

Autore: Luciano Viviani

Classe: TERZA INFORMATICA SERALE (3IS)

Anno scolastico: 2003/2004

Scuola: Itis Euganeo

MANUALE ESSENZIALE MYSQL

Manuale

La dispensa vuole fornire agli studenti delle classi quinte del corso di informatica alcune informazioni essenziali sull'uso del database MySql.

In particolare viene trattato in modo abbastanza approfondito il modo con cui vengono gestite le protezioni degli accessi alle tabelle del database da utenti che si collegano da una rete Internet-Intranet. Non è trattato in modo approfondito il linguaggio SQL, per il quale si fa riferimento al libro di testo e ai numerosi manuali esistenti.



Introduzione

Questo testo ha lo scopo di descrivere il primo approccio al database MySQL. Non contiene quindi una descrizione completa del linguaggio ma le sue parti ritenute essenziali. Come prerequisito è richiesta una conoscenza del linguaggio SQL cui si fa riferimento in svariati punti. Il manuale completo può essere scaricato dal sito www.mysql.com/documentation/

Gli esempi non sono stati verificati, possono quindi contenere degli errori: il manuale è stato realizzato per puro scopo didattico, chi usa MySQL in applicazioni commerciali faccia riferimento alla documentazione ufficiale. La versione di riferimento è la 3.23.47.

In questo manuale faremo riferimento all'esecuzione di comandi SQL dal terminale a linea di comando fornito con MySQL, altri manuali essenziali faranno riferimento alle interfacce realizzabili in PHP e Java. MySQL è un server, di fatto si collega alla porta 3306 di default, tutte le applicazioni quindi useranno lo schema client-server. Nel manuale inoltre non sono riportati i comandi SQL, per questi si può fare riferimento ad un qualsiasi testo.

Connessione e disconnessione dal server

MySQL protegge le connessioni ai suoi database mediante uno schema di protezione abbastanza complesso, che stabilisce i diritti degli utenti in base all'host da cui si collegano, al nome utente e ad una password. Appena installato MySQL fornisce un accesso completo a due utenti: root e ad un utente anonimo senza alcuna password. Questo accesso è troppo libero per un utente inesperto, in particolare è possibile modificare tutti i diritti di accesso, cosa sconsigliata per chi non conosce ancora il meccanismo delle protezioni, spiegato alla fine di questo manuale. Per evitare problemi *non modificate mai le tabelle del database mysql*, almeno fino a che non avrete compreso il meccanismo delle protezioni. In ogni caso, per modificare i diritti di accesso è preferibile usare i comandi SQL GRANT e REVOKE

La sintassi per connettersi al database, dalla directory bin di mysql è la seguente (nell'ipotesi che MySQL sia installato sotto Windows nella directory C:\mysql):

```
C:\mysql\bin> mysql -h host -u user -p
Enter password: *****
```

Dato che inizialmente MySQL prevede un utente anonimo senza password funziona ugualmente il comando:

```
C:\mysql\bin> mysql
```

A questo punto compare un benvenuto con alcune informazioni

```
shell> mysql -h host -u user -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 3.23.47 nt
```

Type 'help' or \h for help.

```
mysql>
```

A questo punto siete pronti per eseguire i vostri comandi SQL, prima però è bene che create un vostro database, o che usiate il database test fornito da MySQL, nel primo caso eseguite i comandi:

```
mysql>create database prova;
mysql>use prova;
```

Nel secondo caso inserite solo il comando:

```
mysql>use test;
```

E avete buone probabilità di non causare guai. Finito il lavoro potete uscire dal programma con il comando QUIT. Da notare che i comandi SQL non sono case-sensitive, potete usare maiuscole o minuscole indifferentemente, questo non vale in generale per i nome delle colonne.

