

Normalizzazione di Schemi Relazionali

Prof. Alfredo Pulvirenti

Prof. Salvatore Alaimo

(Atzeni-Ceri Capitolo 9)

PROBLEMA GENERALE

- La progettazione concettuale e logica produce uno schema relazionale che rappresenta la realtà dei dati nella nostra applicazione.
- E' importante che questo schema abbia alcune proprietà.
- Studieremo queste proprietà e daremo degli algoritmi per produrre schemi buoni. Anche se spesso accade che la progettazione logica da noi descritta produce schemi già normalizzati.
- In ogni caso potremo usare queste tecniche per verificare le proprietà.

Ridondanze

Il telefono è ripetuto per ogni esame (*ridondanza*)

| Matricola | Nome | Telefono | Corso | Voto |
|-----------|------|----------|-------|------|
| | | | | |

Il Nome ed il Telefono sono funzione solo della Matricola e non dipendono dagli esami. Quindi non vanno associati ad ogni esame

Viceversa gli esami hanno bisogno solo della matricola.

Anomalie

- Se il telefono dello studente cambia allora questo deve essere aggiornato in tutti i record dello studente (*anomalia di aggiornamento*)
- Se vengono annullati gli esami dati non rimane traccia dello studente (*anomalia di cancellazione*)
- Similmente se uno studente non ha ancora dato esami non puo' essere inserito (*anomalia di inserimento*)
- Soluzione:decomporre in due relazioni!!

Dipendenze Funzionali

- Definizione:

Una **Dipendenza Funzionale** è un particolare vincolo di integrità che esprime legami funzionali tra gli attributi di una relazione.

- Esempio:

il valore di Matricola implica quelli di Nome e Telefono.
Inoltre Matricola e Corso implicano il Voto.

Matricola \rightarrow Nome, Telefono

Matricola, Corso \rightarrow Voto

Definizione di Dipendenza Funzionale

- Sia $R(A_1, A_2, \dots, A_n)$ uno schema di relazione, X ed Y sottoinsiemi di $\{A_1, A_2, \dots, A_n\}$. Diciamo che **X implica funzionalmente Y** , in simboli $X \rightarrow Y$, per ogni relazione r dello schema R , se due tuple t_1 e t_2 di r coincidono su tutti gli attributi di X allora devono anche coincidere su tutti gli attributi di Y .
- Esempio :
 - Matricola, Corso ---> Voto

Notazione

- A, B, \dots attributi;
- U, V, W, X, Y, Z insiemi di attributi;
- R schema di relazione, r relazione;
- ABC sta per $\{A, B, C\}$, XY sta per $X \cup Y$, XA e AX stanno per $X \cup \{A\}$;

Soddisfazione di Dipendenze Funzionali

- Diciamo che una relazione r soddisfa la dipendenza funzionale $X \rightarrow Y$ se per ogni coppia di tuple t_1 e t_2 in r
 $t_1[X] = t_2[X]$ implica $t_1[Y] = t_2[Y]$.

Logica delle Dipendenze Funzionali : Semantica

- Sia F un insieme di dipendenze funzionali per uno schema di relazione R e sia $X \rightarrow Y$ una dipendenza funzionale. Diciamo che F **logicamente implica** $X \rightarrow Y$, e si scrive $F \models X \rightarrow Y$, se per ogni relazione r di R che soddisfa tutte le dipendenze di F , r soddisfa anche $X \rightarrow Y$.
- Esempio: $\{A \rightarrow B, B \rightarrow C\} \models A \rightarrow C$.

Chiusura di un insieme di dipendenze funzionali

- Dato un insieme F di dipendenze funzionali la sua chiusura F^+ è l'insieme delle dipendenze funzionali che sono implicate logicamente da F , in simboli
- $F^+ = \{X \rightarrow Y \mid F \models X \rightarrow Y\}.$

Chiavi per uno schema con insieme di dipendenze funzionali

- Sia $R(A_1, A_2, \dots, A_n)$ uno schema, F un insieme di dipendenze funzionali su R ed X un sottoinsieme di $\{A_1, A_2, \dots, A_n\}$. Si dice che X è una **chiave** di (R, F) se:
 - $X \rightarrow A_1 A_2 \dots A_n \in F^+$;
 - Per ogni sottoinsieme proprio Y di X la dipendenza $Y \rightarrow A_1 A_2 \dots A_n \notin F^+$

Commenti

- Una dipendenza funzionale è **dettata dalla semantica degli attributi** di una relazione e non può essere inferita da una particolare istanza dello schema
- Una istanza di uno schema che rispetti una data dipendenza funzionale viene detta **istanza legale** dello schema rispetto alla data dipendenza funzionale
- Se X è una chiave in uno schema R allora ogni altro attributo di R dipende funzionalmente da X
- Dire che $X \rightarrow Y$ significa asserire che i valori della componente Y dipendono da (sono determinati da) i valori della componente X
- Se $X \rightarrow Y$ non necessariamente risulta anche $Y \rightarrow X$
- Il concetto di superchiave si esprime facendo uso delle Dipendenze Funzionali:

$K \subseteq T$ è superchiave di $R(T)$ se e solo se $K \rightarrow T$

Necessità di un Calcolo Logico

- Quindi il problema è quello di calcolare la chiusura di un insieme F di dipendenze funzionali. Per far ciò definiamo un calcolo logico tale che $F \models X \rightarrow Y$ se e soltanto se $X \rightarrow Y$ si può sintatticamente dedurre da F nel calcolo logico.
- I punti di partenza e le regole del calcolo sono i seguenti:

Assiomi di Armstrong

- Sia $U=\{A_1, A_2, \dots, A_n\}$ un universo di attributi
- Riflessività
 - Se $Y \subset X \subset U$ allora $F \vdash X \rightarrow Y$
- Aumento
 - Se $F \vdash X \rightarrow Y$ allora $F \vdash XZ \rightarrow YZ$
- Transitività
 - Se $F \vdash X \rightarrow Y$ e $F \vdash Y \rightarrow Z$ allora $F \vdash X \rightarrow Z$
- Notazione $X \rightarrow Y$ si deduce da F applicando gli assiomi di Armstrong si indica con $F \vdash X \rightarrow Y$

