



## Algebra relazionale

**Dario Maio**  
<http://bias.csr.unibo.it/maio/>

---

algebra relazionale 



## Linguaggi di manipolazione 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 d'interrogazione:

- **Calcolo e algebra sono equivalenti in termini di potere espressivo** ("ciò che riescono a calcolare").
- L'**algebra relazionale (AR)** è la base per comprendere "come le interrogazioni vengano effettivamente elaborate da un RDBMS".

---

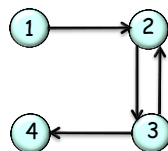
algebra relazionale 



## Algebra relazionale: premesse (1)

- Le limitazioni espressive dell'algebra (e quindi del calcolo) relazionale sono in parte dettate dall'esigenza di garantire una soluzione efficiente al problema dell'ottimizzazione delle interrogazioni, soluzione che non risulterebbe possibile nel caso di un linguaggio general-purpose.
- La principale limitazione dell'AR è legata all'impossibilità di esprimere interrogazioni ricorsive (il caso paradigmatico è il calcolo della chiusura transitiva di una relazione binaria).
- La relazione (Start,End), chiusura transitiva di (From,To), non è computabile mediante algebra o calcolo relazionale.

From	To
1	2
3	2
2	3
3	4



Start	End
1	2
3	2
2	3
3	4
1	3
1	4
2	4



## Algebra relazionale: premesse (2)

- 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**,  $\sigma$ , permette di selezionare un **sottoinsieme delle tuple di una relazione**, applicando a ciascuna di esse una formula booleana F.

Espressione:  $\sigma_F(R)$

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

- F si compone di **predicati** connessi da AND ( $\wedge$ ), OR ( $\vee$ ) e NOT ( $\neg$ ).
- Ogni predicato è del tipo  $A \theta c$  o  $A \theta B$ , dove:
  - A e B sono attributi in X;
  - $c \in \text{dom}(A)$  è una costante;
  - $\theta$  è un operatore di confronto,  $\theta \in \{=, \neq, <, >, \leq, \geq\}$ .


  
 algebra relazionale
   
 5

## 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


  
 algebra relazionale
   
 6

## 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

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

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

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

Giornata	Casa	Ospite	GolCasa	GolOspite
4	Venezia	Bologna	0	1

algebra relazionale  7

## Proiezione

- L'operatore di **proiezione**,  $\pi$ , è ortogonale alla selezione, in quanto permette di selezionare un **sottoinsieme Y degli attributi di una relazione**.


Espressione:  $\pi_Y(R)$

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

Input                      Output


Y

Proiezione





Y	X-Y

Selezione



X

algebra relazionale  8

 **Proiezione: esempi (1)**

Corsi



CodCorso	Titolo	CodDocente	Anno
483	Analisi	0201	1
729	Analisi	0021	1
913	Sistemi Informativi	0123	2


$\pi_{\text{CodCorso, CodDocente}}(\text{Corsi})$

CodCorso	CodDocente
483	0201
729	0021
913	0123

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

CodCorso	Anno
483	1
729	1
913	2

 algebra relazionale  9

 **Proiezione: esempi (2)**

Corsi



CodCorso	Titolo	CodDocente	Anno
483	Analisi	0201	1
729	Analisi	0021	1
913	Sistemi Informativi	0123	2

$\pi_{\text{Titolo}}(\text{Corsi})$

Titolo
Analisi
Sistemi Informativi

$\pi_{\text{CodDocente}}(\text{Corsi})$

Docente
0201
0021
0123

 algebra relazionale  10

## Proiezione: cardinalità del risultato

- In generale, la cardinalità di  $\pi_Y(r)$  è minore o uguale della cardinalità 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  $t_1$  e  $t_2$  tali che  $t_1[Y] = t_2[Y]$ .

(Solo se) Se  $Y$  non è superchiave allora è possibile costruire un'istanza legale  $r$  con due tuple distinte  $t_1$  e  $t_2$  tali che  $t_1[Y] = t_2[Y]$ . 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

esempio:  $\pi_{\text{CodDocente}}(\text{Corsi})$ .

algebra relazionale



11

## 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	CodDocente	Anno
483	Analisi	0201	1
729	Analisi	0021	1
913	Sistemi Informativi	0123	2

Esami  $\bowtie$  Corsi

Matricola	CodCorso	Voto	Lode	Titolo	CodDocente	Anno
29323	483	28	no	Analisi	0201	1
39654	729	30	sì	Analisi	0021	1
29323	913	26	no	Sistemi Informativi	0123	2
35467	913	30	no	Sistemi Informativi	0123	2

algebra relazionale



12

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

algebra relazionale



13

## Join naturale: esempi (1)

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

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  $\bowtie$  Linee

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

algebra relazionale



14


### Join naturale: esempi (2)

Voli  $\bowtie$  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  $\bowtie$  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


 algebra relazionale 15

### Join naturale: osservazioni

- È possibile che una tupla di una delle relazioni (operandi) non faccia match con nessuna tupla dell'altra relazione; in tal caso questa 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:  
 $\text{la cardinalità del join, } |r_1 \bowtie r_2|, \text{ è compresa tra } 0 \text{ e } |r_1| * |r_2|.$
- 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  $|r_1 \bowtie r_2| = |r_2|$ .

