MySQL

Introduction

> Check the **status** of **mysql-server.**

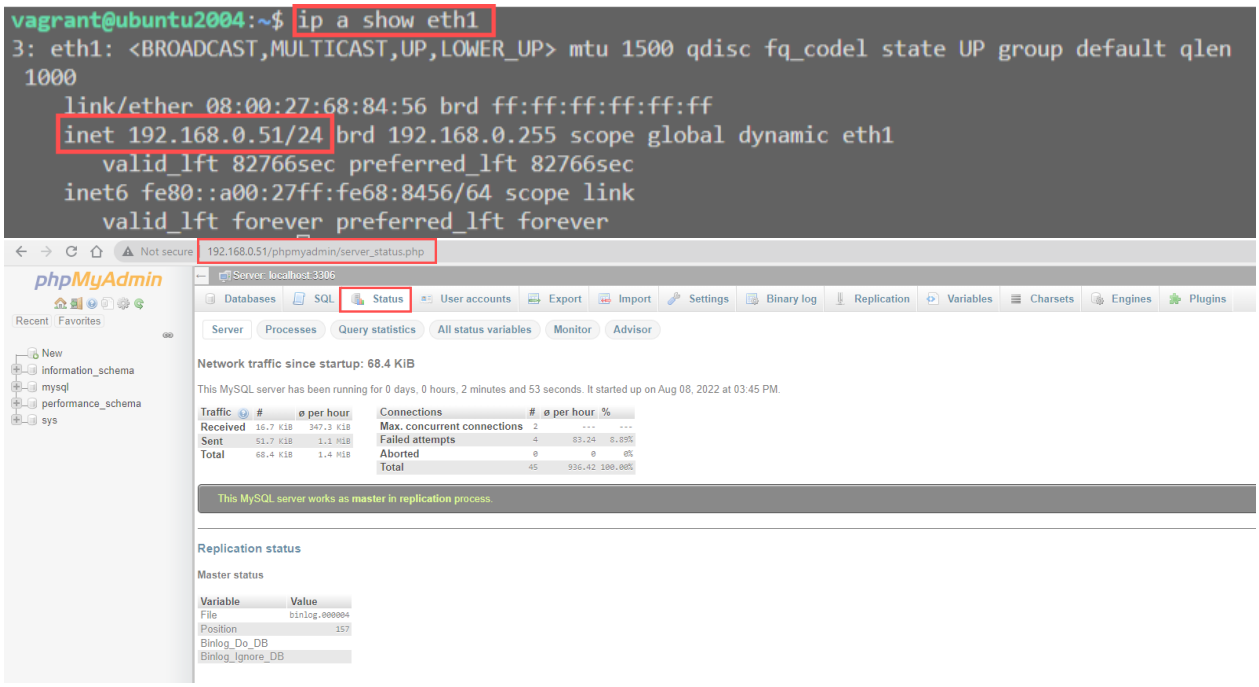
```
sudo service mysql status
```

```
-- Output --
● mysql.service - MySQL Community Server
     Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor
preset: enabled)
     Active: active (running) since Mon 2022-08-08 15:00:04 UTC; 44min ago
   Main PID: 7157 (mysqld)
     Status: "Server is operational"
```

> Try to **stop**, and **restart mysql-server**

```
sudo service mysql stop
sudo service mysql start
```

> Log in to phpMyAdmin and check the Status page



> Use the **CLI** to check all the **server variables** available

```
mysqladmin -u root -p extended-status
```

> Use the **man** command to check the available options for the **mysqladmin** command

```
man mysqladmin
```

> Check the **active threads**

```
mysqladmin -u root -p status
```

```
-- Output --
Uptime: 800  Threads: 2  Questions: 188  Slow queries: 0  Opens: 208  Flush
tables: 3  Open tables: 127  Queries per second avg: 0.235
```

Manage Users

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

> Check the users in the database.

```
SELECT user FROM mysql.user;
```

> Check the privileges for the users

```
show GRANTS
```

Create a new user

> - Username : marie
> - Password: password
> - Only allow from localhost

```
CREATE USER 'marie'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```

> PASSWORD ERROR

```
SHOW VARIABLES LIKE 'validate_password%';
```

```
SET GLOBAL validate_password.policy  = LOW;
```

.

Create a new database

> - Database name: marie
> - Marie should have all privileges on the Database 'marie'.

