Universidad Nacional de Luján
Departamento de Ciencias Básicas
11411 - Base de Datos

ALGEBRA RELACIONAL

Lic. Guillermo Cherencio

INTRODUCCION:

En el año 1970, en el artículo original presentado por E. F. Codd se introdujo al Modelo Relacional y el Algebra Relacional. El Algebra Relacional fué definido por Codd como un lenguaje procedimental para la manipulación de relaciones. Posteriormente, en 1971, Codd introdujo el Cálculo Relacional como un lenguaje no procedimental. Pudo demostrar que El Algebra Relacional y El Cálculo Relacional son lógicamente equivalentes, es decir, que cualquier consulta que podía ser formulada en Cálculo Relacional también se podía formular en Algebra Relacional y viceversa.

Estos dos lenguajes fueron el aspecto más importante del nuevo modelo presentado por Codd ya que, permitían la manipulación de datos basándose únicamente en sus características lógicas y sentaron las bases para el desarrollo de los lenguajes relacionales comerciales. Para que un lenguaje pueda ser llamado relacionalmente completo debería ser tan poderoso como El Algebra Relacional o El Cálculo Relacional, de otra manera, habría ciertas consultas que no podrían resolverse con dicho lenguaje comercial.

Tanto El Algebra Relacional como El Cálculo Relacional son lenguajes teóricos y sólo estamos interesados en los aspectos conceptuales de los mismos, por lo tanto, será bastante libre e informal la definición y el uso de la sintaxis; no será de esta forma cuando estudiemos el lenguaje SQL, el cual tiene una sintaxis bien precisa.

ALGEBRA RELACIONAL:

El Algebra Relacional tiene un conjunto de operaciones básicas que nos permiten crear nuevas relaciones utilizando una o más relaciones previamente existentes, a su vez, esta nueva relación puede usarse de entrada para una nueva operación; este poderoso concepto, hace posible una variedad infinita de manipulaciones posibles sobre los datos y hace considerablemente más fácil la solución de las consultas, debido a que se puede experimentar con soluciones parciales hasta alcanzar el objetivo buscado.

El Algebra Relacional consta de las siguientes nueve operaciones: unión, intersección, diferencia, producto, selección, proyección, reunión, división y asignación. Las cuatro primeras operaciones fueron tomadas de la teoría de conjuntos, la razón de esto, es que, las relaciones en sí mismas son conjuntos, de esta manera, se les pueden aplicar dichas operaciones. Las cuatro operaciones siguientes son nuevas y aplicables específicamente al Modelo Relacional. Por último, la asignación es utilizada para darle un nombre a una nueva relación que es creada a partir de relaciones existentes. Utilizando la implementación del D.E.R. del Mundial de Fútbol (Ejercicio Nº 10 de la práctica de DER) vista en clase, vamos a proceder a explicar cada una de las operaciónes básicas del Algebra Relacional:

UNION:

Permite combinar los datos de dos relaciones, ambas relaciones deberán tener exactamente la misma cantidad de columnas, tanto en número como en el dominio de cada una de ellas (no es necesario que las columnas tengan el mismo nombre), sólo en este caso se dice que las dos relaciones son unión compatible y ésto también es requisito para la operación intersección y diferencia. No podemos combinar dos relaciones que no sean compatibles, por ej., la unión entre AUTORIDAD y EQUIPO no sería posible y su resultado no podría ser una relación. Otra observación es que si una fila se encuentra en ambas relaciones, ésta aparecerá sólo una vez en la relación resultante, obviamente un elemento de un conjunto sólo podrá estar una vez el mismo.

