Progettazione di database relazionali

28 dicembre 2008

Sommario

Lista delle Figure

Lista delle Tabelle

Lista degli Esempi

Introduzione

Questo documento è un supporto per chi inzia a studiare i database relazionali, o basi di dati, e la loro progettazione.

Per la progettazione si seguirà una metodologia che procede attraverso una sequenza di livelli di astrazione. Verrano visti anche degli esempi, rappresentando semplici tabelle su un foglio bianco, ma senza realizzare un «vero» database elettronico; l'unico software che verrà usato sarà quello che permette di preparare il progetto del database attraverso la produzione di schemi grafici, e cioè DIA (http://projects.gnome.org/dia).

Qualsiasi contributo, suggerimento e segnalazione sono benvenuti a questo indirizzo: fabio PUNTO proietti CHIOCCIOLA istruzione PUNTO it

Capitolo 1. Terminologia e definizioni

Sommario

Dato e informazione

Anche se questa può sembrare un'affermazione banale, ricordiamo che l'importanza delle informazioni è dovuta al fatto che esse possono essere archiviate in modo permanente e ritrovate quando devono essere utilizzate. Il vantaggio che se ne trae è quello di poter ottenere, dall'elaborazione dei dati ritrovati, informazioni che potrebbero rivelarsi indispensabili o strategiche nelle decisioni.

Anche se potrebbero sembrare due sinonimi, i termini *dato* e *informazione* hanno un significato diverso:

dato

: è un valore astratto, per esempio un numero, che per essere utilizzato deve essere interpretato, ad esempio, il numero 21;

informazione

: è il significato contenuto nel dato che viene alla luce quando il dato viene interpretato. Ad esempio, il precedente numero 21 assume significati diversi se si parla di una classe composta da 21 alunni o dei 21 cm che misura il lato di un foglio A4.

Archivi tradizionali e database

È molto importante decidere il modo in cui i dati vengono conservati perchè ciò determinerà successivamente anche il modo in cui i dati potranno essere estratti; ad esempio, potrebbe capitare che non preoccupandosi dell'organizzazione dei dati all'interno degli archivi durante la progettazione , si abbiano difficoltà o complicazioni eccessive durante la fase del loro recupero.

Mentre una volta gli archivi potevano essere su supporto cartaceo, oggi quasi tutti gli archivi si trovano in forma digitale su supporto elettronico, e ciò rende possibile la loro elaborazione automatizzata. Per esempio, un archivio può avere la forma di un file di dati opportunamente organizzati.

Utilizzare un singolo file per creare un archivio di dati può essere pratico per piccole applicazioni, ma in generale, questo approccio, può dar luogo a diversi problemi. Per esempio: se una Banca utilizza un file per conservare i dati dei propri clienti e un file per conservare i dati dei relativi conti correnti, le modifiche agli eventuali dati in comune tra i due file devono essere fatte in modo «sincronizzato». Se la Banca permette l'aggiornamento degli archivi da parte di più filiali, i problemi aumentano.

I database nascono proprio per superare i limiti e gli inevitabili problemi che si potrebbero incontrare usando gli archivi tradizionali. Tra i classici problemi vanno ricordati quello della *ridondanza* dei dati e dell'*inconsistenza* degli archivi. Il primo si verifica quando in diversi archivi si trovano memorizzati gli stessi tipi di dati (il problema è quello di dover fare gli aggiornamenti in tutti gli archivi interessati). Il secondo si verifica come conseguenza del primo, quando questi aggiornamenti non vengono fatti in tutti gli archivi in cui si dovrebbe, e ciò provoca incongruenze tra dati nuovi e dati vecchi.

Ricordiamo il significato di alcuni termini:

archivio

> oggetto memorizzato in modo permanente, costituito da una sequenza di elementi dello stesso tipo e dimensione (record), ognuno dei quali può essere composto da più campi, di tipo e dimensione diversi tra loro;

database

> insieme di archivi integrati (ovvero, collegati) secondo un «modello logico» e nel rispetto di determinate regole (ovvero, vincoli).

Gestione degli archivi e DBMS

Un'altra differenza tra gli archivi tradizionali e i database si trova nei software utilizzati per la loro gestione.

I software per la gestione degli archivi sono quei programmi utili per poter inserire, modificare ed eliminare un dato nell'archivio; i programmatori di solito usano in questi casi le funzioni (o istruzioni) per eseguire operazioni di input/output sui File. Un programmatore che scrive programmi di questo tipo (di solito in linguaggio procedurale) sa bene che oltre a pensare alle attività dell'utente, si deve preoccupare anche di: diritti di accesso ai file, operazioni di lettura, scrittura, ecc.

I database invece sono gestiti tramite sofware più complessi chiamati DBMS; sono software che hanno già integrate tutte le principali funzioni necessarie e che possono essere «personalizzati» al caso particolare in cui verranno applicati. Ne sono esempi: PostgreSQL, MySQL, Oracle, Informix, MS Access, HSQLDB, ecc. Alcune delle funzioni integrate nei DBMS sono: la possibilità di integrazione (ovvero, di collegamento) degli archivi, l'integrità dei modelli di dati, la capacità di gestire database accentrati o distribuiti, la gestione della multiutenza e della sicurezza, la gestione dell'accesso concorrente ai dati, la gestione delle transazioni, la gestione dei backup, ecc...

Poichè queste funzioni sono integrate nel DBMS, il lavoro del programmatore risulta alleggerito; il programmatore che lavora con i DBMS non utilizza però solo i classici linguaggi di programmazione procedurali(es: Pascal), ma anche linguaggi di programmazione dichiarativi (es: SQL).

La spiegazione delle differenze tra linguaggi procedurali e dichiarativi non è oggetto di questo documento.

Esercizio

Dopo aver studiato e compreso il significato dei termini visti in questo capitolo, rispondere alle seguenti domande:

Qual è la differenza tra dato e informazione?

Quali problemi si possono incontrare utilizzando gli archivi tradizionali?

Cosa è un DBMS?

Citare i nomi di alcuni DBMS.

Capitolo 2. Livello concettuale

Sommario

Livelli di astrazione

Quando si deve affrontare un problema complesso come quello del progetto di un database è conveniente scomporre il procedimento in 3 fasi, ovvero affrontare il problema in 3 livelli di astrazione, dal più astratto al più concreto.

Livello concettuale

> di solito il problema della realizzazione di un database viene sottoposto da parte di un cliente, il quale cerca di descrivere a parole i suoi bisogni (il progettista procede come in un'intervista); bisogna tenere conto che il problema può essere descritto in forma vaga, o su un argomento su cui non si è compentenenti. Per questo motivo la prima analisi consiste nella ricerca, all'interno della descrizione del problema, dei termini piu' significativi, con gli eventuali sinonimi, che indicano quali sono i contenuti del database che si vuole costruire. È utile scrivere questi termini e spiegare il loro significato, come in una specie di vocabolario. Al termine di questa fase viene prodotto uno schema chiamato Schema E-R che rappresenta in forma grafica i collegamenti tra gli oggetti astratti individuati. Lo Schema E-R è si basa su un modello teorico matematico, che vedremo più avanti.

Livello logico

> In questa fase si parte dallo schema E-R ottenuto nel livello precedente, per ottenere un altro schema, detto anche schema logico. Questo schema spiega concretamente come realizzare il vero e proprio database cioè gli archivi e i *collegamenti* tra gli archivi. A seconda di come si preferisca realizzare questi collegamenti, si può adottare un modello a scelta tra i seguenti: modello relazionale, modello gerarchico, modello reticolare e modello ad oggetti. Si vedrannno più avanti questi modelli, in particolare il modello logico relazionale e come esso realizza i collegamenti tra gli archivi.

Livello fisico

> In questa fase il problema è quello di individuare la forma migliore in cui salvare i dati che costituiscono gli archivi. Per fare ciò è necessario rispondere a domande come:
>
> - Su quale tipo di supporto fisico verranno salvati i dati?
> - Che struttura logica avranno i dati nei file?
> - Quale file system si utilizzerà?
>
> Quest'ultima fase di solito è influenzata, oltre che dai requisiti del progetto, anche dal DBMS sul quale è caduta la scelta, e dal file system usato, e non sarà trattata.

Esempio 2.1. Gli alunni e le classi

Pre comprendere pienamente in cosa consistono le tre fasi della progettazione, si può fare un esempio di progetto; in questo livello di studio non si pretende di risolvere il problema, ma piuttosto di capire in che modo esso può essere affrontato.

Testo del problema

Si vuole realizzare il progetto di un database per un'istituto di istruzione secondaria. È necessario poter memorizzare nel computer della segreteria dell'istituto i dati relativi agli alunni iscritti (cognome, nome, matricola, telefono, data di nascita) e alle classi a cui appartengono (sezione, anno di frequenza ed eventuale indirizzo di studio).

Soluzione del problema

Il problema si deve considerare risolto quando si ottiene la descrizione delle tabelle che potrebbero contenere i dati di cui si parla nel testo. Sarebbe poco didattico procedere a costruire delle tabelle per tentativi. Esiste un procedimento logico che permette di procedere per livelli di semplificazione.

1. **Livello concettuale**

   Si devono individuare nel testo i termini di particolare importanza; questi termini possono essere anche sinonimi tra loro oppure possono non avere rilevanza per il progetto, e in questo caso vanno scartati.

   1. **Vocabolario dei termini**

      Prima di tutto si crea un *vocabolario dei termini significativi* : il lettore è fortemente invitato a rileggere il testo, a sottolineare da solo i termini che ritiene più significativi e a darne una brevissima descrizione:

      > **Nota**
      >
      > Nel livello concettuale i nomi delle categorie individuate sono preferibilmente al singolare.

      istituto (di istruzione)

      > la scuola che commissiona il progetto: poichè *non* si devono registrare informazioni sulla scuola, questo termine è irrilevante per il progetto; non è una categoria;

alunno (iscritto)

è la categoria che indica tutti gli individui attualmente iscritti a tale istituto, e che sono caratterizzati da una *matricola*, oltre che dal nome, cognome, data di nascita e dal telefono;

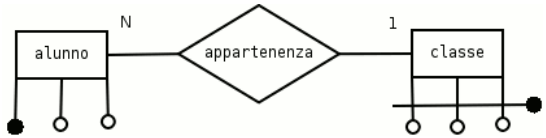
classe

è la categoria che descrive gruppi di alunni omogenei, da non confondere con l'aula dove si svolgono le lezioni. Ogni classe è individuata da un numero intero, una lettera (la sezione) e un indirizzo di studio. Esiste un particolare legame con la precedente categoria, in quanto un alunno può appartenere ad una sola classe. Chiameremo questo legame Appartenenza.

2. **Schema E-R**

A seconda delle categorie che sono state precedentemente individuate nel vocabolario e a seconda dei legami che intercorrono tra esse, si realizza uno schema chiamato schema E-R. Lo schema E-R che segue ha solo uno scopo illustrativo, poichè a questo livello di studio sarebbe inutile spiegare come ottenere tale schema.

