

Security and MySQL

[AUTHENTICATING CONNECTIONS](#) [AUTHENTICATING REQUESTS](#) [CONFIGURING DATABASES FOR SECURITY](#)
[GRANT, REVOKE AND THE GRANT TABLES](#) [ACCESS ROLES](#)

DBMS security is grounded on the principle that the DBMS must control all its resources, authorising every connection and activity via strong authentication.

The MySQL manual describes the MySQL security system as advanced and non-standard. Practically speaking, how advanced MySQL security is depends on what your requirements are. If you expect well differentiated tools and switches for configuring security, you will be pleased; if you expect built-in support for default databases and default languages for users, or if you hope to use views to implement need-to-know access control, or if you expect support for role-based and group-based access control, you will be disappointed. MySQL does not, for example, provide direct built-in support even for standard security roles like SYSADM, DBA, DBMAINT, SSO or SYSOPR. Also non-standard is MySQL's inclusion of the *user host* in the definition of a user name: a user is specified by `username@hostname`, though `host` accepts wildcards. These departures from the norm create issues that must be dealt with. Still, the MySQL security implementation is nimble and powerful.

GRANT, REVOKE and the MySQL grant tables

The job of the security system is to determine who may connect, and after connection, who may do what with a database. The kernel of the MySQL security system is a set of six tables in the `mysql` database—`user`, `host`, `db`, `tables_priv`, `columns_priv`, `procs_priv`—loaded into memory when the server starts unless the [skip-grant-tables](#) option has been specified. Every connection request, and every database access and activity request, is checked against these in-memory grant tables:

- `user` holds connection rights and *global* or *superuser* privileges,
- `db` holds privileges which are scoped to specified *databases*,
- `host` records hosts in addition to `db.host` at which a given user has been GRANTED *database* privileges,
- `tables_priv` holds privileges scoped to specified *tables*,
- `columns_priv` holds privileges scoped to specified table *columns*,
- `procs_priv` holds privileges for stored routines (since version 5.0.3).

GRANT and REVOKE commands ([Chapter 6](#)) are the preferred methods of adding, changing and removing privileges stored in these tables; direct INSERTs, UPDATES and DELETES also change privileges, but require FLUSH PRIVILEGES for immediate effect.

The five kinds of security specification permitted by the GRANT command map to specific tables and columns in the `mysql` database (Table 19-1):

To read the rest of this and other chapters, [buy a copy of the book](#)

[TOC](#) [Previous](#) [Next](#)
