

# Algebra e Calcolo Relazionale (II)

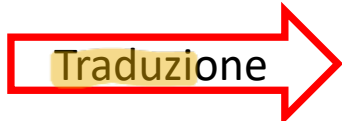
---

PROF. DIOMAIUTA CRESCENZO

UNIVERSITÀ DEGLI STUDI DELLA CAMPANIA «*LUIGI VANVITELLI*»

# Equivalenza di espressioni algebriche

- Due espressioni sono **equivalenti** se producono lo stesso risultato qualunque sia l'istanza attuale della base di dati
- L'equivalenza è importante in pratica perché i DBMS cercano di eseguire espressioni equivalenti a quelle date, ma **meno "costose"**

LINGUAGGIO SQL  Algebra relazionale

- Vengono quindi utilizzati spesso **trasformazioni di equivalenza**

# Trasformazioni

---

- Atomizzazione delle selezioni: una congiunzione di selezioni può essere sostituita da una cascata di selezioni atomiche
  - $\sigma_{F1 \wedge F2}(E) \equiv \sigma_{F1}(\sigma_{F2}(E))$
- Idempotenza della proiezione: una proiezione può essere trasformata in una cascata di proiezioni che eliminano i vari attributi in varie fasi
  - $\pi_X(E) \equiv \pi_X(\pi_{XY}(E))$

**SONO OPERAZIONI PRELIMINARI AD ALTRE**

# Anticipazione selezione rispetto al join

➤ Pushing selections down (se A è attributo di R2 )

➤  $\sigma_F (E1 \bowtie E2) = E1 \bowtie \sigma_F (E2)$

➤ ES:  $\sigma_{A=10} (R1 \bowtie R2) = R1 \bowtie \sigma_{A=10} (R2)$

SOLO SE ➤ se la condizione F coinvolge solo attributi della sottoespressione E2

➤ Riduce in modo significativo la dimensione del risultato intermedio (e quindi il costo dell'operazione)

# Nota

---

- In questo corso, ci preoccupiamo poco dell'efficienza: l'obiettivo è di scrivere interrogazioni corrette e leggibili
- Motivazione: i DBMS si preoccupano di scegliere le strategie realizzative efficienti

# Anticipazione proiezione rispetto al join

## ➤ Pushing projections down

➤  $\pi_{x_1 y_2} (E_1 \bowtie E_2) \equiv E_1 \bowtie \pi_{x_1 y_2} (E_2)$

SOLO SE  $x_1 y_2$  SONO ATTRIBUTI DI  $E_2$ .

# Trasformazioni

## ➤ Inglobamento di una selezione in un prodotto cartesiano a formare un join

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

➤ ES: Supervisione  $\bowtie_{\text{Impiegato=Matricola}}$  Impiegati

Cioè fare il theta join, che in caso di uguaglianza diventa equi-join

NOTA: Posso anche fare join di una tabella per sé stessa a volte

# Trasformazioni: Esempio

➤ Supponiamo di voler trovare i numeri di matricola dei capi di impiegati con meno di trenta anni

IMPIEGATI

Matr	Nome	Età	Stipendio
101	Mario Rossi	34	40
103	Mario Bianchi	23	35
104	Luigi Neri	38	61
105	Nico Bini	44	38
210	Marco Celli	49	60
231	Siro Bisi	50	60
252	Nico Bini	44	70
301	Sergio Rossi	34	70
375	Mario Rossi	50	65

$\pi_{\text{Capo}} (\sigma_{\text{Matr}=\text{Imp} \wedge \text{Eta}<30} (\text{Impiegati} \bowtie \text{Supervisione}))$

Capo	Impiegato
210	101
210	103
210	104
231	105
301	210
301	231
375	252

**POCO EFFICIENTE!!**

↓ Può essere trasformata con le proprietà di prima

SUPERVISIONE



# Trasformazioni: Esempio

➤ Supponiamo di voler trovare i numeri di matricola dei capi di impiegati con meno di trenta anni

