# Le basi del
# Data Base

## Data Base Management System: il modello gerarchico dei dati

**di Andrea de Prisco**

*Questo mese parleremo dei sistemi gerarchici di gestione basi di dati. Dopo aver visto nel numero scorso il modello semantico dei dati, tuttora oggetto di studio e ricerca in tutto il mondo, faremo di fatto un passo indietro.*
*Fra i Data Base, i sistemi gerarchici sono stati i primi ad apparire sul mercato nel "lontano 1969" ad opera della onnipresente IBM.*
*15 anni di vita, un'eternità nel mondo dell'informatica.*

**Quarta parte**

### I sistemi gerarchici

Soffermeremo maggiormente la nostra attenzione sulle differenze col modello dei dati visto lo scorso mese, mostrando come sia possibile adattare uno schema semantico a uno schema gerarchico.

La struttura dei dati memorizzabili in una base di tipo gerarchico è ad albero, per intenderci pensate un attimo ad un albero genealogico. Avremo una radice, alcuni nodi e i relativi archi tra questi, per descrivere delle associazioni.

Nel caso dell'albero genealogico, la radice sarà il capostipite, ai nodi saranno presenti i discendenti, gli archi rappresenteranno l'associazione padre-figlio. In figura 1 un esempio di albero genealogico che per ragioni più o meno storiche dovrebbe essere abbastanza importante.

La struttura di una base di dati gerarchica è anch'essa ad albero: ogni nodo sarà un insieme di dati, in relazione di tipo gerarchico con altri insiemi della base.

Una base di questo tipo è, per definizione, un insieme di esemplari gerarchici. Un esemplare gerarchico è un albero con radice un elemento dell'insieme radice e come nodi discendenti tutti gli elementi (dei nodi seguenti) in relazione diretta o indiretta con l'elemento della radice.

Un albero genealogico come quello di figura 1 è un esemplare gerarchico della base mostrata in figura 2, dove il primo nodo è l'insieme di tutti i bisnonni, il secondo l'insieme di tutti i nonni, e così via.

Adamo è la radice ed è un elemento di Bisnonni; Caino, Abele e Set (sempre per definizione) sono la famiglia di Adamo nell'insieme Nonni. Enoch e Enos sono elementi di Padri e rispettivamente appartengono alla famiglia di Caino e di Set in Padri. A loro volta Irad e Kenan, appartenenti alle famiglie di Enoch e Enos in Figli, sono elementi dell'insieme Figli. Sembra un discorso un po' contorto, ma è esatta-

