Algebra e Calcolo Relazionale

Linguaggi per le basi di dati

- Operazioni sullo **schema**
  - **Data Definition Language** (DDL)
- Operazioni sui **dati**
  - **Data Manipulation Language** (DML)
    - Interrogazione (*query*)
    - Aggiornamento (*update*)

Linguaggi di interrogazione per le basi di dati

- Cosa si intende per **interrogazione**?
  - Operazione di **lettura** sulla base di dati che può richiedere l'accesso a **più** di una **tabella**
- Cosa è necessario fare per specificare il **significato** di una interrogazione?
- Due formalismi:
  - **Modo dichiarativo**: si specificano le proprietà del risultato ("che cosa")
  - **Modo procedurale**: si specificano le modalità di generazione del risultato ("come")

Linguaggi di interrogazione per le basi di dati

- Si definisce il **comportamento** delle interrogazioni in **modo procedurale** utilizzando le espressioni dell'**algebra relazionale**
- Si definisce qual è il **risultato** di un'interrogazione in **modo dichiarativo** utilizzando le espressioni del **calcolo relazionale**
- Il calcolo relazionale è l'effettiva **semantica** del linguaggio
  - Le interrogazioni sono espresse ad **alto livello**
    - **Nessun** concetto di **costo**
- Con l'algebra relazionale si definisce il **modo** in cui il DBMS **esegue** un'interrogazione

Algebra Relazionale

- **Algebra = dati + operatori**
- **Algebra relazionale**:
  - Dati: relazioni
  - Operatori:
    - su relazioni,
    - che producono relazioni,
    - e che possono essere composti

Operatori dell'Algebra Relazionale

- Operatori su insiemi:
  - unione, intersezione, differenza
- Operatori su relazioni:
  - ridenominazione
  - selezione
  - proiezione
  - join:
    - naturale, prodotto cartesiano, theta

Notazione

- $R, R_1, R_2, \ldots$ indicano nomi di relazione
- $A, B, C, A_1, A_2, \ldots$ indicano nomi di attributo
- $X, Y, X_1, X_2, \ldots$ indicano insiemi di attributi
- XY è un'abbreviazione di $X \cup Y$

- Una relazione con n-uple $t_1, t_2, \ldots, t_n$ è indicata con l'insieme $\{t_1, t_2, \ldots, t_n\}$
- $t_i[A_j]$ indica il valore dell'attributo A_j nella n-upla t_i
- $t[X]$ indica l'n-upla ottenuta da t considerando solo gli attributi in X

Operatori su Insiemi

- Le **relazioni** sono **insiemi**
- I **risultati** devono essere **relazioni**
- Si possono applicare gli operatori su insiemi solo a **relazioni definite sugli stessi attributi**
  - In modo che il risultato sia una relazione sugli stessi attributi

Unione

- L'unione di due relazioni sullo stesso insieme di attributi X è una relazione su X che contiene le n-uple sia dell'una che dell'altra relazione originarie

Laureati

Matricola	Nome	Età
7274	Rossi	32
7432	Neri	24
9824	Verdi	25

Specialisti

Matricola	Nome	Età
9297	Neri	33
7432	Neri	24
9824	Verdi	25

Laureati ∪ Specialisti

Matricola	Nome	Età
7274	Rossi	32
7432	Neri	24
9824	Verdi	25
9297	Neri	33

Intersezione

- L'intersezione di due relazioni sullo stesso insieme di attributi X è una relazione su X che contiene le n-uple appartenenti a entrambe le relazioni originarie

Laureati

Matricola	Nome	Età
7274	Rossi	32
7432	Neri	24
9824	Verdi	25

Specialisti

Matricola	Nome	Età
9297	Neri	33
7432	Neri	24
9824	Verdi	25

Laureati ∩ Specialisti

Matricola	Nome	Età
7432	Neri	24
9824	Verdi	25

Differenza

- La differenza tra due relazioni sullo stesso insieme di attributi X è una relazione su X che contiene le n-uple appartenenti alla prima relazione che non appartengono anche alla seconda

Laureati

Matricola	Nome	Età
7274	Rossi	32
7432	Neri	24
9824	Verdi	25

Specialisti

Matricola	Nome	Età
9297	Neri	33
7432	Neri	24
9824	Verdi	25

Laureati − Specialisti

Matricola	Nome	Età
7274	Rossi	32

Unione Impossibile?

- Sebbene abbia senso, come effettuare l'unione delle due relazioni seguenti?

Paternità

Padre	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

Maternità

Madre	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

Paternità $\cup$ **Maternità**

???

Ridenominazione

- Operatore con **un solo operando** ("monadico")
- **Modifica lo schema** dell'operando, **lasciandone inalterata l'istanza**
- Data una relazione R, in generale, questo operatore si scrive come:

$$\rho_{B_1 B_2 \ldots \leftarrow A_1 A_2 \ldots}(R)$$

- da leggersi:
  - L'attributo A_1 viene sostituito dall'attributo B_1
  - L'attributo A_2 viene sostituito dall'attributo B_2
  - ...

Paternità

Padre	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

Maternità

Madre	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

$\rho_{\textbf{Genitore} \leftarrow \textbf{Padre}}(\textbf{Paternità})$

$\rho_{\textbf{Genitore} \leftarrow \textbf{Madre}}(\textbf{Maternità})$

Paternità

Genitore	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco

Maternità

Genitore	Figlio
Eva	Abele
Eva	Set
Sara	Isacco

$$\rho_{\textbf{Genitore}\leftarrow\textbf{Padre}}(\textbf{Paternità}) \cup \rho_{\textbf{Genitore}\leftarrow\textbf{Madre}}(\textbf{Maternità})$$

Genitore	Figlio
Adamo	Abele
Adamo	Caino
Abramo	Isacco
Eva	Abele
Eva	Set
Sara	Isacco

Selezione

- Operatore con **un solo operando** ("monadico")
- Produce un risultato che:
  - ha lo **stesso schema** dell'operando
  - contiene un **sottoinsieme delle n-uple** dell'operando
  - solo quelle n-uple che soddisfano una **condizione** fissata

Selezione

- Data una relazione $R(X)$, in generale, questo operatore si scrive come:

$$\sigma_F(R)$$

- dove:
  - F è una **espressione Booleana** ottenuta componendo con gli **operatori logici AND**, **OR** e **NOT** delle **condizioni atomiche**
- Una **condizione atomica** ha la forma:
  - $A \star B$, dove A e B sono **attributi** di X con domini **compatibili** e $\star$ è un **operatore di confronto**
  - $A \star k$ dove A è un **attributo** di X, k è una **costante** con dominio **compatibile** con A e $\star$ è un **operatore di confronto**

Esempio

- Impiegati che guadagnano più di 50000 euro

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
9553	Milano	Milano	32
5698	Neri	Napoli	40

$$\sigma_{\textbf{Stipendio}>50}(\textbf{Impiegati})$$

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64

Esempio

- Impiegati che guadagnano più di 50000 euro e lavorano a Milano

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
9553	Milano	Milano	32
5698	Neri	Napoli	40

$$\sigma_{\textbf{Stipendio} >50 \textbf{ AND Filiale}=\textbf{'Milano'}}(\textbf{Impiegati})$$

Matricola	Cognome	Filiale	Stipendio
5998	Neri	Milano	64

Selezione e Valori Nulli

- La condizione atomica è vera solo per valori non nulli **in qualsiasi attributo**

