Training - Grundlagen MySQL Admininstration (Windows)

Agenda

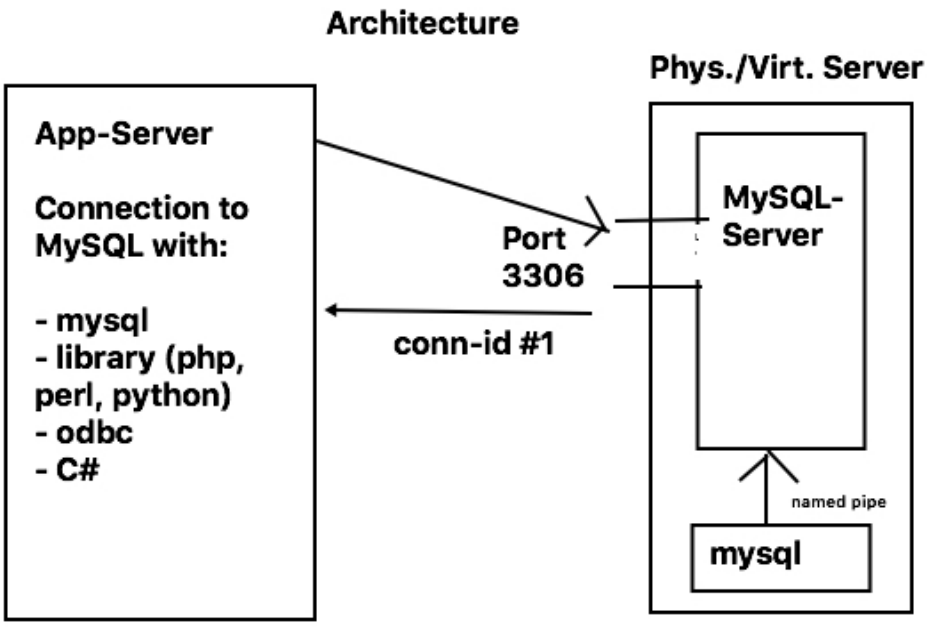
1. Architecture of MySQL
    - MySQL Architectur
    - Verarbeitungsschritte Server (Schritte)
    - InnoDB Struktur
    - Storage Engines
    - Unterschiede MySQL 5.7 -> 8
    - Ort Datenverzeichnis Windows
2. Installation
    - MySQL auf andere Platte installieren
    - Start/Status/Stop/Enable von MySQL
    - Lauscht mysql nach draussen ?
    - 2. Instanz von MySQL erstellen
3. Konfiguration
    - Konfiguration anpassen und neu starten
4. Datenbank - Objekte
    - Databases
    - Tables
    - Events
    - Views
5. Administration
    - Globale und Session Variablen (Server System Variables)
    - Global and Session Status
    - Error-log
    - Slow Query Log
    - MySQL - Client - Tools (most important)
    - Manage max_connections
6. Backup
    - Backup mit mysqldump - best practices
    - mysqldump through the air
    - Backups PIT (Point-In-Time recovery)
    - Backup und Wiederherstellen in neuer Datenbank
    - mysqldump mit asynchroner Verschlüsselung
    - mydumper und myloader
7. Sicherheit
    - Absichern von Server/Client mit ssl
    - Prüfen ob socket verwendet, bei lokalem System
    - mysql_secure_installation - validate plugin aktivieren
    - general log deaktivieren

- plugin vs. components
- User Passwort-Länge und Passwort-Ablauf
- Trigger ohne Super-Rechte anlegen

8. Sicherheit (CIS)
    - 1.1 Place Databases on Non-System Partitions
    - 1.2 Use dedicated Least Privileged Account
9. Tools
    - Testdatenbank Sakila installieren
10. Performance - Analyse
    - Tool pt-query-digest
11. Authentifizierung / User-Management
    - Für User altes Password-Verfahren mysql_native_password verwenden in MySQL 8
    - Wildcard-Rechte für Datenbank
    - Rollen
12. Replication
    - Overview
    - Multi-Source-Replication
    - Binlog format
    - Change-Replication-Filter
    - What to do on error (sql-thread not running)
13. Upgrade
    - Upgrade von MySQL 5.7 -> 8
14. Windows
    - Welchen Benutzer für den Service verwenden?
15. Tipps & Tricks
    - Version von MySQL rausfinden
    - Show Information_schema within MySQL Workbench
    - Set path in Windows for User to easily use mysql
    - Security with outfile mysql
16. Documentation
    - Server System Variables - Reference
    - MySQL Performance Dokument - en

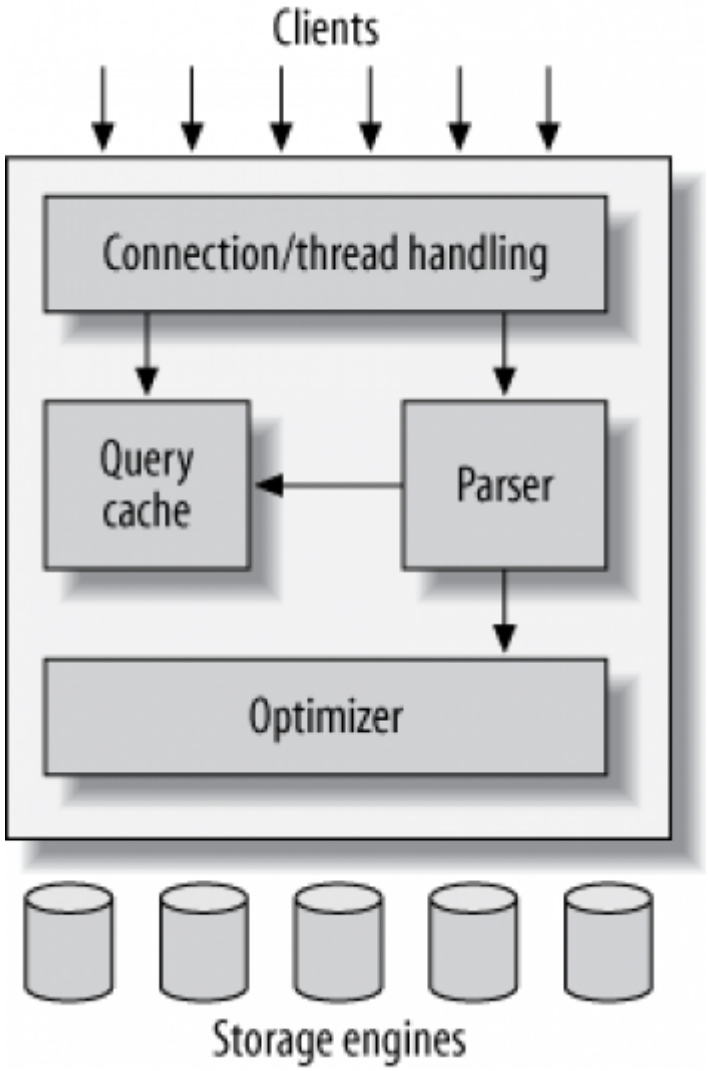
Architecture of MySQL

MySQL Architectur



Verarbeitungsschritte Server (Schritte)

