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"**

Equivalenza Importante

- **Push selections down**:

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

  dove A è un attributo di R_2 e k è una costante sul dominio di A

- Riduce in modo significativo la dimensione del risultato intermedio, e quindi il costo dell'operazione

Equivalenza Importante

- **Push projections down**:

$$\pi_{X_1 Y_2}\left(R_1 \bowtie R_2\right) \equiv R_1 \bowtie \pi_{Y_2}\left(R_2\right)$$

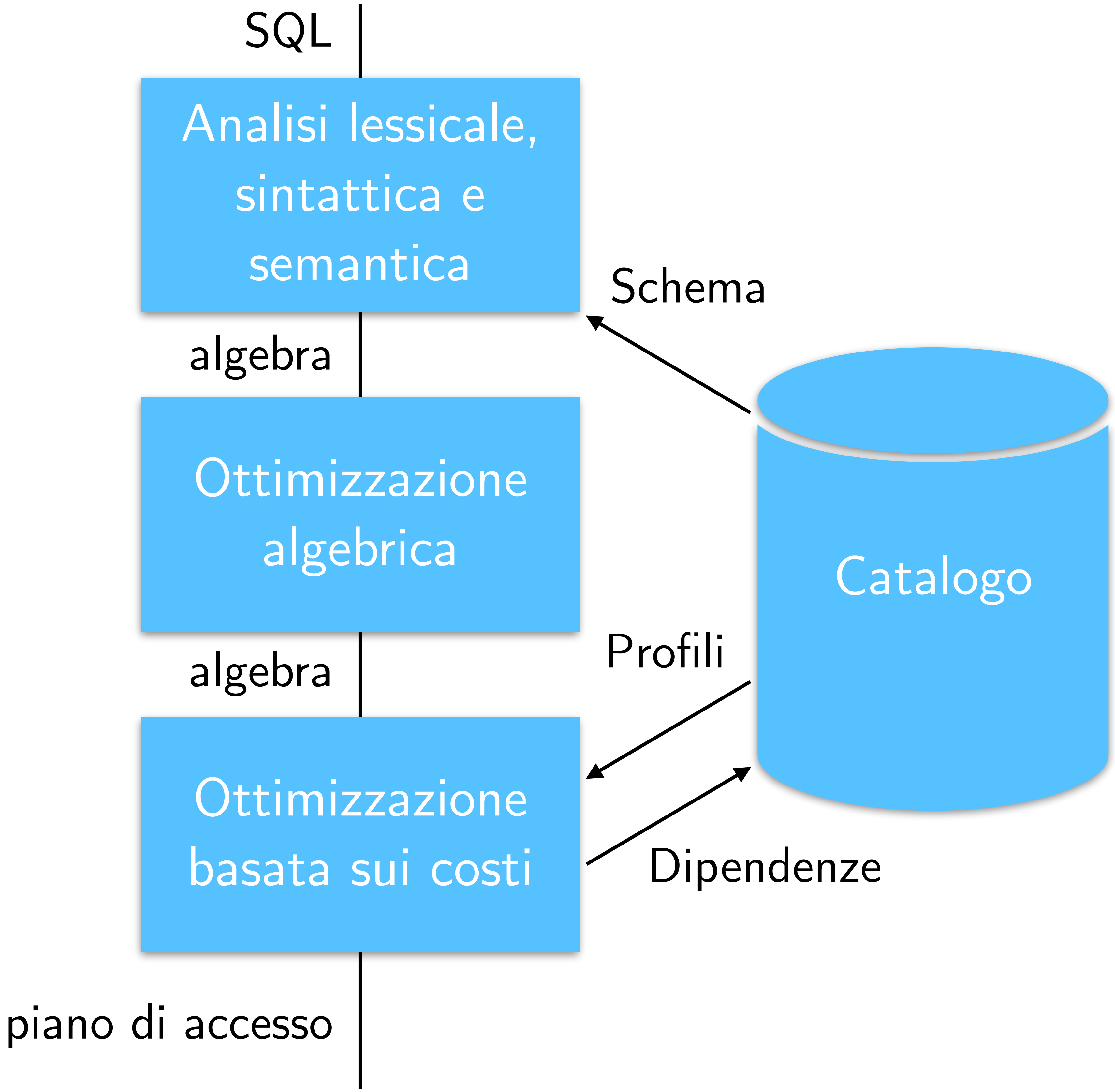
dove X_1 sono gli attributi di R_1, X_2 sono gli attributi di R_2, e gli attributi $X_2 - Y_2$ non sono coinvolti nel join

