Corso di Informatica Generale 1 – IN1

Linguaggio SQL

Marco Liverani
(liverani@mat.uniroma3.it)

Sommario

- Prima parte: *le basi dati relazionali*
  - Basi di dati: scopi, caratteristiche, ambiti applicativi
  - Il modello relazionale dei dati
  - Le prime tre forme normali, forma normale di Boyce e Codd
- Seconda parte: *il linguaggio SQL*
  - Linguaggio SQL: scopi e caratteristiche generali, DML e DDL
  - Data Manipulation Language: le istruzioni *insert*, *select*, *update*, *delete*
  - Operazioni di gruppo

Prima parte

Basi dati e cenni sulla teoria relazionale dei dati

Archivi informatici (digitali)

- L'archiviazione dei dati è una delle applicazioni più diffuse ed importanti dei calcolatori
- Tra gli obiettivi dell'archiviazione informatica dei dati evidenziamo i seguenti:
  - Garantire una elevata capacità di archiviazione
  - Garantire la possibilità di selezionare in modo efficiente le informazioni desiderate
  - Garantire l'integrità fisica e logica delle informazioni archiviate
  - Garantire la possibilità di utilizzare le informazioni archiviate attraverso diversi strumenti informatici
  - Garantire la protezione delle informazioni archiviate, impedendo la lettura o l'alterazione delle informazioni da parte di persone non autorizzate a farlo

Archiviazione su file

- Ogni programma applicativo può archiviare su file i dati di propria competenza
- Questo tipo di archiviazione tuttavia non tiene conto di alcuni aspetti importanti:
  - Il formato con cui sono memorizzate le informazioni su file è arbitrario, non standard
  - Altre applicazioni software possono accedere ai dati solo se il formato dei dati ed il loro significato vengono resi noti da chi ha progettato l'applicazione di archiviazione
  - Devono essere gestite situazioni di accesso concorrente
  - Chi sviluppa il software di archiviazione deve farsi carico anche degli aspetti "fisici" (efficienza nella scrittura e nella selezione delle informazioni, protezione delle informazioni riservate, modalità di accesso ai file, ecc.)

Database Management System

- Per far fronte a questi problemi, a partire dagli anni '70 sono state sviluppate applicazioni dedicate all'archiviazione di informazioni: DBMS.
- Caratteristiche di una base dati gestita attraverso un DBMS:
  - Può avere una grande dimensione (anche superiore a quella di un singolo disco del computer)
  - Può essere condivisa tra più applicazioni
  - È persistente
  - È affidabile (il DBMS fornisce strumenti per mantenere integro e gestire dei *backup* della base dati)
  - È sicuro e garantisce la riservatezza (consente il controllo degli accessi, solo gli utenti autorizzati possono accedere ai dati)
  - È efficiente: le operazioni elementari sull'archivio vengono eseguite molto rapidamente

Modello di database

- Ogni DBMS utilizza un modello astratto attraverso cui rappresentare le informazioni (e trasformare i singoli dati in vere e proprie informazioni)
- Negli anni sono stati proposti diversi modelli, tra i quali:
  - Gerarchico, ad albero
  - Reticolare
  - Relazionale
  - Relazionale ad oggetti
- Il modello che ci interessa è il modello relazionale, progettato inizialmente da IBM (anni '70) ed oggi lo *standard di fatto* per la maggioranza delle basi di dati e degli strumenti DBMS (si parla di RDBMS = *relational database management system*)

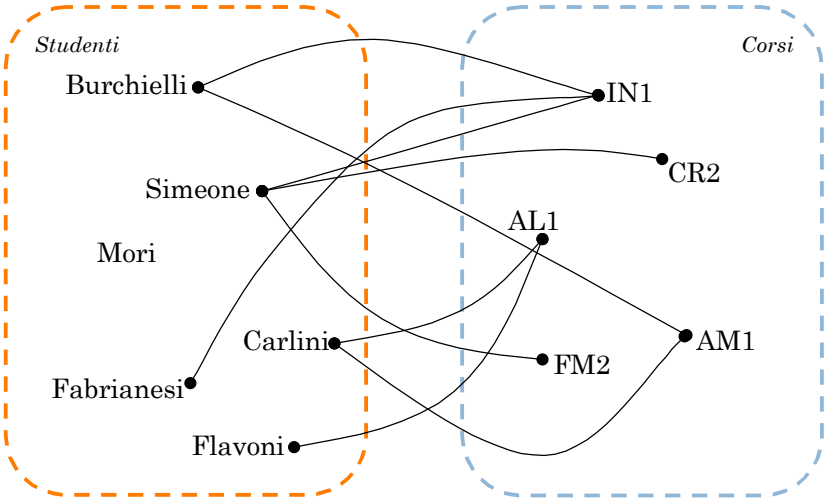
Dicembre 2004 M. Liverani - Linguaggio SQL 7

