Capítulo 2: Modelo Relacional

Database System Concepts, 5th Ed.

Capítulo 2: Modelo Relacional

- Estructura de las BD relacionales
- Operaciones fundamentales del algebra relacional
- Otras operaciones del álgebra relacional
- Operaciones del álgebra relacional extendida
- Valores nulos
- Modificación de la BD

Ejemplo

account_number	*branch_name*	*balance*
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Estructura Básica

- D_1.. Es el dominio
- Formalmente, dado un conjunto D_1, D_2, …. D_n una **relación** *r* es un subconjunto

  D_1 x D_2 x … x D_n

  Así que, una relación es un conjunto de *n*-tuplas (a_1, a_2, …, a_n) donde cada $a_i \in D_i$
- Ejemplo: Si
  - *customer_name* = {Jones, Smith, Curry, Lindsay, …}
    /* Set of all customer names */
  - *customer_street* = {Main, North, Park, …} /* set of all street names*/
  - *customer_city* = {Harrison, Rye, Pittsfield, …} /* set of all city names */

  Entonces *r* = { (Jones, Main, Harrison),
  (Smith, North, Rye),
  (Curry, North, Rye),
  (Lindsay, Park, Pittsfield) }

  es una relación

  customer_name x customer_street x customer_city

Tipo de Atributos

- Cada atributo tiene un nombre
- El conjunto de valores permitidos para cada atributo se llama **dominio**
- Los valores de los atributos son (normalmente) requieren ser **atómicos,** es indivisible.
  - Ejemplo, el valor de un atributo puede ser un número de cuenta pero no puede ser un conjunto de números de cuenta.
- Dominio se dice que es atómico si todos sus miembros son atómicos
- Relación y tupla en vez de tabla y fila
- El valor especial nulo es un miembro de cada dominio
- El valor nulo causa complicaciones en la definición de operaciones
  - Ignoramos el efecto del valor nulo por ahora.

Esquema de la BD
diseño lógido

- $A_1, A_2, \ldots, A_n$ son atributos

- $R = (A_1, A_2, \ldots, A_n)$ es una relación de esquema o esquema de la relación

  Ejemplo:

  Customer_schema = (*customer_name, customer_street, customer_city*)

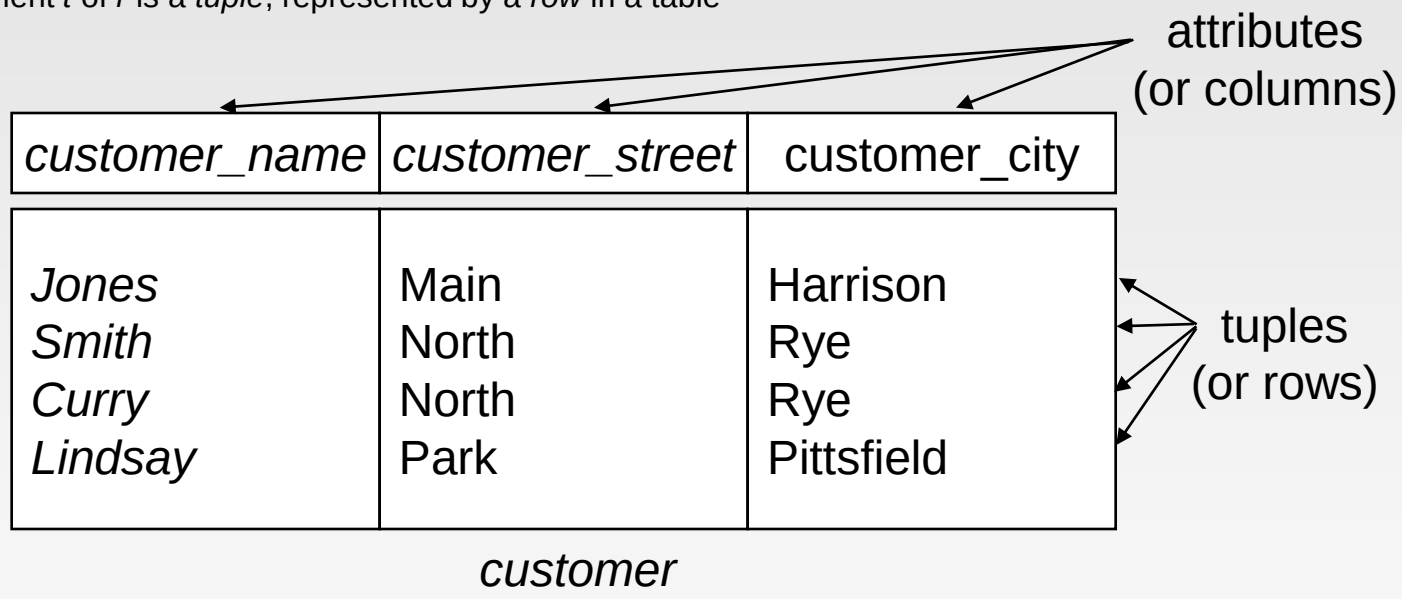
- $r(R)$ denota una relación r en la relación esquema R

  Ejemplo:

  customer (Customer_schema)

Relación
Relation Instance

- Los valores actuales del ejemplar de relación son especificados por una tabla.
- The current values (*relation instance*) of a relation are specified by a table
- Un elemento t de r es una tupla, representada por una fila en la tabla
- An element *t* of *r* is a *tuple*, represented by a *row* in a table