Figura 2.1. Lo schema E-R dell'esempio Gli alunni e le classi



> **Nota**
>
> La parte che segue della soluzione è riportata solo per concludere l'esempio in questo paragrafo: si suggerisce perciò di trascurarla momentaneamente e riprenderla solo dopo aver affrontato il prossimo argomento.

In questo livello sono state individuate due categorie di argomenti e che saranno chiamate entità; siccome nello schema E-R ogni entità è rappresentata da un rettangolo, qui ci saranno due rettangoli, chiamati: *classe* e *alunno* .

All'interno del testo è sott'inteso che le due categorie sono legate dal fatto che ogni alunno può appartenere ad una sola classe; il progettista deve rappresentare esplicitamente anche questo legame aggiungendo un'associazione che chiameremo *appartenenza* e che rappresenteremo con un rombo. Bisogna essere molto precisi e spiegare in che modo l'alunno è legato alla classe e viceversa. Per farlo è utile scrivere delle frasi, chiamate *regole di lettura* per ogni associazione e quindi anche per *appartenenza* , e cioè:

- ogni alunno deve appartenere ad una sola (1) classe
- ad ogni classe devono appartenere più (N) alunni

Si tratta di un'associazione 1:N.

I cerchi vuoti e pieni sono gli attributi, che verranno spiegati più avanti.

2. **Livello logico**

A questo livello di studio non possiamo ancora affrontare la discussione del livello logico.

Entità

Entità

(in inglese Entity) rappresenta una categoria di elementi; per esempio, gli elementi che appartengono ad un entità alunno, sono tutti gli alunni. In questo livello di studio però non ci interessa individuare i singoli elementi, ma solo la loro categoria. Nel database tutti gli elementi appartenenti a quell'entità hanno delle caratteristiche in comune. Le caratteristiche in comune vengono chiamate attributi.

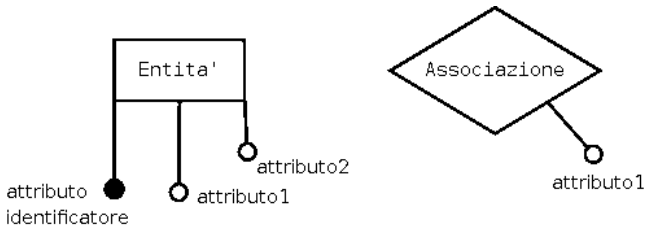
Attributi

sono caratteristiche che descrivono gli elementi di un'entità; per esempio: ogni alunno avrà un nome e un cognome. Gli attributi assumono valori diversi su ogni elemento che fa parte di quella entità, mentre il nome degli attributi è lo stesso per tutti gli elementi dell'entità. Per esempio: ogni alunno avrà sempre un nome e un cognome, ma avranno valori diversi da quelli degli altri alunni. Ci sono diversi tipi di attributi che verranno analizzati più avanti.

Schema E-R

E-R è l'abbreviazione di Entity-Relationship, cioè entità-associazione. Anche se ancora non è chiaro il concetto di associazione, si sappia che nello schema E-R viene riassunto tutto il lavoro fatto nella fase di progettazione a livello concettuale: le entità sono rappresentate come rettangoli, gli attributi come cerchi, le associazioni come rombi.

Figura 2.2. Elementi dello schema E-R



Associazioni

L'associazione (in inglese Relationship) descrive eventuali legami concettuali tra una, due o più entità. Per semplicità conviene considerare, per ora, solo le associazioni binarie, cioè quelle che legano solo due entità (che chiameremo *prima* entità e *seconda* entità).

Classificazione per molteplicità

Le associazioni possono essere classificate in base a quanti elementi appartenenti ad un'entità sono coinvolte nel legame stabilito con l'altra entità (caratteristica che in alcuni casi è chiamata moltemplicità dell'associazione). In particolare le associazioni possono essere dei seguenti tipi:

uno a molti

legano un solo elemento appartenente alla prima entità con più elementi della seconda entità; inoltre, viceversa, legano un solo elemento della seconda con uno della prima; rivedere a tal proposito l'esempio Gli alunni e le classi

molti a molti

legano un solo elemento appartenente alla prima entità con più elementi della seconda altra entità; viceversa, inoltre, legano un solo elemento della seconda con più elementi della prima;

uno a uno

legano un solo elemento appartenente alla prima entità con quello della seconda entità e viceversa.

Avvertimento

Le associazioni «molti a uno»*non* esistono perchè corrispondono alle associazioni uno a molti.

Rappresentazione

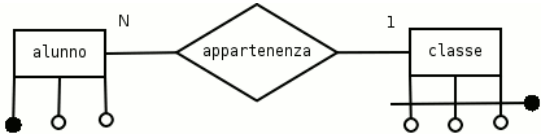
Nello schema E-R ogni tipo di associazione viene rappresentato sempre con il classico rombo, quello che cambia è solo l'aspetto del segmento che unisce l'associazione con le entità, ovvero, il rombo con il rettangolo.

Il numero (1 opppure N) vicino ad ogni entità legata all'associazione, indica la molteplicità con cui l'entità è coinvolta nel legame:

quando, nell'associazione, si parla di "molti elementi" non è necessario indicare precisamente quanti siano, basta sapere solo che ce ne sarà più di uno, scrivendo N;

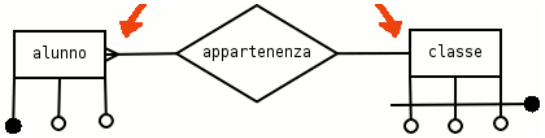
quando il legame coinvolge un solo elemento, basta scrivere 1.

Figura 2.3. Rappresentazione delle associazioni: dettaglio



In alternativa ai numeri si possono usare degli speciali simboli posti agli estremi dei segmenti, come indicato nell seguente figura.

Figura 2.4. Rappresentazione delle associazioni (alternativa)



Classificazione per opzionalità e rappresentazione

Oltre alla molteplicità bisogna stabilire se il legame dell'associazione sia obbligatorio oppure opzionale, per esempio: è necessario domandarsi se sia obbligatorio o meno che ogni alunno iscritto all'istituto faccia sempre parte di una classe, oppure, possano esistere nel database alcuni alunni che non appartengano a nessuna classe; inoltre, viceversa, se possano esistere nel database classi che non contengono nessun alunno.

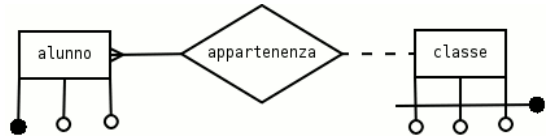
Esempio 2.2. C'è una classe vuota

Per continuare con lo stesso esempio, si ipotizza di aver ottenuto risposta affermativa all'ultima domanda.

> **Nota**
> anche se può sembrare poco realistico questa anomalia è stata introdotta solo per rendere l'esempio più completo dal punto di vista didattico

Anche questa caratteristica del legame viene rappresentata graficamente nello schema E-R, solitamente tratteggiando il segmento di unione tra l'associazione e l'entità coinvolta. In questo esempio l'associazione risulta essere obbligatoria-opzionale, ma esistono anche associazioni obbligatorie-obbligatorie oppure opzionale-opzionale.

Figura 2.5. Rappresentazione delle associazioni con opzionalità



Regole di lettura

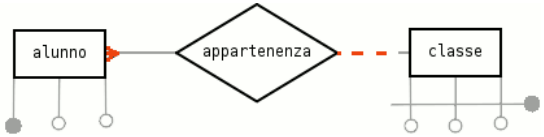
Come visto nell'esempio Gli alunni e le classi, oltre che per mezzo della rappresentazione grafica, le associazioni possono essere descritte anche in forma verbale (a parole) scrivendo le cosiddette *regole di lettura*; ricordiamo che da un'associazione *binaria* si ottengono sempre *due* regole di lettura. Analizzeremo l'associazione binaria del solito esempio Gli alunni e le classi appena perfezionato nella figura Rappresentazione delle associazioni con opzionalità ; per motivi didattici è conveniente oscurare momentaneamente in grigio alcune parti dello schema E-R ed evidenziarne altre in rosso:

Figura 2.6. Comprendere la prima regola di lettura

percorrendo la figura da sinistra verso destra, si può formulare la prima regola di lettura, mentre, percorrendo la figura successiva da destra verso sinistra, si può formulare la seconda regola di lettura:

- ogni alunno *deve* appartenere ad *una* sola classe
- ad ogni classe *possono* appartenere *uno o più* alunni

Figura 2.7. Comprendere la seconda regola di lettura



In ogni figura le parti evidenziate in rosso corrispondono proprio alle parole evidenziate nelle regole di lettura. Tali parole servono a porre l'attenzione su quei termini che differenziano sostanzialmente le due regole: il verbo *potere* usato al posto di *dovere*, per indicare l'opzionalità o l'obbligatorietà; il termine *una sola* usato al posto di *uno o più*, per indicare la molteplicità.

> ***Importante***
> *«ad ogni classe possono appartenere uno o più alunni» significa che ci possono essere classi alle quali non appartiene nessun alunno.*

> **Avvertimento**
> la seconda regola di lettura è leggermente diversa dalla formulazione originale fatta nell'esempio Gli alunni e le classi perchè è diverso anche lo schema E-R

> **Suggerimento**
> Per non scrivere regole di lettura sbagliate, basta ricordare di iniziare sempre con le parole *ogni* oppure *per ogni* e che le entità siano sostantivi al singolare

Associazioni unarie o ricorsive

Le associazioni unarie sono quelle che coinvolgono due volte la stessa entità e per questo motivo sono chiamate anche ricorsive; ci si accorge di dover usare queste particolari associazioni quando è necessario disegnare due volte la stessa entità nello stesso schema E-R. Segue un esempio.

Esempio 2.3. I vicini di banco

Testo del problema

Si vogliono registrare i dati degli alunni di una sola classe per sapere quali sono tra loro vicini di banco.

Soluzione del problema

1. **Livello concettuale**
   1. **Vocabolario dei termini**

      classe

      la classe dove è ambientato l'esempio: poichè *non* si devono registrare informazioni sulla classe, questo termine è irrilevante per il progetto; non è una categoria;

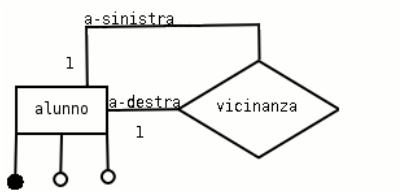
      alunno

      categoria che descrive tutti gli individui che si trovano nella classe; per distinguerli tra loro si può ipotizzare che possiedano una *matricola*, oltre al nome e al cognome;

   2. **Schema E-R**

      In questo livello è stata individuata una categoria (ovvero un'entità) chiamata *alunno*. All'interno del testo è chiaro che ogni alunno è legato ad un altro alunno e ciò verrà rappresentato aggiungendo un'associazione che chiameremo *vicinanza*. Sarebbe sbagliato indicare due volte la stessa categoria alunno, quindi si procede come in figura:

      Figura 2.8. Lo schema E-R de I vicini di banco



Per non fare confusione nelle *regole di lettura* si può usare uno pseudonimo diverso per l'entità alunno, per ogni verso di lettura dell'associazione: alunno-a-sinistra e alunno-a-destra.Dalle regole di lettura si conclude che si tratta di un'associazione uno a uno:

- ogni alunno-a-sinistra deve avere vicino un solo (1) alunno-a-destra
- ogni alunno-a-destra deve avere vicino un solo (1) alunno-a-sinistra

2. **Livello logico**

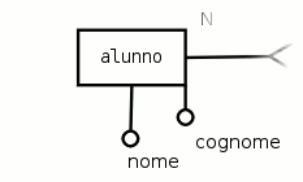
   A questo livello di studio non possiamo ancora affrontare la discussione del livello logico.

