

Algebra Relazionale



Linguaggi di interrogazione relazionale

- ❖ Linguaggi di interrogazione: permettono la manipolazione e il reperimento di dati da una base di dati
- ❖ Il modello relazionale supporta LI semplici e potenti:
 - Forte base formale basata sulla logica
 - Ottimizzazione
- ❖ Linguaggi di interrogazione **diversi da** linguaggi di programmazione!
 - I LI non sono necessariamente “Turing completi”
 - I LI non sono fatti per essere usati in calcoli complessi
 - I LI supportano un accesso semplice ed efficiente a grandi insiemi di dati

Linguaggi formali di interrogazione relazionale

- ❖ Due linguaggi di interrogazione matematici formano la base per i linguaggi “reali” (es. SQL) e per l’implementazione:
 - Algebra relazionale: più operativa, utilissima per rappresentare i piani di esecuzione
 - Calcolo relazionale: permette agli utenti di descrivere ciò che vogliono, piuttosto che il modo in cui calcolarlo (non operativa, dichiarativo)

✉ *Capire l’algebra e il calcolo è la chiave per la comprensione dell’SQL e dell’elaborazione delle interrogazioni!*



Nozioni Preliminari

- ❖ Una interrogazione si applica alle istanze di relazione, e il risultato di una interrogazione è anch'esso una istanza di relazione
 - Gli schemi delle relazioni in ingresso sono fissi (ma l'interrogazione viene eseguita indipendentemente dall'istanza)
 - Anche lo schema per il risultato di una data interrogazione è fisso. Determinato dalla definizione dei costrutti del linguaggio di interrogazione.
- ❖ Notazione posizionale verso notazione nominale dei campi:
 - La notazione posizionale è più semplice per definizioni formali, la notazione nominale è più leggibile
- ❖ In SQL sono usate entrambe



Schema di Esempio

- ❖ *Velisti*(vid:integer,vnome:string,esperienza:integer,età:real)
- ❖ *Barche*(bid:integer,bnome:string,colore:string)
- ❖ *Prenotazioni*(vid:integer,bid:integer,giorno:data)

Istanze di esempio

- ❖ Relazioni “Velisti” e “Prenota” per i nostri esempi
- ❖ Useremo la notazione posizionale o a campi nominati, assumeremo che i nomi dei campi nei risultati delle interrogazioni siano “ereditati” dai nomi dei campi delle relazioni in ingresso dell’interrogazione

P1

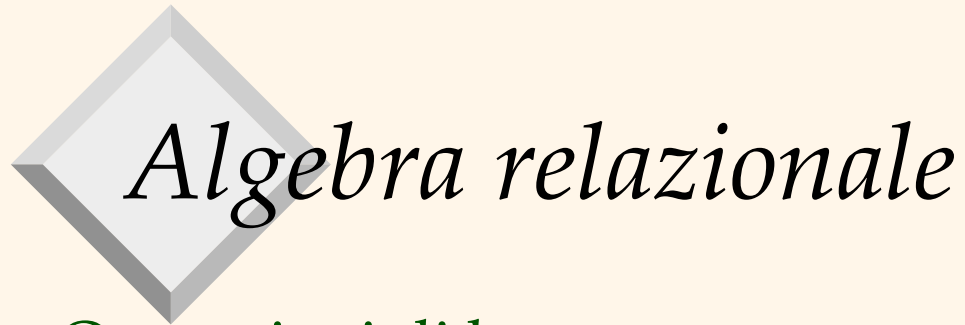
<u>sid</u>	<u>bid</u>	<u>day</u>
22	101	10/10/96
58	103	11/12/96

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 relazionale

❖ Operazioni di base

- **Selezione (σ)** Seleziona un sottoinsieme di righe della relazione
- **Proiezione (π)** Cancella colonne non desiderate dalla relazione
- **Prodotto cartesiano (\times)** Consente di combinare due relazioni
- **Differenza insiemistica ($-$)** Tuple presenti nella relazione 1, ma non nella relazione 2
- **Unione (\cup)** Tuple presenti nella relazione 1 e nella relazione 2

❖ Altre operazioni:

- **intersezione, join, divisione, ridenominazione**: non essenziali ma (molto!) utili
- Poiché ogni operazione restituisce una relazione, le operazioni possono essere composte (l'algebra è "chiusa")

