Bases de Datos I
09/10/2020
Resumen Cap 08 09/10
IS-501
José Inestroza

Capítulo 08 Fundamentals

El álgebra relacional y el cálculo relacional
Dos lenguajes formales para el modelo relacional: el álgebra relacional y el cálculo relacional. Históricamente, el álgebra relacional y el cálculo se desarrollaron antes que el lenguaje SQL. SQL se basa principalmente en conceptos del cálculo relacional y se ha ampliado para incorporar algunos conceptos del álgebra relacional también. El conjunto básico de operaciones para el modelo relacional formal es el álgebra relacional. Estas operaciones permiten al usuario especificar solicitudes de recuperación básicas como expresiones de álgebra relacional. El resultado de una consulta de recuperación es una nueva relación. Las operaciones de álgebra producen así nuevas relaciones, que pueden manipularse aún más utilizando operaciones de la misma álgebra. Una secuencia de operaciones de álgebra relacional forma una expresión de álgebra relacional, cuyo resultado también será una relación que representa el resultado de una consulta de base de datos (o solicitud de recuperación). Razones por las que el álgebra relacional es importante:

- Primero, proporciona una base formal para las operaciones del modelo relacional.
- En segundo lugar, y quizás más importante, se utiliza como base para implementar y optimizar consultas en los módulos de optimización y procesamiento de consultas que son parte integral de los sistemas de gestión de bases de datos relacionales (RDBMS),
- En tercer lugar, algunos de sus conceptos están incorporados en el lenguaje de consulta estándar SQL para RDBMS. Aunque la mayoría de los RDBMS comerciales que se utilizan hoy en día no proporcionan interfaces de usuario para consultas de álgebra relacional, las operaciones y funciones centrales en los módulos internos de la mayoría de los sistemas relacionales se basan en operaciones de álgebra relacional.

Mientras que el álgebra define un conjunto de operaciones para el modelo relacional, el cálculo relacional proporciona un lenguaje declarativo de nivel superior para especificar consultas relacionales. En una expresión de cálculo relacional, no hay un orden de operaciones para especificar cómo recuperar el resultado de la consulta, solo qué información debe contener el resultado. El álgebra relacional sus operaciones se pueden dividir en dos grupos:

- Incluye operaciones de conjuntos de la teoría matemática de conjuntos, las operaciones de configuración incluyen UNION, INTERSECTION, SET DIFFERENCE y PRODUCTO CARTESIANO (también conocido como PRODUCTO CRUZADO).
- Consiste en operaciones desarrolladas específicamente para bases de datos relacionales; estas incluyen SELECT, PROJECT y JOIN, entre otras. Primero, describimos las operaciones SELECT y PROJECT.

Hay dos variaciones de cálculo relacional. El cálculo relacional de tuplasy el cálculo relacional de dominio.
8.1 Operaciones relacionales unarias

La operación SELECT se usa para elegir un subconjunto de las tuplas de una relación que satisface una condición de selección. Podemos considerar que la operación SELECT es un filtro que mantiene solo aquellas tuplas que satisfacen una condición de calificación. Alternativamente, podemos considerar la operación SELECT para restringir las tuplas en una relación solo a aquellas tuplas que satisfacen la condición. La operación SELECT también se puede visualizar como una partición horizontal de la relación en dos conjuntos de tuplas: las tuplas que satisfacen la condición y se seleccionan, y las tuplas que no cumplen la condición y se filtran.

En general, el resultado de una operación SELECT se puede determinar cómo sigue. La <condición de selección> se aplica independientemente a cada tupla t individual en R. Esto se hace sustituyendo cada aparición de un atributo Ai en la condición de selección con su valor en la tupla t [Ai]. Si la condición se evalúa como VERDADERA, se selecciona la tupla t. Todas las tuplas seleccionadas aparecen en el resultado de la operación SELECT. Las condiciones booleanas Y, O y NO tienen su interpretación normal, de la siguiente manera:
- (cond1 AND cond2) es TRUE si tanto (cond1) como (cond2) son TRUE; de lo contrario, es FALSO.
- (cond1 OR cond2) es TRUE si (cond1) o (cond2) o ambos son TRUE; de lo contrario, es FALSO.
- (NOT cond) es TRUE si cond es FALSE; de lo contrario, es FALSO.
El operador SELECT es unario; es decir, se aplica a una sola relación.

8.1.2 La operación PROYECTO
Si pensamos en una relación como una tabla, la operación SELECT elige algunas de las filas de la tabla mientras descarta otras filas. La operación PROYECTO, por otro lado, selecciona ciertas columnas de la tabla y descarta las otras columnas. Si estamos interesados solo en ciertos atributos de una relación, usamos la operación PROYECTO para proyectar la relación solo sobre estos atributos. Por tanto, el resultado de la operación PROYECTO se puede visualizar como una partición vertical de la relación en dos relaciones: uno tiene las columnas necesarias (atributos) y contiene el resultado de la operación, y el otro contiene las columnas descartadas.

El resultado de la operación PROYECTO solo tiene los atributos especificados en <lista de atributos> en el mismo orden en que aparecen en la lista. Por lo tanto, su grado es igual al número de atributos en <lista de atributos>. Si la lista de atributos incluye solo atributos no clave de R, es probable que se produzcan tuplas duplicadas. La operación PROYECTO elimina cualquier tupla duplicada, por lo que
El resultado de la operación PROYECTO es un conjunto de tuplas distintas y, por lo tanto, una relación válida. Esto se conoce como eliminación de duplicados.

Si no se eliminan los duplicados, el resultado sería un conjunto múltiple o una bolsa de tuplas en lugar de un conjunto. Esto no estaba permitido en el modelo relacional formal, pero está permitido en SQL. El número de tuplas en una relación resultante de una operación PROYECTO es siempre menor o igual que el número de tuplas en R.

