudo como un lenguaje de mayor nivel que el álgebra relacional porque una expresión de cálculo relacional declara *qué* queremos recuperar sin preocuparse del *cómo* hacerlo.

Explicamos la sintaxis de las consultas de cálculo relacional usando tanto variables de tupla como de dominio. Presentamos los gráficos de consulta como una representación interna de las consultas en los cálculos relacionales, y tratamos también los cuantificadores existencial (∃) y universal (∀). Mostramos que las variables de cálculo relacional están ligadas a estos cuantificadores. Describimos con detalle la forma de escribir la cuantificación universal, y comentamos el problema de especificar consultas seguras cuyo resultado es finito. También vimos las reglas que rigen la transformación de un cuantificador en otro. Son estos cuantificadores los que dan potencia expresiva a los cálculos relacionales, haciéndolos equivalentes al álgebra relacional. No existen en los cálculos racionales básicos una analogía para las funciones de agrupamiento y agregación, aunque algunas extensiones las han sugerido.

Preguntas de repaso

6.1. Enumere las operaciones del álgebra relacional y el objetivo de cada una de ellas.

6.2. ¿Qué es la compatibilidad de unión? ¿Por qué es necesario que las operaciones UNIÓN, INTERSECCIÓN y DIFERENCIA requieran que las relaciones en las que se aplican sean compatibles con la unión?

6.3. Comente algunos de los tipos de consultas en los que sea necesario renombrar los atributos para evitar la ambigüedad en la consulta.

6.4. Comente los distintos tipos de operaciones de *concatenación interna*. ¿Por qué es necesaria una ASOCIACIÓN?

6.5. ¿Qué papel juega el concepto de *foreign key* a la hora de especificar los tipos más comunes de operaciones de concatenación?

6.6. ¿Qué es la operación FUNCIÓN? ¿Para qué se utiliza?

6.7. ¿En qué difieren las operaciones CONCATENACIÓN EXTERNA e INTERNA? ¿Y la UNIÓN EXTERNA de la UNIÓN?

6.8. ¿En qué difiere el cálculo relacional del álgebra relacional, y en qué se parecen?

6.9. ¿En qué se diferencian los cálculos relacionales de tupla y de dominio?

6.10. Comente el significado del cuantificador existencial (∃) y del universal (∀).

6.11. Defina los siguientes términos respecto a los cálculos de tupla: *variable de tupla*, *relación de rango*, *átomo*, *fórmula* y *expresión*.

6.12. Defina los siguientes términos respecto a los cálculos de dominio: *variable de dominio*, *relación de rango*, *átomo*, *fórmula* y *expresión*.

6.13. ¿Cuál es el significado de una *expresión segura* en los cálculos relacionales?

6.14. ¿Cuándo se dice que un lenguaje de consulta es relacionalmente completo?

Ejercicios

6.15. Muestre el resultado de las consultas de ejemplo de la Sección 6.5 aplicadas a la base de datos de la Figura 5.6.

6.16. Especifique las siguientes consultas del esquema de base de datos EMPRESA de la Figura 5.5 usando los operadores relacionales comentados en este capítulo. Muestre también el resultado de cada consulta aplicada a la base de datos de la Figura 5.6.

a. Recupere los nombres de todos los empleados del departamento 5 que trabajan más de 10 horas por semana en el proyecto ProductoX.

b. Enumere los nombres de todos los empleados que tienen un subordinado con el mismo nombre.

c. Localice los nombres de todos los empleados que están supervisados directamente por 'Alberto Campos'.

d. Por cada proyecto, enumere su nombre y el número total de horas semanales (de todos los empleados) dedicadas al mismo.

e. Recupere los nombres de todos los empleados que trabajan en cada proyecto.

f. Recupere los nombres de todos los empleados que no trabajan en ningún proyecto.

g. Por cada departamento, recupere su nombre y el salario medio de todos los empleados que trabajan en él.

h. Recupere el salario medio de todas las empleadas.

i. Busque los nombres y las direcciones de todos los empleados que trabajan en, al menos, un proyecto localizado en Madrid, pero cuyo departamento no lo esté.

j. Liste los apellidos de todos los directores de departamento que no tengan subordinados.