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	**NULL**
9553	Milano	Milano	32
5698	Neri	Napoli	40

$$\sigma_{\textbf{Filiale='Milano'}}(\textbf{Impiegati})$$

Matricola	Cognome	Filiale	Stipendio
9553	Milano	Milano	32

Selezione e Valori Nulli

- Per riferirsi a valori nulli esistono condizioni apposite: **IS NULL** e **IS NOT NULL**

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	**NULL**
9553	Milano	Milano	32
5698	Neri	Napoli	40

$$\sigma_{(\textbf{Filiale='Milano'})\ \textbf{OR}\ (\textbf{Stipendio IS NULL})}(\textbf{Impiegati})$$

Matricola	Cognome	Filiale	Stipendio
5998	Neri	Milano	**NULL**
9553	Milano	Milano	32

Proiezione

- Operatore con **un solo operando** (“monadico”)
- Produce un risultato che:
  - ha un **sottoinsieme degli attributi** dell’operando
  - contiene **tutte le n-uple** cui contribuiscono tutti i **valori esistenti** dell’operando

Proiezione

- Data una relazione $R(X)$ e un insieme di attributi $Y \subseteq X$, in generale, questo operatore si scrive come:

$$\pi_Y(R)$$

- Il risultato è una relazione su Y che contiene l'insieme delle n-uple di R ristrette ai soli attributi di Y
- Ricordarsi che il risultato è un insieme
  - Non può contenere n-uple uguali

Esempio

- Calcolare matricola e cognome di tutti gli impiegati

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Neri	Napoli	55
5998	Neri	Milano	64
9553	Rossi	Roma	32
5998	Rossi	Roma	40

$\pi_{\text{Matricola,Cognome}}(\text{Impiegati})$

Matricola	Cognome
7309	Neri
5998	Neri
9553	Rossi
5998	Rossi

Esempio

- Calcolare cognome e filiale di tutti gli impiegati

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Neri	Napoli	55
5998	Neri	Milano	64
9553	Rossi	Roma	32
5998	Rossi	Roma	40

$\pi_{\text{Cognome,Filiale}}(\text{Impiegati})$

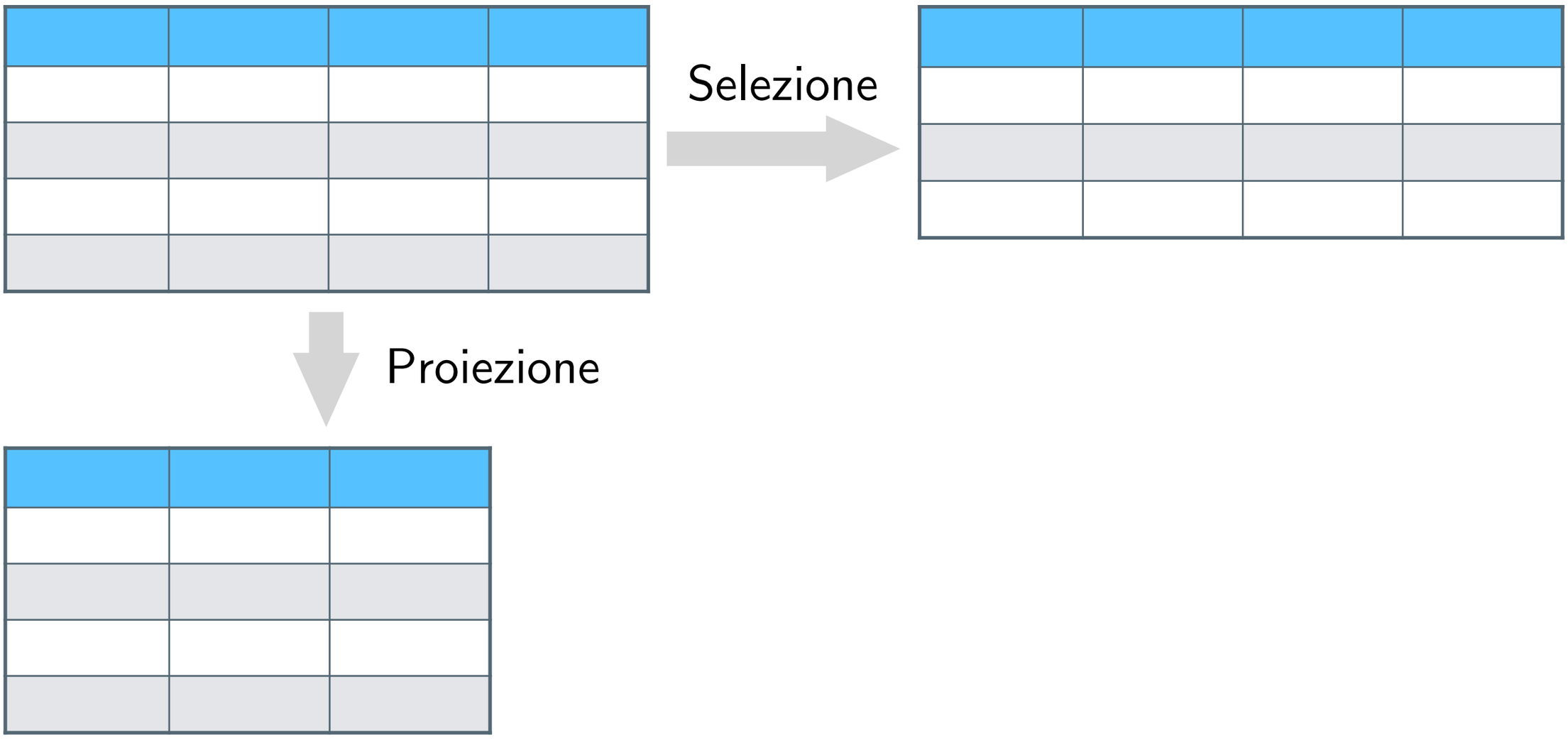
Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma
Rossi	Roma

Cardinalità delle Proiezioni

- Una proiezione
  - Può contenere al più tante n-uple quante ne ha l'operando
  - Può contenerne di meno
- Se X è una superchiave di R allora $\pi_X(R)$ contiene esattamente tante tuple quante ne ha R

Proiezione e Selezione

- **Selezione** σ: decomposizione **orizzontale**
- **Proiezione** π: decomposizione **verticale**



Esempio

- Calcolare matricola e cognome degli impiegati che guadagnano più di 50000 euro

Impiegati

Matricola	Cognome	Filiale	Stipendio
7309	Rossi	Roma	55
5998	Neri	Milano	64
9553	Milano	Milano	32
5698	Neri	Napoli	40

$$\pi_{\textbf{Matricola},\textbf{Cognome}}\left(\sigma_{\textbf{Stipendio}\ >50}(\textbf{Impiegati})\right)$$

Matricola	Cognome
7309	Rossi
5998	Neri

Selezione e Proiezione

- Combinando selezione e proiezione possiamo estrarre informazioni da **una sola** relazione
- **Non** possiamo **combinare informazioni** presenti in **relazioni diverse**
- **Non** possiamo **combinare informazioni** presenti in n-**uple diverse** della stessa relazione

Esempio

- Prove scritte in un concorso pubblico:
  - I compiti sono anonimi e a ognuno è associata una busta chiusa con il nome del candidato
  - Ogni compito e la relativa busta è contrassegnato con uno stesso numero

Esempio

Numero	Voto
1	25
2	13
3	27
4	28

Numero	Candidato
1	Mario Rossi
2	Nicola Russo
3	Mario Bianchi
4	Remo Neri

Numero	Candidato	Voto
1	25	Mario Rossi
2	13	Nicola Russo
3	27	Mario Bianchi
4	28	Remo Neri

