Azzolini Riccardo 2019-11-26

Progettazione logica

1 Progettazione logica

La progettazione concettuale produce uno schema concettuale (ad esempio, un diagramma ER), che rappresenta i dati di interesse in modo formale, non ambiguo, e ad alto livello, indipendentemente dal tipo di DBMS.

Si svolge poi la progettazione logica, che ha l'obiettivo di tradurre questo schema concettuale in uno schema logico equivalente (cioè con la stessa capacità di rappresentazione delle informazioni) per il DBMS scelto. Essa si articola in due fasi:

1. **ristrutturazione** dello schema concettuale;
2. **traduzione** dello schema concettuale ristrutturato in uno schema logico.

2 Fase di ristrutturazione

La fase di ristrutturazione genera uno **schema ER ristrutturato**, equivalente a quello di partenza, al fine di semplificarne la successiva traduzione:

- si eliminano i costrutti non direttamente rappresentabili nel modello logico del DBMS, che, nel caso del modello relazionale, sono:
  - attributi composti,
  - attributi multi-valore,
  - gerarchie di generalizzazione;
- si effettuano ulteriori ristrutturazioni che tengano conto degli aspetti relativi alle prestazioni, in base a un'analisi del carico di lavoro.

Siccome dipende dal carico di lavoro, e da altre considerazioni sulla realizzazione della base di dati, la ristrutturazione non è sempre univoca. Inoltre, lo schema risultante non è più un vero schema "concettuale", poiché tiene conto, appunto, di aspetti realizzativi.

3 Fase di traduzione

Nella fase di traduzione, lo schema ER ristrutturato viene tradotto in un equivalente schema logico (ad esempio, relazionale), mediante l'applicazione di una serie di regole di trasformazione per i vari costrutti del modello ER.

Come la ristrutturazione, anche la traduzione non è sempre univoca: quando sono possibili più soluzioni, la scelta dipende dal carico di lavoro.

4 Output della progettazione logica

L'attività di progettazione logica produce, come output:

- lo schema logico, compresi i vari vincoli, ecc.;
- la documentazione, contenente dizionari, informazioni sulle scelte progettuali (ad esempio, quelle relative alla ristrutturazione e traduzione), ecc.

5 Eliminazione degli attributi composti

Esistono due soluzioni per eliminare un attributo composto A da un'entità E:

1. eliminare i sotto-attributi di A, "collassandoli" in un singolo attributo semplice;
2. considerare i sotto-attributi di A come direttamente attributi di E (cioè "scomporre" A nei suoi sotto-attributi).

Inoltre, indipendentemente dalla soluzione adottata:

- eventuali vincoli di cardinalità per l'attributo composto vengono riportati sui nuovi attributi generati tramite la ristrutturazione;
- se i sotto-attributi sono a loro volta composti, si riapplica a cascata la procedura di ristrutturazione.

Entrambe queste soluzioni permettono di rappresentare tutti i dati che potevano essere rappresentati dall'attributo composto, ma hanno vantaggi/svantaggi diversi:

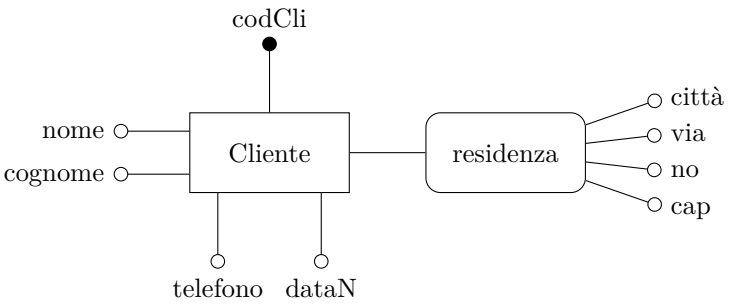
- Soluzione 1:
  - i vincoli sui contenuti dei sotto-attributi nella nuova rappresentazione "collassata" devono essere garantiti dall'applicazione;
  - è adatta se è sufficiente avere accesso all'informazione complessiva.
- Soluzione 2:

– si perde il collegamento logico tra i sotto-attributi (ma, solitamente, si cerca di suggerirlo attraverso i nomi degli attributi generati);

– è adatta se è necessario accedere separatamente a ciascun sotto-attributo, (ad esempio, per effettuare delle ricerche in base ai valori di alcuni di essi).