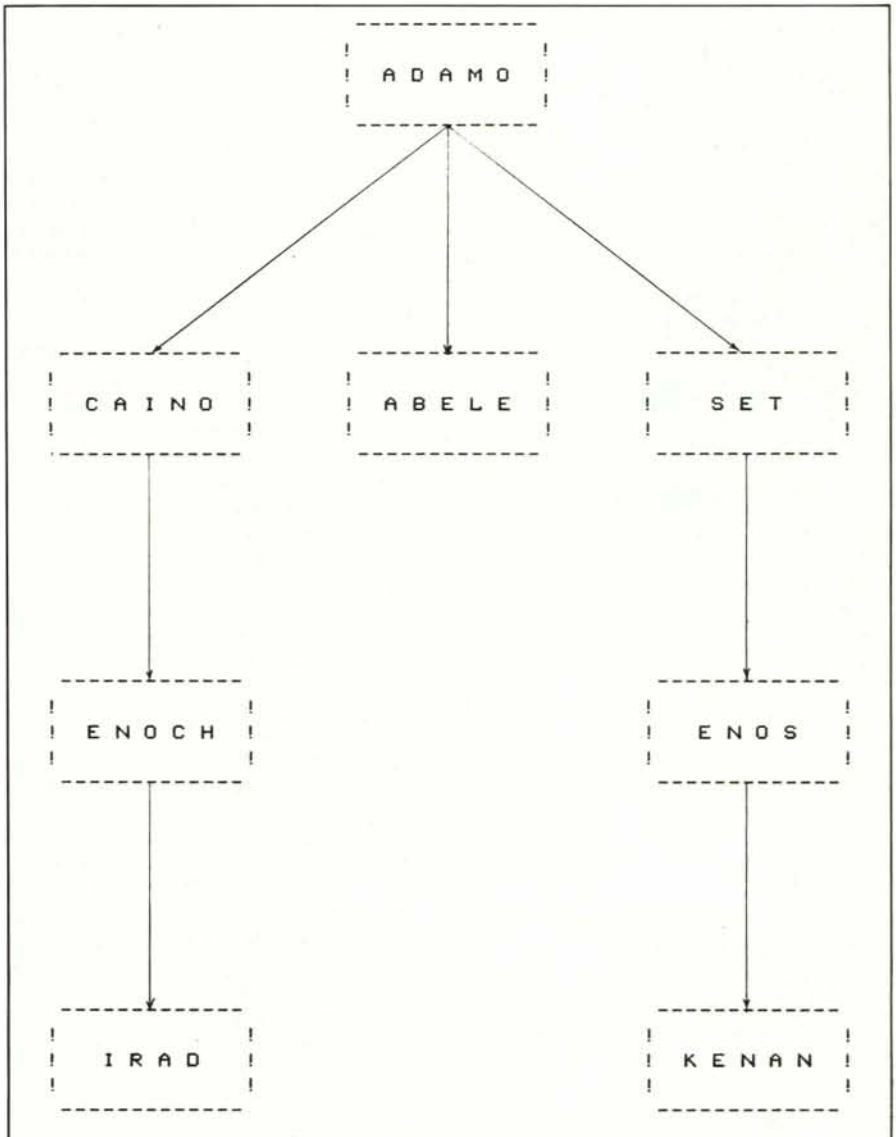



*Figura 1 - Esempio di esemplare gerarchico.*

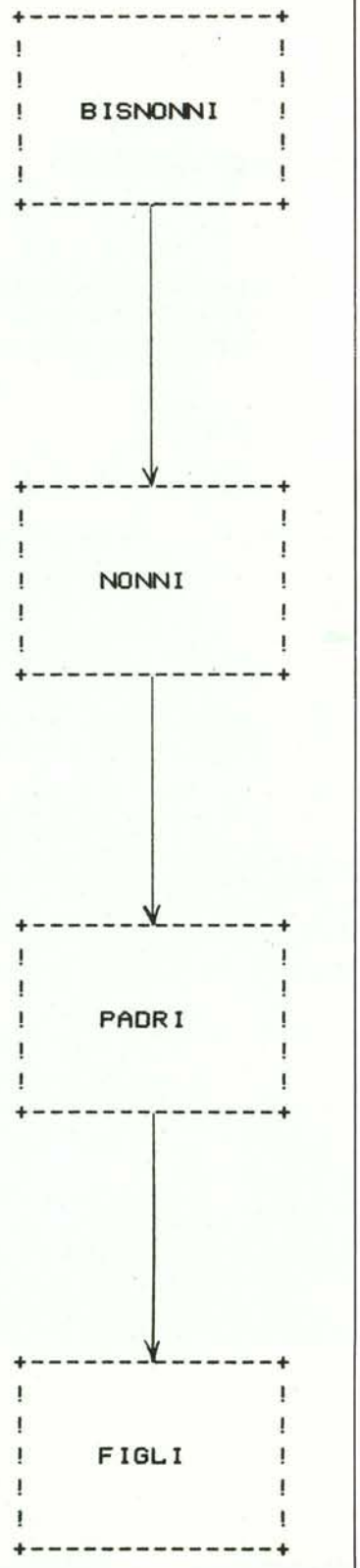



*Figura 2 - Schema della base di dati "Alberi genealogici".*

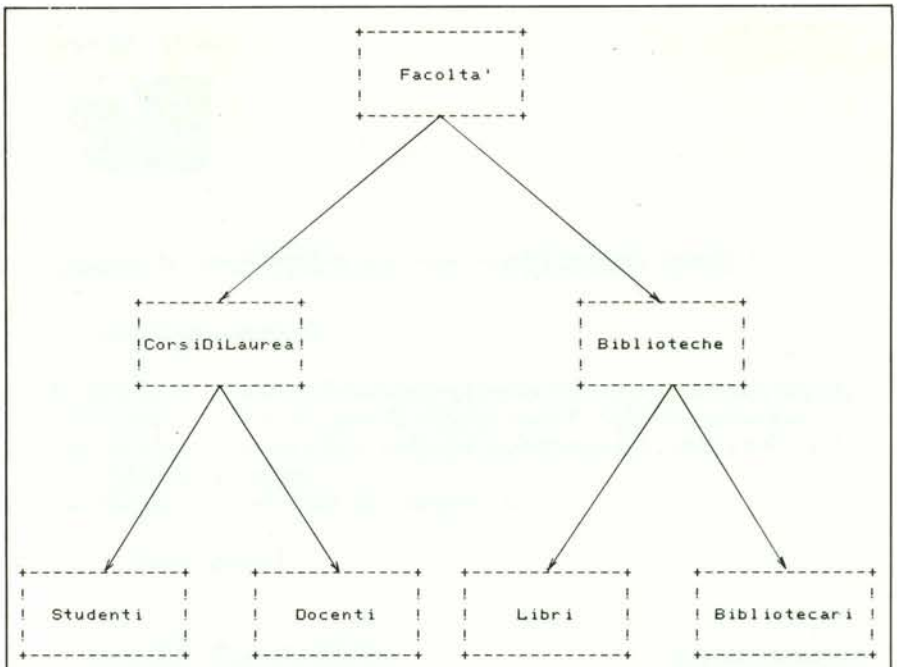



*Figura 3 - Base di dati "Università degli studi".*

mente come nella realtà, come si "manipolano" le discendenze.

Si è usato questo tema in quanto maggiormente adeguato ad essere rappresentato con strutture ad albero. Di fatto però le basi di dati gerarchiche servono per manipolare qualsiasi insieme di dati fra loro correlati.

Assumeremo che un insieme sia precedente ad un altro se lo precede nell'ordinamento gerarchico; discorso analogo per insiemi seguenti. In figura 2 l'insieme dei Nonni è il precedente dell'insieme dei Padri e il seguente dell'insieme dei Bisnonni. Quest'ultimo è precedente a tutti gli insiemi: è detto insieme radice. Analogamente l'insieme Figli è seguente a tutti gli altri: è detto insieme terminale.

Esistono vincoli propri del tipo di organizzazione, che non possono essere violati.

* le associazioni tra i dati hanno tutte diretta multipla e inversa univoca.
* l'inversa è sempre totale.

Diretta multipla vuol dire che ogni elemento in un insieme può essere associato con più elementi dell'insieme seguente. Inversa univoca vuol dire che ogni elemento di ogni insieme diverso dall'insieme radice è in associazione con un unico elemento dell'insieme precedente (deve appartenere a qualche famiglia). Totale vuol dire che questo elemento esiste sempre.

