How to Install phpMyAdmin With Nginx on Ubuntu?

written by sysadmin | 27 December 2025

The previous article has explained how to install the phpMyAdmin application on Linux using the Apache web server. This article explains how to configure phpMyAdmin using nginx on Ubuntu.

Problem

How to install phpMyAdmin with nginx on Ubuntu?

Solution

Follow the steps below to install phpMyAdmin with Nginx on Ubuntu:

1. Update repo

Use the command below to update the Ubuntu repo:

```
sudo apt update -y
```

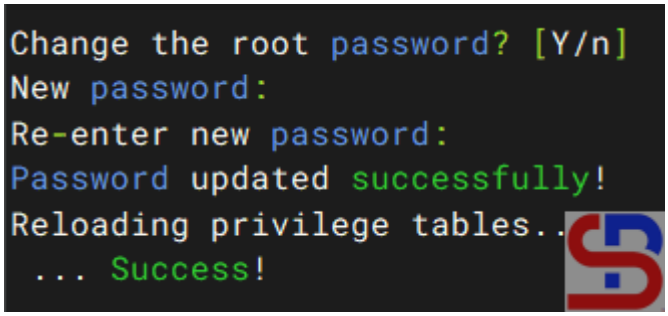
2. Install MariaDB

Next, install the MariaDB database using the command:

```
sudo apt install mariadb-server mariadb-client -y
```

Once finished, use the command below to change the root password in MariaDB:

```
sudo mysql_secure_installation
```



Change the root password

Then check whether the database is up or not using the command below:

```
sudo systemctl status mariadb
```

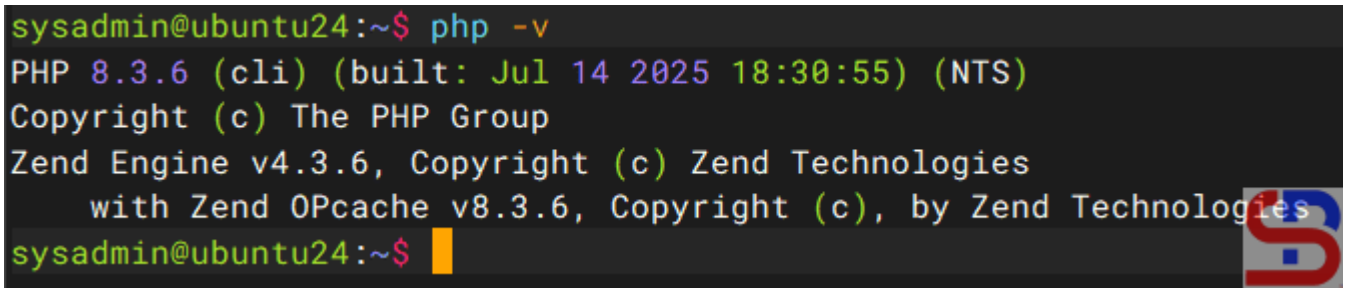
3. Install PHP

Install PHP by using the command below:

```
sudo apt install php php-fpm php-mysql php-cli php-curl php-gd php-mbstring php-xml php-zip -y
```

Then check the version of PHP that you just installed by using the command below:

```
php -v
```



Check the php version

Usually, when installing PHP, the Apache package will also be installed on the server. Therefore, delete Apache using the command:

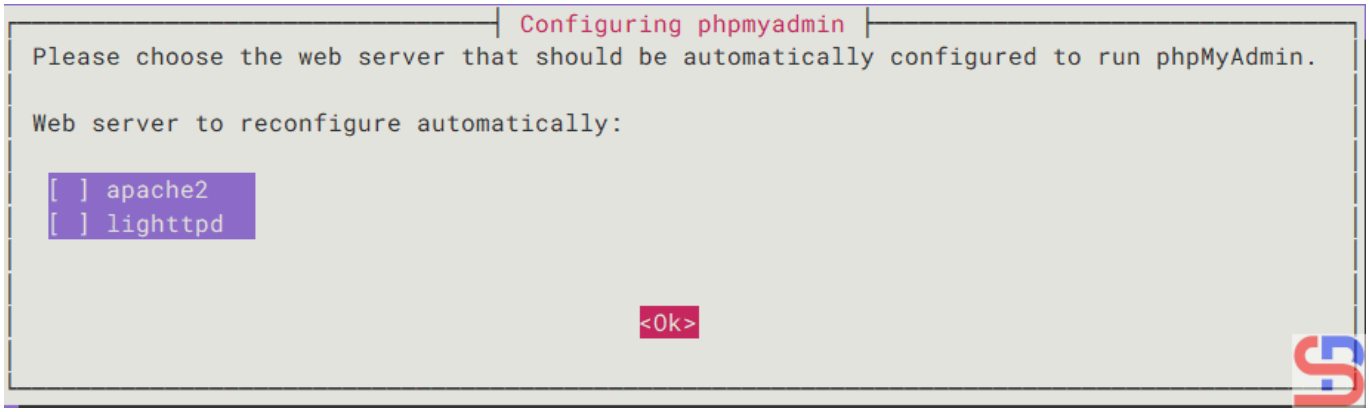
```
sudo apt remove apache2-* -y
```

4. Install phpMyAdmin

Use the command below to install phpMyAdmin:

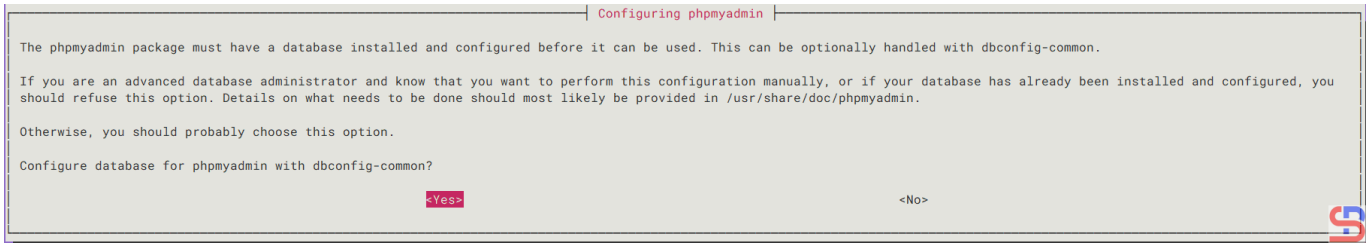
```
sudo apt install phpmyadmin -y
```

At the time of installation, there are several pop-ups that you must answer, such as the selection of the web server you are using, as shown in the image below:



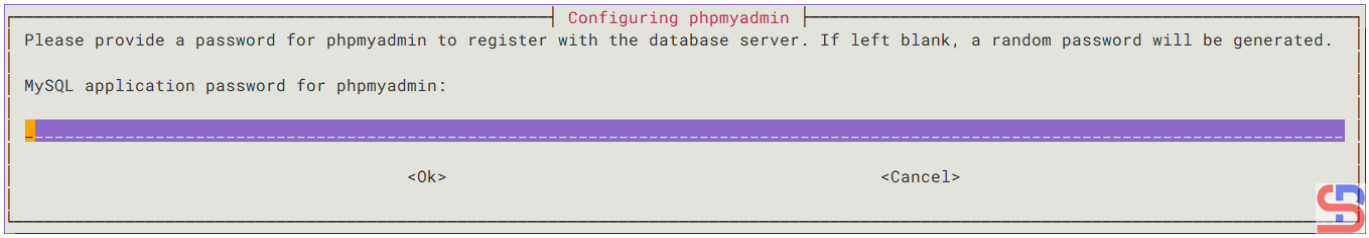
Choose the Ok button

Just select the button **Ok**, then the process of installation will continue. A few seconds later, there will be a pop-up as below to insert the phpMyAdmin in the database:



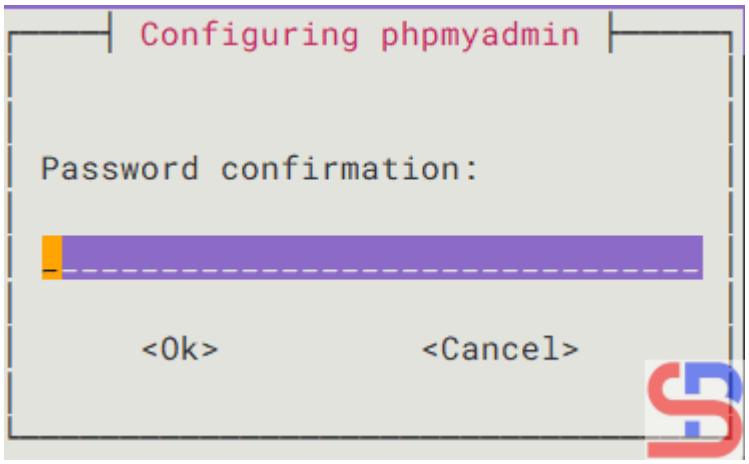
Choose the Yes button

Select the button **Yes**, and there’s a pop-up to enter the password for the user phpmyadmin in the database as in the picture below:



Enter the password for the phpmyadmin user

Enter the password you want, select the **OK** button, and there will be another pop-up to confirm the password as in the image below:



Password confirmation

Enter the same password and select the **Ok** button, then the phpMyAdmin installation process will continue until completion.

