Capítulo 2: Modelo entidad-relación
Algebra Relacional

Fundamentos de Bases de datos, 5ª Edición.

Capítulo 2: Modelo entidad-relación

- Estructura de bases de datos relacionales
- Operaciones fundamentales del álgebra relacional
- Otras operaciones del álgebra relacional
- Operaciones del álgebra relacional extendida
- Valores nulos
- Modificación de la base de datos

Ejemplo de una relación

numero_cuenta	*nombre_sucursal*	*saldo*
C-101	Centro	500
C-215	Alta Cordoba	700
C-102	General Paz	400
C-305	Cerro	350
C-201	Recta	900
C-222	Juan B Justo	700
C-217	Urca	750

Estructura básica

- Formalmente, dados los conjuntos D_1, D_2, …. D_n una **relación** r es un subconjunto de

  D_1 x D_2 x … x D_n

  Así, una relación es un conjunto de n-tuplas (v_1, v_2, …, v_n) donde cada

  $v_i \in D_i$

- Ejemplo: Si

  nombre_cliente = {Gómez, López, Santos, Pérez}

  calle_cliente = {Arenal, Mayor, Goya}

  ciudad_cliente = {León, Cerceda, Vigo}

  Entonces r = { (Gómez, Arenal, León),
  (López, Goya, León),
  (Santos, Arenal, Vigo),
  (Pérez, Mayor, Vigo) }

  es una relación sobre

  nombre_cliente x calle_cliente x ciudad_cliente

Atributos

- Cada atributo de una relación tiene un nombre
- El conjunto de valores permitidos para cada atributo se denomina **dominio** del atributo
- Se requiere (normalmente) que los valores de los atributos sean **atómicos**; esto es , indivisibles
  - Nota: los valores de los atributos multivalorados son no atómicos
  - Nota: los valores de los atributos compuestos son no atómicos
- El valor especial *null* es miembro de todos los dominios
- El valor null crea complicaciones en la definición de algunas operaciones
  - Ignoraremos el efecto de los valores null en una primera presentación y consideraremos sus efectos con posterioridad

Esquema de la relación

- A_1, A_2, …, A_n son atributos

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

  Ejemplo:

  Esquema_cliente = (*nombre_cliente, calle_cliente, ciudad_cliente*)

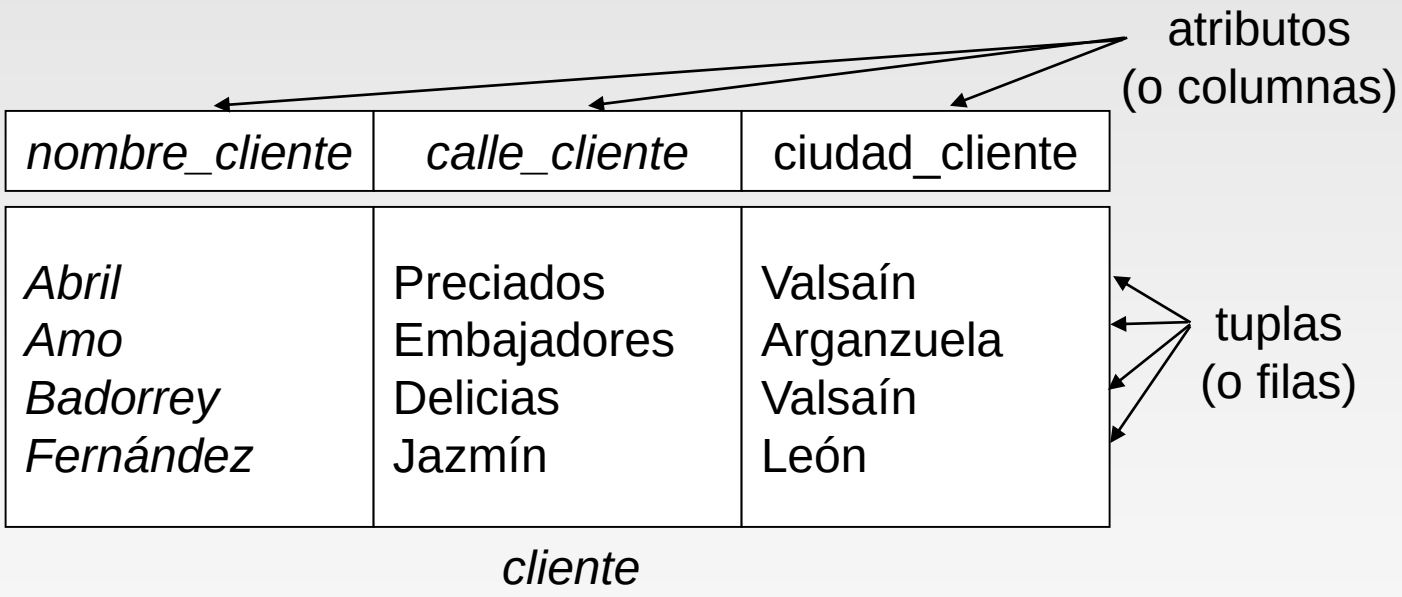
- *r*(*R*) es una *relación* en el *esquema de la relación R*