6.17. Considere el esquema de la base de datos LINEA_AEREA de la Figura 5.8, el cual se describió en el Ejercicio 5.12. Especifique las siguientes consultas en forma de álgebra relacional:

a. Por cada vuelo, liste el número del mismo, el aeropuerto de partida del primer plan de vuelo y el aeropuerto de llegada del último.

b. Enumere los números de vuelo y los días de la semana de todos los vuelos, o planes de vuelo, que salgan del Aeropuerto internacional de Houston (código de aeropuerto 'IAH') y lleguen al Aeropuerto internacional de Los Ángeles (código de aeropuerto 'LAX').

c. Obtenga el número de vuelo, el código del aeropuerto de salida, la hora programada para el despegue, el código del aeropuerto de llegada, la hora programada para el aterrizaje y los días de la semana de todos los vuelos, o planes de vuelo, que salgan de algún aeropuerto de la ciudad de Houston y lleguen a alguno de Los Ángeles.

d. Liste todas las tarifas del número de vuelo 'CO197'.

e. Recupere el número de asientos disponibles del vuelo 'CO197' el día '09-10-1999'.

6.18. Considere el esquema de base de datos BIBLIOTECA de la Figura 6.14, la cual se emplea para controlar los libros, los solicitantes de los mismos y los préstamos. Las restricciones de integridad referencial aparecen como arcos dirigidos en la Figura 6.14, del mismo modo que en la notación de la Figura 5.7. Escriba expresiones relacionales para las siguientes consultas:

a. ¿Cuántas copias del libro *La cabaña del tío Tom* son propiedad de la delegación cuyo nombre es 'Sharpstown'?

b. ¿Cuántas copias del libro *La cabaña del tío Tom* pertenecen a cada delegación de la biblioteca?

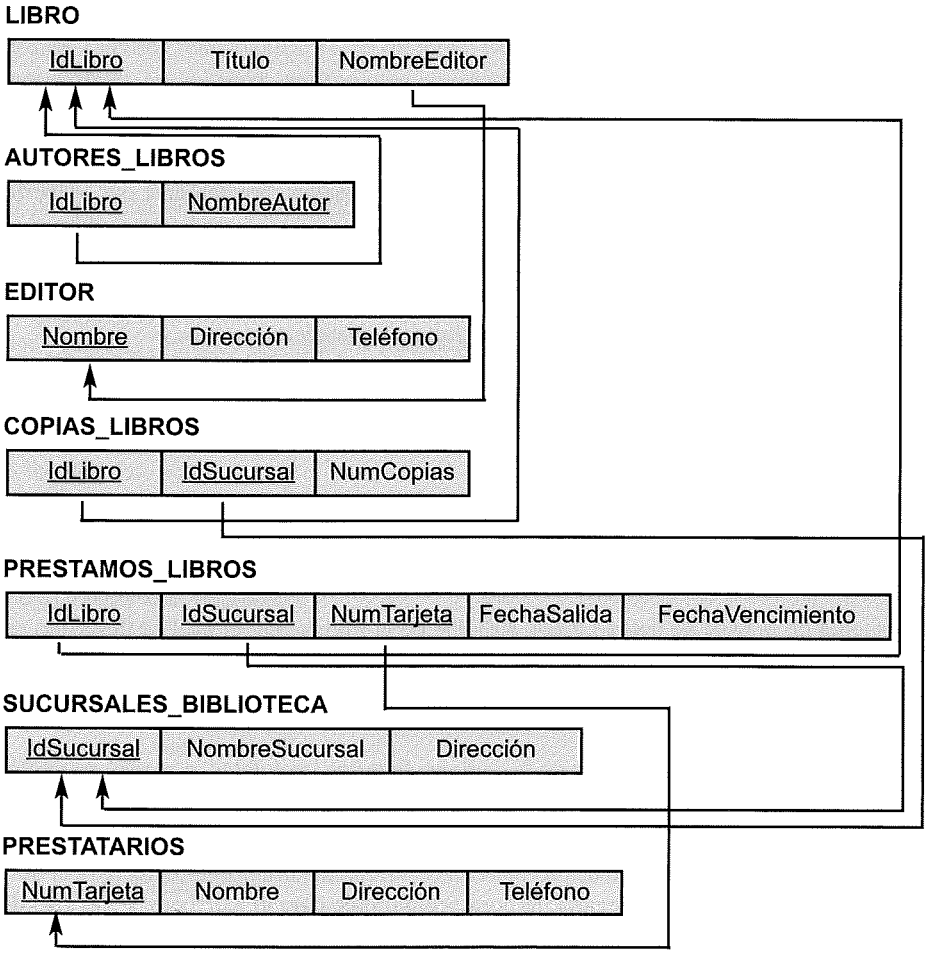
c. Recupere los nombres de todos los prestatarios que no tienen ningún libro prestado.

