Algebra relazionale

Annalisa Franco, Dario Maio
Università di Bologna

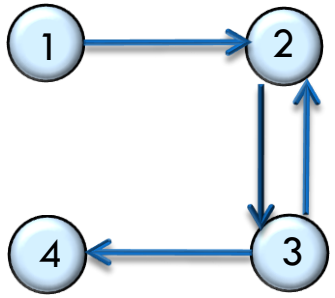
Linguaggi di manipolazione per DB

- Un linguaggio di manipolazione, o DML, permette di interrogare e modificare istanze di basi di dati.
- A parte i linguaggi utente, quali SQL, ne esistono altri, formalmente definiti, che rivestono notevole importanza in quanto enfatizzano gli aspetti "essenziali" dell'interazione con un DB relazionale.
- In particolare due linguaggi che si concentrano sugli aspetti d'interrogazione sono:
  - calcolo relazionale
    - linguaggio dichiarativo basato sulla logica dei predicati del primo ordine;
  - algebra relazionale
    - linguaggio procedurale di tipo algebrico i cui operandi sono relazioni.
  - Calcolo relazionale e algebra relazionale sono equivalenti in termini di potere espressivo ("ciò che possono calcolare").
  - L'algebra relazionale (AR) costituisce le basi formali per le operazioni del modello relazionale e per la loro implementazione in un RDBMS.
  - Il linguaggio SQL incorpora aspetti di calcolo relazionale e algebra relazionale.

Algebra relazionale: premesse

- Le limitazioni espressive dell'algebra e 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. L'insieme delle operazioni dell'AR non è Turing-completo.
- 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 né mediante algebra relazionale né con 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: introduzione

- L'algebra relazionale (AR) è costituita da un insieme di operatori di base che si applicano a una o più relazioni e che producono una relazione:
  - operatori di base unari: 1 selezione σ, 2 proiezione π, 3 ridenominazione ρ;
  - operatori di base binari: 4 unione $\cup$, 5 differenza $-$, 6 join (naturale) $\rhd\lhd$.
  - Altri operatori derivati possono essere definiti a partire da quelli di base.

- La semantica di ogni operatore si definisce specificando:
  - come lo schema (insieme di attributi) del risultato dipende dallo schema degli operandi;
  - come lo stato della relazione risultato dipende dagli stati delle relazioni 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 il momento, si assume che non siano presenti valori nulli.**

Completezza dell'insieme degli operatori

- Si dimostra che l'insieme $\{\sigma, \pi, \rho, \cup, -, \rhd\lhd\}$ degli operatori di base dell'algebra relazionale è completo, ovvero ogni altra operazione può essere espressa come composizione di operazioni di questo insieme.
- In altri testi si preferisce indicare come insieme di base $\{\sigma, \pi, \rho, \cup, -, \times\}$ essendo $\times$ il prodotto cartesiano.
- In realtà un join naturale può essere specificato con un prodotto cartesiano preceduto da una ridenominazione e seguito dalle operazioni di selezione e proiezione; anche un theta join (la forma più generale di join) può essere espresso come sottoinsieme del prodotto cartesiano $\times$.
- Dunque, ai fini del potere espressivo dell'AR le varie operazioni di join non sono strettamente necessarie, ma è importante considerarle separatamente perché sono più "comode" da usare e sono eseguite frequentemente nei RDBMS.
- In questa sede si sceglie $\{\sigma, \pi, \rho, \cup, -, \rhd\lhd\}$ come set degli operatori di base.

1 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
Stato	r	$\sigma_F(r) = \{ t \mid t \in r \text{ AND } F(t) = \text{vero} \}$

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

Valutazione della formula F

- Data una formula booleana F e una tupla t, per determinare se F(t) è vera si procede come appresso riportato.

- Per ogni predicato in F:
  - A θ c è vero per t se t[A] è in relazione θ con c

    (ad esempio: A $\neq$ c è vero se t[A] $\neq$ c)
  - A θ B è vero per t se t[A] è in relazione θ con t[B]

    (ad esempio: A $\geq$ B è vero se t[A] $\geq$ t[B])

  - Per gli operatori booleani $\wedge$, $\vee$ e $\neg$ valgono le regole dell'algebra di Boole.

Selezione: esempio (1)

Nome Relazione 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

$\sigma_{(Voto = 30)\ AND\ (Lode = 'no')}(ESAMI)$

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

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

Matricola	CodCorso	Voto	Lode
39654	729	30	sì
35467	913	30	no
31283	729	30	no

Selezione: esempio (2)

PARTITE

Giornata	Casa	Ospite	GolCasa	GolOspite
4	Crotone	Inter	0	2
4	Fiorentina	Bologna	2	1
5	Cagliari	Sassuolo	0	1
5	Bologna	Inter	1	1
5	Lazio	Napoli	1	4

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

Giornata	Casa	Ospite	GolCasa	GolOspite
5	Bologna	Inter	1	1

$\sigma_{(Ospite = \text{'Inter'})\ AND\ (GolCasa \leq GolOspite)}(PARTITE)$

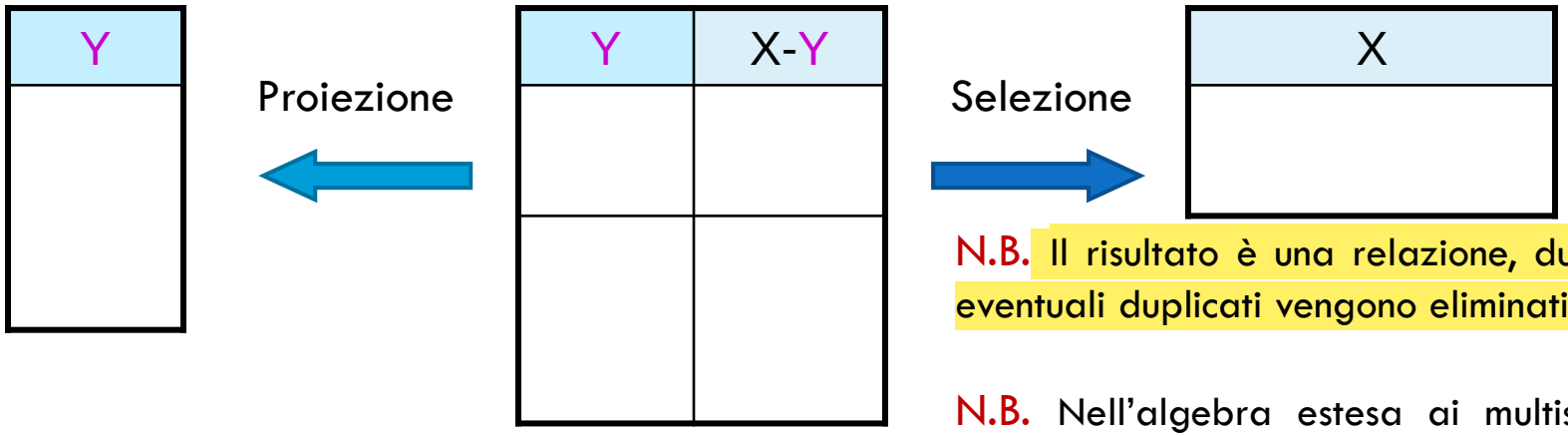
Giornata	Casa	Ospite	GolCasa	GolOspite
4	Crotone	Inter	0	2
5	Bologna	Inter	1	1

2 Proiezione

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

Espressione: $\pi_Y(R)$

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



N.B. Il risultato è una relazione, dunque eventuali duplicati vengono eliminati.

N.B. Nell'algebra estesa ai multiset si definisce anche l'operazione di proiezione senza eliminazione di repliche.

Proiezione: esempio (1)

CORSI

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

$\pi_{CodCorso,CodDocente}(CORSI)$

CodCorso	CodDocente
483	0201
729	0021
913	0123

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

CodCorso	Anno
483	1
729	1
913	2

Proiezione: esempio (2)

CORSI

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

$\pi_{Titolo}(CORSI)$

Titolo
Analisi
Sistemi Informativi

→ Eliminazione dei Duplicati

$\pi_{CodDocente}(CORSI)$

CodDocente
0201
0021
0123

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). perché la Superchiave ha valori Univoci
- Dimostrazione:
  - (Se) Se Y è una superchiave di R(X), in ogni stato 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 uno stato 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. Nell'esempio precedente: $\pi_{CodDocente}(CORSI)$

Join naturale

- L'operatore di join naturale, $\rhd\lhd$, combina le tuple di due relazioni sulla base dell'uguaglianza dei valori degli attributi comuni alle due relazioni, cioè quelli presenti in X1 $\cap$ X2.
- Ogni tupla che compare nel risultato del join naturale di r1 e r2, estensioni rispettivamente di R1(X1) e R2(X2), è ottenuta come combinazione ("match") di una tupla di r1 con una tupla di r2 sulla base dell'uguaglianza dei valori degli attributi comuni.
- Inoltre, lo schema della relazione risultato è l'unione X1 $\cup$ X2 degli schemi degli operandi.