Inserimento di query

Anche se non ci sono ancora tabelle nel vostro database potete già eseguire dei comandi di interrogazione su funzioni o costanti predefinite di MySQL, per esempio il comando:

```
mysql> SELECT VERSION(), CURRENT_DATE;
```

fornirà un output del tipo:

```
+-----+-----+
| version() | CURRENT_DATE |
+-----+-----+
| 3.27.47-nt | 2003-11-16   |
+-----+-----+
1 row in set (0.00 sec)
```

mysql>

Questa query fornisce alcune informazioni sul funzionamento di MySQL:

- Un comando normalmente consiste in uno statement SQL seguito da punto e virgola;
- quando date un comando mysql lo manda al server per l'esecuzione e mostra il risultato;
- l'output è mostrato come una tabella, formata da righe e colonne. La prima riga contiene in genere i nomi delle colonne delle tabelle interrogate, in questo caso fornisce le espressioni valutate.
- MySQL fornisce inoltre il numero di righe del risultato e un valore (approssimato) del tempo impiegato nell'esecuzione.

La query che segue dimostra come MySQL può essere usato per compiere dei calcoli:

```
mysql> SELECT SIN(PI()/4), (4+1)*5;
+-----+-----+
| SIN(PI()/4) | (4+1)*5 |
+-----+-----+
| 0.707107 | 25 |
+-----+-----+
```

Creare e usare un database

In questa sezione sarà mostrato come:

- Creare un database
- Creare una tabella
- Inserire dati in una tabella
- Recuperare i dati dalla tabella

Innanzitutto usiamo la statement SHOW per vedere quanti database esistono nel server:

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| mysql    |
| test     |
| prova    |
+-----+
```

Ora potete selezionare il database prova con il seguente comando:

```
mysql> USE prova
Database changed
```

Creazione di una tabella

A questo punto il database dovrebbe essere vuoto, potete verificarlo con il comando SHOW TABLES:

```
mysql> SHOW TABLES;
Empty set (0.00 sec)
```

A questo punto dovete decidere quali tabelle creare e quali dati inserire in esse, usate il comando CREATE TABLE, nel nostro esempio inseriremo Cognome, Nome, Data di Nascita per creare una tabella che ci ricordi i compleanni

dei nostri amici:

```
mysql> CREATE TABLE compleanni (cognome VARCHAR(20), nome VARCHAR(20),
-> nascita DATE);
Query OK, 0 rows affected (0.02 sec)
```

Come si vede uno statement può essere scritto su più righe, il simbolo -> viene usato da mysql quando non avete terminato una riga con punto e virgola. A questo punto ripetiamo il comando:

```
mysql> SHOW TABLES;
+-----+
| Tables_in_prova |
+-----+
| compleanni      |
+-----+
```

Se si vuole vedere come è stata creata la tabella si usa il comando:

```
mysql> DESCRIBE compleanni;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| cognome | varchar(20) | YES  |     | NULL    |       |
| nome   | varchar(20) | YES  |     | NULL    |       |
| nascita | date         | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
```

Inserire dati nella tabella

Dopo aver creato la tabella bisogna inserirvi dei dati, si usa il comando insert.

```
mysql> INSERT INTO compleanni
-> VALUES ('Viviani', 'Luciano', '1948-03-29');
Query OK, 1 row affected (0.06 sec)
```

È da notare che i dati vanno inseriti come stringhe (delimitate da ‘’) e che la data deve essere inserita come anno-mese-giorno, non nel formato con cui siamo abituati.

Ritrovare i dati da una tabella

Per ritrovare i dati inseriti in una tabella si usa il comando SELECT nella forma generale:

```
SELECT nomi delle colonne
FROM quale tabella
WHERE condizioni da soddisfare
```

La clausola WHERE è opzionale, la forma più semplice di SELECT trova tutti i dati:

```
mysql> SELECT * FROM compleanni;
+-----+-----+-----+
| cognome | nome   | nascita |
+-----+-----+-----+
| Viviani | Luciano | 1948-03-29 |
+-----+-----+-----+
```

Questa forma di select è utile quando avete pochi dati nella tabella. Per il normale uso la clausola WHERE vi permette di selezionare solo i dati che vi interessano, specificando per esempio il nome o il cognome. Nello stesso modo invece di * possiamo elencare le colonne che compaiono nell’output. Esistono inoltre delle funzioni che permettono di estrarre dalla data solo l’anno il mese o il giorno. Per esempio se volessimo vedere solo chi compie gli anni il 29 marzo dovremmo usare una sintassi del tipo:

```
mysql> SELECT * FROM compleanni
-> WHERE (MONTH(nascita)=3 AND DAYOFMONTH(nascita)=29);
```

Gestione dei privilegi di accesso a MySQL

Le informazioni relative ai privilegi sono memorizzate nelle tabelle **user**, **db**, **host**, **tables_priv** e **columns_priv** nel database **mysql**, solo l'amministratore del sistema deve avere accesso a queste tabelle e deve utilizzarle con molta cautela dopo aver letto queste note.

Tipi di privilegio

Privilegio	Nome della colonna	Significato
select	Select_priv	Permette (Y) o vieta (N) l'operazione SELECT su una tabella
insert	Insert_priv	Permette o vieta l'operazione SQL INSERT su una tabella
update	Update_priv	Permette o vieta la modifica dei dati di una riga
delete	Delete_priv	Permette o vieta la cancellazione di una riga
index	Index_priv	Permette o vieta la creazione di un indice
alter	Alter_priv	Permette o vieta la modifica della struttura di una tabella
create	Create_priv	Permette la creazione di database, tabelle, indici
drop	Drop_priv	Permette l'eliminazione di tabelle o database
grant	Grant_priv	Permette di assegnare privilegi ad altri utenti
references	References_priv	Permette di creare riferimenti tra tabelle
reload	Reload_priv	Ricarica o rinfresca le tabelle
shutdown	Shutdown_priv	Può disattivare il servizio
process	Process_priv	Può visualizzare e terminare i processi in corso
file	File_priv	Permette la lettura o scrittura di file interni

Questa tabella concede privilegi su tutto il contenuto dei database nel server, compreso mysql stesso. I permessi a questo livello (valori Y nei campi di privilegio) dovrebbero essere concessi solo agli amministratori del sistema, un utente normale dovrebbe quindi contenere tutti N in questi campi. Una gestione corretta viene gestita dai comandi GRANT e REVOKE.

Come lavora il sistema di privilegi di MySQL

1. Il privilegio viene accordato in funzione del NomeUtente e del nome del computer da cui ci si collega, l'identificazione viene fatta con entrambi i valori
2. Nel caso in cui `utente@computer` abbia diritto di accesso ogni operazione sul database viene controllata con i diritti d'accesso per quell'operazione

Il server usa le tabelle **user**, **db**, **host** per controllare gli accessi, i campi delle tabelle sono i seguenti:

Nome tabella	user	db	host
Campi di scopo	Host	Host	Host
	User	Db	Db
	Password	User	
Campi di privilegi	Select_priv	Select_priv	Select_priv
	Insert_priv	Insert_priv	Insert_priv
	Update_priv	Update_priv	Update_priv
	Delete_priv	Delete_priv	Delete_priv
	Index_priv	Index_priv	Index_priv
	Alter_priv	Alter_priv	Alter_priv
	Create_priv	Create_priv	Create_priv
	Drop_priv	Drop_priv	Drop_priv
	Grant_priv	Grant_priv	Grant_priv
	References_priv		
	Reload_priv		
	Shutdown_priv		
	Process_priv		
File_priv			

Per controllare il diritto a compiere determinate operazioni vengono consultate, per gli utenti che comunque hanno l'accesso anche le tabelle `tables_priv` e `columns_priv`.

Tabelle	<code>tables_priv</code>	<code>columns_priv</code>
Campi di scopo	Host	Host
	Db	Db
	User	User
	Table_name	Table_name
		Column_name
Campi di privilegi	Table_priv	Column_priv
	Column_priv	
Altri campi	Timestamp	Timestamp
	Grantor	

Ciascuna tabella contiene campi di scopo e di privilegio, i campi di scopo determinano il contesto cui si applicano i privilegi per ciascun record della tabella. Per esempio un record nella tabella `user` con i valori:

Host: uno.sistemi.net **User:** luciano

sarà utilizzato per determinare i privilegi dell'utente luciano quando si collega dell'host uno.sistemi.net , ovviamente se la password di accesso è corretta.

In modo simile, nella tabella `db` un record con i valori:

Host: uno.sistemi.net **User:** luciano **Db:** anagrafe

determinerà i privilegi dell'utente luciano, che accede da uno.sistemi.net, limitatamente al database anagrafe, in modo analogo funzionano le tabelle `tables_priv` e `columns_priv`.

Tutti i confronti, tranne Host (compatibilità con Windows), tengono conto di maiuscole-minuscole

I campi di privilegio indicano quali operazioni possono essere compiute da un determinato utente, il server combina le informazioni nelle varie tabelle per creare una descrizione completa di ciascun utente, le regole saranno descritte in seguito.

I campi di scopo sono stringhe di caratteri, dichiarati come segue, e con valore di default la stringa vuota

Nome del campo	Tipo del campo
Host	CHAR(60)
User	CHAR(16)
Password	CHAR(16)
Db	CHAR(64) CHAR(60) per le tabelle <code>tables_priv</code> e <code>columns_priv</code>
Table_name	CHAR(60)
Column_name	CHAR(60)

Nelle tabelle `user`, `db` e `host` tutti i campi di privilegio sono dichiarati ENUM ('N', 'Y') con valore di default 'N'

Nelle tabelle `tables_priv` e `columns_priv` i campi di privilegio sono dichiarati come SET con i seguenti valori possibili:

Nome tabella	Nome campo	Elementi del SET
<code>tables_priv</code>	Table_priv	'Select', 'Insert', 'Update', 'Delete', 'Create', 'Drop', 'Grant', 'References', 'Index', 'Alter'
<code>tables_priv</code>	Column_priv	'Select', 'Insert', 'Update', 'References'