5.1 Esempio

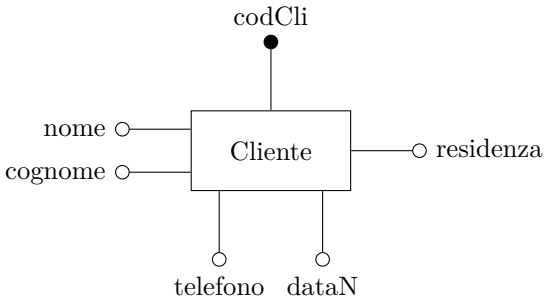
Nel diagramma seguente, l'entità Cliente ha un attributo composto, residenza:



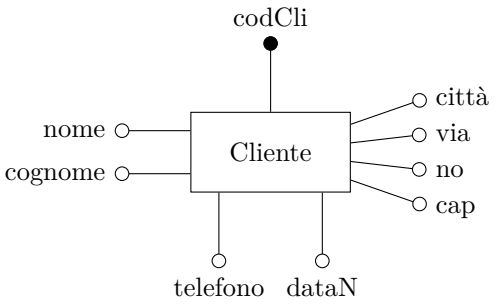
I domini dei sotto-attributi di residenza sono:

$$\text{dom(città)} = \text{string}$$
$$\text{dom(via)} = \text{string}$$
$$\text{dom(no)} = \text{string}$$
$$\text{dom(cap)} = \text{string}$$

Applicando la soluzione 1, residenza diventa un attributo semplice, con dom(residenza) = string:



Invece, con la soluzione 2, si ottiene il diagramma



e i domini rimangono invariati.

6 Eliminazione degli attributi multi-valore

Per eliminare un attributo multi-valore, si definisce una nuova entità, dotata di un attributo mono-valore che:

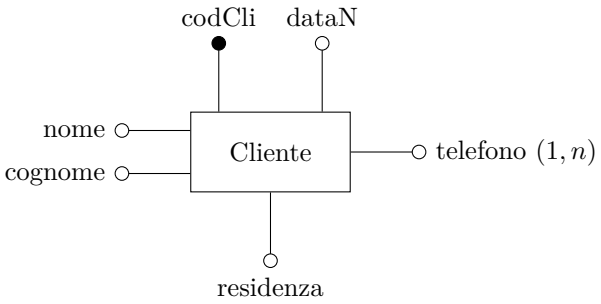
- contiene uno dei valori dell'attributo multi-valore originale;
- identifica l'entità.

Questa nuova entità è legata a quella di partenza da un'apposita associazione, alla quale:

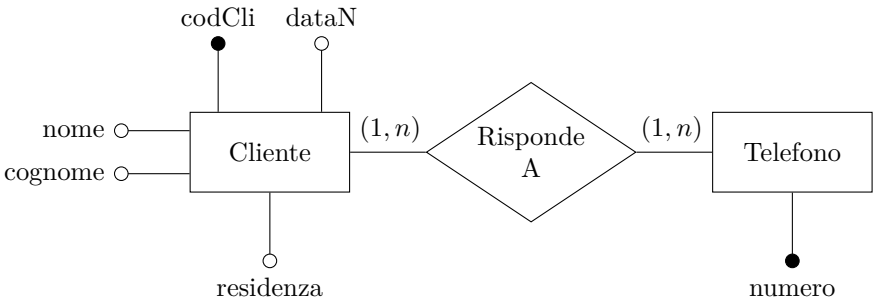
- l'entità originale partecipa con una cardinalità corrispondente a quella dell'attributo multi-valore;
- l'entità generata partecipa con una cardinalità che dipende dal dominio.

6.1 Esempio

Data l'entità Cliente, con l'attributo multi-valore telefono,



il diagramma ristrutturato corrispondente è



dove la cardinalità $(1, n)$ con la quale il cliente partecipa all'associazione corrisponde alla cardinalità originale dell'attributo multi-valore. Anche l'entità Telefono partecipa con cardinalità $(1, n)$, perché:

- ogni numero di telefono è associato ad almeno un cliente;
- uno stesso numero di telefono può essere condiviso da più clienti.

