- Blog
- Docs
- Get Support
- Contact Sales

- Products

  Featured Products

  - DropletsScalable virtual machines
  - KubernetesManaged Kubernetes clusters
  - PaperspaceBuild & scale AI models
  - CloudwaysManaged cloud hosting
  - App PlatformGet apps to market faster
  - DatabasesWorry-free setup & maintenance

  Compute

  DropletsCPU-Optimized DropletsKubernetesApp PlatformFunctionsGPUs

  Backups & Snapshots

  BackupsSnapshotsSnapShooterCloudways Autonomous

  Storage

  Spaces Object StorageVolumes Block Storage

  Networking

  Virtual Private Cloud (VPC)Cloud FirewallsLoad BalancersDNSDDoS Protection

  Managed Databases

  MongoDBKafkaMySQLPostgreSQLManaged Databases for Redis®

  Developer Tools

  APICLISupport PlansMonitoringUptime

  AI/ML

  Paperspace

  Cloud Website Hosting

  Cloudways
  See all products
  See what's new at DigitalOcean
- Solutions

  Cloudways

    - Managed WordPressMake your website faster and more secure.
    - Managed WoocommerceHassle-free ecommerce website hosting.
    - Managed MagentoSimple, efficient ecommerce hosting.

  By industry

  Data AnalyticsDigital Marketing AgenciesEcommerceAdvertising and Marketing TechIT ConsultingDeveloper ToolsCI/CD PipelinesView All Industries

  By use case

  Cloud VPNWeb & Mobile AppsGame DevelopmentVideo StreamingVirtual MachinesBlockchainWebsite HostingVPS HostingView All Use Cases

  Questions?

  Speak With an Expert

  See all solutions

- Developers

  Our community

    - Developer CenterLearn, build, and scale.
    - Community HomeDevOps and development guides.
    - CSS-TricksAll things web design.

  Resources

  TutorialsQuestions And AnswersMarketplaceToolsWrite for DOnationsCustomer StoriesDigitalOcean Blog

  Get Involved

  Hatch Startup ProgramOpen Source SponsorshipsHacktoberfestDO ImpactNonprofits

  Documentation

  Quick StartDropletsStorageApp PlatformAPI ReferenceDomains and DNS

- Businesses

  Grow your business

    - The WaveContent to level up your business.
    - Find a partnerWork with a partner to get up and running in the cloud.
    - Become a partnerJoin our Partner Pod to connect with SMBs and startups like yours
    - UGURUSElite training for agencies & freelancers.

  Resources

  Customer StoriesHatch Startup ProgramPrice Estimate CalculatorOne-Click AppsSecurityMigrate Workloads to DigitalOcean

  Featured articles

  Cloud cost optimization best practicesRead more
  How to choose a cloud providerRead more

CONTENTS

Tutorial

How To Create a New User and Grant Permissions in MySQL

Updated on March 18, 2022

- MySQL

By Etel Sverdlov

English

Introduction

MySQL is an open-source relational database management system. It is commonly deployed as part of the *LAMP stack* (which stands for **L**inux, **A**pache, **M**ySQL, and **P**HP) and, as of this writing, is the most popular open-source database in the world.

This guide outlines how to create a new MySQL user and grant them the permissions needed to perform a variety of actions.

Prerequisites

In order to follow along with this guide, you'll need access to a MySQL database. This guide assumes that this database is installed on a virtual private server running Ubuntu 20.04, though the principles it outlines should be applicable regardless of how you access your database.

If you don't have access to a MySQL database and would like to set one up yourself, you can follow one of our guides on How To Install MySQL. Again, regardless of your server's underlying operating system, the methods for creating a new MySQL user and granting them permissions will generally be the same.

You could alternatively spin up a MySQL database managed by a cloud provider. For details on how to spin up a DigitalOcean Managed Database, see our product documentation.

Please note that any portions of example commands that you need to change or customize will be highlighted `like this` throughout this guide.

Creating a New User

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. This means that you need to precede the `mysql` command with `sudo` to invoke it with the privileges of the **root** Ubuntu user in order to gain access to the **root** MySQL user:

```
sudo mysql
```

Copy

Note: If your **root** MySQL user is configured to authenticate with a password, 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 with the correct password:

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

Copy

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

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

Copy

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:

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

Copy

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:

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

Copy

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:

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

Copy

After creating your new user, you can grant them the appropriate privileges.

Granting a User Permissions

The general syntax for granting user privileges is as follows:

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

Copy

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. 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:

```
GRANT CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT, REFERENCES, RELOAD on *.*
TO 'sammy'@'localhost' WITH GRANT OPTION;
```

Copy

Note that this statement also includes `WITH GRANT OPTION`. This will allow your MySQL user to grant any permissions 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:

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

Copy

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.

Many guides suggest running the `FLUSH PRIVILEGES` command immediately after a `CREATE USER` or `GRANT` statement in order to reload the grant tables to ensure that the new privileges are put into effect:

```
FLUSH PRIVILEGES;
```

Copy

However, according to the official MySQL documentation, when you modify the grant tables indirectly with an account management statement like `GRANT`, the database will reload the grant tables immediately into memory, meaning that the `FLUSH PRIVILEGES` command isn’t necessary in our case. On the other hand, running it won’t have any negative effect on the system.

If you need to revoke a permission, the structure is almost identical to granting it:

```
REVOKE type_of_permission ON database_name.table_name FROM 'username'@'host';
```

Copy

Note that when revoking permissions, the syntax requires that you use `FROM`, instead of `TO` which you used when granting the permissions.

