Database (FnF) - Parte 2

Raffaele Daniele Facendola

05/02/12

Indice

1 Istruzioni DML

Il DML è la parte di un linguaggio che si occupa della manipolazione dei dati (Data Manipulation Language). Le tre principali operazioni che è in grado di eseguire sono le seguenti:

- **Inserimento**: inserisce dei record nella base di dati
  - Sintassi SQL: **INSERT INTO** NomeTabella [(Elenco Attributi)] **VALUES** (Elenco Valori)
  - Il campo "Elenco Valori" può anche essere il risultato di una query
- **Modifica:** modifica i valori dei record esistenti che soddisfano una serie di condizioni
  - Sintassi SQL: **UPDATE** NomeTabella **SET** Attributo1 = (Espressione), [Attributo2 = (Espressione)], ... **WHERE** (Condizione)
- **Cancellazione**: cancella dei record esistenti che soddisfano una serie di condizioni
  - Sintassi SQL: **DELETE FROM** NomeTabella [**WHERE** (Condizione)]

2 Vincoli di integrità

I vincoli di integrità sono delle regole che devono essere verificate in ogni momento (durante l'inserimento, la modifica o la cancellazione di record) affinchè i record di una base di dati siano coerenti tra loro e con il contesto applicativo. Se le condizioni del vincolo non sono rispettate l'azione che causerebbe la violazione viene vietata.

- Sintassi SQL: **CREATE ASSERTION** NomeAsserzione **CHECK** (Condizioni)

3 Progettazione di un database

La progettazione di un db può essere suddivisa in tre sottocategorie:

- **Concettuale**: esprime i requisiti di un sistema con una descrizione adatta all'analisi. Un progetto concettuale è:
  - **Formale**: non presenta ambiguità
  - **Integrato**: la descrizione si riferisce all'intero ambiente e non solo ad una parte di esso
  - **Indipendente** dalla realizzazione fisica
- **Logico**: esprime l'organizzazione dei dati dal punto di vista del loro contenuto informativo, descrivendo la struttura di ciascun record e i collegamenti tra record diversi.
- **Fisica**: descrive fisicamente l'organizzazione dei dati sul sistema informativo.

3.1 Dal modello allo schema concettuale

Ciascuna classe di oggetti è chiamata **entità**, esse sono identificate mediante la seguente convenzione 

Un legame logico tra due o più entità (non necessariamente diverse) è detto **relazione** e si identifica mediante la seguente 

Ciascuna entità o relazione possiede delle caratteristiche le quali vengono chiamate **attributi**. Un attributo semplice non chiave si indica con il simbolo 

. Se un attributo singolo è una chiave primaria allora il simbolo usato è 

; se la chiave primaria è costituita da più elementi allora si adotta la seguente convenzione  .

La **cardinalità di una relazione** indica il numero di volte che una data istanza di un entità deve (o può) partecipare alla relazione. La convenzione usata è del tipo (min, max):

- (1,1): obbligatoria, una sola volta
- (1,n): obbligatoria, almeno una volta
- (0,1): opzionale, una sola volta

- (0,n): opzionale, al massimo n volte

La **cardinalità di un attributo**, invece, indica quanti e quali valori può assumere quell'attributo:

- **Scalare**: attributo semplice, un solo campo: 

- **Multivalore**: Il valore del campo può essere scelto tra un set di n elementi. La convenzione usata è (1, n): 

- **Composto**: atrributo composto da più campi: 

- **Multivalore composto**: attributo composto da più campi che possono assumere un valore in un set di n elementi. La convenzione usata è (1,n): 

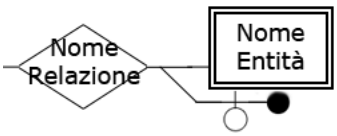
3.1.1 Entità vs Attributi vs Relazioni

- Se il concetto è significativo per il contesto applicativo allora è bene creare un'**entità**.
- Se il concetto è marginale o è descrivibilr in modo semplice allora è bene creare un **attributo**.
- Se il concetto definisce un legame tra diverse entità allora è bene creare una **relazione** (associazione)

3.1.2 Entità deboli

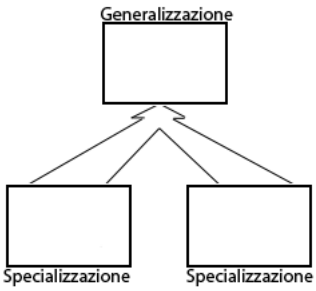
Un'entità si dice **debole** se i suoi attributi non sono sufficienti ad identificarla. In altre parole le entità deboli contengono istanze la cui presenza è ammessa solo se sono presenti determinate istanze di altre entità da cui queste dipendono.

In caso di eliminazione di istanze "principali", anche quelle deboli devono essere cancellate a catena. L'identificativo univoco dell'entità debole contiene l'identificativo delle entità da cui dipende.



