76.1 Struttura e preparazione

MySQL [1] è un DBMS (*Data base management system*) relazionale. Il nome contiene la sigla «SQL», a indicare che si tratta di un DBMS in grado di comprendere le istruzioni SQL.

L'installazione del servente MySQL può essere molto laboriosa, se non si dispone di un pacchetto già pronto per la propria distribuzione

GNU. La descrizione di come installare MySQL viene omessa, perché questa appare nella documentazione di MySQL in modo molto dettagliato. Qui si fa riferimento a una situazione relativamente «comune», a seguito dell'installazione automatica di un pacchetto pronto.

In generale, il servente MySQL è costituito dal demone '**mysqld**', avviato attraverso uno script della procedura di inizializzazione del sistema, che potrebbe corrispondere a '`/etc/init.d/mysql`'. Assieme a questo demone ci sono altri programmi di servizio che servono principalmente per l'amministrazione e la manutenzione delle basi di dati, il più importante dei quali è '**mysqladmin**'.

L'installazione del servente MySQL richiede, nell'ambito del sistema operativo, la preparazione di un utente e un gruppo speciali, entrambi con il nome '**mysql**'. Questa utenza dovrebbe essere già disponibile oppure potrebbe essere creata automaticamente dal sistema di installazione dei pacchetti della propria distribuzione. La directory personale associata a questo utente speciale dovrebbe rappresentare il contenitore dei file che compongono le basi di dati gestite da MySQL: '`~mysql/`'.

Il servente MySQL è sottoposto al controllo di un file di configurazione, che in condizioni normali potrebbe corrispondere a '`/etc/mysql/my.cnf`', oppure solo '`/etc/my.cnf`'. Tuttavia, questo file è suddiviso in sezioni e contiene anche la configurazione relativa ai programmi clienti, mentre una configurazione specifica del solo servente può essere collocata nel file '`~mysql/my.cnf`'.

L'amministratore del DBMS con MySQL si chiama convenzional-

mente ‘**root**’ e si prevede che in un sistema Unix coincida con l’utente ‘**root**’, ma ciò non è strettamente necessario. L’accesso da parte di questo utente, come degli altri, è sottoposto alla presentazione di una parola d’ordine che viene stabilita con l’ausilio di ‘**mysqladmin**’, o attraverso istruzioni SQL appropriate; pertanto, anche se ci può essere una corrispondenza tra utenze di MySQL e utenze del sistema operativo, le parole d’ordine usate non sono collegate tra loro.

MySQL ha una gestione particolare delle proprie utenze, distinguendole in base alla provenienza degli accessi. A seconda del contesto può capitare di utilizzare notazioni del tipo ‘**tizio@dinkel.brot.dg**’, dove si intende fare riferimento all’utente denominato ‘**tizio**’ che accede dal nodo *dinkel.brot.dg*. Deve essere subito chiaro che non si tratta di un indirizzo di posta elettronica e nemmeno di un’utenza Unix. Pertanto, anche l’utente ‘**root**’ (l’amministratore del DBMS), deve essere qualificato meglio e solitamente si fa riferimento a ‘**root@localhost**’ (l’utente ‘**root**’ che accede al servente attraverso lo stesso elaboratore che offre il servizio).

76.1.1 Configurazione del servente

Come accennato, la configurazione generale di MySQL è contenuta nel file ‘`/etc/mysql/my.cnf`’, oppure ‘`/etc/my.cnf`’, a seconda dell’organizzazione della propria distribuzione GNU, con la possibilità di usare il file ‘`~mysql/my.cnf`’ per ciò che riguarda strettamente il funzionamento del servente. Molto probabilmente, il file di configurazione generale è già predisposto per consentire un accesso locale al servente MySQL; di solito può essere utile verificare

o ritoccare solo alcune direttive, specialmente per ciò che riguarda l'abilitazione dell'accesso attraverso la rete.

Il file di configurazione contiene direttive che assomigliano in pratica all'assegnamento di variabili, nella forma seguente:

```
nome = valore
```

Queste direttive sono suddivise in sezioni, dichiarate tra parentesi quadre:

```
[sezione]
direttiva
...
```

È attraverso la presenza di queste sezioni che è possibile distinguere le direttive relative al servente da quelle di altre componenti.

Come spesso accade nei file di configurazione comuni, il simbolo '`#`' introduce un commento, fino alla fine della riga; inoltre, le righe bianche o vuote sono ignorate.

Le direttive che riguardano il funzionamento del servente sono contenute nelle sezioni '`[mysqld_safe]`' e '`[mysqld]`'. Per il momento conviene soffermarsi solo su alcune della seconda di queste sezioni:

```
[mysqld]
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
log             = /var/log/mysql.log
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
language        = /usr/share/mysql/english
```

Si comprende intuitivamente il significato di queste direttive: il demone '**`mysqld`**' viene avviato con i privilegi dell'utente '**`mysql`**'; viene usato il file '`/var/run/mysqld/mysqld.pid`' per annotare il numero PID del demone in funzione; viene utilizzato il file '`/var/run/mysqld/mysqld.sock`' come socket di dominio Unix per le comunicazioni locali, oppure la porta 3306 per le comunicazioni attraverso la rete; nel file '`/var/log/mysql.log`' vengono annotate le informazioni relative al suo funzionamento; la directory '`/usr/`' è il punto di partenza dell'installazione dei programmi e di altre componenti; la directory '`/var/lib/mysql/`' è il punto di inizio dei file che compongono le basi di dati (equivale a '`~mysql/`'); la directory usata per i file temporanei è '`/tmp/`'; infine, i messaggi mostrati dal demone sono quelli contenuti nella directory '`/usr/share/mysql/english/`', ovvero quelli espressi in lingua inglese.