Sempre in merito alla genealogia, è facile verificare che ogni elemento di ogni insieme può avere uno o più figli (elementi nell'insieme seguente) e deve avere un padre (elemento nell'insieme precedente). Eccezion fatta per la radice e per l'ultimo insieme: assumiamo che al momento dell'installazione della base di dati nulla si sappia in merito ai padri dei bisnonni e ai figli dei figli.

In figura 3 è mostrato un caso più generale: l'organizzazione di una università. Come si può notare, ogni insieme non terminale è in relazione con due insiemi seguenti.

Nell'insieme radice abbiamo le varie facoltà (Medicina, Ingegneria, Farmacia, Scienze ecc.). Ogni facoltà ha uno o più corsi di laurea e una o più biblioteche. A loro volta ogni Corso di Laurea ha dei docenti e degli studenti; ogni biblioteca ha un certo numero di Libri e del personale addetto.

Lo studente Tizio, ad esempio, potrebbe essere iscritto al Corso di Laurea in Matematica, facoltà di Scienze. Tizio sarà un elemento di Studenti, appartenente alla famiglia di Scienze Matematiche in Studenti. Scienze Matematiche sarà un elemento dell'insieme Corsi di Laurea, facente parte della famiglia di Scienze; quest'ultima è un elemento di Facoltà.