Relaciones desordenadas

- El orden de las tuplas es irrelevante (tuplas pueden almacenarse arbitrarimente
- Ejemplo: relación cuenta con tuplas desordenadas
  - *account* relation with unordered tuples

account_number	*branch_name*	*balance*
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750

Base de Datos

- Una BD consiste de múltiples relaciones
- La información de una empresa se puede descomponer en partes, con cada relación almacemando una parte de la información.

  account o cuenta: almacena la información de la cuentas
  depositor depositante: almacena la información acerca de que cliente posee que cuenta

  customer : almacena la información acerca de los clientes

- Almacenar toda la información como una sola relación tal como:

  bank(*account_number, balance, customer_name*, ..)
  resulta en

  - Repetición de información
    - E.j. si dos clientes tienen la misma cuenta (¿Qué se repite?)
  - Si se necesita usar valores nulos (null)
    - E.j. para representar a un cliente sin cuenta

- La teoría de normalización (cap7) se hace cargo de como diseñar los esquemas relacionales.

La relación cliente

customer_name	*customer_street*	*customer_city*
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Green	Walnut	Stamford
Hayes	Main	Harrison
Johnson	Alma	Palo Alto
Jones	Main	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Putnam	Stamford
Williams	Nassau	Princeton

La Relación Depositante

customer_name	*account_number*
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Claves

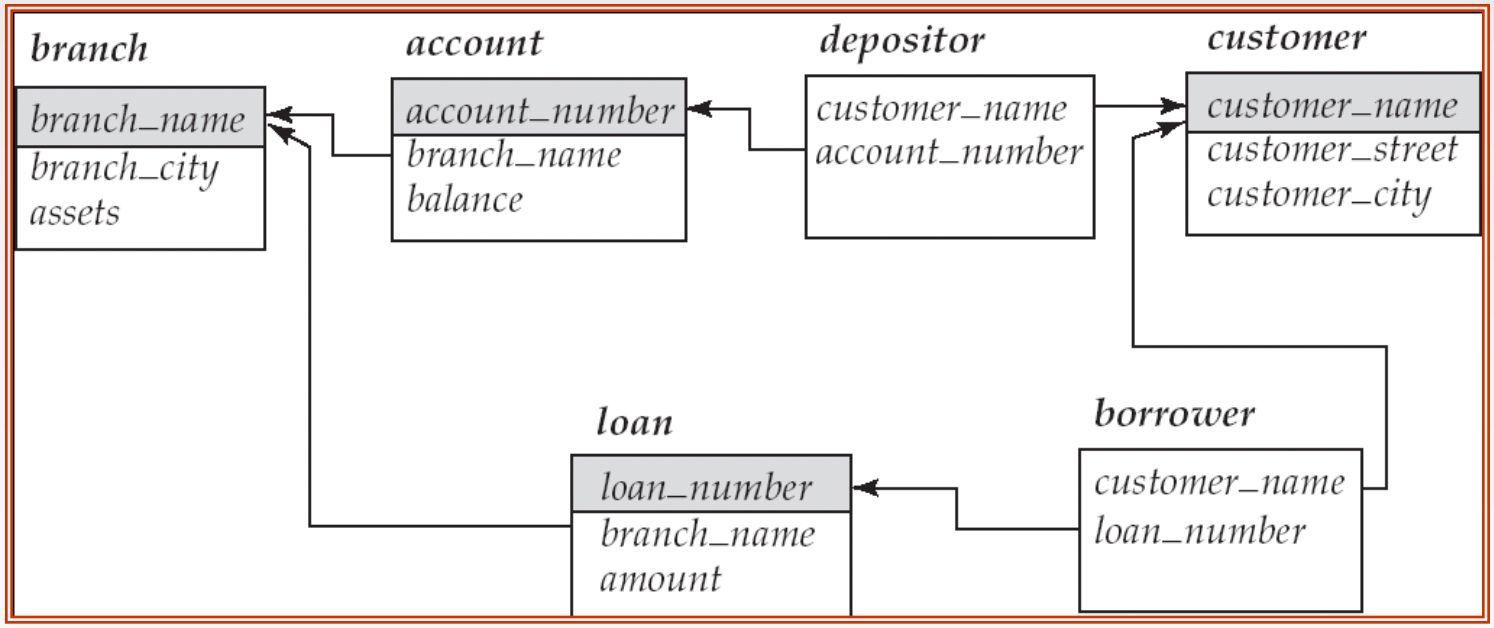
- Sea K $\subseteq$ R
- *K* es una **superclave** de *R* si los valores de *K* son suficientes para identificar una tupla unívocamente en la relación *r(R)*
  - Por "posible *r* " entenderemos una relación *r* que puede existir en la empresa que estamos modelando
  - Ejemplo: {*customer_name, customer_street*} and
    {*customer_name*}
    son superclaves de *Customer*, si no hay dos clientes que posiblemente tengan el mismo nombre
    - En la vida real, una atributo tal como *customer_id* debería ser usado en lugar de *customer_name* para identificar unívocamente a los clientes, pero omitiremos esto para tener ejemplos pequeños y asumimos que los nombres son únicos.

Claves (Cont.)

