Basi di Dati: Teoria Relazionale

F.M. Malvestuto

(Corso di Laurea in INFORMATICA, A. A. 2002/2003)

INDICE

PROGRAMMA

Capitoli 1, 2, 3

Capitolo 4: Par. 4.1, Par. 4.2 (Lemma 4.1 con la dimostrazione, Teoremi 4.1 e 4.2 senza dimostrazione), Par. 4.3 (Lemma 4.2 con la dimostrazione, Lemma 4.3 e Teorema 4.3 senza dimostrazione), Par. 4.4 (Teorema 4.4 senza dimostrazione, fine pagina 60 ed inizio pagina 61 esclusi), Par. 4.5 (solo il Test di Validità e l'enunciato del Teorema 4.5), Par. 4.6 (solo il Test di Normalità e l'enunciato del Teorema 4.6).

Capitolo 5: Par. 5.1, Par 5.2 (solo Teorema 5.2 con dimostrazione).

Capitolo 6: Par. 6.1 e 6.2.

Capitolo 7: Par. 7.1 (Lemma 7.1 senza dimostrazione), Par. 7.2 (Teoremi 7.1 e 7.2 con dimostrazione), Par. 7.3 (Teoremi 7.3 e 7.4 senza dimostrazione), Par. 7.4 (Teorema 7.6 senza dimostrazione), Par. 7.5 (Lemmi 7.2 e 7.3 con dimostrazione), Par. 7.6 (Teoremi 7.10 e 7.11 senza dimostrazione),

INTRODUZIONE

Lo sviluppo dei mezzi e delle tecniche per la registrazione delle informazioni ha sempre accompagnato il progetto di costruire archivi di conoscenze e di esperienze. Si ritiene che storicamente l'idea di raccogliere ed organizzare informazioni scritte abbia avuto origine in due regioni, la Mesopotamia e l'Egitto, circa 5000 anni fa. Le prime raccolte sumere ed egizie di informazioni registrate in cuneiforme su tavolette ed in geroclifico su papiro, contenevano informazioni di natura legale ed economica — leggi e trattati, contratti, rogiti notarili, operazioni fiscali e lasciti testamentari. In questi ed altri documenti è difficile distinguere i concetti di archivio e di biblioteca. Dal Medio Oriente il concetto di raccolta di documenti entrò nel mondo greco e romano. I re romani istituzionalizzarono i censimenti delle popolazioni e delle proprietà fin dal VI secolo a. C. La grande Biblioteca di Alessandria d'Egitto, fondata nel III secolo a. C., conteneva un'immensa raccolta di papiri dove erano registrati atti catastali e fiscali. Di fatto, si tratta di un *sistema informativo* ante litteram. Lo splendore della cultura araba dall'VIII al XIII secolo lo si può in larga parte attribuire alla disponibilità di biblioteche pubbliche e private. La *Bayt al-Hikmah* ("La Casa della Saggezza"), fondata nell'830 a Bagdad, conteneva una biblioteca pubblica con una voluminosa raccolta di materiale su argomenti diversi, e la biblioteca del califfo al-Hakam a Cordova vantava più di 400.000 volumi.

Il rapido sviluppo (anche se tardivo) delle biblioteche europee a partire dal XVI secolo seguì l'invenzione della stampa. Fin dall'inizio del XVII secolo, la pubblicazione letteraria è diventata il mezzo principale per la diffusione delle conoscenze. L'espressione *letteratura primaria* è usata per designare informazioni di contenuto originale stampate in formato vario: giornali e periodici, monografie, atti di convegni, rapporti, brevetti, bollettini. La rivista scientifica, classico mezzo di comunicazione nelle comunità scientifiche, nasce nel 1665. Alla fine del XX secolo, si è stimato che il numero di riviste scientifiche stampate nel mondo superi i 60.000 titoli, riflesso non solo della crescita del numero degli appassionati di scienza e dell'espansione del corpo delle conoscenze, ma anche del credito riconosciuto alle pubblicazioni. Per rendere questo patrimonio di conoscenze più accessibile e gestibile, una porzione sempre più grande viene conservata anche in versione elettronica.

La quantità dell'informazione stampata ha impedito per qualche tempo di trarne il massimo profitto in senso conoscitivo. Strumenti come indici e sommari, che aiutano ad identificare e localizzare informazioni d'interesse nella letteratura primaria, sono stati in uso sin dal XVI secolo e portarono allo sviluppo di quella che nel XIX secolo è

stata chiamata *letteratura secondaria*, che risponde all'esigenza di filtrare le fonti informative primarie.

Il fiorire della tecnologia digitale dalla metà del XX secolo ha rivoluzionato il modo di registrare e di diffondere le informazioni. Nei primi anni '60, per la prima volta i computer vennero usati per registrare testi con il proposito di ridurre i costi ed i tempi richiesti dalla pubblicazione di due repertori, l'*Index Medicus* della National Library of Medicine ed i *Scientific and Technical Aerospace Reports* della NASA. Dalla fine degli anni '60, queste basi di dati bibliografici e numerici sono diventate una nuova fonte informativa che è proliferata anche fuori degli ambiti istituzionali tradizionali e si è diffusa grazie al progresso tecnologico.

L'enorme quantità delle informazioni in formato digitale oggi esistenti pongono in maniera drammatica il problema della loro utilizzazione, che trova soluzione solo se esse sono organizzate con sistematicità e se esistono meccanismi per la ricerca ed il reperimento di singoli elementi su cui di volta in volta si appunta l'interesse dell'utente.

Il complesso delle rappresentazioni digitali delle informazioni in un moderno sistema informativo viene chiamato il suo *insieme di dati* ("data set"). Tipicamente, ogni insieme di dati si compone di una o più *basi di dati*, ed ogni base di dati è un insieme di file mutuamente connessi. Gli insiemi di dati sono organizzati in diverse *strutture di dati* che ne facilitano la creazione, l'accesso e l'aggiornamento ed ottimizzano la gestione delle risorse fisiche. Dal punto di vista della registrazione dell'informazione in forma digitale, è utile distinguere tra dati strutturati, come avviene negli elenchi e nei cataloghi dove gli oggetti sono rappresentati da brevi stringhe di simboli e da numeri, e dati nonstrutturati come nei testi scritti in un linguaggio naturale o nelle immagini figurative. L'obiettivo principale è quello di facilitare l'elaborazione dei dati sulla base delle loro relazioni. La scelta di una particolare struttura di dati è dettata dalla ricchezza espressiva che essa consente in vista dei compiti elaborativi assegnati al sistema informativo. Nei sistemi informativi i cui archivi contengono basi di dati nonstrutturati (ad esempio, documenti scritti), l'obiettivo è quello di ricercare record o loro porzioni in base alla presenza di certe parole od espressioni linguistiche oggetto dell'interrogazione. Grazie ad un indice (un altro file) che fornisce l'informazione necessaria a localizzare parole ed espressioni nei record della base di dati, le relazioni d'interesse (ad esempio, l'adiacenza di parole) possono essere desunte dall'indice; così, il testo stesso può essere registrato come un semplice file di record ordinato e sequenziale. Nelle basi di dati strutturati, le relazioni tra i dati individuali sono rappresentate nella struttura dei record e, fino agli anni '70, due soli tipi di struttura venivano comunemente usati: quella "gerarchica" e quella "reticolare". Dagli anni Settanta, si è affermato prepotentemente l'uso delle strutture "relazionali" il cui

successo è dovuto alla sua semplicità concettuale, al rigore della sua teoria (l'algebra relazionale) ed alla disponibilità di programmi software per il trattamento delle relazioni tra i dati con una logica indipendente dalla rappresentazione fisica.

I sistemi informativi sono oggi il veicolo primario per la raccolta, l'organizzazione, la registrazione, l'elaborazione e la presentazione delle informazioni. In linea di principio, un sistema informativo è un qualsiasi sistema che conservi registrazioni — ad esempio, una rubrica telefonica. Ma l'aspetto che distingue i moderni sistemi informativi è la loro dimensione digitale, che consente il trattamento veloce ed automatico di dati registrati in forma digitale e la loro traduzione da ed in formato analogico. Dare una classificazione dei sistemi informativi non è facile a causa della loro varietà e della continua evoluzione delle loro funzioni e delle loro strutture. La distinzione tra manuali ed automatici, oppure tra interattivi ("on-line") e non ("off-line") oppure ancora tra quelli che operano in "real-time" e quelli in "batch" non è più molto utile. È forse più ragionevole distringuerli dal punto di vista funzionale, cioè in base alle loro applicazioni; li distingueremo così in due categorie: i sistemi informativi che operano in contesti organizzativi ed i sistemi informativi che espletano servizi di pubblica utilità. La prima categoria comprende i sistemi informativi utilizzati per lo svolgimento sia delle funzioni direttive che delle attività operative. La seconda categoria comprende i prodotti dei cosiddetti "venditori di basi di dati", vale a dire basi di dati che contengono documenti testuali, dati aziendali ed industriali, statistiche e cataloghi di prodotti e servizi.

Il crescente interesse nel trattamento dell'informazione e nei sistemi informativi ha dato vita ad un campo di studi interdisciplinare che va sotto il nome di *scienza dell'informazione* e dove confluiscono discipline come l'informatica, le scienze cognitive, l'intelligenza artificiale, l'ingegneria, la matematica, le scienze sociali e del comportamento, la linguistica e l'archivistica. Come i fenomeni che essa studia, la scienza dell'informazione è dunque un campo composito d'indagine la cui struttura è in continua evoluzione.

CAPITOLO 1

BASI DI DATI RELAZIONALI

1.1 Sistemi informativi

Sotto il profilo squisitamente tecnico, un sistema informativo è un complesso di dati organizzati fisicamente in una memoria secondaria e gestiti in maniera tale da curarne la creazione, l'aggiornamento e l'interrogazione. I dati sono organizzati concettualmente in aggregati di informazioni omogenee che costitutiscono le componenti del sistema informativo, ed ogni operazione di aggiornamento ha per oggetto un singolo aggregato mentre un'interrogazione può coinvolgerne uno o più.

Comunque, le caratteristiche tecniche seppure eccellenti di un sistema informativo non valgono nulla se le informazioni in esso contenute non sono veritiere, se cioè non c'è corrispondenza fra l'immagine del mondo fornita dal sistema informativo e l'effettivo stato delle cose nel mondo stesso. Possiamo così riassumere la questione:

1. C'è un mondo che vogliamo descrivere.
2. Vogliamo descriverlo usando un sistema informativo.
3. Il problema è di stabilire se le informazioni riflettono lo stato reale del mondo.
4. Per verificare la corrispondenza fra le informazioni ed il modo in cui stanno realmente le cose, abbiamo bisogno di un metalinguaggio.

1.2 Relazioni

Una base di dati relazionale è un sistema informativo le cui componenti hanno una struttura tabellare e sono chiamate relazioni, e ciascuna componente ha un nome proprio usato per essere identificata sia nelle operazioni di aggiornamento che nelle interrogazioni.

Una *variabile relazionale* è una variabile strutturata $\boldsymbol{r}$ il cui tipo è specificato da una coppia R, d dove R è un insieme di nomi (*attributi*), che chiamiamo lo *schema* di $\boldsymbol{r}$, e d è una funzione che associa ad ogni attributo A in R un insieme finito o infinito ma numerabile, che chiamiamo *dominio* dell'attributo A.

I valori della variabile $\boldsymbol{r}$ sono definiti come insiemi di "ennuple" con "schema" R. Un'*ennupla* (*tuple*, in inglese) con *schema* R è una funzione definita su R che associa ad ogni attributo A in R un elemento di $d(A)$. Se t è un'ennupla con schema R ed A è un attributo in R, allora $t(A)$ starà ad indicare l'elemento di $d(A)$ che è il valore assunto dalla funzione t in corrispondenza dell'attributo A. L'insieme delle ennuple con schema R prende il nome di dominio di R che indichiamo con $dom(R)$. Una *relazione* con *schema* R è un sottoinsieme finito (eventualmente vuoto) di $dom(R)$. Dunque, un *valore* della variabile relazionale $\boldsymbol{r}$ è una qualsiasi relazione con schema R. Solitamente,

una relazione con schema R viene rappresentata come una tabella, le cui colonne corrispondono agli attributi in R (presi in un ordine qualsiasi) e le cui righe corrispondono alle ennuple della relazione (anch'esse prese in un ordine qualsiasi).

Esempio 1.1 Consideriamo la variabile relazionale di nome **partenze** il cui tipo è definito dallo schema {**VOLO**, **DATA**, **PILOTA**, **ORA**} e dalla funzione d così definita:

d(**VOLO**) = insieme di codici numerici formati da tre cifre decimali

d(**DATA**) = insieme di stringhe alfanumeriche

d(**PILOTA**) = insieme di stringhe alfabetiche

d(**ORA**) = insieme di stringhe del tipo "k: n" dove k {0, …, 23} ed n {00, …, 59}

Un esempio di ennupla con schema {**VOLO**, **DATA**, **PILOTA**, **ORA**} è dato dalla funzione t che assume i seguenti valori:

t(**VOLO**) = 112
t(**DATA**) = 6 giugno
t(**PILOTA**) = Rossi
t(**ORA**) = 10:30

Un valore possibile di **partenze** è la relazione riportata in Tabella 1.1.

VOLO	DATA	PILOTA	ORA
1126	giugno	Rossi	10:30
1127	giugno	Verdi	10:30
2009	giugno	Verdi	12:00
11210	giugno	Rossi	10:30

Tabella 1.1 Un valore della variabile relazionale **partenze** ◊

1.3 Basi di dati

Una *base di dati* (*relazionale*) ***D*** è definita da

un insieme $\{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n\}$ di variabili relazionali,
un insieme finito U di attributi,
una famiglia $\{R_1, \ldots, R_n\}$ di sottoinsiemi non vuoti di U, che chiamiamo lo *schema* della base di dati, dove R_i è designato come lo schema di $\boldsymbol{r}_i$,
una famiglia Δ di domini,
una funzione d definita su U che associa ad ogni attributo A un insieme in Δ.

Lo *stato* della base di dati ***D*** ad un certo istante è un insieme di relazioni $D = \{r_1, \ldots, r_n\}$, dove r_i è un valore della variabile relazionale $\boldsymbol{r}_i$, $1 \leq i \leq n$, a cui ci riferiremo come la *relazione di nome* $\boldsymbol{r}_i$ contenuta nello stato D di ***D***.

I due esempi di basi di dati qui di seguito riportati verranno frequentemente richiamati nei prossimi capitoli con i nomi di "Base di Dati Aeroportuali" e "Basi di Dati Dinastici".

Esempio 1.2 Consideriamo una base di dati ***D*** che contenga quattro variabili relazionali: **inventario** con schema {**A#**, **PARTE-DI**, **NOME**}, **dotazione** con schema {**A#**, **VETTORE**, **NUMERO**}, **scorta** con schema {**A#**, **AEROSTAZIONE**, **LIVELLO**} e **partenze** con schema {**VOLO**, **DATA**, **PILOTA**, **ORA**}. Supponiamo che lo stato attuale D della base di dati ***D*** contenga le tre relazioni riportate nelle Tabelle 1.2, 1.3 e 1.4.

A#	PARTE-DI	NOME
211	0	poltrona di II classe
2114	211	fodera per poltrona
216	211	cintura di sicurezza
21163	2116	gancio per cinture
21164	2116	attacco per cinture
206	0	pannello superiore
2066	206	dispositivo luminoso
2068	206	dispositivo d'areazione

Tabella 1.2 La relazione di nome **inventario** contenuta in D

A#	VETTORE	NUMERO
211	707	86
211	727	134
2114	707	86
2114	727	134
2116	707	244
2116	727	296
21164	707	488
21164	727	592

Tabella 1.3 La relazione di nome **dotazione** contenuta in D

A#	AEROSTAZIONE	LIVELLO
211	Fiumicino	106
211	Ciampino	28
211	Linate	6
2114	Malpensa	57

Tabella 1.4 La relazione di nome **scorta** contenuta in D

VOLO	DATA	PILOTA	ORA
1126	giugno	Rossi	10:30
1127	giugno	Verdi	10:30

2009 giugno	Verdi	12:00
11210 giugno	Rossi	10:30

Tabella 1.5 La relazione di nome **partenze** contenuta in *D* ◊

Esempio 1.3 Consideriamo una base di dati ***D*** che contenga due variabili relazionali: **regno** con schema {**SOVRANO**, **INIZIO**, **FINE**} e **dinastia** con schema {**NOME**, **SESSO**, **NASCITA**, **MORTE**}. Supponiamo che lo stato attuale *D* della base di dati ***D*** contenga le due relazioni riportate nelle Tabelle 1.6 e 1.7 (le informazioni ivi contenute riguardano la dinastia scozzese degli Stuart).

SOVRANO	**INIZIO**	**FINE**
Giacomo I	1603	1625
Carlo I	1625	1648
Carlo II	1660	1685
Giacomo II	1685	1688
Maria II	1688	1694
Anna	1702	1714

Tabella 1.6 La relazione di nome **regno** contenuta in *D*

NOME	**SESSO**	**NASCITA**	**MORTE**
Giacomo I	M	1566	1625
Elisabetta	F	1590	1662
Carlo I	M	1600	1649
Carlo II	M	1630	1685
Maria	F	1631	1659
Giacomo II	M	1633	1701
Enrichetta A.	F	1640	1670
Maria II	F	1662	1694
Anna	F	1665	1714
Giacomo E.M		1686	1766

Tabella 1.7 La relazione di nome **dinastia** contenuta in *D*

CAPITOLO 2

INTERROGAZIONE DI UNA BASE DI DATI

Come già detto, una base di dati è essenzialmente una raccolta di dati accessibili ad una comunità di soggetti che formano l'utenza del sistema informativo. Oltre a rendere reperibili per intero le relazioni contenute nella base di dati, i linguaggi di interrogazione permettono anche di selezionare porzioni di singole relazioni ed infine di combinare informazioni provenienti da distinte relazioni.

Esempio 2.1 Ecco due esempi di domande sulla Base di Dati Aeroportuali:

1. "Quali sono le parti dell'articolo di codice 211?"
Risposta:

NOME
fodera per poltrona
cintura di sicurezza

Tabella 2.1

2. "Quante **poltrone di II classe** sono usate sul vettore 727?"
Risposta:

NUMERO
134

Tabella 2.2 ◊

Esempio 2.2 Ecco tre esempi di domande sulla Base di Dati Dinastici:

1. "Chi furono i sovrani che precedettero Carlo II, e in che anno salirono al trono ed in che anno lo lasciarono?".
Risposta:

SOVRANO	INIZIO	FINE
Giacomo I	1603	1625
Carlo I	1625	1648

Tabella 2.3

2. "Chi furono il primo re e la prima regina?"
Risposta:

SOVRANO
Giacomo I
Maria II

Tabella 2.4

3. “Chi fu l’ultimo sovrano ed in che anno nacque?”
Risposta:

SOVRANO	NASCITA
Anna	1665

Tabella 2.5 ◊

Esistono due modelli formali per interrogare una base di dati relazionale: il primo, logico, è un linguaggio “dichiarativo”; il secondo, algebrico, è un linguaggio “procedurale”. Ad essi sono dedicati i prossimi paragrafi.

2.1 L’APPROCCIO LOGICO

L’interrogazione di una base di dati relazionale ***D*** avviene formulando le domande con espressioni del *Calcolo Relazionale*, che è un’estensione della logica dei predicati del prim’ordine. Qui, le ennuple vengono rappresentate da variabili (*x, y, z,* …). Inoltre, le variabili relazionali in ***D*** compaiono nelle formule del Calcolo relazionale con funzione di *predicati* (di appartenza di un’ennupla ad una relazione). Prima di definire formalmente il nostro linguaggio, vogliamo illustrare queste nozioni servendoci degli Esempi 2.1 e 2.2.

Esempio 2.1 (seguito). Prendiamo la domanda “Quali sono le parti dell’articolo di codice 211?”. Una sua formulazione logica è la seguente:

Quali sono i valori *x* dell’attributo **NOME** per cui

esiste un’ennupla *y* con schema {**A#**, **PARTE-DI**, **NOME**} tale che
y appartiene alla relazione **inventario**, *y*(**PARTE-DI**) = 211 e
y(**NOME**) = *x* ?

In maniera più formale, possiamo selezionare i valori della variabile *x* che entreranno a far parte della relazione cercata (vedi Tabella 2.1) con la formula

$$y(\mathbf{A\#}, \mathbf{PARTE\text{-}DI}, \mathbf{NOME})$$
$$(\mathbf{inventario}(y) \quad y(\mathbf{PARTE\text{-}DI}) = 211 \quad y(\mathbf{NOME}) = x(\mathbf{NOME}))$$

Prendiamo ora la domanda “Quante poltrone di II classe sono usate sul vettore 727?”. Una formulazione logica della domanda è la seguente:

Qual è il valore *x* dell’attributo **NUMERO** per cui

esiste un’ennupla *y* con schema {**A#**, **VETTORE**, **NUMERO**} tale che
y appartiene alla relazione **dotazione**,
y(**VETTORE**) = 727,
y(**NUMERO**) = *x*

ed esiste un'ennupla *z* con schema {**A#**, **PARTE-DI**, **NOME**} tale che
z appartiene alla relazione **inventario**,
z(**A#**) = *y*(**A#**) e
z(**NOME**) = poltrona di II classe

?

In maniera più formale, possiamo selezionare i valori della variabile *x* che entreranno a far parte della relazione cercata (vedi Tabella 2.2) con la formula

y(**A#**, **VETTORE**, **NUMERO**)
(**dotazione**(*y*) *y*(**VETTORE**) = 727 *y*(**NUMERO**) = *x*(**NUMERO**)
z(**A#**, **PARTE-DI**, **NOME**)
(**inventario**(*z*) *z*(**A#**) = *y*(**A#**) *z*(**NOME**) = poltrona di II classe))

◊

Esempio 2.2 (seguito). Prendiamo la domanda "Chi furono i sovrani che precedettero Carlo II, ed in che anno salirono al trono ed in che anno lo lasciarono?". Una sua formulazione logica è la seguente:

Quali sono le terne *x* con schema {**SOVRANO**, **INIZIO**, **FINE**} tali che

x appartiene alla relazione **regno**,

ed esiste una terna *y* con schema {**SOVRANO**, **INIZIO**, **FINE**} tale che

y appartiene alla relazione **regno**,
y(**SOVRANO**) = Carlo II e
x(**FINE**) ≤ *y*(**INIZIO**)

?

In maniera più formale, possiamo selezionare le terne *x*(**SOVRANO**, **INIZIO**, **FINE**) che entreranno a far parte della relazione cercata (vedi Tabella 2.3) con la formula

regno(*x*) (*y*(**SOVRANO**, **INIZIO**, **FINE**)
(**regno**(*y*) *y*(**SOVRANO**) = Carlo II *x*(**FINE**) ≤ *y*(**INIZIO**)))

Prendiamo poi la domanda "Chi furono il primo re e la prima regina?". Una sua formulazione logica è la seguente:

Quali sono i valori *x* dell'attributo **SOVRANO** per cui

esiste un'ennupla *y* con schema {**SOVRANO**, **INIZIO**, **FINE**} tale che
y appartiene alla relazione **regno**,
y(**SOVRANO**) = *x* e
per ogni ennupla *y'* con schema {**SOVRANO**, **INIZIO**, **FINE**}

se *y'* appartiene alla relazione **regno** ed *y*(**INIZIO**) > *y'*(**INIZIO**)
allora

esiste un'ennupla z con schema {**NOME**, **SESSO**, **NASCITA**, **MORTE**} tale che

z appartiene alla relazione **dinastia**,
z(**NOME**) = x ed

esiste un'ennupla z' con schema {**NOME**, **SESSO**, **NASCITA**, **MORTE**} tale che

z' appartiene alla relazione **dinastia**,
z'(**NOME**) = y'(**SOVRANO**) e
z'(**SESSO**) ≠ z(**SESSO**)

?

In maniera più formale, possiamo selezionare i valori della variabile x che entreranno a far parte della relazione cercata (vedi Tabella 2.4) con la formula

$$
\begin{aligned}
&y(\textbf{SOVRANO}, \textbf{INIZIO}, \textbf{FINE}) \\
&\quad (\textbf{regno}(y) \quad y(\textbf{SOVRANO}) = x(\textbf{SOVRANO}) \\
&\qquad y'(\textbf{SOVRANO}, \textbf{INIZIO}, \textbf{FINE}) \\
&\qquad\quad (\neg\, \textbf{regno}(y') \quad y(\textbf{INIZIO}) \leq y'(\textbf{INIZIO}) \\
&\qquad\qquad z(\textbf{NOME}, \textbf{SESSO}, \textbf{NASCITA}, \textbf{MORTE}) \\
&\qquad\qquad\quad (\textbf{dinastia}(z) \quad z(\textbf{NOME}) = x(\textbf{SOVRANO}) \\
&\qquad\qquad\qquad z'(\textbf{NOME}, \textbf{SESSO}, \textbf{NASCITA}, \textbf{MORTE}) \\
&\qquad\qquad\qquad\quad (\textbf{dinastia}(z') \quad z'(\textbf{NOME}) = y'(\textbf{SOVRANO}) \\
&\qquad\qquad\qquad\quad z'(\textbf{SESSO}) \neq z(\textbf{SESSO})))))
\end{aligned}
$$

Prendiamo infine la domanda "Chi fu l'ultimo sovrano ed in che anno nacque?". Una formulazione logica della domanda è la seguente:

Quali sono le ennuple x con schema {**SOVRANO**, **NASCITA**} per cui

esiste un'ennupla y con schema {**SOVRANO**, **INIZIO**, **FINE**} tale che

y appartiene alla relazione **regno**,
y(**SOVRANO**) = x(**SOVRANO**) e

per ogni ennupla y' con schema {**SOVRANO**, **INIZIO**, **FINE**}
se y' appartiene alla relazione **regno**
allora y'(**INIZIO**) ≤ y(**INIZIO**),

ed esiste un'ennupla z con schema {**NOME**, **SESSO**, **NASCITA**, **MORTE**} tale che

z appartiene alla relazione **dinastia**,
z(**NOME**) = x(**SOVRANO**) e
z(**NASCITA**) = x(**NASCITA**)

?

In maniera più formale, possiamo selezionare i valori della variabile *x* che entreranno a far parte della relazione cercata (vedi Tabella 2.5) con la formula

$$
\begin{aligned}
&(\ y(\mathbf{SOVRANO}, \mathbf{INIZIO}, \mathbf{FINE}) \\
&\qquad (\mathbf{regno}(y) \quad y(\mathbf{SOVRANO}) = x(\mathbf{SOVRANO}) \\
&\qquad\quad y'(\mathbf{SOVRANO}, \mathbf{INIZIO}, \mathbf{FINE}) \\
&\qquad\qquad\qquad (\neg\, \mathbf{regno}(y') \quad y'(\mathbf{INIZIO}) \leq y(\mathbf{INIZIO})))
\end{aligned}
$$

$$
\begin{aligned}
&(\ z(\mathbf{NOME}, \mathbf{SESSO}, \mathbf{NASCITA}, \mathbf{MORTE}) \\
&\qquad (\mathbf{dinastia}(z) \quad z(\mathbf{NOME}) = x(\mathbf{SOVRANO}) \quad z(\mathbf{NASCITA}) = x(\mathbf{NASCITA})))
\end{aligned}
$$

◊

2.1.1 Calcolo Relazionale

Passiamo ora a definire il nostro linguaggio logico di interrogazione su una base di dati relazionale ***D***, che chiamiamo *Calcolo Relazionale su* ***D***. Le espressioni del Calcolo Relazionale su ***D*** sono definite attraverso le nozioni di "formula" e di "formula ben formata". Per costruire tali formule, abbiamo anche bisogno di un insieme Θ di operazioni di confronto tra valori di attributi in *U* che comprende almeno le operazioni di eguaglianza e di diseguaglianza. Se θ è un elemento di Θ, diremo che due attributi *A* e *B* sono *confrontabili* rispetto a θ (per brevità, θ-*confrontabili*) se l'operazione θ è definita sul prodotto cartesiano $d(A) \times d(B)$ dei domini di *A* e *B*. Assumeremo sempre che ogni attributo sia confrontabile con se stesso rispetto all'eguaglianza ed alla diseguaglianza. Solitamente, ci si limita a considerare l'insieme Θ che comprende le seguenti operazioni di confronto: $=, \neq, <, \leq, >$ e $\geq$.

2.1.2 Formule del CR

Una formula è definita ricorsivamente a partire da *formule atomiche*.

A1. Per ogni variabile relazionale ***r*** in ***D*** e per ogni variabile *x*,

$$\boldsymbol{r}(x)$$

è una formula atomica.

A2. Per ogni coppia di variabili *x* ed *y* (non necessariamente distinte), per ogni operazione θ in Θ e per ogni coppia di attributi *A* e *B* in *U* che siano θ-confrontabili,

$$x(A)\ \theta\ y(B)$$

è una formula atomica.

A3. Per ogni variabile x, per ogni operatore θ in Θ e per ogni attributo A in U tale che θ sia definita su $d(A)$, se a è un elemento di $d(A)$, allora

$$a\ \theta\ x(A) \quad \text{ed} \quad x(A)\ \theta\ a$$

sono formule atomiche.

Abbiamo già incontrato questi tre esempi di formule atomiche:

$$\textbf{inventario}(y)$$
$$y(\textbf{PARTE-DI}) = 211$$
$$z(\textbf{NOME}) = x(\textbf{SOVRANO}).$$

Una *formula* è costruita a partire da formule atomiche utilizzando le parentesi, i connettivi logici ed i quantificatori esistenziale ed universale.

F1. Ogni formula atomica è una formula.

F2. Se f è una formula, allora (f) è una formula.

F3. Se f è una formula, allora $\neg(f)$ è una formula.

F4. Se f e g sono formule, allora $(f) \quad (g)$ ed $(f) \quad (g)$ sono formule.

F5. Se x è una variabile, f una formula che contiene il simbolo x ed R un sottoinsieme di U, allora

$$x(R)\ (f) \qquad\qquad x(R)\ (f)$$

sono formule.

Ecco altri esempi di formule:

$y(\textbf{PARTE-DI}) = 211 \quad y(\textbf{NOME}) = x(\textbf{NOME})$

$\textbf{inventario}(y) \quad y(\textbf{PARTE-DI}) = 211 \quad y(\textbf{NOME}) = x(\textbf{NOME})$

$y(\textbf{A\#}, \textbf{PARTE-DI}, \textbf{NOME})$
$(\textbf{inventario}(y) \quad y(\textbf{PARTE-DI}) = 211 \quad y(\textbf{NOME}) = x(\textbf{NOME}))$

2.1.3 Espressioni del CR

Introdurremo ora la nozione di "formula ben formata", la quale serve ad escludere formule del tipo

dotazione(x) x(**AEROSTAZIONE**) = Fiumicino

in cui la presenza della formula atomica **dotazione**(x) sottintende che x stia a rappresentare ennuple con schema {**A#**, **VETTORE**, **NUMERO**} laddove la formula atomica x(**AEROSTAZIONE**) = Fiumicino vorrebbe che la variabile x fosse definita anche sull'attributo **AEROSTAZIONE**.