Alcune direttive particolari non hanno la forma dell'assegnamento e contengono una sola stringa, nella forma seguente:

```
skip-nome
```

La presenza di queste direttive indica la disabilitazione di ciò che

viene descritto sinteticamente dalla parte finale del loro nome; per esempio, la direttiva '**skip-networking**' serve a disabilitare l'accesso attraverso la rete. Di solito, la configurazione predefinita di MySQL prevede proprio l'uso di questa direttiva per impedire l'accesso attraverso la rete, rendendo necessaria la modifica di questo file per consentirlo in modo esplicito.

Un altro tipo di direttiva, che comunque rientra nel genere normale di assegnamento, serve a dichiarare delle variabili, intese come opzioni di funzionamento, a cui si assegna anche un valore. Si tratta di direttive che hanno l'aspetto seguente:

```
set-variable = nome=valore
```

L'utilizzo di queste variabili può dipendere dal modo in cui si compilano i sorgenti di MySQL; pertanto può essere necessario stabilire su cosa si può intervenire, avviando il demone '**mysqld**' con l'opzione '**--help**', anche senza privilegi particolari:

```
$ /usr/bin/mysqld --help [Invio]
```

Oltre alla sintassi relativa alla riga di comando, si dovrebbe osservare la presenza dell'elenco seguente, che qui viene abbreviato:

```
Variables (--variable-name=value)
and boolean options {FALSE|TRUE}  Value (after reading options)
--------------------------------- ----------------------------
auto-rehash                       TRUE
character-sets-dir                (No default value)
default-character-set             (No default value)
compress                          FALSE
database                          (No default value)
…
net_buffer_length                 16384
select_limit                      1000
max_join_size                     1000000
```

76.1.1.1 Controllo dell'accesso dalla rete

Un problema comune che si può avvertire quando si cerca di mettere in funzione un servente MySQL, sta nel concedere gli accessi attraverso la rete.

Generalmente, la configurazione contenuta nel file '`/etc/mysql/my.cnf`' (o solo '`/etc/my.cnf`') prevede una direttiva che limita gli accessi al solo elaboratore locale. Ci sono due possibilità:

```
[mysqld]
…
skip-networking
…
bind-address = 127.0.0.1
…
```

Se si usa la direttiva '**`skip-networking`**', si intende concedere esclusivamente un accesso locale, tramite un socket di dominio Unix; se si usa la direttiva '**`bind-address = 127.0.0.1`**' (senza però usare anche '**`skip-networking`**'), si concede l'accesso

attraverso la rete, ma in pratica si concede esclusivamente un accesso locale, tramite l'indirizzo 127.0.0.1.

In pratica, per consentire un accesso remoto al DBMS, occorre eliminare (o commentare) entrambe queste direttive.

76.1.2 Avvio e arresto del servente

«

L'avvio e l'arresto del servente dovrebbe essere gestito da uno script della procedura di inizializzazione del sistema, che in generale potrebbe corrispondere a '`/etc/init.d/mysql`'. Se le cose stanno così, è sufficiente avviare il servizio con il comando seguente:

```
# /etc/init.d/mysql start [Invio]
```

Nello stesso modo, per arrestare il servizio basta il comando seguente:

```
# /etc/init.d/mysql stop [Invio]
```

Può essere interessante approfondire cosa succede realmente all'interno di questo script per comprendere come è organizzato MySQL nell'ambito del proprio sistema operativo.

In generale ci sono da considerare due aspetti: il demone '**mysqld**' non viene avviato direttamente, ma attraverso un altro programma, '**mysqld_safe**'; in secondo luogo, l'arresto del funzionamento del servente avviene attraverso '**mysqladmin**', che richiede l'indicazione di una parola d'ordine.

L'avvio del servente attraverso '**mysqld_safe**' non richiede la comprensione di altre questioni, a parte la necessità di accertarsi che non ci sia già un servente MySQL in funzione. Comunque, la presenza di questo programma ('**mysqld_safe**') fa sì che esista anche

una sezione apposita nella configurazione, denominata nello stesso modo:

```
[mysqld_safe]
err-log         = /var/log/mysql/mysql.err
socket          = /var/run/mysqld/mysqld.sock
```

Il problema dell'arresto del servizio è invece più complesso, in quanto si deve usare un altro programma, '**mysqladmin**', che può portare a termine l'operazione soltanto se si utilizzano i privilegi dell'amministratore del servizio, per ottenere i quali occorre fornire la parola d'ordine relativa:

```
# mysqladmin --password="ciaociao" shutdown [Invio]
```

L'amministratore di MySQL ha il nome '**root**' (in questo caso sarebbe precisamente '**root@localhost**'), che, come accennato nella premessa, coincide volutamente con il nome dell'amministrazione di un sistema Unix, pur non essendo la stessa cosa. Dal momento che ogni utente di MySQL può predisporre nella propria directory personale un file di configurazione personalizzato, corrispondente a '`~/.my.cnf`', in questo file si può anche inserire la propria parola d'ordine, con una direttiva della sezione '**[client]**', che in questo caso è riferita all'utente '**root**':

```
[client]
user      = root
password  = ciaociao
```

Così facendo, come viene chiarito in seguito, quando un utente del sistema operativo accede a un servente MySQL locale, se l'utenza di MySQL coincide con il nominativo usato nell'ambito del sistema operativo, può fare a meno di fornire la propria parola d'ordine perché già definita nella configurazione personale. Secondo questo

principio, l'utente '**root**' del sistema operativo potrebbe fare altrettanto per consentire a script che vengono avviati automaticamente di svolgere il loro compito.