Tenga en cuenta que si eliminamos la palabra clave DISTINCT de esta consulta SQL, los duplicados no se eliminarán. Esta opción no está disponible en el álgebra relacional formal, pero el álgebra puede extenderse para incluir esta operación y permitir que las relaciones sean de conjuntos múltiples.

8.1.3 Secuencias de operaciones y operación RENAME

En general, para la mayoría de las consultas, necesitamos aplicar varias operaciones de álgebra relacional una tras otra. O podemos escribir las operaciones como una sola expresión de álgebra relacional anidando las operaciones, o podemos aplicar una operación a la vez y crear relaciones de resultado intermedias. En el último caso, debemos dar nombres a las relaciones que contienen los resultados intermedios. También podemos utilizar esta técnica para cambiar el nombre de los atributos en las relaciones intermedia y de resultado. Esto puede ser útil en conexión con operaciones más complejas como UNION y JOIN, Para cambiar el nombre de los atributos en una relación, simplemente enumeramos los nombres de los nuevos atributos entre paréntesis, como en el siguiente ejemplo: TEMP ← σDno = 5 (EMPLEADO) R (Nombre, Apellido, Salario) ← πFnombre, Lnombre, Salario (TEMP).

También podemos definir una operación RENAME formal, que puede cambiar el nombre de la relación o los nombres de los atributos, o ambos, como un operador unario. El cambio de nombre en SQL se logra mediante la creación de alias mediante AS.

8.2 Operaciones de álgebra relacional a partir de la teoría de conjuntos

8.2.1 Las operaciones UNION, INTERSECTION y MINUS

El siguiente grupo de operaciones de álgebra relacional son las operaciones matemáticas estándar en conjuntos. Varias operaciones teóricas de conjuntos se utilizan para fusionar los elementos de dos conjuntos de diversas formas, incluidas UNION, INTERSECTION y SET DIFFERENCE (también llamado MINUS o EXCEPT). Estas son operaciones binarias; es decir, cada uno se aplica a dos conjuntos (de tuplas). Cuando estas operaciones se adaptan a bases de datos relacionales, las dos relaciones sobre las que se aplica cualquiera de estas tres operaciones deben tener el mismo tipo de tuplas; esta condición se ha denominado compatibilidad de unión o compatibilidad de tipos. Podemos definir las tres operaciones UNION, INTERSECTION y SET DIFFERENCE en dos relaciones R y S compatibles con la unión como sigue:

- **UNIÓN:** El resultado de esta operación, denotado por R ∪ S, es una relación que incluye todas las tuplas que están en R o en S o en R y S. Se eliminan las tuplas duplicadas.
- **INTERSECCIÓN:** El resultado de esta operación, denotado por R ∩ S, es una relación que incluye todas las tuplas que están tanto en R como en S.
- **SET DIFFERENCE (o MINUS):** el resultado de esta operación, denotado por R - S, es una relación que incluye todas las tuplas que están en R pero no en S.

Siempre es posible cambiar el nombre de los atributos en el resultado utilizando el operador de cambio de nombre. Tanto UNION como INTERSECTION pueden tratarse como operaciones n-arias aplicables a cualquier número de relaciones porque ambas son también operaciones asociativas; la operación MINUS no es conmutativa; es decir, en general, R - S ≠ S – R. Tenga en cuenta que la INTERSECCIÓN se puede expresar en términos de unión y establecer la diferencia de la siguiente manera:
R ∩ S = ((R ∪ S) - (R - S)) - (S - R) En SQL, hay tres operaciones — UNION, INTERSECT y EXCEPT — que corresponden a las operaciones de conjunto descritas aquí. Además, hay operaciones de varios conjuntos (UNION ALL, INTERSECT ALL y EXCEPT ALL) que no eliminan los duplicados.

8.2.2 El PRODUCTO CARTESIANO (PRODUCTO CRUZADO)

Operación A continuación, analizamos la operación PRODUCTO CARTESIANO, también conocida como PRODUCTO CRUZADO o UNIÓN CRUZADA, que se indica con ×. Esta también es una operación de conjunto binario, pero las relaciones en las que se aplica no

tienen que ser compatibles con la unión. En su forma binaria, esta operación de conjunto produce un nuevo elemento combinando cada miembro (tupla) de una relación (conjunto) con cada miembro (tupla) de la otra relación (conjunto). La operación PRODUCTO CARTESIANO n-ario es una extensión del concepto anterior, que produce nuevas tuplas al concatenar todas las combinaciones posibles de tuplas de n relaciones subyacentes. La operación de PRODUCTO CARTESIANO aplicada por sí misma generalmente no tiene sentido. Es sobre todo útil cuando va seguido de una selección que coincide con los valores de los atributos procedentes de las relaciones de los componentes.

El PRODUCTO CARTESIANO crea tuplas con los atributos combinados de dos relaciones. Podemos SELECCIONAR tuplas relacionadas sólo de las dos relaciones especificando una condición de selección apropiada después del producto cartesiano, como hicimos en el ejemplo anterior. Porque esta secuencia de PRODUCTO CARTESIANO seguido de SELECT se usa con bastante frecuencia para combinar tuplas relacionadas de dos relaciones, **una operación especial, llamada JOIN, fue creada** para especificar esta secuencia como una sola operación. Aternativamente, si hay dos tablas en la cláusula FROM y no hay una condición de unión correspondiente en la cláusula WHERE de la consulta SQL, el resultado también será el PRODUCTO CARTESIANO de las dos tablas.

8.3 Operaciones relacionales binarias: JOIN y DIVISION

8.3.1 La operación JOIN

La operación JOIN, denotada por, se usa para combinar tuplas relacionadas de dos relaciones en tuplas simples "más largas". Esta operación es muy importante para cualquier base de datos relacional con más de una relación porque nos permite procesar relaciones entre relaciones. La operación UNIR se puede especificar como una operación PRODUCTO CARTESIANO seguida de una operación SELECCIONAR. Sin embargo, JOIN es muy importante porque se usa con frecuencia al especificar consultas de base de datos.