Overview

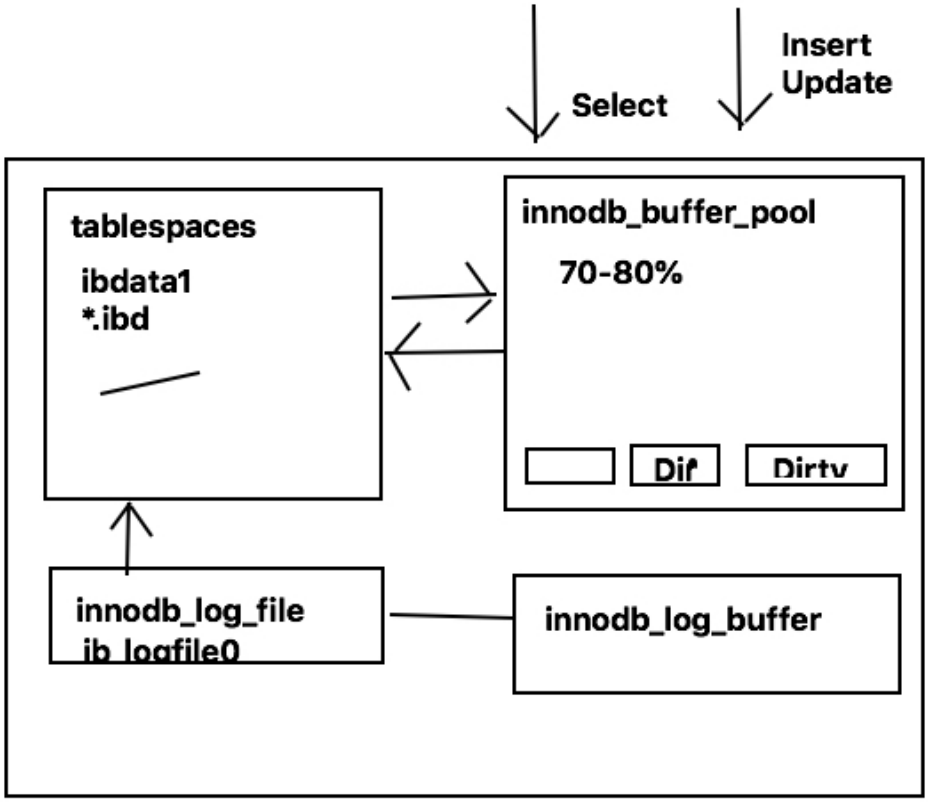


Changes in MySQL 8

- There is no query cache

InnoDB Struktur

Overview



Details

- InnoDB Buffer Pool consists of pages of 16Kbytes Size (default)

Storage Engines

Why ?

```
Decide:
How to save your data internally
```

What do they do ?

- In charge for: Responsible for storing and retrieving all data stored in MySQL
- Each storage engine has its:
  - Drawbacks and benefits
- Server communicates with them through the storage engine API
  - this interface hides differences
  - makes them largely transparent at query layer
  - api contains a couple of dozen low-level functions
    - e.g. “begin a transaction”,
    - “fetch the row that has this primary key”

What do they not do ?

- Storage Engines do not parse SQL

- Storage Engines do not communicate with each other
- They simply .....
    - They simply respond to requests from the server

Which are the most important one ?

- MyISAM
- InnoDB
- Memory
- CSV
- Blackhole (/dev/null)
- Archive
- Partition
- (Federated)

In Detail: MyISAM - Storage Engine

- table locks
    - Locks are done table-wide
- no automatic data-recovery
    - you can loose more data on crashes than with e.g. InnoDB
    - you can loose up to 8 seconds of data
- no transactions
- only indices are saved in memory through MySQL
- compact saving (data is saved really dense)
- table scans are quick

In Detail: InnoDB - Storage Engine

Features

- support hot backups (because of transactions)
- transactions are supported
- foreign keys are supported
- row-level locking
- multi-versioning

Internally

- indexes refer to the data through primary keys
- indexes can quickly get huge in size
    - if size of primary index is not small
- InnoDB puts Data in Buffer Pool
- The Buffer Pool is in memory

Unterschiede MySQL 5.7 -> 8

In Version 8 schnellere Feature-Wechsel

```
Von minor zu minor version, sehr viele neue Features

8.0.23 -> 8.0.24

Das war vorher eher stabiler in der Form ganz wenigen bis gar keinen neuen Features
```

Wegfall von *.frm - Datein von MySQL 5.7 -> 8

Set persist (neu in Version 8)

```
Während der Laufzeit server system variablen persistent setzen
```

mysql ssl verbindung

```
--ssl geht nicht mehr in MySQL 8
stattdessen:
--ssl-mode=REQUIRED
```

Komponenten / Components

```
## Alternative zu den Plugins

Components/Komponenten sind neu in MySQL 8.
```

Ort Datenverzeichnis Windows

Installation

MySQL auf andere Platte installieren

Walkthrough

```
## 1. Download installer

## 2. Run Default installation to C: (with service) without installing service

## 3. Move installation directory to new destination (e.g. D:\

## 4. Create new service with new destination (config-file)
## Erstellt den Service
## Rechte muss stimmen / Rechte müssen gleich bleiben
mysqld.exe --install --defaults-file=D:\mymysqldir\my-defaults.cnf

## 5. service starten
## oder unter Verwaltung
net start mysql
```

```
### Start/Status/Stop/Enable von MySQL
```

```
### starten /stoppen
```

cmd.exe als Administratoren ausführen (Rechte Maustaste) net start MySQL80

stoppen

net stop MYSQL80

```
### Alternativ -> Dienste
```

Dienste -> und in der Liste MySQL80 -> Rechte Maustaste

```
### Lauscht mysql nach draussen ?


### Wie finde ich das raus ?
```

lsof -i | grep mysql

localhost means it does NOT listen to the outside now

mysqld 5208 mysql 19u IPv4 56942 0t0 TCP localhost:mysql (LISTEN)

```
### 2. Instanz von MySQL erstellen


### Walkthrough
```

1. We stop MySQL

net stop MySQL80

2. Copy the Instance - Data C:\ProgramData\MySQL\MySQL Server 8.0

-> to e.g. C:\ProgramData\MySQL\MySQL-Instance-2

3. Change Permission

Add user "NETWORK SERVICE" with all permissions (beside special per permission)

4. Adjust my.ini in new folder MySQL-Instance-2

Adjust:

to new folder

datadir=

port

to new port (not 3306) - must be unused

5. Open cmd.exe as Administrator

then cd to bin-folder of mysql

cd C:\Program Files\MySQL\MySQL Server 8.0\bin

install service

mysqld.exe --install mysql2 --defaults-file=C:\ProgramData\MySQL\MySQL-Instanz-2\my.ini

6. Go to service and refresh
7. Go to properties and change user to:

NETWORK SERVICE (clear password - lines and press o.k.)