  Ejemplo:

  cliente (Esquema_cliente)

Instancia de relación

- Los valores actuales (*instancia de relación*) de una relación se especifican en una tabla
- Un elemento *t* de *r* es una *tupla*, representada por una *fila* en una tabla



Las relaciones no son ordenadas

- El orden de la tuplas es irrelevante (las tuplas se pueden guardar con un orden arbitrario)
- Ejemplo: la relación *cuenta* con las tuplas desordenadas

numero_cuenta	*nombre_sucursal*	*saldo*
C-101	Centro	500
C-215	Alta Cordoba	700
C-102	General Paz	400
C-305	Cerro	350
C-201	Recta	900
C-222	Juan B Justo	700
C-217	Urca	750

Base de datos

- Una base de datos consta de múltiples relaciones
- La información sobre una empresa se divide en partes, en la que cada relación almacena una parte de la información

  cuenta : almacena la información de cuentas
  impositor : almacena información de los clientes con sus cuentas
  cliente : almacena información sobre las cuentas

- Almacena toda la información como una única relación tal como
  banco(*númro_cuenta, saldo, nombre_cliente*, …)
  resultando
  - Repetición de información (ej., dos clientes comparten una cuenta)
  - Necesidad de valores null (ej., representar un cliente sin ninguna cuenta)
- La teoría de normalización (Capítulo 7) trata de cómo diseñar esquemas de relación

La relación *proveedores*

S#	SNOMBRE	SITUACION	CIUDAD
S1	SALAZAR	20	LONDRES
S2	JAIMES	10	PARIS
S3	BERNAL	30	PARIS
S4	CORONA	20	LONDRES
S5	ALDANA	30	ATENAS

La relación *partes*

P#	PNOMBRE	COLOR	PESO	CIUDAD
P1	TUERCA	ROJO	12	LONDRES
P2	PERNO	VERDE	17	PARIS
P3	BIRLO	AZUL	17	ROMA
P4	BIRLO	ROJO	14	LONDRES
P5	LEVA	AZUL	12	PARIS
P6	ENGRANAJE	ROJO	19	LONDRES

La relación *envios*

S#	P#	CANT
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	400
S4	P2	200
S4	P4	300
S4	P5	400

Claves

- Sea K $\subseteq$ R
- *K* es una **superclave** de *R* si los valores de *K* son suficientes para identificar una tupla única de entre todas de la relación *r(R)*
  - Por las “posibles *r* ” quiere decir una relación *r* que pudiera existir en la empresa que se está modelando.
  - Ejemplo: {*nombre_cliente, calle_cliente*} y

    {*nombre_cliente*}

    son ambas superclaves de *Cliente*, si no hubiese dos posibles clientes que pudiesen tener el mismo nombre.
  - El concepto de superclave no es suficiente para nuestro trabajo
- *K* es una **clave candidata** si *K* es minima

  Ejemplo: {*nombre_cliente*} es una clave candidata para *Cliente*, ya que es una superclave (suponiendo que no pueda haber dos clientes con el mismo nombre), y ningún subconjunto de ella es una superclave.