Join Naturale

- Operatore con **due operandi** (generalizzabile)
- Produce un risultato:
  - **sull'unione degli attributi** degli operandi
  - contiene **le n-uple** costruite **ciascuna** a partire da **una n-upla** di **ognuno** degli operandi

Join Naturale

- Date due relazioni $R_1(X_1)$ e $R_2(X_2)$, in generale questo operatore si scrive come

$$R_1 \bowtie R_2$$

- Il risultato è una relazione $R(X_1 \cup X_2)$ definita come

$$R(X_1 \cup X_2) = R_1(X_1) \bowtie R_2(X_2) =$$
$$= \{t \mid \text{esistono } t_1 \in R_1 \text{ e } t_2 \in R_2$$
$$\text{con } t[X_1] = t_1 \text{ e } t[X_2] = t_2\}$$

Esempio

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

$\bowtie$

Reparto	Capo
A	Mori
B	Bruni

$=$

Impiegato	Reparto	Capo
Rossi	A	Mori
Neri	B	Bruni
Bianchi	B	Bruni

Ogni n-upla contribuisce al risultato: **join completo**

Esempio

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

$\bowtie$

Reparto	Capo
B	Mori
C	Bruni

$=$

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori

Esempio

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

$\bowtie$

Reparto	Capo
D	Mori
C	Bruni

$=$

Impiegato	Reparto	Capo

Esempio

Impiegato	Reparto
Rossi	B
Neri	B

$\bowtie$

Reparto	Capo
B	Mori
B	Bruni

$=$

Impiegato	Reparto	Capo
Rossi	B	Mori
Rossi	B	Bruni
Neri	B	Mori
Neri	B	Bruni

Cardinalità del Join

- Il join di R_1 e R_2 contiene un numero di n-uple:
  - Compreso fra 0 e il prodotto di $|R_1|$ e $|R_2|$
- Se il join coinvolge una chiave di R_2 allora il numero di n-uple è
  - Compreso fra 0 e $|R_1|$
- Se il join coinvolge una chiave di R_2 e un vincolo di integrità referenziale allora il numero delle n-uple è
  - Uguale a $|R_1|$

Cardinalità del Join

- Il join di $R_1(A, B)$ e $R_2(B, C)$ contiene un numero di n-uple:

$$0 \leq |R_1 \bowtie R_2| \leq |R_1| \times |R_2|$$

- Se B è una chiave di R_2 allora il numero di n-uple è

$$0 \leq |R_1 \bowtie R_2| \leq |R_1|$$

- Se B è una chiave di R_2 ed esiste un vincolo di integrità referenziale fra B (in R_1) e R_2 allora il numero delle n-uple è

$$|R_1 \bowtie R_2| = |R_1|$$

Join, una difficoltà

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

$\bowtie$

Reparto	Capo
B	Mori
C	Bruni

$=$

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori

Alcune n-uple non contribuiscono al risultato: vengono “tagliate fuori”

Join Esterno

- Il **join esterno** estende, con **valori nulli**, le n-uple che verrebbero tagliate fuori da un **join (interno)**
- Ne esistono **tre versioni**:
  - **Sinistro**: mantiene tutte le n-uple **del primo operando**, estendendole con valori nulli se necessario
  - **Destro**: mantiene tutte le n-uple **del secondo operando**, estendendole con valori nulli se necessario
  - **Completo**: mantiene tutte le n-uple **di entrambi gli operandi**, estendendole con valori nulli se necessario

Join Esterno Sinistro

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

$\bowtie_{\text{LEFT}}$

Reparto	Capo
B	Mori
C	Bruni

=

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori
Rossi	A	**NULL**

Join Esterno Destro

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

$\bowtie_{\text{RIGHT}}$

Reparto	Capo
B	Mori
C	Bruni

=

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori
NULL	C	Bruni

Join Esterno Completo

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

$\bowtie_{\text{FULL}}$

Reparto	Capo
B	Mori
C	Bruni

=

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori
Rossi	A	**NULL**
NULL	C	Bruni

Join e Proiezioni

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparto	Capo
B	Mori
C	Bruni

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Mori

Impiegato	Reparto
Neri	B
Bianchi	B

Reparto	Capo
B	Mori

Join e Proiezioni

- Date due relazioni $R_1(X_1)$ e $R_2(X_2)$

$$\pi_{X_1}\left(R_1 \bowtie R_2\right) \subseteq R_1$$

Join e Proiezioni

Impiegato	Reparto	Capo
Neri	B	Mori
Bianchi	B	Bruni
Verdi	A	Bini

Impiegato	Reparto
Neri	B
Bianchi	B
Verdi	A

Reparto	Capo
B	Mori
B	Bruni
A	Bini

Impiegato	Reparto	Capo
Neri	B	Mori
Neri	B	Bruni
Bianchi	B	Mori
Bianchi	B	Bruni
Verdi	A	Bini

Join e Proiezioni

- Date due relazioni $R_1(X_1)$ e $R_2(X_2)$

$$\pi_{X_1}\left(R_1 \bowtie R_2\right) \subseteq R_1$$

- Data una relazione $R(X)$ con $X = X_1 \cup X_2$

$$\left(\pi_{X_1}(R) \bowtie \pi_{X_2}(R)\right) \supseteq R$$

Prodotto Cartesiano

- Date due relazioni $R_1(X_1)$ e $R_2(X_2)$ senza attributi a comuni, cioè $X_1 \cap X_2 = \emptyset$, la definizione di join naturale funziona ugualmente
- La relazione risultante contiene sempre un numero di n-uple pari al prodotto delle cardinalità degli operandi
  - Tutte le n-uple sono combinabili tra loro
- La relazione risultante corrisponde al **prodotto cartesiano** delle relazioni:

$$R = R_1 \bowtie R_2 = R_1 \times R_2$$

Esempio

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Codice	Capo
A	Mori
B	Bruni

Impiegato	Reparto	Codice	Capo
Rossi	A	A	Mori
Rossi	A	B	Bruni
Neri	B	A	Mori
Neri	B	B	Bruni
Bianchi	B	A	Mori
Bianchi	B	B	Bruni

Theta-Join

- Nella pratica, il prodotto cartesiano ha senso (quasi) solo se seguito da una selezione:

$$\sigma_F \left(R_1 \times R_2 \right)$$

- Questa composizione di operatori è un **operatore derivato** chiamato **theta-join** e indicato come:

$$R_1 \bowtie_F R_2$$

- La condizione F è spesso una **congiunzione** (**AND**) di atomi di confronto $A_1 \ \vartheta \ A_2$ dove ϑ è un **operatore di confronto** ($\leq , < , = , \ldots$) e A_1 e A_2 sono attributi di relazioni diverse
- Se l'operatore di confronto è l'uguaglianza (=) allora si parla di **equi-join**

Esempio

Impiegati

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B

Reparti

Codice	Capo
A	Mori
B	Bruni

Impiegati $\bowtie_{\textbf{Reparto}=\textbf{Codice}}$ **Reparti**

Impiegato	Reparto	Codice	Capo
Rossi	A	A	Mori
Neri	B	B	Bruni
Bianchi	B	B	Bruni

Esercizio

Impiegati

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45
5998	Bianchi	37	38
9553	Neri	42	35
5698	Bruni	43	42
4076	Mori	45	50
8123	Lupi	46	60

