L'algebra relazionale

Sistemi Informativi L-A

Home Page del corso:
http://www-db.deis.unibo.it/courses/SIL-A/

Versione elettronica: Algebra.pdf

Linguaggi di manipolazione (DML) per DB

- Un linguaggio di manipolazione, o DML, permette di interrogare e modificare istanze di Basi di Dati
- A parte i linguaggi utente, quali SQL, esistono altri linguaggi, formalmente definiti, che rivestono notevole importanza in quanto enfatizzano gli aspetti "essenziali" dell'interazione con un DB relazionale
- In particolare:
  - Calcolo relazionale
    - linguaggio dichiarativo basato sulla logica dei predicati del primo ordine
  - Algebra relazionale
    - linguaggio procedurale di tipo algebrico i cui operandi sono relazioni

  sono due linguaggi che si concentrano sugli aspetti di interrogazione:
    - Calcolo e algebra sono equivalenti in termini di potere espressivo ("ciò che riescono a calcolare")
    - L'algebra è la base per capire "come le interrogazioni vengono effettivamente elaborate da un DBMS"

Algebra relazionale: premesse

- L'algebra relazionale (AR) è costituita da un insieme di operatori che si applicano a una o più relazioni e che producono una relazione
  - Operatori di base unari: selezione, proiezione e ridenominazione
  - Operatori di base binari: join (naturale), unione e differenza
  - … più altri derivati da questi

- La semantica di ogni operatore si definisce specificando
  - come lo schema (insieme di attributi) del risultato dipende dallo schema degli operandi
  - come l'istanza risultato dipende dalle istanze in ingresso
- Gli operatori si possono comporre, dando luogo a espressioni algebriche di complessità arbitraria
- Gli operandi sono o (nomi di) relazioni del DB o espressioni (ben formate)
- Per iniziare, si assume che non siano presenti valori nulli

Selezione

- L'operatore di selezione, σ, permette di selezionare un sottoinsieme delle tuple di una relazione, applicando a ciascuna di esse una formula booleana F

Espressione: $\sigma_F(R)$

	Input	Output
Schema	R(X)	X
Istanza	r	$\sigma_F(r)$ = { t | t r AND F(t) = vero }

- F si compone di predicati connessi da AND ( ), OR ( ) e NOT (¬)
- Ogni predicato è del tipo A θ c o A θ B, dove:
  - A e B sono attributi in X
  - c   dom(A) è una costante
  - θ è un operatore di confronto, θ   {=, ≠, <, >, ≤, ≥}

Selezione: esempi (1)

Esami

Matricola	CodCorso	Voto	Lode
29323	483	28	NO
39654	729	30	SÌ
29323	913	26	NO
35467	913	30	NO
31283	729	30	NO

$\sigma_{(\text{Voto} = 30) \text{ AND } (\text{Lode} = \text{NO})}(\text{Esami})$

Matricola	CodCorso	Voto	Lode
35467	913	30	NO
31283	729	30	NO

$\sigma_{(\text{CodCorso} = 729) \text{ OR } (\text{Voto} = 30)}(\text{Esami})$

Matricola	CodCorso	Voto	Lode
39654	729	30	SÌ
35467	913	30	NO
31283	729	30	NO

Selezione: esempi (2)

Partite

Giornata	Casa	Ospite	GolCasa	GolOspite
4	Venezia	Bologna	0	1
5	Brescia	Atalanta	3	3
5	Inter	Bologna	1	0
5	Lazio	Parma	0	0

$\sigma_{(\text{Giornata} = 5) \text{ AND } (\text{GolCasa} = \text{GolOspite})}(\text{Partite})$

Giornata	Casa	Ospite	GolCasa	GolOspite
5	Brescia	Atalanta	3	3
5	Lazio	Parma	0	0

$\sigma_{(\text{Ospite} = \text{Bologna}) \text{ AND } (\text{GolCasa} < \text{GolOspite})}(\text{Partite})$

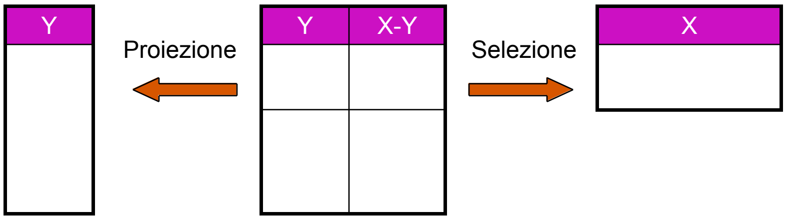
Giornata	Casa	Ospite	GolCasa	GolOspite
4	Venezia	Bologna	0	1

