MySQL on Docker Cheat Sheet

Tags

Supported tags and respective Dockerfile links

`8.0.2`, `8.0`, `8`

`5.7.19`, `5.7`, `5`, `latest`

`5.6.37`, `5.6`

`5.5.57`, `5.5`

Quickstart

```Bash
$ docker run --name some-mysql \
    -e MYSQL_ROOT_PASSWORD=my-secret-pw \
    -d mysql:tag
```

Environment Variables

`MYSQL_ROOT_PASSWORD`

This variable is mandatory and specifies the password that will be set for the MySQL root superuser account. In the above example, it was set to my-secret-pw.

`MYSQL_DATABASE`

This variable is optional and allows you to specify the name of a database to be created on image startup. If a user/password was supplied (see below) then that user will be granted superuser access (corresponding to GRANT ALL) to this database.

`MYSQL_USER`, `MYSQL_PASSWORD`

These variables are optional, used in conjunction to create a new user and to set that user's password. This

user will be granted superuser permissions (see above) for the database specified by the `MYSQL_DATABASE` variable. Both variables are required for a user to be created.

Do note that there is no need to use this mechanism to create the root superuser, that user gets created by default with the password specified by the `MYSQL_ROOT_PASSWORD` variable.

`MYSQL_ALLOW_EMPTY_PASSWORD`

This is an optional variable. Set to yes to allow the container to be started with a blank password for the root user.

NOTE: Setting this variable to yes is not recommended unless you really know what you are doing, since this will leave your MySQL instance completely unprotected, allowing anyone to gain complete superuser access.

`MYSQL_RANDOM_ROOT_PASSWORD`

This is an optional variable. Set to yes to generate a random initial password for the root user (using pwgen). The generated root password will be printed to stdout (GENERATED ROOT PASSWORD: .....).

`MYSQL_ONETIME_PASSWORD`

Sets root (not the user specified in `MYSQL_USER`!) user as expired once init is complete, forcing a password change on first login.

NOTE: This feature is supported on MySQL 5.6+ only. Using this option on MySQL 5.5 will throw an appropriate error during initialization.

Custom Configuration File

Using a custome MySQL configuration file

The MySQL startup configuration is specified in the file `/etc/mysql/my.cnf`, and that file in turn includes any files found in the `/etc/mysql/conf.d` directory that end with `.cnf`. Settings in files in this directory will augment and/or override settings in `/etc/mysql/my.cnf`. If you want to use a customized MySQL configuration, you can create your alternative configuration file in a directory on the host machine and then mount that directory location as `/etc/mysql/conf.d` inside the mysql container.

If `/my/custom/config-file.cnf` is the path and name of your custom configuration file, you can start your mysql container like this (note that only the directory path of the custom config file is used in this command):

```bash
$ docker run --name some-mysql \
    -v /my/custom:/etc/mysql/conf.d \
    -e MYSQL_ROOT_PASSWORD=my-secret-pw \
    -d mysql:tag
```

This will start a new container some-mysql where the MySQL instance uses the combined startup settings from `/etc/mysql/my.cnf` and `/etc/mysql/conf.d/config-file.cnf`, with settings from the latter taking precedence.

Note that users on host systems with SELinux enabled may see issues with this. The current workaround is to assign the relevant SELinux policy type to your new config file so that the container will be allowed to mount it:

```bash
$ chcon -Rt svirt_sandbox_file_t /my/custom
```

Configuration without a `.cnf` file

Many configuration options can be passed as flags to mysqld. This will give you the flexibility to customize the container without needing a cnf file. For example, if you want to change the default encoding and collation for all tables to use `UTF-8` (utf8mb4) just run the following:

```bash
$ docker run --name some-mysql \
    -e MYSQL_ROOT_PASSWORD=my-secret-pw \
    -d mysql:tag \
    --character-set-server=utf8mb4 \
    --collation-server=utf8mb4_unicode_ci
```