d. Por cada libro que está prestado fuera de la delegación de Sharpstown y cuya FechaPrestamo sea hoy, recuperar el título de la obra y el nombre y la dirección del prestatario.

e. Por cada delegación, recuperar el nombre de la misma y el número total de libros prestados fuera de esa delegación.

f. Recuperar los nombres, direcciones y número de libros prestados de todos los prestatarios que tengan más de cinco obras prestadas.

Figura 6.14. Esquema de base de datos relacional para una base de datos BIBLIOTECA.



g. Por cada libro escrito (o coescrito) por Stephen King, recuperar el título y el número de copias de la delegación cuyo nombre es Central.

6.19. Especificar las siguientes consultas en álgebra relacional para el esquema de base de datos ofrecido en el Ejercicio 5.14:

a. Listar el NumeroPedido y la FechaSalida de todos los pedidos procedentes del NumeroAlmacen W2.

b. Listar la información de ALMACEN que abastece al CLIENTE José López. En el listado debe aparecer el NumeroEntrega y el NumeroAlmacen.

c. Obtener un listado con los datos relativos al NombreCliente, NumeroDePedidios y MediaPedidos, donde la columna central es el número total de pedidos del cliente y la última es el importe medio de ese cliente.

d. Enumere los pedidos que no fueron expedidos a los 30 días de su petición.

e. Recupere el NumeroPedido de todos los pedidos que fueron expedidos de *todos* los almacenes que la empresa tiene en Nueva York.

6.20. Especifique las siguientes consultas en álgebra relacional del esquema de base de datos del Ejercicio 5.15:

a. Obtenga todos los detalles de los viajes que sobrepasen los 2.000 euros de gastos.

b. Imprima el Dni del vendedor que realiza viajes a Honolulu.

c. Imprima los gastos totales de viajes del vendedor con DNI='234567890'.

6.21. Especifique las siguientes consultas en álgebra relacional en la base de datos del Ejercicio 5.16:

a. Liste el número de cursos recibidos por todos los estudiantes llamados Juan Pérez en el verano de 1999 (esto es, trimestre=V99).

b. Obtener una lista de los libros de texto (incluyendo NumeroCurso, ISBNLibro, TituloLibro) de los cursos ofrecidos por el departamento 'CC' que han usando más de dos libros.

c. Muestre cualquier departamento que haya adoptado libros publicados por 'AWL Publishing'.

6.22. Considere las dos tablas *T*1 y *T*2 mostradas en la Figura 6.15. Muestre los resultados de las siguientes operaciones:

a. $T1 \bowtie_{T1.P = T2.A} T2$

b. $T1 \bowtie_{T1.Q = T2.B} T2$

c. $T1 \rhd\!\bowtie_{T1.P = T2.A} T2$

d. $T1 \bowtie\!\lhd_{T1.Q = T2.B} T2$

e. $T1 \cup T2$

f. $T1 \bowtie_{(T1.P = T2.A \text{ AND } T1.R = T2.C)} T2$

6.23. Especifique las siguientes consultas en álgebra relacional sobre la base de datos del Ejercicio 5.17:

a. Para la vendedora 'Manuela Pris', obtenga la siguiente información de todos los coches que vendió: NumeroBastidor, Fabricante y PrecioVenta.

b. Liste el NumeroBastidor y el Modelo de los coches que no tengan extras.

c. Considere una operación de CONCATENACIÓN_ NATURAL entre VENDEDOR y VENTA. ¿Cuál es el significado de una concatenación externa izquierda para estas tablas (no cambie el orden de las relaciones)? Explíquelo con un ejemplo.

d. Escriba una consulta en álgebra relacional que comporte una selección y un conjunto de operación y exprese con palabras lo que dicha consulta hace.