- *K* es una **clave candidata** si *K* es mínima
  E.j: {*customer_name*} es una clave candidata de *Customer*, desde que esta es una superclave y no un subconjunto de una superclave.
- **Clave primario:** una clave candidata que se escoge como el principal medio de identificación de tuplas dentro de una relación
- **Primary key:** a candidate key chosen as the principal means of identifying tuples within a relation
  - Debes escoger un atributo cuyo valor nunca o rara vez cambia
  - E.j. email es único, pero puede cambiar

Claves Externas
Foreign Keys

- Un esquema de la relación puede tener un atributo que corresponde a la llave primario de otra relación. El atributo es llamado **clave externa (foreign key**.)
  - E.j. *customer_name* y *account_number* atributo de *depositor* son claves externar de *customer* y *account* respectivamente.
  - Only values occurring in the primary key attribute of the **referenced relation** may occur in the foreign key attribute of the **referencing relation.**
- **Diagrama de Esquema**



Lenguaje de Consultas

- Lenguaje por el cual los usuarios solicitan la información de la BD
- Categorías de lenguajes
  - Procedural
  - Non-procedural, or declarativo
- Lenguajes “Puros” :
  - Álgebra relacional
  - Calculo relacional de tuplas
  - Domain relational calculus
- Los lenguajes puros forman la base principal de los lenguajes de consulta más usado.

Relational Algebra

- Procedural language
- Six basic operators
  - select: σ
  - project: Π
  - union: $\cup$
  - set difference: –
  - Cartesian product: x
  - rename: ρ
- The operators take one or two relations as inputs and produce a new relation as a result.

Select Operation – Example

- Relation r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

- $\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10

Select Operation

- Notation: $\sigma_p(r)$
- p is called the **selection predicate**
- Defined as:

$$\sigma_p(\boldsymbol{r}) = \{t \mid t \in r \ \mathbf{y}\ p(t)\}$$

Where p is a formula in propositional calculus consisting of **terms** connected by : $\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**)
Each **term** is one of:

<attribute> *op* <attribute> or <constant>

where *op* is one of: $=, \neq, >, \geq, <, \leq$

- Example of selection:

$$\sigma_{branch\_name=\text{“Perryridge”}}(account)$$

Project Operation – Example

- Relation *r*:

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

$\Pi_{A,C}(r)$

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

Operación Proyección

- Notación:

$$\Pi_{A_1, A_2, \ldots, A_k}(r)$$

  donde A_1, A_2 son nombres de atributos y *r el nombre de una* relación.

- El resultado es definido como la relación de k columnas obtenidas por borrar columnas que no son listadas.
- Las filas duplicadas son removidas del resultado, ya que las relaciones son conjuntos.
- Ejemplo: Para eliminar los nombres de las sucursales (*branch_name) que es un atributo de account*

$$\Pi_{account\_number,\ balance}\ (account)$$

Operación Unión – Ejemplo

- Relaciones *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- r ∪ s:

A	B
α	1
α	2
β	1
β	3

Operación Unión

- Notación: $r \cup s$
- Defino como:

$$r \cup s = \{t \mid t \in r \text{ o } t \in s\}$$

- Para que $r \cup s$ sea valida.

  1. *r, s* deben tener la misma **aridad** (el mismo número de atributos)

  2. Los dominios de los atributos deben ser **compatibles** (ejemplo: 2^{nda} columna de *r* tiene el mismo tipo de valores como la 2^{nda} columna de *s*)

- Ejemplo: Para encontrar los clientes que tienen cuenta o préstamo

$$\Pi_{customer\_name}(depositor) \cup \Pi_{customer\_name}(borrower)$$

Operación de diferencia de conjuntos. Ejemplo

- Relaciones *r*, *s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- *r* – *s*:

A	B
α	1
β	1

Operación de diferencia de conjuntos.

- Notación $r - s$
- Defina como:

$$r - s = \{t \mid t \in r \textbf{ and } t \notin s\}$$

- Las diferencias de conjuntos deben ser entre relaciones **compatibles**.
  - r y s deben tener la misma aridad
  - Dominios de atributos de *r y s deben ser co*mpatibles

Operación producto cartesiano -Ejemplo

- Relaciones *r, s*:

A	*B*
α	1
β	2

r

C	*D*	*E*
α	10	a
β	10	a
β	20	b
γ	10	b

s

- *r* x *s*:

A	*B*	*C*	*D*	*E*
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Operación producto cartesiano

- Notación $r \times s$
- Definido como:

$$r \times s = \{t\, q \mid t \in r \ \textbf{y}\ q \in s\}$$

- Asume que los atributos de r(R) y s(S) son disjuntos. (es decir, $R \cap S = \varnothing$).
- Si los atributos de $r(R)$ y $s(S)$ no son disjuntos, la operación de renombramiento debe ser usada.

Composición de Operaciones

- Podemos construir expresiones usando múltiples operaciones
- Ejemplo: $\sigma_{A=C}(r \times s)$
- $r \times s$

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

- $\sigma_{A=C}(r \times s)$

A	B	C	D	E
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

Operación Renombramiento

- Nos permite dar nombre y referirnos a los resultados de las expresiones de álgebra relacional.
- Nos permite referirnos a una relación con más de un nombre.
- Ejemplo:

$$\rho_X(E)$$

devuelve el resultado de la expresión *E* con el nombre *X*

- Si una expresión del álgebra relacional *E* tiene una aridad *n*, entonces

$$\rho_{X(A_1,A_2,\ldots,A_n)}(E)$$

devuelve el resultado de la expresión *E* con el nombre *X*, y con los atributos renombrados a A_1, A_2, …., A_n.