Attributi

Attributi di entità

Gli attributi sono le caratteristiche di un oggetto dello schema E-R (per es. di un'entità) che hanno rilevanza per il database; per esempio, per un alunno sono attibuti il suo nome e il suo cognome; gli attributi sono rappresentati nello schema E-R con dei cerchi vuoti all'interno:

Figura 2.9. Gli attributi di un'entità



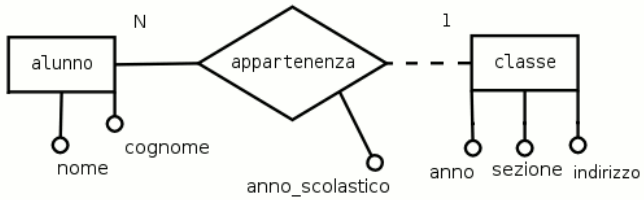
Attributi di associazioni

Può capitare che un attributo sia una caratteristica legata contemporaneamente a più entità; in questo caso l'attributo non può essere assegnato ad una singola entità, ma nemmeno ad entrambe contemporaneamente. La soluzione corretta è di assegnare l'attributo all'associazione che collega le due entità.

Esempio 2.4. Un nuovo anno scolastico

Riprendendo l'esempio nella sua ultima versione C'è una classe vuota, se si desidera che sia possibile registare nel database, oltre alle classi di quest'anno, anche quelle degli anni scolastici futuri, si deve poter registrare anche *l'anno scolastico* di appartenenza della *classe* da parte di ogni *alunno*. L'anno scolastico quindi è un attributo relativo ad entrambe le entità, ovvero, specifica il periodo temporale di validità dell'associazione appartenenza.

Figura 2.10. Gli attributi di un'associazione



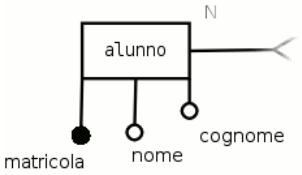
Attributi composti

Continuando a fare riferimento allo stesso esempio C'è una classe vuota, sarebbe sbagliato usare il termine *indirizzo* per indicare dove risiede l'alunno. Questo non può essere considerato un attributo *semplice* perchè può essere scomposto nei seguenti elementi: via, numero, città e provincia. Per questo motivo nello schema E-R devono essere assolutamente *evitati* gli attributi composti.

Attributo identificatore

Hanno il ruolo di identificare univocamente ogni elemento che appartiene all'entità; per esempio, per identificare univocamente ogni alunno può essere usato un numero unico stabilito dalla segreteria al momento dell'iscrizione, che di solito si chiama *matricola*. È importante sottolineare che il valore assegnato all'attributo identificatore deve essere diverso per ogni elemento dell'entità (cioè non devono esistere due alunni con la stessa matricola). Gli atributi identificatori sono rappresentati da un cerchio pieno. Gli attributi identificatori sono molto importanti, ed esiste la seguente regola: ogni entità *deve* possedere un attributo identificatore. In altre parole, non si deve mai usare un'entità senza attributo identificatore in uno schema E-R. Un altro frequente e grossolano errore è quello di considerare tale attributo, l'identificatore della categoria, invece che degli elementi che la compongono. Per le associazioni la regola è diversa: le associazioni rifiutano l'attributo identificatore in quanto esse *non* rappresentano una categoria di elementi (come l'entità).

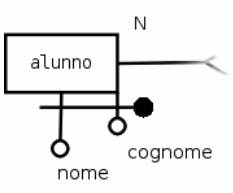
Figura 2.11. L'attributo identificatore di alunno



Attributo identificatore composto

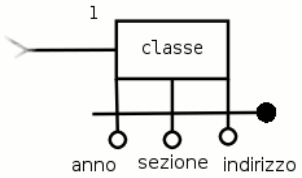
Il divieto di uso degli attributi composti, si riferisce solo agli attributi semplici, non quelli identificatori. Quando nella realtà non si riesce a trovare un attributo che soddisfi i requisiti di unicità necessari per l'identificazione degli elementi di una categoria, si può usare un gruppo di attributi per realizzare un unico attributo identificatore *composto*. Per esempio, se non esistesse la matricola dell'alunno, potremmo usare, per identificare un alunno, il suo nome insieme al suo cognome, nell'ipotesi semplificativa che non esistano mai due alunni omonimi in tale istituto.

Figura 2.12. L'attributo identificatore composto di alunno



Un altro esempio di attributo identificatore composto può essere osservato nell'entità classe, in cui, l'identificazione può essere fatta considerando l'insieme degli attributi costituiti da anno, sezione e indirizzo; come conseguenza non ci potranno essere due classi dello stesso anno, sezione e indirizzo. Un'altra ottima soluzione sarebbe stata quella di aggiungere un attributo identificatore ideato dal progettista e chiamato *codice-classe*.

Figura 2.13. L'attributo identificatore composto di classe



Altre regole ed esempi

- Evitare le entità che possiedono solo l'attributo identificatore, perchè probabilmente non c'è bisogno di quelle entità (e quindi nemmeno dell'eventuale associazione che le lega ad un altra entità);
- cercare di usare un attributo identificatore che occupi meno memoria possibile in termini di byte; ad esempio, usare come identificatore un codice alfanumerico o numerico di solito è una soluzione migliore di quella di usare una lunga parola che fornisce una descrizione completa.

Esempio 2.5. I canili in città

Testo del problema

Si vuole costruire un database per raccogliere dati relativi ai cani che si trovano in diversi canili di una stessa città, ma ad indirizzi diversi. Aggiungere le eventuali ipotesi necessarie. Sviluppare una possibile soluzione concettuale (vocabolario dei termini, schema E-R e regole di lettura) Rappresentare lo schema E-R utilizzando DIA o qualsiasi altro software grafico.

Soluzione del problema

Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente) e di ripetere ogni volta l'esercizio a partire dal punto in cui si sono trovate le eventuali differenze nello svolgimento.

1. **Livello concettuale**
   1. **Vocabolario dei termini**

      cane

      : singolo animale individuabile dal tatuaggio o chip elettronico;

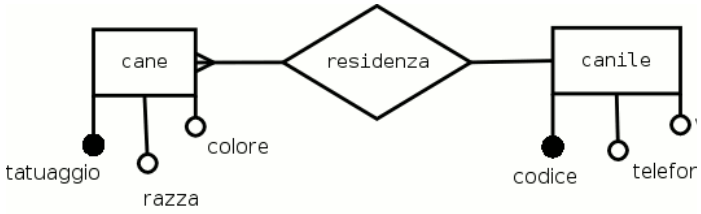
      canile

      : edificio con indirizzo, a cui viene assegnato un codice univoco, che è legato a più cani;

   2. **Schema E-R**

      In questo livello sono state individuate due categorie e un'associazione uno a molti.

      Figura 2.14. Lo schema E-R de I canili in città



Le regole di lettura:

- Ogni cane deve risiedere in un solo canile;
- In ogni canile devono risiedere uno o più cani.

2. **Livello logico**

A questo livello di studio non possiamo ancora affrontare la discussione del livello logico.

Esempio 2.6. Gli esami non finiscono mai

Testo del problema

Si vuole costruire un database per raccogliere dati relativi agli studenti di una facoltà e agli esami che hanno sostenuto nelle diverse materie. Aggiungere le eventuali ipotesi necessarie. Sviluppare una possibile soluzione concettuale (vocabolario dei termini, schema E-R e regole di lettura) Rappresentare lo schema E-R utilizzando DIA o qualsiasi altro software grafico.

Soluzione del problema

Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente) e di ripetere ogni volta l'esercizio a partire dal punto in cui si sono trovate le eventuali differenze nello svolgimento.

1. **Livello concettuale**
   1. **Vocabolario dei termini**

      studente

      individuo riconoscibile dalla matricola di iscrizione, con nome, cognome, telefono, via, città;

      materia

      disciplina (con codice unico) su cui può essere svolta una verifica positiva da ogni studente;

      esame

      superamento della verifica di una materia da parte di uno studente, ha bisogno dell'incontro di entrambe le 2 parti (studente e materia); è ipotizzabile voler registrare anche la data e il voto del superamento dell'esame

      voto

      numero intero da 18 a 30, registrato al superamento dell'esame;

      data

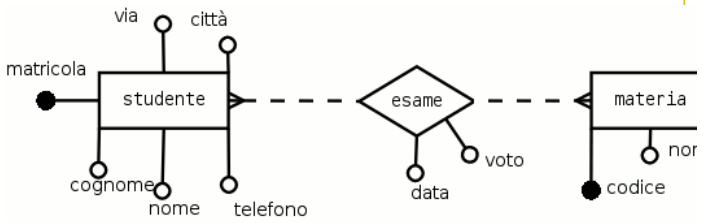
      la data in cui viene avviene il superamento dell'esame.

> **Suggerimento**
> Non è consigliabile ai principianti fare ipotesi che complicano eccessivamente la soluzione del problema

2. **Schema E-R**

In questo livello sono state individuate due categorie e un'associazione molti a molti.

Figura 2.15. Lo schema E-R de Gli esami non finiscono mai



Le regole di lettura:

- Ogni studente può essere esaminato su una o più materie;
- Per ogni materia possono essere esaminati uno o più studenti.

Schema E-R alternativo

In alternativa si poteva conserare anche «esame» come una terza entità da collegare tra le altre due, ma sarebbe stato necessario aggiungere un codice identificativo anche all'esame. Inoltre, in questo caso sarebbe stato possibile per uno studente superare due volte un esame sulla stessa materia... Si tratta di una ipotesi realistica?

2. **Livello logico**

   A questo livello di studio non possiamo ancora affrontare la discussione del livello logico.

Esempio 2.7. Le unioni dei cittadini

Testo del problema

Si vuole costruire un database per raccogliere dati relativi alle unioni tra i cittadini (ad es. matrimoni tra maschi e femmine). Si vuole memorizzare sia la data dell'unione, sia il codice fiscale, la data di nascita, il nome e il cognome di ogni cittadino (che si sia unito o meno). Aggiungere le eventuali ipotesi necessarie. Sviluppare una possibile soluzione concettuale (vocabolario dei termini, schema E-R e regole di lettura) Rappresentare lo schema E-R utilizzando DIA o qualsiasi altro software grafico.