8. Start new service and be happy

```
### Debugging

  * Is the path to binary correct in service
  * Is the defaults-file path correct in service
  * Is user NETWORK SERVICE added to new folder
  * Is datadir correct in new my.ini
  * Is port not same as in old my.ini



## Konfiguration

### Konfiguration anpassen und neu starten

## Datenbank - Objekte

### Databases


### Explanations
```

open a connection to the mysql-server by entering

mysql

then you will get

mysql>

Comments within mysql-client

three - in a row

```
### Show databases
```

mysql mysql> ;; from here i leave out mysql> ;; so you can easily copy & paste the lines hereafter show databases

-- -- or --

show schemas

-- -- or by using information_schema --

select * from information_schema.schemata;

```
### Use a specific database
```

use specific database

use sakila;

Create database

```
create database training
create schema training2
```

Tables

Show tables

```
## within mysql>
## so on the command-line enter:
## mysql (as root)
USE sakila
SHOW TABLES

-- or --

select * from information_schema.TABLES
```

Create table

```
-- only if you want to create table in a completely new database
create schema training;
USE training
CREATE TABLE people (id INT NOT NULL AUTO_INCREMENT, name VARCHAR(20), PRIMARY
KEY(id));
```

Find out the structure of the table

```
## you have to connect to db first with
## mysql
## within mysql>
DESCRIBE people
SHOW CREATE TABLE people
-- or : if you want to know more --
SELECT * from INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='people' AND
TABLE_SCHEMA='training' \G
```

Show indexes

```
SHOW INDEX FROM actor
SHOW INDEXES FROM ACTOR
```

Change table (Add field)

```
--- We want to add a field before name
--- IMPORTANT: BEFORE does not exist
ALTER TABLE people ADD first_name VARCHAR(10) AFTER id;

ALTER TABLE schulungen ADD seats TINYINT unsigned DEFAULT 1, ADD price DECIMAL(6,2);
ALTER TABLE schulungen ADD (room TINYINT unsigned DEFAULT 1,  discount DECIMAL(6,2));
```

Modify a field in table (Change property)

```
ALTER TABLE people
    MODIFY COLUMN first_name VARCHAR(20);
```

Drop a field from the table

```
ALTER TABLE people ADD middle_name VARCHAR(25) BEFORE name;
DESCRIBE people;
ALTER TABLE people DROP COLUMN middle_name;
```

```
## More Examples
--
ALTER TABLE actor ADD in_rente BOOLEAN default true
INSERT INTO actor (first_name,last_name,in_rente) values ('Jochen','Metzger',false)
-- Wieder loswerden
ALTER TABLE actor DROP in_rente;
## add and drop in once command
ALTER TABLE actor ADD in_rente2 BOOLEAN default true, DROP in_rente;
```

Deleting table data (truncate)

```
USE sakila
-- Create table based on other table
CREATE TABLE actorcopy as SELECT * FROM actor;
-- Fields ?
```

```
SELECT * FROM actorcopy;
-- Empty it
TRUNCATE TABLE actorcopy;
-- Emptry ?
SELECT COUNT(*) FROM actorcopy;
```

Delete table data (with delete)

Explanation

- Do not use delete when you want to use data of complete table
  - truncate is quicker in this case.
- DELETE FROM ... WHERE ... does a SELECT first

Example

```
USE sakila
CREATE TABLE actorbackup AS SELECT * FROM actor;
SELECT COUNT(*) FROM actorbackup;
DELETE FROM actorbackup WHERE actor_id > 100;
SELECT COUNT(*) FROM actorbackup;
```

Delete complete table

```
USE sakila
DROP TABLE actorbackup;
```

Events

- https://www.mysqltutorial.org/mysql-triggers/working-mysql-scheduled-event/

Views

Walkthrough

```
CREATE VIEW `avorname` AS
    SELECT
        first_name AS vorname
    FROM
        actor;


select * from avorname;

## Abfrage
## das geht nicht -> weil view der Column nicht bekannt ist
select * from avorname where first_name like 'A%';

## So muss es sein
select * from avorname where vorname like 'A%';
```

Administration

Globale und Session Variablen (Server System Variables)

Find out with show and @@

```
mysql> show session variables like 'PERFORMANCE%schema';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| performance_schema | ON    |
+--------------------+-------+
1 row in set (0.00 sec)

mysql> select @@performance_schema;
+----------------------+
| @@performance_schema |
+----------------------+
|                    1 |
+----------------------+
1 row in set (0.00 sec)

mysql> select @@SESSION.performance_schema;
ERROR 1238 (HY000): Variable 'performance_schema' is a GLOBAL variable
mysql> select @@performance_schema;
+----------------------+
| @@performance_schema |
+----------------------+
|                    1 |
+----------------------+
1 row in set (0.00 sec)

mysql> select @@GLOBAL.long_query_time;
+---------------------------+
| @@GLOBAL.long_query_time |
+---------------------------+
|                10.000000 |
+---------------------------+
1 row in set (0.00 sec)

mysql> select @@SESSION.long_query_time;
+---------------------------+
| @@SESSION.long_query_time |
+---------------------------+
|                 10.000000 |
+---------------------------+
1 row in set (0.00 sec)

mysql> set SESSION long_query_time=0.000001
    -> ;
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> select @@SESSION.long_query_time;
+----------------------------+
| @@SESSION.long_query_time |
+----------------------------+
|                   0.000001 |
+----------------------------+
1 row in set (0.00 sec)

mysql> select @@GLOBAL.long_query_time;
+----------------------------+
| @@GLOBAL.long_query_time |
+----------------------------+
|                 10.000000 |
+----------------------------+
1 row in set (0.00 sec)

mysql>
```

SET PERSISTENT

```
## Set variable to be use also after restart of mysql-server
SET PERSIST long_query_time = 0.000001

## will we in
C:\ProgramData\MySQL\MySQL Server 8.0\Data\mysql-auto.cnf
## <- as json
## loaded after my.ini
```

Get GLOBAL/SESSION variable directly from performance_schema (starting from MySQL 8)

```
use performance_schema
select * from global_variables;
select * from session_variables

## or Alternative is (without use):
use sakila;
select * from performance_schema.global_variables
select * from performance_schema.session_variables
```

Global and Session Status

What for ?

```
## Counts a number of values, like Com_select (how many selects)
## - since the server runs:
show global variables like 'Com_select';
## - in session && since last flush in session
show variables like 'Com_select'
```

flush status

```
## flushes session status
## only values that are specific in session, like Com_select
show variables like 'Com_select';
flush status;
show variables like 'Com_select';
```