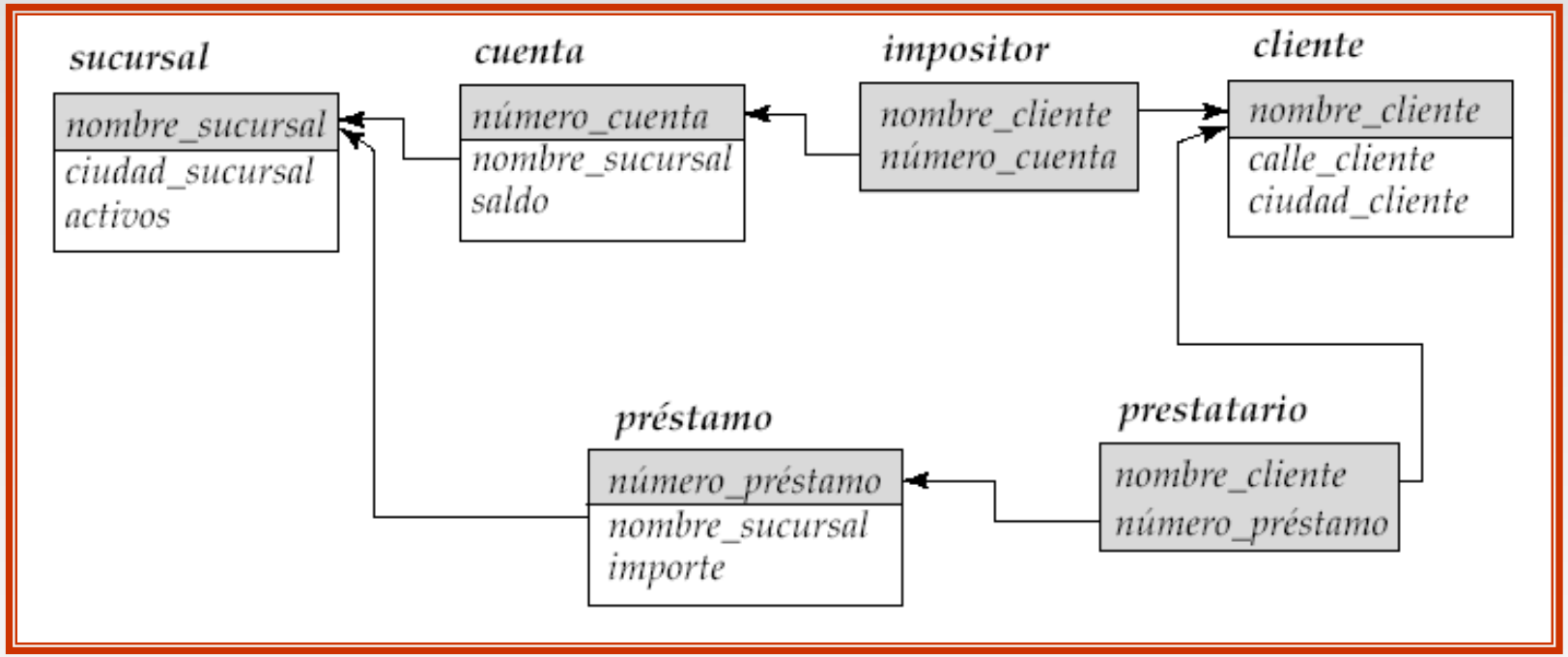
Claves (cont.)

- *R puede contener varias claves candidatas* K_1, K_2, K_3
  Las claves candidatas deben escogerse con cuidado.
  El nombre de una persona no es suficiente, ya que puede haber varias personas con el mismo nombre.
- Se denomina **Clave primaria** a una clave candidata que ha elegido el diseñador de la base de datos como medio principal para la identificación de las tuplas de una relación.
- Las claves candidatas, primaria y superclave son propiedades de toda la relación, no de una tupla.
- La selección de una clave representa una restricción de la empresa del mundo real que se está modelando.
- La clave primaria debe escogerse de manera que los valores de sus atributos no se modifiquen nunca, o muy rara vez.
  Por ej. el atributo domicilio no debe formar parte de una clave primaria.