Supervisione

Impiegato	Capo
7309	5698
5998	5698
9553	4076
5698	4076
4076	8123

Impiegati

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45

Supervisione

Impiegato	Capo
7309	5698

- Trovare matricola, nome, età e stipendio degli impiegati che guadagnano più di 40

$$\sigma_{\textbf{Stipendio}>40}\left(\textbf{Impiegati}\right)$$

Impiegati

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45

Supervisione

Impiegato	Capo
7309	5698

- Trovare matricola, nome ed età degli impiegati che guadagnano più di 40

$$\pi_{\mathbf{Matricola},\mathbf{Nome},\mathbf{Età}}\Big(\sigma_{\mathbf{Stipendio}>40}\left(\mathbf{Impiegati}\right)\Big)$$

Impiegati

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45

Supervisione

Impiegato	Capo
7309	5698

- Trovare i capi degli impiegati che guadagnano più di 40

$$\pi_{\textbf{Capo}}\Big(\textbf{Supervisione} \bowtie_{\textbf{Impiegato}=\textbf{Matricola}} \sigma_{\textbf{Stipendio}>40}\left(\textbf{Impiegati}\right)\Big)$$

Impiegati

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45

Supervisione

Impiegato	Capo
7309	5698

- Trovare nome e stipendio dei capi degli impiegati che guadagnano più di 40

$$\pi_{\mathbf{Nome,Stipendio}}(\mathbf{Impiegati} \bowtie_{\mathbf{Matricola}=\mathbf{Capo}} \pi_{\mathbf{Capo}}(\mathbf{Supervisione} \bowtie_{\mathbf{Impiegato}=\mathbf{Matricola}} \sigma_{\mathbf{Stipendio}>40}(\mathbf{Impiegati})))$$

Impiegati

Matricola	Nome	Età	Stipendio
7309	Rossi	34	45

Supervisione

Impiegato	Capo
7309	5698

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

$$\pi_{\mathbf{Matr,Nome,Stip,MatrC,NomeC,StipC}}\Big(\sigma_{\mathbf{Stip>StipC}}\Big(\rho_{\mathbf{MatrC,NomeC,StipC,EtàC\leftarrow Matr,Nome,Stip,Età}}(\mathbf{Impiegati}) \bowtie_{\mathbf{MatrC=Capo}} \mathbf{Supervisione} \bowtie_{\mathbf{Impiegato=Matr}} \mathbf{Impiegati}\Big)\Big)$$

Operatori Relazionali Fondamentali

- Sorprendentemente, cinque operatori relazionali sono sufficienti per qualsiasi interrogazione
  - Selezione $\sigma_C(R)$
  - Proiezione $\pi_X(R)$
  - Prodotto cartesiano $R_1 \times R_2$
  - Unione $R_1 \cup R_2$
  - Differenza $R_1 - R_2$
- Tutti gli altri sono operatori derivati/di convenienza

Divisione

- Dati due insiemi di **attributi disgiunti** X_1 e X_2, una relazione r su $X_1 \cup X_2$ e una relazione r_2 su X_2, la **divisione** $r \div r_2$ è una relazione su X_1 che contiene le n-uple ottenute come "proiezione" di n-uple di r che si combinano con tutte le n-uple di r_2 per formare n-uple di r:

$$r \div r_2 = \{t_1 \text{ su } X_1 \mid \text{per ogni } t_2 \in r_2 \text{ esiste } t \in r \text{ con } t[X_1] = t_1 \text{ e } t[X_2] = t_2\}$$

Divisione

Sedi

Filiale	Ufficio
Roma	Acquisti
Roma	Vendite
Roma	Studi
Milano	Acquisti
Milano	Vendite
Milano	Studi
Napoli	Acquisti
Napoli	Vendite

Uffici

Ufficio
Acquisti
Vendite
Studi

Sedi ÷ Uffici

Filiale
Milano
Roma

Divisione

- **L'operatore divisione è derivato** perché può essere espresso con altri operatori nel seguente modo:

$$r \div r_2 = \pi_{X_1}(r) - \pi_{X_1}\left(\left(\pi_{X_1}(r) \times r_2\right) - r\right)$$

- dove
  - $\pi_{X_1}(r) \times r_2$ contiene le n-uple di $\pi_{X_1}(r)$ "estese" con tutti i possibili valori di r_2
  - $\left(\pi_{X_1}(r) \times r_2\right) - r$ contiene le "estensioni" di $\pi_{X_1}(r)$ che non compaiono in r
  - $\pi_{X_1}\left(\left(\pi_{X_1}(r) \times r_2\right) - r\right)$ contiene le n-uple di $\pi_{X_1}(r)$ per le quali un qualche "completamento" con r_2 non compare in r
  - Togliendo queste ultime n-uple a $\pi_{X_1}(r)$ otteniamo le n-uple di $\pi_{X_1}(r)$ che si "combinano" con tutte le n-uple di r_2, cioè il risultato della divisione

Chiusura transitiva

- 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

- Nell'esempio precedente, basterebbe eseguire il join della relazione con se stessa, previa opportuna ridenominazione
- Aggiungiamo una nuova $n-$upla

Impiegato	Capo
Rossi	Lupi
Neri	Bruni
Lupi	Falchi
Falchi	Leoni

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

Chiusura transitiva

- Non esiste la possibilità di esprimere l'interrogazione che calcoli la chiusura transitiva di una relazione qualunque
- In algebra relazionale l'operazione si simulerebbe con un numero di **join illimitato**

Equivalenza di Espressioni

- Due **espressioni** sono **equivalenti** se producono lo stesso risultato qualunque sia l'istanza attuale della base di dati
- L'equivalenza è importante nella pratica perché i DBMS cercano di eseguire **espressioni equivalenti** a quelle date, ma **meno "costose"**
- Il **costo dell'esecuzione** di un'interrogazione viene valutato in termini delle **dimensioni dei risultati intermedi** della valutazione dell'espressione dell'algebra relazionale

Equivalenze I

- **Atomizzazione delle selezioni**: una congiunzione di selezioni può essere sostituita da una sequenza di selezioni atomiche

$$\sigma_{F_1 \wedge F_2}(E) \equiv \sigma_{F_1}\big(\sigma_{F_2}(E)\big)$$

  con F_1 e F_2 espressioni Booleane ed E espressione qualsiasi

- **Idempotenza delle proiezioni**: una proiezione può essere trasformata in una sequenza di proiezioni che eliminano i vari attributi in varie fasi

$$\pi_X(E) \equiv \pi_X\big(\pi_{XY}(E)\big)$$

  con E espressione definita su un insieme di attributi che contiene X e Y

Equivalenze II

- **Push selections down**:

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

se la condizione F coinvolge solo attributi della espressione E_2

- **Push projections down**:

