una chiave esterna che riferisce la chiave primaria Matricola della relazione Studenti, e infatti vale che $adom(\text{Candidato}) \subseteq adom(\text{Matricola})$.

Il vincolo sulla chiave esterna non era previsto nella prima definizione del modello relazionale, ma è molto utile ed è previsto in quasi tutti i sistemi relazionali commerciali. Altri vincoli più generali sulle possibili estensioni di una base di dati relazionale potrebbero essere espressi con un opportuno linguaggio, ma qui non si prendono in considerazione. Le chiavi esterne rappresentano il modo standard di rappresentare le associazioni nel modello relazionale. Questo meccanismo permette di rappresentare direttamente la direzione unaria delle associazioni binarie uno a molti (che sono chiamate anche associazioni *gerarchiche*), e quindi anche delle associazioni uno ad uno; vedremo in seguito come possa essere usato per rappresentare associazioni molti a molti, non binarie o con attributi.

Negli schemi di esempio si evidenzieranno gli attributi della chiave primaria di uno schema di relazione sottolineandoli, mentre si userà l'asterisco per indicare gli attributi chiave esterna.

4.1.3 Una rappresentazione grafica di schemi relazionali

Come per il modello ad oggetti, anche per il modello relazionale è possibile definire un formalismo grafico in cui si rappresentano solo gli schemi di relazione e le loro associazioni, o più precisamente le chiavi esterne. In questo formalismo, una relazione è rappresentata da un rettangolo che ne contiene il nome. La presenza di una chiave esterna in R che riferisce la chiave primaria di S è rappresentata da una freccia che va da R ad S. La freccia sarà tagliata, in maniera analoga all'indicazione di parzialità del modello concettuale, quando la chiave esterna può assumere il valore null. Quando sia utile, la freccia può essere etichettata con il nome degli attributi che formano la chiave esterna, e ulteriori attributi della relazione possono essere rappresentati come si è potuto vedere nel Capitolo 2. Nei diagrammi dove sono specificati gli attributi, verrà aggiunta l'etichetta ≪PK≫ alle chiavi primarie, ≪CK≫ alle altre chiavi, e ≪FK≫(*Relazione*) alle chiavi esterne. Riportiamo, a titolo di esempio, in Figura 4.1 una rappresentazione grafica della base di dati con le relazioni Studenti e Esami.

4.1.4 Operatori

Il modello relazionale è stato il primo ad introdurre la possibilità di operare su insiemi di dati con operatori insiemistici, mentre nei modelli precedenti lo stile di programmazione era basato su operatori di scansione simili agli operatori *open, read, write, next, eof, close* con i quali i linguaggi imperativi manipolano i *file*.

I linguaggi per interrogare una base di dati relazionale permettono di definire la relazione risultato a partire dalle relazioni che compongono la base di dati. Essi si possono dividere in tre famiglie, che verranno esemplificate più avanti:

1. *linguaggi algebrici*: un'interrogazione è definita da un'espressione con operatori su relazioni che producono come risultato altre relazioni;

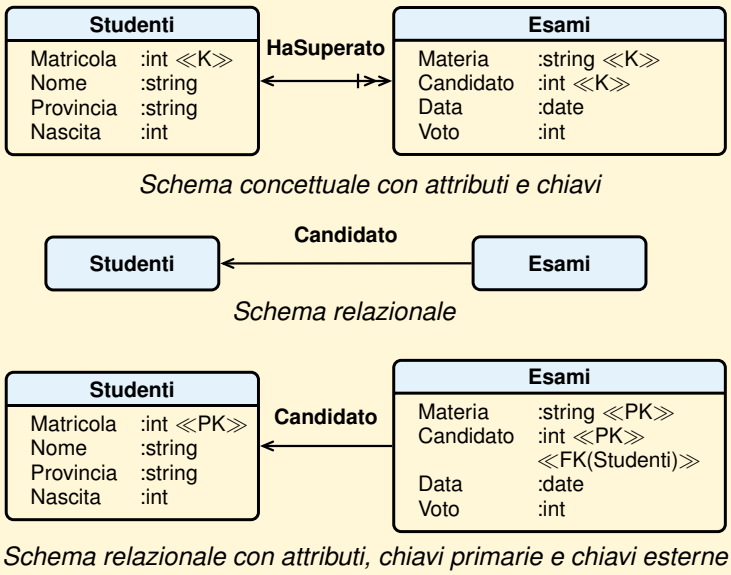


Figura 4.1: La rappresentazione grafica di uno schema relazionale

2. *linguaggi basati sul calcolo dei predicati*: un'interrogazione è definita da una formula del calcolo dei predicati del primo ordine;
3. *linguaggi logici*: sono linguaggi ispirati dal linguaggio logico Prolog in cui un'interrogazione è definita da un insieme di formule del calcolo dei predicati del primo ordine, mutuamente ricorsive ma con una struttura molto rigida (formule *Horn*).

Mentre una formula del calcolo si limita a specificare quali ennuple fanno parte del risultato, un'espressione algebrica stabilisce quali operazioni applicare per produrre il risultato. In genere, in un sistema relazionale il programmatore definisce il risultato della propria interrogazione fornendo una formula del calcolo, ed è il sistema che sceglie l'espressione algebrica equivalente da eseguire. A questo scopo il sistema per prima cosa trasforma la formula in un'espressione algebrica equivalente e poi applica a questa espressione una serie di *riscritture*, ovvero di trasformazioni che non ne modificano il risultato ma ne rendono la valutazione più efficiente. Vedremo, nella Sezione 4.3.3 un esempio di tali trasformazioni.