En JOIN, solo las combinaciones de tuplas que satisfacen la condición de unión aparecen en el resultado, mientras que en el PRODUCTO CARTESIANO todas las combinaciones de tuplas se incluyen en el resultado. Cada combinación de tupla para la que la condición de unión se evalúa como VERDADERA se incluye en la relación resultante Q como una única tupla combinada. Una operación JOIN con una condición de unión tan general se denomina THETA JOIN. Las tuplas cuyos atributos de combinación son NULL o cuya condición de combinación es FALSE no aparecen en el resultado. En ese sentido, la operación JOIN no necesariamente conserva toda la información en las relaciones de participación, porque las tuplas que no se combinan con las coincidentes en la otra relación no aparecen en el resultado.

8.3.2 Variaciones de JOIN: EQUIJOIN y NATURAL JOIN

El uso más común de JOIN implica condiciones de combinación con solo comparaciones de igualdad. Tal JOIN, donde el único operador de comparación utilizado es =, se llama EQUIJOIN. Debido a que uno de cada par de atributos con valores idénticos es superfluo, una nueva operación llamada NATURAL JOI —denotada por *: fue creado para deshacerse del segundo atributo (superfluo) en una condición EQUIJOIN. La definición estándar de NATURAL JOIN requiere que los dos atributos de combinación (o cada par de atributos de combinación) tengan el mismo nombre en ambas relaciones. Si este no es el caso, primero se aplica una operación de cambio de nombre. Si los atributos en los que se especifica la combinación natural ya tienen los mismos nombres en ambas relaciones,

no es necesario cambiar el nombre. n general, la condición de combinación para NATURAL JOIN se construye equiparando cada par de atributos de combinación que tienen el mismo nombre en las dos relaciones y combinando estas condiciones con AND. Puede haber una lista de atributos de combinación de cada relación y cada par correspondiente debe tener el mismo nombre.

En general, si R tiene nR tuplas y S tiene nS tuplas, el resultado de una operación JOIN R <condición de unión> S tendrá entre cero y nR * nS tuplas. El tamaño esperado del resultado de la unión dividido por el tamaño máximo nR * nS conduce a una relación denominada selectividad de la unión, que es una propiedad de cada condición de unión. Si no hay una condición de unión, todas las combinaciones de tuplas califican y la UNIÓN degenera en un PRODUCTO CARTESIANO, también llamado PRODUCTO CRUZADO o UNIÓN CRUZADA. Como podemos ver, una sola operación JOIN se usa para combinar datos de dos relaciones para que la información relacionada se pueda presentar en una sola tabla. Estas operaciones también se conocen como combinaciones internas, para distinguirlas de una variación de combinación diferente llamada combinaciones externas.

De manera informal, una unión interna es un tipo de operación de emparejamiento y combinación definida formalmente como una combinación de PRODUCTO CARTESIANO y SELECCIÓN. Tenga en cuenta que a veces se puede especificar una unión entre una relación y él mismo, La operación NATURAL JOIN o EQUIJOIN también se puede especificar entre varias tablas, lo que lleva a una unión de n vías.

En SQL, JOIN se puede realizar de varias formas diferentes. El primer método es especificar las <condiciones de unión> en la cláusula WHERE, junto con cualquier otra condición de selección. La segunda forma es usar un relación anidada, otra forma es utilizar el concepto de tablas unidas, l constructo
de tablas unidas se agregó a SQL2 para permitir al usuario especificar explícitamente todos los diversos tipos de uniones, porque los otros métodos eran más limitados. También permite al usuario distinguir claramente las condiciones de combinación de las condiciones de selección en la cláusula WHERE.

8.3.3 Un conjunto completo de operaciones de álgebra relacional

Se ha demostrado que el conjunto de operaciones de álgebra relacional {σ, π, ∪, ρ, -, ×} es un conjunto completo; es decir, cualquiera de las otras operaciones originales del álgebra relacional puede expresarse como una secuencia de operaciones de este conjunto. Por ejemplo, la operación de INTERSECCIÓN se puede expresar usando UNIÓN y MENOS de la siguiente manera: R ∩ S ≡ (R ∪ S) - ((R - S) ∪ (S - R)) Aunque, estrictamente hablando, la INTERSECCIÓN no es necesaria, es inconveniente especificar esta expresión compleja cada vez que deseamos especificar una intersección. Como otro ejemplo, una operación JOIN se puede especificar como un PRODUCTO CARTESIANO seguida de una operación SELECT, De manera similar, una UNIÓN NATURAL se puede especificar como un CARTESIANO. PRODUCTO precedido de RENAME y seguido de las operaciones SELECT y PROJECT. Por lo tanto, las diversas operaciones JOIN tampoco son estrictamente necesarias para el poder expresivo del álgebra relacional. Sin embargo, es importante incluirlos como operaciones independientes porque son convenientes de usar y se aplican con mucha frecuencia en aplicaciones de bases de datos.

8.3.4 La operación DIVISION

La operación DIVISION, denotada por ÷, es útil para un tipo especial de consulta que a veces ocurre en aplicaciones de bases de datos. En general, la operación DIVISION se aplica a dos relaciones R (Z) ÷ S (X), donde los atributos de S son un subconjunto de los

atributos de R; es decir, $X \subseteq Z$. Sea Y el conjunto de atributos de R que no son atributos de S; es decir, $Y = Z - X$ (y por tanto $Z = X \cup Y$).

La operación DIVISION se puede expresar como una secuencia de operaciones π, × y - de la siguiente manera:
$T1 \leftarrow \pi Y (R)$
$T2 \leftarrow \pi Y ((S \times T1) - R)$
$T \leftarrow T1 - T2$
La operación DIVISION se define por conveniencia para tratar consultas que implican cuantificación universal o la condición total. La mayoría de las implementaciones de RDBMS con SQL como lenguaje de consulta principal no implementan directamente la división. SQL tiene una forma indirecta de tratar con la división.