Proiezione

- L'operatore di proiezione, π, è ortogonale alla selezione, in quanto permette di selezionare un sottoinsieme Y degli attributi di una relazione

Espressione: $\pi_Y(R)$

	Input	Output
Schema	R(X)	Y
Istanza	r	$\pi_Y(r) = \{ t[Y] \mid t \quad r \}$



Proiezione: esempi (1)

Corsi

CodCorso	Titolo	Docente	Anno
483	Analisi	Biondi	1
729	Analisi	Neri	1
913	Sistemi Informativi	Castani	2

$\pi_{CodCorso,Docente}(Corsi)$

CodCorso	Docente
483	Biondi
729	Neri
913	Castani

$\pi_{CodCorso,Anno}(Corsi)$

CodCorso	Anno
483	1
729	1
913	2

Proiezione: esempi (2)

Corsi

CodCorso	Titolo	Docente	Anno
483	Analisi	Biondi	1
729	Analisi	Neri	1
913	Sistemi Informativi	Castani	2

$\pi_{Titolo}(Corsi)$

Titolo
Analisi
Sistemi Informativi

$\pi_{Docente}(Corsi)$

Docente
Biondi
Neri
Castani

Proiezione: cardinalità del risultato

- In generale, la cardinalità di $\pi_Y(r)$ è minore o uguale a quella di r (la proiezione "elimina i duplicati")
- L'uguaglianza è garantita se e solo se Y è una superchiave di R(X)

Dimostrazione:

(Se) Se Y è una superchiave di R(X), in ogni istanza legale r di R(X) non esistono due tuple distinte t1 e t2 tali che t1[Y] = t2[Y]

(Solo se) Se Y non è superchiave allora è possibile costruire un'istanza legale r con due tuple distinte t1 e t2 tali che t1[Y] = t2[Y]. Tali tuple "collassano" in una singola tupla a seguito della proiezione

- Si noti che il risultato ammette la possibilità che "per caso" la cardinalità non vari anche se Y non è superchiave (es: $\pi_{Docente}(Corsi)$)

Join naturale

- L'operatore di join naturale, $\bowtie$, combina le tuple di due relazioni sulla base dell'uguaglianza dei valori degli attributi comuni alle due relazioni

Esami

Matricola	CodCorso	Voto	Lode
29323	483	28	NO
39654	729	30	SÌ
29323	913	26	NO
35467	913	30	NO

Corsi

CodCorso	Titolo	Docente	Anno
483	Analisi	Biondi	1
729	Analisi	Neri	1
913	Sistemi Informativi	Castani	2

Esami $\bowtie$ Corsi

Matricola	CodCorso	Voto	Lode	Titolo	Docente	Anno
29323	483	28	NO	Analisi	Biondi	1
39654	729	30	SÌ	Analisi	Neri	1
29323	913	26	NO	Sistemi Informativi	Castani	2
35467	913	30	NO	Sistemi Informativi	Castani	2

Join naturale: definizione

- Ogni tupla che compare nel risultato del join naturale di r_1 e r_2, istanze rispettivamente di $R_1(X_1)$ e $R_2(X_2)$, è ottenuta come combinazione ("match") di una tupla di r_1 con una tupla di r_2 sulla base dell'uguaglianza dei valori degli attributi comuni (cioè quelli in $X_1 \cap X_2$)
- Inoltre, lo schema del risultato è l'unione degli schemi degli operandi

Espressione: $R_1 \bowtie R_2$

	Input	Output
Schema	$R_1(X_1), R_2(X_2)$	X_1X_2
Istanza	r_1, r_2	$r_1 \bowtie r_2 = \{ t \mid t[X_1] \quad r_1 \text{ AND } t[X_2] \quad r_2 \}$

Join naturale: esempi (1)

Voli

Codice	Data	Comandante
AZ427	21/07/2001	Bianchi
AZ427	23/07/2001	Rossi
TW056	21/07/2001	Smith

Linee

Codice	Partenza	Arrivo
AZ427	FCO	JFK
TW056	LAX	FCO

Prenotazioni

Codice	Data	Classe	Cliente
AZ427	21/07/2001	Economy	Anna Bini
AZ427	21/07/2001	Business	Franco Dini
AZ427	23/07/2001	Economy	Ada Cini

Voli ⊳⊲ Linee

Codice	Data	Comandante	Partenza	Arrivo
AZ427	21/07/2001	Bianchi	FCO	JFK
AZ427	23/07/2001	Rossi	FCO	JFK
TW056	21/07/2001	Smith	LAX	FCO

Join naturale: esempi (2)