 algebra relazionale 16





## 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 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 ⋈ VoliNoSmoking

Codice	Data
SC278	28/07/2001


algebra relazionale

17



## Join naturale e prodotto Cartesiano

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

algebra relazionale

18



## Unione e differenza

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

Espressione:  $R_1 \cup R_2$

Schema	$R_1(X), R_2(X)$	X
Istanza	$r_1, r_2$	$r_1 \cup r_2 = \{t \mid t \in r_1 \text{ OR } t \in r_2\}$
Input		Output

Espressione:  $R_1 - R_2$

Schema	$R_1(X), R_2(X)$	X
Istanza	$r_1, r_2$	$r_1 - r_2 = \{t \mid t \in r_1 \text{ AND } t \notin r_2\}$
Input		Output

- N.B. L'intersezione si può anche scrivere come:  $r_1 \cap r_2 = r_1 - (r_1 - r_2)$ .



algebra relazionale

19



## 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

$VoliCharter \cup VoliNoSmoking$

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



algebra relazionale

20



## 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 effettuano l'unione e la differenza?

Studenti


Matricola	CodiceFiscale	Cognome	Nome	DataNascita
29323	BNCGRG78H21A944V	Bianchi	Giorgio	21/06/1978
35467	RSSNNA78L53G125J	Rossi	Anna	13/07/1978


Come si esegue il join? Redditi

CF	Imponibile
BNCGRG78H21A944V	10000



algebra relazionale

21



## Ridenominazione

- L'operatore di **ridenominazione**,  $\rho$ , 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:  $\rho_{Y \leftarrow X}(R)$


Schema

$R(XZ)$	$YZ$
Input	Output


X	Z


Ridenominazione

Y	Z



algebra relazionale

22



### Ridenominazione: esempi

**Redditi**

CF	Imponibile
BNCGRG78H21A944V	10000

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


CodiceFiscale	Imponibile
BNCGRG78H21A944V	10000


**VoliNoSmoking**

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


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

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



algebra relazionale 

23



### Self-join

- La ridenominazione permette di eseguire il join di una relazione con sé 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:**


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


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

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


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

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



algebra relazionale 

24



## 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**,  $\div$ , 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:  $R_1 \div R_2$


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

Input


Output




La divisione si può esprimere come:  $\pi_{X_1}(r_1) - \pi_{X_1}((\pi_{X_1}(r_1) \bowtie r_2) - r_1)$ .



algebra relazionale



25



## 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


$Voli \div Linee$

Codice	Data
	21/07/2001
	24/07/2001


$(Voli \div Linee) \bowtie Linee$

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".



algebra relazionale



26




## 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  $A \theta B$ , con  $A \in X_1$  e  $B \in X_2$ .

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



algebra relazionale

  
27



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

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



algebra relazionale

  
28

### *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 sono esplicitati*.
- In questo caso, per garantire l'univocità (distinguibilità) degli attributi nello schema risultato, è necessario adottare "alcuni trucchi" (ad es. usare il nome dello schema; 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**  $\bowtie$  **Prog**  
 (CodProgetto=Sigla) AND  
 (Ric.Nome  $\neq$  Prog.Nome)


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


  
 algebra relazionale 29

### *Algebra con valori nulli*

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


  
 algebra relazionale 30

  $\pi, \cup, -$  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

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


Nome	Ufficio
Rossi	A12
Verdi	NULL
Verdi	A27


**Responsabili**

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

**Impiegati  $\cup$  Responsabili**

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

algebra relazionale  31

  $\sigma$  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

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

- 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 vale per  $\sigma_{\text{Ufficio} \neq \text{A12}}(\text{Impiegati})$  !

algebra relazionale  32





### Logica a tre valori

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

**NOT**

V	F
F	V
?	?

**AND**

V	V	F	?
F	F	F	F
?	?	F	?