Proiezione

- ❖ Cancella gli attributi che non sono nella lista di proiezione
- ❖ Lo schema del risultato contiene esattamente i campi nella lista di proiezione, con gli stessi nomi che avevano nella (unica) relazione in ingresso
- ❖ L'operatore di proiezione deve eliminare i duplicati (perché?)
 - Nota: i sistemi reali tipicamente non eliminano i duplicati a meno che l'utente non lo richieda esplicitamente

Proiezione

sname	rating
yuppy	9
lubber	8
guppy	5
rusty	10

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

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

age
35.0
55.5

$\pi_{age}(S2)$



Selezione

- ❖ Seleziona le righe che soddisfano una condizione di selezione
- ❖ Niente duplicati nel risultato
- ❖ Lo schema del risultato è identico allo schema della (unica) relazione in ingresso
- ❖ La relazione risultato può essere l'ingresso per un'altra operazione di algebra relazionale! (Composizione di operatori)

Selezione

- ❖ Condizione di selezione:
 - Attributo OP Costante
 - Attributo1 OP Attributo2
 - OP= $<$, $<=$, $=$, \neq , $>=$, $>$
 - Attributo specificabile per nome o per posizione

Selezione

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

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

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

sname	rating
yuppy	9
rusty	10

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



Selezione e proiezione

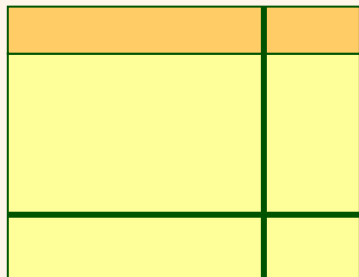
Sono due operatori "ortogonali"

❖ **selezione:**

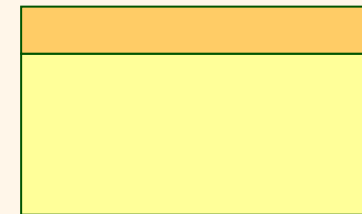
- decomposizione orizzontale

❖ **proiezione:**

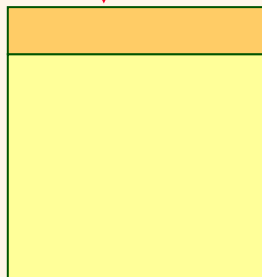
- decomposizione verticale

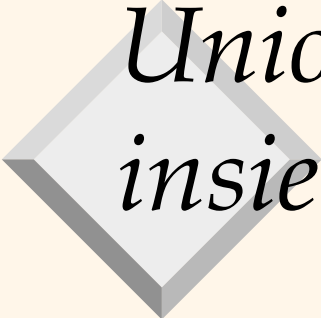


-**selezione**



-**proiezione**





Unione, intersezione, differenza insiemistica

- ❖ Tutte queste operazioni prendono in ingresso due relazioni che devono essere compatibili rispetto all'unione:
 - stesso numero di campi
 - campi "corrispondenti" hanno lo stesso dominio
- ❖ qual è lo schema del risultato?

Unione

S1

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

<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

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

$S1 \cup S2$

Intersezione

S1

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

<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

S2

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

$S1 \cap S2$

Differenza

S1

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

sid	sname	rating	age
22	dustin	7	45.0

$S1 - S2$

<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

S2

Prodotto cartesiano

- ❖ Ciascuna riga di S1 è accoppiata con ciascuna riga di P1
- ❖ Lo schema del risultato ha un campo per ogni campo di V1 e P1, i cui nomi sono, se possibile, “ereditati”
 - Conflitto: sia S1 che P1 hanno un campo chiamato *sid*

S1

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

P1

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

Prodotto cartesiano

(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

✉ Operatore per ridenominare i campi
 $\rho (C(1 \rightarrow \text{sid1}, 5 \rightarrow \text{sid2}), V1 \times P1)$

Operatore ρ rinominare i campi

- ❖ $\rho(R(F),E)$
 - ❖ E espressione dell'algebra relazionale
 - ❖ R istanza di nuova relazione
 - ❖ F lista di ridenominazione
- ❖ $\rho(C(1 \rightarrow \text{sid1}, 5 \rightarrow \text{sid2}), V1 \times P1)$



- ❖ Molto usato, permette di correlare dati in relazioni diverse
- ❖ Può essere ottenuto da un prodotto cartesiano seguito da selezioni e proiezioni
- ❖ Molte varianti dell'operazione di join

Join condizionale

- ❖ Join condizionale: $P \times_c S = \sigma_c(P \times S)$
 - Lo schema del risultato è lo stesso del prodotto cartesiano
 - Meno tuple del prodotto cartesiano, potrebbe essere calcolato più efficientemente
 - A volte chiamato theta-join