Deducibilità di dipendenze funzionali

- Diciamo che $F \vdash X \rightarrow Y$ se $X \rightarrow Y$ si può dedurre da F applicando un numero finito di volte gli assiomi di Armstrong. Cioè esiste una catena $D_1 D_2 \dots D_k = X \rightarrow Y$ tale che D_i è in F oppure si ottiene da precedenti mediante gli assiomi di Armstrong.
- Esempio $\{A \rightarrow B, B \rightarrow C, C \rightarrow D\} \vdash A \rightarrow D$ (applicando 2 volte la transitività)

Correttezza e Completezza

- **Correttezza:**

Se $F \vdash X \rightarrow Y$ allora $F \models X \rightarrow Y$

- **Completezza:**

Se $F \models X \rightarrow Y$ allora $F \vdash X \rightarrow Y$

Dimostrazione della Correttezza

- Basta dimostrare che gli assiomi di Armstrong sono corretti
- **Riflessività**
Se $Y \subset X \subset U$ allora $F \vdash X \rightarrow Y$
- **Dimostrazione:**
se $t_1[X]=t_2[X]$ allora ovviamente anche $t_1[Y]=t_2[Y]$ perché ogni attributo di Y sta anche in X .

Correttezza dell'Aumento

- **Aumento**

Se $F \vdash X \rightarrow Y$ allora $F \vdash XZ \rightarrow YZ$

- **Dimostrazione:**

Supponiamo che $r \models F$ allora, per ipotesi, $r \models X \rightarrow Y$. Supponiamo che $t_1[XZ] = t_2[XZ]$ allora $t_1[X] = t_2[X]$ da cui segue, per ipotesi, $t_1[Y] = t_2[Y]$ e quindi $t_1[YZ] = t_2[YZ]$.

Correttezza della TRANSITIVITA'

- **Transitività**

Se $F \vdash X \rightarrow Y$ e $F \vdash Y \rightarrow Z$ allora $F \vdash X \rightarrow Z$

- **Dimostrazione:**

Supponiamo che $r \models F$ allora, per ipotesi, $r \models X \rightarrow Y$ e $r \models Y \rightarrow Z$. Supponiamo che $t_1[X]=t_2[X]$ allora per la prima $t_1[Y]=t_2[Y]$ da cui per la seconda $t_1[Z]=t_2[Z]$.

Correttezza

- Supponiamo che $F \vdash X \rightarrow Y$ e sia $D_1 D_2 \dots D_n = X \rightarrow Y$ la relativa catena. Procedendo per induzione, se D_i è in F allora ovviamente $F \models D_i$. Se D_i si ottiene da precedenti D_j mediante gli assiomi di Armstrong allora per l'ipotesi induttiva $F \models D_j$ e per la correttezza dei singoli assiomi si ha $F \models D_i$. Segue $F \models D_n$ cioè $F \models X \rightarrow Y$

- Prima di verificare la completezza degli assiomi di amstrong introduciamo altre regole per derivare dipendenze funzionali

Primo lemma preliminare: La regola di Decomposizione

- Se $F \vdash X \rightarrow Y$ e $Z \subset Y$
allora $F \vdash X \rightarrow Z$

- Dimostrazione:

Se $F \vdash X \rightarrow Y$ e $Z \subset Y$ allora per riflessività
 $F \vdash Y \rightarrow Z$ da cui per transitività si ottiene
 $F \vdash X \rightarrow Z$.

Secondo lemma preliminare: La Regola dell'Unione

- Se $F \vdash X \rightarrow Y$ e $F \vdash X \rightarrow Z$ allora $F \vdash X \rightarrow YZ$

Dimostrazione:

Se $F \vdash X \rightarrow Y$ e $F \vdash X \rightarrow Z$ aumentando la prima di X e la seconda di Y si ha $F \vdash X \rightarrow XY$ e $F \vdash XY \rightarrow ZY$ che per transitività implicano $F \vdash X \rightarrow YZ$.

Terzo lemma preliminare: La Regola di pseudotransitivita'

- Se $F \vdash X \rightarrow Y$ e $F \vdash WY \rightarrow Z$ allora $F \vdash WX \rightarrow Z$

Dimostrazione:

Se $F \vdash X \rightarrow Y$ e $F \vdash WY \rightarrow Z$ aumentando la prima di W si ha $F \vdash WX \rightarrow WY$ e $F \vdash WY \rightarrow Z$ che per transitività implicano $F \vdash WX \rightarrow Z$.

Chiusura di un insieme funzionale

- La chiusura di un insieme funzionale F^+ è un lavoro che consuma molto tempo perché F^+ può essere molto grande anche se F è piccolo.

Lemma Fondamentale

- Definiamo $X_F^+ = \{A \mid F \vdash X \rightarrow A\}$.
Ometteremo l'indice F quando non c'è ambiguità e scriveremo semplicemente X^+ .

LEMMA

$F \vdash X \rightarrow Y$ se e solo se $Y \subseteq X^+$.

Dimostrazione:

(\Leftarrow) Sia $Y = A_1 A_2 \dots A_n$ e supponiamo che $Y \subseteq X^+$. Allora per definizione $F \vdash X \rightarrow A_i$ per ogni $i = 1, 2, \dots, n$. Da questa per la regola dell'unione si ha $F \vdash X \rightarrow Y$.

(\Rightarrow) Viceversa se $F \vdash X \rightarrow Y$ allora per la regola di decomposizione si ha $F \vdash X \rightarrow A_i$ per ogni $i = 1, 2, \dots, n$, e quindi $Y \subseteq X^+$.