Voli ⊳⊲ Prenotazioni

Codice	Data	Comandante	Classe	Cliente
AZ427	21/07/2001	Bianchi	Economy	Anna Bini
AZ427	21/07/2001	Bianchi	Business	Franco Dini
AZ427	23/07/2001	Rossi	Economy	Ada Cini

LInee ⊳⊲ Prenotazioni

Codice	Partenza	Arrivo	Data	Classe	Cliente
AZ427	FCO	JFK	21/07/2001	Economy	Anna Bini
AZ427	FCO	JFK	21/07/2001	Business	Franco Dini
AZ427	FCO	JFK	23/07/2001	Economy	Ada Cini

Join naturale: osservazioni

- È possibile che una tupla di una delle relazioni operande non faccia match con nessuna tupla dell'altra relazione; in tal caso tale tupla viene detta "dangling"
- Nel caso limite è quindi possibile che il risultato del join sia vuoto; all'altro estremo è possibile che ogni tupla di r_1 si combini con ogni tupla di r_2
- Ne segue che

  la cardinalità del join, $| r_1 \bowtie r_2 |$, è compresa tra 0 e $| r_1 | * | r_2 |$

- Se il join è eseguito su una superchiave di $R_1(X_1)$, allora ogni tupla di r_2 fa match con al massimo una tupla di r_1, quindi $| r_1 \bowtie r_2 | \leq | r_2 |$
- Se $X_1 \cap X_2$ è la chiave primaria di $R_1(X_1)$ e foreign key in $R_2(X_2)$ (e quindi c'è un vincolo di integrità referenziale) allora $| r_1 \bowtie r_2 | = | r_2 |$

Join naturale e intersezione

- Quando le due relazioni hanno lo stesso schema ($X_1 = X_2$) allora due tuple fanno match se e solo se hanno lo stesso valore per tutti gli attributi, ovvero sono identiche, per cui:

Se $X_1 = X_2$ il join naturale equivale all'intersezione ($\cap$) delle due relazioni

VoliCharter

Codice	Data
XY123	21/07/2001
SC278	28/07/2001
XX338	18/08/2001

VoliNoSmoking

Codice	Data
SC278	28/07/2001
SC315	30/07/2001

VoliCharter $\bowtie$ VoliNoSmoking

Codice	Data
SC278	28/07/2001

Join naturale e prodotto Cartesiano

- Viceversa, quando non ci sono attributi in comune ($X_1 \cap X_2 = \varnothing$), allora due tuple fanno sempre match, per cui:

  Se $X_1 \cap X_2 = \varnothing$ il join naturale equivale al prodotto Cartesiano

Si noti che in questo caso, a differenza del caso matematico, il prodotto Cartesiano non è ordinato

VoliCharter

Codice	Data
XY123	21/07/2001
SC278	28/07/2001
XX338	18/08/2001

VoliNoSmoking

Numero	Giorno
SC278	28/07/2001
SC315	30/07/2001

VoliCharter $\bowtie$ VoliNoSmoking

Codice	Data	Numero	Giorno
XY123	21/07/2001	SC278	28/07/2001
SC278	28/07/2001	SC278	28/07/2001
XX338	18/08/2001	SC278	28/07/2001
XY123	21/07/2001	SC315	30/07/2001
SC278	28/07/2001	SC315	30/07/2001
XX338	18/08/2001	SC315	30/07/2001

Unione e differenza

- Poiché le relazioni sono insiemi, sono ben definite le operazioni di unione, , e differenza, -
- Entrambe si applicano a relazioni con lo stesso insieme di attributi

Espressione: $R_1 \cup R_2$

	Input	Output
Schema	$R_1(X)$, $R_2(X)$	X
Istanza	r_1, r_2	r_1 r_2 = { t | t r_1 OR t r_2 }

Espressione: R_1 - R_2

	Input	Output
Schema	$R_1(X)$, $R_2(X)$	X
Istanza	r_1, r_2	$r_1 - r_2$ = { t | t r_1 AND t r_2 }

- Si noti che l'intersezione si può anche scrivere come: $r_1 \cap r_2 = r_1 - (r_1 - r_2)$

Unione e differenza: esempi

VoliCharter

Codice	Data
XY123	21/07/2001
SC278	28/07/2001
XX338	18/08/2001

VoliNoSmoking

Codice	Data
SC278	28/07/2001
SC315	30/07/2001

VoliCharter VoliNoSmoking

Codice	Data
XY123	21/07/2001
SC278	28/07/2001
XX338	18/08/2001
SC315	30/07/2001

VoliCharter - VoliNoSmoking

Codice	Data
XY123	21/07/2001
XX338	18/08/2001

VoliNoSmoking - VoliCharter

Codice	Data
SC315	30/07/2001

Il problema dei nomi

- Il join naturale, l'unione e la differenza operano (sia pur diversamente) sulla base degli attributi comuni a due schemi

VoliCharter

Codice	Data
XY123	21/07/2001
SC278	28/07/2001
XX338	18/08/2001

VoliNoSmoking

Numero	Giorno
SC278	28/07/2001
SC315	30/07/2001

Come si fa l'unione e la differenza?