5. Install nginx

Install nginx by using the command below:

```
sudo apt install nginx -y
```

After that, configure Nginx so that it can be integrated with phpMyAdmin. Copy the default file using the command below:

```
sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-available/default.ori
```

Then in the default file, copy the script below:

```
server {
    listen 80;
    server_name _;
    root /var/www/html;
    index index.php index.html;

    location / {
        try_files $uri $uri/ =404;
    }

    location /phpmyadmin {
        root /usr/share/;
        index index.php;

        location ~ ^/phpmyadmin/(.+\.php)$ {
            try_files $uri =404;
            root /usr/share/;
            fastcgi_pass unix:/run/php/php8.3-fpm.sock;
            fastcgi_index index.php;
            fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
            include fastcgi_params;
        }

        location ~*
^/phpmyadmin/(.+\.(css|js|jpg|jpeg|gif|png|ico|html|xml|txt))$ {
            root /usr/share/;
        }
    }
}
```

Warning

You have to be careful when writing the php-fpm version in the **fastcgi_pass_unix** section because there will be an error if the version is different from the one installed on the server. To see the version installed on the server, use the command below:

```
ls -l /run/php/
```

After that, use the command below to check whether the nginx configuration has errors or not:

```
sudo nginx -t
```

If there are no errors, then you can run the command below to reload nginx:

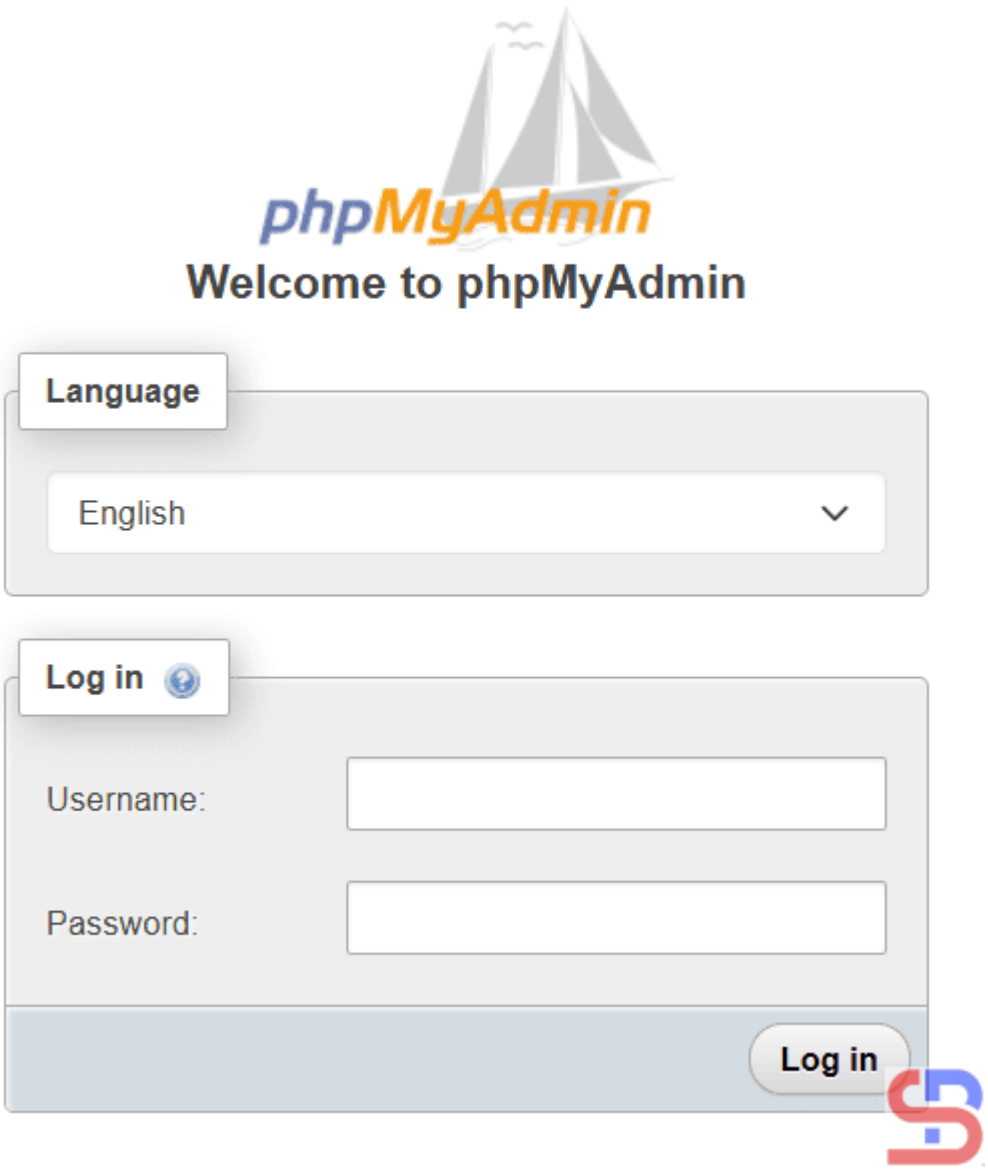
```
sudo systemctl reload nginx
```

6. Open phpMyAdmin

Open your browser and type:

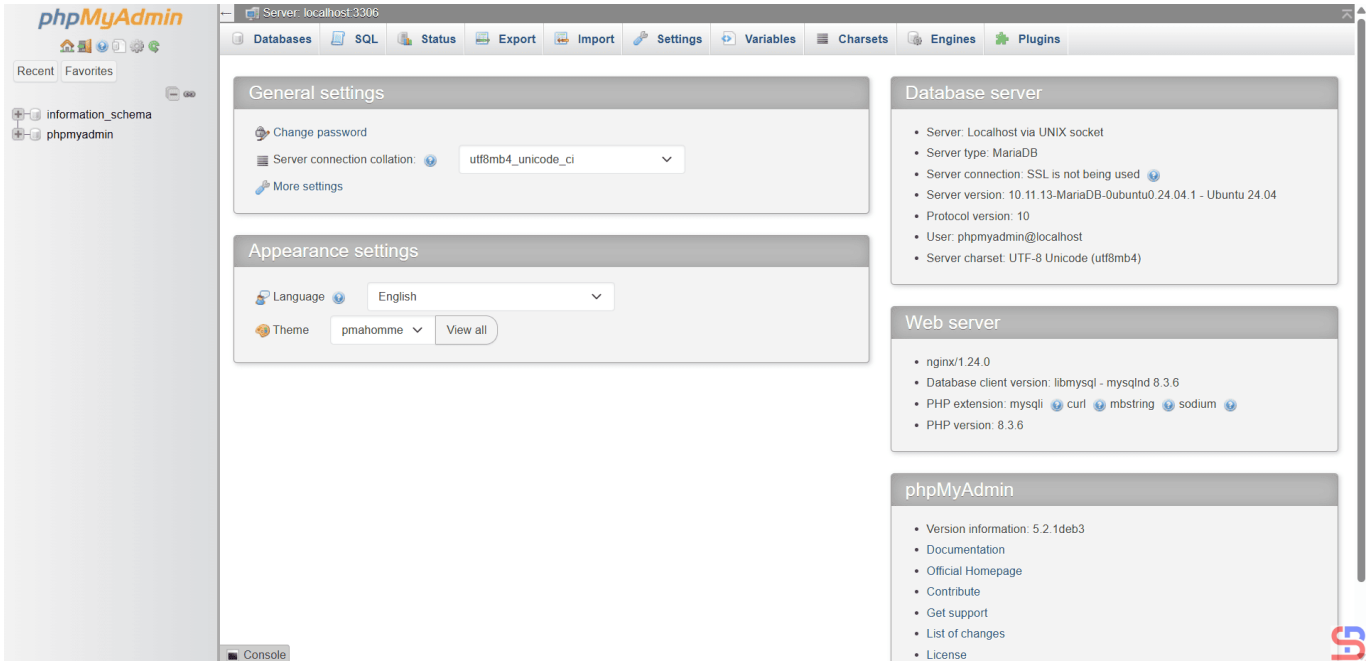
```
http://ip_server/phpmyadmin
```

Then there should be a display like below:



Open phpMyAdmin in the browser

Enter the database username and password, for example, using the user phpmyadmin with the password that you created when installing phpMyAdmin, then there will be a display like the one below:



Display of phpMyAdmin using phpmyadmin user

Note

If you want the phpMyAdmin application to be more secure, you can look at this page.