➤ **Atomizzazione selezioni**

$\pi_{\text{Capo}} (\sigma_{\text{Matr}=\text{Imp}} (\sigma_{\text{Eta}<30} (\text{Impiegati} \bowtie \text{Supervisione})))$

Spezziamo la selezione

➤ **Inglobamento selezione e anticipazione della selezione rispetto al join**

$\pi_{\text{Capo}} (\sigma_{\text{Eta}<30} (\text{Impiegati}) \bowtie_{\text{Matr}=\text{Imp}} \text{Supervisione})$

Fondiamo selezione con prodotto cartesiano e anticipiamo la selezione

➤ **Anticipazione proiezione rispetto al join**

$\pi_{\text{Capo}} (\pi_{\text{Matr}} (\sigma_{\text{Eta}<30} (\text{Impiegati})) \bowtie_{\text{Matr}=\text{Imp}} \text{Supervisione})$

Eliminiamo dal primo argomento gli attributi non necessari

# Trasformazioni

*Selezione dell'unione = unione delle selezioni*

## ➤ Distributività della selezione rispetto all'unione

➤  $\sigma_F(E_1 \cup E_2) \equiv \sigma_F(E_1) \cup \sigma_F(E_2)$

## ➤ Distributività della selezione rispetto alla differenza

➤  $\sigma_F(E_1 - E_2) \equiv \sigma_F(E_1) - \sigma_F(E_2)$

# Trasformazioni

## ➤ Distributività della proiezione rispetto all'unione

➤  $\pi_X(E_1 \cup E_2) \equiv \pi_X(E_1) \cup \pi_X(E_2)$  *Come le partizioni*

➤  $\sigma_{F_1 \vee F_2}(E) = \sigma_{F_1}(E) \cup \sigma_{F_2}(E)$

➤  $\sigma_{F_1 \wedge F_2}(E) = \sigma_{F_1}(E) \cap \sigma_{F_2}(E) = \sigma_{F_1}(E) \bowtie \sigma_{F_2}(E)$

➤  $\sigma_{F_1 \wedge \neg F_2}(E) = \sigma_{F_1}(E) - \sigma_{F_2}(E)$

# Algebra con valori nulli

---

- Estendere la semantica degli operatori per la presenza di valori nulli
- Bisogna estendere il join naturale per non scartare le tuple dangling, ma generare valori nulli
- Esistono diversi approcci al trattamento dei valori nulli, nessuno dei quali è completamente soddisfacente (per ragioni formali e/o pragmatiche)
- L'approccio che vedremo è quello "tradizionale", che ha il pregio di essere molto simile a quello adottato in SQL (e quindi dai DBMS relazionali)

# Proiezione, Unione e Differenza con 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
1	Rossi	A1
2	Verdi	NULL
3	Verdi	A2
4	Verdi	NULL

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

Nome	Ufficio
Rossi	A1
Verdi	NULL
Verdi	A2

Responsabili

Cod	Nome	Ufficio
1	Rossi	A1
NULL	NULL	A2
4	Verdi	NULL

$\text{Impiegati} \cup \text{Responsabili}$

Cod	Nome	Ufficio
1	Rossi	A1
2	Verdi	NULL
3	Verdi	A2
4	Verdi	NULL
NULL	NULL	A2

# Selezione 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

Matricola	Cognome	Filiale	Età
7309	Rossi	Roma	32
5998	Neri	Milano	45
9553	Bruni	Milano	NULL

la condizione atomica è vera solo per valori **non nulli**

$\sigma_{\text{Età} > 40}(\text{Impiegati})$

- La seconda tupla fa parte del risultato e la prima no, la terza? Non si può decidere

# Un risultato non desiderabile

---

- Le selezioni vengono valutate separatamente!

$$\sigma_{\text{Età} > 30}(\text{Impiegati}) \cup \sigma_{\text{Età} \leq 30}(\text{Impiegati}) \neq \text{Impiegati}$$