Claves (cont.)

- Formalmente, sea *R* el esquema de una relación. Si se dice que un subconjunto *C* de *R* es una *superclave* de *R*, se restringe la consideración a las relaciones *r(R)* en las que no hay dos tuplas diferentes que tengan los mismos valores en todos los atributos de C. Es decir, si t_1 y t_2 están en *r* y $t_1 <> t_2$, entonces $t_1[C] <> t_2[C]$.
- El esquema de una relación, por ejemplo r_1, puede incluir entre sus atributos la clave primaria de otro esquema de relación, por ejemplo r_2. Este atributo se denomina **clave externa** de r_1, que hace referencia a r_2. La relación r_1 también se denomina **relación referenciante** de la dependencia de clave externa y r_2 se denomina **relación referenciada** de la clave externa.
- El esquema de la base de datos, junto con las dependencias de clave primaria y externa, se puede mostrar gráficamente mediante **diagramas de esquema.**

Diagrama de esquema



Lenguajes de consulta

- Lenguaje en el que un usuario solicita información de la base de datos Estos lenguajes suelen ser de un nivel superior de los lenguajes de programación habitual.
- Categorías de los lenguajes
  - Procedimental
    El usuario indica al sistema que lleve a cabo una serie de operaciones en la base de datos para obtener el resultado esperado
  - No procedimental, o declarativo
    El usuario describe la información deseada sin dar un procedimiento para obtenerla
- Lenguajes “puros”:
  - Álgebra relacional
  - Cálculo relacional de tuplas
  - Cálculo relacional de dominios
- Los lenguajes puros son la base de los lenguajes de consulta que utilizan los usuarios.

Algebra relacional

- Lenguaje procedimental
- Seis operadores básicos
  - selección: σ
  - proyección: Π
  - unión: $\cup$
  - diferencia de conjuntos: –
  - producto cartesiano: x
  - renombramiento: ρ
- Los operadores tienen como operandos una o dos relaciones y producen como resultado una nueva relación.

Operación selección – Ejemplo

- Relación 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

Operación selección

- Notación: $\sigma_p(r)$
- p se denomina **predicado de selección**
- Se define como:

$$\sigma_p(\boldsymbol{r}) = \{t \mid t \in r \textbf{ and } p(t)\}$$

Donde p es una fórmula del cálculo proposicional que consta de **términos** conectados por : $\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**)
Cada **término** puede ser uno de :

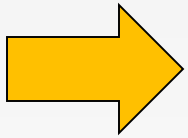
<atributo> *op* <atributo> o <constante>

donde *op* es una de las siguientes: $=, \neq, >, \geq, <, \leq$

- Ejemplo de selección:

$$\sigma_{snombre=\text{“SALAZAR”}}(\text{proveedores})$$

S#	SNOMBRE	SITUACION	CIUDAD
S1	SALAZAR	20	LONDRES
S2	JAIMES	10	PARIS
S3	BERNAL	30	PARIS
S4	CORONA	20	LONDRES
S5	ALDANA	30	ATENAS



S#	SNOMBRE	SITUACION	CIUDAD
S1	SALAZAR	20	LONDRES

Operación proyección – Ejemplo

- Relación *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:

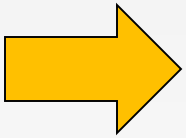
$$\Pi_{A_1, A_2, \ldots, A_k}(r)$$

  donde A_1, A_2 son nombres de atributos y r es un nombre de relación.

- El resultado se define como una relación de k columnas que se obtiene eliminando las columnas que no están en la lista.
- Las filas duplicadas se eliminan del resultado, ya que las relaciones son cunjuntos
- Ejemplo: Para eliminar los atributos *snum* y *snombre* de *proveedores*

$$\Pi_{situacion,\ ciudad}(proveedores)$$

S#	SNOMBRE	SITUACION	CIUDAD
S1	SALAZAR	20	LONDRES
S2	JAIMES	10	PARIS
S3	BERNAL	30	PARIS
S4	CORONA	20	LONDRES
S5	ALDANA	30	ATENAS



SITUACION	CIUDAD
20	LONDRES
10	PARIS
30	PARIS
30	ATENAS

Operación unión – Ejemplo

- Relaciones *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- r $\cup$ s:

A	B
α	1
α	2
β	1
β	3

Operación unión

- Notación: $r \cup s$
- Se define como:

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

- Para todo $r \cup s$ válido.