Ej.: Supongamos que tenemos dos relaciones: JUG_ARG(NOMBRE) y JUG_BRA(NOMBRE) que representan a todos los jugadores de Argentina y Brasil respectivamente, puedo crear una tercera relación llamada JUG_ARGBRA(NOMBRE) que reúna tanto a los jugadores de Argentina como a los de Brasil:

JUG_ARGBRA := JUG_ARG ∪ JUG_BRA

El símbolo ":=" indica la operación de asignación para crear la tercer relación JUG_ARGBRA

INTERSECCION:

Permite identificar las filas que son comunes a dos relaciones. Ambas relaciones tienen que ser unión compatibles.

Ej.: Supongamos que tenemos dos relaciones: AUTLINEA(IDAUT) y AUTARBITRO(ID) que representan a todas las autoridades que participaron en partidos como jueces de línea y como árbitros respectivamente, puedo crear una tercera relación llamada AUTLIN_ARB(IDAUT) que contenga a todas las autoridades que han participado como jueces de línea y como árbitros en diferentes partidos:

AUTLIN_ARB := AUTLINEA ∏ AUTARBITRO

DIFERENCIA:

Permite identificar filas que están en una relación y no en la otra. Ambas relaciones tienen que ser unión compatibles y es muy importante el orden en la operación, recuerde que A – B no es lo mismo que B – A.

Ej.: ¿Cuáles son los jugadores que todavía no jugaron?. Suponiendo que tengo dos relaciones JUGJUG(NOMBRE) y JUG(NOMBRE) que indican respectivamente los nombres de todos los jugadores que ya han jugado y los nombres de todos los jugadores:

NOJUGARON := JUG - JUGJUG

PRODUCTO:

Crea el producto cartesiano de dos relaciones. Las relaciones A y B tienen los atributos X, Y y W, Z respectivamente; el producto de A y B es C, por lo tanto, C tendrá los atributos X, Y, W y Z. La cantidad de filas de C será igual a la cantidad de filas de A multiplicadas por las de B, es decir, todas las posibles combinaciones entre las filas de A y B. Se toma la primera fila de A, se completan los valores para c/u de los atributos de A y se la combina con c/u de las filas de B, así sucesivamente hasta llegar a la última fila de A.

Ej.: Supongamos que tenemos dos relaciones: EQUIPO1(NOMBRE) y EQUIPO2(NOMBRE) que representan dos listas de nombres de equipos diferentes, yo quiero realizar una combinatoria entre ambas listas para obtener una nueva relación llamada PARTIDOS_POSIBLES en dónde guardaría los posibles partidos que se podrían realizar:

PARTIDOS_POSIBLES := EQUIPO1 * EQUIPO2

SELECCION:

Se usa para crear una relación a partir de otra relación, seleccionando sólo aquellas filas que satisfacen una condición específica.

Ej.: ¿Cuales son los jugadores de Argentina?:

Dada la relación JUGADOR(**NOMBRE**,NOMBRE_EQ):

SELECT (JUGADOR: NOMBRE_EQ = 'ARGENTINA')

Entre paréntesis primero se indica el nombre de la relación sobre la cual se aplicará la selección y seguido de dos puntos se indica la condición como una expresión condicional a ser evaluada para cada una de las filas de la relación.

PROYECCION:

Debido a que la operación de selección siempre selecciona filas enteras, se necesita alguna manera de eliminar las columnas no deseadas. Si la selección puede pensarse como la manera de eliminar filas no deseadas, entonces la proyección puede pensarse como la manera de eliminar columnas no deseadas.

Ej.: Mostrar los nombres de los jugadores:

Dada la relacion JUGADOR(**NOMBRE**,NOMBRE_EQ), para obtener una nueva relación con sólo la columna NOMBRE:

JUGADOR [NOMBRE]

Tener en cuenta que si hubiese más de un jugador con el mismo nombre, éste aparecería una sola vez, por la misma razón que explicamos en la operación UNION. Cuando se trate de proyectar por más de una columna, se pueden especificar entre corchetes y separadas por comas las distintas columnas a proyectar.