6.24. Especifique las consultas a, b, c, e, f, i y j del Ejercicio 6.16 en forma de cálculo relacional de dominio y de tupla.

6.25. Especifique las consultas a, b, c y d del Ejercicio 6.17 en forma de cálculo relacional de dominio y de tupla.

6.26. Especifique las consultas c, d y f del Ejercicio 6.18 en forma de cálculo relacional de dominio y de tupla.

6.27. En una consulta de cálculo relacional de tupla con *n* variables de tupla, ¿cuál sería el número mínimo de condiciones de conexión típico? ¿Por qué? ¿Cuál sería el efecto de tener un número menor de condiciones de conexión?

Figura 6.15. Estado de una base de datos para las relaciones *T*1 y *T*2.

TABLA T1

P	Q	R
10	a	5
15	b	8
25	a	6

TABLA T2

A	B	C
10	b	6
25	c	3
10	b	5

6.28. Reescriba las consultas de cálculo relacional de dominio que siguen a la C0 en la Sección 6.7 en el estilo abreviado mostrado en la C0A, donde el objetivo es minimizar el número de variables de dominio escribiendo constantes donde sea posible.

6.29. Considere esta consulta: recupere los Dni de los empleados que trabajan en, al menos, los mismos proyectos que los que tienen el Dni=123456789. Esto puede declararse como (**FORALL** x) (**IF** P **THEN** Q), donde:

x es una variable de tupla que abarca la relación PROYECTO.

$P \equiv$ EMPLEADO con Dni=123456789 trabaja en el PROYECTO x.

$Q \equiv$ EMPLEADO e trabaja en el PROYECTO x.

Exprese la consulta en forma de cálculos relacionales de tupla usando estas reglas:

(**FORALL** x)($P(x)$) ≡ **NOT**(∃x)(**NOT**($P(x)$)).

(**IF** P **THEN** Q) ≡ (**NOT**(P) **OR** Q).

6.30. Indique cómo podría especificar las siguientes operaciones de álgebra relacional en forma de cálculos relacionales de tupla y de dominio.

a. $\sigma_{A=C}(R(A, B, C))$

b. $\pi_{<A, B>}(R(A, B, C))$

c. $R(A, B, C) * S(C, D, E)$

d. $R(A, B, C) \cup S(A, B, C)$

e. $R(A, B, C) \cap S(A, B, C)$

f. $R(A, B, C) - S(A, B, C)$

g. $R(A, B, C) \times S(D, E, F)$

h. $R(A, B) \div S(A)$

6.31. Sugiera las extensiones necesarias de los cálculos relacionales que permitan expresar los siguientes tipos de operaciones que fueron tratados en la Sección 6.4: (a) funciones agregadas y de agrupamiento; (b) operaciones de CONCATENACIÓN EXTERNA; (c) consultas de finalización recursivas.

6.32. Una consulta anidada es una consulta dentro de otra. De forma más específica puede decirse que una consulta anidada es una consulta con paréntesis cuyo resultado puede usarse como valor en muchos otros lugares, como en lugar de una relación. Especifique las siguientes consultas de la base de datos de la Figura 5.5 usando el concepto de consulta anidada y los operadores relacionales estudiados en este capítulo. Muestre también el resultado de cada consulta aplicada a la base de datos de la Figura 5.6.

a. Liste los nombres de todos los empleados que trabajan en el departamento que cuenta con el empleado de mayor salario de todos los trabajadores.

b. Obtenga los nombres de todos los empleados cuyo supervisor de su supervisor tenga como Dni el valor '888665555'.

c. Recupere los nombres de todos los empleados que ganen, al menos, 10.000 euros más que el empleado con el sueldo más bajo de toda la compañía.

