SAPIENZA
UNIVERSITÀ DI ROMA

Corso di Laurea in Ingegneria Informatica

I Vincoli di Dipendenza Funzionale nel Modello Relazionale

Logica e Informatica

Professore:
Maurizio Lenzerini

Studente:
Gabriele Onorato
1979543

Anno Accademico 2023/2024

Indice

1 Introduzione

Questo progetto nasce con lo scopo di esaminare l'utilizzo della Logica nelle Basi di Dati, in particolare nel modello relazionale, a tale motivo si visionerà il concetto di vincolo di "Dipendenza Funzionale" nelle relazioni, e come esso viene applicato negli schemi di relazioni e nella progettazione dello schema di una base di dati.

Partiremo con l'obiettivo di analizzare un problema concreto, da esso svilupperemo progressivamente i vincoli di dipendenza funzionali e i concetti associati, tra cui un sistema di deduzione e vari algoritmi, fino ad ottenere poi la definizione di Chiave in logica, così come le forme normali.

1.1 Uno schema di esempio

Prendiamo una relazione di esempio, cioè quella usata in [1]:

STUDENTE	DIPARTIMENTO	SUPERVISORE
Clelia	DIAG	Cristian
Amin	DIAG	Cristian
Andrea	DIET	Leonardo
Pietro	DIET	Leonardo
Elisabetta	DIAG	Cristian
Federico	DIET	Leonardo

I problemi evidenziati sono i seguenti:

- Ridondanza: I supervisori Cristian e Leonardo sono ripetuti più volte.
- Inconsistenza: Se il supervisore di Elisabetta cambiasse, vuol dire che il dipartimento ha due supervisori, o è cambiato supervisore?

Molti di questi problemi vengono risolti in un'ottima progettazione concettuale e logica a partire dai requisiti, il nostro scopo è, tramite le *dipendenze funzionali* formalizzare nel modello relazionale le intenzioni esplicate dal modello entità-relazione.

1.2 La dipendenza funzionale

Come visto in 1.1, ci è possibile suppore che per ogni tupla della relazione, il valore che essa assume in DIPARTIMENTO determini quello di SUPERVISORE, come una tipologia di vincolo semantico sui dati per le relazioni legali.

Pertanto affermeremo che esista una *dipendenza funzionale* tra DIPARTIMENTO e SUPERVISORE che scriveremo come DIPARTIMENTO $\longrightarrow$ SUPERVISORE, ovvero il valore di DIPARTIMENTO determina il valore di SUPERVISORE.

Definizione 1.1 (Dipendenza Funzionale)**.** Sia R uno schema di relazione, siano X, Y degli insiemi di attributi tali che $X \cup Y \subseteq R$, sia I una qualsiasi relazione legale senza valori nulli. Si dirà che $X \longrightarrow Y$ è una *dipendenza funzionale* (FD) su uno schema R quando per ogni tupla $u, v \in I$ si ha $u[X] = v[X] \Longrightarrow u[Y] = v[Y]$.

Possiamo poi affermare le seguenti proprietà:

1. Dato un insieme Σ di FD, diremo che I soddisfa Σ se I soddisfa tutte le FD presenti in Σ .
2. Una qualsiasi FD $X \longrightarrow Y$ implica che un'altra qualsiasi FD $X \longrightarrow W$ è vera quando $Y \supset W$.
3. Due insiemi di FD sono *equivalenti* $\Sigma \equiv \Delta$ se sono soddisfatti dallo stesso insieme di relazioni legali.
4. Un insieme di FD Σ è *ridondante* se esiste $\Delta \subset \Sigma$ tale che $\Sigma \equiv \Delta$.

Risciviamo ora usando la logica le proprietà enunciate:

Definizione 1.2 (Soddisfacibilità ($\vDash$) delle FD)**.** Dato un insieme Σ di FD, $I \vDash \Sigma$ se $\forall \sigma \in \Sigma$ si ha $I \vDash \sigma$.

Definizione 1.3 (Equivalenza ($\equiv$) delle FD)**.** $\Delta \equiv \Sigma$ se e solo se $\Delta \vDash \sigma,\ \forall \sigma \in \Sigma$ e $\Sigma \vDash \delta,\ \forall \delta \in \Delta$.