<code>columns_priv</code>	Column_priv	'Select', 'Insert', 'Update', 'References'

I tipi di dati ENUM e SET sono estensioni MySQL, il primo è un elenco di valori (massimo 65535) di cui uno solo può essere attribuito alla variabile, il secondo è un oggetto di tipo stringa che può assumere zero o più valori (massimo 64) che devono essere scelti dalla lista .

In breve, il server usa le tabelle dei privilegi di accesso nel seguente modo:

- I campi di scopo della tabella user determinano se accettare o rifiutare una connessione;
- se la connessione è accettata i privilegi indicati nel record della tabella relativo all'utente si applicano a tutte le basi dati del server, sono cioè privilegi globali. in particolare i privilegi relativi all'amministrazione sono solo globali e si trovano solo in questa tabella.
- Le tabelle db e host sono utilizzate insieme:
 - i campi di scopo della tabella db determinano quali utenti possono aver l'accesso, a quali basi dati e da quali host, i campi di privilegio determinano le operazioni ammesse.
 - La tabella host è usata come estensione della tabella db se si vuole che un utente, identificato dalla tabella db, possa accedere a un data base da più host. In questo caso il campo Host nella tabella db deve essere vuoto e per ogni host si deve creare un record nella tabella host.
 - Le tabelle tables_priv e columns_priv funzionano come db, a un livello di dettaglio maggiore.

Il server legge le tabelle dei diritti di accesso solo quando parte, se si desidera che le nuove impostazioni abbiano effetto senza far ripartire il server si deve eseguire l'istruzione FLUSH PRIVILEGES

Controllo dell'accesso, Stadio1: verifica della connessione

Quando un utente tenta la connessione al server, il server esegue le seguenti verifiche:

1. host dal quale l'utente si connette;
2. nome dell'utente
3. password

Se l'identità è accertata correttamente la connessione viene stabilita e il server passa allo stadio 2, altrimenti la connessione viene rifiutata.

La verifica viene eseguita impiegando i tre campi di scopo della tabella user: Host, User e Password nel seguente modo:

- il campo Host deve contenere un nome di host, 'localhost' o un indirizzo IP;
- si può usare il carattere jolly % o lasciare il campo vuoto, in quest'ultimo caso tutti gli host sono accettati;
- nel campo User non sono accettati caratteri jolly, se viene lasciato vuoto qualsiasi nome utente viene accettato;
- il campo Password può essere vuoto, in questo caso l'utente può connettersi senza specificare una password
- le password sono crittografate mediante la funzione PASSWORD()

Esempi di host e utenti

Host	User	valori accettati nella connessione
'uno.sistemi.net'	'luciano'	luciano quando si connette da uno.sistemi.net
'uno.sistemi.net'	' '	chiunque si connetta da uno.sistemi.net
'%'	'luciano'	luciano, da qualunque host si connetta
'%'	' '	chiunque si connetta da qualsiasi host
'%.sistemi.net'	'luciano'	luciano, che si connette da un host qualsiasi del dominio sistemi.net
'192.168.1.1'	'luciano'	luciano, quando si connette da 192.168.1.1
'192.168.1.%'	'luciano'	luciano, quando si connette da un host nella rete di classe C 192.168.1

Può capitare il caso che una connessione sia verificata da più record nella tabella, dato che l'identificazione deve essere unica il server procede come segue:

1. la tabella viene ordinata in memoria mettendo prima le indicazioni più specifiche e dopo le più generiche, per esempio, nella nostra tabella, le righe più specifiche sono la prima e la sesta (la corrispondenza deve essere esatta), mentre la quarta è la più generica;
2. viene usata la prima corrispondenza trovata (cioè la più specifica tra quelle che verificano la connessione).
3. nell'ordinamento il campo Host viene usato per primo, a parità di host si utilizza il campo User.

Controllo dell'accesso, Stadio2: verifica delle richieste

Stabilita la connessione, il server entra nel secondo stadio. Per ogni richiesta (comando SQL) verifica se l'utente ha sufficienti privilegi per eseguirla. La verifica viene fatta, controllando nell'ordine le tabelle:

1. user se il permesso è concesso la richiesta viene eseguita, altrimenti si prosegue con:
2. db se il permesso è concesso la richiesta viene eseguita, altrimenti si prosegue con:
3. host se il permesso è concesso la richiesta viene eseguita, altrimenti si prosegue con:
4. tables_priv se il permesso è concesso la richiesta viene eseguita, altrimenti si prosegue con:
5. columns_priv se il permesso è concesso la richiesta viene eseguita, altrimenti rifiutata

Come si vede, si procede dal permesso più generale al più specifico, è quindi se voglio concedere un permesso solo a livello più basso (per esempio per una specifica colonna) devo negarlo a tutti i livelli più alti.

Le tabelle che concedono l'accesso sono normalmente manipolate con i comandi GRANT e REVOKE

Sintassi di GRANT e REVOKE

I comandi SQL GRANT e REVOKE servono a concedere o revocare permessi ad un utente, GRANT crea anche un nuovo utente, se questo non esiste.