Ejemplo Bancario

branch (branch_name, branch_city, assets)

customer (customer_name, customer_street, customer_city)

account (account_number, branch_name, balance)

loan (loan_number, branch_name, amount)

depositor (customer_name, account_number)

borrower (customer_name, loan_number)

Ejemplos de Consultas

- Encuentra todos los préstamos (loans) mayores $1200

$$\sigma_{amount > 1200}(loan)$$

- Encuentra el número de préstamo (loan number) para cada préstamo cuyo monto es mayor que $1200

$$\Pi_{loan\_number}(\sigma_{amount > 1200}(loan))$$

- Encuentra los nombres de todos los clientes quienes tengan un préstamo, una cuenta o las dos en el banco (loan, account)

$$\Pi_{customer\_name}(borrower) \cup \Pi_{customer\_name}(depositor)$$

Ejemplos de Consultas

- Encuentra los nombres de todos los clientes quienes tengan un préstamo (loan) en la sucursal Perryridge (branch).

$$\Pi_{customer\_name} (\sigma_{branch\_name=\text{“Perryridge”}} (\sigma_{borrower.loan\_number = loan.loan\_number}(borrower \times loan)))$$

- Encuentra los nombres de todos los clientes quienes tengan un préstamo (loan) en la sucursal Perryridge (branch) pero no tengan cuenta en ninguna sucursal del banco.

$$\Pi_{customer\_name} (\sigma_{branch\_name = \text{“Perryridge”}} (\sigma_{borrower.loan\_number = loan.loan\_number}(\text{borrower x loan}))) - \Pi_{customer\_name}(\text{depositor})$$

Ejemplos de Consultas

- Encuentra los nombres de todos los clientes quienes tengan un préstamo (loan) en la sucursal Perryridge (branch).
  - Consulta 1

$$\Pi_{customer\_name} (\sigma_{branch\_name = \text{“Perryridge”}} (\sigma_{borrower.loan\_number = loan.loan\_number} (borrower \times loan)))$$

  - consulta 2

$$\Pi_{customer\_name}(\sigma_{loan.loan\_number = borrower.loan\_number} ((\sigma_{branch\_name = \text{“Perryridge”}} (loan)) \times borrower))$$

Ejemplos de Consultas

- Encuentra el saldo más grande (account balance)
  - Estrategia:
    - Encuentra todos los balance que no son los más grandes
      - Renombra la relación cuenta (*account)* como *d así podemos comparar cada saldo (*account balance) con los otros
    - Usa la diferencia de conjuntos en estos saldos (account balances) que no se encontraron en el paso anterior. Fig 2.15-2.17
  - La consulta es:

$$\Pi_{balance}(account) - \Pi_{account.balance}$$
$$(\sigma_{account.balance < d.balance} (account \times \rho_{d} (account)))$$

Definición Formal

- Las expresiones fundamentales del álgebra relacional se componen de alguno de los siguientes elementos:
  - Una relación de la BD
  - Una relación constante
- Sean E_1 y E_2 expresiones del álgebra relacional; las siguientes son todas las expresiones del álgebra relacional:
  - $E_1 \cup E_2$
  - $E_1 - E_2$
  - E_1 x E_2
  - $\sigma_p (E_1)$, P es un predicado de atributos de E_1
  - $\prod_s (E_1)$, S es una lista que se compone de algunos de los atributos de E_1
  - $\rho_x (E_1)$, x es el nuevo nombre del resultado de E_1

Operaciones Adicionales

Se definen otras operaciones que no añaden poder al álgebra, pero que simplifican las consultas habituales .

- Intersección de conjuntos
- Reunión natural
- División
- Asignación

Operación Intersección de conjuntos

- Notación: $r \cap s$
- Definido como:
- $r \cap s = \{ t \mid t \in r$ **y** $t \in s \}$
- Asume:
  - r, s tienen la misma *aridad*
  - atributos de r y s son compatibles
- Nota: $r \cap s = r - (r - s)$

Operación Intersección de conjuntos–Ejemplo

- Relación *r, s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cap s$

A	B
α	2

Operación Reunión Natural

- Notación: $r \bowtie s$
- Sean *r* y *s* relaciones de los esquemas *R* y *S* respectivamente. Entonces, $r \bowtie s$ es una relación del esquema $R \cup S$ obtenido como sigue:
  - Considere cada par de tuplas t_r de *r* y t_s de *s*.
  - Si t_r y t_s tiene el mismo valor en cada atributo en $R \cap S$, añada una tupla *t* al resultado, donde
    - *t* tiene el mismo valor como t_r en *r*
    - *t* tiene el mismo valor como t_s en *s*
- Ejemplo:

  $R = (A, B, C, D)$

  $S = (E, B, D)$

  - Resulta el esquema = (A, B, C, D, E)
  - $r \bowtie s$ es definido como:

$$\Pi_{r.A,\ r.B,\ r.C,\ r.D,\ s.E}\ (\sigma_{r.B = s.B \wedge r.D = s.D}\ (r \times s))$$

Operación Reunión Natural– Ejemplo

- Relaciones r, s:

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

r

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	$\in$

s

- r ⋈ s

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

Operación División

- Notación: $r \div s$
- Resulta adecuada para las consultas que incluyen la expresión “para todos”.
- Sean *r* y *s* relaciones en los esquemas *R* y *S* respectivamente donde
  - $R = (A_1, \ldots, A_m, B_1, \ldots, B_n)$
  - $S = (B_1, \ldots, B_n)$

  La resultante de r ÷ s es una relación del esquema

  $R - S = (A_1, \ldots, A_m)$

  $$r \div s = \{ t \mid t \in \prod_{R-S} (r) \wedge \forall u \in s (tu \in r) \}$$

  donde *tu* significa la concatenación de tuplas *t* y *u* para producir una sola tupla