- Riduce in modo significativo la dimensione del risultato intermedio, e quindi il costo dell'operazione

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 $1 \le i \le k$, 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)

Rappresentazione Interna delle Interrogazioni

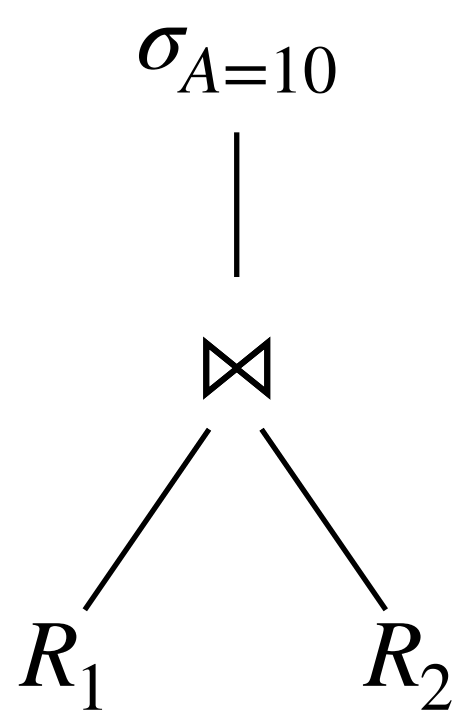
- **Alberi**:
  - **Foglie**: **dati** (relazioni, file)
  - **Nodi intermedi**: **operatori** (operatori algebrici, poi effettivi operatori di accesso ai dati)

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

Rappresentazione Interna delle Interrogazioni

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

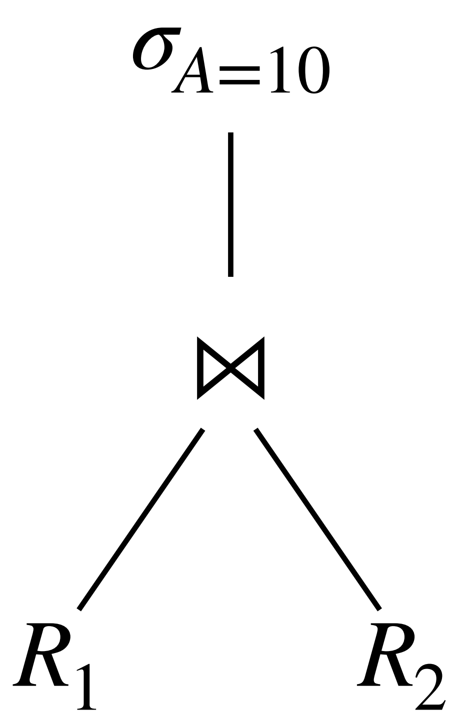
$\sigma_{A=10}(R_1 \bowtie R_2)$



Rappresentazione Interna delle Interrogazioni

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

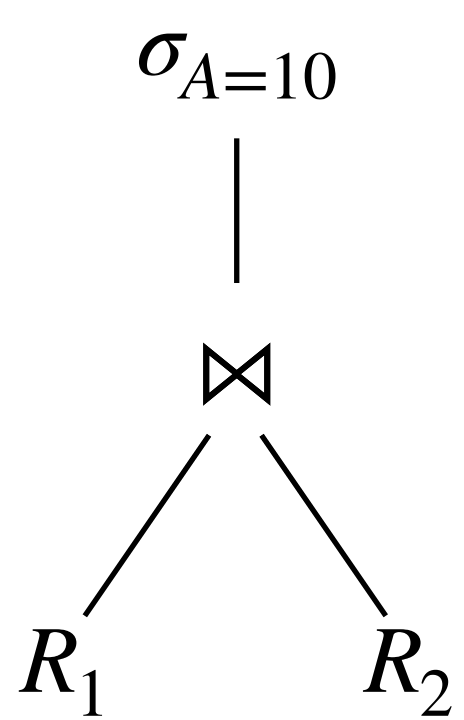
$\sigma_{A=10}(R_1 \bowtie R_2)$ $\qquad\qquad$ $R_1 \bowtie \sigma_{A=10}(R_2)$