Definizione 1.4 (Ridondanza)**.** Σ è *ridondante* se esiste una FD $\sigma \in \Sigma$ tale che $\Sigma - \{\sigma\} \vDash \sigma$.

Definizione 1.5 (Indipendenza)**.** Un insieme di FD Σ è *indipendente* se **non** è ridondante, cioè $\neg\exists \sigma \in \Sigma$ tale che $\Sigma - \{\sigma\} \vDash \sigma$.

Vediamo ora come si relazionano invece le dipendenze funzionali con l'algebra relazionale, in particolare:

Teorema 1.1. *Sia I una qualsiasi relazione su R, una FD $X \longrightarrow Y$ dove $X \cup Y \subset R$ è soddisfatta da I se e solo se $\pi_{XY}(I)$ soddisfa $X \longrightarrow Y$.*

È stato provato che la FD possono essere interpretate come formule proposizionali, basta riscrivere una FD $A_1...A_k \longrightarrow B$ come una formula di Horn, e quindi possiamo valutare la soddisfacibilità delle FD in tempo polinomiale.

In ambito generale bisogna anche considerare l'esistenza di relazioni infinite, sebbene in pratica si abbia a che fare solo con relazioni finite, comunque sia *L'implicazione* $\models$ *e l'implicazione finita* $\models_f$ *coincidono per le FD.*

1.3 La riduzione delle FD

Abbiamo visto che la Unit-Propagation ci permette di risolvere in tempo polinomiale il problema dell'implicazione. Ci domandiamo ora, esiste una procedura la quale data una FD $X \longrightarrow Y$, ne trovi una equivalente con il numero minimo di attributi?

Definizione 1.6 (Reduced)**.** Una FD $X \longrightarrow Y$ è *reduced* se $\neg\exists W \subset X$ tale che $\Sigma \vDash W \longrightarrow Y$. Inoltre diremo che un insieme Σ di FD è *reduced* se tutte le FD al suo interno lo sono.

Illustriamo ora un'algoritmo che dato un insieme Σ di FD, ne restituisce uno Δ con tutte le FD *reduced*:

Algorithm 1: REDUCED

Data: $\Sigma \leftarrow$ un insieme di FD in input.
$\Delta \leftarrow \Sigma$
for *each FD* $X \longrightarrow Y \in \Delta$ **do**
  for *each attribute* $A \in X$ **do**
    if $\Delta \vDash X - A \longrightarrow Y$ **then**
      $X \leftarrow X - A$ dove $X \longrightarrow Y$
  end
end
return Δ

Questo algoritmo ha quindi generato un insieme Δ di FD dove ogni componente di esso è *reduced*, dimostriamolo nel seguente modo:

Teorema 1.2. *Dato un insieme Σ di FD finito su una relazione R finita, l'algoritmo 1 termina e ritorna un insieme Δ di FD* ***reduced***

Dimostrazione. Premessa, se Σ non fosse finito, il primo ciclo *for* itererebbe indefinivamente, inoltre il secondo ciclo *for* itererebbe indefinivamente se X avesse infiniti attributi, ma se ci restringiamo a relazioni finite ciò non è possibile.

Per ogni FD scorriamo la sua lista di attributi, e proviamo a rimuoverne uno, se Δ comunque soddisfa questa FD, allora quell'attributo non è cruciale per la FD, e può essere rimosso, aggiornando l'FD corrispondente in Δ.

Ciò date le ipotesi ha chiaramente terminazione finita e ritorna grazie all'implicazione logica un'insieme di FD equivamente a quello dato. □

Per valutare il costo di quest'algoritmo partiamo dal cercare l'operazione dominante, che risulta ovviamente essere l'implicazione logica. che in questo caso particolare ha costo polinomiale, possiamo quindi affermare che il costo è polinomiale rispetto all'input.

2 Deduzione sulle FD

2.1 Un sistema formale di Prova per le FD

Un sistema formale è stato costruito per le FD, e pertanto ci è possibile dire che esse sono una teoria finita assiomatizzabile, adesso illustreremo il sistema che fu primo tra tutti studiato da W.W. Armstrong nel 1974.