Dimostrazione di Completezza degli Assiomi di Armstrong

- Dobbiamo dimostrare che $F \models X \rightarrow Y$ allora $F \vdash X \rightarrow Y$.
- Basta far vedere che se **not** $F \vdash X \rightarrow Y$ allora **not** $F \models X \rightarrow Y$.
- Infatti supponiamo che **not** $F \vdash X \rightarrow Y$ allora per il Lemma $Y \not\subseteq X^+$ e $Y \neq X^+$. Allora è possibile considerare la relazione r dello stesso schema fatta dalle due tuple t_1 e t_2

$t_1 =$ 1 1 ... 1 1 1 ... 1
 X^+ Not X^+

$t_2 =$ 1 1 ... 1 0 0 ... 0
 X^+ Not X^+

- Facciamo vedere che **r soddisfa tutte le dipendenze di F .**
 - Infatti, supponiamo che esista una $V \rightarrow W \in F$ tale che r non la soddisfa.
 - Questo vuol dire che t_1 e t_2 coincidono su V ma non su W . Questo vuol dire che $V \subseteq X^+$ e $W \not\subseteq X^+$.
 - Per il lemma segue che $F \vdash X \rightarrow V$ che assieme a $F \vdash V \rightarrow W$ per transitività da $F \vdash X \rightarrow W$ che contraddice $W \not\subseteq X^+$.
 - Quindi r soddisfa tutte le dipendenze di F .
- Tuttavia **r non soddisfa $X \rightarrow Y$.** Infatti $t_1[X]=t_2[X]$ ma $t_1[Y] \neq t_2[Y]$ poiché $Y \not\subseteq X^+$ in quanto, per ipotesi, ***not*** $F \vdash X \rightarrow Y$.
- Quindi ***not*** $F \models X \rightarrow Y$, che conclude la dimostrazione di completezza.

Chiusure, Equivalenze e Ricoprimenti Minimi

Prof. Alfredo Pulvirenti

Prof. Salvatore Alaimo

Calcolo delle Chiusure

- Ricordiamo che
 - $F^+ = \{X \rightarrow Y \mid F \models X \rightarrow Y\}$.
- Il calcolo può essere molto costoso in quanto ad esempio se
- $F = \{A \rightarrow B_1, A \rightarrow B_2, \dots, A \rightarrow B_n\}$ allora F^+ include $A \rightarrow Y$ per ogni Y sottoinsieme di $\{B_1, B_2, \dots, B_n\}$. Quindi ha cardinalità almeno 2^n .

Chiusura di un insieme di attributi

- Facciamo vedere invece come è possibile un calcolo efficace di $X_F^+ = \{A | F \vdash X \rightarrow A\}$.
- Il calcolo avviene attraverso il seguente Algoritmo

Algoritmo per X^+

1. $X^{(0)} := X, j = 0$
2. *REPEAT*
3. $X^{(j+1)} := X^{(j)} \cup \{A | \exists Y \rightarrow Z \in F | A \in Z \wedge Y \subseteq X^{(j)}\}$
4. *UNTIL* $(X^{(j+1)} = X^{(j)})$
5. *SET* $X^+ := X^{(j)}$

Equivalenze di dipendenze funzionali

- Siano F, G insiemi di dipendenze funzionali allora diciamo che sono **equivalenti** se

$$F^+ = G^+.$$

- La relazione di equivalenza tra insiemi di dipendenze ci permette di capire quando due schemi di relazione rappresentano gli stessi fatti: basta controllare che gli attributi siano uguali e abbiano le stesse dipendenze.
- L'algoritmo è il seguente:

Algoritmo di equivalenza

- Per ogni $Y \rightarrow Z$ in F controlliamo se essa è in G^+ calcolando Y_G^+ e controllando se $Z \subseteq Y_G^+$. Questo implica $F^+ \subseteq G^+$.
- Viceversa in maniera analoga si può controllare se $G^+ \subseteq F^+$.

Insiemi di Dipendenze Minimali

Un insieme di dipendenze funzionali F è **minimale** se:

1. Ogni lato destro di una dipendenza è un singolo attributo.
2. Per ogni dipendenza $X \rightarrow A$ in F , $F \setminus \{X \rightarrow A\}$ non è equivalente a F
3. Per ogni $X \rightarrow A$ in F e $Z \subset X$, $F \setminus \{X \rightarrow A\} \cup \{Z \rightarrow A\}$ non è equivalente a F

La regola 2 garantisce che nessuna dipendenza in F è ridondante. La regola 3 garantisce che nessun attributo in qualunque primo membro sia ridondante.

Ricoprimenti Minimali

- Dato F si dice che G è un suo **ricoprimento minimale** se G è minimale ed è equivalente a F .
- TEOREMA: *Ogni insieme di dipendenze funzionali ha un ricoprimento minimale*

Dimostrazione del teorema del ricoprimento minimale

- Dato l'insieme F costruiamo un insieme F' ad esso equivalente con la proprietà 1.
- Per garantire la proprietà 2. Basta cancellare ogni dipendenza $X \rightarrow A$ che non soddisfa 2.
- Analogamente se esiste una regola che non soddisfa la 3 si accorcia e si continua il processo che avrà termine ottenendo il ricoprimento minimale.

Si può dimostrare che basta applicare la regola 3. E solo alla fine la 2. Ma non viceversa!

Decomposizioni di uno schema, Decomposizioni che preservano i dati (loss-less join)

Prof. Alfredo Pulvirenti
Prof. Salvatore Alaimo

Decomposizione di uno schema

- Dato uno schema $R=\{A_1,A_2,...A_n\}$ una sua **decomposizione** è un insieme $d = \{R_1,R_2,...R_k\}$ di sottoinsiemi di R tali che:

$$R = R_1 \cup R_2 \cup \dots \cup R_k$$

ESEMPIO

Consideriamo lo schema

StudentiEsamiCorsi={Matricola, Nome, Indirizzo, Telefono, Corso, Professore, Voto}

L'insieme $d=\{\text{Studenti}, \text{Esami}, \text{Corsi}\}$ con

Studenti={Matricola, Nome, Indirizzo, Telefono}

Esami={Matricola, Corso, Voto}

Corsi={Corso, Professore}

è una sua decomposizione.