Esistono comunque dei raggiri differenti, per evitare di costringere l'utente '**root**' del sistema operativo a inserire la parola d'ordine nel file di configurazione '`~/.my.cnf`'. In particolare, la distribuzione GNU/Linux Debian definisce un utente speciale, denominato '**debian-sys-maint**' ('**debian-sys-maint@localhost**'), con i privilegi necessari, a cui associa il file di configurazione '`/etc/mysql/debian.cnf`', avviando poi '**mysqladmin**' con l'opzione '**--defaults-extra-file**':

```
# mysqladmin --defaults-extra-file=/etc/mysql/debian.cnf ... [Invio]
```

Naturalmente, il file '`/etc/mysql/debian.cnf`' non deve essere leggibile agli utenti comuni del sistema operativo, dal momento che contiene la parola d'ordine per le funzioni amministrative gestite in modo automatico dal sistema stesso:

```
# Automatically generated for Debian scripts. DO NOT TOUCH!
[client]
host     = localhost
user     = debian-sys-maint
password = sddgetGRFhyjftuP
```

76.1.2.1 Struttura iniziale dei dati

«

Perché il servente MySQL possa essere avviato e funzionare, è necessario che sia stata predisposta una struttura iniziale di dati, che serve successivamente ad accogliere le basi di dati. Si tratta di file di una base di dati speciale, che si colloca assieme alle altre nella directory '`~mysql/`'.

Questa struttura iniziale si crea con il comando '**mysql_install_db**', che corrisponde a uno script. Si usa semplicemente così:

```
# mysql_install_db [Invio]
```

Può succedere che la propria distribuzione GNU, pur predisponendo un pacchetto per il servente MySQL, non provveda alla creazione di questa struttura iniziale dei dati; in tal caso non si riesce ad avviare il servente e occorre provvedere da soli a usare il comando '**mysql_install_db**'.

76.1.3 Utenze

MySQL distingue le proprie utenze in base a un nominativo e al nome dell'elaboratore da cui questi accedono. Il nominativo può anche essere assente e in tal caso si parla di utenze anonime. L'accesso può essere sottoposto al controllo di una parola d'ordine; inoltre, può anche essere consentito l'accesso a una base di dati senza essere utenti conosciuti.

Dal momento che l'utente è sempre riferito a un certo elaboratore, diventa necessario provvedere a un sistema di risoluzione dei nomi, anche se si tratta solo del file '`/etc/hosts`'; è comunque assolutamente indispensabile che sia stato definito il nome '**localhost**', riferito all'indirizzo IPv4 127.0.0.1, per consentire almeno l'accesso all'utente '**root@localhost**' che è l'amministratore quando accede dallo stesso elaboratore che ospita il servente MySQL.

Un primo controllo di accesso si ottiene con la configurazione del servente MySQL, dove è possibile escludere l'accesso attraverso la rete con la direttiva '**skip-networking**', cosa che è utile fare quando il sistema di utenze e dei privilegi relativi non è ancora stato definito.

Per poter iniziare a organizzare le basi di dati e le utenze di MySQL, occorre agire con i privilegi dell'utente '**root**'. Quando si installa MySQL per la prima volta, è molto probabile che questo utente risulti essere sprovvisto di parola d'ordine, consentendo in pratica a qualunque utente di dichiararsi '**root**', senza bisogno di altre forme di riconoscimento (ecco perché inizialmente è bene che la configurazione impedisca almeno l'accesso dall'esterno, attraverso la rete). Per associare una parola d'ordine all'utente '**root**' presso l'elaboratore che ospita il servente MySQL (quindi '**root@localhost**') quando questo ne è ancora sprovvisto, basta usare il comando seguente:

```
# mysqladmin -u root password 'ciaociao' [Invio]
```

Come si intende, la parola d'ordine che appare nell'esempio è proprio «ciaociao».

A partire dal momento in cui la parola d'ordine è stata definita, ogni accesso al servente compiuto da questo utente richiede la sua indicazione, a meno che questa parola d'ordine risulti inserita nel file di configurazione personale '`~/.my.cnf`':

```
[client]
user     = root
password = ciaociao
```

In questo caso, facendo riferimento alla sezione '**[client]**', la con-

figurazione riguarda qualunque situazione in cui si accede al servente; in pratica riguarda qualunque programma. Eventualmente, si possono definire contesti più precisi; per esempio come nel caso seguente, in cui si consente di non specificare la parola d'ordine solo quando si utilizza il programma '**mysqladmin**':

```
[mysqladmin]
user     = root
password = ciaociao
```

Il sistema di gestione delle utenze di MySQL è abbastanza complesso, dove per esempio, è possibile anche concedere dei privilegi di accesso a utenti qualunque provenienti da un certo nodo, o da un certo gruppo di nodi.

Inizialmente dovrebbe essere previsto l'utente '**root@localhost**', come già descritto più volte, ma potrebbe esistere anche un utente '**root@*hostname***', ovvero l'utente '**root**' che accede dal nodo corrispondente al nome che si ottiene con il comando '**hostname**'. Entrambe queste utenze hanno tutti i privilegi possibili ed entrambe potrebbero essere inizialmente senza parola d'ordine.
Assieme alle utenze '**root@**...' potrebbero essere stati definiti dei privilegi di accesso molto limitati per qualunque nominativo-utente che accede dal nodo locale, ovvero '**@localhost**' (ma si rappresenta necessariamente come '**''@localhost**'). Di solito ciò consente di fare degli esperimenti con la base di dati denominata '**test**', prima ancora di avere studiato i criteri di controllo degli accessi.

Per quanto riguarda l'indicazione dei nominativi-utente, esistono solo due possibilità: indicare un nome preciso, oppure non indicarlo affatto. Nel secondo caso si intende fare riferimento a qualunque utente nell'ambito del nodo o del gruppo di nodi a cui la definizione dei privilegi è associata.

