Álgebra Relacional

ORGANIZACIÓN DE ARCHIVOS Y BASE DE DATOS I

LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN

UNIVERSIDAD NACIONAL DE SAN LUIS

AÑO 2006

1. Introducción

Un modelo de bases de datos es una herramienta que nos permite almacenar y administrar información. Para ello, nos provee de los medios necesarios para definir las restricciones asociadas a los datos, especificar las estructuras de almacenamiento utilizadas y realizar consultas y/o actualizaciones en la base. Para especificar las estructuras de datos y restricciones se utiliza un lenguaje de definición de datos (DDL) y para la manipulación se utiliza un lenguaje de manipulación de datos (DML).

Existen diferentes modelos de bases de datos; cada uno utiliza alguna herramienta matemática para la descripción de la base. En los Modelos de Red, Semánticos, Orientados a Objetos se utilizan los grafos; en el Modelo Jerárquico, los árboles; y en el Modelo Relacional, las relaciones.

El *Modelo Relacional* fue presentado por Codd en una publicación en 1970, refiriéndose a un modelo de datos específico, con *Relaciones* como único objeto de tratamiento en el modelo, un *álgebra* como lenguaje de consulta, a la que llamó *Álgebra Relacional* (AR), y ninguna manera de expresar actualizaciones, restricciones y/o cálculos sobre el modelo. Posteriormente, Codd presentó otro lenguaje basado en el *Cálculo de Predicados de la Lógica de Primer Orden*, mostrando que era equivalente en su poder expresivo al AR primeramente presentada; a este segundo lenguaje lo denominó *Cálculo Relacional* (CR).

El AR se considera un *lenguaje procedural*, dado que, como se verá más adelante, el cálculo de una nueva relación se hace especificando las operaciones a realizar y el orden en que deben ser evaluadas, semejante a un lenguaje de programación de tipo procedural. El CR en cambio, es un *lenguaje no procedural*; se deben especificar las condiciones que debe satisfacer la relación resultante, pero de ninguna manera se indica el procedimiento para lograrlo.

En las siguientes publicaciones, Codd presentó la primera restricción de integridad: *Dependencia Funcional*. Luego de este inicio, numerosos investigadores continuaron dedicándose a profundizar en el tema, definiendo más restricciones de integridad, elaborando una rica teoría del Modelo Relacional y creando lenguajes basados en el AR/CR con mayor poder expresivo, con adaptaciones basadas en el paradigma de la programación lógica.

En el Modelo Relacional, las relaciones son almacenadas en estructuras de datos simples tal como la lista secuencial desordenada, comúnmente denominada *tabla*. Los administradores de bases de datos relacionales proveen la facilidad de crear estructuras de acceso [1], en función de especificaciones brindadas por el usuario. Estas estructuras de acceso, por lo general, son *árboles B,* comúnmente denominados *índices*.

[1] Estas estructuras de acceso pueden ser vistas como servicios secundarios sobre la relación. En este caso, no hay servicio primario, sino un espacio primario de nuplas

Resulta sencillo entender la idea subyacente en el modelo, por lo que es accesible a una gran cantidad de usuarios. Por otro lado, el Modelo Relacional ha dado pie a que investigadores de la teoría de bases de datos lo utilicen como marco formal de estudio, dado sus aspectos teóricos, la variedad de herramientas, técnicas y líneas de investigación que provee, que permiten la comprensión y el estudio de otros modelos de datos.

Intuitivamente, la información es representada en tablas, en las cuales cada fila representa un objeto específico o un conjunto de objetos, y todas las filas con un sentido y estructura uniforme son agrupadas en una entidad. Cada tabla almacena una *relación*. En la siguiente sección veremos algunos conceptos preliminares sobre relaciones antes de introducirnos en el tema.

2. Relaciones

Desde un punto de vista matemático, una relación se define del siguiente modo: dados los conjuntos $D_1, D_2, \ldots, D_n$ llamados *dominios*, una relación R es un subconjunto del producto cartesiano de n dominios, R $\subseteq D_1 \times D_2 \times \cdots \times D_n$.

Ejemplo:
$Proveedores \subseteq \#Proveedor \times Nombre\_P \times Categoría\_P \times Ciudad\_P$
$\#Proveedor = Alfa\_numérico^*$
$Nombre\_P = Alfa^*$
$Categoría\_P = \{10, \cdots, 50\}$
$Ciudad\_P = Alfa^*$

Donde : $Alfa$={a, b,. . ., z, á, é, í, ó, ú, ü, ø} y $Alfa\_numérico$ = $Alfa \cup \{0, 1, 2, \ldots, 9\}$
□

En este caso, a los elementos de R los llamamos ***nuplas***, distinguiendo que las componentes de las mismas guardan un orden establecido por el orden en que se dispusieron los dominios en el producto cartesiano. Así, el producto cartesiano $D_1 \times D_2 \times \cdots \times D_n$ es distinto de $D_2 \times D_1 \times \cdots \times D_n$, por lo que, por ejemplo, la nupla $(d_1, d_2, d_3, \ldots, d_n)$ es diferente de $(d_2, d_1, d_3, \ldots, d_n)$

Esto significa que, desde un enfoque matemático, el orden en que se disponen los dominios en una relación, tiene importancia, no así el nombre de los mismos. Incluso, algunos autores, consideran un dominio universal sobre el que se define la relación: R $\subseteq D^n$, donde D puede ser visto como la unión de los D_i, es decir $D = \bigcup_{i=1}^{n} D_i$. Bajo esta perspectiva, los atributos no necesariamente deben llevar nombre, a lo sumo interesa la cantidad de dominios, denominada ***aridad*** de la relación.