Sia f una formula. Come facciamo a sapere se f è una formula ben formata? Per rispondere a tale domanda, abbiamo bisogno di alcune nozioni che ora introduciamo. Una variabile x presente in f potrà comparirvi una o più volte e le sue presenze in f sono dette le *occorrenze* di x in f. Vedremo che un'occorrenza di una variabile che compare in f può essere *libera* o *legata*. Inoltre, le variabili con almeno un'occorrenza libera in f potranno essere di tre tipi: quelle *ben strutturate*, quelle *parzialmente strutturate* e quelle *mal strutturate*. Infine, ad ogni variabile x che abbia almeno un'occorrenza libera in f e che sia ben strutturata o anche parzialmente strutturata, faremo corrispondere un insieme (non vuoto) di attributi che chiameremo lo *schema* di x in f ed indicheremo con $\Sigma(x, f)$.

Riassumendo, se f è una qualsiasi formula, per le variabili che compaiono in f avremo:

- variabili con almeno un'occorrenza libera:
  - — variabili ben strutturate: $x_1, \ldots, x_n$ — schema di x_i : $\Sigma(x_i, f)$
  - — variabili parzialmente strutturate: $y_1, \ldots, y_m$ — schema di y_i : $\Sigma(y_i, f)$
  - — variabili mal strutturate: $z_1, \ldots, z_p$ — lo schema di z_i non è definito
- variabili con occorrenze legate: $u_1, \ldots, u_q$ — lo schema di u_i non è definito

Diamo ora la definizione (di natura ricorsiva) di *formula ben formata*, la quale essenzialmente richiede che nessuna variabile che abbia almeno un'occorrenza libera sia mal strutturata.
Cominciamo con le formule atomiche. Queste formule sono per definizione tutte ben formate.

> A1. $f = \boldsymbol{r}(x)$
> L'occorrenza della variabile x è libera in f. La variabile x è ben strutturata in f e, se R è lo schema di $\boldsymbol{r}$, lo schema di x in f è dato da R: $\Sigma(x, f) := R$.

Ad esempio, se f = **regno**(y), allora la variabile y è ben strutturata in f; inoltre $\Sigma(x, f)$:= {**SOVRANO**, **INIZIO**, **FINE**}.

A2. $f = x(A)\ \theta\ y(B)$
Entrambe le occorrenze delle variabili x ed y sono libere in f. Le variabili x ed y sono parzialmente strutturate in f. I loro schemi in f sono rispettivamente $\Sigma(x, f) := \{A\}$ e $\Sigma(y, f) := \{B\}$ se $x \neq y$; altrimenti, $\Sigma(x, f) := \{A, B\}$.

Ad esempio, se $f = x$(**FINE**) $\leq y$(**INIZIO**), allora le variabili x e y sono ben strutturate in f; inoltre, $\Sigma(x, f)$:= {**FINE**} e $\Sigma(y, f)$:= {**INIZIO**}.

A3. $f = x(A)\ \theta\ a$ oppure $f = a\ \theta\ x(A)$.
L'occorrenza della variabile x è libera in f. La variabile x è parzialmente strutturata in f. Lo schema di x in f è $\Sigma(x, f) := \{A\}$.

Ad esempio, se $f = y$(**SOVRANO**) = Carlo II, allora la variabile y è ben strutturata in f; inoltre, $\Sigma(x, f)$:= {**SOVRANO**}.

Passiamo ora alle formule non atomiche. Una condizione necessaria perché una formula non atomica sia ben formata è che ogni sua componente sia essa stessa una formula ben formata. Tale condizione è anche sufficiente per le formule non atomiche di tipo F2, $f = (g)$, e di tipo F3, $f = \neg(g)$. Per queste si ha inoltre che tutte le proprietà delle variabili in g restano immutate in f; vale a dire,

un'occorrenza di una variabile presente in g è libera in f se e solo se lo è in g,

una variabile con almeno un'occorrenza libera in f è ben strutturata o parzialmente strutturata se e solo se lo è in g,

lo schema di una variabile che abbia almeno un'occorrenza libera in f e che non sia mal strutturata in f è identico allo schema che essa ha in g.

Passiamo ora a considerare le formule non atomiche di tipo F4 ed F5.

F4. $f = (g)\quad (h)$ oppure $f = (g)\quad (h)$
dove le componenti g e h di f sono formule ben formate. Per ogni variabile x presente in f, un'occorrenza di x in f è libera se e solo se la corrispondente occorrenza di x in g o in h è libera.
Sia x una variabile con almeno un'occorrenza libera in f. Se le occorrenze libere di x in f sono tutte e solo le sue occorrenze libere in g (rispettivamente, in h), allora tutte le proprietà di x in f

verranno mutuate da *g* (rispettivamente, da *h*). Supponiamo invece che *x* abbia occorrenze libere sia in *g* che in *h*. Distinguiamo i seguenti quattro casi:

> (1) La variabile *x* è ben strutturata sia in *g* che in *h*. Allora *x* è ben strutturata in *f* se $\Sigma(x, g) = \Sigma(x, h)$; altrimenti, è mal strutturata. Se *x* è ben strutturata in *f*, allora il suo schema in *f* è definito come $\Sigma(x, f) := \Sigma(x, g)$.
>
> (2) La variabile *x* è ben strutturata in *g* ma lo è parzialmente in *h*. Allora *x* è ben strutturata in *f* se $\Sigma(x, h) \quad \Sigma(x, g)$; altrimenti, è mal strutturata. Se *x* è ben strutturata in *f*, allora il suo schema in *f* è definito come $\Sigma(x, f) := \Sigma(x, g)$.
>
> (3) La variabile *x* è ben strutturata in *h* ma lo è parzialmente in *g*. Allora *x* è ben strutturata in *f* se $\Sigma(x, g) \quad \Sigma(x, h)$; altrimenti, è mal strutturata. Se *x* è ben strutturata in *f*, allora il suo schema in *f* è definito come $\Sigma(x, f) := \Sigma(x, h)$.
>
> (4) La variabile *x* è parzialmente strutturata sia in *g* che in *h*. Allora *x* è parzialmente strutturata in *f* ed il suo schema in *f* è definito come $\Sigma(x, f) := \Sigma(x, g)$ $\Sigma(x, h)$.

La formula *f* è ben formata se nessuna variabile che abbia un'occorrenza libera in *f* è mal strutturata.

F5. $f = \quad x(R)\,(g)$ oppure $f = \quad x(R)\,(g)$

dove la componente *g* di *f* è una formula ben formata. Ogni occorrenza di *x* in *f* è legata (al quantificatore). Per ogni altra variabile *y* presente in *f*, le sue proprietà in *f* vengono mutuate da *g*. La formula *f* è ben formata se:

> *x* ha un'occorrenza libera in *g*,
> $\Sigma(x, g) = R$ nel caso che *x* sia ben strutturata in *g*,
> $\Sigma(x, g) \quad R$ nel caso che *x* sia parzialmente strutturata in *g*.

Esempio 2.1 (seguito). Consideriamo la formula *g*

inventario(*y*) $\quad$ *y*(**PARTE-DI**) = 211 $\quad$ *y*(**NOME**) = *x*(**NOME**)

Tutte le occorrenze delle variabili *x* ed *y* sono libere in *g*. Inoltre, *x* è parzialmente strutturata in *g* ed il suo schema è $\Sigma(x, g) = \{$**NOME**$\}$. La variabile *y* è ben strutturata in *g* ed il suo schema è uguale allo schema di **inventario**, cioè $\Sigma(y, g) = \{$**A#**, **PARTE-DI**, **NOME**$\}$. Dunque, la formula *g* è ben formata.
Consideriamo poi la formula *f*

y(**A#**, **PARTE-DI**, **NOME**) (*g*)

L'occorrenza della variabile *x* è libera in *f*, mentre le occorrenze della variabile *y* in *f* sono tutte legate al quantificatore esistenziale. La variabile *x* è parzialmente strutturata in *f* ed il suo schema in *f* è $\Sigma(x, f) = \Sigma(x, g) = \{$**NOME**$\}$. Dunque, anche la formula *f* è ben formata. ◊

Con queste definizioni, definiamo un'*espressione* del Calcolo Relazionale sulla base di dati relazionale ***D*** una qualsiasi stringa della forma

$\{x(R) \mid f\}$

dove

R è un sottoinsieme di *U*,
f è una formula ben formata,
x è l'unica variabile che ha occorrenze libere in *f* e
$\Sigma(x, f)$ $\quad$ *R*, dove è richiesta l'uguaglianza se *x* è ben strutturata in *f*.

2.1.4 Valutazione delle espressioni del CR

Sia $E = \{x(R) \mid f\}$ un'espressione del Calcolo Relazionale sulla base di dati ***D***, e sia *D* uno stato di ***D***. Il *valore* dell'espressione *E* calcolato *su D* è l'insieme delle ennuple appartenenti a *dom*(*R*) che, sostituite alle occorrenze libere di *x* in *f*, rendono la formula *f* vera. Resta da chiarire le modalità con cui viene valutata la formula *f* quando le occorrenze libere della variabile *x* vengono sostituite con un'ennupla con schema *R*.

In generale, parleremo di "sostituzione" di una variabile in una formula se quella variabile ha almeno un'occorrenza libera nella formula. Parleremo invece di "interpretazione" di una formula solo se nessuna variabile che vi compare ha occorrenze libere.

SOSTITUZIONE. Sia *f* una qualsiasi formula ben formata ed *x* una variabile con un'occorrenza libera di *f*. Se *x* è ben strutturata in *f*, allora potremo sostituire le occorrenze libere di *x* in *f* con una qualsiasi ennupla appartenente al dominio del suo

schema $\Sigma(x, f)$. Se invece x è solo parzialmente strutturata in f, allora potremo sostituire le occorrenze libere di x in f con una qualsiasi ennupla appartenente al dominio di un qualsiasi sottoinsieme di U che contenga lo schema $\Sigma(x, f)$. Esplicitamente, se indichiamo con T l'insieme delle ennuple che possono essere sostituite ad x, allora nel primo caso abbiamo

$$T = dom(\Sigma(x, f))$$

e nel secondo

$$T = \bigcap_{Y:\, \Sigma(x, f) \subseteq Y \subseteq U} dom(Y)$$

La *sostituzione* di un'occorrenza libera di x in f con un'ennupla t appartenente a T produce una formula, che indicheremo con $f(x)|_{x=t}$, la quale si ottiene modificando ogni formula atomica g in f che contenga un'occorrenza libera di x in f nella maniera seguente:

> Se $g = \boldsymbol{r}(x)$ allora g viene sostituito con il valore VERO se t appartiene alla relazione di nome $\boldsymbol{r}$ contenuta in D; altrimenti g viene sostituito con il valore FALSO.
>
> Se $g = x(A)\theta y(B)$ allora g viene sostituito con $t(A)\theta y(B)$. Lo stesso vale per $y(B)\theta x(A)$. Qualora poi sia $x=y$, allora g viene sostituito con il valore VERO se $t(A)\theta t(B)$, e con il valore FALSO in caso contrario.
>
> Se $g = x(A)\theta a$ allora g viene sostituito con il valore VERO se $t(A)\theta a$, e con il valore FALSO in caso contrario; analogamente, se $g = a\theta x(A)$.

A sostituzione avvenuta, la formula $f(x)|_{x=t}$ che si ottiene non contiene variabili con occorrenze libere; inoltre, se non fosse per la presenza delle costanti logiche VERO e FALSO, sarebbe ancora (e verrà trattata nel seguito alla stregua di) una formula ben formata.

Esempio 2.1 (seguito). Consideriamo la formula $f(x)$

$$y(\mathbf{A\#}, \mathbf{AEROSTAZIONE}, \mathbf{LIVELLO})\ (g(x, y))$$

dove $g(x, y)$ è la formula

$$\mathbf{inventario}(y) \quad y(\mathbf{PARTE\text{-}DI}) = 211 \quad y(\mathbf{NOME}) = x(\mathbf{NOME}).$$

Se t è il valore **fodera per poltrona** dell'attributo **NOME**, allora $f(x)|_{x=t}$ è

$$y(\mathbf{A\#}, \mathbf{AEROSTAZIONE}, \mathbf{LIVELLO})\ (g(x, y)|_{x=t})$$

dove $g(x, y)|_{x=t}$ è

inventario(y) y(**PARTE-DI**) = **211** y(**NOME**) = fodera per poltrona.

Consideriamo poi la formula $g(x, y)|_{x=t}$. Questa contiene occorrenze libere solo della variabile y, che possiamo sostituire ad esempio con una terna t' appartenente al dominio dell'insieme degli attributi {**A#**, **PARTE-DI**, **NOME**}. Se prendiamo t' = (**2116**, **211**, **cintura di sicurezza**) allora $g(x, y)|_{x=t,y=t'}$ è

VERO VERO FALSO ◊

INTERPRETAZIONE. Sia f una formula in cui ogni variabile che vi compare ha solo occorrenze legate ed in cui possono essere presenti le costanti logiche VERO e FALSO. L'*interpretazione* di f, indicata con $\mathsf{I}[f]$, è una costante logica definita ricorsivamente come segue:

1. Se f è una costante logica, allora $\mathsf{I}[f] = f$.

2. Se $f = (g)$, allora $\mathsf{I}[f] = \mathsf{I}[g]$.

3. Se $f = \neg(g)$ allora $\mathsf{I}[f] = \neg\mathsf{I}[g]$.

4. Se $f = (g)\ \ (h)$ oppure $f = (g)\ \ (h)$, allora nel primo caso si ha

$$\mathsf{I}[f] = \mathsf{I}[g]\ \ \mathsf{I}[h],$$

e nel secondo

$$\mathsf{I}[f] = \mathsf{I}[g]\ \ \mathsf{I}[h].$$

5. Se $f =\ x(R)(g)$ oppure $f =\ x(R)(g)$ (sicché x è l'unica variabile che ha occorrenze libere in g) allora nel primo caso si ha

$$\mathsf{I}[f] = \ _{t\ \ dom(R)}\ \mathsf{I}[g(x)|_{x=t}]$$

e nel secondo

$$\mathsf{I}[f] = \ _{t\ \ dom(R)}\ \mathsf{I}[g(x)|_{x=t}].$$

Vale la pena osservare che se la formula f è del tipo

$$f =\ x(R)\ (\boldsymbol{r}(x)\ \ g(x))$$

allora, indicato con r la relazione di nome $\boldsymbol{r}$ contenuta in D, l'interpretazione di f si semplifica come segue

$$I[f] = \bigvee_{t \in dom(R)} (I[\boldsymbol{r}(x)|_{x=t}] \wedge I[g(x)|_{x=t}]) =$$

$$= (\bigvee_{t \in r} (I[\boldsymbol{r}(x)|_{x=t}] \wedge I[g(x)|_{x=t}])) \vee (\bigvee_{t \in r} (I[\boldsymbol{r}(x)|_{x=t}] \wedge I[g(x)|_{x=t}])) =$$

$$= (\bigvee_{t \in r} (I[\text{VERO}] \wedge I[g(x)|_{x=t}])) \vee (\bigvee_{t \in r} (I[\text{FALSO}] \wedge I[g(x)|_{x=t}])) =$$

$$= (\bigvee_{t \in r} (\text{VERO} \wedge I[g(x)|_{x=t}])) \vee (\bigvee_{t \in r} (\text{FALSO} \wedge I[g(x)|_{x=t}])) =$$

$$= (\bigvee_{t \in r} I[g(x)|_{x=t}]) \vee (\bigvee_{t \in r} \text{FALSO}) =$$

$$= (\bigvee_{t \in r} I[g(x)|_{x=t}]) \vee \text{FALSO} =$$

$$= \bigvee_{t \in r} I[g(x)|_{x=t}]$$

Se invece la formula f è del tipo

$$f = \forall x(R) (\neg \boldsymbol{r}(x) \vee g(x))$$

allora, indicato con r la relazione di nome $\boldsymbol{r}$ contenuta in D, l'interpretazione di f si semplifica come segue

$$I[f] = \bigwedge_{t \in dom(R)} (\neg I[\boldsymbol{r}(x)|_{x=t}] \vee I[g(x)|_{x=t}]) =$$

$$= (\bigwedge_{t \in r} (\neg I[\boldsymbol{r}(x)|_{x=t}] \vee I[g(x)|_{x=t}])) \wedge (\bigwedge_{t \in r} (\neg I[\boldsymbol{r}(x)|_{x=t}] \vee I[g(x)|_{x=t}])) =$$

$$= (\bigwedge_{t \in r} (I[\text{FALSO}] \vee I[g(x)|_{x=t}])) \wedge (\bigwedge_{t \in r} (I[\text{VERO}] \vee I[g(x)|_{x=t}])) =$$

$$= (\bigwedge_{t \in r} (\text{FALSO} \vee I[g(x)|_{x=t}])) \wedge (\bigwedge_{t \in r} (\text{VERO} \vee I[g(x)|_{x=t}])) =$$

$$= (\bigwedge_{t \in r} I[g(x)|_{x=t}]) \wedge (\bigwedge_{t \in r} \text{VERO}) =$$

$$= (\bigwedge_{t \in r} I[g(x)|_{x=t}]) \wedge \text{VERO} =$$

$$= \bigwedge_{t \in r} I[g(x)|_{x=t}]$$

A questo punto sappiamo come interpretare, dato uno stato D della base di dati $\boldsymbol{D}$ ed un'espressione $E = \{x(R)|f\}$, la formula $f(x)|_{x=t}$ dove t è una qualsiasi ennupla appartenente a $dom(R)$. Chiamiamo *valore* di E su D l'insieme (eventualmente infinito) delle ennuple t appartenenti a $dom(R)$ per le quali $I[f(x)|_{x=t}] = \text{VERO}$.

2.2 L'APPROCCIO ALGEBRICO

L'interrogazione di una base di dati avviene formulando le domande mediante *espressioni algebriche*. A tale scopo, vengono usati degli operatori, detti *operatori*

relazionali, ognuno dei quali ha come argomento una o più relazioni e calcola una relazione.

2.2.1 Operatori relazionali

I primi operatori relazionali che consideriamo scaturiscono dalla natura insiemistica delle relazioni e sono gli operatori insiemistici di *unione* ( ), *intersezione* (∩) e *differenza* (–). Oltre agli operatori insiemistici, esistono specifici operatori relazionali che ora introduciamo.

SELEZIONE. L'operatore di selezione si applica ad un'unica relazione e genera un suo sottoinsieme. Sia *r* una relazione con schema *R*; una "formula proposizionale" *p* su *R* è definita ricorsivamente a partire da "proposizioni atomiche" e dai connettivi logici. Come al solito, indichiamo con Θ un insieme di operazioni di confronto.
Sia θ una operazione in Θ, siano *A* e *B* due attributi in *R* che siano θ-confrontabili, e sia *a* un elemento di *d*(*A*); le espressioni

$$A\ \theta\ B \qquad \text{e} \qquad A\ \theta\ a$$

sono *proposizioni atomiche su R.*
Ogni proposizione atomica è una *formula proposizionale.*
Se *p* e *q* sono formule proposizionali su *R*, allora lo sono anche

$$\neg\,(p) \qquad (p)\quad (q) \qquad (p)\quad (q).$$

Sia *t* un'ennupla appartenente a *dom*(*R*) e sia *p* una formula proposizionale su *R*; l'interpretazione di *p* relativa a *t* è una costante logica che indichiamo con *p*(*t*) ed è definita come segue. Se $p = A\ \theta\ B$, allora $p(t)$ = VERO se e solo se la condizione

$$t(A)\ \theta\ t(B)$$

è vera; analogamente, se $p = A\ \theta\ a$, allora $p(t)$ = VERO se e solo se la condizione

$$t(A)\ \theta\ a$$

è vera.
La *selezione* su *r* sotto la condizione *p* dà come risultato ancora una relazione con schema *R*, che indicheremo con $\sigma_p(r)$, così definita

$$\sigma_p(r) = \{t \quad r : p(t) = \text{VERO}\}.$$

È facile dimostrare che la selezione è

— commutativa rispetto alla composizione, cioè $\sigma_p(\sigma_q(r)) = \sigma_q(\sigma_p(r))$,

— distributiva rispetto all'unione ed all'intersezione:

$$\sigma_p(r \cup s) = \sigma_p(r) \cup \sigma_p(s)$$
$$\sigma_p(r \cap s) = \sigma_p(r) \cap \sigma_p(s).$$

PROIEZIONE. L'operatore di proiezione si applica ad un'unica relazione e genera una relazione. Sia r una relazione con schema R e sia X un sottoinsieme proprio di R. Se t è un'ennupla in r, chiameremo *restrizione* di t ad X la funzione t' definita su X tale che, per ogni A in X, $t'(A) = t(A)$. La restrizione dell'ennupla t ad X verrà indicata con $t[X]$. Si osservi poi che, se Y è un sottoinsieme proprio di X, la restrizione di t ad Y coincide con la restrizione di $t[X]$ ad Y, cioè $t[Y] = (t[X])[Y]$.
La *proiezione* di r su X è la relazione con schema X, che indicheremo con $\pi_X(r)$, la quale contiene le restrizioni ad X delle ennuple in r, cioè

$$\pi_X(r) = \{t \in dom(X) : t' \in r \text{ tale che } t'[X] = t\}.$$

Esempio 2.1 La proiezione della Tabella 1.1 su {**DATA**, **ORA**} è

DATA	**ORA**
6 giugno	10:30
7 giugno	10:30
9 giugno	12:00
10 giugno	10:30

e la sua proiezione su {**VOLO**, **PILOTA**} è

VOLO	**PILOTA**
112	Rossi
112	Verdi
200	Verdi

◊

PRODOTTO JOIN. A differenza della proiezione, il prodotto join si applica ad una coppia di relazioni ed il risultato è ancora una relazione. Siano r_1 ed r_2 due relazioni rispettivamente con schemi R_1 ed R_2. Il *prodotto join* di r_1 ed r_2 è una relazione con schema $R = R_1 \cup R_2$, che indicheremo con $r_1 \bowtie r_2$, così definita

$$r_1 \bowtie r_2 = \{t \in dom(R) : t_1 \in r_1 \text{ ed } t_2 \in r_2 \text{ tali che } t_1 = t[R_1] \text{ e } t_2 = t[R_2]\}.$$

Si osservi che, se $R_1 \cap R_2 = \emptyset$, allora il prodotto join di r_1 ed r_2 è una sorta di prodotto cartesiano ($\times$) di r_1 ed r_2 e l'ennupla t ottenuta combinando le due ennuple t_1 in r_1 e t_2 in r_2 — cioè tale che $t_1 = t[R_1]$ e $t_2 = t[R_2]$ — la indichiamo con $t_1 \times t_2$. Il prodotto join gode della proprietà commutativa e di quella associativa. Quest'ultima proprietà permette di definire senza ambiguità il prodotto join di tre o più relazioni.

DIVISIONE. Siano r ed s due relazioni rispettivamente con schemi R ed S, dove S è un sottoinsieme proprio di R. La *divisione* di r per s dà come risultato una relazione con schema $R–S$, che indicheremo con $r \div s$, la quale contiene tutte e sole le ennuple t con schema $R–S$ tali che, per ogni ennupla t' in s, l'ennupla $t \times t'$ appartiene alla relazione r, cioè,

$$r \div s = \{t \quad dom(R–S) : \quad t' \quad s \quad t \times t' \quad r\}.$$

La relazione $r \div s$ è chiamata il *quoziente* della divisione di r per s; essa può essere anche definita in termini degli operatori già visti nei modi seguenti

$$r \div s = \pi_{R–S}(r) – \pi_{R–S}((\pi_{R–S}(r) \quad s) – r)$$

$$r \div s = \cap_{v \quad s} \pi_{R–S}(\sigma_{S=v}(r))$$

RIDENOMINAZIONE. Siano r una relazione con schema R, A un attributo in R e B un attributo non appartenente ad R con $d(A) = d(B)$. Allora, r con A *ridenominato* B è una relazione con schema $R–\{A\} \quad \{B\}$, che indicheremo con $\delta_{A \leftarrow B}(r)$, così definita

$$\delta_{A \leftarrow B}(r) = \{t \quad dom(R–\{A\} \quad \{B\}) : \quad t' \quad r \text{ tale che } t'[R–\{A\}] = t[R–\{A\}] \quad t'(A) = t(B)\}.$$

Infine, introduciamo alcuni operatori che si ottengono per estensione degli operatori di selezione e di prodotto join, purché tra gli elementi di uno o più domini sia possibile effettuare dei confronti che non siano la pura eguaglianza. Siano $\theta_1, \ldots, \theta_k$ operazioni di confronto (non necessariamente distinte) nell'insieme Θ, e siano r ed s due relazioni rispettivamente con schemi R ed S disgiunti. Se $A_1, \ldots, A_k$ sono attributi in R e $B_1, \ldots, B_k$ sono attributi in S tali che gli attributi A_h e B_h siano θ_h-confrontabili, $1 \leq h \leq k$, allora la forma generalizzata del prodotto join è

$$r\ [A_1\theta_1B_1, \ldots, A_k\theta_kB_k]\ s = \{t \quad dom(R \quad S) : \quad t' \quad r, t'' \quad s \text{ tali che } t' = t[R] \text{ e } t'' = t[S] \text{ e } t'(A_1)\theta_1t''(B_1) \text{ e } \ldots\ t'(A_k)\theta_kt''(B_k)\}\ .$$

Questo nuovo operatore relazionale è chiamato *prodotto theta-join* (θ-*join*). Nel caso particolare che ogni θ_h sia l'operazione di eguaglianza, allora il prodotto theta-join prende il nome di *prodotto equi-join.* Si osservi che il prodotto theta-join si può esprimere mediante gli operatori di prodotto join e di selezione; ad esempio

$$r\ [A\theta B]\ s = \sigma_{A\theta B}(r \quad s).$$

Viceversa, il prodotto join di due relazioni con schemi non disgiunti si può esprimere utilizzando gli operatori di prodotto equi-join, di ridenominazione e di proiezione.

Siano r ed s due relazioni rispettivamente con schemi R ed S e sia $R \cap S = \{A_1, \ldots, A_k\}$; allora si ha

$$r \quad s = \pi_{R \quad S}(r\ [A_1 = B_1, \ldots, A_k = B_k]\ (\delta_{A_1 \leftarrow B_1}(\ldots(\delta_{A_k \leftarrow B_k}(s)))))$$

2.2.2 Algebra Relazionale

Per definire un'espressione dell'Algebra Relazionale sulla base di dati relazionale $\boldsymbol{D}$ abbiamo ancora bisogno di

- Θ un insieme di operazioni di confronto tra valori di attributi che comprende almeno l'eguaglianza e la diseguaglianza, le quali sono definite per ciascun dominio in Δ,
- O l'insieme dei seguenti operatori relazionali

  unione,
  intersezione,
  differenza,
  proiezione,
  prodotto join,
  divisione,
  ridenominazione,
  selezione con operazioni in Θ,
  connettivi logici,
  prodotto theta-join con operazioni in Θ.

Un'espressione dell'Algebra Relazionale su $\boldsymbol{D}$ si ottiene a partire dalle variabili relazionali in $\boldsymbol{D}$ e da costanti relazionali (cioè, relazioni anonime aventi per schemi sottoinsiemi di U) utilizzando gli operatori in O e le parentesi. La loro definizione, che ora daremo, è di natura ricorsiva e specifica per ognuna di esse anche un insieme di attributi che prende il nome di *schema* dell'espressione.

1. Se $\boldsymbol{r}$ è una variabile relazionale in $\boldsymbol{D}$, allora $\boldsymbol{r}$ è un'espressione che ha per schema lo schema di $\boldsymbol{r}$.

2. Se r è una costante relazionale con schema R, allora r è un'espressione il cui schema è R.

3. Se E è un'espressione con schema R e p è una formula proposizionale su R, allora $\sigma_p(E)$ è un'espressione con schema R.

3. Se E_1 ed E_2 sono espressioni entrambi con schema R, allora

$E_1 \quad E_2$ è un'espressione con schema R,

$E_1 \cap E_2$ è un'espressione con schema R,

$E_1 - E_2$ è un'espressione con schema R.

5. Se E è un'espressione con schema R ed X è un sottoinsieme non vuoto di R, allora

$$\pi_X(E)$$

è un'espressione con schema X.

6. Se E_1 ed E_2 sono espressioni con schemi R_1 ed R_2 e se R_2 è un sottoinsieme proprio di R_1, allora

$$E_1 \div E_2$$

è un'espressione con schema $R_1 - R_2$.

7. Se E_1 ed E_2 sono espressioni con schemi R_1 ed R_2 allora

$$E_1 \quad E_2$$

è un'espressione con schema $R_1 \quad R_2$.
Se poi $R_1 \cap R_2 = \emptyset$ e se $A_1, \ldots, A_k$ sono attributi in R_1 e $B_1, \ldots, B_k$ sono attributi in R_2 tali che gli attributi A_h e B_h sono θ_h-confrontabili, $1 \leq h \leq k$, allora

$$E_1 \; [A_1 \theta_1 B_1, \ldots, A_k \theta_k B_k] \; E_2$$

è un'espressione con schema $R_1 \quad R_2$.

8. Se E è un'espressione con schema R e se l'insieme $\{A_1, \ldots, A_k\}$ è un sottoinsieme di R e l'insieme $\{B_1, \ldots, B_k\}$ ha intersezione vuota con R, allora

$$\delta_{A_1 \leftarrow B_1, \ldots, A_k \leftarrow B_k}(E)$$

è un'espressione con schema $R - \{A_1, \ldots, A_k\} \quad \{B_1, \ldots, B_k\}$.

Sia $D = \{r_1, \ldots, r_n\}$ uno stato della base di dati $\boldsymbol{D}$, dove r_i è una relazione con schema R_i, e sia E un'espressione dell'Algebra Relazionale su $\boldsymbol{D}$. Il *valore* di E su D è la relazione che ha per schema lo schema di E e che si ottiene applicando gli operatori in E alle relazioni contenute in D i cui nomi figurano in E.

