Capitolo 2

Modello relazionale

Il modello relazionale, sebbene non sia stato il modello dei dati usato nei primi DBMS, ha rivoluzionato, sin dalla sua definizione nel 1970 ad opera di Codd, il mondo delle basi di dati ed è rapidamente divenuto il modello dei dati più diffuso, oggi comunemente adottato dalla larga maggioranza dei DBMS disponibili a livello commerciale. Il modello relazionale è basato su una semplice struttura dati – la *relazione* – ed è caratterizzato da precise basi matematiche, avendo come fondamento teorico la teoria degli insiemi e la logica dei predicati del primo ordine. I principali vantaggi del modello relazionale rispetto ai modelli dei dati di precedente definizione possono essere individuati nella semplice rappresentazione dei dati e nella facilità con cui possono essere espresse interrogazioni anche complesse. La semplicità del modello relazionale, infatti, ha reso possibile lo sviluppo di linguaggi *dichiarativi* e semplici da usare per specificare ricerche sui dati, che consentono l'accesso ai dati da parte di utenti ed applicazioni anche in base a modalità non previste a priori. Le interrogazioni sulle relazioni possono essere espresse in due formalismi di base:

- **algebra relazionale**, in cui le interrogazioni sono espresse applicando operatori specializzati alle relazioni;
- **calcolo relazionale**, in cui le interrogazioni sono espresse per mezzo di formule logiche che devono essere verificate dalle tuple ottenute come risposta all'interrogazione.

Un risultato teorico stabilisce che, sotto determinate assunzioni, i due formalismi hanno lo stesso potere espressivo: ognuno di essi può esprimere qualsiasi interrogazione che l'altro formalismo può esprimere, ma non di più. Questi formalismi, ovviamente, non costituiscono veri linguaggi per basi di dati, in quanto mancano delle operazioni di modifica alle relazioni e di numerose funzionalità utili nell'uso pratico. Le operazioni di modifica e varie funzionalità addizionali saranno illustrate nella trattazione del linguaggio SQL (vedi Capitolo 3), linguaggio standard per basi di dati basate sul modello relazionale, sviluppato sulla base di algebra e calcolo relazionale.

Questo capitolo, oltre ad introdurre il modello dei dati relazionale, illustra gli aspetti fondamentali relativi ai linguaggi di interrogazione per basi di dati relazionali, presentando le caratteristiche principali di algebra e calcolo relazionale.

2.1 Modello dei dati

In questo paragrafo introdurremo le principali nozioni alla base del modello dei dati relazionale.

2.1.1 Relazioni

Il concetto alla base del modello relazionale è la *relazione*. Per definire la nozione di relazione è necessario partire dalla nozione di *dominio*. Un dominio è un insieme (anche infinito) di valori.

Esempio 2.1 L'insieme dei numeri interi è un dominio. L'insieme delle stringhe di caratteri è un dominio. L'insieme {0,1} è un dominio. □

Nel seguito indicheremo con $\mathcal{D}$ l'insieme di tutti i domini, che assumeremo contenere i numeri interi (`int`), i numeri reali (`real`), le stringhe (`string`) e le date (`date`). I valori dei domini costituiscono i valori atomici che popoleranno la base di dati, a partire dai quali vengono costruite le *tuple* delle relazioni, effettuando un prodotto cartesiano di tali valori.

Definizione 2.1 (Prodotto cartesiano) *Siano $D_1, D_2, \ldots, D_k \in \mathcal{D}$ k domini. Il prodotto cartesiano di tali domini, indicato con $D_1 \times D_2 \times \ldots \times D_k$, è definito come l'insieme $\{(v_1, v_2, \ldots, v_k) \mid v_1 \in D_1, \ldots, v_k \in D_k\}$.* ◇

Gli elementi appartenenti al prodotto cartesiano sono detti *tuple*. Il prodotto cartesiano di k domini ha *grado* k.

