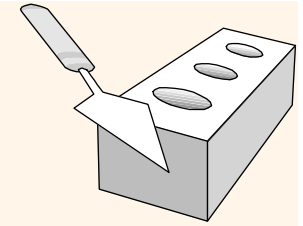


Algebra Relacional

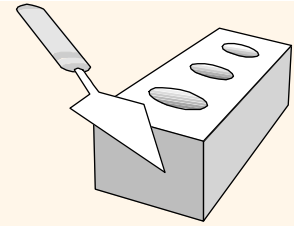
Capítulo 4, Parte A



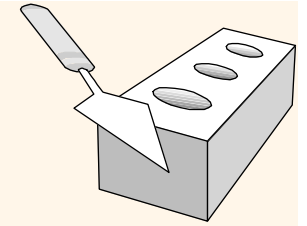
Relational Query Languages

- ❖ Lenguajes de consulta: Permiten manipulación y **recuperación de datos** de una base de datos.
- ❖ El modelo relacional soporta lenguajes de consulta simples pero poderosos:
 - Sólido fundamento formal basado en lógica.
 - Permite mucha optimización.
- ❖ Lenguajes de consulta **!=** lenguajes programación!
 - LCs no necesitan ser “Turing completos”.
 - LCs no están diseñados para cálculos complejos.
 - LCs soportan acceso simple y eficiente a grandes volúmenes de datos.

Lenguajes de Consulta Relacionales Formales



- ❖ Dos lenguajes de consulta matemáticos forman la base de los lenguajes “reales” (p. ej. SQL) y de su implementación:
 - Algebra Relacional: Más **operacional**, muy útil para representar planes de ejecución..
 - Cálculo Relacional: Permite a los usuarios describir lo que desean más que computarlo. (**No-operacional**, o declarativo.)

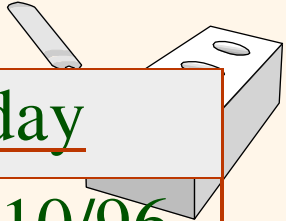


Preliminares

- ❖ Una consulta se aplica a una *instancia de relación*, y el resultado es también otra instancia de relación.
 - Los *esquemas de entrada* de la relación para una consulta están *fijos* (¡pero la consulta debe funcionar para cualquier instancia!)
 - El *esquema para el resultado* de una consulta dada está también *fijo*, y está determinado por la definición de los operadores del lenguaje de consulta.
- ❖ Notación posicional vs. notación de campos:
 - Notación posicional es más fácil para las definiciones formales; la notación de campos es más legible.
 - Ambas son usadas en SQL

Ejemplo de Instancias

R1



<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

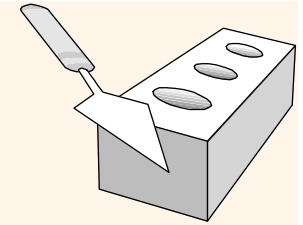
- ❖ Las relaciones “Marinos” (“Sailors”) and “Reservas” para nuestros ejemplos.
- ❖ Usaremos notación posicional, y asumiremos que los nombres de los campos en los resultados de las consultas son “heredados” de los nombres en los campos de las relaciones de entrada de la consulta.

S1

<u>sid</u>	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

S2

<u>sid</u>	sname	rating	age
28	yuppy	9	35.0
31	lubber	8	55.5
44	guppy	5	35.0
58	rusty	10	35.0



Algebra Relacional

❖ Operaciones básicas:

- Selección (σ) Selecciona un subconjunto de las tuplas de una relación.
- Proyección (π) Borra columnas no deseadas de una reln.
- Producto (π) Permite combinar dos relaciones.
- Diferencia (\times) \dashv Tuplas en reln. 1, pero not en reln. 2.
- Union (\cup) Tuplas en reln. 1 y en reln. 2.

❖ Operaciones adicionales:

- Intersección, join, división, renombramiento: No son esenciales, pero sí (muy!) útiles.