8.3.5 Notación para árboles de consultas
En esta sección, describimos una notación que se usa típicamente en DBMS relacionales (RDBMS) para representar consultas internamente. La notación se denomina árbol de consultas o, a veces, se conoce como árbol de evaluación de consultas o árbol de ejecución de consultas. Incluye las operaciones de álgebra relacional que se están ejecutando 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 de árbol que corresponde a una expresión de álgebra relacional. Representa las relaciones de entrada de la consulta como nodos hoja del árbol y representa las operaciones de álgebra relacional como nodos internos. Una ejecución del árbol de consulta consiste en ejecutar una operación de nodo interno siempre que sus operandos (representados por sus nodos secundarios) estén disponibles, y luego reemplazar ese nodo interno por la relación que resulta de ejecutar la operación. La ejecución termina cuando se ejecuta el nodo raíz y produce la relación de resultado para la consulta.

Las operaciones de álgebra relacional en la expresión están representadas por nodos de árbol internos. El árbol de consultas significa un orden explícito de ejecución en el siguiente sentido. Para ejecutar Q2, el nodo marcado (1) en la Figura 8.9 debe comenzar la ejecución antes que el nodo (2) porque algunas tuplas resultantes de la operación (1) deben estar disponibles antes de que podamos comenzar a ejecutar la operación (2). De manera similar, el nodo (2) debe comenzar a ejecutarse y producir resultados antes de que el nodo (3) pueda iniciar la ejecución, y así sucesivamente. En general, un árbol de consultas proporciona una buena representación visual y comprensión de la consulta en términos de las operaciones relacionales que utiliza y se recomienda como un medio adicional para expresar consultas en álgebra relacional.

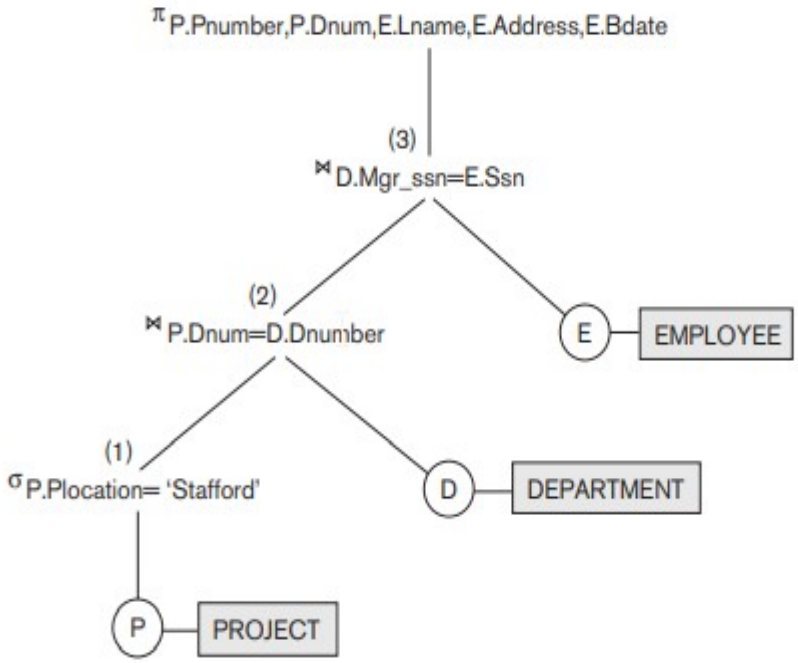

Figure 8.9
Query tree corresponding to the relational algebra expression for Q2.

Table 8.1 Operations of Relational Algebra

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{<\text{selection condition}>}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{<\text{attribute list}>}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{<\text{join condition}>} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{<\text{join condition}>} R_2$, OR $R_1 \bowtie_{(<\text{join attributes 1}>), (<\text{join attributes 2}>)} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{<\text{join condition}>} R_2$, OR $R_1 *_{(<\text{join attributes 1}>), (<\text{join attributes 2}>)} R_2$ OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

8.4 Operaciones relacionales adicionales

Algunas solicitudes de bases de datos comunes, que son necesarias en aplicaciones comerciales para RDBMS, no se pueden realizar con las operaciones originales de álgebra relacional.

8.4.1 Proyección generalizada

La operación de proyección generalizada amplía la operación de proyección al permitir que se incluyan funciones de atributos en la lista de proyección. La forma generalizada se puede expresar como:

πF1, F2,..., Fn (R).

Donde F1, F2,…, Fn son funciones sobre los atributos en relación R y pueden involucrar operaciones aritméticas y valores constantes. Esta operación es útil cuando se desarrollan informes donde los valores calculados deben producirse en las columnas del resultado de una consulta.

8.4.2 Funciones agregadas y agrupación

Otro tipo de solicitud que no se puede expresar en el álgebra relacional básica es especificar funciones matemáticas agregadas en colecciones de valores de la base de datos. Ejemplos de tales funciones incluyen recuperar el salario promedio o total de todos

los empleados o el número total de tuplas de empleados. Estas funciones se utilizan en consultas estadísticas simples que resumen la información de las tuplas de la base de datos. Las funciones comunes aplicadas a las colecciones de valores numéricos incluyen SUMA, PROMEDIO, MÁXIMO y MÍNIMO. La función COUNT se utiliza para contar tuplas o valores.
Otro tipo común de solicitud implica agrupar las tuplas en una relación por el valor de algunos de sus atributos y luego aplicar una función agregada independientemente a cada grupo.

En cada uno de estos pares, <función> es una de las funciones permitidas, como SUMA, PROMEDIO, MÁXIMO, MÍNIMO, RECUENTO, y <atributo> es un atributo de la relación especificada por R. La relación resultante tiene los atributos de agrupación más un atributo para cada elemento en la lista de funciones. Si no se especifican atributos de agrupación, las funciones se aplican a todas las tuplas de la relación, por lo que la relación resultante tiene una única tupla.

