Corso sul linguaggio SQL

Modulo L2B (SQL)

4.3 - Congiunzione

Prerequisiti

- Creazione e gestione tabelle
- Uso ambiente SQL
- Congiunzione in algebra relazionale

Introduzione

In questa Unità vediamo i comandi SQL con i quali è possibile eseguire *query* complesse che comportano l'operazione di congiunzione.

Informazioni generali

N.B. – A solo scopo didattico:

- i caratteri MAIUSCOLI indicano parole chiave del linguaggio;
- i caratteri *corsivi* indicano elementi che dovranno essere specificati dal programmatore;
- le parentesi quadre indicano opzione
- la barra verticale "|" indica alternativa.

Congiunzione

Considerando le tabelle indicate, vediamo come il comando **SELECT** possa servire anche per realizzare la **congiunzione**.

PATERNITA

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo

MATERNITA

Madre	Figlio
Luisa	Maria
Luisa	Luigi
Anna	Olga
Anna	Filippo
Maria	Andrea
Maria	Aldo

Congiunzione
- Sintassi

La **congiunzione** (*join*) è realizzata dalla **SELECT** con la sintassi seguente:

SELECT *lista_attributi* **FROM** *lista_tab*
[**WHERE** *condizione*];

Si noti la presenza di più tabelle in ingresso

dove

- *lista_attributi* indica l'elenco degli attributi da visualizzare
- *condizione* indica la condizione di selezione (opzionale)
- *lista_tab* indica le tabelle coinvolte

Congiunzione
- Equi-join

ATTIVITA': scrivere il comando SQL per elencare *padre*, *madre* e *figlio* dalle tabelle PATERNITA' e MATERNITA'.

La sintassi col "." serve ad indicare un campo di una tabella in caso di ambiguità

Padre	Madre	Figlio
Luigi	Anna	Olga
Luigi	Anna	Filippo
Franco	Maria	Andrea
Franco	Maria	Aldo

Il comando è il seguente:

SELECT padre, madre, paternita.figlio **FROM** maternita, paternita
WHERE paternita.figlio = maternita.figlio;

che realizza una **congiunzione naturale** (è anche un *equi-join*).

Congiunzione
- Join esplicito

In modo alternativo si può esprimere il **join esplicito** con la sintassi

SELECT *lista_attributi*
FROM *tab* { **JOIN** *tab* **ON** *condizione* }
[**WHERE** *condizione*];

Per cui l'elenco delle famiglie (come la precedente) è dato anche da:

SELECT padre, madre, paternita.figlio
FROM maternita **JOIN** paternita **ON**
paternita.figlio = maternita.figlio;

Congiunzione
- Join esterno

Il **join esterno** (sinistro, destro o pieno) prevede la creazione di tuple con attributi di valore **NULL**.

Sintassi:

SELECT *lista_attributi*
FROM *tab* { **LEFT|RIGTH|FULL JOIN** *tab* **ON** *condizione* }
[**WHERE** *condizione*];

Congiunzione
- Join esterno

ATTIVITA': scrivere l'elenco delle famiglie complete, sulle tabelle PATERNITA' e MATERNITA', tenendo conto della eventuale mancanza della madre

Il comando è il seguente:

SELECT padre, madre, paternita.figlio
FROM paternita **LEFT JOIN**
maternita **ON**
paternita.figlio=maternita.figlio;

Padre	Madre	Figlio
Sergio	NULL	Franco
Luigi	Anna	Olga
Luigi	Anna	Filippo
Franco	Maria	Andrea
Franco	Maria	Aldo

Congiunzione
- Inner join

ATTIVITA': dati gli schemi:
Categorie(Codice, Nome)
Veicoli (Targa, Modello, IDCategoria, Cilindrata, Posti)
elencare i dati di tutti i veicoli la cui categoria ha nel campo *Nome* il valore "Autovettura" o "Camion"

Il comando è il seguente:

SELECT *
FROM Veicoli, Categorie
WHERE Categorie.Codice = Veicoli.IDCategoria **AND**
Categorie.Nome **IN** ("Autovettura","Camion");

Non è un *equi-join*, in quanto la condizione non è di uguaglianza.

Interrogazioni composte

Con il comando **SELECT** possiamo anche realizzare **interrogazioni composte** che possono essere scritte anche in modo **nidificato**. Consideriamo, ad esempio, le tabelle seguenti:

PATERNITA

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo

MATERNITA

Madre	Figlio
Luisa	Maria
Luisa	Luigi
Anna	Olga
Anna	Filippo
Maria	Andrea
Maria	Aldo

PERSONE

Nome	Età	Reddito
Andrea	27	21
Aldo	25	15
Maria	55	42
Anna	50	35
Filippo	26	30
Luigi	50	40
Franco	60	20
Olga	30	41
Sergio	85	35
Luisa	75	87

Interrogazioni composte

ATTIVITA': scrivere il comando SQL per visualizzare nome e reddito del padre di Franco

SELECT Nome, Reddito **FROM** Persone, Paternita
WHERE Nome = Padre **AND** Figlio = 'Franco';

ma anche come **interrogazione nidificata**:

SELECT Nome, Reddito **FROM** Persone
WHERE Nome = (**SELECT** Padre from Paternita **WHERE** figlio='Franco')

Interrogazioni composte

Sulle interrogazioni composte sono necessarie le seguenti osservazioni:

- la forma nidificata è talvolta più leggibile (richiede meno variabili)
- la forma piana e quella nidificata possono essere combinate
- le sottointerrogazioni non possono contenere operatori insiemistici ("l'unione si fa solo al livello esterno"); la limitazione non è significativa
- l'interrogazione interna viene eseguita una volta per ciascuna tupla dell'interrogazione esterna

SQL e algebra relazionale

In conclusione, l'operatore **SELECT** in SQL svolge le funzioni che in algebra relazionale sono tipiche di diverse operazioni.
In particolare:

- *prodotto cartesiano*: espresso dalla clausola **FROM**
- *selezione*: espressa dalla clausola **WHERE**
- *proiezione*: espressa dalla clausola **SELECT**

SQL e algebra relazionale

Ad esempio, date due relazioni R1(A1,A2) R2(A3,A4), l'interrogazione

SELECT R1.A1, R2.A4
FROM R1, R2
WHERE R1.A2 = R2.A3

corrisponde all'operazione relazionale

PROJ $_{A1,A4}$ (**SEL**$_{A2=A3}$ (R1 **JOIN** R2))

Esecuzione delle interrogazioni

- Le espressioni SQL sono dichiarative e quindi ne usiamo direttamente la semantica
- In pratica, i DBMS eseguono le operazioni in modo efficiente, ad esempio:
  - eseguono le selezioni in modo prioritario
  - se possibile, eseguono *join* e non prodotti cartesiani
- Dato che di solito il DBMS "ottimizza" l'esecuzione delle interrogazioni, non è strettamente necessario preoccuparsi della loro efficienza quando le si specificano; la chiarezza è un fattore determinante.

Argomenti

- Congiunzione
  - Sintassi
  - Equi-join
  - Join esplicito
  - Join esterno
  - Inner join
- Interrogazioni composte
- SQL e algebra relazionale
- Esecuzione delle interrogazioni

Altre fonti di informazione

- Atzeni, Ceri, Paraboschi, Torlone, Basi di dati - McGraw-Hill, 1996-2002
- A. Lorenzi-D.Rossi – Le basi di dati e il linguaggio SQL – ed. ATLAS