Modello relazionale

- È basato sul concetto di relazione (nel senso algebrico del termine) e di tabella (concetto intuitivo)
- Data una collezione di insiemi A_1, A_2,..., A_n, il prodotto cartesiano $A_1 \times A_2 \times \ldots \times A_n$ è l'insieme delle n-ple $\{(a_1, a_2, \ldots, a_n)$ tali che $a_i \in A_i \ \forall \ i=1, \ldots, n\}$
- Una relazione su A_1, A_2, ..., A_n è un sottoinsieme del prodotto cartesiano $A_1 \times A_2 \times \ldots \times A_n$

Dicembre 2004 M. Liverani - Linguaggio SQL 8

Relazioni tra insiemi



Dicembre 2004 M. Liverani - Linguaggio SQL 9

Relazioni e prodotto cartesiano

Relazione

Studente	Corso
Burchielli	IN1
Burchielli	CR2
Burchielli	AL1
Burchielli	AM1
Burchielli	FM2
Simeone	IN1
Simeone	CR2
Simeone	AL1
…	…

Prodotto cartesiano

Dicembre 2004 M. Liverani - Linguaggio SQL 10

Domini e attributi

- Nell'ambito della teoria relazionale dei dati è utile poter considerare le n-ple di una relazione come sequenze non ordinate (a differenza di quanto avviene nella teoria degli insiemi)
- Quindi ad esempio $(a_1, a_2, a_3) = (a_3, a_1, a_2)$
- In questo caso, per distinguere i dati presenti in una n-pla, è utile assegnare dei nomi ai valori delle n-ple; tali nomi vengono detti attributi
- L'insieme A_i (finito o infinito) dei valori che un certo attributo può assumere è il dominio dell'attributo stesso

Relazioni e tabelle

- È possibile rappresentare una relazione mediante una tabella:

Relazione X		
Attributo A	Attributo B	Attributo C
A_1	B_7	C_4
A_9	B_7	C_1

- Il numero di colonne (il numero di attributi) è il grado della relazione
- Il numero di righe è la cardinalità della relazione
- La relazione dell'esempio ha cardinalità 2 e grado 3

Chiavi primarie

- Consideriamo la seguente relazione rappresentata mediante una tabella:

Esami				
Studente	***Matricola***	***Corso***	Docente	Voto
Bianchi	***102030***	***IN1***	Liverani	24
Verdi	***213243***	***AL1***	Fontana	27
Rossi	***376114***	***IN1***	Liverani	25
Bianchi	***102030***	***AL1***	Fontana	22

- Una chiave primaria (*primary key*) è un campo (attributo) o un insieme di campi della tabella che permettono di individuare univocamente un record (riga della tabella)
- Nell'esempio la chiave primaria è la coppia "*Matricola*" + "*Corso*"

Dicembre 2004 M. Liverani - Linguaggio SQL 13

Chiavi esterne e correlazioni fra tabelle

- Per ridurre la ridondanza dei dati (evitare ripetizioni inutili) è possibile suddividere le informazioni su più tabelle:

Studente	
Matricola	Nome
102030	Bianchi
213243	Verdi
376114	Rossi

Corso	
Nome	Docente
AL1	Fontana
IN1	Liverani

Esame		
Corso	***Matricola***	Voto
IN1	***102030***	24
AL1	***213243***	27
IN1	***376114***	25
AL1	***102030***	22

- Le chiavi esterne (*foreign key*) sono attributi di una tabella X che consentono di correlare il record con un record di una tabella Y attraverso la ripetizione del valore della chiave primaria della tabella Y

Dicembre 2004 M. Liverani - Linguaggio SQL 14

Diagramma Entità/Relazioni

- In questo modo è possibile costruire una rappresentazione dei dati attraverso un diagramma che rappresenta le relazioni esistenti fra le varie entità
- Le relazioni possono essere di tre tipi:
  - "1:1": ad ogni record dell'entità A corrisponde uno ed un solo record dell'entità B
  - "1:M": ad ogni record dell'entità A corrispondono diversi record dell'entità B
  - "M:M": ad ogni record di A corrispondono molti record di B e viceversa.

