Lezione 16 - Decomposizioni che hanno un join senza perdita - Esercizi

Prof.ssa Maria De Marsico
demarsico@di.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

Parte del materiale è tratto dal materiale preparato dalla prof.ssa Moscarini per il canale A-L

Definizione

- Se si decompone uno schema di relazione R si vuole che la **decomposizione** $\{R_1, R_2,\ldots, R_k\}$ ottenuta sia tale che **ogni istanza legale *r* di *R*** sia **ricostruibile** mediante **join naturale** ($\bowtie$) **da un' istanza legale** $\{r_1, r_2,\ldots, r_k\}$ **dello schema decomposto** $\{R_1, R_2,\ldots, R_k\}$. Poiché per **ricostruire una tupla *t* di r è necessario che** $t[R_i]\in r_i$, $i=1,\ldots,k$, **si deve avere** $r_i=\pi_{R_i}(r)$, $i=1,\ldots,k$.
-

- **Definizione** Sia R uno schema di relazione. Una decomposizione $\rho=\{R_1, R_2,\ldots, R_k\}$ di R *ha un join senza perdita* se **per ogni istanza legale *r*** di R si ha $r=\pi_{R1}(r)\bowtie\pi_{R2}(r)\bowtie\ldots\bowtie\pi_{Rk}(r)$.
-

Problema

- Partiamo da una decomposizione data, e cerchiamo un modo per verificare che goda della proprietà desiderata.

Algoritmo di verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Algoritmo – verifica che una decomposizione abbia del join senza perdita

Input uno schema di relazione ***R***, un insieme ***F*** di dipendenze funzionali su R, una decomposizione $\rho=\{R_1, R_2,\dots, R_k\}$ di R;

Output decide se ρ ha un join senza perdita;

begin

Costruicsi una tabella ***r*** nel modo seguente:

una colonna per ogni attributo di R e una riga per ogni elemento della decomposizione (sottoschema)

- r ha $|R|$ colonne e $|\rho|$ righe
- all'incrocio dell'i-esima riga e della j-esima colonna metti
- il simbolo a_j **se l'attributo** $A_j \in R_i$
- il simbolo b_{ij} altrimenti

l'attributo A_j **fa parte del sottoschema** R_i

Indice i = elemento della decomposizione = riga
Indice j = attributo = colonna

repeat

for every $X \rightarrow Y \in F$

do **if** ci sono due tuple t_1 e t_2 in r tali che $t_1[X]=t_2[X]$ e $t_1[Y]\neq t_2[Y]$

then for every attribute A_j in Y **do if** $t_1[A_j]=$ 'a_j' **then** $t_2[A_j]:=t_1[A_j]$

else $t_1[A_j]:=t_2[A_j]$

gestiamo correttamente anche il caso in cui $t_2[A_j]=$'a_j'

until r ha una riga con tutte 'a' **or** r non è cambiato;

if r ha una riga con tutte 'a' **then** ρ ha un join senza perdita

else ρ non ha un join senza perdita

end

anche in questo caso l'algoritmo termina sempre! occorre poi verificare se in r c'è la tupla che cerchiamo

Osservazione

- Perché entra in gioco F?

- Perché la proprietà $m_\rho(r)=r$ deve valere **per ogni istanza legale** di *R*, cioè per ogni istanza che soddisfa **TUTTE le dipendenze in F.** Se ne troviamo **anche una sola** per cui questo non vale, possiamo dire che la **decomposizione NON HA un join senza perdita.**

- L'algoritmo costruisce proprio una istanza legale **che ci permette la verifica,** e per costruirla fa in modo che soddisfi tutte le **dipendenze in *F*.**

Esempio 1

Dato il seguente schema di relazione

$$R = (A, B, C, D, E)$$

e il seguente insieme di dipendenze funzionali

$$F = \{ C \to D, AB \to E, D \to B \}$$

dire se la decomposizione

$$\rho = \{ AC, ADE, CDE, AD, B \}$$

ha un join senza perdita

Esempio 1: verifica

- Cominciamo a costruire la relativa tabella

	A	B	C	D	E
AC	a1	b12	a3	b14	b15
ADE	a1	b22	b23	a4	a5
CDE	b31	b32	a3	a4	a5
AD	a1	b42	b43	a4	b45
B	b51	a2	b53	b54	b55

