Raffinamento della progettazione di Basi di Dati

Dipendenze funzionali e Chiavi

Introduzione

- La modellazione concettuale ci fornisce un insieme di relazioni e di vincoli di integrità che costituiscono una buona base di partenza per la progettazione di una base di dati relazionale
- Tuttavia, i vincoli di integrità possono essere utilizzati per un ulteriore raffinamento dello schema relazionale
- Questo raffinamento permette di evitare alcune anomalie molto generali che possono rendere inconsistente o difficile da mantenere la nostra base di dati

Possibili anomalie della base di dati

- Immaginiamo di avere la seguente tabella:

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

Esempio

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

- Lo stipendio per ciascun impiegato è unico ed è funzione del solo impiegato, indipendentemente dai progetti cui partecipa
- Il bilancio per ciascun progetto è unico ed è funzione del solo progetto, indipendentemente dagli impiegati che vi partecipano.
- Ogni impiegato, in ciascun progetto cui partecipa, svolge una sola funzione, eventualmente diversa da progetto a progetto

Esempio

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

Questa relazione può causare «anomalie»:

- **Ridondanza**: lo stipendio di un impiegato si ripete in ogni tupla in cui compare l’impiegato
  - **Aggiornamento**: Se cambia lo stipendio di un impiegato, devo aggiornare tutte le tuple in cui compare quell’impiegato
- **Inserimento**: non posso inserire un nuovo impiegato senza un progetto a cui è assegnato
- **Cancellazione**: se un impiegato non partecipa più ad alcun progetto devo cancellarlo dalla tabella (!!)

Esempio

Anomalia di cancellazione: Se un impiegato interrompe la partecipazione a tutti i progetti senza lasciare l'azienda, dobbiamo cancellarlo definitivamente, a meno di ammettere valori nulli sulla chiave *Progetto*, il che è **inammissibile**.

Anomalia di inserimento: Un nuovo impiegato senza progetto non può essere inserito.

<u>Impiegato</u>	Stipendio	<u>Progetto</u>	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

Ridondanza: Lo stipendio di ciascun impiegato è ripetuto in tutte le tuple relative.

Anomalia di aggiornamento: Se lo stipendio di un impiegato varia, è necessario andarne a modificare il valore in diverse tuple.

Causa delle anomalie

- L’origine delle anomalie è che abbiamo informazioni diverse in un'unica relazione:
  - gli impiegati e i loro stipendi;
  - i progetti con i relativi bilanci;
  - Il legame tra i progetti e gli impiegati (e il loro ruolo in ogni progetto)
- Per studiare in maniera sistematica questi aspetti, introduciamo il concetto di dipendenza funzionale (DF)
  - Una DF descrive legami di tipo funzionale tra gli attributi di una relazione

Esempio

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

- Lo **Stipendio** di un **Impiegato** è unico, indipendentemente dai progetti a cui collabora o dalla funzione nel progetto
  - Di conseguenza, il valore dell’attributo **Impiegato** *determina funzionalmente* il valore dell’attributo **Stipendio** (in altri termini, dato il valore di impiegato, il valore di stipendio è definito)
- Lo **Bilancio** di un **Progetto** è unico, indipendentemente dai progetti a cui collabora o dalla funzione nel progetto
  - Di conseguenza, il valore dell’attributo **Progetto** *determina funzionalmente* il valore dell’attributo **Bilancio**

Definizione formale di DF

Il concetto di dipendenza funzionale può essere formalizzato come segue:

- sia **r** una relazione sullo schema R(X)
- Siano Y e Z due sottoinsiemi non vuoti di X
- Z dipende funzionalmente da Y ($Y \rightarrow Z$) se per ogni coppia di tuple t_1 e t_2 di **r** vale che:

$$\text{se } t_1[Y]=t_2[Y], \text{ allora } t_1[Z]=t_2[Z]$$

[Ovvero se t_1 e t_2 hanno gli stessi valori su Y, allora hanno gli stessi valori anche su Z]

Come si identificano le DF?