Esempio 2.1 (seguito). Per la domanda "Quali sono le parti dell'articolo di codice 211?" possiamo usare la seguente espressione dell'Algebra Relazionale:

$$E = \pi_{\mathbf{NOME}}(\sigma_{\mathbf{PARTE\text{-}DI}=211}(\mathbf{inventario}))$$

Per la domanda "Quante poltrone di II classe sono usate sul vettore 727?" possiamo usare l'espressione E dell'Algebra Relazionale così definita. Siano

$$E_1 = \mathbf{inventario} \quad \mathbf{dotazione}$$
$$E_2 = \sigma_{\mathbf{NOME}=\text{poltrona di II classe} \quad \mathbf{VETTORE}=727}(E_1)$$

ed

$$E = \pi_{\mathbf{NUMERO}}(E_2)$$

Esempio 2.2 (seguito). Per la domanda "Chi furono il primo re e la prima regina?" possiamo usare l'espressione E dell'Algebra Relazionale così definita. Siano

$$E_1 = \pi_{\mathbf{SOVRANO},\mathbf{INIZIO}}(\mathbf{regno})$$
$$E_2 = \pi_{\mathbf{NOME},\mathbf{SESSO}}(\mathbf{dinastia})$$
$$E_3 = E_1[\mathbf{SOVRANO}=\mathbf{NOME}]E_2$$
$$E_4 = \pi_{\mathbf{SOVRANO},\mathbf{INIZIO},\mathbf{SESSO}}(E_3)$$
$$E_5 = \delta_{\mathbf{SOVRANO}\leftarrow\mathbf{NOME},\ \mathbf{INIZIO}\leftarrow\mathbf{I},\ \mathbf{SESSO}\leftarrow\mathbf{S}}(E_4)$$
$$E_6 = \sigma_{\mathbf{SESSO}\neq\mathbf{S} \quad \mathbf{INIZIO}\leq\mathbf{I}}(E_4 \ E_5)$$
$$E_7 = E_6 \div E_5$$
$$E = \pi_{\mathbf{SOVRANO}}(E_7)$$

Allora, abbiamo che sullo stato corrente della Base di Dati Dinastici (vedi Capitolo 1) i valori delle espressioni E_1, …, E_7 sono le relazioni riportate rispettivamente nelle Tabelle 2.6, …, 2.12.

SOVRANO	INIZIO
Giacomo I	1603
Carlo I	1625
Carlo II	1660
Giacomo II	1685
Maria II	1688
Anna	1702

Tabella 2.6 Il valore di E_1

NOME	SESSO
Giacomo I	M
Elisabetta	F
Carlo I	M
Carlo II	M
Maria	F
Giacomo II	M
Enrichetta A.	F
Maria II	F
Anna	F
Giacomo E.	M

Tabella 2.7 Il valore di E_2

SOVRANO	INIZIO	NOME	SESSO
Giacomo I	1603	Giacomo I	M
Carlo I	1625	Carlo I	M
Carlo II	1660	Carlo II	M
Giacomo II	1685	Giacomo II	M
Maria II	1688	Maria II	F
Anna	1702	Anna	F

Tabella 2.8 Il valore di E_3

SOVRANO	INIZIO	SESSO
Giacomo I	1603	M
Carlo I	1625	M
Carlo II	1660	M
Giacomo II	1685	M
Maria II	1688	F
Anna	1702	F

Tabella 2.9 Il valore di E_4

NOME	I	S
Giacomo I	1603	M
Carlo I	1625	M
Carlo II	1660	M
Giacomo II	1685	M
Maria II	1688	F
Anna	1702	F

Tabella 2.10 Il valore di E_5

SOVRANO	INIZIO	SESSO	NOME	I	S
Giacomo I	1603	M	Giacomo I	1603	M
Giacomo I	1603	M	Carlo I	1625	M
Giacomo I	1603	M	Carlo II	1660	M
Giacomo I	1603	M	Giacomo II	1685	M
Giacomo I	1603	M	Maria II	1688	F
Giacomo I	1603	M	Anna	1702	F
Carlo I 1625	M		Carlo I	1625	M
Carlo I 1625	M		Carlo II	1660	M
Carlo I 1625	M		Giacomo II	1685	M
Carlo I 1625	M		Maria II	1688	F
Carlo I 1625	M		Anna	1702	F
Carlo II1660	M		Carlo II	1660	M
Carlo II1660	M		Giacomo II	1685	M
Carlo II1660	M		Maria II	1688	F
Carlo II1660	M		Anna	1702	F
Giacomo II	1685	M	Giacomo II	1685	M
Giacomo II	1685	M	Maria II	1688	F
Giacomo II	1685	M	Anna	1702	F
Maria II1688	F		Giacomo I	1603	M
Maria II1688	F		Carlo I	1625	M
Maria II1688	F		Carlo II	1660	M
Maria II1688	F		Giacomo II	1685	M
Maria II1688	F		Maria II	1688	F
Maria II1688	F		Anna	1702	F
Anna	1702	F	Giacomo I	1603	M
Anna	1702	F	Carlo I	1625	M
Anna	1702	F	Carlo II	1660	M
Anna	1702	F	Giacomo II	1685	M
Anna	1702	F	Anna	1702	F

Tabella 2.11 Il valore di E_6

SOVRANO	INIZIO	SESSO
Giacomo I	1603	M
Maria II	1688	F

Tabella 2.12 Il valore di E_7

A questo punto, è chiaro che il valore di E su D è proprio la relazione riportata nella Tabella 2.4.

Per la domanda "Chi fu l'ultimo sovrano ed in che anno nacque?" possiamo usare l'espressione E dell'Algebra Relazionale così definita. Siano

$E_1 = \pi_{\mathbf{SOVRANO,INIZIO}}(\mathbf{regno})$
$E_2 = \delta_{\mathbf{SOVRANO} \leftarrow \mathbf{S},\ \mathbf{INIZIO} \leftarrow \mathbf{I}}(E_1)$
$E_3 = \sigma_{\mathbf{INIZIO} \geq \mathbf{I}}(E_1 \ E_2)$
$E_4 = E_3 \div E_2$
$E_5 = \pi_{\mathbf{NOME,NASCITA}}(\mathbf{dinastia})$
$E_6 = (\pi_{\mathbf{SOVRANO}}(E_4))[\mathbf{SOVRANO} = \mathbf{NOME}]E_5$
$E = \pi_{\mathbf{SOVRANO,NASCITA}}(E_6)$

Allora, abbiamo che i valori delle espressioni E_1, …, E_6 sono le relazioni riportate rispettivamente nelle Tabelle 2.13, …, 2.18.

SOVRANO	INIZIO
Giacomo I	1603
Carlo I	1625
Carlo II	1660
Giacomo II	1685
Maria II	1688
Anna	1702

Tabella 2.13 Il valore di E_1

S	I
Giacomo I	1603
Carlo I	1625
Carlo II	1660
Giacomo II	1685
Maria II	1688
Anna	1702

Tabella 2.14 Il valore di E_2

SOVRANO	INIZIO	S	I	
Giacomo I	1603		Giacomo I	1603
Carlo I	1625		Giacomo I	1603
Carlo I	1625		Carlo I	1625
Carlo II	1660		Giacomo I	1603
Carlo II	1660		Carlo I	1625
Carlo II	1660		Carlo II	1660
Giacomo II	1685		Giacomo I	1603
Giacomo II	1685		Carlo I	1625
Giacomo II	1685		Carlo II	1660
Giacomo II	1685		Giacomo II	1685
Maria II	1688		Giacomo I	1603
Maria II	1688		Carlo I	1625
Maria II	1688		Carlo II	1660
Maria II	1688		Giacomo II	1685
Maria II	1688		Maria II	1688
Anna	1702		Giacomo I	1603
Anna	1702		Carlo I	1625
Anna	1702		Carlo II	1660
Anna	1702		Giacomo II	1685
Anna	1702		Maria II	1688
Anna	1702		Anna	1702

Tabella 2.15 Il valore di E_3

SOVRANO	INIZIO
Anna	1702

Tabella 2.16 Il valore di E_4

NOME	NASCITA
Giacomo I	566
Elisabetta	590
Carlo I	1600
Carlo II	1630
Maria	1631
Giacomo II	1633
Enrichetta A.	1640
Maria II	1662
Anna	1665
Giacomo E.1686	

Tabella 2.17 Il valore di E_5

SOVRANO	NOMENASCITA	
Anna	Anna	1665

Tabella 2.18 Il valore di E_6

A questo punto, è chiaro che il valore di E su D è proprio la relazione riportata nella Tabella 2.5. ◊

2.3 POTERE ESPRESSIVO

I due linguaggi di interrogazione che abbiamo presentato nei paragrafi precedenti possono essere confrontati in base al loro "potere espressivo". In generale, dati due linguaggi di interrogazione L ed L' su una base di dati $\boldsymbol{D}$, diremo che L è *espressivo* almeno quanto L' se per ogni espressione E' di L' esiste un'espressione E di L equivalente ad E' nel senso che, per ogni stato D di $\boldsymbol{D}$, i valori di E e di E' su D coincidono. Dimostriamo ora che il Calcolo Relazionale è espressivo almeno quanto l'Algebra Relazionale per induzione sul numero degli operatori.

BASE. La seguente tabella riporta per ognuna delle espressioni dell'Algebra Relazionale priva di operatori un'espressione equivalente del Calcolo Relazionale

Algebra Relazionale	Calcolo Relazionale
$A\theta B$	$\{x(A, B) \mid x(A)\theta x(B)\}$
$A\ \theta\ a$	$\{x(A) \mid x(A)\theta a\}$
$\boldsymbol{r}$	$\{x(R) \mid \boldsymbol{r}(x)\}$

dove R è lo schema della variabile relazionale $\boldsymbol{r}$ (in $\boldsymbol{D}$).

PASSO INDUTTIVO. Assumiamo che la cosa sia vera per ogni espressione dell'Algebra Relazionale con meno di k operatori, $k \geq 1$, e sia E un'espressione dell'Algebra Relazionale con k operatori. È sufficiente considerare i seguenti casi:

Caso 1. $E = \delta_{A \leftarrow B}(E_1)$

Caso 2. $E = \sigma_{A\theta B}(E_1)$ oppure $E' = \sigma_{A\theta a}(E_1)$

Caso 3. $E = \pi_S(E_1)$

Caso 4. $E = E_1 \quad E_2$

Caso 5. $E = E_1 \quad E_2$

Caso 6. $E = E_1 \cap E_2$

Caso 7. $E = E_1 - E_2$

Siano R_i lo schema di E_i, $i = 1, 2$, ed f_i la formula usata in un'espressione del Calcolo Relazionale equivalente ad E_i. Si noti che nei casi 5, 6 e 7 è $R_1 = R_2$. Allora, un'espressione del Calcolo Relazionale equivalente ad E è rispettivamente:

Caso 1. $\{x(R) \mid \quad y(R_1)\ f_1(y) \quad g(y, x)\}$, dove

$R = R_1 - \{A\} \quad \{B\}$ e $g(y, x) = (\ _{C \quad R_1 - \{A\}}\ y(C)=x(C)) \quad (y(A)=x(B))$.

Caso 2. $\{x(R_1) \mid\ f_1(x) \quad x(A)\theta x(B)\}$ e $\{x(R_1) \mid\ f_1(x) \quad x(A)\theta a\}$

Caso 3. $\{x(S) \mid \quad y(R_1)\ f_1(y) \quad g(y, x)\}$, dove

$g(y, x) = (\ _{A \quad S}\ y(A)=x(A))$.

Caso 4. $\{x(R_1 \quad R_2) \mid \quad y(R_1)\ f_1(y) \quad g(y,x) \quad z(R_2)\ f_2(z) \quad h(z, x)\}$, dove

$g(y, x) = \ _{A \quad R_1}\ y(A)=x(A)$ ed $h(z, x) = \ _{A \quad R_2}\ z(A)=x(A)$.

Caso 5. $\{x(R_1) \mid f_1(x) \quad f_2(x)\}$

Caso 6. $\{x(R_1) \mid f_1(x) \quad f_2(x)\}$

Caso 7. $\{x(R_1) \mid f_1(x) \quad (\neg f_2(x))\}$

Abbiamo così provato che il Calcolo Relazionale è espressivo almeno quanto l'Algebra Relazionale. In realtà, il Calcolo Relazionale è più espressivo dell'Algebra Relazionale, perché esistono espressioni del Calcolo Relazionale come

$\{x(R) \mid \neg \boldsymbol{r}(x)\}$

che non hanno equivalenti nell'Algebra Relazionale. Va però osservato che la superiorità del Calcolo Relazionale rispetto all'Algebra Relazionale è in qualche modo artificiosa. Il punto è che il valore di un'espressione dell'Algebra Relazionale sullo stato *D* della base di dati è sempre una relazione dove compaiono solo valori degli attributi che o sono già presenti in *D* o sono costanti introdotte dall'espressione, mentre questo può non verificarsi per un'espressione del Calcolo Relazionale. Pertanto, il valore di un'espressione dell'Algebra Relazionale dipende dai domini degli attributi solo per il tramite dello stato *D* della base di dati su cui essa viene valutata; mentre il valore di un'espressione del Calcolo Relazionale potrebbe contenere valori di un attributo che non compaiono in *D*. Per tornare all'esempio precedente, consideriamo la relazione *r* riportata nella Tabella 2.19

A	**B**
0	1
1	0

Tabella 2.19

e supponiamo che essa sia il valore della variabile relazionale ***r*** definita nella base di dati ***D*** in cui entrambi attributi **A** e **B** hanno assegnato il dominio {0, 1}. Allora il valore dell'epressione $\{x(\mathbf{A}, \mathbf{B}) \mid \neg \boldsymbol{r}(x)\}$ su *r* è dato dalla relazione riportata nella Tabella 2.20

A	**B**
0	0
1	1

Tabella 2.20

Ma se la stessa variabile ***r*** fosse definita in un'altra base di dati ***D'*** che differisse da ***D*** solo perché i domini degli attributi **A** e **B** sono ora entrambi {0, 1, 2}, allora il valore dell'epressione sempre su *r* sarebbe la relazione riportata nella Tabella 2.21.

A	B
0	0
0	2
1	1
1	2
2	0
2	1
2	2

Tabella 2.21

Per eliminare questo tipo di dipendenza del valore di un'espressione del Calcolo Relazionale dalla definizone dei domini all'interno della base di dati, si usa la nozione di "dominio attivo" di un attributo. Dato uno stato D di una base di dati relazionale $\boldsymbol{D}$, per ogni attributo A in U, si chiama *dominio attivo* di A rispetto a D il sottoinsieme $d(A)$ di $d(A)$ formato dai valori che assume l'attributo A nelle relazioni in D i cui schemi contengano l'attributo A. Inoltre, per un qualsiasi sottoinsieme R di U, chiamiamo *dominio attivo* di R rispetto a D l'insieme delle ennuple t con schema R tali che $t(A)$ appartenga al dominio attivo $d(A)$ di A. Se indichiamo con $dom(R)$ il dominio attivo di R rispetto a D, abbiamo

$$dom(R) = \{t \quad dom(R) : \quad A \quad R\, t(A) \quad d(A)\}.$$

Data poi un'espressione del Calcolo Relazionale $E = \{x(R) \mid f(x)\}$, chiamiamo *valore attivo* di E su D la relazione (!) fomata dalle ennuple t appartenenti a $dom(R)$ per le quali l'interpretazione di $f(x)|_{x=t}$ dà il valore VERO, avendo cura di sostituire anche le altre variabili eventualmente presenti in f (ad esempio, quelle quantificate) con ennuple appartenenti ai domini attivi dei loro schemi.

La nozione di dominio attivo di un attributo è anche utilizzata per arricchire il potere espressivo dell'Algebra Relazionale quando si introduca l'operatore di "complementazione attiva". Per una relazione r con schema R, la *complementazione attiva* di r è la relazione

$$\bar{r} = dom(R) - r.$$

Così, abbiamo un nono tipo di espressione algebrica:

9. Se E è un'espressione con schema R, allora $\bar{E}$ è un'espressione con schema R.

Prima chiudere questo capitolo, vogliamo osservare che un limite dei due linguaggi di interrogazione qui esaminati è nell'impossibilità di esprimere certe domande come quella che richiede la chiusura transitiva di una relazione binaria (ad esempio, padre-figlio). Così, non possiamo con i linguaggi visti ottenere da una base di dati araldici un'intera linea di discendenza di una persona, ma solo la sua linea di discendeza fino ad un livello (arbitrariamente) prefissato.

CAPITOLO 3

AGGIORNAMENTO DI UNA BASE DI DATI

3.1 Operazioni di aggiornamento

Nella maggior parte dei casi una base di dati è soggetta a continue operazioni di aggiornamento affinché le informazioni in essa contenute siano sempre attendibili. Ogni operazione di aggiornamento ha per oggetto una singola variabile relazionale e consiste o nell'eliminazione di un'ennupla, oppure nell'aggiunta di un'ennupla oppure ancora nella modifica dei valori di uno o più attributi in un'ennupla. Vediamo ora come si esprimono le singole operazioni di aggiornamento supposto che la variabile relazionale in oggetto sia ***r***, che $R = \{A_1, \ldots, A_n\}$ sia lo schema di ***r*** e che r sia la relazione di nome ***r*** contenuta nello stato attuale della base di dati.

AGGIUNTA. L'aggiunta dell'ennupla $t = \{t(A_1) = a_1, \ldots, t(A_n) = a_n\}$ si esprime come

aggiungere (***r***; $A_1 = a_1, \ldots, A_n = a_n$).

CANCELLAZIONE. La cancellazione dell'ennupla $t = \{t(A_1) = a_1, \ldots, t(A_n) = a_n\}$ si esprime come

cancellare (***r***; $A_1 = a_1, \ldots, A_n = a_n$).

MODIFICA. La modifica nell'ennupla $t = \{t(A_1) = a_1, \ldots, t(A_n) = a_n\}$ dei valori degli attributi $B_1, \ldots, B_m$, dove $\{B_1, \ldots, B_m\}$ è un sottoinsieme di R, si esprime come

cambiare (***r***; $A_1 = a_1, \ldots, A_n = a_n$; $B_1 = b_1, \ldots, B_m = b_m$).

Una tale operazione di aggiornamento è *sintatticamente corretta* se i valori assegnati ai singoli attributi appartengono ai rispettivi domini e se:

nel caso dell'aggiunta, l'ennupla t effettivamente non compare in r, e

nel caso della cancellazione o della modifica, l'ennupla t effettivamente compare in r.

Dopo aver effettuato i controlli di correttezza sintattica, l'operazione di aggiornamento verrà eseguita ed il risultato diventerà la relazione di nome ***r*** nel nuovo stato della base di dati. La procedura *standard* per eseguire un'operazione di aggiornamento consiste nell'aggiungere, cancellare o modificare l'ennupla t dalla relazione di nome ***r*** contenuta nello stato attuale della base di dati. (Si osservi che in questo modo una modifica è nient'altro che una cancellazione seguita da un'aggiunta.)

Tuttavia, il risultato dell'applicazione della procedura standard può essere una relazione inaccettabile sotto il profilo semantico se gli attributi presenti nello schema R di $\boldsymbol{r}$ hanno proprietà semantiche che pongono dei vincoli sui possibili valori di $\boldsymbol{r}$. Allora, come vedremo, il risultato dell'applicazione della procedura standard sarà sottoposto ad un esame semantico, che se non è superato porterà ad una "rettifica" della relazione ottenuta.

Un tipo di vincolo semantico (ed è l'unico che prenderemo in considerazione) è la *dipendenza funzionale* di un sottoinsieme X dello schema R di $\boldsymbol{r}$ da un altro sottoinsieme Y di R: tale vincolo, che scriviamo $X \rightarrow Y$ e leggiamo "*X determina Y*", impone che nessun valore di $\boldsymbol{r}$ possa contenere due ennuple t_1 e t_2 tali che

$$t_1[X] = t_2[X] \qquad \text{e} \qquad t_1[Y] \neq t_2[Y].$$

In altri termini, la dipendenza funzionale $X \rightarrow Y$ richiede che i valori accettabili di $\boldsymbol{r}$ sono tutte e solo le relazioni r con schema R tali che, per ogni elemento x di $dom(X)$, la relazione $\pi_Y(\sigma_{X=x}(r))$ contiene al più un'ennupla. Diremo infine che una dipndenza funzionale $X \rightarrow Y$ è semplice se $|Y| = 1$.

Vediamo ora in dettaglio come si può decidere se un valore r di $\boldsymbol{r}$ sia o meno accettabile. Il seguente algoritmo fornisce la risposta corretta in termini della variabile logica *test* che assumerà il valore VERO se e solo se r soddisfa la dipendenza funzionale $X \rightarrow Y$.

(1) *test* := VERO.

(2) Con il criterio lessicografico ordinare le ennuple in r usando X come chiave dell'ordinamento. Sia L la lista risultante.

(3) Per ogni coppia t e t' di elementi consecutivi della lista L,

se $t'[X] = t[X]$ e $t'[Y] \neq t[Y]$, allora *test* := FALSO ed Uscire.

Si osservi che, se $X \cup Y = R$, l'Istruzione 3 può semplificarsi come segue

(3) Per ogni coppia t e t' di elementi consecutivi della lista L,

se $t'[X] = t[X]$, allora *test* := FALSO ed Uscire.

Dal punto di vista computazionale, l'algoritmo richiede l'ordinamento lessicografico di $|r|$ oggetti basato sulla chiave X dell'ordinamento lessicografico (Istruzione 2) e, poi, al più $(|r|-1)$ $(|X|+|Y|)$ operazioni di confronto (Istruzione 3). Pertanto, il costo

dell'Istruzione 2 è una funzione crescente sia di $|r|$ che di $|X|$, ed il costo dell'Istruzione 3 sarà una funzione crescente di $|r|$ e di $|X|+|Y|$. Dunque, la complessità dell'algoritmo è una funzione crescente di $|r|$ e di $|X|+|Y|$.

Nel primo degli esempi che ora discutiamo, vedremo che il risultato dell'applicazione della procedura standard è un valore inaccettabile ed il motivo è dovuto alla presenza di certi attributi composti che sono "sovradimensionati". Un attributo è *composto* se il suo dominio è formato da ennuple. Un tipico esempio è l'attributo **DATA** i cui valori possono essere terne, in cui le componenti sono i valori di giorni, mesi ed anni. Sia A un attributo composto con dominio $d(A)$. Se A appartiene ad uno schema R, diremo che A è *sovradimensionato* se

— esiste un attributo B in R tale che $A \rightarrow B$, ed

— è possibile definire due attributi A_1 ed A_2 tali che

$$dom(\{A_1, A_2\}) = d(A) \qquad \text{e} \qquad A_1 \rightarrow B.$$

In altre parole, A è sovradimensionato se la dipendenza funzionale di B da A è solo dovuta ad A_1.

Esempio 3.1 La variabile relazionale **compleanno** ha schema {**NOME**, **DATA**, **SEGNO**}. Assumiamo che l'attributo **NOME** determini (com'è naturale) **DATA** e **SEGNO** e, inoltre, che la **DATA** determini il **SEGNO**: **NOME** → {**DATA**, **SEGNO**} e **DATA** → **SEGNO**. La Tabella 3.1 riporta un valore di **compleanno** che soddisfa entrambi i vincoli.

NOME	DATA	SEGNO
Alfredo	7 giugno 1949	Gemelli
Maria	14 Luglio 1963	Cancro
Vincenzo	30 Gennaio 1959	Aquario
Giuseppe	30 Gennaio 1959	Aquario

Tabella 3.1 La relazione attuale di nome **compleanno**

Supponiamo ora di aggiungere l'ennupla

Carla	7 giugno 1968	Leone

Si osservi che la relazione che si ottiene applicando la procedura standard ancora soddisfa entrambi i vincoli. Tuttavia, **Alfredo** e **Carla** pur festeggiando il compleanno lo stesso giorno dello stesso mese hanno distinti segni zodiacali. La ragione è che **DATA** è un attributo composto sovradimensionato e che solo una sua parte (giorno e mese) determina **SEGNO**. Meglio sarebbe stato se alla variabile **compleanno** si fosse assegnato lo schema {**NOME**, **GM**, **A**} dove l'attributo **GM** contiene il giorno ed il mese e l'attributo **A** contiene l'anno. ◊

Eliminare attributi composti sovradimensionati è cosa facile. Tutt'altro che facile è l'anomalia causata da certe dipendenze funzionali che ora esamineremo prima con un esempio e, poi, in generale.

Esempio 3.2 Consideriamo la Base di Dati Aeroportuali e la variabile relazionale **partenze**. Si ricordi che essa ha schema {**VOLO**, **DATA**, **PILOTA**, **ORA**}. Supponiamo che come valori accettabili di **partenze** si prendano tutte le relazioni con schema {**VOLO**, **DATA**, **PILOTA**, **ORA**} che soddisfano le tre dipendenze funzionali:

1. «Ogni volo ha un'ora fissata»
   VOLO → **ORA**

2. «Esiste un solo volo con un dato pilota, in un dato giorno ad una data ora»
   {**DATA**, **PILOTA**, **ORA**} → **VOLO**.

3. «C'è un solo pilota di un dato volo in un dato giorno»
   {**VOLO**, **DATA**} → **PILOTA**.

La Tabella 3.2 riporta la relazione di nome **partenze** contenuta nello stato attuale della base di dati.

VOLO	DATA	PILOTA	ORA
1126	giugno	Rossi	10:30
1127	giugno	Verdi	10:30
2009	giugno	Verdi	12:00
11210	giugno	Rossi	10:30

Tabella 3.2 Un valore della variabile relazionale **partenze**

Si noti che essa soddisfa tutte e tre le dipendenze funzionali. Supponiamo che la si voglia modificare con l'operazione

cambiare (**partenze**; **VOLO** = 112, **DATA** = 6 giugno, **ORA** = 10:30; **ORA** = 9:30).

Dopo aver eseguita questa operazione con la procedura standard, avremo la relazione riportata in Tabella 3.3

VOLO	DATA	PILOTA	ORA
112	6 giugno	Rossi	9:30
112	7 giugno	Verdi	10:30
200	9 giugno	Verdi	12:00
112	10 giugno	Rossi	10:30

Tabella 3.3 La relazione di nome **partenze** aggiornata

la quale vìola la dipendenza funzionale **VOLO** → **ORA**. Così, essa andrà "rettificata" in maniera tale che il valore dell'attributo **ORA** per il **VOLO** = 112 sia sempre 9:30, cioè in tutte e tre le ennuple della Tabella 3.2:

112	6 giugno	Rossi	10:30
112	7 giugno	Verdi	10:30

112	10 giugno	Rossi	10:30

Alla fine, si ottiene la relazione riportata in Tabella 3.4

VOLO	DATA	PILOTA	ORA
112	6 giugno	Rossi	9:30
112	7 giugno	Verdi	9:30
200	9 giugno	Verdi	12:00
112	10 giugno	Rossi	9:30

Tabella 3.4 La relazione di nome **partenze** rettificata

Si osservi che la procedura di rettifica deve trovare tutte le ennuple con **VOLO** = 112 ed aggiornarvi il valore dell'attributo **ORA**. Le cose sarebbero invece andate meglio, cioè non ci sarebbe stato bisogno della procedura di rettifica, se al momento della progettazione della base di dati avessimo introdotto anziché la variabile relazionale **partenze** due variabili relazionali, **giornale** e **imbarco**, la prima con schema {**VOLO**, **DATA**, **PILOTA**} e la seconda con schema {**VOLO**, **ORA**}. Allora, lo stato della base di dati prima dell'esecuzione dell'operazione di modifica dell'ora del volo 112 conterrebbe, al posto della Tabella 3.2, le due relazioni riportate nelle Tabelle 3.5 e 3.6:

VOLO	DATA	PILOTA
112	6 giugno	Rossi
112	7 giugno	Verdi
200	9 giugno	Verdi
112	10 giugno	Rossi

Tabella 3.5 La relazione attuale di nome **giornale**

VOLO	ORA
112	10:30
200	12

Tabella 3.6 La relazione attuale di nome **imbarco**

cosicché l'operazione di modifica avrebbe coinvolto solo la variabile relazionale **imbarco**:

cambiare (**imbarco**; **VOLO** = 112, **ORA** = 10:30; **ORA** = 9:30)

ed avrebbe potuto essere eseguita con la procedura standard, cioè semplicemente modificando il valore **ORA** nella prima ennupla della relazione di nome **imbarco**. Il risultato dell'esecuzione sarebbe la relazione riportata nella Tabella 3.7.

VOLO	ORA
112	9:30
200	12

Tabella 3.7 La relazione di nome **imbarco** aggiornata ◊

In generale, dato lo schema R di una variabile relazionale ***r***, una dipendenza funzionale $X \rightarrow Y$ in cui tanto X quanto Y sono sottoinsiemi di R la diremo *anomala* per R se esiste un valore di ***r*** che contenga due o più ennuple $t_1, \ldots, t_n$, $n > 1$, con

$$t_1[X] = \ldots = t_n[X].$$

La diciamo anomala perché, se la relazione di nome ***r*** contenuta nello stato attuale della base di dati contenesse n ennuple, $n > 1$, con identici valori in X e vi si andasse a modificare una qualsiasi di queste n ennuple nei valori degli attributi che appartengono ad Y, allora l'applicazione della procedura standard non sarebbe sufficiente ed il suo risultato andrebbe rettificato nelle restanti n–1 ennuple. Un'analoga anomalia si presenterebbe se la relazione di nome ***r*** contenuta nello stato attuale della base di dati venisse aggiornata aggiungendo un'ennupla t e già esistesse un'altra ennupla t' con $t'[X] = t[X]$. Quanto poi alla cancellazione, la dipendenza funzionale $X \rightarrow Y$ porta ad una nuova anomalia. Supponiamo che la relazione di nome ***r*** contenuta nello stato attuale della base di dati contenga una sola ennupla t con $t[X] = x$ e $t[Y] = y$. Se cancelliamo t, otteniamo sì una relazione che soddisfa la dipendenza funzionale $X \rightarrow Y$, ma noi abbiamo perso l'informazione che il valore y di Y è abbinato al valore x di X.

Esempio 3.3 Consideriamo una variabile relazionale **esame** con schema {**STUDENTE**, **MATERIA**, **DOCENTE**, **VOTO**} e la dipendenza funzionale **MATERIA** → **DOCENTE**. Supponiamo che la relazione di nome **esame** contenuta nello stato attuale della base di dati sia la Tabella 3.8

STUDENTE	MATERIA	DOCENTE	VOTO
Gödel	Assiomatica	Hilbert	30
Gödel	Algebra	Fermat	27
Wiles	Algebra	Fermat	30

Tabella 3.8 La relazione attuale di nome **esame**

e che essa venga aggiornata con l'operazione

cancellare (**esame**; **STUDENTE** = Gödel, **MATERIA** = Assiomatica, **DOCENTE** = Hilbert, **VOTO** = 30)

La nuova relazione di nome **esame** sarà quella riportata in Tabella 3.9

STUDENTE	MATERIA	DOCENTE	VOTO
Gödel	Algebra	Fermat	27
Wiles	Algebra	Fermat	30

Tabella 3.9 La relazione di nome **esame** aggiornata

così che avremo perso l'informazione che Hilbert è il docente della materia Assiomatica. ◊

L'altra faccia delle anomalie è la *ridondanza* dei dati. Consideriamo ancora una dipendenza funzionale $X \rightarrow Y$ anomala per lo schema di una variabile relazionale $\boldsymbol{r}$ e supponiamo che la relazione attuale r di $\boldsymbol{r}$ contiene due o più ennuple $t_1, \ldots, t_n$, $n > 1$, con $t_1[X] = \ldots = t_n[X]$. Allora, per ogni i, $2 \leq i \leq n$, abbiamo che $t_i[Y] = t_1[Y]$. Cioè, tutti i dati $t_2[Y], \ldots, t_n[Y]$ sono *ridondanti* e noi potremmo anche non riportarli (registrarli) perché essi sono univocamente determinati e calcolabili. In altre parole, c'è uno spreco di memoria nella tabella che rappresenta r.