Definizione 2.2 (Relazione) *Siano $D_1, D_2, \ldots, D_k \in \mathcal{D}$ domini. Una relazione su $D_1, D_2, \ldots, D_k$ è un sottoinsieme finito del prodotto cartesiano $D_1 \times D_2 \times \ldots \times D_k$.* ◇

Una relazione, sottoinsieme del prodotto cartesiano di k domini, ha grado k. Ogni tupla di una relazione di grado k ha pertanto k componenti, una per ogni dominio su cui è definita la relazione cui la tupla appartiene. Dati una relazione R di grado k, una tupla $t \in R$ ed un intero $i \in \{1, \ldots, k\}$, la notazione $t[i]$ denota la i-esima componente di t. La *cardinalità* di una relazione è il numero di tuple appartenenti alla relazione. È importante notare che mentre i domini che partecipano ad un prodotto cartesiano possono essere insiemi infiniti (questa è anzi la situazione più comune), una relazione è sempre un insieme finito.

In base alle definizione, una relazione è un insieme di tuple, in quanto tali ordinate al loro interno: l'i-esimo valore di ciascuna tupla proviene dall'i-esimo dominio su cui è definito il prodotto cartesiano; è cioè definito un ordinamento fra i domini su cui è definita la relazione. Viceversa, essendo una relazione un insieme, non è definito alcun ordinamento fra le tuple di una relazione. Inoltre, sempre poiché una relazione è un insieme, le tuple di una relazione sono distinte l'una dall'altra, cioè non vi sono tuple duplicate.

Esempio 2.2 Dati $D_1 = \{0, 1, 2\}$ e $D_2 = \{\mathtt{d}, \mathtt{v}\}$, il prodotto cartesiano $D_1 \times D_2$ ha grado 2 e contiene le seguenti tuple: {(0,d),(0,v),(1,d),(1,v),(2,d),(2,v)}.

Esempi di relazioni, sottoinsiemi del prodotto cartesiano $D_1 \times D_2$, sono: {(0,d),(0,v),(1,d)} e {(1,d),(2,v)}. Le due relazioni hanno, rispettivamente, cardinalità 3 e 2. Data la tupla t = (0,d) appartenente alla prima delle due relazioni precedenti, $t[1] = 0$ e $t[2] = \mathtt{d}$. □

Una formulazione più conveniente del modello relazionale associa un nome, detto *nome di attributo*, ad ogni componente delle tuple in una relazione. La coppia (nome di attributo, dominio) è detta *attributo*. L'uso degli attributi permette di denotare le componenti di ogni tupla *per nome* piuttosto che *per posizione*, come è invece necessario nella formulazione precedente della nozione di relazione. Un altro importante vantaggio nell'uso degli attributi è di fornire maggiori informazioni semantiche sulle proprietà che ogni componente delle tuple in una relazione modella. In nome di una relazione e l'insieme dei suoi attributi ne costituiscono lo *schema*, come specificato dalla seguente definizione.

Definizione 2.3 (Schema di relazione) *Siano R un nome di relazione, $\{A_1, A_2, \ldots, A_n\}$ un insieme di nomi di attributi, $dom : \{A_1, A_2, \ldots, A_n\} \rightarrow \mathcal{D}$ una funzione totale che associa ad ogni nome di attributo in $\{A_1, A_2, \ldots, A_n\}$ il corrispondente dominio. La coppia $(R(A_1, A_2, \ldots, A_n), dom)$ è uno schema di relazione. U_R denota l'insieme dei nomi di attributi di R, cioè $\{A_1, A_2, \ldots, A_n\}$.* ◇

Nel seguito, quando saranno chiare dal nome dell'attributo o non saranno rilevanti per la trattazione, ometteremo le informazioni relative al dominio degli attributi ed indicheremo lo schema di una relazione R semplicemente con $R(A_1, A_2, \ldots, A_n)$. Un insieme di schemi di relazione costituisce uno *schema di base di dati*, come specificato dalla definizione seguente.

Definizione 2.4 (Schema di base di dati) *Siano $S_1, S_2, \ldots, S_n$ schemi di relazioni, con nomi di relazione diversi, $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ è detto schema di base di dati.* ◇

Le definizioni seguenti introducono le nozioni di tupla e di relazione sulla base della precedente definizione di schema di relazione.