**OR**

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 valori NULL si introduce l'operatore di confronto **IS**, ad es. `A IS NULL`.
- `NOT ( A IS NULL)` si scrive anche `A IS NOT NULL`.

algebra relazionale

33



### Selezione con valori nulli: esempi

**Impiegati**

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

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

Cod	Nome	Ufficio
123	Rossi	A12

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

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

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

Cod	Nome	Ufficio
373	Verdi	A27

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

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

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


Cod	Nome	Ufficio
231	Verdi	NULL

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

Cod	Nome	Ufficio
-----	------	---------

algebra relazionale

34

 **▷◁ 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



algebra relazionale 35

 **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,  $r_1 \cap r_2 = r_1 \bowtie r_2$ , 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

Impiegati - Responsabili


Cod	Nome	Ufficio
231	Verdi	NULL
373	Verdi	A27

Responsabili

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

Impiegati - (Impiegati - Responsabili)


Cod	Nome	Ufficio
123	Rossi	A12
435	Verdi	NULL



algebra relazionale 36

## 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** ( $\Rightarrow \bowtie \Leftarrow$ ): sono incluse solo le tuple dangling dell'operando sinistro, e completate con null.
  - **Right** ( $\Leftarrow \bowtie \Rightarrow$ ): sono incluse solo le tuple dangling dell'operando destro, e completate con null.
  - **Full** ( $\Rightarrow \bowtie \Leftarrow$ ): sono considerate le tuple dangling di entrambi gli operandi, e completate con null.


  
 algebra relazionale

37

## Outer join: esempi

Ricercatori

Nome	CodProgetto
Rossi	HK27
Bianchi	HK27
Verdi	HK28

Progetti

CodProgetto	Responsabile
HK27	Bianchi
HAL2000	Neri

Ricercatori  $\Rightarrow \Leftarrow$  Progetti


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

Ricercatori  $\Rightarrow \Leftarrow$  Progetti

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

Ricercatori  $\Leftarrow \Rightarrow$  Progetti

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


  
 algebra relazionale

38

## 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.



- 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** =  $\sigma_{\text{Nome} = \text{Rossi}}(\text{Ricercatori} \bowtie \text{Progetti})$

algebra relazionale



39

## 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	1000
E007	Violetti	S01	Programmatore	700
E008	Aranci	S02	Programmatore	1200

Sedi

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


Prog

CodProg	Sede
P01	S01
P01	S02
P02	S02

algebra relazionale



40



### Espressioni: esempi (1)

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

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

oppure:

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

CodImp	Sede	Stipendio
E001	S01	2000
E002	S02	1500
E005	S02	2500

2) Sedi, responsabili e città dove vi sono impiegati che guadagnano più di 1300 Euro:

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

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

Sede	Responsabile	Città
S01	Biondi	Milano
S02	Mori	Bologna


3) Progetti nelle città delle sedi dove vi sono impiegati che guadagnano più di 1300 Euro:

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

CodProg
P01
P02

algebra relazionale

41



### Espressioni: esempi (2)

4) Responsabili delle sedi senza sistemisti

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

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

$\pi_{Responsabile}$

$\bowtie$

Sedi

$-$

$\pi_{Sede}$

Sedi

$\pi_{Sede}$

$\sigma_{Ruolo = Sistemista}$

Imp

$\pi_{Responsabile}$

$-$

$\bowtie$

Sedi

$\sigma_{Ruolo = Sistemista}$

Imp

Sedi

$\bowtie$

$\sigma_{Ruolo = Sistemista}$

Imp

algebra relazionale

42

### Espressioni: esempi (3)

ma anche (!!):

Responsabile
Fulvi

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

algebra relazionale 43

### Espressioni: esempi (4)

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

Responsabile
Biondi
Mori

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

algebra relazionale 44

## *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 costituisce quindi una modalità specifica 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  $R$  ( $E1 \equiv_R E2$ ) 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  $E1 \equiv_R E2$  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  $\pi_{AB}(R_1) \bowtie \pi_{BC}(R_2) \equiv_R \pi_{ABC}(R_1 \bowtie R_2)$ , 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 durante la fase di **ottimizzazione** delle interrogazioni; infatti la conoscenza delle **regole di equivalenza** può consentire di eseguire **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:

$$E_1 \bowtie E_2 \equiv E_2 \bowtie E_1 \quad (E_1 \bowtie E_2) \bowtie E_3 \equiv E_1 \bowtie (E_2 \bowtie E_3) \equiv E_1 \bowtie E_2 \bowtie E_3$$

- Selezione e proiezione si possono raggruppare:

$$\sigma_{F_1}(\sigma_{F_2}(E)) \equiv \sigma_{F_1 \text{ AND } F_2}(E) \quad \pi_Y(\pi_{YZ}(E)) \equiv \pi_Y(E)$$

- Selezione e proiezione commutano (F si riferisce esclusivamente 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  $E_1$ ):

$$\sigma_F(E_1 \bowtie E_2) \equiv \sigma_F(E_1) \bowtie E_2$$

algebra relazionale



47

## Sommario:

- 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 : **selezione**, **proiezione**, **ridenominazione** (operatori unari), **join naturale**, **unione** e **differenza** (operatori binari). 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, una **query** 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.

algebra relazionale



48