6.33. Indique si las siguientes conclusiones son verdaderas o falsas:

a. **NOT** ($P(x)$ **OR** $Q(x)$) → (**NOT** ($P(x)$) **AND** (**NOT** ($Q(x)$))

b. **NOT** (∃ x) ($P(x)$) → ∀x (**NOT** ($P(x)$)

c. (∃ x) ($P(x)$) → ∀x (($P(x)$)

Ejercicios de práctica

6.34. Especifique y ejecute las siguientes consultas en álgebra relacional usando el intérprete RA del esquema de base de datos relacional EMPRESA.

a. Liste los nombres de todos los empleados del departamento 5 que trabajen más de 10 horas a la semana en el proyecto ProductoX.

b. Obtenga los nombres de todos los empleados que tengan un subordinado con su mismo nombre.

c. Recupere los nombres de los empleados que están supervisados directamente por Alberto Campos.

d. Enumere los nombres de los empleados que trabajan en cada proyecto.

e. Muestre los nombres de todos los empleados que no trabajan en ningún proyecto.

f. Indique los nombres y las direcciones de los empleados que trabajan en, al menos, un proyecto localizado en Madrid, pero cuyo departamento no esté localizado en esa ciudad.

g. Muestre los nombres de los directores de departamento que no tengan subordinados.

6.35. Considere el siguiente esquema relacional PEDIDOS_CORREO que describe los datos de una compañía de envío de pedidos por correo.

REPUESTOS(NumeroRep, NombreRep, ACuenta, Precio, ONivel)

CLIENTES (NumeroCliente, Nombre, Direccion, CP, Telefono)

EMPLEADOS(NumeroEmpleado, NombreEmpleado, CP, Hdate)

CODIGOS_POSTALES(CP, Ciudad)

PEDIDOS(NumeroPedido, NumeroCliente, NumeroEmpleado, Recibido, Enviado)

DETALLE_PEDIDO(NumeroPedido, NumeroProyecto, Cantidad)

Los nombres de los atributos son auto explicativos: ACuenta mantiene la *cantidad entregada a cuenta*. Especifique y ejecute las siguientes consultas usando el intérprete RA del esquema PEDIDOS_CORREO.

a. Recupere los nombres de los repuestos que cuestan menos de 20 euros.

b. Recupere los nombres y las ciudades de los empleados que han efectuado pedidos de repuestos que cuestan más de 50 euros.

c. Recupere los números de cliente de aquéllos que vivan en el mismo código postal.

d. Recupere los nombres de los clientes que tengan pedidos de repuestos a representantes que vivan en Valencia.

e. Recupere los nombres de los clientes que hayan solicitado repuestos que cuesten menos de 20 euros.

f. Recupere los nombres de los clientes que no hayan hecho pedidos.

g. Recupere los nombres de los clientes que hayan hecho dos pedidos.

6.36. Considere el siguiente esquema relacional PLAN_ESTUDIOS que describe los datos del plan de estudios de un profesor concreto. (*Nota*. Los atributos A, B, C y D de CURSOS almacenan las abreviaturas de las notas).

CATALOGO(Cno, TituloCatalogo)

ESTUDIANTES(NumeroEstudiante, Nombre, Apellidos)

CURSOS(Term, NumSec, Cno, A, B, C, D)

MATRICULAS(NumeroEstudiante, Term, NumSec)

Especifique y ejecute las siguientes consultas usando el intérprete RA sobre el esquema PLAN_ESTUDIOS.

a. Recupere los nombres de los estudiantes matriculados en la clase Autómatas durante el tercer trimestre de 1996.

b. Recupere los valores de NumeroEstudiante de los matriculados en CSc226 y CSc227.

c. Recupere los valores de NumeroEstudiante de los matriculados en CSc226 o CSc227.

d. Recupere los nombres de los estudiantes que no están matriculados en ninguna clase.

e. Recupere los nombres de los estudiantes matriculados en todos los cursos de la tabla CATALOGO.