Esta perspectiva es denominada por algunos autores como ***no nombrada***.

Desde un punto de vista computacional, podemos relajar la condición de orden, puesto que no resulta relevante: cualquier permutación de dominios representa la misma información.

Para ello, tenemos que distinguir entre ***atributo*** y ***dominio***. El atributo representa un dato particular de un objeto. Lleva nombre propio y un dominio asociado de donde toma sus valores, como por ejemplo: enteros, cadenas de caracteres, valores booleanos, etc.

El conjunto de atributos determina la constitución de la relación, y tal conjunto es llamado ***esquema de la relación***. Al conjunto de elementos que conforman la relación en un determinado instante del tiempo se lo denomina ***instancia de la relación***.

Cada elemento de la instancia de la relación es llamado ***tupla***. Una tupla t es definida como una función o aplicación sobre el conjunto de atributos, donde cada valor que toma la tupla es obtenido de un conjunto de constantes, correspondiente al *dominio del atributo*. Formalmente:

Sean:
$U = \{A_1, A_2, \ldots, A_n\}$
$dom : U \mapsto \{D_1, D_2, \ldots, D_n\}$
$D = \bigcup_{i=1}^{n} D_i$

Una tupla t se define como:

$$t : U \mapsto D \; tal \; que \; t(A_i) = v_s \Rightarrow v_s \in dom(A_i), para \; i = 1 \ldots n$$

Así, una relación se define mediante su esquema, y la instancia se define como un conjunto de aplicaciones o funciones totales, del siguiente modo:

$$r \subseteq \{t/t : U \mapsto D \wedge t(A_i) \in dom(A_i)\}$$

Notar que, hablar de esquema e instancia de relación es análogo al concepto tipo - valor de una variable: la declaración de una variable se corresponde con la especificación del esquema de una relación; y el valor de la variable en un instante de tiempo dado se corresponde con el concepto de instancia de la relación .

Esta perspectiva es denominada por algunos autores como ***nombrada***.

Ejemplo:

Para la relación definida en el primer ejemplo, el esquema de la relación es:

$$Proveedores = \{\#Proveedor,\ Nombre\_P,\ Categoría\_P, Ciudad\_P\}$$

Para cada atributo del esquema tenemos los siguientes dominios:

$dom(\#Proveedor) = Alfa - numérico^*$
$dom(Nombre\_P) = Alfa^*$
$dom(Categoría\_P) = \{10, \ldots, 50\}$
$dom(Ciudad\_P) = Alfa^*$

Una instancia posible de esta relación es $Prov = \{t_1, t_2\}$, donde cada tupla está definida del siguiente modo:

$t_1(\#Proveedor) = S1$	$t_2(\#Proveedor) = S2$
$t_1(Nombre\_P) = Smith$	$t_2(Nombre\_P) = Jones$
$t_1(Categoría\_P) = 20$	$t_2(Categoría\_P) = 10$
$t_1(Ciudad\_P) = Londres$	$t_2(Ciudad\_P) = París$

Otra forma de expresar $Prov$ es:

$$Prov = \{\ \{(\#Proveedor, S1), (Nombre\_P, Smith), (Categoría\_P, 20), (Ciudad\_P, Londres)\},$$
$$\{(\#Proveedor, S2), (Nombre\_P, Jones), (Categoría\_P, 10), (Ciudad\_P, París)\}\ \}$$

□

La correspondencia entre ambas perspectivas es natural, dado que:

1. Una tupla $t = \{(A_1, a_1), (A_2, a_2), \ldots, (A_n, a_n)\}$ definida como función, puede ser vista como una nupla $\langle a_1, a_2, \ldots, a_n\rangle$ considerando un orden sobre los atributos $A_1, A_2, \ldots, A_n$.
2. Dada una nupla $\langle a_1, a_2, \ldots, a_n\rangle$ puede ser interpretada como una función sobre el conjunto de los enteros $\{1, 2, \ldots, n\}$ con $t(i) = a_i$ para $i = 1, 2, \ldots, n$.

Esta correspondencia, nos permite ir de una perspectiva a otra, según la conveniencia eventual. Por ello, es común encontrar en la bibliografía el uso de una u otra, de acuerdo a la elección del autor.

Nos referiremos a *subtupla (subnupla)* cuando necesitemos considerar parte de la misma. La parte que nos interese se corresponderá con un subconjunto de atributos del esquema de la relación. Veamos esto un poco más en detalle.

Desde la perspectiva nombrada, una tupla t es una función, y por consiguiente podemos restringirla a un subconjunto de su dominio[2]. Sea R un esquema de relación, r una instancia con esquema R, $t \in r$, $X \subseteq R$, la **proyección de la tupla** t en X, que denotaremos con $\Pi_X(t)$, se define como :

$$\Pi_X(t) = t_{|X}$$

Observación: cuando X es un conjunto vacío, $\Pi_X(t) = \lambda$ siendo λ el valor nulo.