Definizione 2.5 (Tupla e relazione) *Sia $(R(A_1, A_2, \ldots, A_n), dom)$ uno schema di relazione. Una tupla t definita su R è un insieme di funzioni totali $f_1, f_2, \ldots, f_n$, dove $f_i : A_i \rightarrow dom(A_i)$, $i = 1, ..., n$, associa all'attributo di nome A_i un valore del dominio di tale attributo. Una relazione definita su uno schema di relazione è un insieme finito di tuple definite su tale schema; tale relazione è anche detta istanza dello schema.* ◇

Dato uno schema di relazione $(R(A_1, A_2, \ldots, A_n), dom)$, una tupla t su tale schema può essere rappresentata tramite la notazione $[A_1 : v_1, A_2 : v_2, \ldots, A_n : v_n]$ dove v_i, $i = 1, \ldots, n$, è un valore appartenente a $dom(A_i)$. La notazione $t[A_i]$,

$i = 1, ..., n$, denota il valore dell'attributo A_i della tupla t (cioè v_i). Analogamente, dato un insieme $A \subseteq U_R$ di attributi, $t[A]$ denota la tupla costituita dalle componenti di t i cui attributi appartengono all'insieme A.

Dalle definizioni precedenti possiamo notare che, dato uno schema di relazione, è in realtà possibile definire più relazioni che siano istanze di tale schema; pertanto può essere utile distinguere tra il nome di relazione usato in uno schema di relazione ed il nome (i nomi) della relazione (relazioni) istanza (istanze) di tale schema. Nell'uso comune, tuttavia, e nel resto della trattazione, il nome di relazione utilizzato nello schema è usato per denotare anche la relazione istanza dello schema. Pertanto ogni schema di relazione ha, ad ogni dato istante, una sola relazione istanza. Ovviamente tale istanza cambia il suo *stato* nel tempo a seguito di modifiche sui dati.

Esempio 2.3 Consideriamo la relazione `Film` della base di dati della videoteca illustrata nella Figura 1.1 (qui riprodotta come Figura 2.1). In tale figura, ed in tutto il testo, le relazioni sono rappresentate come tabelle in cui ad ogni colonna è associato un nome di attributo ed ogni riga corrisponde ad una tupla. Lo schema della relazione `Film` è:

`Film(titolo,regista,anno,genere,valutaz)`
$dom($`titolo`$) = dom($`regista`$) = dom($`genere`$) =$ `string`
$dom($`anno`$) =$ `int`$, dom($`valutaz`$) =$ `real`.

La relazione è un sottoinsieme di: `string` $\times$ `string` $\times$ `int` $\times$ `string` $\times$ `real`. Una tupla appartenente a tale relazione è: `[titolo:'ed wood',regista:'tim burton',anno:1994,genere:'drammatico',valutaz:4.00]`.

Consideriamo ora le relazioni di Figura 2.2, sempre relative alla base di dati della videoteca. Il nome della prima relazione, che modella informazioni sui clienti della videoteca, è `Cliente` e lo schema è:

`Cliente(codCli,nome,cognome,telefono,dataN,residenza)`
$dom($`nome`$) = dom($`cognome`$) = dom($`telefono`$) = dom($`residenza`$) =$ `string`
$dom($`codCli`$) =$ `int`$, dom($`dataN`$) =$ `date`.

Tali attributi rappresentano, rispettivamente, il codice identificativo, il nome, il cognome, il numero di telefono, la data di nascita e l'indirizzo di residenza del cliente. □

2.1.2 Valori nulli

Un aspetto importante nella modellazione dei dati riguarda il fatto che non sempre sono disponibili tutte le informazioni sulle entità del dominio applicativo che vengono rappresentate nella base di dati. In termini del modello relazionale questo vuol dire che alcune tuple possono non avere un valore per un qualche attributo.

Non permettere l'inserzione nella base di dati di queste tuple sarebbe troppo rigido rispetto alle più comuni esigenze applicative. L'approccio adottato è quello di introdurre un valore speciale, detto *valore nullo*, il quale denota la mancanza di un valore. Usare ad esempio 0 per indicare un valore nullo non è una soluzione appropriata in quanto 0 è un valore legale per vari domini (ad esempio quelli numerici) e quindi il suo uso non permetterebbe di distinguere il caso in cui 0 sia effettivamente il valore dell'attributo dal caso in cui 0 indichi il valore nullo. Nella trattazione assumiamo di denotare il valore nullo con il simbolo '`?`' (sono tuttavia possibili altre convenzioni). Il valore nullo è un valore ammissibile per ogni dominio. I linguaggi come SQL permettono comunque di specificare nella definizione di una relazione quali attributi non possono mai assumere valore nullo (vedi Capitolo 3). La presenza di valori nulli introduce interessanti questioni riguardanti l'esecuzione di interrogazioni sui dati. Tali questioni saranno trattate approfonditamente nel Capitolo 3 nel contesto del linguaggio SQL. Nell'esempio seguente, ed in tutto il testo, useremo la convenzione di evidenziare negli schemi con un circoletto gli attributi che possono assumere valori nulli.