Operación División– Ejemplo

- Relaciones *r, s*:

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
$\in$	6
$\in$	1
β	2

r

B
1
2

s

- *r* ÷ *s*:

A
α
β

Operación División– Otro ejemplo

- Relaciones *r, s*:

A	*B*	*C*	*D*	*E*
α	a	α	a	*1*
α	a	γ	a	*1*
α	a	γ	b	*1*
β	a	γ	a	*1*
β	a	γ	b	*3*
γ	a	γ	a	*1*
γ	a	γ	b	*1*
γ	a	β	b	*1*

r

D	*E*
a	*1*
b	*1*

s

- *r* ÷ *s*:

A	*B*	*C*
α	a	γ
γ	a	γ

Operación División(Cont.)

- Sean r(R) y s(S) relaciones y S contenido en R, es decir todos los atributos del esquema S están en el esquema R. La relación $r \div s$ es una relación del esquema R-S (es decir, del esquema que contiene todos los atributos del esquema R que no están en el esquema S).
- Una tupla t está en $r \div s$ sss:
- 1. t esta en $\Pi_{R-S}(r)$
- 2. Para cada tupla t_s de s hay una tupla t_r de r que cumple:
  - $t_r[S]=t_s[S]$
  - $t_r[R-S] = t$
    - $r \div s = \Pi_{R-S}(r) - \Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$

$\Pi_{R-S}(r)$ proporciona todas las t que cumplen la 1era condición

$\Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$ elimina la t que no cumplen la 2da condición

Operación División(Cont.)

- Propiedad
  - Sea $q = r \div s$
  - Entonces q es la relación más grande que satisface que q x $s \subseteq r$
- Definición en términos del algebra operacional fundamental

  Sea $r(R)$ y $s(S)$ relaciones, y sea $S \subseteq R$

$$r \div s = \Pi_{R-S}(r) - \Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$$

  Para ver porque
  - $\Pi_{R-S,S}(r)$ simplemente reorderna los atributos de r
  - $\Pi_{R-S}(\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$ da las tuplas de t en
  - $\Pi_{R-S}(r)$ tal que algunas tuplas $u \in s$, $tu \notin r$.

Operación Asignación

- La operación asignación ($\leftarrow$) provee una manera conveniente de expresar consultas complejas.
  - Escribe una consulta como un programa secuencial que consta
    - De una serie de asignaciones
    - Seguida por una expresión cuyo valor se muestra como resultado de la consulta
  - La asignación siempre debe hacerse a una variable de relación temporal
- Ejemplo: Escribe $r \div s$ como

$$temp1 \leftarrow \Pi_{R-S}(r)$$
$$temp2 \leftarrow \Pi_{R-S}((temp1 \times s) - \Pi_{R-S,S}(r))$$
$$result = temp1 - temp2$$

  - El resultado de la derecha de la $\leftarrow$ es la asignación a la variable relación en la izquierda de $\leftarrow$.
  - Esta variable relación puede usarse en expresiones posteriores.

Ejemplo de Consultas Bancarias

- Encuentra todos los clientes quienes tiene un préstamo (loan) y una cuenta (account) en el banco.

$$\Pi_{customer\_name}(borrower) \cap \Pi_{customer\_name}(depositor)$$

- Encuentra todos los clientes quienes tiene un préstamo (loan) en el banco y el valor del préstamo (loan amount)

$$\Pi_{customer\_name,\ loan\_number,\ amount}(borrower \bowtie loan)$$

Ejemplo de Consultas Bancarias

- Encuentra todos los clientes quienes tienen al menos una cuenta en las sucursales “Downtown” y Uptown”.
  - Consulta1

$$\Pi_{customer\_name}(\sigma_{branch\_name = \text{“Downtown”}}(depositor \bowtie account)) \cap$$

$$\Pi_{customer\_name}(\sigma_{branch\_name = \text{“Uptown”}}(depositor \bowtie account))$$

  - Consulta2

$$\Pi_{customer\_name,\ branch\_name}(depositor \bowtie account) \div \rho_{temp(branch\_name)}(\{(\text{“Downtown”}), (\text{“Uptown”})\})$$

Note que la consulta 2 usa una relación constante.

Ejemplo de Consultas Bancarias

- Encuentra todos los clientes que tienen cuenta en todas las sucursales que se encuentran en la ciudad Brooklyn.

$$\prod_{customer\_name,\ branch\_name} (depositor \bowtie account)$$
$$\div \prod_{branch\_name} (\sigma_{branch\_city = \text{“Brooklyn”}} (branch))$$

Operaciones del Algebra Relacional Extendida

- Proyección Generalizada
- Funciones de Agregación
- Reunión Externa

Proyección Generalizada

- Extiende la proyección permitiendo que se utilicen funciones aritméticas en la lista de proyección

$$\Pi_{F_1, F_2, \ldots, F_n}(E)$$

- E es cualquier expresión del álgebra relacional
- Cada F_1, F_2, …, F_n *son expresiones aritméticas que incluyen constantes y atributos del esquema E.*
- Supóngase que dado una relación información de crédito Fig. 2.24 *credit_info(customer_name, limit, credit_balance),*
- Encuentra cuanto cada persona tiene de importe disponible (credit available) Fig 2.25:

$$\Pi_{customer\_name,\ limit - credit\_balance}\ (credit\_info)$$

Figure 2.24: The *credit_info* relation

customer_name	*limit*	*credit_balance*
Curry	2000	1750
Hayes	1500	1500
Jones	6000	700
Smith	2000	400