L'*algebra relazionale* e il *calcolo relazionale di ennuple* sono esempi di linguaggi delle prime due famiglie e furono proposti da Codd come esempi di linguaggi equivalenti e con le capacità minime di calcolo su relazioni. Egli, inoltre, introdusse il concetto di *completezza* di un linguaggio relazionale, intendendo con ciò che si possa formulare in esso un'espressione equivalente (cioè che valuti lo stesso risultato) ad una qualunque espressione dell'algebra relazionale, cioè una espressione formata da variabili e costanti di relazioni e da operatori dell'algebra. In realtà i linguaggi esistenti per sistemi relazionali offrono anche altri operatori che li rendono più che "completi". Ad esempio, quasi tutti offrono le operazioni aritmetiche e operazioni come la media,

somma, minimo o massimo, per agire su insiemi di valori per ottenere una singola quantità. I linguaggi logici sono anche più espressivi perché riescono ad esprimere interrogazioni "ricorsive", per calcolare ad esempio la chiusura transitiva di una relazione, ma pagano questa espressività con una maggiore difficoltà nell'ottimizzazione delle interrogazioni.

4.2 Progettazione logica relazionale

Il modello relazionale non si presta bene, per la sua povertà espressiva, alla progettazione concettuale; in questa fase si preferisce utilizzare il modello ad oggetti o il modello entità-relazioni. Al termine di questa fase poi, se si decide di realizzare la base di dati utilizzando un sistema relazionale, si trasforma lo schema concettuale prodotto in uno schema logico relazionale.

La trasformazione di uno schema ad oggetti in uno schema relazionale avviene eseguendo i seguenti passi:

PASSO 1: Rappresentazione delle associazioni uno a molti e uno ad uno.
PASSO 2: Rappresentazione delle associazioni molti a molti.
PASSO 3: Rappresentazione delle gerarchie fra classi.
PASSO 4: Definizione delle chiavi primarie.
PASSO 5: Rappresentazione degli attributi multivalore.
PASSO 6: Appiattimento degli attributi composti.

Questa trasformazione dovrebbe essere guidata dai seguenti obiettivi:

- rappresentare le stesse informazioni;
- minimizzare la ridondanza;
- produrre uno schema comprensibile, per facilitare la scrittura e manutenzione delle applicazioni.

In teoria, durante la trasformazione non si dovrebbe tenere conto di considerazioni relative all'efficienza delle applicazioni, poiché questi aspetti dovrebbero riguardare la fase di progettazione fisica (che in questo testo non viene presa in considerazione).

In generale nella conversione occorre duplicare certe informazioni e non si possono sempre rappresentare direttamente tutti i vincoli imposti dai meccanismi del modello ad oggetti. Per garantire la coerenza dei dati duplicati, e il rispetto dei vincoli non esprimibili nel modello relazionale, occorrerà quindi definire opportunamente le operazioni che modificano la base di dati.

Nel seguito verrà descritto un metodo per trasformare uno schema descritto con il modello ad oggetti in uno equivalente descritto con il modello relazionale, prendendo in considerazione solo gli aspetti riconducibili a dati (aspetti strutturali). Nel prossimo capitolo sulla teoria relazionale dei dati vedremo invece un altro approccio alla progettazione di uno schema relazionale, basato su un metodo formale che parte da una descrizione dei fatti da trattare data in modo completamente diverso. Alcuni

risultati di questa teoria consentono però di giustificare anche la ragionevolezza delle regole di trasformazione presentate in questo capitolo.

Il metodo che definiremo applica i primi tre passi alla rappresentazione grafica di uno schema ad oggetti, producendo, come risultati intermedi, dei disegni che contengono sempre meno costrutti ad oggetti, fino ad arrivare ad uno schema totalmente relazionale. Gli attributi delle classi vengono quindi presi in considerazione solo per gli ultimi tre passi. Il metodo sarà esemplificato applicandolo alo schema studiato nel Capitolo 2, rappresentato in Figura 4.2.