Es importante notar que, en general, los duplicados no se eliminan cuando se aplica una función agregada; de esta manera, se calcula la interpretación normal de funciones como SUMA y PROMEDIO Sin embargo, los valores NULOS no se consideran en la agregación, Vale la pena enfatizar que el resultado de aplicar una función agregada es una relación, no un número escalar, incluso si tiene un solo valor. Esto hace que el álgebra relacional sea un sistema matemático cerrado.

8.4.3 Operaciones de cierre recursivo
Otro tipo de operación que, en general, no se puede especificar en el álgebra relacional original básica es el cierre recursivo. Esta operación se aplica a una relación recursiva entre tuplas del mismo tipo, como la relación entre un empleado y un supervisor.

8.4.4 Operaciones OUTER JOIN
A continuación, discutimos algunas extensiones adicionales a la operación JOIN que son necesarias para especificar ciertos tipos de consultas. Las operaciones JOIN descritas anteriormente coinciden con tuplas que satisfacen la condición de combinación. También se eliminan las tuplas con valores NULL en los atributos de combinación. Este tipo de combinación, donde se eliminan las tuplas sin coincidencia, se conoce como combinación interna.

Esto equivale a la pérdida de información si el usuario desea que el resultado de JOIN incluya todas las tuplas en una o más de las relaciones de componentes. Se desarrolló un conjunto de operaciones, llamadas uniones externas, para el caso en que el usuario quiera mantener todas las tuplas en R, o todas las de S, o todas las de ambas relaciones en el resultado de la UNIÓN, independientemente de si tienen tuplas coincidentes en la otra relación. Esto satisface la necesidad de consultas en las que las tuplas de dos tablas deben combinarse haciendo coincidir las filas correspondientes, pero sin perder cualquier tupla por falta de valores coincidentes.

La operación LEFT OUTER JOIN mantiene cada tupla en la primera relación, o izquierda, R en R S; si no se encuentra una tupla coincidente en S, los atributos de S en el resultado de la combinación se rellenan o rellenan con valores NULL.

Una operación similar, RIGHT OUTER JOIN, denotada por, mantiene cada tupla en la segunda, o la derecha, relación S en el resultado de R S. Una tercera operación, FULL OUTER JOIN, denotado por, mantiene todas las tuplas en las relaciones izquierda y

derecha cuando no se encuentran tuplas coincidentes, rellenándolas con valores NULL según sea necesario.

8.4.5 La operación UNIÓN EXTERIOR

La operación OUTER UNION se desarrolló para tomar la unión de tuplas de dos relaciones que tienen algunos atributos comunes, pero no son compatibles con la unión (tipo). Esta operación tomará la UNIÓN de tuplas en dos relaciones R (X, Y) y S (X, Z) que son parcialmente compatibles, lo que significa que solo algunos de sus atributos, digamos X, son compatibles con la unión. Los atributos que son compatibles con la unión se representan solo una vez en el resultado, y aquellos atributos que no son compatibles con la unión de cualquiera de las relaciones también se mantienen en la relación de resultado T (X, Y, Z). Por lo tanto, es lo mismo que FULL OUTER JOIN en los atributos comunes Las tuplas en cualquiera de las relaciones que no tienen tuplas coincidentes en la otra relación se rellenan con valores NULL.

8.6 El cálculo relacional de tuplas

El lenguaje conocido como cálculo relacional de tuplas y una variación llamada cálculo relacional de dominio. N ambas variaciones del cálculo relacional, escribimos una expresión declarativa para especificar una solicitud de recuperación; por lo tanto, no hay una descripción de cómo o en qué orden evaluar una consulta. Una expresión de cálculo especifica qué se recuperará en lugar de como recuperarlo. Por tanto, se considera que el cálculo relacional es un lenguaje no procedimental. Esto difiere del álgebra relacional, donde debemos escribir una secuencia de operaciones para especificar una solicitud de recuperación en un orden particular de aplicación de las operaciones; por tanto, puede considerarse como una forma procedimental de plantear una consulta.

Es posible anidar operaciones de álgebra para formar una sola expresión; sin embargo, un cierto orden entre las operaciones siempre se especifica explícitamente en una expresión de álgebra relacional. Este orden también influye en la estrategia para evaluar la consulta. Una expresión de cálculo se puede escribir de diferentes formas, pero la forma en que se escribe no influye en cómo se debe evaluar una consulta. Se ha demostrado que cualquier recuperación que se pueda especificar en el álgebra relacional básica también se puede especificar en el cálculo relacional y viceversa; en otras palabras, el poder expresivo de los lenguajes es idéntico. Esto llevó a la definición del concepto de un lenguaje relacionalmente completo. Un lenguaje de consulta relacional L se considera relacionalmente completo si podemos expresar en L cualquier consulta que pueda expresarse en cálculo relacional. La completitud relacional se ha convertido en una base importante para comparar el poder expresivo de los lenguajes de consulta de alto nivel. Ciertas consultas requeridas con frecuencia en aplicaciones de bases de datos no se pueden expresar en álgebra relacional básica o cálculo. La mayoría de los lenguajes de consulta relacionales son relacionalmente completos pero tienen más poder expresivo que el álgebra relacional o el cálculo relacional debido a operaciones adicionales como funciones agregadas, agrupamiento y ordenamiento.

El cálculo relacional es importante por dos razones. Primero, tiene una base firme en lógica matemática. En segundo lugar, el lenguaje de consulta estándar (SQL) para RDBMS tiene su base básica en el cálculo relacional de tuplas.

8.6.1 Variables de tupla y relaciones de rango

El cálculo relacional de tuplas se basa en especificar una serie de variables de tuplas. Cada variable de tupla generalmente varía sobre una relación de base de datos particular,

lo que significa que la variable puede tomar como valor cualquier tupla individual de esa relación.