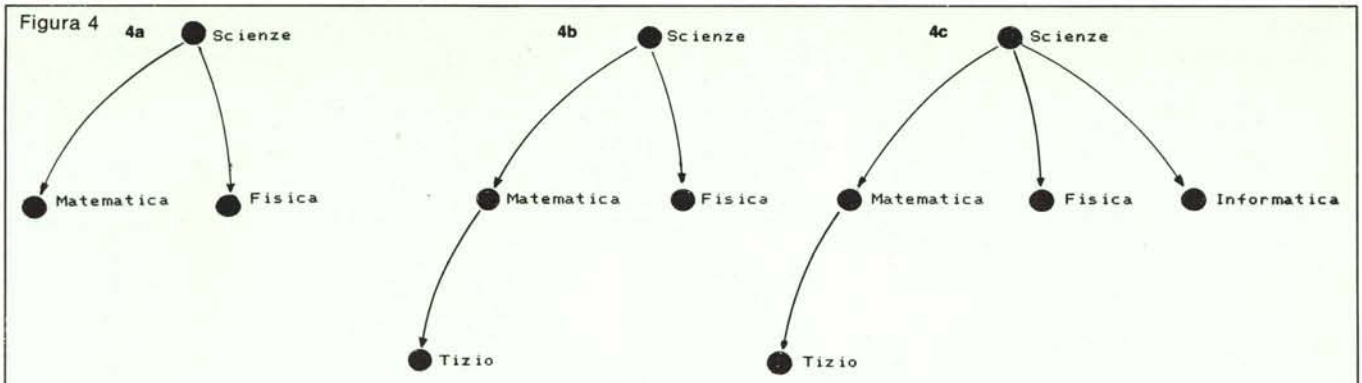
## Gli operatori

Vedremo ora come è possibile costruire una base di dati gerarchica e come si opera su essa per prelevare o inserire elementi. Nella definizione di ogni insieme, come già visto per le classi del modello semantico dei dati, si dovranno elencare gli attributi di ogni ennupla: es. nome, cognome, indirizzo ecc.

Bisognerà, inoltre, dichiarare quali sono gli insiemi (nodi seguenti) in relazione con l'insieme che si sta definendo. Nessun riferimento ai nodi precedenti: il sistema è in grado di riconoscerli automaticamente.

Prima di parlare di inserimento e ricerca, chiariremo un punto molto importante: l'ordinamento di un albero. Un esemplare gerarchico, come visto, è in generale

Figura 4




*Un esemplare gerarchico della base di figura 3.* *Inserimento dello studente "Tizio".* *Inserimento del corso di laurea in Informatica.*

una struttura ad albero. Anche in strutture di questo tipo, è possibile definire una relazione di ordinamento: un meccanismo per sapere se un elemento viene prima o dopo di un altro. Basta mettersi daccordo una volta per tutte. Nei sistemi gerarchici, il tipo di ordinamento adottato per gli elementi di un albero è quello anticipato: il nodo prima della sua famiglia, ricorsivamente. Ordiniamo l'esemplare gerarchico di figura 1: il primo elemento è Adamo, poi viene la sua famiglia. Questa è composta da Caino, Abele e Set. Si comincia da Caino e si passa alla sua famiglia: Enoch. A sua volta la sua famiglia è Irad. Abbiamo finito con Caino, si sale su e il successore è Abele: non ha famiglia, si passa a Set che ha come famiglia Enos che ha come famiglia Kenan.

La visita dell'albero è completa: ricapitolando, l'ordinamento sequenziale gerarchico in quest'albero è: Adamo, Caino, Enoch, Irad, Abele, Set, Enos, Kenan.

La semantica degli operatori dei sistemi gerarchici si basa, per l'appunto, sull'ordinamento degli esemplari. Per l'esattezza gli operatori hanno una semantica espressa in termini di un elemento detto di riferimento corrente.