- Per chiarezza applichiamo le dipendenze funzionali nell'ordine e vediamo i cambiamenti che vengono effettuati sulla tabella (ricordiamo che ogni cambiamento corrisponde a fare in modo che venga soddisfatta una dipendenza funzionale, per ottenere alla fine dell'algoritmo una tabella che rappresenta un'istanza legale dello schema):
- Indicheremo col simbolo $\rightarrow$ le modifiche ai valori della tabella e con un apice l'ordine delle sostituzioni quando opportuno

Esempio 1: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

	A	B	C	D	E
AC	a1	b12	a3	b14→a4(1)	b15
ADE	a1	b22→b12(2)	b23	a4	a5
CDE	b31	b32→b12(2)	a3	a4	a5
AD	a1	b42→b12(2)	b43	a4	b45
B	b51	a2	b53	b54	b55

F = { C→D, AB → E, D → B }

C → D la prima e la terza riga coincidono sull'attributo C =a3, quindi cambiamo b14 in a4 in modo che la dipendenza funzionale sia soddisfatta (se le righe hanno valori uguali in C, devono avere valori uguali in D)

AB → E non viene utilizzata in questo passo: la dipendenza funzionale è gia' soddisfatta, in quanto non ci sono (ancora) tuple uguali su AB e diverse su E, quindi non devono essere effettuati cambiamenti

D → B nelle prime quattro righe D= a4, quindi cambiamo b22 in b12, b32 in b12, b42 in b12 (potevano scegliere una diversa sostituzione delle b, purché le rendesse tutte uguali)

dire che due valori **diventano** uguali **non significa necessariamente** che diventano ***a***

Abbiamo completato la prima iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

Esempio 1: verifica

	A	B	C	D	E
AC	a1	b12	a3	a4	b15 $\rightarrow$ a5
ADE	a1	b12	b23	a4	a5
CDE	b31	b12	a3	a4	a5
AD	a1	b12	b43	a4	b45 $\rightarrow$ a5
B	b51	a2	b53	b54	b55

F = { C $\rightarrow$ D, AB $\rightarrow$ E, D $\rightarrow$ B }

C $\rightarrow$ D non viene utilizzata in questo passo: la dipendenza funzionale è già soddisfatta, in quanto non ci sono tuple uguali su C e diverse su D

AB $\rightarrow$ E la prima, la seconda e la quarta riga coincidono sugli attributi AB = <a1, b12>, quindi cambiamo b15 in a5 e b45 in a5 in modo che la dipendenza funzionale sia soddisfatta (se le righe hanno valori uguali in AB, devono avere valori uguali in E)

D $\rightarrow$ B non viene utilizzata in questo passo: la dipendenza funzionale è gia' soddisfatta, in quanto non ci sono tuple uguali su D e diverse su B, quindi non devono essere effettuati cambiamenti

Abbiamo completato l'iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

dire che due valori **sono** uguali **non significa necessariamente** che sono entrambi ***a***

Esempio 1: verifica

	A	B	C	D	E
AC	a1	b12	a3	a4	a5
ADE	a1	b12	b23	a4	a5
CDE	b31	b12	a3	a4	a5
AD	a1	b12	b43	a4	a5
B	b51	a2	b53	b54	b55

F = { C→D, AB → E, D → B }

C → D non viene utilizzata in questo passo

AB → E non viene utilizzata in questo passo

D → B non viene utilizzata in questo passo

La tabella non cambia più e quindi l'algoritmo **termina**.

Ora occorre verificare la presenza della tupla con tutte ***a***

Poiché non c'è una riga con tutte *a*, il join NON È senza perdita.

Esempio 2

- Dato lo schema di relazione **R = ABCDEHI** e l' insieme di dipendenze funzionali
- **F = { A $\rightarrow$ B, B $\rightarrow$ AE, DI $\rightarrow$ B, D $\rightarrow$ HI, HI$\rightarrow$ C, C $\rightarrow$ A}**
- dire se la decomposizione
- **ρ = { ACD, BDEH, CHI}** ha un join senza perdita

- Indicheremo col simbolo $\rightarrow$ le modifiche ai valori della tabella e con un apice l' ordine delle sostituzioni quando opportuno

Esempio 2: verifica

$F = \{ A \rightarrow B, B \rightarrow AE, DI \rightarrow B, D \rightarrow HI, HI \rightarrow C, C \rightarrow A\}$

	A	B	C	D	E	H	I
ACD	a1	b12	a3	a4	b15	b16	b17
BDEH	b21	a2	b23	a4	a5	a6	b27
CHI	b31	b32	a3	b34	b35	a6	a7