Soluzione del problema

Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente) e di ripetere ogni volta l'esercizio a partire dal punto in cui si sono trovate le eventuali differenze nello svolgimento.

1. **Livello concettuale**
   1. **Vocabolario dei termini**

      maschio

      individuo di sesso maschile individuato dal CF, che si puo' abbinare ad una sola femmina;

      femmina

      individuo di sesso femminile individuato dal CF, che si puo' abbinare ad un solo maschio;

      cittadino

      categoria che raggruppa le due precedenti, senza discriminazione per il sesso;

      unione

      l'unione puo' avvenire tra solo due cittadini alla volta e non è obbligatoria;

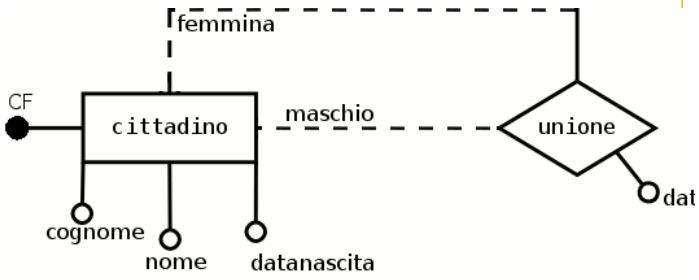
      data

      la data in cui avviene l'unione.

   2. **Schema E-R**

      In questo livello sono state individuate due categorie (maschie e femmina) che potrebbero essere entrambe sostituite dalla categoria «cittadino» e un'associazione uno ad uno.

      Figura 2.16. Lo schema E-R de Le unioni dei cittadini

      Le regole di lettura:

      - Ogni cittadino maschio può essere unito ad un solo cittadino femmina;
      - Ogni cittadino femmina può essere unito ad un solo cittadino maschio.

2. **Livello logico**

   A questo livello di studio non possiamo ancora affrontare la discussione del livello logico.

Capitolo 3. Livello logico

Sommario

Le relazioni matematiche

In questo capitolo si analizzeranno le più importanti «regole di derivazione». Le «Regole di derivazione» servono per passare dalla fase di progettazione del «livello concettuale» a quella del «livello logico», cioè per trasformare lo schema E-R nello schema delle relazioni. Se non fosse ancora abbastanza chiaro, si consiglia di rivedere il significato dei termini Livelli di astrazione e Associazioni, da non confondere con il termine «Relazioni».

Prima di passare alle vere e proprie regole, dobbiamo precisare quale modello logico verrà adottato tra quello relazionale, reticolare, gerarchico ed orientato ad oggetti; ovviamente, poichè lo scopo di questo corso è quello di affrontare il progetto di database relazionali, sarà adottato il modello logico relazionale. Convinti del fatto che prima sia meglio vedere qualche esempio e poi leggere le definizioni matematiche, tratteremo lo studio teorico del «modello relazionale» nel prossimo capitolo, ma nulla vieta al lettore di studiare i due argomenti nell'ordine che preferisce.

Il modello logico «relazionale» si occupa dello studio delle «Relazioni matematiche». A questo livello di studio dovrebbe essere sufficiente sapere che una «relazione matematica», appartenente al livello logico, può essere rappresentata graficamente, nel livello fisico, come una «tabella».

> **Avvertimento**
> Quella appena data *non* è una definizione di relazione: il concetto di «relazione» e quello di «tabella» sono distinti, come lo sono quelli di livello logico e fisico. Molti libri di testo, purtroppo, non li distinguono.

Con la dovuta cautela, si può, descrivendo come è fatta una tabella, descrive anche come è fatta la corrispondente relazione che l'ha generata. Una tabella è divisa in righe e in colonne: rispettando la terminologia usata negli archivi elettronici, le righe sono chiamate anche «record» e le colonne sono chiamate anche «campi». I record (righe) di una tabella corrispondono agli «elementi della relazione», mentre i campi (colonne) agli «attributi della relazione».

> **Nota**
> Non confondere il nuovo termine «attributi di relazione» con quello di Attributi di entità.

Tabella 3.1. Esempio di tabella (vuota e senza nome)

attributo1	attributo2	attributo3	attributo4
...	...	...	...
...	...	...	...

Per rappresentare una relazione sarebbe troppo faticoso disegnare ogni

volta un'intera tabella (vuota), per questo si usa spesso la seguente convenzione alternativa o «rappresentazione sintetica»:

```
Nome_della_relazione (attributo1, attributo2, attributo3, attributo4)
```

Si faccia attenzione a non fare confusione con i nuovi termini, che possiedono un significato importante e che evidenziano come, passando da un livello di progettazione all'altro, alcuni elementi cambino di ruolo e anche di nome. Per aiutare il lettore viene fornita una tabella riassuntiva che raggruppa i termini per livello di astrazione e indica a quale livello di progettazione ci si riferisca quando se ne usa uno in particolare.

Avvertimento

La tabella va letta esclusivamene seguendo il verso delle colonne, leggendola secondo le righe si possono ottenere informazioni sbagliate.

Tabella 3.2. Elenco dei termini usati nei diversi livelli di astrazione

Livello concettuale	Livello logico	Livello fisico
entità	relazione	tabella
attributo identific.	chiave primaria	campo chiave primaria
attributo (di entità)	chiave esterna	campo chiave esterna
associazione	attributo (di relazione)	record
	elemento di relazione	

Regola per entità

Le «regole di derivazione» permettono di ottenere lo schema logico a partire dallo schema concettuale. La prima regola di derivazione che si studia è quella che trasforma le entità dello schema E-R in relazioni nello schema logico relazionale.

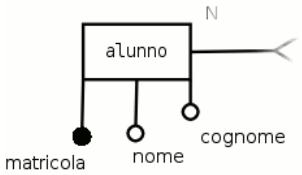
Importante

Regola 1: ogni entità dello schema E-R viene trasformata in una relazione.

Esempio 3.1. Derivazione dell'entità alunno

Come esempio si può considerare l'entità «alunno» e i suoi attributi: «matricola», «nome» e «cognome».

Figura 3.1. Entità con attributo identificatore



Seguendo la regola appena vista si ottiene una nuova relazione, che chiameremo «alunni» (al plurale), per non fare confusione con il nome dell'entità «alunno» (al singolare). La rappresentazione sarebbe la seguente:

```
alunni ()
```

Per poter chiarire meglio e completare questo esempio è necessario studiare anche la prossima regola di derivazione.

Regola per attributi

Questa regola si riferisce esclusivamente agli attributi delle entità, mentre per quello che riguarda gli attributi delle associazioni seguiranno ulteriori precisazioni.

> ***Importante***
> *Regola 2: ogni attributo proprio di un'entità diventa un attributo proprio della relazione che ne è stata derivata.*

Esempio 3.2. Derivazione degli attributi di alunno

Come esempio si può considerare sempre l'entità «alunno» e i suoi attributi: «matricola», «nome» e «cognome». Vedi Figura Entità con attributo identificatore

Dopo aver applicato la prima, si applica la seconda regola e si ottiene una nuova relazione con 3 attributi, la cui rappresentazione sarebbe:

```
alunni ( matricola, nome, cognome )
```

Per chiarezza segue anche la rappresentazione di una tabella vuota di esempio:

Tabella 3.3. Tabella «alunni» (vuota)

matricola	nome	cognome
...	...	...
...	...	...

Regola per attributi identificatori ed esempi

Questa regola si riferisce esclusivamente agli attributi identificatori.

> ***Importante***
> *Regola 3: ogni attributo identificatore proprio di un'entità diventa la chiave primaria della relazione che ne è stata derivata.*

Come ogni altro attributo, anche l'attributo identificatore dell'entità diventa un attributo della relazione. L'attributo identificatore svolge però l'importante funzione di identificare in modo univoco e di rendere accessibile ogni elemento della relazione, cioè ogni riga della tabella, perciò tale attributo della relazione viene chiamato «chiave primaria». Verrà adottata la convenzione di sottolineare i campi chiave primaria. Questa regola si può applicare sia agli attributi identificatori semplici che composti.

Per poter assolvere il suo compito indentificativo, la chiave primaria di una relazione deve assumere valori *unici* (non può essere ripetuto due volte lo stesso valore) ed *obbligatori* (non ci può essere un elemento della relazione senza valore di chiave primaria). Siccome ogni entità può avere un solo attributo identificatore, ne segue che ogni relazione può avere una sola chiave primaria.

Esempio 3.3. Derivazione della matricola dell'alunno

Consideriamo il solito esempio dell'entità «alunno» con i suoi 3 attributi; siccome «matricola» è l'attributo a cui va applicata la regola, possiamo dire che non ci possono essere due alunni con la stessa matricola e nemmeno un alunno senza matricola. Vedi Figura Entità con attributo identificatore

La rappresentazione che indica «matricola» come chiave primaria è la seguente:

```
alunni ( matricola, nome, cognome )
```

Questa è la tabella corrispondente, riempita con alcuni valori di esempio:

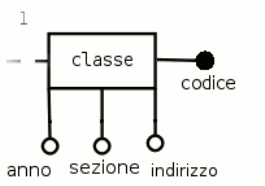
Tabella 3.4. Tabella «alunni» (ripempita)

matricola	nome	cognome
0102	Mario	Rossi
0103	Anna	Bianchi
0154	Giulio	Bianchi

Esempio 3.4. Derivazione dell'entità classe

Si provi ad applicare le prime tre regole di derivazione all'entità «classe» nel caso in cui si sia aggiunto un codice numerico come attributo identificatore.

Figura 3.2. Entità classe con attributo identificatore semplice



Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente)

```
classi ( codice, anno, sezione, indirizzo )
```

Esempio 3.5. Derivazione di un'attributo identificatore composto

Si provi ad applicare le prime tre regole di derivazione all'entità «classe» nel caso in cui si siano usati i suoi 3 attributi per costituire un'attributo identificatore composto. Vedi Figura L'attributo identificatore composto di classe

Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente)

```
classi ( anno, sezione, indirizzo )
```

Nota

Nell'ultimo esempio, la linea di sottolineatura *non* si deve interrompere tra un attributo e l'altro. Questo sta ad indicare che i tre attributi costituiscono, nel loro insieme, un'unica chiave primaria. Per questo motivo *non* è consentito ripetere due volte la stessa combinazione anno-sezione-indirizzo ed è obbligatorio inserire i valori in ogni campo.

Regola per associazioni uno a molti

Dopo aver derivato tutte le entità e tutti gli attributi, si può passare alla derivazione delle associazioni (che sono i rombi dello schema E-R); il tipo più semplice da derivare è quello «uno a molti» (1:N).