Error-log

```
show variables like 'log_error';
+---------------+--------------------------+
| Variable_name | Value                    |
+---------------+--------------------------+
| log_error     | /var/log/mysql/error.log |
+---------------+--------------------------+
1 row in set (0.00 sec)
```

Slow Query Log

Walkthrough

```
mysql> show variables like '%slow%';
+------------------------------+-------------------------------------+
| Variable_name                | Value                               |
+------------------------------+-------------------------------------+
| log_slow_admin_statements    | OFF                                 |
| log_slow_extra               | OFF                                 |
| log_slow_replica_statements | OFF                                 |
| log_slow_slave_statements    | OFF                                 |
| slow_launch_time             | 2                                   |
| slow_query_log               | OFF                                 |
| slow_query_log_file          | /var/lib/mysql-data/mysql2-slow.log |
+------------------------------+-------------------------------------+
7 rows in set (0.01 sec)

mysql> show variables like '%long%';
+-----------------------------------------------------------+-----------+
| Variable_name                                             | Value     |
+-----------------------------------------------------------+-----------+
| long_query_time                                           | 10.000000 |
| performance_schema_events_stages_history_long_size        | 10000     |
| performance_schema_events_statements_history_long_size    | 10000     |
| performance_schema_events_transactions_history_long_size | 10000     |
| performance_schema_events_waits_history_long_size         | 10000     |
+-----------------------------------------------------------+-----------+
5 rows in set (0.00 sec)

mysql> set slow_query_log = on;
ERROR 1229 (HY000): Variable 'slow_query_log' is a GLOBAL variable and should be set
with SET GLOBAL
```

```
mysql> set global slow_query_log = on;
Query OK, 0 rows affected (0.00 sec)

mysql> set global long_query_time = 0.000001;
Query OK, 0 rows affected (0.00 sec)

mysql> show session variables like 'long_query_time';
+-----------------+-----------+
| Variable_name   | Value     |
+-----------------+-----------+
| long_query_time | 10.000000 |
+-----------------+-----------+
1 row in set (0.00 sec)

mysql> show global variables like 'long_query_time';
+-----------------+----------+
| Variable_name   | Value    |
+-----------------+----------+
| long_query_time | 0.000001 |
+-----------------+----------+
1 row in set (0.01 sec)

mysql> quit
Bye
root@mysql2:/etc/mysql/mysql.conf.d# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.27-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, 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> show session variables like 'long_query_time';
+-----------------+----------+
| Variable_name   | Value    |
+-----------------+----------+
| long_query_time | 0.000001 |
+-----------------+----------+
1 row in set (0.00 sec)
```

MySQL - Client - Tools (most important)

- mysql (mysql-client)
- mysqldump (backup of data)
- mysqlbinlog (read mysqlbinlog - files, because they are in binary format)

Manage max_connections

Max Connections default

```
select @@max_connections;
```

Error like

```
Too many connections

### increase or debug first by using status
show status like '%max%conn%';
## Variable_name    Value
Connection_errors_max_connections    16
Max_used_connections    3
Max_used_connections_time    2021-11-30 11:01:37
```

Change in config

```
## my.ini
[mysqld]
## or even more
max_connections = 151

## restart server
net stop MySQL80
net start MySQL80
```

Backup

Backup mit mysqldump - best practices

Useful options for PIT (before MySQL 8.0.27)

```
## —quick not needed, because included in —opt which is enabled by default

## on local systems using socket, there are no huge benefits concerning --compress
## when you dump over the network use it for sure
mysqldump -uroot -p --all-databases --single-transaction --master-data=2 --routines --
events --flush-logs --compress > /usr/src/all-databases.sql;
```

Same, but MySQL >= 8.0.27

```
mysqldump -uroot -p --all-databases --single-transaction --source-data=2 --routines --
events --flush-logs --compress > /usr/src/all-databases.sql;
```

With PIT_Recovery you can use --delete-master-logs

- All logs before flushing will be deleted

```
mysqldump --all-databases --single-transaction --gtid --master-data=2 --routines --
events --flush-logs --compress --delete-master-logs > /usr/src/all-databases.sql;
```

Version with zipping

```
mysqldump —-all-databases —-single-transaction —-gtid —-master-data=2 —-routines
--events —-flush-logs --compress | gzip > /usr/src/all-databases.sql.gz
```

Performance Test mysqldump (1.7 Million rows in contributions)

```
date; mysqldump --all-databases --single-transaction --gtid --master-data=2 --routines
--events --flush-logs --compress > /usr/src/all-databases.sql; date
Mi 20. Jan 09:40:44 CET 2021
Mi 20. Jan 09:41:55 CET 2021
```

Seperated sql-structure files and data-txt files including master-data for a specific database

```
 # backups needs to be writeable for mysql
 mkdir /backups
 chmod 777 /backups
 chown mysql:mysql /backups
 mysqldump --tab=/backups contributions
 mysqldump --tab=/backups --master-data=2 contributions
 mysqldump --tab=/backups --master-data=2 contributions > /backups/master-data.tx
```

mysqldump through the air

```
mysqldump --all-databases --single-transaction --source-data=2 --routines --events --
flush-logs --delete-source-logs -uroot -p | mysql -uroot -p --port=3308
```

Backups PIT (Point-In-Time recovery)

Meta - Weg

```
## Es ist wichtig, diese Reihenfolge
1) Spielen ein dump aus:

## --delete-source-logs nur wenn keine master-slave-Replikation
mysqldump --events --routines --flush-logs --source-data=2 --delete-source-logs --all-
databases > /usr/src/all-databases

2) Machen wir Änderungen in der sakral

use sakila
insert into actor (last_name,first_name) values (‚Hans‘,’Metzger’);
insert into actor (last_name,first_name) values (‚Hansi‘,’Metzgerei’);
delete from actor where id > 200;

——
```

```
3) Recovery vor Delete
1. Rausfinden wann der Fehler aufgetreten ist.
cd /var/lib/mysql
mysqlbinlog -vv bin-log.000005

2. Einschränken des binlogs und in recovery.sql ausspielen
## bei mehreren binlogs im Zeitraum bitte alle angeben
mysqlbinlog -vv --start-position=156 --stop-position=462 bin-log.000005 >
/usr/src/recovery.sql


3. Vollständigen Dump einspielen
mysql < /usr/src/all-databases.sql

4. recovery.sql einspielen
mysql < /src/src/recovery.sql

5. Überprüfen, ob die beiden Datensätze wieder da sind .
mysql> use sakila; select * from actor;
```