3.2 Modelli semantici

Abbiamo visto che la semantica degli attributi presenti nello schema di una variabile relazionale $\boldsymbol{r}$ impone dei vincoli (nella fattispecie, dipendenze funzionali) che, restringendo il campo dei valori di $\boldsymbol{r}$ accettabili, vanno tenuti presenti quando si eseguono operazioni di aggiornamento sulla relazione attuale di nome $\boldsymbol{r}$ per garantire che la versione aggiornata sia ancora un valore accettabile. in ogni istante la relazione di nome $\boldsymbol{r}$ contenuta nella base di dati sia semanticamente valida. È per questo che, in fase di progettazione della base di dati, per ogni variabile relazionale $\boldsymbol{r}$ vengono specificati non solo il tipo R, d ma anche un insieme F (eventualmente vuoto) di dipendenze funzionali a garanzia dell'integrità semantica dei dati. Ovviamente, ogni dipendenza funzionale in F deve essere *applicabile* ad R nel senso che se F contiene la dipendenza funzionale $X \rightarrow Y$ allora tanto X quanto Y devono essere sottoinsiemi di R. La coppia $M = [R, F]$ definisce il *modello* (*semantico*) della variabile relazionale $\boldsymbol{r}$ e diremo che un valore r di $\boldsymbol{r}$ è *conforme* al modello M se la relazione r soddisfa tutte le dipendenze funzionali presenti in F. Dunque, i valori accettabili di $\boldsymbol{r}$ sono solo le relazioni conformi al modello M. Così, quando la relazione r di nome $\boldsymbol{r}$ contenuta nello stato attuale della base di dati viene aggiornata, dopo l'applicazione della procedura standard di aggiornamento andrà controllata la conformità del nuovo valore r' di $\boldsymbol{r}$ al modello M e, se r' non lo è, allora la relazione r' verrà rettificata. Vediamo come nell'ipotesi che r sia conforme ad M. Ovviamente, se r' è il risultato della cancellazione di un'ennupla da r allora sicuramente t' sarà conforme ad M e non interverrà nessuna rettifica. Se invece r' è il risultato di un'aggiunta o di una modifica di un'ennupla allora, detta t' l'ennupla in $r' - r$, si scandirà la relazione r' e per ogni ennupla t, $t \neq t'$, se esiste una dipendenza funzionale $X \rightarrow Y$ in F tale che $t[X] = t'[X]$ e $t[Y] \neq t'[Y]$, allora si porrà $t[Y] := t'[Y]$.

3.3 Dipendenze funzionali banali

Una dipendenza funzionale $X \rightarrow Y$ è *banale* se, per ogni sovrainsieme R di X Y, la dipendenza $X \rightarrow Y$ è soddisfatta da ogni relazione con schema R. Daremo fra un momento una caratterizzazione delle dipendenze funzionali banali. Sia $R = \{A_1, \ldots, A_n\}$ e sia $Z = \{A_1, \ldots, A_m\}$, $m < n$, un sottoinsieme proprio di R. Un *prototipo* di

relazione con schema R e *radice* Z è una qualsiasi relazione di tipo R, d formata da due sole ennuple t_1 e t_2 le quali appartengono a $dom(R)$ e sono tali che $t_2[Z] = t_1[Z]$ e $t_2(A) \neq t_1(A)$ per ogni attributo A appartenente ad $R–Z$.

A_1	…	A_m	A_{m+1}	…	A_n
a_1	…	a_m	a_{m+1}	…	a_n
a_1	…	a_m	b_{m+1}	…	b_n

È immediato verificare che tale relazione vìola una dipendenza funzionale $X \rightarrow Y$ applicabile ad R se e solo se $X \quad Z$ e $Y–Z \neq \emptyset$.

***Teorema* 3.1** Una dipendenza funzionale $X \rightarrow Y$ è banale se e solo se Y è un sottoinsieme di X.

Dimostrazione. (*se*) Sia r un'arbitraria relazione con schema R, dove R è un sovrainsieme di $X \ Y$. Se r è una relazione vuota, allora $X \rightarrow Y$ è banalmente soddisfatta da r. Altrimenti, siccome $t[Y] = (t[X])[Y]$ per ogni ennupla t in r, prese arbitrariamente due ennuple t_1 e t_2 in r tali che $t_1[X] = t_2[X]$, abbiamo $t_1[Y] = (t_1[X])[Y] = (t_2[X])[Y] = t_2[Y]$, di qui la tesi.

(*solo se*) Sia $X \rightarrow Y$ una dipendenza funzionale banale. Supponiamo per assurdo che Y contenga un attributo che non appartiene ad X. Allora, arbitrariamente scelto un sovrainsieme R di $X \ Y$, ogni prototipo di relazione con schema R e radice X vìola la dipendenza funzionale $X \rightarrow Y$ (contraddizione). □

Mostriamo ora che il numero delle dipendenze funzionali banali applicabili ad uno schema R cresce in maniera esponenziale rispetto al numero di attributi contenuti in R. Qui e nel seguito, indicheremo spesso gli insiemi X ed Y che compaiono in una dipendenza funzionale $f = X \rightarrow Y$, rispettivamente con $det(f)$ e $dip(f)$. Dunque, per il Teorema 3.1, sono banali tutte e solo le dipendenze funzionali f con $\emptyset \neq dip(f)$ $det(f)$. Così, se $|R| = n$; allora fissato un sottoinsieme non vuoto X di R con k attributi, $1 \leq k \leq n$, il numero di dipendenze funzionali banali f con $det(f) = X$ è pari a $2^k–1$, e lo stesso vale per qualsiasi altro sottoinsieme di R formato da k attributi. Ora, siccome i sottoinsiemi di R di cardinalità k sono in numero pari a $\binom{n}{k}$, abbiamo che il numero di dipendenze funzionali banali f con $|det(f)| = k$ è uguale a

$$\binom{n}{k}(2^k - 1)$$

In definitiva, le dipendenze funzionali banali applicabili ad R sono in numero pari a

$$\sum_{k=1}^{n} \binom{n}{k}(2^k - 1) = \sum_{k=0}^{n} \binom{n}{k}(2^k - 1) = \sum_{k=0}^{n} \binom{n}{k} 2^k - \sum_{k=0}^{n} \binom{n}{k} = 3^n - 2^n;$$

ciononostante, esse costituiscono solo una frazione asintoticamente infinitesima dal numero totale — uguale a $(2^n-1)^2$ — delle dipendenze funzionali applicabili ad *R*.

3.4 Modelli normali

Dato un modello $M = [R, F]$, una dipendenza funzionale f applicabile ad R si dice *valida* in M se, arbitrariamente scelto il tipo di relazione $\langle R, d \rangle$, ogni relazione di tipo $\langle R, d \rangle$ che sia conforme ad M soddisfa anche f. Va da sé che ogni dipendenza funzionale contenuta in F è valida in M così come è valida in M ogni dipendenza funzionale banale applicabile ad *R*.

Diremo che un modello $M = [R, F]$ è *normale* se

(*i*) non contiene attributi composti sovradimensionati, e

(*ii*) nessuna dipendenza funzionale valida in M è anomala.

Qui, la condizione (*ii*) assicura che in ogni relazione conforme ad M non avremo mai ridondanza né incontreremo mai anomalie in fase di aggiornamento. Restano però aperti il problema di sapere quali sono le dipendenze funzionali valide in M ed il problema di saper riconoscere tra queste quelle anomale. Per risolvere il primo problema, dobbiamo analizzare più a fondo il concetto di validità. A tale scopo, ne stabiliamo alcune proprietà.

***Teorema* 3.2** (*Proprietà Additiva*). Dato un modello $M = [R, F]$, se le dipendenze funzionali $X \rightarrow Y$ ed $X \rightarrow Z$ sono valide in M, allora lo è anche la dipendenza funzionale $X \rightarrow YZ$.

Dimostrazione. Sia r una arbitraria relazione conforme al modello M. Per ipotesi, per ogni coppia t_1 e t_2 di ennuple presenti in r, si ha che se $t_1[X] = t_2[X]$ allora $t_1[Y] = t_2[Y]$, e $t_1[Z] = t_2[Z]$. Ne viene che non possono esistere due ennuple t_1 e t_2 in r con $t_1[X] = t_2[X]$ e $t_1[YZ] \neq t_2[YZ]$ perché, altrimenti, siccome per ogni ennupla t si ha che $t[Y] = (t[YZ])[Y]$ e $t[Z] = (t[YZ])[Z]$, si dovrebbe avere che $t_1[Y] \neq t_2[Y]$ o $t_1[Z] \neq t_2[Z]$. □

***Teorema* 3.3** (*Proprietà Proiettiva*). Dato un modello $M = [R, F]$, se la dipendenza funzionale $X \rightarrow Z$ è valida in M ed Y è un sottoinsieme nonvuoto di Z, allora anche la dipendenza funzionale $X \rightarrow Y$ è valida in M.

Dimostrazione. Sia r una arbitraria relazione conforme al modello M. Si osservi che, se Y è un sottoinsieme di Z, allora per ogni ennupla con schema R si ha $t[Y] = (t[Z])[Y]$. Ora, per ipotesi, per ogni coppia di ennuple t_1 e t_2 in r, se $t_1[X] = t_2[X]$ allora $t_1[Z] =$

$t_2[Z]$. Ma se $t_1[Z] = t_2[Z]$ allora si ha pure che $t_1[Y] = (t_1[Z])[Y] = (t_2[Z])[Y] = t_2[Y]$, di qui la tesi. □

***Teorema* 3.4** (*Proprietà Transitiva*). Se le dipendenze funzionali $X \rightarrow Y$ e $Y \rightarrow Z$ sono valide in M, allora lo è anche la dipendenza funzionale $X \rightarrow Z$.

Dimostrazione. Sia r una arbitraria relazione conforme al modello M. Assumiamo che le dipendenze funzionali $X \rightarrow Y$ ed $Y \rightarrow Z$, siano entrambe valide in M cosicché per ogni coppia di ennuple t_1 e t_2 in r, si ha

(1) se $t_1[X] = t_2[X]$ allora $t_1[Y] = t_2[Y]$ e
(2) se $t_1[Y] = t_2[Y]$ allora $t_1[Z] = t_2[Z]$.

Allora, per ogni coppia di ennuple t_1 e t_2 in r con $t_1[X] = t_2[X]$ si ha per la condizione (1) che $t_1[Y] = t_2[Y]$, la qual cosa per la condizione (2) implica che $t_1[Z] = t_2[Z]$ e, dunque, $t_1[Y\ Z] = t_2[Y\ Z]$, di qui la tesi. □

3.5 Chiavi e sovrachiavi

In questo paragrafo introdurremo la nozione di "chiave" di un modello, nozione basilare non solo nella progettazione concettuale di una base di dati ma anche nell'organizzazione fisica dei dati.

Sia $M = [R, F]$ un modello. Una *chiave* di M è un sottoinsieme minimale X di R tale che la dipendenza funzionale $X \rightarrow R$ sia valida in M; in altre parole, una chiave di M è un sottoinsieme minimale di R il cui valore identifica ciascuna ennupla in ogni relazione conforme ad M. Per la proprietà proiettiva, abbiamo che X è una chiave di M se e solo se X è un sottoinsieme minimale X di R tale che, per ogni attributo A in R, la dipendenza funzionale $X \rightarrow A$ è valida in M. Se soprassediamo al requisito di minimalità, otteniamo la nozione di "sovrachiave" (*superkey*, in inglese): un sottoinsieme X di R è una *sovrachiave* di M se la dipendenza funzionale $X \rightarrow R$ è valida in M. Ovviamente, ogni chiave è anche una sovrachiave. Si osservi che se X è una sovrachiave (in particolare, una chiave) di M ed $Y \rightarrow X$ è una qualsiasi dipendenza valida in M, allora per la proprietà transitiva anche Y è una sovrachiave di M.

La nozione di sovrachiave consente di riconoscere tra le dipendenze funzionali valide in un modello quelle anomale.

***Teorema* 3.5** Sia $M = [R, F]$ un modello. Una dipendenza funzionale f valida in M non è anomala se e solo se *det*(f) è una sovrachiave di M.

Dimostrazione. (*se*) Se *det*(f) è una sovrachiave di M allora *det*(f) determina R e, pertanto, non possono esistere relazioni conformi ad M con due distinte ennuple che concordino su *det*(f).

(*solo se*) Sia $det(f) = X$ e sia Z l'unione degli insiemi Y tali che $X \rightarrow Y$ sia una dipendenza funzionale valida in M. Per la proprietà additiva, la dipendenza funzionale $X \rightarrow Z$ è valida in M. Supponiamo per assurdo che X non sia una sovrachiave, cioè $Z \neq R$. Sia r un prototipo di relazione con schema R e radice Z. Allora r soddisfa la dipendenza funzionale $X \rightarrow Y$. D'altra parte, non può esistere una dipendenza funzionale $X' \rightarrow Y'$ in F che sia violata da r perché, allora, si avrebbe $X' \quad Z$ e $Y' - Z \neq \emptyset$. Ma allora la dipendenza funzionale $Z \rightarrow X'$ sarebbe valida in M e, per la proprietà transitiva, anche le dipendenze funzionali $Z \rightarrow Y'$ e $X \rightarrow Y'$ sarebbero valide in M il che $Y' \quad Z$ (contraddizione). Dunque r è conforme ad M e, siccome X è un sottoinsieme di Z, conterrebbe due ennuple che concordano su X, cosa che contraddice l'ipotesi che f non è anomala. □

Le sovrachiavi e, in particolare, le chiavi consentono di semplificare la sintassi delle operazioni di aggiornamento.
Sia $M = [R, F]$ il modello di una variabile relazionale ***r***, dove $R = \{A_1, \ldots, A_n\}$. Supponiamo che $X = \{A_1, \ldots, A_k\}$, $k < n$, sia una sovrachiave di M; allora possiamo utilizzare X come *chiave d'accesso* alla relazione di nome ***r*** ed esprimere le operazioni di aggiornamento su ***r*** più semplicemente nel modo seguente.

AGGIUNTA. L'aggiunta dell'ennupla $t = \{t(A_1) = a_1, \ldots, t(A_k) = a_k, t(A_{k+1}) = a_{k+1}, \ldots, t(A_n) = a_n\}$ si esprime come

aggiungere (***r***; $A_1 = a_1, \ldots, A_k = a_k$; $A_{k+1} = a_{k+1}, \ldots, A_n = a_n$).

CANCELLAZIONE. La cancellazione dell'ennupla t con $t(A_1) = a_1, \ldots, t(A_k) = a_k$ si esprime come

cancellare (***r***; $A_1 = a_1, \ldots, A_k = a_k$).

MODIFICA. La modifica nell'ennupla t con $t(A_1) = a_1, \ldots, t(A_k) = a_k$ dei valori degli attributi $B_1, \ldots, B_m$, si esprime come

cambiare (***r***; $A_1 = a_1, \ldots, A_k = a_k$; $B_1 = b_1, \ldots, B_m = b_m$).

Esempio 3.2 (seguito). Qui abbiamo due chiavi {**VOLO**, **DATA**} e {**PILOTA**, **DATA**, **ORA**} del modello. Supponiamo di scegliere la coppia di attributi {**VOLO**, **DATA**} come chiave di accesso a **partenze**. Allora, l'operazione di modifica:

cambiare (**partenze**; **VOLO** = 112, **DATA** = 6 giugno, **PILOTA** = Rossi, **ORA** = 10:30; **ORA** = 9:30).

si può formulare più semplicemente come

cambiare (**partenze**; **VOLO** = 112, **DATA** = 6 giugno; **ORA** = 9:30). ◊

CAPITOLO 4

TEORIA DELLE DIPENDENZE FUNZIONALI

In questo capitolo forniremo un sistema formale per il trattamento delle dipendenze funzionali, il quale ci permetterà di elaborare un semplice algoritmo per il riconoscimento delle dipendenze funzionali valide (*test di validità*) in un assegnato modello ed un algoritmo per decidere se un dato modello è o meno normale (*test di normalità*).

4.1 Sistemi formali

Una *forma funzionale* è una coppia ordinata X, Y di variabili di tipo insiemistico. Una *regola di derivazione* per forme funzionali è un'espressione del tipo

Dato Φ, se p allora φ

dove Φ è una lista (eventualmente vuota) di forme funzionali, p è una condizione (che potrebbe essere assente) sulle variabili insiemistiche che compaiono in Φ, e φ è una forma funzionale. Le forme funzionali presenti in Φ prendono il nome di *antecedenti* della regola di derivazione, e φ è chiamato il suo *conseguente*.

Esempi di regole di derivazione per forme funzionali sono

A1. Se X è un insieme non vuoto allora X, X .

A2. Dato X, Z , se $Y \quad Z$ allora X, Y .

A3. Dati X, V ed Y, Z , se $Y \quad V$ allora $X, V \quad Z$.

A4. Dato Y, Z , se $Y \quad X$ allora X, Z .

A5. Dati X, Y e V, Z , se $Y \quad V$ allora $X \quad V, Z$.

A6. Dati X, Y ed X, Z , allora $X, Y \quad Z$.

Un insieme α di regole di derivazione per forme funzionali prende il nome di *sistema formale* per forme funzionali.

Dato un insieme R di attributi, un'*applicazione* di una regola di derivazione consiste nell'assegnare ad ogni variabile insiemistica presente nella regola di derivazione un sottoinsieme di R in maniera tale che la condizione p sia soddisfatta; allora, il *risultato* dell'applicazione è la dipendenza funzionale f applicabile ad R con $det(f)$ uguale al

valore della prima componente di φ e con *dip*(*f*) uguale al valore della seconda componente di φ. Consideriamo poi un insieme arbitrario F $\{f\}$ di dipendenze funzionali applicabili ad R. Una α-*derivazione* di f da F è una sequenza finita (f_1, …, f_n), $n \geq 1$, di dipendenze funzionali applicabili ad R e tali che $f = f_n$ e, per ogni i, $1 \leq i \leq n$,

1. f_i appartiene ad F oppure

2. f_i è il risultato dell'applicazione di una regola di derivazione in α in cui ogni antecedente è sostituito da una dipendenza funzionale che precede f_i nella sequenza.

Una dipendenza funzionale applicabile ad R si dice α-*derivabile* da F se ne esiste una α-derivazione da F.

Esempio 4.1 Siano $R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{G}, \mathbf{H}\}$ ed $F = \{\mathbf{AB} \rightarrow \mathbf{E}, \{\mathbf{AG} \rightarrow \mathbf{D}, \mathbf{BE} \rightarrow \mathbf{C}, \mathbf{E} \rightarrow \mathbf{G}, \mathbf{CG} \rightarrow \mathbf{H}\}$. Consideriamo il sistema formale $\alpha = \{A1, A4\}$. Mostriamo che la dipendenza funzionale $\mathbf{ABC} \rightarrow \mathbf{E}$ è α-derivabile da F.

$f_1 = \mathbf{AB} \rightarrow \mathbf{E}$		in F	
$f_2 = \mathbf{AB} \rightarrow \{\mathbf{A}, \mathbf{B}\}$		A1	$\{\mathbf{A}, \mathbf{B}\}/X$
$f_3 = \mathbf{ABC} \rightarrow \mathbf{E}$	A4(f_1)		$\mathbf{AB}/X, \mathbf{E}/Y, \mathbf{ABC}/Z$

◊

Esempio 4.2 Siano $R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{G}, \mathbf{H}\}$ ed $F = \{\mathbf{AB} \rightarrow \mathbf{E}, \mathbf{AG} \rightarrow \mathbf{D}, \mathbf{BE} \rightarrow \mathbf{C}, \mathbf{E} \rightarrow \mathbf{G}, \mathbf{CG} \rightarrow \mathbf{H}\}$. Consideriamo il sistema formale $\alpha = \{A1, A2, A3\}$. Mostriamo che la dipendenza funzionale $\mathbf{AB} \rightarrow \mathbf{GH}$ è α-derivabile da F.

$f_1 = \mathbf{CE} \rightarrow \mathbf{CE}$	A1	
$f_2 = \mathbf{E} \rightarrow \mathbf{G}$		in F
$f_3 = \mathbf{CE} \rightarrow \mathbf{CEG}$		A3(f_1, f_2)
$f_4 = \mathbf{CE} \rightarrow \mathbf{CG}$	A2(f_3)	
$f_5 = \mathbf{CG} \rightarrow \mathbf{H}$		in F
$f_6 = \mathbf{CE} \rightarrow \mathbf{CGH}$		A3(f_4, f_5)
$f_7 = \mathbf{CE} \rightarrow \mathbf{GH}$	A2(f_6)	
$f_8 = \mathbf{AB} \rightarrow \mathbf{AB}$	A1	
$f_9 = \mathbf{AB} \rightarrow \mathbf{E}$		in F
$f_{10} = \mathbf{AB} \rightarrow \mathbf{ABE}$		A3(f_8, f_9)
$f_{11} = \mathbf{BE} \rightarrow \mathbf{C}$	in F	
$f_{12} = \mathbf{AB} \rightarrow \mathbf{ABCE}$		A3(f_{10}, f_{11})
$f_{13} = \mathbf{AB} \rightarrow \mathbf{ABCEG}$		A3(f_{12}, f_4)
$f_{14} = \mathbf{AB} \rightarrow \mathbf{ABCEGH}$		A3(f_{13}, f_7)
$f_{15} = \mathbf{AB} \rightarrow \mathbf{GH}$		A2(f_{14})

◊

Sia α un sistema formale (eventualmente vuoto). Ci chiediamo qual è il potere deduttivo di α. Si osservi che se α è vuoto (cioè non contiene regole di derivazione), allora per ogni insieme *R* di attributi e per ogni insieme *F* di dipendenze funzionali applicabili ad *R*, l'insieme delle dipendenze funzionali che sono α-derivabili da *F* coincide con *F* e, quindi, il suo potere deduttivo è praticamente nullo. Almeno ci aspettiamo che α sappia derivare (oltre alle dipendenze funzionali in *F*) tutte le dipendenze funzionali banali applicabili ad *R*, e questo è il caso ad esempio del sistema formale {A1, A2}. Nel caso migliore, α dovrà essere tale che:

(I) per ogni modello $M = [R, F]$, ogni dipendenza funzionale α-derivabile da *F* sia valida in *M*,

(II) per ogni modello $M = [R, F]$, ogni dipendenza funzionale valida in *M* sia α-derivabile da *F*.

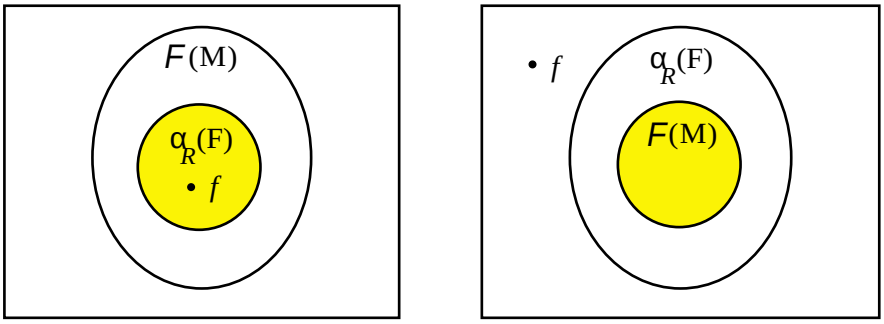
Diremo che un sistema formale α è *corretto* se soddisfa la proprietà I, e lo diremo *completo* se soddisfa la proprietà II. Così, dato un modello $M = [R, F]$, se indichiamo con $\alpha_R(F)$ ed $\mathcal{F}(M)$ rispettivamente l'insieme delle dipendenze funzionali α-derivabili da *F* e l'insieme delle dipendenze funzionali valide in *M*, abbiamo che α è

— corretto se $\alpha_R(F) \quad \mathcal{F}(M)$ per ogni modello $M = [R, F]$,

— completo se $\mathcal{F}(M) \quad \alpha_R(F)$ per ogni modello $M = [R, F]$,

— corretto e completo se $\alpha_R(F) = \mathcal{F}(M)$ per ogni modello $M = [R, F]$.

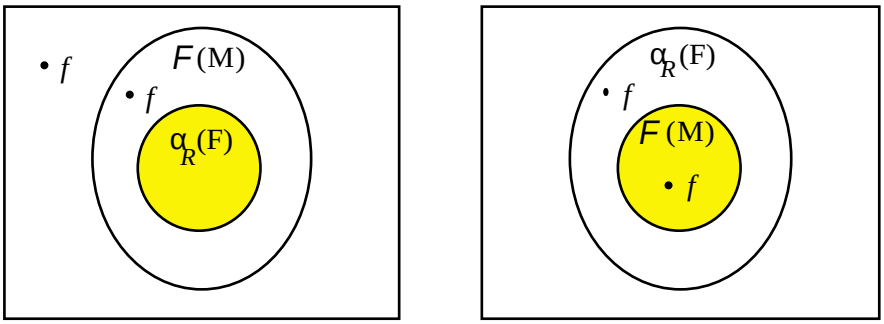
Si osservi che, dato un sistema formale corretto (rispettivamente, completo) α ed un modello $M = [R, F]$, se riusciamo a provare che una dipendenza funzionale *f* applicabile ad *R* è (rispettivamente, non è) α-derivabile da *F*, allora possiamo senz'altro concludere che *f* è (rispettivamente, non è) valida in *M* — vedi Figura 4.1(*a*) (rispettivamente, Figura 4.1(*b*)) —;



(a) (b)

Figura 4.1

mentre, se pure riusciamo a provare che f non è (rispettivamente, è) α-derivabile da F, in mancanza della proprietà di completezza (rispettivamente, di correttezza), dovremo astenerci dal prendere una decisione in merito alla validità di f in M — vedi Figura 4.2(a) (rispettivamente, Figura 4.2(b)) —.



(a) (b)

Figura 4.2

Infine vogliamo segnalare che l'operatore α_R gode delle proprietà caratteristiche degli operatori di chiusura; cioè che per ogni coppia di insiemi di dipendenze funzionali F e G si ha:

— $F \quad \alpha_R(F)$,
— se $F \quad G$ allora $\alpha_R(F) \quad \alpha_R(G)$,
— $\alpha_R(\alpha_R(F)) = \alpha_R(F)$.

4.2 Criterio di completezza

Daremo in questo paragrafo un criterio perché un sistema formale corretto sia anche completo. Dato un sistema formale α, sia F un qualsiasi insieme di dipendenze funzionali. Per ogni insieme nonvuoto X di attributi, consideriamo la famiglia $D_\alpha(X)$ di tutti gli insiemi Y tali che la dipendenza funzionale $X \rightarrow Y$ sia α-derivabile da F.
Diremo che un sistema formale corretto α è un'*assiomatizzazione* delle dipendenze funzionali se, per ogni R ed F, si ha che

(*i*) ogni dipendenza funzionale banale applicabile ad R è α-derivabile da F, e
(*ii*) per ogni sottoinsieme non vuoto X di R, la famiglia $D_\alpha(X)$ è chiusa rispetto all'inclusione e all'unione.

Dunque, se α è un'assiomatizzazione allora, per ogni insieme nonvuoto X, la famiglia $D_\alpha(X)$ conterrà l'insieme che risulta dall'unione di tutti i suoi membri, insieme che indicheremo con $\alpha_F(X)$, cioè

$$\alpha_F(X) = \bigcup_{Y \in D_\alpha(X)} Y,$$

così come conterrà tutti i sottoinsiemi nonvuoti di $\alpha_F(X)$. Il simbolo α_F può essere visto come un operatore che, dato F, associa ad un insieme di attributi un altro insieme (non necessariamente distinto) di attributi; cioè, α_F: $(R) \mapsto$ (R). Dunque, se α è un'assiomatizzazione, allora α gode della seguente proprietà che vale per ogni insieme F di dipendenze funzionali e per ogni coppia X ed Y di insiemi (nonvuoti) di attributi:

B. Una dipendenza funzionale $X \rightarrow Y$ è α-derivabile da F se e solo se Y è un sottoinsieme di $\alpha_F(X)$.

Esempio 4.3 Consideriamo il sistema formale {A1, A2, A6}. Siano $R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$ ed $F = \{\mathbf{A} \rightarrow \mathbf{B}, \mathbf{B} \rightarrow \mathbf{C}\}$. Per ogni sottoinsieme non vuoto X di R, riportiamo qui di seguito i valori di $D_\alpha(X)$ e di $\alpha_F(X)$.

X	$D_\alpha(X)$	$\alpha_F(X)$
{**A**}	{**A**}, {**B**}, {**A**, **B**}	{**A**, **B**}
{**B**}	{**B**}, {**C**}, {**B**, **C**}	{**B**, **C**}
{**C**}	{**C**}	{**C**}
{**A**, **B**}	{**A**}, {**B**}, {**A**, **B**}	{**A**, **B**}
{**A**, **C**}	{**A**}, {**C**}, {**A**, **C**}	{**A**, **C**}
{**B**, **C**}	{**B**}, {**C**}, {**B**, **C**}	{**B**, **C**}
{**A**, **B**, **C**}	{**A**}, {**B**}, {**C**}, {**A**, **B**}, {**A**, **C**}, {**B**, **C**}, {**A**, **B**, **C**}	{**A**, **B**, **C**}

◊

Data un'assiomatizzazione α ed un insieme F di dipendenze funzionali, diremo che α è *dotata di operatore di chiusura* se α_F gode delle seguenti tre proprietà per ogni coppia di insiemi di attributi X e Y:

C1. $X \quad \alpha_F(X)$

C2. se $X \quad Y$ allora $\alpha_F(X) \quad \alpha_F(Y)$

C3. $\alpha_F(\alpha_F(X)) = \alpha_F(X)$.

Se α è dotata di operatore di chiusura, chiameremo l'insieme $\alpha_F(X)$ la α-*chiusura* di X *rispetto* ad F. Si osservi che, siccome α è un'assiomatizzazione, per la condizione (*i*) ogni dipendenza funzionale banale (ed in particolare $X \rightarrow X$) è α-derivabile da F; pertanto, la proprietà C1 è sempre soddisfatta perché segue dalla proprietà B.

Esempio 4.3 (seguito). L'assiomatizzazione {A1, A2, A6} non è dotata di operatore di chiusura, perché la condizione C2 non vale ad esempio per $X = \{\mathbf{B}\}$ ed $Y = \{\mathbf{A}, \mathbf{B}\}$.

◊

***Lemma* 4.1**. Sia α un'assiomatizzazione dotata di operatore di chiusura. Se $F \quad \{f\}$ è un insieme di dipendenze funzionali tale che f non sia α-derivabile da F allora, per ogni modello $M = [R, F]$, f non è valida in M.