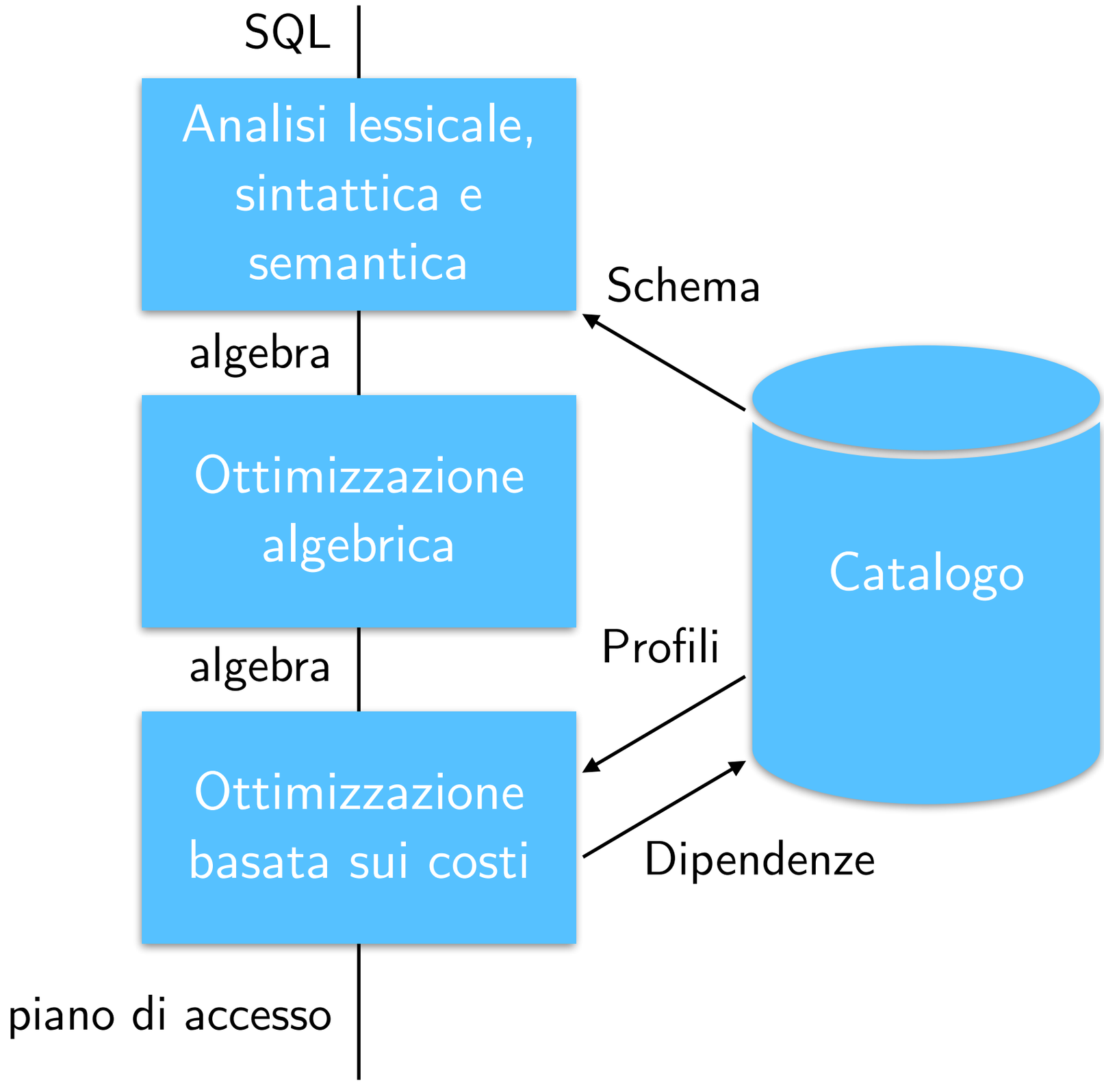
$$\pi_{X_1 Y_2}\left(E_1 \bowtie E_2\right) \equiv E_1 \bowtie \pi_{Y_2}\left(E_2\right)$$

dove X_1 sono gli attributi di E_1, X_2 sono gli attributi di E_2, $Y_2 \subseteq X_2$ e gli attributi $X_2 - Y_2$ non sono coinvolti nel join (cioè $X_1 \cap X_2 \subseteq Y_2$)

Ottimizzazione delle Interrogazioni

- ***Query processor*** (od **ottimizzatore**): un modulo del DBMS
- Più importante nei sistemi attuali che in quelli "vecchi" (gerarchici e reticolari):
  - Le interrogazioni sono espresse **ad alto livello** (ricordare il concetto di indipendenza dei dati):
    - insiemi di n-uple
    - poca proceduralità
- L'ottimizzatore sceglie la **strategia realizzativa** (di solito fra diverse alternative), a partire dall'istruzione SQL

Esecuzione delle Interrogazioni



Profili delle Relazioni

- **Informazioni quantitative**:
  - **cardinalità** di ciascuna relazione
  - **dimensioni** delle n-uple
  - **dimensioni** dei valori
    - **numero** di valori distinti degli attributi
    - valore **minimo** e **massimo** di ciascun attributo
- Sono **memorizzate** nel "catalogo" e **aggiornate** con comandi del tipo `update statistics`
- Utilizzate nella **fase finale** dell'ottimizzazione, per **stimare** le **dimensioni** dei **risultati intermedi**

Ottimizzazione Algebrica

- Il termine **ottimizzazione** è **improprio** (anche se efficace) perché il processo utilizza **euristiche**
- Si basa sulla nozione di **equivalenza**:
  - Due **espressioni** sono **equivalenti** se **producono lo stesso risultato** qualunque sia l'istanza attuale della base di dati
- I DBMS cercano di eseguire espressioni equivalenti a quelle date, ma **meno "costose"**
- Euristica fondamentale:
  - selezioni e proiezioni il più presto possibile (per **ridurre le dimensioni dei risultati intermedi**):
    - "**push selections down**"
    - "**push projections down**"

Grafo

- Un **grafo** $G = (V, E)$ consiste in:
  - un insieme V di vertici (o nodi)
  - un insieme E di coppie di vertici, detti archi
    - ogni arco connette due vertici
- Grafo **orientato** (o **diretto**): ogni arco è orientato e rappresenta relazioni orientate tra coppie di oggetti
- Grafo **non orientato** (o **non diretto**): gli archi non hanno un orientazione e rappresentano relazioni simmetriche tra coppie di oggetti

Cammino e Ciclo

- Un **cammino** in un grafo $G = (V, E)$ da un vertice x ad un vertice y è dato da una sequenza di vertici $(v_0, v_1, \ldots, v_k)$ di V con $v_0 = x$ e $v_k = y$ tale che per ogni $1 \leq i \leq k$, l'arco $(v_{i-1}, v_i) \in E$
- Un **cammino** $(v_0, v_1, \ldots, v_k)$ tale che $v_0 = v_k$ è detto **ciclo**
- Un **grafo diretto** è detto **aciclico** se non contiene cicli

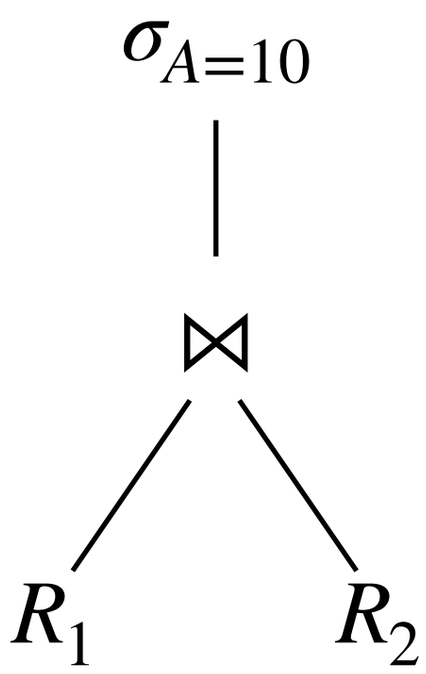
Albero

- Un **grafo non orientato** si dice **connesso** se esiste un cammino tra ogni coppia di vertici.
- Un **albero** è un grafo non orientato nel quale due vertici qualsiasi sono connessi da uno e un solo cammino

