

Installation und Konfiguration von MariaDB im LXC-Container

Einleitung

MariaDB ist ein leistungsfähiges, relationales Open-Source-Datenbankmanagementsystem (RDBMS), das als freier Fork von MySQL entwickelt wurde. Es entstand, nachdem Oracle MySQL übernommen hatte, um die Offenheit und Community-Entwicklung sicherzustellen.

MariaDB wird überall dort eingesetzt, wo strukturierte Daten gespeichert, verarbeitet und abgefragt werden müssen – von kleinen Webanwendungen bis hin zu großen Unternehmenssystemen. Typische Einsatzbereiche sind:

- **Webanwendungen:** Speicherung von Benutzerdaten, Inhalten und Konfigurationsdaten.
- **Business-Anwendungen:** Verwaltung von Bestellungen, Kundendaten oder Lagerbeständen.
- **Datenanalyse & Reporting:** Grundlage für Business-Intelligence-Tools.
- **Cloud- und Container-Umgebungen:** als skalierbare, hochverfügbare Datenbanklösung.

Durch ihre hohe Kompatibilität zu MySQL, gute Performance, Replikations- und Clustering-Funktionen ist MariaDB eine der beliebtesten Datenbanken in der Open-Source-Welt.

Hinweis zu Begrifflichkeiten:

Obwohl MariaDB ein eigenständiges Datenbankmanagementsystem ist, werden in der Befehlszeile und in Tools häufig weiterhin Begriffe wie `mysql` verwendet – zum Beispiel bei Befehlen wie `mysql_secure_installation` oder beim Datenbank-Client `mysql`.

Der Grund dafür ist die vollständige Kompatibilität zu MySQL: MariaDB nutzt bewusst die gleichen Befehlsnamen und Schnittstellen, damit bestehende Anwendungen, Skripte und Dokumentationen ohne Änderungen weiterverwendet werden können.

Das Auftreten des Begriffs „mysql“ bedeutet also nicht, dass MySQL installiert ist – es handelt sich lediglich um beibehaltene Namen zur Sicherstellung der Kompatibilität.

Voraussetzungen

- LXC-Container mit Ubuntu 20.04/22.04/24.04 (getestet mit Ubuntu 24.04)
- Netzwerzugriff auf den Container
- Root- oder `sudo`-Berechtigungen
- User `pdal` mit sudo-Rechten eingerichtet

1. Paketliste aktualisieren

```
sudo apt update
```

2. MariaDB installieren

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

```
pdal@mariadb120:~$ sudo apt install mariadb-server -y
Reading package lists... done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
galera-4 gawk libcgi-fast-perl libcgi-pm-perl libclone-perl libcom32c0_1.02-1 libfcgi-perl libfcgi0t64 libgdbm-compat4t64 libhtml-parser-perl libio-html-perl liblwp-mediatypes-perl liblzo2-2 libmariadb3 libmemcached4.0 libtimedate-perl liburi-perl liburing2 mariadb-client mariadb-client-5.7 mariadb-plugin-provider-lzma mariadb-plugin-provider-lzo mariadb-socat
Suggested packages:
gawk-doc libmldb-perl libnet-daemon-perl libsql-statement-perl libbusiness-isbn-perl libregexp-ipv6-perl libwww-perl mailx mariadb-docs libtap-harness-archive-perl doc-base
```

Überprüfen der Installation:

```
mariadb --version
```

```
pdal@mariadb120:~$ mariadb --version
mariadb Ver 15.1 Distrib 10.11.13-MariaDB, for debian-linux-gnu (x86_64) using EditLine wrapper
pdal@mariadb120:~$
```

3. MariaDB-Dienst starten und aktivieren

```
sudo systemctl start mariadb
sudo systemctl enable mariadb
```

```
pdal@mariadb120:~$ sudo systemctl start mariadb
sudo systemctl enable mariadb
Synchronizing state of mariadb.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable mariadb
pdal@mariadb120:~$
```