Dimostrazione. Per dimostrare la tesi è sufficiente provare che, per ogni modello $M = [R, F]$, esiste una relazione conforme ad M la quale vìola f. Sia Z la α-chiusura di $det(f)$ rispetto ad F, cioè $Z = \alpha_F(det(f))$. Prendiamo un qualsiasi modello $M = [R, F]$ e sia r un prototipo del tipo di relazione R, d con radice Z. Dimostreremo che

(*a*) r è conforme al modello M, e

(*b*) r vìola f.

(*a*) Per provare la conformità di r ad M occorre verificare che r soddisfa tutte le dipendenze funzionali in F. Se $F = \emptyset$, allora la cosa è banalmente vera. Altrimenti, sia $X \rightarrow Y$ una qualsiasi dipendenza funzionale in F. Distinguiamo due casi a seconda che X sia o meno un sottoinsieme di Z.

> (Caso 1) Se $X \quad Z$, allora $t_1[X] = t_2[X]$. Ora, siccome ogni dipendenza funzionale in F è α-derivabile da F, per la proprietà B delle assiomatizzazioni, si ha $Y \quad \alpha_F(X)$. D'altra parte, per le proprietà C2 e C3 dell'operatore α_F, abbiamo $\alpha_F(X) \quad \alpha_F(Z) = Z$. Pertanto, Y è un sottoinsieme di Z e, quindi, $t_1[Y] = t_2[Y]$. Dunque, r soddisfa la dipendenza funzionale $X \rightarrow Y$.
>
> (Caso 2) Se $X - Z \neq \emptyset$, allora $t_1[X] \neq t_2[X]$ e, quindi, banalmente r soddisfa la dipendenza funzionale $X \rightarrow Y$.

(*b*) Siccome $det(f) \quad Z$, abbiamo che $t_1[det(f)] = t_2[det(f)]$. D'altra parte, siccome la dipendenza funzionale f non è α-derivabile da F, per la proprietà B delle

assiomatizzazioni, l'insieme $dip(f)$ non è un sottoinsieme di Z e, pertanto, $t_1[dip(f)] \neq t_2[dip(f)]$ cosicché r vìola f. □

Siamo ora in grado di stabilire il seguente risultato fondamentale della teoria delle dipendenze funzionali.

***Teorema* 4.1** (*Criterio di Completezza*). Un sistema formale per dipendenze funzionali è corretto e completo se e solo se è un'assiomatizzazione dotata di operatore di chiusura.

Dimostrazione. (*se*) Siccome α è un sistema formale corretto, ogni dipendenza funzionale che sia α-derivabile da F è valida in ogni modello $M = [R, F]$. Così, per dimostrare la tesi basta dimostrare che, per ogni modello $M = [R, F]$, ogni dipendenza valida in M è anche α-derivabile da F. Supponiamo per assurdo che esistano un modello $M = [R, F]$ ed una dipendenza funzionale f valida in M che non sia α-derivabile da F. Allora, per il Lemma 4.1, esisterebbe una relazione conforme ad M la quale vìola f, cosa che è in contrasto con il fatto che f è valida in M.

(*solo se*) Siccome α è un sistema formale corretto e completo, ogni dipendenza funzionale è α-derivabile da F se e solo se è valida in ogni modello $M = [R, F]$. Allora, per le proprietà additiva e proiettiva delle dipendenze valide, α è un'assiomatizzazione. Basta allora dimostrare che, per ogni insieme F di dipendenze funzionali, l'operatore α_F è un operatore di chiusura. Già sappiamo che la proprietà C1 è sempre soddisfatta. Quanto alla proprietà C2, se $X \quad Y$, allora la dipendenza $Y \rightarrow X$ è banalmente valida e la dipendenza funzionale $X \rightarrow \alpha_F(X)$ è anche valida perché α-derivabile da F. Allora, per la proprietà transitiva, anche la dipendenza funzionale $Y \rightarrow \alpha_F(X)$ è valida e, quindi, α-derivabile da F cosicché, per la proprietà B delle assiomatizzazioni, si ha $\alpha_F(X) \quad \alpha_F(Y)$. Infine, per dimostrare la proprietà C3, osserviamo innanzitutto che, in virtù delle proprietà C1 e C2, abbiamo che $\alpha_F(X) \quad \alpha_F(\alpha_F(X))$. Dunque, basta dimostrare che $\alpha_F(\alpha_F(X)) \quad \alpha_F(X)$. Ora, si osservi che le due dipendenze funzionali $X \rightarrow \alpha_F(X)$ e $\alpha_F(X) \rightarrow \alpha_F(\alpha_F(X))$ sono entrambe α-derivabili da F e, quindi, valide. Per la proprietà transitiva, anche la dipendenza funzionale $X \rightarrow \alpha_F(\alpha_F(X))$ è valida e, quindi, α-derivabile da F. Ma allora per la proprietà B delle assiomatizzazioni abbiamo $\alpha_F(\alpha_F(X)) \quad \alpha_F(X)$. □

Dal Teorema 4.1 e dalla proprietà II delle assiomatizzazioni ricaviamo il seguente risultato.

***Teorema* 4.2** Sia α un'assiomatizzazione delle dipendenze funzionali dotata di operatore di chiusura. Per ogni modello $M = [R, F]$, una dipendenza funzionale f è valida in M se e solo se $dip(f)$ è un sottoinsieme della α-chiusura di $det(f)$ rispetto ad F.

Si osservi che il Teorema 4.1 non ci dice nulla sull'esistenza di un'assiomatizzazione dotata di operatore di chiusura, ma implica che, se ne sapessimo costruire una avremmo così trovato un sistema formale corretto e completo. A quel punto, se anche sapessimo calcolare la chiusura di un qualsiasi insieme di attributi rispetto ad un qualsiasi insieme di dipendenze funzionali, per il Teorema 4.2 avremmo anche un test di validità.

4.3 Il sistema formale {A1, A2, A3}

Mostreremo ora che il sistema formale {A1, A2, A3} è corretto e completo. Cominciamo con il dimostare che è un'assiomatizzazione.

***Lemma* 4.2**. Il sistema formale {A1, A2, A3} è corretto.

Dimostrazione. (A1) Per il Lemma 1.4, $X \rightarrow X$ è una dipendenza funzionale banale ed è quindi valida in ogni modello.

(A2) Per il Lemma 1.2.

(A3) Siano $M = [R, F]$ un qualsiasi modello ed r una arbitraria relazione conforme al modello M. Assumiamo che le dipendenze funzionali $X \rightarrow V$ ed $Y \rightarrow Z$, $Y \quad V$, siano entrambe valide in M cosicché per ogni coppia di ennuple t_1 e t_2 in r, si ha

(1) se $t_1[X] = t_2[X]$ allora $t_1[V] = t_2[V]$ e

(2) se $t_1[Y] = t_2[Y]$ allora $t_1[Z] = t_2[Z]$.

Allora, per ogni coppia di ennuple t_1 e t_2 in r con $t_1[X] = t_2[X]$ si ha per la condizione (1) che $t_1[Y] = (t_1[V])[Y] = (t_2[V])[Y] = t_2[Y]$, la qual cosa per la condizione (2) implica che $t_1[Z] = t_2[Z]$ e, dunque, $t_1[V \ Z] = t_2[V \ Z]$. Pertanto, anche la dipendenza funzionale $X \rightarrow V \ Z$ è soddisfatta da r e, quindi è valida in M data l'arbitrarietà con cui è stata scelta la relazione r. □

***Lemma* 4.3**. Il sistema formale {A1, A2, A3} è un'assiomatizzazione.

Dimostrazione. Per il Lemma 4.2, è sufficiente provare le proprietà (*i*) e (*ii*) delle assiomatizzazioni. Quanto alla proprietà (*i*), si osservi che, se $Y \ X$, allora la dipendenza funzionale $X \rightarrow Y$ è derivabile da F con le regole A1 ed A2. Proviamo ora la proprietà (*ii*), cioè che, per ogni insieme F di dipendenze funzionali e per ogni insieme nonvuoto X di attributi, la famiglia $D_\alpha(X)$ è chiusa rispetto all'inclusione ed all'unione. La sua chiusura rispetto all'inclusione è garantita dalla regola A2. Dimostriamo ora la chiusura di $D_\alpha(X)$ rispetto all'unione, cioè che, se Y e Z sono due insiemi tali che le dipendenze funzionali $X \rightarrow Y$ e $X \rightarrow Z$ sono entrambe derivabili da F, allora lo è anche la dipendenza funzionale $X \rightarrow Y \ Z$. Supponiamo che $(g_1, \ldots, g_m)$ sia una derivazione da F della dipendenza funzionale $X \rightarrow Y$ e che $(h_1, \ldots, h_n)$ sia una

derivazione da F della dipendenza funzionale $X \to Z$. Allora, possiamo costruire la seguente derivazione ($f_1, \ldots, f_{m+n+4}$) da F della dipendenza funzionale $X \to Y\ Z$.

$f_1 = g_1$
…
$f_m = g_m$ (= $X \to Y$)
$f_{m+1} = h_1$
…
$f_{m+n} = h_n$ (= $X \to Z$)
$f_{m+n+1} = X \to X$ A1
$f_{m+n+2} = X \to X\ Y$ A3(f_{m+n+1}, f_m)
$f_{m+n+3} = X \to X\ Y\ Z$ A3(f_{m+n+2}, f_{m+n})
$f_{m+n+4} = X \to Y\ Z$ A2(f_{m+n+3})

Dunque, la famiglia $D_\alpha(X)$ è chiusa anche rispetto all'unione, di qui la tesi. □

***Teorema* 4.3**. Il sistema formale {A1, A2, A3} è corretto e completo.

Dimostrazione. Indichiamo con α il sistema formale {A1, A2, A3}. Per il Teorema 4.1, basta provare che α è un'assiomatizzazione delle dipendenze funzionali dotata di operatore di chiusura. Per il Lemma 4.3, α è un'assiomatizzazione. Resta da dimostrare che α_F è un operatore di chiusura per ogni insieme F di dipendenze funzionali, cioè gode delle proprietà C1, C2 e C3. Come si è già osservato, la proprietà C1 vale per ogni assiomatizzazione. Quanto alla proprietà C2, si osservi che, se $X\ Y$, allora la dipendenza funzionale $Y \to X$ è derivabile da F con le regole A1 ed A2. D'altra parte, per la proprietà B delle assiomatizzazioni, anche la dipendenza funzionale $X \to \alpha_F(X)$ è derivabile da F. A questo punto, applicando la regola A3 alle dipendenze funzionali $Y \to X$ e $X \to \alpha_F(X)$, deriviamo la dipendenza funzionale $Y \to X\ \alpha_F(X)$; ma, per la proprietà C1, $X\ \alpha_F(X) = \alpha_F(X)$. Finalmente, la proprietà B ci porta a concludere che $\alpha_F(X)$ è un sottoinsieme di $\alpha_F(Y)$. Infine, per dimostrare che vale la proprietà C3, cioè che $\alpha_F(\alpha_F(X)) = \alpha_F(X)$, osserviamo che, per le proprietà C1 e C2 appena dimostrate, si ha senz'altro che $\alpha_F(X)\ \alpha_F(\alpha_F(X))$. Basterà allora provare che $\alpha_F(X)\ \alpha_F(\alpha_F(X))$. Ora per la proprietà B le dipendenze funzionali $X \to \alpha_F(X)$ ed $\alpha_F(X) \to \alpha_F(\alpha_F(X))$ sono derivabili. Allora, applicando la regola A3 alle dipendenze funzionali

$X \to \alpha_F(X)$ $\alpha_F(X) \to \alpha_F(\alpha_F(X))$,

deriviamo la dipendenza funzionale

$X \to \alpha_F(X)\ \alpha_F(\alpha_F(X))$;

ma, per la proprietà C1, $\alpha_F(X) \quad \alpha_F(\alpha_F(X)) = \alpha_F(\alpha_F(X))$. Di nuovo, la proprietà B ci porta a concludere che $\alpha_F(\alpha_F(X))$ è un sottoinsieme di $\alpha_F(X)$, e l'assunto è provato. □

Un altro sistema formale corretto e completo è fornito dalle tre regole di derivazione A1, A4 ed A5. Se ne lascia la dimostrazione come esercizio. Si osservi che questa può essere data provando alternativamente o che {A1, A4, A5} è un'assiomatizzazione dotata di operatore di chiusura oppure che i due sistemi formali {A1, A2, A3} ed {A1, A4, A5} sono equivalenti per potere deduttivo.

4.4 Algoritmo di Chiusura

Se α è un sistema formale corretto e completo distinto da {A1, A2, A3}, allora per ogni *R*, *F* ed *X* si ha che $\alpha_F(X)$ coincide con la chiusura di *X* rispetto ad *F* calcolata nel sistema formale {A1, A2, A3}; per questo, nel seguito parleremo di chiusura di *X* rispetto ad *F* senza specificare altro e, quando l'insieme delle dipendenze funzionali *F* è implicito nel contesto, semplicemente di chiusura di *X* e la indicheremo con X^+. Il seguente algoritmo consente di determinare X^+ dati *X* ed *F*.

Algoritmo di Chiusura

Dati: un insieme *F* di dipendenze funzionali ed un insieme *X* di attributi.
Risultato: un sovrainsieme *V* di *X*.

Procedura

(1) $U := \emptyset$; $V := X$;

(2) Fintantoché $U \neq V$ ripetere:

(2.1) $U := V$;
(2.2) per ogni dipendenza funzionale $Y \rightarrow Z$ in *F*

se $Y \quad V$ allora $V := V \quad Z$.

***Teorema* 4.4**. Siano *F* un insieme di dipendenze funzionali ed *X* un insieme di attributi. L'Algoritmo di Chiusura calcola correttamente la chiusura di *X* rispetto ad *F*.

Dimostrazione. Sia *V* il risultato dell'Algoritmo di Chiusura. La maniera con cui l'insieme *V* viene costruito simula una derivazione da *F* della dipendenza funzionale $X \rightarrow V$ nel sistema formale {A1, A2, A3}. Infatti, l'istruzione (1) rimanda alla regola A1 e l'istruzione (2.2) alla regola A3. Pertanto, la dipendenza funzionale $X \rightarrow V$ è

derivabile da F e, per la proprietà B delle assiomatizzazioni, V è un sottoinsieme di X^+. Basterà allora dimostrare che l'insieme $U = X^+ - V$ è vuoto. Supponiamo per assurdo che $U \neq \emptyset$. Per la proprietà B delle assiomatizzazioni, la dipendenza funzionale $X \rightarrow U$ è derivabile da F e, siccome il sistema formale {A1, A2, A3} è corretto, la dipendenza funzionale $X \rightarrow U$ deve essere valida in ogni modello $M = [R, F]$. Si consideri allora un prototipo r di relazione con schema R e radice V ed una qualsiasi dipendenza funzionale $Y \rightarrow Z$ presente in F. Ora, se $Y - V \neq \emptyset$ allora banalmente la relazione r soddisfa la dipendenza funzionale $Y \rightarrow Z$. Mentre, se $Y \quad V$ allora anche Z deve essere un sottoinsieme di V, perché ad un certo passo i.esimo l'Algoritmo di Chiusura si troverà ad esaminare la dipendenza funzionale $Y \rightarrow Z$ e, trovando che Y è un sottoinsieme del valore corrente V_i di V, aggiungerà a V tutti gli attributi in Z (che non sono già in V_i) cosicché $Z \quad V_{i+1}$. Dunque, siccome $Y \quad Z \quad V$, la relazione r soddisfa la dipendenza funzionale $Y \rightarrow Z$. In definitiva, la relazione r soddisfa tutte le fipendenze funzionali in F e, quindi, è conforme ad M. Pertanto, siccome la dipendenza funzionale $X \rightarrow U$ è valida in M, r deve soddisfare anche la dipendenza funzionale $X \rightarrow U$; ma, siccome $X \quad V$ ed $U \cap V = \emptyset$, r vìola la dipendenza funzionale $X \rightarrow U$ (contraddizione). □

Esempio 4.4. Siano $F = \{\mathbf{A} \rightarrow \mathbf{D}, \mathbf{AB} \rightarrow \mathbf{E}, \mathbf{BF} \rightarrow \mathbf{E}, \mathbf{CD} \rightarrow \mathbf{F}, \mathbf{E} \rightarrow \mathbf{C}\}$ ed $X = \{\mathbf{A}, \mathbf{E}\}$. Inizialmente, $V = \{\mathbf{A}, \mathbf{E}\}$. Durante la prima scansione di F, la dipendenza funzionale $\mathbf{A} \rightarrow \mathbf{D}$ è usata per aggiungere **D** a V, e la dipendenza funzionale $\mathbf{E} \rightarrow \mathbf{C}$ per aggiungere **C** a V; così, al termine della prima esecuzione del ciclo (2) si ha $V = \{\mathbf{A}, \mathbf{C}, \mathbf{D}, \mathbf{E}\}$. Durante la seconda scansione di F, la dipendenza funzionale $\mathbf{CD} \rightarrow \mathbf{F}$ è usata per aggiungere **F** a V; così, al termine della seconda esecuzione del ciclo (2), troviamo $V = \{\mathbf{A}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}\}$. La terza scansione di F lascia V invariato e, dunque, $\{\mathbf{A}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}\}$ è il valore calcolato dall'Algoritmo di Chiusura. ◊

Analizziamo il funzionamento dell'Algoritmo di Chiusura. Il caso peggiore si ha quando, ad ogni esecuzione del ciclo (2.2), un solo attributo viene aggiunto a V. Questo realmente capita ad esempio se $R = \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_{n+1}\}$ ed $F = \{\mathbf{A}_2 \rightarrow \mathbf{A}_1, \mathbf{A}_3 \rightarrow \mathbf{A}_2, \ldots, \mathbf{A}_{n+1} \rightarrow \mathbf{A}_n\}$, quando si voglia calcolare la chiusura di $\mathbf{A}_{n+1}$ rispetto ad F. Allora, ogni esecuzione del ciclo (2.2) richiede un tempo $O(n^2)$ perché occorrono n+1 operazioni elementari per valutare l'inclusione tra due sottoinsiemi di R; inoltre, il ciclo (2) viene eseguito n volte prima di vedere soddisfatta l'uguaglianza $U = V$.

Questo esempio suggerisce come ridurre il costo computazionale dell'Algoritmo di Chiusura. Il miglioramento può essere così ottenuto. Primo, se durante la sua esecuzione per qualche dipendenza funzionale $Y \rightarrow Z$ si trova che Y è incluso in V, allora Z viene aggiunto a V e la dipendenza funzionale $Y \rightarrow Z$ non sarà più esaminata. Secondo, se si introduce una tabella che dinamicamente riporti quali dipendenze funzionali f tra quelle presenti in F hanno $det(f)$ contenuto nel valore corrente di V,

allora nella scansione di F potrà limitarsi ad esaminare solo queste dipendenze funzionali per poi subito cancellarle una volta ampliato V. Così, ogni dipendenza funzionale in F verrà esaminata solo una volta. Tenendo conto di queste osservazioni, si può raffinare l'Algoritmo di Chiusura ed otterne una versione lineare, cioè di complessità dell'ordine di $O(|X| + ||F||)$, dove il parametro

$$||F|| = \sum_{f \in F} (|det(f)| + |dip(f)|).$$

è chiamato la *dimensione* di F.

Nota. Come abbiamo visto, l'insieme X^+ viene costruito dall'Algoritmo di Chiusura usando solo le regole A1 ed A3 del sistema formale {A1, A2, A3}. Pertanto, per ogni dipendenza funzionale della forma $X \rightarrow Y$ che sia derivabile da F, possiamo trovare una derivazione da F che chiamiamo *regolare*, in cui la regola A2 è usata al più una volta cioè quando viene generata l'ultima dipendenza funzionale, cioè $X \rightarrow Y$, se la regola A3 è usata per generare dipendenze funzionali f con $det(f) = X$.

Esempio 4.2 (seguito). Una derivazione regolare della dipendenza funzionale **AB → GH** è la seguente:

f_1 = **AB → AB**		A1
f_2 = **AB → E**		in F
f_3 = **AB → ABE**		A3(f_1, f_2)
f_4 = **BE → C**		in F
f_5 = **AB → ABCE**		A3(f_3, f_4)
f_6 = **E → G**		in F
f_7 = **AB → ABCEG**		A3(f_5, f_6)
f_8 = **CG → H**		in F
f_9 = **AB → ABCEGH**		A3(f_7, f_8)
f_{10} = **AB → GH**	A2(f_9)	

◊

Vogliamo infine fare un'osservazione che ci permette di chiarire un aspetto a proposito delle relazioni conformi ad un dato modello $M = [R, F]$. Per definizione, una relazione r conforme ad M soddisfa tutte le dipendenze funzionali valide in M; questo però non esclude che r soddisfi anche altre dipendenze funzionali. Ad esempio, se r è una relazione vuota o contiene esattamente un'ennupla, allora r soddisfa tutte le possibili dipendenze funzionali applicabili ad R. Così, una questione che si pone è se esista una relazione conforme ad M che soddisfi tutte e solo le dipendenze funzionali valide in M. Mostriamo che è proprio questo il caso e che è anche facile costruire esplicitamente una tale relazione. Per ogni dipendenza funzionale f applicabile ad R che non sia valida in M, possiamo usare la dimostrazione del Lemma 4.1 per costruire una relazione $r^{(f)}$ che sia conforme ad M e che vìoli f. Data l'infinita numerabilità del

dominio di ogni attributo in R, possiamo scegliere tali relazioni in maniera tale che, prese arbitrariamente due distinte dipendenze funzionali f e g applicabili ad R che non appartengono ad $F(M)$, per ogni attributo A in R gli insiemi dei valori di A in $r^{(f)}$ ed in $r^{(g)}$ siano disgiunti. Allora, è facile vedere che la relazione

$$\bar{r} = \bowtie_{f \in F(M)} r^{(f)}$$

soddisfa tutte e solo le dipendenze funzionali valide in M. In verità, per costruire $\bar{r}$, possiamo anche prendere l'unione delle sole relazioni $r^{(f)}$ per $f \in \bar{F}$, se $\bar{F}$ è un sottoinsieme di dipendenze funzionali f non valide in M tale che, per ogni altra dipendenza funzionale f non valida in M, la condizione $f \in F(M)$ è conseguenza del fatto che $\bar{F} \cap F(M) = \emptyset$. Per esempio, se la dipendenza funzionale $X \rightarrow Y$ è stata inclusa in $\bar{F}$, allora possiamo ignorare (cioè escludere da $\bar{F}$) ogni dipendenza funzionale della forma $X' \rightarrow Y'$ se $X \subseteq X'$ e $Y' \subseteq Y$.

4.5 Test di Validità

In virtù della correttezza e completezza del sistema formale {A1, A2, A3}, abbiamo che una dipendenza funzionale f è valida in un modello $M = [R, F]$ se e solo se $dip(f)$ è un sottoinsieme di X^+. Così, per decidere se f è o meno valida in M possiamo impiegare il seguente algoritmo.

Test di Validità

Dati:	Un insieme R di attributi, un insieme F di dipendenze funzionali applicabili ad R ed una dipendenza funzionale $X \rightarrow Y$ applicabile ad R
Risultato:	Un valore della variabile logica *test.*

Procedura

(1) *test* := FALSO.
(2) Calcolare la chiusura X^+ di X.
(3) Se Y è un sottoinsieme di X^+, allora *test* := VERO.

Diremo che il Test di Validità *dà esito positivo* (rispettivamente, *negativo*) se il valore finale della variabile *test* è VERO (rispettivamente, FALSO)

***Teorema* 4.5** Dato un modello $M = [R, F]$ ed una dipendenza funzionale $X \rightarrow Y$ applicabile ad R, la dipendenza funzionale $X \rightarrow Y$ è valida in M se e solo se il test di validità dà esito positivo.

Dimostrazione. Per i Teoremi 4.3 e 4.4. □

Usando la versione lineare dell'algoritmo di Chiusura abbiamo un test di validità la cui complessità è dell'ordine di $O(|R| + ||F||)$.

Prima di chiudere questo capitolo, presentiamo un altro test di validità, il quale è basato sulla logica proposizionale.
Ad ogni dipendenza funzionale applicabile ad R associamo una formula della logica proposizionale nel modo seguente. Innazitutto, per ogni attributo A in R introduciamo una distinta variabile logica a. Sia ora $X \rightarrow A$ una dipendenza funzionale semplice applicabile ad R, con $X = \{A_1, \ldots, A_k\}$; associamo a questa dipendenza funzionale la formula della logica proposizionale

$$\bar{a}_1 \quad \ldots \quad \bar{a}_k \quad a$$

dove il letterale $\bar{a}_h$ sta per $\neg a_h$. Si osservi che questa formula è identicamente vera se e solo se l'attributo A appartiene ad X; in altri termini, una dipendenza funzionale f è banale se e solo se la formula proposizionale associata ad f è identicamente vera. Le formule proposizionali associate alle dipendenze funzionali prendono il nome di *formule condizionali*, perché ognuna di esse è equivalente logicamente ad una proposizione del tipo

$$(a_1 \quad \ldots \quad a_k) \quad a$$

dove denota il connettivo condizionale ("se … allora") della logica proposizionale. Dato un insieme $P = \{p_1, \ldots, p_n\}$ di formule condizionali, una formula condizionale p si dice *conseguenza logica* di P se ogni valutazione delle variabili logiche che renda vera ogni formula p_i in P rende vera anche p. Equivalentemente, p è una conseguenza logica di P se e solo se la formula

$$(p_1 \quad \ldots \quad p_n) \quad p$$

è identicamente vera, ovvero se e solo se la formula

$$p_1 \quad \ldots \quad p_n \quad \neg p$$

è identicamente falsa. Un metodo diretto per decidere se p è una conseguenza logica di P fa uso delle tabelle di verità. Più efficiente è invece è l'*algoritmo di risoluzione*, che andiamo a presentare. Inizialmente, viene creato un insieme di stringhe S così fatto. Ogni formula condizionale

$$\bar{a}_1 \quad \ldots \quad \bar{a}_k \quad a$$

in P viene rappresentata con la stringa di letterali

$\bar{a}_1 \ldots \bar{a}_k \; a$

che viene posta in *S*. Fino a questo punto, *S* è una rappresentazione congiuntiva della formula

$p_1 \quad \ldots \quad p_n$

Poi, se

$p = \bar{b}_1 \quad \ldots \quad \bar{b}_m \quad b,$

allora ad *S* vengono aggiunte *m*+1 stringhe ciascuna formata da un singolo letterale:

$b_1 \qquad \ldots \qquad b_m \qquad \bar{b}$

Così, alla fine *S* è una rappresentazione congiuntiva della formula

$p_1 \quad \ldots \quad p_n \quad \neg p$

L'algoritmo di risoluzione procede cercando di volta in volta una variabile a tale che a sia una stringa in *S* e ci sia una stringa in *S* che inizi per il letterale $\bar{a}$. Quando una tale variabile viene trovata, allora

il letterale $\bar{a}$ viene cancellato in ogni stringa in *S* che inizia per $\bar{a}$.

L'algoritmo di risoluzione termina se *S* contiene la stringa vuota (ε) oppure se non esiste nessuna variabile a tale che a sia una stringa in *S* e ci sia una stringa in *S* che inizi per $\bar{a}$. La conclusione è che la formula condizionale p è una conseguenza logica di P se e solo se alla fine *S* contiene la stringa vuota.

Esempio 4.6. Siano

$p = \bar{a} \quad \bar{b} \quad d$

ed

$P = \{\bar{a} \quad \bar{b} \quad c, \; \bar{b} \quad \bar{c} \quad a, \; \bar{b} \quad \bar{c} \quad d, \; \bar{c} \quad \bar{d} \quad a, \; \bar{c} \quad \bar{d} \quad b\}.$

Inizialmente, l'insieme *S* contiene le seguenti stringhe di letterali:

$\bar{a}\ \bar{b}\ c$	$\bar{b}\ \bar{c}\ a$	$\bar{b}\ \bar{c}\ d$	$\bar{c}\ \bar{d}\ a$	$\bar{c}\ \bar{d}\ b$
a	b	$\bar{d}$		

Al primo passo, scegliamo la variabile a ed il contenuto di *S* viene modificato in

$\bar{b}\ c$	$\bar{b}\ \bar{c}\ a$	$\bar{b}\ \bar{c}\ d$	$\bar{c}\ \bar{d}\ a$	$\bar{c}\ \bar{d}\ b$
a	b	$\bar{d}$		

Al secondo passo, scegliamo la variabile b ed il contenuto di *S* diventa

c	$\bar{c}$ a	$\bar{c}$ d	$\bar{c}$ $\bar{d}$ a	$\bar{c}$ $\bar{d}$ b
a	b	$\bar{d}$		

Al terzo passo, scegliamo la variabile c ed il contenuto di *S* viene modificato in

c	a	d	$\bar{d}$ a	$\bar{d}$ b
a	b	$\bar{d}$		

Al quarto passo, scegliamo la variabile d ed il contenuto di *S* viene a contenere la stringa vuota.

c	a	d	a	b
a	b	ε		

Ne concludiamo che p è una conseguenza logica di P. ◊

Il nostro interesse per le formule condizionali e per l'algoritmo di risoluzione deriva dal fatto (che ci limitiamo solo ad enunciare) che una dipendenza funzionale semplice *f* è valida in ogni modello *M* = [*R, F*] se e solo se la formula condizionale associata ad *f* è una conseguenza logica dell'insieme delle formule condizionali associate ad *F*. La dimostrazione di questo risultato si basa sulla correttezza e completezza per formule condizionali di un qualsiasi sistema formale che sia un'assiomatizzazione delle dipendenze funzionali. Come conseguenza abbiamo che come test di validità possiamo anche usare l'algoritmo di risoluzione.

Esempio 4.6 (seguito). Sia dato il modello *M* = [*R, F*] dove

$R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$

$F = \{\mathbf{AB} \rightarrow \mathbf{C}, \mathbf{BC} \rightarrow \mathbf{A}, \mathbf{BC} \rightarrow \mathbf{D}, \mathbf{CD} \rightarrow \mathbf{A}, \mathbf{CD} \rightarrow \mathbf{B}\}$

Ci chiediamo se {**A**, **B**} è una chiave di *M*. Vista la presenza della dipendenza **AB** → **C** in *F*, la risposta a tale quesito sarà positiva se e solo se la dipendenza funzionale **AB** → **D** è valida in *M*. Grazie al Teorema 2.7, questo è vero se e solo se la formula condizionale associata alla dipendenza funzionale **AB** → **D** è una conseguenza logica dell'insieme delle formule condizionali associate alle dipendenze in *F*. Ma questo è proprio quello che abbiamo provato applicando l'algoritmo di risoluzione. Dunque la dipendenza funzionale **AB** → **D** è valida nel modello *M* e, quindi, {**A**, **B**} è una chiave di *M*. ◊

4.6 Test di Normalità

Siamo finalmente in grado di formulare un algoritmo per decidere se un modello, privo di attributi composti sovradimensionati, è o meno normale.

