

La maggior parte dei sistemi di basi di dati oggi sul mercato si basa sul modello relazionale, che fu proposto, in una pubblicazione scientifica (di E.F. Codd [26]), nel 1970, al fine di superare le limitazioni dei modelli all'epoca utilizzati a livello logico, che non permettevano di realizzare efficacemente la proprietà di indipendenza dei dati, già riconosciuta fondamentale. L'affermazione del modello relazionale è stata abbastanza lenta, a causa proprio dell'alto livello di astrazione: non è stato immediato individuare realizzazioni efficienti per strutture significativamente diverse da quelle all'epoca utilizzate. Infatti, nonostante i primi prototipi di sistemi relazionali siano stati realizzati già nei primi anni Settanta, i primi sistemi relazionali sono apparsi sul mercato nel 1981, acquisendone una frazione significativa solo a metà degli anni Ottanta.

La presentazione del modello relazionale è articolata in tre capitoli, questo e i due successivi. In questo capitolo sono illustrate le caratteristiche strutturali del modello, cioè le modalità secondo cui esso permette di organizzare i dati. Dopo una breve discussione sui vari modelli logici, si mostra come il concetto di relazione possa essere mutuato dalla teoria degli insiemi ed utilizzato, con alcune varianti, per rappresentare le informazioni di interesse in una base di dati. Viene in particolare approfondito il fatto che le corrispondenze fra dati in strutture diverse sono rappresentate per mezzo dei dati stessi. Poi, dopo una breve discussione delle modalità per la rappresentazione di informazione incompleta, l'attenzione viene volta ai vincoli di integrità, che permettono di specificare proprietà aggiuntive che debbono essere soddisfatte dalle basi di dati.

La presentazione del modello relazionale è completata nei due capitoli successivi, il primo dedicato alla specifica delle operazioni di interrogazione di basi di dati relazionali e il secondo al linguaggio SQL, che, nei DBMS oggi esistenti, permette di definire, aggiornare e interrogare basi di dati.

## 2.1 Modelli logici nei sistemi di basi di dati

Il *modello relazionale* si basa su due concetti, *relazione* e *tabella*, di natura diversa ma facilmente riconducibili l'uno all'altro. La nozione di *relazione* proviene dalla matematica, in particolare dalla teoria degli insiemi, mentre il concetto di *tabella* è semplice ed intuitivo. La presenza contemporanea di questi due concetti, l'uno

formale e l'altro intuitivo, è responsabile del grande successo ottenuto dal modello relazionale. Infatti, le tabelle risultano naturali e comprensibili anche per gli utenti finali (che spesso le utilizzano in tanti contesti per scopi diversi, senza riferimento alle basi di dati). D'altra parte, la disponibilità di una formalizzazione al tempo stesso semplice e chiara ha permesso anche uno sviluppo teorico a supporto del modello con risultati di interesse concreto.

Il modello relazionale risponde al requisito dell'indipendenza dei dati, che, come abbiamo visto nel capitolo 1 prevede una distinzione, nella descrizione dei dati, fra il livello *fisico* e il livello *logico*: gli utenti che accedono ai dati e i programmatori che sviluppano le applicazioni fanno riferimento solo al livello logico; i dati descritti al livello logico sono poi realizzati per mezzo di opportune strutture fisiche, ma per accedere ai dati non è necessario conoscere le strutture fisiche stesse. Il modello relazionale fu proposto alla fine degli anni Sessanta per permettere, a livello logico, una descrizione efficace. Infatti, i modelli precedentemente proposti, quello reticolare e quello gerarchico, includevano espliciti riferimenti alla sottostante struttura realizzativa, attraverso l'uso di puntatori e l'ordinamento fisico dei dati.

Una precisazione è utile prima di passare all'introduzione del modello relazionale. Il termine *relazione* viene utilizzato in questo testo (e in generale con riferimento alle basi di dati) in tre accezioni che, nei dettagli, differiscono in modo importante:

- *Relazione matematica*, secondo la definizione normalmente data nella teoria degli insiemi elementare. Da questa nozione, che verrà richiamata nel prossimo paragrafo, derivano le altre due.
- *Relazione* secondo la definizione del modello relazionale che, come vedremo nel paragrafo 2.3, presenta alcune differenze rispetto a quella della teoria degli insiemi.
- *Relazione*, come traduzione di *Relationship*,<sup>1</sup> costruito dal modello concettuale *Entità-Relazione* (in inglese *Entity-Relationship*) utilizzato, come vedremo nella discussione della progettazione concettuale (capitolo 5), per descrivere legami tra entità del mondo reale.

## 2.2 Relazioni e tabelle

Ricordiamo che, dati due insiemi  $D_1$  e  $D_2$ , si chiama *prodotto cartesiano* di  $D_1$  e  $D_2$ , in simboli  $D_1 \times D_2$ , l'insieme delle coppie ordinate  $(v_1, v_2)$ , tali che  $v_1$  è un elemento di  $D_1$  e  $v_2$  è un elemento di  $D_2$ . Ad esempio, dati gli insiemi  $A = \{1, 2, 4\}$  e  $B = \{a, b\}$ , il prodotto cartesiano  $A \times B$  è costituito dall'insieme di tutte le possibili coppie in cui il primo elemento appartiene ad  $A$  e il secondo a  $B$ . Poiché  $A$  ha tre elementi e  $B$  due, si tratta quindi di sei coppie:

$$\{(1, a), (1, b), (2, a), (2, b), (4, a), (4, b)\}$$

<sup>1</sup>Nei due casi precedenti si usa in inglese il termine *relation*, che quindi non è ambiguo rispetto a *relationship*. In questo contesto, si usano talvolta altre traduzioni, diverse da *relazione*, come *associazione* o *correlazione*.

Una *relazione matematica* sugli insiemi  $D_1$  e  $D_2$  (chiamati *domini* della relazione) è un sottoinsieme di  $D_1 \times D_2$ . Dati gli insiemi  $A$  e  $B$  di cui sopra, una possibile relazione matematica su  $A$  e  $B$  è costituita dall'insieme di coppie  $\{(1,a),(1,b),(4,b)\}$ .

Vale la pena fare una annotazione, importante dal punto di vista formale (anche se pressoché ovvia da quello pratico). Finora non abbiamo detto niente riguardo alla finitezza degli insiemi che consideriamo, e quindi abbiamo implicitamente ammesso la possibilità di insiemi infiniti (e quindi di relazioni infinite). In pratica, poiché le nostre basi di dati debbono essere memorizzate in sistemi di calcolo di dimensione finita, le relazioni sono necessariamente finite. Peraltro, in talune trattazioni teoriche, che comunque esulano dagli interessi di questo testo, vengono talvolta ammesse relazioni infinite. Al tempo stesso, è comodo talvolta che i domini abbiano dimensione infinita (in modo che sia sempre possibile assumere l'esistenza di un valore non presente nella base di dati). Pertanto, assumeremo ove necessario che le nostre basi di dati siano costituite da relazioni finite su domini eventualmente infiniti.

Le relazioni possono essere rappresentate graficamente, in maniera utile ed espressiva, sotto forma tabellare. Le due tabelle riportate nella figura 2.1 descrivono il prodotto cartesiano  $A \times B$  e la relazione matematica su  $A$  e  $B$  dei capoversi precedenti.

1	a
1	b
2	a
2	b
4	a
4	b

1	a
1	b
4	b

Figura 2.1 Rappresentazione tabellare di un prodotto cartesiano e una relazione

Le definizioni precedenti di prodotto cartesiano e relazione matematica fanno riferimento a due insiemi, ma possono essere generalizzate rispetto al numero di insiemi. Dati  $n > 0$  insiemi  $D_1, D_2, \dots, D_n$ , non necessariamente distinti, il prodotto cartesiano di  $D_1, D_2, \dots, D_n$ , indicato con  $D_1 \times D_2 \times \dots \times D_n$ , è costituito dall'insieme delle  $n$ -uple  $(v_1, v_2, \dots, v_n)$ , tali che  $v_i$  appartiene a  $D_i$ , per  $1 \leq i \leq n$ . Una relazione matematica sui domini  $D_1, D_2, \dots, D_n$  è un sottoinsieme del prodotto cartesiano  $D_1 \times D_2 \times \dots \times D_n$ . Il numero  $n$  delle componenti del prodotto cartesiano (e quindi di ogni  $n$ -upla) viene detto *grado* del prodotto cartesiano e della relazione. Il numero di elementi (cioè di  $n$ -uple) della relazione viene chiamato, come di solito nella teoria degli insiemi, *cardinalità* della relazione. Nella figura 2.2 sono mostrate le rappresentazioni tabellari del prodotto cartesiano e di una relazione di grado tre sui domini  $C = \{x, y\}$ ,  $D = \{a, b, c\}$  ed  $E = \{3, 5\}$ . La relazione ha cardinalità pari a sei.

