docker run --name mingxie-mysql -p 32xxx:3306 -e MYSQL\_ROOT\_PASSWORD=123456 -d mysql:latest

--name 后面的是docker容器名

-p 32xxx:3306 这里需要注意 `32xxx` 是映射到宿主机的端口，在外部通过这个端口访问docker运行的mysql服务。

-e MYSQL\_ROOT\_PASSWORD 是设置mysql的root账号密码

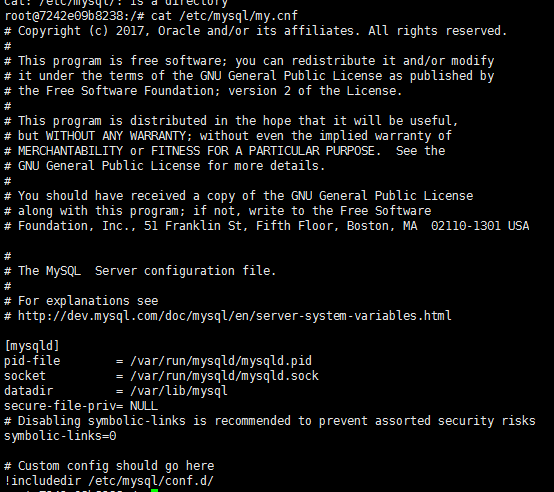
-d mysql 是你的镜像标签

这条指令会从 Docker 的仓库下载最新的 MySQL 稳定版镜像，然后创建一个叫做 mingxie-mysql的容器，我们可以使用密码为 123456 的 root 账户通过 3306 端口连接到这个数据库。

如果要将docker中的mysql服务的相关文件挂在出来，需要用-v参数。

docker run --name=mysql-sherefile -p 3306:3306 -v /opt/mysql/var/run/mysqld:/var/run/mysqld -v /opt/mysql/var/lib/mysql:/var/lib/mysql -v /opt/mysql/etc/mysql:/etc/mysql -v /opt/mysql/var/lib/mysql-files:/var/lib/mysql-files/ -e MYSQL\_ROOT\_PASSWORD=123456 -d mysql:latest

具体需要挂载那些目录，主要看mysql的配置文件my.c



v /opt/mysql/var/lib/mysql-files:/var/lib/mysql-files/

/var/lib/mysql-files/

docker  使用mysql8镜像时加载指定的外部配置文件和存储目录，报出

mysqld: Error on realpath() on '/var/lib/mysql-files' (Error **2** - No such file or directory

当指定了外部配置文件与外部存储路径时，也需要指定 /var/lib/mysql-files的外部目录，

Mysql加密策略修改

ALTER USER 'root'@'%' IDENTIFIED WITH mysql\_native\_password BY '123456';

查看源

 yum list all |grep mariadb

Mysql客户端安装

yum install -y mariadb.x86\_64 mariadb-libs.x86\_64

问题：

[user@cluster2 ~]$ mysql -uroot -p

Enter password:

ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (2)

