Sistemi di Elaborazione delle Informazioni

Informatica II

Ing. Mauro Iacono
Seconda Università degli Studi di Napoli
Facoltà di Studi Politici e per l'Alta Formazione Europea e Mediterranea "Jean Monnet"
-
PARSeC Research Group

Macchine per l'elaborazione dell'informazione

Parte prima: Fondamenti di basi di dati

Basi di dati

(Fadini-Savy cap. 8)

Cosa è una base di dati

- I dati relativi ad una certa applicazione sono tra loro interdipendenti
- E' necessario poter disporre dei dati per poterli utilizzare
- I dati costituiscono un patrimonio
- Avere dati non utilizzabili o difficilmente accessibili equivale a non averli
- Una base di dati è una collezione di dati correlati organizzati in maniera da poter essere memorizzati, gestiti ed elaborati efficacemente e efficientemente
- Per gestire le basi di dati si usano software detti DBMS (Data Base Management System)

Modello dei dati

- Per rappresentare correttamente l'informazione tramite i dati si usano appositi modelli con particolari proprietà desiderate
- Modelli di rappresentazione (usati per la implementazione delle basi di dati): sono detti *modelli logici*
  - Modello relazionale (universalmente adottato), modello gerarchico, modello reticolare (obsoleti)
- Modelli di analisi (usati per la progettazione delle basi di dati): sono detti *modelli concettuali*
  - Modello ER (Entità-Relazione), modello ad oggetti

DBMS

- Architettura strutturata a 2 livelli:
  - livello interno (struttura fisica degli archivi, memorizzazione, tecniche interne per la ricerca dei dati, efficienza del sistema, ...)
  - livello esterno (struttura logica dei dati così come vista dagli utenti – modello relazionale, in genere)
- Teoria delle basi di dati: tratta il livello esterno
- Alcuni DBMS:
  - IBM DB/2, Informix, Oracle, MySQL, PostgreSQL, Microsoft SQL Server
- Linguaggi per l'interrogazione:
  - SQL (standard)
  - QBE (Query By Example)

Modello relazionale

- Fondato sul concetto matematico di *relazione* o corrispondenza tra insiemi
  - dal punto di vista pratico: lega tra loro alcuni dati per rappresentare informazione
  - *Schema*: Nome e Cognome – Data di Nascita – Indirizzo – Codice fiscale
  - *Istanza*: “Gennaro Esposito” - 1/1/1970 - “via Roma 2 Napoli” - “GNRSPS70A01F839V”
  - Ogni informazione è costituita nell'ordine da una *stringa di caratteri*, una *data*, una *stringa di caratteri*, una *stringa a lunghezza fissa di 16 caratteri con formato*, ognuna delle quali legata a un *attributo* dello schema (in base all'ordine)
- Le relazioni si rappresentano in forma di *tabelle*
- *Una base di dati relazionale è un insieme di relazioni*

Chiavi

- Un insieme di attributi è detto *chiave* di una relazione se identifica univocamente una istanza della relazione
  - se un insieme di attributi è una chiave, non esistono combinazioni di valori ripetute su tali attributi
  - possono esistere più chiavi per una stessa relazione
- Ogni relazione è caratterizzata da una e una sola *chiave primaria*
  - è una chiave come le altre, scelta tra tutte dal progettista per questioni di convenienza (valori numerici, minimo numero di attributi, significato semantico importante nel modello)
  - nel caso in cui la relazione ammetta una sola chiave la scelta è banale

Vincoli di integrità referenziale

- Detti anche *relazioni tra tabelle* se si usa il termine “tabella” in luogo di “relazione” (vedere Fadini-Savy)
- Si ricorre ad essi per ricostruire informazioni complesse a partire dalle informazioni elementari contenute in ciascuna relazione
  - il più comune è il *vincolo di chiave esterna*: si tratta di una corrispondenza stabilita tra uno o più attributi di una relazione e la chiave di un'altra (composta ovviamente dallo stesso numero di attributi degli stessi tipi nell'ordine) realizzate tramite valori uguali
  - detto anche vincolo 1 a molti, perchè consente di legare a un certo valore della chiave di una relazione molti valori di attributi della relazione su cui sussiste il vincolo
- Tra più relazioni può crearsi un vincolo *molti a molti*