Questo sistema formale di prova è formato da un assioma e quattro regole di inferenza, di cui tre essenziali, così riportate:

FDA0 o (FD1): (Riflessività) $\vdash X \longrightarrow X$

FDA1 o (FD2): (Transitività) $X \longrightarrow Y, Y \longrightarrow Z \vdash X \longrightarrow Z$

FDA2: $X \longrightarrow Y \vdash W \longrightarrow Z$ se $X \subseteq W$ e $Z \subseteq Y$

FD3: $X \longrightarrow Y \vdash X \cup Z \longrightarrow Y \cup Z$

FDA4: $X \longrightarrow Y, Z \longrightarrow W \vdash X \cup Z \longrightarrow Y \cup W$

Lemma 2.1. *FD2 non è più una regola di inferenza se sono presenti valori nulli nelle relazioni.*

Prendiamo la seguente relazione di esempio, dove $A \longrightarrow B$ e $B \longrightarrow C$, e di conseguenza FD2 inferenzia $A \longrightarrow C$:

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

Sostituiamo la colonna B con $NULL$ se interpretassimo i valori nulli come mancanza di informazione, nella seguente relazione FD2 non è valida:

A	B	C
a_1	NULL	c_1
a_2	NULL	c_2

Teorema 2.2. *Un sistema formale per le **FD** formato da FD1, FD2, FD3, è **sound** e **complete***

Dimostrazione. Ricordiamo che un sistema formale si dice **sound** quando se $\Sigma \vdash \sigma$ allora $\Sigma \vDash \sigma$, e si dice **complete** quando se $\Sigma \vDash \sigma$ allora $\Sigma \vdash \sigma$.
Sound:
FDA0 è triviale, dimostriamo che FDA1 sia sound, usando la definizione di FD date

le tuple $u, v \in I$ relazione su R contenente X, Y, Z, abbiamo che se $u[X] = v[X]$ allora $u[Y] = v[Y]$ ma allora anche $u[Z] = v[Z]$ e quindi è sound.

Dimostriamo che FD3 sia sound, supponiamo I soddisfi la FD $X \longrightarrow Y$ allora se $u[X \cup Z] = v[X \cup Z]$ ovviamente coincidono $u[X] = v[X]$ e $u[Z] = v[Z]$, data la FD abbiamo $u[Y] = v[Y]$ e di conseguenza $u[Z \cup Y] = v[Z \cup Y]$.
Complete:
Molto più complessa, ma visionabile in [1]. □

2.2 Un algoritmo per il problema dell'implicazione delle FD

Possiamo ora costruire un algoritmo polinomiale per l'implicazione delle FD, basato sulla costruzione efficiente della chiusura X^+ rispetto a Σ (anche chiamata $\Sigma-$Closure(X) o X^+_Σ) o più semplicemente $cl_\Sigma(X)$.

Lemma 2.3. *Sia Σ un insieme di FD soddisfatto da tutte le relazioni legali su R, siano $X, Y \in R$ allora è vero che:*

$$\Sigma \vDash X \longrightarrow Y \Longleftrightarrow Y \subseteq cl_\Sigma(X)$$

Pertanto per vedere se una FD $X \longrightarrow Y$ è implicata da Σ ci basta costruire $cl_\Sigma(X)$.

Definizione 2.1 (Closed)**.** Una FD $X \longrightarrow Y$ è *closed* se $Y = cl_\Sigma(X)$ e un insieme Δ di FD è *closed* se tutte le FD lo sono.

Definizione 2.2 (Forma Canonica)**.** Una FD $X \longrightarrow Y$ è in *forma canonica* se $|Y| = 1$.