6.37. Considere una base de datos compuesta por las siguientes relaciones.

PROVEEDOR(NumeroProveedor, NombreProveedor)

REPUESTOS(NumeroRep, NombreRep)

PROYECTO(NumeroProyecto, NombreProyecto)

SUMINISTRO(NumeroProveedor, NumeroRep, NumeroProyecto)

La base de datos registra información acerca de los proveedores, repuestos y proyectos e incluye una relación ternaria entre ellos. Esta relación es de tipo muchos-muchos-muchos. Especifique y ejecute las siguientes consultas usando el intérprete RA.

a. Obtenga los números de repuesto que son suministrados a exactamente dos proyectos.

b. Obtenga los nombres de los proveedores que suministran más de dos repuestos al proyecto 'J1'.

c. Obtenga los números de repuesto suministrados por cada proveedor.

d. Obtenga los nombres de los proyectos suministrados sólo por el proveedor 'S1'.

e. Obtenga los nombres de los proveedores que suministran, al menos, dos repuestos diferentes a dos proyectos distintos.

6.38. Especifique y ejecute las siguientes consultas para la base de datos del Ejercicio 5.16 usando el intérprete RA.

a. Recupere los nombres de los estudiantes que están matriculados en un curso que utiliza libros de texto de la editorial Addison Wesley.

b. Recupere los nombres de los cursos en los que los libros se han cambiado una vez al menos.

c. Recupere los nombres de los departamentos que adoptan libros publicados sólo por Addison Wesley.

d. Recupere los nombres de los departamentos que adoptan libros escritos por Navathe y publicados por Addison Wesley.

e. Recupere los nombres de los estudiantes que nunca han usado un libro (en un curso) escrito por Navathe y publicado por Addison Wesley.

6.39. Repita los Ejercicios de prácticas del 6.34 al 6.38 en DRC (Cálculo relacional de dominio, *Domain Relational Calculus*) usando el intérprete DRC.

Bibliografía seleccionada

Codd (1970) definió el álgebra relacional básica. Date (1983a) abordó las concatenaciones externas. El trabajo para la extensión de las operaciones relacionales fue tratado por Carlis (1986) y Ozsoyoglu y otros (1985). Cammarata y otros (1989) expande las restricciones de integridad del modelo relacional y las concatenaciones.

Codd (1971) presentó el lenguaje Alpha, que está basado en los conceptos de los cálculos relacionales de tupla. Alpha incluye también la noción de función agregada, la cual va más allá de los cálculos relacionales. La definición formal original de los cálculos relacionales fue ofrecida por Codd (1972), el cual proporcionó

también un algoritmo que transforma cualquier expresión de cálculo relacional de tupla en álgebra relacional. El QUEL (Stonebraker y otros, 1976) está basado en los cálculos relacionales de tupla, con cuantificadores existenciales implícitos, pero no cuantificadores universales, y fue implementado en el sistema Ingres como un lenguaje comercial. Ullman (1988) ofrece una prueba de la equivalencia del álgebra relacional con las expresiones seguras del cálculo relacional de dominio y de tupla. Abiteboul y otros (1995), junto con Atzeni y de Antonellis (1993), ofrecen un tratamiento detallado de los lenguajes relacionales formales.

Aunque las ideas de los cálculos relacionales de dominio fueron propuestas inicialmente en el lenguaje QBE (Zloof 1975), el concepto fue definido formalmente por Lacroix y Pirotte (1977). La versión experimental del sistema QBE está descrita en Zloof (1977). El ILL (Lacroix y Pirotte 1977a) está basado en los cálculos relacionales de dominio. Whang y otros (1990) amplía el QBE con los cuantificadores universales. Los lenguajes de consulta visual, de los que QBE es un ejemplo, se propusieron como un medio de consultar las bases de datos; varios grupos como el Visual Database Systems Workshop (por ejemplo, Arisawa y Catarci [2000] o Zhou y Pu [2002]) ofrecen una gran cantidad de propuestas para lenguajes de este tipo.