❖ Como cada una retorna una relación, **las operaciones se pueden componer!** (El Algebra es “cerrada”.)

Proyección

- ❖ Borra atributos que no están en la *lista de proyección*.
- ❖ *El Esquema* de los resultados contiene exactamente los campos en la lista de proyección, con los mismos nombres que ellos tenían en la (única) relación de entrada.
- ❖ El operador de proyección elimina los *duplicados*! (¿Porqué?)
 - Nota: los sistemas relaes típicamente no eliminan duplicados a menos que el usuario lo explicita. (¿Porqué no?)



sname	rating
yuppy	9
lubber	8
guppy	5
rusty	10

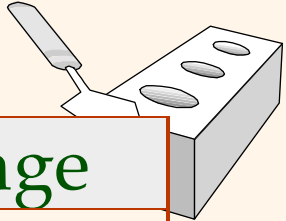
$\pi_{sname, rating}(S2)$

age
35.0
55.5

$\pi_{age}(S2)$

Selección

- ❖ Selecciona tuplas que satisfacen la *condición de selección*.
- ❖ No hay duplicados en el resultado! (¿Porqué?)
- ❖ *El Esquema* del resultado es idéntico al esquema de la (única) relación de entrada.
- ❖ *El resultado* puede ser la entrada de otra operación del algebra. (*Composición*).



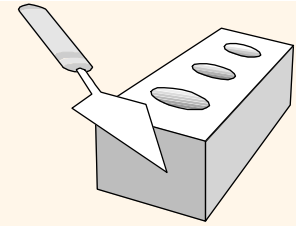
sid	sname	rating	age
28	yuppy	9	35.0
58	rusty	10	35.0

$$\sigma_{rating > 8}(S2)$$

sname	rating
yuppy	9
rusty	10

$$\pi_{sname, rating}(\sigma_{rating > 8}(S2))$$

Unión, Intersección, Diferencia



- ❖ Todas estas operaciones toman *dos* relaciones como entrada, las que deben ser compatibles:
 - Mismo número de campos.
 - Campos 'correspondientes' tienen el mismo tipo.

- ❖ ¿Cuál es el the *esquema* del resultado?

sid	sname	rating	age
22	dustin	7	45.0

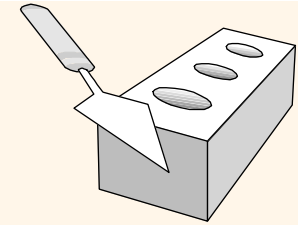
$S1 - S2$

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0
44	guppy	5	35.0
28	yuppy	9	35.0

$S1 \cup S2$

sid	sname	rating	age
31	lubber	8	55.5
58	rusty	10	35.0

$S1 \cap S2$



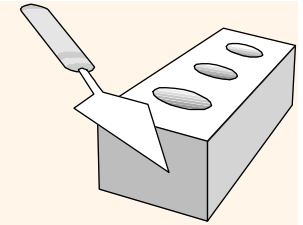
Producto (cartesiano)

- ❖ Cada tupla de S1 se alinea con cada tupla de R1
- ❖ *El esquema resultante*: un campo por cada uno de S1 y R1, con nombres “heredados” si es posible.
 - *Conflicto*: Ambas S1 y R1 tienen campo *sid*.

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

- *Operator de renombramiento*:

$$\rho (C(1 \rightarrow sid1, 5 \rightarrow sid2), S1 \times R1)$$



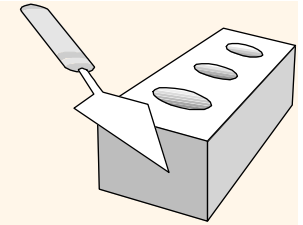
Reuniones (Joins)

❖ Reunión Condicional: $R \bowtie_c S = \sigma_c (R \times S)$

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	58	103	11/12/96

$$S1 \bowtie_{S1.sid < R1.sid} R1$$

- ❖ *Esquema resultante*: el mismo del producto.
- ❖ Menos tuplas que el producto; puede ser computados más eficientemente.
- ❖ A veces llamado *theta-join*.