Teorema 2.4. *Ogni FD $X \longrightarrow Y$ può essere convertita in forma canonica.*

Dimostrazione. Sia $Y = A_1, ..., A_k$, allora certamente $X \longrightarrow Y \vDash X \longrightarrow A_j$, e l'insieme $\{\forall j : X \longrightarrow A_j\} \vDash X \longrightarrow Y$. □

Calcoliamo ora la chiusura con il seguente algoritmo:

Algorithm 2: CLOSURE

Data: $\Sigma \leftarrow$ un insieme di FD in input.
Data: $X \leftarrow$ un insieme di attributi in input
$Y \leftarrow X$
while *esiste una FD $S \longrightarrow T$ tale che $S \subseteq Y$ e $T \not\subseteq Y$* **do**
  $Y \leftarrow Y \cup T$
end
return Y

Teorema 2.5. *L'algoritmo 2 restituisce correttamente $cl_\Sigma(X)$.*

Dimostrazione. L'algoritmo termina per un numero finito di FD e si interrompe quando $Y \subseteq cl_\Sigma(X)$.
Dimostriamo ora che è impossibile che l'algoritmo ritorni $Y \subset cl_\Sigma(X)$ per far ciò supponiamo $B \in cl_\Sigma(X)$ ma $B \notin Y$, per la prima affermazione però abbiamo che $\Sigma \vDash X \longrightarrow B$

Sia ora I una relazione legale, e u, v due tuple tali che $u[A] = v[A]$, $A \in Y$ e sia $S \longrightarrow T$ una FD violata, e quindi avremmo avuto $S \subseteq Y$ e $T \not\subseteq Y$ ma in quel caso ci saremmo trovati in una esecuzione parziale dell'algoritmo. Di conseguenza $\Sigma \nvDash X \longrightarrow B$ che è una contraddizione. □

2.3 La non ridondanza per le FD

Definizione 2.3 (Cover tra insiemi di FD)**.** Dati due insiemi Σ e Δ di FD, una è una *cover* per l'altra se $\Delta \equiv \Sigma$ (come in 1.3).

Definizione 2.4 (Cover minima)**.** Sia Δ una *cover* dell'insieme Σ di FD, Δ sarà una *cover minima* se contiene il numero minimo di FD.

Illustriamo ora un algoritmo che a partire da un insieme, ne trovi un *cover*.

Algorithm 3: NONREDUND

Data: $\Sigma \leftarrow$ un insieme di FD in input.
$\Delta \leftarrow \Sigma$
for *each FD* $\sigma \in \Delta$ **do**
  if $\Delta - \{\sigma\} \vDash \sigma$ **then**
    $\Delta \leftarrow \Delta - \{\sigma\}$
end
return Δ

Quest'algoritmo non trova necessariamente *cover minimi*, ce ne è possibile costruire uno usando questo teorema:

Teorema 2.6. *Sia* Δ *un cover non ridondante per* Σ *allora se* Δ *è closed è anche minimo.*

Algorithm 4: MINCOVER

Data: $\Sigma \leftarrow$ un insieme di FD in input.
$\Delta \leftarrow \Sigma$
for *each FD* $\sigma = X \longrightarrow Y \in \Sigma$ **do**
  $\Delta \leftarrow \Delta - \{\sigma\}$
  if $Y \not\subseteq CLOSURE(\Delta, X)$ **then**
    $Z \leftarrow CLOSURE(\Delta, Y)$
    $\Delta \leftarrow \Delta \cup \{X \longrightarrow Z\}$
end
return Δ

Dove a ogni rimozione verifichiamo che la dipendenza rimossa Y si trovi nella chiusura, e in caso negativo la calcoliamo e l'aggiungiamo come nuova FD a Δ. Il costo dell'algoritmo è $O(|\Sigma| \times \tau)$ dove τ è il costo dell'esecuzione dell'algoritmo 2.

Definizione 2.5 (Cover canonica)**.** Dato un insieme Σ di FD, è possibile trovare un insieme Δ di FD che sia una *cover canonica*, ovvero che sia una *cover* non ridondante, in cui ogni FD è sia *reduced* che in *forma canonica*. Le cover canoniche non sono uniche.

2.4 Il problema degli insiemi di FD minimi

Dato uno schema R con attributi $A_1, ..., A_k$, e dato U l'insieme di tutti i possibili attributi.

Cioè formalmente sia $R = \{A_1, ..., A_k\} \subset U$ lo schema su cui definiamo tutte le relazioni, prese ora tutte le possibili relazioni legali che soddisfano un insieme Σ di FD, ne calcoliamo l'insieme Δ il quale è *cover closed non ridondante*, cioè **minimo**, e poniamo $K = \{X_k\}$ tale che $\forall X \in K, \Delta \vdash X \longrightarrow R$ e non esiste altra Y che invece $\Delta \vdash Y \longrightarrow R$, cioè abbiamo trovato un insieme di FD minimo su tutto lo schema R a partire da K. Inoltre non esiste altro Y con questa stessa proprietà in R.