- In generale, sono derivate da vincoli del mondo reale sul significato degli attributi di una relazione R
- Da notare:
  - data un’istanza di una relazione, è possibile solo inferire che una certa DF *potrebbe sussistere* tra certi attributi
  - viceversa, se si verifica la presenza di tuple che violano una certa DF, quello che si può inferire con certezza è che una certa DF non sussiste

Esempio

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

- Prendiamo la tabella popolata qui sopra:
  - da un lato, possiamo inferire che la dipendenza funzionale **Text → Course** *potrebbe sussistere*, poiché non ci sono tuple che con identici valori di Text abbiamo diversi valori di Course (ma nulla esclude che potrebbero esserci in un'altra istanza della relazione!)
  - dall'altro, possiamo concludere con assoluta certezza che le DF **Teacher → Course** , **Teacher → Text** e **Course → Text** *non possono sussistere*, dato che ci sono tuple con valori identici per l'attributo a sinistra della DF e valori diversi per l'attributo a destra

Esercizio

- Quali DF potrebbero sussistere per la seguente istanza di relazione?
- Quali invece non possono sussistere con certezza?

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

DF banali

- Prendiamo la possibile DF:

  Impiegato, Stipendio → Stipendio

- Essa è sicuramente verificata, in quanto il valore di Stipendio nella stessa tupla non può mai essere diverso da se stesso!
- Si dice quindi che si tratta di una DF *banale* (è sempre soddisfatta)
  - Una DF Y → A è banale se A appartiene a Y
  - Una DF Y → Z è non banale se nessun attributo in Z appartiene a Y
- Essendo le DF banali sempre soddisfatte, d'ora in poi ci riferiremo sempre e soltanto a DF non banali.

DF e chiavi

- Se prendiamo una chiave K di una relazione R si può facilmente verificare che esiste una DF tra K e ogni altro attributo dello schema di R:
  - per definizione, in R non possono esistere due tuple con lo stesso valore su K, e quindi una DF che ha K al primo membro sarà sempre soddisfatta.
  - perciò, esisterà sempre una DF tra una chiave di una relazione e tutti gli attributi dello schema della relazione(esclusi quelli della chiave stessa)
- Per questo si dice che **il concetto di DF generalizza quello di vincolo di chiave**
- Più precisamente, una DF $Y \rightarrow Z$ su uno schema R(X) **degenera in un vincolo di chiave** se Y U Z = X. In tal caso, infatti, Y è (super)chiave per lo schema R(X).

Esercizio

Quali DF *non banali* possiamo derivare dal fatto che gli attributi **Impiegato, Progetto** formano la chiave di questa tabella?

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

DF e anomalie

- Abbiamo detto che
  - **Impiegato □ Stipendio**
  - **Progetto □ Bilancio**
  - **Impiegato, Progetto □ Funzione**

  sono intuitivamente DF della Tabella esempio
- Tuttavia, tutte le anomalie discusse in precedenza appaiono legate solo alle prime due DF e non alla terza.
- Come possiamo spiegare questa differenza?

DF e anomalie

- Osservazione:
  - **Impiegato → Stipendio** e **Progetto → Bilancio** non contengono una chiave nella parte sinistra
  - **Impiegato, Progetto → Funzione** invece sì (la coppia di attributi <Impiegato, Progetto> è una chiave)
- Una DF che contiene una chiave nella sua parte sinistra per definizione non può generare ridondanze,
- Ovvero non è possibile che alla coppia <Impiegato, Progetto> corrispondano funzioni diverse (altrimenti <Impiegato, Progetto> non sarebbe una chiave!)
- **Le anomalie sono perciò causate solo dalle DF Y → Z tali che Y non contiene una chiave**

Un po' di teoria delle DF

- Prima di vedere come risolvere in modo generale i problemi appena discussi, è necessario introdurre alcuni concetti e algoritmi generali su DF e chiavi di una relazione:
  - chiusura di un insieme di dipendenze funzionali
  - chiusura di un insieme di attributi
  - individuazione delle superchiavi di una relazione
  - copertura minimale per un insieme di dipendenze funzionali
- Questo ci fornirà gli strumenti teorici per affrontare la normalizzazione delle relazioni

Chiusura di un insieme di dipendenze funzionali

- Dato un insieme *F* di dipendenze funzionali (DF), è possibile calcolare altre DF che sono logicamente implicate da *F*.
  - Esempio: se $\mathbf{X} \rightarrow \mathbf{Y}$ e $\mathbf{Y} \rightarrow \mathbf{Z}$, allora $\mathbf{X} \rightarrow \mathbf{Z}$
- Un insieme *F* di DF **implica logicamente** una DF $\mathbf{X} \rightarrow \mathbf{Y}$ se ogni istanza che soddisfa F soddisfa anche $\mathbf{X} \rightarrow \mathbf{Y}$
- L'insieme di **tutte** le DF logicamente implicate da *F* si chiama la **chiusura** di *F* e viene indicate con la notazione F^+

Problema: come facciamo a calcolare la chiusura di un insieme di dipendenze funzionali *F*?