Esempio 2: verifica

F = { A → B, B → AE, DI → B, D → HI, HI→ C, C → A}

	A	B	C	D	E	H	I
ACD	a1	b12	a3	a4	b15	b16→a6(1)	b17
BDEH	b21→a1(3)	a2	b23→a3(2)	a4	a5	a6	b27→b17(1)
CHI	b31→a1(3)	b32	a3	b34	b35	a6	a7

A → B non si applica in questa iterazione
B → AE non si applica in questa iterazione
DI → B ci sono due tuple uguali su D ma non su I - non si applica in questa iterazione
D → HI la prima e la seconda riga coincidono sull'attributo D =a4, quindi cambiamo H e I **ma separatamente** b16 → a6 mentre b27 → b17
HI → C ora abbiamo due tuple uguali su HI (la prima e la seconda entrambe con valori <a6, b17> quindi modifichiamo i valori della C nelle stesse tuple – b23 →a3
C → A le tuple sono tutte uguali su C, quindi le facciamo diventare uguali su A, e poiché abbiamo la prima con valore a, diventano tutte a (b21 →a1, b31 →a1)

dire che due valori **diventano** uguali **non significa necessariamente** che diventano ***a***

Abbiamo completato la prima iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

Esempio 2: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

$F = \{ A \rightarrow B,\ B \rightarrow AE,\ DI \rightarrow B,\ D \rightarrow HI,\ HI \rightarrow C,\ C \rightarrow A\}$

	A	B	C	D	E	H	I
ACD	a1	b12→a2(1)	a3	a4	b15→a5(2)	a6	b17
BDEH	a1	a2	a3	a4	a5	a6	b17
CHI	a1	b32→a2(1)	a3	b34	b35→a5(2)	a6	a7

A → B le tuple sono tutte uguali su A, quindi le facciamo diventare uguali su B, e poiché abbiamo la seconda con valore a, diventano tutte a (b12 →a2, b32 →a2)

B → AE ora le tuple sono tutte uguali su B, quindi devono diventare uguali anche su AE; su A sono già uguali, la seconda tupla ha una a sull'attributo E, quindi diventano tutte a (b15 →a5, b35 →a5)

DI → B la prima e la seconda tupla sono uguali su DI <a4, b17>, quindi devono diventare uguali su B ma lo sono già

D → HI la dipendenza è già soddisfatta – niente da modificare

HI → C la dipendenza è già soddisfatta – niente da modificare

C → A la dipendenza è già soddisfatta – niente da modificare

Abbiamo completato l'iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

Esempio 2: verifica

SAPIENZA
Università di Roma
Dipartimento di Informatica

F = { A → B, B → AE, DI → B, D → HI, HI→ C, C → A}

	A	B	C	D	E	H	I
ACD	a1	a2	a3	a4	a5	a6	b17
BDEH	a1	a2	a3	a4	a5	a6	b17
CHI	a1	a2	a3	b34	a5	a6	a7

A → B la dipendenza è già soddisfatta – niente da modificare
B → AE la dipendenza è già soddisfatta – niente da modificare
DI → B la dipendenza è già soddisfatta – niente da modificare
D → HI la dipendenza è già soddisfatta – niente da modificare
HI → C la dipendenza è già soddisfatta – niente da modificare
C → A la dipendenza è già soddisfatta – niente da modificare

Abbiamo completato l'iterazione del for e la tabella <u>non è stata modificata</u>, quindi l'algoritmo termina

Ora occorre verificare la presenza della tupla con tutte ***a***

Poiché non c'è una riga con tutte *a*, il join <u>NON È</u> senza perdita.

Esempio 3

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Dato il seguente schema di relazione

$$R = (A, B, C, D, E, G)$$

e il seguente insieme di dipendenze funzionali

$$F = \{ AB \rightarrow C, DG \rightarrow B, G \rightarrow D, E \rightarrow G \}$$

dire se la decomposizione

$$\rho = \{ ABD, AEG, BCE \}$$

ha un join senza perdita e descrivere come si è arrivati alla risposta.