Trovare questo K che risulta essere il più piccolo ed elementare insieme di FD per R nonché unico è un problema NP-Completo.

3 Le FD per la progettazione delle Basi di Dati

Il nostro intento è ora applicare queste conoscenze sulle dipendenze funzionali nelle basi di dati.

Definizione 3.1 (Schema di Relazione)**.** Uno *schema di relazione* (R, Σ) è formato da un insieme di relazioni $R \subseteq U$, dove $U = \cup_{j=1}^{k} R_k$, e da un insieme Σ di FD.

Definizione 3.2 (Schema di Base di Dati)**.** Uno *schema di base di dati* $\mathcal{D}$ è un insieme di schemi di relazioni $\{(R_1, \Sigma_1), ..., (R_k, \Sigma_k)\}$, dove $\Sigma = \cup_{j=1}^{k} \Sigma_j$.

Definizione 3.3 (Base di Dati)**.** Una base di dati $\mathcal{B}$ sullo schema $\mathcal{D}$ non è nient'altro che un assegnamento ragionevole a ogni relazione negli schemi di relazione in $R_j \in \mathcal{D}$.

L'obiettivo primario nella progettazione è prevenire quelle che sono dette anomalie di Codd, le quali non devono sussistere durante le operazioni sulla base di dati. Questo problema ha protato alla creazione di quelle che sono dette le *Forme Normali* dei schemi di basi di dati.

3.1 Le chiavi nelle Basi di Dati con le FD

Diamo ora alcune definizioni fondamentali per il modello relazionale delle basi di dati, basati sulla logica e sulle analisi precedenti delle dipendenze funzionali.

Definizione 3.4 (Determinante)**.** Dato uno schema di relazione (R, Σ), un insieme di attributi $X \subseteq R$ è un *determinante* per R se esiste almeno un attributo $A \in R - X$ tale che $\Sigma \vDash X \longrightarrow A$.

Definizione 3.5 (Superchiave)**.** Dato uno schema di relazione (R, Σ), un insieme di attributi $X \subseteq R$. Se $\forall A \in R$ abbiamo $\Sigma \vDash X \longrightarrow A$ allora diremo che X è una *superchiave* per R.

Definizione 3.6 (Chiave)**.** Data una *superchiave* X su uno schema di relazione (R, Σ), se $\forall B \in X, \Sigma \vDash X - B \not\longrightarrow A$ allora ci riferiremo a X come *chiave* per R.

Abbiamo tutti gli strumenti per scrivere un algoritmo che costruisce chiavi per uno schema di relazione (R, Σ).

Algorithm 5: MINIMAL - KEY

Data: $R \leftarrow$ un insieme di relazioni in input.
Data: $\Sigma \leftarrow$ un insieme di FD in input.
$\Delta \leftarrow \Sigma$
$X \leftarrow R$
for *each* $A \in R$ **do**
  if $\Delta \vDash X - A \longrightarrow R$ **then**
    $X \leftarrow X - A$
end
return X

In una base di dati può essere molto interessante sapere tutte le chiavi esistenti per uno schema di relazione (R, Σ), l'algoritmo qui sotto calcola queste chiavi con un approccio di tipo brute-force.

Algorithm 6: ALL - MINIMAL - KEYS

Data: $R \leftarrow$ un insieme di relazioni in input.
Data: $\Sigma \leftarrow$ un insieme di FD in input.
$Q \leftarrow \mathcal{P}^R$
$K \leftarrow \emptyset$
while $Q \neq \emptyset$ **do**
  $X \in Q$
  if $\Sigma \vDash X \longrightarrow R$ **then**
    $Z \leftarrow NONREDUND(X)$
    $K \leftarrow K \cup \{Z\}$
    Siano $S_1, ..., S_r \in Q$ tali che $X \subset S_i, \forall i = 1, .., r$
    $Q \leftarrow Q - X - S_1 - ... - S_r$
  end
  else
    $Q \leftarrow Q - X$
  end
end
return K

Dove $\mathcal{P}^R$ è l'insieme delle parti di R.