Esempio

- Semplice base di dati per una casa editrice
  - Ipotesi (non banali):
    - ogni autore può scrivere più libri
    - ogni libro può avere più autori
    - ogni contratto lega un libro ai suoi autori
    - ogni responsabile può esserlo per più generi
    - più generi possono essere collocati insieme

Autori

NOME	NATO	INDIR	CF
N1	D1	I1	F1
N2	D2	I2	F2
N3	D3	I3	F3
N4	D4	I4	F4
N5	D5	I5	F5
N6	D6	I6	F6

Autori-Libri

CF	CODLIB
F1	C1
F1	C2
F1	C4
F2	C2
F2	C3
F3	C4
F4	C4
F5	C5

Libri

CODLIB	TITOLO	COSTO	CONTR	GENERE
C1	T1	LC1	CT1	G1
C2	T2	LC2	CT2	G2
C3	T3	LC3	CT3	G1
C4	T4	LC4	CT4	G3
C5	T5	LC5	CT5	G2

Genere

NOME	COLLOC	RESP
G1	L1	R1
G2	L2	R2
G3	L3	R3
G4	L4	R4

Esempio: chiavi primarie

- La base di dati è formata da 4 relazioni
- Le chiavi delle 4 relazioni sono NOME (che si ipotizza unico tra gli autori) e CF per Autori, CODLIB e CONTR (unico per ogni libro) per Libri, la coppia CF-CODLIB per Autori-Libri, GENERE per genere
- Vengono scelte tra le chiavi CF per Autori e CONTR per Libri come chiavi primarie (per le altre relazioni la scelta è ovvia)
- Si usa la notazione [nomerelazione].[nomeattributo] per identificare univocamente gli attributi

Esempio: vincoli

- Sussistono i vincoli di chiave esterna:
  - Autori-Libri.CF->Autori, Autori-Libri.CODLIB->Libri, Libri.GENERE->Genere
  - Non si indica l'attributo destinazione in quanto si tratta necessariamente della chiave della relazione
  - Tramite più relazioni legate da vincoli uno a molti si ottengono vincoli *molti a molti*

Definizione di una base di dati

- Una base di dati va *progettata* perchè la struttura deve essere tale da evitare *ridondanza dei dati*
- La *progettazione logica*, ovvero dello schema relazionale, consiste nel definire in base a criteri di ottimalità:
  - lo schema delle relazioni (ovvero gli attributi che la caratterizzano e i relativi tipi)
  - la chiave primaria
  - gli eventuali vincoli (tra cui quelli di chiave esterna)
- Una volta definita la struttura, si può implementare nel DBMS tramite il Data Definition Language (SQL)

Esempio di definizione

- Definizione della relazione Autori in SQL:
  - CREATE TABLE Autori

```
(CF          CHAR(16),
 Nome        CHAR VARYING(30),
 DataNasc    DATE,
 Indirizzo CHAR VARYING(50),
 PRIMARY KEY(CF)
)
```

- In maiuscolo sono state evidenziate per chiarezza le *parole chiave* del linguaggio SQL, ma il linguaggio non è *case sensitive*

Esempio di definizione

- Definizione della relazione Autori-Libri in SQL:
  - CREATE TABLE Autori-Libri

```
(CF          CHAR(16),
 CODLIB      CHAR(18),
 PRIMARY KEY(CF, CODLIB)
 FOREIGN KEY(CF) REFERENCES Autori,
 FOREIGN KEY(CODLIB) REFERENCES Libri
)
```

