14 DAYS Upcoming Tech Talk: Developing on Windows With the WSL

Community

TUTORIAL

How To Install MySQL on Ubuntu 20.04

Ubuntu MySQL Databases Ubuntu 20.04

By Mark Drake

Last Validated on July 30, 2020 • Originally Published on April 23, 2020 292.5k

English

Not using Ubuntu 20.04?
Choose a different version or distribution.

Ubuntu 20.04

A previous version of this tutorial was written by Hazel Virdó

Introduction

MySQL is an open-source database SCROLL TO TOP em, commonly installed as part of the popular LAMP (Linux, Apache, MySQL, PHP/Python/Perl) stack. It implements the relational model and uses Structured Query Language (better known as SQL) to manage its data.

This tutorial will go over how to install MySQL version `8.0` on an Ubuntu 20.04 server. By completing it, you will have a working relational database that you can use to build your next website or application.

Prerequisites

To follow this tutorial, you will need:

- One Ubuntu 20.04 server with a non-root administrative user and a firewall configured with UFW. To set this up, follow our initial server setup guide for Ubuntu 20.04.

Step 1 — Installing MySQL

On Ubuntu 20.04, you can install MySQL using the APT package repository. At the time of this writing, the version of MySQL available in the default Ubuntu repository is version 8.0.19.

To install it, update the package index on your server if you've not done so recently:

```
$ sudo apt update
```

Then install the `mysql-server` package:

```
$ sudo apt install mysql-server
```

This will install MySQL, but will not prompt you to set a password or make any other configuration changes. Because this leaves your installation of MySQL insecure, we will address this next.

Step 2 — Configuring MySQL

For fresh installations of MySQL, you'll want to run the DBMS's included security script. This script changes some of the less secure default options for things like remote root logins and sample users.

Run the security script with `sudo`:

SCROLL TO TOP

```
$ sudo mysql_secure_installation
```

This will take you through a series of prompts where you can make some changes to your MySQL installation’s security options. The first prompt will ask whether you’d like to set up the Validate Password Plugin, which can be used to test the password strength of new MySQL users before deeming them valid.

If you elect to set up the Validate Password Plugin, any MySQL user you create that authenticates with a password will be required to have a password that satisfies the policy you select. The strongest policy level — which you can select by entering `2` — will require passwords to be at least eight characters long and include a mix of uppercase, lowercase, numeric, and special characters:

```
Output
Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: Y

There are three levels of password validation policy:

LOW    Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary                  file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG:
 2
```

Regardless of whether you choose to set up the Validate Password Plugin, the next prompt will be to set a password for the MySQL **root** user. Enter and then confirm a secure password of your choice:

```
Output
Please set the password for root here.
```

SCROLL TO TOP

```
New password:

Re-enter new password:
```

Note that even though you've set a password for the **root** MySQL user, this user is not currently configured to authenticate with a password when connecting to the MySQL shell.

If you used the Validate Password Plugin, you'll receive feedback on the strength of your new password. Then the script will ask if you want to continue with the password you just entered or if you want to enter a new one. Assuming you're satisfied with the strength of the password you just entered, enter `Y` to continue the script:

```
Output
Estimated strength of the password: 100
Do you wish to continue with the password provided?(Press y|Y for Yes, any other key for No) : Y
```

From there, you can press `Y` and then `ENTER` to accept the defaults for all the subsequent questions. This will remove some anonymous users and the test database, disable remote root logins, and load these new rules so that MySQL immediately respects the changes you have made.

Once the script completes, your MySQL installation will be secured. You can now move on to creating a dedicated database user with the MySQL client.

Step 3 — Creating a Dedicated MySQL User and Granting Privileges

Upon installation, MySQL creates a **root** user account which you can use to manage your database. This user has full privileges over the MySQL server, meaning it has complete control over every database, table, user, and so on. Because of this, it's best to avoid using this account outside of administrative functions. This step outlines how to use the **root** MySQL user to create a new user account and grant it privileges.

In Ubuntu systems running MySQL `5.7` (and later versions), the **root** MySQL user is set to authenticate using the `auth_socket` plugin by default rather than with a password. This plugin requires that the name of the operating system user that invokes the MySQL client matches the name of the MySQL user specified in the command, so you must invoke `mysql` with `sudo` privileges to gain access to the **root** MySQL user:

SCROLL TO TOP

```
$ sudo mysql
```

Note: If you installed MySQL with another tutorial and enabled password authentication for **root**, you will need to use a different command to access the MySQL shell. The following will run your MySQL client with regular user privileges, and you will only gain administrator privileges within the database by authenticating:

```
$ mysql -u root -p
```

Once you have access to the MySQL prompt, you can create a new user with a `CREATE USER` statement. These follow this general syntax:

```
mysql> CREATE USER 'username'@'host' IDENTIFIED WITH authentication_plugin BY 'password';
```

After `CREATE USER`, you specify a username. This is immediately followed by an `@` sign and then the hostname from which this user will connect. If you only plan to access this user locally from your Ubuntu server, you can specify `localhost`. Wrapping both the username and host in single quotes isn't always necessary, but doing so can help to prevent errors.