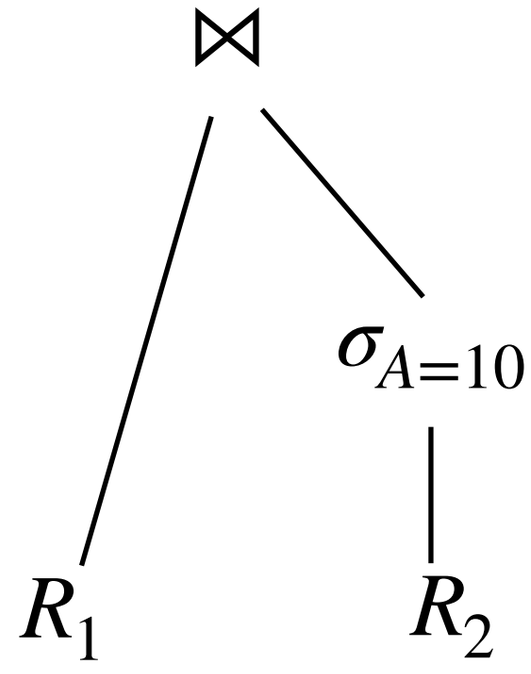
Rappresentazione Interna delle Interrogazioni

- **Alberi**:
  - **Foglie**: **dati** (relazioni, file)
  - **Nodi intermedi**: **operatori** (operatori algebrici, poi effettivi operatori di accesso ai dati)

$\sigma_{A=10}(R_1 \bowtie R_2)$



$R_1 \bowtie \sigma_{A=10}(R_2)$



Procedura Euristica di Ottimizzazione

1. **Decomporre** le **selezioni congiuntive** in successive selezioni atomiche
2. **Anticipare** il più possibile le **selezioni**
3. In una sequenza di selezioni, **anticipare** le più **selettive**
4. **Combinare prodotti cartesiani** e **selezioni** per formare **join**
5. **Anticipare** il più possibile le **proiezioni** (anche introducendone di nuove)

Esempio

- $R_1(ABC)$, $R_2(DEF)$, $R_3(GHI)$
- Interrogazione:

  SELECT A, E

  FROM R_1, R_2, R_3

  WHERE

  $$B > 100 \textbf{ AND } H = 7 \textbf{ AND } I > 2 \textbf{ AND } C = D \textbf{ AND } F = G$$

- dove:
  - FROM: prodotto cartesiano
  - WHERE: selezione
  - SELECT: proiezione

$$\pi_{AE}\left(\sigma_{B>100 \textbf{ AND } H=7 \textbf{ AND } I>2 \textbf{ AND } C=D \textbf{ AND } F=G}(r_1 \bowtie r_2 \bowtie r_3)\right)$$

Esempio

- L'espressione

$$\pi_{AE}\left(\sigma_{B>100}\ \textbf{AND}\ _{H=7}\ \textbf{AND}\ _{I>2}\ \textbf{AND}\ _{C=D}\ \textbf{AND}\ _{F=G}(r_1 \bowtie r_2 \bowtie r_3)\right)$$

- diventa (passi 1, 2, 3 e 4)

$$\pi_{AE}\left(\sigma_{B>100}(r_1) \bowtie_{C=D} r_2\right) \bowtie_{F=G} \sigma_{I>2}\left(\sigma_{H=7}(r_3)\right)$$

- diventa (passo 5)

$$\pi_{AE}\left(\pi_{AEF}((\pi_{AC}(\sigma_{B>100}(r_1))) \bowtie_{C=D} r_2) \bowtie_{F=G} \pi_G(\sigma_{I>2}(\pi_{GI}(\sigma_{H=7}(r_3))))\right)$$

Esercizio

- Si consideri il seguente schema di base di dati
  - Film(CodiceFilm, Titolo, CodiceRegista, Anno)
  - Produzione(CasaProduzione, Nazionalità, CodiceFilm, Costo, Incasso1annoSala)
  - Artista(CodiceAttore, Cognome, Nome, Sesso, DataDiNascita, Nazionalità)
  - Interpretazione(CodiceFilm, CodiceAttore, Personaggio, SessoPersonaggio)
  - Regista(CodiceRegista, Cognome, Nome, Sesso, DataDiNascita, Nazionalità)
  - Noleggio(CodiceFilm, Incasso1annoVideo, Incasso1annoDVD)
- Formulare in algebra relazionale la seguente interrogazione:
  - Nomi e cognomi dei registi che hanno diretto film che hanno incassato il primo anno di uscita meno nelle sale che per il noleggio di DVD

Esercizio

- Formulare in algebra relazionale la seguente interrogazione:
  - Nomi e cognomi dei registi che hanno diretto film che hanno incassato il primo anno di uscita meno nelle sale che per il noleggio di DVD

$$
\begin{aligned}
&\pi_{\mathbf{N,C}}(\\
&\quad \pi_{\mathbf{N,C,CF}}(\pi_{\mathbf{N,C,CR}}(\mathbf{Regista}) \bowtie \pi_{\mathbf{CF,CR}}(\mathbf{Film})) \\
&\qquad \bowtie \\
&\quad \pi_{\mathbf{CF}}(\sigma_{\mathbf{Inc1Sala}<\mathbf{Inc1DVD}}(\\
&\qquad\quad \pi_{\mathbf{Inc1Sala,CF}}(\mathbf{Produzione}) \\
&\qquad\qquad \bowtie \\
&\qquad\quad \pi_{\mathbf{Inc1DVD,CF}}(\mathbf{Noleggio}) \\
&\quad)) \\
&)
\end{aligned}
$$

Relazioni Derivate

- **Relazioni di base**: contenuto autonomo
- **Relazioni derivate**: contenuto funzione del contento di altre relazioni
  - Rappresentazioni **diverse** per gli **stessi** dati
  - Definite per mezzo di **interrogazioni**
  - Le relazioni derivate possono essere definite su altre relazioni derivate ma...
- Due tipi di relazioni derivate:
  - **Viste materializzate** e
  - **Viste virtuali**, o più semplicemente **viste**

Esempio di Vista

Afferenza

Impiegato	Reparto
Rossi	A
Neri	B
Bianchi	B
Verdi	C

Direzione

Reparto	Capo
A	Mori
B	Bruni
C	Leoni

- Una vista:

$$\textbf{Supervisione} = \pi_{\textbf{Impiegato},\textbf{Capo}} \left(\textbf{Afferenza} \bowtie \textbf{Direzione}\right)$$

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 derivate **non memorizzate nella base di dati**
- Sono supportate da tutti i DBMS
- Una interrogazione su una vista è eseguita "ricalcolando" la vista (o quasi)

Interrogazioni su viste

- Sono eseguite sostituendo alla vista la sua definizione:
- L'interrogazione

$$\sigma_{\textbf{Capo='Leoni'}}\left(\textbf{Supervisione}\right)$$

- è eseguita come

$$\sigma_{\textbf{Capo='Leoni'}}\left(\textbf{Supervisione}\right) =$$

$$= \sigma_{\textbf{Capo='Leoni'}}\Big($$

$$\pi_{\textbf{Impiegato,Capo}}\left(\textbf{Afferenza} \bowtie \textbf{Direzione}\right)$$

$$\Big)$$

Perché le viste?