Le relazioni (e le corrispondenti tabelle) possono essere utilizzate per rappresentare i dati di interesse per qualche applicazione. Ad esempio, la relazione nella figura 2.3 contiene i dati relativi ai risultati di un insieme di partite di calcio. Essa

x	a	3
x	a	5
x	b	3
x	b	5
x	c	3
x	c	5
y	a	3
y	a	5
y	b	3
y	b	5
y	c	3
y	c	5

x	a	3
x	a	5
x	c	5
y	a	3
y	c	3
y	c	5

Figura 2.2 Rappresentazione tabellare di un prodotto cartesiano e una relazione ternari

Juventus	Lazio	3	1
Lazio	Milan	2	0
Juventus	Roma	1	2
Roma	Milan	0	1

Figura 2.3 Una relazione con i risultati di partite di calcio

è definita con riferimento a due domini *intero* e *stringa*, ognuno dei quali compare due volte. La relazione è infatti un sottoinsieme del prodotto cartesiano:

$$\text{Stringa} \times \text{Stringa} \times \text{Intero} \times \text{Intero}$$

## 2.3 Relazioni con attributi

Sulle relazioni e sulle loro rappresentazioni tabellari, possiamo fare varie osservazioni. In base alla definizione, una relazione matematica è un *insieme* di  $n$ -uple ordinate  $(v_1, v_2, \dots, v_n)$ , con  $v_1 \in D_1$ ,  $v_2 \in D_2$ ,  $\dots$ ,  $v_n \in D_n$ . Con riferimento all'uso che facciamo delle relazioni per organizzare i dati nelle nostre basi di dati, possiamo dire che ciascuna  $n$ -upla contiene dati fra loro collegati, anzi, stabilisce un legame fra loro: ad esempio la prima  $n$ -upla della relazione nella figura 2.3 stabilisce un legame fra i valori "Juventus," "Lazio," "3," "1", ad indicare che il risultato della partita fra Juventus e Lazio è 3 a 1. Possiamo poi ricordare che una relazione è un *insieme* e quindi:

- non è definito alcun ordinamento fra le  $n$ -uple; nelle tabelle che le rappresentano c'è, per necessità, un ordine, ma è "occasionale": due tabelle con le stesse righe, ma in ordine diverso, rappresentano la stessa relazione;

- le  $n$ -uple di una relazione sono distinte l'una dall'altra, in quanto tra gli elementi di un insieme non possono essere presenti due elementi uguali; quindi una tabella rappresenta una relazione solo se le sue righe sono l'una diversa dall'altra.

Al tempo stesso, ciascuna  $n$ -upla è, al proprio interno, *ordinata*: l' $i$ -esimo valore di ciascuna proviene dall' $i$ -esimo dominio. È cioè definito un ordinamento fra i domini, che è significativo ai fini dell'interpretazione dei dati nelle relazioni: se, nella relazione nella figura 2.3, scambiassimo il terzo dominio con il quarto, cambieremmo completamente il significato della nostra relazione, in quanto i risultati delle partite verrebbero invertiti. In effetti, questo accade perché nella relazione ciascuno dei due domini *intero* e *stringa* compare due volte, e le due occorrenze sono distinte attraverso la posizione: la prima occorrenza del dominio *stringa* fa riferimento alla squadra di casa e la seconda a quella ospitata; analogamente le due occorrenze del dominio *intero*.

L'ordinamento che abbiamo appena evidenziato fra i domini di una relazione corrisponde in effetti ad una caratteristica insoddisfacente del concetto di relazione matematica rispetto alla possibilità di organizzare e utilizzare i dati. Infatti, in vari contesti dell'informatica si tende a privilegiare notazioni *non posizionali* (quali quelle che permettono, in un linguaggio ad alto livello, di far riferimento ai campi di un record per mezzo di nomi simbolici) rispetto a quelle posizionali (utilizzate ad esempio negli array per indicare il primo elemento, il secondo, e così via): si tende ad utilizzare le notazioni posizionali solo quando l'ordinamento corrisponde ad una esigenza intrinseca, come accade ad esempio nei problemi di natura matematica, in cui gli array permettono di rappresentare in modo ovvio e diretto vettori e matrici. Risulta evidente come le informazioni che siamo interessati a organizzare nelle relazioni delle nostre basi di dati hanno una struttura che si può naturalmente ricondurre a quella dei record: una relazione è sostanzialmente un insieme di record omogenei, cioè definiti sugli stessi campi. Nel caso dei record, ad ogni campo è associato un nome: associamo a ciascuna occorrenza di dominio nella relazione un nome, detto *attributo*, che descrive il "ruolo" giocato dal dominio stesso. Ad esempio, per la relazione relativa alle partite, possiamo usare nomi quali *SquadraDiCasa*, *SquadraOspitata*, *RetiCasa*, *RetiOspitata*; nella rappresentazione tabellare, utilizziamo gli attributi come intestazioni per le colonne (vedi figura 2.4); sottolineiamo che, dovendo identificare univocamente le componenti, gli attributi di una relazione (e quindi le intestazioni delle colonne delle tabelle) debbono essere diversi l'uno dall'altro.

SquadraDiCasa	SquadraOspitata	RetiCasa	RetiOspitata
Juventus	Lazio	3	1
Lazio	Milan	2	0
Juventus	Roma	1	2
Roma	Milan	0	1

Figura 2.4 Una relazione con attributi

Modificando la definizione di relazione con l'introduzione degli attributi, e prima ancora di dare la definizione formale, possiamo vedere che l'ordinamento degli attributi (e delle colonne nella rappresentazione tabellare) risulta irrilevante: non è più necessario parlare di primo dominio, secondo dominio e così via, è sufficiente far riferimento agli attributi. La figura 2.5 mostra un'altra rappresentazione tabellare della relazione nella figura 2.4, con gli attributi (e quindi le colonne) in ordine diverso (secondo lo stile americano in cui la squadra di casa viene indicata dopo quella ospitata).

SquadraOspitata	SquadraDiCasa	RetiOspitata	RetiCasa
Lazio	Juventus	1	3
Milan	Lazio	0	2
Roma	Juventus	2	1
Milan	Roma	1	0

Figura 2.5 Un'altra rappresentazione per la relazione nella figura 2.4

Facoltativo

Per formalizzare i concetti, stabiliamo la corrispondenza fra attributi e domini per mezzo di una funzione  $\text{DOM} : X \rightarrow \mathcal{D}$ , che associa a ciascun attributo  $A \in X$  un dominio  $\text{DOM}(A) \in \mathcal{D}$ . Poi, diciamo che una *tupla*<sup>2</sup> su un insieme di attributi  $X$  è una funzione  $t$  che associa a ciascun attributo  $A \in X$  un valore del dominio  $\text{DOM}(A)$ . Possiamo quindi dare la nuova definizione di relazione: una *relazione* su  $X$  è un insieme di tuple su  $X$ . La differenza fra questa definizione e quella tradizionale di relazione matematica risiede solo nella definizione di tupla: nella relazione matematica abbiamo  $n$ -uple i cui elementi sono individuati per posizione, mentre, nelle tuple della nuova definizione, gli elementi sono individuati per mezzo degli attributi, cioè con una tecnica non posizionale.

Introduciamo una notazione che utilizzeremo molto in seguito. Se  $t$  è una tupla su  $X$  e  $A \in X$ ,  $t[A]$  (o  $t.A$ ) indica il valore di  $t$  su  $A$ . Ad esempio, se  $t$  è la prima tupla della relazione<sup>3</sup> nella figura 2.5, possiamo dire che

$$t[\text{SquadraOspitata}] = \text{Lazio}$$

La stessa notazione è estesa anche ad insiemi di attributi, nel qual caso denota tuple:

$$t[\text{SquadraOspitata}, \text{RetiOspitata}]$$

è una tupla su due attributi.<sup>4</sup>

<sup>2</sup>Traslitterazione dell'inglese "tuple." In italiano sarebbe forse più corretto usare il termine *ennupla*, ma è ormai diffuso il termine *tupla*, che permette di sottolineare la differenza con l'usuale concetto di  $n$ -upla ordinata visto in precedenza.

<sup>3</sup>Più precisamente, dovremmo dire "la tupla rappresentata dalla prima riga della tabella ...", ma sarebbe un inutile appesantimento.

<sup>4</sup>C'è qui una incoerenza nella notazione: se  $A$  è un attributo, allora  $t[A]$  è un valore, mentre se  $X$  è un insieme di attributi, allora  $t[X]$  è una tupla, cioè una funzione. Inoltre, come vedremo, insiemi costituiti da un singolo attributo verranno denotati dal nome dell'attributo stesso; quindi,

## 2.4 Relazioni e basi di dati

Come già notato, una relazione può essere utilizzata per organizzare dati rilevanti nell'ambito di una applicazione di interesse. Peraltro, di solito non è sufficiente allo scopo una singola relazione: una base di dati è in generale costituita da più relazioni, le cui tuple contengono valori comuni, ove necessario per stabilire corrispondenze. Approfondiamo questo concetto commentando la base di dati nella figura 2.6:

- la prima relazione contiene informazioni relative ad un insieme di studenti, con numero di matricola, cognome, nome e data di nascita;
- la terza relazione contiene informazioni su alcuni corsi, con codice, titolo e docente;
- la seconda relazione contiene informazioni relative ad esami: il numero di matricola dello studente, il codice del corso e il voto; questa relazione fa riferimento ai dati contenuti nelle altre due: agli studenti, attraverso i numeri di matricola e ai corsi attraverso i relativi codici.

