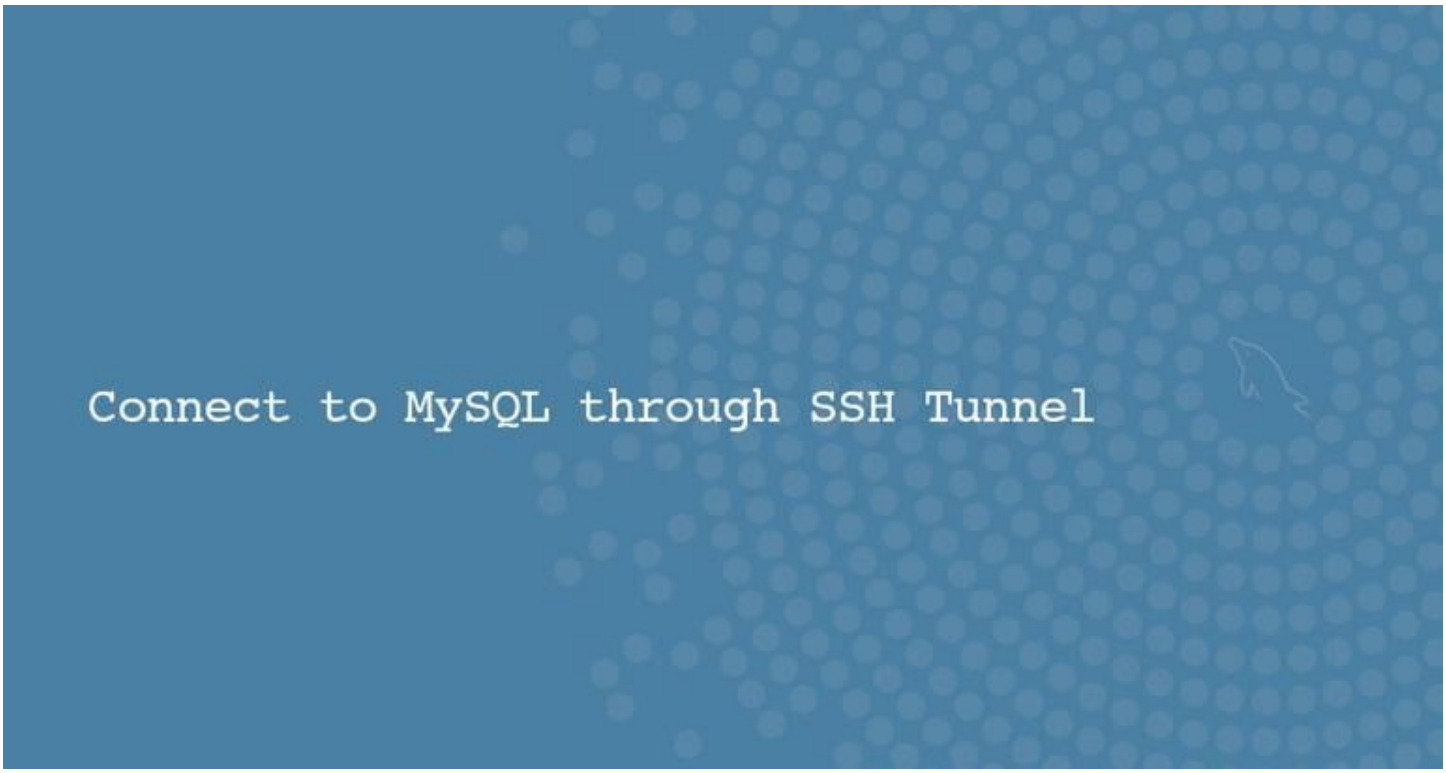
# How to Connect to MySQL through SSH Tunnel

Posted Sep 23, 2019 • 4 min read




By default, the MySQL server listens only on localhost, which means it can be accessed only by applications running on the same host.

However, in some situations, you might want to connect to the server from remote locations. One option would be to configure the MySQL server to allow remote connections , but that requires administrative privileges, and it may cause security risks.

A more secure alternative would be to create an SSH tunnel from the local system to the server. SSH tunneling is a method of creating an encrypted SSH connection between a client and a server machine through which services ports can be relayed.