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

  2. Los dominios de los atributos deben ser **compatibles** (ejemplo: 2ª columna de *r* trata el mismo tipo de valores que la 2ª columna de *s*)

- Ejemplo: para buscar a todos los proveedores y las partes

$$\Pi_{snombre}(proveedores) \cup \Pi_{pnombre}(partes)$$

SNOMBRE
SALAZAR
JAIMES
BERNAL
CORONA
ALDANA
TUERCA
PERNO
BIRLO
LEVA
ENGRANAJE

Operación 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 diferencia de conjuntos

- Notación $r - s$
- Definido como:

$$r - s = \{t \mid t \in r \text{ y } t \notin s\}$$

- Las diferencias de conjuntos deben realizarse entre relaciones **compatibles**.
  - r y s deben tener la **misma** cardinalidad
  - los dominios de los atributos de r y s deben ser compatibles

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*:

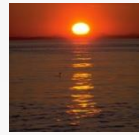
A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Operación producto cartesiano

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

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

- Supone 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, se debe utilizar el renombramiento.



Composición de operaciones

- Permite obtener expresiones que utilizan operaciones múltiples
- 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 nombrar y por lo tanto referirnos a los resultados de expresiones de álgebra relacional.
- Nos permite referirnos a una relación por más de un nombre.

  Ejemplo:

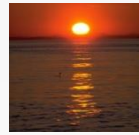
$$\rho_X(E)$$

  devuelve la expresión *E* bajo el nombre *X*

  Si una expresión de álgebra relacional *E* tiene cardinalidad *n*, entonces

$$\rho_{X(A1, A2, \ldots, An)}(E)$$

  devuelve el resultado de la expresión *E* bajo el nombre *X*, y con los atributos renombrados como *A1, A2, …., An*.



Definición formal

- Una expresión básica en el álgebra relacional se compone de una de las siguientes:
  - Una relación en la base de datos
  - Una relación constante
- Sean E_1 y E_2 expresiones de álgebra relacional. Todas las siguientes son 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 alguno de los atributos de E_1
  - $\rho_x (E_1)$, x es el nuevo nombre del resultado de E_1

Operaciones adicionales

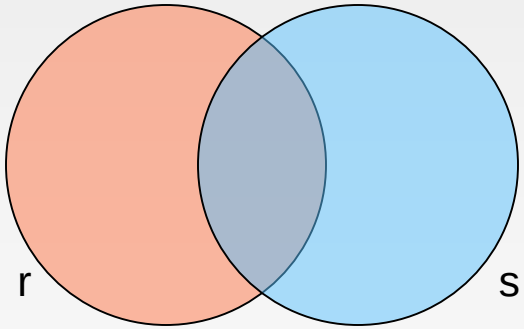
Definimos operaciones adicionales que no añaden potencia al álgebra relacional, pero que simplifican las consultas habituales.

- Intersección de conjuntos
- Reunión natural
- División (salteamos)
- Asignación

Operación intersección de conjuntos

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

  $r \cap s = \{ t \mid t \in r \textbf{ y } t \in s \}$
- Suponiendo:
  - r, s tienen la *misma cardinalidad*
  - los 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 en el esquema $R \cup S$ que se obtiene de la siguiente forma:
  - Considerando cada par de tuplas t_r de *r* y t_s de *s*.
  - Si t_r y t_s tienen el mismo valor en cada uno de los atributos de $R \cap S$, se añade una tupla *t* al resultado, donde
    - *t* tiene el mismo valor que t_r en *r*
    - *t* tiene el mismo valor que t_s en *s*
- Ejemplo:

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

  $S = (E, B, D)$
- Esquema resultante = (A, B, C, D, E)
- $r \bowtie s$ se define como:

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

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 asignación

- La operación asignación (←) proporciona una forma conveniente de expresar consultas complejas
  - Escribir una consulta como un programa secuencial compuesto de
    - una series de asignaciones
    - seguidas por una expresión cuyo valor se muestra como resultado de la consulta.
  - La asignación siempre debe realizarse a una variable de relación temporal.
- Ejemplo: Escribir $r \div s$ 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 a la derecha de ← se asigna a la variable de relación situada a la izquierda de ←.
  - Esta variable puede utilizarse en expresiones posteriores.