STUDENTI	Matricola	Cognome	Nome	Data di nascita
	276545	Rossi	Maria	25/11/1971
	485745	Neri	Anna	23/04/1972
	200768	Verdi	Fabio	12/02/1972
	587614	Rossi	Luca	10/10/1971
	937653	Bruni	Mario	01/12/1971

ESAMI	Studente	Voto	Corso
	276545	28	01
	276545	27	04
	937653	25	01
	200768	24	04

CORSI	Codice	Titolo	Docente
	01	Analisi	Giani
	03	Chimica	Melli
	04	Chimica	Belli

Figura 2.6 Una base di dati relazionale

$t[A]$  denota sia un valore sia una tupla su un attributo. Peraltro, l'ambiguità sarà di solito irrilevante e verrà risolta esplicitamente quando necessario.

La base di dati nella figura 2.6 mostra una delle caratteristiche fondamentali del modello relazionale, che viene spesso indicata dicendo che esso è "basato su valori:" i riferimenti fra dati in relazioni diverse sono rappresentati per mezzo di valori dei domini che compaiono nelle tuple. Va notato che gli altri modelli logici, reticolare e gerarchico (definiti prima del modello relazionale ma tuttora in uso), realizzano le corrispondenze in modo esplicito attraverso puntatori e vengono pertanto detti modelli "basati su record e puntatori." Rimandando all'appendice A per una presentazione articolata del modello reticolare, schematizziamo le caratteristiche fondamentali di un modello ideale basato su record e puntatori nella figura 2.7, che rappresenta la stessa base di dati nella figura 2.6, dove abbiamo utilizzato puntatori al posto dei riferimenti realizzati tramite valori (i numeri di matricola degli studenti e i codici dei corsi).

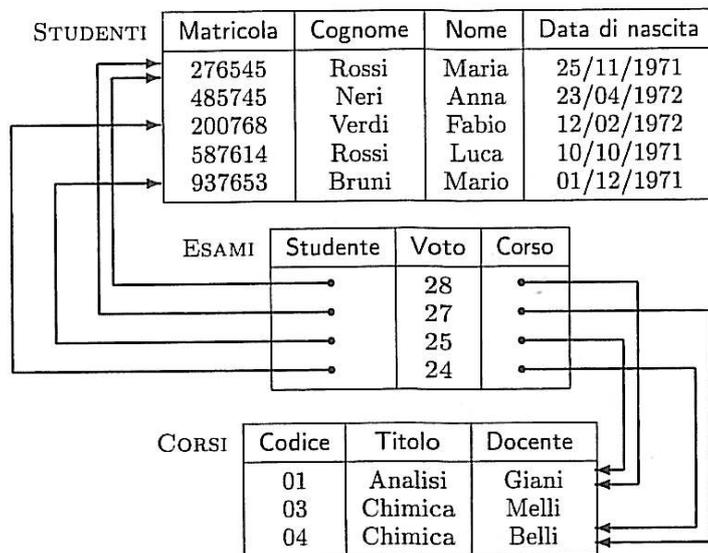


Figura 2.7 Una base di dati con puntatori

Rispetto ad un modello basato su record e puntatori, il modello relazionale, basato su valori, presenta diversi vantaggi:

- esso richiede di rappresentare solo ciò che è rilevante dal punto di vista dell'applicazione (dell'utente); i puntatori sono qualcosa di aggiuntivo, legato ad aspetti realizzativi; nei modelli con puntatori, il programmatore delle applicazioni fa riferimento a dati che non sono significativi per l'applicazione;
- essendo tutta l'informazione contenuta nei valori, è relativamente semplice trasferire i dati da un contesto ad un altro (per esempio se si deve trasferire una base di dati da un calcolatore ad un altro); in presenza di puntatori,

l'operazione è più complessa, perché i puntatori hanno un significato locale al singolo sistema, che non sempre è immediato esportare;

- la rappresentazione logica dei dati (costituita dai soli valori) non fa alcun riferimento a quella fisica, che può anche cambiare nel tempo: il modello relazionale permette quindi di ottenere l'indipendenza fisica dei dati.

A titolo di inciso, vale la pena notare che anche in una base di dati relazionale, a livello fisico, i dati possono essere rappresentati secondo modalità che prevedono l'uso di puntatori. La differenza, rispetto ai modelli basati su puntatore è nel fatto che qui i puntatori non sono visibili a livello logico. Inoltre, sottolineiamo come nei sistemi di basi di dati ad oggetti, che rappresentano una delle direzioni di evoluzione delle basi di dati e che discuteremo nel capitolo 11, vengono introdotti gli identificatori di oggetto, che, pur ad un livello di astrazione più alto, presentano alcune delle caratteristiche dei puntatori.

Possiamo a questo punto riassumere le definizioni relative al modello relazionale, con un po' di precisione, distinguendo il livello degli schemi da quello delle istanze

- Uno *schema di relazione* è costituito da un simbolo, detto *nome della relazione*  $R$  e da un insieme di (nomi di) *attributi*  $X = \{A_1, A_2, \dots, A_n\}$ , il tutto di solito indicato  $R(X)$ . A ciascun attributo è associato un dominio, come visto in precedenza.
- Uno *schema di base di dati* è un insieme di schemi di relazione con nomi diversi:

$$\mathbf{R} = \{R_1(X_1), R_2(X_2), \dots, R_n(X_n)\}$$

I nomi di relazione hanno come scopo principale quello di distinguere le varie relazioni nella base di dati.

- Una *istanza di relazione* (o semplicemente *relazione*) su uno schema  $R(X)$  è un insieme  $r$  di tuple su  $X$ .
- Un'*istanza di base di dati* (o semplicemente *base di dati*) su uno schema  $\mathbf{R} = \{R_1(X_1), R_2(X_2), \dots, R_n(X_n)\}$  è un insieme di relazioni  $r = \{r_1, r_2, \dots, r_n\}$ , dove ogni  $r_i$ , per  $1 \leq i \leq n$ , è una relazione sullo schema  $R_i(X_i)$ .

Per esemplificare, possiamo dire che lo schema della base di dati nella figura 2.6 è così definito (con opportune definizioni per i domini):

$$\mathbf{R} = \{\text{STUDENTI (Matricola, Cognome, Nome, Data di nascita),} \\ \text{ESAMI (Studente, Voto, Corso),} \\ \text{CORSI (Codice, Titolo, Docente)}\}$$

Per comodità, accenniamo brevemente alle convenzioni che adotteremo nel seguito (e che abbiamo già usato nelle definizioni e negli esempi), allo scopo di favorire la sinteticità della notazione senza compromettere la comprensione:

- gli attributi (quando non si utilizzeranno nomi significativi dal punto di vista dell'applicazione) verranno indicati con lettere iniziali dell'alfabeto, maiuscole, eventualmente con indici e/o pedici:  $A, B, C, A', A_1, \dots$

- insiemi di attributi verranno indicati con lettere finali dell'alfabeto, maiuscole,  $X, Y, Z, X', X_1, \dots$ ; un insieme in cui si vogliono evidenziare gli attributi componenti verrà denotato dalla giustapposizione dei nomi degli attributi stessi: scriveremo cioè  $X = ABC$  anziché  $X = \{A, B, C\}$ ; analogamente, l'unione di insiemi verrà denotata dalla giustapposizione dei relativi nomi: scriveremo  $XY$  anziché  $X \cup Y$ ; riunendo le due convenzioni, scriveremo  $XA$  anziché  $X \cup \{A\}$ ;
- per i nomi di relazione (sempre nel caso in cui si possa o debba fare a meno di nomi significativi) utilizzeremo la  $R$  e le lettere circostanti, maiuscole:  $R_1, S, S', \dots$ ; per le relazioni, gli stessi simboli dei corrispondenti nomi di relazione, ma in lettere minuscole;

Per approfondire ancora i concetti fondamentali del modello relazionale, discutiamo un altro paio di esempi.

In primo luogo notiamo come, secondo la definizione, siano ammissibili relazioni su un solo attributo. Ciò può avere senso in particolare in basi di dati su più relazioni, in cui la relazione su singolo attributo contiene valori che appaiono come valori di un attributo di un'altra relazione. Ad esempio, in una base di dati in cui sia presente la relazione **STUDENTI** nella figura 2.6, si può utilizzare (vedi figura 2.8) un'altra relazione sul solo attributo **Matricola** per indicare gli studenti lavoratori (attraverso i relativi numeri di matricola, che debbono apparire nella relazione **STUDENTI**).