Espressione: $R_1 \rhd\lhd R_2$

	Input	Output
Schema	$R_1(X_1), R_2(X_2)$	X_1X_2
Stati	r_1, r_2	$r_1 \rhd\lhd r_2 = \{ t \mid t[X_1] \in r_1 \text{ AND } t[X_2] \in r_2 \}$

Join naturale: esempio (1)

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

match

$X_1 \cap X_2 = \{CodCorso\}$

$X_1 \cup X_2 = \{Matricola, CodCorso, Voto, Lode, Titolo, CodDocente, Anno\}$

ESAMI $\triangleright\!\triangleleft$ 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

Join naturale: esempio (2)

VOLI

CodVolo	Data	CodComandante
AZ427	21/07/2017	C002314
AZ427	23/07/2017	C126721
AA056	21/07/2017	C201205

ROTTE

CodVolo	Partenza	Arrivo
AZ427	FCO	JFK
AA056	LAX	FCO

PRENOTAZIONI

CodVolo	Data	Posto	CodCliente
AZ427	21/07/2017	12A	BNCGRG84H21A944K
AZ427	21/07/2017	27B	DNMFNC52L70F839FU
AZ427	23/07/2017	14H	MAIMRA61P18F205P

VOLI ⋈ ROTTE

CodVolo	Data	CodComandante	Partenza	Arrivo
AZ427	21/07/2017	C002314	FCO	JFK
AZ427	23/07/2017	C126721	FCO	JFK
AA056	21/07/2017	C201205	LAX	FCO

Join naturale: esempio (3)

VOLI

CodVolo	Data	CodComandante
AZ427	21/07/2017	C002314
AZ427	23/07/2017	C126721
AA056	21/07/2017	C201205

ROTTE

CodVolo	Partenza	Arrivo
AZ427	FCO	JFK
AA056	LAX	FCO

PRENOTAZIONI

CodVolo	Data	Posto	CodCliente
AZ427	21/07/2017	12A	BNCGRG84H21A944K
AZ427	21/07/2017	27B	DNMFNC52L70F839FU
AZ427	23/07/2017	14H	MAIMRA61P18F205P

VOLI ▷◁ PRENOTAZIONI

CodVolo	Data	CodComandante	Posto	CodCliente
AZ427	21/07/2017	C002314	12A	BNCGRG84H21A944K
AZ427	21/07/2017	C002314	27B	DNMFNC52L70F839FU
AZ427	23/07/2017	C126721	14H	MAIMRA61P18F205P

Join naturale: esempio (4)

VOLI

CodVolo	Data	CodComandante
AZ427	21/07/2017	C002314
AZ427	23/07/2017	C126721
AA056	21/07/2017	C201205

ROTTE

CodVolo	Partenza	Arrivo
AZ427	FCO	JFK
AA056	LAX	FCO

PRENOTAZIONI

CodVolo	Data	Posto	CodCliente
AZ427	21/07/2017	12A	BNCGRG84H21A944K
AZ427	21/07/2017	27B	DNMFNC52L70F839FU
AZ427	23/07/2017	14H	MAIMRA61P18F205P

ROTTE ▷◁ PRENOTAZIONI

CodVolo	Partenza	Arrivo	Data	Posto	CodCliente
AZ427	FCO	JFK	21/07/2001	12A	BNCGRG84H21A944K
AZ427	FCO	JFK	21/07/2001	27B	DNMFNC52L70F839FU
AZ427	FCO	JFK	23/07/2001	14H	MAIMRA61P18F205P

Join naturale: proprietà e osservazioni

- Il join naturale è commutativo e associativo:
  - $r1 \bowtie r2 = r2 \bowtie r1$
  - $r1 \bowtie r2 \bowtie r3 = (r1 \bowtie r2) \bowtie r3$
- È possibile che una tupla di una delle relazioni (operandi) non faccia match con nessuna tupla dell'altra relazione; in tal caso questa tupla è denominata "dangling".
- Nel caso limite è quindi possibile che il risultato del join sia vuoto; all'altro estremo è possibile che ogni tupla di r1 si combini con ogni tupla di r2. Ne consegue che per la cardinalità del join, $|r1 \bowtie r2|$:

$$0 \leq |r1 \bowtie r2| \leq |r1| * |r2|$$

- Se il join è eseguito su una superchiave di R1(X1), allora ogni tupla di r2 fa match con al massimo una tupla di r1, quindi $|r1 \bowtie r2| \leq |r2|$.
- Se $X1 \cap X2$ è una chiave di R1(X1), e foreign key in R2(X2) (e quindi vi è un vincolo d'integrità referenziale) allora $|r1 \bowtie r2| = |r2|$. Questa affermazione è vera in assenza di valori nulli.

Join naturale: note cardinalità (1)

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 $|r_1 \triangleright\!\triangleleft r_2| \leq |r_2|$.

R1

A	**B**	**C**
1	X1	C2
2	Y4	C5
3	Z3	C2

R2

A	**B**	**D**
1	X1	D1
3	Z2	D3

Join naturale su AB che è superchiave di R1

R1 $\triangleright\!\triangleleft$ R2

A	**B**	**C**	**D**
1	X1	C2	D1

Con R2

A	**B**	**D**
1	X1	D1
3	Z3	D3

si ottiene invece

R1 $\triangleright\!\triangleleft$ R2

A	**B**	**C**	**D**
1	X1	C2	D1
3	Z3	C2	D3

Join naturale: note cardinalità (2)

Se $X_1 \cap X_2$ è una chiave di $R_1(X_1)$, e foreign key in $R_2(X_2)$ (e quindi vi è un vincolo d'integrità referenziale) allora $|r_1 \rhd\lhd r_2| = |r_2|$. Questa affermazione è vera in assenza di valori nulli.

R1

A	B	C
1	X1	C2
2	Y4	C5
3	Z3	C2

R2

A	D	E
1	D1	E1
3	D2	E3
3	D3	E3

Join naturale su A che è primary key di R1 e foreign key per R2

R1 $\rhd\lhd$ R2

A	B	C	D	E
1	X1	C2	D1	E1
3	Z3	C2	D2	E3
3	Z3	C2	D3	E3

Join naturale e intersezione

- Quando le due relazioni hanno lo stesso schema (X1=X2) allora due tuple fanno match se e solo se hanno lo stesso valore per tutti gli attributi, ovvero sono identiche, per cui:

se X1 = X2 il join naturale equivale all'intersezione (∩) delle due relazioni

VOLI_CHARTER

Codice	**Data**
IB123	21/01/2018
FR278	28/01/2018
VY338	18/02/2018

VOLI_NON_STOP

Codice	**Data**
FR278	28/01/2018
FR315	30/01/2018

VOLI_CHARTER ⋈ VOLI_NON_STOP

Codice	**Data**
FR278	28/01/2018

Join naturale e prodotto cartesiano

- Se non vi sono attributi in comune ($X1 \cap X2 = \emptyset$) allora, non essendovi condizioni di join, due tuple fanno sempre match, per cui:

  se $X1 \cap X2 = \emptyset$ il join naturale equivale al prodotto cartesiano

- Si noti che, a differenza del caso matematico, il prodotto cartesiano non è ordinato.

VOLI_CHARTER

Codice	Data
IB123	21/01/2018
FR278	28/01/2018
VY338	18/02/2018

VOLI_NON_STOP

Numero	Giorno
FR278	28/01/2018
FR315	30/01/2018

VOLI_NON_STOP ▷◁ VOLI_CHARTER

Codice	Data	Numero	Giorno
IB123	21/01/2018	FR278	28/01/2018
FR278	28/01/2018	FR278	28/01/2018
VY338	18/02/2018	FR278	28/01/2018
IB123	21/01/2018	FR315	30/01/2018
FR278	28/01/2018	FR315	30/01/2018
VY338	18/02/2018	FR315	30/01/2018

4 Unione e differenza 5

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

Espressione: $R_1 \cup R_2$

Schema	$R_1(X), R_2(X)$	X
Stati	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
Stati	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: $r1 \cap r2 = r1 - (r1 - r2)$.

Unione e differenza: esempi

VOLI_CHARTER

Codice	Data
IB123	21/01/2018
FR278	28/01/2018
VY338	18/02/2018

VOLI_NON_STOP

Codice	Data
FR278	28/01/2018
FR315	30/01/2018

VOLI_CHARTER $\cup$ VOLI_NON_STOP

Codice	Data
IB123	21/01/2018
FR278	28/01/2018
VY338	18/02/2018
FR315	30/01/2018

VOLI_CHARTER $-$ VOLI_NON_STOP

Codice	Data
IB123	21/01/2018
VY338	18/02/2018

VOLI_NON_STOP $-$ VOLI_CHARTER

Codice	Data
FR315	30/01/2018

N.B. Unione e intersezione sono operazioni commutative, mentre la differenza non è commutativa:

$R \cup S = S \cup R;\quad R \cap S = S \cap R;\quad R - S \neq S - R$

Intersezione: $r1 \cap r2 = r1 - (r1 - r2)$

L'intersezione $r_1 \cap r_2$ si può esprimere tramite l'operatore differenza: $r_1 \cap r_2 = r_1 - (r_1 - r_2)$. È pertanto un operatore derivato.