L'elemento corrente è sempre l'ultimo elemento inserito o l'ultimo elemento ripescato nella base di dati. A partire da un elemento corrente è possibile richiedere il successivo (nell'ordinamento), specificando l'insieme in cui effettuare la ricerca e eventualmente una condizione.

La prima operazione da compiere è l'inizializzazione dell'elemento corrente. Supponiamo di avere lo schema di figura 2, con inserito l'esemplare gerarchico mostrato in figura 1.

Inizializziamo come elemento di riferimento Adamo. Scriveremo qualcosa del tipo:

```
get Bisnonni with Nome = "Adamo"
```

Bisnonni è l'insieme in cui effettuare la ricerca; Nome = "Adamo" la condizione da verificare. Per muoverci sull'esemplare esiste l'operatore next, seguito da insieme e, facoltativamente, condizione. Esempio:

```
next Nonni
```

restituisce Caino (è il successore di Adamo, nell'insieme dei Nonni, secondo l'ordinamento anticipato visto sopra). Caino diventa il nuovo elemento corrente. Possiamo spostarci specificando una condizione:

```
next Nonni with Nome = "Set"
```

l'elemento di riferimento diventa ora Set: è lui il prossimo in Nonni, secondo l'ordinamento, con Nome = "Set".

Facciamo un altro esperimento: inizializziamo nuovamente l'elemento di riferimento corrente, nel seguente modo:

```
get Figli
```

ci posizioniamo su Irad. Con:

```
next Figli
```

otteniamo Kenan. Irad, nell'esemplare gerarchico è il primo nell'insieme Figli che si incontra. Il suo successore, nello stesso insieme è Kenan. Fin qui tutto regolare. Esiste un altro operatore, nextd, che fa lo stesso lavoro, senza però allontanarsi dalla famiglia dell'elemento di riferimento. Se dopo esserci posizionati su Irad eseguiamo un:

```
nextd Figli
```

Il sistema genererà errore non esistendo altri elementi in Figli, della stessa famiglia di Irad (= altri elementi figli di Enoch).

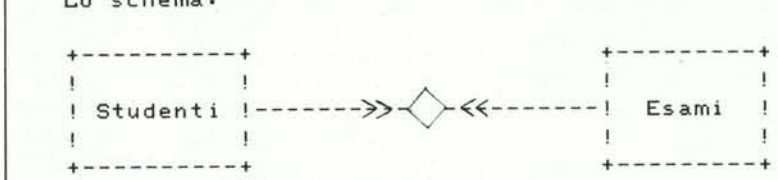
Per inserire un elemento nella base di dati esiste l'operatore make. La sua sintassi è:

```
make Insieme Elemento
```

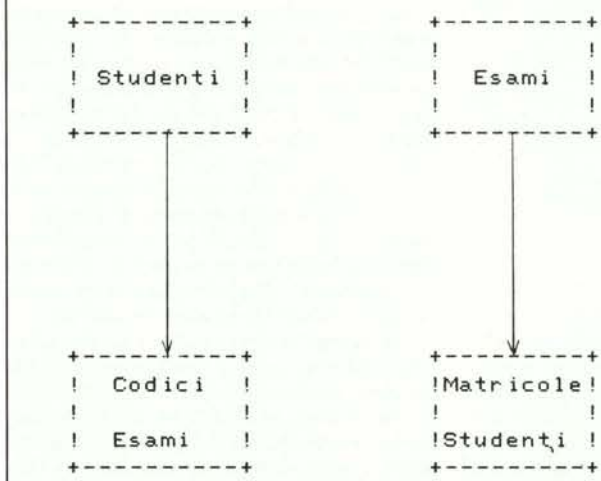
Elemento è l'elemento da inserire; Insieme è l'insieme in cui vogliamo inserirlo. L'operatore make costruisce un elemento in Insieme e lo pone come successivo dell'elemento corrente, nell'ordinamento sequenziale gerarchico. L'elemento inserito