Preservazione dei dati

Ovviamente decomporre lo schema iniziale comporta il vantaggio di evitare ridondanze nella rappresentazione. Inoltre nell'esempio considerato *per ritrovare i dati dello schema iniziale basta considerare la giunzione naturale degli elementi della decomposizione*. Questa proprietà della decomposizione si chiama *preservazione dei dati(loss-less joins)*

Decomposizioni che preservano i dati (loss-less joins)

- Dato uno schema R con un insieme F di dipendenze funzionali, una sua decomposizione $d=\{R_1, R_2, \dots, R_k\}$ si dice che **preserva i dati** (o che ha loss-less joins) se per ogni relazione r di R che soddisfa tutte le dipendenze di F si ha:

$$r = \pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \dots \bowtie \pi_{R_k}(r)$$

Nota sulla decomposizione

Siano R, F e d come sopra.

E sia $m_d(r) = \pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \cdots \bowtie \pi_{R_k}(r)$

allora

$$r \subseteq m_d(r)$$

ma $m_d(r) \subseteq r$ non è sempre vero.

Decomposizione che non preserva i dati

Sia $R=\{A,B,C\}$, $R_1=\{A,B\}$, $R_2=\{B,C\}$, e sia
 $r=\{a_1b_1c_1, a_2b_1c_2\}$.

Allora

$$\pi_{R_1}(r) = \{a_1b_1, a_2b_1\},$$

$$\pi_{R_2}(r) = \{b_1c_1, b_1c_2\}$$

e $a_1b_1c_2$ appartiene a $m_d(r)$ ma non appartiene ad r .

Quindi la decomposizione non preserva i dati

Algoritmo per controllare se una decomposizione preserva i dati

Input:

$$R=\{A_1,A_2,...,A_n\}, F, d=\{R_1,R_2,...R_k\}$$

Output:

Yes/No se d preserva i dati.

INIZIALIZZAZIONE:

Consideriamo una matrice

$$M = \{R_1,R_2,...R_k\} \times \{A_1,A_2,...,A_n\}$$

dove nell'elemento $R_i A_j$ mettiamo a_j se A_j è in R_i

altrimenti mettiamo b_{ij} .

Passo Iterativo

- ITERAZIONE:

Applichiamo finché è possibile ogni dipendenza $X \rightarrow Y$ in F nel seguente modo: se esistono due righe di M che coincidono su X allora facciamole coincidere anche in Y :

- se ho uno dei due a_j allora cambiamo l'altro (b_{ij}) in a_j ;
- Altrimenti prendiamo uno dei due e lo facciamo uguale all'altro.

Test Finale

- Se durante il passo precedente si produce la riga $\mathbf{a_1a_2...a_n}$ allora rispondi *YES*
(*La decomposizione preserva i dati*)
- Altrimenti rispondi *NO*
(*La decomposizione non preserva i dati*)

Esempio

$R=\{A,B,C,D\}$, $F=\{A \rightarrow B\}$

$R_1=\{A,B\}$, $R_2=\{A,C,D\}$

La matrice M è

$$\begin{array}{c} A \quad B \quad C \quad D \\ R_1 \left(\begin{array}{cccc} a_1 & a_2 & b_{13} & b_{14} \end{array} \right) \\ R_2 \left(\begin{array}{cccc} a_1 & b_{22} & a_3 & a_4 \end{array} \right) \end{array}$$

Applicando $A \rightarrow B$ si ottiene

continua

$$\begin{array}{c} A \quad B \quad C \quad D \\ R_1 \left(\begin{array}{cccc} a_1 & a_2 & b_{13} & b_{14} \\ R_2 \left(\begin{array}{cccc} a_1 & a_2 & a_3 & a_4 \end{array} \right) \end{array} \right)$$

- Quindi la decomposizione conserva i dati

Esempio negativo

- $R=\{A,B,C,D\}$, $F=\{A \dashrightarrow C\}$, $R_1=\{A,B\}$, $R_2=\{A,C,D\}$

La matrice M è

$$\begin{array}{c} A \quad B \quad C \quad D \\ R_1 \left(\begin{array}{cccc} a_1 & a_2 & b_{13} & b_{14} \\ R_2 \left(\begin{array}{cccc} a_1 & b_{22} & a_3 & a_4 \end{array} \right) \end{array} \right)$$

- Applicando $A \rightarrow C$ si ottiene

continua

$$\begin{array}{cccc} A & B & C & D \\ R_1 & \left(a_1 & a_2 & a_3 \right. \\ R_2 & \left. a_1 & b_{22} & a_3 \right. \end{array} \begin{array}{c} b_{14} \\ a_4 \end{array}$$

- Non si può applicare nessun'altra dipendenza: quindi NON preserva i dati
- Infatti la relazione $r = \{a_1 a_2 a_3 b_{14}, a_1 b_{22} a_3 a_4\}$ soddisfa F
 $\pi_{R_1}(r) = \{a_1 a_2, a_1 b_{22}\}$, $\pi_{R_2}(r) = \{a_1 a_3 b_{14}, a_1 a_3 a_4\}$, l'elemento $a_1 a_2 a_3 a_4$ sta in $m_d(r)$ ma non sta in r .

Un caso particolare

Teorema

- Se $d=\{R_1, R_2\}$ è una decomposizione di R con dipendenze F . Allora d preserva i dati se e solo se soddisfa una delle due:

$$F \vdash R_1 \cap R_2 \rightarrow R_1 - R_2$$

$$F \vdash R_1 \cap R_2 \rightarrow R_2 - R_1$$

Dimostrazione:

- La tabella iniziale è

| | $R_1 \cap R_2$ | $R_1 - R_2$ | $R_2 - R_1$ |
|----|----------------|-------------|-------------|
| R1 | aa...a | aa...a | bb...b |
| R2 | aa...a | bb...b | aa...a |

Dimostrazione

- Osserviamo che si forma la riga tutta a...a se e solo se
 - $F \vdash R_1 \cap R_2 \rightarrow R_1 - R_2$ (si forma la seconda riga di a...a) oppure
 - $F \vdash R_1 \cap R_2 \rightarrow R_2 - R_1$ (si forma la prima riga di a...a).