3.2 La Forma Normale di Boyce-Codd

Possiamo ora affermare che la Forma Normale di Boyce-Codd è la più forte di tutte le forme normali che incontreremo più aventi, e ne diamo la definizione:

Definizione 3.7 (BCNF)**.** Uno schema di basi di dati $\mathcal{D}$ soddisfa la *BCNF* quando se X è un *determinante* di R allora X è una *superchiave* per R, dove R fa parte di un qualunque schema di relazione in $\mathcal{D}$. Ciò implica che per $A \in R - X$ dove $X \longrightarrow A$ è una FD, si ha necessariamente X *chiave*.

Definizione 3.8 (Proiezione su insiemi di FD)**.** Dato un insieme di attributi $X \subset R$ nel contesto (R,Σ) definiamo la proiezione di Σ su X come $\pi_X(\Sigma) = \{W \longrightarrow A \in \Sigma$ e inoltre $W \cup A \subseteq X\}$

3.2.1 Verifica della BCNF

Supponiamo che da $\mathcal{D}$ vogliamo ottenere $\mathcal{E}$ in *BCNF*. pertanto, sia $\Sigma_j = \pi_{R_j}(\Sigma)$, se $\forall X \longrightarrow A \in \Sigma_j$ si ottiene $\Sigma_j \vDash X \longrightarrow R_j$ allora (R_j,Σ_j) è in *BCNF*.

3.3 Normalizzazione tramite decomposizione

Sia $R = ABCDE$ uno schema di relazione tale che per ogni relazione si abbia la FD $E \longrightarrow CD$ valida. Considereremo le decomposizioni di R su ABE e CDE, in particolare costruiamo ragionevolmente una relazione I di R dove la FD è valida e che i valori di A,B siano riempiti casualmente:

A	B	C	D	E
0	0	0	0	1
0	1	0	0	1
1	0	1	1	0
1	1	1	1	0

Si nota che $\pi_{ABE}(I) * \pi_{CDE}(I) = I$ è vero $\forall I$, questa non è una coincidenza visto che la presenza di FD permettere di garantire decomposizioni lossless così come affermato dal seguente teorema.

Teorema 3.1. *Sia I definita su $R = XYZ$ tale che I soddisfi la FD $X \longrightarrow Y$ allora $\pi_{XY}(I) * \pi_{XZ}(I) = I$, cioè le proiezioni sono lossless.*

Per normalizzare usando la decomposizione assumiamo $\mathcal{D} = \{(R_1, \Sigma_1, ..., (R_k, \Sigma_k)\}$ non in *BCNF* e che (R_j, Σ_j) sia la causa della violazione e che quindi esista un X che sia *determinante* per R_j ma che non sia *superchiave*, cioè esiste un attributo $A \in R_j - X$ tale che $\Sigma \vDash X \longrightarrow A$ dove $\Sigma = \cup_{j=1}^{k} \Sigma_j$, usando il Teorema 3.1 sostituiamo (R_j, Σ_j) con $D_1 = (\pi_{XA}(R_j), \Sigma_j^1)$ e $D_2 = (\pi_{R_j - A}(R_j), \Sigma_j^2)$ e continuiamo questo processo finché D_2 non è in *BCNF*.

Dato $\mathcal{D}$ come sopra, e dato $\mathcal{E} = (U, \Sigma)$ detto schema universale, ci chiediamo ora se è sensato dire che $\mathcal{D}$ rappresenti $\mathcal{E}$, per far ciò intituivamente dobbiamo dire che non ci debbano essere perdite di informazioni rispetto alla base di dati di partenza, e che tutte le dipendenze funzionali siano mantenute, cioè la trasformazione applicata deve essere *lossless* e *dependency preserving*, ciò viene riassunto dalle due proprietà qui sotto:

Definizione 3.9 (Trasformazione Lossless)**.** Sia I una relazione su U che soddisfa Σ, allora la decomposizione di I in $\pi_{R_j}(I), j = 1, ..., k$ è lossless.