De manera informal, necesitamos especificar la siguiente información en una expresión de cálculo relacional de tupla:

- Para cada variable tupla t, la relación de rango R de t. Este valor se especifica mediante una condición de la forma R (t). Si no especificamos una relación de rango, entonces la variable t abarcará todas las tuplas posibles "en el universo" ya que no está restringida a ninguna relación.
- Una condición para seleccionar combinaciones particulares de tuplas. Como las variables de tupla varían en sus respectivas relaciones de rango, la condición se evalúa para cada combinación posible de tuplas para identificar las combinaciones seleccionadas para las cuales la condición se evalúa como VERDADERA.
- Un conjunto de atributos a recuperar, los atributos solicitados. Los valores de estos atributos se recuperan para cada combinación seleccionada de tuplas. Antes de discutir la sintaxis formal del cálculo relacional de tuplas, considere otra consulta.

8.6.2 Expresiones y fórmulas en el cálculo relacional de tuplas

Una expresión general del cálculo relacional de tuplas tiene la forma
{t1.Aj, t2.Ak,..., tn.Am | COND (t1, t2,..., tn, tn + 1, tn + 2,..., tn + m)}
Donde t1, t2,…, tn, tn + 1,…, tn + m son variables de tupla, cada Ai es un atributo de la relación en la que ti varía, y COND es una condición o fórmula13 del cálculo relacional de tuplas. Una fórmula está formada por átomos de cálculo de predicados, que pueden ser uno de los siguientes:

- Un átomo de la forma R (ti), donde R es un nombre de relación y ti es una variable de tupla. Este átomo identifica el rango de la variable tupla ti como la relación cuyo nombre es R. Se evalúa como VERDADERO si ti es una tupla en la relación R, y se evalúa como FALSO en caso contrario.
- Un átomo de la forma ti.A op tj.B, donde op es uno de los operadores de comparación en el conjunto {=, <, ≤,>, ≥, ≠}, ti y tj son variables de tupla, A es una atributo de la relación en la que ti varía, y B es un atributo de la relación en la que tj varía.
- Un átomo de la forma ti.A op c o c op tj.B, donde op es uno de los operadores de comparación en el conjunto {=, <, ≤,>, ≥, ≠}, ti y tj son variables de tupla, A es un atributo de la relación en la que ti varía, B es un atributo de la relación en la que tj varía y c es un valor constante. Cada uno de los átomos precedentes se evalúa como VERDADERO o FALSO para una combinación específica de tuplas; esto se llama el valor de verdad de un átomo.

En general, una variable de tupla t abarca todas las tuplas posibles del universo. Para átomos de la forma R (t), si t se asigna a una tupla que es miembro de la relación R especificada, el átomo es VERDADERO; de lo contrario, es FALSO. En los átomos de los tipos 2 y 3, si las variables de tupla se asignan a tuplas de manera que los valores de los atributos especificados de las tuplas satisfacen la condición, entonces el átomo es VERDADERO. Una fórmula (condición booleana) se compone de uno o más átomos conectados mediante los operadores lógicos Y, O y NO y se define de forma recursiva por las Reglas 1 y 2 de la siguiente manera:

- Regla 1: cada átomo es una fórmula.
- Regla 2: Si F1 y F2 son fórmulas, también lo son (F1 Y F2), (F1 O F2), NO (F1) y NO (F2).

Los valores de verdad de estas fórmulas se derivan de sus fórmulas componentes F1 y F2 de la siguiente manera:

- a. (F1 Y F2) es VERDADERO si F1 y F2 son VERDADEROS; de lo contrario, es FALSO.
- segundo. (F1 O F2) es FALSO si F1 y F2 son FALSOS; de lo contrario, es VERDADERO.
- C. NO (F1) es VERDADERO si F1 es FALSO; es FALSO si F1 es VERDADERO.
- D. NO (F2) es VERDADERO si F2 es FALSO; es FALSO si F2 es VERDADERO.

8.6.3 Los cuantificadores existenciales y universales

Además, en las fórmulas pueden aparecer dos símbolos especiales llamados cuantificadores; estos son el cuantificador universal (∀) y el cuantificador existencial (∃). De manera informal, una variable de tupla t está limitada si se cuantifica, lo que significa que aparece en una cláusula (∃t) o (∀t); de lo contrario, es libre. Formalmente, definimos una variable de tupla en una fórmula como libre o ligada de acuerdo con las siguientes reglas:

- Una ocurrencia de una variable tupla en una fórmula F que es un átomo está libre en F.
- Una ocurrencia de una variable tupla t es libre o está ligada en una fórmula formada por conectivos lógicos— (F1 AND F2), (F1 OR F2), NOT (F1) y NOT (F2) - dependiendo de si es libre o enlazado en F1 o F2 (si ocurre en cualquiera).

Observe que en una fórmula de la forma F = (F1 Y F2) o F = (F1 O F2), una variable tupla puede estar libre en F1 y limitada en F2, o viceversa; en este caso, una aparición de la variable tupla está ligada y la otra está libre en F.

- Todas las apariciones libres de una variable tupla t en F están ligadas en una fórmula F ′ de la forma F ′ = (∃t) (F) o F ′ = (∀t) (F).

Regla 3: si F es una fórmula, también lo es (∃t) (F), donde t es una variable de tupla. La fórmula (∃t) (F) es VERDADERA si la fórmula F se evalúa como VERDADERA para alguna (al menos una) tupla asignada a apariciones libres de t en F; de lo contrario, (∃t) (F) es FALSO.
Regla 4: si F es una fórmula, también lo es (∀t) (F), donde t es una variable tupla. La fórmula (∀t) (F) es VERDADERA si la fórmula F se evalúa como VERDADERA para cada tupla (en el universo) asignada a apariciones libres de t en F; de lo contrario, (∀t) (F) es FALSO.

El cuantificador (∃) se llama cuantificador existencial porque una fórmula (∃t) (F) es VERDADERA si existe alguna tupla que hace que F sea VERDADERA. Para el cuantificador universal, (∀t) (F) es VERDADERO si cada posible tupla que se puede asignar a apariciones libres de t en F se sustituye por t, y F es VERDADERO para cada sustitución de este tipo. Se llama cuantificador universal o para todos porque cada tupla en el universo de tuplas debe hacer F VERDADERO para que la fórmula cuantificada sea VERDADERA.

