


Restore MySQL Master Slave Replication

Introduction


This page describes the process of rebuilding a Slave-DB & Replication. Since the MySQL Dump command locks the tables, there is no need to create it when there is no traffic on the machine. It can be done during operational hours. With --master-data the command mysqldump stores the correct position for inserting the replication on the slave server.

Prerequisites

 If the hard disk of the slave is full, please refer to the instructions on the bottom of this page

Step-By-Step Guide

General Advice

 jtel does not take responsibility for any mishaps which result in attempting to execute the procedure described on this page, and also does not advise attempting this procedure, if no previous experience with database operations is present. The minimum requirement is that you have basis knowledge about relational databases in general and the MySQL database in particular.

STOP SLAVE

Login to the Slave Database MySQL and stop the slave SQL. Leave MySQL again afterwards. Use the following commands for this:

```
mysql -uUSER -pPWD
STOP SLAVE;
QUIT;
```

Phase 1 - MySQL Dump

A MySQL Dump of the master database is now created. Perform the following steps to create a MySQL Dump and save it to the STORE:

mysqldump command

 The mysqldump command is different, depending on the jtel portal release, as well as the MySQL software release installed on the databases. All different options and how to find out which one to choose is specified below.

Master-Master Replication

 The mysqldump commands on this page can NOT be used to realign a master-master replication. Visit the following page for that description [Restore MySQL Master-Master Replication](#)

jtel Portal software release

The following commands are designed to be executed on the STORE as jtel user

```
# change to jtel user
su jtel
# Find out which software release is installed
cd /srv/jtel/shared/JTELCarrierPortal
git status

# If /srv/jtel/.. does not exist on the STORE, attempt this
cd /home/jtel/shared/JTELCarrierPortal
git status

# Expected output
release-stable/3.XX
```

Create Backup Directory

The following commands are designed to be executed on the STORE as jtel user



The following cd commands depend on the variable JT_DATE_TIME, which is set at the beginning of the next part. If the variable is not set, commands will fail.

```
# Create backup directory
JT_DATE_TIME=$(date +%F)
mkdir /srv/jtel/shared/backup/${JT_DATE_TIME}
# If /srv/jtel/.. does not exist on the load balancer, attempt this
JT_DATE_TIME=$(date +%F)
mkdir /home/jtel/shared/backup/${JT_DATE_TIME}
```

Create MySQL Dump

CAUTION - CREDENTIALS+IP-Adresses



Credentials and IP-Adresses need to be changed before the following mysqldump commands can be executed

To Check for MySQL Version, log in to the master database server and execute the following command

```
mysql --version
```

The following commands are designed to be executed on the STORE as jtel user

MySQL Dump - Until jtel Portal release 3.12

In versions 3.12, 3.14 and 3.15



If someone logs on to the portal while the dump is being pulled, it will go wrong. Enclosed a SQL query. If the time changes after executing the query, a login has taken place.

If this happens, the dump has to be pulled again and in the meantime it has to be permanently checked if a login has taken place. Only if this is not the case, the dump can be replicated error-free to the slave.

SELECT Max(dtAcidLoggedIn) FROM Users;

In versions 3.11 and below and version 3.16 this problem does not exist.

```
# change to jtel user
su jtel
# Change to backup directory
cd /srv/jtel/shared/backup/${JT_DATE_TIME}
# Create MySQL Dump
mysqldump -uUSER -pPWD -h<IP-Address-OR-Alias-GOOD-MASTER> --single-transaction --master-data=1 --databases JTELWeb JTELStats JTELLog --add-drop-database --add-drop-table --events --routines --triggers > /srv/jtel/shared/backup/${JT_DATE_TIME}/acd-dbm_<yyyymmdd>.sql
```

MySQL Dump - From jtel Portal release 3.12 until 3.31

```
# change to jtel user
su jtel
# Change to backup directory
cd /srv/jtel/shared/backup/${JT_DATE_TIME}
# Create MySQL Dump
mysqldump -uUSER -pPWD -h<IP-Address-OR-Alias-GOOD-MASTER> --single-transaction --master-data=1 --databases JTELWeb JTELStats JTELStats2 JTELLog --add-drop-database --add-drop-table --events --routines --triggers > /srv/jtel/shared/backup/${JT_DATE_TIME}/acd-dbm_<yyyymmdd>.sql
```

MySQL Dump - From Release 3.32

```
# change to jtel user
su jtel
# Change to backup directory
cd /srv/jtel/shared/backup/${JT_DATE_TIME}
# Create MySQL Dump
mysqldump -uUSER -pPWD -h<IP-Address-OR-Alias-GOOD-MASTER> --single-transaction --master-data=1 --databases JTELWeb JTELStats JTELStats2 JTELLog JTELCustomer --add-drop-database --add-drop-table --events --routines --triggers > /srv/jtel/shared/backup/${JT_DATE_TIME}/acd-dbm_<yyyymmdd>.sql
```

MySQL Dump - From MySQL 8.0.27

```
# change to jtel user
su jtel
# Change to backup directory
cd /srv/jtel/shared/backup/${JT_DATE_TIME}
# Create MySQL Dump
mysqldump -uUSER -pPWD -h<IP-Address-OR-Alias-GOOD-MASTER> --single-transaction --source-data --databases JTELWeb JTELStats JTELStats2 JTELLog --add-drop-database
--add-drop-table --events --routines --triggers > /srv/jtel/shared/backup/${JT_DATE_TIME}/acd-dbm_<yyyymmdd>.sql
```

Brief Explanation of the mysqldump command

- <Name LogFile> and <Position LogFile> are saved through --master-data or --source-data in the dump
- use only the JTEL databases '--databases'
- delete all databases before import '--add-drop-database'
- delete all tables of the databases before import '--add-drop-table'
- Implement all routines and procedures in the dump '--events --routines'
- Additionally the parameter --default_character_set utf8 can be used

Phase 2 - Import Dump on Slave Database

The following commands are designed to be run from the STORE as root user. After the source <acd-dbm_<yyyymmdd>.sql> is executed, the dump will start importing into the slave database. Once it is finished, the Slave SQL is started.