Esempio 2.4 Consideriamo la relazione `Noleggio` illustrata nella Figura 2.2, che modella i noleggi effettuati dai clienti della videoteca. Lo schema della relazione `Noleggio` è:

`Noleggio(colloc,dataNol,codCli,dataRest`$_\circ$`)`
$dom($`codCli`$) = dom($`colloc`$) =$ `int`, $dom($`dataNol`$) = dom($`dataRest`$) =$ `date`.

Tali attributi rappresentano, rispettivamente, la collocazione del video noleggiato, la data di inizio noleggio, il codice identificativo del cliente che effettua il noleggio e la data di restituzione del video. Le tuple corrispondenti ai noleggi in corso hanno valore nullo per l'attributo `dataRest`. □

2.1.3 Chiavi

Una *chiave* di una relazione è un insieme di attributi che distingue fra loro le tuple della relazione, come formalizzato dalla seguente definizione.

Definizione 2.6 (Chiave e super-chiave) *Sia $R(A_1, ..., A_n)$ uno schema di relazione. Un insieme $X \subseteq U_R$ di attributi di R è chiave di R se verifica entrambe le seguenti proprietà:*

1. *qualsiasi sia lo stato di R, non esistono due tuple distinte di R che abbiano lo stesso valore per tutti gli attributi in X;*
2. *nessun sottoinsieme proprio[1] di X verifica la proprietà (1).*

[1] Dati due insiemi S ed S', S' è sottoinsieme proprio di S (indicato come $S' \subset S$) se tutti gli elementi di S' appartengono ad S ed esiste almeno un elemento di S che non appartiene ad S'.

Film

titolo	regista	anno	genere	valutaz
underground	emir kusturica	1995	drammatico	3.20
edward mani di forbice	tim burton	1990	fantastico	3.60
nightmare before christmas	tim burton	1993	animazione	4.00
ed wood	tim burton	1994	drammatico	4.00
mars attacks	tim burton	1996	fantascienza	3.00
il mistero di sleepy hollow	tim burton	1999	horror	3.50
big fish	tim burton	2003	fantastico	3.10
la sposa cadavere	tim burton	2005	animazione	3.50
la fabbrica di cioccolato	tim burton	2005	fantastico	4.00
io non ho paura	gabriele salvatores	2003	drammatico	3.50
nirvana	gabriele salvatores	1997	fantascienza	3.00
mediterraneo	gabriele salvatores	1991	commedia	3.80
pulp fiction	quentin tarantino	1994	thriller	3.50
le iene	quentin tarantino	1992	thriller	4.00

Video

colloc	titolo	regista	tipo
1111	underground	emir kusturica	v
1112	underground	emir kusturica	d
1113	big fish	tim burton	v
1114	big fish	tim burton	d
1115	edward mani di forbice	tim burton	d
1116	nightmare before christmas	tim burton	v
1117	nightmare before christmas	tim burton	d
1118	ed wood	tim burton	d
1119	mars attacks	tim burton	d
1120	il mistero di sleepy hollow	tim burton	d
1121	la sposa cadavere	tim burton	d
1122	la fabbrica di cioccolato	tim burton	d
1123	la fabbrica di cioccolato	tim burton	d
1124	io non ho paura	gabriele salvatores	d
1125	nirvana	gabriele salvatores	d
1126	mediterraneo	gabriele salvatores	d
1127	pulp fiction	quentin tarantino	v
1128	pulp fiction	quentin tarantino	d
1129	le iene	quentin tarantino	d

Figura 2.1: Base di dati relativa alla videoteca (1)

Cliente

codCli	nome	cognome	telefono	dataN	residenza
6610	anna	rossi	01055664433	05-Ott-1979	via scribanti 16 16131 genova
6635	paola	bianchi	0104647992	12-Apr-1976	via dodecaneso 35 16146 genova
6642	marco	verdi	3336745383	16-Ott-1972	via lagustena 35 16131 genova