[2]Recordar lo visto en Estructuras de Datos y Algoritmos sobre restricción de una función a un subconjunto del dominio

Desde una perspectiva no nombrada, una nupla es una secuencia de valores. Sea R una relación de aridad n, y sea m una nupla de R . Utilizaremos la notación $m(i)$ para referirnos a la i-ésima componente de la nupla m; notar que $m(i) \in D_i$. Luego, si $X = \langle i_1, i_2, \ldots, i_k \rangle$ con $i_j \in \{1, 2, \ldots, n\}, i_j < i_{j+1}$, la subnupla $m(X)$, también denominada *X-valor*, se define como:

$$m(X) = (m(i_1), m(i_2), \ldots, m(i_k))$$

Resumiendo tenemos:

No nombrada	***Nombrada***
Nuplas (secuencias ordenadas)	Tuplas (funciones)
Dominios	Atributos(nombre y dominio)
Relación (producto cartesiano y aridad)	Relación (esquema e instancia)

3. El Modelo Relacional

El modelo de base de datos relacional maneja una única componente, que es la relación. **Una base de datos relacional es un conjunto de relaciones** que representan una realidad dada. Por lo tanto, para describir una base, desde laperspectiva no nombrada, debemos especificar cuáles son las relaciones con sus respectivas aridades. Y desde la perspectiva nombrada, interesa distinguir entre el ***esquema de la base de datos***, que especifica la estructura de la misma, formado por un conjunto de esquemas de relaciones; y la ***instancia de la base de datos***, que especifica el contenido actual, formado por un conjunto de instancias de relaciones definidas según los esquemas antes mencionados.

Desde una perspectiva nombrada, los atributos son vistos como parte de la base de datos y pueden ser usados en los lenguajes de consulta y/o actualización y en la especificación de restricciones (dependencias). Y desde una perspectiva no nombrada, los atributos son ignorados, y sólo la aridad es de interés.

En este curso utilizaremos la siguiente notación:

Constantes:	a, b, c
Atributos:	A, B, C
Conjuntos de atributos:	U, V, W, X
Esquemas de relación:	R, S
Instancias de relación:	r, s
Esquemas de base de datos :	σ, μ
Instancias de base de datos:	$\mathcal{A}_\sigma, \mathcal{A}_\mu$
Tuplas:	t, u

Para abreviar la notación, denotaremos con *esq(r)* al esquema de la instancia de relación r, y con *inst(R)* a una instancia del esquema de relación R. Para denotar todas las instancias de relaciones con esquema R, usaremos la notación *Rel(R)*.

4. Definición del Álgebra Relacional

El Álgebra Relacional (AR) se define como una 7-upla $\Re = \langle U, D, dom, \sigma, \mathcal{A}_\sigma, \theta, \beta \rangle$ donde:

1. U es un conjunto de atributos llamado ***Esquema de la Relación Universal***, $U = \{A_1, A_2, \ldots, A_n\}$. A cada atributo A_i le corresponde un dominio D_i de donde toma sus valores.
2. D es el conjunto formado por todos los dominios D_i: $D = \{D_1, D_2, \ldots, D_n\}$.
3. ***dom*** es una función, $dom : U \mapsto D$ tal que $dom(Ai) \in D$.
4. σ es un conjunto de esquemas de relación, $\sigma = \{R_1, R_2, \ldots, R_p\}$, donde $U = \bigcup_{i=1}^{p} R_i$. σ es llamado *descomposición de U* y conforma el ***Esquema de la Base de Datos***.
5. $\mathcal{A}_\sigma$ es un conjunto de instancias de relaciones $\mathcal{A}_\sigma = \{r_1, r_2, \ldots, r_p\}$, donde cada r_i es una instancia de relación con esquema de relación R_i, y conforma la ***Instancia de la Base de Datos***.
6. θ es un conjunto de operadores relacionales que permiten comparar valores del dominio que sean compatibles. Se utlizan los siguientes operadores relacionales: $<, >, \leq, \geq, =, \neq$. Con estos operadores se arman fórmulas según la siguiente sintaxis:

   - Términos:
     Si $A_i \in U$, entonces A_i es término.
     Si $c \in D_j$ para algún j, entonces c es un término.

- Átomos de comparación
  La expresión *Término* $\otimes$ *Término* es un átomo de comparación, donde $\otimes \in \theta$.
- Fórmulas
  Un átomo de comparación es una fórmula.
  Si F_1 y F_2 son fórmulas, entonces: $F_1 \wedge F_2, F_1 \vee F_2, \neg F_1$ son fórmulas.
  Nada más es fórmula

Para interpretar estas fórmulas y obtener un valor de verdad *Verdadero o Falso*, es necesario darle semántica a los operadores de comparación y a los conectivos lógicos. Entonces para:

- Los operadores de comparación: se debe definir un orden total sobre cada dominio de atributo del esquema universal.
- Los operadores lógicos: consideraremos las funciones [3] correspondientes a tales conectivos como comúnmente se conocen.

7. β es un conjunto de operadores del AR cuyos operandos son las relaciones. La evaluación de estos operadores devuelve como resultado una relación. Los operadores algebraicos se clasifican en *Primitivos y No primitivos* Los operadores algebraicos primitivos son seis: Proyección(Π) Selección(σ), Unión ($\cup$), Diferencia (-), Producto cartesiano extendido ($\times$) y Renombre(δ). Los operadores algebraicos no primitivos son varios, pero nosotros sólo veremos los siguientes: Intersección($\cap$), Ensamble Natural ($\bowtie$), Ensamble con selección ($\bowtie_F$) y División($\div$).

Una ***Expresión algebraica E***, es una expresión legalmente formada[4], que define una función desde un conjunto relaciones a una única relación. El esquema y la instancia de la relación resultante dependen de los esquemas y de las instancias corrientes del conjunto de relaciones intervinientes en la expresión.