7 Eliminazione delle gerarchie di generalizzazione

Le gerarchie di generalizzazione vengono sostituite da entità e associazioni. Esistono tre opzioni diverse per come effettuare tale ristrutturazione:

- *eliminazione delle entità figlie* (accorpamento nell'entità padre);
- *eliminazione dell'entità padre* (accorpamento nelle entità figlie);
- *sostituzione della generalizzazione con delle associazioni.*

La scelta tra di esse dipende sia dal carico di lavoro, che dal tipo di gerarchia (perché l'eliminazione del padre non è sempre possibile).

7.1 Eliminazione delle entità figlie

Le entità figlie vengono eliminate, e i loro attributi vengono inseriti nell'entità padre, come attributi *opzionali*. Inoltre, viene aggiunto al padre anche un attributo *tipo*, che permette di distinguere a quali entità figlie appartiene ogni istanza:[1]

- per le generalizzazioni totali, è un attributo obbligatorio (non può mai assumere un valore nullo);
- per le generalizzazioni parziali, è opzionale (cioè può assumere un valore nullo);

[1] A volte, in fase di ottimizzazione, l'attributo tipo può essere rimosso, perché le entità figlie a cui appartiene un'istanza possono essere dedotte dai valori degli attributi (ad esempio, osservando quali di essi hanno valori nulli, purché tali attributi non fossero già opzionali nelle entità figlie, prima della ristrutturazione).

- per le generalizzazioni esclusive, è mono-valore;
- per le generalizzazioni condivise, è multi-valore.

In seguito, è necessario definire (nella documentazione) dei vincoli di integrità, per imporre che i valori degli attributi aggiunti all'entità padre rispettino il tipo dell'istanza e il tipo di generalizzazione:

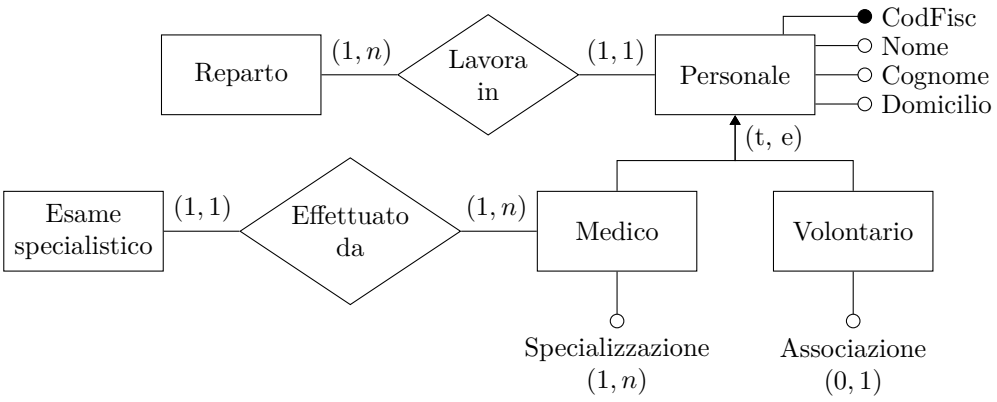
- se la generalizzazione è totale, devono assumere valori non nulli gli attributi di almeno un'entità figlia;
- se la generalizzazione è esclusiva, possono assumere valori non nulli gli attributi di al più un'entità figlia.

In pratica, si rendono espliciti i vincoli che erano stabiliti implicitamente dalla gerarchia.