3.1.3 Gerarchia ISA

Una gerarchia ISA (dall'inglese "is a", "è un") si ha quando un'entità può "specializzarsi" in altre sotto-entità, o, in altre parole, quando più entità condividono una base comune che può essere svincolata dalle singole entità e trattata come un'unica entità "generale". La convenzione usata è la seguente:



Una gerarchia è essere **totale (t)** se l'unione insiemistica di tutte le sotto-entità è uguale all'entità generale (non può esistere nessuna istanza dell'entità generale che non faccia parte di almeno una specializzata). E' **parziale (p)** se può esistere almeno un istanza che fa parte dell'entità generale che non può far parte di nessuna delle sotto-entità.

Una gerarchia, inoltre, può essere **esclusiva (e)** se ogni istanza può far parte di una ed una sola entità della gerarchia, oppure **overlapping (o)** se una stessa istanza può far parte di più entità contemporaneamente.

3.2 Dallo schema concettuale a quello logico

Il processo per passare dall'uno all'altro è meccanico:

- Eliminazione delle gerarchie ISA
- Normalizzazione degli attributi composti e multipli (non supportati da SQL)
- Traduzione di entità forti in tabelle con gli stessi attributi dell'entità
- Traduzione delle relazioni molti a molti in tabelle con gli stessi attributi della relazione in aggiunta a tutti gli attributi che sono anche chiave primaria di tutte le entità coinvolte nella relazione
- Se una relazione è uno a molti da A a B allora tutti gli attributi che fanno parte della relazione e gli attributi che sono anche chiave primaria dell'entità B vengono aggiunti alla tabella A (la relazione non viene tradotta in tabella)
- Traduzione delle entità deboli in tabelle con gli stessi attributi dell'entià debole cui si aggiungono tutti quelli della chiave primaria dell'entità forte
- Traduzione delle entità deboli

4 Forme normali

4.1 Dipendenze funzionali

Sia R uno schema di una relazione e siano $V \subseteq R$, $W \subseteq R$, v elemento di V e w elemento di W allora esiste una **dipendenza funzionale** e la indichiamo con $V \to W$ se e solo se $t_1[v] = t_2[v] \Rightarrow t_1[w] = t_2[w]$ con t_1 e t_2 due tuple su R.
In questo caso diremo che **V dtetermina W** oppure, equivalentemente, che **W dipende funzionalmente da V**.
NB: Se K è superchiave allora $t_1[k] = t_2[k] \Rightarrow t_1 = t_2$.

4.1.1 Assiomi di Armstrong

- **Riflessività**: $Y \subseteq X \Rightarrow X \to Y$ (la dipendenza si dice banale).
- **Aumento**: $X \to Y \Rightarrow (W, X) \to (W, Y)$
- **Transitività:** $X \to Y \wedge Y \to Z \Rightarrow X \to Z$
- **Unione\Decomposizione**: $X \to Y \wedge X \to Z \Leftrightarrow X \to (Y, Z)$
- **Pseudotransitività**: $X \to Y \wedge (W, Y) \to Z \Rightarrow (W, X) \to Z$

4.2 Prima forma normale (1FN)

Una tabella si dice in prima forma normale se

- I domini degli attributi sono atomici (indivisibili)
- Ogni valore di un attributo deve essere singolo (semplice)

4.3 Seconda forma normale (2FN)

Una tabella si dice in seconda forma normale se

- E' in prima forma normale
- Tutte le chiavi candidate dipendono completamente dalla chiave primaria (e non da parte di essa)

4.4 Terza forma normale (3FN)

Una tabella si dice in terza forma normale se

- E' in seconda forma normale
- $X \to Y$ implica che X è superchiave **oppure** che Y è membro di una chiave della relazione

4.5 Forma normale di Boyce e Codd (BCFN)

Una relazione si dice in forma normale di Boyce e Codd se

- E' in terza forma normale
- $X \to Y$ implica che X è superchiave per la relazione.

5 Transazioni

Una transazione è un'unità di lavoro minima per un database server: essa rappresenta una sequenza di interrogazioni da eseguire in sequenza. Per ogni sistema transazionale (la maggior parte) devono essere sempre garantite le proprietà ACIDe:

- **Atomicità:** l'insieme delle interrogazioni devono essere eseguite come fossero una soltanto: se una fallisce tutte le altre vengono annullate (**rollback**)
- **Consistenza**: nessun vincolo di integrità deve essere violato
- **Isolamento**: Non deve essere influenzata da nessuna transazione concorrente
- **Durabilità**: una transazione andata a buon fine modifica i dati in maniera permanente (**commit work**).

Una transazione inizia con un comando di inizio BOT (**begin of transaction**) e termina con uno di fine EOT (**end of transaction**).

6 Fruizione dei dati

Le modalità con cui un client può richiedere dati ad un database server variano a seconda di quanti agenti sono coinvolti nell'operazione e chi di questi si occupa di gestire la logica dell'applicazione.

Un sistema in cui un client comunica con un db server attraverso la rete senza intermediari viene detto **two-tier**. Se il client comunica con un intermediario (che si occupa di gestire la logica dell'applicazione) allora il sistema viene detto **three-tier**.

Il client di un sistema two-tier si dice **thin** se la logica dell'applicazione è demandata al server, si dice **thick,** invece, se è esso stesso ad espletare le suddette funzioni (con tutti i problemi di sicurezza derivanti).