Importante

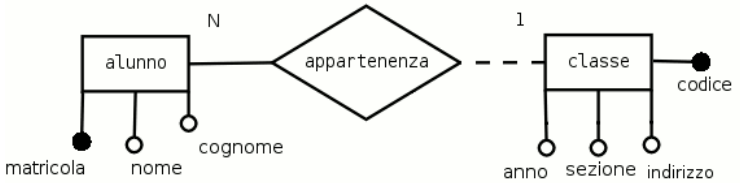
Regola 4: (Prima parte) per ogni associazione binaria di tipo 1:N è necessario aggiungere, alla relazione che partecipa con molteplicità N, un nuovo attributo, detto «chiave esterna», collegato logicamente alla chiave primaria dell'altra relazione (a molteplicità 1).

La comparsa della chiave esterna tra gli attributi, permette di realizzare un collegamento tra le due relazioni. Si adotterà la convenzione di sopralineare la chiave esterna

Esempio 3.6. Derivazione dell'associazione appartenenza

Si applicheranno tutte le regole di derivazione note al seguente schema E-R.

Figura 3.3. Schema E-R dell'associazione tra alunno e classe



Applicando le prime tre regole di derivazione alle due entità si ottengono le seguenti due relazioni:

```
alunni ( matricola, nome, cognome ) ; classi ( codice, anno, sezione, indirizzo )
```

A questo punto le due relazioni sono ancora indipendenti l'una dall'altra, quindi si possono scrivere anche affiancate.

La regola di lettura dice che «ad ogni classe possono appartenere uno o più alunni», quindi è necessario aggiungere alla relazione «alunni» (a molteplicità N) un nuovo attributo, a cui si darà il nome codice-classe che sarà collegato all'attributo codice della relazione «classi».

```
classi ( codice, anno, sezione, indirizzo ) ;

alunni ( matricola, nome, cognome, codice-classe )
```

Come si sarà notato, questa volta la relazione «classi» precede «alunni», questo perchè, a causa del legame instaurato dalla chiave esterna, è necessario riempire prima la tabella «classi» e successivamente quella di «alunni».

Tabella 3.5. classi

codice	anno	sezione	indirizzo
41	4	A	Mercurio
42	4	A	IGEA
43	5	A	Mercurio

Tabella 3.6. alunni

matricola	cognome	nome	codice-classe
025221	Rossi	Mario	41
025223	Bianchi	Anna	41
025101	Verdi	Giuseppe	42

Il collegamento tra una chiave esterna e una primaria, fa in modo che la prima possa assumere solo i valori stabiliti dalla seconda. Questo legame logico tra le relazioni (e tra le tabelle che le rappresentano) viene chiamato tecnicamente «vincolo di integrità referenziale», e verrà studiato più avanti.

Sebbene la chiave esterna sia collegata alla chiave primaria vi sono profonde differenze tra i due tipi di attributi: la «chiave esterna» può essere *non unica* (nella chiave esterna si possono ripetere più volte gli stessi valori) e può essere *non obbligatoria* (questo accade solo quando le regole di lettura dell'associazione lo permettono, per esempio se ci potessero essere alunni senza una classe). Come esercizio, verificare alcune delle proprietà appena esposte nella tabella alunni del precedente esempio.

Suggerimento

Arrivati a questo punto del progetto, *non* si deve tornare indietro a modificare lo schema E-R per aggiungere qualche attributo alle entità. La chiavi esterne si aggiungono esclusivamente nel livello logico, non nel livello concettuale.

Nel caso in cui anche l'associazione uno a molti avesse attributi, la regola continua così:

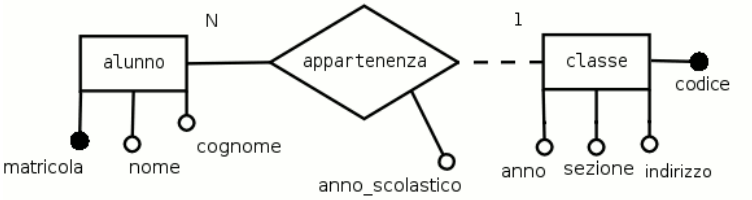
Importante

Regola 4: (Seconda parte) Gli eventuali attributi dell'associazione uno a molti vanno messi nella stessa relazione in cui si trova la chiave esterna

Esempio 3.7. Derivazione dell'associazione appartenenza con attributo

Lo schema E-R:

Figura 3.4. Schema E-R di associazione con attributo



Lo schema delle relazioni che si ottiene è:

```
classi (codice, anno, sezione, indirizzo);

alunni (matricola, cognome, nome, codice-classe, anno_scolastico)
```

Le tabelle sono lasciate come esercizio al lettore.

Regola per associazioni molti a molti

Dopo aver applicato le regole di derivazione per entità e attributi di entità, nel caso di associazioni molti a molti si procede come segue.

> ***Importante***
> *Regola 5 (prima parte): per ogni associazione binaria N:N tra due entità, è necessario aggiungere alla due relazioni derivate dalle due entità, una terza relazione i cui attributi sono chiavi esterne logicamente collegate alle chiavi primarie delle prime due relazioni.*

Nel caso in cui nell' associazione molti a molti siano presenti attributi, la regola continua così:

> ***Importante***
> *Regola 5 (seconda parte): gli eventuali attributi dell'associazione N:N diventano attributi della relazione che contiene anche le chiavi esterne.*

Esempio 3.8. Derivazione dell'associazione esame

Derivare Lo schema E-R de Gli esami non finiscono mai :

Le due entità dànno luogo alle relazioni «studenti» e «materie» a cui va aggiunta, a causa della regola appena vista, la relazione «esami». In questa relazione saranno presenti due chiavi esterne «matricola-studente» e «codice-materia», collegate alle chiavi primarie delle altre due relazioni: «matricola» in studenti e «codice» in materie. L'ordine con cui vengono scritte le tre relazioni è importante.

```
studenti ( matricola, cognome, nome, telefono, via, citta);

materie (codice, nome ) ;

esami ( matricola-studente, codice-materia )
```

Per concludere l'esempio è necessario aplicare anche la seconda parte della Regola 5:

```
studenti ( matricola, cognome, nome, telefono, via, citta);

materie (codice, nome ) ;

esami ( matricola-studente, codice-materia, data, voto )
```

Per esercizio, verificare che i dati nelle seguenti tabelle rispettino i vincoli delle chiavi esterne e contengano una casistica abbastanza varia da permettere di verificare le regole di lettura scritte in precedenza.

Tabella 3.7. studenti

matricola	cognome	nome	via	citta	telefono
025221	Rossi	Mario	via stretta, 5	Pisa	050111111
025223	Verdi	Giuseppe	p.zza italia, 3	Roma	06123456

Tabella 3.8. materie

codice	nome
01	chimica
02	informatica
03	inglese

Tabella 3.9. esami

matricola-studente	codice-materia	data	voto
025221	02	2008/11/31	28
025223	02	2007/01/29	19
025221	01	2009/01/01	25

> **Nota**
> Nell'ultimo esempio è stato volutamente omesso l'accento sul termine «città» perchè potrebbe capitare che alcuni software non ammettano l'uso di lettere accentate durante la realizzazione di tabelle.

Esercizi di comprensione

Esempio 3.9. I proprietari dei veicoli

Testo del problema

Si desidera realizzare il database degli autoveicoli e dei loro proprietari di un certo paese, ricordando che un autoveicolo può essere anche di proprietà di più persone (co-intestazione). In particolare si vuole registrare anche quale tipo di alimentazione abbia ogni veicolo; le possibilità di alimentazione devono essere le seguenti: benzina, diesel, gpl, metano, elettrico, idrogeno, ibrido (per considerare autoveicoli alimentabili con più tipi di propellenti). Aggiungere le eventuali ipotesi necessarie. E' richiesto di realizzare lo schema ER, le regole di lettura, lo schema sintetico delle relazioni e le tabelle di prova (testing).

Soluzione del problema

Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente) e di ripetere ogni volta l'esercizio a partire dal punto in cui si sono trovate le eventuali differenze nello svolgimento.

1. **Livello concettuale**
    1. **Vocabolario dei termini**

        cittadino/proprietario

            individuo identificato dal codice fiscale;

        autoveicolo

            oggetto con legame di proprietà con cittadino/ni, identificato dalla targa e caratterizzato dalla tipologia di carburante con cui viene alimentato;

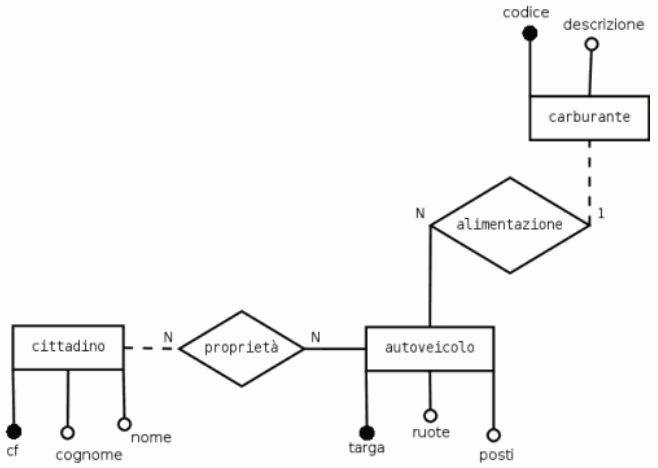
        carburante

            l'oggetto carburante (o tipo-di-carburante) rappresenta una «tipologia» di sostanze ed è identificato da un codice numerico.

    2. **Schema E-R**

        L'esercizio potrebbe essere risolto utilizzando «carburante» come un semplice attributo dell'entità autoveicolo, che può assumere solo 7 valori (stringhe). Esiste però una soluzione alternativa (più efficiente soprattutto nei grandi database) che consiste nel realizzare un'ulteriore entità denominata (tipo-di-)carburante, in modo che venga assegnato un codice ad ogni tipo di carburante. In questo modo, nelle tabelle, non si dovrà più ripetere tante volte il nome del carburante (ad esempio «benzina»), ma basterà indicare, in sua sostituzione, il codice che lo rappresenta (ad esempio «1»), con notevole risparmio di memoria.

Figura 3.5. schema ER dell'autoveicolo e dei proprietari



    3. **Le regole di lettura:**

        - Ogni autoveicolo deve essere di proprietà di una o più persone;
        - Ogni persona può essere proprietaria di uno o più autoveicoli;
        - Ogni autoveicolo deve essere alimentato con un (tipo di) carburante;
        - Ogni (tipo di) carburante può alimentare uno o più autoveicoli.

2. **Livello logico**

    Dalle 3 entità si ottengono 3 relazioni, più un'ultima relazione dovuta alla presenza dell'associazione molti a molti (N:N). Si è evitato di usare lettere accentate sui nomi delle relazioni.

    ```
    cittadini (cf, cognome, nome)

    carburanti (codice, descrizione)

    autoveicoli (targa, posti, codice-carburante)
    ```

proprieta (cf-cittadino, targa-auto)

3. **Esempio di testing:**
   è importante verificare che le tabelle possano contenere più cittadini proprietari di piu' autoveicoli (o di nessuno).