Togliamo da r1 ció che é uguale a r2

VOLI_CHARTER

r_1

Codice	Data
IB123	21/01/2018
FR278	28/01/2018
VY338	18/02/2018

VOLI_NON_STOP

r_2

Codice	Data
FR278	28/01/2018
FR315	30/01/2018

VOLI_CHARTER − VOLI_NON_STOP

Codice	Data
IB123	21/01/2018
VY338	18/02/2018

$r_1 - r_2$

VOLI_CHARTER ∩ VOLI_NON_STOP

Codice	Data
FR278	28/01/2018

$r_1 - (r_1 - r_2)$

Il problema dei nomi

- Il join naturale, l'unione e la differenza operano, seppur diversamente, sulla base degli attributi comuni a due schemi. Ciò comporta alcuni problemi come si può desumere dagli esempi appresso riportati.

VOLI_CHARTER

Codice	Data
IB123	21/01/2018
FR278	28/01/2018
VY338	18/02/2018

VOLI_NON_STOP

Numero	Giorno
FR278	28/01/2018
FR315	30/01/2018

Come si possono effettuare l'unione e la differenza?

IMPIEGATI

Matricola	CodiceFiscale	Cognome	Nome	DataNascita
29323	BNCGRG84H21A944K	Bianchi	Giorgio	21/06/1984
35467	RSSNNA90L53G125Z	Rossi	Anna	13/07/1990

Come si esegue il join?

REDDITI

CF	Imponibile
BNCGRG84H21A944K	27000

Prodotto cartesiano: chiarimenti (1)

- La definizione di prodotto cartesiano assume che gli insiemi degli attributi di R1 e R2 siano disgiunti, cioè $X1 \cap X2 = \emptyset$, ed è dunque coincidente con la definizione data per il join naturale.

Espressione: $R_1 \times R_2$

	Input	Output
Schema	$R_1(X_1)$, $R_2(X_2)$ con $X_1 \cap X_2 = \emptyset$	X_1X_2
Stati	r_1, r_2	$r_1 \rhd\lhd r_2 = \{ t \mid t[X_1] \in r_1 \text{ AND } t[X_2] \in r_2 \}$

- N.B. Se $X1 \cap X2 \neq \emptyset$ e se si vuole effettivamente eseguire un prodotto cartesiano si deve procedere a una ridenominazione degli attributi comuni, in modo da rendere diversi i loro nomi.

Esempio di cross product

PIETANZE

IdPietanza	Nome
101	Omelette con verdure
123	Insalata di pollo
321	Frittura di calamari

BEVANDE

IdBevanda	NomeBevanda
01	Calice di vino
04	Birra 33 cl
11	Acqua ½ litro

Possibili menu: PIETANZE × BEVANDE

IdPietanza	Nome	IdBevanda	NomeBevanda
101	Omelette con verdure	01	Calice di vino
101	Omelette con verdure	04	Birra 33 cl
101	Omelette con verdure	11	Acqua ½ litro
123	Insalata di pollo	01	Calice di vino
123	Insalata di pollo	04	Birra 33 cl
123	Insalata di pollo	11	Acqua ½ litro
321	Frittura di calamari	01	Calice di vino
321	Frittura di calamari	04	Birra 33 cl
321	Frittura di calamari	11	Acqua ½ litro

3 Ridenominazione

- L'operatore di ridenominazione, ρ, modifica lo schema di una relazione, cambiando i nomi di uno o più attributi. La definizione formale si omette per semplicità d'esposizione. È sufficiente ricordare che:
  - dato lo schema R(XZ), $\rho_{Y \leftarrow X}(R)$ cambia lo schema in YZ, lasciando invariati i valori delle tuple;
  - nel caso in cui si cambi il nome di più attributi, allora l'ordine in cui si elencano è significativo.

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

Schema	R(XZ)	YZ
	Input	Output

Ridenominazione

X	Z

Y	Z

- In alcuni testi l'operatore ρ ha anche una forma per modificare il nome della relazione, ad esempio: $\rho_{S(Y \leftarrow X)}(R)$ modifica R(XZ) in S(YZ).

Ridenominazione: esempi

REDDITI

CF	Imponibile
BNCGRG84H21A944K	27000

VOLI_NON_STOP

Numero	Giorno
FR278	28/01/2018
FR315	30/01/2018

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

CodiceFiscale	Imponibile
BNCGRG84H21A944K	27000

$\rho_{Codice,Data \leftarrow Numero,Giorno}(VOLI\_NON\_STOP)$

Codice	Data
FR278	28/01/2018
FR315	30/01/2018

Self-join

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

GENITORI

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

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

Attributi da Cambiare

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

Per trovare nonni e nipoti: $\rho_{Nonno,Genitore \leftarrow Genitore,Figlio}(GENITORI) \bowtie GENITORI$

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

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

Self-join: un altro esempio

Trovare gli impiegati che lavorano allo stesso progetto a cui lavora Rossi.

IMPIEGATI

ID	Cognome	Progetto
1	Rossi	A
2	Neri	A
3	Neri	B
4	Bianchi	B

$\rho_{ID1,Imp \leftarrow ID,Cognome}(IMPIEGATI)$

ID1	Imp	Progetto
1	Rossi	A
2	Neri	A
3	Neri	B
4	Bianchi	B

$R = \rho_{ID1,Imp \leftarrow ID,Cognome}(IMPIEGATI) \triangleright\!\triangleleft IMPIEGATI$

ID1	Imp	Progetto	ID	Cognome
1	Rossi	A	1	Rossi
1	Rossi	A	2	Neri
2	Neri	A	1	Rossi
2	Neri	A	2	Neri
3	Neri	B	3	Neri
3	Neri	B	4	Bianchi
4	Bianchi	B	3	Neri
4	Bianchi	B	4	Bianchi

$\sigma_{Imp='Rossi'}(R)$

per eliminare Rossi dal risultato

$\pi_{ID,Cognome}(\sigma_{Imp='Rossi' \text{ and } Cognome<>'Rossi'}(R))$

<> = significa "not equal to"

Operatori derivati: la divisione

- Gli operatori sinora visti definiscono completamente l'algebra relazionale. Tuttavia, per praticità, è talvolta utile ricorrere ad altri operatori "derivati", quali la divisione e il theta-join.
- La divisione, $\div$, di r1 per r2 , con r1 su R1(X1X2) e r2 su R2(X2), è il più grande insieme di tuple $t \in \pi_{X_1}(r_1)$, e dunque con schema X1, tale che, facendo il prodotto cartesiano con r2, ciò che si ottiene è una relazione contenuta in r1 o uguale a r1.

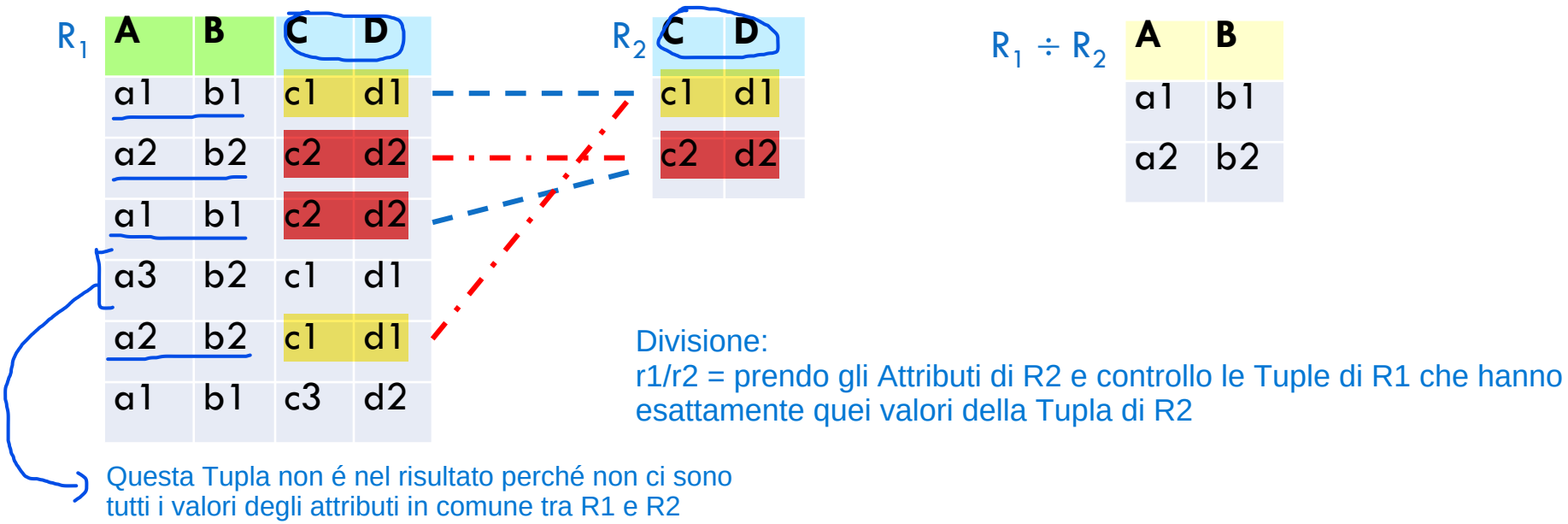
Espressione: $R_1 \div R_2$

	Input	Output
Schema	$R_1(X_1\ X_2), R_2(X_2)$	X_1
Stati	r_1, r_2	$r_1 \div r_2 = \{ t \mid \{t\} \rhd\lhd r_2 \subseteq r_1 \}$

In modo equivalente si definisce $r_1 \div r_2 = \{ t \mid t \in \pi_{X_1}(r_1) \wedge \forall\, u \in r_2 (tu \in r_1) \}$

$R_1 \div R_2$ si può esprimere come: $\pi_{X_1}(R_1) - \pi_{X_1}((\pi_{X_1}(R_1) \rhd\lhd R_2) - R_1)$.