Noleggio

colloc	dataNol	codCli	dataRest
1111	01-Mar-2006	6635	02-Mar-2006
1115	01-Mar-2006	6635	02-Mar-2006
1117	02-Mar-2006	6635	06-Mar-2006
1118	02-Mar-2006	6635	06-Mar-2006
1111	04-Mar-2006	6642	05-Mar-2006
1119	08-Mar-2006	6635	10-Mar-2006
1120	08-Mar-2006	6635	10-Mar-2006
1116	08-Mar-2006	6642	09-Mar-2006
1118	10-Mar-2006	6642	11-Mar-2006
1121	15-Mar-2006	6635	18-Mar-2006
1122	15-Mar-2006	6635	18-Mar-2006
1113	15-Mar-2006	6635	18-Mar-2006
1129	15-Mar-2006	6635	20-Mar-2006
1119	15-Mar-2006	6642	16-Mar-2006
1126	15-Mar-2006	6610	16-Mar-2006
1112	16-Mar-2006	6610	18-Mar-2006
1114	16-Mar-2006	6610	17-Mar-2006
1128	18-Mar-2006	6642	20-Mar-2006
1124	20-Mar-2006	6610	21-Mar-2006
1115	20-Mar-2006	6610	21-Mar-2006
1124	21-Mar-2006	6642	22-Mar-2006
1116	21-Mar-2006	6610	?
1117	21-Mar-2006	6610	?
1127	22-Mar-2006	6635	?
1125	22-Mar-2006	6635	?
1122	22-Mar-2006	6642	?
1113	22-Mar-2006	6642	?

Figura 2.2: Base di dati relativa alla videoteca (2)

Un insieme di attributi che verifica la proprietà (1), ma non la proprietà (2), è detto super-chiave di R. ◇

Una relazione può avere più di un insieme S di attributi che verificano le proprietà *(1)* e *(2)* della precedente definizione. In tal caso viene utilizzato a volte il termine *chiavi candidate* per indicare tutte le chiavi. Notiamo inoltre che una relazione ha sicuramente almeno una chiave. Infatti, poiché una relazione, essendo un insieme di tuple, non contiene tuple duplicate, è sicuramente possibile distinguere una tupla da ogni altra tupla della relazione considerando *tutti* i suoi attributi. L'insieme di attributi in U_R, infatti, soddisfa sempre la proprietà *(1)* precedente.

Nel caso in cui una relazione abbia più chiavi candidate, è possibile selezionare tra queste una *chiave primaria*. Le altre chiavi vengono a volte indicate come *chiavi alternative*. Intuitivamente, come discuteremo nel Paragrafo 2.1.4, possiamo fare riferimento ad una tupla mediante i valori dei campi di una sua chiave. In linea di principio, possiamo usare qualunque chiave, non solo la primaria, per riferirci ad una tupla. Usare la chiave primaria è però preferibile perché è ciò per cui il DBMS ottimizza le operazioni (vedi Capitolo 7). Come discuteremo nel Capitolo 6, è quindi importante selezionare oculatamente la chiave primaria. Un criterio nella scelta della chiave primaria è scegliere tra le chiavi candidate quella che contiene il minor numero di attributi o quella più frequentemente usata nelle interrogazioni. Un insieme di attributi selezionato come chiave primaria deve inoltre verificare la proprietà che nessuno degli attributi possa assumere valore nullo. I linguaggi come SQL permettono comunque di specificare nella definizione di una relazione quali attributi costituiscono la chiave primaria e quali eventuali chiavi alternative (vedi Capitolo 3).

Nell'esempio seguente ed in tutto il testo utilizzeremo la convenzione di sottolineare gli attributi chiave primaria di una relazione. Eventuali altre chiavi candidate verranno indicate in italico, quando la notazione non crea ambiguità, o elencate a parte, altrimenti. Come evidenziato dall'esempio seguente, le chiavi delle relazioni vengono individuate mediante esame del dominio applicativo e dei relativi vincoli.

Esempio 2.5 Consideriamo la base di dati della videoteca costituita dalle relazioni nelle Figure 2.1 e 2.2. Un cliente viene identificato attraverso il codice, quindi una chiave della relazione `Cliente` è l'attributo `codCli`. Se assumiamo che non possano esistere clienti con lo stesso nome e cognome, nati lo stesso giorno, una chiave alternativa per `Cliente` è costituita dai tre attributi `nome`, `cognome` e `dataN`. L'attributo `codCli` è preferibile, perché la chiave risulta costituita da un solo attributo. Assumiamo invece che possano esistere clienti che condividono lo stesso numero di telefono, quindi l'attributo `telefono` non è chiave per `Cliente`.

