Basi di Dati – Antonio Vivace

Progettazione Logica

Fasi della progettazione logica

Le attività principali della progettazione logica sono:

- **Ristrutturazione dello schema Entità-Relazione**, è una fase indipendente dal modello logico scelto e si basa su criteri di ottimizzazione dello schema e di semplificaizone della fase successiva.
- **Traduzione verso il modello logico**, fa riferimento a uno specifico modello logico(nel nostro caso il modello relazionale) e può includere una ulteriore ottimizzazione che si basa sulle caratteristiche del modello logico stesso.

L'input della prima fase è costituito dallo schema concettuale e il carico applicativo previsto, in termini di dimensione dei dati e caratteristiche delle operazioni. Il risultao è uno schema E-R ristrutturato, che non è più uno schema concettuale nel senso stretto del termine, in quanto costituisce una rappresentazione dei dati che **tiene conto degli aspetti realizzativi**.
Questo schema e il modello logico scelto costituiscono l'input della seconda fase, che produce lo schema logico della nostra base di dati. In questa seconda fase è possibile effettuare verifiche della qualità dello schema ed eventuali ulteriori ottimizzazioni mediante tecniche basate sulle caratteristiche del modello logico.
Lo schema logico finale, i vincoli di integrità definiti su di esso e la relativa documentazione, costituiscono i prodotti finali della progettazione logica.

Analisi delle prestazioni su schemi E-R

Uno schema E-R può essere modificato per ottimizzare alcuni indici di prestazione del progetto. Parliamo di indici di prestazione e non di prestazioni perchè in realtà le prestazioni di una base di dati non sono valutabili in maniera precise in sede di progettazione logica, in quanto dipendenti anche da parametri fisici, del DBMS che verrà utilizzato e da altri fattori difficilmente prevedibili in questa fase.
Facendo uso di alcune schematizzazioni è comunque possibile effettuare studi di massima dei due parametri che **generalmente** regolano le prestazioni dei sistemi software:

- **costo di una operazione**: valutato in termini di numero di occorrenze di entità e associazioni che mediamente vanno visitate per rispondere a una operazione sulla base di dati;

- **occupazione di memoria**: valutato in termini dello spazio di memoria necessario per memorizzare i dati descritti dallo schema.

Per studiare questi parametri abbiamo bisogno di conoscere lo schema e le seguenti informazioni:

- **Volume dei dati**, ovvero:
  - numero di occorrenze di ogni entità e associazone dello schema;
  - dimensioni di ciascun attributo (di entità o associazione).
- **Caratteristiche delle operazioni**, ovvero:
  - tipo dell'operazione (interattiva o batch);
  - frequenza (numero medio di esecuzioni in un certo intervallo di tempo);
  - dati coinvolti (entità e/o associazioni).

Ristrutturazione di schemi E-R

Questa fase si può suddividere in una serie di passi da effettuare in sequenza

- **Analisi delle ridondanze**. Si decide se eliminare o mantenere eventuali ridondanze presenti nello schema.
- **Eliminazione delle generalizzazioni**. Tutte le generalizzazioni presenti nello schema vengono analizzate e sostituite da altri costrutti.
- **Partizionamento/accorpamento di entità e associazioni**. Si decide se è opportuno partizionare concetti dello schema (entità e/o associazioni) in più concetti o, viceversa, accorpare concetti separati in un unico concetto.
- **Scelta degli identificatori principali**. Si seleziona un identificatore per quelle entità che ne hanno più di uno.

Analisi delle ridondanze

Una ridondanza in uno schema concettuale corrispond alla presenza di un dato che può essere derivato (ovvero ottinibile attraverso una serie di operazioni) da altri dati. In uno schema E-R si possono presentare varie forme di ridondanza.

- Attributi derivabili, occorenza per occorenza, da altri attributi della stessa entità (o associazione);
- Attributi derivabili da attributi di altre entità (o associazioni), di solito attraverso funzoni aggregative;
- Attributi derivabili da operazioni di conteggio di occorrenze;
- Attributi derivabili dalla composizione di altre associazioni in presenza di cicli.

La presenza di un dato derivato presenta il vantaggio di **ridurre** gli accessi necessari per

calcolarlo, ma anche lo svantaggio di una maggiore occupazione di memoria (costo molto spesso trascurabile) e la necessità di effetuare **operazioni aggiuntive** per mantenere il dato derivato aggiornato.