Status prüfen:

```
sudo systemctl status mariadb
```

```
pdal@mariadb120:~$ systemctl status mariadb
● mariadb.service - MariaDB 10.11.15 database server
    Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset: enabled)
    Active: active (running) since Fri 2025-06-27 09:37:44 UTC; 6min ago
      Docs: man:mariadb(8)
             https://mariadb.com/kb/en/library/systemd/
   Main PID: 7305 (mariadb)
     Status: "Taking your SQL requests now..."
        Tasks: 9 (limit: 28969)
       Memory: 37.9M (peak: 84.5M swap: 43.3M swap peak: 43.3M)
          CPU: 692ms
        CGroup: /system.slice/mariadb.service
                 `--7305 /usr/sbin/mariadb
pdal@mariadb120:~$
```

Nach dem Starten und Aktivieren des MariaDB-Dienstes läuft der Datenbankserver nun im Hintergrund und ist standardmäßig über localhost erreichbar. Das bedeutet, dass lokale Anwendungen oder Befehle wie `mysql -u root -p` sofort eine Verbindung zur Datenbank herstellen können.

In den nächsten Schritten wird die Datenbank zunächst abgesichert, um unbefugten Zugriff zu verhindern. Dazu wird das integrierte Sicherheits-Skript ausgeführt: `sudo mysql_secure_installation`

Dieses Skript hilft dabei, ein Root-Passwort zu setzen, anonyme Benutzer zu entfernen, den entfernten Root-Zugriff zu deaktivieren und die Testdatenbank zu löschen. Erst danach werden die notwendigen Anpassungen vorgenommen, um einen sicheren Remote-Zugriff auf die Datenbank zu ermöglichen.

🔒 4. MariaDB absichern

```
sudo mysql_secure_installation
```

Antwortempfehlungen:

switch to unix socket authentication? N

change root password? Y

Remove anonymous users? Y

Disallow root login remotely? Y (für Produktivsysteme)

Remove test database and access to it? Y

Reload privilege tables? Y

```
pdal@mariadb120:~$ sudo mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] n
... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
... Success!

Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.
```

🌐 5. Remote-Zugriff erlauben (optional)

externe Verbindungen z.B. von einem Webserver-Container erlauben. Hierfür wird die Listen-Address geändert.

Konfigurationsdatei anpassen:

```
sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

Suche folgende Zeile:

```
bind-address = 127.0.0.1

GNU nano 7.2                               /etc/mysql

# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]

#
# * Basic Settings
#


#user          = mysql
pid-file      = /run/mysqld/mysqld.pid
basedir        = /usr
#datadir       = /var/lib/mysql
#tmpdir        = /tmp

# Broken reverse DNS slows down connections considerably and name res
# safe to skip if there are no "host by domain name" access grants
#skip-name-resolve

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address      = 127.0.0.1

#
# * Fine Tuning
```

Ändere sie zu:

```
bind-address = 192.168.137.120 (IP-Adresse des eigenen LXC-Containers)

# Broken reverse DNS slows down connections considerably and n
# safe to skip if there are no "host by domain name" access gr
#skip-name-resolve

# Instead of skip-networking the default is now to listen only
# localhost which is more compatible and is not less secure.
bind-address      = 192.168.137.120