References

markaicode.com
linuxbabe.com
hostman.com

How to Display MariaDB Database Metric Values?

written by sysadmin | 27 December 2025

I want to see the metric values □□of my MariaDB database.

Problem

How to display MariaDB database metric values?

Solution

As far as I know, there are 2 tools that you can use to display the values of metrics on your MariaDB database.

1. mariadb-report

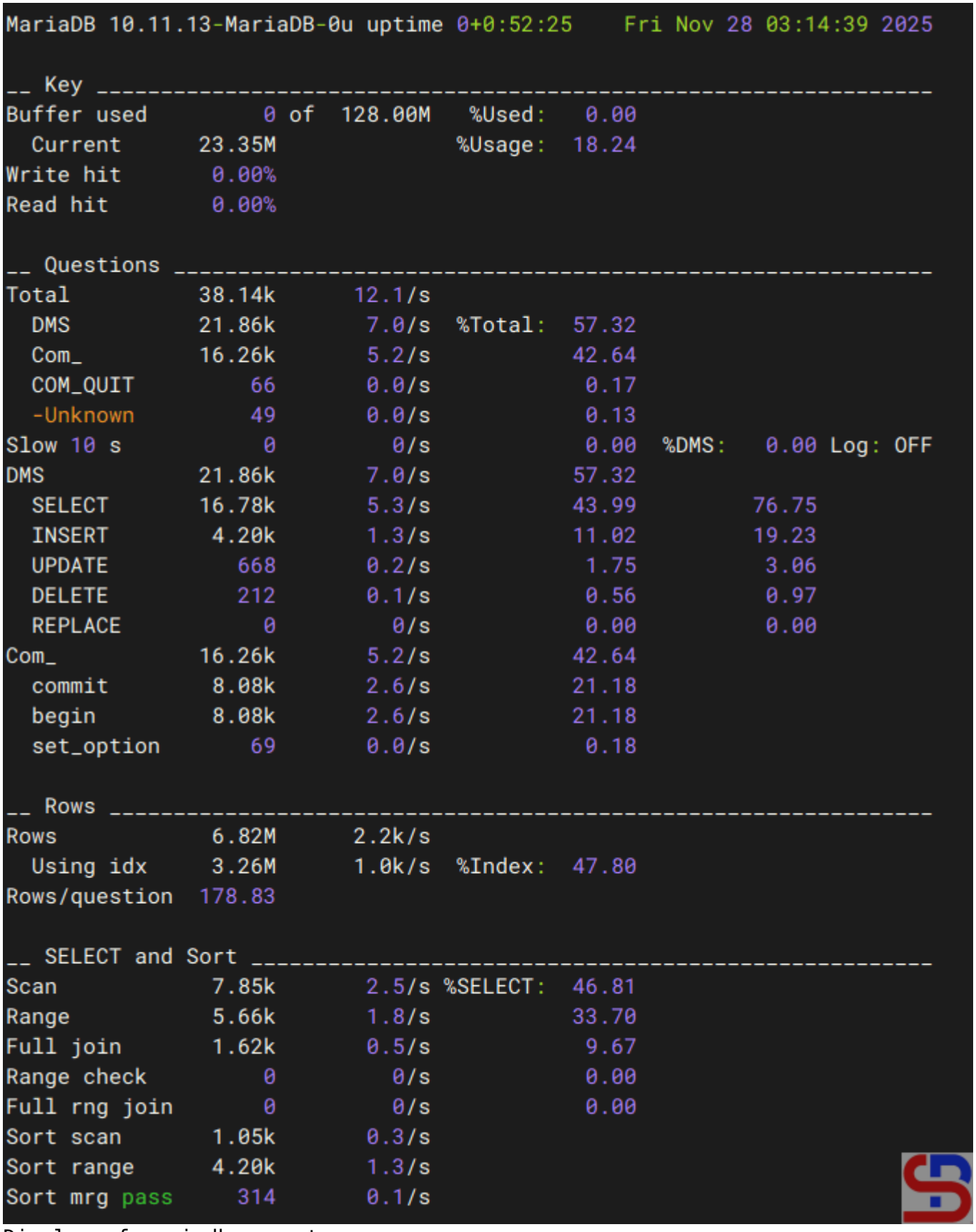
This tool is built into MariaDB, so when you install the MariaDB database or mariadb-client on your server, this tool will automatically be installed on your server. In general, use the format below to run this command:

```
mariadb-report --user=username --password your_password
```

If you want to display MariaDB database metrics using the root user, then use the command below:

```
mariadb-report --user=root --password
```

Enter the password for the root user, and a display will appear as below:



Display of mariadb-report

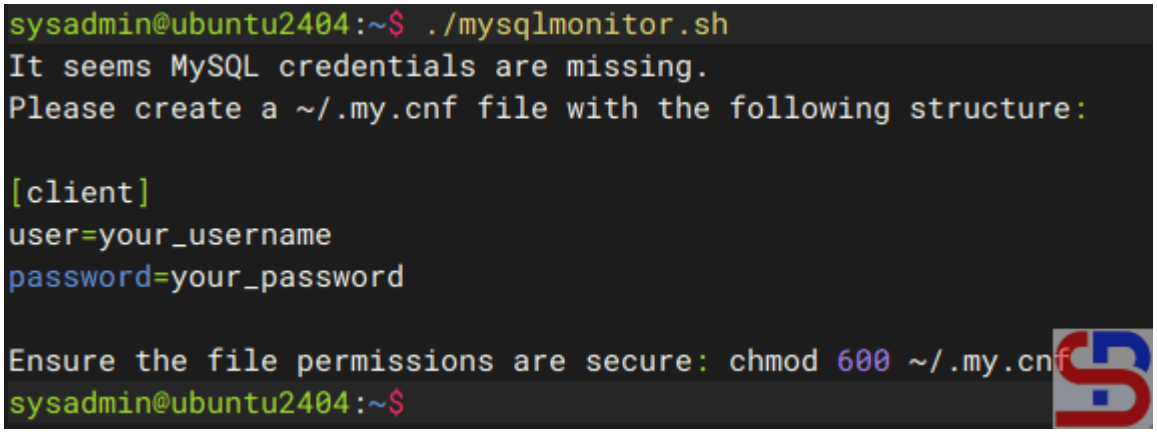
If you want to put the results into a file, you can use the command below:

```
mariadb-report --user=root --password --outfile /tmp/metric.txt
```

Enter the password, and the results will be entered into the /tmp/metric.txt file.