Backup und Wiederherstellen in neuer Datenbank

```
## using --databases sakila  instead does not work here
mysqldump --events --routines sakila > /usr/src/sakila.sql
cd /usr/src

## Version 1
## echo "create schema sakilatraining" | mysql

## Version 2 - works also on Windows
mysql -e 'create schema sakilatraining'
mysql sakilatraining < sakila.sql
```

mysqldump mit asynchroner Verschlüsselung

```
## Asynchrones Schlüsselpaar erstellen
openssl req -x509 -nodes -newkey rsa:2048 -keyout mysqldump-key.priv.pem -out
mysqldump-key.pub.pem

## Öffentlichen Schlüssel Verwenden zum Verschlüsseln
mysqldump --routines --events --triggers --all-databases | openssl smime -encrypt -
binary -text -aes256 -out database.sql.enc -outform DER mysqldump-key.pub.pem

## Entschlüsseln
openssl smime -decrypt -in database.sql.enc -binary -inform DEM -inkey mysqldump-
key.priv.pem -out mysql-backup.sql
mysql < mysql-backup.sql
```

mydumper und myloader

- https://github.com/maxbube/mydumper

Sicherheit

Absichern von Server/Client mit ssl

Teil 1: 1-Weg-Sicherheit (nur auf Server validiert)

```
Bei MySQL 8 werden Zertfikate in der Regel bereits erstellt.
Ob ssl funktioniert können wir mit

mysql>show variables like '%HAVE_SSL%';

## Es funktioniert bereits, allerdings mit den automatisch
## erstellten Zertifikaten
```

Herausfinden, ob SSL verwendet wird

```
## auf client auf dem mysql-server
mysql
mysql>status
SSL:            Not in use
Connection:        Localhost via UNIX socket
```

Bitte das nicht verwenden, weil man damit nicht den Common Name setzen kann

```
sudo mysql_ssl_rsa_setup --uid=mysql
```

CA (Certificate Authority) und Server-Key erstellen

```
## On Server - create ca and certificates
mkdir -p /etc/mysql/ssl
cd /etc/mysql/ssl

## create ca.
openssl genrsa 4096 > ca-key.pem

## create ca-certificate
## Common Name: MariaDB CA
openssl req -new -x509 -nodes -days 365 -key ca-key.pem -out ca-cert.pem

## create server-cert
## Common Name: MariaDB Server
## Password: --- leave empty ----
openssl req -newkey rsa:2048 -days 365 -nodes -keyout server-key.pem -out server-
req.pem

## Next process the rsa - key
openssl rsa -in server-key.pem -out server-key.pem

## Now sign the key
openssl x509 -req -in server-req.pem -days 365 -CA ca-cert.pem -CAkey ca-key.pem -
set_serial 01 -out server-cert.pem
```

Zertifikate validieren

```
openssl verify -CAfile ca-cert.pem server-cert.pem
```

Configure Server

```
## create file
## /etc/mysql/mysql.cnf.d/mysqld.cnf
[mysqld]
ssl-ca=/etc/mysql/ssl/ca-cert.pem
ssl-cert=/etc/mysql/ssl/server-cert.pem
ssl-key=/etc/mysql/ssl/server-key.pem
### Set up TLS version here. For example TLS version 1.2 and 1.3 ##
tls_version = TLSv1.2,TLSv1.3

## Set ownership
chown -vR mysql:mysql /etc/mysql/ssl/
```

Restart and check for errors

```
systemctl restart mysql
journalctl -u mysql
```

Externen user auf server einrichten

```
## Einloggen in mysql-client als root

mysql> create user ext@'%' identified by 'P@ssw0rd';
Query OK, 0 rows affected (0.01 sec)

mysql> grant all on sakila.* to ext@'%';
Query OK, 0 rows affected (0.00 sec)

mysql> alter user ext@'%' REQUIRE SSL;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from user where user = 'ext' \G

ssl_type: ANY
ssl_cipher: 0x
x509_issuer: 0x
x509_subject: 0x
```

Test on Client (1. Versuch)

```
## Er verbindet sich per SSL
## Zertifikatsüberprüfung findet nur auf SERVER statt
mysql -uext -p -h<ip-des-servers>
mysql> status
mysql> exit
## Wir probieren es ohne SSL
```

```
mysql -uext -p -h<ip-des-servers> --ssl-mode=DISABLED
## Trotz richtigem Passwort
Enter password:
ERROR 1045 (28000): Access denied for user 'ext'@'139.59.215.179' (using password:
YES)
```

Client verpflichten ein eigenes Zertifikat zu haben

```
## auf server als root
mysql>ALTER USER ext@'%' REQUIRE X509
```

On Client - fails because of missing client certificate

```
mysql -uext -p -h159.223.23.99
Enter password:
ERROR 1045 (28000): Access denied for user 'ext'@'139.59.215.179' (using password:
YES)
```

Teil 2: 2-Weg-Sicherheit (auf Server und Client validiert)

Client - Zertifikate auf Server erstellen

- Wir verwenden die gleiche CA wie beim Server

```
## auf dem Server
cd /etc/mysql/ssl
## Bitte Common-Name: MariaDB Client
openssl req -newkey rsa:2048 -days 365 -nodes -keyout client-key.pem -out client-
req.pem

## process RSA - Key
openssl rsa -in client-key.pem -out client-key.pem

## sign certficate with CA
openssl x509 -req -in client-req.pem -days 365 -CA ca-cert.pem -CAkey ca-key.pem -
set_serial 01 -out client-cert.pem
```

Client - Zertifikate validieren

```
openssl verify -CAfile ca-cert.pem client-cert.pem
```

Zertifikate für Client zusammenpacken

```
mkdir cl-certs; cp -a client* cl-certs; cp -a ca-cert.pem cl-certs ; tar cvfz cl-
certs.tar.gz cl-certs
```

Zertifikate auf Client transferieren

```
scp cl-certs.tar.gz 11trainingdo@<ip-des-clients>:/tmp
```

Zertifikate einrichten

```
## auf client
mv /tmp/cl-certs.tar.gz /etc/mysql/
cd /etc/mysql; tar xzvf cl-certs.tar.gz

cd /etc/mysql/cl-certs
ls -la


cd /etc/mysql/conf.d
vi mysql.cnf
[mysql]
ssl-ca=/etc/mysql/cl-certs/ca-cert.pem
ssl-cert=/etc/mysql/cl-certs/client-cert.pem
ssl-key=/etc/mysql/cl-certs/client-key.pem
```

Zertifikate testen

```
## Auf Server überprüfen dass X509 für user eingestellt ist
select user,ssl_type from mysql.user where user='ext'

## Auf Client zum server connecten
## Sollte die Verbindung nicht klappen stimmt auf dem
## Client etwas mit der Einrichtung nicht
mysql -uext -p -h<ip-des-mysql-servers>
mysql> status
```

Ref

- https://dev.mysql.com/doc/refman/8.0/en/alter-user.html```

Prüfen ob socket verwendet, bei lokalem System

Voraussetzung

```
Linux - System
Applikation und Datenbank-Server sind auf gleichen Virtuellen bzw. Physischen Server
```

Testfolge

```
lsof -i
localhost:mysql
```

mysql_secure_installation - validate plugin aktivieren

Sicherstellen, dass die Komponente Validate als Passwort-Mechanismus aktiviert wird

```
mysql_secure_installation

und keine root-benutzer von extern erlauben
```

```
mysql>
select * from mysql.user where user='root' and host != 'localhost'
    -> ;
Empty set (0.00 sec)
```

general log deaktivieren

Warum ?