También podemos combinar diferentes operaciones del Algebra Relacional, por Ej.: Mostrar los nombres de los jugadores de Argentina:

SELECT (JUGADOR: NOMBRE_EQ = 'ARGENTINA') [NOMBRE]

REUNION:

La operación de reunión o join se utiliza para conectar datos a través de relaciones, quizás sea la operación más importante en cualquier lenguaje de base de datos. Existen varias versiones: reunión natural (natural join), reunión theta (theta join) y la reunión externa (outer join).

Reunión Natural (natural join): Si tomamos la relación PARTIDO(**FECHA, NOMBRE_EQ1**, NOMBRE_EQ2, IDAUT) podemos observar que en la misma existen una serie de atributos como por ej. IDAUT que permitirían realizar "conexiones lógicas" con otras relaciones, en este caso, con la relación AUTORIDAD(**IDAUT**, NOMBRE). Por Ej.; Si queremos saber el nombre del árbitro que dirigió cada uno de los partidos, debemos reunir las relaciones PARTIDO y AUTORIDAD; ésto puede hacerse en los siguientes pasos:

1. Se realiza un producto cartesiano entre PARTIDO y AUTORIDAD
2. De ésta nueva relación se eliminan todas las filas excepto aquellas en donde PARTIDO.IDAUT = AUTORIDAD.IDAUT
3. Se eliminan todas las columnas que posean mismo nombre e idéntica información, resultando así en la reunión natural (natural join) de PARTIDO y AUTORIDAD.

Relación PARTIDO:

FECHA	NOMBRE_EQ1	NOMBRE_EQ2	IDAUT
01/08/2001	ARGENTINA	BRASIL	1
04/08/2001	ARGENTINA	COREA	3
07/08/2001	ARGENTINA	JAPON	2

Relación AUTORIDAD:

IDAUT	NOMBRE
1	PHILIPAKIS, JOHN

2	SMITH, GEORGE
3	JAMES, KEN
4	CAMERON, PHIL

JOIN(PARTIDO, AUTORIDAD)

FECHA	NOMBRE_EQ1	NOMBRE_EQ2	(*) IDAUT	(**)IDAUT	NOMBRE
01/08/2001	ARGENTINA	BRASIL	1	1	PHILIPAKIS, JOHN
04/08/2001	ARGENTINA	COREA	3	3	JAMES, KEN
07/08/2001	ARGENTINA	JAPON	2	2	SMITH, GEORGE

(*) representa el atributo PARTIDO.IDAUT
(**) representa el atributo AUTORIDAD.IDAUT

En este caso, hay sólo una columna de reunión: PARTIDO.IDAUT que debe coincidir con AUTORIDAD.IDAUT, si bien este es el caso más común, puede ser que se trate de un conjunto de columnas de reunión, en tal caso, todos los valores de c/u de esas columnas deberán coincidir con otro conjunto de columnas equivalentes en la otra relación. Esta es una de las operaciones más poderosas y más utilizadas del Algebra Relacional.

Reunión Theta: Suele utilizarse cuando se pretende realizar una operación de reunión de una relación contra ella misma, es muy común en casos como la relación EMPLEADO que posee un atributo llamado JEFE en donde se guarda el ID del empleado del jefe de este empleado. En estos casos el join se realiza no sobre los atributos en común, sino a través de una condición de join que debemos especificar.

Ej.: sobre la relación PARTIDO, para cada uno de dichos partidos, quiero verificar que no se dé la posibilidad de que, x ej., si Argentina juega con Brasil el 1/8/2001, Brasil no tenga tampoco un partido programado para la misma fecha, nos interesa reportar todos los partidos que estén en esa situación.