2. mysqlmonitor

This is a simple bash script created to give system administrators and database administrators a fast summary of MySQL metrics. It presents essential metrics such as InnoDB buffer utilization, query efficiency, and system memory, accompanied by concise descriptions of each parameter. To run this script, you must create MariaDB credentials in the my.cnf file, because if not, there will be an error like the one below:



Error when running mysqlmonitor

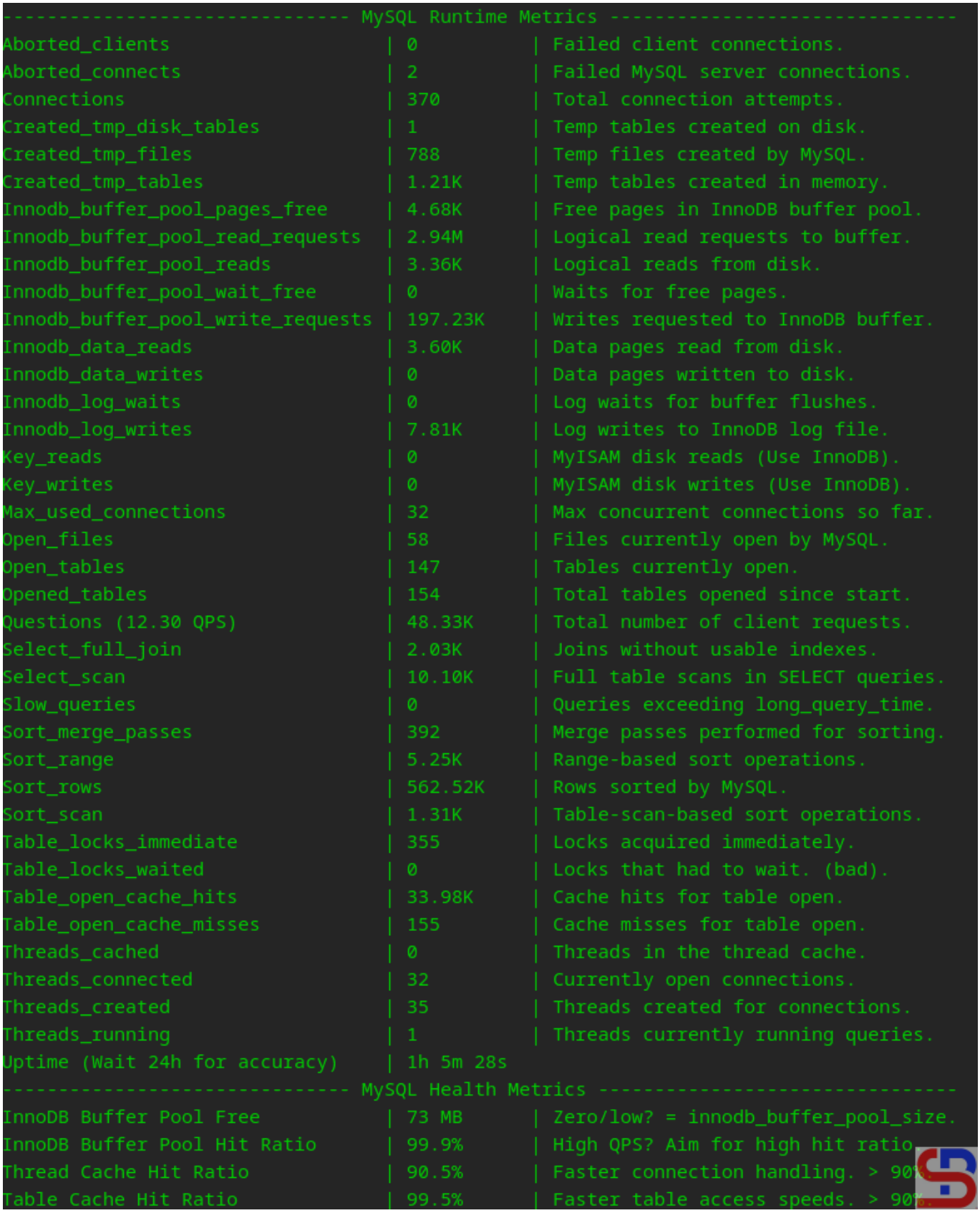
Copy the script below and insert it into the **my.cnf** file:

```
[client]
user=username
password=your_password
```

To download and run the script, run the commands below:

```
curl -O
https://raw.githubusercontent.com/haydenjames/mysqlmonitor-script/main/mysqlm
onitor.sh &&
chmod +x mysqlmonitor.sh &&
./mysqlmonitor.sh
```

There will be a display like the image below:



```
------------------------------ MySQL Runtime Metrics ------------------------------
Aborted_clients                  | 0          | Failed client connections.
Aborted_connects                 | 2          | Failed MySQL server connections.
Connections                      | 370        | Total connection attempts.
Created_tmp_disk_tables          | 1          | Temp tables created on disk.
Created_tmp_files                | 788        | Temp files created by MySQL.
Created_tmp_tables               | 1.21K      | Temp tables created in memory.
Innodb_buffer_pool_pages_free    | 4.68K      | Free pages in InnoDB buffer pool.
Innodb_buffer_pool_read_requests | 2.94M      | Logical read requests to buffer.
Innodb_buffer_pool_reads         | 3.36K      | Logical reads from disk.
Innodb_buffer_pool_wait_free     | 0          | Waits for free pages.
Innodb_buffer_pool_write_requests | 197.23K    | Writes requested to InnoDB buffer.
Innodb_data_reads                | 3.60K      | Data pages read from disk.
Innodb_data_writes               | 0          | Data pages written to disk.
Innodb_log_waits                 | 0          | Log waits for buffer flushes.
Innodb_log_writes                | 7.81K      | Log writes to InnoDB log file.
Key_reads                        | 0          | MyISAM disk reads (Use InnoDB).
Key_writes                       | 0          | MyISAM disk writes (Use InnoDB).
Max_used_connections             | 32         | Max concurrent connections so far.
Open_files                       | 58         | Files currently open by MySQL.
Open_tables                      | 147        | Tables currently open.
Opened_tables                    | 154        | Total tables opened since start.
Questions (12.30 QPS)            | 48.33K     | Total number of client requests.
Select_full_join                 | 2.03K      | Joins without usable indexes.
Select_scan                      | 10.10K     | Full table scans in SELECT queries.
Slow_queries                     | 0          | Queries exceeding long_query_time.
Sort_merge_passes                | 392        | Merge passes performed for sorting.
Sort_range                       | 5.25K      | Range-based sort operations.
Sort_rows                        | 562.52K    | Rows sorted by MySQL.
Sort_scan                        | 1.31K      | Table-scan-based sort operations.
Table_locks_immediate            | 355        | Locks acquired immediately.
Table_locks_waited               | 0          | Locks that had to wait. (bad).
Table_open_cache_hits            | 33.98K     | Cache hits for table open.
Table_open_cache_misses          | 155        | Cache misses for table open.
Threads_cached                   | 0          | Threads in the thread cache.
Threads_connected                | 32         | Currently open connections.
Threads_created                  | 35         | Threads created for connections.
Threads_running                  | 1          | Threads currently running queries.
Uptime (Wait 24h for accuracy)   | 1h 5m 28s
------------------------------ MySQL Health Metrics ------------------------------
InnoDB Buffer Pool Free          | 73 MB      | Zero/low? = innodb_buffer_pool_size.
InnoDB Buffer Pool Hit Ratio     | 99.9%      | High QPS? Aim for high hit ratio.
Thread Cache Hit Ratio           | 90.5%      | Faster connection handling. > 90%.
Table Cache Hit Ratio            | 99.5%      | Faster table access speeds. > 90%.
```

Display of mysqlmonitor

From the image above, you can see that the results from the mysqlmonitor tool are more concise than the mariadb-report tool. To exit this tool, press the **q** key.

Note

If you want detailed metric results, then you can use the mariadb-report tool. However, if you want more concise metrics, you can use the mysqlmonitor tool.

References

linuxblog.io
mariadb.com
github.com

How to Install and Run tuning-primer?

written by sysadmin | 27 December 2025

The previous article explained the mysqltuner script to provide recommendations to increase MariaDB performance. This article will explain the primary-tuning script, which is an alternative or may also be an addition to providing recommendations for MariaDB.

Problem

How to install and run tuning-primer?