You can review a user’s current permissions by running the `SHOW GRANTS` command:

```
SHOW GRANTS FOR 'username'@'host';
```

Copy

Just as you can delete databases with `DROP`, you can use `DROP` to delete a user:

```
DROP USER 'username'@'localhost';
```

Copy

After creating your MySQL user and granting them privileges, you can exit the MySQL client:

```
exit
```

Copy

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

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

Copy

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

Conclusion

By following this tutorial, you’ve learned how to add new users and grant them a variety of permissions in a MySQL database. From here, you could continue to explore and experiment with different permissions settings for your MySQL user, or you may want to learn more about some higher-level MySQL configurations.

For more information about the basics of MySQL, you can check out the following tutorials:

- How To Create and Manage Databases in MySQL and MariaDB on a Cloud Server
- How To Set Up Replication in MySQL
- How To Configure MySQL Group Replication on Ubuntu 20.04

About the authors

[Etel Sverdlov]

author

Still looking for an answer?

[Ask a question][Search for more help]

Was this helpful?
YesNo

Comments
10 Comments

Leave a comment...
This textbox defaults to using **Markdown** to format your answer.

You can type **!ref** in this text area to quickly search our full set of tutorials, documentation & marketplace offerings and insert the link!

[Sign In or Sign Up to Comment]
[petreoctavian] • [December 12, 2013]

Hi,

I have done all the above but I can't seem to log with the new user with:

mysql -u [username]-p

I have the error:

ERROR 1045 (28000): Access denied for user 'prosper202'@'localhost' (using password: YES)

What may be the problem?

- Reply

[ejkuhn] • [September 26, 2016]

Thanks for the great tutorial. One comment: apparently FLUSH PRIVILEGES is unnecessary in most cases:

[http://dbahire.com/stop-using-flush-privileges/]

- Reply

asad • June 26, 2014

You guys have an article for everything…so psycho its awesome!

- Reply

Etel Sverdlov
DigitalOcean Employee

• March 6, 2013

Updated

- Reply

chris96043 • February 13, 2013

You can grant multiple privileges in one command by separating them with commas: eg: “GRANT UPDATE, SELECT ON [database name].[table name] TO ‘[username]’@‘localhost’;”

- Reply

Renata Cardoso • December 8, 2023

Thanks for the great tutorial!

- Reply

humayun • November 1, 2023

Thanks

It saves my day

Though granting individual privilege didn’t work for me, so I used `GRANT ALL` instead.

- Reply

williambradley • October 18, 2023

Make sure to grant only the minimum required privileges for security best practices.

- Reply

Emmanuel Katto • August 30, 2023

Thank you for sharing your knowledge with us!

- Reply

lajmonline su • July 8, 2023

i don’t know anything know anything

- Show replies
- Reply

Load More Comments

Popular Topics

- Ubuntu
- Linux Basics
- JavaScript
- Python
- MySQL
- Docker
- Kubernetes
- All tutorials
- Talk to an expert

Join the Tech Talk

Success! Thank you! Please check your email for further details.

Please complete your information!

Sign up

Congratulations on unlocking the whale ambience easter egg! Click the whale button in the bottom left of your screen to toggle some ambient whale noises while you read.

Thank you to the Glacier Bay National Park & Preserve and Merrick079 for the sounds behind this easter egg.

Interested in whales, protecting them, and their connection to helping prevent climate change? We recommend checking out the Whale and Dolphin Conservation.

Reset easter egg to be discovered again / Permanently dismiss and hide easter egg

Get our biweekly newsletter

Sign up for Infrastructure as a Newsletter.

- DO Impact
- Nonprofits

Products

- Products Overview
- Droplets
- Kubernetes
- Paperspace
- App Platform
- Functions
- Cloudways
- Managed Databases
- Spaces
- Marketplace
- Load Balancers
- Block Storage
- Tools & Integrations
- API
- Pricing
- Documentation
- Release Notes
- Uptime

Community

- Tutorials
- Q&A
- CSS-Tricks
- Write for DOnations
- Currents Research
- Hatch Startup Program
- deploy by DigitalOcean
- Shop Swag
- Research Program
- Open Source
- Code of Conduct
- Newsletter Signup
- Meetups

Solutions

- Website Hosting
- VPS Hosting
- Web & Mobile Apps
- Game Development
- Streaming
- VPN
- SaaS Platforms
- Cloud Hosting for Blockchain
- Startup Resources

Contact

- Support
- Sales
- Report Abuse
- System Status
- Share your ideas

Company

- About
- Leadership
- Blog
- Careers
- Customers
- Partners
- Referral Program
- Affiliate Program
- Press
- Legal
- Privacy Policy
- Security
- Investor Relations
- DO Impact
- Nonprofits

Products

- Products Overview
- Droplets
- Kubernetes
- Paperspace
- App Platform
- Functions
- Cloudways
- Managed Databases
- Spaces
- Marketplace
- Load Balancers
- Block Storage
- Tools & Integrations
- API
- Pricing
- Documentation
- Release Notes
- Uptime

Community

- Tutorials
- Q&A
- CSS-Tricks
- Write for DOnations
- Currents Research
- Hatch Startup Program
- deploy by DigitalOcean
- Shop Swag
- Research Program
- Open Source
- Code of Conduct
- Newsletter Signup
- Meetups

Solutions

- Website Hosting
- VPS Hosting
- Web & Mobile Apps

- Game Development
- Streaming
- VPN
- SaaS Platforms
- Cloud Hosting for Blockchain
- Startup Resources

Contact

- Support
- Sales
- Report Abuse
- System Status
- Share your ideas

Loading...

Community

Product Docs

Marketplace

DigitalOcean Blog

navigate go exit