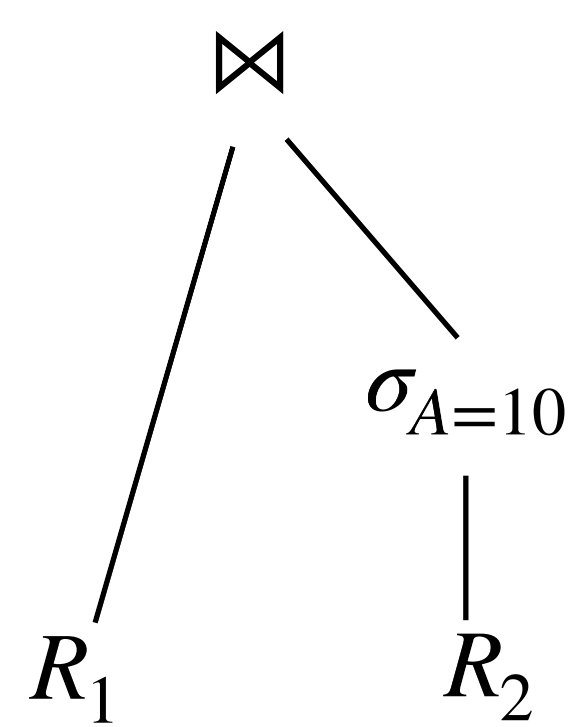
Rappresentazione Interna delle Interrogazioni

- **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

  $B > 100$ **AND** $H = 7$ **AND** $I > 2$ **AND** $C = D$ **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

Esempio

- L'espressione

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

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)

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)

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

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)

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

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

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

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

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

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

- è eseguita come

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

$$= \sigma_{\textbf{Capo}='\textbf{Leoni}'}\Big(\pi_{\textbf{Impiegato},\textbf{Capo}}\left(\textbf{Afferenza} \bowtie \textbf{Direzione}\right)\Big)$$

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

Convenzione

- Ignoriamo il join naturale
  - Vale a dire che non consideriamo implicitamente condizioni su attributi con nomi uguali
- Per "riconoscere" attributi con lo stesso nome gli premettiamo il nome della relazione seguita da "."
- Usiamo "assegnazioni", cioè viste, per ridenominare le relazioni
  - E gli attributi solo quando serve per l'unione

Esempio

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

$$\pi_{\textbf{Matr,Nome,Stip,MatrC,NomeC,StipC}}\Big($$

$$\sigma_{\textbf{Stip}>\textbf{StipC}}\big($$

$$\rho_{\textbf{MatrC,NomeC,StipC,EtàC}\leftarrow\textbf{Matr,Nome,Stip,Età}}(\textbf{Imp})$$

$$\bowtie$$

$$(\textbf{Sup} \bowtie_{\textbf{Imp}=\textbf{Matr}} \textbf{Imp})\big)\Big)$$

Esempio

Esempio

$$\pi_{\textsf{Matr,Nome,Stip,MatrC,NomeC,StipC}}\Big($$

$$\sigma_{\textsf{Stip}>\textsf{StipC}}\big($$

$$\rho_{\textsf{MatrC,NomeC,StipC,EtàC}\leftarrow\textsf{Matr,Nome,Stip,Età}}(\textsf{Imp})$$

$$\bowtie$$

$$(\textsf{Sup} \bowtie_{\textsf{Imp}=\textsf{Matr}} \textsf{Imp})\big)\Big)$$

Esempio

$$\pi_{\textsf{Matr,Nome,Stip,MatrC,NomeC,StipC}}\Big($$
$$\sigma_{\textsf{Stip}>\textsf{StipC}}\big($$
$$\rho_{\textsf{MatrC,NomeC,StipC,EtàC}\leftarrow\textsf{Matr,Nome,Stip,Età}}(\textsf{Imp})$$
$$\bowtie$$
$$(\textsf{Sup} \bowtie_{\textsf{Imp}=\textsf{Matr}} \textsf{Imp})\big)\Big)$$

Capi := Imp

Esempio

$$\pi_{\textbf{Matr,Nome,Stip,MatrC,NomeC,StipC}}\Big(\sigma_{\textbf{Stip}>\textbf{StipC}}\big(\rho_{\textbf{MatrC,NomeC,StipC,EtàC}\leftarrow\textbf{Matr,Nome,Stip,Età}}(\textbf{Imp}) \bowtie (\textbf{Sup} \bowtie_{\textbf{Imp}=\textbf{Matr}} \textbf{Imp})\big)\Big)$$