Dicembre 2004 M. Liverani - Linguaggio SQL 15

Ottimizzazione di una base dati

- La possibilità di definire correlazioni tra le entità del database ci permette di ottimizzarne la struttura
- L'operazione di ottimizzazione consiste nella suddivisione dei dati su tabelle/entità distinte, in modo da ridurre la *ridondanza*; questa attività si chiama normalizzazione della base dati
- Esempio di base dati **non** normalizzata

Esami							
Nome	Matricola	AL1	Prof_AL1	IN1	Prof_IN1	GE1	Prof_GE1
Rossi	203040	24	Fontana	22	Liverani	-	Lopez
Verdi	251497	-	Fontana	-	Liverani	27	Lopez
Bianchi	337782	29	Fontana	30	Liverani	26	Lopez

Dicembre 2004 M. Liverani - Linguaggio SQL 16

Finalità della normalizzazione

- Perché normalizzare la base dati? Per risolvere i seguenti problemi:
  - ridondanza: inutile ripetizione di uno stesso dato
  - anomalia di aggiornamento: per mantenere integre e coerenti le informazioni è necessario modificare più volte uno stesso dato
  - anomalia di cancellazione: eliminando una informazione dal DB se ne perdono anche altre
  - anomalia di inserimento: non è possibile inserire informazioni incomplete, anche quando sarebbe necessario farlo

Esami							
Nome	***Matricola***	AL1	Prof_AL1	IN1	Prof_IN1	GE1	Prof_GE1
Rossi	***203040***	24	Fontana	22	Liverani	-	Lopez
Verdi	***251497***	-	Fontana	-	Liverani	27	Lopez
Bianchi	***337782***	29	Fontana	30	Liverani	26	Lopez

Dicembre 2004 M. Liverani - Linguaggio SQL 17

Forme normali 1

- Esistono delle regole che devono essere rispettate dalla base dati affinché questa sia correttamente normalizzata; queste regole sono note come forme normali
- Prima forma normale

  in una tabella non possono esistere colonne definite per contenere una molteplicità di valori

NO!

Esami				
Matr	Esami sostenuti			
	AL1	IN1	GE1	TIB
102030	25	27	22	30
123987	28	21	29	20
874329	25	26	26	24

OK!

Esame		
Matr	***Corso***	Voto
102030	***AL1***	25
123987	***GE1***	29
102030	***IN1***	27
874329	***TIB***	24

Dicembre 2004 M. Liverani - Linguaggio SQL 18

Forme normali 2

- Seconda forma normale

  in una tabella in cui la chiave primaria è composta da più attributi tutte le colonne devono dipendere dalla chiave primaria

NO!

Esame			
Matr	**Corso**	Voto	Docente
102030	AL1	25	Fontana
123987	GE1	29	Lopez
102030	IN1	27	Liverani
874329	AL1	24	Fontana

OK!

Esame		
Matr	**Corso**	Voto
102030	AL1	25
123987	GE1	29
102030	IN1	27
874329	TIB	

Corso	
Nome	Docente
AL1	Fontana
GE1	Lopez
IN1	Liverani

Dicembre 2004 M. Liverani - Linguaggio SQL 19

Forme normali 3

- Terza forma normale

  Non esistono dipendenze tra colonne di una tabella se non basate sulla chiave primaria

NO!

Esame			
Matr	**Corso**	Voto	Crediti
102030	AL1	25	6
123987	GE1	29	6
102030	IN1	27	9
874329	TIB	24	3

Corso	
Nome	Docente
AL1	Fontana
GE1	Lopez
IN1	Liverani

OK!

Esame		
Matr	**Corso**	Voto
102030	AL1	25
123987	GE1	29
102030	IN1	27
874329	TIB	24

Corso		
Nome	Docente	Crediti
AL1	Fontana	6
GE1	Lopez	6
IN1	Liverani	9

Dicembre 2004 M. Liverani - Linguaggio SQL 20

Forme normali 4

- Forma normale di Boyce e Codd

Una tabella è in forma normale se per ogni dipendenza funzionale A→B definita su di essa, A contiene una chiave della tabella.

Esame					
Matr	***Corso***	Voto	Docente	Crediti	Studente
102030	***AL1***	25	Fontana	6	Rossi
123987	***GE1***	29	Lopez	6	Bianchi
102030	***IN1***	27	Liverani	9	Rossi
874329	***AL1***	24	Fontana	3	Verdi

NO!