- Wird sehr schnell, sehr groß
- Schlecht für die Performance

Überprüfung ?

```
select @@general_log
## sollte auf 0 stehen
show variables like 'general_log'
OFF
```

plugin vs. components

Components

- Abgeschlossene Einheiten
- MySQL-Server ist ein Komponente
- Eine weitere Komponenten kann geschrieben werden.
- Diese kommunziert über einen Service mit der anderen Komponenten

Plugins

- Server, stellt eine API / bzw. verschiedene bereit
- Auf dieses greift das Plugin dann zu
- Alles innerhalb der Komponenten MySQL-Server
- Oftmals schlecht implementiert
    - Eigentlich respektiv auf bestimmte apis
    - In der Realität, scope ist oft auf alle api
- Plugin kann alles auslesen

Vorteile von Komponentens

- Keine Endung mehr beim Laden notwendig

User Passwort-Länge und Passwort-Ablauf

Passwort-Ablauf pro User setzten

```
## Passwort läuft nach 60 Tagen ab und muss neu gesetzt werden
ALTER USER training@localhost PASSWORD EXPIRE INTERVAL 60 day;
```

- https://dev.mysql.com/doc/refman/8.0/en/password-management.html

Läßt sich eine Passwort - Länge pro User festlegen ?

```
Nein.
Nur auf Server-Ebene für alle Benutzer möglich (über Validation Komponente)
```

- https://dev.mysql.com/doc/refman/8.0/en/validate-password.html

Trigger ohne Super-Rechte anlegen

Warum ist es so ?

- Trigger können nicht auf DETERMINISTIC gesetzt werden.
- Wenn ein Trigger nicht-deterministisch ist, kann es zu Problemen kommen
- In diesem Fall kann es beim BINLOG_FORMAT=STATEMENT zu Problemen beim Slave kommen

Test auf MySQL 8.0.27 BINLOG_FORMAT = ROW

```
## User training@localhost mit folgenden Rechten
show grants;
+-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------+
| Grants for training@localhost
|
+-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------+
| GRANT USAGE ON *.* TO `training`@`localhost`
|
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, REFERENCES, INDEX, ALTER, CREATE
TEMPORARY TABLES, LOCK TABLES, EXECUTE, CREATE VIEW, SHOW VIEW, CREATE ROUTINE, ALTER
ROUTINE, EVENT, TRIGGER ON `sakila`.* TO `training`@`localhost` |
+-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------+
2 rows in set (0.01 sec)

## Test - Case
use sakila;

## aus bug-report
mysql> CREATE TABLE t1 ( a int );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TRIGGER g1 BEFORE INSERT ON t1 FOR EACH ROW SET new.a=new.a+1;
ERROR 1419 (HY000): You do not have the SUPER privilege and binary logging is enabled
(you *might* want to use the less safe log_bin_trust_function_creators variable)
```

(Dirty-)Fix

```
## Either set it in the config or as SUPER-privileges user:
/etc/mysql/mysql.conf.d/mysqld.cnf
log-bin-trust-function-creators = 1
systemctl restart mysql

## now login as unprivileged (NON SUPERUSER PERMS) and try again
## on localhost
mysql -utraining -p
```

```
use sakila
mysql> CREATE TABLE if not exists t1 ( a int );
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> DROP TRIGGER IF EXISTS g1;
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> CREATE TRIGGER g1 BEFORE INSERT ON t1 FOR EACH ROW SET new.a=new.a+1;
Query OK, 0 rows affected (0.00 sec)
```

Refs:

- https://bugs.mysql.com/bug.php?id=39489

Sicherheit (CIS)

1.1 Place Databases on Non-System Partitions

Überprüfen, wo das datadir liegt

```
mysql> select @@datadir;
+----------------------+
| @@datadir            |
+----------------------+
| /var/lib/mysql-data/ |
+----------------------+
1 row in set (0.00 sec)

mysql> show variables like 'datadir';
+---------------+----------------------+
| Variable_name | Value                |
+---------------+----------------------+
| datadir       | /var/lib/mysql-data/ |
+---------------+----------------------+
1 row in set (0.01 sec)
```

Walkthrough

```
## /etc/apparmod.d/
vi usr.sbin.mysqld
## ---> change these lines
## Allow data dir access
##  /var/lib/mysql/ r,
##  /var/lib/mysql/** rwk,
  /var/lib/mysql-data/ r,
  /var/lib/mysql-data/** rwk,
### <-----

systemctl stop mysql
systemctl restart apparmor
systemctl status apparmor
aa-status
```

```
## Change config of mysql
## datadir
cd /etc/mysql/mysql.conf.d/
vi mysqld.cnf
## change datadir to /var/lib/mysql-data # on seperate partition
datadir=/var/lib/mysql-data

cd /var/lib
cp -a mysql mysql-data
systemctl restart mysql

## Bei Erfolg ist das Datadir jetzt geändert
mysql>
show variables like 'datadir';
```

Debuggen bei Problemen

```
journalctl -u mysql -e
/var/log/mysql/error_log
```

systemctl stop mysql

1.2 Use dedicated Least Privileged Account

Check

```
## simple
ps aux | grep mysql

## sophistiscated
ps aux | head -n 1 && ps aux | grep mysql | grep -v grep
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mysql       18341  0.5 42.4 1842172 426204 ?      Ssl  14:22   0:01 /usr/sbin/mysqld
```

Tools

Testdatenbank Sakila installieren

```
cd /usr/src
wget https://downloads.mysql.com/docs/sakila-db.tar.gz
tar xvf sakila-db.tar.gz
cd sakila-db/
ls -la
mysql < sakila-schema.sql
mysql < sakila-data.sql
```

Performance - Analyse

Tool pt-query-digest

Authentifizierung / User-Management

Für User altes Password-Verfahren mysql_native_password verwenden in MySQL 8

```
create user scanner@localhost identified with mysql_native_password by 'Passw0rd';
```

```
## even possible without changing password
alter user training3@localhost identified with caching_sha2_password
```

Wildcard-Rechte für Datenbank

Why ?

- Allow a user to connect to all databases starting with prod_
- Also those, that are present yet .