Il nodo di accesso può essere indicato per nome, per numero, oppure si può fare riferimento a un gruppo di nodi con l'uso di caratteri jolly oppure associando a un indirizzo una maschera di rete. I caratteri jolly in questione sono il simbolo di percentuale ('`%`') e il trattino basso ('_'). Segue una tabellina che descrive alcuni esempi di associazione tra utenti e nodi di accesso.

utente*@*nodo	Descrizione
tizio@dinkel.brot.dg	L'utente '**`tizio`**' che accede dal nodo *dinkel.brot.dg*.
@dinkel.brot.dg	Qualunque utente che accede dal nodo *dinkel.brot.dg*.
tizio@%	L'utente '**`tizio`**' che accede da qualunque nodo.
@%	Qualunque utente che accede da qualunque nodo.
tizio@%.brot.dg	L'utente '**`tizio`**' che accede da un nodo che appartiene al dominio *brot.dg*.
tizio@brot.brot.%	L'utente '**`tizio`**' che accede da un nodo che appartiene a domini che iniziano per *dinkel.brot.*$*$.
tizio@111.112.113.114	L'utente '**`tizio`**' che accede dal nodo corrispondente all'indirizzo IPv4 111.112.113.114.

utente@*nodo*	Descrizione
tizio@111.112.113.%	L'utente '**tizio**' che accede da nodi con indirizzi IPv4 111.112.113.∗.
tizio@111.112.113.0/255.255.255.0	L'utente '**tizio**' che accede da nodi con indirizzi IPv4 111.112.113.∗ (come nell'esempio precedente).

A seconda del contesto, le coordinate di un utente si indicano come nella tabella, inserendo il carattere '**@**' per separare le due parti, oppure in campi distinti. In molti casi, nell'ambito delle istruzioni di MySQL è necessario proteggere il nome dell'utente e l'indicazione del nodo o dei nodi di accesso attraverso apici singoli. Per esempio, per poter indicare qualunque utente, come nel secondo esempio della tabella, è necessario delimitare la stringa nulla: '**''@dinkel.brot.dg**'. Così, quando si usa il carattere jolly '**%**' è indispensabile delimitare l'indicazione riferita al nodo: '**tizio@'%.brot.dg'**'. In generale non si sbaglia se si delimitano sempre queste due componenti, anche se non dovesse servire: '**''tizio'@'dinkel.brot.dg'**'.

76.1.4 Accesso comune al servente

Prima di poter definire delle politiche di accesso, è necessario prendere un po' di confidenza con il programma '**mysql**', che consente di interagire con il servente MySQL. Si è già accennato al fatto che inizialmente dovrebbe risultare permesso l'accesso da parte di

qualunque utente, inoltre dovrebbe essere disponibile la base di dati '**test**', a cui qualunque utente può accedere.

```
mysql [opzioni] [base_di_dati]
```

```
cat file_sql | mysql [opzioni] [base_di_dati]
```

I modelli sintattici mostrano la possibilità di indicare delle opzioni ed eventualmente il nome di una base di dati da aprire inizialmente. Nel secondo modello, si vede in particolare come si può alimentare il programma '**mysql**' con uno script SQL. La tabella successiva descrive alcune delle opzioni che possono essere utilizzate.

Opzione	Descrizione
`-h` ***nodo*** `--host=`***nodo***	Specifica il nome del nodo a cui ci si vuole connettere.
`-p` [***parola_d'ordine***] `--password` [`=`***parola_d'ordine***]	Specifica la parola d'ordine da fornire per ottenere l'accesso; se non viene indicata come argomento dell'opzione, viene richiesta in modo interattivo. In generale è meglio evitare di fornire la parola d'ordine già nella riga di comando, per non renderla evidente nell'elenco dei processi elaborativi.

Opzione	Descrizione
`-P` ***n_porta*** `--port=`***n_porta***	Specifica la porta da utilizzare per accedere al servizio; in modo predefinito è la numero 3306.
`-u` ***utente*** `--user=`***utente***	Specifica il nominativo-utente con il quale si vuole accedere.

Il programma '**`mysql`**' è uno di quelli che prende in considerazione la configurazione di MySQL, soprattutto per quanto riguarda il file '`~/.my.cnf`', dove gli utenti possono inserire il proprio nominativo-utente da utilizzare con MySQL e la parola d'ordine, per non dover ogni volta utilizzare le opzioni '**`-u`**' e '**`-p`**':

```
[client]
user     = tizio
password = supersegreta
```

Per esempio, se un certo utente del sistema operativo ha la necessità di identificarsi con il nominativo '**`tizio`**' e la sua parola d'ordine:

`$` **`mysql -u tizio -p`** … [*Invio*]

`Enter password:` ***digitazione_all'oscuro*** [*Invio*]

Oppure, disponendo del file '`~/.my.cnf`' mostrato sopra, può fare a meno di queste opzioni e dell'indicazione della parola d'ordine.

Viene mostrato un esempio riferito all'utente '**`tizio`**' che accede a un servente MySQL locale:

`$` **`mysql -u tizio`** [*Invio*]

Molto probabilmente non è necessario inserire alcuna parola d'ordine, dal momento che inizialmente dovrebbe essere stata inserita automaticamente l'utenza anonima '**@localhost**' senza parola d'ordine:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14 to server version: 4.0.13-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Da questo momento appare un invito speciale, rappresentato dalla stringa '**mysql>**'. La prima cosa che può essere conveniente fare è verificare la disponibilità di basi di dati a cui si può accedere:

```
mysql> SHOW DATABASES; [Invio]
```

Molto probabilmente si ottiene precisamente quanto segue, ovvero il riferimento all'unica base di dati, '**test**', a cui è consentito accedere con questo utente:

```
+----------+
| Database |
+----------+
| test     |
+----------+
1 row in set (0.03 sec)
```

Si può selezionare la base di dati con il comando seguente:

```
mysql> USE test; [Invio]

Database changed
```

Quindi si può tentare di consultare l'elenco delle relazioni disponibili, con il comando seguente, ma inizialmente la basi di dati '**test**' non ne contiene alcuna:

```
mysql> SHOW TABLES; [Invio]

  Empty set (0.00 sec)
```