- In questo esempio si evidenzia la definizioni di vincoli di chiave esterna

Indici

- Per velocizzare le ricerche, il DBMS permette di definire insieme al modello della base di dati anche gli *indici*
- Gli indici non fanno parte del modello logico, ma in fase di implementazione velocizzano l'accesso ai dati in base ai criteri di ricerca più frequenti (sulla base dei quali sono creati dall'amministratore)
- Gli indici possono essere pensati come tabelle aggiuntive al DB (pertanto occupano spazio aggiuntivo su disco)
- Un indice su una chiave facilita le ricerche, un indice su un attributo individua sottogruppi delle relazioni

Interrogazioni

- Una volta che un DB è stato definito e creato inizia il suo utilizzo: è necessario inserire i dati di interesse per poi utilizzarli alla bisogna
- E' possibile eseguire sullo schema definito quattro tipi di operazioni (*query*):
  - interrogazioni (*select*)
  - inserimenti (*insert*)
  - cancellazioni (*delete*)
  - aggiornamenti (*update*)
- La parte di Data Manipulation Language (DML) di SQL permette di effettuare tutti i 4 tipi

Interrogazioni (select)

- Le operazioni di interrogazione (di tipo select) non alterano il contenuto del DB ma consentono di accedere ai dati di interesse
- Ci sono 3 tipi di interrogazioni base che possono anche essere combinati tra loro:
  - selezione
  - proiezione
  - join
- In SQL vengono realizzati tutti con l'unica istruzione select

Selezione

- Una operazione di selezione prende solo alcune “righe” di una “tabella”, ovvero quelle che soddisfano una *condizione* specificata nell'operazione
- Una selezione da una rubrica telefonica di tutti i contatti di nome “Mauro” restituisce una tabella con tutti i dati di tutti i Mauro presenti e basta
- La sintassi SQL (supponendo di avere una relazione di nome Rubrica con un attributo Nome) è:

  SELECT * FROM Rubrica WHERE Nome=”Mauro”

- * indica “tutti gli attributi”

Condizioni di selezione

- La condizione deve avere un valore VERO o FALSO
- Nelle condizioni atomiche di possono usare operatori di confronto:
  - = per uguaglianza (Nome = “Mauro”)
  - > e < su tipi ordinati (Stipendio > 500)
  - LIKE per confronto tra stringhe, con ? per una lettera generica e * per una sottostringa generica (Nome LIKE “Mar?o” - trova Mario e Marco, Nome LIKE Mar*o – trova Mario, Marco e Marcello)
- Usa connettivi logici AND, OR e NOT per ottenere condizioni complesse da condizioni atomiche
  - ((Nome = “Mauro”) AND (Età>30)) OR (Nome LIKE Mar?o)

Proiezione

- Una operazione di proiezione prende solo alcune “colonne” di una “tabella”, fornite nell'operazione
- Una proiezione da una rubrica telefonica degli attributi Nome, Cognome e Numero restituisce una tabella con soli nome, cognome e numero di tutti
- La sintassi SQL (supponendo di avere una relazione di nome Rubrica con un attributo Nome) è:

```
SELECT DISTINCT Nome, Cognome, Numero
    FROM Rubrica
```

- DISTINCT evita i duplicati che si possono creare

Join

- Permette di ricostruire una informazione complessa combinando informazioni da più relazioni
- L'informazione viene ricostruita combinando (parte delle) “righe” di una relazione con (parte delle) “righe” di un'altra
- Il criterio di combinazione è una condizione imposta tra valori di un certo numero di attributi di una relazione e un ugual numero di attributi dell'altra (curando la corrispondenza dei tipi)
- In genere sugli attributi conivolti vige un vincolo di chiave esterna
- Si costruisce una “riga” per ogni combinazione possibile

Esempio

- Ricostruiamo il rapporto tra docenti e corsi a partire dalle informazioni sui docenti e quelle sui corsi
  - Attenzione! La relazione join contiene ridondanze!