Assiomi di Armstrong

- F^+ può essere calcolato mediante gli **Assiomi di Armstrong (AA):**
  - **Assioma di riflessività:** se $Y \subseteq X$, allora $X \rightarrow Y$
  - **Assioma di aumento:** se $X \rightarrow Y$, allora $XZ \rightarrow YZ$
  - **Assioma di transitività:** se $X \rightarrow Y$ e $Y \rightarrow Z$, allora $X \rightarrow Z$

- Dagli AA si possono ottenere altre due regole derivate:
  - **Unione**: Se $X \rightarrow Y$ e $X \rightarrow Z$, allora $X \rightarrow YZ$
  - **Scomposizione**: Se $X \rightarrow YZ$, allora $X \rightarrow Y$ e $X \rightarrow Z$

Assiomi di Armstrong: correttezza e completezza

- Gli Assiomi di Armstrong (AA) sono:
  - **Corretti (sound)**: se gli AA generano una nuova dipendenza funzionale *f* a partire da un insieme *F*, allora *f* è logicamente implicata da *F*
  - **Completi (complete)**: se una dipendenza funzionale *f* è logicamente implicata da F, allora gli AA sono in grado di generarla

Questo equivale a dire che gli AA sono in grado di generare **TUTTE e SOLE** le dipendenze funzionali che sono logicamente implicate da *F*

Esempio di applicazione degli AA

- Assumiamo di avere una relazione R con l'insieme di DF:

  R = (A, B, C, G, H, I)

  F = {$A \rightarrow B$, $A \rightarrow C$, $CG \rightarrow H$, $CG \rightarrow I$, $B \rightarrow H$}

- Alcuni membri di F+ sono:
  - $A \rightarrow H$
  - $AG \rightarrow I$
  - $CG \rightarrow HI$

Esempio di applicazione degli AA

- Assumiamo di avere una relazione R con l'insieme di DF:

  $R = (A, B, C, G, H, I)$

  $F = \{A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H\}$
- Alcuni membri di F+ sono:
  - $A \rightarrow H$
    - per transitivita' da $A \rightarrow B$ e $B \rightarrow H$
  - $AG \rightarrow I$
    - arricchendo $A \rightarrow C$ con G e poi utilizzando la transitività con $CG \rightarrow I$
  - $CG \rightarrow HI$
    - arricchimento di $CG \rightarrow I$ con CG per ottenere $CG \rightarrow CGI$
    - arricchimento di $CG \rightarrow H$ con I per ottenere $CGI \rightarrow HI$
    - infine applicazione della transitività

Chiusura di un insieme di attributi

- Algoritmo calcolare la chiusura di un insieme di attributi X:
  - Si ponga $(X)^+$ = X
  - Se esiste una DF Z -> W in F dove Z $\subseteq (X)^+$, si aggiunga W a $(X)^+$
  - Ripetere la procedura finché non è possibile aggiungere più nulla a $(X)^+$

Chiusura di un insieme di attributi - Esempio

- Sia F = {A -> B, B -> C, AC -> D }
- Qual è la chiusura dell'attributo A (ovvero $(A)^+$)?
- Applichiamo l'algoritmo di chiusura:
  - Poniamo $(X)^+$ **= A**
  - Deriviamo **B** (applicando la DF **A -> B**)
  - Poniamo $(X)^+$ **= AB**
  - Deriviamo **C** (applicando la DF **B -> C**)
  - Poniamo $(X)^+$ = **ABC**
  - Deriviamo **D** (applicando la DF **AC -> D**)
  - Poniamo $(X)^+$ = **ABCD**
- La chiusura di A (denotata da $(A)^+$) è l'insieme di attributi ABCD

Copertura minimale di F

- Il calcolo della chiusura di un insieme F di DF può essere estremamente dispendioso, dato che potrebbe richiedere di derivare un numero esponenziale di DF

- Per ovviare a tale problema, è possibile introdurre una versione più semplice (ma logicamente equivalente) di un insieme F di DF chiamata **copertura minimale**

- Una **copertura minimale** di F è un insieme di DF F_{min} tale che:
  - $F_{min}^{+} = F^{+}$ (ovvero, i due insiemi sono logicamente equivalenti, dato che la loro chiusura è identica)
  - Tutte le DF in F_{min} sono della forma X -> A
  - Se rimuoviamo una DF o un attributo dal lato sinistro di una DF in F_{min} , allora F_{min} non è più equivalente a F, ovvero $F_{min}^{+} \neq F^{+}$

NB: In generale, potrebbe esserci più di una copertura minimale per F!