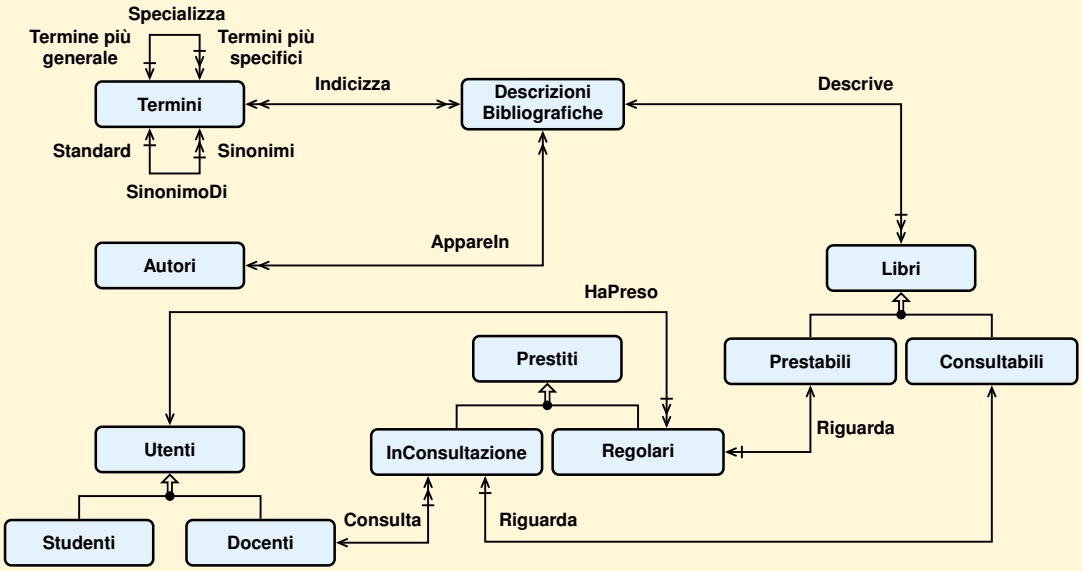


Figura 4.2: Lo schema ad oggetti da trasformare di Figura 2.7

4.2.1 PASSO 1: Rappresentazione delle associazioni uno a molti e uno ad uno

Come abbiamo già visto, le associazioni uno a molti si rappresentano aggiungendo agli attributi della relazione rispetto a cui l'associazione è univoca e totale una chiave esterna che riferisce l'altra relazione. Ad esempio, l'associazione tra Studenti e Esami, essendo univoca e totale rispetto a Esami si rappresenta aggiungendo a Esami una chiave esterna Candidato. Se l'associazione ha degli attributi, questi si aggiungono alla relazione in cui è presente la chiave esterna.

Se invece l'associazione nella direzione univoca è parziale, come capita comunemente nelle relazioni ricorsive di una classe, si possono scegliere due diverse rappresentazioni nello schema logico relazionale, come si mostra in Figura 4.3. nel caso della classe Termini. Nella prima si aggiunge allo schema una nuova relazione che contiene due chiavi esterne che riferiscono le due relazioni coinvolte dall'associazione, e gli eventuali attributi dell'associazione; questa relazione contiene un'ennupla per ogni istanza dell'associazione. La chiave primaria di questa relazione è la chiave esterna

che riferisce la relazione rispetto a cui l'associazione è multipla. Nella seconda soluzione, invece, si procede come nel caso precedente, aggiungendo una chiave esterna con possibili valori nulli alla relazione rispetto a cui l'associazione è univoca.

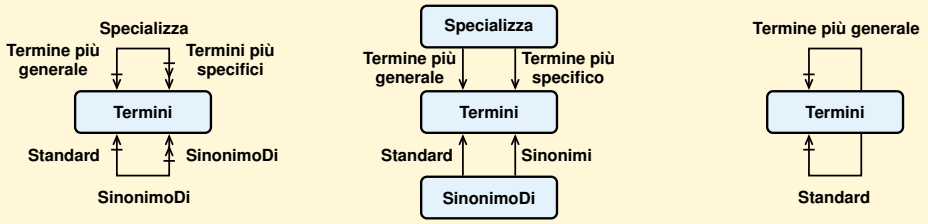


Figura 4.3: Esempio di trasformazione di associazioni ricorsive

Per le associazioni uno a uno si procede in maniera analoga, considerandole un caso particolare di associazioni molti a uno. Si consideri che quando una associazione ha sia la diretta che l'inversa, e si aggiunge una relazione con due chiavi esterne, una qualunque delle due può essere scelta come chiave primaria.

Graficamente, questo significa operare le sostituzioni specificate in Figura 4.4 su tutte le associazioni uno a molti e uno a uno presenti nello schema, supponendo che in ogni relazione sia definita la chiave primaria.

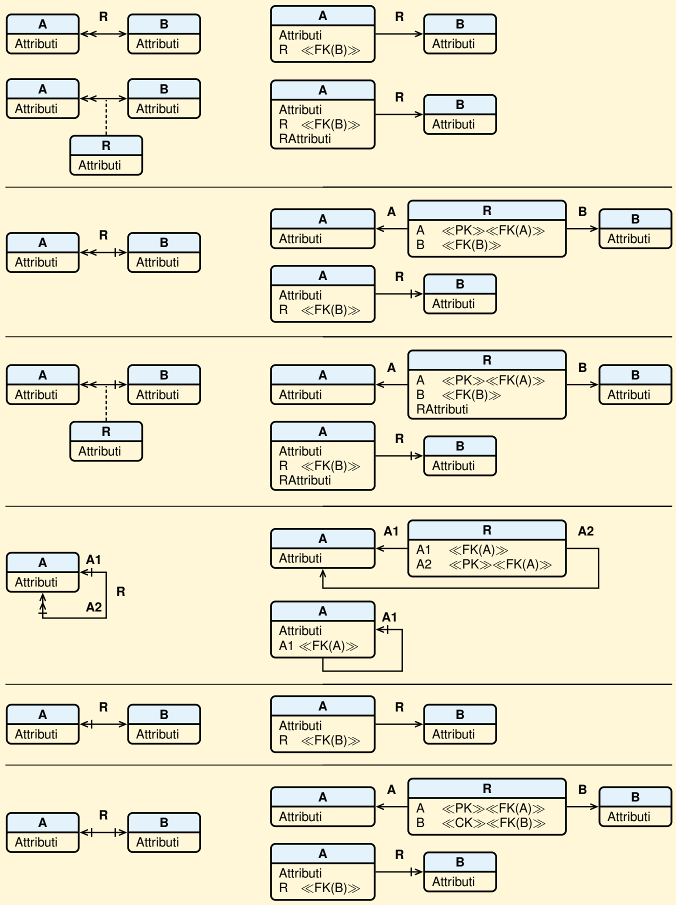
Ad esempio, applicando questo passo di trasformazione alla Figura 4.2, si ottiene il disegno in Figura 4.6.