Test di Normalità

Dati: Un insieme R di attributi, ed un insieme F di dipendenze funzionali applicabili ad R

Risultato: Un valore della variabile logica *test.*

Procedura

(1) *test* := VERO.

(2) Per ogni dipendenza funzionale nonbanale f in F,

Calcolare la chiusura di *det*(f).
Se la chiusura di *det*(f) è diversa da R, allora *test* := FALSO ed Uscire.

Diremo che il Test di Normalità *dà esito positivo* (rispettivamente, *negativo*) se il valore finale della variabile *test* è VERO (rispettivamente, FALSO)

Esempio 4.7 Consideriamo il modello con schema {**CITTÀ**, **VIA**, **CAP**} in cui siano definite le due dipendenze funzionali:

{**CITTÀ**, **VIA**} → **CAP** **CAP** → **CITTÀ**

Siccome la chiusura di {**CAP**} è {**CITTÀ**, **CAP**}, {**CAP**} non è una sovrachiave e, quindi, il Test di Normalità dà esito negativo. ◊

***Teorema* 4.6** Sia $M = [R, F]$ un modello privo di attributi composti sovradimensionati. Il modello M è normale se e solo se il test di normalità dà esito positivo.

Dimostrazione. Se M è normale allora, per ogni dipendenza funzionale nonbanale valida f, la chiusura di *det*(f) coincide con R; in particolare per tutte le dipendenze funzionali nonbanali presenti in F. Pertanto, il test di normalità darà esito positivo.

Viceversa, assumiamo che il test di normalità dia esito positivo, cioè che, per ogni dipendenza funzionale nonbanale f in F, la chiusura di *det*(f) coincida con R. Supponiamo poi che esista una dipendenza funzionale valida $X \rightarrow Y$ tale che $X^+ \neq R$. Dimostreremo che $X \rightarrow Y$ è una dipendenza funzionale banale. Infatti, per ogni dipendenza funzionale nonbanale f in F, *det*(f) non può essere un sottoinsieme di X perché, altrimenti, si avrebbe che la chiusura di X sarebbe strettamente contenuta nella chiusura di un suo sottoinsieme, la qual cosa contraddice la proprietà C2 dell'operatore α_F. Ma allora, se calcoliamo X^+ con l'Algoritmo di Chiusura, otterremo $X^+ = X$. D'altra parte, siccome la dipendenza funzionale $X \rightarrow Y$ è valida, l'insieme Y deve essere un sottoinsieme di X^+ e, quindi, di X. Vale a dire, la dipendenza funzionale $X \rightarrow Y$ non può che essere banale. □

Usando la versione lineare dell'Algoritmo di Chiusura abbiamo un test di normalità la cui complessità è dell'ordine di $O(|F| \times ||F||)$.

CAPITOLO 5

PROBLEMI SU CHIAVI

5.1 Ricerca delle chiavi

Dato un modello $M = [R, F]$, quante sono le chiavi di M? Come dimostrato dal seguente esempio, il loro numero può crescere in maniera esponenziale con $|R|$ ed $|F|$ così che pensare di generare tutte le chiavi di un modello è un problema provatamente intrattabile.

Esempio 5.1 Siano $R = \{\mathbf{A_1}, \mathbf{B_1}, \mathbf{A_2}, \mathbf{B_2}, \ldots, \mathbf{A_k}, \mathbf{B_k}\}$ ed $F = \{\mathbf{A_1} \rightarrow \mathbf{B_1}, \mathbf{B_1} \rightarrow \mathbf{A_1}, \mathbf{A_2} \rightarrow \mathbf{B_2}, \mathbf{B_2} \rightarrow \mathbf{A_2}, \ldots, \mathbf{A_k} \rightarrow \mathbf{B_k}, \mathbf{B_k} \rightarrow \mathbf{A_k}\}$. È chiaro che un sottoinsieme X di R è una chiave se e solo se contiene k attributi e per ogni h, $1 \leq h \leq k$, X contiene o $\mathbf{A_h}$ oppure $\mathbf{B_h}$. Siccome di siffatti insiemi ne esistono 2^k, abbiamo che il numero di chiavi è 2^k. ◊

Abbandoniamo dunque l'idea di generare tutte le chiavi di un modello, ed accontentiamoci di poterne trovare una. Il seguente algoritmo, la cui complessità è dell'ordine di $O(|R| \times ||F||)$, fa proprio questo.

Calcolo di una Chiave

Dati: Un insieme R di attributi ed un insieme F di dipendenze funzionali applicabili ad R

Risultato: Un insieme di attributi X.

Procedura

(1) $X := R$.

(2) Per ogni A in X

$Y := X - \{A\}$;
calcolare la chiusura di Y;
se $Y^+ = R$, allora $X := Y$.

Il seguente esempio mostra un'applicazione dell'algoritmo ed evidenzia la dipendenza del risultato che se ne ottiene dall'ordine in cui gli attributi sono esaminati.

Esempio 5.2 Siano $R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}\}$ ed $F = \{\mathbf{AB} \rightarrow \mathbf{C}, \mathbf{AC} \rightarrow \mathbf{B}, \mathbf{D} \rightarrow \mathbf{E}\}$. Se l'attributo $\mathbf{C}$ è esaminato prima dell'attributo $\mathbf{B}$, allora la chiave che si trova è $\{\mathbf{A}, \mathbf{B}, \mathbf{D}\}$, altrimenti è $\{\mathbf{A}, \mathbf{C}, \mathbf{D}\}$. ◊

5.2 Chiavi minime

Confortati dal successo, ci spingiamo oltre ed andiamo alla ricerca di una *chiave minima* cioè di una chiave col minor numero di attributi. Dimostreremo che questo problema è intrattabile e che appartiene alla classe dei cosiddetti problemi NP-ardui, per i quali non esiste a tutt'oggi un algoritmo polinomiale né, si congettura, mai se ne potrà trovare uno (è questa la famosa congettura $P \neq NP$). Per dimostrarlo, consideriamo il seguente problema decisionale

> *p*. Dati un insieme R di attributi, un insieme F di dipendenze funzionali ed un intero positivo $k \leq |R|$, esiste una sovrachiave di cardinalità non superiore a k ?

È chiaro che, se si sapesse risolvere il problema della chiave minima, si saprebbe risolvere il problema *p*. Viceversa, supponendo di saper risolvere il problema *p*, potremmo determinare una chiave minima prendendo la sovrachiave che si trova risolvendo il problema *p* in corrispondenza del più piccolo valore di k per cui si ottiene una risposta affermativa (cioè provando con $k = 1, 2, \ldots$ finché non si otterrà una risposta affermativa). Pertanto, se esistesse un algoritmo polinomiale per il problema *p*, allora anche potremo trovare una chiave minima in tempo polinomiale, e viceversa. Dunque la questione che ora si pone è la seguente: esiste un algoritmo polinomiale per risolvere il problema *p*? Risponderemo a questa domanda dimostrando che il problema *p* non è meno difficile del seguente problema decisionale π (notoriamente *NP*-completo)

> π. Dati un grafo $G = (V, E)$ ed un intero positivo $k \leq |V|$, esiste un sottoinsieme U di V con $|U| \leq k$ tale che ogni arco di G abbia almeno un estremo in U ?

Più precisamente, dimostreremo che π è *riconducibile* al problema *p* nel senso che esiste una trasformazione polinomiale di π in un esempio p^* di *p* tale che la questione posta da π ammette risposta positiva se e solo se la questione posta dal problema p^* ammette anch'essa risposta positiva. In tal modo, avremo la prova che il problema π non è più difficile del problema *p*. Per prima cosa, vediamo come viene costruito p^*. Si procede alla maniera seguente. Per ogni vertice v di G introduciamo un attributo che indichiamo con $A(v)$, e per ogni arco e di G introduciamo un attributo che indichiamo con $A(e)$. Siano poi

$$X_G = \{A(e): e \quad E\} \qquad Y_G = \{A(v): v \quad V\} \qquad R_G = X_G \quad Y_G.$$

Sia poi F_G l'insieme delle $2|E| + 1$ dipendenze funzionali così definite:

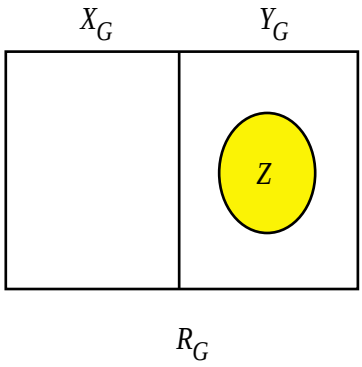
> per ogni arco e di G, $A(v) \rightarrow A(e)$ se v è un estremo di e,

$X_G \rightarrow Y_G$.

Allora, il problema p^* suona così:

> Dati l'insieme R_G di attributi, l'insieme F_G di dipendenze funzionali e l'intero positivo k, esiste una sovrachiave di cardinalità non superiore a k ?

Se chiamiamo *copertura* di G un qualsiasi insieme del tipo di U, dimostrare che la questione posta da π ammette risposta positiva se e solo se la questione posta dal problema p^* ammette anch'essa risposta positiva significa provare che esiste una copertura di G di cardinalità non superiore a k se e solo se esiste una sovrachiave di $[R_G, F_G]$ di cardinalità non superiore a k. Per far questo, basterà provare che le coperture di G corrispondono alle sovrachiavi di $[R_G, F_G]$ e viceversa. A questo scopo, prendiamo un qualsiasi sottoinsieme U di V e sia Z il sottoinsieme di Y_G corrispondente ad U, cioè $Z = \{A(u): u \quad U\}$.



Qual è la chiusura Z^+ di Z rispetto ad F_G? Se applichiamo l'Algoritmo di Chiusura, aggiungeremo a Z tutti gli attributi $A(e)$ in X_G tali che e abbia un estremo in U; a questo punto, avremo finito ottenendo

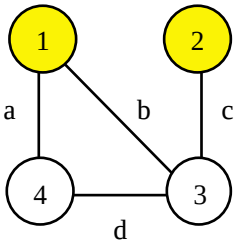
$$Z^+ = Z \quad \{A(e): \text{un estremo di } e \text{ appartiene ad } U\},$$

a meno che non siamo riusciti ad aggiungere a Z tutti gli attributi in X_G (cioè a meno che U non sia una copertura di G), perché allora, grazie alla dipendenza funzionale $X_G \rightarrow Y_G$, potremo aggiungere anche tutti gli attributi in $Y_G – Z$ ottenendo

$$Z^+ = R_G.$$

Ne segue che Z è una sovrachiave se e solo se U è una copertura di G.

Esempio 5.3 Sia G il grafo mostrato in Figura 5.1. L'insieme $U = \{1, 2\}$ non è, ovviamente, una copertura di G.



U = {1, 2}

Figura 5.1

All'insieme di vertici U corrisponde l'insieme di attributi $Z = \{\mathbf{A}(1), \mathbf{A}(2)\}$ la cui chiusura rispetto ad F_G non è una sovrachiave perché non contiene i tre attributi $\mathbf{A}(3)$, $\mathbf{A}(4)$ ed $\mathbf{A}(3, 4)$ (vedi Figura 5.2).

X_G		Y_G	
A(a)	A(c)	A(1)	A(2)
A(b)	A(d)	A(4)	A(3)

Z = {A(1), A(2)}

Figura 5.2

◊

Il seguente risultato asserisce che non esiste algoritmo polinomiale per risolvere il problema p a meno che $P = NP$.

***Teorema* 5.1** Il problema p è NP-completo.
Dimostrazione. Abbiamo da provare che:

(1) è possibile verificare in tempo polinomiale se un assegnato sottoinsieme di R è una sovrachiave con un numero di attributi non superiore a k, e

(2) esiste un problema decisionale NP-completo π riconducibile a p.

(1). Per decidere se un assegnato sottoinsieme X di R sia una sovrachiave basta applicare l'Algoritmo di Chiusura che è polinomiale. Inoltre, la verifica che $|X| \leq k$ può ovviamente essere fatta in tempo polinomiale.

(2). Basta prendere il problema π del ricoprimento di un grafo. □

L'ultima questione sulle chiavi che trattiamo è quella dell'unicità di una chiave. Il seguente teorema caratterizza i casi in cui esiste un'unica chiave.

***Teorema* 5.2** Dato un modello $M = [R, F]$, sia $Z_f = dip(f)–det(f)$ per ogni f in F e sia $Y = R – (\ _f\ _F Z_f)$. Esiste un'unica chiave di M e questa allora coincide con Y se e solo se Y è una sovrachiave di M.

Dimostrazione. (*se*) Mostreremo che, se l'insieme Y è una sovrachiave di M, allora Y è l'unica chiave di M. Per far questo, sarà sufficiente provare che l'insieme Y è contenuto in ogni sovrachiave di M. Supponiamo per assurdo che esista una sovrachiave X di M che non contenga Y. Sia allora A un attributo in $Y–X$. Siccome A è in Y esso non appartiene ad alcun insieme Z_f e quindi, se compare in un $dip(f)$ per qualche f in F, deve allora comparire anche in $det(f)$. Ne viene che, quando calcoliamo X^+ con l'Algoritmo di Chiusura, non troviamo mai una dipendenza funzionale f in F che produca l'aggiunta di A ad X^+. Pertanto, X^+ non contiene A e quindi X non è una sovrachiave di M (contraddizione).

(*solo se*) Per ogni f in F, l'insieme $R–Z_f$ è ovviamente una sovrachiave di M. Ora, siccome esiste un'unica chiave di M, allora questa è contenuta in ogni sovrachiave $R–Z_f$ e, quindi, nella loro intersezione $\cap_{f\ F}(R – Z_f)$, che è uguale ad $R – (\ _f\ _F Z_f) = Y$. Dunque, l'insieme Y è una sovrachiave. □

CAPITOLO 6

EQUIVALENZA TRA MODELLI

6.1 Test di equivalenza

Nella progettazione di una base di dati, al momento di specificare il modello di una variabile relazionale, potrebbero presentarsi due o più "alternative", e due modelli $M = [R, F]$ ed $N = [R, G]$ sono tra loro *alternativi* se $F(M) = F(N)$. Allora, è buona regola scegliere il modello che abbia l'insieme di dipendenze di dimensione minima.

Esempio 6.1 Siano $M = [R, F]$ ed $N = [R, G]$, dove $R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}\}$, $F = \{\mathbf{A} \rightarrow \mathbf{B}, \mathbf{A} \rightarrow \mathbf{C}, \mathbf{CD} \rightarrow \mathbf{E}\}$ e $G = \{\mathbf{A} \rightarrow \mathbf{BC}, \mathbf{CD} \rightarrow \mathbf{E}\}$. I modelli M ed N sono tra loro alternativi. Infatti, la dipendenza funzionale $\mathbf{A} \rightarrow \mathbf{BC}$ in G è valida in M per la proprietà additiva, e la dipendenza funzionale $\mathbf{CD} \rightarrow \mathbf{E}$ lo è perché è già in F. D'altro canto, le dipendenze funzionali $\mathbf{A} \rightarrow \mathbf{B}$ ed $\mathbf{A} \rightarrow \mathbf{C}$ in F sono valide in N per la proprietà proiettiva, e la dipendenza funzionale $\mathbf{CD} \rightarrow \mathbf{E}$ lo è perché è già in G. ◊

Se due modelli $M = [R, F]$ ed $N = [R, G]$ sono alternativi, allora gli insiemi di dipendenze funzionali F e G sono detti essere *equivalenti*. Discuteremo ora il problema dell'equivalenza, mentre rimandiamo al paragrafo 6.3 il problema della ricerca di un modello alternativo di costo minimo equivalente ad un modello assegnato.

Il primo problema si risolve facilmente sfruttando il Test di Validità.

Test di Equivalenza

Dati: Un insieme R di attributi e due insiemi F e G di dipendenze funzionali applicabili ad R

Risultato: Il valore della variabile logica *test*

Procedura

(1) *test* := VERO.

(2) Per ogni g in G
Applicare il Test di Validità per decidere se g è valida in M;
se il Test di Validità dà esito negativo, allora *test* := FALSO
ed Uscire.

(3) Per ogni f in F
Applicare il Test di Validità per decidere se f è valida in N;

se il Test di Validità dà esito negativo, allora *test* := FALSO ed Uscire.

Diremo che il Test di Equivalenza *dà esito positivo* (rispettivamente, *negativo*) se il valore finale della variabile *test* è VERO (rispettivamente, FALSO).

6.2 Forma Canonica

Un insieme F di dipendenze funzionali è

— *minimale* se non esiste alcun sottoinsieme proprio G di F tale che G sia equivalente ad F,

— *ridotto* se ogni dipendenza funzionale f in F è "irriducibile", nel senso che non esiste nessun sottoinsieme proprio X di $det(f)$ tale che l'insieme

$$(F - \{f\}) \quad \{X \rightarrow dip(f)\}$$

risulta equivalente ad F,

— *in forma canonica* se è minimale, ridotto ed ogni dipendenza funzionale f in F è semplice.

Per decidere se un insieme F di dipendenze funzionali è minimale facciamo uso della seguente nozione di dipendenza funzionale "ridondante". Una dipendenza funzionale f presente in F è *ridondante* se $F-\{f\}$ è equivalente ad F. Per appurarlo basta applicare il seguente *Test di Ridondanza.*

Test di Ridondanza

Dati: Un insieme F di dipendenze funzionali ed una dipendenza funzionale f in F.
Risultato: Un valore della variabile logica *test.*

Procedura

(1) *test* := FALSO.
(2) $G := F-\{f\}$.
(3) Se l'applicazione del Test di Validità ad f e G dà esito positivo, allora *test* := VERO.

Diremo che il Test di Ridondanza *dà esito positivo* (rispettivamente, *negativo*) se il valore finale della variabile *test* è VERO (rispettivamente, FALSO).

Così, per decidere se un insieme F di dipendenze funzionali è minimale è sufficiente applicare il Test di ridondanza ad ognuna delle dipendenze funzionali presenti in F. Concluderemo che F è minimale se e solo se ogni volta il Test di ridondanza dà esito negativo. Se invece F non fosse minimale, possiamo con la seguente procedura trovare un insieme minimale di dipendenze funzionali che sia equivalente ad F.

Algoritmo di Semplificazione

Dati: Un insieme F di dipendenze funzionali.
Risultato: Un sottoinsieme G di F.

Procedura

(1) $G := F$.

(2) Per ogni dipendenza funzionale f in F

Applicare il Test di Ridondanza ad f e G;
se il Test di Ridondanza dà esito positivo allora $G := G - \{f\}$.

Vogliamo avvertire che il risultato dell'Algoritmo di Semplificazione può dipendere dall'ordine in cui le dipendenze funzionali in F vengono esaminate. Così, con due diversi ordinamenti si possono ottenere due distinte forme minimali.

Esempio 6.2 Consideriamo l'insieme $F = \{\mathbf{A} \rightarrow \mathbf{B},\ \mathbf{A} \rightarrow \mathbf{C},\ \mathbf{B} \rightarrow \mathbf{A},\ \mathbf{B} \rightarrow \mathbf{C}\}$. Si osservi che le dipendenze funzionali $\mathbf{A} \rightarrow \mathbf{C}$ e $\mathbf{B} \rightarrow \mathbf{C}$ sono ridondanti in F. Supponiamo ora di applicare l'Algoritmo di Semplificazione e di esaminare le dipendenze funzionali in F nello stesso ordine in cui le abbiamo elencate. Allora, delle due dipendenze ridondanti $\mathbf{A} \rightarrow \mathbf{C}$ e $\mathbf{B} \rightarrow \mathbf{C}$, andremo ad eliminare solo la seconda. Elimineremmo invece solo la prima se, ad esempio, esaminassimo le dipendenze funzionali nell'ordine $\mathbf{A} \rightarrow \mathbf{B}$, $\mathbf{B} \rightarrow \mathbf{C}$, $\mathbf{B} \rightarrow \mathbf{A}$, $\mathbf{A} \rightarrow \mathbf{C}$. ◊

Il prossimo algoritmo permette di generare un insieme ridotto di dipendenze funzionali equivalente ad uno assegnato.

Algoritmo di Riduzione

Dati: Un insieme F di dipendenze funzionali.
Risultato: Un insieme G di dipendenze funzionali.

Procedura

(1) $G := F$.

(2) Per ogni dipendenza funzionale f: $X \rightarrow Y$ in F con $|X| \neq 1$

per ogni attributo A in X

$Z := X - \{A\}$;
calcolare la chiusura di Z;
se $Y \quad Z^+$ allora $X := Z$;

$G := (G - \{f\}) \quad \{X \rightarrow Y\}$.

Utilizzando infine gli Algoritmi di Riduzione e di Semplificazione è facile (e lo si lascia come esercizio) trovare un modello in forma canonica equivalente ad un modello assegnato. se è minimale ed ogni dipendenza funzionale è semplice ed irriducibile.

6.3 Ottimimalità

Il seguente problema:

> q. Dato un insieme F di dipendenze funzionali, trovare un insieme di dipendenze funzionali che sia equivalente ad F ed abbia dimensione minima.

è intrattabile e lo proveremo dimostrando che esiste un problema NP-arduo riconducibile al problema q. Come problema NP-arduo prendiamo il problema della chiave minima

> Dati un insieme R di attributi ed un insieme F di dipendenze funzionali, trovare una chiave di cardinalità minima.

e mostriamo che esso è riconducibile a q.
Siano A e B due attributi estranei ad R e siano

$R' = R \quad \{A, B\}, \qquad f = R \quad \{A\} \rightarrow B \qquad$ ed $\qquad F' = F \quad \{f\}$.

Consideriamo il seguente esempio q' del problema q:

> q'. Dato l'insieme F' di dipendenze funzionali, trovare un insieme di dipendenze funzionali che sia equivalente ad F' ed abbia dimensione minima.

Dimostreremo che una soluzione del problema q' permette di costruire facilmente una soluzione del problema della chiave minima. Più precisamente, data una soluzione G del problema q', dimostreremo che G contiene una e solo una dipendenza funzionale g con $B \quad dip(g)$, e che l'insieme $det(g) - \{A\}$ è una chiave minima di $[R, F]$. Per far questo, utilizzeremo la seguente ovvia proprietà della chiusura di un insieme di attributi rispetto ad F' (e, quindi, rispetto a G): per ogni sottoinsieme X di R', indicata con X^* la chiusura di X rispetto ad F', si ha che

> se A appartiene ad X (per i due casi possibili si veda Figura 5.3) e se $X \cap R$ è una sovrachiave di $[R, F]$, allora $X^* = R'$;

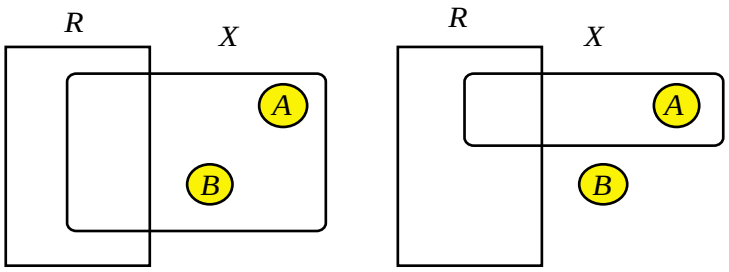


Figura 6.3

> altrimenti (per i due casi possibili si veda Figura 6.4), $X^* = (X - \{B\})^+ \quad X - R$.

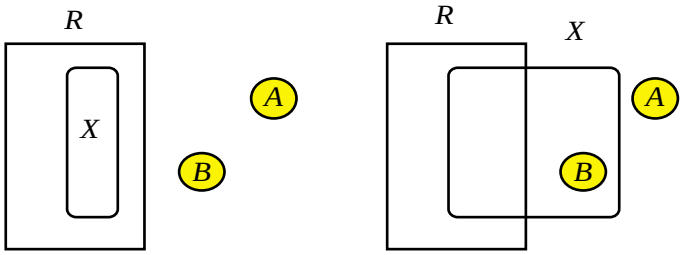


Figura 6.4

Siccome G è equivalente ad F', deve esistere in G almeno una dipendenza funzionale g tale che $B \quad dip(g)$ perché altrimenti la dipendenza funzionale f non sarebbe implicata da G. Inoltre, $det(g)$ non può contenere B perché altrimenti $dip(g)$ sarebbe riducibile eliminando B e G non avrebbe dimensione minima. Ora, per la proprietà summenzionata di α_G, $det(g)$ deve essere della forma $det(g) = X \quad \{A\}$ dove X è una

sovrachiave di [*R*, *F*]. Ma, siccome neanche *det*(*g*) può essere riducibile, *X* deve essere una chiave minima di [*R*, *F*]. Infine, non può esistere in *G* un'altra dipendenza funzionale *g'* con *B* *dip*(*g'*) perché allora *det*(*g'*) sarebbe ancora della forma *det*(*g'*) = *X'* {*A*} dove *X'* è una chiave minima di [*R*, *F*] e *dip*(*g'*) potrebbe essere ridotto eliminando *B*. Dunque, esiste in *G* una e una sola dipendenza funzionale *g* con *B* *dip*(*g*) sicché, una volta che questa è stata trovata, si può ottenere una chiave minima di [*R*, *F*] semplicemente prendendo l'insieme *det*(*g*) – {*A*}.

CAPITOLO 7

NORMALIZZAZIONE

Consideriamo ancora una volta la Base di Dati Aeroportuali ***D*** che contiene la variabile relazionale **partenze** il cui modello M ha $R = \{\textbf{VOLO}, \textbf{DATA}, \textbf{PILOTA}, \textbf{ORA}\}$ ed $F = \{\textbf{VOLO} \rightarrow \textbf{ORA}, \{\textbf{DATA}, \textbf{PILOTA}, \textbf{ORA}\} \rightarrow \textbf{VOLO}, \{\textbf{VOLO}, \textbf{DATA}\} \rightarrow \textbf{PILOTA}\}$. Abbiamo visto che M non è un modello normale. Vediamo allora come sarebbero andate le cose se avessimo progettato la Base di Dati Aeroportuali facendo uso, anziché della variabile relazionale **partenze**, delle due variabili relazionali **giornale** e **imbarco**, a cui avessimo assegnato rispettivamente i modelli $M_1 = [R_1, F_1]$ ed $M_2 = [R_2, F_2]$ con

$R_1 = \{\textbf{VOLO}, \textbf{DATA}, \textbf{PILOTA}\}$ $\quad F_1 = \{\{\textbf{VOLO}, \textbf{DATA}\} \rightarrow \textbf{PILOTA}\}$

$R_2 = \{\textbf{VOLO}, \textbf{ORA}\}$ $\quad F_2 = \{\textbf{VOLO} \rightarrow \textbf{ORA}\}$.

È facile vedere che entrambi i modelli M_1 ed M_2 sono normali. Sia ***D′*** la Base di Dati Aeroportuali così ottenuta, cioè ***D′*** = ***D*** – {**partenze**} {**giornale**, **imbarco**}. Siamo sicuri che le cose ora vanno nel giusto verso? Quello che ci aspetteremmo è che le due basi di dati ***D*** e ***D′*** diano le stesse informazioni. Consideriamo allora le tre domande poste (nell'Algebra Relazionale) a ***D***

E_1 = (**partenze**)
$E_2 = \pi_{\{\textbf{VOLO}, \textbf{DATA}, \textbf{PILOTA}\}}$ (**partenze**)
$E_3 = \pi_{\{\textbf{VOLO}, \textbf{ORA}\}}$ (**partenze**)

e le correspondenti domande poste a ***D′***:

E'_1 = (**giornale**) (**imbarco**)
E'_2 = (**giornale**)
E'_3 = (**partenze**)

Ci aspettiamo che i valori delle espressioni E_i ed E'_i, $1 \leq i \leq 3$, siano identici in ogni istante. Sia r la relazione di nome **partenze** contenuta nello stato attuale D di ***D*** ad un certo istante e siano r_1 ed r_2 rispettivamente le relazioni di nome **giornale** e **imbarco** contenute nello stato attuale D' di ***D′*** allo stesso istante. Ovviamente, r è una relazione conforme ad M così come r_1 e r_2 sono relazioni conformi rispettivamente ad M_1 ed M_2. Ora, per i valori delle espressioni E_i su D abbiamo:

$E_1(D) = r$

$E_2(D) = \pi_{\{\mathbf{VOLO}, \mathbf{DATA}, \mathbf{PILOTA}\}}(r)$
$E_3(D) = \pi_{\{\mathbf{VOLO}, \mathbf{ORA}\}}(r)$

e per i valori delle espressioni E'_i su D' abbiamo:

$E'_1(D') = r_1 \quad r_2$
$E'_2(D') = r_1$
$E'_3(D') = r_2$

Pertanto, dato $D = \{r, \ldots\}$ le relazioni di nome **giornale** e **imbarco** contenute in D' devono essere rispettivamente $\pi_{R_1}(r)$ e $\pi_{R_2}(r)$, ed inoltre deve aversi che

$r = \pi_{R_1}(r) \quad \pi_{R_2}(r)$.

Viceversa, dato $D' = \{r_1 , r_2, \ldots\}$ la relazione $r_1 \quad r_2$ deve essere conforme al modello *M*.

Se queste condizioni sono soddisfatte in ogni istante siamo certi che $\boldsymbol{D}$ e $\boldsymbol{D'}$ sono basi di dati equivalenti, cioè con lo stesso contenuto informativo. Altrimenti, abbiamo che la sosituzione di $\boldsymbol{D}$ con $\boldsymbol{D'}$ ha provocato la perdita di informazione. Nei prossimi paragrafi preciseremo il significato dell'espressione "perdita di informazione". Dopodiché, daremo le condizioni ed i metodi per normalizzare il modello di una variabile relazionale che non sia normale.

7.1 L'operatore di Proiezione-Join

Sia *R* un insieme di attributi e sia $\boldsymbol{S} = \{R_1, \ldots, R_k\}$ un ricoprimento di *R*, cioè gli insiemi $R_1, \ldots, R_k$ non sono confrontabili rispetto all'inclusione e la loro unione coincide con *R*. Due relazioni *r* ed *r'*, entrambe con schema *R*, si dicono *equivalenti* rispetto ad $\boldsymbol{S}$, $r \equiv_{\boldsymbol{S}} r'$, se

$\pi_{R_1}(r) = \pi_{R_1}(r'), \ \ldots, \pi_{R_k}(r) = \pi_{R_k}(r')$.

Si osservi che se $r \equiv_{\boldsymbol{S}} r'$ allora si ha anche $r \equiv_{\boldsymbol{S}} r \quad r'$. Pertanto, se indichiamo con $[r]_{\boldsymbol{S}}$ la classe di equivalenza rispetto ad $\boldsymbol{S}$ che contiene la relazione *r*, allora anche l'unione delle relazioni in $[r]_{\boldsymbol{S}}$ appartiene a $[r]_{\boldsymbol{S}}$. Chiameremo la relazione data dall'unione delle relazioni in $[r]_{\boldsymbol{S}}$ l'*elemento massimale* di $[r]_{\boldsymbol{S}}$.