Studenti

Matricola	CodiceFiscale	Cognome	Nome	DataNascita
29323	BNCGRG78F21A	Bianchi	Giorgio	21/06/1978
35467	RSSNNA78D13A	Rossi	Anna	13/04/1978

Come si fa il join?

Redditi

CF	Imponibile
BNCGRG78F21A	10000

Ridenominazione

- L'operatore di ridenominazione, ρ, modifica lo schema di una relazione, cambiando i nomi di uno o più attributi
- La definizione formale, oltremodo complessa, si omette; è sufficiente ricordare che $\rho_{Y \leftarrow X}(r)$, con r su R(XZ), cambia lo schema in YZ, lasciando invariati i valori delle tuple, e che nel caso si cambi più di un attributo, allora l'ordine in cui si elencano è significativo

Espressione: $\rho_{Y \leftarrow X}(R)$

	Input	Output
Schema	R(XZ)	YZ

Ridenominazione: esempi

Redditi

CF	Imponibile
BNCGRG78F21A	10000

$\rho_{CodiceFiscale \leftarrow CF}$(Redditi)

CodiceFiscale	Imponibile
BNCGRG78F21A	10000

VoliNoSmoking

Numero	Giorno
SC278	28/07/2001
SC315	30/07/2001

$\rho_{Codice,Data \leftarrow Numero,Giorno}$(VoliNoSmoking)

Codice	Data
SC278	28/07/2001
SC315	30/07/2001

Self-join

- La ridenominazione permette di eseguire il join di una relazione con se stessa ("self-join") in modo significativo (si ricordi che $r \bowtie r = r$!)

Genitori

Genitore	Figlio
Luca	Anna
Maria	Anna
Giorgio	Luca
Silvia	Maria
Enzo	Maria

Per trovare nonni e nipoti:

$\rho_{Nonno,Genitore \leftarrow Genitore,Figlio}(Genitori)$

Nonno	Genitore
Luca	Anna
Maria	Anna
Giorgio	Luca
Silvia	Maria
Enzo	Maria

$\rho_{Nonno,Genitore \leftarrow Genitore,Figlio}(Genitori) \bowtie Genitori$

Nonno	Genitore	Figlio
Giorgio	Luca	Anna
Silvia	Maria	Anna
Enzo	Maria	Anna

… poi si può ridenominare Figlio in Nipote e proiettare su {Nonno,Nipote}

Operatori derivati: la divisione

- Gli operatori sinora visti definiscono completamente l'AR. Tuttavia, per praticità, è talvolta utile ricorrere ad altri operatori "derivati", quali la divisione e il theta-join
- La divisione, ÷, di r_1 per r_2 , con r_1 su $R_1(X_1X_2)$ e r_2 su $R_2(X_2)$, è (il più grande) insieme di tuple con schema X_1 tale che, facendo il prodotto Cartesiano con r_2, ciò che si ottiene è una relazione contenuta in r_1

Espressione: $R_1 \div R_2$

	Input	Output
Schema	$R_1(X_1 X_2), R_2(X_2)$	X_1
Istanza	r_1, r_2	$r_1 \div r_2 = \{ t \mid \{t\} \bowtie r_2 \quad r_1 \}$

La divisione si può esprimere come: $\pi_{X1}(r_1) - \pi_{X1}((\pi_{X1}(r_1) \bowtie r_2) - r_1)$

Divisione: esempio

Voli

Codice	Data
AZ427	21/07/2001
AZ427	23/07/2001
AZ427	24/07/2001
TW056	21/07/2001
TW056	24/07/2001
TW056	25/07/2001

Linee

Codice
AZ427
TW056

Voli ÷ Linee

Data
21/07/2001
24/07/2001

(Voli ÷ Linee) $\bowtie$ Linee

Codice	Data
AZ427	21/07/2001
AZ427	24/07/2001
TW056	21/07/2001
TW056	24/07/2001

La divisione trova le date con voli per tutte le linee

In generale, la divisione è utile per interrogazioni di tipo “universale”