Eliminazione delle generalizzazioni

I sistemi tradizionali per la gestione delle basi di dati non consentono di rappresentare direttamente una generalizzazione. Risulta dunque necessario trasformare questo costrutto in altri costrutti del modello E-R per i quali invece esiste un implementazione naturale: entità e associazioni.

Le seguenti ristrutturazioni eliminano le generalizzazioni:

- **Accorpamento delle figlie della generalizzazione nel genitore**. Le entità figlie vengono eliminate e le loro proprietà (attributi e partecipazioni ad associazioni e generalizzazioni) vengono aggiunte all'entità genitore. A tale entità viene aggiunto un ulteriore attributo che serve a distinguere il "tipo" di una occorrenza, cioè se e a quale figlia apparteneva.
  E' conveniente quando le operazioni non fanno molta distinzione tra le occorrenze e tra gli attributi delle entità. Anche se con questa scelta si consumerà più memoria per la presenza di valori nulli, essa ci assicura un numero minore di accessi rispetto alle altre nelle quali le occorrenze e gli attributi sono distribuiti tra le varie entità.
- **Accorpamento del genitore della generalizzazione nelle figlie**. L'entità genitore viene eliminata e tutti i suoi attributi, il suo identificatore e le sue relazioni a cui tale entità partecipava, vengono aggiunti alle entità figlie.
  Questa alternativa è possibile solo se la generalizzazione è totale, altrimenti le occorrenze dell'entità genitore che non sono occorrenze di alcuna delle entità figlie non sarebbero rappresentabili.
  E' conveniente quando ci sono operazioni che si riferiscono solo a occorrenze delle entità figlie e avremmo un risparmio di memoria rispetto alla prima alternativa perchè, in linea teoria, gli attributi non assumono mai valori nulli. C'è inoltre una riduzione degli accessi rispetto alla prossima scelta descritta perchè non si deve visitare l'entità padre per accedere ad alcuni degli attributi delle entità figlie.
- **Sostituzione della generalizzazione con associazioni**. La generalizzazione si trasforma in due associazioni uno a uno che legano rispettivamente l'entità genitore con le entità figlie. Non ci sono trasferimenti di attributi o associazioni ma vanno aggiunti dei vincoli.
  E' conveniente quando la generalizzazione non è totale (sebbene non sia necessario) e ci sono operazioni che si riferiscono solo a occorenze delle entità figlie o a solo occorrenze dell'entità genitore. Abbiamo un risparmio di memoria

rispetto alla prima alternativa per l'assenza di valori nulli, ma c'è un incremento degli accessi per mantenere la consistenza delle occorrenze rispetto ai vincoli introdotti.

La scelta tra le varie alternative deve essere fatta considerando gli scenari e vantaggi e svantaggi di ognuna delle scelte possibili relativamente alla **occupazione di memoria e al costo delle operazioni coinvolte**.

In alcuni casi critici vanno però considerati **altri fattori** quali le dimensioni dei domini degli attributi e la quantità di dati che è possibile recuperare con una sola operazione di accesso a memoria secondaria.

Partizionamento/accorpamento di concetti

Entità e associazioni in uno schema E-R possono essere partizionati o accorpati per garantire una maggior efficienza delle operazioni in base al seguente principio: gli accessi si riducono separando attributi di uno stesso concetto che vengono acceduti da operazioni diverse e raggrupando attributi di concetti diversi che vengono acceduti dalle medesime operazioni. Le stesse tecniche discusse per la l'analisi delle generalizzazioni possono essere usate per prendere decisioni di questo tipo.

Partizionamenti di entità

Questa ristrutturazione è conveniente se le operazioni che coinvolgono frequentemente l'entità originaria richiedono attributi solo in una partizione alla volta. Un partizionamento di questo tipo è un esempio di *decomposizione verticale* di un'entità, nel senso che suddivide il concetto operando sui suoi attributi. E' comunque possibile una decomposizione orizzontale, nelle quali la suddivisione avviene sulle occorrenze dell'entità (questi partizionamenti hanno però l'effetto collaterale di dover duplicare tutte le associazioni in cui l'entità originaria partecipa).