- Anche le condizioni atomiche vengono valutate separatamente!

$$\sigma_{\text{Età} > 30 \vee \text{Età} \leq 30}(\text{Impiegati}) \neq \text{Impiegati}$$

# Logica a tre valori

- Si potrebbe usare (ma non serve) una "logica a tre valori" (vero, falso, 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

**IS NULL**

**IS NOT NULL**

*Quanto dato mettere in OR e in IS null per quelle da cui se voglio prendere*



# Clausola IS

---

$$\begin{aligned} & \sigma_{\text{Età} > 30}(\text{Impiegati}) \cup \sigma_{\text{Età} \leq 30}(\text{Impiegati}) \cup \sigma_{\text{Età IS NULL}}(\text{Impiegati}) \\ &= \\ & \sigma_{\text{Età} > 30 \vee \text{Età} \leq 30 \vee \text{Età IS NULL}}(\text{Impiegati}) \\ &= \\ & \text{IMPIEGATI} \end{aligned}$$

# Selezione con valori nulli

---

Impiegati

Matricola	Cognome	Filiale	Età
5998	Neri	Milano	45
9553	Bruni	Milano	NULL

$\sigma_{(Età > 40) \text{ OR } (Età \text{ IS NULL})}(\text{Impiegati})$

*Why important; we find employee if we consider null.*

# Selezione con valori nulli: esempi

Impiegati

Cod	Nome	Ufficio
1	Rossi	A1
2	Verdi	NULL
3	Verdi	A2
4	NULL	A2

$\sigma_{\text{Ufficio} = A1}(\text{Impiegati})$

Cod	Nome	Ufficio
1	Rossi	A1

$\sigma_{(\text{Ufficio} = A1) \text{ OR } (\text{Ufficio} \neq A1)}(\text{Impiegati})$

Cod	Nome	Ufficio
1	Rossi	A1
3	Verdi	A2
4	NULL	A2

$\sigma_{(\text{Ufficio} = A2) \text{ AND } (\text{Nome} = \text{Verdi})}(\text{Impiegati})$

Cod	Nome	Ufficio
3	Verdi	A2

$\sigma_{(\text{Ufficio} = A2) \text{ OR } (\text{Nome} = \text{Verdi})}(\text{Impiegati})$

Cod	Nome	Ufficio
2	Verdi	NULL
3	Verdi	A2
4	NULL	A2

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

Cod	Nome	Ufficio
2	Verdi	NULL

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

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

# Join 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
1	Rossi	A1
2	Verdi	NULL
3	Verdi	A2
4	Verdi	NULL

Responsabili

Ufficio	Cod
A1	1
A2	NULL
NULL	2

Impiegati ⋈ Responsabili

Cod	Nome	Ufficio
1	Rossi	A1
3	Verdi	A2

# Viste

→ Le viste rischiano di appesantire la base dati: se una delle relazioni sono memorizzate, causa ridondanza  
Se ho vista virtuale, la tabella generata vale solo a runtime

- Rappresentazioni diverse per gli stessi dati (**schema esterno**)
- **Relazioni derivate:**
  - relazioni il cui contenuto è funzione del contenuto di altre relazioni (definito per mezzo di interrogazioni)
- **Relazioni di base:** contenuto autonomo
- Le relazioni derivate il cui contenuto è funzione di quello di altre relazioni ... ma deve esistere un ordinamento fra le relazioni derivate

# Viste virtuali e materializzate

---

- Due tipi di relazioni derivate:
  - Viste materializzate
  - Relazioni virtuali (o viste)

# Viste materializzate

---

- Relazioni derivate memorizzate nella base di dati
  - Vantaggi:
    - Immediatamente disponibili per le interrogazioni
  - Svantaggi:
    - Ridondanti
    - Appesantiscono gli aggiornamenti
    - Sono raramente supportate dai DBMS