Join condizionale

S1

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

P1

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

(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

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

Equi-Join

- ❖ Equi-join: un caso speciale di join condizionale dove la condizione c contiene solo uguaglianze

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

$$S1 \bowtie_{sid} R1$$

- ❖ Lo schema del risultato è simile al prodotto cartesiano, ma c'è solo una copia dei campi per i quali è specificata l'uguaglianza
- ❖ **Join naturale**: equijoin su tutti i campi in comune

Divisione

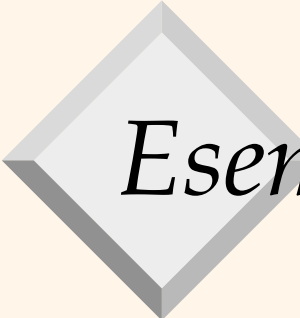
- ❖ Non supportata come operatore primitivo, ma utile per esprimere interrogazioni come:

trovare i velisti che hanno prenotato tutte le barche

- ❖ Sia A una relazione con due campi, x e y ; sia B una relazione con il solo campo y :

- $A/B = \{\langle x \rangle \mid \langle x, y \rangle \in A, \langle y \rangle \in B\}$
- cioè, A/B contiene tutte le tuple x (velisti) tali che per ogni tupla y (barca) in B , ci sia una tupla xy in A
- Oppure. Se l'insieme di valori y (barche) associato con un valore x (velista) in A contiene tutti i valori y in B , il valore x è in A/B

- ❖ In generale, x e y possono essere un qualunque elenco di campi; y è l'elenco di campi in B , e $x \cup y$ è l'elenco dei campi in A



Esempi di divisione 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

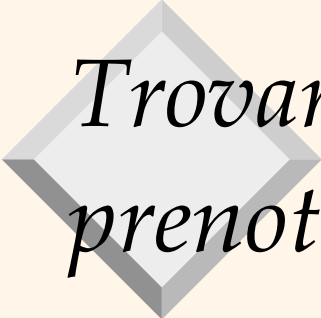
A/B3

Esprimere A/B usando operatori di base

- ❖ La divisione non è una operazione essenziale; solo un'utile scorciatoia
 - (Vero anche per i join, ma i join sono così comuni che i sistemi li implementano esplicitamente)
- ❖ Idea: per A/B , calcolare tutti i valori x che non sono "interdetti" da qualche valore y in B
 - Un valore x è "interdetto" se associandogli valori y da B otteniamo una tupla xy che non è in A

Valori x interdetti : $\pi_x ((\pi_x(A) \times B) - A)$

A/B : $\pi_x(A) -$ Tutte le tuple interdette



Trovare i nomi dei velisti che hanno prenotato la barca #103

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

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

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

$\pi_{sname}(Temp2)$

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

Trovare i nomi dei velisti che hanno prenotato una barca rossa


- Informazioni sul colore sono disponibili solo in Barche; quindi abbiamo bisogno di un ulteriore join:

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

❖ Una soluzione più efficiente:

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

✉ *Un ottimizzatore di interrogazioni può trovare la seconda soluzione data la prima!*



Trovare i nomi dei velisti che hanno prenotato una barca rossa o una verde

- ❖ Possiamo identificare tutte le barche rosse o verdi, poi trovare i velisti che hanno prenotato una di esse :

$$\rho (Tempboats, (\sigma_{color = 'red' \vee color = 'green'} Boats))$$
$$\pi_{sname}(Tempboats \bowtie Reserves \bowtie Sailors)$$

- ❖ Possiamo anche definire TempBarche usando l'unione (come?)
- ❖ Che succede se \vee è sostituito da \wedge in questa interrogazione?

Trovare i nomi dei velisti che hanno prenotato una barca rossa e una verde

- ❖ L'approccio precedente non funziona! Dobbiamo identificare i velisti che hanno prenotato barche rosse, i velisti che hanno prenotato barche verdi, poi trovare l'intersezione (notate che *vid* è una chiave per *Velisti*)

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

Trovare i nomi dei velisti che hanno prenotato tutte le barche

- ❖ Usa la divisione; gli schemi delle relazioni in ingresso alla divisione devono essere scelti con cura:

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

$$\pi_{sname} (Tempoids \bowtie Sailors)$$

- ❖ Per trovare i velisti che hanno riservato tutte le barche “Interlake”:

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