Solution

The primary-tuning script was created by Matthew Montgomery using a bash script to provide recommendations against a MySQL/MariaDB database. This script takes information from

“SHOW STATUS LIKE…” and “SHOW VARIABLES LIKE…” to produce recommendations for tuning server variables. To download this script, use the command below:

```
git clone https://github.com/mattiabasone/tuning-primer.git
```

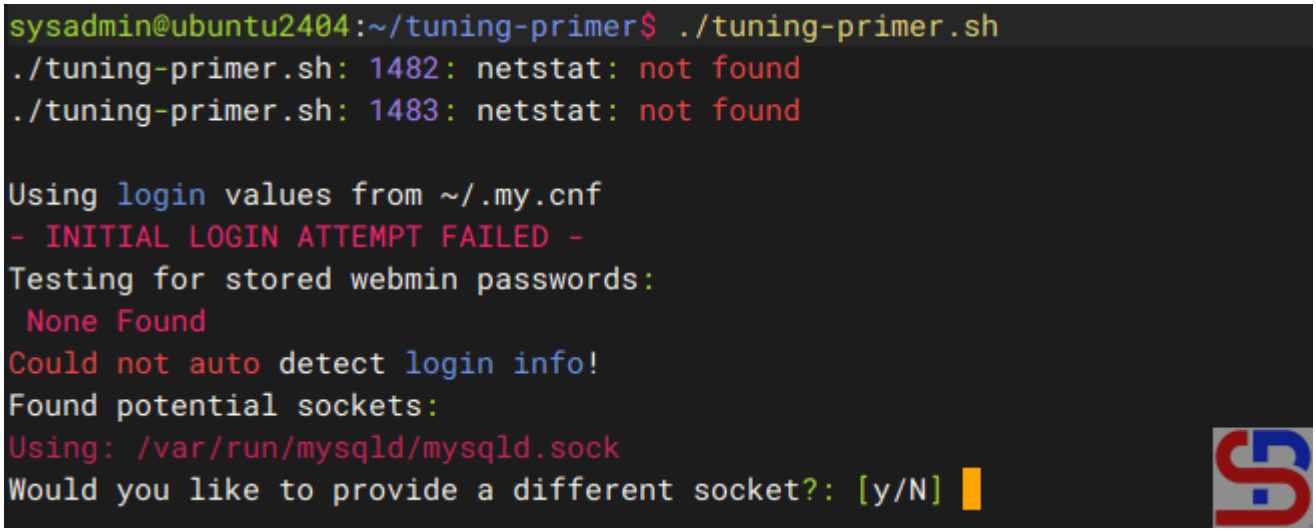
Go to the primary-tuning folder and permit so that the script can be run:

```
chmod +x tuning-primary.sh
```

Then, run the command below to run the primary-tuning script:

```
./tuning-primer.sh
```

There will be a display like the image below:



Run tuning-primer script

Press **Enter**, then you have to enter your username and password so that this script can access your MariaDB database, as in the image below:

```
Do you have your login handy ? [y/N] : y
User: root
Password: qwerty

Would you like me to create a ~/.my.cnf file for you? [y/N] :
```

Enter username and password

Warning

This script writes the password clearly, so you have to be careful when using this script.

After that, the script will display recommendations as in the image below:

```
        -- MYSQL PERFORMANCE TUNING PRIMER --
             - By: Matthew Montgomery -

MySQL Version 10.11.13-MariaDB-0ubuntu0.24.04.1 x86_64

Uptime = 0 days 1 hrs 41 min 2 sec
Avg. qps = 14
Total Questions = 88210
Threads Connected = 33

Warning: Server has not been running for at least 48hrs.
It may not be safe to use these recommendations

To find out more information on how each of these
runtime variables effects performance visit:
http://dev.mysql.com/doc/refman/10.11/en/server-system-variables.html
Visit http://www.mysql.com/products/enterprise/advisors.html
for info about MySQL's Enterprise Monitoring and Advisory Service

SLOW QUERIES
The slow query log is NOT enabled.
Current long_query_time = 10.000000 sec.
You have 0 out of 88224 that take longer than 10.000000 sec. to complete
Your long_query_time seems to be fine

BINARY UPDATE LOG
The binary update log is NOT enabled.
You will not be able to do point in time recovery
See http://dev.mysql.com/doc/refman/10.11/en/point-in-time-recovery.html

WORKER THREADS
Current thread_cache_size = 151
Current threads_cached = 0
Current threads_per_sec = 0
Historic threads_per_sec = 0
Your thread_cache_size is fine
```

tuning-primer script result display

Note

Because this script was created in 2018 and there has been no update, there are several errors that occur where on lines 1482 and 1483 (see first image) in the netstat command where at this time (November 2025), the netstat command has

been changed to **ss** and also the password is written as plain text which is very dangerous if known by the unauthorized user.

References

linuxblog.io
github.com

How to Install And Run mysqltuner?

written by sysadmin | 27 December 2025

If you have a MariaDB database, then you definitely want the performance of the database to improve. Therefore, you should have done several configurations to achieve your goals. There is a script called mysqltuner that you can use to improve the performance of your MariaDB database by providing recommendations.

Problem

How to install and run mysqltuner?

Solution

mysqltuner is a Perl script designed to quickly assess a MySQL setup and implement changes to enhance performance and stability. It supports ~300 indicators for MySQL/MariaDB/Percona Server in this latest version and is actively maintained, supporting many configurations such as Galera Cluster, TokuDB, Performance schema, Linux OS metrics, InnoDB, MyISAM, Aria, and so on. To download it, you can run the command:

```
git clone https://github.com/major/MySQLTuner-perl.git
```

Or use the commands below:

```
wget http://mysqltuner.pl/ -O mysqltuner.pl
wget
https://raw.githubusercontent.com/major/MySQLTuner-perl/master/basic_password
s.txt -O basic_passwords.txt
wget
https://raw.githubusercontent.com/major/MySQLTuner-perl/master/vulnerabilitie
s.csv -O vulnerabilities.csv
```

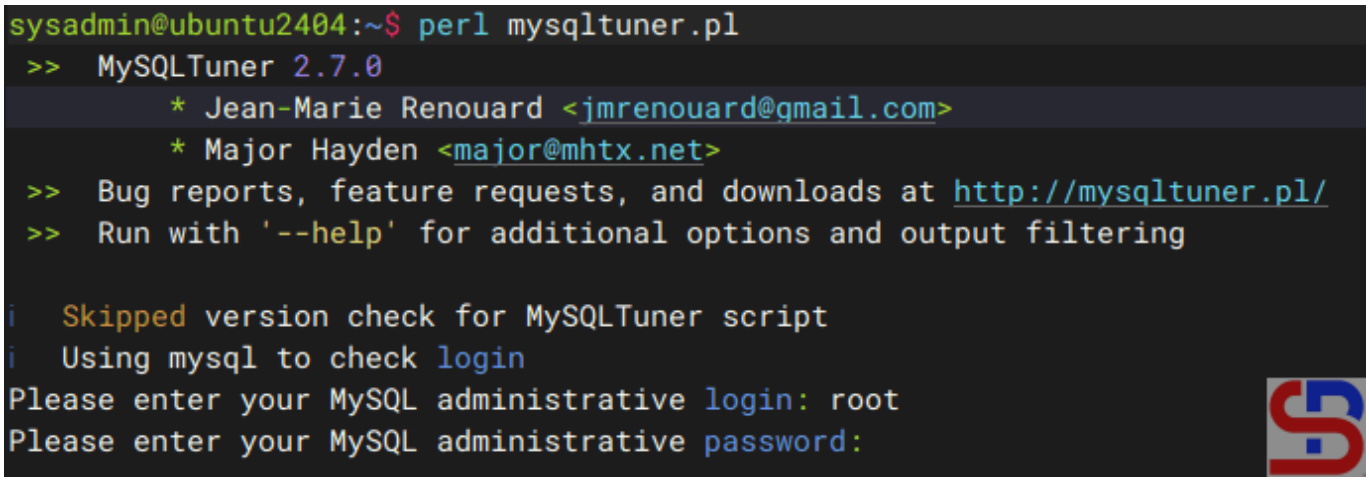
To run this script, you must have Perl installed on your server. So, to run this script, you can use the command (if you download mysqltuner using git, you have to go to **the MySQLTuner-perl** folder):

```
perl mysqltuner.pl
```

Or you permit this script to be executed by using the command:

```
chmod +x mysqltuner.pl
./mysqltuner.pl
```

