Informatica di Base[1] — Linea 1

Jianyi Lin

Dipp. di Matematica e Scienze dell'Informazione
Università degli Studi di Milano, Italia
jianyi.lin@unimi.it

a.a. 2011/12

Lezione XVI: Basi di dati (cont.)

Database relazionali

- I dati sono conservati in forma tabellare: ogni riga rappresenta una delle istanze (record) di un'entità da memorizzare, le colonne sono gli attributi memorizzati;
- Un insieme di attributi in grado di identificare univocamente un record si dice chiave.
- I dati possono essere identificati utilizzando tre semplici operazioni algebriche
  1. selezione
  2. proiezione
  3. prodotto cartesiano
- A queste generalmente si aggiunge il join, una selezione di un prodotto cartesiano in cui si prendono in considerazione solo i record correlati da una qualche chiave comune.

Progettazione di DB

Quando si progetta un database relazionale, occorre definire le entità da memorizzare e quali attributi conservare.
Un obiettivo importante è quello di evitare ridondanze nei dati:

- le stesse informazioni vengono memorizzate piú volte: spreco di spazio di memorizzazione
- ogni volta che un dato viene cambiato occorre aggiornare tutte le copie: incoerenza dei dati

Ridondanza

L'Università mantiene una tabella degli studenti iscritti:
`(matricola, nome, cognome, indirizzo, corso)`
La biblioteca tiene un elenco degli studenti e dei libri in prestito
`(matricola, nome, cognome, indirizzo, tessera)`
`(matricola, libro)`
Che succede se uno studente cambia indirizzo? Occorre assicurarsi che venga cambiato nelle due tabelle!
Invece:
`(matricola, corso)`
`(matricola, nome, cognome, indirizzo)`
`(matricola, tessera)`
`(matricola, libro)`
A questo punto il join diventa molto utile!

SQL

I DB vengono interrogati (query) per conoscere i dati che rispettano determinati criteri.
SQL (Structured Query Language) è un linguaggio formale per l'interrogazione di DB relazionali, molto diffuso (dagli anni '70 del '900).

SELECT attributi **FROM** prodotto cartesiano di tabelle **WHERE** condizione;

In pratica con **SELECT** si fa la proiezione (sic!) e con **WHERE** la selezione **FROM** un insieme ottenuto come prodotto cartesiano di tabelle.

SELECT nome, cognome **FROM** studente **WHERE** matricola $>$ 50000;

SQL esempi

Tabella Docenti: (identificativo, nome, cognome)
Tabella Esami: (nomeEsame, docente)

```
1 SELECT nome, cognome, nomeEsame
2 FROM Docenti, Esami
3 WHERE Docenti.identificativo = Esami.docente;
```

Esempi

Tabella World: (name, region, area, population, gdp)
Esempio: (“Afghanistan”, “South Asia”, 652225, 26000000)

```
1  SELECT population FROM World
2    WHERE name = 'France';
3
4  SELECT name, population/area FROM World
5    WHERE area > 5000000;
6
7  SELECT name , region
8    FROM World
9    WHERE population < 2000000
10     AND gdp > 5000000000;
11
12 SELECT name, population FROM World
13   WHERE name = 'Ireland' OR name = 'Iceland' OR name = 'Denmark');
14
15 SELECT name FROM World
16   WHERE name LIKE 'D%';
```

Esempi

Tabella PingPongWinners: (games, color, who, country)
Esempio: (1988, "gold", "Yoo Nam-Kyu", "KOR")
Tabella Country: (id, name)
Esempio: ("ALG", "Algeria")

```
1 SELECT who, Country.name
2   FROM PingPongWinners JOIN Country
3           ON (PingPongWinners.country=Country.id)
4  WHERE games = 2000;
5
6 SELECT who, Country.name
7   FROM PingPongWinners, Country
8  WHERE PingPongWinners.country=Country.id AND games = 2000;
```