Operatori derivati: il theta-join

- Il theta-join è la combinazione di prodotto Cartesiano e selezione:

$$r_1 \bowtie_F r_2 = \sigma_F(r_1 \bowtie r_2)$$

con r_1 e r_2 senza attributi in comune e F composta di “predicati di join”, ossia del tipo A θ B, con A X_1 e B X_2

- Se F è una congiunzione di uguaglianze, si parla più propriamente di equi-join

Theta-join: esempi

Ricercatori

Nome	CodProgetto
Rossi	HK27
Verdi	HAL2000
Bianchi	HK27
Verdi	HK28
Neri	HAL2000

Progetti

Sigla	Responsabile
HK27	Bianchi
HAL2000	Neri
HK28	Verdi

Ricercatori $\bowtie_{\text{CodProgetto=Sigla}}$ Progetti

Nome	CodProgetto	Sigla	Responsabile
Rossi	HK27	HK27	Bianchi
Verdi	HAL2000	HAL2000	Neri
Bianchi	HK27	HK27	Bianchi
Verdi	HK28	HK28	Verdi
Neri	HAL2000	HAL2000	Neri

Ricercatori $\bowtie_{\text{(CodProgetto=Sigla) AND (Nome} \neq \text{Responsabile)}}$ Progetti

Nome	CodProgetto	Sigla	Responsabile
Rossi	HK27	HK27	Bianchi
Verdi	HAL2000	HAL2000	Neri

Theta-join: una precisazione

- Così come è stato definito, il theta-join richiede in ingresso relazioni con schemi disgiunti
- In diversi libri di testo e lavori scientifici (e anche nei DBMS), viceversa, il theta-join accetta relazioni con schemi arbitrari e "prende il posto" del join naturale, ossia: tutti i predicati di join vengono esplicitati
- In questo caso, per garantire l'univocità (distinguibilità) degli attributi nello schema risultato, è necessario adottare "dei trucchi" (ad es. usare il nome della relazione; DB2 usa un suffisso numerico: 1, 2, ecc.)

Ric

Nome	CodProgetto
Rossi	HK27
Bianchi	HK27
Verdi	HK28

Prog

Sigla	Nome
HK27	Bianchi
HK28	Verdi

Ric $\bowtie_{\text{(CodProgetto=Sigla) AND (Ric.Nome} \neq \text{Prog.Nome)}}$ Prog

Ric.Nome	CodProgetto	Sigla	Prog.Nome
Rossi	HK27	HK27	Bianchi

Algebra con valori nulli

- La presenza di valori nulli nelle istanze richiede un'estensione della semantica degli operatori
- Inoltre, è utile considerare una estensione del join naturale che non scarta le tuple dangling, ma genera valori nulli

- Va premesso che esistono diversi approcci al trattamento dei valori nulli, nessuno dei quali è completamente soddisfacente (per ragioni formali e/o pragmatiche)
- L'approccio che qui si presenta è quello "tradizionale", che ha il pregio di essere molto simile a quello adottato in SQL (e quindi dai DBMS relazionali)

π, e − con i valori nulli

- Proiezione, unione e differenza continuano a comportarsi usualmente, quindi due tuple sono uguali anche se ci sono dei NULL

Impiegati

Cod	Nome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
435	Verdi	NULL

$\pi_{Nome,Ufficio}$(Impiegati)

Nome	Ufficio
Rossi	A12
Verdi	NULL
Verdi	A27

Responsabili

Cod	Nome	Ufficio
123	Rossi	A12
NULL	NULL	A27
435	Verdi	NULL

Impiegati Responsabili

Cod	Nome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
435	Verdi	NULL
NULL	NULL	A27

σ con valori nulli

- Per la selezione il problema è stabilire se, in presenza di NULL, un predicato è vero o meno per una data tupla

Impiegati

Cod	Nome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27

$\sigma_{Ufficio = A12}$(Impiegati)

- Sicuramente la prima tupla fa parte del risultato e la terza no
- Ma la seconda? Non si hanno elementi sufficienti per decidere…
- … e lo stesso varrebbe per $\sigma_{Ufficio \neq A12}$(Impiegati)!!

Logica a tre valori

- Oltre ai valori di verità Vero (V) e Falso (F), si introduce "Sconosciuto" (?)

NOT	
V	F
F	V
?	?

AND	V	F	?
V	V	F	?
F	F	F	F
?	?	F	?

OR	V	F	?
V	V	V	V
F	V	F	?
?	V	?	?