- In altri termini, F_{min} contiene una quantità minima di informazione per descrivere tutte le DF logicamente implicate da F

[E' possibile dimostrare che uno schema <R(**U**), F> è in BCNF sse le condizioni per essere in BCNF sono soddisfatte da F_{min}. Per questa ragione possiamo limitarci a considerare F_{min}]

Calcolo della Copertura minimale

- Per calcolare la Copertura minimale di una relazione **R** è sufficiente applicare i seguenti passi:
  - **Step 1**: ogni DF con più di un attributo sul lato destro (**X -> ABC…**) va normalizzata (messa in **forma canonica**):
    - X -> ABC… viene trasformata in un insieme di DF della forma:
      - **X -> A**
      - **X -> B**
      - **X -> C**
      - **…**

Calcolo della Copertura minimale

-
  - **Step 2 (calcolo degli attributi estranei o ridondanti)**: dato in insieme M di DF:
    - prendiamo una DF della forma **XA -> B**
    - verifichiamo se $\mathrm{B} \in (X)^{+}$ in M:
      - se sì, allora **A** è ridondante e può essere rimossa dal lato sinistro della DF
        - otteniamo M' rimuovendo da M la dipendenza XA → B e sostituendola con la dipendenza X → B
      - altrimenti **A** non può essere rimossa

Calcollo della Copertura minimale

-
  - **Step 3 (eliminazione delle DF ridondanti)**: sia M un insieme di DF per una relazione R.
    - prendiamo una DF della forma **X -> A**
    - otteniamo M' rimuovendo da M la dipendenza **X → A**
      - se $\mathbf{A} \in (X)^{+}$ usando M', allora **X -> A** è ridondante e può essere rimossa
      - altrimenti **X → A** non è ridondante

Copertura minimale - Esempio

Sia < R(A,B,C,D,E), F>, con F = { **A -> BCE**, **CDB -> A**, **CD -> E**, **E -> B** }

- Normalizzazione:
  - Da **A -> BCE** ricaviamo **A -> B**, **A -> C**, **A -> E**,
  - Otteniamo l'insieme { **A -> B**, **A -> C**, **A -> E**, **CDB -> A**, **CD -> E**, **E -> B** }

- Rimuoviamo gli attributi ridondanti dal lato sinistro delle DF:
  - $(CD)^+$ = CDAE**B**, per cui B è ridondante
  - $(C)^+ = C$ (niente da fare)
  - $(D)^+ = D$ (niente da fare)

- Rimuoviamo le DF ridondanti:
  - Dall'attributo A è possibile derivare B senza usare **A -> B**
  - Dall'insieme CD posso ricavare E senza usare **CD -> E**
  - Quindi possiamo rimuovere **A -> B** e **CD -> E**

$F_{min} = \boldsymbol{A \rightarrow C, A \rightarrow E, CD \rightarrow A, E \rightarrow B}$

Individuazione delle superchiavi di una relazione

- Come primo passo, possiamo usare la chiusura di un insieme di attributi X per verificare se X è una superchiave di una relazione R

- Metodo:
  - Partiamo da una relazione R con attributi **U** e un insieme di dipendenze funzionali F:

    < R(**U**), F >

  - Dato un insieme di attributi X $\subseteq$ **U**, calcoliamo $(X)^+$ e verifichiamo se $(X)^+$= **U**, ovvero se vale che

    X -> U

  - Se vale, X è una superchiave per < R(**U**), F >, altrimenti non lo è

- NB: l'insieme di attributi X potrebbe non essere una chiave minimale!!