Docenti

MATR	NOME	COGNOME	INTERNO	RUOLO
10005	Antonio	Esposito	F	Ricerc
9981	Enrico	Coppini	V	Ordin
10009	Emilio	Coppetiello	V	Ricerc
8021	Claudio	Raimondi	F	Assoc
9000	Oreste	Antonelli	V	Ricerc
10013	Antonio	Esposito	F	Assoc

Corsi

NOME	ANNO	DOCENTE
Informatica	1	10005
Inglese	1	9981
Francese	2	8021
Letteratura italiana	2	10013
Storia	1	10009
Informatica giuridica	3	10005
Storia contemporanea	2	10009

Vincolo ch. est.: Corsi.DOCENTE->Docenti

Docenti JOIN Corsi (SELECT * FROM Docenti, Corsi WHERE Docenti.MATR = Corsi.DOCENTE)

MATR	NOME	COGNOME	INTERNO	RUOLO	NOME	ANNO
10005	Antonio	Esposito	F	Ricerc	Informatica	1
10005	Antonio	Esposito	F	Ricerc	Informatica giuridica	3
9981	Enrico	Coppini	V	Ordin	Inglese	1
10009	Emilio	Coppetiello	V	Ricerc	Storia	1
10009	Emilio	Coppetiello	V	Ricerc	Storia contemporanea	2
8021	Claudio	Raimondi	F	Assoc	Francese	2
10013	Antonio	Esposito	F	Assoc	Letteratura italiana	2

Join completo

- Nell'esempio precedente una “riga” di Docenti non partecipa alla creazione di “righe” nella relazione join in quanto non esiste una “riga” della seconda che soddisfa la condizione
  - N.B.: Essendoci il vincolo, non è possibile che la situazione si verifichi a ruoli invertiti tra le relazioni
- Qualora tutte le “righe” di entrambe le relazioni partecipino alla creazione di almeno una “riga” della relazione join, si parla di *join completo*
- Se il join non è completo, disponendo della sola relazione join si perdono le informazioni relative alle “righe” che non hanno partecipato

Insert, delete, update

- Per inserire dati (nuovi) in una relazione, cancellare dati o modificarli
- Inserimento: in SQL si usa l'istruzione INSERT
  - INSERT INTO Corsi (Nome, Anno, Docente) VALUES (“Musica”, 3, 9000)
- Cancellazione: in SQL si usa l'istruzione DELETE
  - DELETE FROM Corsi WHERE Docente=9000
- Aggiornamento: in SQL si usa l'istruzione UPDATE
  - UPDATE Docenti SET Interno=V WHERE Matr=8021
- N.B.: ESISTONO SINTASSI PIU' ARTICOLATE!

QBE: Query By Example

- Modalità di interazione grafica con il DBMS
  - Permette l'esecuzione delle stesse operazioni viste finora ma specificandole tramite la scelta da menu degli elementi che compariranno nelle interrogazioni
  - Nasconde l'SQL sottostante all'utente inesperto
- Si può usare sia in luogo del DDL che del DML
  - Esempio: uso di Microsoft Access
- Attenzione!
  - Può non consentire l'esecuzione di tutte le possibili query
  - Poichè toglie controllo all'utente, può dare risultati non previsti dovuti a ipotesi errate dell'utente sul comportamento