Una chiave della relazione `Film` è data dagli attributi `(titolo,regista)`. Notiamo, infatti, che possono esistere film con lo stesso titolo, quindi il titolo da solo non permette di identificare un film. Al contrario, assumiamo che uno stesso regista non possa realizzare due film con lo stesso titolo. Poiché assumiamo che

potrebbero in generale essere realizzati due film con lo stesso titolo nello stesso anno, `(titolo,anno)` non viene individuata come chiave alternativa. Notiamo che, ad esempio, `(titolo,regista,anno)` è una super-chiave per la relazione `Film`.

Per quanto riguarda la relazione `Video`, la collocazione permette di identificare univocamente un video, quindi la chiave è costituita dal solo attributo `colloc`.

Consideriamo infine la relazione `Noleggio`. Se assumiamo che ogni video venga noleggiato (e quindi restituito) al più una volta ogni giorno, `(colloc,dataNol)` e `(colloc,dataRest)` sono chiavi candidate per `Noleggio`. Al contrario, siccome uno stesso cliente può di solito noleggiare (e/o restituire) più video nello stesso giorno e può noleggiare più volte lo stesso video, né `(codCli,dataNol)`, né `(codCli,dataRest)`, né `(codCli,colloc)` sono chiavi per la relazione `Noleggio`. Poiché `dataRest` può assumere valore nullo per i noleggi in corso e le chiavi primarie non possono contenere valori nulli, viene selezionata `(colloc,dataNol)` come chiave primaria per `Noleggio`.

Possiamo quindi riassumere lo schema della base di dati come segue:

`Cliente(`<u>`codCli`</u>`,`*nome*`,`*cognome*`,telefono,`*dataN*`,residenza)`
`Film(`<u>`titolo`</u>`,`<u>`regista`</u>`,anno,genere,valutaz`$_\circ$`)`
`Video(`<u>`colloc`</u>`,titolo,regista,tipo)`
`Noleggio(`<u>*colloc*</u>`,`<u>`dataNol`</u>`,codCli,`*dataRest*$_\circ$`)`.

Lo schema evidenzia anche che per la valutazione di un film sono ammessi valori nulli, in caso tale valutazione non sia nota. □

2.1.4 Chiavi esterne

Un aspetto importante nella modellazione di una qualsiasi realtà applicativa riguarda la rappresentazione dei collegamenti tra informazioni memorizzate in relazioni diverse, cioè le associazioni tra entità introdotte nel Capitolo 1. Nel modello relazionale, tali collegamenti sono rappresentati tramite l'uso di *chiavi esterne*.

Definizione 2.7 (Chiave esterna) *Siano R ed R' due relazioni, sia $Y \subseteq U_{R'}$ una chiave per R' e sia $X \subseteq U_R$ un insieme di attributi di R tale che Y e X contengano lo stesso numero di attributi e di dominio compatibile.*[2] *X è una chiave esterna di R su R' se, qualsiasi siano gli stati di R ed R', per ogni tupla t di R esiste una tupla t' di R' tale che $t[X] = t'[Y]$. R viene detta relazione referente e R' viene detta relazione riferita.* ◇

Il vincolo d'integrità semantica che assicura che, se una tupla t di R fa riferimento, tramite i valori di una chiave esterna, ai valori della chiave di una tupla t' di R', t' sia effettivamente presente in R', viene indicato con il termine *vincolo di integrità referenziale*.

[2] Due domini sono compatibili se sono uguali o se i valori di uno possono essere comunque accettati come valori dell'altro. Ad esempio, interi e reali sono compatibili.