Figure 2.25

customer_name	*credit_available*
Curry	250
Jones	5300
Smith	1600
Hayes	0

Funciones de Agregación y sus operaciones

- **Función Agregación** toman un conjunto de valores y devuelven como resultado un único valor.

  avg: valor de la media
  min: valor mínimo
  max: valor maximo

  sum: suma de valores
  count: número de valores del conjunto

- **Operación Agregar** en algebra relacional

$$_{G_1,G_2,\ldots,G_n}\vartheta_{F_1(A_1),F_2(A_2,\ldots,F_n(A_n)}(E)$$

  E es cualquier expresión del álgebra relacional

  - $G_1, G_2 \ldots, G_n$ es la lista de atributos sobre los que se realiza la agrupación (puede ser vacía)
  - cada F_i es una función de agregación
  - cada A_i *es el nombre de un atributo*

Ejemplo- Operación Agregar

- Relación *r*:

$$_{G_1,G_2,\ldots,G_n}\vartheta_{F_1(A_1),F_2(A_2,\ldots,F_n(A_n)}(E)$$

A	B	C
α	α	7
α	β	7
β	β	3
β	β	10

- $\mathcal{g}_{\text{sum(c)}}(\text{r})$

sum(*c*)
27

Ejemplo- Operación Agregar

- Relación cuenta (*account)* agrupado por sucursal (*branch-name)*:

branch_name	*account_number*	*balance*
Perryridge	A-102	400
Perryridge	A-201	900
Brighton	A-217	750
Brighton	A-215	750
Redwood	A-222	700

$$branch\_name\ \mathcal{G}_{\mathbf{sum}(balance)}\ (account)$$

branch_name	**sum**(*balance*)
Perryridge	1300
Brighton	1500
Redwood	700

Operación Agregar (Cont.)

- Resultado de Agregar no tiene nombre
- Se puede usar la operación renombrar (renombramiento) para darle nombre
  - Por conveniencia, nos permitimos renombrar como parte de la operación agregar.

$$branch\_name\ \mathcal{G}\ \textbf{sum}(balance)\ \textbf{as}\ sum\_balance\ (account)$$

Reunión Externa

- Es una extensión de la operación unión que evita la perdida de información.
- Calcula la unión y luego añade tuplas de una relación que no coinciden con las tuplas de la otra relación dando como resultado la unión.
- Usa valores *nulos*
  - *nulo* significa que un valor es desconocido o no existe
  - Todas las comparaciones que envuelven el *nulo son* falsas por definición.
    - Estudiaremos el significado de comparaciones con el valor nulo mas adelante

Reunion Externa– Ejemplo

- Relación préstamo (*loan)*

loan_number	*branch_name*	*amount*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

- Relación prestatario (*borrower)*

customer_name	*loan_number*
Jones	L-170
Smith	L-230
Hayes	L-155

Reunion Externa– Ejemplo

- Unión
  loan ⋈ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith

- Unión externa por la izquierda

loan ⟕ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*

Reunion Externa– Ejemplo

- Unión externa por la derecha
- *loan* ⟖ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-155	*null*	*null*	Hayes

- Unión externa completa
- *loan* ⟗ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*
L-155	*null*	*null*	Hayes

Valores Nulos

- Es posible para las tuplas tener un valor nulo, denotado por *nulo*, para algunos de sus atributos.
- *Nulo* significa un valor desconocido o no existente.
- El resultado de cualquier operación aritmética que involucre al *nulo* es *nulo.*
- Las funciones de agregación simplemente ignoran los valores nulos (como en SQL)
- Para eliminación de duplicados y agrupación, el nulo se trata como cualquier otro valor, y dos nulos se asume que tienen el mismo valor (como en SQL)

Valores Nulos

- Comparaciones con valores devuelven el valor especial de verdad: desconocido
  - Si falso es usado en lugar de desconocido, entonces no (A<5) no sería equivalente a *A >= 5*
- Se definen las 3 operaciones lógicas tratan el valor lógico *desconocido*:
  - O: (desconocido **o** cierto) = *cierto*,
    (desconocido **o** *falso*) = desconocido
    (*desconocido* **o** *desconocido*) = *desconocido*
  - Y: (*cierto* **y** *desconocido*) = *desconocido,*
    (*falso* **y** *desconocido*) = *falso,*
    (*desconocido* **y** *desconocido*) = *desconocido*
  - NO*:* (**no** *desconocido*) = *desconocido*
  - En SQL "*P* **es** *desconocido*" se evalúa cierto si el predicado devuelve el valor *desconocido*
- El resultado de un predicado seleccionado se trata como falso si se evalúa como desconocido

Modificación de la BD

- El contenido de la BD se puede modificar usando las siguientes operaciones:
  - Borrar
  - Inserción
  - Actualización
- Todas estas operaciones son expresadas usando la operación asignación

Borrar

- Las solicitudes de borrado se expresan básicamente igual que las consultas. Sin embargo, en lugar de mostrar las tuplas al usuario, se eliminan de la BD las tuplas seleccionadas.
- Sólo se pueden borrar tuplas enteras; no se pueden borrar valores de atributos concretos.
- En el algebra relacional los borrados se expresan mediante:

$$r \leftarrow r - E$$

donde r es una relación y E es una consulta del álgebra relacional.