Dipendenze funzionali:

A→B: il valore di A determina il valore di B

Corso → Docente

Corso → Crediti

Matricola, Corso → Voto

Matricola → Studente

Dicembre 2004 M. Liverani - Linguaggio SQL 21

Forme normali 5

- Forma normale di Boyce e Codd

Una tabella è in forma normale se per ogni dipendenza funzionale A→B definita su di essa, A contiene una chiave della tabella.

Esame		
Matr	***Corso***	Voto
102030	***AL1***	25
123987	***GE1***	29
102030	***IN1***	27
874329	***TIB***	24

Corso		
Nome	Docente	Crediti
AL1	Fontana	6
GE1	Lopez	6
IN1	Liverani	9

Studente	
Matricola	Nome
102030	Bianchi
213243	Verdi
376114	Rossi

OK!

Dipendenze funzionali:

A→B: il valore di A determina il valore di B

Corso → Docente

Corso → Crediti

Matricola, Corso → Voto

Matricola → Studente

M. Liverani - Linguaggio SQL 22

Seconda parte

Linguaggio SQL per l'interrogazione di basi dati relazionali

Linguaggio SQL: caratteristiche

- Per operare su una base dati relazionale è stato progettato un linguaggio standard: SQL (*Structured Query Language*)
- È un linguaggio di interrogazione e manipolazione della base dati e delle informazioni in essa contenute
- Non è un linguaggio imperativo/procedurale, è un linguaggio dichiarativo
- È costituito da tre insiemi di istruzioni:
  - DDL (*Data Definition Language*): per definire la struttura della base dati
  - DCL (*Data Control Language*): per gestire i criteri di protezione e di accesso ai dati
  - DML (*Data Manipulation Language*): per operare sui dati

Data Definition Language

- È il set di istruzioni di SQL per la definizione della struttura della base dati.
- Sono tre le istruzioni principali del DDL:
  - **`create`**: per la creazione di database, tabelle, indici, viste, ecc.
  - **`alter`**: per la modifica della struttura di una tabella o di altri oggetti interni ad una base dati
  - **`drop`**: per l'eliminazione di una tabelle, di un intero database o di altri oggetti.
- Creazione di un database:

```
create database studenti
```

- Cancellazione di un database:

```
drop database studenti
```

Dicembre 2004 M. Liverani - Linguaggio SQL 25

Dominio di definizione degli attributi

- Definire una tabella significa definirne gli attributi e il dominio degli stessi attributi
- I domini sono quelli tipici di ogni linguaggio di programmazione:
  - numeri interi (int, integer)
  - numeri floating point (float, real, double)
  - stringhe di caratteri (char, varchar)
  - date (date, time, timestamp)
- Altri tipi possono essere definiti sulla base di specifiche tipiche di un determinato RDBMS
- Ogni attributo può essere anche caratterizzato da un valore di default e una serie di vincoli (es. "**`not null`**" per i campi obbligatori di una tabella)
- Tra i vincoli vi è la possibilità di aggiungere un attributo alla chiave primaria della tabella

Dicembre 2004 M. Liverani - Linguaggio SQL 26

Creazione di una tabella

```
create table corso (
    sigla char(5) not null primary key,
    nome char(20) not null,
    docente char(30),
    crediti integer default 6
)
```

```
create table esame (
    s_corso char(5) not null references
         corso.sigla,
    matr_studente char(10) not null
         references studente.matricola,
    voto integer not null,
    primary key (s_corso, matr_studente)
)
```


Data Control Language

- È il set di istruzioni di SQL per la definizione dei permessi di accesso sui database e sulle tabelle e per la gestione degli account utente.
- Sono due le istruzioni principali del DCL:
  - `grant`: per assegnare un determinato permesso ad un utente
  - `revoke`: revocare un determinato permesso ad un utente
- Concessione di permessi:

  `grant` *privilegi* `on` *risorsa* `to` *utenti* `with grant option`
- Esempio:

```
grant select on studenti to liverani
```

Data Manipulation Language

- È il set di istruzioni di SQL per la gestione delle informazioni presenti nella base dati
- Sono quattro le istruzioni principali del DML:
  - **insert**: per inserire dati in una tabella
  - **select**: per selezionare in base a determinati criteri o condizioni i dati presenti in una o più tabelle
  - **update**: per modificare i dati di una tabella sulla base di un determinato criterio di selezione
  - **delete**: per eliminare i record di una tabella corrispondenti ad una determinata condizione