Denotaremos con *esq(E)* al esquema de la relación resultante de la evaluación de E. Primero definiremos el esquema de E en forma recursiva. Para la definición de su instancia, veremos los operadores en detalle individualmente más adelante. Entonces definimos el *esq(E)* según las siguientes reglas:

1. Si E es r_i entonces $esq(E)$ es el esquema de relación para r_i, $esq(E) = esq(r_i) = R_i$.
2. Si E es una relación constante, entonces el $esq(E)$ es el esquema de la *relación constante*, $esq(E) = esq(r_i) = R_i$.
3. Si $E = (E_1 \cup E_2)$, o $E = (E_1 \cap E_2)$, o $E = (E_1 - E_2)$, o $E = \sigma_F(E_1)$, entonces el $esq(E) = esq(E_1)$.
4. Si $E = \Pi_X E_1$ entonces $esq(E) = X$
5. Si $E = (E_1 \div E_2)$ entonces el $esq(E) = esq(E_1) - esq(E_2)$.
6. Si $E = E_1 \bowtie E_2$ entonces el $esq(E) = esq(E_1) \cup esq(E_2)$.
7. Si $E = E_1 \bowtie_F E_2$ entonces el $esq(E) = esq(E_1)\ más$ [5] $esq(E_2)$
8. Si $E = E_1 \times E_2$ entonces el $esq(E) = esq(E_1)\ mas\ esq(E_2)$, con renombre de atributos.
9. Si $E = \delta_{A_i \leftarrow B_i}(E_1)$ entonces el $esq(E) = esq(E_1) - \{A_i\} \cup \{B_i\}$.

Si E involucra las relaciones $r_1, r_2, \ldots, r_n$ con esquemas $R_1, R_2, \ldots, R_n$, entonces E es la función:

$$E:\ Rel(R_1) \times Rel(R_2) \times \cdots \times Rel(R_n) \rightarrow Rel(esq(E))$$

Cualquier relación resultante de la evaluación de una expresión algebraica tiene asociado un esquema y una instancia de relación, pero no un nombre, por lo que ella no pertenece al conjunto de relaciones que conforman la base de datos. Es una relación temporal, que puede ser utilizada como operando de otras operaciones algebraicas sólo si la expresión que la define es una subexpresión de otra más general que las abarque.

[3] Es decir, las tablas de verdad conocidas
[4] Mediante reglas sintácticas. Ver cada operador
[5] Es *más* porque van todos los atributos participantes; los repetidos se renombran automáticamente

5. Los operadores algebraicos

5.1. Los operadores primitivos

Los operadores algebraicos denominados primitivos, son aquellos para los que no hay una combinación de operadores primitivos que produzcan el mismo resultado.

Un operador es considerado unario si toma una única relación como operando y binario si toma dos relaciones como operandos. Los operadores algebraicos primitivos y unarios son la Proyeccion, Selección y Renombre. Los operadores algebraicos primitivos y binarios son Unión, Diferencia y Producto Cartesiano.

PROYECCIÓN
Sean R un esquema de relación, X un subconjunto de R y r una instancia de relación con esquema R. Denotamos con $\Pi_X(r)$ a la proyección de r sobre el conjunto X. La relación resultante tiene esquema X y su instancia está conformada por el conjunto de tuplas obtenidas proyectando todas las tuplas t de r a X. **Ejemplos:**

$$Proveedores = \{\#Proveedor,\ Nombre\_P,\ Categoría, Ciudad\_P\}$$

$\mathbf{Prov}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

$\mathbf{\Pi_{\#Proveedor,Nombre\_P}(Prov)}$:

# Proveedor	*Nombre_P*
S1	Smith
S2	Jones
S3	Blake
S4	Clark
S5	Adams

$\mathbf{\Pi_{Categoría,Ciudad\_P}(Prov)}$:

Categoría_P	*Ciudad_P*
20	Londres
10	París
30	París
30	Atenas

□

Sea $\Pi_X(r) = s$, luego tenemos los siguientes casos extremales:

- $X = \emptyset$ entonces $esq(s) = \emptyset$ y $s = \emptyset$.
- $X = esq(r) = R$ entonces $esq(s) = esq(r) = R$ y $s = r$.
- $r = \emptyset$ entonces $esq(s) = X$ y $s = \emptyset$.

SELECCIÓN
Sean R un esquema de relación, r una instancia de relación con esquema R y F una fórmula, descripta en términos de los atributos del esquema de la relación. Denotamos con $\sigma_F(r)$ a la selección de tuplas de r respecto de F. La relación resultante tiene esquema R, y su instancia está conformada por el conjunto de tuplas que cumplen con una condición establecida por medio de la evaluación de F.

Ejemplos:

$\mathbf{Prov}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	20	Atenas

$\sigma_{(\mathbf{Categoría\_P=30})}(\mathbf{Prov})$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S3	Blake	30	París

$\sigma_{(\mathbf{Categoría\_P<30 \wedge (Ciudad\_P=Atenas \vee Ciudad\_P=París)})}(\mathbf{Prov})$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S2	Jones	10	París
S5	Adams	20	Atenas

□

Sea $\sigma_F(r) = s$ luego tenemos los siguientes casos extremales:

- Ninguna tupla de r satisface F, entonces $s = \emptyset$
- Todas las tuplas satisfacen la fórmula F , entonces $s = r$

UNIÓN

Sean R un esquema de relación, r y s instancias de relación con esquema R. Denotamos con $r \cup s$ a la unión de tuplas de ambas relaciones. La relación resultante tiene esquema R, y su instancia está conformada por el conjunto de tuplas que pertenecen a r o a s o ambas.