- Le viste sono uno **strumento di programmazione**:
  - Si può semplificare la scrittura di interrogazioni: espressioni complesse e sotto-espressioni ripetute
- L'uso delle viste virtuali **non influisce sull'efficienza** delle interrogazioni

Esempio

- Supponiamo di avere le seguenti relazioni:

$$R_1(ABC),\ R_2(DEF),\ R_3(GH)$$

- e di definire la seguente vista R:

$$R = \sigma_{A>D}(R_1 \bowtie R_2)$$

- Un'interrogazione può essere definita:
  - Senza vista:

$$\sigma_{B=G}\left(\sigma_{A>D}\left(R_1 \bowtie R_2\right) \bowtie R_3\right)$$

  - Con vista:

$$\sigma_{B=G}\left(R \bowtie R_3\right)$$

Viste e 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**!
- Ben **pochi aggiornamenti sono ammissibili** sulle viste

Calcolo Relazionale

- Famiglia di **linguaggi dichiarativi** basati sul **calcolo dei predicati del primo ordine**
- Diverse versioni:
  - calcolo **relazionale sui domini**
    - **calcolo sui domini**, in breve
  - calcolo **su n-uple con dichiarazione di *range***
    - **calcolo sulle tuple**, in breve
    - base per il linguaggio SQL

Assunzioni

- I **simboli di predicato** corrispondono alle **relazioni** presenti nella base di dati (più alcuni predicati standard quali uguaglianza e diseguaglianza)
  - Non compaiono simboli di funzione
- Nel calcolo relazionale vengono utilizzate prevalentemente **formule aperte**, cioè formule con variabili libere, il cui valore di verità dipende dai valori assegnati alle variabili libere
  - Il risultato di un'interrogazione (formula aperta) è costituito dalle tuple di valori che, sostituiti alle variabili libere, la rendono vera
- In coerenza con quanto fatto in algebra relazionale (attributi con nome), utilizzeremo una **notazione non posizionale**

Calcolo sui domini

- **Sintassi**: le **espressioni** hanno la forma:

$$\{A_1 : x_1, \ldots, A_k : x_k \,|\, f\}$$

- dove:
  - $A_1, \ldots, A_k$ sono attributi distinti (possono anche non comparire nello schema della base di dati)
  - $x_1, \ldots, x_k$ sono variabili (che assumiamo essere distinte, nonostante non sia strettamente necessario)
  - $A_1 : x_1, \ldots, A_k : x_k$ è chiamata ***target list*** (lista degli obiettivi), e descrive il risultato
  - f è una **formula** costruita a partire da formule atomiche utilizzando eventualmente i connettivi Booleani e quantificatori $\exists x$ e $\forall x$, con x variabile

Formule atomiche

- $R(A_1 : x_1, \ldots, A_p : x_p)$, dove $R(A_1, \ldots, A_p)$ è uno **schema di relazione** e $x_1, \ldots, x_p$ sono variabili
  - Interpretabile come $[x_1, \ldots, x_p] \in R$
- x_i OP x_j
  - dove x_i e yx_j sono variabili e OP è un **operatore di confronto** $< , > = , \leq , \geq , \neq$
- x_i OP c oppure c OP x_i
  - dove c è una costante (nel dominio A_i di x_i)

Formule

- La formule atomiche sono formule
- Se f è una formula, allora anche $\neg f$ lo è
- Se f_1 e f_2 sono formule, allora anche $f_1 \wedge f_2$ lo è
- Se f_1 e f_2 sono formule, allora anche $f_1 \vee f_2$ lo è
- Se f è una formula e x una variabile, allora anche $\exists x(f)$ e $\forall x(f)$ sono formule, dove $\exists$ e $\forall$ sono **quantificatori**
- Per convenienza, useremo laddove necessario le parentesi
- Per convenienza, raggrupperemo le variabili usate nei quantificatori con la medesima formula; per esempio $\exists x(\exists y(f)) \equiv \exists x, y(f)$

Calcolo sui domini

- Il **valore di verità di una formula** è definito nel modo seguente (assumiamo per semplicità che tutti gli attributi abbiano lo stesso dominio):
  - una formula atomica $R(A_1 : x_1, \ldots, A_p : x_p)$ è vera sui valori di $x_1, \ldots, x_p$ che costituiscono una n-upla di R
  - una formula atomica $x\theta y$ $(x\theta c)$ è vera sui valori x e y che rendono vera la condizione θ
  - il valore di verità di $\wedge$, $\vee$ e $\neg$ è definito nel modo usuale
  - una formula della forma $\exists x(f)$ (rispettivamente, $\forall x(f)$) è vera se esiste almeno un elemento del dominio che (rispettivamente, ogni elemento del dominio), sostituito ad x, rende vera f

Calcolo sui domini

- Un'espressione del calcolo sui domini:

$$\{A_1 : x_1, \ldots, A_k : x_k \,|\, f\}$$

- può essere "interpretata" come una **formula logica** del tipo

$$\{x_1, \ldots, x_k \,|\, f(x_1, \ldots, x_k)\}$$

- dove
  - $x_1, \ldots, x_k$ sono **variabili** o **costanti**
  - $f(x_1, \ldots, x_k)$ è un **predicato** che può essere VERO o FALSO
- **Semantica**: il risultato di un'espressione del calcolo sui domini è una relazione su $A_1, \ldots, A_k$ che contiene n-uple di tutti i possibili valori per $x_1, \ldots, x_k$ che rendono vero il predicato $f(x_1, \ldots, x_k)$ rispetto a un'istanza di base di dati a cui l'espressione è applicata

Base di dati per gli esempi

- Impiegato(Matr, Nome, Età, Stipendio)
- Supervisione(Matr, Capo)

Esempio

- Trovare matricola e nome di tutti gli impiegati che guadagnano più di 40

$$\{\text{Matr: } m, \text{ Nome: } n \mid$$
$$\text{Impiegati(Matr: } m, \text{ Nome: } n, \text{ Età: } e, \text{ Stipendio: } s) \wedge s > 40\}$$

- La formula "Impiegati(Matr: m, Nome: n, Età: e, Stipendio: s)" ci assicura che le variabili m, n, e, s assumano valori che compaiono nelle n-uple di Impiegati
- La formula "$s > 40$" ci assicura che la variabile s, che assume valori nel dominio Stipendio, assuma solo valori maggiori di 40
- Come risultati ci interessano solo gli attributi matricola e nome

Esempio

- Trovare la matricola degli impiegati con un capo di nome “Luca”

- Ci interessa la matricola m degli impiegati...

  $\{$Matr: m | Impiegati(Matr: m, Nome: n, Età: e, Stipendio: s) ...

- ... per i quali esiste una n-upla in Supervisione...

  ... $\wedge \exists m', c'$(Supervisione(Matr: m', Capo: c') ...

- ...con la stessa matricola e con attributo Capo con valore “Luca”

  ... $\wedge\, m = m' \wedge c' =$ "Luca")$\}$

- Mettendo tutto insieme:

  $\{$Matr: m | Impiegati(Matr: m, Nome: n, Età: e, Stipendio: s) $\wedge$

  $\exists m', c'$(Supervisione(Matr: m', Capo: c') $\wedge\, m = m' \wedge c' =$ "Luca")$\}$

Esempio

- Trovare i nomi dei capi che supervisionano almeno due impiegati

- Procediamo passo passo:

$$\{\text{Capo: } c \mid \text{Supervisione(Matr: } m\text{, Capo: } c) \ \ldots$$

$$\ldots \ \wedge \exists m', c' (\text{Supervisione(Matr: } m'\text{, Capo: } c') \ \ldots$$

$$\ldots \ \wedge c = c' \wedge m \neq m')\}$$

Esempio

- Trovare matricola e nome dei capi i cui impiegati guadagnano più di 40

$$
\begin{aligned}
&\{\text{Matr: } c, \text{ Nome: } n \mid \\
&\quad \text{Impiegati(Matr: } c, \text{ Nome: } n, \text{ Età: } e, \text{ Stipendio: } s) \wedge \\
&\quad \forall m', n', e', s' (\\
&\qquad \text{Impiegati(Matr: } m', \text{ Nome: } n', \text{ Età: } e', \text{ Stipendio: } s') \wedge \\
&\qquad \text{Supervisione(Capo: } c, \text{ Impiegato: } m') \wedge \\
&\qquad s' > 40 \\
&\quad) \\
&\}
\end{aligned}
$$