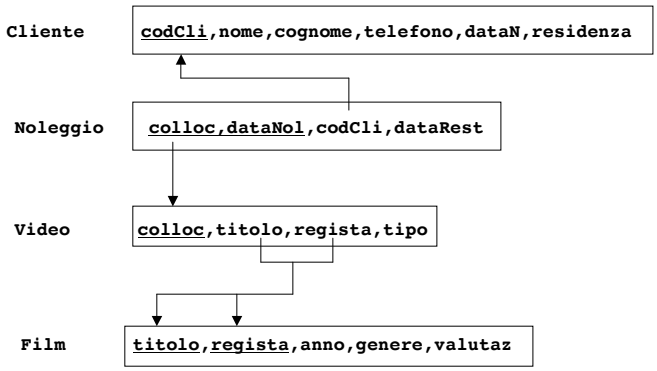

Figura 2.3: Rappresentazione grafica dello schema della base di dati della videoteca

Le chiavi esterne permettono di collegare tra loro tuple di relazioni diverse e costituiscono, pertanto, un meccanismo per realizzare le associazioni. Abbiamo visto nel Capitolo 1 come l'approccio alla modellazione delle associazioni basato su chiavi esterne sia detto *per valore*; infatti una tupla che debba riferire un'altra tupla include tra i suoi attributi degli attributi il cui *valore* è il valore della chiave della seconda tupla.

Nell'esempio seguente ed in tutto il testo useremo la convenzione di indicare, quando questo non crea ambiguità, le chiavi esterne utilizzando il nome della relazione riferita come apice del nome dell'attributo che appartiene alla chiave esterna. Altrimenti, le chiavi esterne e le corrispondenti relazioni riferite verranno elencate a parte. Per evidenziare le chiavi esterne e le corrispondenti relazioni riferite, gli schemi relazionali vengono a volte rappresentati mediante una notazione grafica quale quella presentata nella Figura 2.3 relativamente alla base di dati della videoteca.

Esempio 2.6 Consideriamo nuovamente la base di dati della videoteca. Ogni video contiene un determinato film. L'informazione del film contenuto nel video è rappresentata dagli attributi `titolo` e `regista` in `Video`, che sono una chiave esterna sulla relazione `Film`. La relazione `Noleggio`, a sua volta, contiene due chiavi esterne: `codCli` su `Cliente`, rappresentante il cliente che ha effettuato il noleggio, e `colloc` su `Video`, rappresentante il video noleggiato. Lo schema con l'indicazione delle chiavi esterne diventa quindi:

$$\texttt{Cliente}(\underline{\texttt{codCli}}, \textit{nome}, \textit{cognome}, \texttt{telefono}, \textit{dataN}, \texttt{residenza})$$
$$\texttt{Film}(\underline{\texttt{titolo}}, \underline{\texttt{regista}}, \texttt{anno}, \texttt{genere}, \texttt{valutaz}_\circ)$$
$$\texttt{Video}(\underline{\texttt{colloc}}, \texttt{titolo}^{\texttt{Film}}, \texttt{regista}^{\texttt{Film}}, \texttt{tipo})$$
$$\texttt{Noleggio}(\underline{\textit{colloc}}^{\texttt{Video}}, \underline{\texttt{dataNol}}, \texttt{codCli}^{\texttt{Cliente}}, \textit{dataRest}_\circ).$$

Le relazioni nelle Figure 2.1 e 2.2 verificano il vincolo di integrità referenziale. Infatti ogni tupla della relazione `Video` ha come valore degli attributi `titolo` e `regista` valori che sono anche valori della chiave di una qualche tupla della relazione `Film`. Analogamente, ogni tupla della relazione `Noleggio` ha come valore dell'attributo `colloc` un valore che è valore della chiave di una tupla della relazione `Video` e come valore dell'attributo `codCli` un valore che è valore della chiave di una tupla della relazione `Cliente`. Supponiamo, invece, che la relazione `Noleggio` contenga, in aggiunta alla tuple presenti nella Figura 2.2, la tupla:
`[colloc:1137,dataNol:22-Mar-2006,codCli:6635,dataRest:?]`.
Tale tupla viola il vincolo d'integrità referenziale in quanto non esiste alcun video con numero di collocazione 1137; pertanto tale tupla riferisce un video non esistente. □

L'integrità referenziale può essere violata da inserimenti e modifiche (del valore della chiave esterna) nella relazione referente e da cancellazioni e modifiche (del valore della chiave) nella relazione riferita. Data la rilevanza di tale vincolo, i linguaggi per basi di dati quali SQL permettono all'utente, nella definizione di chiavi esterne, di specificare quali azioni eseguire nel caso in cui operazioni di modifica violino tale vincolo (vedi Capitolo 3).