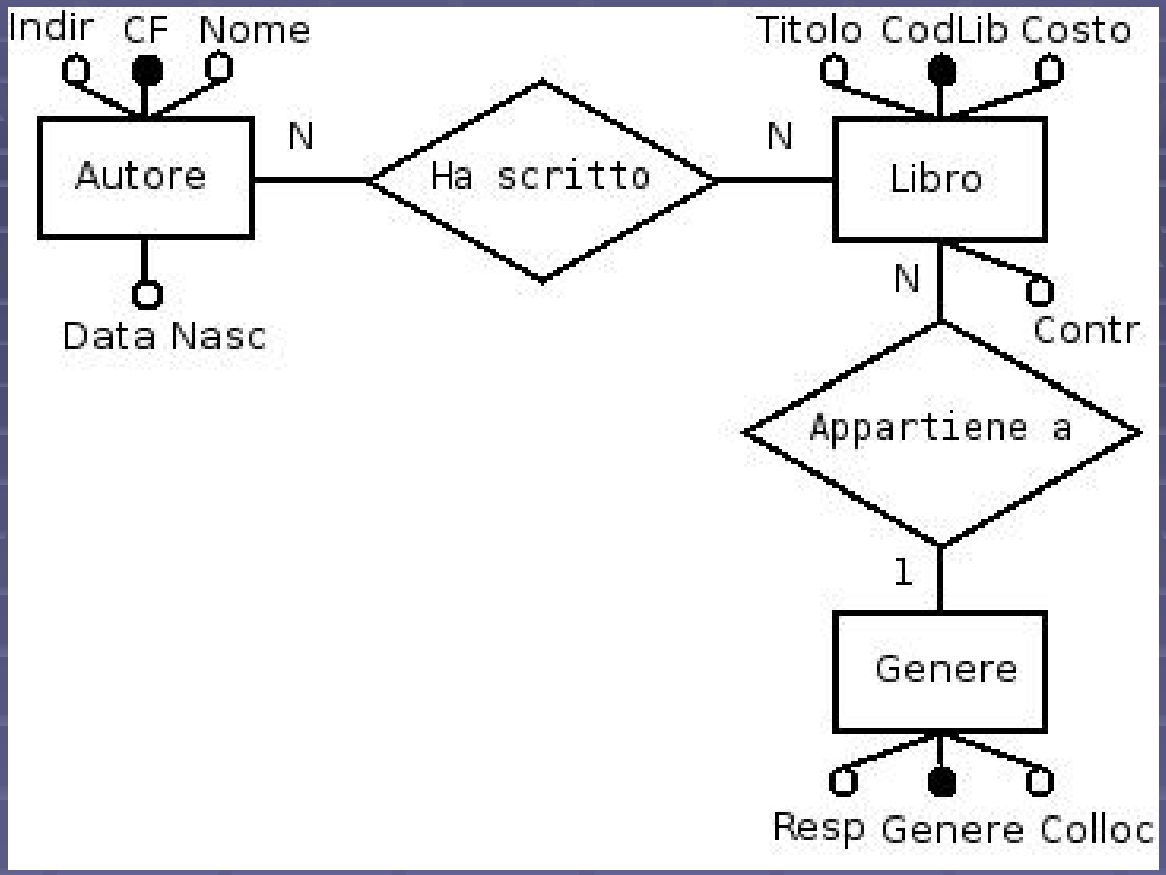
Progettazione

- Per progettare una base di dati è necessario analizzare il problema che essa deve risolvere
- L'analisi del dominio procede per raffinamenti incrementali di modelli concettuali, che servono a fissare gli aspetti rilevanti del problema
  - il modello concettuale è una descrizione essenziale e semiformale del problema (in generale grafica)
- Il modello concettuale più usato per i DB è il modello *Entità-Relazione*
  - da non confondere “relazione” in questa accezione con il concetto di relazione del modello relazionale
- Dal modello concettuale si trae poi il modello logico

Modello E-R

- Modella la realtà con *entità* che rappresentano gli elementi significativi e autonomi del problema e *relazioni* tra di esse che ne modellano i rapporti
- Una entità rappresenta l'astrazione di un oggetto concreto o astratto della realtà da rappresentare, dotata di dignità autonoma (è una classe di oggetti)
  - Esempio dell'editoria: Libro, Autore, Genere
- Una relazione lega tra loro due o più entità correlate
  - Esempio dell'editoria: Pubblicazione tra Autore e Libro, Appartenenza tra Libro e Genere
- Relazioni e entità posseggono attributi
  - analoghi agli attributi del modello relazionale