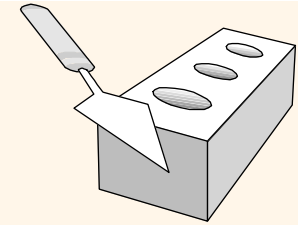


Reuniones

- ❖ Reunión de igualdad: Un caso especial de la reunión condicional, donde c contiene sólo igualdades.

sid	sname	rating	age	bid	day
22	dustin	7	45.0	101	10/10/96
58	rusty	10	35.0	103	11/12/96

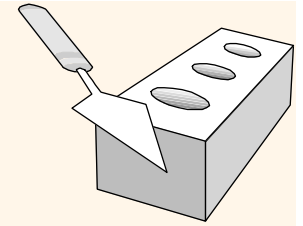
- ❖ Esquema resultante: $S1 \times_{sid} R1$ similar al producto, pero sólo una copia de cada pareja de campos que aparece en cada igualdad.
- ❖ Reunión natural: Reunión de igualdad sobre *todos* los campos de nombres comunes.



División

- ❖ No es soportada como operador primitivo, pero es muy útil para consultas como:
Marinos que han reservado todos los botes.
- ❖ Sea $A(x,y)$ y $B(y)$ dos esquemas de relación:
 - $A/B = \{ \langle x \rangle \mid \exists \langle x, y \rangle \in A \ \forall \langle y \rangle \in B \}$
 - i.e., **A/B contiene todas las tuplas x (sailors) tal que para cada tupla y (boat) de B , hay una tupla xy en A .**
 - *Alt:* si el conjunto de valores y (boats) asociados con un valor x (sailor) en A contiene todos los valores y en B , el valor x está en A/B .
- ❖ En general, x e y pueden ser cualquier lista de campos; y es la lista en B , y xy es la lista en A .

Ejemplos de división A/B



sno	pno
s1	p1
s1	p2
s1	p3
s1	p4
s2	p1
s2	p2
s3	p2
s4	p2
s4	p4

A

pno
p2

B1

sno
s1
s2
s3
s4

A/B1

pno
p2
p4

B2

sno
s1
s4

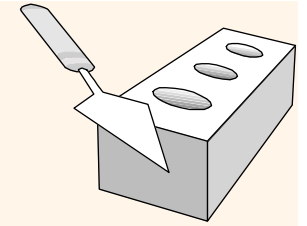
A/B2

pno
p1
p2
p4

B3

sno
s1

A/B3



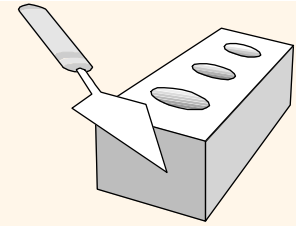
Expresando A/B con op. básicos

- ❖ La división no es esencial, es sólo un alias.
 - (Esto también es cierto para las reuniones, pero las reuniones son tan comunes que los sistemas las implementan especialmente.)
- ❖ *Idea:* Para A/B , compute todos los valores x que no son 'descalificados' por algún valor de B .
 - Valor x está *descalificado* si al agregar el valor y de B , obtenemos una tupla xy que no está en A .

Disqualified x values: $\pi_x ((\pi_x(A) \times B) - A)$

A/B : $\pi_x(A) -$ Todas las tuplas descalificadas

Encuentre los nombres de marinos que han reservado el bote con identificador #103



❖ Solución 1: $\pi_{sname}((\sigma_{bid=103} Reserves) \times Sailors)$

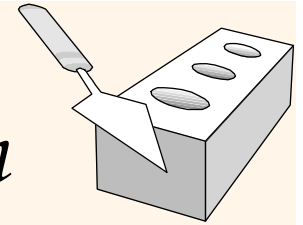
❖ Solución 2: $\rho(Temp1, \sigma_{bid=103} Reserves)$

$\rho(Temp2, Temp1 \times Sailors)$

$\pi_{sname}(Temp2)$

❖ Solución 3: $\pi_{sname}(\sigma_{bid=103}(Reserves \times Sailors))$

Encuentre los nombres de marinos que han reservado un bote rojo.