Definizione 3.10 (Trasformazione Dependency Preserving)**.** Sia $\Delta = \cup_{j=1}^{k} \Sigma_j$, allora $\Sigma \vDash \Delta$ e $\Delta \vDash \Sigma$.

Lemma 3.2. *Uno schema di base di dati $\mathcal{E}$ a partire da $\mathcal{D}$ rappresenta $\mathcal{D}$ se essa subisce una trasformazione che verifica 3.9 e 3.10.*

La decomposizione effettuata usando il Teorema 3.1 come sopra garantisce 3.9 ma solo metà di 3.10 è soddisfatta. Sfortunatamente appare che non sia possibile trovare efficientemente una decomposizione che porti alla *BCNF* che soddisfi entrambe queste condizioni, enunciamo ora un Teorema che afferma questo.

Teorema 3.3 (C. Beeri e P.A. Bernstein)**.** *Sia $\mathcal{D}$ uno schema di base di dati, verificare che ci sia una violazione alla* ***BCNF*** *è un problema NP-Completo.*

3.4 La Terza Forma Normale e la normalizzazione via sintesi

Assumiamo che $\mathcal{D}$ sia uno schema di base di dati, a partire da $\mathcal{D}$ ottenere un altro schema in *BCNF* che lo rappresenti potrebbe non essere fattibile ed è accettabile fermarsi a ottenere una forma più debole, cioè la Terza Forma Normale (3NF). La definizione di 3NF può essere data a partire da quella di *determinante forte*.

Definizione 3.11 (Primalità)**.** Un attributo $A \in R_j$ è detto *primo* se esiste una chiave Z di R_j tale che $A \in Z$. Determinare la *primalità* di un attributo è un problema NP-Completo.

Definizione 3.12 (Determinante Forte)**.** Z sarà un *determinante forte* di R_j se $Z \subseteq R_j$ ed esiste un A *non primo* tale che $A \in R_j - Z$ soddisfi $\Sigma \vDash Z \longrightarrow A$. Non è necessario che Z sia una superchiave.

Teorema 3.4 (Propagazione della *primalità* in catene di FD)**.** *Sia (R, Σ) uno schema di relazione, $\forall j : 1 \leq j \leq n$ sia $A_j \longrightarrow A_{j+1} \in \Sigma$, sia dato K* ***superchiave****, se $A_{n+1} \in K$ allora $A_1 \in K$.*

Dimostrazione. Data una FD $A \longrightarrow B$, si ha che A è *primo* se B lo è. Per questo senza perdita di generalità supponiamo $B \not\longrightarrow A$ la *primalità* di B significa che $K - B \not\longrightarrow R$ per una superchiave K tale che $B \in K$, se $A \in K$ il risultato segue. se $A \notin K$ allora si avrebbe che $\{K \longrightarrow A, B \not\longrightarrow A\} \vDash K - B \longrightarrow A$, cioè una contraddizione. □

Definizione 3.13 (Terza Forma Normale)**.** Uno schema di base di dati $\mathcal{D}$ soddisfa la Terza Forma Normale (3NF) quando se Z è un *determinante forte* per R in $\mathcal{D}$, allora Z è una superchiave per R.

Determinare che uno schema di relazione (R_j,Σ_j) sia in *3NF* può essere fatto nei due seguenti modi:

- Mostrare che per tutte le FD $X \longrightarrow A \in \Delta_j$, X è una *superchiave*, o A è *primo*, dove $\Delta_j = CANONICAL(\Sigma_j)$, e quindi la *3NF* è rispettata.
- Trovare una FD $X \longrightarrow A \in \Delta_j$, tale che X **non** è una *superchiave* e A **non** è *primo*, e quindi la *3NF* è violata.