Walktrough

```
## as root
mysql> create user schulung@localhost identified by 'my_super_secret_pass';
mysql> -- give permission to all databases starting with prod_
mysql> grant all on `prod\_%`.* to schulung@localhost
mysql> create schema prod_db1; create schema prod_db2;
mysql> use prod_db1; create table data (id int);
mysql> use prod_db2; create table data (id int);

## as user schulung@localhost
## connect and find out if you cann access db's
## mysql -uschulung -p




### Rollen


### Konzept

  * Rollen

### Walkthrough
```

Rolle anlegen

mysql> CREATE ROLE sakiladb

Berechtigungen der Rolle zuordnen / alle Berechtigungen für sakila

mysql> GRANT ALL ON sakila.* TO sakiladb

Nutzer anlegen

mysql> CREATE USER roleuser@localhost identified by 'P@ssw0rd';

Die Rolle dem Nutzer zugeordnet

mysql> GRANT sakiladb TO roleuser@localhost

Die Standardrolle festlegen, wenn er sich einloggt

SET DEFAULT ROLE ALL TO roleuser@localhost

```
### Weitere Rolle für den Nutzer
```

CREATE ROLE mysqldb; GRANT ALL ON mysql.* TO mysqldb; GRANT mysqldb TO roleuser@localhost;

Important. SET DEFAULT ROLE must be executed once

again to have mysqldb selected after login

SET DEFAULT ROLE ALL TO roleuser@localhost;

```
### Revoke a role from a user
```

REVOKE mysqldb FROM roleuser@localhost;

```
### Abgekürzte From
```

create user roleuser2@localhost identified by 'P@ssw0rd' DEFAULT ROLE sakiladb;

```
### Ref
  * https://www.mysqltutorial.org/mysql-roles/



## Replication

### Overview

![Overview Multi-Source-Replication](/images/multi-source-replication.jpg)

### Multi-Source-Replication
```

```
### Background

  * Aggregate multiple sources into one slave
  * Uses channels (FOR CHANNEL 'replicant-1')

### Walkthrough (replicant-1)
```

-> ON master/replicant:

1. create replication user

event better IP-Range instead of % -> 192.168.56.%

CREATE USER repl_multi@'%' identified by 'your_secret_pass' GRANT REPLICATION SLAVE ON . TO 'repl_multi'@'%'

2. test connection with that user

in our case on same server

explicitly host, because that's how we use it in CHANGE MASTER

mysql -urepl_multi -p -h127.0.0.1

3. Daten auf master ausspielen und master-data notieren

CHANGE MASTER steht relativ am Anfang der Datei

mysqldump --all-databases --single-transaction --source-data=2 --routines --events --flush-logs --delete-source-logs -uroot -p > all-databases.sql

-> ON slave/replica:

1. be sure, that server does not have same server_id // server uuid

--> Delete auto.cnf in datadir

-> change server_id in my.cnf in [mysqld] section to > 1

must be unique across all servers in master/slave replications network

e.g.

server_id = 2

2. Restart server (in our case mysql2 is the replica)

net stop mysql2 net start mysql2

3. Import data into slave/replica from master

port of our replica is 3308

mysql -uroot -p --port=3308 -h 127.0.0.1 < all-databases.sql

4. Construct change master -> sql command

with master_pos, master_log_file from dump

CHANGE MASTER is the same as CHANGE REPLICATION SOURCE

CHANGE REPLICATION SOURCE TO SOURCE_HOST='127.0.0.1', SOURCE_USER='repl_multi', SOURCE_PASSWORD='password', SOURCE_LOG_FILE='binlog.000026', SOURCE_LOG_POS=156 FOR CHANNEL 'replicant-1';

5. start replica

START REPLICA FOR CHANNEL 'replicant-1'

6. Check on slave if you succeeded

show replica status;

or

show slave status;

look for slave_io_running -> YES

look for slave_sql_running -> YES

If not look for errors within the output

```
### Walkthrough (replicant-2)
```

1. neues Verzeicnnis erstellen
2. richtige Rechte geben. -> NETWORK SERVICES Trainer08 IT-Schulungen11:19
3. my.ini kopieren aus bisheriger Instanz
4. my.ini anpassen: port, datadir, server-id
5. cmd.exe als Admin ausführen
6. in das bin -verzeichnis wechseln cd C:\Program Files\MySQL\MySQL Server 8.0\bin
7. Datenverzeichnis initial erstellen mysqld --defaults-file=C:\ProgramData\MySQL\MYSQL-basic\my.ini --initialize --console
8. password kopieren / temporäres IXahAsElk4)V
9. Service installieren

10. mit Server verbinden (konsole) und password setzen mysql -uroot -p --port=3310 mysql> alter user root@localhost identified by 'password'
11. show master status; Daten notieren.
12. Replications-User einrichten create user multi@'%' identified by 'password'; grant slave replication on . to multi@'%';
13. Test connection with multi user show grants
14. On slave CHANGE REPLICATION SOURCE TO SOURCE_HOST='127.0.0.1', SOURCE_USER='multi', SOURCE_PORT=3310, SOURCE_PASSWORD='password', SOURCE_LOG_FILE='binlog.000002', SOURCE_LOG_POS=10403 FOR CHANNEL 'replicant-2';

CHANGE REPLICATION FILTER REPLICATE_DO_DB = (dauerversuche) FOR CHANNEL 'replicant-2'START REPLICA FOR CHANNEL 'replicant-2'; START REPLICA FOR CHANNEL 'replicant-2'; SHOW REPLICA STATUS;

```
### Show the state of replication in performance schema
```

all slaves

show slave status;

specific slave

show slave status for channel 'replicant1';

more information in performance_schema

use performance_schema; select * from performance_schema.replication_connection_status \G

```
### Binlog format


### What ?
```

The binlog format determines how the data is written into the binary log

```
### Which options ?

  * STATEMENT (first one in MySQL over development time)
  * ROW
  * MIXED

### STATEMEMT

  * The exact statement executed on replicant (master) will be written to binlog
```

```
### ROW

  * If you execute an update, it will not write the update itself but the results
  * Example: update last_name = 'Testuser' from sakila.actor where actor > 100 (200
dataset)
  * In binlog systems writes 100 dataset, each dataset with the exact data for that
row

### MIXED

  * Systems decided if the sql was determenistic
  * If not -> uses row
  * if yes -> uses Statement

### DEFAULT

   * ROW (safest option)

### Change-Replication-Filter


### Why ?

  * Allows to ignore db or only replicate specific db's
  * also possible: different for channel

### Example
```

CHANGE REPLICATION FILTER REPLICATE_DO_DB = (dauertest) FOR CHANNEL replicant-1; CHANGE REPLICATION FILTER REPLICATE_DO_DB = (d1);

```
### Start and Stop the slave (important)
```

STOP REPLICA SQL_THREAD FOR CHANNEL 'replicant-1'; CHANGE REPLICATION FILTER REPLICATE_DO_DB = (dauerversuche) FOR CHANNEL 'replicant-1'; START REPLICA SQL_THREAD FOR CHANNEL 'replicant-1';

```
### Rausnehmen
```

STOP REPLICA SQL_THREAD FOR CHANNEL 'replicant-1'; CHANGE REPLICATION FILTER REPLICATE_DO_DB = () FOR CHANNEL 'replicant-1'; START REPLICA SQL_THREAD FOR CHANNEL 'replicant-1';

```
### Reference

  * https://dev.mysql.com/doc/refman/8.0/en/change-replication-filter.html

### What to do on error (sql-thread not running)


### Situation
```

Auf dem slave/replica

show slave status;

Wir sehen

slave_sql_running -> no

And we will see an

and it show us at which place the error occured

binlog -> and binlog_pos

Last_SQL_Error

```
### Troubleshoot
```

We look into the binlog in the master for this channel

and have a look what it did there

cmd.exe

cd

mysqlbinlog -vv bin-log.0000026 | more

So now we look at the command which should have executed

and also look, if the next would be o.k.