Operaciones de álgebra relacional extendida

- Proyección generalizada
- Funciones de agregación
- Reunión externa (salteamos)

Proyección generalizada

- Extiende la operación de proyección al permitir que las funciones aritméticas se utilicen en la lista de proyección.

$$\Pi_{F1, F2, \ldots, Fn}(E)$$

- E es cualquier expresión de álgebra relacional
- $F_1, F_2, \ldots, F_n$ son expresiones aritméticas que implican constantes y atributos en el esquema de E.
- Dada la relación *información-crédito(nombre-cliente, límite, saldo-crédito),* averiguar cuánto más puede gastar cada persona:

$$\Pi_{\textit{nombre-cliente, límite – saldo-crédito}} (\textit{información-crédito})$$

Funciones de agregación y operaciones

- La **función de agregación** toma una colección de valores y devuelve como resultado un valor simple.

  avg: valor medio
  min: valor mínimo
  max: valor máximo
  sum: suma de valores
  count: número de valores

- La **operación de agregación** en el álgebra 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 de álgebra relacional
  - G_1, G_2 …, G_n es una lista de atributos sobre los que se agrupa (puede estar vacía)
  - Cada F_i es una función agregada
  - Cada A_i es un nombre de atributo

Operación de agregación – Ejemplo

- Relación *r*:

A	*B*	*C*
α	α	7
α	β	7
β	β	3
β	β	10

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

sum(*c*)
27

Operación de agregación – Ejemplo

- Relación *cuenta* agrupada por *nombre-sucursal*:

nombre-sucursal	*número-cuenta*	*saldo*
Navacerrada	A-102	400
Navacerrada	A-201	900
Barcelona	A-217	750
Barcelona	A-215	750
Reus	A-222	700

$$\textit{nombre-sucursal}\ \mathcal{g}_{\ \mathbf{sum}(\textit{saldo})}\ (\textit{cuenta})$$

nombre-sucursal	*saldo*
Navacerrada	1300
Barcelona	1500
Reus	700

Funciones de agregación (cont.)

- El resultado de la agregación no tiene nombre
  - Puede utilizarse la operación renombrar para darle un nombre
  - Por conveniencia, permitimos renombrarlo como parte de la operación de agregación

$$\textit{nombre-sucursal}\ \mathcal{G}\ \textbf{sum}(\textit{saldo})\ \textbf{as}\ \textit{suma-saldo}\ (\textit{cuenta})$$

Valores nulos

- Es posible que las tuplas tengan valores nulos para algunos de sus atributos, denotados como *null*
- *null* significa valores desconocidos o no existentes.
- El resultado de una expresión aritmética que contenga un *null* es *null.*
- Las funciones de agregación simplemente ignoran los valores nulos
- Para eliminar duplicados y hacer grupos, los nulos se tratan como cualquier otro valor. Se supone que los valores representados por los nulos son los mismos

Valores nulos

- Las comparaciones con valores nulos devuelven el valor especial cierto *desconocido*
  - Si se utilizó *falso* en lugar de *desconocido*, entonces *no (A < 5)* no sería equivalente a *A >= 5*
- Tres valores lógicos utilizando el valor cierto *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 como cierto si el predicado *P* se evalúa como *desconocido*
- El resultado de la selección de predicado se trata como *falso* si el predicado se evalúa como *desconocido*

Modificación de la base de datos

- El contenido de la base de datos puede ser modificado utilizando las siguientes operaciones:
  - Borrado
  - Inserción
  - Actualización
- Todas estas operaciones se expresan utilizando la operación de asignación.