#
```

⌚ Erklärung: Mit dieser Änderung lauscht der MariaDB-Server nicht mehr nur auf localhost, sondern zusätzlich auf der angegebenen Container-IP-Adresse (192.168.137.120). Dadurch ist die Datenbank nun auch aus dem lokalen Netzwerk erreichbar, z. B. von einem anderen LXC-Container oder einem externen Server. Ein Client kann sich nun mit folgendem Befehl verbinden: `mysql -h`

192.168.137.120 -u BENUTZERNAME -p ⚠ Wichtig: Für die Verbindung von externen Clients muss auch die Firewall (falls aktiv) den Zugriff auf Port 3306/tcp erlauben. Außerdem muss sichergestellt sein, dass der verwendete Datenbankbenutzer entsprechende Rechte für % (alle Hosts) oder die spezifische IP des Clients hat.

Alternativ kann man die IP-Adresse **0.0.0.0** verwenden. Damit lauscht MariaDB auf allen Ports. Diese Option ist dann zu empfehlen, wenn man einen DHCP-/DNS-Server verwendet und man nicht sicherstellen kann, dass der Container/Server immer die gleiche IP-Adresse zugewiesen bekommt.

abc 6. Standard-Zeichensatz und Kollation festlegen

i Einführung: In einer Datenbank legen der **Standard-Zeichensatz** (`character set`) und die **Kollation** (`collation`) fest, wie Textdaten gespeichert, verglichen und sortiert werden.

- Der Zeichensatz definiert, welche Zeichen überhaupt gespeichert werden können (z.B. Buchstaben, Zahlen, Sonderzeichen, Emojis).
- Die Kollation bestimmt, wie diese Zeichen miteinander verglichen werden, z.B. ob Groß- und Kleinschreibung berücksichtigt wird oder welche Sortierreihenfolge verwendet wird.

Eine konsistente Einstellung von Zeichensatz und Kollation ist wichtig, um Probleme bei Abfragen, Vergleichen und Datenmigrationen zu vermeiden und sicherzustellen, dass alle Anwendungen die Daten korrekt interpretieren.

Füge in der gleichen Konfigurationsdatei (Abschnitt [mysqld]) sicherheitshalber die Zeichensatz- und Kollationsparameter hinzu oder passe sie an:

Vorher:

```
character-set-server = utf8mb4
collation-server = utf8mb4_general_ci
```

Ändern zu:

```
character-set-server = utf8mb4
collation-server = utf8mb4_general_ci
```

```
#  
  
# MySQL/MariaDB default is Latin1, but in Debian we rather default  
# utf8 4-byte character set. See also client.cnf  
character-set-server = utf8mb4  
collation-server     = utf8mb4_general_ci  
  
#  
# * InnoDB  
#  
  
# InnoDB is enabled by default with a 10MB datafile in /var/lib/mysqld  
# Read the manual for more InnoDB related options. There are many!  
#
```

 **Erklärung:** Mit `character-set-server = utf8mb4` wird sichergestellt, dass die Datenbank standardmäßig den modernen UTF-8-Zeichensatz verwendet, der alle Unicode-Zeichen unterstützt (z. B. Emojis oder Sonderzeichen). Bei der Kollation gibt es zwei Varianten:

- `utf8mb4_general_cs` → **case-sensitive**, Groß- und Kleinschreibung wird beim Vergleich berücksichtigt (z.B. „Test“ ≠ „test“).
 - `utf8mb4_general_ci` → **case-insensitive**, Groß- und Kleinschreibung wird nicht berücksichtigt (z.B. „Test“ = „test“). Die Einstellung `utf8mb4_general_ci` wird häufig verwendet, da sie in den meisten Anwendungsfällen für Benutzernamen, Logins oder Suchfunktionen praktischer ist.

```
# * Character sets
#
# MySQL/MariaDB default is Latin1, but in Debian we rather default to the full
# utf8 4-byte character set. See also client.cnf
character-set-server = utf8mb4
collation-server     = utf8mb4_general_ci

#
# * InnoDB
#
# InnoDB is enabled by default with a 10MB datafile in /var/lib/mysql/.
# Read the manual for more InnoDB related options. There are many!
# Most important is to give InnoDB 80 % of the system RAM for buffer use:
# https://mariadb.com/kb/en/innodb-system-variables/#innodb_buffer_pool_size
```

Dann MariaDB neu starten:

```
sudo systemctl restart mariadb
```

7. MariaDB-Nutzer für externen Zugriff erstellen

i Einführung: In diesem Schritt wird ein Datenbank-Benutzer erstellt, der auf MariaDB von einem externen Rechner oder Container zugreifen kann. Es handelt sich dabei um einen **reinen Datenbank-User**, der nicht zwingend als Systembenutzer (`pda1`) auf dem Server existieren muss. Der Name und das Passwort können frei gewählt werden. Mit den gesetzten Berechtigungen (`GRANT ALL PRIVILEGES`) erhält der Benutzer Zugriff auf alle Datenbanken und Tabellen und kann Berechtigungen an andere Benutzer vergeben.

In die Datenbank einloggen:

```
sudo mariadb
```

```
pdal@mariadb120:~$ sudo mariadb
[sudo] password for pdal:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 52
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)]> 
```

Benutzer erstellen und Berechtigungen setzen:

```
CREATE USER 'pdal'@'192.168.137.%' IDENTIFIED BY 'JadeHS20';
GRANT ALL PRIVILEGES ON *.* TO 'pdal'@'192.168.137.%' WITH GRANT OPTION;
FLUSH PRIVILEGES;
EXIT;
```

💡 Erklärung der Befehle:

- **CREATE USER 'pdal'@'192.168.137.%' IDENTIFIED BY 'JadeHS20';** Erstellt einen neuen Datenbankbenutzer namens **pdal**, der sich von allen IP-Adressen im Netzwerkbereich **192.168.137.%** verbinden kann, mit dem Passwort **JadeHS20**.
- **GRANT ALL PRIVILEGES ON *.* TO 'pdal'@'192.168.137.%' WITH GRANT OPTION;** Vergibt alle Rechte auf alle Datenbanken und Tabellen an diesen Benutzer. Mit **WITH GRANT OPTION** kann der Benutzer auch Berechtigungen an andere Benutzer weitergeben.
- **FLUSH PRIVILEGES;** Aktualisiert die Berechtigungen, damit die Änderungen sofort wirksam werden.
- **EXIT;** Trennt die Verbindung zur MariaDB-Datenbank und kehrt zur Shell zurück.

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
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)]> CREATE USER 'pdal'@'192.168.137.%'
-> IDENTIFIED BY 'JadeHS20';
Query OK, 0 rows affected (0.039 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'pdal'@'192.168.137.%' WITH GRANT OPTION;
Query OK, 0 rows affected (0.039 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> EXIT;
Bye
pdal@mariadb120:~$ 
```

Hinweis: Soll ein Zugriff von außerhalb des lokalen Netzwerkes erfolgen (z. B. vom Client-PC), müsste '`pdal'@'%'` verwendet werden. Damit ist jede IP-Adresse erlaubt.

🔍 8. Port-Freigabe prüfen (optional)

Standardport für MariaDB ist 3306. Teste mit:

```
ss -tulpen | grep 3306
```

Die Ausgabe des Befehls `ss -tulpen | grep 3306` zeigt die Netzwerkverbindungen (oder Sockets), die auf Port 3306 (Standard-Port für MySQL/MariaDB) aktiv sind. Du hast zusätzlich die Optionen `-tulpen` verwendet:

- t: TCP
- u: UDP
- l: nur lauschende (listening) Sockets
- p: Prozessinformationen anzeigen
- e: erweiterte Informationen (z. B. inode, sk)
- n: keine DNS-Namen, sondern IPs/Ports anzeigen

Erklärung deiner Zeile (aufgeschlüsselt):

```
tcp      LISTEN  0          80      192.168.137.120:3306      0.0.0.0:*      uid:108
ino:27953 sk:2005 cgroup:/system.slice/mariadb.service <->
```