# Viste virtuali

---

- relazioni virtuali (o viste): relazioni definite per mezzo di funzioni non memorizzate nella base dati, ma utilizzabili come se lo fossero
- sono supportate dai DBMS (tutti)
- una interrogazione su una vista viene eseguita "ricalcolando" la vista (o quasi)



# Viste: Esempio

- una vista: Se siamo interessati solo agli impiegati coi relativi capi
- **Supervisione** =  $\pi_{\text{Impiegato, Capo}}$  (Afferenza  $\triangleright \triangleleft$  Direzione)

**Afferenza**

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B
Bianchi	B

**Direzione**

Reparto	Capo
A	Mori
B	Bruni
B	Bruni

# Interrogazioni sulle viste

---

- Sono eseguite sostituendo alla vista la sua definizione:

$\sigma_{\text{Capo}='Leoni'}$  (Supervisione)

- Viene eseguita come

$\sigma_{\text{Capo}='Leoni'}(\pi_{\text{Impiegato, Capo}}(\text{Afferenza} \triangleright \triangleleft \text{Direzione}))$

# Viste: motivazioni

---

- Schema esterno: ogni utente vede solo
  - ciò che gli interessa e nel modo in cui gli interessa, senza essere distratto dal resto
  - ciò che è autorizzato a vedere (autorizzazioni)
- Strumento di programmazione:
  - si può semplificare la scrittura di interrogazioni: espressioni complesse e sotto espressioni ripetute
- Utilizzo di programmi esistenti su schemi ristrutturati
- Invece:
  - L'utilizzo di viste non influisce sull'efficienza delle interrogazioni

# Viste come strumento di programmazione

□ Trovare gli impiegati che hanno lo stesso capo di Rossi

□ Senza vista:

□  $\pi_{\text{Impiegato}} (\text{Afferenza} \bowtie \text{Direzione}) \bowtie \rho_{\text{ImpR,RepR} \leftarrow \text{Imp,Reparto}} (\sigma_{\text{Impiegato}='Rossi'} (\text{Afferenza} \bowtie \text{Direzione}))$

□ Con la vista:

□  $\pi_{\text{Impiegato}} (\text{Supervisione}) \bowtie \rho_{\text{ImpR,RepR} \leftarrow \text{Imp,Reparto}} (\sigma_{\text{Impiegato}='Rossi'} (\text{Supervisione}))$

# Viste: Aggiornamenti

- Vogliamo inserire, nella vista, il fatto che Lupi ha come capo Bruni; oppure che Belli ha come capo Falchi; come facciamo?

## Afferenza

Impiegato	Reparto
Rossi	A
Neri	B
Verdi	A

## Direzione

Reparto	Capo
A	Mori
B	Bruni
C	Bruni

## Supervisione

Impiegato	Capo
Rossi	Mori
Neri	Bruni
Verdi	Mori

# Viste: Aggiornamenti

- ❑ "Aggiornare una vista":
  - ❑ modificare le relazioni di base in modo che la vista, "ricalcolata" rispecchi l'aggiornamento
- ❑ L'aggiornamento sulle relazioni di base corrispondente a quello specificato sulla vista deve essere univoco
- ❑ In generale però non è univoco!

*Aggiornamento non è più univoco*

## Supervisione

Impiegato	Capo
Rossi	Mori
Neri	Bruni
Verdi	Mori

L'inserimento di una tupla nella vista non corrisponde univocamente a un insieme di aggiornamenti sulle relazioni di base, poiché non c'è un valore per l'attributo

Reparto

- ❑ Ben pochi aggiornamenti sono ammissibili sulle viste

# Una convenzione e notazione alternativa per i join

---

- ❑ Nota: è sostanzialmente l'approccio usato in SQL
- ❑ Ignoriamo il join naturale (cioè non consideriamo implicitamente condizioni su attributi con nomi uguali)
- ❑ Per "riconoscere" attributi con lo stesso nome gli premettiamo il nome della relazione
- ❑ Usiamo "assegnazioni" (viste) per ridenominare le relazioni (e gli attributi solo quando serve per l'unione)