A titolo di esempio si vuole creare la relazione '**Indirizzi**' con il codice SQL seguente:

```
CREATE TABLE Indirizzi (
        Codice          INTEGER,
        Cognome         CHAR(40),
        Nome            CHAR(40),
        Indirizzo       VARCHAR(60),
        Telefono        VARCHAR(40)
    );
```

Con il programma '**mysql**' si può fare così:

```
mysql> CREATE TABLE Indirizzi ( [Invio]

    -> Codice INTEGER, [Invio]

    -> Cognome CHAR(40), [Invio]

    -> Nome CHAR(40), [Invio]

    -> Indirizzo VARCHAR(60), [Invio]

    -> Telefono VARCHAR(40) [Invio]

    -> ); [Invio]

  Query OK, 0 rows affected (0.06 sec)
```

Come si può osservare, finché l'istruzione SQL non risulta completa, appare un invito differente: '**->**'.

Per completare l'esempio si può inserire qualche dato e poi si può visualizzare il contenuto della relazione:

```
mysql> INSERT INTO Indirizzi VALUES (01, 'Pallino', 'Pinco', [Invio]
    -> 'Via Biglie 1', '0222,222222'); [Invio]

  Query OK, 1 row affected (0.06 sec)

mysql> SELECT * FROM Indirizzi; [Invio]

  +--------+---------+-------+-------------+-------------+
  | Codice | Cognome | Nome  | Indirizzo   | Telefono    |
  +--------+---------+-------+-------------+-------------+
  |      1 | Pallino | Pinco | Via Biglie 1 | 0222,222222 |
  +--------+---------+-------+-------------+-------------+
  1 row in set (0.00 sec)
```

Per concludere il funzionamento di '**mysql**' basta il comando interno '**quit**', che si può esprimere anche come '**\q**':

```
mysql> \q[Invio]

  Bye
```

76.1.5 Base di dati amministrativa

«

MySQL gestisce le proprie informazioni amministrative all'interno della base di dati '**mysql**', a cui si dovrebbe poter accedere soltanto in qualità di utente '**root**' (possibilmente '**root@localhost**'), o comunque solo attraverso utenze speciali.

```
# mysql -u root -p[Invio]

Enter password: digitazione all'oscuro [Invio]
```

In questo modo si vuole accedere al servente MySQL locale, in qualità di utente '**root**' ('**root@localhost**'), fornendo anche la

parola d'ordine.

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15 to server version: 4.0.13-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Si controlla la disponibilità di basi di dati esistenti:

```
mysql> SHOW DATABASES; [Invio]

+----------+
| Database |
+----------+
| mysql    |
| test     |
+----------+
2 rows in set (0.01 sec)
```

Si accede alla base di dati '**mysql**':

```
mysql> USE mysql; [Invio]

Database changed
```

Si elencano le relazioni disponibili:

```
mysql> SHOW TABLES; [Invio]
```

```
+-----------------+
| Tables_in_mysql |
+-----------------+
| columns_priv    |
| db              |
| func            |
| host            |
| tables_priv     |
| user            |
+-----------------+
6 rows in set (0.00 sec)
```

Il significato dettagliato di queste relazioni può essere studiato nella documentazione originale che accompagna MySQL. Ciò che è importante comprendere è che non si può consentire l'accesso a queste relazioni a utenti che non hanno un ruolo amministrativo. Tuttavia, durante la fase di studio di MySQL da parte di chi deve poi amministrarne il servizio, è utile imparare a consultare queste relazioni. Per esempio, è utile essere al corrente delle utenze che sono effettivamente previste:

```
mysql> SELECT User, Host, Password FROM user; [Invio]

+------------------+-----------+------------------+
| user             | host      | password         |
+------------------+-----------+------------------+
| root             | localhost | 576gtd435e967361 |
| root             | dinkel    |                  |
|                  | localhost |                  |
|                  | dinkel    |                  |
| debian-sys-maint | localhost | 69b3c178kbvcd325 |
+------------------+-----------+------------------+
5 rows in set (0.00 sec)
```

Quello che si vede qui è quanto si potrebbe ottenere dopo aver installato MySQL in una distribuzione GNU/Linux Debian, attraverso pacchetti appositi, in un elaboratore dove il comando '**hostname**' restituisce il nome '**dinkel**'. Si può notare la presenza dell'utente speciale '**debian-sys-maint**', a cui si è già accennato, ma si deve portare attenzione alle utenze «normali». Si può osservare che ci sono due utenti '**root**'; precisamente si tratta di '**root@localhost**' e di '**root@dinkel**'. La seconda di queste due utenze rappresenta l'utente '**root**' che accede localmente, ma attraverso la rete, da un'interfaccia che è associata al nome dell'elaboratore (in questo caso *dinkel.brot.dg*). Può anche darsi che un'utenza annotata in questo modo non risulti funzionante, ma rimane il problema, dato il fatto che si tratta di un'utenza importante e senza parola d'ordine per accedere (per quanto riguarda l'utenza '**root@localhost**' è già stato mostrato come definire la parola d'ordine ed è per questo che appare qualcosa nella colonna '**password**'). Eventualmente, si possono fare considerazioni simili per l'utenza anonima '**@dinkel**'.

Probabilmente è superfluo precisare che nella colonna '**password**', se esiste una parola d'ordine associata all'utenza, appare una stringa che rappresenta la parola d'ordine cifrata.

```
mysql> \q[Invio]

  Bye
```

Durante la fase della prima installazione di MySQL uno script si occupa di creare le directory che devono ospitare i file che compongono le basi di dati, inserendo la base di dati amministrativa ('**mysql**'). In pratica, è questo script che definisce anche le utenze iniziali, senza parola d'ordine. Questo script è '**mysql_install_db**'.