Brief explanation - tmux: tmux is a tool used to create sub-sessions within a ssh terminal. The tool is a good option for this specific operation, since the sub-session can be exited but the processes which were started will still be running in the background. This prevents failure by accidental loss of the ssh terminal, and provides the ability to view the MySQL process list to check the progress of the import

Tmux cheat sheet: <https://tmuxcheatsheet.com/>

Import Dump - tmux

```
# switch to root user
sudo -s
# start tmux sub-session
tmux new
# change to the directory where the dump file is located
JT_DATE_TIME=$(date +%F)
cd /srv/jtel/shared/backup/${JT_DATE_TIME}/
# copy the dump-filename
ls -als
# log in to mysql and import the dump
mysql -hacd-dbs -uroot -p<password>
source <acd-dbm_<yyyymmdd>.sql>;
```

Import Dump - No tmux

```
# switch to root user
sudo -s
# change to the directory where the dump file is located
JT_DATE_TIME=$(date +%F)
cd /srv/jtel/shared/backup/${JT_DATE_TIME}/
# copy the dump-filename
ls -als
# log in to mysql and import the dump
mysql -hacd-dbs -uroot -p<password>
source <acd-dbm_<yyyymmdd>.sql>;
```

Check status

You can check the progress by executing the following command on acd-dbs

```
watch 'mysql -uroot -p<password> -e "SHOW PROCESSLIST;"'
```

Start Slave - tmux

```
# If not attached to the sub-session anymore
tmux attach
# Start Slave SQL
START SLAVE;
# Check Slave Status
SHOW SLAVE STATUS \G
```

Start Slave - No tmux

```
# Start Slave SQL
START SLAVE;
# Check Slave Status
SHOW SLAVE STATUS \G
```

Slave Disk Full

There are several reasons why the slave disk can become full. Following is a summary of the different cases and further below is a detailed explanation:

Case

Temp directory full
Lots of "relay logs" available
ibdata files on the slave "very large"
Still no disk space > 20% free

tmp Directory Is Full

Introduction

Every time a query creates a tmp table, it is written to the temp directory, usually /tmp. This happens when the maximum size exceeds the maximum "in memory" table size. This is defined with the variables `tmp_table_size` aswell as `max_heap_table_size` .

The tables in /tmp are kept until the respective DB connection is closed or a DROP TEMPORARY TABLE is called. If the /tmp directory is full, it is likely that a DROP TEMPORARY TABLE is missing somewhere. This can also happen through customer queries to the DB.

The installation of **tmpwatch** creates help in a permanent way

Hints:

- On CentOS 6 the /tmp directory is then by default freed from files that have not been accessed for **> 10 days**
 - Circumstantially this may not be sufficient
- On CentOS 7 the /tmp directory is then freed by default from files that have not been accessed for > 1 day

See also <https://dev.mysql.com/doc/refman/5.6/en/internal-temporary-tables.html> for more information.

Procedure

1. Space must first be made available. Just mercilessly delete everything from /tmp.
2. If this provides enough space, then restart the mysql service first: **service mysql restart**
3. If there is enough space (at least about 20%), then proceed with replication as described above.

Lots of "relay logs" available

Introduction

MySQL first writes the relay logs from the master to a file. Once replication is interrupted, but the slave relay process continues to operate, the disk is filled by relay logs.

This step should be done in any case, especially before the next one (ibdata too large) to make room.

Procedure

The files for the database are usually located in **/var/lib/mysql**

If not, the location can be found in **/etc/my.cnf** The corresponding entry is **datadir=(pfad)**

Delete all relay logs:

```
cd /var/lib/mysql
rm mysql-relay-bin*
```

Restart MySQL service

```
service mysqld restart
```

If enough space is available (at least 20%) then proceed with the slave recovery as described above.

ibdata files on the slave "very large"

Introduction

The files for the database are usually located in **/var/lib/mysql**

Due to not clearly documented MySQL internals, the file **/var/lib/mysql/ibdata1** can be huge compared to the master database. To remedy this, you have to proceed a little more rigorously.

Procedure

If no less than 100% disk can be achieved by the steps above:

Unsubscribe the MySQL service from the autostart:

```
systemctl disable mysqld
```

Restart the computer with :

```
systemctl reboot now
```

Only works if less than 100% disk is reached (possibly after reboot above) then

Log on to the mysql server:

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

Drop all JTEL databases:

```
SET FOREIGN_KEY_CHECKS=0;
DROP DATABASE JTELLog;
DROP DATABASE JTELStats;
DROP DATABASE JTELWeb;
SET FOREIGN_KEY_CHECKS=1;
```

Press CTRL+C to return to the command line, and then

```
service mysqld stop
rm /var/lib/mysql/ibdata*
rm /var/lib/mysql/ib_log_*
```

Start MySQL Server, and enable it again if necessary:

```
service mysqld start
service mysqld enable
```

Check disk space, and proceed with Slave Restore as described above.

Still no disk space > 20% free

Introduction

The slave disk space is still full, more than 80% of the space on the drive is used by normal operation

Solution

In this case the slave is simply too small. The hard disk must be expanded (as with extending the STORE role, apply only to the logical volume where the MySQL database data resides). Then perform the steps again, if the disk is too small, then restore the slave as described above.

Or the slave is completely rebuilt with a larger plate.