You have several options when it comes to choosing your user's authentication plugin. The `auth_socket` plugin mentioned previously can be convenient, as it provides strong security without requiring valid users to enter a password to access the database. But it also prevents remote connections, which can complicate things when external programs need to interact with MySQL.

As an alternative, you can leave out the `WITH authentication plugin` portion of the syntax entirely to have the user authenticate with MySQL's default plugin, `caching_sha2_password`. The MySQL documentation recommends this plugin for users who want to log in with a password due to its strong security features.

Run the following command to create a user that authenticates with `caching_sha2_password`. Be sure to change `sammy` to your preferred username and `password` to a strong password of your choosing:

```
mysql> CREATE USER 'sammy'@'localhost' IDENTIFIED BY 'password';
```

SCROLL TO TOP

Note: There is a known issue with some versions of PHP that causes problems with `caching_sha2_password`. If you plan to use this database with a PHP application — phpMyAdmin, for example — you may want to create a user that will authenticate with the older, though still secure, `mysql_native_password` plugin instead:

```
mysql> CREATE USER 'sammy'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password';
```

If you aren't sure, you can always create a user that authenticates with `caching_sha2_plugin` and then `ALTER` it later on with this command:

```
mysql> ALTER USER 'sammy'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password';
```

After creating your new user, you can grant them the appropriate privileges. The general syntax for granting user privileges is as follows:

```
mysql> GRANT PRIVILEGE ON database.table TO 'username'@'host';
```

The `PRIVILEGE` value in this example syntax defines what actions the user is allowed to perform on the specified `database` and `table`. You can grant multiple privileges to the same user in one command by separating each with a comma. You can also grant a user privileges globally by entering asterisks (`*`) in place of the database and table names. In SQL, asterisks are special characters used to represent "all" databases or tables.

To illustrate, the following command grants a user global privileges to `CREATE`, `ALTER`, and `DROP` databases, tables, and users, as well as the power to `INSERT`, `UPDATE`, and `DELETE` data from any table on the server. It also grants the user the ability to query data with `SELECT`, create foreign keys with the `REFERENCES` keyword, and perform `FLUSH` operations with the `RELOAD` privilege. Lastly, it grants this user the `REPLICATION CLIENT` privileges which will allow it to perform some operations related to managing database replication. However, you should only grant users the permissions they need, so feel free to adjust your own user's privileges as necessary.

You can find the full list of available privileges in the official MySQL documentation.

Run this `GRANT` statement, replacing `sammy` with your own MySQL user's name, to grant these privileges to your user:

SCROLL TO TOP

```
mysql> GRANT CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT, REFERENCES, RELOAD, REPLICAT
```

Note that this statement also includes `WITH GRANT OPTION`. This will allow your MySQL user to grant any that it has to other users on the system.

Warning: Some users may want to grant their MySQL user the `ALL PRIVILEGES` privilege, which will provide them with broad superuser privileges akin to the **root** user's privileges, like so:

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'sammy'@'localhost' WITH GRANT OPTION;
```

Such broad privileges **should not be granted lightly**, as anyone with access to this MySQL user will have complete control over every database on the server.

Following this, it's good practice to run the `FLUSH PRIVILEGES` command. This will free up any memory that the server cached as a result of the preceding `CREATE USER` and `GRANT` statements:

```
mysql> FLUSH PRIVILEGES;
```

Then you can exit the MySQL client:

```
mysql> exit
```

In the future, to log in as your new MySQL user, you'd use a command like the following:

```
$ mysql -u sammy -p
```

The `-p` flag will cause the MySQL client to prompt you for your MySQL user's password in order to authenticate.

Finally, let's test the MySQL installation.

Step 4 — Testing MySQL

Regardless of how you installed it, N SCROLL TO TOP e started running automatically. To test this, check its status.

```
$ systemctl status mysql.service
```

You'll see output similar to the following:

```
Output
● mysql.service - MySQL Community Server
     Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2020-04-21 12:56:48 UTC; 6min ago
   Main PID: 10382 (mysqld)
     Status: "Server is operational"
      Tasks: 39 (limit: 1137)
     Memory: 370.0M
     CGroup: /system.slice/mysql.service
             └─10382 /usr/sbin/mysqld
```

If MySQL isn't running, you can start it with `sudo systemctl start mysql`.

For an additional check, you can try connecting to the database using the `mysqladmin` tool, which is a client that lets you run administrative commands. For example, this command says to connect as a MySQL user named **sammy** (`-u sammy`), prompt for a password (`-p`), and return the version. Be sure to change `sammy` to the name of your dedicated MySQL user, and enter that user's password when prompted:

```
$ sudo mysqladmin -p -u sammy version
```

You should see output similar to this:

```
Output
mysqladmin  Ver 8.0.19-0ubuntu5 for Linux on x86_64 ((Ubuntu))
Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Server version      8.0.19-0ubuntu5
Protocol version    10
Connection      Localhost via UNIX socket
UNIX socket     /var/run/mysqld/mysqld.sock
Uptime:         10 min 44 sec
```

SCROLL TO TOP

```
Threads: 2  Questions: 25  Slow queries: 0  Opens: 149  Flush tables: 3  Open tables: 69  Queries pe
```

This means MySQL is up and running.

Conclusion

You now have a basic MySQL setup installed on your server. Here are a few examples of next steps you can take:

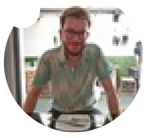
- Set up a LAMP stack or a LEMP stack
- Practice running queries with SQL
- Manage your MySQL installation with phpMyAdmin

Was this helpful? Yes No

4

Report an issue

About the authors



Mark Drake

Technical Writer @ DigitalOcean

Still looking for an answer?

Ask a question

Search for more help

SCROLL TO TOP

RELATED

Now Available: Managed MySQL Databases
Product

How To Use Telepresence on Kubernetes for Rapid Development on Ubuntu 20.04
Tutorial

How To Deploy a Static HTML Website with Ansible on Ubuntu 20.04 (Nginx)
Tutorial

Comments

4 Comments

Leave a comment...

Sign In to Comment

1 **vlada972010** May 1, 2020

Hello,

How I can install MySQL 5? This is very important to me for using Magento 1 version.

Thank you.

Reply Report

0 **mdessaintes** September 1, 2020

I have this error at mysql*secure*inst

SCROLL TO TOP

Re-enter new password:

... Failed! Error: Password hash should be a 41-digit hexadecimal number

Reply Report

0 lun91yong December 25, 2020

For those who wonder why this isn't working, it is because `sudo apt install mysql-server` install latest version of mysql, which is mysql 8 or latest. mysql 5.7 installation need to refers to other tutorial

Reply Report

0 mohadesehthm January 19, 2021

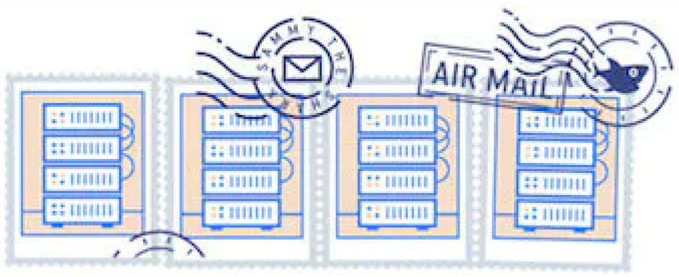
hi

I followed your tutorial on ubuntu 16.4 and it seems to be working all fine, now I have another question, my software instruction asks: the table engine must be “MyISAM”. With new MySQL versionit's always InnoDB. Set “default-storage-engine” option to “MyISAM” in “/etc/mysql/my.cnf”

can you guide me on how to do that?

Thanks

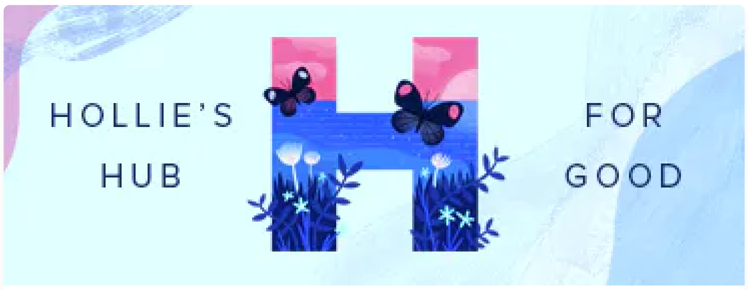
Reply Report



SCROLL TO TOP

GET OU ETTER

Sign up for Infrastructure as a Newsletter.



HOLLIE'S HUB FOR GOOD

Working on improving health and education, reducing inequality, and spurring economic growth? We'd like to help.

BECOME A CONTRIBUTOR

You get paid; we donate to tech nonprofits.

Featured on Community Kubernetes Course Learn Python 3 Machine Learning in Python Getting started with Go Intro to Kubernetes

DigitalOcean Products Virtual Machines SCROLL TO TOP Managed Kubernetes Block Storage Object Storage Marketplace VPC Load Balancers

Welcome to the developer cloud

DigitalOcean makes it simple to launch in the cloud and scale up as you grow – whether you're running one virtual machine or ten thousand.

Learn More

Company

About
Leadership
Blog
Careers
Partners
Referral Program
Press
Legal
Security & Trust Center

Products	Community	Contact
Pricing	Tutorials	Get Support
Products Overview	Q&A	Trouble Signing In?
Droplets	Tools and Integrations	Sales
Kubernetes	Tags	Report Abuse
Managed Databases	SCROLL TO TOP	System Status
Spaces	Write for DigitalOcean	