```
show databases;
```

```
CREATE DATABASE marie;
GRANT ALL PRIVILEGES ON marie.* TO 'marie'@'localhost';
```

> Log in with user 'marie' to verify.
>
> - Verify 'maire' can access the Database 'marie'

```
mysql -u marie -p
```

```
show databases;
```

> - Verify 'maire' can create a new Table inside the 'maire' Database

```
use marie
```

```
CREATE TABLE test (     id int, naam varchar(255));
```

> Verify the table has been created

```
show tables;
```

```
-- Output --
+-----------------+
| Tables_in_marie |
+-----------------+
| test            |
+-----------------+
```

> Insert some data into the table TEST

```
INSERT INTO test VALUES(1,"cédric");
SELECT * FROM test;
```

> Create a new user
>
> - Username : Guy
> - Password: password
> - only from localhost, no privileges.

```
mysql -u root -p

CREATE USER 'guy'@'localhost' IDENTIFIED BY 'password';
```

> Verify: log in with Guy and take a look at the database Guy can see

```
mysql -u guy -p

SHOW DATABASES;

-- Output --
+--------------------+
| Database           |
+--------------------+
| information_schema |
| performance_schema |
+--------------------+
```

> Allow Guy to only select and add data inside the marie database

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

```
    GRANT SELECT , INSERT ON marie.* TO 'guy'@'localhost';
```

> Verify

```
select * from test
```

```
INSERT into TEST VALUES("2", "guy");
```

> Allow Guy to only update the 'naam' field in the table test.

```
    GRANT UPDATE (`naam`) ON marie.test TO 'guy'@'localhost';
```

> Verify

```
UPDATE test
SET
    naam = 'josef'
WHERE
    id = 2;
```

> Allow marie to create databases

```
GRANT CREATE ON *.* TO 'marie'@'localhost';
```

Backups

> Download the test-database (zip) from GitHub

```
curl -LO https://github.com/datacharmer/test_db/archive/master.zip
```

> Unzip the file and install the test-database (https://github.com/datacharmer/test_db)

```
unzip master.zip
```

```
vagrant@ubuntu2004:~$ ls
master.zip  test_db-master

vagrant@ubuntu2004:~$ cd test_db-master/
vagrant@ubuntu2004:~/test_db-master$ ls
Changelog                   employees.sql          load_dept_emp.dump
load_salaries1.dump  load_titles.dump  sakila
test_employees_md5.sql
employees_partitioned_5.1.sql  images                 load_dept_manager.dump
load_salaries2.dump  objects.sql       show_elapsed.sql
test_employees_sha.sql
employees_partitioned.sql      load_departments.dump  load_employees.dump
load_salaries3.dump  README.md         sql_test.sh       test_versions.sh
```

> Install the employees.sql database

```
mysql -u root -p  < employees.sql
```

```
INFO
CREATING DATABASE STRUCTURE
INFO
storage engine: InnoDB
INFO
LOADING departments
INFO
LOADING employees
INFO
LOADING dept_emp
INFO
LOADING dept_manager
INFO
LOADING titles
INFO
LOADING salaries
...
```

> Verify

```
mysql -u root -p
show databases;
use employees;
show tables;
DESCRIBE titles;
```

> Database information (mysql database size)

```
use information_schema;
show tables;