Divisione: esempio (1) - a



In generale, la divisione è utile per interrogazioni di tipo "universale".

Divisione: esempio (1) - b

$R_1 \div R_2$ si può esprimere come: $\pi_{X_1}(R_1) - \pi_{X_1}((\pi_{X_1}(R_1) \rhd\lhd R_2) - R_1)$ $X_1 = \{A,B\}$ $X_2 = \{C,D\}$

R_1

A	B	C	D
a1	b1	c1	d1
a2	b2	c2	d2
a1	b1	c2	d2
a3	b2	c1	d1
a2	b2	c1	d1
a1	b1	c3	d2

R_2

C	D
c1	d1
c2	d2

$\pi_{X_1}(R_1)$

A	B
a1	b1
a2	b2
a3	b2

$\pi_{X_1}(R_1) \rhd\lhd R_2$

A	B	C	D
a1	b1	c1	d1
a1	b1	c2	d2
a2	b2	c1	d1
a2	b2	c2	d2
a3	b2	c1	d1
a3	b2	c2	d2

$(\pi_{X_1}(R_1) \rhd\lhd R_2) - R_1$

A	B	C	D
a3	b2	c2	d2

$\pi_{X_1}((\pi_{X_1}(R_1) \rhd\lhd R_2) - R_1)$

A	B
a3	b2

$R_1 \div R_2 = \pi_{X_1}(R_1) - \pi_{X_1}((\pi_{X_1}(R_1) \rhd\lhd R_2) - R_1)$

A	B
a1	b1
a2	b2

Divisione: esempio (2)

VOLI

Codice	Data
AZ427	21/07/2017
AZ427	23/07/2017
AZ427	24/07/2017
AA056	21/07/2017
AA056	24/07/2017
AA056	25/07/2017

LINEE_VOLI

Codice
AZ427
AA056

VOLI ÷ LINEE_VOLI

Data
21/07/2017
24/07/2017

- La divisione trova le date con voli effettuati da tutte le linee aeree.

(VOLI ÷ LINEE_VOLI) ▷◁ LINEE_VOLI

Codice	Data
AZ427	21/07/2017
AZ427	24/07/2017
AA056	21/07/2017
AA056	24/07/2017

Divisione: esempio (3)

MANSIONI

Tecnico	Reparto
Bianchi	Produzione
Bianchi	Vendite
Gialli	Marketing
Gialli	Produzione
Gialli	Vendite
Neri	Produzione
Neri	Vendite
Rossi	Marketing
Rossi	Produzione
Rossi	Vendite

REPARTI

Reparto
Marketing
Produzione
Vendite

MANSIONI ÷ REPARTI

TECNICO
Gialli
Rossi

La divisione trova i tecnici che lavorano in tutti i reparti.

Operatori derivati: theta-join

- L'operatore theta-join, $\bowtie_F$, è la combinazione di prodotto cartesiano e selezione:

Espressione: $R_1 \bowtie_F R_2$

	Input	Output
Schema	$R_1(X_1)$, $R_2(X_2)$ con $X_1 \cap X_2 = \emptyset$	X_1X_2
Stati	r_1, r_2	$r_1 \bowtie_F r_2 = \sigma_F(r_1 \bowtie r_2)$

con R_1 e R_2 senza attributi in comune e F formula composta di "predicati di join", ossia del tipo $A\ \theta\ B$, con $A \in X_1$ e $B \in X_2$.

Quindi solo Predicati tra Attributi, no costanti

- Se F è una congiunzione di uguaglianze, si parla più propriamente di equijoin (o equi-join).
- Il natural join può essere simulato per mezzo della ridenominazione, dell'equijoin e della proiezione.
- Il theta-join e il join naturale sono detti anche inner join.

Theta-join: esempi

PARTECIPAZIONI

CodRicercatore	CodProgetto
115623	HK27
100104	HAL2000
116232	HK27
100104	HK28
201401	HAL2000

PARTECIPAZIONI $\triangleright\triangleleft_{\text{CodProgetto=Sigla}}$ PROGETTI

CodRicercatore	CodProgetto	Sigla	CodResponsabile
115623	HK27	HK27	116232
100104	HAL2000	HAL2000	201401
116232	HK27	HK27	116232
100104	HK28	HK28	100104
201401	HAL2000	HAL2000	201401

PROGETTI

Sigla	CodResponsabile
HK27	116232
HAL2000	201401
HK28	100104

PARTECIPAZIONI $\triangleright\triangleleft_{\text{(CodProgetto=Sigla) AND (CodRicercatore} \neq \text{CodResponsabile)}}$ PROGETTI

CodRicercatore	CodProgetto	Sigla	CodResponsabile
115623	HK27	HK27	116232
100104	HAL2000	HAL2000	201401

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 RDBMS), 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 accorgimenti", ad esempio usare anche il nome dello schema per denotare un attributo.

PARTECIPAZIONI

CodRicercatore	CodProgetto
115623	HK27
116232	HK27
100104	HK28

PROGETTI

Sigla	CodRicercatore
HK27	116232
HK28	100104

PARTECIPAZIONI $\triangleright\triangleleft_{\text{(CodProgetto=Sigla) AND (PARTECIPAZIONI.CodRicercatore} \neq \text{PROGETTI.CodRicercatore)}}$ PROGETTI

RICERCATORI.CodRicercatore	CodProgetto	Sigla	PROGETTI.CodRicercatore
115623	HK27	HK27	116232

Theta-join: un esempio d'uso di self join (1)

- Dato lo schema ABBONAMENTI(Provider, CostoAnnuo) trovare i Provider il cui abbonamento ha costo annuo minimo.

ABBONAMENTI

Provider	CostoAnnuo
A	100
B	120
C	100
D	110
E	130

1) Si opera una ridenominazione $\rho_{P,C \leftarrow Provider,CostoAnnuo}(ABBONAMENTI)$

2) Si esegue il theta-join tra ABBONAMENTI e la sua ridenominazione

$T = ABBONAMENTI \triangleright\!\triangleleft_{(CostoAnnuo>C)} (\rho_{P,C \leftarrow Provider,CostoAnnuo}(ABBONAMENTI))$

3) Si effettua la differenza $\pi_{Provider}(ABBONAMENTI) - \pi_{Provider}(T)$

Theta-join: un esempio d'uso di self join (2)

ABBONAMENTI

Provider	CostoAnnuo
A	100
B	120
C	100
D	110
E	130

$A = \rho_{P,C \leftarrow Provider,CostoAnnuo}(ABBONAMENTI)$

P	C
A	100
B	120
C	100
D	110
E	130

$T = ABBONAMENTI \triangleright\!\triangleleft_{(CostoAnnuo>C)} A$

Provider	CostoAnnuo	P	C
B	120	A	100
B	120	C	100
B	120	D	110
D	110	A	100
D	110	C	100
E	130	A	100
E	130	B	120
E	130	C	100
E	130	D	110

$\pi_{Provider}(ABBONAMENTI)$

Provider
A
B
C
D
E

$\pi_{Provider}(T)$

Provider
B
D
E

Provider Piú Costosi

$\pi_{Provider}(ABBONAMENTI) - \pi_{Provider}(T)$

Provider
A
C

Semijoin

Tra i due Schemi, eseguo prima il Join Naturale e poi faccio la Proiezione sugli Attributi del primo schema (Left Join) oppure del secondo schema (Right Join)

- Il semijoin (o semi-join) da S a R, indicato con $R \ltimes S$, è la proiezione del natural join $R \rhd\lhd S$ sugli attributi dello schema R; è detto anche left semijoin.