STUDENTI

Matricola	Cognome	Nome	Data di nascita
276545	Rossi	Maria	25/11/1971
485745	Neri	Anna	23/04/1972
200768	Verdi	Fabio	12/02/1972
587614	Rossi	Luca	10/10/1971
937653	Bruni	Mario	01/12/1971

LAVORATORI

Matricola
276545
485745
937653

Figura 2.8 Una relazione su un solo attributo

Discutiamo ora un esempio un po' più complesso, che mostra come, sia pur indirettamente, il modello relazionale permetta di rappresentare informazione strutturata in modo articolato. Nella figura 2.9 sono schematizzate tre ricevute fiscali emesse da un ristorante. Esse hanno una struttura che prevede (a parte le frasi prestampate che abbiamo riportato in grassetto) alcune informazioni fisse (numero, data e totale) e un numero di righe variabile, ognuna relativa ad un insieme di portate omogenee (con quantità, descrizione e importo complessivo). Poiché le nostre relazioni hanno una struttura fissa, non è possibile rappresentare l'insieme delle ricevute con un'unica relazione (non sarebbe possibile rappresentare le righe in numero non predeterminato). Possiamo però rappresentare le relative informazioni per mezzo di due relazioni, come mostrato nella figura 2.10: la relazione

"DA MARIO"		
Ricevuta n. 2357		
del 5/5/92		
3	coperti	6000
2	antipasti	12000
3	primi	27000
2	bistecche	36000
Totale		81000

"DA MARIO"		
Ricevuta n. 2334		
del 4/7/92		
2	coperti	4000
1	antipasti	6000
2	primi	15000
2	orate	50000
2	caffé	3000
Totale		78000

"DA MARIO"		
Ricevuta n. 2007		
del 4/8/92		
3	coperti	6000
2	antipasti	14000
3	primi	20000
1	orate	25000
1	caprese	8000
2	caffé	3000
Totale		76000

Figura 2.9 Alcune ricevute fiscali

RICEVUTE contiene i dati presenti una sola volta in ciascuna ricevuta (numero, data e totale) e la relazione DETTAGLIO contiene le varie righe di ciascuna ricevuta (con quantità, descrizione e importo complessivo), associate alla ricevuta stessa tramite il relativo numero.

RICEVUTE			
Num	Data	Totale	
2357	5/5/92	81000	
2334	4/7/92	78000	
2007	4/8/92	76000	

DETTAGLIO			
Num	Q.tà	Descr	Importo
2357	3	coperti	6000
2357	2	antipasti	12000
2357	3	primi	27000
2357	2	bistecche	36000
2334	2	coperti	4000
2334	1	antipasti	6000
2334	2	primi	15000
2334	2	orate	50000
2334	2	caffé	3000
2007	3	coperti	6000
2007	2	antipasti	14000
2007	3	primi	20000
2007	1	orate	25000
2007	1	caprese	8000
2007	2	caffé	3000

Figura 2.10 Una base di dati per le ricevute fiscali nella figura 2.9

È opportuno notare che la base di dati nella figura 2.10 rappresenta correttamente le ricevute solo a due condizioni:

- non interessa mantenere traccia dell'ordine con cui le righe compaiono in ciascuna ricevuta: infatti, poiché nessun ordinamento è definito fra le tuple di una relazione, le tuple di DETTAGLIO non sono in alcun modo ordinate;
- in una ricevuta non compaiono due righe uguali (il che potrebbe accadere in presenza di ordinazioni diverse relative alle stesse pietanze con le stesse quantità).

In entrambi i due casi, si può risolvere il problema aggiungendo un attributo, che indica la posizione della riga sulla ricevuta (vedi figura 2.11): in questo modo è sempre possibile ricostruire perfettamente il contenuto di tutte le ricevute. In generale, possiamo dire che la soluzione di figura 2.10 è da preferirsi quando le informazioni sulla ricevuta interessano solo in quanto tali (e nelle ricevute non vi sono righe ripetute), mentre quella di figura 2.11 permette di tenere traccia dell'effettiva impaginazione di ciascuna ricevuta. L'esempio ci permette quindi di notare che, anche in una stessa situazione, i dati da rappresentare nella base di dati possono essere diversi a seconda degli specifici obiettivi che ci si prefigge.

RICEVUTE

Num	Data	Totale
2357	5/5/92	81000
2334	4/7/92	78000
2007	4/8/92	76000

DETTAGLIO

Num	Riga	Q.tà	Descr	Importo
2357	1	3	coperti	6000
2357	2	2	antipasti	12000
2357	3	3	primi	27000
2357	4	2	bistecche	36000
2334	1	2	coperti	4000
2334	2	1	antipasti	6000
2334	3	2	primi	15000
2334	4	2	orate	50000
2334	5	2	caffé	3000
2007	1	3	coperti	6000
2007	2	2	antipasti	14000
2007	3	3	primi	20000
2007	4	1	orate	25000
2007	5	1	caprese	8000
2007	6	2	caffé	3000

Figura 2.11 Un'altra base di dati per le ricevute fiscali

## 2.5 Informazione incompleta e valori nulli

La struttura del modello relazionale, come discussa nei paragrafi precedenti, è indubbiamente molto semplice e potente. Al tempo stesso, essa impone però un certo grado di rigidità, in quanto le informazioni debbono essere rappresentate per mezzo di tuple di dati omogenee: in particolare, in ogni relazione possiamo rappresentare solo tuple corrispondenti allo schema della relazione stessa. In effetti, in molti casi, i dati disponibili possono non corrispondere esattamente al formato previsto. Ad esempio, in una relazione sullo schema:

PERSONE(Cognome, Nome, Indirizzo, Telefono)

il valore dell'attributo Telefono potrebbe non essere disponibile per tutte le tuple. Vale la pena notare che non sarebbe corretto utilizzare un valore del dominio per rappresentare l'assenza di informazione, in quanto si potrebbe in tal modo ingenerare confusione. In questo caso, supponendo i numeri telefonici rappresentati per mezzo di interi, potremmo ad esempio utilizzare lo zero per indicare l'assenza di un valore significativo. In generale, però, questa scelta non risulta soddisfacente, per due motivi. In primo luogo, essa richiede l'esistenza di un valore del dominio mai utilizzato per valori significativi: nel caso dei numeri di telefono, lo zero è chiaramente distinguibile, ma in altri casi non esiste un valore disponibile allo scopo; ad esempio, in un attributo che rappresenti la data di nascita e che utilizzi come dominio un tipo Data correttamente definito, non esistono elementi "non utilizzati" per valori significativi e quindi utilizzabili per denotare assenza di informazione. In secondo luogo, l'uso di valori del dominio può generare confusione: la distinzione fra valori "veri" e valori fittizi è nascosta, e quindi i programmi che accedono alla base di dati debbono tenerne conto, distinguendo opportunamente (e tenendo conto di quali sono, in ciascun caso, i valori fittizi). Per rappresentare in modo semplice, ma al tempo stesso comodo, la non disponibilità di valori, il concetto di relazione viene di solito esteso prevedendo che una tupla possa assumere, su ciascun attributo, o un valore del dominio, come visto finora, oppure un valore speciale, detto *valore nullo*, che denota appunto l'assenza di informazione, ma è un valore aggiuntivo rispetto a quelli del dominio, e ben distinto da essi. Nelle rappresentazioni tabellari, utilizzeremo per il valore nullo il simbolo NULL, come nella figura 2.12. Con riferimento alla tabella in figura, possiamo notare come in effetti i tre valori nulli che compaiono in essa sono dovuti a motivazioni diverse, come segue.

Città	IndirizzoPrefettura
Roma	Via Quattro Novembre
Firenze	NULL
Tivoli	NULL
Prato	NULL

Figura 2.12 Una relazione con valori nulli

- Firenze è capoluogo di provincia e quindi ha certamente una prefettura. Al momento, non disponiamo del suo indirizzo. Il valore nullo sostituisce un valore ordinario, non noto alla base di dati: per questo diciamo che si tratta di un valore *sconosciuto*.
- Tivoli non è capoluogo di provincia e quindi non ha una prefettura. Quindi l'attributo IndirizzoPrefettura non può avere un valore per questa tupla. Il valore nullo denota l'inapplicabilità dell'attributo o l'inesistenza del valore: il valore è *inesistente*.
- La provincia di Prato, nel momento in cui scriviamo, è da poco stata istituita e non sappiamo se la prefettura sia già stata costituita, né conosciamo il suo indirizzo (già operativo o previsto). In sostanza, non sappiamo se il valore esista e, in caso affermativo, non lo conosciamo. Sostanzialmente, ci troviamo in una situazione che corrisponde alla disgiunzione logica (l'"or") delle due precedenti: il valore è inesistente oppure sconosciuto. Questo tipo di valore nullo viene di solito chiamato *senza informazione*, perché non ci dice assolutamente niente: il valore può esistere o non esistere, e se esiste non sappiamo quale sia.