PARTIDO1 := PARTIDO
PARTIDO2 := PARTIDO
JOIN (PARTIDO1, PARTIDO2: PARTIDO1.FECHA = PARTIDO2.FECHA AND
PARTIDO1.NOMBRE_EQ1 = PARTIDO2.NOMBRE_EQ2)

Reunión Externa (outer join): Volviendo al ejemplo dado en la Reunión Natural (natural join), supongamos que deseamos listar todos los árbitros junto con los partidos que ellos dirigieron y en caso de que no hayan dirigido aún ningún partido, sólo se mostrarán los datos del árbitro y la información relacionada con el partido quedará con valores nulos. Observe que el árbitro 4, CAMERON, PHIL no existe en la relación final del natural join debido a que él aún no ha dirigido ningún partido. La Reunión Externa nos asegura que cada registro de ambas relaciones sea listado en la relación reunida (join relation) al menos una vez. Primero se ejecuta una Reunión Natural y luego si un registro en una relación no se corresponde con un registro en la otra relación, ése registro se añade a la join relation y las columnas adicionales se llenan con valores nulos, así entonces:

OUTERJOIN (AUTORIDAD, PARTIDO)

(**)IDAUT	NOMBRE	FECHA	NOMBRE_EQ1	NOMBRE_EQ2	(*) IDAUT
1	PHILIPAKIS, JOHN	01/08/2001	ARGENTINA	BRASIL	1
2	SMITH, GEORGE	07/08/2001	ARGENTINA	JAPON	2
3	JAMES, KEN	04/08/2001	ARGENTINA	COREA	3
4	CAMERON, PHIL				

(*) representa el atributo PARTIDO.IDAUT
(**) representa el atributo AUTORIDAD.IDAUT

nos devolverá una fila más que en el caso del natural join. Esto nos permitirá hacer selecciones por valores nulos sobre las columnas que deberán llenarse con dichos valores para determinar cuando se está en esta situación.

DIVISION:

Supongamos que en la relación PARTIDO, tenemos adicionalmente dos atributos nuevos llamados IDLINEA1 y IDLINEA2 cuyos valores se corresponden con la relación AUTORIDAD para indicar que dicha autoridad actuó como juez de línea 1 o juez de línea 2. Si nos interesa saber que AUTORIDAD ha sido árbitro o juez de línea de todos los equipos que han jugado podríamos utilizar la operación de división para solucionar esta consulta, de la siguiente manera:

1. Obtenemos la lista de los equipos que han jugado, llamamos TABEQ a esta relación
2. Obtenemos una lista en donde indicamos código de autoridad y equipo; en dicha lista están incluidas todas las autoridades que actuaron como árbitro o juez de línea en partidos en donde participaba dicho equipo. Esta lista se formará con la lista de autoridades que actuaron como árbitro UNION lista de autoridades que actuaron como línea 1 UNION lista de autoridades que actuaron como línea 2. Llamamos AUTEQ a esta relación.
3. Por último, para resolver esta consulta debemos realizar la división entre ambas relaciones para asegurarnos de que una autoridad esté relacionada con todos los equipos que han jugado.

La solución completa de esta consulta es la siguiente:

TABEQ := PARTIDO [NOMBRE_EQ1] ∪ PARTIDO [NOMBRE_EQ2]
AUTAR := PARTIDO[IDARBITRO,NOMBRE_EQ1] ∪ PARTIDO[IDARBITRO,NOMBRE_EQ2]
AUTL1 := PARTIDO [IDLINEA1,NOMBRE_EQ1] ∪ PARTIDO [IDLINEA1,NOMBRE_EQ2]
AUTL2 := PARTIDO [IDLINEA2,NOMBRE_EQ1] ∪ PARTIDO [IDLINEA2,NOMBRE_EQ2]
AUTEQ := AUTAR ∪ AUTL1 ∪ AUTL2
PRESOLUCION:= AUTEQ / TABEQ
SOLUCION := JOIN(PRESOLUCION,AUTORIDAD) [AUTORIDAD.NOMBRE]

ASIGNACION:

En el ejemplo anterior podemos observar varios ejemplos de asignaciones. Guardo en la relación TABEQ la unión de dos proyecciones, en este caso, PARTIDO [NOMBRE_EQ1] y PARTIDO [NOMBRE_EQ2].