Esempio 7.1 Siano **A**, **B** e **C** tre attributi binari. Si consideri la relazione *r* con schema {**A**, **B**, **C**} riportata in Tabella 7.1.

ABC
000
010

101

Tabella 7.1

Se $\boldsymbol{S}$ = {{**A**, **B**}, {**B**, **C**}}, allora $[r]_{\boldsymbol{S}}$ contiene sei relazioni e l'elemento massimale di $[r]_{\boldsymbol{S}}$ è la relazione riportata in Tabella 7.2.

ABC
000
001
010
100
101

Tabella 7.2 ◊

Data una relazione r con schema R ed un ricoprimento $\boldsymbol{S} = \{R_1, \ldots, R_k\}$ di R, consideriamo l'operatore relazionale $\psi_{\boldsymbol{S}}$ così definito:

$$\psi_{\boldsymbol{S}}(r) = \pi_{R_1}(r) \quad \ldots \quad \pi_{R_k}(r) =$$

$$= \{t \quad dom(R): \quad t_1, \ldots, t_k \quad r \quad t_1[R_1] = t[R_1], \ldots, t_k[R_k] = t[R_k]\}$$

prende il nome di operatore di *proiezione-join* generato da $\boldsymbol{S}$.

Esempio 7.1 (seguito). Sia r la relazione riportata in Tabella 7.1 e sia $\boldsymbol{S} = \{\{\mathbf{A}, \mathbf{B}\}, \{\mathbf{B}, \mathbf{C}\}\}$. Allora la relazione $\psi_{\boldsymbol{S}}(r)$ è quella riportata in Tabella 7.2. ◊

L'operatore $\psi_{\boldsymbol{S}}$ gode delle proprietà della chiusura. Infatti, la proprietà

$$r \quad \psi_{\boldsymbol{S}}(r)$$

segue dal fatto che, se $t \quad r$, allora $t \quad \psi_{\boldsymbol{S}}(r)$ (basta prendere $t_1 = \ldots = t_k = t$).
Inoltre, vale la proprietà

se $r \quad r'$ allora $\psi_{\boldsymbol{S}}(r) \quad \psi_{\boldsymbol{S}}(r')$

perché, se $t \quad \psi_{\boldsymbol{S}}(r)$, allora

$$t_1, \ldots, t_k \quad r \quad t_1[R_1] = t[R_1], \ldots, t_k[R_k] = t[R_k]$$

ma, siccome $r \quad r'$, le ennuple $t_1, \ldots, t_k$ appartengono anche ad r' e quindi $t \quad \psi_{\boldsymbol{S}}(r')$.
Infine per provare la proprietà

$$\psi_{\boldsymbol{S}}(\psi_{\boldsymbol{S}}(r)) = \psi_{\boldsymbol{S}}(r)$$

basta dimostrare che

$$\psi_{\boldsymbol{S}}(\psi_{\boldsymbol{S}}(r)) \quad \psi_{\boldsymbol{S}}(r)$$

Ora per definizione abbiamo

$$\psi_{\boldsymbol{S}}(\psi_{\boldsymbol{S}}(r)) = \{t \quad dom(R): \quad \tau_1, \ldots, \tau_k \quad \psi_{\boldsymbol{S}}(r) \quad \tau_1[R_1] = t[R_1], \ldots, \tau_k[R_k] = t[R_k]\}$$

ma, per ogni h, $1 \leq h \leq k$, l'appartenenza di τ_h a $\psi_{\boldsymbol{S}}(r)$ richiede l'esistenza in r di un'ennupla t_h (che dipende da h) tale che $\tau_h[R_h] = t_h[R_h]$. Pertanto, t appartiene anche a $\psi_{\boldsymbol{S}}(r)$.

Una relazione r con schema R si dice essere un *punto fisso* di $\psi_{\boldsymbol{S}}$ se $\psi_{\boldsymbol{S}}(r) = r$. Naturalmente, siccome $\psi_{\boldsymbol{S}}(\psi_{\boldsymbol{S}}(r)) = \psi_{\boldsymbol{S}}(r)$, la relazione $\psi_{\boldsymbol{S}}(r)$ è un punto fisso di $\psi_{\boldsymbol{S}}$ (vedi Figura 7.1).

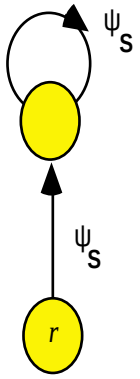


Figura 7.1

Dimostreremo ora che $\psi_{\boldsymbol{S}}(r)$ coincide con l'elemento massimale di $[r]_{\boldsymbol{S}}$ (vedi Figura 7.2).

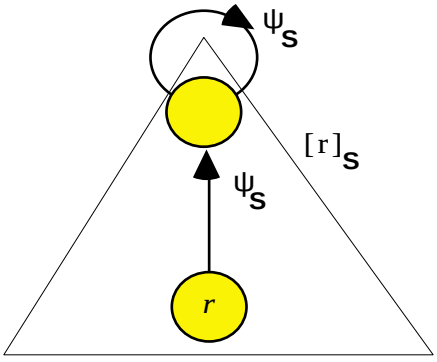


Figura 7.2

***Lemma* 7.1**. Sia r una relazione con schema R e sia $\boldsymbol{S}$ un ricoprimento di R. Allora $\psi_{\boldsymbol{S}}(r)$ è l'elemento massimale della classe $[r]_{\boldsymbol{S}}$.

Dimostrazione. Sia r^* l'elemento massimale della classe $[r]_{\boldsymbol{S}}$. Si osservi che, siccome $r' \equiv_{\boldsymbol{S}} \psi_{\boldsymbol{S}}(r')$ per ogni r' in $[r]$, la relazione $\psi_{\boldsymbol{S}}(r^*)$ è contenuta in r^*. Allora, per la

prima delle proprietà di chiusura di ψ_S, si ha $\psi_S(r^*) = r^*$. D'altra parte, $\psi_S(r) = \psi_S(r')$ per ogni r' in $[r]_S$ e, quindi, $\psi_S(r) = \psi_S(r^*) = r^*$. □

Sia R un insieme di attributi finito non vuoto. Dati due ricoprimenti $\boldsymbol{S}_1$ ed $\boldsymbol{S}_2$ di R, diremo che $\boldsymbol{S}_1$ è *più fine* di $\boldsymbol{S}_2$, $\boldsymbol{S}_1 \prec \boldsymbol{S}_2$, se ogni insieme in $\boldsymbol{S}_1$ è sottoinsieme di un insieme in $\boldsymbol{S}_2$. Particolari ricoprimenti di R sono la *partizione puntuale* di R, cioè $\boldsymbol{P}_0 = \{\{A\}: A \in R\}$, e la *partizione banale* di R, cioè $\boldsymbol{P}_1 = \{R\}$. Ovviamente, la partizione puntuale è il ricoprimento più fine di R e la partizione banale è il ricoprimento meno fine di R. Definiamo infine l'addizione (+) come

$\boldsymbol{S}_1 + \boldsymbol{S}_2$ è il più fine dei ricoprimenti di R che sono meno fini tanto di $\boldsymbol{S}_1$ che di $\boldsymbol{S}_2$.

Si lascia come esercizio la dimostrazione che gli insiemi in $\boldsymbol{S}_1 + \boldsymbol{S}_2$ sono gli insiemi massimali nella famiglia $\boldsymbol{S}_1 \cup \boldsymbol{S}_2$, e che inoltre valgono le seguenti proprietà.

$$\boldsymbol{S} + \boldsymbol{P}_0 = \boldsymbol{S} \qquad\qquad \boldsymbol{S} + \boldsymbol{P}_1 = \boldsymbol{P}_1$$

Siano $\boldsymbol{S}_1$ e $\boldsymbol{S}_2$ due ricoprimenti di R con $\boldsymbol{S}_1 \prec \boldsymbol{S}_2$. Allora, è ovvio che due relazioni equivalenti rispetto ad $\boldsymbol{S}_2$ sono anche equivalenti rispetto ad $\boldsymbol{S}_1$. In altri termini, la classe $[r]_{\boldsymbol{S}_2}$ è un sottoinsieme della classe $[r]_{\boldsymbol{S}_1}$ e la relazione $\psi_{\boldsymbol{S}_2}(r)$ è contenuta nella relazione $\psi_{\boldsymbol{S}_1}(r)$ (vedi Figura 7.3).

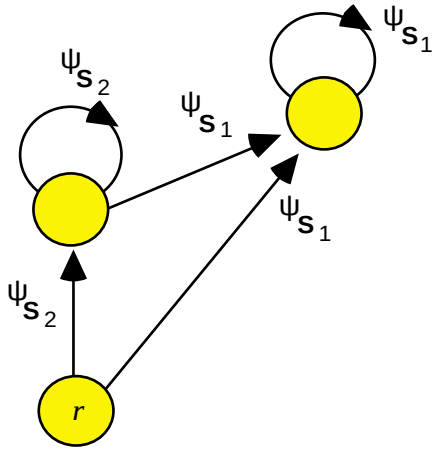


Figura 7.3

7.2 Scomposizioni

Sia $M = [R, F]$ un modello. Dato un ricoprimento $\boldsymbol{S} = \{R_1, \ldots, R_k\}$ di R, chiamiamo *scomposizione* di M *indotta* da $\boldsymbol{S}$ l'insieme dei modelli $M_1 = [R_1, F_1], \ldots, M_k = [R_k, F_k]$

dove F_h, $1 \leq h \leq k$, è l'insieme delle dipendenze funzionali valide in M che sono applicabili ad R_h, cioè

$$F_h = \{f \quad F(M): det(f) \quad dip(f) \quad R_h\} .$$

Chiamiamo i modelli $M_1, \ldots, M_k$ le *proiezioni* di M su $\boldsymbol{S}$.
Diremo che la scomposizione di M indotta da $\boldsymbol{S}$

— *conserva i dati* se ogni relazione conforme ad M coincide con il prodotto join delle sue proiezioni su $R_1, \ldots, R_k$ o, in termini equivalenti, se e solo se ogni relazione conforme ad M è un punto fisso di $\psi_{\boldsymbol{S}}$;

— *conserva le dipendenze* se, per ogni relazione r_1 conforme ad M_1, …, per ogni relazione r_k conforme ad M_k, il prodotto join delle relazioni $r_1, \ldots, r_k$ è una relazione conforme ad M;

— è *conservativa* se conserva sia i dati che le dipendenze.

Dunque, se una scomposizione non conserva i dati allora, date k relazioni $r_1, \ldots, r_k$ conformi rispettivamente ai modelli $M_1, \ldots, M_k$, in genere non abbiamo modo di ricostruire una relazione conforme ad M. Inoltre, se una scomposizione conserva i dati allora, date k relazioni $r_1, \ldots, r_k$ conformi rispettivamente ai modelli $M_1, \ldots, M_k$, in genere il prodotto join di $r_1, \ldots, r_k$ è una relazione conforme ad M solo se la scomposizione conserva le dipendenze; in assenza di tale requisito, prima di affermare che il prodotto join di $r_1, \ldots, r_k$ è una relazione conforme ad M occorre controllare che esso soddisfi tutte le dipendenze funzionali in $F - G$.

***Teorema* 7.1**. Dato un modello $M = [R, F]$, siano $\boldsymbol{S}_1$ e $\boldsymbol{S}_2$ due ricoprimenti di R con $\boldsymbol{S}_1 \prec \boldsymbol{S}_2$. Se la scomposizione di M indotta da $\boldsymbol{S}_1$ conserva i dati, allora anche la scomposizione di M indotta da $\boldsymbol{S}_2$ conserva i dati. Se la scomposizione di M indotta da $\boldsymbol{S}_2$ non conserva i dati, allora anche la scomposizione di M indotta da $\boldsymbol{S}_1$ non conserva i dati.
Dimostrazione. Per una qualsiasi relazione r su R, si ha

$$r \quad \psi_{\boldsymbol{S}_2}(r) \quad \psi_{\boldsymbol{S}_1}(r).$$

Pertanto se, r è un punto fisso di $\psi_{\boldsymbol{S}_1}$ allora, a fortiori, lo è di $\psi_{\boldsymbol{S}_2}$; se invece r non è un punto fisso di $\psi_{\boldsymbol{S}_2}$ allora non può esserlo di $\psi_{\boldsymbol{S}_1}$. □

In merito alla conservazione delle dipendenze, si osservi che quello che si richiede è che, per ogni relazione r_1 conforme ad M_1, …, per ogni relazione r_k conforme ad M_k, il

prodotto join delle relazioni $r_1, \ldots, r_k$ soddisfi tutte le dipendenze funzionali in F. Ovviamente, la relazione

$$r_1 \quad \ldots \quad r_k$$

soddisfa tutte le dipendenze funzionali nel sottoinsieme (eventualmente improprio, per esempio se $\boldsymbol{S} = \boldsymbol{P}_1$)

$$G = \quad {}_{h=1, \ldots, k}\ F_h$$

di $F(M)$. Chiamiamo il modello $N = [R, G]$ il *sottomodello* di M *generato* da $\boldsymbol{S}$. Dunque, perché si abbia la conservazione delle dipendenze occorre e basta che il modello N sia equivalente ad M o, che è lo stesso, che ogni dipendenza funzionale in F sia valida in N.

***Teorema* 7.2**. Dato un modello $M = [R, F]$, siano $\boldsymbol{S}_1$ e $\boldsymbol{S}_2$ due ricoprimenti di R con $\boldsymbol{S}_1 \prec \boldsymbol{S}_2$. Se la scomposizione di M indotta da $\boldsymbol{S}_1$ conserva le dipendenze, allora anche la scomposizione di M indotta da $\boldsymbol{S}_2$ conserva le dipendenze. Se la scomposizione di M indotta da $\boldsymbol{S}_2$ non conserva le dipendenze, allora anche la scomposizione di M indotta da $\boldsymbol{S}_1$ non conserva le dipendenze.

Dimostrazione. Siano $N_1 = [R, G_1]$ ed $N_2 = [R, G_2]$ i sottomodelli di M generati rispettivamente da $\boldsymbol{S}_1$ e $\boldsymbol{S}_2$. Si noti innanzitutto che $\boldsymbol{S}_1 \prec \boldsymbol{S}_2$ implica che $G_1 \quad G_2$ e, quindi, che $F(N_1) \quad F(N_2) \quad F(M)$. Ora, se la scomposizione di M indotta da $\boldsymbol{S}_1$ conserva le dipendenze, si ha che $F(N_1) = F(M)$ e, quindi, anche che $F(N_2) = F(M)$. Se, invece, la scomposizione di M indotta da $\boldsymbol{S}_2$ non conserva le dipendenze, allora si ha che $F(N_2) \quad F(M)$, e quindi, anche che $F(N_1) \quad F(M)$. □

Esempio 7.2 Sia $R = \{$**CITTÀ**, **VIA**, **CAP**$\}$ e sia F dato dalle due dipendenze funzionali:

$$\{\textbf{CITTÀ}, \textbf{VIA}\} \rightarrow \textbf{CAP} \qquad\qquad \textbf{CAP} \rightarrow \textbf{CITTÀ}$$

Avevamo già visto che il modello $M = [R, F]$ non è normale. Ora, la scomposizione generata dal ricoprimento $\{\{$**CITTÀ**, **VIA**$\}, \{$**CITTÀ**, **CAP**$\}, \{$**VIA**, **CAP**$\}\}$ di R non conserva le dipendenze per via della dipendenza funzionale $\{$**CITTÀ**, **VIA**$\} \rightarrow$ **CAP**. Per il Teorema 7.2, ogni scomposizione generata da un ricoprimento nonbanale di $\{$**CITTÀ**, **VIA**, **CAP**$\}$ non conserva le dipendenze. ◊

7.3. Conservazione dei dati

Sia $M = [R, F]$ un modello ed $\boldsymbol{S}$ un ricoprimento di R. Daremo un algoritmo per decidere se la scomposizione di M indotta da $\boldsymbol{S}$ conserva i dati. Senza perdere in generalità, assumiamo che M sia un modello in forma canonica. Rappresentiamo il ricoprimento $\boldsymbol{S}$ con una matrice di simboli che chiamiamo il *tableau* associato ad $\boldsymbol{S}$.

Siano $A_1, \ldots, A_n$ ed $R_1, \ldots, R_k$ due ordinamenti arbitrari rispettivamente degli attributi in R e degli insiemi in $\boldsymbol{S}$, e sia poi $Q = \{1, \ldots, q\}$, dove

$$q = n\,(k+1) - \sum_{h=1,\ldots k} |R_h|.$$

Il tableau associato ad $\boldsymbol{S}$ è la matrice $T = ((T_{h,i}))$, $1 \leq h \leq k$ ed $1 \leq i \leq n$, che si ottiene con la seguente procedura:

(1) $p := n$
(2) Per $h = 1, \ldots, k$
(3) Per $i = 1, \ldots, n$
se $A_i \quad R_h$, allora $T_{h,i} := i$, altrimenti $T_{h,i} := p+1$; $p := p+1$.

Se X è un sottoinsieme di R, con $T_h[X]$ indicheremo la stringa di numeri che nella riga T_h di T corrispondono agli attributi presenti in X. Dato un modello in forma canonica $M = [R, F]$ ed il tableau T associato ad un ricoprimento $\boldsymbol{S}$ di R, il seguente algoritmo consente di decidere se la scomposizione di M indotta da $\boldsymbol{S}$ conserva o meno i dati.

Test di Conservazione dei Dati

Dati: Un insieme R di attributi, un insieme F di dipendenze funzionali semplici applicabili ad R ed un ricoprimento $\boldsymbol{S}$ di R.

Risultato: Un valore della variabile logica *test.*

Procedura

(1) *test* := FALSO.

(2) Finché il tableau T non possa essere ulteriormente modificato, applicare ripetutamente le seguente procedura:

Per ogni dipendenza funzionale $X \rightarrow A_i$ in F,
se T contiene due righe $T_{h'}$ e $T_{h''}$ tali che

$$T_{h'}[X] = T_{h''}[X] \text{ e } T_{h',i} \neq T_{h'',i},$$

allora:

p' := min $(T_{h',i}, T_{h'',i})$; p'' = max $(T_{h',i}, T_{h'',i})$;
per $h = 1, \ldots, k$ se $T_{h,i} = p''$ allora $T_{h,i} := p'$.

(3) Se T contiene la riga $(1, \ldots, n)$, allora *test* := VERO.

Diremo che il Test di Conservazione dei Dati *dà esito positivo* (rispettivamente, *negativo*) se il valore finale della variabile *test* è VERO (rispettivamente, FALSO).

Esempio 7.3 Siano $R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ ed $F = \{\mathbf{A} \rightarrow \mathbf{B}, \mathbf{AC} \rightarrow \mathbf{D}\}$. Il tableau associato al ricoprimento $\boldsymbol{S} = \{\{\mathbf{A}, \mathbf{B}\}, \{\mathbf{A}, \mathbf{C}, \mathbf{D}\}\}$ di R è riportato in Tabella 7.3.

A	**B**	**C**	**D**
1	2	5	6
1	7	3	4

Tabella 7.3

La dipendenza funzionale $\mathbf{A} \rightarrow \mathbf{B}$ porta a sostituire (nella seconda riga) 7 con 2, ed il nuovo tableau è riportato in Tabella 7.4.

A	B	C	D
1	2	5	6
1	2	3	4

Tabella 7.4

La dipendenza funzionale **AC** → **D** non modifica il contenuto di questo tableau e, siccome questo contiene la riga (1, 2, 3, 4), possiamo concludere che la scomposizione di *M* indotta da ***S*** conserva i dati. ◊

Esempio 7.4 Siano $R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ ed $F = \{\mathbf{A} \rightarrow \mathbf{B}, \mathbf{C} \rightarrow \mathbf{D}\}$. Il tableau associato al ricoprimento $\boldsymbol{S} = \{\{\mathbf{A}, \mathbf{B}\}, \{\mathbf{C}, \mathbf{D}\}\}$ di *R* è riportato in Tabella 7.5.

A	B	C	D
1	2	5	6
7	8	3	4

Tabella 7.5

Esso non viene modificato né da **A** → **B** né da **C** → **D**. Pertanto la scomposizione di *M* indotta da ***S*** non conserva i dati. ◊

Teorema* 7.3**. Sia $M = [R, F]$ un modello in forma canonica e sia ***S un ricoprimento di *R*. Se la scomposizione di *M* indotta da ***S*** conserva i dati, allora il Test di Conservazione dei Dati dà esito positivo.

Dimostrazione. Siano $R = \{A_1, \ldots, A_n\}$ ed $\boldsymbol{S} = \{R_1, \ldots, R_k\}$. Siano poi *T* il tableau iniziale e *T'* il tableau finale prodotto dal Test di Conservazione dei Dati. Ai numeri 1, …, *q* associamo altrettanti elementi $a_1, \ldots, a_q$ dell'insieme

$$d(A_1) \quad \ldots \quad d(A_n)$$

in maniera tale che, se il numero *p*, $1 \leq p \leq q$, compare nella colonna *i*.esima di *T*, allora a_p è un elemento del dominio di A_i. In questo modo, a *T* ed a *T'* restano associate due relazioni con schema *R*; indichiamole con *r* ed *r'*. Si osservi che, per ogni *h*, $1 \leq h \leq k$, la sottomatrice di *T* ottenuta considerando le sole colonne corrispondenti agli attributi in R_h contiene una riga i cui elementi sono tutti minori o uguali ad *n*; pertanto, la proiezione di *r* su R_h contiene un'ennupla i cui valori appartengono all'insieme $\{a_1, \ldots, a_n\}$. Ovviamente, questa proprietà vale anche per la relazione *r'* cosicché la relazione $\psi_{\boldsymbol{S}}(r')$ conterrà l'ennupla $(a_1, \ldots, a_n)$. D'altro canto, *r'* risulta essere una relazione conforme ad *M* perché, ogni volta che si è incontrata una violazione di una dipendenza funzionale in *F*, il tableau è stato modificato per rimuoverla. Ora, siccome la scomposizione di *M* indotta da ***S*** conserva i dati, *r'* è un punto fisso di $\psi_{\boldsymbol{S}}$ e quindi

$\psi_{\boldsymbol{S}}(r') = r'$. Dunque, r' contiene l'ennupla $(a_1, \ldots, a_n)$. Ne segue che T' contiene la riga $(1, \ldots, n)$. □

In verità, si può dimostrare un risultato più forte di quello enunciato nel Teorema 7.3 che ci limitiamo ad enunciare.

***Teorema* 7.4**. Siano R un insieme di attributi, F un insieme di dipendenze funzionali applicabili ad R in forma canonica, ed $\boldsymbol{S}$ un ricoprimento di R. Per ogni modello $M = [R, F]$, la scomposizione di M indotta da $\boldsymbol{S}$ conserva i dati se e solo se il Test di Conservazione dei Dati dà esito positivo.

Il Teorema 7.4 ha come conseguenza il seguente risultato la cui dimostrazione si lascia come esercizio.

***Teorema* 7.5.** Siano R un insieme di attributi, F un insieme di dipendenze funzionali applicabili ad R ed $\boldsymbol{S} = \{R_1, R_2\}$ un ricoprimento di R. Per ogni modello $M = [R, F]$, la scomposizione di M indotta da $\boldsymbol{S}$ conserva i dati se e solo se almeno una delle due dipendenze funzionali $R_1 \cap R_2 \rightarrow R_1 - R_2$ ed $R_1 \cap R_2 \rightarrow R_2 - R_1$ è valida in M.

7.4 Conservazione delle dipendenze

Sia $M = [R, F]$ un modello in forma canonica e sia $\boldsymbol{S}$ un ricoprimento di R. Daremo un algoritmo per decidere se la scomposizione di M indotta da $\boldsymbol{S}$ conserva le dipendenze, cioè se ogni dipendenza funzionale in F è valida nel sottomodello $N = [R, G]$ di M generato da $\boldsymbol{S}$. L'algoritmo è basato sul fatto che una dipendenza funzionale $X \rightarrow A$ in F è valida in N se e solo se l'attributo A appartiene alla chiusura X^* di X rispetto a G. Così, è sufficiente calcolare X^* per ogni dipendenza funzionale $X \rightarrow A$ appartenente ad F. Ora, non conviene applicare l'Algoritmo di Chiusura, perché questo richiede il calcolo preliminare di G che è un insieme esageratamente grande.

Test di Conservazione delle Dipendenze

Dati: Un insieme R di attributi, un insieme F di dipendenze funzionali semplici applicabili ad R ed un ricoprimento $\boldsymbol{S} = \{R_1, \ldots, R_k\}$ di R.

Risultato: Un valore della variabile logica *test.*

Procedura

(1) *test* := VERO

(2) Per ogni dipendenza funzionale $X \rightarrow A$ in F

$V := X$.
Finché V non possa essere ulteriormente ampliato ripetere:

Per $h = 1, \ldots, k$,

aggiungere a V tutti gli attributi in R_h che appartengono a $(V \cap R_h)^+$, cioè

$$V := V \quad (R_h \cap (V \cap R_h)^+).$$

Se $A \quad V$ allora *test* := FALSO ed Uscire.

Diremo che il Test di Conservazione delle Dipendenze *dà esito positivo* (rispettivamente, *negativo*) se il valore finale della variabile *test* è VERO (rispettivamente, FALSO).

***Teorema* 7.6**. Siano R un insieme di attributi, F un insieme di dipendenze funzionali applicabili ad R ed in forma canonica, ed $\boldsymbol{S}$ un ricoprimento di R. Per ogni modello $M = [R, F]$, la scomposizione di M indotta da $\boldsymbol{S}$ conserva le dipendenze se e solo se il Test di Conservazione delle Dipendenze dà esito positivo.

Dimostrazione. È sufficiente provare che, per ogni dipendenza funzionale $X \rightarrow A$ in F, il valore V trovato al passo (2) dell'algoritmo coincide con la chiusura X^* di X rispetto a G. Lo proveremo dimostrando che V è un sottoinsieme di X^*, e poi che X^* è un sottoinsieme di V.

1: $V \quad X^*$. Siano $V_1, \ldots, V_p$ i valori man mano assunti da V durante l'esecuzione dell'istruzione (2) dell'algoritmo. Così, $V_1 = X$. Dimostreremo che per ogni i, $1 \leq i \leq p$, $V_i \quad X^*$. La dimostrazione è per induzione su i.

PASSO BASE. Per $i = 1$, l'inclusione V_i X^* è ovvia.

PASSO INDUTTIVO. Supponiamo per ipotesi induttiva che l'inclusione V_i X^* valga fino ad un certo i, $i < p$. Dimostriamo allora che vale anche l'inclusione V_{i+1} X^*. Sia $R_{h(i)}$, $1 \le h(i) \le k$, l'insieme in $\boldsymbol{S}$ che ha portato i valore di V da V_i a V_{i+1}, e sia Y_i, $1 \le i \le p-1$, l'insieme di attributi che vengono aggiunti a V_i per generare V_{i+1}, cioè $V_{i+1} = V_i$ Y_i. Così, l'insieme Y_i è contenuto tanto in $R_{h(i)}$ quanto in $(V_i \cap R_{h(i)})^+$. Ora, siccome Y_i è un sottoinsieme di $(V_i \cap R_{h(i)})^+$, la dipendenza funzionale $V_i \cap R_{h(i)} \rightarrow Y_i$ è valida in M e, siccome Y_i è anche un sottoinsieme di $R_{h(i)}$, questa è una dipendenza funzionale applicabile ad $R_{h(i)}$. Pertanto, la dipendenza funzionale $V_i \cap R_{h(i)} \rightarrow Y_i$ è presente in $F_{h(i)}$ e, quindi, in G. Dunque, Y_i è un sottoinsieme di $(V_i \cap R_{h(i)})^*$ e, quindi, di V_i^* cosicché $V_{i+1} = V_i$ Y_i V_i^*. D'altra parte, per l'ipotesi induttiva V_i X^*; pertanto, si ha che V_{i+1} V_i^* X^*.
Dunque per ogni i, $1 \le i \le p$, vale l'inclusione V_i X^*. In particolare, per $i = p$, abbiamo che il valore finale di V calcolato dall'algoritmo è un sottoinsieme di X^*.

2: X^* V. Sia $(f_1, \ldots, f_2, \ldots, f_n)$ una derivazione regolare della dipendenza funzionale $X \rightarrow X^*$ da G, dove f_i, $2 \le i \le n$, è l'i.esima dipendenza funzionale ottenuta applicando la regola A3, e sia $X_i = dip(f_i)$ per ogni i, $1 \le i \le n$. Così, f_i è $X \rightarrow X_i$ ed inoltre $X_1 = X$ e $X_n = X^*$. Dimostriamo ora per induzione che ogni X_i, $1 \le i \le n$, è un sottoinsieme di V. Ovviamente, X_1 V perché X V. Supponiamo per ipotesi induttiva che l'inclusione X_i V valga fino ad un certo i, $i < n$; dimostriamo che anche X_{i+1} è un sottoinsieme di V. La dipendenza funzionale f_{i+1} è stata ottenuta applicando la regola A3 con la sostituzione degli antecedenti rispettivamente con f_i e con una dipendenza funzionale $Y \rightarrow Z$ in G dove Y X_i così che $X_{i+1} = X_i$ Z. Ora, siccome Y è un sottoinsieme di X_i, per l'ipotesi induttiva abbiamo che

$$Y \quad V.$$

D'altra parte, siccome la dipendenza funzionale $Y \rightarrow Z$ è in G, essa è valida in M, cioè

$$Z \quad Y^+ ,$$

ed è applicabile ad un qualche insieme R_h in $\boldsymbol{S}$, cioè

$$Y \quad R_h \qquad \text{e} \qquad Z \quad R_h.$$

Dunque, per Y abbiamo che Y $V \cap R_h$ da cui

$$Y^+ \quad (V \cap R_h)^+$$

e per Z abbiamo invece

$Z \quad R_h \cap Y^+ \quad R_h \cap (V \cap R_h)^+$.

Pertanto, ad un certo punto nell'esecuzione dell'istruzione (2) dell'algoritmo tutti gli attributi in Z entreranno a far parte di V. In conclusione, anche X_{i+1} è un sottoinsieme di V. Dunque per ogni i, $1 \leq i \leq n$, vale l'inclusione $X_i \quad V$. In particolare, per $i = n$, abbiamo che X^* è un sottoinsieme di V. □

Quanto alla complessità del Test di Conservazione delle Dipendenze, si osservi che il numero delle fasi è minore o uguale a $|R|-1$, ed ogni fase richiede l'esecuzione di k operazioni in ciascuna delle quali è dominante il calcolo di $(V \cap R_h)^+$ che richiede un tempo lineare nella dimensione di M. Se questa è q, allora la complessità dell'algoritmo è $O(|R| \times q \times |\boldsymbol{S}|)$.