Tabella 3.10. cittadini

cf	cognome	nome
RSSMRA	Rossi	Mario
VRDGPP	Verdi	Giuseppe
BNCGPP	Bianchi	Giuseppe
GLLGNN	Gialli	Gianna

Tabella 3.11. carburanti

codice	descrizione
1	Benzina
2	Diesel
3	Metano
4	GPL
5	Elettrico
6	Idrogeno
7	Ibrido

Tabella 3.12. autoveicoli

targa	posti	codice-carburante
AZ12345	5	1
BZ12345	5	1
OIL9876	7	2
EE33333	5	5

Tabella 3.13. proprieta

targa-auto	cf-cittadino
AZ12345	RSSMRA
AZ12345	VRDGPP
OIL9876	GLLGNN
EE33333	GLLGNN

Per esercizio, risolvere lo stesso problema registrando anche la data di acquisto dell'autoveicolo.

Esempio 3.10. Dirigenti e dipendenti

Testo del problema

Si desidera realizzare il database di un'azienda in cui lavorano due tipi di impiegati: gli impiegati che hanno un ruolo di manodopera e quelli che hanno un ruolo di dirigente. Tutti gli impiegati sono identificati da un numero di matricola, e inoltre hanno un cognome e un nome. Si desidera conoscere anche chi sia il dirigente di ogni impiegato. Aggiungere le ipotesi necessarie. E' richiesto di realizzare lo schema ER, le regole di lettura, lo schema sintetico delle relazioni e le tabelle di prova (testing).

Soluzione del problema

Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente) e di ripetere ogni volta l'esercizio a partire dal punto in cui si sono trovate le eventuali differenze nello svolgimento.

1. **Livello concettuale**
   1. **Vocabolario dei termini**

      impiegato

      persona fisica identificata dalla matricola, appartenente all'azienda;

      dirigente

      persona fisica identificata dalla matricola, può far parte degli impiegati stessi dell'azienda (soluzione 2), oppure può essere considerata una categoria a parte perchè è un superiore;

   2. **Schema E-R**

      La soluzione con due entità è suggerita se nel database verranno eseguite ricerche esclusivamente sui dipendenti o sui dirigenti.

      Figura 3.6. schema ER della prima soluzione

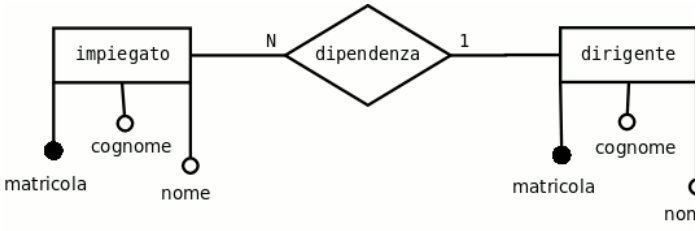
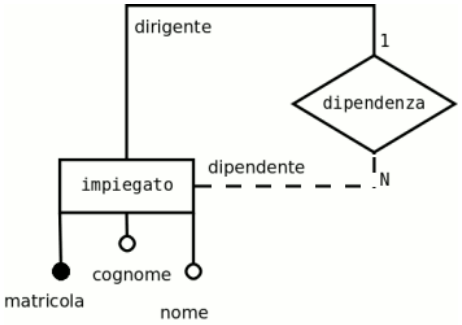


Figura 3.7. schema ER della seconda soluzione



3. **Le regole di lettura:**
   Poichè le regole di lettura sono abbastanza simili nelle due soluzioni, si riporta solo quella della prima:

   - Ogni dipendente deve dipendere da un dirigente;
   - Ogni dirigente deve avere alle sue dipendenze uno o più impiegati.

2. **Livello logico**

   Dalle 2 entità si ottengono 2 relazioni, la presenza dell'associazione uno a molti (1:N) non genera una relazione, ma solo una chiave esterna.

   Prima soluzione

   ```
   dirigenti (matricola, cognome, nome)

   impiegati (matricola, posti, matricola-dirigente)
   ```

   Seconda soluzione

   ```
   dipendente (matricola, cognome, nome, matricola-dirigente)
   ```

   Nel caso di associazioni ricorsive le regole di derivazione rimangono le stesse, facendo finta che esistano due entità distinte. Si riveda l'esempio I vicini di banco. In questo caso gli impiegati dirigenti avranno la chiave esterna vuota perchè non hanno un dirigente.

3. **Esempio di testing:**
   è lasciato come esercizio.

Regola per associazioni uno ad uno

Se si è in possesso di un libro di testo «semplice», è probabile avervi trovato, nel caso delle associazioni «uno a uno», una regola di derivazione diversa da quella che verrà descritta in queste pagine. Nei libri si potrebbe trovare una regola che dice di fondere le due entità collegate dall'associazione uno ad uno in un'unica relazione (che dà luogo ad un'unica tabella).

Escludendo il caso banale in cui una delle due entità fosse costituita da un solo attributo, ci si dovrebbe domandare a cosa sia servita l'analisi concettuale del problema e l'individuazione delle due entità, concettualmente diverse, se poi si deve di nuovo fonderle in un'unica relazione. In risposta a questa domanda si può affermare che è conveniente continuare a mantenere separate le due tabelle, soprattutto se queste sono destinate a contenere dati di diverse categorie. Per contro, questo costringe a seguire una regola leggermente più complessa, ma che ha come vantaggio un utilizzo più efficiente dello spazio sui dispositivi di memoria.

Questa regola si divide in tre casi, essendo l'unica che distingue anche se l'associazione è obbligatoria o meno.

Prerequisito per affrontare questa regola sono: la Regola per associazioni uno a molti, la Regola per associazioni molti a molti e l'esempio Le unioni dei cittadini

Avvertimento

La derivazione di associazioni molti a molti in relazioni, era stata ottenuta consentendo di ripetere più volte il valore di una chiave esterna. Per ottenere associazioni uno ad uno, quindi, si dovrebbe impedire questa possibilità, aggiungendo un particolare tipo di vincolo alla relazione, ma tali vincoli saranno studiati solo nel prossimo capitolo.

Regola per associazioni non obbligatorie

Dopo aver derivato le entità e gli attributi delle entità, per le associazioni uno a uno a cui *entrambe* le entità *non* partecipano obbligatoriamente, si procede come segue:

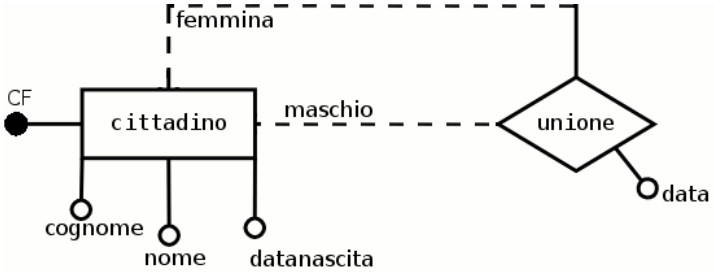
Importante

Regola 6 (prima parte): ogni associazione uno ad uno non obbligatoria viene derivata come se fosse un'associazione molti a molti. La stessa regola vale per gli eventuali attributi dell'associazione.

Esempio 3.11. Derivazione dell'associazione unione

Nell'esempio Le unioni dei cittadini si aveva il seguente schema E-R:

Figura 3.8. Lo schema E-R de Le unioni dei cittadini



L'entità «cittadino» diventa la relazione «cittadini» con i suoi attributi, inoltre, applicando la regola appena vista, si aggiunge una la relazione «unioni»; che dovrà contenere due chiavi esterne. Poichè l'associazione «unione» collega due volte il cittadino, come se fossero due entità distinte, ci sarà una chiave esterna (codice fiscale) per cittadino-maschio e un'altra per cittadino-femmina, che chiameremo cf-maschio e cf-femmina.

```
cittadini (cf, cognome, nome, datadinascita)

unioni (cf-maschio, cf-femmina)
```

Tabella 3.14. cittadini

cf	cognome	nome	datadinascita
RSSMRA	Rossi	Mario	1950-12-25
VRDGPP	Verdi	Giuseppe	1950-12-31
BNCNNA	Bianchi	Anna	1960-02-28
RSSMRC	Rossi	Marco	1960-02-02
NRENRC	Neri	Enrico	1962-01-01
GLLGNN	Gialli	Gianna	1930-03-30

Tabella 3.15. unioni

cf-maschio	cf-femmina	data
VRDGPP	BNCNNA	1970-12-23

Regola per associazioni parzialmente obbligatorie

Dopo aver derivato le entità e gli attributi delle entità, per le associazioni uno a uno *parzialmente* obbligatorie si procede come segue:

Importante

Regola 6 (seconda parte): ogni associazione uno ad uno in cui una sola entità partecipa obbligatoriamente viene derivata come se fosse un'associazione uno a molti, immaginando che l'entità che vi partecipa obbligatoriamente sia quella con molteplicità maggiore di uno.

Alla relazione ottenuta da quest'ultima entità va aggiunta la chiave esterna. Anche gli eventuali attributi dell'associazione uno ad uno vanno nella relazione che contiene la chiave esterna.

Esempio 3.12. Derivazione dell'associazione occupazione

Testo del problema

Si desidera realizzare per un albergo un database che raccolga le informazioni sulle camere e sull'ospite che la occupa in quel momento. Quando si libera la camera i dati del cliente vengono eliminati dal database. Anche nel caso di camere con più ospiti si segnano sempre solo i dati di un solo cliente. Si vogliono registrare i dati anagrafici del cliente e le caratteristiche della camera (numero camera, numero letti, superificie). Aggiungere le eventuali ipotesi necessarie. E` richiesto di realizzare lo schema ER, le regole di lettura, lo schema sintetico delle relazioni e le tabelle di prova (testing).

Si suggerisce di leggere la soluzione solo dopo aver provato a risolvere l'esercizio (anche parzialmente) e di ripetere ogni volta l'esercizio a partire dal punto in cui si sono trovate le eventuali differenze nello svolgimento.

Soluzione del problema

1. **Livello concettuale**
   1. **Vocabolario dei termini**

      cliente/ospite

      > l'individuo a cui risulta occupata una camera, identificato dal codice fiscale;

      camera

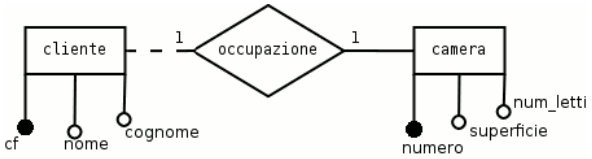
      > stanza dotata di un certo numero identificativo, di un numero di letti e di una certa superficie (metri quadrati);

   2. **Schema E-R**

      Le due categorie individuate sono due entità, legate tra di

loro da un'associazione uno ad uno parzialmente obbligatoria, perchè ci possono essere stanze libere, ma non clienti senza una stanza, perchè i clienti che lasciano l'albergo vengono anche eliminati dal database.