Capi := Imp

$$\pi_{\textbf{Imp.Matr,Imp.Nome,Imp.Stip,Capi.Matr,Capi.Nome,Capi.Stip}}\Big(\sigma_{\textbf{Imp.Stip}>\textbf{Capi.Stip}}\big(\textbf{Capi} \bowtie_{\textbf{Capi.Matr}=\textbf{Capo}} (\textbf{Sup} \bowtie_{\textbf{Imp}=\textbf{Imp.Matr}} \textbf{Imp})\big)\Big)$$

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 n-uple**, in breve

Calcolo sui domini

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

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

- dove:
  - f è una **formula** (con connettivi Booleani e quantificatori)
  - A_i è un nome di **attributo**
  - x_i è un nome di **variabile**
  - $A_1 : x_1, \ldots, A_n : x_n$ è chiamata ***target list***, e descrive il risultato
- **Semantica**: il **risultato** è una relazione su $A_1, \ldots, A_k$ che contiene n-uple di valori per $x_1, \ldots, x_k$ che rendono vera la formula f rispetto a un'istanza di base di dati a cui l'espressione è applicata

Formula

- f è una **formula** secondo le seguenti **regole**:
  - Esistono **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
    - $x\theta y$ o $x\theta c$, dove x e y sono variabili, c è una costante, e θ è un **operatore di confronto**
  - Se f_1 e f_2 sono formule, allora lo sono anche $f_1 \wedge f_2$, $f_1 \vee f_2$ e $\neg f_1$, e si possono usare le **parentesi**
  - Se f è una formula e x una variabile, allora anche $\exists x(f)$ e $\forall x(f)$ dove $\exists$ e $\forall$ sono **quantificatori**

Base di dati per gli esempi

- Impiegato(<u>Matr</u>, Nome, Età, Stipendio)
- Supervisione(Capo, <u>Impiegato</u>)

Esempio

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

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

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

Esempio

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

$$\pi_{\textbf{Matr},\textbf{Nome}}\left(\sigma_{\textbf{Stipendio}>40}(\textbf{Impiegati})\right)$$

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

Esempio

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

$$\{\text{Matr: } c, \text{ Nome: } n \mid$$
$$\text{Impiegati(Matr: } c, \text{ Nome: } n, \text{ Età: } e, \text{ Stipendio: } s) \land$$
$$\forall m' \forall n' \forall e' \forall s' :$$
$$\text{Impiegati(Matr: } m', \text{ Nome: } n', \text{ Età: } e', \text{ Stipendio: } s') \land$$
$$\text{Supervisione(Capo: } c, \text{ Impiegato: } m') \land s' > 40\}$$

Calcolo sui domini: discussione

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

Calcolo sulle n-uple

- Le **espressioni** hanno la forma:

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

- dove:
  - T è la ***target list***, con elementi del tipo
    - $Y : x \, . \, Z$
    - $x \, . \, Z \equiv Z : x \, . \, Z$
    - $x \, . \, * \equiv X : x \, . \, X$
    - x è una **variabile**
    - Y e Z sono liste di **attributi**
    - Gli attributi di Z devono comparire nello schema della relazione che costituisce il campo di variabilità, o ***range***, di x
  - L è la ***range list***, che elenca le variabili libere della formula f con i relativi campi di variabilità, o **range**
  - f è una **formula**

Esempio

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

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

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

Esempio

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

$$\pi_{\textbf{Matr},\textbf{Nome}}\left(\sigma_{\textbf{Stipendio}>40}(\textbf{Impiegati})\right)$$

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

Esempio

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

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

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**

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**

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:

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

Divisione

Sedi

Uffici

Sedi ÷ Uffici

Divisione

Sedi

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

Uffici

Sedi ÷ Uffici

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 $\div$ Uffici

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:

$$r \div r_2 = \pi_{X_1}(r) - \pi_{X_1}\Big(\big(\pi_{X_1}(r) \times r_2\big) - r\Big)$$

- 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
  - $\big(\pi_{X_1}(r) \times r_2\big) - r$ contiene le "estensioni" di $\pi_{X_1}(r)$ che non compaiono in r
  - $\pi_{X_1}\Big(\big(\pi_{X_1}(r) \times r_2\big) - r\Big)$ 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