Figura 5

**5a** *Associazione con diretta e inversa multiple.*

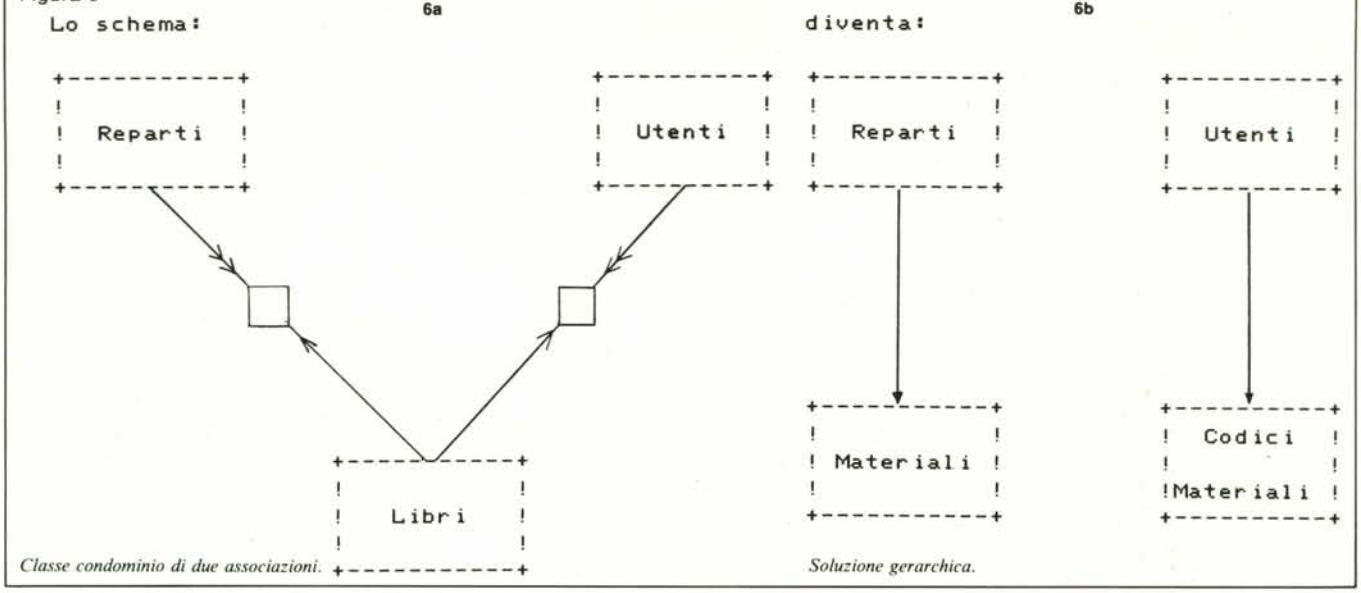



**5b** diventa:




*Soluzione gerarchica.*

Figura 6




*Classe condominio di due associazioni.*

*Soluzione gerarchica.*

diventa il nuovo riferimento corrente. Facciamo qualche esempio. È data la base di dati di figura 3. Immaginiamo che siano già stati inseriti alcuni dati, tra cui l'esemplare gerarchico di figura 4a: facoltà di Scienze con i due figli (o la famiglia) in CorsiDiLaurea, Matematica e Fisica.

Vogliamo inserire lo studente Tizio, appartenente alla famiglia del Corso di Laurea in Matematica, in Studenti. Come sempre, la prima operazione da compiere è inizializzare l'elemento di riferimento corrente. Dovendo modificare i figli di Matematica eseguiremo:

```
getfirst CorsiDiLaurea with Nome = "Matematica"
```

in parole povere cerchiamo nell'insieme CorsiDiLaurea l'elemento (in generale il primo elemento) che ha per nome "Matematica". Con:

```
make Studenti(Tizio)
```

avremo la situazione di figura 4b. Come promesso, Tizio è posto dopo Matematica, nell'ordinamento sequenziale gerarchico. Tizio diventa il nuovo riferimento corrente. Supponiamo a questo punto di voler inserire il corso di laurea in Informatica (il nome vero è Scienze dell'Informazione, ndr). Non c'è bisogno di spostare l'elemento corrente, dato che a noi va bene che (Informatica) sia posto come successore (in CorsiDiLaurea) di Tizio. Basterà un:

```
make CorsiDiLaurea (Informatica)
```

per avere la situazione di figura 4c.