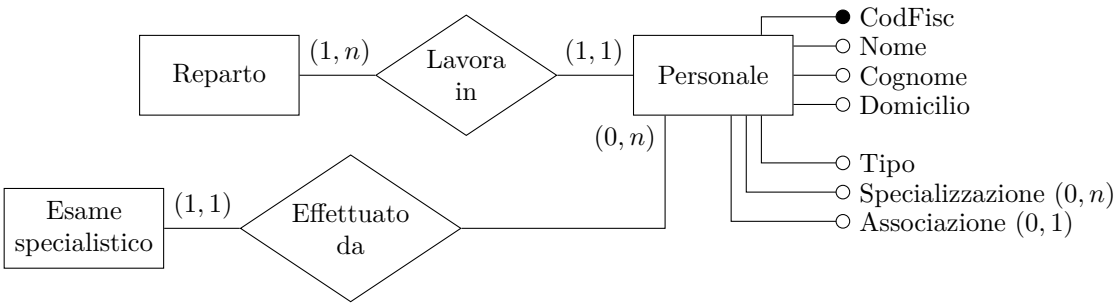
Come gli attributi, anche le associazioni delle entità figlie vengono "spostate" sul padre, e la partecipazione diventa opzionale (se rimanesse obbligatoria, dovrebbero partecipare tutte le istanze del padre, nonostante l'associazione originale riguardasse solo una delle figlie). Si aggiungono poi ulteriori vincoli, per specificare quali tipi di istanze possono partecipare a tali associazione.

7.1.1 Esempio

Dato il seguente diagramma ER,



nel quale (t, e) indica una generalizzazione totale ed esclusiva, eliminando le entità figlie si ottiene questo diagramma ristrutturato:



Inoltre, vengono definiti alcuni vincoli per garantire il rispetto del tipo di generalizzazione:

- V_1: per ogni istanza di Personale di tipo Medico, deve essere specificata almeno una Specializzazione, mentre l'Associazione non deve essere specificata.
- V_2: per ogni istanza di Personale di tipo Volontario, non deve essere specificata alcuna Specializzazione (mentre l'Associazione è opzionale).
- V_3: possono partecipare all'associazione "Effettuato da" solo le istanze di Personale di tipo Medico, e ciascuna di esse deve obbligatoriamente partecipare ad almeno un'istanza di tale associazione.

7.2 Eliminazione dell'entità padre

Questa soluzione è applicabile solo per le generalizzazioni totali: eliminando l'entità padre, si perde infatti la possibilità di rappresentare le istanze del padre che non sono anche istanze di almeno una delle figlie.

Gli attributi dell'entità padre vengono inserite in ciascuna delle entità figlie.

Se la generalizzazione è condivisa, un'istanza appartenente a più figlie viene rappresentata attraverso istanze separate delle varie entità figlie, aventi gli stessi valori per gli identificatori. Viceversa, se la generalizzazione è esclusiva, si aggiunge un vincolo di integrità, per indicare che non possono esistere istanze di due entità figlie con valori degli identificatori uguali.

Ogni associazione a cui partecipava il padre viene sostituita con n nuove associazioni, una per ogni entità figlia:

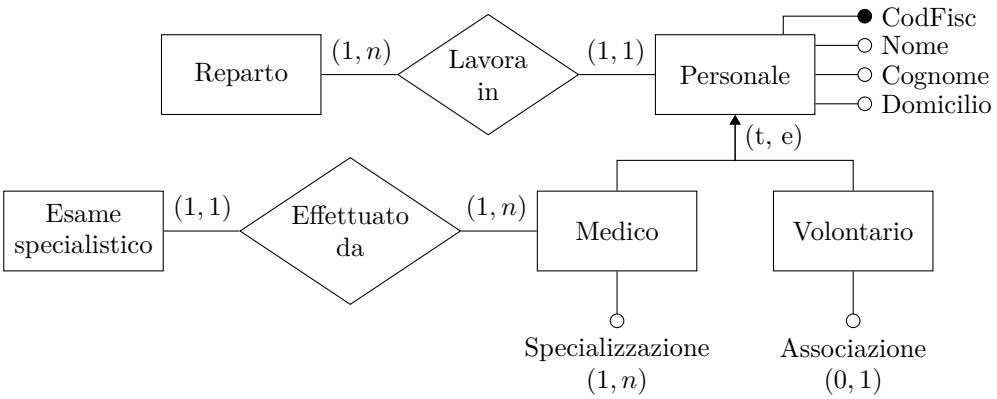
- le figlie mantengono la stessa cardinalità con la quale partecipava il padre;
- per l'entità dall'altro lato dell'associazione, la partecipazione alle associazioni generate diventa opzionale[2] (altrimenti, ogni istanza di tale entità, che prima era

[2]Se il documento di specifica contiene informazioni aggiuntive sull'associazione di quest'entità con le varie figlie, la partecipazione ad alcune delle nuove associazioni (o anche a tutte) potrebbe rimanere obbligatoria.