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22	b23	b24	a5	a6
BCE	b31	a2	a3	b34	a5	b36

Esempio 3: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22	b23	b24	a5	a6
BCE	b31	a2	a3	b34	a5	b36→a6(1)

AB → C non si applica in questa iterazione

DG → B non si applica in questa iterazione

G → D non si applica in questa iterazione

E → G la seconda e la terza riga coincidono sull'attributo E =a5, quindi cambiamo G b36 → a6

Abbiamo completato la prima iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

Esempio 3: verifica

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22	b23	b24	a5	a6
BCE	b31	a2	a3	b34→ b24(1)	a5	a6

AB → C non si applica in questa iterazione

DG → B non si applica in questa iterazione

G → D la seconda e la terza riga coincidono sull'attributo G =a6, quindi rendiamo uguali i valori di D b34 → b24

E → G non si applica in questa iterazione

Abbiamo completato l'iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

Esempio 3: verifica

Sapienza Università di Roma
Dipartimento di Informatica

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22→ a2(1)	b23	b24	a5	a6
BCE	b31	a2	a3	b24	a5	a6

AB → C non si applica in questa iterazione

DG → B la seconda e la terza tupla sono uguali <b24, a6>, quindi facciamo diventare uguali le tuple su B; la terza ha il valore a quindi b22 → a2

G → D non si applica in questa iterazione

E → G non si applica in questa iterazione

Abbiamo completato l'iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

Esempio 3: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { AB $\rightarrow$ C, DG $\rightarrow$ B, G $\rightarrow$ D, E $\rightarrow$ G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	a2	b23 $\rightarrow$ b13$^{(1)}$	b24	a5	a6
BCE	b31	a2	a3	b24	a5	a6

- AB $\rightarrow$ C prima e seconda tupla uguali su AB <a1, a2> quindi modifichiamo C b23 $\rightarrow$ b13
- DG $\rightarrow$ B non si applica in questa iterazione
- G $\rightarrow$ D non si applica in questa iterazione
- E $\rightarrow$ G non si applica in questa iterazione
- **Abbiamo completato l' iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo**

Esempio 3: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

$F = \{ AB \rightarrow C, DG \rightarrow B, G \rightarrow D, E \rightarrow G \}$

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	a2	b13	b24	a5	a6
BCE	b31	a2	a3	b24	a5	a6

$AB \rightarrow C$ non si applica in questa iterazione

$DG \rightarrow B$ non si applica in questa iterazione

$G \rightarrow D$ non si applica in questa iterazione

$E \rightarrow G$ non si applica in questa iterazione

Abbiamo completato l'iterazione del for e la tabella non è stata modificata, quindi l'algoritmo termina

Ora occorre verificare la presenza della tupla con tutte ***a***

Poiché non c'è una riga con tutte *a*, il join NON È senza perdita.

Esempio 4

- Dato lo schema di relazione
- $R = ABCDEHI$
- e l' insieme di dipendenze funzionali
- $F = \{ H \rightarrow B, DI \rightarrow H, D \rightarrow I, B \rightarrow I, B \rightarrow E, E \rightarrow C\}$
- dire se la decomposizione
- $\rho = \{ ABDE, CDH, AHI\}$
- ha un join senza perdita

	A	B	C	D	E	H	I
ABDE	a1	a2	b13	a4	a5	b16	b17
CDH	b21	b22	a3	a4	b25	a6	b27
AHI	a1	b32	b33	b34	b35	a6	a7

Esempio 4: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { H → B, DI → H, D → I, B → I, B → E, E → C}

	A	B	C	D	E	H	I
ABDE	a1	a2	b13	a4	a5	b16	b17→ a7(3)
CDH	b21	b22	a3	a4	b25	a6	b27→ b17(2) b17→ a7(3)
AHI	a1	b32→b22(1)	b33→ a3(5)	b34	b35→ b25(4)	a6	a7

H→ B seconda e terza tupla uguali su H quindi le modifichiamo su B: b32→b22

DI → H non si applica in questa iterazione

D → I prima e seconda tupla uguali su D quindi le modifichiamo su I: b27→ b17

B → I seconda e terza tupla ora sono uguali (b22) quindi le modifichiamo su I: b17(2)→ a7 (notare che alla prossima iterazione riapplicando D → I anche sulla prima tupla potremo trasformare b17→ a7, quindi possiamo anticipare cioè trasformare tutti i valori che sono già uguali tra loro)

B → E seconda e terza tupla uguali su B (b22) quindi le modifichiamo su E: b35 → b25

E → C seconda e terza tupla uguali su E (b25) quindi le modifichiamo su C: b33→ a3

Abbiamo completato la prima iterazione del for e la tabella <u>è stata modificata</u>, quindi continuiamo

Esempio 4: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

$F = \{ H \rightarrow B, DI \rightarrow H, D \rightarrow I, B \rightarrow I, B \rightarrow E, E \rightarrow C\}$