Eliminazione di attributi multivalore

Questa ristrutturazione si rende necessaria perchè, come per le generalizzazioni, il modello relazione non permette di rappresentare in maniera diretta un attributo di tipo multivalore.

Accorpamento di entità

L'accorpamento è l'operazione inversa del partizionamento, viene di solito effettuato su associazioni di tipo uno a uno, raramente su associazioni uno a molti e praticamente mai su relazioni molti a molti (queste ultime generano ridondanze). E' facile dimostrare che si possono presentare ridondanze su attributi non chiave dell'entità che partecipava all'associazione originaria con una cardinalità massima pari a N.

Altri tipi di partizionamento/accorpamento
I concetti di partizionamento e accorpamento possono essere applicati anche alle associazioni. Può convenire **decomporre** un'associazioni tra due entità in due (o più) associazioni tra le medesime entità, per separare occorrenze dell'associazioni originale accedute sempre separatamente e, viceversa, **accorpare** due (o più) associazioni tra le medesime entità (che si riferiscono però a due aspetti dello stesso concetto) in un'unica associazione, quando le relative occorrenze vengono sempre accedute contemporaneamente.

Scelta degli identificatori principali

La scelta degli identificatori principali è essenziale nelle traduzioni verso il modello relazionale perchè in questo modello le chiavi vengono usate per stabilire legami tra dati in relazioni diverse. Inoltre i DBMS richiedono generalmente di specificare una chiave primaria sulla quale vengono costruite automaticamente delle strutture ausiliarie, dette indici, per il reperimento efficiente dei dati. Nei casi in cui esistono entità per le uqali sono stati specificati più identificatori, **bisogna decidere quale di questi identificatori verrà utilizzato come chiave primaria**.
I criteri di decisione per questa scelta sono i seguenti.

- **Gli attributi con valori nulli non possono costituire identificatori principali**. Tali attributi non garantiscono l'accesso a tutte le occorrenze.
- **Un identificatore composto da uno o da pochi attributi è da preferire a identificatori costituiti da molti attributi**. Questo garantisce che le strutture ausiliarie create per accedere ai dati (gli indici) siano di dimensioni ridotte, permette un risparmio di memoria nella realizzazione dei legami logici tra le varie relazioni e facilita le operazioni di join.
- **Un identificatore interno con pochi attributi è da preferire a un identificatore esterno, che magari coinvolge diverse entità**. Gli identificatori esterni vengono tradotti in chiavi che includono gli identificatori delle entità coinvolte nell'intificazione esterna: chiaramente in questa maniera si possono generare chiavi con molti attributi.
- **Un identificatore che viene utilizzato da molte operazioni per accedere alle occorrenze di un'entità è da preferire rispetto agli altri**. In questa maniera infatti tali operazioni possono essere eseguite efficientemente perchè possono trarre vantaggio dagli indici creati automaticamente dal DBMS.

Se, a questo punto, nessuno degli identificatori candidati soddisfa tali requisiti, è possibile pensare di introdurre un ulteriore attributo dell'entità: questo attributo conterrà valori speciali (detti *codici*) generati appositamente per identificare le occorrenze delle entità.
E' inoltre possibile definire degli *indici secondari*, ovvero quegli identificatori non

selezionati come prinicpali ma utilizzati da qualche operazione per accedere ai dati.

Traduzione verso il modello relazionale

Questa fase della progettazione logica consiste in una traduzione tra modelli di dati diversi: a partire da uno schema E-R ristrutturato si costruisce uno schema logico equivalente, in grado cioè di rappresentare le medesime informaizioni.

Entità e associazioni molti a molti

La traduzione naturale nel modello relazionale di questo caso prevede:

- per ogni entità, una relazione con lo stesso nome avente per attributi i medesimi attributi dell'entità e per chiave il suo identificatore;
- per l'associazione, una relazione con lo stesso nome avente per attributi gli attributi dell'associazione e gli identificatori delle entità coinvolte; tali identificatori formano la chiave della relazione.

Se gli attributi originali di entità o associazioni sono opzionali, i corrispondenti attributi di relazione possono assumere valori nulli.

Associazioni uno a molti

todo

Entità con identificatore esterno

todo

Associazioni uno a uno

todo

Title 1

Title 2

Title 3

Title 4

Paragraph Text

`Monospaced Text`