Ejemplos de Borrar

- Borrar todas las cuentas (account) de la sucursal Perryridge (branch).

$$account \leftarrow account - \sigma_{branch\_name = \text{“Perryridge”}}(account)$$

- Borrar todos los préstamos (loan) con importes entre 0 y 50

$$loan \leftarrow loan - \sigma_{amount \geq 0 \text{ y } amount \leq 50}(loan)$$

- Borrar todas las cuentas (accounts) de las sucursales (branches) localizadas en Needham.

$$r_1 \leftarrow \sigma_{branch\_city = \text{“Needham”}}(account \bowtie branch)$$

$$r_2 \leftarrow \Pi_{account\_number,\ branch\_name,\ balance}(r_1)$$

$$r_3 \leftarrow \Pi_{customer\_name,\ account\_number}(r_2 \bowtie depositor)$$

$$account \leftarrow account - r_2$$

$$depositor \leftarrow depositor - r_3$$

Ejemplo Bancario

branch (branch_name, branch_city, assets)

customer (customer_name, customer_street, customer_city)

account (account_number, branch_name, balance)

loan (loan_number, branch_name, amount)

depositor (customer_name, account_number)

borrower (customer_name, loan_number)

Inserción

- Para insertar datos en una relación hay que :
  - Especificar la tupla que se va a insertar
  - Escribir una consulta cuyo resultado sea el conjunto de tuplas que se van a insertar
- El álgebra relacional expresa las inserciones mediante:

$$r \leftarrow r \cup E$$

  donde r es una relación y E es una expresión del álgebra relacional
- La inserción de una sola tupla se expresa haciendo que E sea una relación constante que contiene una tupla

Ejemplos de Inserción

- Supóngase que se desea insertar el hecho de que Smith tiene $1200 en la cuenta (account) A-973 en la sucursal Perryridge (branch)

$$account \leftarrow account \cup \{(\text{“A-973”}, \text{“Perryridge”}, 1200)\}$$

$$depositor \leftarrow depositor \cup \{(\text{“Smith”}, \text{“A-973”})\}$$

- Supóngase que se desea ofrecer una nueva cuenta de ahorro con $200 como regalo a todos los clientes con préstamos concedidos en la sucursal Perryridge. Se usará el número de préstamo como número de esta cuenta de ahorro.

$$r_1 \leftarrow (\sigma_{branch\_name = \text{“Perryridge”}}(borrower \bowtie loan))$$

$$account \leftarrow account \cup \prod_{loan\_number,\ branch\_name,\ 200}(r_1)$$

$$depositor \leftarrow depositor \cup \prod_{customer\_name,\ loan\_number}(r_1)$$

Ejemplo Bancario

branch (branch_name, branch_city, assets)

customer (customer_name, customer_street, customer_city)

account (account_number, branch_name, balance)

loan (loan_number, branch_name, amount)

depositor (customer_name, account_number)

borrower (customer_name, loan_number)

Actualización

- Un mecanismo para modificar un valor de una tupla sin modificar todos los valores de esa tupla.
- Se usa el operador de proyección generalizada para llevar acabo esta tarea

$$r \leftarrow \prod_{F_1, F_2, \ldots, F_l} (r)$$

- Cada F_i es
  - el I ésimo atributo de r, si el I iésimo atributo no se va a actualizar, o,
  - Si el atributo va ser actualizado F_i es una expresión con constantes y atributos de r, que proporciona el nuevo valor del atributo.

Ejemplos de Actualización

- Supóngase que se va hacer el pago de los intereses y hay que incremental todos los saldos (balance) en un 5%

$$account \leftarrow \prod_{account\_number,\ branch\_name,\ balance * 1.05} (account)$$

- Las cuentas con saldos superiores a $10,000 reciben 6% de interes, mientras que el resto recibe un 5%:

$$account \leftarrow \prod_{account\_number,\ branch\_name,\ balance * 1.06} (\sigma_{BAL > 10000} (account)) \cup \prod_{account\_number,\ branch\_name,\ balance * 1.05} (\sigma_{BAL \leq 10000} (account))$$

Tarea
Ejercicios Prácticos
5ta Edición 2.1 al 2.3

Fin del Capítulo 2

Figure 2.3. The *branch* relation

branch_name	*branch_city*	*assets*
Brighton	Brooklyn	7100000
Downtown	Brooklyn	9000000
Mianus	Horseneck	400000
North Town	Rye	3700000
Perryridge	Horseneck	1700000
Pownal	Bennington	300000
Redwood	Palo Alto	2100000
Round Hill	Horseneck	8000000

Figure 2.6: The *loan* relation

loan_number	*branch_name*	*amount*
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

Figure 2.7: The *borrower* relation

customer_name	*loan_number*
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

Figure 2.9

Result of $\sigma_{branch\_name = \text{“Perryridge”}}$ (*loan*)

loan_number	*branch_name*	*amount*
L-15	Perryridge	1500
L-16	Perryridge	1300

Figure 2.10: Loan number and the amount of the loan

loan_number	*amount*
L-11	900
L-14	1500
L-15	1500
L-16	1300
L-17	1000
L-23	2000
L-93	500

Figure 2.11: Names of all customers who have either an account or an loan

customer_name
Adams
Curry
Hayes
Jackson
Jones
Smith
Williams
Lindsay
Johnson
Turner