Esempio 7.5 Consideriamo il modello $M = [R, F]$ in forma canonica con

$$R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\} \qquad \text{ed} \qquad F = \{\mathbf{A} \rightarrow \mathbf{B}, \mathbf{B} \rightarrow \mathbf{C}, \mathbf{C} \rightarrow \mathbf{D}, \mathbf{D} \rightarrow \mathbf{A}\}.$$

La chiusura rispetto ad F di ognuno dei quattro attributi è sempre R e tutte le 2^8 possibili dipendenze funzionali applicabili ad R sono valide in M. Consideriamo la scomposizione di M indotta dal ricoprimento $\boldsymbol{S} = \{\{\mathbf{A}, \mathbf{B}\}, \{\mathbf{B}, \mathbf{C}\}, \{\mathbf{C}, \mathbf{D}\}\}$ di R. Le proiezioni di M su $\boldsymbol{S}$ sono

$$M_1 = [\{\mathbf{A}, \mathbf{B}\}, d_1, F_1], M_2 = [\{\mathbf{B}, \mathbf{C}\}, d_2, F_2], M_3 = [\{\mathbf{C}, \mathbf{D}\}, d_3, F_3]$$

ed ognuno degli F_h, $1 \leq h \leq 3$, contiene tutte e nove le dipendenze funzionali applicabili ad R_h, e

$$G = F_1 \quad F_2 \quad F_3$$

contiene 3×2^4 dipendenze funzionali, tra cui le dipendenze funzionali $\mathbf{A} \rightarrow \mathbf{B}$, $\mathbf{B} \rightarrow \mathbf{C}$ e $\mathbf{C} \rightarrow \mathbf{D}$. Dunque, ad eccezione di $\mathbf{D} \rightarrow \mathbf{A}$, tutte le altre dipendenze funzionali in F sono già in G e, pertanto, il modello $N = [R, G]$ è equivalente ad M se e solo se la dipendenza funzionale $\mathbf{D} \rightarrow \mathbf{A}$ è valida in N. Ebbene, è facile verificare che è proprio questo il caso perché F_1, F_2 ed F_3 contengono rispettivamente le dipendenze funzionali $\mathbf{B} \rightarrow \mathbf{A}$, $\mathbf{C} \rightarrow \mathbf{B}$ and $\mathbf{D} \rightarrow \mathbf{C}$, e quindi, per la proprietà transitiva, la dipendenza funzionale $\mathbf{D} \rightarrow \mathbf{A}$ è valida in N. Alla stessa conclusione arriviamo verificando che l'attributo $\mathbf{A}$ appartiene a $\{\mathbf{D}\}^*$. Inizialmente, V è formato dal singolo attributo $\mathbf{D}$. Una prima scansione di $\boldsymbol{S}$ dà il seguente risultato: gli insiemi $\{\mathbf{A}, \mathbf{B}\}$ e $\{\mathbf{B}, \mathbf{C}\}$ lasciano V invariato; invece, per $\{\mathbf{C}, \mathbf{D}\}$ si ha

$$V := V \quad (\{\mathbf{C}, \mathbf{D}\} \cap (V \cap \{\mathbf{C}, \mathbf{D}\})^+)$$

cioè

$$\begin{aligned} V := \quad & \{\mathbf{D}\} \quad (\{\mathbf{C}, \mathbf{D}\} \cap (\{\mathbf{D}\} \cap \{\mathbf{C}, \mathbf{D}\}^+) = \\ & = \{\mathbf{D}\} \quad (\{\mathbf{C}, \mathbf{D}\} \cap \{\mathbf{D}\}^+) = \\ & = \{\mathbf{D}\} \quad (\{\mathbf{C}, \mathbf{D}\} \cap \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}) = \\ & = \{\mathbf{D}\} \quad \{\mathbf{C}, \mathbf{D}\} = \{\mathbf{C}, \mathbf{D}\} . \end{aligned}$$

Ad una seconda scansione di $\boldsymbol{S}$, per effetto di {**B**, **C**}, V diventa {**B**, **C**, **D**}. Finalmente, al terzo esame di $\boldsymbol{S}$, V viene a contenere anche **A**. Dunque, possiamo concludere che la dipendenza funzionale **D** → **A** è valida in N e, quindi, che la scomposizione di M indotta da $\boldsymbol{S}$ conserva le dipendenze. ◊

7.5 Gradi di Normalità

Dato un modello $M = [R, F]$, ci domandiamo se è possibile trovare un ricoprimento $\boldsymbol{S}$ di R tale che la scomposizione di M indotta da $\boldsymbol{S}$ sia conservativa e che le proiezioni di M su $\boldsymbol{S}$ siano tutti modelli normali. La risposta a questo quesito è negativa nel caso generale dal momento che esistono casi in cui non esiste proprio un tale ricoprimento. Inoltre, per quei casi in cui pure esiste, costruirlo è in generale un problema NP-arduo.

Esempio 7.2 (seguito). Riprendiamo il modello $M = [R, F]$ con R = {**COMUNE**, **VIA**, **CAP**} ed F = {{**COMUNE**, **VIA**} → **CAP**, **CAP** → **COMUNE**}. Consideriamo la scomposizione di M indotta da $\boldsymbol{S}$ = {{**COMUNE**, **VIA**}, {**COMUNE**, **CAP**}, {**VIA**, **CAP**}}. I modelli prodotti dalla scomposizione sono normali, ma la scomposizione non conserva le dipendenze perché la dipendenza funzionale {**COMUNE**, **VIA**} → **CAP** non è valida nel sottomodello di M generato da $\boldsymbol{S}$. In realtà, per il Teorema 7.2 non esiste nessuna scomposizione di M che conservi le dipendenze fatta eccezione per quella indotta dalla partizione banale di R. ◊

Vediamo allora come indebolire il requisito di normalità quanto basti per essere certi che esista sempre una scomposizione conservativa. Introduciamo prima alcune nozioni.

Sia $M = [R, F]$ un modello e sia $X \rightarrow A$ una dipendenza funzionale semplice, non banale e valida in M. Diremo che in M l'attributo A

> *dipende parzialmente* (rispettivamente, *totalmente*) da X se esiste (rispettivamente, non esiste) un sottoinsieme proprio Y di X tale che la dipendenza funzionale $Y \rightarrow A$ sia ancora valida in M
>
> *dipende transitivamente* da X se esiste un sottoinsieme Y di $R–\{A\}$ tale che:
>
> > $X \rightarrow Y$ ed $Y \rightarrow A$ sono dipendenze funzionali valide in M, e
> > $Y \rightarrow X$ non è una dipendenza funzionale valida in M.

***Lemma* 7.2**. Siano M un modello ed X una chiave di M. Se un attributo dipende parzialmente da X allora esso dipende transitivamente da X.

Dimostrazione. Infatti se un attributo A dipende parzialmente da X, allora esiste un sottoinsieme proprio Y di X tale che la dipendenza funzionale $Y \rightarrow A$ non è banale ed è valida in M. Inoltre, per il Lemma 1.4 la dipendenza funzionale $X \rightarrow Y$ è banalmente valida in M. D'altra parte, siccome X è una chiave di M, Y non può essere una

sovrachiave cosicché, per il Lemma 1.5, la dipendenza funzionale $Y \rightarrow X$ non è valida in *M*. Dunque, *A* dipende transitivamente da *X*. □

Un attributo in un modello *M* si dice *primario* per *M* se appartiene ad almeno una chiave di *M*, e *secondario* altrimenti.

Un modello è in *prima forma normale* se nessun attributo è composto o multiplo.

Un modello è in *seconda forma normale* se è in prima forma normale ed ogni attributo secondario dipende totalmente da ogni chiave.

Un modello è in *terza forma normale* se è in prima forma normale e se non esistono né una chiave *X* né un attributo secondario *A* tali che *A* dipenda transitivamente da *X*.

Un modello è in *forma normale* di Boyce-Codd se è in prima forma normale e se non esistono né una chiave *X* né un attributo *A* tali che *A* dipenda transitivamente da *X*.

Si osservi che, se un modello è in terza forma normale allora, per il Lemma 7.2, esso è anche in seconda forma normale. Ovviamente, se un modello è in forma normale di Boyce-Codd, allora è anche in terza forma normale.

Esempio 7.6 Riprendiamo il modello della variabile relazionale **partenze** della Base di Dati Aeroportuali. Esso non è in seconda forma normale perché la dipendenza funzionale **VOLO** → **ORA** fa sì che l'attributo secondario **ORA** dipenda parzialmente dalla chiave {**VOLO**, **DATA**}. ◊

Esempio 7.7 Riprendiamo il modello della variabile relazionale **esame** dell'Esempio 3.3. Esso non è in seconda forma normale perché l'attributo secondario **DOCENTE** dipende parzialmente dalla chiave {**STUDENTE**, **MATERIA**}. ◊

Il seguente esempio mostra un modello che è in terza forma normale ma non in forma normale di Boyce-Codd.

Esempio 7.2 (seguito). Consideriamo il modello con schema {**CITTÀ**, **VIA**, **CAP**} in cui siano definite le due dipendenze funzionali: {**CITTÀ**, **VIA**} determina **CAP**, e **CAP** determina **CITTÀ**. Allora, {**CITTÀ**, **VIA**} e {**VIA**, **CAP**} sono le uniche chiavi del modello e, quindi, i tre attributi sono tutti primari. Dunque, il modello è in terza forma normale. Il modello non è però in forma normale di Boyce-Codd perché l'attributo (primario) **CITTÀ** dipende parzialmente e quindi transitivamente dalla chiave {**VIA**, **CAP**}. ◊

***Lemma* 7.3**. Un attributo *A* di un modello *M* non dipende transitivamente da alcuna chiave di *M* se e solo se, per ogni dipendenza funzionale non banale $Y \rightarrow A$ che sia valida in *M*, l'insieme *Y* è una sovrachiave di *M*.

Dimostrazione. Dalla stessa definizione di dipendenza transitiva applicata ad una chiave X, abbiamo che un attributo A dipende transitivamente da X se esiste un sottoinsieme Y di R tale che

$Y \rightarrow A$ è una dipendenza funzionale non banale valida in M, ed
$Y \rightarrow X$ non è una dipendenza funzionale valida in M.

dove la seconda condizione è equivalente a richiedere che Y non sia una sovrachiave. □

Come conseguenza del Lemma 7.3, abbiamo le seguenti caratterizzazioni dei modelli in terza forma normale ed in forma normale di Boyce-Codd.

***Teorema* 7.7**. Un modello M che sia in prima forma normale è in terza forma normale se e solo se, per ogni dipendenza funzionale semplice non banale $Y \rightarrow A$ che sia valida in M, A è un attributo primario oppure Y è una sovrachiave di M.

***Teorema* 7.8**. Un modello M che sia in prima forma normale è in forma normale di Boyce-Codd se e solo se, per ogni dipendenza funzionale semplice non banale f che sia valida in M, l'insieme $det(f)$ è una sovrachiave di M.

Il Teorema 7.8 ha il seguente corollario.

***Teorema* 7.9**. Un modello che sia in prima forma normale è normale se e solo se è in forma normale di Boyce-Codd.

Così dato un modello M in prima forma normale, per decidere se M è o non è in forma normale di Boyce Codd basta applicare il test di normalità.

7.6 Scomposizione in Terza Forma Normale

Dato un modello $M = [R, F]$, ci domandiamo se è possibile trovare un ricoprimento $\boldsymbol{S}$ di R tale che la scomposizione di M indotta da $\boldsymbol{S}$ sia conservativa ed in terza forma normale. La risposta a questo quesito è affermativa. Per dimostrarlo, daremo prima un algoritmo che permette di costruire un ricoprimento $\boldsymbol{S}$ di R che induce una scomposizione in terza forma normale di M la quale conserva le dipendenze.

Scomposizione in Terza Forma Normale

Input: Un insieme di attributi R ed un insieme di dipendenze funzionali F applicabili ad R che sia in forma canonica.
Output: Un ricoprimento $\boldsymbol{S}$ di R.

Procedura

(1) $\boldsymbol{S} := \emptyset$.

(2) Se Z è l'insieme di attributi in R che non sono presenti in nessuna delle dipendenze funzionali contenute in F, allora

$$\boldsymbol{S} := \boldsymbol{S} + \{Z\}.$$

(3) Per ogni dipendenza funzionale $X \rightarrow A$ in F

$$\boldsymbol{S} := \boldsymbol{S} + \{X \ \ \{A\}\}.$$

Esempio 7.8 Siano

$$R = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}, \mathbf{G}\}$$

$$F = \{\mathbf{A} \rightarrow \mathbf{B}, \mathbf{B} \rightarrow \mathbf{A}, \mathbf{CD} \rightarrow \mathbf{E}, \mathbf{CE} \rightarrow \mathbf{D}, \mathbf{DE} \rightarrow \mathbf{C}\}.$$

L'applicazione dell'algoritmo di Scomposizione in Terza Forma Normale genera il ricoprimento $\boldsymbol{S} = \{\mathbf{AB}, \mathbf{CDE}, \mathbf{FG}\}$ di R. ◊

***Teorema* 7.10**. Sia R un insieme di attributi e sia F un insieme di dipendenze funzionali applicabili ad R che sia in forma canonica. Per ogni modello $M = [R, F]$, l'algoritmo di Scomposizione in Terza Forma Normale costruisce un ricoprimento $\boldsymbol{S}$ di R tale che la scomposizione di M indotta da $\boldsymbol{S}$ conserva le dipendenze e le proiezioni di M su $\boldsymbol{S}$ sono tutte in terza forma normale.
Dimostrazione. Sia $\boldsymbol{S} = \{R_1, \ldots, R_k\}$, e sia M_h la proiezione di M su R_h, $1 \leq h \leq k$. Siccome ogni dipendenza funzionale è applicabile ad uno degli insiemi R_h, la scomposizione di M indotta da $\boldsymbol{S}$ conserva le dipendenze in F. Resta da provare che ogni modello M_h è in terza forma normale. Se $R_h = Z$ (vedi Istruzione 2), allora R_h è l'unica chiave di M_h e, quindi, M_h è banalmente in terza forma normale. Altrimenti sia $X \rightarrow A$ la dipendenza funzionale in F che ha portato R_h in $\boldsymbol{S}$, cioè $R_h = X \ \ \{A\}$ (vedi Istruzione 3). Si noti innanzitutto che X è una chiave di M_h perché la dipendenza funzionale $X \rightarrow R_h$ è valida in M_h e perché nessuna dipendenza funzionale in F è

riducibile. Dunque, ogni attributo in X è primario; inoltre, solo l'attributo A potrebbe essere secondario. In tal caso, se M_h non fosse in terza forma normale allora, per il Lemma 7.3, A dovrebbe dipendere parzialmente da X, la qual cosa è in contrasto con il fatto che ogni dipendenza funzionale in F è irriducibile. Dunque, la scomposizione di M indotta da $\boldsymbol{S}$ conserva le dipendenze e ciascuna delle proiezioni di M su $\boldsymbol{S}$ è un modello in terza forma normale. □

Come dimostrato nel prossimo teorema, l'algoritmo di Scomposizione in Terza Forma Normale con l'aggiunta della seguente istruzione riesce a costruire un ricoprimento $\boldsymbol{S}^*$ che induce una scomposizione in terza forma normale di M che conserva anche i dati.

(4) Trovare una chiave K di M, e porre $\boldsymbol{S}^* := \boldsymbol{S} + \{K\}$.

***Teorema* 7.11**. Sia R un insieme di attributi e sia F un insieme di dipendenze funzionali applicabili ad R che sia in forma canonica. Per ogni modello $M = [R, F]$, l'algoritmo di Scomposizione in Terza Forma Normale con l'aggiunta dell'Istruzione 4 costruisce un ricoprimento $\boldsymbol{S}^*$ di R tale che la scomposizione di M indotta da $\boldsymbol{S}^*$ è conservativa e le proiezioni di M su $\boldsymbol{S}^*$ sono tutte in terza forma normale.

Dimostrazione. Sia $\boldsymbol{S} = \{R_1, \ldots, R_k\}$ il ricoprimento costruito dall'algoritmo di Scomposizione in Terza Forma Normale. Siccome la scomposizione di M indotta da $\boldsymbol{S}$ conserva le dipendenze allora, per il Teorema 7.2, anche $\boldsymbol{S}^* = \{R_1, \ldots, R_k, K\}$ induce una scomposizione che conserva le dipendenze. Inoltre, siccome la proiezione di M su K è banalmente un modello in terza forma normale, per il Teorema 7.10 le proiezioni di M su $\boldsymbol{S}^*$ sono tutte in terza forma normale. Così, resta solo da provare che la scomposizione di M indotta da $\boldsymbol{S}^*$ conserva i dati, cioè che il Test di Conservazione dei Dati dà esito positivo. Consideriamo allora il tableau associato ad $\boldsymbol{S}^*$; questo si ottiene dal tableau associato ad $\boldsymbol{S}$, sia esso T, aggiungendo la riga T_{k+1} corrispondente all'insieme K. Dunque, $T_{k+1,j} = j$ oppure $T_{k+1,j} > n$, dove $n = |R|$, a seconda che j corrisponda o meno ad un attributo in K. Applichiamo ora il Test di Conservazione dei Dati e contestualmente l'Algoritmo di Chiusura per calcolare K^+, ed assumiamo che l'ordine in cui vengono esaminate le dipendenze funzionali dai due algoritmi sia identico a quello usato dall'algoritmo di Scomposizione in Terza Forma Normale. Sia V il valore corrente di K^+ ottenuto all'inizio di una certa iterazione dall'Algoritmo di Chiusura e sia A un attributo che viene aggiunto a V per effetto della dipendenza funzionale $X \rightarrow A$ in F; così, $X \quad V$ ed $A \quad V$. Ora, se l'attributo A corrisponde alla colonna i.esima di T e se l'insieme di attributi $X \quad \{A\}$ è l'insieme R_h, allora quando l'Algoritmo di Chiusura aggiunge A a V, il Test di Conservazione dei dati si trova ad esaminare un tableau in cui:

la riga T_h contiene i valori $T_{h,j} = j$ per ogni j che corrisponda ad un attributo in $X \quad \{A\}$

la riga T_{k+1} contiene i valori $T_{k+1,j} = j$ per ogni j che corrisponda ad un attributo in X e $T_{k+1,i} > n$.

Siccome $X \quad V$ ed $A \quad V$, abbiamo

$$T_{k+1}[X] = T_h[X]$$

e

$$T_{k+1,i} > n \qquad \text{e} \qquad T_{h,i} = i.$$

Allora, $T_{k+1,i}$ verrà posto uguale ad i ($\leq n$). Pertanto, dopo aver esaminato tutte le dipendenze funzionali in F, avendo l'Algoritmo di Chiusura generato $K^+ = R$ (visto che K è una chiave di M), il Test di Conservazione dei Dati avrà prodotto un tableau la cui riga T_{k+1} contiene solo valori minori o uguali ad n e questo, per il Teorema 7.5, dimostra che la scomposizione di M indotta da $\boldsymbol{S^*}$ conserva anche i dati. □

7.7 Scomposizione in Forma Normale di Boyce-Codd

Sia $M = [R, F]$ un modello. Se una dipendenza funzionale semplice non banale f che sia valida in M è tale che $det(f)$ non sia una sovrachiave di M, allora diremo che *f vìola la condizione di Boyce-Codd*.

***Lemma* 7.4**. Sia $M = [R, F]$ un modello in Prima Forma Normale.

(*i*) Se $|R| = 2$ allora M è in forma normale di Boyce-Codd.

(*ii*) Se M non è in forma normale di Boyce-Codd, allora esiste una dipendenza funzionale semplice non banale $X \rightarrow A$ valida in M tale che $|X| = |R| - 2$.

Dimostrazione. (*i*) Ovvio. (*ii*). Sia $X \rightarrow A$ una violazione della condizione di Boyce-Codd. La cardinalità di X non può essere uguale ad $|R| - 1$ perché altrimenti $X \rightarrow R$ sarebbe valida in M e, quindi, X conterrebbe una chiave di M. La tesi allora segue facilmente. □

in merito alla condizione (*ii*), si osservi che può darsi il caso che M sia in forma normale di Boyce-Codd e che tuttavia esista una dipendenza funzionale semplice non banale $X \rightarrow A$ valida in M tale che $|X| = |R| - 2$. Un esempio è dato dal modello con $R =$ **ABC** ed $F = \{$**A** $\rightarrow$ **BC**$\}$.

Dato un modello semplice $M = [R, F]$, il prossimo algoritmo permette di trovare un ricoprimento $\boldsymbol{S}$ di R tale che la scomposizione di M indotta da $\boldsymbol{S}$ conservi i dati e le proiezioni su $\boldsymbol{S}$ siano tutte in forma normale di Boyce-Codd. Prima di darne i dettagli, introduciamo la nozione di "connessione" per lo schema R di M. Dati due attributi A e B in R, diremo che B non *incide* su A se la dipendenza funzionale

$$R - \{A, B\} \rightarrow A$$

è valida in M. Se lo schema R di M contiene due attributi A e B tali che B non incida su A, allora diremo che R non è *connesso*. Così, R è connesso se per ogni attributo A in R si ha che, se $X \rightarrow A$ è una dipendenza funzionale nonbanale valida in M, allora

$$X = R - \{A\}.$$

***Lemma* 7.5**. Sia M un modello in Prima Forma Normale. Se lo schema di M è connesso, allora M è in forma normale di Boyce-Codd.
Dimostrazione. Per il Lemma 7.4, se M non è in forma normale di Boyce-Codd, allora R non è connesso, di qui la tesi. □

Il seguente algoritmo determina se R è connesso e, se non lo è, trova due attributi A e B tali che B non incide su A.

Test di Connessione

Dati: Un insieme $R = \{A_1, \ldots, A_n\}$ di attributi e un insieme F di dipendenze funzionali applicabili ad R

Risultato: Il valore della variabile logica *test*

Procedura

(1) *test* := VERO.

(2) Se $n = 2$, allora Uscire.

(3) Per $i = 1, \ldots, n–1$
per $j = i + 1, \ldots, n$

calcolare la chiusura di $R – \{A_i, A_j\}$;
se A_i $(R – \{A_i, A_j\})^+$ allora
$A := A_i$; $B := A_j$;
test := FALSO ed Uscire;

se A_j $(R – \{A_i, A_j\})^+$ allora
$A := A_j$; $B := A_i$;
test := FALSO ed Uscire.

Diremo che il Test di Connessione *dà esito positivo* (rispettivamente, *negativo*) se il valore finale della variabile *test* è VERO (rispettivamente, FALSO).

Scomposizione in Forma Normale di Boyce-Codd

Dati: Un insieme $R = \{A_1, \ldots, A_n\}$ di attributi e un insieme F di dipendenze funzionali applicabili ad R

Risultato: Un ricoprimento $\boldsymbol{S}$ di R.

Procedura

(1) $\boldsymbol{S} := \{R\}$; $V := R$.

(2) Applicare il Test di Connessione a V e se il test dà esito negativo,

(2.1) applicare la routine Separazione ($V, A, B; Y, Z$);
(2.2) $\boldsymbol{S} := (\boldsymbol{S} - \{V\}) + \{Y, Z\}$;
(2.3) $V := Z$ e Ripetere (2).

Routine di Separazione ($V, A, B; Y, Z$).

(I) $Y := V - \{B\}$.

(II) Fintantoché l'applicazione del Test di Connessione all'insieme Y dà esito negativo, $Y := Y - \{B\}$.

(III) $Z := V - \{A\}$.

Esempio 7.9 Consideriamo l'insieme di attributi $R = \{\mathbf{A}, \mathbf{C}, \mathbf{D}, \mathbf{O}, \mathbf{S}, \mathbf{T}\}$ dove **A** sta per Aula, **C** per Corso, **D** per Docente, **O** per Ora, **S** per Studente e **T** per Tutore. È ragionevole assumere l'insieme di dipendenze funzionali

$$F = \{\mathbf{AO} \rightarrow \mathbf{C}, \mathbf{C} \rightarrow \mathbf{D}, \mathbf{CS} \rightarrow \mathbf{T}, \mathbf{DO} \rightarrow \mathbf{A}, \mathbf{OS} \rightarrow \mathbf{A}\}.$$

Applichiamo ad R ed F l'algoritmo di Scomposizione in Forma Normale di Boyce-Codd con l'intesa di esaminare gli attributi in ordine alfabetico.

PRIMA ITERAZIONE.

(1) $V := \{\mathbf{A}, \mathbf{C}, \mathbf{D}, \mathbf{O}, \mathbf{S}, \mathbf{T}\}$.
(2) L'applicazione del Test di Connessione all'insieme V dà esito negativo e restituisce gli attributi $A := \mathbf{A}$ e $B := \mathbf{C}$.

(2.1) A questo punto viene eseguita la routine Separazione con input $(\{\mathbf{A}, \mathbf{C}, \mathbf{D}, \mathbf{O}, \mathbf{S}, \mathbf{T}\}, \mathbf{A}, \mathbf{C})$.

(I) Y := {**A**, **D**, **O**, **S**, **T**}.

(II) L'applicazione del Test di Connessione all'insieme Y dà esito negativo e restituisce gli attributi A := **A** e B := **D**.

Y := {**A**, **O**, **S**, **T**}. L'applicazione del Test di Connessione all'insieme Y dà esito negativo e restituisce gli attributi A := **A** e B := **T**.

Y := {**A**, **O**, **S**}. L'applicazione del Test di Connessione all'insieme Y dà esito positivo.

(III) Z := {**C**, **D**, **O**, **S**, **T**}

(2.2) $\boldsymbol{S}$:= {{**A**, **O**, **S**}, {**C**, **D**, **O**, **S**, **T**}}

(2.3) V := {**C**, **D**, **O**, **S**, **T**} e tutto è pronto per la seconda iterazione.

SECONDA ITERAZIONE.

(2) L'applicazione del Test di Connessione all'insieme V dà esito negativo e restituisce gli attributi A := **C** e B := **D**.

(2.1) A questo punto viene eseguita la routine Separazione con input ({**C**, **D**, **O**, **S**, **T**}, **C**, **D**).

(I) Y := {**C**, **O**, **S**, **T**}.

(II) L'applicazione del Test di Connessione all'insieme Y dà esito negativo e restituisce gli attributi A := **C** e B := **T**.

Y := {**C**, **O**, **S**}. L'applicazione del Test di Connessione all'insieme Y dà esito positivo.

(III) Z := {**D**, **O**, **S**, **T**}

(2.2) $\boldsymbol{S}$:= {{**A**, **O**, **S**}, {**C**, **O**, **S**}, {**D**, **O**, **S**, **T**}}

(2.3) V := {**D**, **O**, **S**, **T**} e tutto è pronto per la terza iterazione.

TERZA ITERAZIONE.

(2) L'applicazione del Test di Connessione all'insieme V dà esito negativo e restituisce gli attributi A := **D** e B := **T**.

(2.1) A questo punto viene eseguita la routine Separazione con input ({**D**, **O**, **S**, **T**}, **D**, **T**).

(I) Y := {**D**, **O**, **S**}. L'applicazione del Test di Connessione all'insieme Y dà esito positivo.

(III) Z := {**O**, **S**, **T**}

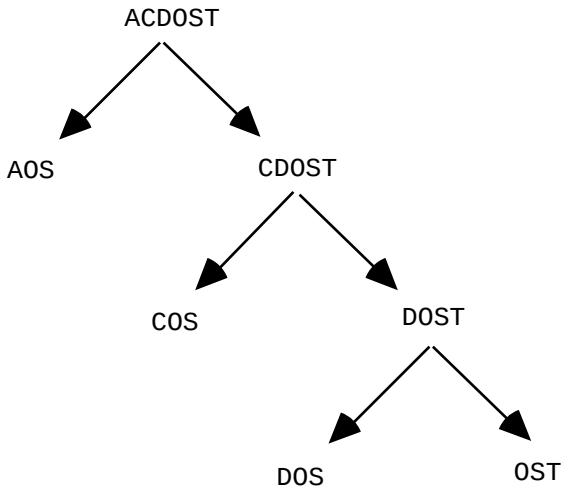
(2.2) $\boldsymbol{S}$:= {{**A**, **O**, **S**}, {**C**, **O**, **S**}, {**D**, **O**, **S**}, {**O**, **S**, **T**}}

(2.3) V := {**O**, **S**, **T**} e tutto è pronto per la quarta iterazione.

QUARTA ITERAZIONE.

(2) L'applicazione del Test di Connessione all'insieme V dà esito positivo e l'algoritmo termina.

L'intero processo di scomposizione può essere rappresentato graficamente con l'albero orientato:



◊

Si osservi che la scomposizione di un modello indotta dal ricoprimento che si trova applicando l'algoritmo di Scomposizione in Forma Normale di Boyce-Codd non necessariamente conserva le dipendenze.

***Teorema* 7.10**. Sia R un insieme di attributi e sia $\boldsymbol{F}$ un insieme di dipendenze funzionali semplici applicabili ad R. Per ogni modello $M = [R, F]$, l'algoritmo di Scomposizione in Forma Normale di Boyce-Codd calcola un ricoprimento $\boldsymbol{S}$ di R tale che la scomposizione di M indotta da $\boldsymbol{S}$ conservi i dati ed le proiezioni di M su $\boldsymbol{S}$ siano tutte in forma normale di Boyce-Codd.

Dimostrazione. Che la scomposizione conservi i dati lo si dimostra per induzione. Inizialmente, $\boldsymbol{S} = \{R\}$ e la proprietà è banalmente vera. Consideriamo poi il generico passo del processo di scomposizione. Sia $\boldsymbol{T}$ $\{V\}$ il valore corrente di $\boldsymbol{S}$ e sia $\boldsymbol{T}$ + $\{Y, Z\}$, il suo valore aggiornato per effetto della dipendenza funzionale $X \rightarrow A$. Così, la proiezione M' di M su V verrà scomposto nelle sue proiezioni su Y e Z. Per l'ipotesi induttiva, $\boldsymbol{T}$ $\{V\}$ è una scomposizione di M che conserva i dati; vale a dire, per ogni relazione r conforme ad M, abbiamo

$$r = \psi_{\boldsymbol{S}}(r) = \psi_{\boldsymbol{T}}(r) \quad \pi_V(r).$$

Inoltre, per il Teorema 7.5, anche la scomposizione di M' indotta da $\{Y, Z\}$ conserva i dati. Dunque, siccome $\pi_V(r)$ è una relazione conforme ad M', si ha pure che

$$\pi_V(r) = (\pi_Y(\pi_V(r))) \quad (\pi_Z(\pi_V(r))) = (\pi_Y(r)) \quad (\pi_Z(r)).$$

Combinando le due formule, si ottiene allora

$$r = \psi_{\boldsymbol{T}}(r) \quad (\pi_Y(r)) \quad (\pi_Z(r)),$$

il che prova che anche la scomposizione di M indotta dal $\boldsymbol{T} + \{Y, Z\}$ conserva i dati. Infine per dimostrare che il valore finale di $\boldsymbol{S}$ è tale che le proiezioni di M su $\boldsymbol{S}$ sono tutte in forma normale di Boyce-Codd, è sufficiente osservare che, quando M' viene scomposto nelle sue proiezioni su Y e Z, la proiezione di M' su Y è in forma normale di Boyce-Codd per il Lemma 7.5. □