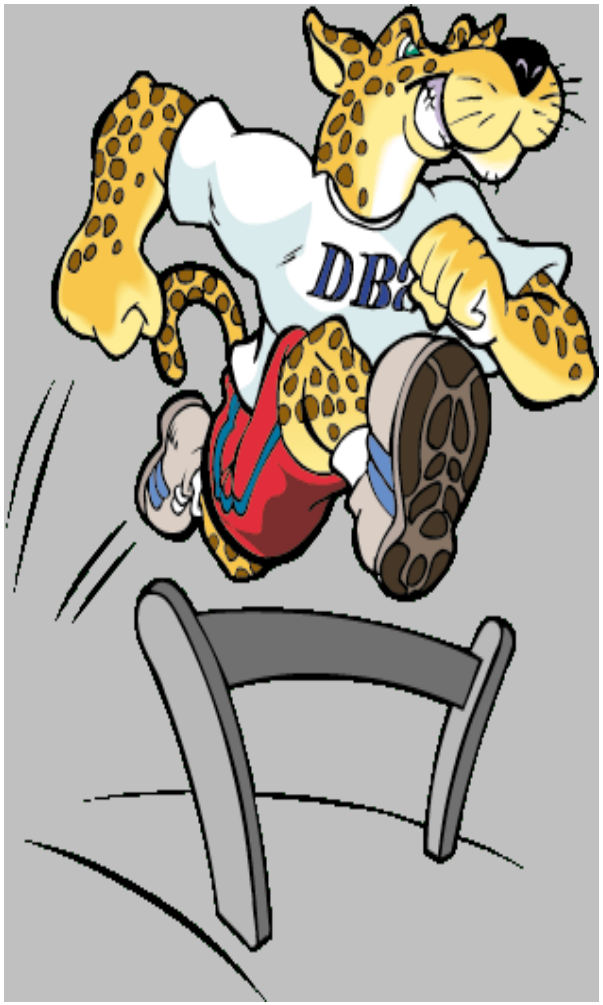
DB2 Data Management Software



Il linguaggio SQL

I Sistemi di Gestione di Basi di Dati

Paolo Avallone
Sr Consulting – IT Specialist
DB2, Data Management

Marzo 2004

LEGGERE LE SEGUENTI ATTENZIONI

- Le informazioni contenute in questa presentazione si basano su tecniche, algoritmi, e documentazioni pubblicate da diversi autori e società, oltre ad essere risultato di ricerche. Questa presentazione è soggetta a cambiamenti nel tempo.

- Le informazioni contenute in questa presentazione non sono sottoposte ad alcun test formale, esse sono distribuite "così come sono" senza alcuna garanzia, nè esplicita o implicita.

- L'utilizzo di queste informazioni oppure l'implementazione di alcune di queste tecniche è responsabilità del cliente e dipende dalla sua abilità di valutazione la decisione se integrare queste tecniche nel suo ambiente operativo.

I Sistemi di Dati

Partiamo subito da una semplice ma importantissima riflessione: i dati sono la risorsa principale per qualunque organizzazione.
Quindi per poter meglio utilizzare queste risorse possiamo dire che:

- i dati devono poter essere utilizzati da diversi utenti (*Data Sharing*);
- i dati non devono essere logicamente sovrabbondanti (*Data Redundancy*);
- i programmi che li gestiscono devono essere indipendenti dalla loro sistemazione fisica (*Data Indipendence*).

Un sistema di dati che ha tali caratteristiche si chiama DBMS (*Data Base Management System*).

I Sistemi di Dati - caratteristiche

- **DATA SHARING**
  - Con questo termine si intende il processo mediante il quale diverse applicazioni utilizzano dati comuni, scambiandosi tutte le informazioni relative alla variazione di ciascun dato

- **DATA REDUNDANCY**
  - Gestire i dati in modo tradizionale, comporta una proliferazione di archivi con conseguente ridondanza di dati. La necessita' di tenere sotto controllo questo fenomeno e' un ottimo motivo per passare alla tecnologia DBMS

- **DATA INDIPENDENCE**
  - In tutti i sistemi informatici il numero dei programmi e i tipi di dati e le relazioni cambiano nel tempo. La tecnologia DBMS consente di scorporare la parte di definizione dei dati in modo che programmi e dati potranno essere incrementati o decrementati senza interessare il DBMS