Lemma 3.5. *Uno schema di base di dati $\mathcal{D}$ se soddisfa la BCNF allora soddisfa anche la 3NF.*

Teorema 3.6. *Sia (R, Σ) uno schema di relazione in 3NF, se per ogni coppia di chiavi K_1, K_2 si ha $K_1 \cap K_2 = \emptyset$ allora (R, Σ) è in BCNF.*

Ecco ora un algoritmo di sintesi della 3NF a partire da uno schema di relazione (U, Σ):

Algorithm 7: 3NF

Data: $U \leftarrow$ un insieme di relazioni in input.
Data: $\Sigma \leftarrow$ un insieme di FD in input.
$\mathcal{D} \leftarrow \phi$
$\Delta \leftarrow CANONICAL(U, \Sigma)$
$X \leftarrow MINIMAL - KEY(U, \Delta)$
for *each FD* $Y \longleftarrow A \in \Delta$ **do**
  $\mathcal{D} \leftarrow \mathcal{D} \cup (YA, \pi_{YA}(\Delta))$
end
$\mathcal{D} \leftarrow \mathcal{D} \cup (X, \phi)$
return $\mathcal{D}$

L'algoritmo inizialmente calcola una *cover canonica* per Σ, poi ne cerca una *chiave*. Successivamente genera uno schema di relazione per ogni FD, e infine aggiunge la chiave a $\mathcal{D}$.

3.5 Altri aspetti progettuali sulle FD

Immaginiamo di avere il seguente schema di relazione:

$$\textit{Libri}(\text{Codice,Autore,Titolo,Grandezza,Npagine})$$

Di cui è una possibile istanza la tabella qui sotto:

Codice	Autore	Titolo	Grandezza	Npagine
111	Elisabetta	Cucina per casa	Poster	150
111	Elisabetta	Cucina per casa	A5	300
222	Gabriele	Dipendenze Funzionali	A5	400
222	Gabriele	Dipendenze Funzionali	PBack	320
222	Gabriele	Dipendenze Funzionali	Tascabile	160

Qui si evidenzia come nella giusta interpretazione bisogni avere la FD $Codice \longrightarrow Autore$ mentre si nota che anche gli altri parametri dipendano dal Codice, sebbene non funzionalmente, Questo perché per esempio $Grandezza$ è sì determinato da $Codice$, così come $Titolo$, ma non rappresenterebbe correttamente il mondo che ci interessa modellare.

Pertanto uno schema di relazione può anche essere rappresentato da una tripla (R, Σ_1, Σ_0) tale che:

- Σ_1 è l'insieme delle FD valide σ_i^1, dove $\Sigma_1 \cup \Sigma_0 \vDash \sigma_i^1$
- Σ_0 è l'insieme delle FD escluse σ_i^0, dove $\Sigma_1 \cup \Sigma_0 \vDash \sigma_i^0$

E quindi sebbene $Codice \longrightarrow Grandezza$ possa essere un'ottima FD candidata, la correttezza è data da un attenta analisi dei requisiti di progetto che vanno a modellare la realtà di interesse.

4 Conclusione

Abbiamo avuto modo di esplorare l'utilizzo delle dipendenze funzionali nelle basi di dati, in particolare quelle che sfruttano il modello relazionale, dalla sintassi fino al ragionamento e visionato un sistema formale di prova per la deduzione di FD, partendo da ciò siamo riusciti a dare due esempi reali e pratici che sfruttano le FD, cioè la Forma Normale di Boyce-Codd e la 3NF, in realtà esistono forme normali più deboli (Seconda, Prima...) così come altre generalizzazioni (Quarta...) che non sono state trattate in questo progetto.

In verità è rimarcabile affermare come un ottima progettazione concettuale così come una eccellente progettazione logica a partire dai requisiti di progetto, portino alla realizzazione di Schemi di Basi di Dati già in forme normali. Evidente nel caso della ristrutturazione della progettazione logica, dove in caso di tabelle **lascamente accoppiate** era previsto un accorpamento per *denormalizzazione* per facilitare l'accesso ai dati e ridurre il costo computazionale.

In via definitiva le FD sono un mattone fondamentale della progettazione di basi di dati nel modello relazionale, apparendo silenziosamente in tutti i progetti ben svolti, e averne un'accurata conoscenza aiuta in tutte le fasi del ciclo di vita della base di dati.

Riferimenti bibliografici

[1] K. Viswanathan Iyer. An introduction to functional dependency in relational databases. 2016.

[2] Maurizio Lenzerini. *Materiale del corso "Logica e Informatica"*. 2024.

[3] Maurizio Lenzerini. *Materiale del corso "Basi di Dati"*. 2024.