76.2 Gestione del DBMS

Una volta superata la fase della configurazione del servizio di MySQL; una volta compreso come utilizzare gli strumenti essenziali per interagire con questo, si può passare alla gestione del DBMS, che implica l'amministrazione delle utenze e delle basi di dati.

76.2.1 Controllo delle utenze e degli accessi

Il modo normale per creare un'utenza con MySQL è quello di concedere dei privilegi di accesso, attraverso l'istruzione '**`GRANT`**'; se poi quell'utenza esiste già, i privilegi in questione vengono solo aggiunti a quelli già esistenti. Per eliminare un'utenza, invece, non è sufficiente privarla di tutti i privilegi con l'istruzione '**`REVOKE`**', perché occorre anche eliminare la tupla corrispondente all'utenza nella relazione '**`user`**' della base di dati '**`mysql`**'.

Il comportamento di MySQL è abbastanza diverso rispetto allo standard ANSI, per quanto riguarda l'identificazione delle utenze e l'attribuzione o l'eliminazione dei privilegi relativi. In particolare, è importante il fatto che MySQL distingua le utenze in base al nodo di provenienza; inoltre è possibile concedere privilegi complessivi: per una base di dati completa, o anche per tutte le basi di dati esistenti. A questo si aggiunga che è possibile fare riferimento a una relazione di una base di dati indicando esplicitamente la base di dati relativa, con la forma: '***base_di_dati*** **.** ***relazione***'.

Gli schemi seguenti mostrano la sintassi semplificata per l'uso delle istruzioni '**`GRANT`**' e '**`REVOKE`**' con MySQL:

```
GRANT privilegio [, privilegio] ...
  ON {relazione | * | base_di_dati.* | *.*}
  TO utenza [IDENTIFIED BY [PASSWORD] 'parola_d'ordine']
    [, utenza [IDENTIFIED BY [PASSWORD] 'parola_d'ordine']] ...
  [WITH GRANT OPTION]
```

```
REVOKE privilegio [, privilegio] ...
    ON {relazione | * | base_di_dati.* | *.*}
    FROM utenza [, utenza] ...
```

I privilegi che possono essere concessi o revocati sono espressi attraverso una parola chiave. Alcuni di questi privilegi sono descritti nell'elenco seguente:

Privilegio	Descrizione
`ALL [PRIVILEGES]`	concede tutti i privilegi disponibili;
`ALTER`	consente la modifica delle relazioni;
`CREATE`	consente la creazione delle relazioni;
`DELETE`	consente la cancellazione dei dati contenuti nelle relazioni;
`DROP`	consente l'eliminazione delle relazioni;
`INDEX`	consente di creare ed eliminare degli indici;
`INSERT`	consente l'inserimento di dati nelle relazioni;

Privilegio	Descrizione
`SELECT`	consente la lettura dei dati nelle relazioni;
`UPDATE`	consente la modifica dei dati all'interno delle relazioni;
`USAGE`	serve solo a creare un utente senza privilegi;
`GRANT OPTION`	consente di dare ad altri i propri privilegi.

Si osservi che MySQL prevede anche privilegi «particolari», che dipendono dalle proprie specificità rispetto allo standard ANSI. In generale, si può considerare che l'utente '**`root`**' deve possedere tutti i privilegi, mentre possono esistere delle utenze amministrative fittizie (come nel caso di '**`debian-sys-maint`**') con privilegi particolari legati alla possibilità di arrestare il funzionamento del servente MySQL.

Lo standard ANSI prevede la concessione di privilegi su relazioni singole, mentre MySQL permette di fare riferimento a basi di dati complete. Pertanto, nel modello sintattico mostrato sono apparse delle notazioni speciali:

Modello	Descrizione
relazione	rappresenta una relazione singola, che può contenere anche l'indicazione della basi di dati che la contiene, nella forma '***base_di_dati . relazione***';
`*`	l'asterisco rappresenta tutte le relazioni della basi di dati attiva, ma se non è stata selezionata una base di dati in precedenza, si fa riferimento a tutte le basi di dati;

Modello	Descrizione
base_di_dati . *	l'indicazione di una base di dati con un asterisco al posto del nome della relazione, serve a fare riferimento a tutta la base di dati nel complesso;
* . *	l'indicazione di due asterischi separati da un punto serve a fare riferimento esplicito a tutte le relazioni di tutte le basi di dati.

Quando si utilizza l'istruzione '**GRANT**' è consentito l'uso dei caratteri jolly '_' e '%', con il significato comune nell'ambito del linguaggio SQL. Ciò consente di fare riferimento a gruppi di relazioni e a gruppi di basi di dati, secondo la corrispondenza del modello. Tuttavia, nel caso questi simboli debbano essere usati in modo letterale, è necessario proteggerli con la barra obliqua inversa: '_' e '\%' (in generale, è improbabile che si dia un nome a una base di dati o a una relazione che contenga il simbolo di percentuale, ma è più probabile che si pensi invece di usare il trattino basso).

Come già accennato altre volte, l'utenza tiene conto anche della provenienza dell'accesso, secondo la forma seguente:

nominativo_utente [@*nodo_di_provenienza*]

Pertanto, l'indicazione del nominativo è obbligatoria, mentre si può omettere la specificazione del nodo di provenienza, se si vuole fare riferimento a qualunque origine.

Sono già stati descritti i modi in cui si può rappresentare un utente secondo il modello '*utente*@*nodo*'; in particolare è già stato descrit-

to l'utilizzo dei caratteri jolly (anche se sono stati mostrati soltanto esempi con il simbolo '`%`').

Si ricorda comunque che si possono usare i caratteri jolly soltanto nella parte che descrive i nodi di provenienza

Infine, è già stata indicata la necessità di usare gli apici singoli per delimitare separatamente il nominativo e la specificazione del nodo di provenienza quando questi nomi contengono caratteri «particolari», compresi i caratteri jolly '`%`' e '_'.