Nei sistemi di basi di dati relazionali, è di solito prevista una gestione molto semplice, ma tutto sommato efficace, del valore nullo, sul quale non viene fatta alcuna ipotesi e quindi in pratica ci si trova nell'ultimo caso, quello del valore senza informazione.

Per una ulteriore riflessione sui valori nulli, consideriamo la base di dati nella figura 2.13 che è definita sullo stesso schema della base di dati di figura 2.6. Il valore nullo sulla data di nascita nella prima tupla della relazione STUDENTI è tutto sommato ammissibile, in quanto si può pensare che l'informazione non sia in questo contesto essenziale. Viceversa, un valore nullo sul numero di matricola o sul codice di un corso genera problemi maggiori, in quanto questi valori, come abbiamo discusso con riferimento alla figura 2.6, sono utilizzati per stabilire correlazioni fra

STUDENTI

Matricola	Cognome	Nome	Data di nascita
276545	Rossi	Maria	NULL
NULL	Neri	Anna	23/04/1972
NULL	Verdi	Fabio	12/02/1972

ESAMI

Studente	Voto	Corso
276545	28	01
NULL	27	NULL
200768	24	NULL

CORSI

Codice	Titolo	Docente
01	Analisi	Giani
03	Chimica	NULL
NULL	Chimica	Belli

Figura 2.13 Una base di dati con molti valori nulli

tuple di relazioni diverse. Poi, la presenza di valori nulli nella relazione ESAMI rende addirittura inutilizzabili le informazioni: ad esempio, la seconda tupla, con il solo voto e due valori nulli non fornisce alcuna informazione utile. Infine, la presenza di molteplici valori nulli in una relazione può addirittura generare dubbi sull'effettiva significatività e identità delle tuple: le ultime due tuple della relazione CORSI possono essere diverse o addirittura coincidere! È evidente quindi come sia necessario controllare opportunamente la presenza dei valori nulli nelle nostre relazioni: solo alcune configurazioni debbono essere ammesse. In genere, quando si definisce una relazione, è possibile specificare che i nulli sono ammessi solo su alcuni attributi e non su altri. Alla fine del prossimo paragrafo vedremo quale possa essere un criterio fondamentale per individuare alcuni attributi su cui è essenziale evitare la presenza di valori nulli.

## 2.6 Vincoli di integrità

Le strutture del modello relazionale ci permettono di organizzare le informazioni di interesse per le nostre applicazioni. In molti casi, però, non è vero che qualsiasi insieme di tuple sullo schema rappresenti informazioni corrette per l'applicazione. Abbiamo già brevemente discusso il problema relativamente alla presenza di valori nulli. Ora, approfondiamo il problema con riferimento anche a relazioni prive di valori nulli. Consideriamo ad esempio la base di dati nella figura 2.14 e notiamo in essa varie situazioni che in pratica non si dovrebbero presentare.

- Nella prima tupla della relazione ESAMI abbiamo un voto pari a 36 che, nel sistema italiano, non è ammissibile, in quanto i voti debbono essere compresi fra 0 e 30 (o, con riferimento al superamento dell'esame, fra 18 e 30).

STUDENTI

Matricola	Cognome	Nome	Data di nascita
200768	Verdi	Fabio	12/02/1792
937653	Rossi	Luca	10/10/1971
937653	Bruni	Mario	01/12/1971

ESAMI

Studente	Voto	Lode	Corso
200768	36		05
937653	28	lode	01
937653	30	lode	04
276545	25		01

CORSI

Codice	Titolo	Docente
01	Analisi	Giani
03	Chimica	Melli
04	Chimica	Belli

Figura 2.14 Una base di dati con informazioni scorrette

- Nella seconda tupla ancora della relazione ESAMI viene indicato che è stata attribuita la lode in un esame in cui il voto è 28, il che è impossibile: la lode può essere attribuita solo se il voto è 30.
- Le ultime due tuple della relazione STUDENTI contengono informazioni su due studenti diversi con lo stesso numero di matricola: ancora una situazione impossibile, in quanto il numero di matricola serve appunto ad identificare univocamente gli studenti.
- La quarta tupla della relazione ESAMI presenta, per l'attributo *Studente*, un valore che non compare fra i numeri di matricola nella relazione STUDENTI: anche questa è una situazione indesiderabile, in quanto i numeri di matricola ci forniscono informazioni solo come tramite, verso le corrispondenti tuple della relazione STUDENTI. Analogamente, la prima ennupla presenta un codice di corso che non compare nella relazione CORSI.

In una base di dati, è opportuno evitare situazioni come quelle appena descritte. Allo scopo, è stato introdotto il concetto di *vincolo di integrità*, come proprietà che deve essere soddisfatta dalle istanze che rappresentano informazioni corrette per l'applicazione. Ogni vincolo può essere visto come un *predicato* che associa ad ogni istanza il valore *vero* o *falso*. Se il predicato assume il valore *vero* diciamo che l'istanza *soddisfa* il vincolo. In generale, ad uno schema di base di dati associamo un insieme di vincoli e consideriamo *corrette* (o *lecite*, o *ammisibili*) le istanze che soddisfano tutti i vincoli. In ciascuno dei quattro casi sopra discussi potrebbe essere introdotto un vincolo che vieti la situazione indesiderata.

È possibile classificare i vincoli a seconda degli elementi di una base di dati che ne sono coinvolti. Distinguiamo due categorie, la prima delle quali ha alcuni casi particolari:

- Un vincolo è *intrarelazionale* se il suo soddisfacimento è definito rispetto a singole relazioni della base di dati; i primi tre casi sopra discussi corrispondono a vincoli intrarelazionali; talvolta, il coinvolgimento riguarda le tuple (o addirittura i valori) separatamente le une dalle altre:
  - un *vincolo di tupla* è un vincolo che può essere valutato su ciascuna tupla indipendentemente dalle altre: i vincoli relativi ai primi due casi rientrano in questa categoria;
  - come caso ancora più specifico, un vincolo definito con riferimento a singoli valori (è il caso del primo esempio, in cui sono ammessi solo valori dell'attributo *Voto* compresi fra 18 e 30) viene detto *vincolo su valori* o *vincolo di dominio*, in quanto impone una restrizione sul dominio dell'attributo.
- Un vincolo è *interrelazionale* se coinvolge più relazioni; è questo il caso del quarto esempio, in cui la situazione indesiderata può essere vietata richiedendo che un numero di matricola compaia nella relazione ESAMI solo se compare nella relazione STUDENTI.

Nei paragrafi seguenti esaminiamo con un certo dettaglio le tre categorie più importanti:

- una classe interessante di vincoli di tupla;
- i vincoli di chiave, che sono i più importanti vincoli intrarelazionali;
- i vincoli di integrità referenziale, che sono i vincoli interrelazionali di maggiore interesse.

## 2.7 Vincoli di tupla

Come abbiamo detto, i vincoli di tupla esprimono condizioni sui valori di ciascuna tupla, indipendentemente dalle altre tuple.

Una possibile sintassi per esprimere vincoli di questo tipo è quella che permette di definire espressioni booleane (cioè con connettivi AND, OR e NOT) con atomi che confrontano (con gli operatori di uguaglianza, disuguaglianza e ordinamento) valori di attributo o espressioni aritmetiche su valori di attributo. I vincoli violati individuati nei primi due esempi della lista del paragrafo precedente potrebbero essere descritti con le seguenti espressioni:

$$(\text{Voto} \geq 18) \text{ AND } (\text{Voto} \leq 30)$$

$$(\text{NOT } (\text{Lode} = \text{'lode'})) \text{ OR } (\text{Voto} = 30)$$

In particolare, il secondo vincolo indica che è ammissibile la lode solo se il voto è pari a 30 (dicendo che o non c'è la lode, oppure il voto è pari a 30). Il primo vincolo è in effetti un vincolo di dominio in quanto coinvolge un solo attributo.

La definizione che abbiamo dato ammette anche espressioni più complesse, purché definite sui valori delle singole tuple. Ad esempio, su una relazione sullo schema:

PAGAMENTI(Data,Importo,Ritenute,Netto)

è possibile definire il vincolo che impone, come naturale, che il netto sia pari alla differenza fra l'importo originario e le ritenute, nel modo seguente:

$$\text{Netto} = \text{Importo} - \text{Ritenute}$$

## 2.8 Chiavi

In questo paragrafo discutiamo i vincoli di chiave, che sono senz'altro i più importanti del modello relazionale; potremmo addirittura affermare che senza di essi il modello stesso non avrebbe senso. Cominciamo con un esempio. Nella relazione nella figura 2.15 possiamo notare che i valori delle varie tuple sull'attributo Matricola sono tutti diversi l'uno dall'altro: come già notato più volte, ribadiamo che il valore della matricola *identifica univocamente* gli studenti; a suo tempo il concetto stesso di numero di matricola fu introdotto proprio per avere uno strumento semplice ed efficace per far riferimento agli studenti in modo non ambiguo. Analogamente, possiamo notare che nella relazione non vi sono coppie di tuple con gli stessi valori su ciascuno dei tre attributi Cognome, Nome e DataNascita:

anche i dati anagrafici identificano univocamente le persone.<sup>5</sup> Anche altri insiemi di attributi identificano univocamente le tuple della relazione nella figura 2.15: ad esempio, *Matricola* e *Corso*, come è in effetti ovvio, visto che *Matricola* è da solo sufficiente.