# Convenzione e notazioni: Esempio (1)

- Trovare gli impiegati che guadagnano più del proprio capo, mostrando matricola, nome e stipendio dell'impiegato e del capo

$\pi_{\text{Matr, Nome, Stip, MatrC, NomeC, StipC}}$   
 $(\sigma_{\text{Stip} > \text{StipC}} ( \rho_{\text{MatrC, NomeC, StipC, EtàC} \leftarrow \text{Matr, Nome, Stip, Età} (\text{Impiegati})$   
 $\quad \triangleright \triangleleft_{\text{MatrC} = \text{Capo}}$   
 $\quad (\text{Supervisione } \triangleright \triangleleft_{\text{Impiegato} = \text{Matricola}} \text{ Impiegati})))$



# Convenzione e notazioni: Esempio (2)

□ Assegniamo la relazione Impiegato a Capo (Vista)

**Capi := Imp**

$\pi_{\text{Imp.Matr}, \text{Imp.Nome}, \text{Imp.Stip}, \text{Capi.Matr}, \text{Capi.Nome}, \text{Capi.Stip}} (\sigma_{\text{Imp.Stip} > \text{Capi.Stip}} (\mathbf{Capi} \bowtie \text{Capi.Matr} = \text{Capo} (\mathbf{Sup} \bowtie \text{Imp} = \text{Imp.Matr} \mathbf{Imp})))$

→ Per risolvere problema in cui ho una tabella strutturata in un certo modo in cui certe informazioni devono essere ritate in maniera ricorsiva.

# Chiusura transitiva

Supervisione (Impiegato, Capo)

□ Per ogni impiegato, trovare tutti i superiori (cioè il capo, il capo del capo, e così via...)

Impiegato	Capo
Rossi	Lupi
Neri	Bruni
Lupi	Falchi

Impiegato	Superiore
Rossi	Lupi
Neri	Bruni
Lupi	Falchi
Rossi	Falchi

# Chiusura transitiva: come fare?

➤ Nell'esempio, basterebbe il join della relazione con se stessa, previa opportuna ridenominazione

➤ Ma:

Impiegato	Capo
Rossi	Lupi
Neri	Bruni
Lupi	Falchi
Falchi	Leoni

Impiegato	Superiore
Rossi	Lupi
Neri	Bruni
Lupi	Falchi
Rossi	Falchi
Lupi	Leoni
Rossi	Leoni

# Chiusura transitiva: come fare?

---

- Non esiste in algebra relazionale la possibilità di esprimere l'interrogazione che, per ogni relazione binaria, ne calcoli la chiusura transitiva
- Per ciascuna relazione, è possibile calcolare la chiusura transitiva, ma con un'espressione ogni volta diversa:
- quanti join servono? non c'è limite!

---

# ESERCIZI

# Esercizio 1

---

Considerare lo schema di base di dati contenente le relazioni:

**Film**(CodiceFilm, Titolo, Regista, Anno, CostoNoleggior)  
**Artisti**(CodiceAttore, Cognome, Nome, Sesso, DataNascita, Nazionalità)  
**Interpretazioni**(CodiceFilm, CodiceAttore, Personaggio)

formulare in algebra relazionale le seguenti interrogazioni:

1. I titoli dei film nei quali Al Pacino sia stato interprete;
2. I titoli dei film per i quali il regista sia stato anche interprete;
3. I titoli dei film in cui gli attori noti siano tutti dello stesso sesso.