Dicembre 2004 M. Liverani - Linguaggio SQL 29

Inserimento dati

- Sintassi dell'istruzione **insert**:

  insert into *tabella* **(***campo*$_1$**, …,** *campo*$_k$**) values (**valore$_1$**, …,** *valore*$_k$**)**

- Esempio:

```
insert into corso (sigla,
crediti, nome, docente) values
('IN1', 9, 'Informatica 1',
'Marco Liverani')
```

Dicembre 2004 M. Liverani - Linguaggio SQL 30

Selezione di record

- Sintassi dell'istruzione **select**:

```
select (tabella_1.campo_1, ..., tabella_k.campo_k)
from tabella_1, ..., tabella_k
where (condizione)
order by tabella_i.campo_i, ..., tabella_j.campo_j
```

- Esempio:

```
select (corso.sigla, corso.nome,
corso.docente) from corso where
crediti>=6 order by corso.docente,
corso.sigla
```

- Anche il risultato di una **select** è una relazione: una **select** su una tabella produce una tabella

Dicembre 2004 M. Liverani - Linguaggio SQL 31

Join

- È possibile eseguire operazioni di selezione che coinvolgano più tabelle grazie alle correlazioni stabilite tra queste attraverso le chiavi primarie e le chiavi esterne
- Questa operazione si chiama *join*
- Esempio:

```
select studente.nome, studente.cognome,
corso.nome, esame.voto
from studente, corso, esame
where studente.matricola=esame.matr_studente
and esame.s_corso=corso.sigla
order by esame.voto
```

Dicembre 2004 M. Liverani - Linguaggio SQL 32

Join e prodotto cartesiano di tabelle 1

- Ogni operazione di *join* fra più tabelle consiste innanzi tutto in un prodotto cartesiano tra le tabelle coinvolte nella **`select`**
- Dalla relazione (tabella) ottenuta come prodotto cartesiano si selezionano solo le righe che corrispondono con la "*where condition*"
- Una operazione di *join* può essere quindi molto onerosa per la macchina: la base dati deve essere progettata in modo da minimizzare il numero di *join* necessarie per ottenere le informazioni di interesse

Dicembre 2004 M. Liverani - Linguaggio SQL 33

Join e prodotto cartesiano di tabelle 2

Esame		
Matr	Corso	Voto
102030	AL1	25
123987	GE1	29
102030	IN1	27

Studente	
Matricola	Nome
102030	Bianchi
123987	Verdi
376114	Rossi

```
select esame.corso, esame.voto, studente.nome from
esame, studente where esame.matr=studente.matricola
```

Join: esame × studente				
Matr	Corso	Voto	Matricola	Nome
102030	AL1	25	102030	Bianchi
102030	AL1	25	123987	Verdi
102030	AL1	25	376114	Rossi
123987	GE1	29	102030	Bianchi
123987	GE1	29	123987	Verdi

Dicembre 2004 M. Liverani - Linguaggio SQL 34

Aggiornamento

- Sintassi dell'istruzione **update**:

  update *tabella*
  set $campo_1$=$valore_1$, …, $campo_k$=$valore_k$
  where (*condizione*)

- Esempio:

```
update studente set nome='Mori'
where matricola=102030
```

- Attenzione: una istruzione **update** priva di una condizione esegue l'aggiornamento su tutte le righe (record) della tabella

Cancellazione

- Sintassi dell'istruzione **delete**:

  delete from *tabella*
  where (*condizione*)

- Esempio:

```
delete from studente where nome
like 'Mo%' or matricola=213243
```

- Attenzione: una istruzione **delete** priva di una condizione esegue la cancellazione di tutti i record (tutte le righe) della tabella

Operazioni di gruppo

- È possibile ottenere attraverso una **`select`** anche valori non presenti nei campi della tabella, ma calcolati raggruppando più record sulla base di un criterio di aggregazione
- Esempio:

```
select studente.nome, average(esame.voto)
from studente, esame
where studente.matricola=esame.matr
group by studente.nome
```

- Tipiche funzioni di gruppo sono
  - **`sum`**: somma i valori numerici dei campi
  - **`average`**: calcola la media aritmetica dei valori numerici dei campi

Bibliografia

- Atzeni, Ceri, Paraboschi, Torlone, *Basi di dati – Concetti, linguaggi e architetture*, McGraw-Hill, 1996
- Guidi, Dorbolò, *Guida a SQL*, McGraw-Hill, 1996
- Codd, *A relational model for large shared data banks*, Communication of the ACM, 13(6), June 1986