Esempio 2.7 Consideriamo lo schema dell'Esempio 2.6 e la chiave esterna `codCli` da `Noleggio` su `Cliente`. Possiamo avere violazioni dell'integrità referenziale:

- se inseriamo un nuovo noleggio: l'inserimento in `Noleggio` della tupla `[colloc:1126,dataNol:22-Mar-2006,codCli:6638,dataRest:?]` causa una violazione del vincolo, perché non è presente in `Cliente` una tupla con valore di chiave 6638;
- se modifichiamo il codice del cliente che ha effettuato il noleggio: la modifica del codice di cliente di una tupla della relazione in 6638 causa la violazione del vincolo;
- se cancelliamo un cliente: la cancellazione del cliente con codice 6635 renderebbe invalide 12 tuple della relazione `Noleggio`;
- se modifichiamo il codice di un cliente: la modifica del codice del cliente Paola Bianchi da 6635 a 6638 renderebbe invalide 12 tuple della relazione `Noleggio`. □

Notiamo che nello schema dell'Esempio 2.6 abbiamo scelto di utilizzare lo stesso nome per gli attributi (chiave e chiave esterna) nelle due relazioni. Tale scelta è comoda perché permette l'utilizzo dell'operazione di join naturale, che vedremo nel Paragrafo 2.2, ma non è strettamente necessaria. Ad esempio, l'attributo che modella il cliente che effettua il noleggio potrebbe chiamarsi `cliente` in `Noleggio` invece di `codCli`. In particolare, gli attributi avranno sicuramente nomi diversi tutte le volte che la relazione referente e la relazione riferita coincidono, cioè la

chiave esterna contiene un riferimento alla relazione stessa. Ad esempio, la relazione `Film` potrebbe contenere come chiave esterna su `Film` stessa una coppia di attributi `(titoloPre,registaPre)`, contenente titolo e regista del film di cui il film è eventualmente il seguito. Come evidenziato dall'Esempio 2.6, inoltre, una relazione può contenere più chiavi esterne, eventualmente anche sulla stessa relazione. Rimarchiamo inoltre che le chiavi esterne, come del resto le chiavi, devono essere esplicitamente specificate in uno schema di relazione. Il fatto di avere attributi con lo stesso nome e domini compatibili in relazioni diverse non offre di per sé alcuna garanzia relativamente al mantenimento dell'integrità referenziale. Notiamo infine che, se non esplicitamente impedito mediante la specifica di un apposito vincolo, le chiavi esterne possono assumere valore nullo.

Per concludere la trattazione del modello dei dati relazionale, sottolineiamo che i vincoli di integrità che il modello permette di rappresentare direttamente, e che abbiamo discusso in questo paragrafo, non sono sufficienti a garantire che il contenuto della base di dati rispecchi in modo fedele le informazioni del dominio applicativo da rappresentare. Discuteremo più in dettaglio nel Capitolo 3 (vedi Paragrafo 3.4) altre categorie di vincoli, oltre ai vincoli di dominio, di obbligatorietà, di chiave e di chiave esterna qui discussi, cui il modello relazionale non offre diretto supporto.

2.2 Algebra relazionale

L'algebra relazionale è costituita da varie operazioni per la manipolazione delle relazioni. Più precisamente, l'algebra è composta da cinque operazioni di base: *proiezione*, *selezione*, *prodotto cartesiano*, *unione* e *differenza*. Queste operazioni definiscono completamente l'algebra relazionale. Ogni operazione ha come argomento una o due relazioni (a seconda dell'operazione) e restituisce come risultato una relazione; è pertanto possibile applicare un'operazione al risultato di un'altra operazione (proprietà detta di *chiusura*). Esistono operazioni addizionali, che possono essere espresse in termini delle cinque operazioni di base. Tali operazioni non estendono il potere espressivo dato dalle cinque operazioni di base, ma sono utili come abbreviazione. Tra le operazioni addizionali, l'operazione detta di *join* è la più rilevante. È da notare che la definizione delle varie operazioni è leggermente diversa a seconda che si consideri la definizione del modello relazionale per posizione o la definizione per nome (vedi Paragrafo 2.1); la differenza principale consiste nel modo in cui vengono denotate le componenti delle tuple in alcune delle operazioni che hanno una o più di tali componenti come ulteriore argomento. Nella trattazione seguente, considereremo la definizione delle operazioni in base alla notazione per nome. In riferimento a tale notazione, viene introdotta un'ulteriore operazione, di *ridenominazione*, che permette di modificare i nomi degli attributi.