Este operador cumple con las propiedades conmutativa y asociativa.

Ejemplo:

$\mathbf{Prov1}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

$\mathbf{Prov2}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S6	Jones	40	Londres
S2	Jones	10	París
S7	Adams	10	Roma

$\mathbf{Prov1 \cup Prov2}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas
S6	Jones	40	Londres
S7	Adams	10	Roma

□

Sea $r \cup s = v$, luego tenemos los siguientes casos extremales:

- $esq(r) = \emptyset$, entonces $esq(v) = \emptyset$ y $v = \emptyset$.
- $r = \emptyset$ entonces $v = s$.
- $r = s$ entonces $v = r = s$.

DIFERENCIA
Sean R un esquema de relación, r y s instancias de relación con esquema R. Denotamos con $r - s$ a la diferencia de tuplas de ambas relaciones. La relación resultante tiene esquema R, y su instancia está conformada por el conjunto de tuplas que pertenecen a r y no pertenecen a s.

Ejemplo:

Prov1 :

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

Prov2 :

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S6	Jones	40	Londres
S2	Jones	10	París
S7	Adams	10	Roma

Prov1 – Prov2 :

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

□

Sea $r - s = v$, luego tenemos los siguientes casos extremales:

- $esq(r) = \emptyset$ entonces $esq(v) = \emptyset$ y $v = \emptyset$.
- $r = \emptyset$ entonces $v = \emptyset$.
- $r = s$ entonces $v = \emptyset$.

PRODUCTO CARTESIANO
Sean R y S esquemas de relaciones, r y s instancias de relación con esquema R y S respectivamente. Denotamos con $r \times s$ al producto cartesiano *extendido* de ambas relaciones. La relación resultante tiene esquema $R \cup S$ con renombre de atributos repetidos. La instancia de la relación resultante está conformada por el conjunto de tuplas t tal que, hay una tupla t_r en r y hay una tupla t_s en s que cumplen las siguientes restricciones:

- $\Pi_R(t) = t_r$
- $\Pi_S(t) = t_s$

Este operador cumple con las propiedades conmutativa y asociativa.

Ejemplo:

$$Proveedores = \{\#Proveedor,\ Nombre\_P,\ Categoría,\ Ciudad\}$$
$$Proyectos = \{\#Proyecto,\ Nombre,\ Ciudad\}$$

Prov :

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París

Proy :

#Proyecto	*Nombre*	*Ciudad*
J1	Ordenador	París
J2	Perforadora	Roma

Prov $\times$ **Proy** :

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad*	*#Proyecto*	*Nombre*	*Ciudad'*
S1	Smith	20	Londres	J1	Ordenador	París
S2	Jones	10	París	J1	Ordenador	París
S3	Blake	30	París	J1	Ordenador	París
S1	Smith	20	Londres	J2	Perforadora	Roma
S2	Jones	10	París	J2	Perforadora	Roma
S3	Blake	30	París	J2	Perforadora	Roma

Notar que :
$esq(Prov \times Proy) = \{\#Proveedor,\ Nombre\_P,\ Categoría,\ Ciudad,\ \#Proyecto,\ Nombre,\ Ciudad'\}$
□

Sea $r \times s = v$, luego tenemos los siguientes casos extremales:

- $esq(r) = \emptyset$ entonces $esq(v) = esq(s) = S$ y $v = \emptyset$.
- $r = \emptyset$ entonces $v = \emptyset$.

RENOMBRE
Sean R un esquema de relación, r una instancia de relación con esquema R, A y B atributos tales que: $A \in R$ y $B \notin R$. Denotamos con $\delta_{A \leftarrow B}(r)$ al cambio de nombre del atributo A por B. La relación resultante tiene esquema $(R - \{A\} \cup \{B\})$. La instancia de la relación no varía.

Ejemplo:

$\mathbf{Prov}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

$\delta_{\mathbf{Ciudad \leftarrow Ciudad\_P}}(\mathbf{Prov})$

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

□

5.2. Los operadores no primitivos

Los siguientes operadores algebraicos son no primitivos y binarios.

INTERSECCIÓN
Sean R un esquema de relación, r y s instancias de relaciones con esquema R. Denotamos con $r \cap s$ a la intersección de tuplas de ambas relaciones. La relación resultante tiene esquema R, y su instancia está conformada por el conjunto de tuplas que pertenecen a r y a s.

Propiedades: conmutativo, asociativo.

Ejemplo:

$\mathbf{Prov1}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

$\mathbf{Prov2}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S6	Jones	40	Londres
S2	Jones	10	París
S7	Adams	10	Roma

$\mathbf{Prov1} \cap \mathbf{Prov2}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad_P*
S2	Jones	10	París

□

Sea $r \cap s = v$, luego tenemos los siguientes casos extremales:

- $esq(r) = \emptyset$ entonces $esq(v) = \emptyset$ y $v = \emptyset$.
- $r = \emptyset$ entonces $v = \emptyset$.
- $r = s$ entonces $v = r$.