4.2.2 PASSO 2: Rappresentazione di associazioni molti a molti

Un'associazione molti a molti tra due classi si rappresenta aggiungendo allo schema una nuova relazione che contiene due chiavi esterne che riferiscono le due relazioni coinvolte; la chiave primaria di questa relazione è costituita dall'insieme di tutti i suoi attributi. Questa relazione contiene un'ennupla per ogni istanza dell'associazione. Se l'associazione ha degli attributi, questi attributi vengono aggiunti alla nuova relazione, e non vanno a far parte della chiave della nuova relazione.

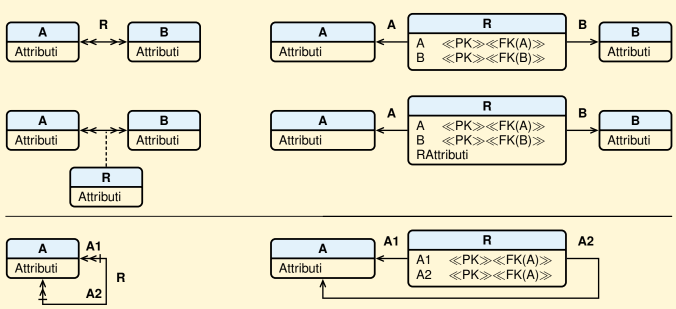
Graficamente, questo significa operare la sostituzione specificata in Figura 4.5 su tutte le associazioni molti a molti nello schema, che non cambia in presenza di indicazioni di parzialità nei primi due casi.

Ad esempio, applicando questa trasformazione alla Figura 4.6, si ottiene il disegno in Figura 4.7, dove la relazione Indicizza rappresenta l'associazione molti a molti tra Termini e DescrizioniBibliografiche.