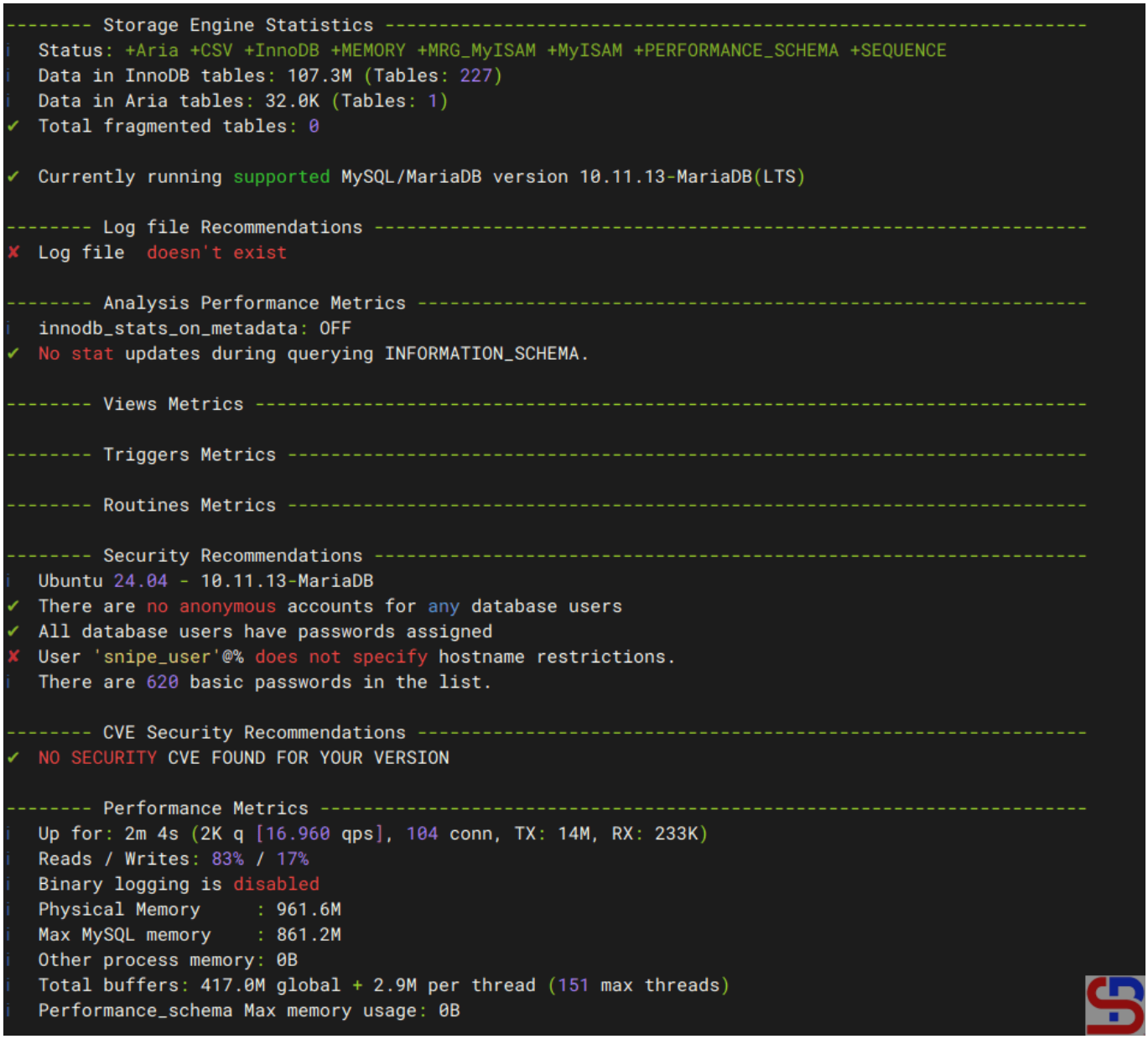
After you run the command, there will be a display as below:



Enetr username and password

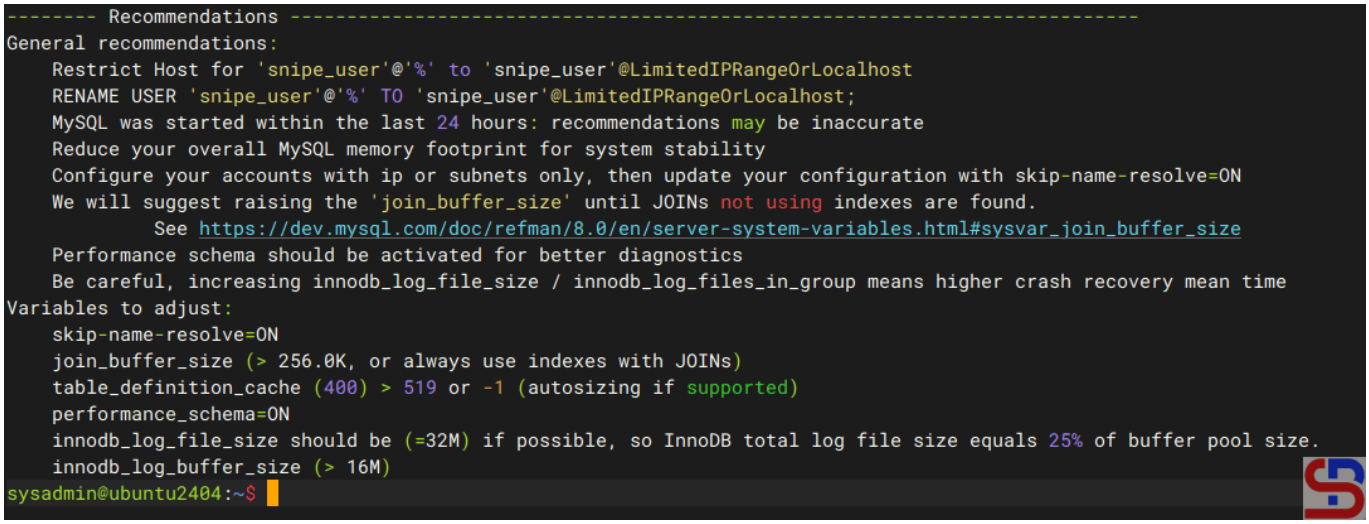
Enter the username and password, and if there is no error,

mysqltuner displays your MariaDB state as shown in the image below:



mysqltuner view

And at the end, mysqltuner will recommend that your MariaDB improve its performance:



Recommendations on mysqltuner

Note

If you want to run mysqltuner with the verbose option, use the command below:

```
perl mysqltuner.pl --verbose
```

If you want to display Maximum Output Information around MySQL, like display database(s) and table(s) on mysqltuner, use the command below:

```
perl mysqltuner.pl --buffers --dbstat --idxstat --sysstat --pfstat --tbstat
```

Use the command below to use CVE(Common Vulnerabilities and Exposures) in mysqltuner:

```
perl mysqltuner.pl --cvefile=vulnerabilities.csv
```

Use the command below to save the results from mysqltuner in a file without displaying it on the screen:

```
perl mysqltuner.pl --silent --outputfile /tmp/result_mysqltuner.txt
```

To update mysqltuner, run the command below:

```
perl mysqltuner.pl --checkversion --updateversion
```

References

mysqltuner.com
github.com
hevodata.com

How to Display All Crontabs Running Using a Bash Script?

written by sysadmin | 27 December 2025

I would like to know who is running crontab on the Linux server.

Problem

How to display all crontabs running using a Bash script?

Solution

By default, run the command below if you want to display the crontab command:

```
crontab -l
```

However, if you want to display another user, for example, user john, use the command below:

```
sudo crontab -l -u john
```

If you want to see all users running crontab, then you can run the command below:

Ubuntu/Debian

```
ls -l /var/spool/cron/crontabs
```

```
sysadmin@ubuntu2404:~$ sudo ls -l /var/spool/cron/crontabs
[sudo] password for sysadmin:
total 8
-rw------- 1 john     crontab 1154 Nov 26 07:50 john
-rw------- 1 sysadmin crontab 1215 Nov 26 08:02 sysadmin
sysadmin@ubuntu2404:~$
```

Displays all users running crontab

RockyLinux/AlmaLinux/RHEL/CentOS

```
ls -l /var/spool/cron/
```

Use the command below if you want to see all users running crontab and display each user's crontab at the same time:

```
awk -F: '$3>=1000 || $1=="root" {print $1}' /etc/passwd |
while read u; do
    if sudo crontab -l -u "$u" &>/dev/null; then
        echo
        echo "=== $u ==="
        sudo crontab -l -u "$u" | grep -v '^#' | sed '/^\s*$/d'
        echo ""
    fi
done
```

```
sysadmin@ubuntu2404:~$ awk -F: '$3>=1000 || $1=="root" {print $1}' /etc/passwd |
while read u; do
    if sudo crontab -l -u "$u" &>/dev/null; then
        echo
        echo "=== $u ==="
        sudo crontab -l -u "$u" | grep -v '^#' | sed '/^\s*$/d'
        echo ""
    fi
done

=== sysadmin ===
@reboot sleep 900 &&    cd /home/sysadmin;./time.sh
0 0 * * 6       cd /home/sysadmin;./saturday.sh
15 7 24-31 * 6          date > time.txt


=== john ===
30 10 * * *     sudo systemctl restart apache2

sysadmin@ubuntu2404:~$
```