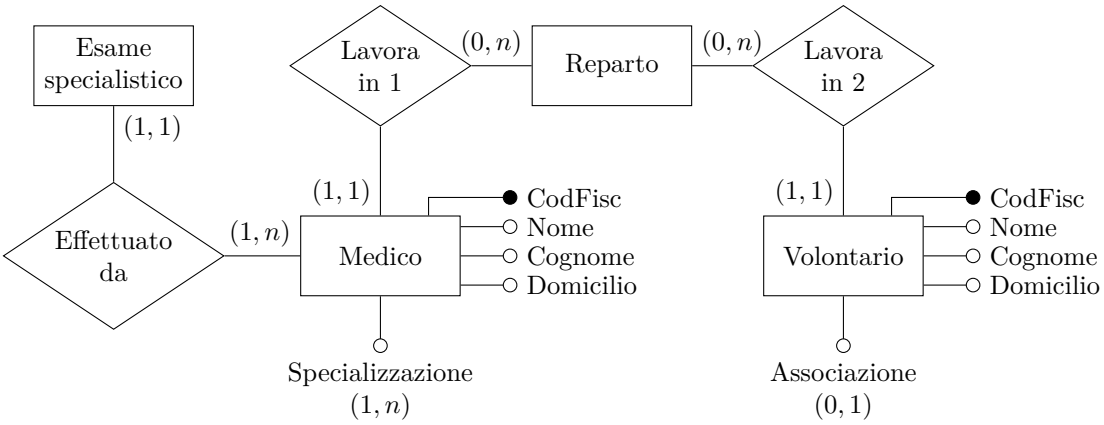
obbligata a essere associata almeno a un'istanza del padre, dovrebbe per forza associarsi almeno a un'istanza di *ciascun tipo* di figlia, cioè, in totale, almeno a n istanze delle figlie), e, se prima era obbligatoria, ciò viene specificato mediante l'aggiunta di un vincolo nella documentazione.

7.2.1 Esempio

Eliminando l'entità padre, il diagramma



diventa



e vengono aggiunti i seguenti vincoli di integrità:

- V_1: per ogni Medico, non può esistere un Volontario con lo stesso CodFisc, e viceversa.
- V_2: ogni istanza di Reparto deve obbligatoriamente partecipare almeno a una delle due associazioni "Lavora in".

In questo caso, la partecipazione di Reparto a entrambe le associazioni "Lavora in" è diventata opzionale, come avviene in generale. Se invece, ad esempio, le specifiche indicassero che in ogni reparto lavora almeno un medico, allora l'entità Reparto potrebbe avere partecipazione obbligatoria a "Lavora in 1" (e non sarebbe più necessario aggiungere esplicitamente il vincolo V_2).

7.3 Sostituzione della generalizzazione con delle associazioni

Le entità e associazioni esistenti non vengono modificate, e la gerarchia viene sostituita da n associazioni uno a uno, ciascuna delle quali lega l'entità padre a una delle figlie. Per ognuna di queste associazioni:

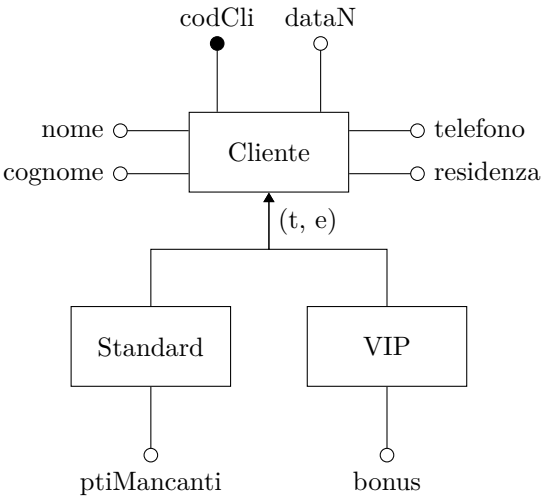
- l'entità figlia partecipa sempre con cardinalità $(1, 1)$, ed è identificata esternamente dall'entità padre (dato che, in una gerarchia, le entità figlie non hanno identificatori propri, ma sfruttano invece quelli del padre, che ereditano);
- l'entità padre partecipa con cardinalità $(0, 1)$, anche se la generalizzazione è totale (perché, comunque, un'istanza del padre potrebbe essere associata solo ad alcune delle figlie, non per forza a tutte).