I Sistemi di Dati – il modello gerarchico

Il primo sistema di gestione di base di dati apparso sui mercati era di tipo gerarchico; alla fine degli anni '60 IBM mise in commercio IMS/1 sui sistemi 360. Successivamente altri sistemi si allinearono al modello gerarchico:

- System 2000
- Tdms
- Mark IV
- RFMS

Nel sistema gerarchico gli oggetti sono rappresentati mediante i segmenti e i campi. Il campo è la più piccola unità di dato al quale viene associato un nome. Un segmento corrisponde a un insieme di campi. I rapporti tra segmenti sono di tipo 1:N, a 1 segmento padre sono associati N figli. Ogni segmento figlio possiede 1 segmento padre unico.
Il linguaggio di manipolazione dei dati gerarchici è chiamato DL/1.

I Sistemi di Dati – il modello gerarchico (2)

- **Vantaggi**

  - buone prestazioni
  - diffusa commercializzazione
  - lo schema riflette i bisogni dell'utente

- **Svantaggi**

  - mancanza di legami (N:M) tra entità
  - costo proibitivo di eventuali trattamenti non previsti a priori, poichè non riflettono la struttura dati
  - povertà di interfacce utente non procedurali. L'utente si posiziona nella base di dati costruendo progressivamente il risultato della propria richiesta (navigazione)

I Sistemi di Dati – il modello reticolare

Fautori di questo modello furono Charles BACHMAN e il gruppo DBTG (Data Base Task Group) del comitato CODASYL nel 1971, che svilupparono questo modello a partire da un prodotto pre-relazionale (a volte definito "reticolare") creato dallo stesso BACHMAN alla General Electric: questo prodotto era l'IDS (Integrated Data Store).

Altri sistemi in commercio sono:

- IDS II (Bull)
- IDMS (Cullinane)
- EDMS (Xerox)
- DMS/1100 (Univac)
- DBMS-10/20 (Digital)
- PHOLAS (Philips)
- TOTAL (Cincom)

I Sistemi di Dati – il modello reticolare (2)

- **Vantaggi**

  - buone prestazioni
  - buona diffusione commerciale
  - buona rappresentazione dei legami N:M tra entità

- **Svantaggi**

  - l'inconveniente maggiore deriva dal linguaggio di manipolazione di dati. Esso è di tipo procedurale; l'accesso ai dati, quindi la manipolazione nella rete logica, costituita da registrazioni e puntatori, è a carico dell'applicazione

I Sistemi di Dati – il modello relazionale

Nel 1970 Edgar F. Codd, allora ricercatore IBM del laboratorio di San Josè in California, pubblicò uno studio dal titolo *"A relational model of large shared data banks"*, (Un modello relazionale di dati per grandi banche dati condivisibili).

Questo studio proponeva un nuovo metodo generale di gestire i dati, basato sulla semplicità delle righe e delle colonne di una tabella.

I concetti e la novità introdotti da Codd erano rivoluzionari nel 1970, e in netto contrasto con l'approccio allora prevalente di sfruttare le possibilità offerte dalle diverse architetture hardware e software.

I Sistemi di Dati – il modello relazionale (2)

Da quel momento Codd condusse una vera e propria battaglia su più fronti:

- contro il management IBM che vedeva la sua proposta come una minaccia mortale all'IMS, il database gerarchico considerato "stategico" da Big Blue;
- contro i grandi utenti e i loro "esperti", che vedevano il modello relazionale, come una minaccia al monopolio delle conoscenze sulla gestione dei dati;
- infine contro i concorrenti IBM, che vedevano avviato gli studi per la definizione di uno standard da contrapporre all'IMS e che vedevano nelle teorizzazioni di Codd un tentativo mascherato da parte IBM di confondere le acque dal punto di vista teorico.

Da quel momento la strada per Codd fu tutta in discesa (o quasi), finchè altre società (ad esempio ORACLE e Relational Technology) cominciarono a mettere a punto un DBMS basato sul modello relazionale, in ambiente mini. Successivamente anche l'IBM riconosceva i suoi errori e avviava uno sviluppo di quello che poi sarebbe divenuto il DB2.

I Sistemi di Dati – il modello relazionale (3)

Definizioni

I dati di un sistema relazionale sono rappresentati sotto forma di tabelle di valori chiamate RELAZIONI. Una linea di una relazione è una TUPLA, una colonna caratterizzata da un nome è chiamata ATTRIBUTO, il numero di TUPLE di una relazione prende il nome di CARDINALITA'.