# Esercizio 1

---

**I titoli dei film nei quali Al Pacino sia stato interprete**

$\Pi_{\text{Titolo}}( \text{FILM} \bowtie (\sigma (\text{Nome}=\text{'Al'}) \wedge (\text{Cognome}=\text{'Pacino'}) (\text{ARTISTI}) \bowtie \text{INTERPRETAZIONI} ) )$

# Esercizio 1

---

**I titoli dei film per i quali il regista sia stato anche interprete**

**$\Pi_{\text{Titolo}}( \sigma_{(\text{Regista}=\text{CodiceAttore})} (\text{INTERPRETAZIONI} \bowtie \text{FILM}))$**



# Esercizio 1

---

**I titoli dei film in cui gli attori noti siano tutti dello stesso sesso**

$\Pi_{\text{Titolo}}(\text{FILM}) - \Pi_{\text{Titolo}}(\text{FILM}) \sigma_{\text{Sex} \leftrightarrow \text{Sex1}}((\text{ARTISTI} \bowtie \text{INTERPRETAZIONI}) \bowtie \rho_{\text{Sex1} \leftarrow \text{Sex}}(\Pi_{\text{CodiceFilm, Sex}}(\text{ARTISTI} \bowtie \text{INTERPRETAZIONI})))$

# Esercizio 2

---

Considerare lo schema di base di dati contenente le relazioni:

**MATERIE** (Codice, Facoltà, Denominazione, Professore)

**STUDENTI** (Matricola, Cognome, Nome, Facoltà)

**PROFESSORI** (Matricola, Cognome, Nome)

**ESAMI** (Studente, Materia, Voto, Data)

**PIANIDISTUDIO** (Studente, Materia, Anno)

formulare in algebra relazionale le seguenti interrogazioni:

1. Gli studenti che hanno riportato in almeno un esame una votazione pari a 30, mostrando, per ciascuno di essi, nome e cognome e data della prima di tali occasioni;
2. Gli studenti che hanno superato tutti gli esami previsti dal rispettivo piano di studio;
3. Nome e cognome degli studenti che hanno sostenuto almeno un esame con un professore che ha il loro stesso nome proprio.

## Esercizio 2

**Gli studenti che hanno riportato in almeno un esame una votazione pari a 30, mostrando, per ciascuno di essi, nome e cognome e data della prima di tali occasioni**

$$\begin{aligned} & \Pi_{\text{Nome, Cognome, Data}} \\ & \left( \left( \Pi_{\text{Studente, Data}} (\sigma_{\text{Voto} = '30'}(\text{ESAMI})) - \right. \right. \\ & \left. \Pi_{\text{Studente, Data}} ((\sigma_{\text{Voto} = '30'}(\text{ESAMI}) \bowtie (\text{Studente} = \text{Studente1}) \wedge (\text{Data} > \text{Data1})) \right. \\ & \left. \left. \rho_{\text{Studente1, Corso1, Voto1, Data1} \leftarrow \text{Studente, Corso, Voto, Data}} \sigma_{\text{Voto} = '30'}(\text{ESAMI})) \right) \right. \\ & \left. \bowtie_{\text{Studente} = \text{Matricola}} (\text{STUDENTE}) \right) \end{aligned}$$

# Esercizio 2

---

**Gli studenti che hanno superato tutti gli esami previsti dal  
rispettivo piano di studio**

**$\Pi_{\text{Studente}}(\text{PIANIDISTUDIO}) -$   
 $(\Pi_{\text{Studente}}(\Pi_{\text{Studente,Materia}}(\text{PIANIDISTUDIO}) - \Pi_{\text{Studente, Materia}}(\text{ESAMI})))$**

# Esercizio 2

---

**Nome e cognome degli studenti che hanno sostenuto almeno un esame con un professore che ha il loro stesso nome proprio**

$\Pi_{\text{Nome,Cognome}} (\sigma_{\text{Nome}=\text{NomeP}} ((\text{ESAMI} \bowtie_{\text{Studente}=\text{Matricola}} \text{STUDENTI}) \bowtie_{\text{Materia}=\text{Codice}} (\rho_{\text{NomeP,CognomeP} \leftarrow \text{Nome,Cognome}} (\text{PROFESSORI} \bowtie_{\text{Matricola}=\text{Professore}} \text{MATERIE}))))$