DIVISIÓN

Sean R y S esquemas de relación, $S \subseteq R$, r y s instancias de relación con esquema R y S respectivamente. Denotamos con $r \div s$ a la división de r por s. La relación resultante tiene esquema $(R - S)$. Esta operación realiza una selección de subtuplas de r, considerando que las mismas estén relacionadas con todas las tuplas dadas en s. Es decir, la instancia de la relación está conformada por el conjunto de tuplas t, tal que para toda tupla t_s de s, hay una tupla t_r en r que cumplen las siguientes restricciones:

- $\Pi_{R-S}(t_r) = t$
- $\Pi_S(t_r) = t_s$

Ejemplos:

• Obtener los proyectos provistos por todos los proveedores

$$Pp = \{\#Proveedor, \#Proyecto\}$$
$$Pr = \{\#Proveedor\}$$

pp :

#Proveedor	*#Proyecto*
S1	J1
S1	J4
S1	J2
S1	J3
S2	J1
S2	J3
S3	J1
S3	J3

pr :

#Proveedor
S1
S2
S3

Pp ÷ Pr

#Proyecto
J1
J3

• Obtener todos los proyectos a los que el proveedor $S1$ provee las partes $P1$ y $P2$, y el proveedor $S2$ provee la parte $P3$

$$PPP = \{\#Proveedor, \#Parte, \#Proyecto\}$$
$$R = \{\#Proveedor, \#Parte\}$$

ppp :

#Proveedor	*#Parte*	*#Proyecto*
S1	P1	J1
S1	P1	J4
S1	P2	J1
S1	P1	J2
S1	P4	J1
S1	P3	J3
S1	P2	J2
S2	P3	J1
S2	P3	J2
S2	P3	J3
S2	P3	J4

r :

#Proveedor	*#Parte*
S1	P1
S1	P2
S2	P3

ppp ÷ r

#Proyecto
J1
J2

□

Sea $r \div s = v$, luego tenemos los siguientes casos extremales:

- $esq(r) = esq(s)$ entonces $esq(v) = \emptyset$ y $v = \emptyset$.

- $esq(r) = \emptyset$ entonces $esq(v) = \emptyset$ y $v = \emptyset$.
- $esq(s) = \emptyset$ entonces $esq(v) = esq(r)$ y $v = r$.
- $s = \emptyset$ entonces $v = \Pi_{(R-S)}(r)$.
- $r = \emptyset$ entonces $v = \emptyset$

ENSAMBLE NATURAL

Sean R y S esquemas de relaciones, r y s instancias de relación con esquema R y S respectivamente. Denotamos con $r \bowtie s$ al ensamble natural de r y s. La relación resultante tiene esquema $R \cup S$. Esta operación realiza el ensamblaje de tuplas por atributos comunes con iguales valores. Es decir, la instancia de la relación está conformada por el conjunto de tuplas t tal que hay una tupla t_r en r y hay una tupla t_s en s que cumplen las siguientes restricciones:

- $\Pi_R(t) = t_r$
- $\Pi_S(t) = t_s$
- $\Pi_{(R \cap S)}(t_r) = \Pi_{(R \cap S)}(t_s)$

Este operador cumple con las propiedades conmutativa y asociativa.

Ejemplo:

$$Proveedores = \{\#Proveedor, Nombre\_P, Categoría\_P, Ciudad\}$$
$$Proyectos = \{\#Proyecto, Nombre, Ciudad\}$$

Prov :

#Proveedor	***Nombre_P***	***Categoría_P***	***Ciudad***
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

Proy :

#Proyecto	***Nombre***	***Ciudad***
J1	Ordenador	París
J2	Perforadora	Roma
J3	Impresora	Atenas

Prov $\bowtie$ Proy

#Proveedor	***Nombre_P***	***Categoría_P***	***Ciudad***	***#Proyecto***	***Nombre***
S2	Jones	10	Paris	J1	Ordenador
S3	Blake	30	Paris	J1	Ordenador
S5	Adams	30	Atenas	J3	Impresora

□

Sea $r \bowtie s = v$, luego tenemos los siguientes casos extremales:

- $esq(r) = esq(s)$ entonces $esq(v) = esq(r) = R$ y $v = r \cap s$.
- $esq(r) = \emptyset$ entonces $esq(v) = esq(s) = S$ y $v = \emptyset$.
- $r = \emptyset$ entonces $v = \emptyset$.
- $(R \cap S) = \emptyset$ entonces $v = r \times s$.

ENSAMBLE CON SELECCIÓN

Sean R y S esquemas de relaciones, r y s instancias de relaciones con esquema R y S respectivamente y F una fórmula formada con atributos de R y S. Denotamos con $r \bowtie_F s$ al ensamble de r y s condicionado a F. La relación resultante

tiene esquema $R\ más\ S$ con renombre de atributos repetidos. Esta operación realiza el ensamblaje de tuplas y selecciona las que satisfacen la fórmula F. Es decir, la instancia de la relación está conformada por el conjunto de tuplas t tal que hay una tupla t_r en r y hay una tupla t_s en s que cumplen las siguientes restricciones:

- $\Pi_R(t) = t_r$
- $\Pi_S(t) = t_s$
- t *satisface la fórmula* F

Este operador cumple con las propiedades conmutativa y asociativa.