Matricola	Cognome	Nome	DataNascita	Corso
4328	Rossi	Luigi	29/04/59	Informatica
6328	Rossi	Dario	29/04/59	Informatica
4766	Rossi	Luca	01/05/61	Civile
4856	Neri	Luca	01/05/61	Meccanica
5536	Neri	Luca	05/03/58	Meccanica

Figura 2.15 Una relazione per la discussione delle chiavi

Intuitivamente, una chiave è un insieme di attributi utilizzato per identificare univocamente le tuple di una relazione. Per formalizzare la definizione, procediamo in due passi:

- un insieme  $K$  di attributi è *superchiave* per una relazione  $r$  se  $r$  non contiene due tuple distinte  $t_1$  e  $t_2$  con  $t_1[K] = t_2[K]$
- un insieme  $K$  di attributi è *chiave* per  $r$  se è una superchiave minimale (cioè non esiste un'altra superchiave  $K'$  di  $r$  che sia contenuta in  $K$  come sottoinsieme proprio).

Nell'esempio

- l'insieme  $\{\text{Matricola}\}$  è superchiave; è anche una superchiave minimale, in quanto contiene un solo attributo (l'insieme vuoto non è in grado di identificare tuple), e quindi  $\{\text{Matricola}\}$  è una chiave.
- l'insieme  $\{\text{Cognome}, \text{Nome}, \text{Nascita}\}$  è superchiave; inoltre, nessuno dei suoi sottoinsiemi è superchiave: infatti esistono due tuple (la prima e la seconda) uguali su *Cognome* e *Nascita*, due (le ultime) uguali su *Cognome* e *Nome* e due (la quarta e la quinta) uguali su *Cognome* e *Nascita*; quindi  $\{\text{Cognome}, \text{Nome}, \text{Nascita}\}$  è un'altra chiave;
- l'insieme  $\{\text{Matricola}, \text{Corso}\}$  è superchiave, come già discusso; peraltro esso non è una superchiave minimale, perché un suo sottoinsieme proprio,  $\{\text{Matricola}\}$  è esso stesso superchiave minimale e quindi  $\{\text{Matricola}, \text{Corso}\}$  non è una chiave.
- l'insieme  $\{\text{Nome}, \text{Corso}\}$  non è superchiave, perché nella relazione compaiono due tuple, le ultime due, fra loro uguali sia su *Nome* sia su *Corso*.

<sup>5</sup>Per non appesantire l'esempio, assumiamo qui che, nella realtà, cognome, nome e data di nascita siano sufficienti ad identificare univocamente le persone, il che in generale non è vero.

Per approfondire la discussione, esaminiamo un'altra relazione, quella nella figura 2.16. Essa non contiene tuple fra loro uguali sia su *Cognome* sia su *Corso*. Quindi, per questa relazione, l'insieme {*Cognome*, *Corso*} è superchiave. Poiché vi sono tuple uguali su *Cognome* (le prime due) e su *Corso* (la seconda e la quarta), tale insieme è superchiave minimale e cioè chiave. Ora, in questa relazione, *Cognome* e *Corso* identificano univocamente le tuple; ma possiamo dire che questo è vero in generale? Certamente no, in quanto possono benissimo esistere studenti con uguale cognome iscritti allo stesso corso di laurea.

Matricola	Cognome	Nome	Nascita	Corso
6328	Rossi	Dario	29/04/59	Informatica
4766	Rossi	Luca	01/05/61	Civile
4856	Neri	Luca	01/05/61	Meccanica
5536	Neri	Luca	05/03/58	Civile

Figura 2.16 Un'altra relazione per la discussione delle chiavi

In un certo senso, possiamo quindi dire che {*Cognome*, *Corso*} è "casualmente" una chiave per la relazione nella figura 2.16, mentre a noi interessano le chiavi corrispondenti a vincoli di integrità, soddisfatti da tutte le relazioni lecite su un certo schema. In effetti, definendo uno schema, noi associamo ad esso i vincoli di interesse, corrispondenti a proprietà del mondo reale le cui informazioni vengono rappresentate per mezzo della nostra base di dati. I vincoli sono cioè definiti a livello di schema, con riferimento a tutte le istanze: consideriamo corrette solo le istanze che soddisfano tutti i vincoli. Un'istanza corretta può poi soddisfare altri vincoli oltre a quelli definiti sullo schema. Ad esempio, ad uno schema:

STUDENTI(*Matricola*, *Cognome*, *Nome*, *Nascita*, *Corso*)

vanno associati i vincoli che impongono come chiavi i due insiemi di attributi sopra discussi:

{*Matricola*}  
{*Cognome*, *Nome*, *Nascita*}

Entrambe le relazioni nelle figure 2.15 e 2.16 soddisfano tutti e due i vincoli; la seconda soddisfa anche ("per caso," come abbiamo detto) il vincolo secondo cui {*Cognome*, *Corso*} è un'altra chiave.

Possiamo ora fare alcune riflessioni sul concetto di chiave, che giustificano l'importanza ad esso attribuita. In primo luogo, possiamo notare come ciascuna relazione e ciascuno schema di relazione abbiano sempre una chiave. Una relazione è un insieme, e quindi, come più volte ribadito, è costituita di elementi fra loro diversi; di conseguenza, per ogni relazione  $r(X)$ , l'insieme  $X$  di tutti gli attributi su cui è definita è senz'altro una superchiave per essa. Ora i casi sono due: o tale insieme è anche chiave, nel qual caso confermiamo l'esistenza della chiave stessa, oppure non è chiave, perché esiste un'altra superchiave in esso contenuta; allora possiamo procedere ricorsivamente, ripetendo il ragionamento su quest'ultimo

insieme, e così via; poiché l'insieme di attributi su cui è definita una relazione è finito, il processo termina in un numero finito di passi con una superchiave minimale. Quindi, possiamo affermare con certezza che ogni relazione ha una chiave. Lo stesso ragionamento può essere svolto a livello di schema di relazione: l'insieme di tutti gli attributi è superchiave per ciascuna relazione, e quindi lo è per ciascuna relazione lecita; la ricerca di superchiavi minimali procede poi come sopra.

Il fatto che su ciascuno schema di relazione possa essere definita almeno una chiave garantisce la accessibilità a tutti i valori di una base di dati e la loro univoca identificabilità. Inoltre, permette di stabilire efficacemente quelle corrispondenze fra dati contenuti in relazioni diverse che caratterizzano il modello relazionale come "modello basato su valori." Riconsideriamo l'esempio nella figura 2.6. Nella relazione ESAMI, si fa riferimento agli studenti attraverso i numeri di matricola e ai corsi attraverso i relativi codici: in effetti, *Matricola* è la chiave della relazione STUDENTI e *Codice* è la chiave della relazione CORSI. I valori attraverso cui vengono stabilite le corrispondenze fra tuple di relazioni diverse sono valori delle chiavi delle relazioni a cui si fa riferimento dall'esterno.

## 2.9 Chiavi e valori nulli

Possiamo ora riprendere il discorso avviato alla fine del paragrafo 2.5 sulla necessità di evitare la proliferazione di valori nulli nelle nostre relazioni. In particolare, notiamo come, in presenza di valori nulli, non è più vero che, come abbiamo invece affermato più volte, i valori delle chiavi permettono di identificare univocamente le tuple delle relazioni e di stabilire riferimenti fra tuple di relazioni diverse. Esaminando allo scopo la relazione nella figura 2.17, definita sullo stesso schema della relazione nella figura 2.16 (e quindi con due chiavi, una composta dal solo attributo *Matricola* e l'altra dagli attributi *Cognome*, *Nome* e *Nascita*), notiamo problemi di due tipi. La prima tupla ha valori nulli su *Matricola* e *Nascita* e quindi su almeno un attributo di ciascuna chiave: questa tupla non è identificabile in alcun modo; se vogliamo inserire nella base di dati un'altra tupla relativa ad uno studente di nome Mario Rossi, non possiamo sapere se ci stiamo in effetti riferendo allo stesso studente oppure ad un altro. Inoltre, non è possibile, in altre relazioni della base di dati, fare riferimento a questa tupla, visto che ciò andrebbe fatto attraverso il valore di una chiave. Anche le ultime due tuple nella figura presentano un problema: nonostante ciascuna abbia una chiave completamente specificata (grazie al valore su *matricola* nella terza tupla e ai valori su *Cognome*, *Nome* e *Nascita* nell'ultima), la presenza di valori nulli rende impossibile capire se le due tuple facciano riferimento allo stesso studente oppure no.