Figura 3.9. Lo schema E-R de Derivazione dell'associazione occupazione



- ogni cliente deve occupare una camera
- ogni camera può essere occupata da un cliente

2. **Livello logico**

   Le entità «cliente» e «camera» diventano due relazioni: «clienti» e «camere». Alla relazione derivata dall'entità a partecipazione obbligatoria («camere») si aggiunge la chiave esterna collegata all'associazione derivata dall'entità a partecipazione opzionale («clienti»).

   ```
   camere (numero, superficie, num_letti)

   clienti (cf, cognome, nome, numero-camera)
   ```

3. **Esempio di testing:**

Tabella 3.16. camere

numero	superficie	num_letti
110	9	1
111	9	1
112	9	1
113	13	2
114	15	3

Tabella 3.17. clienti

cf	nome	cognome	numero-camera
RSSMRA	Mario	Rossi	114

cf	nome	cognome	numero-camera
BNCNNA	Anna	Bianchi	111

Avvertimento

Si potrebbe obiettare che utilizzando questo schema sarebbe teoricamente possibile far occupare la stessa camera a due clienti diversi, violando così l'associazione uno ad uno. Questo tipo di vincolo non può essere aggiunto a questo livello di studio e verrà affrontato nel prossimo capitolo.

Nota

Solo le effettive occupazioni vengono registrate, e nonostante l'associazione non sia completamente obbligatoria, nelle due precedenti tabelle non si trova una sola cella vuota: si può quindi affermare che questo è il metodo più efficiente per memorizzare come vengono occupate le camere dai clienti!

Rispondere alla seguente domanda facendo un esempio: si sarebbe ottenuto un database ugualmente efficiente utilizzando una regola di derivazione diversa?

Regola per associazioni obbligatorie

Dopo aver derivato le entità e gli attributi delle entità, per le associazioni uno a uno completamente obbligatorie si procede come segue:

Importante

Regola 6 (terza parte): ogni associazione uno a uno completamente obbligatoria, viene trasformata come se fosse un'associazione uno a molti; in questo caso la chiave esterna può essere messa indifferentemente in una qualsiasi delle relazioni ottenute. Gli eventuali attributi dell'associazione uno a uno vanno nella stessa relazione che contiene la chiave esterna.

Questa volta non c'è nessuna partecipazione opzionale e non c'è la possibilità che nelle tabelle si abbiano delle celle vuote, perciò ovunque venga collocata la chiave esterna si otterrà sempre un database efficiente. Per coloro che si domandano se fosse possibile costruire un'unica relazione con gli attributi di entrambe le entità, la risposta è quasi sempre negativa. E' sempre meglio mantenere separate le due categorie, anche dopo la derivazione delle entità in relazioni, perchè esse rappresentano sempre due concetti distinti che il progettista ha individuato nella fase di progetto concettuale. Rimettere insieme due concetti che erano stati faticosamente individuati e separati, sarebbe considerare inutile il lavoro precedente.

Fase di testing

In quasi tutti gli esempi proposti in questo capitolo è stata aggiunta una tabella piena di dati, per poter capire meglio la regola di derivazione. Anche se il progetto di un database si potrebbe considerare concluso con la stesura delle relazioni in forma sintetica, è una buona pratica quella di disegnare anche le tabelle e soprattutto di riempirle con dei dati, in modo da verificare il funzionamento della struttura nel suo insieme.

Questa operazione è una prova generale della bontà del prodotto che si è ottenuto ed è chiamata anche «fase di testing». Per fare un esempio, quando si deve verificare se una associazione uno a molti è stata ben realizzata con le chiavi esterne, bisogna verificare (nella fase di testing) se sia possibile ripetere più volte alcuni elementi in una delle tabelle, senza violare le regole del modello logico relazionale (per esempio senza che si ripeta due volte lo stesso valore di chiave primaria).

Le regole del modello logico relazionale, o vincoli del modello relazionale, saranno oggetto del prossimo capitolo.

Convenzioni

Nella stesura dello schema delle relazioni abbiamo implicitamente assunto queste convenzioni:

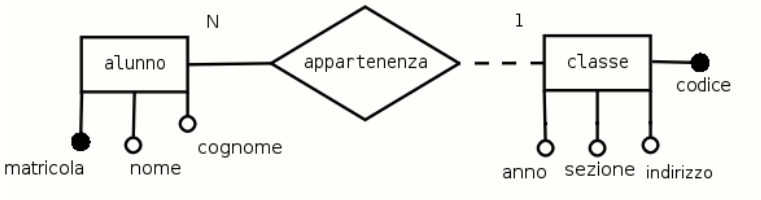
- mentre le entità sono al singolare, i nomi delle relazioni sono al plurare;
- i nomi delle relazioni e degli attributi non devono contenere nè spazi nè lettere accentate, questo perchè alcuni software potrebero non consentirne l'uso;
- l'ordine in cui si scrivono le relazioni è importante: prima si definiscono le relazioni senza chiavi esterne e dopo quelle le cui chiavi esterne sono già state definite. (vedi esempio Derivazione dell'associazione appartenenza).

Convenzioni improprie in alcuni DBMS

Ricordiamo che il progettista di un database svolge il suo progetto passando dal livello concettuale al livello logico relazionale, e infine realizza concretamente il modello logico che ha ottenuto, utilizzando un DBMS. Per questo motivo il DBMS deve «ignorare» completamente tutto quello che che riguarda lo schema E-R (entità, attributi e associazioni) e deve rappresentare, mediante delle tabelle, solo le relazioni che si sono ottenute.

Purtroppo, però, per rappresentare il legame logico, il vincolo, tra una chiave primaria e una chiave esterna di due tabelle, molti DBMS usano una rappresentazione grafica e una terminologia impropria. Per esempio, da questo schema E-R:

Figura 3.10. Schema E-R dell'associazione tra alunno e classe



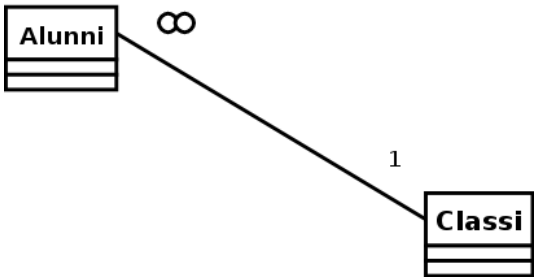
Si ottengono le seguenti due relazioni:

```
alunni ( matricola, nome, cognome, codice-classe )

classi ( codice, anno, sezione, indirizzo )
```

In un DBMS, esse vengono rappresentate come 2 tabelle, e schematizzate in questo modo:

Figura 3.11. Un DBMS ad interfaccia grafica



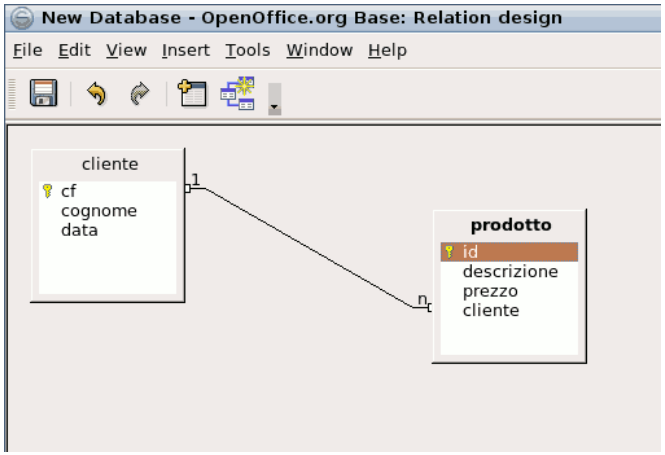
La linea nera che unisce le due tabelle viene *impropriamente* chiamata «Relazione» e ai suoi estremi vengono visualizzati anche dei simboli

numerici, come se si volesse specificare la molteplicità di un'associazione dello schema E-R, ma come detto all'inizio di questo paragrafo, il DBMS ignora quello che riguarda il livello concettuale.

Ricordando queste considerazioni si possono evitare errori di interpretazione di ciò che viene rappresentato da un DBMS.

Purtroppo anche OpenOffice fa questa confusione di termini

Figura 3.12. Interfaccia grafica di OpenOffice



Capitolo 4. Modello relazionale

Sommario

Definizioni

Per introdurre il modello matematico relazionale è necessario ricordare alcune definizioni: insieme, prodotto cartesiano, relazione, tabella, n-uple, t-uple, attributo, dominio

Insieme

una collezione di oggetti chiamati elementi dell'insieme; un elemento può appartenere o non appartenere a un determinato insieme, non ci sono vie di mezzo. Un elemento non può comparire più di una volta in un insieme; gli elementi di un insieme non hanno un ordine di comparizione (l'insieme non è ordinato); gli elementi di un insieme lo caratterizzano univocamente: due insiemi coincidono se e solo se hanno gli stessi elementi.

Prodotto cartesiano

il prodotto cartesiano di due insiemi A e B (si scrive A x B) è l'insieme costituito dalle coppie ordinate (a,b), dove a è un elemento di A e b un elemento di B. Le coppie sono ordinate, cioè la coppia (a,b) è diversa dalla coppia (b,a), perciò il prodotto A x B è diverso dal prodotto B x A.

Esempio 4.1. Esempio di prodotto cartesiano

Dati gli insiemi `A = {Rossi,Verdi}` e `B = {1980, 1990, 1991}` ; il loro Prodotto cartesiano vale `A x B = { (Rossi,1980), (Rossi,1990), (Rossi,1991), (Verdi,1980), (Verdi,1990), (Verdi,1991) }`.

Notare che l'insieme A è di 2 elementi, B di 3 elementi, A x B di 6 elementi. L'ordine con cui sono disposti gli elementi all'interno di un insieme non è importante, ma lo è invece, quello con cui sono disposti gli elementi all'interno delle parentesi tonde.

In generale il prodotto cartesiano può riguardare anche più di *due* insiemi (A, B, C, D,...), e al posto delle *coppie* si possono avere triple, quadruple, e in generale *n-uple* (pronunciato enn-uple).

Relazione (matematica)

Si può definire la Relazione matematica (o più brevemente Relazione) R su A e B incata da R(A,B) un sottoinsieme dell'insieme prodotto cartesiano A x B.

Esempio 4.2. Esempio di relazione matematica

Continuando il precedente esempio, una possibile relazione su A e B potrebbe essere costituita da `R(A,B) = { (Rossi,1990), (Rossi,1991), (Verdi,1980) }`

Come il prodotto cartesiano, anche una relazione può riguardare più di due

insiemi (A, B, C, D,...), e al posto delle coppie si possono avere triple o quadruple.

Nota

Mentre in matematica per il Prodotto cartesiano, l'ordine degli elementi delle coppie (a,b) è importante, nelle relazioni studiate nei database, l'ordine (Rossi,1990) è meno importante. Per questo motivo, al posto del termine «n-uple» viene usato anche il termine «t-uple».