Esempio

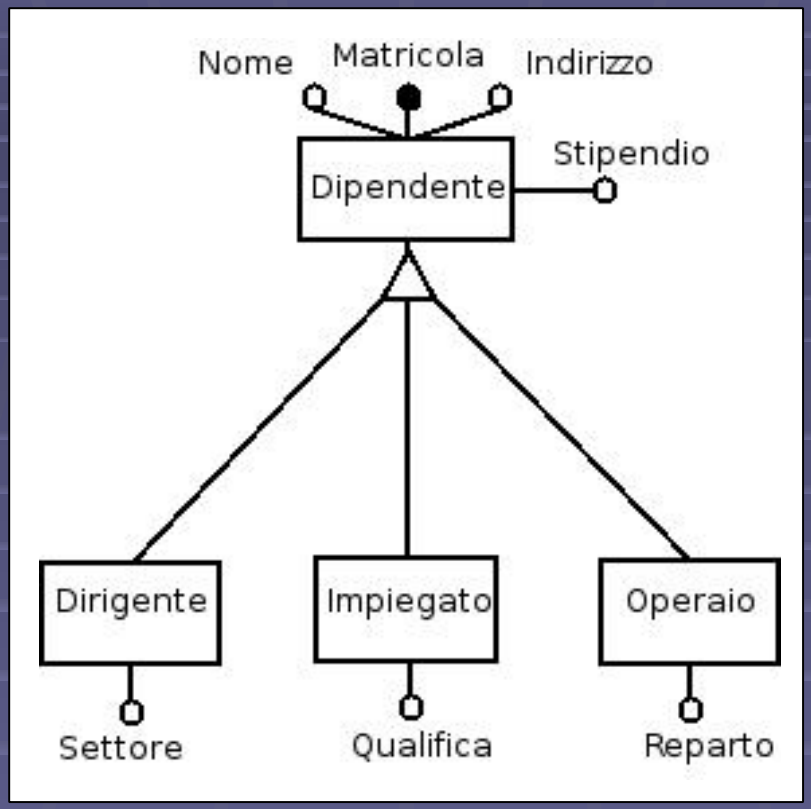


- La relazione N a N “Ha scritto” diventerà nel modello logico “Autori-Libri” e relativi vincoli 1 a N mentre “Appartiene a” diventerà un vincolo 1 a N

Generalizzazione

- La *generalizzazione* o *gerarchia IS A* è un costrutto del modello E-R che permette di ragionare per astrazioni durante la modellazione
  - Lega una *entità padre* ad una o più *entità figlie*
  - Le entità figlie hanno tutti gli attributi e le relazioni del padre più le proprie
  - Una entità figlia rappresenta un sottoinsieme degli elementi del padre con particolari caratteristiche
- Una gerarchia può essere:
  - Completa: non esistono nella realtà elementi concreti del padre
  - Disgiuntiva: i figli sono nella realtà sottoinsiemi disgiunti

Esempio



- Dirigenti, impiegati e operai sono dipendenti (g. disgiunta): in realtà non esistono dipendenti che non ricadano in uno dei tre casi (g. completa)

Normalizzazione

- Procedimento che permette di verificare se lo schema del DB rispetta le caratteristiche necessarie viste
- Si parla di *forme normali*, che assicurano che:
  - ogni relazione rappresenti un concetto diverso;
  - non esistano ridondanze;
  - la struttura sia semplice;
  - aggiornamenti, inserimenti e canzellazioni non causino *anomalie*;
  - gli attributi in una relazione dipendano effettivamente solo dalla chiave della stessa
- N.B: Le forme normali sono formalismi matematici

Ciclo di vita di un DB

- CdV: le vicissitudini del DB dalla nascita alla morte:
  - Progetto dello schema concettuale del DB
  - Generazione dello schema logico del DB (relazioni, attributi, vincoli, chiavi, vincoli di chiave esterna ed eventuali indici)
  - Caricamento iniziale dei dati
  - Messa in sicurezza del DB (protezione dei dati)
  - Elaborazione ordinaria e transazionale
  - Interrogazioni ed elaborazioni occasionali
  - Amministrazione del DB
  - Manutenzione e riorganizzazione
  - Backup