	A	B	C	D	E	H	I
ABDE	a1	a2	b13	a4	a5	b16→a6(1)	a7
CDH	b21	b22	a3	a4	b25	a6	a7
AHI	a1	b22	a3	b34	b25	a6	a7

$H \rightarrow B$ non si applica in questa iterazione

$DI \rightarrow H$ prima e seconda tupla uguali su D I <a4, a7>; le modifichiamo su H: b16 → a6

$D \rightarrow I$ non si applica in questa iterazione

$B \rightarrow I$ non si applica in questa iterazione

$B \rightarrow E$ non si applica in questa iterazione

$E \rightarrow C$ non si applica in questa iterazione

Abbiamo completato l'iterazione del for e la tabella è stata modificata, quindi continuiamo

Esempio 4: verifica

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { H → B, DI → H, D → I, B → I, B → E, E → C}

	A	B	C	D	E	H	I
ABDE	a1	a2	b13→ a3$^{(3)}$	a4	a5	a6	a7
CDH	b21	b22→ a2$^{(1)}$	a3	a4	b25→ a5$^{(2)}$	a6	a7
AHI	a1	b22→ a2$^{(1)}$	a3	b34	b25→ a5$^{(2)}$	a6	a7

H→ B tutte le tuple sono uguali su H, quindi diventano uguali anche su B: b22 → a2

DI → H non si applica in questa iterazione

D → I non si applica in questa iterazione

B → I non si applica in questa iterazione

B → E tutte le tuple sono uguali su B, quindi diventano uguali anche su E: b25 → a5

E → C tutte le tuple sono uguali su E, quindi diventano uguali anche su C: b13 → a3

Abbiamo completato l'iterazione del for e la tabella <u>è stata modificata</u>, quindi dovremmo continuare ma ...

Poiché c'è una riga con tutte *a* (la prima), possiamo fermarci e il join <u>È</u> senza perdita.

Ricordiamo il nostro esempio

Riconsideriamo lo schema

R=(Matricola, Provincia, Comune) con l'insieme di dipendenze funzionali *F={Matricola→Provincia, Comune→Provincia}* (lo schema non è in 3NF per la presenza in F^+ delle dipendenze parziali *Matricola→Provincia* e *Comune→Provincia*, dato che la chiave è (Matricola, Comune) (Comune non è determinato da nessun altro attributo!)

e riconsideriamo la decomposizione:

R1=(Matricola, Provincia) con { *Matricola→Provincia* } e
R2=(Provincia, Comune) con { *Comune→Provincia* }.

Ricordiamo che ale schema **pur preservando tutte le dipendenze in F^+** non è soddisfacente.

Ricordiamo il nostro esempio

SAPIENZA
Università di Roma
Dipartimento di Informatica

Consideriamo l' i stanza **legale** di *R*

R

Matricola	**Provincia**	**Comune**
501	Roma	Tivoli
502	Roma	Mandela

sono veri i due fatti *(501,Roma,Tivoli)* e *(501, Roma,Mandela)* e **non altri**

In base alla decomposizione data, questa istanza si decompone in

R1

Matricola	**Provincia**
501	Roma
502	Roma

R2

Provincia	**Comune**
Roma	Tivoli
Roma	Mandela

E dovrebbe essere possibile ricostruirla **esattamente** tramite join …
invece …

Ricordiamo il nostro esempio

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- ... e invece se si effettua il join delle due istanze legali risultanti dalla decomposizione si ottiene

R

Matricola	Comune	Provincia
501	Roma	Tivoli
502	Roma	Mandela
501	Roma	Mandela
502	Roma	Tivoli

tuple estranee alla realtà di interesse
quindi
perdita di informazione

Proviamo la verifica

- Per comodità rinominiamo gli attributi Matricola =A, Comune =B, Provincia =C e consideriamo lo schema R=ABC con le dipendenze F={$A \rightarrow C$, $B \rightarrow C$} e verifichiamo se la decomposizione **ρ = { AC, BC} ha un join senza perdita**

	A	B	C
AC	a1	b12	a3
BC	b21	a2	a3

- $A \rightarrow C$ non si applica in questa iterazione
- $B \rightarrow C$ non si applica in questa iterazione

- In pratica, la tabella **non viene mai modificata** e **NON ha una riga con tutte *a***, quindi come avevamo verificato empiricamente **la decomposizione non ha un join senza perdita**