Ejemplo:

$$Proveedores = \{\#Proveedor, Nombre\_P, Categoría\_P, Ciudad\}$$
$$Proyectos = \{\#Proyecto, Nombre, Ciudad\}$$

Prov :

#Proveedor	***Nombre_P***	***Categoría_P***	***Ciudad***
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

Proy :

#Proyecto	***Nombre***	***Ciudad***
J1	Ordenador	París
J2	Perforadora	Roma

Proy $\bowtie_{(\mathbf{Ciudad} \neq \mathbf{Ciudad'} \wedge \mathbf{Categoría} < \mathbf{30})}$ **Proy**

#Proveedor	***Nombre_P***	***Categoría_P***	***Ciudad***	***#Proyecto***	***Nombre***	***Ciudad'***
S1	Smith	20	Londres	J1	Ordenador	París
S1	Smith	20	Londres	J2	Perforadora	Roma
S2	Jones	10	París	J2	Perforadora	Roma
S4	Clark	20	Londres	J1	Ordenador	París
S4	Clark	20	Londres	J2	Perforadora	Roma

Notar que:
$esq(Prov \bowtie_{Ciudad \neq Ciudad' \wedge Categoría < 30} Proy) =$
$\{\#Proveedor, Nombre\_P, Categoría\_P, Ciudad, \#Proyecto, Nombre, Ciudad'\}$
□

Sea $r \bowtie_F s = v$, luego tenemos los siguientes casos extremales:

- $esq(r) = esq(s)$ entonces $esq(v) = esq(r)\ más\ esq(s)$ completamente renombrado.
- $esq(r) = \emptyset$ entonces $esq(v) = esq(s) = S$.
- $r = \emptyset$ entonces $v = \emptyset$.
- Ninguna tupla satisface F, entonces $v = \emptyset$.
- Todas las tuplas satisfacen F, entonces $v = r \times s$.

6. Expresividad del Álgebra Relacional

El AR es un lenguaje restringido en cuanto a la cantidad de clases de consultas que puede expresar. A continuación mostraremoscuáles son las clases no expresables.

CONSULTAS DE CLAUSURA

Dado un conjunto A incluído en un universo $\mathcal{U}$, y una propiedad $\mathcal{P}$, ***la clausura de A respecto de*** $\mathcal{P}$ se define como el menor conjunto B ($B \subseteq \mathcal{U}$), tal que: $A \subseteq B$ y B cumple la propiedad $\mathcal{P}$.

Dado que las relaciones son conjuntos de nuplas, este concepto es aplicable a relaciones. En este caso, para las relaciones binarias [6], hay dos clausuras que son de particular interés: la *clausura transitiva* y la *clausura reflexo-transitiva* .

La clausura transitiva de una relación $R \subseteq X \times X$, denotada con R^+, es la menor relación que contiene a R y que cumple con la propiedad transitiva. R^+ puede calcularse de la siguiente manera:

$$R^+ = \bigcup_{i=1}^{\infty} R^i$$

donde: $R^1 = R$
y $R^i = R \circ R^{i-1}$

Análogamente, la clausura reflexo-transitiva de una relación $R \subseteq X \times X$, denotada con $R^\star$, es la menor relación que contiene a R y que cumple con las propiedades reflexiva y transitiva. $R^\star$ puede calcularse de la siguiente manera:

$$R^\star = R^+ \cup R^0$$

donde: $R^0 = \{(x, x)/x \in X\}$

Si X es un conjunto finito, la unión de las R^i también lo será , dado que llegará un punto en que ninguna nupla nueva pueda agregarse a la clausura, es decir existirá un i tal que $R^{i-1} = R^i$. Este i dependerá de la instancia de la relación, en el peor de los casos i llegará hasta n, donde $n =\mid X \mid$.

Aho y Ullman demostraron que la **clausura no es expresable ni en $\mathbf{AR}$ ni en $\mathbf{CR}$**, y por lo tanto tampoco lo son las consultas que involucren cálculo de clausura. Esto se debe a que en AR, no hay manera de expresar una iteración que se realiza un número arbitrario de veces.

Al principio, el poder expresivo de un lenguaje de consultas a bases de datos relacionales se medía en función del AR, considerándoselo relacionalmente completo si era capaz de expresar por lo menos la misma clase de consultas que el AR. A partir de la demostración de Aho y Ullman, el concepto de completitud relacional cambió radicalmente, demostrándose en este nuevo sentido que ni el AR ni el CR eran completos. Para definir completitud se desarrolló una amplia y compleja teoría al respecto, la cual es materia de estudio e investigación en bases de datos.

[6] Notar que estamos trabajando desde una perspectiva no nombrada

Ejemplo:
Sea $G = (P, R)$ un 1-digrafo, donde P es el conjunto de vértices, y $R \subseteq P \times P$ es el conjunto de arcos [7]. Bajo la perspectiva del modelo relacional, la base de datos estará conformada por dos relaciones: la relación P de aridad 1 y la relación R de aridad 2. Supongamos que queremos resolver en AR la siguiente consulta: *obtener los pares de vértices para los cuales existe un camino en G.*

Notar que :

$R^0 = \{(x, x)/x \in P\}$ están todos los pares de vértices para los cuales existe un camino de longitud cero.
$R^1 = R$ están todos los pares de vértices para los cuales existe un camino de longitud uno.
$R^2 = R \circ R$ están todos los pares de vértices para los cuales existe un camino de longitud dos.

$\vdots$

$R^i = R \circ R^{i-1}$ están todos los pares de vértices para los cuales existe un camino de longitud i.

Luego en $R^\star = \bigcup_{i=0}^{n} R^i$ están todos los pares de vértices para los que existe un camino.