- Una selezione produce le sole tuple per cui l'espressione di predicati risulta vera
- Per lavorare esplicitamente con i NULL si introduce l'operatore di confronto IS, ad es. A IS NULL
- NOT (A IS NULL) si scrive anche A IS NOT NULL

Selezione con valori nulli: esempi

Impiegati

Cod	Nome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
385	NULL	A27

$\sigma_{\text{Ufficio = A12}}$(Impiegati)

Cod	Nome	Ufficio
123	Rossi	A12

$\sigma_{\text{(Ufficio = A12) OR (Ufficio} \neq \text{A12)}}$(Impiegati)

Cod	Nome	Ufficio
123	Rossi	A12
373	Verdi	A27
385	NULL	A27

$\sigma_{\text{(Ufficio = A27) AND (Nome = Verdi)}}$(Impiegati)

Cod	Nome	Ufficio
373	Verdi	A27

$\sigma_{\text{(Ufficio = A27) OR (Nome = Verdi)}}$(Impiegati)

Cod	Nome	Ufficio
231	Verdi	NULL
373	Verdi	A27
385	NULL	A27

$\sigma_{\text{Ufficio IS NULL}}$(Impiegati)

Cod	Nome	Ufficio
231	Verdi	NULL

$\sigma_{\text{(Ufficio IS NULL) AND (Nome IS NULL)}}$(Impiegati)

Cod	Nome	Ufficio

▷◁ con valori nulli

- Il join naturale non combina due tuple se queste hanno entrambe valore nullo su un attributo in comune (e valori uguali sugli eventuali altri attributi comuni)

Impiegati

Cod	Nome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
435	Verdi	NULL

Responsabili

Ufficio	Cod
A12	123
A27	NULL
NULL	231

Impiegati ▷◁ Responsabili

Cod	Nome	Ufficio
123	Rossi	A12

Join ≠ intersezione con valori nulli!

- In assenza di valori nulli l'intersezione di r_1 e r_2 si può esprimere
  - mediante il join naturale, $r_1 \cap r_2 = r_1 \triangleright\triangleleft r_2$, oppure
  - sfruttando l'uguaglianza $r_1 \cap r_2 = r_1 - (r_1 - r_2)$
- In presenza di valori nulli, dalle definizioni date si ha che
  - nel primo caso il risultato non contiene tuple con valori nulli
  - nel secondo caso, viceversa, tali tuple compaiono nel risultato

Impiegati

Cod	Nome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
435	Verdi	NULL

Responsabili

Cod	Nome	Ufficio
123	Rossi	A12
NULL	NULL	A27
435	Verdi	NULL

Impiegati - Responsabili

Cod	Nome	Ufficio
231	Verdi	NULL
373	Verdi	A27

Impiegati – (Impiegati – Responsabili)

Cod	Nome	Ufficio
123	Rossi	A12
435	Verdi	NULL

Outer join: mantenere le tuple dangling

- In alcuni casi è utile che anche le tuple dangling di un join compaiano nel risultato
- A tale scopo si introduce l'outer join (join "esterno") che "completa" con valori nulli le tuple dangling
- Esistono tre varianti
  - Left (=▷◁): solo tuple dell'operando sinistro sono riempite con NULL
  - Right (▷◁=): idem per l'operando destro
  - Full (=▷◁=): si riempiono con NULL le tuple dangling di entrambi gli operandi

Outer join: esempi

Ricercatori

Nome	CodProgetto
Rossi	HK27
Bianchi	HK27
Verdi	HK28

Progetti

CodProgetto	Responsabile
HK27	Bianchi
HAL2000	Neri

Ricercatori ⟕ Progetti

Nome	CodProgetto	Responsabile
Rossi	HK27	Bianchi
Bianchi	HK27	Bianchi
Verdi	HK28	NULL

Ricercatori ⟖ Progetti

Nome	CodProgetto	Responsabile
Rossi	HK27	Bianchi
Bianchi	HK27	Bianchi
NULL	HAL2000	Neri

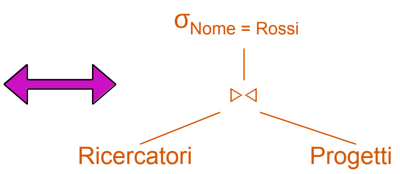
Ricercatori ⟗ Progetti

Nome	CodProgetto	Responsabile
Rossi	HK27	Bianchi
Bianchi	HK27	Bianchi
Verdi	HK28	NULL
NULL	HAL2000	Neri

Espressioni e viste

- Gli operatori dell'AR si possono liberamente combinare tra loro, avendo cura di rispettare le regole stabilite per la loro applicabilità
- Oltre alla rappresentazione "lineare" è anche possibile adottare una rappresentazione grafica in cui l'espressione è rappresentata ad albero

$\sigma_{Nome = Rossi}$(Ricercatori $\Join$ Progetti)