Esempio positivo

- $R=\{A,B,C,D\}$, $F=\{A \twoheadrightarrow B\}$
- $R_1=\{A,B\}$, $R_2=\{A,C,D\}$
- $R_1 \cap R_2=\{A\}$; $R_1-R_2=\{B\}$
- Ovviamente $F \vdash A \rightarrow B$ quindi la decomposizione preserva i dati

Esempio negativo

- $R=\{A,B,C,D\}$, $F=\{A \twoheadrightarrow C\}$
- $R_1=\{A,B\}$, $R_2=\{A,C,D\}$
- $R_1 \cap R_2=\{A\}$; $R_1-R_2=\{B\}$; $R_2-R_1=\{C,D\}$
- Poiché B e D appartengono a A^+ allora non è soddisfatta né $F \vdash R_1 \cap R_2 \rightarrow R_1 - R_2$
né $F \vdash R_1 \cap R_2 \rightarrow R_2 - R_1$ quindi la decomposizione non preserva i dati.

Decomposizioni che conservano le
dipendenze funzionali

Prof. Alfredo Pulvirenti
Prof. Salvatore Alaimo

Conservazione delle Dipendenze

- La *proiezione* $\pi_Z(F)$ di F su un insieme Z di attributi è l'insieme delle dipendenze $X \rightarrow Y$ appartenenti a F^+ tali che $XY \subseteq Z$

$$\pi_Z(F) = \left\{ X \rightarrow Y \mid F \vdash X \rightarrow Y \text{ AND } XY \subseteq Z \right\}$$

Algoritmo per il calcolo della proiezione di un insieme di dipendenze

Input: $(R(A_1, A_2, \dots, A_n), F)$

Output: Una copertura della proiezione di F su $T \subseteq (A_1, A_2, \dots, A_n)$

Begin

$$\pi_T(F) = \emptyset$$

for each $Y \subset T$ **do**

$$Z = Y_F^+ - Y$$

$$\pi_T(F) = \pi_{\textcircled{T}}(F) \cup \{Y \rightarrow (Z \cap T)\}$$

end for

return $\pi_T(F)$

end

Conservazione delle Dipendenze

- Dato uno schema relazionale (R, F) ed una sua decomposizione $d = \{R_1, R_2, \dots, R_k\}$ si dice che essa **conserva le dipendenze funzionali** se F è implicata logicamente dall'unione delle proiezioni $\pi_{R_i}(F)$ di F sugli R_i .

- ES.

Sia $R(A, B, C)$ $F = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$ e $d = \{R_1(A, B), R_2(B, C)\}$ una sua decomposizione. d preserva le dipendenze?

Decomposizione che preserva i dati ma non le dipendenze funzionali

- Consideriamo lo schema $R(C,S,Z)$ che ci dice che *nella città C c'è un palazzo all'indirizzo S con codice postale Z .*
- Valgono le dipendenze $F = \{CS \rightarrow Z, Z \rightarrow C\}$.
- La decomposizione di R in (SZ, CZ) preserva i dati perché $F \vdash SZ \cap CZ = Z \rightarrow (CZ - SZ) = C$
- Tuttavia non preserva le dipendenze funzionali infatti: $\pi_{SZ}(F)$ è vuoto e $\pi_{CZ}(F) = \{Z \rightarrow C\}$ ed essi non implicano $CS \rightarrow Z$.

Algoritmo per controllare se una decomposizione preserva le dipendenze funzionali

- **Input:** $R=\{A_1, A_2, \dots, A_n\}$, F , $d=\{R_1, R_2, \dots, R_k\}$
- **Output:** Yes/No se d preserva F .
- **IDEA DELL'ALGORITMO:**
 - Sia $G = \bigcup_{i=1, \dots, k} \pi_{R_i}(F)$.
 - Dobbiamo provare che G è equivalente ad F (senza calcolare G).
 - Basta provare che per ogni $X \rightarrow Y$ in F , Y è contenuto in X^+ calcolato rispetto a G . Per questo calcolo basta fare la chiusura di X rispetto alle varie proiezioni di F sulle R_i .

(R, \neq)

FORME NORMALI

- Uno schema relazionale R con dipendenze F si dice in **Forma Normale di Boyce-Codd** (BCNF) se per ogni $X \rightarrow A$ di F^+ , se A non appartiene ad X allora
 - X è una *superchiave* di R, cioè è o contiene una chiave.
- Si può dimostrare che se tutte le dipendenze di F sono del tipo $X \rightarrow A$ (F sia minimale), allora basta verificare la suddetta proprietà solo per gli elementi di F e non di F^+ .

X^+

Esempio

- Consideriamo lo schema $R(\underline{C}, \underline{S}, \underline{Z})$ con dipendenze $CS \rightarrow Z$ e $Z \rightarrow C$.
- Le chiavi sono CS e ZS.
- Tuttavia $Z \rightarrow C$ non soddisfa la proprietà perché Z non è una superchiave, quindi **non è in BCNF**.

(R, F)

TERZA FORMA NORMALE

- Uno schema relazionale R con dipendenze F si dice in **Terza Forma Normale** (3NF) se per ogni $X \rightarrow A$ di F^+ , se A non appartiene ad X allora
 - X è una *superchiave* di R oppure A è *primo*, cioè appartiene a qualche chiave.
- Si può dimostrare che se tutte le dipendenze di F sono del tipo $X \rightarrow A$, allora basta verificare la suddetta proprietà solo per gli elementi di F e non di F^+ .

3NF Esempio

- Consideriamo lo schema $R(C, S, Z)$ con dipendenze $\underline{CS} \rightarrow Z$ e $Z \rightarrow C$. Le chiavi sono \underline{CS} e ZS .
- Abbiamo già osservato che **non è in BCNF**.
- Tuttavia essa è in **3NF** infatti, nella dipendenza $Z \rightarrow C$, Z non è una superchiave, tuttavia C è primo perché è elemento della chiave CS .