SELECT table_schema AS "Database", SUM(data_length + index_length) / 1024 /
1024 AS "Size (MB)" FROM information_schema.TABLES GROUP BY table_schema
```

> Give marie all privileges to the employees database (Select, insert, update, delete) and allow her to read and create views. Allow marie to pass those privileges using the WITH GRANT OPTION

```
GRANT ALL PRIVILEGES ON employees.* TO 'marie'@'localhost' WITH GRANT
OPTION;
GRANT SELECT , CREATE VIEW ON employees.* TO 'marie'@'localhost';
FLUSH PRIVILEGES;
```

> Verify

```
mysql -u marie -p
show databases;
use employees
CREATE VIEW TEST_V AS select title from titles;
SHOW FULL TABLES WHERE table_type = 'VIEW';
```

> Use marie's account to give Guy privileges to select entries from the employees database

```
GRANT SELECT ON employees.* TO 'guy'@'localhost';
```

> Allow Guy to use the command line to take a backup of the structure from the employees table.

```
GRANT  SELECT,  PROCESS, LOCK TABLES , SHOW VIEW , TRIGGER  ON *.* TO
'guy'@'localhost';
```

> Using mysqldump

```
mysqldump -u guy --password employees employees > employees.sql
```

> Check the available export options in phpMyAdmin
> Export the table employees

> Restore the table in a new database 'hr', and give Guy all privileges.

```
mysql> create database hr;
mysql> use hr;
mysql> source employees.sql
mysql > show tables
mysql > describe employees
```

Automatic backup

> Use cron to schedule an automatic backup every hour of the database employees.