Calcolo sui domini: discussione

- **Pregi**:
  - Dichiaratività
- **Difetti**:
  - Verbosità (tante variabili!)
  - Possibilità di scrivere espressioni senza senso (**dipendenti dal dominio**)
    - $\{A : x, B : y \mid R(A : x) \wedge y = y\}$
      - Nel risultato compaiono tuple per qualsiasi valore del dominio di B
      - Se il dominio di B è infinito, il risultato è infinito
      - Se il dominio di B cambia, il risultato cambia (**dipendenza dal dominio**)
    - $\{A : x \mid \neg R(A : x)\}$
      - Nel risultato compaiono tuple per qualsiasi valore del dominio di A che non compaiono in R
- Nell'algebra tutte le espressioni hanno un senso (**indipendenza dal dominio**)

Indipendenza dal dominio

- **Un'espressione** di un linguaggio di interrogazione si dice **indipendente dal dominio** se il suo risultato, su ciascuna istanza della base di dati, non varia al variare del dominio rispetto al quale l'espressione viene valutata (purché ogni dominio contenga almeno i valori presenti nell'istanza e nell'espressione)
- **Un linguaggio** si dice **indipendente dal dominio** se tali sono tutte le sue espressioni
- Il **calcolo sui domini** non è indipendente dal dominio
- L'**algebra relazionale** è indipendente dal dominio
  - Costruisce i risultati a partire dalle relazioni presenti nella base di dati, senza far mai riferimento ai domini degli attributi: i valori che compaiono nei risultati sono tutti presenti nell'istanza cui l'espressione viene applicata

Calcolo sulle tuple

- Il **calcolo su domini** presenta anche lo svantaggio di richiedere **numerose variabili**, spesso una per ciascun attributo di ciascuna relazione coinvolta (lo stesso con i quantificatori)
- Dal calcolo su domini al **calcolo su tuple**: **le variabili denotano tuple, non singoli valori**
- **Una variabile** per ciascuna **relazione** coinvolta
- Occorre associare una struttura (insieme degli attributi della relazione) a ciascuna variabile che consenta di individuare le singole componenti delle tuple

Calcolo sulle tuple

- Le **espressioni** hanno la forma:

$$\{T \mid L \mid f\}$$

- dove:
  - T è una ***target list*** (obiettivi dell'interrogazione)
  - L è una ***range list***
  - f è una **formula**

Target List

- Notazione:
  - x è una variabile
  - X è un insieme di attributi di una relazione
  - Y e Z sono sottoinsiemi di attributi di X di pari lunghezza
- T è una lista di elementi del tipo
- $Y : x \,.\, Z$
  - Vogliamo solo gli attributi Z della variabile x, che assumerà valori definiti in L (*range list*), e li chiameremo Y
- $x \,.\, Z \equiv Z : x \,.\, Z$
  - Vogliamo solo gli attributi Z della variabile x, che assumerà valori definiti in L (*range list*), e non li rinomineremo
- $x \,.\, * \equiv X : x \,.\, X$
  - Vogliamo tutti gli attributi della variabile x, che assumerà valori definiti in L (*range list*)

Range List

- L è una lista che contiene, senza ripetizioni, tutte le variabili della *target list*, con la relazione associata da cui sono prelevati i valori assunti dalla variabile
- In altre parole $L \equiv x_1(R_1), \ldots, x_k(R_k)$
  - x_i è una variabile
  - R_i è una relazione
- L è una **dichiarazione di range**: specifica l'insieme dei valori che possono essere assegnati alle variabili
  - Non occorrono più condizioni atomiche che vincolano una tupla ad appartenere ad una relazione.

Formule Atomiche

- Notazione:
  - x_i indica una variabile, c indica una costante
  - A_i indica un attributo, R indica una relazione
  - OP è un operatore di confronto $<, >, =, \leq, \geq, \neq$
- Formule atomiche:
  - $x_i.A_i$ OP $x_j.A_j$
  - $x_i.A_i$ OP c
  - c OP $x_i.A_i$

Formule

- La formule atomiche sono formule
- Se f è una formula, allora anche $\neg f$ lo è
- Se f_1 e f_2 sono formule, allora anche $f_1 \wedge f_2$ lo è
- Se f_1 e f_2 sono formule, allora anche $f_1 \vee f_2$ lo è
- Se f è una formula e x una variabile che indica una n-upla su R, allora anche $\exists x(R)(f)$ e $\forall x(R)(f)$ sono formule, dove $\exists$ e $\forall$ sono **quantificatori**
  - Notare che anche i **quantificatori** contengono ora delle **dichiarazioni di range**
  - $\exists x(R)(f)$ significa "esiste nella relazione R una n-upla x che soddisfa la formula f"

Esempio

- Trovare matricola, nome, età e stipendio degli impiegati che guadagnano più di 40

$$\{i\,.\,* \mid i(\text{Impiegati}) \mid i\,.\,\text{Stipendio} > 40\}$$

Esempio

- Trovare matricola e nome degli impiegati che guadagnano più di 40

$$\{i\,.\,(\text{Matr, Nome}) \mid i(\text{Impiegati}) \mid i\,.\,\text{Stipendio} > 40\}$$

Esempio

- Trovare matricola e nome dei capi i cui impiegati guadagnano più di 40

$$\{\text{Matr, Nome} : i'.(\text{Matr, Nome}) \mid$$
$$i'(\text{Impiegati}),\ s(\text{Supervisione}),\ i(\text{Impiegati}) \mid$$
$$i'.\text{Matr} = s.\text{Capo}$$
$$\wedge\, s.\text{Impiegato} = i.\text{Matr}$$
$$\wedge\, i.\text{Stipendio} > 40\}$$

Calcolo sulle n-uple: discussione

- Nel calcolo sulle n-uple le variabili rappresentano tuple quindi si ha **minore verbosità**
- **Alcune interrogazioni** importanti **non si possono esprimere**, in particolare le unioni: $R_1(AB) \cup R_2(AB)$
  - Ogni variabile nel risultato ha un solo *range*, mentre vorremmo n-uple sia della prima relazione che della seconda
  - Intersezione e differenza sono esprimibili
- Per questa ragione SQL (che è basato su questo calcolo) prevede un **operatore esplicito di unione**, ma non tutte le versioni prevedono intersezione e differenza

Calcolo e algebra: limiti

- **Calcolo** e **algebra** sono sostanzialmente **equivalenti**:
  - per ogni espressione del calcolo relazionale che sia indipendente dal dominio esiste un'espressione nell'algebra relazione equivalente a essa
  - per ogni espressione dell'algebra relazionale esiste un'espressione del calcolo relazionale equivalente a essa (e quindi indipendente dal dominio)
- Ci sono però **interrogazioni** interessanti non **esprimibili**:
  - calcolo di **valori derivati**: possiamo solo **estrarre valori**, non calcolarne di nuovi:
    - a livello di n-upla o di singolo valore (conversioni somme, differenze, etc.)
    - su insiemi di n-uple (somme, medie, etc.)
  - interrogazioni **inerentemente ricorsive**, come la **chiusura transitiva**