Poi, a seconda del tipo di gerarchia, si aggiungono alla documentazione dei vincoli di integrità:

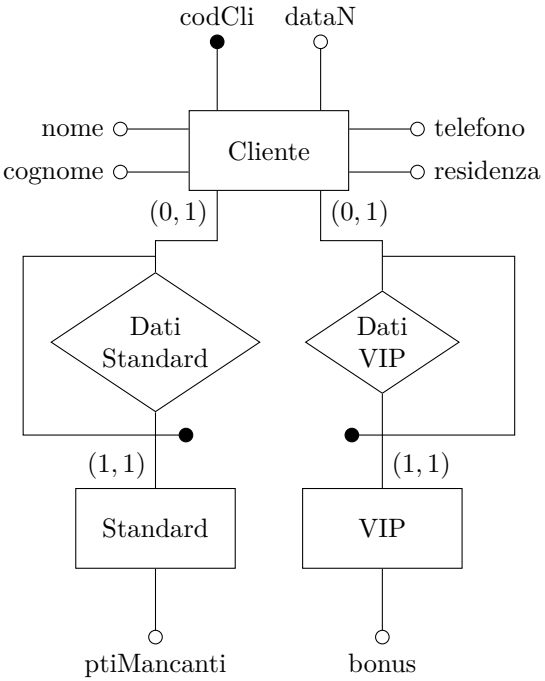
- se la generalizzazione è totale, ogni istanza del padre deve partecipare ad almeno una delle associazioni con le figlie;
- se la generalizzazione è esclusiva, un'istanza del padre può partecipare al massimo a una delle associazioni con le figlie.

7.3.1 Esempio

Dato il diagramma ER seguente,



sostituendo la generalizzazione (totale ed esclusiva) con delle associazioni si ottiene il diagramma ristrutturato



che deve essere accompagnato dal seguente vincolo di integrità, derivato dal tipo di generalizzazione:

- V_1: ogni cliente è alternativamente un cliente standard oppure un cliente VIP.

7.4 Valutazione delle alternative

Eliminazione delle entità figlie:

- Comporta uno spreco di memoria, dovuto alla presenza di valori nulli.
- È appropriata quando:
  - le entità figlie hanno pochi attributi (in questo caso, ci sono pochi valori nulli);
  - la maggior parte delle operazioni di accesso non distinguono tra occorrenze dell'entità padre e delle figlie (così, esse possono semplicemente accedere a una singola entità, il padre, invece che separatamente alle figlie, e diventano allora più efficienti).

Eliminazione dell'entità padre:

- Si può applicare solo per le generalizzazioni totali.
- Non avendo valori nulli, si risparmia memoria rispetto alla soluzione precedente.
- È appropriata quando le operazioni di accesso distinguono spesso tra le occorrenze delle diverse entità figlie: con questa soluzione, tali operazioni diventano più efficienti in quanto possono accedere direttamente alle istanze di determinate figlie, invece di doverle estrarre dall'insieme di tutte le istanze del padre. Viceversa, le operazioni che non fanno distinzioni del genere devono essere replicate separatamente su tutte le entità figlie, e risultano allora meno efficienti.

Sostituzione con delle associazioni:

- Si ha ancora un risparmio di memoria rispetto all'eliminazione delle figlie, data l'assenza di valori nulli.
- È la soluzione più generale, sempre applicabile.
- Può essere dispendioso ricostruire l'informazione di partenza, "completa", costituita sia dagli attributi di un'entità figlia che da quelli del padre (nel modello relazionale, ciò richiede un join).

In alcune situazioni, può essere utile adottare una soluzione ibrida: ad esempio, eliminare alcune delle entità figlie, collassandole nel padre, e mantenere le altre nello schema, mediante delle associazioni.

Infine, nel caso di una generalizzazione a più livelli, si applicano queste strategie su un livello alla volta, partendo dalle foglie (le entità che non hanno ulteriori figlie); per ogni livello, è possibile applicare una strategia diversa.