Tipi di associazioni *Rappresentazione nel modello relazionale*

Figura 4.4: Regole per il **PASSO 1** di trasformazione



Tipi di associazioni *Rappresentazione nel modello relazionale*

Figura 4.5: Regole per il **PASSO 2** di trasformazione

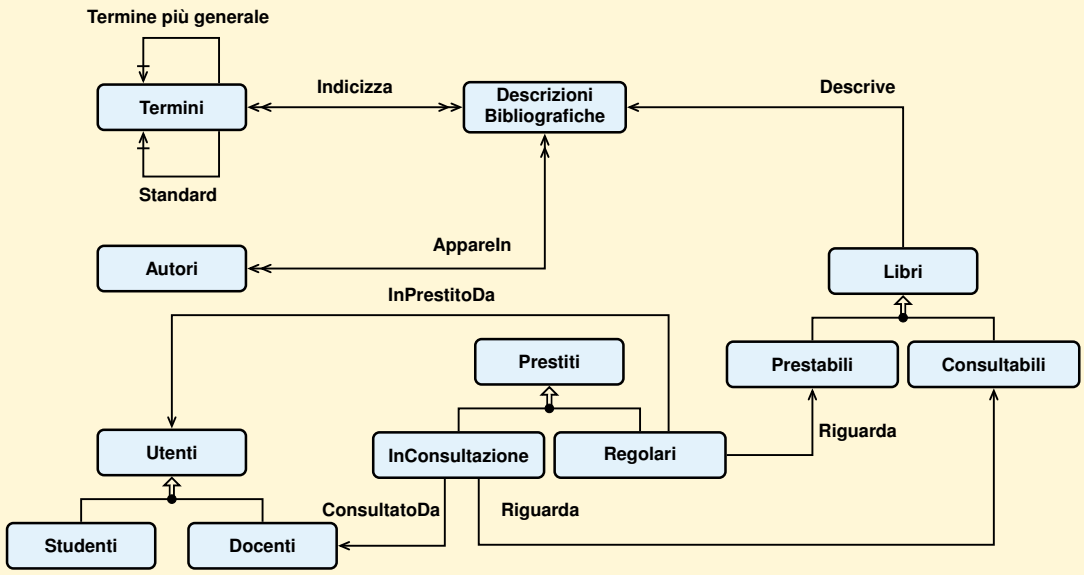


Figura 4.6: Effetto del **PASSO 1** di trasformazione

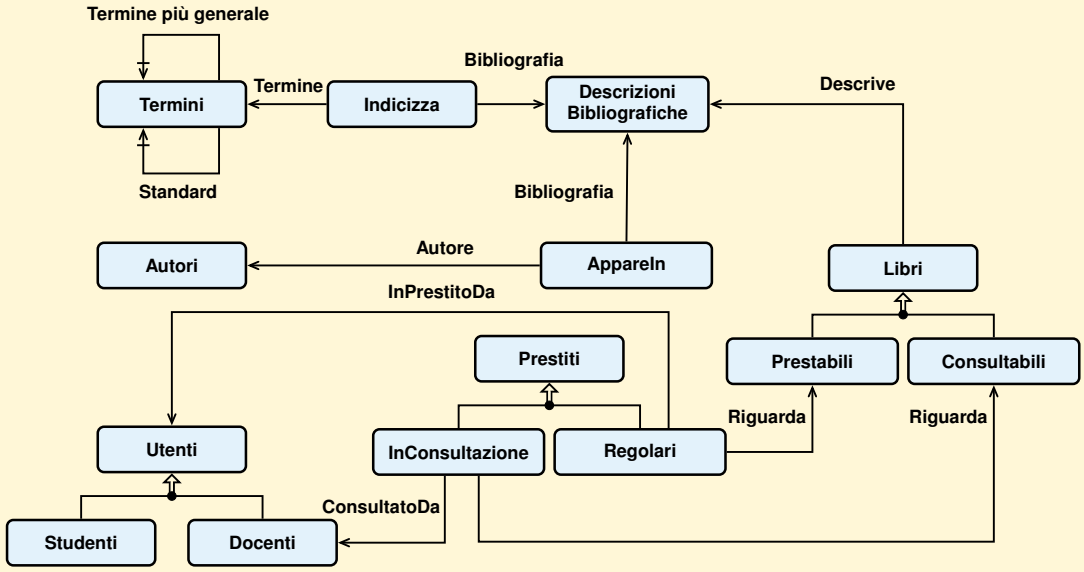


Figura 4.7: Effetto del **PASSO 2** di trasformazione

4.2.3 PASSO 3: Rappresentazione delle gerarchie fra classi

Sia data una classe A con due sottoclassi B e C, tali che i tipi associati alle tre classi abbiano, rispettivamente, attributi (X_A), $(X_A X_B)$ e $(X_A X_C)$, e sia K_A la chiave primaria di A.[3]

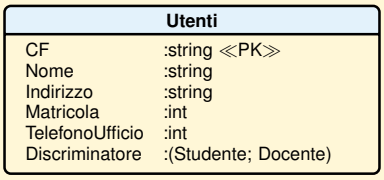
Nel modello relazionale vi sono almeno tre modi diversi di rappresentare questa situazione:

1. *relazione unica*: si definisce un'unica relazione con attributi (X_A, X_B, X_C, Discriminatore) che raccoglie tutti gli elementi delle tre classi; gli attributi X_B, X_C possono assumere il valore null e l'attributo Discriminatore serve a indicare la classe a cui appartiene l'elemento;
2. *partizionamento verticale*: si definiscono tre relazioni $R_A(X_A)$, $R_B(K_A, X_B)$ e $R_C(K_A, X_C)$. R_A contiene tutti gli elementi della classe A, anche se stanno in qualche sottoclasse, mentre R_B ed R_C contengono solo quegli attributi, degli elementi di B e di C, che non sono in X_A (attributi *propri* delle sottoclassi), ed una chiave esterna K_A che permette di ritrovare in R_A il valore degli altri attributi. K_A è anche la chiave primaria di R_B ed R_C;
3. *partizionamento orizzontale*: si definiscono tre relazioni $R_A(X_A)$, $R_B(X_A,X_B)$, $R_C(X_A, X_C)$. R_A contiene solo gli elementi della classe A che non stanno in nessuna delle sottoclassi, mentre R_B ed R_C contengono tutti gli elementi di B e di C; se le sottoclassi costituiscono una copertura, la relazione $R_A(X_A)$ non viene definita perché sarebbe sempre vuota.

Ad esempio, supponiamo di avere una classe Utenti con gli attributi CF (la chiave), Nome e Indirizzo, e con due sottoclassi di tipo partizione: Studenti, con un attributo Matricola, e Docenti, con attributo TelefonoUfficio.

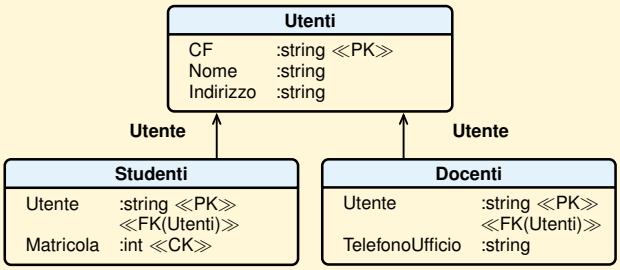
Le tre tecniche precedenti darebbero i seguenti risultati:

1. *relazione unica*: si definisce la relazione



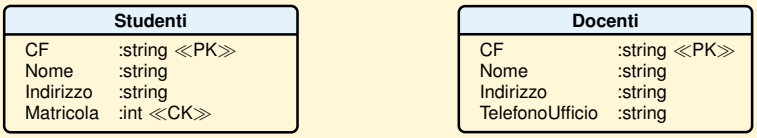
   che raccoglie tutti gli utenti; gli attributi Matricola e TelefonoUfficio possono assumere il valore null, ed il Discriminatore indica se un utente è uno studente o un docente;
2. *partizionamento verticale*: si definiscono tre relazioni

3. Per semplicità, trascuriamo il problema dell'eventuale ridefinizione del tipo degli attributi nelle sottoclassi.



Utenti contiene i dati comuni a tutti gli utenti, mentre le altre due relazioni contengono gli attributi propri delle sottoclassi, nonché il codice fiscale, per risalire al nome e all'indirizzo;

3. *partizionamento orizzontale*: trattandosi di sottoclassi che soddisfano il vincolo di copertura si definiscono solo due relazioni che contengono separatamente tutte le informazioni relative a Studenti e Docenti.