Si possono presentare delle ambiguità nell'individuazione dei privilegi delle utenze. Per esempio, può esistere l'utente anonimo '**`''@dinkel.brot.gd`**', l'utente '**`tizio@'%'`**' e l'utente '**`tizio@dinkel.brot.gd`**': quando un utente accede valgono per lui i privilegi più specifici che gli si possono individuare, altrimenti, in presenza di utenze anonime, i privilegi di queste prevarrebbero.

Come si vede dal modello sintattico dell'istruzione '**`GRANT`**', è possibile specificare una parola d'ordine. Quando si concedono dei privilegi a un utente che non è ancora stato definito e non si stabilisce la parola d'ordine, l'utenza in questione rimane priva di parola d'ordine; pertanto, per accedere non deve essere fornita. Se invece l'utenza esiste già, i privilegi vengono aggiunti e la presenza di una parola d'ordine eventuale serve solo per cambiare quella preesistente. Tuttavia, è possibile cambiare una parola d'ordine anche con l'i-

struzione '**SET PASSWORD**', se si tratta della propria o se si hanno i privilegi dell'amministratore:

```
SET PASSWORD FOR utenza = PASSWORD('parola_d'ordine')
```

In alternativa, è anche possibile usare il comando '**mysqladmin**' (dal sistema operativo), ma solo per cambiare la propria parola d'ordine:

```
$ mysqladmin password 'supersegreta' [Invio]
```

Se si utilizza l'opzione '**GRANT OPTION**', o '**WITH GRANT OPTION**', si permette all'utenza a cui si fa riferimento di concedere ad altri gli stessi privilegi di cui si dispone.

Come accennato, l'eliminazione di un'utenza richiede prima l'eliminazione di tutti i privilegi e quindi la cancellazione dalla relazione '**user**' della base di dati '**mysql**'.

Nel seguito vengono descritti alcuni esempi attraverso una sequenza lineare di operazioni, a cominciare dall'avvio del programma '**mysql**' per interagire con il servente.

```
$ mysql -u root -p [Invio]

Enter password: parola_d'ordine [Invio]

  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 6 to server version: 4.0.13-log

  Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Si crea una base di dati nuova:

```
mysql> CREATE DATABASE Magazzino; [Invio]
```

```
Query OK, 1 row affected (0.26 sec)
```

Si crea un utente amministratore per la base di dati appena creata, che può accedere da dove vuole:

```
mysql> GRANT ALL ON Magazzino.* TO amministratore@'%' ↩
↪      IDENTIFIED BY 'segreta' WITH GRANT OPTION; [Invio]

Query OK, 0 rows affected (0.35 sec)
```

Si osservi che se esiste un'utenza anonima riferita al nodo locale (utenza che verrebbe indicata come '**''@localhost**'), se l'utente appena creato volesse accedere localmente, non verrebbe identificato come amministratore, ma solo come utente anonimo. Per risolvere il problema si potrebbe aggiungere l'utente '**amministratore@localhost**', ma in questo caso si preferisce eliminare l'utenza anonima che interferisce e non si vuole mantenere:

```
mysql> REVOKE ALL ON *.* FROM ''@localhost; [Invio]

Query OK, 0 rows affected (0.15 sec)

mysql> USE mysql; [Invio]

Database changed

mysql> DELETE FROM user WHERE user = '' ↩
↪      AND host = 'localhost'; [Invio]

Query OK, 0 rows affected (0.15 sec)
```

Si crea un'altra utenza in grado di consultare e di modificare i dati delle relazioni che deve contenere la base di dati '**Magazzino**'; anche in questo caso si consente l'accesso da qualunque nodo:

```
mysql> GRANT DELETE, INSERT, SELECT, UPDATE ON Magazzino.* ↩
↪      TO tizio@'%' IDENTIFIED BY 'ottimo'; [Invio]

  Query OK, 0 rows affected (0.05 sec)
```

Supponendo di avere installato MySQL nell'elaboratore *dinkel.brot.dg*, dovrebbero essere presenti anche le utenze '**''@dinkel**' e '**root@dinkel**', che si preferisce eliminare:

```
mysql> REVOKE ALL ON *.* FROM ''@dinkel; [Invio]

  Query OK, 0 rows affected (0.15 sec)

mysql> REVOKE ALL ON *.* FROM root@dinkel; [Invio]

  Query OK, 0 rows affected (0.15 sec)

mysql> USE mysql; [Invio]

  Database changed

mysql> DELETE FROM user WHERE user = '' ↩
↪       AND host = 'dinkel'; [Invio]

  Query OK, 0 rows affected (0.15 sec)

mysql> DELETE FROM user WHERE user = 'root' ↩
↪       AND host = 'dinkel'; [Invio]

  Query OK, 0 rows affected (0.15 sec)

mysql> \q [Invio]