```
GRANT priv_type [(column_list)] [, priv_type [(column_list)] ...]
  ON {tbl_name | * | *.* | db_name.*}
  TO user_name [IDENTIFIED BY 'password']
    [, user_name [IDENTIFIED BY 'password'] ...]
  [WITH GRANT OPTION]
```

```
REVOKE priv_type [(column_list)] [, priv_type [(column_list)] ...]
  ON {tbl_name | * | *.* | db_name.*}
  FROM user_name [, user_name ...]
```

priv_type deve essere specificato con le seguenti parole chiave

ALL PRIVILEGES	FILE	RELOAD
ALTER	INDEX	SELECT
CREATE	INSERT	SHUTDOWN
DELETE	PROCESS	UPDATE
DROP	REFERENCES	USAGE

USAGE serve per creare un utente senza alcun privilegio, REFERENCES non è attualmente implementato

Un utente con l'opzione grant (WITH GRANT OPTION) può attribuire e revocare privilegi agli altri, per toglierli questa possibilità si usa REVOKE GRANT OPTION ON ... FROM ...;

Ovviamente, oltre ad essere corretta sintatticamente, l'istruzione deve anche tener conto del sistema di protezione prima spiegato, per esempio FILE o SHUTDOWN non hanno alcun senso a livello di tabella.

Nella clausola ON il simbolo * si riferisce al database corrente (o ai privilegi globali se non c'è alcun database corrente), *.* indica privilegi globali, db_name.* si riferisce al database db_name.

Il nome dell'utente può essere specificato nella forma utente@host usando eventualmente caratteri jolly o indirizzi IP, in tal caso la stringa deve essere tra virgolette (esempio luciano@"192.168.1.%").

Titolo:	Manuale essenziale MySql
Autore:	Luciano Viviani
Email:	luciano_viviani@libero.it
Classe:	TERZA INFORMATICA SERALE (3IS)
Anno scolastico:	2003/2004
Scuola:	Itis Euganeo Via Borgofuro, 6 Via Borgofuro 6 - 35042 Este (PD) - Italy Telefono 0429.21.16 - 0429.34.72 Fax 0429.41.86 http://www.itiseuganeo.it informazioni@itiseuganeo.it
Note legali:	Nessuna restrizione all'utilizzo