Esempio

- Sia < R(A,B,C,D), F >, dove F = {**A -> B**, **B -> C**, **AC -> D** }
  - $(A)^{+}$= ABCD

    A è una superchiave (di fatto una chiave, visto che è costituito da un solo attributo)
  - $(AC)^{+}$ = ACBD

    AC è una superchiave, ma non è una chiave (non è minimale)
  - $(B)^{+}$ = BC

    B non è una superchiave (e *a fortiori* nemmeno una chiave)

Algoritmo per l’individuazione delle superchiavi

- **Step 1**: determinare l’insieme *I* degli attributi di **R(U)** che non sono né sul lato sinistro né sul lato destro di una DF (attributi *Isolati*)
- **Step 2**: determinare l’insieme *L* degli attributi che sono solo sul lato sinistro di una qualsiasi DF (attributi *Left*)
- **Step 3**: determinare l’insieme *R* degli attributi che sono solo sul lato destro di una qualsiasi DF (attributi *Right*)
- **Step 4**: combinare gli attributi *Isolati* e *Left* (Step 1 e Step 2)
- **Step 5**: calcolare la chiusura degli attributi ottenuti nello Step 4
  - se la chiusura coincide con **U**, allora gli attributi ottenuti nello Step 4 costituiscono la sola chiave candidata
  - l’algoritmo termina
- **Step 6**: altrimenti, individuare gli attributi che non sono stati considerati negli Step 3 e 4 (che quindi sono solo a destra o su entrambi i lati)
- **Step 7**: partendo dalla chiusura degli attributi ottenuti nello Step 4 (se ce ne sono), combinarli uno alla volta con tutti i sottoinsiemi degli attributi ottenuti nello Step 6 e determinare le chiusure di attributi che sono uguali a **U**

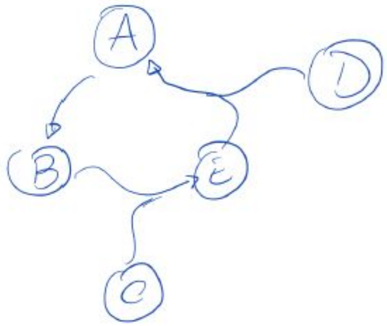
Esempio

- Sia < R(A,B,C,D,E), F >, dove F = {**A -> B, BC -> E, ED ->A** }
- Step 1 (*Isolati*): { }
- Step 2 (*Left*): { C, D }
- Step 3 (*Right*): { }
- Step 4 (unione di *I* e *L*): { C, D }
- Step 5 (chiusura): $(CD)^{+}$= CD (non è una chiave)
- Step 6: { A, B, E }
- Step 7: $(CDA)^{+}$ = ABCDE (chiave candidata)
  $(CDB)^{+}$ = ABCDE (chiave candidata)
  $(CDE)^{+}$ = ABCDE (chiave candidata)

- Aggiungere altri attributi produrrebbe solo superchiavi, per cui l'algoritmo termina
- ACD, BCD e CDE sono le sole chiavi candidate

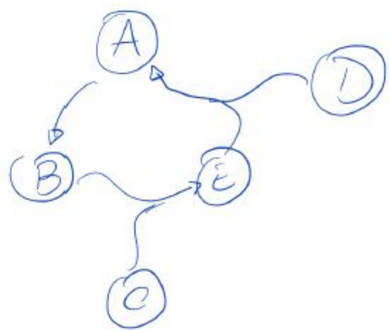
Grafo delle dipendenze

- Per calcolare le chiavi candidate è possibile usare come aiuto il **grafo delle dipendenze**
- Per l'esempio precedente, il grafo è il seguente:



- Le linee con le frecce rappresentano le dipendenze della relazione
- Seguendo le linee, è possibile stabilire se da un certo insieme di attributi posso raggiungere altri attributi

Grafo delle dipendenze



- Osservazione: se un nodo non ha nodi entranti (quindi anche se è isolato), allora deve necessariamente appartenere a ogni chiave candidata (cfr. **Step 4** dell'algoritmo)
  - Quindi C e D devono appartenere a ogni possibile chiave!
- Se considero solo CD, non posso raggiungere alcun altro nodo(**Step 5**)
- Se a C e D aggiungo A, allora da A raggiungo B, da BC raggiungo E. Quindi CDA è una chiave candidata
  - Analogamente se considero gli insiemi CDE e CDB
- Quindi CDA, CDB e CDE sono le tre chiavi candidate di R (**Step 6 e 7**)

Riflessione

Usando il grafo delle dipendenze, è più facile capire a cosa corrispondano i passi dell'algoritmo per il calcolo delle chiavi:

- Cosa rappresentano gli attributi che stanno nell'insieme *Left* o *Isolati*?
- Cosa rappresentano gli attributi che stanno nell'insieme *Right*?
- Perché l'algoritmo inizia considerando la chiusura dell'unione di *Left* e *Isolati*?
- Perché negli Step 6 e 7 vengono considerate tutte le combinazioni di attributi non considerati prima con la chiusura di quelli che appartengono a *Left* e *Isolati*?