La tecnica da usare si può scegliere come segue.

Se gli attributi propri delle sottoclassi sono pochi si sceglie la relazione unica, che è la tecnica più semplice. Altrimenti, si sceglie tra le altre due tecniche tenendo presente che:

- il partizionamento orizzontale divide gli elementi della superclasse in più relazioni diverse, per cui non è possibile mantenere un vincolo referenziale verso la superclasse stessa; in conclusione, questa tecnica non si usa se nello schema relazionale grafico c'è una freccia che entra nella superclasse;
- il partizionamento verticale rende più complessa la ricostituzione di tutte le informazioni relative ad un elemento della sottoclasse, mentre il partizionamento orizzontale rende più complessa l'operazione di visita a tutti gli elementi della superclasse, per cui è necessario valutare l'importanza relativa di queste due operazioni;
- il partizionamento orizzontale è preferibile in presenza di un vincolo di copertura perché ne può garantire il mantenimento; il partizionamento orizzontale funziona al meglio in presenza di un vincolo di disgiunzione, perché, quando un oggetto appartiene a più sottoclassi, questa rappresentazione replica la chiave e gli attributi della superclasse in tutte le relazioni a cui l'oggetto appartiene, e diviene complesso mantenere la consistenza di queste informazioni replicate.

La Figura 4.8 rappresenta l'esecuzione grafica delle tre possibili trasformazioni, mostrando anche l'effetto dell'operazione sulle associazioni che escono ed entrano nei tre rettangoli.

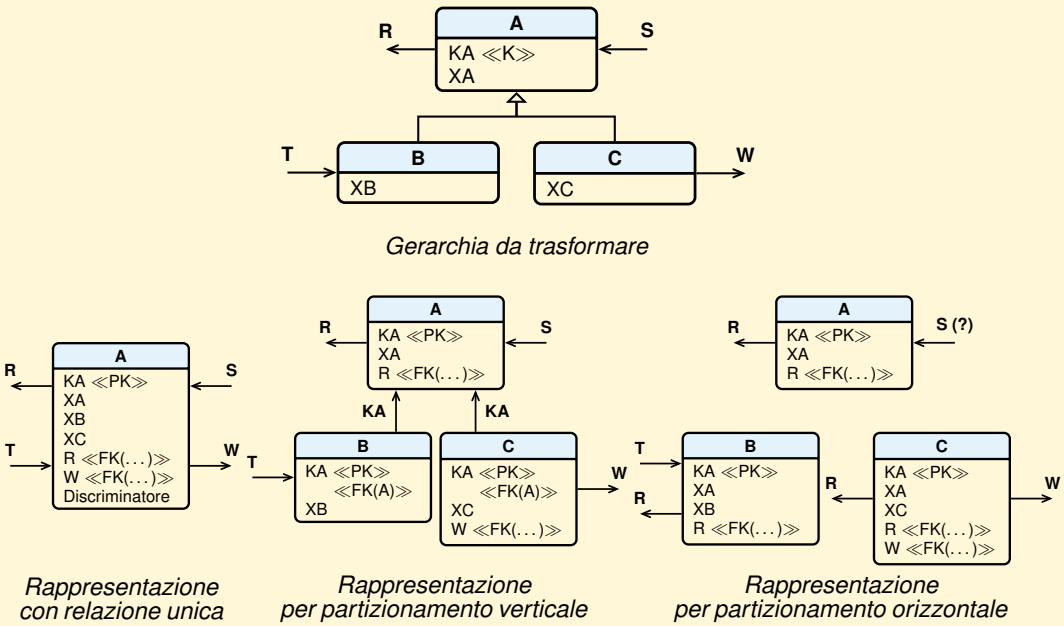


Figura 4.8: Regole per il **PASSO 3** di trasformazione

Applicando questa trasformazione alla Figura 4.7, si ha la Figura 4.9. Per evitare l'uso di attributi con valore null le sottoclassi immediate delle classi Libri e Utenti sono state rappresentate sfruttando il partizionamento verticale, mentre le sottoclassi di Prestiti sono state rappresentate con il partizionamento orizzontale (senza la superclasse perché le due sottoclassi sono una copertura).

4.2.4 PASSO 4: Definizione delle chiavi primarie

Il quarto passo della trasformazione è il primo che non può essere eseguito sullo schema grafico ma va eseguito a partire da un'analisi degli attributi.

In questa fase si deve definire, per ognuna delle relazioni dello schema, un insieme di attributi che funga da chiave primaria.

Per prima cosa, si considerano le relazioni che corrispondono a classi dello schema originale che erano prive di superclassi (classi *radice*). La chiave primaria è di norma un attributo artificiale, tipicamente un numero progressivo assegnato dal sistema. È possibile utilizzare un attributo presente nella classe, purché l'attributo sia univoco, totale e costante. Nei nostri esempi, per semplicità, useremo questo approccio.