8.6.5 Notación para grafos de consulta

Estos tipos de consultas se conocen como consultas de selección-proyecto-unión porque solo involucran estas tres operaciones de álgebra relacional. Las relaciones en la consulta se representan mediante nodos de relación, que se muestran como círculos individuales. Los valores constantes, normalmente de las condiciones de selección de la consulta, se representan mediante nodos constantes, que se muestran como círculos dobles u óvalos.

Las condiciones de selección y unión están representadas por los bordes del grafo. Finalmente, los atributos que se recuperarán de cada relación se muestran entre corchetes encima de cada relación. La representación del gráfico de la consulta no indica un orden particular para especificar qué operaciones realizar primero y, por lo tanto, es una representación más neutral de una consulta selectproject-join que la representación del árbol de consultas donde el orden de ejecución es implícitamente especificado. Solo hay un único gráfico de consulta correspondiente a cada consulta.

Aunque algunas técnicas de optimización de consultas se basaban en gráficos de consultas, ahora se acepta generalmente que los árboles de consultas son preferibles porque, en la práctica, el optimizador de consultas necesita mostrar el orden de las operaciones para la ejecución de consultas, lo que no es posible en los gráficos de consultas.

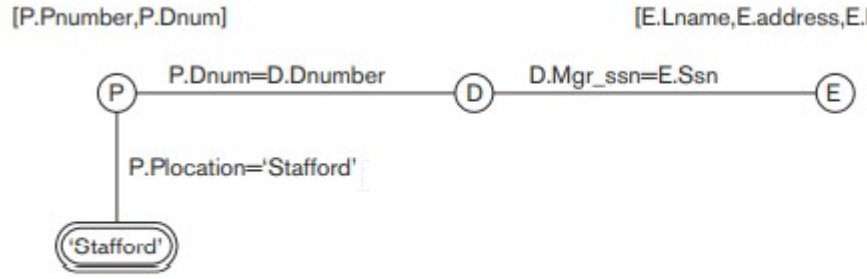

Figure 8.13
Query graph for Q2.

8.6.6 Transformar los cuantificadores universal y existencial

A continuación, presentamos algunas transformaciones conocidas de la lógica matemática que relacionan los cuantificadores universal y existencial. Es posible transformar un cuantificador universal en un cuantificador existencial, y viceversa, para obtener una expresión equivalente. Una transformación general se puede describir informalmente como sigue: Transformar un tipo de cuantificador en otro con negación (precedido por NOT); Y O se reemplazan entre sí; una fórmula negada se vuelve innecesaria; y una fórmula innecesaria se invalida. Algunos casos especiales de esta transformación se pueden enunciar de la siguiente manera, donde el símbolo ≡ significa equivalente a:
(∀x) (P (x)) ≡ NO (∃x) (NO (P (x)))
(∃x) (P (x)) ≡ NO (∀x) (NO (P (x)))
(∀x) (P (x) Y Q (x)) ≡ NO (∃x) (NO (P (x)) O NO (Q (x)))
(∀x) (P (x) O Q (x)) ≡ NO (∃x) (NO (P (x)) Y NO (Q (x)))
(∃x) (P (x)) O Q (x)) ≡ NO (∀x) (NO (P (x)) Y NO (Q (x)))
(∃x) (P (x) Y Q (x)) ≡ NO (∀x) (NO (P (x)) O NO (Q (x)))
Observe también que lo siguiente es VERDADERO, donde el símbolo ⇒ significa:
(∀x) (P (x)) ⇒ (∃x) (P (x))
NO (∃x) (P (x)) ⇒ NO (∀x) (P (x))

8.6.8 Expresiones seguras

Siempre que usemos cuantificadores universales, cuantificadores existenciales o negación de predicados en una expresión de cálculo, debemos asegurarnos de que la expresión resultante tenga sentido. Una expresión segura en cálculo relacional es aquella que está garantizada para producir un número finito de tuplas como resultado; de lo contrario, la expresión se denomina insegura. . Podemos definir expresiones seguras de manera más precisa al introducir el concepto del dominio de una expresión de cálculo relacional de tupla: Este es el conjunto de todos los valores que aparecen como valores constantes en la expresión o existen en cualquier tupla en las relaciones referenciadas en

la expresión. Se dice que una expresión es segura si todos los valores de su resultado son del dominio de la expresión.

8.7 El dominio del cálculo relacional

Existe otro tipo de cálculo relacional llamado cálculo relacional de dominios, o simplemente cálculo de dominios. Históricamente, mientras que SQL, que se basaba en el cálculo relacional de tuplas, estaba siendo desarrollado por IBM Research en San José, California, otro lenguaje llamado QBE (Query-By-Example), que está relacionado con el cálculo de dominios. , se estaba desarrollando casi simultáneamente en el IBM TJ Watson Research Center en Yorktown Heights, Nueva York. La especificación formal del cálculo de dominio se propuso después del desarrollo del lenguaje y sistema QBE. El cálculo de dominios difiere del cálculo de tuplas en el tipo de variables utilizadas en las fórmulas: en lugar de tener un rango de variables sobre tuplas, las variables varían sobre valores individuales de dominios de atributos. Para formar una relación de grado n para el resultado de una consulta, debemos tener n de estas variables de dominio, una para cada atributo. Una expresión del cálculo de dominios tiene la forma {x1, x2,..., xn | COND (x1, x2,..., xn, xn + 1, xn + 2,..., xn + m)}

Donde x1, x2,…, xn, xn + 1, xn + 2,…, xn + m son variables de dominio que abarcan dominios (de atributos) y COND es una condición o fórmula del cálculo relacional de dominio.