  Bye
```

La documentazione di MySQL descrive anche come compiere tutte queste operazioni amministrative intervenendo esclusivamente nella base di dati '**mysql**'. Nel caso si preferisca intervenire in

quel modo, è necessario concludere le operazioni di modifica o aggiornamento delle relazioni amministrative con l'istruzione '**`FLUSH PRIVILEGES`**'.

76.2.2 Amministrazioni varie attraverso il sistema operativo

«

In questo capitolo si è fatto riferimento più volte al programma '**`mysqladmin`**' a proposito della possibilità di assegnare e modificare la parola d'ordine di un utente. Questo programma ha anche altre funzionalità; in particolare consente di creare ed eliminare una base di dati e di arrestare il funzionamento del servente MySQL, senza bisogno di utilizzare istruzioni SQL. Vengono sintetizzate le sintassi da utilizzare per le varie occasioni:

Comando	Descrizione

`mysqladmin [-u root] [-p] ↩`
`↪[-h` ***nodo*** `] ↩`
`↪create database` ***base_di_dati***
	crea la base di dati indicata;

`mysqladmin [-u root] [-p] ↩`
`↪[-h` ***nodo*** `] ↩`
`↪drop database` ***base_di_dati***
	elimina la base di dati indicata con tutto il suo contenuto;

`mysqladmin [-u` ***utente*** `] [-p] ↩`
`↪[-h` ***nodo*** `] ↩`
`↪password` ***parola_d'ordine***
	assegna o cambia la parola d'ordine per accedere;

`mysqladmin [-u root] [-p] ↩` `↪[-h` ***nodo*** `] shutdown`	arresta il funzionamento del servente.

76.2.3 Ripristino della parola d'ordine dell'amministratore

Nel caso fosse necessario modificare la parola d'ordine dell'amministratore del DBMS ('**root**'), senza poter conoscere quella precedente, esiste un procedimento che è bene annotare. Per prima cosa si deve arrestare il servente, nel caso questo fosse in funzione; dovrebbe essere possibile farlo così:

```
# /etc/init.d/mysql stop[Invio]
```

Successivamente si deve avviare '**mysqld_safe**', con l'opzione '**--skip-grant-tables**', in modo da ignorare completamente il controllo dei privilegi concessi (o negati) agli utenti del DBMS:

```
# mysqld_safe --skip-grant-tables &[Invio]
```

Se il servizio si avvia regolarmente, è possibile accedere alle basi di dati senza vincoli; pertanto è possibile cambiare parola d'ordine intervenendo direttamente nella base di dati amministrativa '**mysql**':

```
# mysql -u root[Invio]

  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 2 to server version: ↩
↪4.0.24_Debian-10-log

  Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> USE mysql;[Invio]

  Reading table information for completion of table and column
  names. You can turn off this feature to get a quicker startup
  with -A

  Database changed
```

```
mysql> UPDATE user SET password=password("supersegreta") ↩
↪      WHERE user="root"; [Invio]

  Query OK, 2 rows affected (0.00 sec)
  Rows matched: 2  Changed: 2  Warnings: 0

mysql> \q[Invio]

  Bye
```

Un modo alternativo, ma drastico, di ripristinare l'utenza dell'amministratore senza parola d'ordine, consiste nell'eliminazione «manuale» della base di dati '**mysql**', da ricostruire con l'aiuto di '**mysql_install_db**'. Naturalmente, in questo modo si perdono le informazioni su tutti gli altri utenti del DBMS.

76.2.4 Archiviazione e recupero delle basi di dati

«

MySQL gestisce le sue basi di dati come sottodirectory di '`~mysql/`', che di solito corrisponde a '`/var/lib/mysql/`'. Per esempio, la base di dati '**prova**' corrisponde alla struttura che si articola a partire da '`~mysql/prova/`'.

Per archiviare una base di dati è sufficiente fare una copia della struttura che la riguarda; per ripristinare una base di dati è sufficiente rimpiazzare la struttura esistente con la sua copia fatta in precedenza; per duplicare una base di dati è sufficiente fare la copia della struttura di origine, utilizzando un nome differente. Per esempio, disponendo della base di dati '**prova**', è possibile creare un'altra identica, ma con nome differente, così:

```
# cp -dpRv ~mysql/prova ~mysql/prova2[Invio]
```

In base all'esempio si ottiene una seconda base di dati con il nome '**prova2**', con lo stesso contenuto di '**prova**'.

Tuttavia, con il procedere dello sviluppo di MySQL bisogna considerare la possibilità che il formato dei file usati per descrivere le relazioni delle basi di dati cambi nel tempo. Pertanto, per archiviare una base di dati in modo abbastanza duraturo, è necessario creare un file di testo contenente comandi SQL. Si ottiene questo con il programma '**mysqldump**':

```
# mysqldump -u root -p prova > prova.sql [Invio]
```

L'esempio mostra in che modo archiviare la base di dati '**prova**' nel file '`prova.sql`'. Per fare questo, logicamente, ci si identifica in qualità di amministratore (con il nominativo '**root**') e nell'esempio si usa l'opzione '**-p**', per richiedere l'inserimento della parola d'ordine, supponendo che ciò sia necessario.

Per ripristinare un file ottenuto in questo modo, occorre prima creare o ricreare la base di dati; nell'esempio seguente si cancella prima la base di dati '**prova**', quindi la si ricrea vuota:

```
# mysql -u root -p [Invio]

Enter password: digitazione_all'oscuro [Invio]

  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 6 to server version: 4.0.13-log

  Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> DROP DATABASE prova; [Invio]

  Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CREATE DATABASE prova; [Invio]

  Query OK, 1 row affected (0.00 sec)

mysql> \q[Invio]

  Bye
```

Per recuperare la base di dati archiviata in forma di comandi SQL, dopo un controllo visivo del suo contenuto, si può procedere così:

```
# mysql -u root -p prova < prova.sql [Invio]

Enter password: digitazione_all'oscuro [Invio]
```

In questo modo, si fa leggere a '**mysql**' il contenuto del file '`prova.sql`', che dovrebbe contenere le istruzioni per la rigenerazione delle relazioni della base di dati originaria.

Si osservi che nella riga di comando di '**mysql**' appare l'indicazione della base di dati di destinazione; pertanto, si potrebbe benissimo ricreare una base di dati con un nome differente da quello che aveva nel momento dell'archiviazione in forma di comandi SQL. Di conseguenza, questa tecnica di archiviazione e recupero è anche quella più appropriata per duplicare una base di dati.

76.3 Riferimenti

«

- *MySQL reference manual*, http://dev.mysql.com/doc/mysql/en/index.html

[1] **MySQL** GNU GPL