Perché la BCNF ?

- Lo scopo delle BCNF e' **eliminare ridondanze causate dalle dipendenze**.
- Infatti supponiamo che R sia in BCNF e supponiamo per assurdo che possiamo predire il valore di un attributo da qualche dipendenza (QUINDI CHE CI SIA RIDONDANZA). Quindi avremmo una situazione del tipo:

| X | Y | A |
|-------|---|---|
| ----- | | |
| x | y | a |
| x | z | ? |

- Con y **diversa** da z
- Dove possiamo dedurre che $?=a$. Questo implica che per qualche $Z \subseteq X$ vale la dipendenza $Z \twoheadrightarrow A$. Tuttavia poiché R è in BCNF allora Z è una superchiave così come X . Ma questo implicherebbe $y=z$ che contraddice l'assunzione che le due tuple sono distinte. Assurdo!

$$(\cancel{CSZ} - \cancel{CS}) \rightarrow C$$

E la 3NF ?

Consideriamo lo schema CSZ con le dipendenze $CS \rightarrow Z$ e $Z \rightarrow C$
 Che e' in 3NF ma non in BCNF. Allora da

| C | S | Z |
|----------|---|---|
| ----- | | |
| c | s | z |
| <u>c</u> | t | z |

$$R \quad Z \rightarrow C$$

$$(\cancel{CSZ} - \cancel{CS}) \rightarrow C$$

A

con s **diversa** da t. Si può dedurre che $?=c$ perché $Z \rightarrow C$.

Quindi ci possono essere ridondanze dovute a dipendenze

E' possibile avere tutto?

- E' possibile avere decomposizioni che preservano i dati o le dipendenze ed in cui tutte le componenti sono in forma normale?
 - SI per *conservazione dei dati e BCNF*
 - SI per *conservazione dei dati e delle dipendenze e 3NF*.
 - **NO** per *la conservazione delle dipendenze e BCNF*

Algoritmo per BCNF:alcuni lemmi

$R_1 \quad S_1 \ S_2 \quad \pi_{R_1}(F)$ preliminari

Lemma 1

✓ Sia R uno schema con dipendenze F e sia $d=\{R_1, R_2, \dots, R_k\}$ una decomposizione che preserva i dati rispetto a F , e sia $d'=\{S_1, S_2\}$ una decomposizione di R_1 che preserva i dati rispetto a $\pi_{R_1}(F)$. Allora la decomposizione di R , $d''=\{S_1, S_2, R_2, \dots, R_k\}$ preserva i dati rispetto a F .

$$d = \{R_1 \ R_2 \ \dots \ R_k\} \quad (*)$$

$$d' = \{S_1 \ S_2 \ R_2 \ \dots \ R_k\} \quad (k+1)$$

Algoritmo per BCNF:alcuni lemmi preliminari

Lemma 2

- a) Ogni schema R con due attributi è in BCNF
- b) Se R non è in BCNF allora esistono due attributi A,B tali che: $(R - AB) \rightarrow A$.



Algoritmo per BCNF:alcuni lemmi preliminari

Lemma 3

Dati (R, F) , se proiettiamo su $R_1 \subseteq R$ ottenendo F_1 , e successivamente proiettiamo su $R_2 \subseteq R_1$ ottenendo F_2 , allora si ha che $F_2 = \pi_{R_2}(F)$.

Decomposizioni che preservano i dati con componenti in BCNF

INPUT: Schema R e dipendenze F

OUTPUT: Decomposizione che preserva i dati tale che ogni componente sia in BCNF rispetto alla proiezione di F su quella componente.

L'idea dell'algoritmo è quella di decomporre R in due schemi:

- XA in BCNF, dove vale $X \rightarrow A$ e $R-A$, tali che $(R-A, XA)$ preserva i dati.
- Si riparte da $R-A$, si calcola la proiezione di F (tale step ha un costo esponenziale) su tale schema e si continua come sopra fino a quando ci si riduce a due soli attributi quando, per il Lemma 2, è senz'altro in BCNF.
- La decomposizione trovata non è l'unica, dipende dall'ordine con cui vengono analizzate le dipendenze

INPUT: Schema $R(T)$ e dipendenze F

OUTPUT: Decomposizione che preserva i dati tale che ogni componente sia in BCNF rispetto alla proiezione di F su quella componente.

begin

$$\rho = \{(R_1(T_1), F_1) = (R(T), F)\}, n = 1$$

while $\exists (R_i(T_i), F_i) \in \rho$ non in BCNF per $X \rightarrow A$ **do**

$$n := n + 1$$

$$T' = X^+$$

$$F' = \pi_{T'}(F_i)$$

$$T'' = T_i - (T' - X)$$

$$F'' = \pi_{T''}(F_i)$$

$$\rho = \rho - \{(R_i(T_i), F_i)\} \cup \{(R_i(T'), F'), (R_n(T''), F'')\}$$

end while

end

- Sfortunatamente non è possibile prevedere pregi e difetti delle scomposizioni che si ottengono.

Preservazione delle dipendenze e 3NF

- **Input:** R, F con F *ricoprimento minimale*
- **Output:** Una decomposizione di R che **conserva le dipendenze** e tale che ogni suo elemento e' in **3NF**.

ALGORITMO

- Se ci sono attributi *non presenti in F* essi possono essere raggruppati in un solo schema ed eliminati.
- Se una dipendenza di F coinvolge *tutti gli attributi* di R , allora ritorna (R).
- Altrimenti ritorna la decomposizione fatta da tutti gli XA tali che $X \rightarrow A$ appartiene ad F.

