Opzioni aggiuntive per il comando Select

Per interrogare un database MySQL si usa il comando SELECT. MySQL accetta ed esegue correttamente i comandi del linguaggio SQL, scritti con la sintassi presentata nel Capitolo 6. È però possibile usare il comando SELECT con lievi differenze sintattiche e specifiche opzioni che permettono di scrivere interrogazioni espressive e molto potenti. Per mostrare alcune utili caratteristiche dei comandi SQL di MySQL si useranno le tabelle *Libri* e *Autori*:

Autori (IDAutore, Nome, Cognome, Nascita, EMail)
Libri (IDLibro, Titolo, Pagine, Edizione, Prezzo, Editore, *IDAutore*)

1. Quando si ha a che fare con tabelle come le precedenti, dove la chiave primaria della tabella *Autori* e la chiave esterna associata nella tabella *Libri* hanno il medesimo nome, si possono eseguire congiunzioni di tabelle usando, per la clausola FROM, la seguente sintassi con l'opzione **USING**:

```
FROM Tabella1 INNER JOIN Tabella2 USING (Campo_comune)
```

dove la parola INNER è opzionale.
Per esempio, per elencare i gialli scritti da *Agatha Christie* si può scrivere l'interrogazione:

```
SELECT Libri.*
FROM Libri JOIN Autori USING (IDAutore)
WHERE Nome = 'Agatha' AND Cognome = 'Christie';
```

2. Per elencare tutti gli autori con nome, cognome e il numero di loro libri che sono presenti nella tabella *Libri*, si può scrivere un'interrogazione dove nella clausola SELECT può comparire, oltre al campo usato in GROUP BY (*IDAutore* nell'esempio), anche i campi che sono identificati univocamente dal valore del campo usato per eseguire il raggruppamento, cioè i campi che sono da questo determinati. Nella clausola SELECT dell'esempio, dove oltre a *IDAutore* compaiono anche i campi *Nome* e *Cognome*, questo è possibile perché *IDAutore* è determinante per *Nome*, *Cognome*.

```
SELECT IDAutore, Nome, Cognome, Count(*)
FROM Libri JOIN Autori USING (IDAutore)
GROUP BY IDAutore;

+----------+----------+----------+----------+
| IDAutore | Nome     | Cognome  | count(*) |
+----------+----------+----------+----------+
| AGCH001  | Agatha   | Christie |        7 |
| ERDL001  | Erri     | De Luca  |        4 |
+----------+----------+----------+----------+
```

3. Nello scrivere interrogazioni con raggruppamenti si può usare l'opzione **WITH ROLLUP** nella clausola GROUP BY. Per effetto di questa opzione la funzione di aggregazione è applicata ad ogni raggruppamento e poi sulla totalità delle righe considerate. L'uso dell'opzione WITH ROLLUP nei raggruppamenti è molto utile nella preparazione di rapporti di sintesi per la direzione.

Raggruppando i libri per autore e contando il numero di libri in ogni raggruppamento con l'opzione WITH ROLLUP, si ottiene l'elenco mostrato di seguito al comando che lo genera. Come effetto del ROLLUP il numero di libri dell'intera tabella è inserito nell'ultima riga, evidenziata in colore, con il valore NULL nel campo *IDAutore*.

```
SELECT IDAutore, Count(*)
FROM Libri
GROUP BY IDAutore WITH ROLLUP;

+----------+----------+
| IDAutore | count(*) |
+----------+----------+
| AGCH001  |        7 |
| ERDL001  |        4 |
| NULL     |       11 |
+----------+----------+
```

4. Per controllare il valore dei dati inseriti in una tabella si può usare la clausola **WITH CHECK OPTION** in una vista logica modificabile. La clausola WITH CHECK OPTOIN serve per validare i dati inseriti, che sono controllati con la condizione scritta nella clausola WHERE. Si noti che con il comando:

```
INSERT INTO Libri VALUES
('M1348','Dieci piccoli indiani','Paperback','Granada',12.75,0,'AGCH001');
```

si inserisce nella tabella *Libri* la descrizione di un libro con 0 pagine. Per impedire che accadano situazioni di questo tipo, per esempio per impedire l'inserimento nella tabella *Libri* di record con *Pagine* < = *0* oppure con *Prezzo* < = *0*, si può creare la vista logica *LibriWithCheck*, definita di seguito, e costringere gli utenti a inserire i dati nella tabella usando la vista logica. Come si è visto nel Paragrafo 7, per attuare questa strategia bisogna usare un comando **GRANT** per permettere di usare *Libri* con i soli comandi SELECT e DELETE e concedere l'uso dei comandi INSERT ed UPDATE con la vista logica *LibriWithCheck*.

```
CREATE VIEW LibriWithCheck AS
SELECT IDLibro, Titolo, Pagine, Edizione, Prezzo, Editore, IDAutore
FROM Libri
WHERE Pagine > 0 AND Prezzo > 0
WITH CHECK OPTION;
```

MySQL impedisce che si possano inserire record che violano la condizione scritta nella clausola WHERE di una vista logica dove compare la clausola WITH CHECK OPTION.

```
mysql> INSERT INTO LibriWithCheck VALUES
    -> ('M1348','Dieci piccoli indiani','Paperback','Granada',12.75,
    -> 0,'AGCH001');
ERROR 1369 (HY000): CHECK OPTION failed 'biblioteca.libriwithcheck'
```