## Da schema semantico a schema gerarchico

I meccanismi di astrazione del modello semantico dei dati, visti nel numero scorso, permettono di modellare molto facilmente i vari aspetti della conoscenza, prima dell'installazione di una base di dati. Col modello gerarchico, più precisamente a causa dei suoi vincoli impliciti, modellare determinate situazioni diventa più difficile, a meno che non si ricorra a particolari trucchetti, discussi in seguito.

Un esempio classico è mostrato in figura 5a. Si vuole rappresentare l'associazione multipla Esami-Studenti. Multipla vuol dire che ogni Studente è in relazione con più esami (tutti quelli che ha sostenuto) e ogni esame di una particolare facoltà è stato (ovviamente) superato da più studenti.

Col modello gerarchico, tale situazione non è direttamente realizzabile, essendo l'inversa di ogni associazione unica. In parole povere, uno studente può puntare più esami (associazione diretta multipla) ma ogni esame (associazione inversa) può puntare un solo studente (unica). Come si può notare non torna affatto. Finché si parla di genitori e figli tutto ok (un padre ha più figli, ma ogni figlio ha un solo padre), allontanandoci un pochino da schemi per natura gerarchici, il modello dei dati discusso questo mese mostra subito i suoi limiti, a causa anche della sua considerevole età.

In figura 5b è mostrata una delle possibili soluzioni: si può simulare l'associazione multipla solo a costo di pagare una non indifferente ridondanza (duplicazione di alcuni dati). Si costruiscono due alberi gerarchici, formati da due insieme l'uno. Il primo albero gerarchico è formato dall'insieme Studenti e da un particolare insieme di servizio che contiene codici di esami (un numero che li identifica univocamente). Ogni studente che ha sostenuto esami ha nell'insieme Codici Esami una sua propria famiglia: i codici degli esami da lui superati. Nel secondo albero gerarchico sono presenti altri due insiemi: uno Esami, l'altro di servizio, che contiene matricole di studenti. Stesso meccanismo: ogni esame di Esami ha una sua famiglia in Matricole Studenti, rappresentante le matricole di tutti gli studenti che hanno superato l'esame in considerazione.

L'associazione multipla, a questo punto, funziona così: partendo da Studenti, possiamo conoscere la famiglia in Codici Esami di un particolare studente. Per ogni codice pescato accediamo all'insieme Esami, per conoscere effettivamente gli esami sostenuti. Viceversa, per sapere da quali studenti è stato sostenuto un esame, si accede a Esami, ci si sposta in Matricole Studenti e prese le varie matricole (appartenenti alla famiglia dell'esame in considerazione) si accede in Studenti per conoscere gli altri attributi (nome, cognome, età ecc.).

In figura 6a è mostrato un altro caso non esprimibile direttamente col modello dei dati gerarchico. C'è una classe, libri, puntata da due altre classi: in termini più specifici, si dice condominio di due associazioni. La classe Utenti conterrà gli utenti che accedono alla biblioteca per prendere in prestito libri; la classe Reparti contiene i vari reparti (suddivisi per materia) della biblioteca. Un libro si trova in un solo reparto e può (ovviamente) essere in prestito a un solo utente. Viceversa, in ogni reparto ci sono più libri e ogni utente può averne in prestito più d'uno.

Anche questa situazione non è gerarchica, in figura 6b è mostrata una possibile soluzione.

Come prima, due alberi gerarchici. Il primo non fa altro che realizzare la prima delle due associazioni: Reparti-Libri. Il secondo albero, col solito trucchetto dell'insieme di servizio, completa l'associazione. Codici Libri contiene per l'appunto codici di libri, ognuno identifica univocamente un libro della biblioteca. Se si vuol conoscere quali libri ha in prestito un particolare utente, si accede a Utenti, si scende in Codici Libri. Conoscendo la famiglia in Codici Libri dell'utente in considerazione, grazie ai codici prelevati, si prelevano dall'insieme "Libri", Autore, Titolo e posizione dei materiali ricercati. MC