La concezione di una base dati relazionale è un processo complesso consistente nella definizione, a partire dall'osservazione del mondo reale, di uno schema relazionale normalizzato.

Tra i vari metodi di concezioni della base dati relazionale ne citiamo due tra i più importanti, tra loro complementari: l'approccio sintetico, detto anche Top-Down, ha per obiettivo l'ottenimento di uno schema relazionale, eventualmente non normalizzato, a partire da un insieme di entità-statiche modellizzanti al meglio la realtà da descrivere.

L'approccio analitico, detto anche Bottom-Up, mette in gioco il processo di normalizzazione a partire da uno schema relazionale preesistente.

I Sistemi di Dati – l'SQL

L'SQL è un linguaggio standard per fare interrogazioni e aggiornamenti di DBMS, sia centralizzati che ad architettura client server.
Il suo nome originale era SEQUEL, in alternativa a SQUARE (Specifyng QUeries As Relational Expression). L'acronimo iniziale significava Structured English Query Language. Successivamente la parola English è stata eliminata e si arrivò all'attuale SQL (Structured Query Language).
L'SQL fornisce un'interfaccia utente completa verso un ambiente relazionale ed è capace di rendere disponibili le seguenti funzioni:

- data definition
- data retrieval
- data modification
- data control and security

Pertanto, il linguaggio fornisce tutti gli strumenti necessari per creare e mantenere una base dati relazionale.

I Sistemi di Dati – Il Data Manipulation Language

Il DML, il linguaggio per la manipolazione dei dati, gestisce le funzioni di recupero, inserimento, aggiornamento e cancellazione delle righe di una tabella. Per ottenere cio' sono disponibili i seguenti comandi:

- SELECT (esecuzione di query nel DB)
- UPDATE (modifica tuple del DB)
- DELETE (cancella tuple dal DB)
- INSERT (inserisce nuove tuple nel DB)

INSERT può usare il risultato di una query per eseguire gli inserimenti multipli
DELETE e UPDATE possono fare uso di condizioni per specificare le tuple da cancellare e modificare

I Sistemi di Dati – L'istruzione SELECT

Scopo: L'istruzione SELECT recupera una o più righe da una specifica tabella o tabelle. Solo le righe che soddisfano le condizioni di ricerca sono recuperate.

Formato: **SELECT**
(**ALL/DISTINCT**)
select-list (*nomi colonne*) / *
FROM *indicazione delle tabelle*
(**WHERE** *condizioni di ricerca*)
(**GROUP BY** *nomi colonne*)
(**HAVING** *condizioni di ricerca*)
(**ORDER BY** *indicazione colonne da sortare*)

I Sistemi di Dati – L'istruzione SELECT (2)

ALL / DISTINCT

ALL (è il valore di default) recupera tutte le righe che soddisfano le condizioni di ricerca.

DISTINCT elimina le righe duplicate (tutte le righe dove le colonne specificate sono identiche).

select-list - Specifica i dati che devono essere recuperati. La lista è composta da uno o più elementi, dove ogni elemento può essere:

- un nome colonna
- una costante
- una funzione SQL
- una espressione aritmetica

L'ordine degli elementi può essere specificato liberamente. Quando la clausola FROM specifica più di una tabella (join) è necessario qualificare i nomi delle colonne identiche.

I Sistemi di Dati – L'istruzione UPDATE

Scopo: l'istruzione UPDATE è usata per aggiornare i valori di una o più tabelle. Le righe che devono essere aggiornate sono scelte dalle condizioni di ricerca. Se non è data nessuna condizione di ricerca, tutte le righe nella tabella specificata sono aggiornate.

Formato: **UPDATE** *nome-tabella*
SET *nome-colonna-1=espressione-1*
(*,nome-colonna-2=espressione-2*)
(**WHERE** *condizione di ricerca*)

nome-tabella - Specifica il nome della tabella che contiene i dati che devi aggiornare.
SET - Specifica i nuovi valori di una o più colonne. Tali valori possono essere costanti, espressioni oppure NULL. Una espressione può contenere costanti, nomi colonna, e gli operatori aritmetici **+**, **-**, *****, e **/**.
WHERE - Specifica le condizioni atte a determinare le righe che devono essere aggiornate.

I Sistemi di Dati – L'istruzione DELETE

Scopo: l'istruzione DELETE è usata per cancellare una o più righe in una tabella. Le righe che devono essere cancellate sono scelte dalle condizioni di ricerca. Se non è data nessuna condizione di ricerca, tutte le righe nella tabella sono cancellate.