Osservazione e raffinamento

- In effetti se ho le dipendenze
- $X \rightarrow A_1, \dots, X \rightarrow A_n$, invece delle componenti XA_1, \dots, XA_n basta mettere la sola componente $XA_1 \dots A_n$. Essa infatti non solo conserva le dipendenze ma è anche in 3NF perché X è una chiave

Preservare dati+dipendenze+3NF

- Partiamo da una decomposizione $R = (R_1, R_2, \dots, R_k)$ che *preserva le dipendenze* e tale che ogni R_i è in 3NF rispetto alla proiezione di F su R_i . E sia X una *chiave* per R
- Allora $(R_1, R_2, \dots, R_k, X)$ preserva i dati e le dipendenze ed ogni suo elemento è in 3NF

Correttezza

- Infatti che X sia in 3NF è ovvio, una chiave ogni elemento è primo.
- Ovviamente (R_1, \dots, R_n, X) conserva le dipendenze perché questo è già vero per (R_1, \dots, R_n)
- Infine per vedere che preserva i dati basta applicare l'algoritmo della matrice e vedere che la riga di X diviene a_1, a_2, \dots, a_n .
- Infatti poiché X è una chiave $F \vdash X \rightarrow A_m$ come conseguenza delle proiezioni di F sulle componenti per ogni A_m non in X per cui b_m prima o poi diventa a_m .

Input R(T),F

Begin

$$ND = T - \bigcup_{X \rightarrow A \in F} A$$

Output tutte le chiavi di R

$$SD = \bigcup_{X \rightarrow A \in F} X \cap \bigcup_{X \rightarrow A \in F} A$$

$$Cand = [ND :: (SD)]$$

$$Keys = \phi$$

while $Cand \neq \phi$ do

$$X :: (Y) = First(Cand)$$

$$Cand = Rest(Cand)$$

if $\nexists k \subset X \mid k \in Keys$ then

if $X^+ = T$ then

$$Keys = Keys \cup X$$

else

$$A_1, \dots, A_n = Y - X^+$$

for $i = 1 \dots n$ do

$$cand = cand \cup XA_i :: (A_{i+1} \dots A_n)$$

end for

end if

end if

end while

end

Esercitazione Normalizzazione

Considerare la relazione:

| Docente | Dipartimento | Facoltà | Preside | Corso |
|----------------|---------------------|----------------|----------------|--------------|
| Verdi | Matematica | Ingegneria | Neri | Analisi |
| Verdi | Matematica | Ingegneria | Neri | Geometria |
| Rossi | Fisica | Ingegneria | Neri | Analisi |
| Rossi | Fisica | Scienze | Bruni | Analisi |
| Bruni | Fisica | Scienze | Bruni | Fisica |

individuare le proprietà della corrispondente applicazione.

Individuare inoltre eventuali ridondanze e anomalie nella relazione.

Individuare la chiave e le dipendenze funzionali della relazione

Individuare poi una decomposizione in forma normale di Boyce-Codd.

Si consideri la relazione:

| Prodotto | Componente | Tipo | Q | PC | Fornitore | PT |
|-----------------|-------------------|-------------|----------|-----------|------------------|-----------|
| Libreria | Legno | Noce | 50 | 10.000 | Forrest | 400.000 |
| Libreria | Bulloni | B212 | 200 | 100 | Bolt | 400.000 |
| Libreria | Vetro | Cristal | 3 | 5.000 | Clean | 400.000 |
| Scaffale | Legno | Mogano | 5 | 15.000 | Forrest | 300.000 |
| Scaffale | Bulloni | B212 | 250 | 100 | Bolt | 300.000 |
| Scaffale | Bulloni | B412 | 150 | 300 | Bolt | 300.000 |
| Scrivania | Legno | Noce | 10 | 8.000 | Wood | 250.000 |
| Scrivania | Maniglie | H621 | 10 | 20.000 | Bolt | 250.000 |
| Tavolo | Legno | Noce | 4 | 10.000 | Forrest | 200.000 |

Che rappresenta alcune informazioni sui prodotti di una falegnameria e i relativi componenti. Vengono indicati: il tipo del componente di un prodotto (attributo **Tipo**), la quantità del componente necessaria per un certo prodotto (attributo **Q**), il prezzo unitario del componente di un certo prodotto (attributo **PC**), il fornitore del componente (attributo **Fornitore**) e il prezzo totale del singolo prodotto (attributo **PT**).

Individuare le dipendenze funzionali e la chiave di questa relazione.

Con riferimento alla relazione:

| Prodotto | Componente | Tipo | Q | PC | Fornitore | PT |
|-----------------|-------------------|-------------|----------|-----------|------------------|-----------|
| Libreria | Legno | Noce | 50 | 10.000 | Forrest | 400.000 |
| Libreria | Bulloni | B212 | 200 | 100 | Bolt | 400.000 |
| Libreria | Vetro | Cristal | 3 | 5.000 | Clean | 400.000 |
| Scaffale | Legno | Mogano | 5 | 15.000 | Forrest | 300.000 |
| Scaffale | Bulloni | B212 | 250 | 100 | Bolt | 300.000 |
| Scaffale | Bulloni | B412 | 150 | 300 | Bolt | 300.000 |
| Scrivania | Legno | Noce | 10 | 8.000 | Wood | 250.000 |
| Scrivania | Maniglie | H621 | 10 | 20.000 | Bolt | 250.000 |
| Tavolo | Legno | Noce | 4 | 10.000 | Forrest | 200.000 |

si considerino le seguenti operazioni di aggiornamento:

Inserimento di un nuovo prodotto;

Cancellazione di un prodotto;

Aggiunta di una componente a un prodotto;

Modifica del prezzo di un prodotto.

Discutere i tipi di anomalia che possono essere causati da tali operazioni

Descrivere le ridondanze presenti e individuare una decomposizione della relazione che non presenti tali ridondanze. Fornire infine l'istanza dello schema così ottenuto, corrispondente all'istanza originale. Verificare poi che sia possibile ricostruire l'istanza originale a partire da tale istanza.