Spalte / Feld	Bedeutung
<code>tcp</code>	Protokoll: TCP
<code>LISTEN</code>	Socket-Zustand: horcht auf eingehende Verbindungen
<code>0</code>	<code>Recv-Q</code> : Empfangs-Queue (0 = nichts wartet auf Verarbeitung)
<code>80</code>	<code>Send-Q</code> : Sende-Queue (z. B. wartende Pakete; bei LISTEN fast immer >0)
<code>192.168.137.120:3306</code>	IP-Adresse und Port (MariaDB horcht hier auf diesem Interface/Port)
<code>0.0.0.0:*</code>	Remote-Adresse/Port: akzeptiert Verbindungen von allen IPs (alle Clients)
<code>uid:108</code>	UID des Prozesses (Benutzer, z. B. <code>mysql</code> oder <code>mariadb</code>)

Spalte / Feld	Bedeutung
ino:27953	Inode-Nummer des Sockets (für interne Verwaltung im Kernel)
sk:2005	Socket-ID (Referenz auf Kernel-Socket, nützlich für Debugging)
cgroup:/system.slice/mariadb.service	CGroup (Systemd-Dienst <code>mariadb.service</code>)
<->	Platzhalter (bei LISTEN irrelevant, da keine Gegenstelle existiert)

Zusammenfassung:

Was das konkret bedeutet:

- MariaDB ist erfolgreich gestartet
- hört auf Port 3306
- auf allen Netzwerkinterfaces, also:
 - 127.0.0.1 (localhost)
 - 192.168.137.120 (deine Container-IP)
 - ggf. auch andere lokale IPs
 - Verbindungen aus dem Netzwerk (z. B. 192.168.137.0/24) sind möglich – vorausgesetzt:
 - der User ist korrekt eingerichtet in unserem Fall(pdal@192.168.137.%)
 - es gibt keine Firewallregel, die den Zugriff blockiert

 Tipp: Zugriff testen

Von einem anderen Host im selben Netz:

```
mysql -u pdal -p -h 192.168.137.120
```

```
apache101 login: pdal
Password:
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.12-11-pve x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro
pdal@apache101:~$ mysql -u pdal -p -h 192.168.137.120
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 53
Server version: 5.5.5-10.11.13-MariaDB-0ubuntu0.24.04.1 Ubuntu 24.04

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

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

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

mysql> ■
```

Hierfür benötigt man zumindest einen minimalistischen Mariadb Client. Man kann dafür den [mariadb core client](#) nutzen. Dazu in einem anderen Container einfach

```
sudo apt install mariadb-client-core
```

eingeben und mit [y](#) bestätigen

Wenn das klappt: Netzwerkzugriff funktioniert.

Wenn nicht: Prüfen, ob eine Firewall (z.B. ufw oder iptables) den Port auf dem Client blockiert:

```
sudo ufw status
```

Unsere Anwendungen können so automatisiert die MariaDB direkt erreichen(z.B. direkter Zugriff von unseren Programmen). Für manuellen Zugriff, setzen wir im nächsten Dokument phpMyAdmin zur Administrierung der MariaDB auf.

Quellen

- „MariaDB Documentation | MariaDB Documentation“. Zugegriffen: 25. September 2025. [Online].
Verfügbar unter: [MariaDB Dokumentation](#)
- „Installing MariaDB Server Guide | MariaDB Documentation“. Zugegriffen: 25. September 2025. [Online].
Verfügbar unter: [MariaDB Server Installation](#)
- „Connecting to MariaDB Server | MariaDB Documentation“. Zugegriffen: 25. September 2025. [Online].
Verfügbar unter: [Connecting MariaDB Server](#)
- „Essential Queries Guide | MariaDB Documentation“. Zugegriffen: 25. September 2025. [Online].
Verfügbar unter: [Essential Queries Guide](#)

- „Server Management | MariaDB Documentation“. Zugegriffen: 25. September 2025. [Online]. Verfügbar unter: [Server Management](#)
 - „Security | MariaDB Documentation“. Zugegriffen: 25. September 2025. [Online]. Verfügbar unter: [Security](#)
-

Lizenz

Dieses Werk ist lizenziert unter der **Creative Commons - Namensnennung - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz.**

[Zum Lizenztext auf der Creative Commons Webseite](#)