L'esempio ci suggerisce quindi la necessità di limitare la presenza di valori nulli nelle chiavi delle relazioni. In pratica, si adotta una soluzione semplice, che permette di garantire l'identificazione univoca di tutte le tuple e la possibilità di far riferimento ad esse da parte di altre relazioni: su una delle chiavi (detta la *chiave primaria*) si vieta la presenza di valori nulli; sulle altre, i valori nulli (salvo necessità specifiche) sono ammessi. Gli attributi che costituiscono la chiave primaria vengono spesso evidenziati attraverso la sottolineatura, come mostrato nella figura 2.18. La maggior parte dei riferimenti tra relazioni vengono realizzati attraverso i valori della chiave primaria.

Matricola	Cognome	Nome	Nascita	Corso
NULL	Rossi	Mario	NULL	Informatica
4766	Rossi	Luca	01/05/61	Civile
4856	Neri	Luca	NULL	NULL
NULL	Neri	Luca	05/03/58	Civile

Figura 2.17 Una relazione con valori nulli su tutte le chiavi

È opportuno notare che in quasi tutti i casi reali è possibile trovare attributi i cui valori sono identificanti e sempre disponibili. Quando ciò non accada, è necessario introdurre un attributo aggiuntivo, un codice, probabilmente non significativo dal punto di vista dell'applicazione, che viene in qualche modo generato ed attribuito a ciascuna tupla all'atto dell'inserimento. Tra l'altro, si può dire che molti codici identificativi (quali ad esempio il numero di matricola, il codice fiscale, il numero di targa) siano stati introdotti in passato, prima dell'invenzione o della diffusione delle basi di dati, proprio per garantire l'identificazione univoca dei soggetti di un dominio (rispettivamente gli studenti, i contribuenti, le automobili) e per favorire il riferimento ad essi—esattamente gli obiettivi delle chiavi.

<u>Matricola</u>	Cognome	Nome	Nascita	Corso
3976	Rossi	Mario	NULL	Informatica
4766	Rossi	Luca	01/05/61	Civile
4856	Neri	Luca	NULL	NULL
5591	Neri	Luca	05/03/58	Civile

Figura 2.18 Una relazione con chiave primaria

## 2.10 Vincoli di integrità referenziale

Per discutere la più importante classe di vincoli interrelazionali, consideriamo la base di dati in figura 2.19. In essa, la prima relazione contiene informazioni relative ad un insieme di infrazioni al codice della strada, la seconda agli agenti di polizia che le hanno rilevate e la terza ad un insieme di autoveicoli.<sup>6</sup> Le informazioni della relazione INFRAZIONI sono rese significative e complete attraverso il riferimento alle altre due relazioni: alla relazione AGENTI, per il tramite dell'attributo Agente, che contiene numeri di matricola di agenti corrispondenti alla chiave primaria della

<sup>6</sup>Utilizziamo la vecchia struttura delle targhe automobilistiche, con la sigla della provincia, perché ci permette di svolgere considerazioni interessanti sulle chiavi composte da più attributi.

relazione AGENTI, e alla relazione AUTO per mezzo degli attributi Prov e Targa, che contengono valori degli omonimi attributi che formano la chiave primaria della relazione AUTO. I riferimenti sono significativi in quanto i valori nella relazione INFRAZIONI sono uguali a valori effettivamente presenti nelle altre due: se un valore di Agente in INFRAZIONI non compare come valore della chiave di AGENTI, allora il riferimento non è efficace. Nell'esempio, tutti i riferimenti sono in effetti utilizzabili.

INFRAZIONI	Codice	Data	Agente	Art	Prov	Numero
	143256	25-10-92	567	44	RM	4E5432
	987554	26-10-92	456	34	RM	4E5432
	987554	26-10-92	456	34	RM	2F7643
	630876	15-10-92	456	53	MI	2F7643
	539856	12-10-92	567	44	MI	2F7643

AGENTI	Matricola	CF	Cognome	Nome
	567	RSSM...	Rossi	Mario
	456	NREL...	Neri	Luigi
	638	NREP...	Neri	Piero

AUTO	Prov	Numero	Proprietario	Indirizzo
	RM	2F7643	Verdi Piero	Via Tigli
	RM	1A2396	Verdi Piero	Via Tigli
	RM	4E5432	Bini Luca	Via Aceri
	MI	2F7643	Luci Gino	Via Aceri

Figura 2.19 Una base di dati con vincoli di integrità referenziale

Un vincolo di integrità referenziale (chiamato nella letteratura in inglese *foreign key* o *referential integrity constraint*) fra un insieme di attributi  $X$  di una relazione  $R_1$  e un'altra relazione  $R_2$  è soddisfatto se i valori su  $X$  di ciascuna tupla dell'istanza di  $R_1$  compaiono come valori della chiave (primaria) dell'istanza di  $R_2$ . La definizione precisa richiede un po' di attenzione, in particolare nel caso in cui la chiave della relazione riferita sia composta di più attributi e nel caso in cui vi siano più chiavi. Procediamo gradualmente, vedendo prima il caso in cui la chiave di  $R_2$  è unica e composta di un solo attributo  $B$  (e quindi l'insieme  $X$  è a sua volta costituito da un solo attributo  $A$ ): allora il vincolo di integrità referenziale fra l'attributo  $A$  di  $R_1$  e la relazione  $R_2$  è soddisfatto se, per ogni tupla  $t_1$  in  $R_1$ , esiste una tupla  $t_2$  in  $R_2$  tale che  $t_1[A] = t_2[B]$ . Nel caso più generale, dobbiamo fare attenzione al fatto che ciascuno degli attributi in  $X$  deve corrispondere ad un preciso attributo della chiave primaria  $K$  di  $R_2$ . Allo scopo, è necessario specificare un ordinamento sia nell'insieme  $X$  sia in  $K$ . Indicando gli attributi in ordine,  $X = A_1 A_2 \dots A_p$  e  $K = B_1 B_2 \dots B_p$ , il vincolo è soddisfatto se per ogni tupla  $t_1$

in  $R_1$  esiste una tupla  $t_1$  in  $R_1$  con  $t_1[A_i] = t_2[A_i]$ , per ogni  $i$  compreso fra 1 e  $n$ .

Sullo schema della base di dati nella figura 2.19 ha senso definire i vincoli di integrità referenziale:

- fra l'attributo Agente della relazione INFRAZIONI e la relazione AGENTI
- fra gli attributi Prov e Numero di INFRAZIONI e la relazione AUTO, in cui l'ordine degli attributi nella chiave preveda prima Prov e poi Numero.

La base di dati nella figura 2.19 soddisfa entrambi i vincoli, mentre la base di dati nella figura 2.20 li viola entrambi: il primo perché AGENTI non contiene nessuna tupla con valore di Matricola pari a 456, e il secondo perché AUTO non contiene nessuna tupla con valore 'RM' su Prov e 2F7643 su Numero (notiamo che esistono una tupla con valore 'RM' su Prov e un'altra con valore 2F7643 su Numero, ma questo non è sufficiente, perché è richiesto che ci sia una tupla con entrambi i valori: solo in questo modo, infatti, i due valori possono far riferimento ad una tupla della relazione AUTO).

Codice	Data	Agente	Art	Prov	Numero
987554	26-10-92	456	34	RM	2F7643
630876	15-10-92	456	53	FI	4E5432

Matricola	CF	Cognome	Nome
567	RSSM...	Rossi	Mario
638	NREP...	Neri	Piero

Prov	Numero	Proprietario	Indirizzo
RM	1A2396	Verdi Piero	Via Tigli
FI	4E5432	Bini Luca	Via Aceri
MI	2F7643	Luci Gino	Via Noci

Figura 2.20 Una base di dati che viola vincoli di integrità referenziale

Relativamente al secondo vincolo, notiamo come il ragionamento sull'ordine degli attributi possa apparire pesante, visto che la corrispondenza può, almeno in questo caso, essere realizzata per mezzo dei nomi degli attributi stessi. In generale, però, questo può non accadere, e quindi l'ordinamento è essenziale. Consideriamo ad esempio una base di dati contenente informazioni sui veicoli coinvolti in incidenti stradali. In particolare, supponiamo di voler includere in una relazione, insieme ad altre informazioni, i numeri di targa dei due veicoli coinvolti.<sup>7</sup> Ovviamente, dovremo avere due coppie di attributi, che non potranno avere gli stessi nomi. Ad esempio, lo schema potrebbe essere:

<sup>7</sup>Supponiamo per semplicità che i veicoli coinvolti siano sempre e solo due.

INCIDENTI(Codice, Prov1, Numero1, Prov2, Numero2, ...)

In questo caso, non sarà ovviamente possibile stabilire la corrispondenza nel vincolo di integrità referenziale verso la relazione AUTO per mezzo dei nomi degli attributi, in quanto essi sono diversi da quelli della chiave primaria di AUTO. Solo attraverso l'ordinamento diventa possibile specificare che il riferimento associa Prov1 (attributo di INCIDENTI) a Prov (attributo nella chiave di AUTO) e Numero1 a Numero e, analogamente, Prov2 a Prov e Numero2 a Numero. La base di dati in figura 2.21 soddisfa i due vincoli in questione, mentre quella in figura 2.22 soddisfa quello relativo a Prov1 e Numero1 e viola l'altro, perché nella relazione AUTO non c'è nessun veicolo targato FI T39275.