Espressione: $R \ltimes S$

	Input	Output
Schema	R (X), S (Y)	X
Stati	r, s	$r \ltimes s = \pi_X(r \rhd\lhd s) = r \rhd\lhd \pi_{X \cap Y}(s)$

- Si definisce anche il right semijoin $S \rtimes R$ che equivale a $R \ltimes S$
- Il semijoin è utile in ambiente distribuito in quanto, se r ed s sono su nodi diversi della rete, consente di ridurre la mole dei dati da trasferire; infatti vale la seguente proprietà:

$$(r \ltimes s) \rhd\lhd s = (s \ltimes r) \rhd\lhd r = r \rhd\lhd s$$

N.B. In generale il semijoin non è simmetrico: $(r \ltimes s) \neq (s \ltimes r)$
La definizione di semijoin si può estendere anche al theta-join.

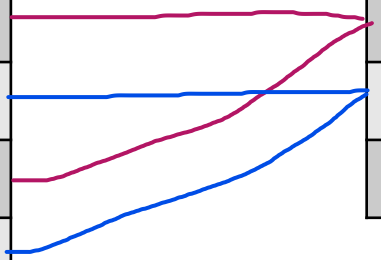
Semijoin: esempio

IMPIEGATI

IdImp	Cognome	Nome	Qualifica
100	Bianchi	Mario	1
200	Neri	Carlotta	2
250	Rossi	Giorgio	1
300	Verdi	Maria	2

QUAL_STIP

Qualifica	Stipendio
1	18000
2	22500
3	30000



- Il semijoin QUAL_STIP ⋉ IMPIEGATI con schema IQ(Qualifica,Stipendio) corrisponde alle qualifiche per le quali vi è almeno un impiegato che percepisce stipendio.

Qualifica	Stipendio	IdImp	Cognome	Nome
1	18000	100	Bianchi	Mario
1	18000	250	Rossi	Giorgio
2	22500	200	Neri	Carlotta
2	22500	300	Verdi	Maria

QUAL_STIP ▷◁ IMPIEGATI

IQ

Qualifica	Stipendio
1	18000
2	22500

$\pi_{Qualifica,Stipendio}$ (QUAL_STIP ▷◁ IMPIEGATI)

≡

QUAL_STIP ⋉ IMPIEGATI

Algebra con valori nulli

- La presenza di valori nulli nelle relazioni richiede un'estensione della semantica degli operatori. Si ricorda d'altra parte quanto sia importante, ai fini pratici, la gestione dei valori nulli.

- 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 presentato in questa sede è quello "tradizionale" e ha il pregio di essere molto simile a quello adottato in SQL, e quindi dai DBMS relazionali.

π, $\cup$, $-$ con i valori nulli

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

N.B. Nell'esempio per motivi di spazio nella slide si omettono altri attributi (es. nome di un impiegato).

IMPIEGATI

CodImp	Cognome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
435	NULL	A35
521	Verdi	NULL

RESPONSABILI

CodImp	Cognome	Ufficio
123	Rossi	A12
NULL	NULL	A27
435	NULL	A35

$\pi_{Cognome,Ufficio}$(IMPIEGATI)

Cognome	Ufficio
Rossi	A12
Verdi	NULL
Verdi	A27
NULL	A35

IMPIEGATI $\cup$ RESPONSABILI

CodImp	Cognome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
435	NULL	A35
521	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. Si consideri ad esempio la selezione

$$\sigma_{\text{Ufficio} = \text{'A12'}}(\text{IMPIEGATI})$$

e lo stato della relazione

IMPIEGATI

CodImp	Cognome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27

- 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 la selezione $\sigma_{\text{Ufficio} \neq \text{'A12'}}(\text{IMPIEGATI})$

Logica a tre valori

- Oltre ai valori di verità Vero (V) e Falso (F), si introduce il valore "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 operare esplicitamente con i valori NULL si introduce l'operatore di confronto IS, ad esempio: A IS NULL.
- NOT (A IS NULL) si scrive anche A IS NOT NULL.

Selezione con valori nulli: esempi

IMPIEGATI

CodImp	Cognome	Ufficio
123	Rossi	A12
231	Verdi	NULL
373	Verdi	A27
385	NULL	A27

$\sigma_{\text{Ufficio = 'A12'}}$ (IMPIEGATI)

CodImp	Cognome	Ufficio
123	Rossi	A12

$\sigma_{\text{(Ufficio = 'A12') OR (Ufficio ≠ 'A12')}}$(IMPIEGATI)

CodImp	Cognome	Ufficio
123	Rossi	A12
373	Verdi	A27
385	NULL	A27

$\sigma_{\text{(Ufficio = 'A27') AND (Cognome = 'Verdi')}}$(IMPIEGATI)

CodImp	Cognome	Ufficio
373	Verdi	A27

$\sigma_{\text{(Ufficio = 'A27') OR (Cognome = 'Verdi')}}$(IMPIEGATI)

CodImp	Cognome	Ufficio
231	Verdi	NULL
373	Verdi	A27
385	NULL	A27

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

CodImp	Cognome	Ufficio
231	Verdi	NULL

$\sigma_{\text{(Ufficio IS NULL) AND (Cognome IS NULL)}}$(IMPIEGATI)

CodImp	Cognome	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

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12
231	Verdi	5	NULL
373	Verdi	6	A27
435	NULL	4	A35
521	Verdi	NULL	A35

DIRIGENTI

CodImp	Livello	Ufficio
123	7	A12
NULL	8	A27
521	NULL	A35

IMPIEGATI ▷◁ DIRIGENTI

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12

Join ≠ intersezione con valori nulli (1)

- In assenza di valori nulli l'intersezione di r_1 e r_2 si può esprimere in due modi:
  - mediante il join naturale, $r_1 \cap r_2 = r_1 \bowtie r_2$;
  - 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

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12
231	Verdi	5	NULL
373	Verdi	6	A27
435	NULL	4	A35
521	Verdi	NULL	A35

DIRIGENTI

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12
NULL	NULL	8	A27
521	Verdi	NULL	A35

IMPIEGATI $\bowtie$ DIRIGENTI

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12

Join ≠ intersezione con valori nulli (2)

IMPIEGATI

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12
231	Verdi	5	NULL
373	Verdi	6	A27
435	NULL	4	A35
521	Verdi	NULL	A35

DIRIGENTI

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12
NULL	NULL	8	A27
521	Verdi	NULL	A35

IMPIEGATI – (IMPIEGATI – DIRIGENTI)

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12
521	Verdi	NULL	A35

IMPIEGATI – DIRIGENTI

CodImp	Cognome	Livello	Ufficio
231	Verdi	5	NULL
373	Verdi	6	A27
435	NULL	4	A35

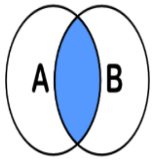
IMPIEGATI ▷◁ DIRIGENTI

CodImp	Cognome	Livello	Ufficio
123	Rossi	7	A12

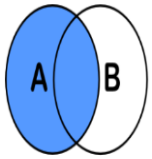
Outer join

- In alcuni casi è utile che anche le tuple dangling di un join compaiano nel risultato.
- A tale scopo si introduce l'operatore outer join (detto anche external join) che "completa" con valori nulli le tuple dangling.
- Esistono tre varianti:
  - Left outer join ($=\triangleright\triangleleft$): sono incluse solo le tuple dangling dell'operando sinistro, e completate con NULL.
  - Right outer join ($\triangleright\triangleleft=$): sono incluse solo le tuple dangling dell'operando destro, e completate con NULL.
  - Full outer join ($=\triangleright\triangleleft=$): sono considerate le tuple dangling di entrambi gli operandi, e completate con NULL.

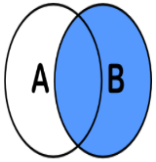
Inner join



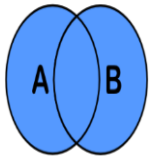
Left outer join



Right outer join



Full outer join



Left outer join: esempio A

CLIENTI

CF	CodComune
BNCGRG84H21A944K	A347
TRLFNC60L31G713L	A944
MAIMRA61P18F839O	A347
RLFRC72L60G713J	M185

FORNITORI

CodFornitore	CodComune
F001	A347
F002	G125
F003	A944
F004	H501

CLIENTI =▷◁ FORNITORI

CF	CodComune	CodFornitore
BNCGRG84H21A944K	A347	F001
TRLFNC60L31G713L	A944	F003
MAIMRA61P18F839O	A347	F001
RLFRC72L60G713J	M185	NULL

Right outer join: esempio A

CLIENTI

CF	CodComune
BNCGRG84H21A944K	A347
TRLFNC60L31G713L	A944
MAIMRA61P18F839O	A347
RLFRC72L60G713J	M185

FORNITORI

CodFornitore	CodComune
F001	A347
F002	G125
F003	A944
F004	H501

CLIENTI ▷◁= FORNITORI

N.B. Il risultato non ammette una chiave primaria sulla base degli attributi definiti. Nei RDBMS si dispone in ogni caso di un identificatore di riga.