Figure 2.12:
Customers with an account but no loan

customer_name
Johnson
Lindsay
Turner

Figure 2.13: Result of *borrower* |X| *loan*

customer_name	*borrower.* *loan_number*	*loan.* *loan_number*	*branch_name*	*amount*
Adams	L-16	L-11	Round Hill	900
Adams	L-16	L-14	Downtown	1500
Adams	L-16	L-15	Perryridge	1500
Adams	L-16	L-16	Perryridge	1300
Adams	L-16	L-17	Downtown	1000
Adams	L-16	L-23	Redwood	2000
Adams	L-16	L-93	Mianus	500
Curry	L-93	L-11	Round Hill	900
Curry	L-93	L-14	Downtown	1500
Curry	L-93	L-15	Perryridge	1500
Curry	L-93	L-16	Perryridge	1300
Curry	L-93	L-17	Downtown	1000
Curry	L-93	L-23	Redwood	2000
Curry	L-93	L-93	Mianus	500
Hayes	L-15	L-11		900
Hayes	L-15	L-14		1500
Hayes	L-15	L-15		1500
Hayes	L-15	L-16		1300
Hayes	L-15	L-17		1000
Hayes	L-15	L-23		2000
Hayes	L-15	L-93		500
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
Smith	L-23	L-11	Round Hill	900
Smith	L-23	L-14	Downtown	1500
Smith	L-23	L-15	Perryridge	1500
Smith	L-23	L-16	Perryridge	1300
Smith	L-23	L-17	Downtown	1000
Smith	L-23	L-23	Redwood	2000
Smith	L-23	L-93	Mianus	500
Williams	L-17	L-11	Round Hill	900
Williams	L-17	L-14	Downtown	1500
Williams	L-17	L-15	Perryridge	1500
Williams	L-17	L-16	Perryridge	1300
Williams	L-17	L-17	Downtown	1000
Williams	L-17	L-23	Redwood	2000
Williams	L-17	L-93	Mianus	500

Figure 2.14

customer_name	*borrower. loan_number*	*loan. loan_number*	*branch_name*	*amount*
Adams	L-16	L-15	Perryridge	1500
Adams	L-16	L-16	Perryridge	1300
Curry	L-93	L-15	Perryridge	1500
Curry	L-93	L-16	Perryridge	1300
Hayes	L-15	L-15	Perryridge	1500
Hayes	L-15	L-16	Perryridge	1300
Jackson	L-14	L-15	Perryridge	1500
Jackson	L-14	L-16	Perryridge	1300
Jones	L-17	L-15	Perryridge	1500
Jones	L-17	L-16	Perryridge	1300
Smith	L-11	L-15	Perryridge	1500
Smith	L-11	L-16	Perryridge	1300
Smith	L-23	L-15	Perryridge	1500
Smith	L-23	L-16	Perryridge	1300
Williams	L-17	L-15	Perryridge	1500
Williams	L-17	L-16	Perryridge	1300

Figure 2.15

customer_name
Adams
Hayes

Figure 2.16

balance
500
400
700
750
350

Figure 2.17
Largest account balance in the bank

balance
900

Figure 2.18: Customers who live on the same street and in the same city as Smith

customer_name
Curry
Smith

Figure 2.19: Customers with both an account and a loan at the bank

customer_name
Hayes
Jones
Smith

Figure 2.20

customer_name	*loan_number*	*amount*
Adams	L-16	1300
Curry	L-93	500
Hayes	L-15	1500
Jackson	L-14	1500
Jones	L-17	1000
Smith	L-23	2000
Smith	L-11	900
Williams	L-17	1000

Figure 2.21

branch_name
Brighton
Perryridge

Figure 2.22

branch_name
Brighton
Downtown

Figure 2.23

customer_name	*branch_name*
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

Figure 2.26: The *pt_works* relation

employee_name	*branch_name*	*salary*
Adams	Perryridge	1500
Brown	Perryridge	1300
Gopal	Perryridge	5300
Johnson	Downtown	1500
Loreena	Downtown	1300
Peterson	Downtown	2500
Rao	Austin	1500
Sato	Austin	1600

Figure 2.27
The *pt_works* relation after regrouping

employee_name	*branch_name*	*salary*
Rao	Austin	1500
Sato	Austin	1600
Johnson	Downtown	1500
Loreena	Downtown	1300
Peterson	Downtown	2500
Adams	Perryridge	1500
Brown	Perryridge	1300
Gopal	Perryridge	5300

Figure 2.28

branch_name	*sum of salary*
Austin	3100
Downtown	5300
Perryridge	8100

Figure 2.29

branch_name	*sum_salary*	*max_salary*
Austin	3100	1600
Downtown	5300	2500
Perryridge	8100	5300

Figure 2.30
The *employee* and *ft_works* relations

employee_name	*street*	*city*
Coyote	Toon	Hollywood
Rabbit	Tunnel	Carrotville
Smith	Revolver	Death Valley
Williams	Seaview	Seattle

employee_name	*branch_name*	*salary*
Coyote	Mesa	1500
Rabbit	Mesa	1300
Gates	Redmond	5300
Williams	Redmond	1500

Figure 2.31

employee_name	*street*	*city*	*branch_name*	*salary*
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500

Figure 2.32

employee_name	*street*	*city*	*branch_name*	*salary*
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500
Smith	Revolver	Death Valley	*null*	*null*

Figure 2.33

employee_name	*street*	*city*	*branch_name*	*salary*
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500
Gates	*null*	*null*	Redmond	5300

Figure 2.34

employee_name	*street*	*city*	*branch_name*	*salary*
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500
Smith	Revolver	Death Valley	*null*	*null*
Gates	*null*	*null*	Redmond	5300