Importante

vista la natura insiemistica della relazione, e visto anche il precedente esempio, si può dedurre che non ci possono essere due n-uple (o t-uple) identiche nella stessa relazione matematica. Nonostante ciò, nella pratica dei DBMS, questo vincolo non esiste, ma è il programmatore che deve aggiungerlo, per fare in modo che le tabelle del DBMS abbiano le proprietà delle relazioni matematiche.

Grado e cardinalità di una Relazione

Il «grado» è pari al numero degli insiemi su cui è affettuata la relazione (A, B, C, D, ...), mentre la «cardinalità» è pari al numero degli elementi che la costituiscono.

Esempio 4.3. Esempio di grado e cardinalità

Continuando il precedente esempio `R(A,B) = { (Rossi,1990), (Rossi,1991), (Verdi,1980) }` è di grado 2 e di cardinalità 3.

Tabella

è una delle possibili rappresentazioni in cui può essere espressa una Relazione, quindi permette di visualizzare gli elementi che la costituiscono, ma parlando in termini strettamente matematici, una tabella *non* è una Relazione.

Esempio 4.4. Esempio di tabella

La precedente Relazione R su A e B potrebbe essere espressa dalla seguente tabella

Tabella 4.1. Rappresentazione di R(A,B)

A	B
Rossi	1990
Rossi	1991
Verdi	1980

I DBMS utilizzano spesso il termine «tabella» in luogo del termine «relazione»; altri DMBS ad interfaccia grafica, addirittura, usano il termine «relazione» per indicare i collegamenti tra le chiavi primarie e le chiavi esterne: si eviti di fare la stessa confusione di termini che fanno questi DBMS.

Attributi

sono gli insiemi che vengono utilizzati per costriure la Relazione, come A e B , oppure Cognome, Nome, ecc.

Dominio

è l'insieme dei valori che può assumere un attributo: per esempio A è un insieme di nomi (stringhe), B è un insieme di numeri interi. Importante: ogni attributo ha un proprio dominio di valori, che non può essere modificato.

Vincoli intrarelazionali

I vincoli intrarelazionali sono regole che descrivono la struttura dei dati all'interno di una singola relazione (rappresentabile da una singola tabella). Si distinguono nei seguenti tipi:

vincolo di unicità

> impedisce che all'interno di un attributo sia inserito due volte lo stesso valore;

vincolo di obbligatorietà

> impedisce che all'interno di un attributo sia lasciato un valore non inserito;

vincolo di chiave primaria

> oltre ad unire insieme i due precedenti vincoli (unicità ed obbligatorietà) consente al DBMS di utilizzare l'attributo indicato come elemento identificatore e di accesso ai dati di ogni t-upla della relazione (ogni riga della tabella);

vincolo di dominio

> indica se nel dominio dei valori di un attributo della relazione ce ne siano alcuni che non possono essere assunti; per esempio, si potrebbe desiderare che l'utente non possa inserire (anche per distrazione) valori negativi all'interno di una colonna chiamata «prezzo»

vincolo di tupla

> fissando una tupla, indica se nel dominio dei valori di un attributo della relazione ce ne siano alcuni che non possano essere assunti, a seconda del valore attualmente presente in un altro attributo; per esempio, si potrebbe desiderare che il «prezzo di vendita» di un prodotto sia sempre superiore al suo «costo di acquisto»

L'unico vincolo intrarelazionale che durante la progettazione del livello logico è stato espresso in forma grafica è stato quello di «chiave primaria», utilizzando convenzionalmente la sottolineatura. Gli altri tipi di vincolo si sarebbero potuti esprimere solo a parole (o in SQL).

Nella derivazione di alcune associazioni uno ad uno, si sarebbe potuto obiettare di aver ottenuto tabelle che non rispettavano il «vincolo di unicità», proprio perchè non si era in grado di esprimere questo tipo di vincolo, come invece, per esempio, quello di «chiave primaria».

Vincoli extrarelazionali

Si tratta di vincoli che esprimono regole del modello logico relazionale che riguardano almeno *due* relazioni. Si vedrà un solo vincolo di tipo extrarelazionale:

vincolo di integrità referenziale

> si *deve* applicare ad ogni chiave esterna di una relazione ed impedisce che quella chiave esterna assuma valori non presenti nella chiave primaria a cui essa fa riferimento.

Si è già usato questo vincolo senza spiegarlo a fondo, ed è necessario usarlo ogni volta che si usa una chiave esterna. Purtroppo in alcuni DBMS è possibile escludere questo vincolo e questo genera confusione. Per rappresentare graficamente questo vincolo è possibile, oltre a sopralineare la chiave esterna, realizzare una freccia che collega la chiave esterna ad una chiave primaria.

Operazioni relazionali

Dopo aver compreso che cosa sia una Relazione matematica, si può comprendere meglio quali siano le possibili operazioni che si possono operare tra di esse. Siccome le relazioni non sono altro che dei particolari insiemi, tra le operazioni relazionali, ricadono anche le classiche operazioni insiemistiche di «unione», «intersezione», «differenza», che non verranno trattate.

Per descrivere le altre operazioni relazionali, si utilizzeranno le tabelle ottenute dal test della soluzione dell'esempio I proprietari dei veicoli

Di solito tali operazioni non vengono usate per costruire un database, ma soprattutto per estrarre informazioni da esso

Selezione

Questa operazione si applica ad *una* sola Relazione alla volta e produce come risultato *una* nuova Relazione, si dice quindi che l'operazione selezione è di tipo unario. La relazione risultante ha sempre lo stesso grado della relazione di partenza, ma solitamente cardinalità inferiore. In pratica si può ottenere una nuova tabella che contenga solo alcune delle righe contenute nella tabella di partenza. Questa operazione si indica anche con il simbolo della lettera greca minuscola «sigma», seguito tra parentesi dal nome della relazione, e avente come pedice la condizione di selezione che deve essere verificata

Esempio 4.5. Esempio di selezione

Tabella 4.2. cittadini

cf	cognome	nome
RSSMRA	Rossi	Mario
VRDGPP	Verdi	Giuseppe
BNCGPP	Bianchi	Giuseppe
GLLGNN	Gialli	Gianna

Per visualizzare dalla tabella cittadini, solo quelli di cognome 'Rossi' si scriverà:

$\sigma_{\texttt{cognome='Rossi'}}(\texttt{cittadini})$

Tabella 4.3. σ (cittadini)

cf	cognome	nome
RSSMRA	Rossi	Mario

Proiezione

Questa operazione si applica ad *una* sola relazione e produce come risultato *una* nuova relazione, quindi questa operazione è di tipo *unario*. La relazione risultante ha la stessa cardinalità della relazione di partenza, ma solitamente grado inferiore. In pratica si può ottenere una nuova tabella che contenga solo alcune delle colonne contenute nella tabella di partenza. Questa operazione si indica anche con il simbolo della lettera greca minuscola «pi», seguita, tra parentesi, dal nome della relazione, e avente come pedice la condizione di proiezione che deve essere verificata.

Esempio 4.6. Esempio di proiezione

per visualizzare dalla tabella cittadini, solo le informazioni contenute in «nome», si scriverà:

$\pi_{\texttt{nome}}(\texttt{cittadini})$

Tabella 4.4. π (cittadini)

nome
Mario
Giuseppe
Giuseppe
Gianna

Nota

Nell'ultimo esempio si può notare che nella tabella sono presenti due righe identiche: si tratta di un'eventualità che non si sarebbe dovuta presentare nel caso in cui la tabella fosse stata la rappresentazione di una Relazione matematica.

Questa anomalia consente all'operatore del DBMS di decidere cosa fare: se volere, come in questo esempio, contare quante siano le omonimie, o di aggiungere altri vincoli che rendano la tabella risultante conforme alle Relazioni matematiche.

Congiunzione

Questa operazione si applica a *due* relazione e produce come risultato *una* sola nuova relazione, quindi è un'operazione *binaria*. La relazione risultante ha solitamente grado superiore a quello delle singole relazioni di partenza,

ma nulla si può affermare sulla sua cardinalità.

Per poter eseguire questa operazione si deve verificare una particolare condizione tra le due tabelle di partenza: devono avere almeno una colonna contenente lo stesso tipo di dati; tale condizione si verifica sicuramente quando tra le due relazioni c'è un «vincolo d'integrità referenziale». In pratica da una congiunziome si può ottenere una nuova tabella che possiede le colonne di entrambe le tabelle di partenza, e le righe che hanno un valore in comune ad entrambe le tabelle.

Questa operazione si indica anche con il simbolo della lettera «⋈». Il simbolo deve essere preceduto e seguito dai nomi delle due relazioni coinvolte e deve avere come pedice la condizione di congiunzione che deve essere verificata.

Esempio 4.7. Esempio di congiunzione

visto che la tabella 'cittadini' e la tabella 'proprieta' hanno una colonna in comune (il codice fiscale del cittadino), per visualizzare quali targhe sono abbinate ad ogni cittadino, si può scrivere:

$$\text{proprieta} \bowtie_{\text{cf-cittadino=cf}} \text{cittadini}$$

Per costruire la tabella finale si deve analizzare una riga alla volta la tabella «proprieta» e verificare se il valore di «cf-cittadino» è presente come valore in «cf» di «cittadini». In caso affermativo, bisogna riportare tutti gli altri campi della riga di «cittadini».

Tabella 4.5. proprieta X cittadini

targa-auto	cf	cognome	nome
AZ12345	RSSMRA	Rossi	Mario
AZ12345	VRDGPP	Verdi	Giuseppe
OIL9876	GLLGNN	Gialli	Gianna
EE33333	GLLGNN	Gialli	Gianna

Nota

Nell'ultimo esempio si può notare che nell'ultima tabella ottenuta il campo «cf» non possiede piu' il vincolo di «chiave primaria». Infatti le tabelle «cittadini» e «proprieta X cittadini» sono due tabelle distinte, e nessuno ha scritto da nessuna parte che la seconda relazione debba possedere gli stessi vincoli intrarelazionali della prima.

Lacune del modello relazionale

Tenendo in mente il concetto di relazionale matematica e le definizioni di operazioni relazionali, va osservato che il modello logico non ha una corrispondenza esatta in tutti i DBMS, perchè non tutti i DBMS sono uguali e non tutti rispettano lo standard SQL. Alcuni dei vincoli potrebbero essere presenti, altri no, mentre ne potrebbero esistere alcuni che qui non sono stati studiati e che quindi non appartengono a questo modello. Lo stesso vale per le operazioni relazionali e tutto diverrà più chiaro quando si studierà il linugaggio SQL.

Capitolo 5. Normalizzazione (DA FARE)

Glossario

Indice

B

D

Appendice A. GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall

subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific

section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent

copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties — for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If

there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include

the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See Copyleft.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright © YEAR YOUR NAME

Permission is granted to copy, distribute and/or modify this document under the
terms of the GNU Free Documentation License, Version 1.3 or any later version
published by the Free Software Foundation; with no Invariant Sections, no
Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in
the section entitled “GNU Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with… Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts
being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.