En AR podemos hacer las sucesivas composiciones de la siguiente manera:

$R^0 : \sigma_{\langle 1^\circ = 2^\circ \rangle}(P \times P)$
$R^1 : R$
$R^2 : \Pi_{1,4}(\sigma_{\langle 2^\circ = 3^\circ \rangle}(R \times R^1))$
$R^3 : \Pi_{1,4}(\sigma_{\langle 2^\circ = 3^\circ \rangle}(R \times R^2))$

$\vdots$

$R^n : \Pi_{1,4}(\sigma_{\langle 2^\circ = 3^\circ \rangle}(R \times R^{n-1}))$

Finalmente : $R^\star : R^0 \cup R^1 \cup R^2 \cup \cdots \cup R^n$
El problema fundamental es que no podemos expresar la unión generalizada como una operación única, sino que debemos hacerla explícitamente para cada n. En general, **no podemos dar una expresión algebraica general para la clausura transitiva de cualquier relación**. Aún considerando que tuviéramos la cardinalidad de P que nos brinda una cota máxima para la longitud de caminos elementales, siempre dependerá de ese número y la expresión variará de acuerdo al mismo.

CONSULTAS DE CARDINALIDAD No es posible contar la cantidad de tuplas que conforman la instancia de una relación, es decir su cardinalidad. No hay manera de seleccionar las tuplas una por una, de modo de poder escribir una expresión que denote un contador de elementos de un conjunto.

CONSULTAS CON OPERACIONES ARITMÉTICAS, MANIPULACIÓN DE CADENAS DE CARACTERES
No es posible realizar cálculos aritméticos entre los valores de atributos de tipo numérico, dado que el AR no está provista de las operaciones aritméticas. Tampoco se pueden realizar operaciones tipo concatenación, subcadena, etc. sobre atributos de tipo carácter.

CONSULTAS DE SELECCIÓN DE LA I_ÉSIMA TUPLA
Cuando se realiza una consulta en AR, obtenemos como respuesta una relación. La relación es un conjunto de tuplas (nuplas), y como tal, solamente tenemos la posibilidad de trabajar con el conjunto completo y no con sus componentes.

El AR no nos provee ningún tipo de operación que permita explícitamente acceder a un elemento de una relación y mucho menos a una componente de una tupla. Puede ocurrir que la relación resultante tenga una sola tupla, pero la misma no podrá ser manipulada como tal, sino que sólo podremos manejar el conjunto unitario conformado por tal tupla.

De igual modo, si en particular nos interesara una subtupla, deberemos tratarla a través del conjunto unitario que contenga a tal subtupla como su único elemento. En estos casos, el tratamiento de "una" tupla en particular puede realizarse mediante las operaciones de ensamble vistas.

Ejemplo:
Obtener la tupla de *Proveedores* correspondiente al Proveedor S3.

[7] Recordar que en un 1-digrafo ,un arco u pueden identificarse por el par de vértices que une

$\mathbf{Prov}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad*
S1	Smith	20	Londres
S2	Jones	10	París
S3	Blake	30	París
S4	Clark	20	Londres
S5	Adams	30	Atenas

$\mathbf{Temp} = \sigma_{\#\mathbf{Proveedor=S3}}(\mathbf{Prov})$

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad*
S3	Blake	30	París

Obtenemos una relación *Temp* conformada por una sola tupla. Luego si quisiéramos manipular la tupla, deberíamos hacerlo a través del conjunto, utilizando los operadores algebraicos. Por ejemplo, si sólo interesara el nombre del proveedor, a nuestra selección anterior deberíamos agregarle una proyección por nombre de proveedor.

$$\mathbf{Temp'} = \mathbf{\Pi_{Nombre\_P}}(\sigma_{\#\mathbf{Proveedor=S3}}(\mathbf{Prov}))$$

Nombre_P
Blake

Pero siempre obtendríamos un conjunto unitario formado por la subtupla correspondiente. Si ese valor de atributo nos hiciera falta para utilizarlo como constante en una consulta, no podríamos usarlo directamente;es distinto expresar *Blake* que $\{$*(Blake)*$\}$.

Si bién no es posible usar *Blake* en forma directa y explícita, sí lo es en forma indirecta a través de un ensamble natural entre *Temp'* y alguna otra relación:

$\mathbf{Prov'}$:

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad*
S2	Jones	10	París
S3	Blake	30	París
S5	Adams	30	Atenas

$\mathbf{Temp''} = \mathbf{Temp'} \bowtie \mathbf{Prov'}$

#Proveedor	*Nombre_P*	*Categoría_P*	*Ciudad*
S3	Blake	30	París

Es incorrecto escribir la consulta anterior como:

$$\sigma_{\#Nombre\_P=Temp'}(Prov')$$

CONSULTAS DE ORDENAMIENTOS No es posible ordenar las tuplas por algún atributo ni ablocarlas por algún criterio. Esto es porque la relación es manejada como un todo.
Δ

BIBLIOGRAFÍA

1. Abiteboul,S; Hull and Vianu, V. ; *Foundations of Databases* ; Addison-Wesley Publishing Company, 1995.

2. Codd, E.F.; *A relational model of data for a large shared data banks* Com of ACM 13/6):377- 387,1970.

3. Date; *Introduction To Database Systems* Vol I,. Addison Wesley, 1981

4. Maier, D; *The theory of relational databases* Computer science press, 1983.

5. Ullman, Jeffrey D.; *Principles of Database and Knowledge Base Systems, Vol I* Computers Science Press, 1988.