CF	CodComune	CodFornitore
BNCGRG84H21A944K	A347	F001
MAIMRA61P18F839O	A347	F001
NULL	G125	F002
TRLFNC60L31G713L	A944	F003
NULL	H501	F004

Full outer join: esempio A

CLIENTI

CF	CodComune
BNCGRG84H21A944K	A347
TRLFNC60L31G713L	A944
MAIMRA61P18F839O	A347
RLFRC72L60G713J	M185

FORNITORI

CodFornitore	CodComune
F001	A347
F002	G125
F003	A944
F004	H501

CLIENTI =▷◁= FORNITORI

N.B. Il risultato non ammette una chiave primaria.

CF	CodComune	CodFornitore
BNCGRG84H21A944K	A347	F001
TRLFNC60L31G713L	A944	F003
MAIMRA61P18F839O	A347	F001
RLFRC72L60G713J	M185	NULL
NULL	G125	F002
NULL	H501	F004

Left outer join: esempio B

STUDENTI

CodStudente	CodEsercizio
S001	E001
S002	E002
S003	NULL
S004	E001

ESERCIZI

CodEsercizio	Argomento
E001	Algebra relazionale
E002	Entity/Relationship
E003	Normalizzazione

STUDENTI =▷◁ ESERCIZI

CodStudente	CodEsercizio	Argomento
S001	E001	Algebra relazionale
S002	E002	Entity/Relationship
S003	NULL	NULL
S004	E001	Algebra relazionale

Right outer join: esempio B

STUDENTI

CodStudente	CodEsercizio
S001	E001
S002	E002
S003	NULL
S004	E001

ESERCIZI

CodEsercizio	Argomento
E001	Algebra relazionale
E002	Entity/Relationship
E003	Normalizzazione

STUDENTI ▷◁= ESERCIZI

N.B. Il risultato non ammette una chiave primaria.

CodStudente	CodEsercizio	Argomento
S001	E001	Algebra relazionale
S002	E002	Entity/Relationship
S004	E001	Algebra relazionale
NULL	E003	Normalizzazione

Full outer join: esempio B

STUDENTI

CodStudente	CodEsercizio
S001	E001
S002	E002
S003	NULL
S004	E001

ESERCIZI

CodEsercizio	Argomento
E001	Algebra relazionale
E002	Entity/Relationship
E003	Normalizzazione

STUDENTI =▷◁= ESERCIZI

N.B. Il risultato non ammette una chiave primaria.

CodStudente	CodEsercizio	Argomento
S001	E001	Algebra relazionale
S002	E002	Entity/Relationship
S003	NULL	NULL
S004	E001	Algebra relazionale
NULL	E003	Normalizzazione

Left outer join: esempio C

PARTECIPAZIONI

CodRicercatore	CodProgetto
115623	HK27
116232	HK27
100104	HK28
201401	HAL2000

PROGETTI

CodProgetto	CodResponsabile
HK27	116232
HAL2000	201401
HK28	NULL
PLUS	201401

PARTECIPAZIONI =▷◁ PROGETTI

In questo caso coincide con il join naturale.

CodRicercatore	CodProgetto	CodResponsabile
115623	HK27	116232
116232	HK27	116232
100104	HK28	NULL
201401	HAL2000	201401

Right outer join: esempio C

PARTECIPAZIONI

CodRicercatore	CodProgetto
115623	HK27
116232	HK27
100104	HK28
201401	HAL2000

PROGETTI

CodProgetto	CodResponsabile
HK27	116232
HAL2000	201401
HK28	NULL
PLUS	201401

PARTECIPAZIONI ▷◁= PROGETTI

CodRicercatore	CodProgetto	CodResponsabile
115623	HK27	116232
116232	HK27	116232
201401	HAL2000	201401
100104	HK28	NULL
NULL	PLUS	20141

N.B. Il risultato non ammette una chiave primaria.

Full outer join: esempio C

PARTECIPAZIONI

CodRicercatore	CodProgetto
115623	HK27
116232	HK27
100104	HK28
201401	HAL2000

PROGETTI

CodProgetto	CodResponsabile
HK27	116232
HAL2000	201401
HK28	NULL
PLUS	201401

PARTECIPAZIONI =▷◁= PROGETTI

CodRicercatore	CodProgetto	CodResponsabile
115623	HK27	116232
116232	HK27	116232
100104	HK28	NULL
201401	HAL2000	201401
NULL	PLUS	20141

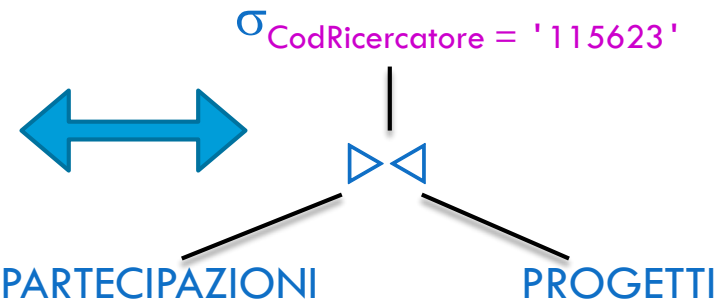
In questo caso coincide con il right outer join.

N.B. Il risultato non ammette una chiave primaria.

Espressioni

- Gli operatori dell'AR si possono liberamente combinare tra loro, avendo cura di rispettare le regole stabilite per la loro applicabilità.
- È anche possibile, oltre alla rappresentazione "lineare", adottare una rappresentazione grafica in cui l'espressione è rappresentata con un albero.
- La valutazione di un'espressione procede "bottom-up".

$\sigma_{CodRicercatore = '115623'}(PARTECIPAZIONI \bowtie PROGETTI)$

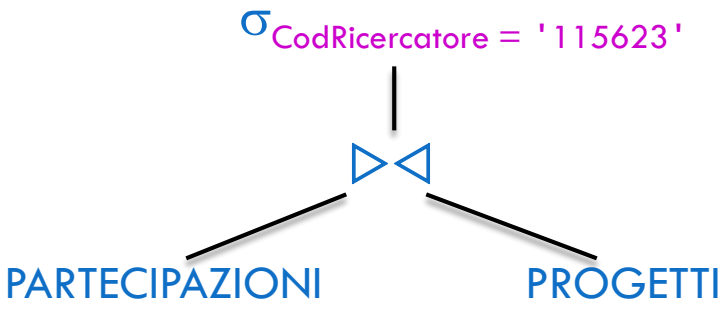


Viste

- 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.
- La sintassi è V := E dove V è il nome della vista ed E è l'espressione.

$$\text{PROGETTI\_115623} := \sigma_{\text{CodRicercatore = '115623'}} (\text{PARTECIPAZIONI} \rhd\lhd \text{PROGETTI})$$

PROGETTI_115623 :=



DB di riferimento per gli esempi

IMPIEGATI

CodImpiegato	Nome	Cognome	Sede	Ruolo	Stipendio
E001	Carlo	Rossi	S01	Analista	2000
E002	Mario	Verdi	S02	Sistemista	1500
E003	Maria	Bianchi	S01	Programmatore	1000
E004	Caterina	Gialli	S03	Programmatore	1000
E005	Ennio	Neri	S02	Analista	2500
E006	Flavio	Grigi	S01	Sistemista	1400
E007	Giuseppe	Biondi	S01	Responsabile	3200
E008	Giorgia	Mori	S02	Responsabile	3000
E009	Carlo	Fulvi	S03	Responsabile	3500

SEDI

Sede	CodResponsabile	Città
S01	E007	Milano
S02	E008	Bologna
S03	E009	Milano

PROGETTI

CodProg	Sede
P01	S01
P01	S02
P02	S02

Espressioni: esempio (1)

Codice, cognome, nome, sede e stipendio degli impiegati che non ricoprono il ruolo di responsabile e che guadagnano più di 1300 Euro

IMPIEGATI_TOP:=

$\pi_{CodImpiegato,Nome,Cognome,Sede,Stipendio}(\sigma_{(Stipendio > 1300)\ AND\ (Ruolo \neq 'Responsabile')}(IMPIEGATI))$

oppure:

IMPIEGATI_TOP:=

$\sigma_{(Stipendio > 1300)\ AND\ (Ruolo \neq 'Responsabile')}(\pi_{CodImpiegato,Nome,Cognome,Sede,Ruolo,Stipendio}(IMPIEGATI))$

IMPIEGATI_TOP

CodImpiegato	Nome	Cognome	Sede	Stipendio
E001	Carlo	Rossi	S01	2000
E002	Mario	Verdi	S02	1500
E005	Ennio	Neri	S02	2500
E006	Flavio	Grigi	S01	1400

N.B. La tabella in figura corrisponde alla prima espressione; la seconda infatti porterebbe a uno schema che include anche l'attributo Ruolo.