Displays all users running crontab and their script(s) on crontab

If you want the result of the above command to be entered as a CSV file, then run the command below:

```
echo "user,cron_entry" > all_user_crontab.csv; \
awk -F: '$3>=1000||$1=="root"{print $1}' /etc/passwd | while read u; do \
    if sudo crontab -l -u "$u" &>/dev/null; then \
        sudo crontab -l -u "$u" | \
        grep -v '^[[:space:]]*#' | grep -v '^[[:space:]]*$' | \
        while read line; do \
            echo "$u,\"$line\"" >> all_user_crontab.csv; \
        done; \
    fi; \
done
```

```
sysadmin@ubuntu2404:~$ echo "user,cron_entry" > all_user_crontab.csv; \
awk -F: '$3>=1000||$1=="root"{print $1}' /etc/passwd | while read u; do \
    if sudo crontab -l -u "$u" &>/dev/null; then \
        sudo crontab -l -u "$u" | \
        grep -v '^[[:space:]]*#' | grep -v '^[[:space:]]*$' | \
        while read line; do \
            echo "$u,\"$line\"" >> all_user_crontab.csv; \
        done; \
    fi; \
done
sysadmin@ubuntu2404:~$ cat all_user_crontab.csv
user,cron_entry
sysadmin,"@reboot       sleep 900 &&    cd /home/sysadmin;./time.sh"
sysadmin,"0 0 * * 6     cd /home/sysadmin;./saturday.sh"
sysadmin,"15 7 24-31 * 6                date > time.txt"
john,"30 10 * * *       sudo systemctl restart apache2"
sysadmin@ubuntu2404:~$
```

Run the command to display all the users and their crontab, and save it to the CSV file

However, if you want to display all crontabs, whether run by the user or the Linux system, then you can use the command below:

```
#!/bin/bash

clean_output() {
    grep -v '^[[:space:]]*#' | grep -v '^[[:space:]]*$'
}

echo "=== SYSTEM CRONTABS ==="
echo ""

# Main crontab system
if [[ -f /etc/crontab ]]; then
    echo "--- /etc/crontab ---"
    clean_output < /etc/crontab
    echo ""
fi

# Cron.d directory
if [[ -d /etc/cron.d ]]; then
    echo "--- /etc/cron.d/ ---"
    for f in /etc/cron.d/*; do
        [[ -f "$f" ]] || continue
        echo "File: $f"
        clean_output < "$f"
```

```
        echo ""
    done
fi

# Cron.daily, cron.hourly, cron.weekly, cron.monthly
for dir in daily hourly weekly monthly; do
    path="/etc/cron.$dir"
    if [[ -d "$path" ]]; then
        echo "--- /etc/cron.$dir/ ---"
        # List script names only (normally no '#' inside)
        ls -1 "$path"
        echo ""
    fi
done

echo "=== USER CRONTABS ==="
echo ""

# All user in /etc/passwd
for user in $(cut -f1 -d: /etc/passwd); do
    uid=$(id -u "$user" 2>/dev/null)
    [[ $? -ne 0 ]] && continue
    if [[ $uid -lt 1000 && $user != "root" ]]; then
        continue
    fi

    crontab_content=$(sudo crontab -l -u "$user" 2>/dev/null | clean_output)

    if [[ -n "$crontab_content" ]]; then
        echo "--- Crontab for user: $user ---"
        echo "$crontab_content"
        echo ""
    fi
done
```

And use the script below if you want to insert the result into a CSV file:

```
#!/bin/bash

OUTPUT="cron_inventory.csv"

echo "type,owner,source,cron_entry" > "$OUTPUT"

filter_clean() {
    grep -v '^[[:space:]]*#' | grep -v '^[[:space:]]*$'
}

################################################
```

```
# SYSTEM CRONTAB
##############################################

if [[ -f /etc/crontab ]]; then
    cat /etc/crontab | filter_clean | while read line; do
        echo "system,root,/etc/crontab,\"$line\"" >> "$OUTPUT"
    done
fi

##############################################
# /etc/cron.d
##############################################

if [[ -d /etc/cron.d ]]; then
    for file in /etc/cron.d/*; do
        [[ -f "$file" ]] || continue
        cat "$file" | filter_clean | while read line; do
            echo "system,root,$file,\"$line\"" >> "$OUTPUT"
        done
    done
fi

##############################################
# cron.daily / cron.hourly / cron.weekly / cron.monthly
##############################################

for dir in daily hourly weekly monthly; do
    path="/etc/cron.$dir"
    if [[ -d "$path" ]]; then
        for script in "$path"/*; do
            [[ -f "$script" ]] || continue
            echo "system,root,$path,$(basename "$script")" >> "$OUTPUT"
        done
    fi
done

##############################################
# USER CRONTABS
##############################################

for user in $(cut -f1 -d: /etc/passwd); do
    uid=$(id -u "$user" 2>/dev/null)
    [[ $? -ne 0 ]] && continue
    if [[ $uid -lt 1000 && $user != "root" ]]; then
        continue
    fi

    crontab -l -u "$user" 2>/dev/null | filter_clean | while read line; do
        echo "user,$user,crontab,\"$line\"" >> "$OUTPUT"
    done
done
```

```
echo "== CSV generated: $OUTPUT =="
```

Note

By displaying users who use crontab on a Linux server, you can save your time and effort investigating if there are commands running at a certain time on that server.

References

stackoverflow.com
unix.stackexchange.com
cyberciti.biz

How to Change MariaDB Port?

written by sysadmin | 27 December 2025

By default, MariaDB uses port 3306. I want to change the default port to another port for security reasons.

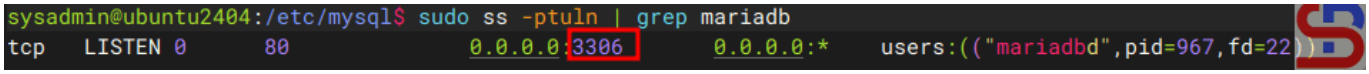
Problem

How to change MariaDB port?

Solution

To see the default MariaDB port, you can use the command below:

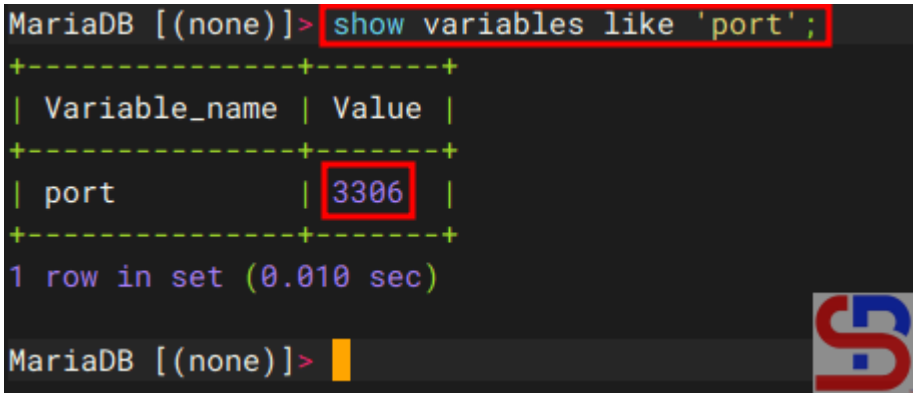
```
sudo ss -ptuln | grep mariadb
```



Display the MariaDB port via netstat

Or you run the query below:

```
show variables like 'port';
```



Display the MariaDB port via query

From the images above, you can see MariaDB is using port 3306. If you want to change the port from 3306 to 4306, for example, then in the file **/etc/mysql/mariadb.conf.d/50-server.cnf** if you are using Ubuntu, add the item below:

```
port                    = 4306
```

Warning

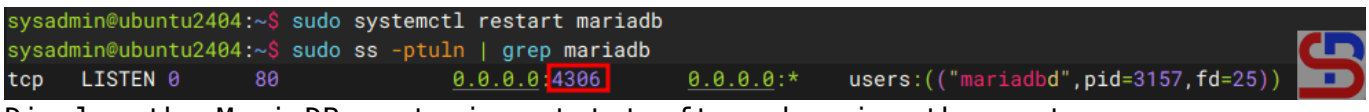
If you're using a distro other than Ubuntu/Debian, you can search for the file by using the command:

```
sudo find / -type f -name "*server.cnf"
```