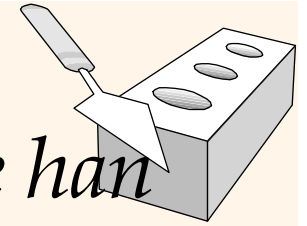
- ❖ Información acerca del color de un bote está sólo disponible en Boats; necesita join extra:

$$\pi_{sname}((\sigma_{color='red'} Boats) \times Reserves \times Sailors)$$

- ❖ Una solución más eficiente:

$$\pi_{sname}(\pi_{sid}((\pi_{bid} \sigma_{color='red'} Boats) \times Res) \times Sailors)$$

¡Un optimizador de consultas puede encontrar ésta si conoce la primera!



Encuentre los nombres de los marinos que han reservado un bote rojo o verde.

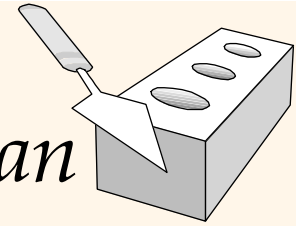
- ❖ Idea: identificar todos los botes rojos o verdes, y luego los marinos que los han reservado:

$$\rho \text{ (Tempboats, } (\sigma_{color='red' \vee color='green'} \text{ Boats}))$$

$$\pi_{sname}(\text{Tempboats} \times \text{Reserves} \times \text{Sailors})$$

- ❖ Alt: definir Tempboats usando unión! (¿Cómo?)
- ❖ ¿Que ocurre si \vee se reemplaza por \wedge en esta consulta?

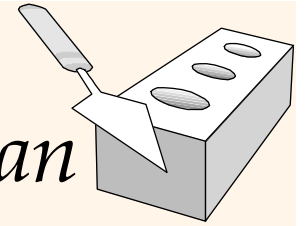
Encontrar los nombres de los marinos que han reservado un bote rojo y un bote verde.



❖ ¡Enfoque anterior no funciona! Hay que encontrar los marinos que han reservado botes rojos, aquellos que han reservado botes verdes, y luego intersectarlos (note que *sid* es una llave para Sailors):

$$\rho \text{ (Tempred, } \pi_{sid}((\sigma_{color='red'} Boats) \times Reserves))$$
$$\rho \text{ (Tempgreen, } \pi_{sid}((\sigma_{color='green'} Boats) \times Reserves))$$
$$\pi_{sname}((Tempred \cap Tempgreen) \times Sailors)$$

Encontrar los nombres de los marinos que han reservado todos los botes.



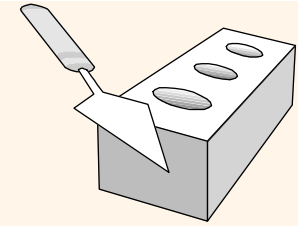
- ❖ Usa división; los esquemas de la relaciones de entrada deben ser elegidos con cuidado:

$$\rho (Tempsids, (\pi_{sid,bid} Reserves) / (\pi_{bid} Boats))$$

$$\pi_{sname} (Tempsids \times Sailors)$$

- ❖ Para encontrar los marinos que reservaron todos los botes 'Interlake':

$$..... / \pi_{bid} (\sigma_{bname='Interlake'} Boats)$$



Resumen

- ❖ El modelo relacional tiene definido rigurosamente lenguajes de consulta simples y poderosos.
- ❖ El algebra relacional es más operacional; muy útil como representación interna de los planes de evaluación.
- ❖ Muchas maneras de expresar una consulta dada; un optimizador de consultas debiera elegir la versión más eficiente.