Espressioni: esempio (2)

Sede, città, e codice del responsabile per ogni sede dove vi sono impiegati, non responsabili, che guadagnano più di 1300 €:

$$SEDI \triangleright\triangleleft (\pi_{Sede}(\sigma_{(Stipendio > 1300)\ AND\ (Ruolo \neq \text{'Responsabile'})}(IMPIEGATI)))$$

oppure:

$$\pi_{Sede,CodResponsabile,Città}(SEDI \triangleright\triangleleft IMPIEGATI\_TOP)$$

Sede	CodResponsabile	Città
S01	E007	Milano
S02	E008	Bologna

- Per ottenere anche il nome e cognome del responsabile si deve eseguire un altro join:

$$TEMP := (\pi_{Sede,CodResponsabile,Città}(SEDI \triangleright\triangleleft IMPIEGATI\_TOP)) \triangleright\triangleleft_{CodImpiegato=CodResponsabile} IMPIEGATI$$

$$\pi_{SEDI.Sede,CodResponsabile,Città,Nome,Cognome}(TEMP)$$

SEDI.Sede	SEDI.CodResponsabile	SEDI.Città	IMPIEGATI.Nome	IMPIEGATI.Cognome
S01	E007	Milano	Giuseppe	Biondi
S02	E008	Bologna	Giorgia	Mori

Espressioni: esempi (3 e 4)

Progetti e città nelle sedi dove vi sono impiegati, non responsabili, che guadagnano più di 1300 Euro:

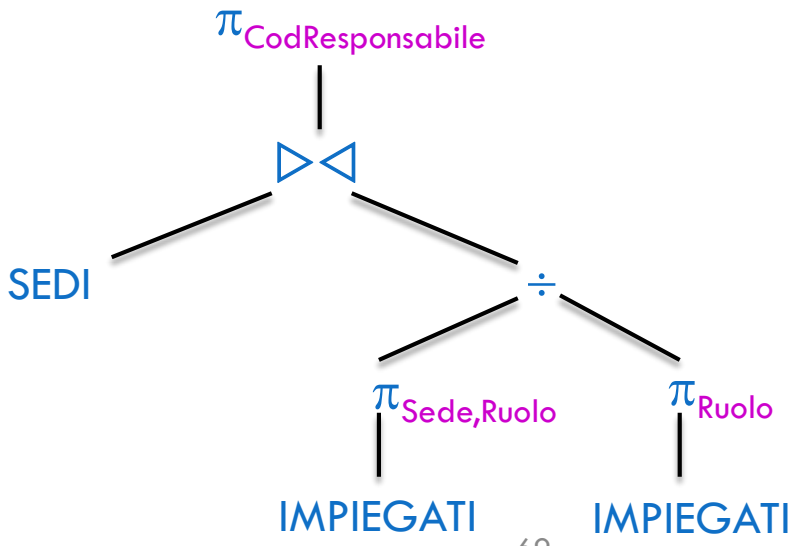
$\pi_{CodProg,\ Città}(PROGETTI \triangleright\triangleleft (SEDI \triangleright\triangleleft IMPIEGATI\_TOP))$

CodProg	Città
P01	Milano
P01	Bologna
P02	Bologna

Codici dei responsabili delle sedi dove sono presenti tutti i ruoli:

$\pi_{CodResponsabile}(SEDI \triangleright\triangleleft (\pi_{Sede,Ruolo}(IMPIEGATI) \div \pi_{Ruolo}(IMPIEGATI)))$

CodResponsabile
EOO7



Esercizio:
si modifichi l'espressione per restituire anche il nome e il cognome dei responsabili.

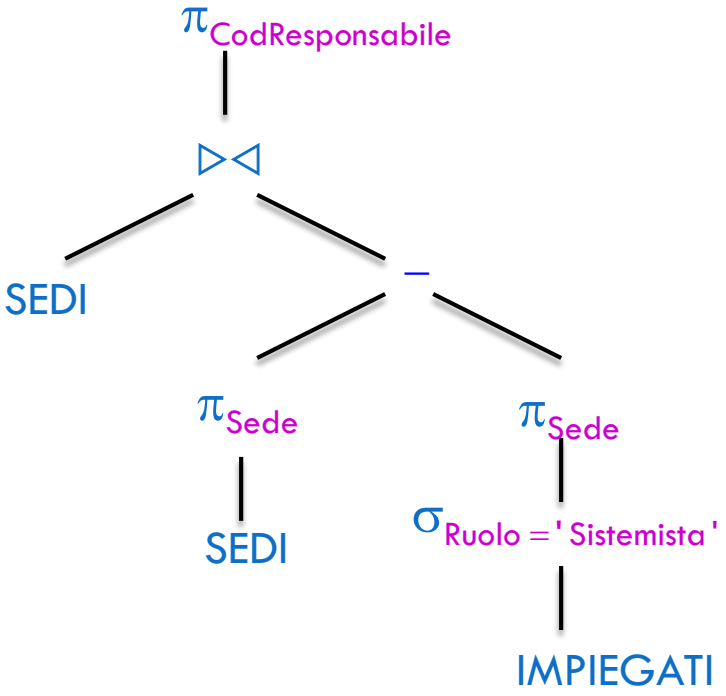
Espressioni: esempio (5)

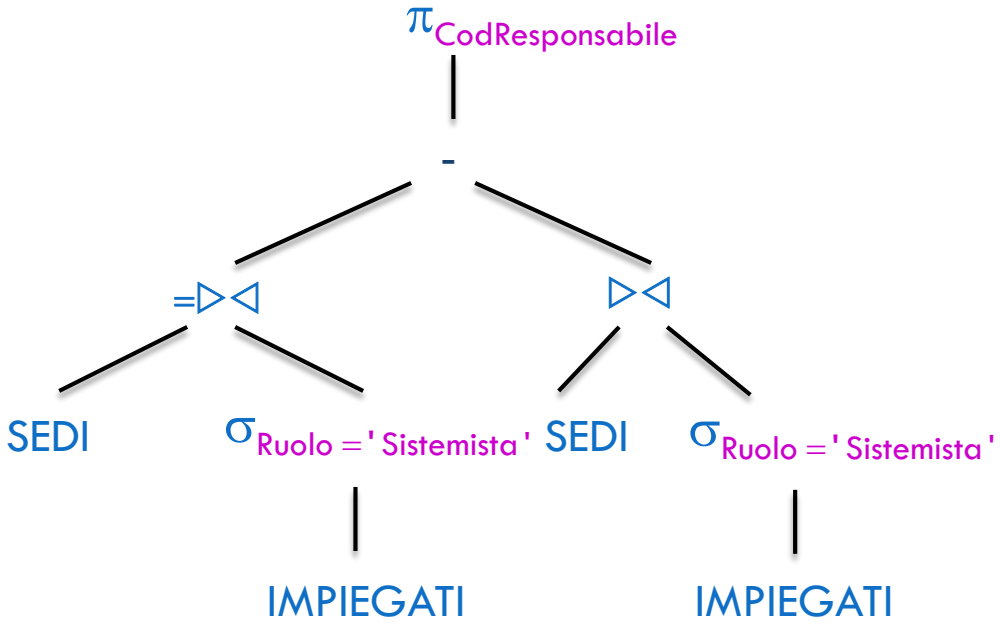
Codici dei responsabili delle sedi che non hanno sistemisti:

CodResponsabile
E009

$$\pi_{CodResponsabile}(SEDI \rhd\lhd (\pi_{Sede}(SEDI) - \pi_{Sede}(\sigma_{Ruolo='Sistemista'}(IMPIEGATI))))$$

oppure: $\pi_{CodResponsabile}((SEDI =\rhd\lhd (\sigma_{Ruolo='Sistemista'}(IMPIEGATI))) - (SEDI \rhd\lhd (\sigma_{Ruolo='Sistemista'}(IMPIEGATI))))$





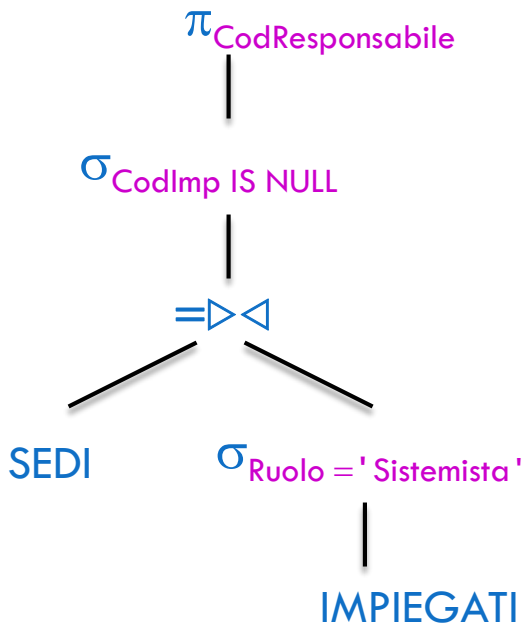
Espressioni: esempio (5 bis)

Un altro modo per ottenere il risultato:

CodResponsabile
E009

$$\pi_{\text{CodResponsabile}}(\sigma_{\text{CodImpiegato IS NULL}}(\text{SEDI} =\triangleright\triangleleft (\sigma_{\text{Ruolo = 'Sistemista'}}(\text{IMPIEGATI}))))$$

Ragionamento: si effettua un left outer join e poi un test sul valore dell'attributo CodImpiegati di IMPIEGATI; se la tupla di SEDI non è dangling, quel valore è sicuramente non nullo.