Then restart mariadb using the command:

```
sudo systemctl restart mariadb
```

After that, you can check the MariaDB port by using one of the commands above, and the MariaDB port should have changed according to the port you want, as shown in the image below:



Display the MariaDB port via netstat after changing the port

Note

If you have changed the default port of MariaDB from 3306 to 4306, for example, then you don’t need to write the port in the Linux command to access MariaDB if you access from localhost:

```
sysadmin@ubuntu2404:~$ sudo ss -ptuln | grep mariadb
tcp   LISTEN 0      80             0.0.0.0:4306       0.0.0.0:*    users:(("mariadbd",pid=3157,fd=25))

sysadmin@ubuntu2404:~$ sudo mariadb -uroot -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 46
Server version: 10.11.13-MariaDB-0ubuntu0.24.04.1 Ubuntu 24.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Access to MariaDB after changing the port from localhost

But, if you access MariaDB from another host, you have to write the option for the port, like in the picture below:

```
sysadmin@docker:~$ mariadb -h 192.168.56.11 -P 4306 -u john -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 50
Server version: 10.11.13-MariaDB-0ubuntu0.24.04.1 Ubuntu 24.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Access to MariaDB after changing the port on another host

References

geeksforgeeks.org
stackoverflow.com

How to Change the Default Port in PostgreSQL?

written by sysadmin | 27 December 2025

By default, PostgreSQL uses port 5432. But for security's sake, I want to change the default port to another one.

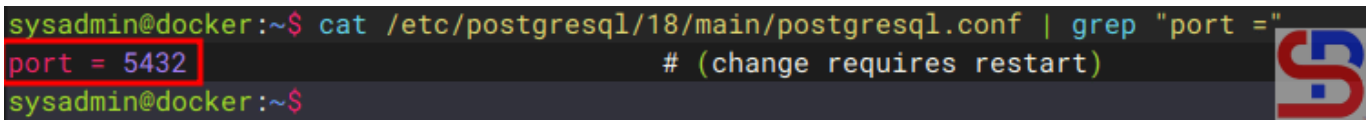
Problem

How to change the default port in PostgreSQL?

Solution

If you want to see the port used by PostgreSQL, you can see it in the postgresql.conf file by using the command (I'm using Ubuntu distro and PostgreSQL version 18):

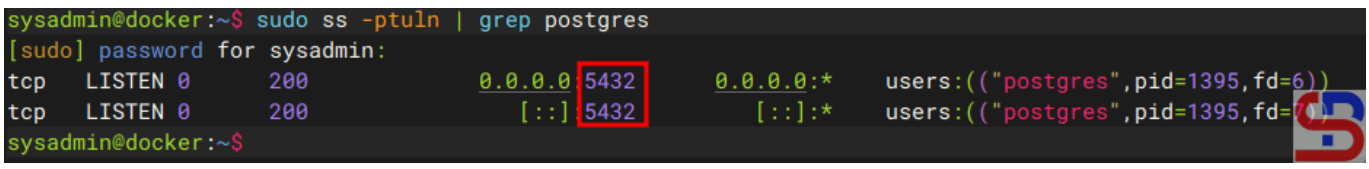
```
cat /etc/postgresql/18/main/postgresql.conf | grep 'port ='
```



Display the PostgreSQL port via postgresql.conf file

Or you can use the command below to see the PostgreSQL port:

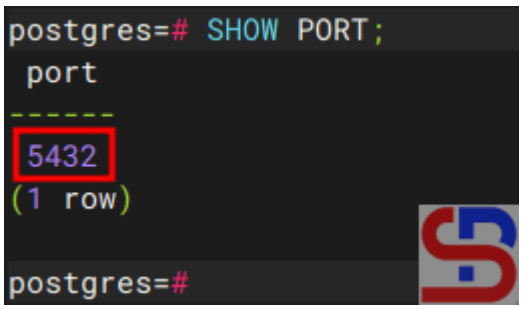
```
sudo ss -ptuln | grep postgres
```



Display the PostgreSQL port via netstat

Or you can use the query in this post to see the port used by PostgreSQL:

```
SHOW PORT;
```



Display the PostgreSQL port via query

From the images above, you can see that PostgreSQL uses port 5432. If you want to change the PostgreSQL port to port 6543, for example, then go to the **/etc/postgresql/18/main/postgresql.conf** file if you use the Ubuntu distro and PostgreSQL version 18, and change the port value from 5432 to 6432.

Warning

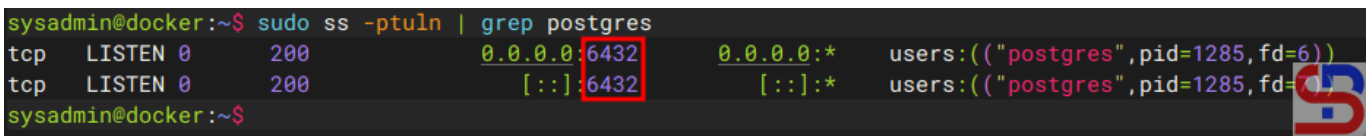
If you're using a distro other than Ubuntu/Debian, you can search for the postgresql.conf file by using the command:

```
sudo find / -type f -name "*postgresql.conf"
```

Then restart PostgreSQL using the command:

```
sudo systemctl restart postgresql
```

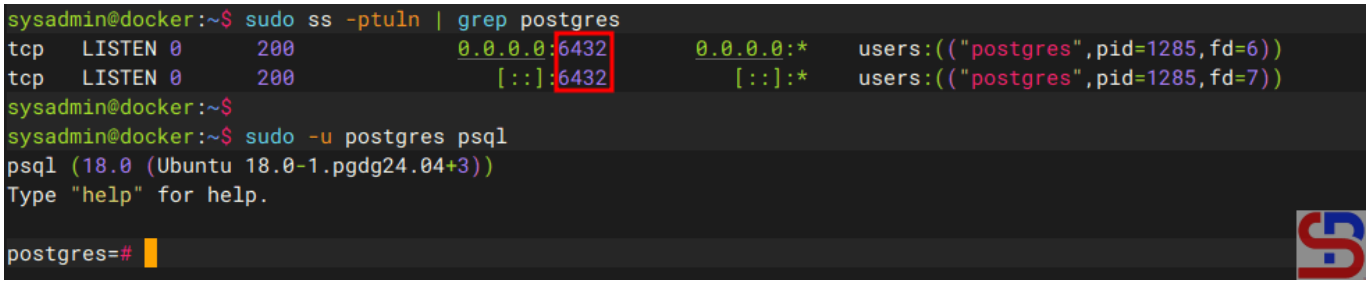
After that, you can check the PostgreSQL port by using one of the commands above, and the PostgreSQL port should have changed according to the port you want as shown in the image below:



Display the PostgreSQL port via netstat after changing the port

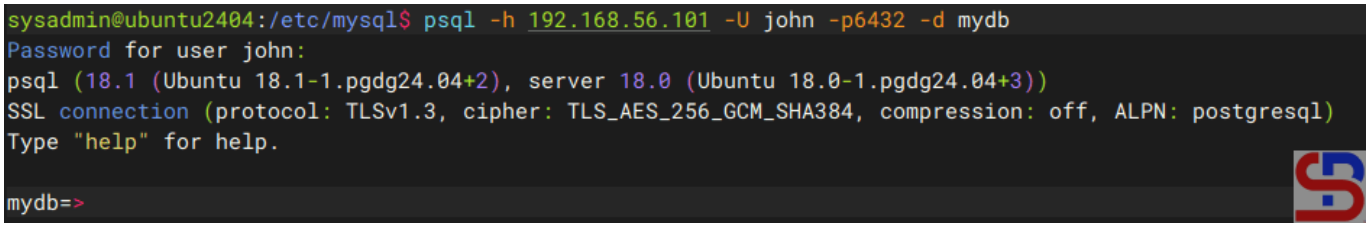
Note

If you have changed the default port of PostgreSQL from 5432 to 6432, for example, then you don’t need to write the port in the Linux command to access PostgreSQL if you access from localhost:



Access PostgreSQL from localhost after changing the default port

But, if you access PostgreSQL from another host, you have to write the option for the port, like in the picture below:



Access PostgreSQL from another host after changing the default port

References

stackoverflow.com
dbvis.com
geeksforgeeks.org