- Al fine di "semplificare" espressioni complesse è anche possibile fare uso di viste, ovvero espressioni a cui viene assegnato un nome e che è possibile riutilizzare all'interno di altre espressioni

ProgettiRossi = $\sigma_{Nome = Rossi}$(Ricercatori $\Join$ Progetti)

DB di riferimento per gli esempi

Imp

CodImp	Nome	Sede	Ruolo	Stipendio
E001	Rossi	S01	Analista	2000
E002	Verdi	S02	Sistemista	1500
E003	Bianchi	S01	Programmatore	1000
E004	Gialli	S03	Programmatore	1000
E005	Neri	S02	Analista	2500
E006	Grigi	S01	Sistemista	1100
E007	Violetti	S01	Programmatore	1000
E008	Aranci	S02	Programmatore	1200

Sedi

Sede	Responsabile	Citta
S01	Biondi	Milano
S02	Mori	Bologna
S03	Fulvi	Milano

Prog

CodProg	Citta
P01	Milano
P01	Bologna
P02	Bologna

Espressioni: esempi (1)

1) Nome, sede e stipendio degli impiegati che guadagnano più di 1300 Euro, definendo la vista ImpRicchi

$$\text{ImpRicchi} = \pi_{\text{Nome,Sede,Stipendio}}(\sigma_{\text{Stipendio} > 1300}(\text{Imp}))$$

oppure:

$$\text{ImpRicchi} = \sigma_{\text{Stipendio} > 1300}(\pi_{\text{Nome,Sede,Stipendio}}(\text{Imp}))$$

ImpRicchi

Nome	Sede	Stipendio
Rossi	S01	2000
Verdi	S02	1500
Neri	S02	2500

2) Sedi, responsabili e città degli impiegati che guadagnano più di 1300 Euro

$$\pi_{\text{Sede,Responsabile,Citta}}(\text{Sedi} \bowtie (\sigma_{\text{Stipendio} > 1300}(\text{Imp})))$$

oppure: $\pi_{\text{Sede,Responsabile,Citta}}(\text{Sedi} \bowtie \text{ImpRicchi})$

Sede	Responsabile	Citta
S01	Biondi	Milano
S02	Mori	Bologna

3) Progetti nelle città delle sedi degli impiegati che guadagnano più di 1300 Euro

$$\pi_{\text{CodProg}}(\text{Prog} \bowtie (\text{Sedi} \bowtie \text{ImpRicchi}))$$

CodProg
P01
P02

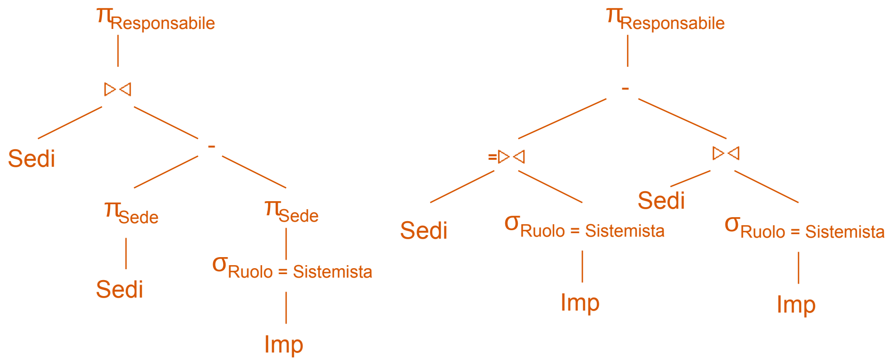
Espressioni: esempi (2)

4) Responsabili delle sedi senza sistemisti

Responsabile
Fulvi

$\pi_{Responsabile}(Sedi \bowtie (\pi_{Sede}(Sedi) - \pi_{Sede}(\sigma_{Ruolo = Sistemista}(Imp))))$

oppure: $\pi_{Responsabile}((Sedi =\!\bowtie (\sigma_{Ruolo = Sistemista}(Imp))) -$
$(Sedi \bowtie (\sigma_{Ruolo = Sistemista}(Imp))))$

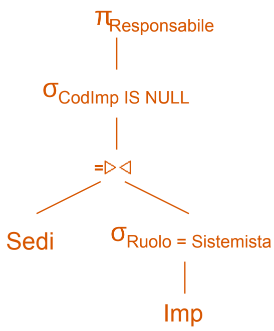


Espressioni: esempi (3)

ma anche (!!):

Responsabile
Fulvi

$\pi_{Responsabile}(\sigma_{CodImp\ IS\ NULL}(Sedi =\!\bowtie (\sigma_{Ruolo = Sistemista}(Imp))))$