In this guide, we will explain how to create an SSH tunnel and connect to the MySQL server from remote clients. The same instructions apply for MariaDB.

- SSH Client.
- SSH access to the system on which the MySQL server runs.

## Create an SSH Tunnel on Linux and macOS

The `ssh` client is preinstalled on most Linux and Unix-based systems.

If you are using Linux or macOS as your operating system, you can create an SSH tunnel using the following command:

```
$ ssh -N -L 3336:127.0.0.1:3306 [USER]@[SERVER_IP]
```

The options used are as follows:

- `-N` - Tells SSH not to execute a remote command.
- `-L 3336:127.0.0.1:3306` - Creates a local port forwarding. The local port ( `3306` ), the destination IP ( `127.0.0.1` ) and the remote port ( `3306` ) are separated with a colon ( `:` ).
- `[USER]@[SERVER_IP]` - The remote SSH user and server IP address.
- To run the command in the background, use the `-f` option.
- If the SSH server is listening on a port other than 22 (the default) specify the port with the `-p [PORT_NUMBER]` option.

Once you run the command, you'll be prompted to enter your SSH user password. After entering it, you will be logged in to the server, and the SSH tunnel will be established. It is a good idea to set up an SSH key-based authentication and connect to the server without entering a password.

database login credentials and access the MySQL server.

For example, to connect to the MySQL server using the command line `mysql` client you would issue:

```
$ mysql -u MYSQL_USER -p -h 127.0.0.1
```

Where `MYSQL_USER` is the remote MySQL user having privileges to access the database.

When prompted, enter the MySQL user password.

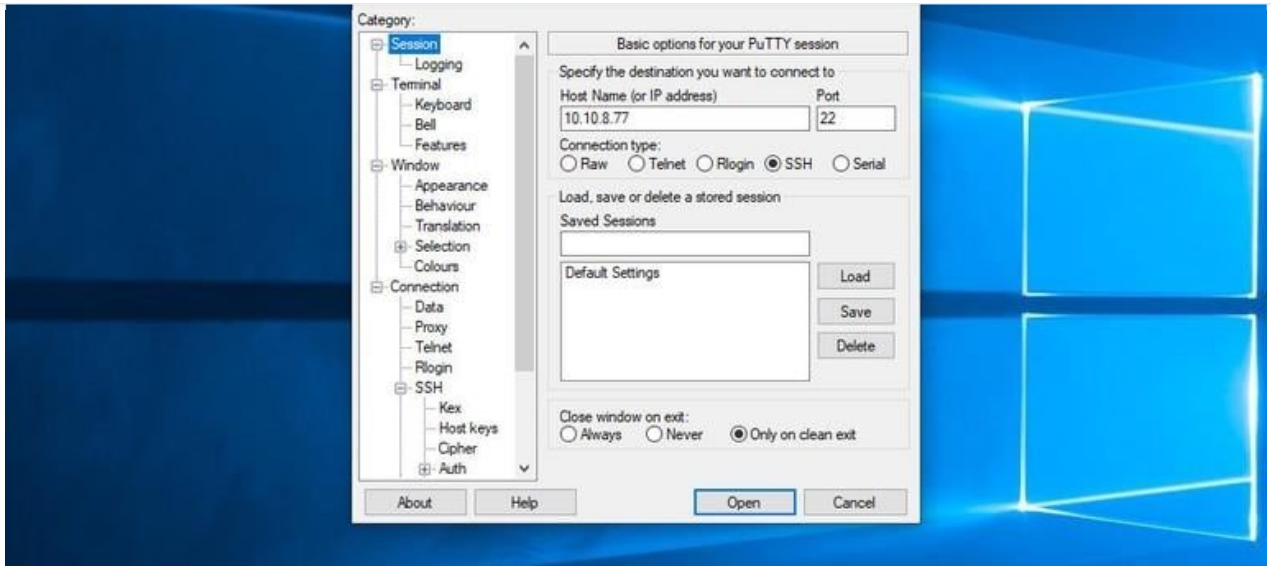
To terminate the SSH tunnel type `CTRL+C` in the console where the ssh client is running.