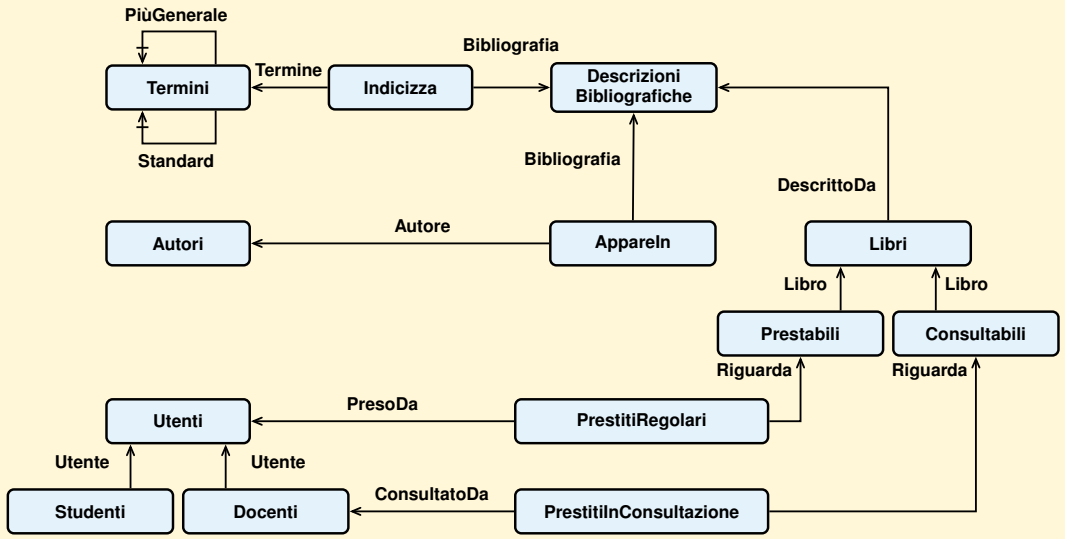


Figura 4.9: Effetto del **PASSO 3** di trasformazione

Poi, per ogni relazione dello schema che corrisponde ad una sottoclasse dello schema originario, la chiave primaria sarà la stessa della superclasse. Infine, per le relazioni che corrispondono ad associazioni molti a molti nello schema originario, la chiave primaria sarà costituita dalla concatenazione delle chiavi esterne.

L'inserimento di una chiave primaria in ogni collezione trasforma effettivamente tutte le classi in relazioni in senso matematico, ovvero in collezioni dove non possono esistere due elementi diversi che coincidono su tutti gli attributi. Tuttavia, per soddisfare il modello relazionale, è necessario eliminare da queste relazioni gli attributi multivalore e strutturati.

In Figura 4.10 è mostrato la versione dello schema di esempio con gli attributi, le chiavi primarie e le chiavi esterne.

4.2.5 PASSO 5: Rappresentazione degli attributi multivalore

Una proprietà multivalore di una classe C si rappresenta eliminando il corrispondente attributo da C e creando una nuova relazione N con una *chiave di due attributi*: una chiave esterna che fa riferimento alla chiave primaria di C ed un attributo che corrisponde all'attributo multivalore da trasformare. Un oggetto di C con chiave primaria k ed in cui l'attributo assume valore $a_1, \ldots, a_n$ si rappresenta poi inserendo nella nuova relazione N le coppie $(k, a_1), \ldots, (k, a_n)$.

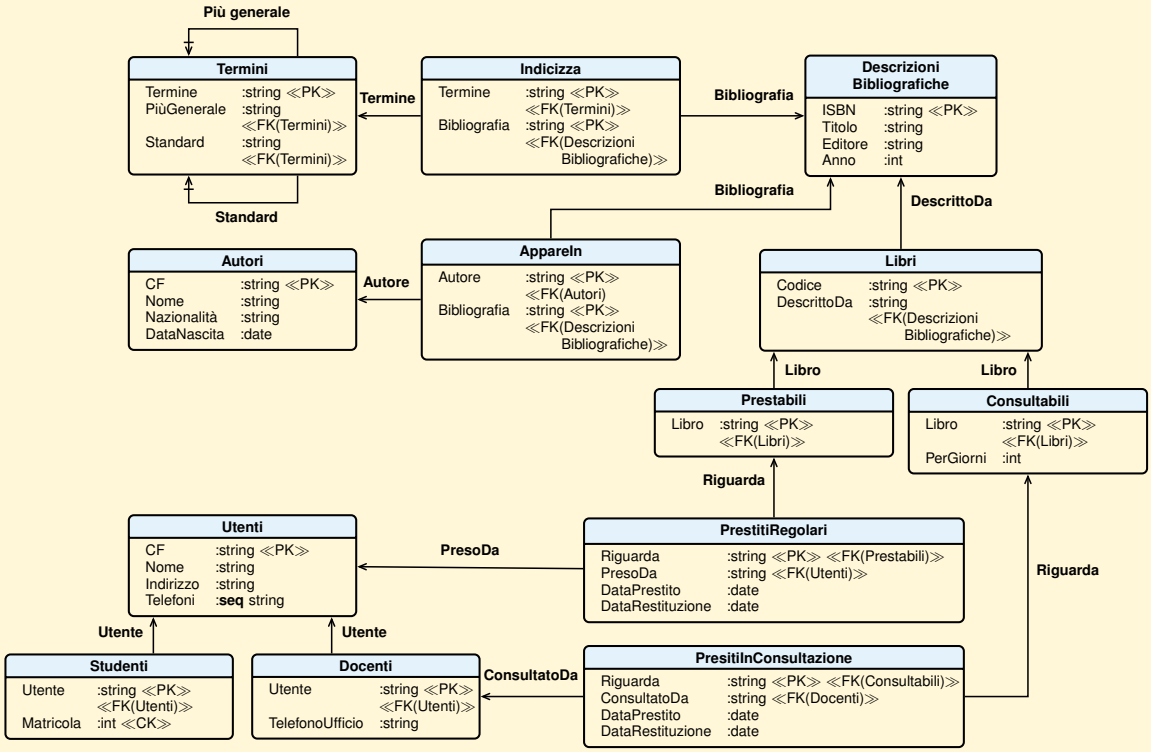
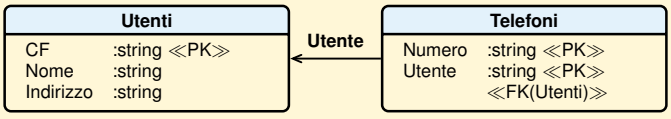