```
mkdir backups
cd backups
pwd
```

> Open crontab

```
sudo crontab -e
```

> Add cron schedule and mysqldump command

```
00 * * * * mysqldump -u root -pStudent1 employees | gzip -c >
/home/vagrant/backups/employees.sql.gz
```

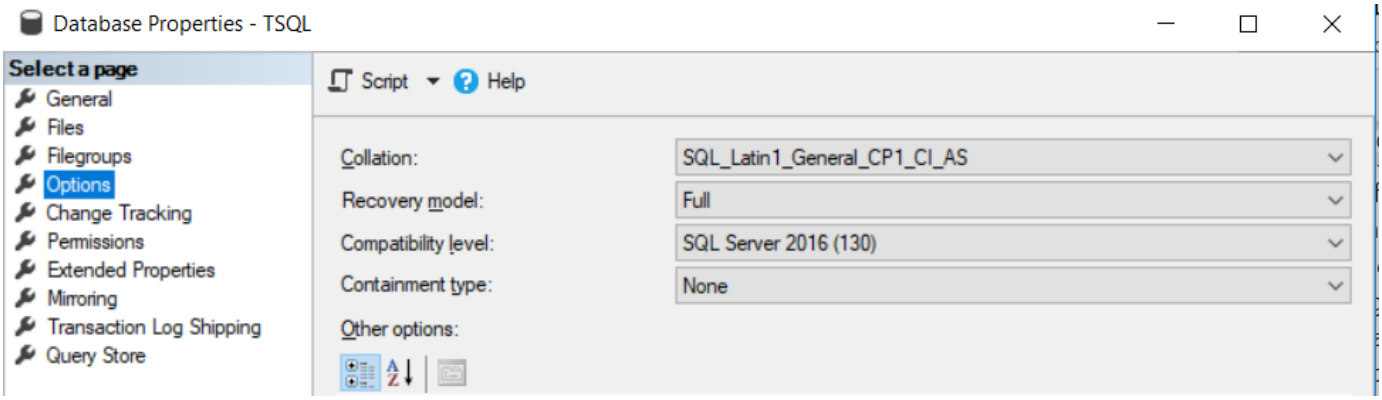
Incremental backup

An incremental backup contains all the changes that have occurred to the database since the last backup.

Transaction Log backup

https://www.sqlshack.com/sql-server-transaction-log-backup-truncate-and-shrink-operations/

The SQL Server Transaction Log backup can be taken only from the database when the recovery model of that database is Full or Bulk-Logged.



Triggers, routines, events

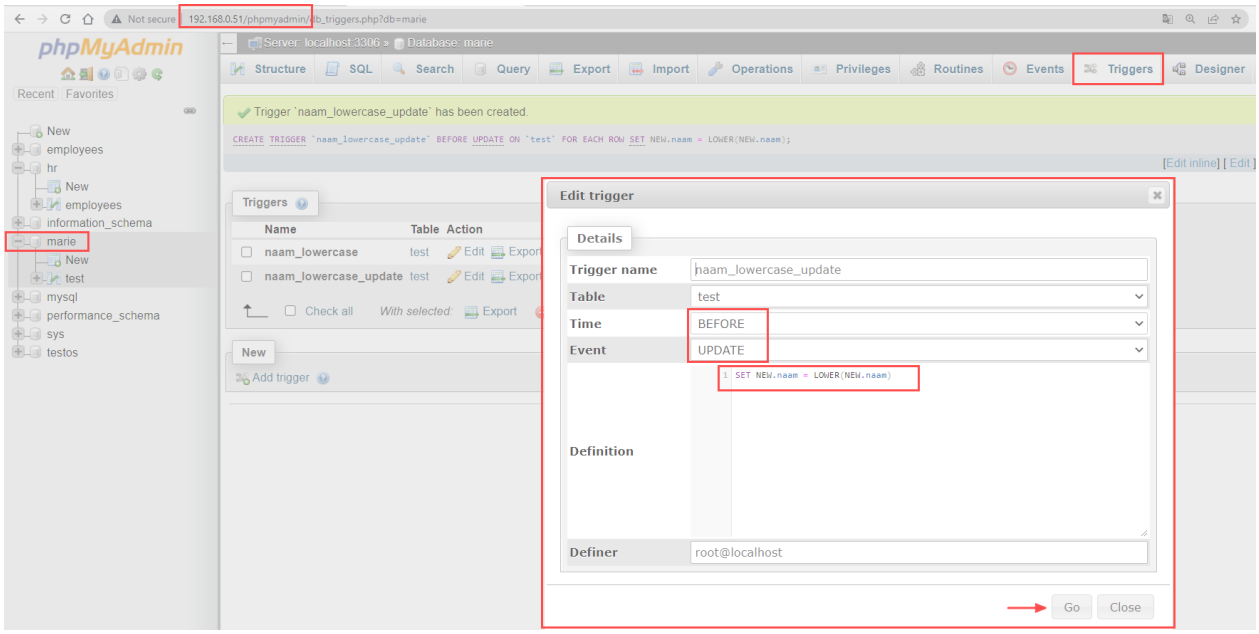
Triggers

> Create a BEFORE INSERT TRIGGER to have control over data modification before committing into the marie database table test, ensure the 'naam' is always in lowercase.

> Create an BEFORE INSERT TRIGGER USING mysql

```
CREATE TRIGGER naam_lowercase BEFORE INSERT
ON test
FOR EACH ROW
SET NEW.naam = LOWER(NEW.naam);
```

> Add an UPDATE TRIGGER using phpMyAdmin



> Verify the TRIGGERS using the mysql SHOW command

```
SHOW TRIGGERS;
```

Events

Create an event that will clear the content of the test table (each hour).

> Verify event scheduler is enabled

```
SHOW VARIABLES LIKE  "event_scheduler";
```

> Create an EVENT

```
CREATE EVENT event_delete_content_test
        ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 HOUR
```

```
        DO
         DELETE FROM test;
```

> Verify the event is successfully created, using the SHOW command

```
SHOW EVENTS;
```

Procedures

Create a procedure getTest() that will show the content of the test-table.

```
delimiter //
CREATE PROCEDURE procedure_show_test()
BEGIN
    SELECT * FROM test;
END //
delimiter ;
```

Call the procedure

```

CALL procedure_show_test();
```