Una fórmula está formada por átomos. Los átomos de una fórmula son ligeramente diferentes de los del cálculo de tuplas y pueden ser uno de los siguientes:

1. Un átomo de la forma R (x1, x2,…, xj), donde R es el nombre de una relación de grado j y cada xi, 1 ≤ i ≤ j, es una variable de dominio. Este átomo establece que una lista de valores de <x1, x2,…, xj> debe ser una tupla en la relación cuyo nombre es R, donde xi es el valor del i-ésimo valor del atributo de la tupla. Para hacer una expresión de cálculo de dominio más concisa, podemos colocar las comas en una lista de variables; así, podemos escribir: {x1, x2,..., xn | R (x1 x2 x3) Y...}

En lugar de: {x1, x2,..., xn | R (x1, x2, x3) Y...}

2. Un átomo de la forma xi op xj, donde op es uno de los operadores de comparación en el conjunto {=, <, ≤,>, ≥, ≠}, y xi y xj son variables de dominio.

3. Un átomo de la forma xi op c o c op xj, donde op es uno de los operadores de comparación en el conjunto {=, <, ≤,>, ≥, ≠}, xi y xj son variables de dominio, y c es un valor constante.

Como en el cálculo de tuplas, los átomos se evalúan como VERDADERO o FALSO para un conjunto específico de valores, llamados valores de verdad de los átomos. En el caso 1, si a las variables de dominio se les asignan valores correspondientes a una tupla de la relación R especificada, entonces el átomo es VERDADERO. En los casos 2 y 3, si a las variables de dominio se les asignan valores que satisfacen la condición, entonces el átomo es VERDADERO.

Como mencionamos anteriormente, se puede demostrar que cualquier consulta que se pueda expresar en el álgebra relacional básica también se puede expresar en el cálculo relacional de dominio o tupla. Además, cualquier expresión segura en el dominio o tupla de cálculo relacional se puede expresar en el álgebra relacional básica. El lenguaje QBE se basó en el cálculo relacional de dominios, aunque esto se realizó más tarde, después de que se formalizó el cálculo de dominios. QBE fue uno de los primeros lenguajes de consulta gráfica con sintaxis mínima desarrollado para sistemas de bases de datos. Fue desarrollado en IBM Research y está disponible como un producto comercial de IBM como parte de la opción de interfaz Query Management Facility (QMF) para DB2. Las ideas básicas utilizadas en QBE se han aplicado en varios otros productos comerciales. Debido a su lugar importante en la historia de los lenguajes relacionales.

8.8 Resumen

En este capítulo presentamos dos lenguajes formales para el modelo relacional de datos. Se utilizan para manipular relaciones y producir nuevas relaciones como respuesta a consultas. Discutimos el álgebra relacional y sus operaciones, que se utilizan para especificar una secuencia de operaciones para especificar una consulta. Luego, presentamos dos tipos de cálculos relacionales llamados cálculo de tuplas y cálculo de dominios. En las secciones 8.1 a 8.3, presentamos las operaciones básicas de álgebra relacional e ilustramos los tipos de consultas para las que se usa cada una. Primero, discutimos los operadores relacionales unarios SELECT y PROJECT, así como la operación RENAME. Luego, discutimos las operaciones teóricas de conjuntos binarios que requieren que las relaciones en
Que se apliquen sean compatibles con la unión (o tipo); estos incluyen UNION, INTERSECTION y SET DIFFERENCE. La operación PRODUCTO CARTESIANO es una operación de conjunto que se puede utilizar para combinar tuplas de dos relaciones, produciendo todas las combinaciones posibles. Rara vez se usa en la práctica; sin embargo, mostramos cómo PRODUCTO CARTESIANO seguido de SELECCIONAR se puede usar para definir tuplas coincidentes
De dos relaciones y conduce a la operación JOIN. Se introdujeron diferentes operaciones JOIN denominadas THETA JOIN, EQUIJOIN y NATURAL JOIN. Los árboles de consulta se introdujeron como una representación gráfica de consultas de álgebra relacional, que también se pueden utilizar como base para estructuras de datos internas que el DBMS puede utilizar para representar una consulta. Discutimos algunos tipos importantes de consultas que no se pueden formular con las operaciones básicas de álgebra relacional, pero que son importantes para situaciones prácticas. Introdujimos PROYECCIÓN GENERALIZADA para usar funciones de atributos en la proyección lista y la operación FUNCIÓN AGREGADA para tratar tipos agregados de solicitudes estadísticas que resumen la información en las tablas. Discutimos las consultas recursivas, para las cuales no hay apoyo directo en el álgebra pero que pueden manejarse en un enfoque paso a paso, como demostramos. Luego presentamos las operaciones OUTER JOIN y OUTER UNION, que extienden JOIN y UNION y permiten que toda la información en las relaciones de origen se conserve en el resultado.

Las dos últimas secciones describieron los conceptos básicos detrás del cálculo relacional, que se basa en la rama de la lógica matemática llamada cálculo de predicados.Hay dos tipos de cálculo relacional: (1) el cálculo relacional de tuplas, que usa variables de tuplas que se extienden sobre tuplas (filas) de relaciones, y (2) el cálculo relacional de dominios, que usa variables de dominio que abarcan dominios (columnas de relaciones). En cálculo relacional, una consulta se especifica en una única declaración declarativa, sin especificar ningún orden o método para recuperar el resultado de la consulta. Por lo tanto, el cálculo relacional a menudo se considera un lenguaje declarativo de mayor nivel que el álgebra relacional, porque una expresión de cálculo relacional establece lo que queremos recuperar independientemente de cómo se ejecute la consulta.
Introdujimos gráficos de consulta como una representación interna para consultas en cálculo relacional. También discutimos el cuantificador existencial (∃) y el cuantificador universal (∀). Discutimos el problema de especificar consultas seguras cuyos resultados son finitos. También discutimos las reglas para transformar cuantificadores universales en existenciales, y viceversa. Son los cuantificadores los que dan poder expresivo al cálculo relacional, haciéndolo equivalente al álgebra relacional básica. No existe una analogía con las funciones de agrupación y agregación en el cálculo relacional básico, aunque se han sugerido algunas extensiones.