Formato: **DELETE FROM** *nome-tabella*
(**WHERE** *condizione-di-ricerca*)

FROM nome-tabella - Specifica il nome della tabella che contiene i dati da cancellare.

WHERE condizioni-di-ricerca - La clausola WHERE specifica le condizioni che si usano per determinare le righe da cancellare.

I Sistemi di Dati – L'istruzione INSERT 1

Scopo: L'istruzione INSERT è usata per inserire una singola riga oppure righe multiple in una tabella esistente in un data base DB2. Esistono due formati base. Il formato 1 inserisce una nuova riga in una tabella.

Formato 1: **INSERT INTO** *nome-tabella (lista-delle-colonne)*
VALUES *(lista-dei-valori)*

INTO nome-tabella - Specifica il nome della tabella nella quale deve essere inserita la nuova riga.

lista-delle-colonne - Questa lista specifica le colonne nella quali i valori vengono inseriti. Tutti i campi non inclusi nella lista avranno valore NULL. Se i dati sono inseriti in tutte le colonne non è necessario specificare la lista.

VALUES - Values specifica l'ordine delle colonne separate dalla virgola.

I Sistemi di Dati – L'istruzione INSERT 2

Scopo: Il formato 2 dell'istruzione INSERT è usato per inserire in una tabella esistente righe che sono state selezionate o calcolate da un data base DB2 con l'istruzione SELECT.

Formato 2: **INSERT INTO** *nome-tabella (lista-delle-colonne)*
SELECT *nomi-colonna* **FROM** *nome-tabella*
WHERE *condizioni-di-ricerca*

INTO (Come il formato 1).
lista-delle-colonne (Come il formato 1).
SELECT nomi-colonna – Lista delle colonne da recuperare. Devono coincidere con quelle specificate nella clausola INTO.
FROM nome-tabella - La clausola FROM specifica la tabella o le tabelle DB2 dalle quali i dati devono essere recuperati.
WHERE condizioni di ricerca - La clausola WHERE contiene le condizioni di ricerca, che determinano la selezione delle righe dalle tabelle specificate.

I Sistemi di Dati – Politiche di "reazione"

Anziché lasciare al programmatore il compito di garantire che a fronte di cancellazioni e modifiche i vincoli di integrità referenziale siano rispettati, si possono specificare opportune politiche di reazione in fase di definizione degli schemi

```
CREATE TABLE Imp (
CodImp char(4) PRIMARY KEY,
Sede char(3),
...

FOREIGN KEY Sede REFERENCES Sedi
ON DELETE CASCADE          -- cancellazione in cascata
ON UPDATE NO ACTION        -- modifiche non permesse
```

Altre politiche: SET NULL e SET DEFAULT

I Sistemi di Dati – Conclusioni

- Un DBMS è un sistema software (complesso!) che gestisce grandi quantità di dati persistenti e condivisi, garantendone l'integrità e la sicurezza mediante l'esecuzione coordinata delle richieste, la protezione da malfunzionamenti e la realizzazione di una politica degli accessi
- Un modello dei dati è una collezione di concetti che vengono utilizzati per descrivere i dati, le loro associazioni, e i vincoli che questi devono rispettare
- Una base di dati si compone di uno schema, che ne descrive la struttura logica, e di un'istanza, che consiste dei dati memorizzati
- Mediante una organizzazione a 3 livelli, un DBMS permette di ottenere gradi di indipendenza fisica e logica dei dati

I Sistemi di Dati – Conclusioni

- Il linguaggio SQL è lo standard de facto per interagire con DB relazionali
- Si discosta dal modello relazionale in quanto permette la presenza di tuple duplicate (tabelle anziché relazioni)
- La definizione delle tabelle permette di esprimere vincoli e anche di specificare politiche di reazione a fronte di violazioni dell'integrità referenziale
- Il comando SELECT seleziona le colonne dalle righe della tabella referenziata nel comando FROM
- La clausola WHERE e' opzionale; quando viene usata permette di ridurre l'insieme di TUPLE (righe) interessate dalla ricerca o modifica o inserimento
- I dati recuperati possono essere organizzati utilizzando la clausola GROUP BY che riorganizza la tabella in gruppi
- La clausola HAVING filtra il raggruppamento
- La clausola ORDER BY ordina in modalita' ascendente/discendente i valori di una o piu' colonne
- L'opzione DISTINCT permette di eliminare le righe duplicate