Equivalenza di espressioni

- Un'interrogazione su un database con schema DB può a tutti gli effetti essere vista come una funzione che a ogni stato db del database associa una relazione risultato con un dato schema.
- Un'espressione E dell'AR costituisce quindi una modalità specifica per esprimere tale funzione; E(db) denota il risultato dell'applicazione di E allo stato db. Due espressioni sono tra loro equivalenti se rappresentano la stessa funzione:

  due espressioni E_1 ed E_2 espresse su un database con schema DB si dicono equivalenti rispetto a DB ($E_1 \equiv_{DB} E_2$) se e solo se per ogni stato db producono lo stesso risultato, $E_1(db) = E_2(db)$.

- Si noti che quando E è un'espressione composta, ad esempio se $E = E_a \bowtie E_b$ allora $E(db) = E_a(db) \bowtie E_b(db)$; il caso di base riguarda uno stato r di una relazione R nell'estensione db del data base con schema DB.

Equivalenza di espressioni

- In alcuni casi l'equivalenza non dipende dallo schema DB specifico, nel qual caso si scrive $E_1 \equiv E_2$ (ossia $E_1 \equiv_{DB} E_2$ è valida per ogni schema DB).

  Esempio: per ogni DB si ha:

  $\pi_{AB}(\sigma_{A=a}(R)) \equiv \sigma_{A=a}(\pi_{AB}(R))$, come è facile verificare;
  a è un generico valore di dom(A).

- D'altra parte l'equivalenza

  $\pi_{AB}(R_1) \bowtie \pi_{BC}(R_2) \equiv_{DB} \pi_{ABC}(R_1 \bowtie R_2)$

  è garantita solo se anche nel secondo caso il join è solo su B, come avviene nell'espressione a sinistra.

Equivalenze: considerazioni

- Due espressioni equivalenti E_1 ed E_2 garantiscono lo stesso risultato, ma ciò non significa che la scelta sia indifferente in termini di "risorse" necessarie.
- Considerazioni di questo tipo sono essenziali per un RDBMS durante la fase di ottimizzazione delle interrogazioni.
- 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 sono presenti valori nulli).

Regole di equivalenza

- Tra le regole base di equivalenza, si ricordano quelle appresso elencate.
- Il join naturale è commutativo e associativo:

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

- Selezione e proiezione si possono raggruppare:

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

- Selezione e proiezione commutano (se F si riferisce esclusivamente ad attributi in Y):

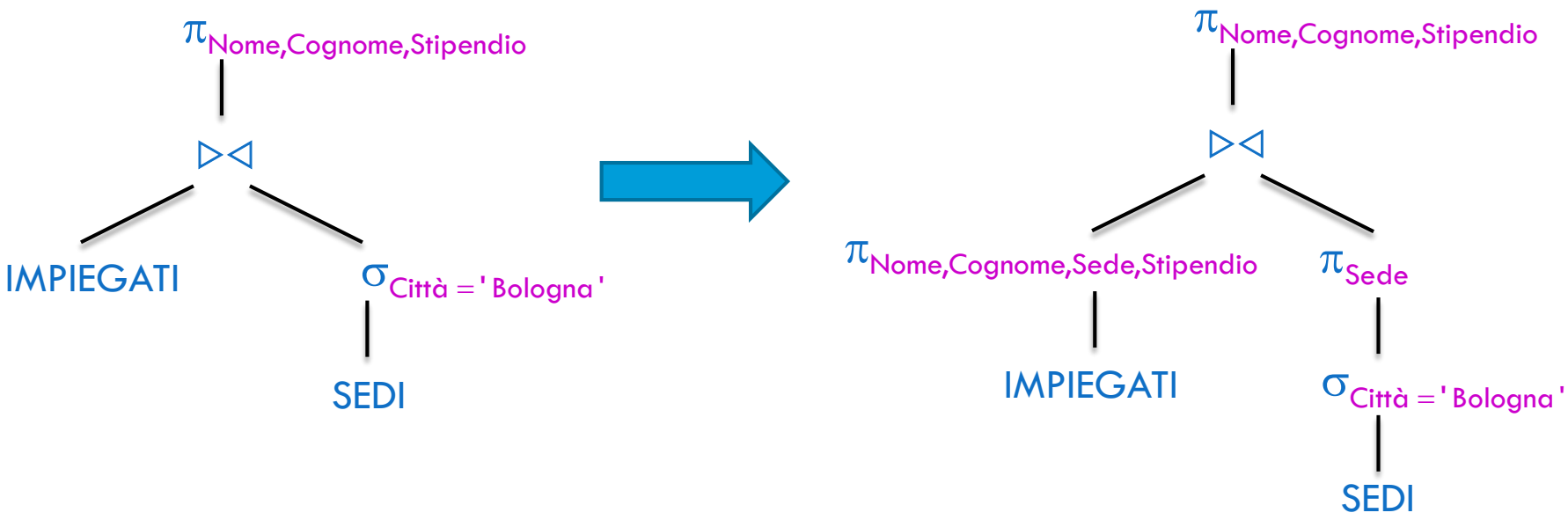
$$\pi_Y(\sigma_F(E)) \equiv \sigma_F(\pi_Y(E))$$

- "Push-down"della selezione rispetto al join (se F è sullo schema di E_1):

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

Push-down delle proiezioni

- Usualmente un RDBMS cerca di eliminare quanto prima gli attributi che non servono per produrre il risultato di una query.
- Un attributo A è utile se è richiesto in output o è necessario per un operatore che non è stato ancora eseguito.
- Esempio: nome, cognome e stipendio degli impiegati che lavorano nelle sedi di Bologna:



Strumenti per AR

- RelaX è un servizio online che permette di eseguire interrogazioni in algebra relazionale con una sintassi simile a quella utilizzata nel corso.
- RelaX, sviluppato l'Università di Innsbruck, consente di scrivere ed eseguire espressioni di algebra relazionale; mette anche a disposizione alcuni DB di prova. Altri DB sono reperibili presso i siti web di alcuni corsi universitari e, nell'ambito di questo corso, sono disponibili esempi nel materiale didattico fornito per le esercitazioni sulla piattaforma virtuale Unibo.
- Relax offre anche la possibilità di scrivere alcuni tipi di query in SQL e mostrare l'albero dell'equivalente espressione in algebra relazionale.
- Esistono anche software, a scopo didattico, per convertire espressioni di algebra relazionale in SQL; un esempio di free software è RAT. Un esempio di interprete di espressioni algebriche relazionali è RA.

Un esempio con RelaX

IMPIEGATI
- CodImpiegato string
- Nome string
- Cognome string
- Sede string
- Ruolo string
- Stipendio number

SEDI
- Sede string
- CodResponsabile string
- Citta string

PROGETTI
- CodProgetto string
- Sede string

Relational Algebra | SQL | Group Editor

π σ ρ ← → τ γ ∧ ∨ ¬ = ≠ ≥ ≤ ∩ ∪ ÷ - × ⋈ ⋈ ⋈ ⋈ ⋉ ⋊ ▷ = -- /* {}

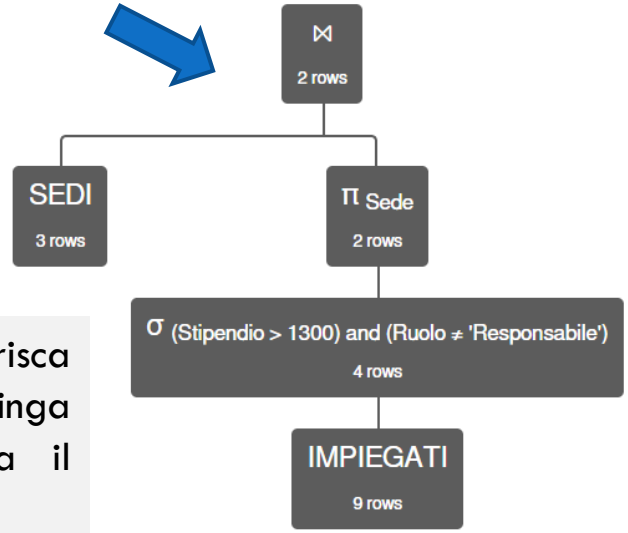
```
1 /*Sede, codice del responsabile e città per ogni sede dove vi sono impiegati, non
  responsabili, che guadagnano più di 1300 €
2 */
3 SEDI ⋈ (π Sede (σ (Stipendio>1300) ∧ (Ruolo≠'Responsabile') (IMPIEGATI) ))
```

espressione

Risultato

SEDI.Sede	SEDI.CodResponsabile	SEDI.Citta
'S01'	'E007'	'Milano'
'S02'	'E008'	'Bologna'

Albero d'esecuzione della query



Si acceda a RelaX; per caricare il DB di prova, indicato in figura, s'inserisca nel campo "Load dataset stored in a gist" la stringa 021f3f0fdac45f4d3cea85dfe7070d71 e successivamente si prema il bottone "Load" .

Domande?