INCIDENTI	<u>Codice</u>	Prov1	Numero1	Prov2	Numero2	...
	6207	RM	2F7643	FI	T39275	...
	6974	FI	4E5432	FI	T39275	...

AUTO	<u>Prov</u>	<u>Numero</u>	Proprietario	Indirizzo
	RM	1A2396	Verdi Piero	Via Tigli
	FI	4E5432	Bini Luca	Via Aceri
	FI	T39275	Bitti Piero	Via Pini
	RM	2F7643	Luci Gino	Via Noci

Figura 2.21 Una base di dati con due vincoli di integrità referenziale simili

INCIDENTI	<u>Codice</u>	Prov1	Numero1	Prov2	Numero2	...
	6207	RM	2F7643	FI	T39275	...
	6974	FI	4E5432	FI	T39275	...

AUTO	<u>Prov</u>	<u>Numero</u>	Proprietario	Indirizzo
	RM	1A2396	Verdi Piero	Via Tigli
	FI	4E5432	Bini Luca	Via Aceri
	RM	2F7643	Luci Gino	Via Noci

Figura 2.22 Una base di dati che viola un vincolo di integrità referenziale

Un'ultima considerazione può essere utile riguardo a relazioni con più chiavi. In questo caso, come abbiamo detto, è opportuno che una delle chiavi sia evidenziata come primaria, ed è ragionevole che i riferimenti siano diretti verso di essa: per questo motivo, nella specifica dei vincoli di integrità referenziale, abbiamo potuto

omettere la citazione esplicita degli attributi che compongono la chiave primaria. Peraltro, va notato che non tutti i sistemi di gestione di basi di dati oggi sul mercato permettono di indicare esplicitamente la chiave primaria: alcuni permettono anche di specificare più chiavi, ma non di evidenziarne una come primaria. In questi casi, il vincolo di integrità referenziale deve indicare esplicitamente gli attributi che compongono la chiave cui si fa riferimento. Ad esempio, consideriamo in una base di dati sullo schema

IMPIEGATI(Matricola, Cognome, Nome, Dipartimento)  
DIPARTIMENTI(Codice, Nome, Sede)

in cui la relazione DIPARTIMENTI sia identificata dall'attributo Codice e, separatamente, dall'attributo Nome (non esistono due dipartimenti con lo stesso codice e non esistono due dipartimenti con lo stesso nome). Metodologicamente, abbiamo detto che è opportuno che una delle due chiavi, ad esempio Codice, sia individuata come primaria ed utilizzata per stabilire i riferimenti. Se però il sistema non prevede il concetto di chiave primaria, il vincolo deve essere espresso indicando esplicitamente gli attributi; dovremo quindi dire che esiste un vincolo di integrità referenziale

- fra l'attributo Dipartimento della relazione IMPIEGATI e la chiave Codice della relazione DIPARTIMENTI.

Questo è il motivo per cui, come vedremo nel capitolo 4, nei sistemi relazionali è prevista una specifica più articolata per i vincoli di integrità referenziale.

## 2.11 Conclusioni

Abbiamo visto in questo capitolo la definizione delle strutture e dei vincoli del modello relazionale. Abbiamo in primo luogo discusso il concetto di relazione, con gli adattamenti necessari per meglio sfruttare i concetti della teoria degli insiemi nel contesto delle basi di dati. Poi, abbiamo mostrato come le relazioni possano essere utilizzate per organizzare insiemi di dati anche complessi, utilizzando i dati stessi per realizzare riferimenti fra le diverse componenti (senza alcun utilizzo di puntatori espliciti). Quindi, dopo aver brevemente accennato alla necessità di utilizzare valori nulli per denotare l'assenza di informazioni, abbiamo discusso il concetto di vincolo di integrità, attraverso tre classi fondamentali: i vincoli di tupla, le chiavi e i vincoli di integrità referenziale.

Nei prossimi due capitoli completeremo la presentazione del modello relazionale da due punti di vista:

- nel capitolo 3 illustreremo i principi su cui si basano le operazioni di interrogazione di una base di dati.
- nel capitolo 4 mostreremo come tutti i concetti, quelli relativi alle strutture e ai vincoli, discussi in questo capitolo, e quelli relativi alle operazioni (capitolo 3), siano effettivamente realizzati nei DBMS reali, attraverso il linguaggio SQL.

## Note bibliografiche

Presentazioni del modello relazionale tipo quella di questo capitolo sono reperibili su tutti i moderni testi sulle basi di dati, a cominciare da quelli di Elmasri e Navathe [30] e Korth e Silberchatz [43]. Formalizzazioni ulteriori e approfondimenti teorici (molto più dettagliati di quelli che presenteremo nei prossimi capitoli) si trovano nei libri di teoria delle basi di dati, in italiano quello di Atzeni, Batini e De Antonellis [4] e in inglese quelli di Ullman [60], Maier [47], Atzeni e De Antonellis [5], Abiteboul, Hull e Vianu [1]. Può essere interessante consultare l'articolo di Codd [26] che contiene la proposta originaria del modello, per constatarne tuttora l'attualità.

## Esercizi

**Esercizio 2.1** Considerare le informazioni per la gestione dei prestiti di una biblioteca personale. Il proprietario presta libri ai suoi amici, che indica semplicemente attraverso i rispettivi nomi o soprannomi (così da evitare omonimie) e fa riferimento ai libri attraverso i titoli (non possiede due libri con lo stesso titolo). Quando presta un libro, prende nota della data prevista di restituzione. Definire uno schema di relazione per rappresentare queste informazioni, individuando opportuni domini per i vari attributi e mostrarne un'istanza in forma tabellare. Indicare la chiave (o le chiavi) della relazione.

**Esercizio 2.2** Rappresentare per mezzo di una o più relazioni le informazioni contenute nell'orario delle partenze da una stazione ferroviaria: numero, orario, destinazione finale, categoria, fermate intermedie, di ciascuno dei treni in partenza.

**Esercizio 2.3** Definire uno schema di base di dati per organizzare le informazioni di un'azienda che ha impiegati (ognuno con codice fiscale, cognome, nome e data di nascita), filiali (con codice, sede e direttore, che è un impiegato). Ogni impiegato lavora presso una filiale. Indicare le chiavi e i vincoli di integrità referenziale dello schema. Mostrare un'istanza della base di dati e verificare che soddisfa i vincoli.

**Esercizio 2.4** Un albero genealogico rappresenta, in forma grafica, la struttura di una famiglia (o più famiglie, quando è ben articolato). Mostrare come si possano rappresentare, per mezzo di una base di dati relazionale le informazioni di un albero genealogico, cominciando eventualmente da una struttura semplificata, in cui si rappresentano solo le discendenze in linea maschile (cioè i figli vengono rappresentati solo per i componenti di sesso maschile) oppure solo quelle in linea femminile.

**Esercizio 2.5** Per ciascuno degli esercizi 2.1-2.4, valutare le eventuali esigenze di rappresentazione di valori nulli, con i benefici e le difficoltà connesse.

**Esercizio 2.6** Descrivere in linguaggio naturale le informazioni organizzate nella base di dati in figura 2.23.

PAZIENTI

Cod	Cognome	Nome
A102	Necchi	Luca
B372	Rossini	Piero
B543	Missoni	Nadia
B444	Missoni	Luigi
S555	Rossetti	Gino

RICOVERI

Paziente	Inizio	Fine	Reparto
A102	2/05/94	9/05/94	A
A102	2/12/94	2/01/95	A
S555	5/10/94	3/12/94	B
B444	1/12/94	2/01/95	B
S555	5/10/94	1/11/94	A

MEDICI

Matr	Cognome	Nome	Reparto
203	Neri	Piero	A
574	Bisi	Mario	B
461	Bargio	Sergio	B
530	Belli	Nicola	C
405	Mizzi	Nicola	A
501	Monti	Mario	A

REPARTI

Cod	Nome	Primario
A	Chirurgia	203
B	Pediatria	574
C	Medicina	530

Figura 2.23 Una base di dati per l'esercizio 2.6

**Esercizio 2.7** Individuare le chiavi e i vincoli di integrità referenziale che sussistono nella base di dati di figura 2.23 e che è ragionevole assumere che siano soddisfatti da tutte le basi di dati sullo stesso schema. Individuare anche gli attributi sui quali possa essere sensato ammettere valori nulli.

**Esercizio 2.8** Definire uno schema di base di dati che organizzi i dati necessari a generare la pagina dei programmi radiofonici di un quotidiano, con stazioni, ore e titoli dei programmi; per ogni stazione sono memorizzati, oltre al nome, anche la frequenza di trasmissione e la sede.