## Create an SSH Tunnel on Windows

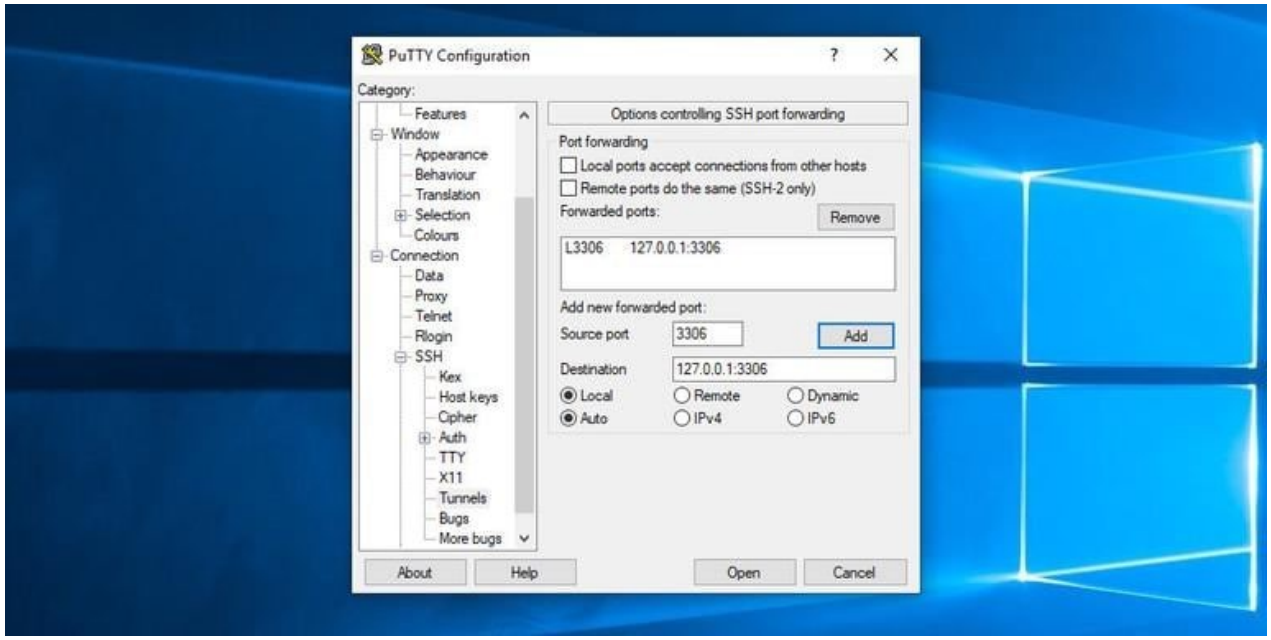
Windows users will first need to download and install an SSH client program. The most popular Windows SSH client is PuTTY. You can download PuTTY here .

Perform the following steps to create an SSH tunnel to the MySQL server with PuTTY:

01. Launch Putty and enter the IP Address of the server in the `Host name (or IP address)` field:



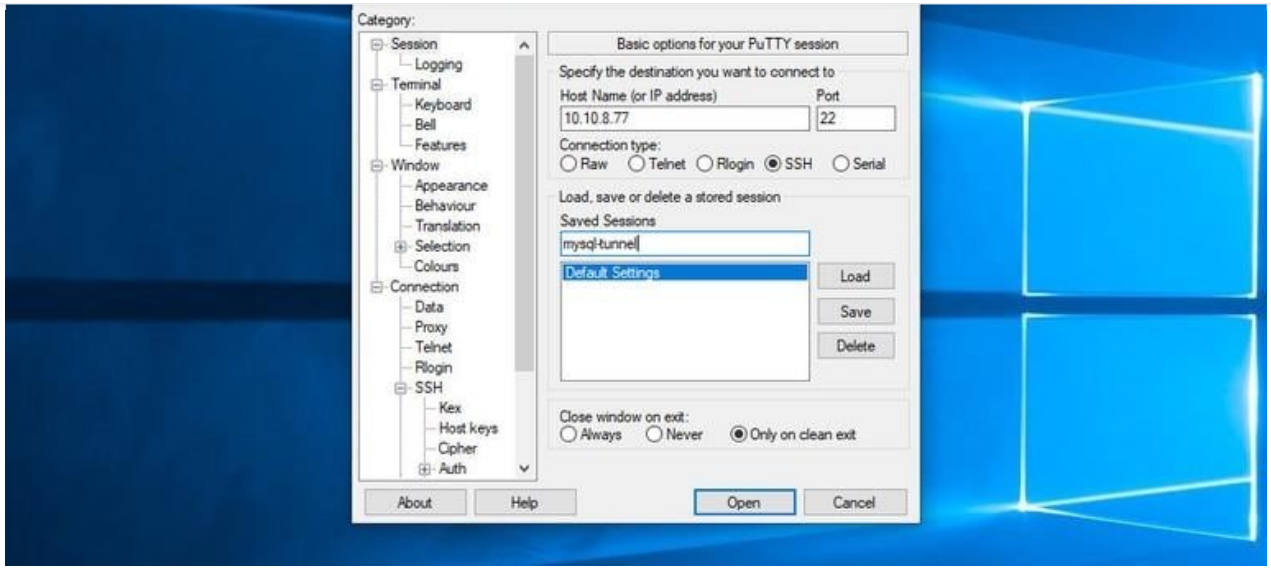

02. Under the `Connection` menu, expand `SSH` and select `Tunnels`. Enter `3306` in the `Source Port` field, and `127.0.0.1:3306` in the `Destination` field:



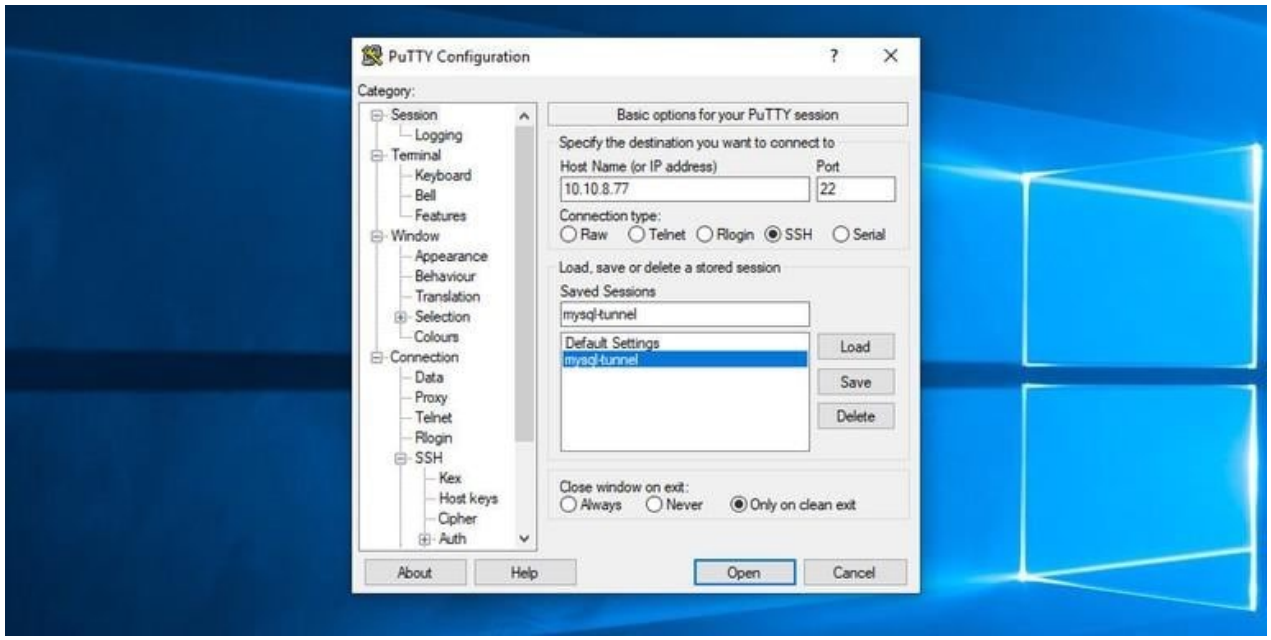

Click on the `Add` button to add the tunnel.