Figura 4.10: Lo schema dell'esempio con gli attributi

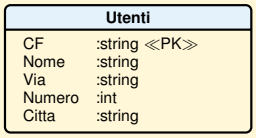
Ad esempio, si consideri la classe Utenti con l'attributo multivalore Telefoni. Con la trasformazione, si ottengono le due seguenti relazioni:



4.2.6 PASSO 6: Appiattimento degli attributi composti

Se un attributo A_i di uno schema di relazione è di tipo $[A_{i1} : T_{i1}, \ldots, A_{ij} : T_{ij}]$, si sostituisce con gli attributi $A_{i1} : T_{i1}, \ldots, A_{ij} : T_{ij}$. Se A_i faceva parte della chiave primaria dello schema di relazione, si sostituisce A_i con gli attributi $A_{i1}, \ldots, A_{ij}$ nella chiave, e poi si verifica che non esista un sottoinsieme degli attributi della nuova chiave primaria che è esso stesso una chiave.

Ad esempio, se l'attributo Indirizzo di utenti invece che String fosse definito composto da tre attributi [Via :string, Numero: int, Citta :string] applicando la trasformazione alla relazione Utenti si otterrebbe:



Al termine di questo passo, lo schema in Figura 4.11 rappresenta uno schema relazionale che equivale a quello ad oggetti di partenza, se non per il fatto che alcuni vincoli sono andati perduti.

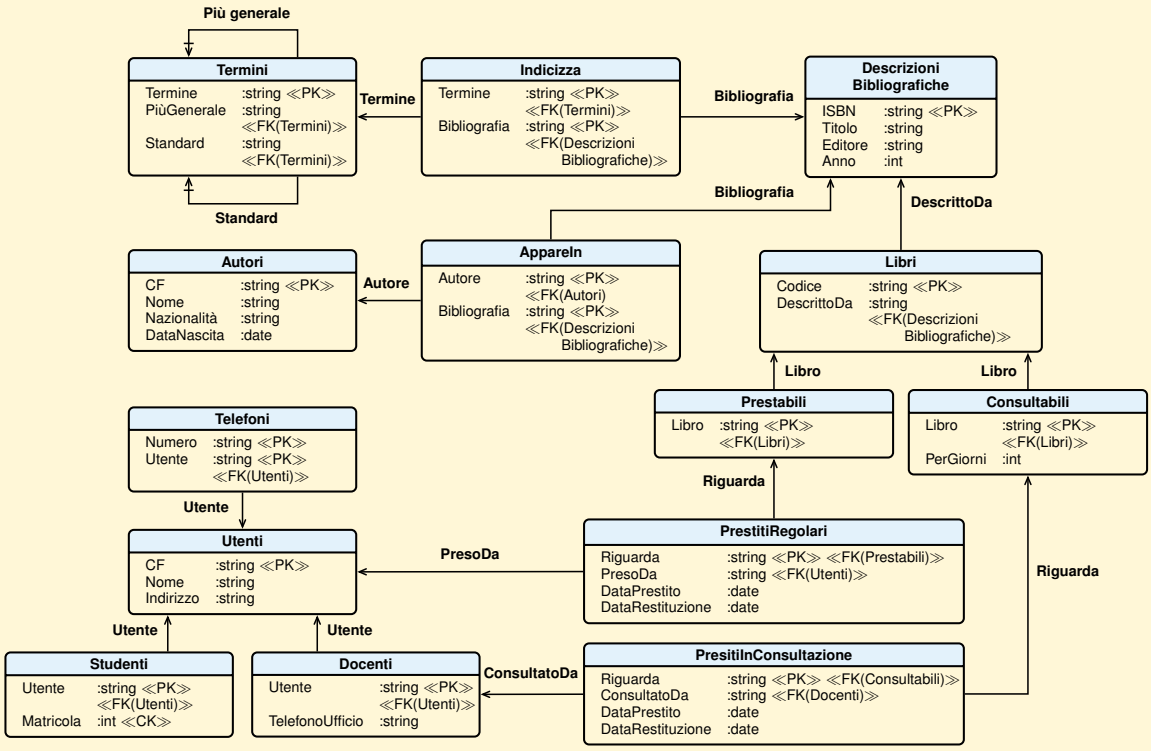


Figura 4.11: Lo schema relazionale finale per la biblioteca