Borrado

- Una petición de borrado se expresa de forma similar a una consulta, salvo que en lugar de mostrar las tuplas al usuario, el borrado elimina las tuplas seleccionadas de la base de datos.
- Solo se pueden borrar tuplas completas, no se pueden borrar valores solo en determinados atributos
- Un borrado se expresa en álgebra relacional como:

$$r \leftarrow r - E$$

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

Ejemplos de borrado

- Borrar todos los registros de cuentas en la sucursal de Navacerrada.

  $cuenta \leftarrow cuenta - \sigma_{nombre\text{-}sucursal = \text{“Navacerrada”}} (cuenta)$

- Borrar todos los registros de préstamos con cantidades entre 0 y 50

  $préstamo \leftarrow préstamo - \sigma_{cantidad \geq 0 \text{ y } cantidad \leq 50} (préstamo)$

- Borrar todas las cuentas en las sucursales situadas en Getafe.

  $r1 \leftarrow \sigma_{ciudad\text{-}sucursal} = \text{“Getafe”} (cuenta \bowtie sucursal)$

  $r2 \leftarrow \Pi_{nombre\text{-}sucursal,\ número\text{-}cuenta,\ saldo} (r1)$

  $r3 \leftarrow \Pi_{nombre\text{-}cliente,\ número\text{-}cuenta} (r2 \bowtie impositor)$

  $cuenta \leftarrow cuenta - r2$

  $impositor \leftarrow impositor - r3$

Inserción

- Para insertar datos en una relación, podemos:
  - Especificar una tupla para insertar
  - Escribir una consulta cuyo resultado sea un conjunto de tuplas para insertar
- En álgebra relacional, una inserción se expresa como:

$$r \leftarrow r \cup E$$

  donde r es una relación y E es una expresión de álgebra relacional.
- La inserción de una tupla simple se expresa tomando E como una relación constante que contiene una tupla.

Ejemplos de inserción

- Insertar información en la base de datos especificando que Pérez tiene 1200€ en la cuenta C-973 de la sucursal de Navacerrada.

$$cuenta \leftarrow cuenta \cup \{(\text{“Navacerrada”}, \text{C-973}, 1200)\}$$

$$\text{impositor} \leftarrow impositor \cup \{(\text{“Pérez”}, \text{C-973})\}$$

- Ofrecer una cuenta de ahorro de 200€ como regalo a todos los clientes con préstamos en la sucursal de Navacerrada. Sea el número de préstamo el utilizado como número de la nueva cuenta de ahorro.

$$r_1 \leftarrow (\sigma_{nombre\text{-}sucursal = \text{“Navacerrada”}} (prestatario \bowtie préstamo))$$

$$cuenta \leftarrow cuenta \cup \prod_{nombre\text{-}sucursal,\ número\text{-}cuenta,200} (r_1)$$

$$\text{impositor} \leftarrow impositor \cup \prod_{nombre\text{-}cliente,\ número\text{-}préstamo,} (r_1)$$

Actualización

- Un mecanismo para cambiar un valor en una tupla sin cambiar *todos* los valores de la tupla
- Utilizar la operación de proyección generalizada para realizar esta tarea

$$r \leftarrow \prod_{F1,\, F2,\, \ldots,\, Fl,} (r)$$

- Cada *F*, es o
  - el *iesimo* atributo de *r*, si el *iesimo* atributo no está actualizado, o
  - si el atributo tiene que ser actualizado F_i es una expresión que contiene solo constantes y los atributos de *r*, que da el nuevo valor para el atributo

Ejemplos de actualización

- Realizar pagos de intereses aumentando todos los saldos un 5 por ciento.

$$cuenta \leftarrow \prod_{número\_cuenta,\ nombre\_sucursal,\ saldo * 1.05} (cuenta)$$

- Pagar a todas las cuentas con saldos por encima de 10,000€ un 6 por ciento de interés y un 5 por ciento a todas las demás cuentas

$$cuenta \leftarrow \prod_{número\_cuenta,\ nombre\_sucursal,\ saldo * 1.06} (\sigma_{SAL > 10000} (cuenta)) \cup \prod_{número\_cuenta,\ nombre\_sucursal,\ saldo * 1.05} (\sigma_{SAL \leq 10000} (cuenta))$$

Fin del capítulo 2