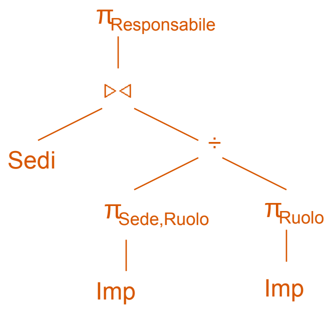


Espressioni: esempi (4)

5) Responsabili delle sedi in cui sono presenti tutti i ruoli

$\pi_{Responsabile}(Sedi \bowtie (\pi_{Sede,Ruolo}(Imp) \div \pi_{Ruolo}(Imp)))$

Responsabile
Biondi
Mori



Equivalenza di espressioni

- Un'interrogazione su un DB con schema **R** può a tutti gli effetti essere vista come una funzione che a ogni istanza **r** di **R** associa una relazione risultato con un dato schema
- Un'espressione dell'AR è quindi un modo specifico per esprimere (rappresentare) tale funzione, e due espressioni sono tra loro equivalenti se rappresentano la stessa funzione:

  Due espressioni E1 ed E2 espresse su un DB **R** si dicono equivalenti rispetto a **R** (E1 $\equiv_{\mathbf{R}}$ E2) se e solo se per ogni istanza **r** di **R** producono lo stesso risultato, E1(**r**) = E2(**r**)

- In alcuni casi l'equivalenza non dipende dallo schema **R** specifico, nel qual caso si scrive E1 $\equiv$ E2 (ossia vale E1 $\equiv_{\mathbf{R}}$ E2 per ogni schema **R**)

Esempio: si ha $\pi_{AB}(\sigma_{A=a}(R)) \equiv \sigma_{A=a}(\pi_{AB}(R))$, come è facile verificare; d'altronde $\pi_{AB}(R_1) \bowtie \pi_{BC}(R_2) \equiv_{\mathbf{R}} \pi_{ABC}(R_1 \bowtie R_2)$, poiché l'equivalenza è garantita solo se anche nel secondo caso il join è solo su B

Equivalenze: considerazioni

- Due espressioni equivalenti E1 ed E2 garantiscono lo stesso risultato, ma ciò non significa che la scelta sia indifferente in termini di “risorse” necessarie
- Considerazioni di questo tipo sono essenziali in fase di ottimizzazione (il discorso verrà ripreso in SI L-B), in cui la conoscenza delle regole di equivalenza può consentire di eseguire delle trasformazioni che possono portare a un’espressione valutabile in modo più efficiente rispetto a quella iniziale
- In particolare le regole più interessanti sono quelle che permettono di ridurre la cardinalità degli operandi e quelle che portano a una semplificazione dell’espressione (es.: $R \bowtie R \equiv R$ se non ci sono valori nulli)

Regole di equivalenza

Tra le regole base di equivalenza, si ricordano qui le seguenti:

- Il join naturale è commutativo e associativo:
  $E_1 \bowtie E_2 \equiv E_2 \bowtie E_1$ $(E_1 \bowtie E_2) \bowtie E_3 \equiv E_1 \bowtie (E_2 \bowtie E_3) \equiv E_1 \bowtie E_2 \bowtie E_3$
- Selezione e proiezione si possono raggruppare:
  $\sigma_{F1}(\sigma_{F2}(E)) \equiv \sigma_{F1 \text{ AND } F2}(E)$ $\pi_Y(\pi_{YZ}(E)) \equiv \pi_Y(E)$
- Selezione e proiezione commutano (F si riferisce solo ad attributi in Y):
  $\pi_Y(\sigma_F(E)) \equiv \sigma_F(\pi_Y(E))$
- “Push-down” della selezione rispetto al join (F è sullo schema di E_1):
  $\sigma_F(E_1 \bowtie E_2) \equiv \sigma_F(E_1) \bowtie E_2$

Riassumiamo:

- L'algebra relazionale (AR) è un linguaggio per DB costituito da un insieme di operatori che si applicano a una o più relazioni e che producono una relazione
- Gli operatori di base sono 6: selezione, proiezione, ridenominazione,: join naturale, unione e differenza. Sulla base di questi si possono poi definire altri operatori, quali divisione e theta-join
- La presenza di valori nulli porta a ridefinire la semantica del join naturale e a fare uso di una logica a tre valori (V,F,?) per calcolare il valore di verità di espressioni booleane con valori nulli
- L'outer-join (left, right e full) permette di includere nel risultato anche tuple dangling, completandole con valori nulli
- In generale, un'interrogazione sul DB può essere rappresentata in AR mediante diverse espressioni, tutte tra loro equivalenti dal punto di vista del risultato, ma non necessariamente dal punto di vista dell'efficienza