Example

Command 1: CREATE DATABASE dauertest

.. but this database already exists on slave

Command 2: SET GIT... would be o.k.

Go back to slave

STOP SLAVE FOR CHANNEL 'your-channel'

so we skip one binlog / relaylog -entry

SET GLOBAL sql_slave_skip_counter = 1 START SLAVE FOR CHANNEL 'your-channel'

Doublecheck if it is running

SHOW REPLICA STATUS

now -> should be

io_thread_running = YES

sql_thread_running = YES

```
### IMPORTANT
```

If it struggles with the next error, DO NOT PROCEED LIKE so. But re-setup the slave

-> Because slave it completely messed up <--

```
## Upgrade

### Upgrade von MySQL 5.7 -> 8


### Walkthrough (Teil 1)
```

Download repo - apt install package on server

cd /usr/src wget https://dev.mysql.com/downloads/repo/apt/ -> mysql-apt.....

dpkg -i mysql-apt-config_0.8.20-1_all.deb

all settings can be like, then select OK

Repostände lokal updaten

apt update

apt install mysql-shell

mysqlsh> JS \c root@localhost

Probleme mit socket evtl durch unterschiedliche Paketherkünfte (MySQL 5.7 -> MySQL 8)

MySQL 5.7. kam von Ubuntu und verwendete einen anderen Socket MySQL 8 verwendet den socket /var/lib/mysql/mysql.sock

```
### Anpassen des Socket in Server und Client - Teil 2
```

cd /etc/mysql/mysql.conf.d/mysqld.cnf

socket geändert in

[mysqld] socket = /var/lib/mysql/mysql.sock

Server neu starten und prüfen ob sockt da ist im Verzeichnis

systemctl restart mysql /var/lib/mysql/

socket geändert für client - config

cd /etc/mysql/conf.d/mysql.cnf

mysql.conf

[mysql] socket = /var/lib/mysql/mysql.sock

Achtung auch für mysqldump setzten z.B. in Datei

client.cnf - neu erstellen

[client] socket = /var/lib/mysql/mysql.sock

```
### Test Script für MySQL Migration aufgerufen
```

mysqlsh

Vorne connection, hinten Options

JS> util.checkForServerUpgrade('root@localhost',{"configPath":"/etc/mysql/my.cnf"})

Dann Ausgabe sieht ungefähr so aus.

The MySQL server at localhost:33060, version 5.7.31-log - MySQL Community

Server (GPL), will now be checked for compatibility issues for upgrade to MySQL

8.0.21...

1. Usage of old temporal type

   No issues found

.....

```
### Ausgabe überprüft, was muss evtl geändert, berücksichtigt werden.
```

z.B. AUTO_CREATE_NO_USER - unkritisch, da in MySQL 8 per default der Fall ist. (Es wird bei grants keine user angelegt, wenn diese nicht existieren)

```
### Sicherung der Datenbank VOR !! Update
```

mysqldump --all-databases --routines --events > /usr/src/all-database.sql

Wenn 0 ausgabe, dann ist das Script erfolgreich durchgelaufen

echo $?

Zur Sicherheit noch letzte in Dump anschauen

Hier muss Dump completed stehen

```
### Server stoppen und deinstallieren
```

systemctl stop mysql apt remove mysql-server-5.7

alte Abhängigkeiten, die nicht mehr benötigt werden, werden gelöscht

apt autoremove

```
### Neuen Server installieren und Fehler bereinigen
```

mysql-server ist in der Regel die neueste, zur Sicherheit nochmal checken

mit apt search mysql-server

apt install mysql-server

mysqld.conf behalten !!

Wenn server nicht starten Fehler bereinigen

Analysieren

/var/log/mysql/error.log

mysqld.conf entsprechend anpassen

danach start probieren, so lange bis es geht !! (fehlerbereinigung -> starten -> fehlerber....)

systemctl start mysqld

Upgrade erfolgt beim Starten in Place sowohl in Installationspackage als auch tar.gz

```
### Refs:
```

```
  * https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-utilities-upgrade.html

## Windows

### Welchen Benutzer für den Service verwenden?


### LocalSystem is nicht gut !!
```

Zu viele Rechte

```
### Optimal wäre: LocalService
```

geht nur auf, wenn applikation und DB-Server auf gleichem Host

mysqld --install --local-service

mysqld --remove "ServiceNamen"

```
### 2. Alternative: NetworkService
```

Frage: Nimmt der installer diesen beider Installatio

```
### Refs:

  * https://www.netikus.net/documents/MySQLServerInstallation/index.html?
moresecurity.htm



## Tipps & Tricks

### Version von MySQL rausfinden
```

mysql> status; -- oder mysql> select version()

```
### Show Information_schema within MySQL Workbench
```

Edit -> Preferences -> SQL Editor and then check the box "Show Metadata and Internal Schemas"

```
### Set path in Windows for User to easily use mysql
```

Find path of binaries

e.g.

C:\Program Files\MySQL\MySQL Server 8.0\bin

Now in Search - Field enter

Umgebungsvariablen

Click on, Click Path and add new entry from above

```
### Security with outfile mysql
```

Generally, you can only write to a specific folder if secure_file_priv isset

Step1: Check setting in global server system variables

select @@secure_file_priv;

or

show variables like 'secure_file_priv';

mysql> select @@secure_file_priv; +---+ | @@secure_file_priv | +---+ | C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\ | +---+ 1 row in set (0.00 sec)

Step 2: than use exactly that path, but either with '/' or '\'

Version 1

Take whatever suffix you want ;o)

mysql>SELECT * from sakila.actor INTO OUTFILE 'C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\actor.txt';

Version 2

mysql>SELECT * from sakila.actor INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/actor.txt';

```
### Alternative
```

-r raw format without table

put -e expression in double quotes "

mysql -e "select * from actor;" -uroot -r -p sakila > test.sql

```
## Documentation

### Server System Variables - Reference

  * https://dev.mysql.com/doc/refman/8.0/en/server-system-variable-reference.html

### MySQL Performance Dokument - en

  * https://schulung.t3isp.de/documents/pdfs/mysql/mysql-performance.pdf
```