03. Go back to the `Session` page to save the settings so that you do not need to enter them again.

Enter the session name in the `Saved Session` field and click on the `Save` button.




04. Select the saved session and log in to the remote server by clicking on the `Open` button.

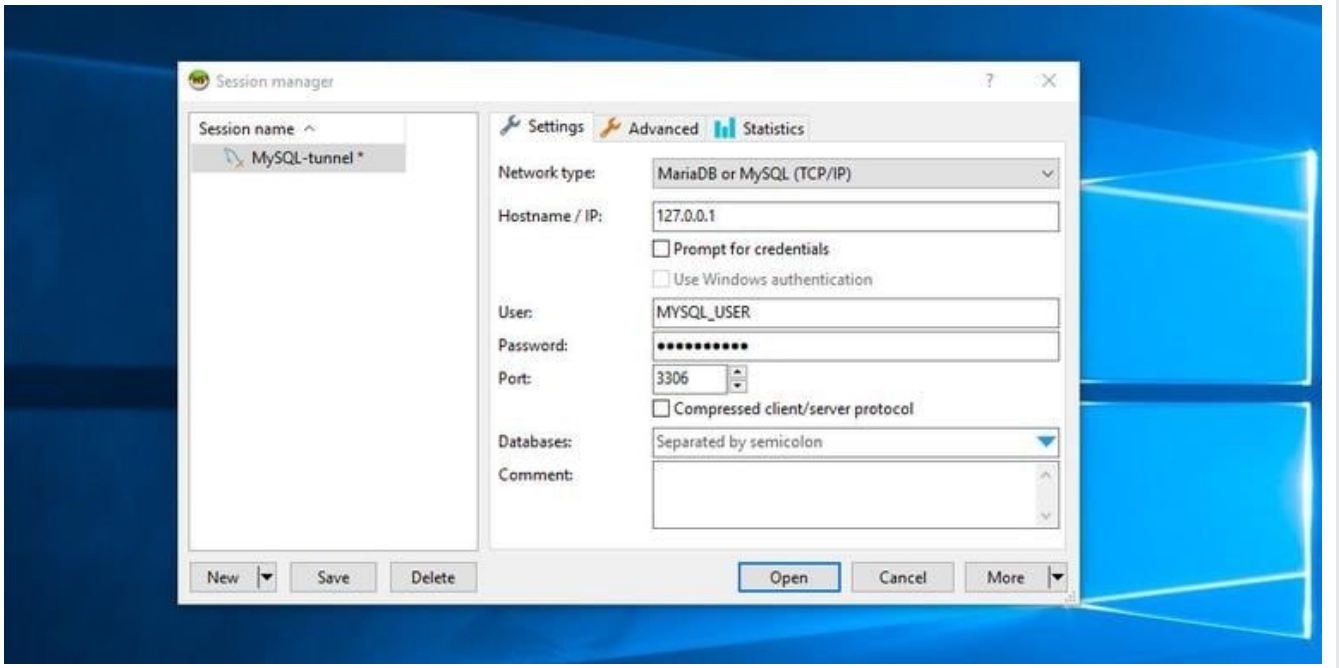



A new window asking for your username and password will show up. Once you enter the username and password, you will be logged in to the server, and the SSH tunnel will be created.

Setting up public key authentication will allow you to connect to the server without entering a password.

You can now connect to the remote database using your local MySQL client.

For example, if you are using HeidiSQL enter `127.0.0.1` in the `Hostname / IP` field and the MySQL user and password in the `User` and `Password` fields:




## Conclusion

MySQL, the most popular open-source database server, listens for incoming connections only on localhost. Creating an SSH tunnel allows you to securely connect to the remote MySQL server from your local client.

mysql mariadb ssh

If you like our content, please consider buying us a coffee.
Thank you for your support!

BUY ME A COFFEE

Sign up to our newsletter and get our latest tutorials and news straight to your mailbox.

Your email... Subscribe

We'll never share your email address or spam you.

## Related Articles


OCT 10, 2019

List (Show) Tables in a MySQL Database


SEP 27, 2019

How to Configure MySQL (MariaDB) Master-Slave Replication on Debian 10


Show comments (3)