Considerare uno schema di relazione R (E, N, L, C, S, D, M, P, A), con le dipendenze

$E \rightarrow NS,$

$NL \rightarrow EMD,$

$EN \rightarrow LCD,$

$C \rightarrow S,$

$D \rightarrow M,$

$M \rightarrow D$

$EPD \rightarrow AE$

$NLCP \rightarrow A.$

Calcolare una copertura minima
per tale insieme e decomporre la relazione in terza forma normale.

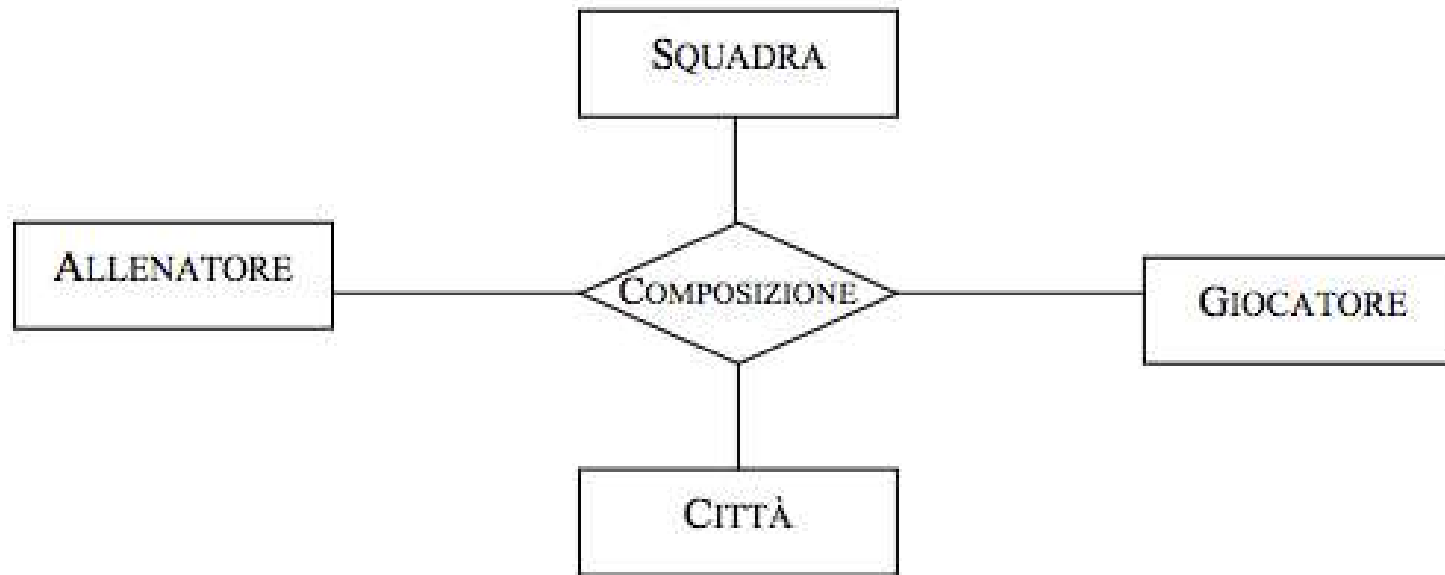
Si consideri la relazione:

| CM | Materia | CS | Sem. | CD | NomeDoc | Dipartimento |
|-----|------------|--------|------|-----|---------|--------------|
| I01 | Analisi I | Inf | I | NR1 | Neri | Matematica |
| I01 | Analisi I | El | I | NR2 | Neri | Matematica |
| I02 | Analisi II | El-Inf | I | NR1 | Neri | Matematica |
| I04 | Fisica I | El | II | BN1 | Bianchi | Fisica |
| I04 | Fisica I | Mec | I | BR1 | Bruni | Meccanica |
| I04 | Fisica I | Inf | I | BR1 | Bruni | Meccanica |
| I05 | Fisica II | El | II | BR1 | Bruni | Meccanica |
| I06 | Chimica | Tutti | I | RS1 | Rossi | Fisica |

in cui CM e CD sono, rispettivamente, abbreviazioni di CodiceMateria e CodiceDocente e l'attributo CS assume valori di tipo stringa che indicano in qualche modo il corso di studio o i corsi di studio cui un corso è destinato. Individuare la chiave (o le chiavi) e le dipendenze funzionali definite su di essa (ignorando quelle che si ritiene siano eventualmente "occasionalmente") e spiegare perché essa non soddisfa la BCNF. '

Decomporla in BCNF nel modo che si ritiene più opportuno

Si consideri lo schema Entità-Relazione:



Sui dati descritti da questo schema valgono le seguenti proprietà:

- Un giocatore può giocare per una sola squadra (o per nessuna);
- Un allenatore può allenare una sola squadra (o nessuna);
- Una squadra ha un solo allenatore, diversi giocatori e appartiene a un'unica città.

Verificare se lo schema soddisfa la forma normale di Boyce-Codd e, in caso negativo, ristrutturarlo in un nuovo schema in maniera che soddisfi tale forma normale.

Consideriamo la relazione:

| Reparto | Cognome | Nome | Indirizzo |
|----------------|----------------|-------------|------------------|
| Vendite | Rossi | Mario | Via Po 20 |
| Acquisti | Rossi | Mario | Via Po 20 |
| Bilancio | Neri | Luca | Via Taro 12 |
| Personale | Rossi | Luigi | Via Taro 12 |

e le sue seguenti possibili decomposizioni:

Reparto, Cognome in una relazione e **Cognome, Nome, Indirizzo** nell'altra;

Reparto, Cognome, Nome in una relazione e **Nome, Indirizzo** nell'altra;

Reparto, Cognome, Nome in una relazione e **Cognome, Nome, Indirizzo** nell'altra;

Individuare, con riferimento sia all'istanza specifica sia all'insieme delle istanze sullo stesso schema (con le proprietà naturalmente associate), quali di tali decomposizioni sono senza perdita.

**Materiale aggiuntivo lo trovate a
questo link:**

[**http://fondamentidibasididati.it/**](http://fondamentidibasididati.it/)