本地linux本地登录报错，指定主机 –h ip 就没问题。

解决：

<http://www.glxxw2018.com/study/blog/detail/sRQopv2Xlo.html>

|  |
| --- |
| 1、具体所错与下所示：  [user@cluster2 ~]$ mysql -uroot -p Enter password: ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (2)  2、根据报错提示，是本地mysql连接服务器时，没有找到/var/lib/mysql/mysql.sock文件。那么从这入手，我们看到mysql容器中的服务器启动后的mysql.sock文件在哪。  我们进入mysql容器中看看：  sudo docker exec -it mysql /bin/bash  查看mysql配置文件my.cnf  root@9e313681fc3e:/# cat /etc/mysql/my.cnf  我们看到下面的一些配置：  [mysqld] pid-file = /var/run/mysqld/mysqld.pid socket = /var/run/mysqld/mysqld.sock datadir = /var/lib/mysql secure-file-priv= NULL # Disabling symbolic-links is recommended to prevent assorted security risks symbolic-links=0  这里可以看到mysqld.sock的目录是在/var/run/mysqld目录下，但是这个目录，我们并没有挂载主机目录，下面我们重新运行mysql容器，挂载相应的容器，如下所示：  sudo docker run --name=mysql -it -p 3306:3306 -v /opt/data/mysql/mysqld:/var/run/mysqld -v /opt/data/mysql/db:/var/lib/mysql -v /opt/data/mysql/conf:/etc/mysql/conf.d -v /opt/data/mysql/files:/var/lib/mysql-files -e MYSQL\_ROOT\_PASSWORD=123456 --privileged=true -d mysql  使用以上命令运行mysql容器，这里使用/opt/data/mysql中的相关目录挂载到mysql容器中，启动容器之后你会发现mysql服务器无法正常启动，报错如下：  2018-12-29T06:25:09.480604Z 0 [ERROR] [MY-010273] [Server] Could not create unix socket lock file /var/run/mysqld/mysqld.sock.lock. 2018-12-29T06:25:09.480619Z 0 [ERROR] [MY-010268] [Server] Unable to setup unix socket lock file. 2018-12-29T06:25:09.480625Z 0 [ERROR] [MY-010119] [Server] Aborting 2018-12-29T06:25:11.289786Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld: Shutdown complete (mysqld 8.0.13) MySQL Community Server - GPL. [user@cluster2 mysql]$ sudo docker logs mysql 2018-12-29T06:25:09.095301Z 0 [Warning] [MY-011070] [Server] 'Disabling symbolic links using --skip-symbolic-links (or equivalent) is the default. Consider not using this option as it' is deprecated and will be removed in a future release. 2018-12-29T06:25:09.095434Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.13) starting as process 1 2018-12-29T06:25:09.478963Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed. 2018-12-29T06:25:09.480604Z 0 [ERROR] [MY-010273] [Server] Could not create unix socket lock file /var/run/mysqld/mysqld.sock.lock. 2018-12-29T06:25:09.480619Z 0 [ERROR] [MY-010268] [Server] Unable to setup unix socket lock file. 2018-12-29T06:25:09.480625Z 0 [ERROR] [MY-010119] [Server] Aborting 2018-12-29T06:25:11.289786Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld: Shutdown complete (mysqld 8.0.13) MySQL Community Server - GPL.  说是无法创建：Could not create unix socket lock file /var/run/mysqld/mysqld.sock.lock。这说明本地的/opt/data/mysql/mysqld目录，mysql容器中无权限防问。  注意：这里的/opt/data/mysql目录下的所示目录都是mysql容器启动时自动创建的，这里只有db目录的用户和组是：polkitd input，其它的就是root root，因此，我们要将其它几个目录的用户和组都改成polkitd input。命令如下：  cd /opt/data sudo chown -R polkitd:input mysql  这样，删除mysql docker 容器。重新创建：  sudo docker rm mysql  sudo docker run --name=mysql -it -p 3306:3306 -v /opt/data/mysql/mysqld:/var/run/mysqld -v /opt/data/mysql/db:/var/lib/mysql -v /opt/data/mysql/conf:/etc/mysql/conf.d -v /opt/data/mysql/files:/var/lib/mysql-files -e MYSQL\_ROOT\_PASSWORD=123456 --privileged=true -d mysql  这样容器中的mysql就能正常运行了。  3、本地主机连接容器的mysql时，需要查到 /var/lib/mysql/mysql.sock。我们启动mysql容器后，在/opt/data/mysql/mysqld目录下有一个mysqld.sock。我们要把这个文件链接到本地主机的var/lib/myql目录中。  sudo ln -s /opt/data/mysql/mysqld/mysqld.sock /var/lib/mysql/mysql.sock  这样在运行mysql -uroot -p输入密码就能正常连接，docker容器中的mysql服务了。  4、本地主机只需要安装mysql-community-client包就可以了。具体参考百度。 |

用上面的命令挂载目录出来，mysql的配置文件就消失了。就是下面这个条命令

|  |
| --- |
| docker run --name=mysql-new2 -p 3306:3306 -v /opt/mysql/var/run/mysqld:/var/run/mysqld -v /opt/mysql/var/lib/mysql:/var/lib/mysql -v /opt/mysql/etc/mysql:/etc/mysql -v /opt/mysql/var/lib/mysql-files:/var/lib/mysql-files/ --privileged=true -e MYSQL\_ROOT\_PASSWORD=123456 -d mysql:latest |

本来mysql配置文件实在容器中的/etc/mysql目录下面，用上面的语句后，mysql配置文件就消失了。倒是不影响mysql服务的运行，但是找不到配置文件还是很慌。

问题的原因：创建容器的时候，应该是从宿主机拷贝配置文件到容器中，如果宿主机中是空目录，这样会清空容器中应该有文件的目录。这就导致了配置文件消失的现象。

解决办法：

1、创建容器的时候，可以挂在一个容器中空目录比如 /mysql/conf

2、容器起来以后，docker exec -it 28b941608555 /bin/bash 进入容器，创建目录/mysql/conf，然后将/etc/mysql中的配置文件拷贝到/mysql/conf中，然后宿主机中对应的目录就有配置文件了cp /etc/mysql/\* /mysql/conf –r

3、最后将现在的容器删掉，重新对应挂载-v /opt/mysql/etc/mysql:/etc/mysql 创建新容器。

4、这样就可以通过宿主机中的配置，来改变docker容器中的mysql服务。

参考文档：<https://blog.csdn.net/u014690615/article/details/83692683>

也可以下面这样：

|  |
| --- |
| 普通安装  1.下载镜像，mysql 5.7  docker pull mysql:5.7  2.创建mysql容器，并后台启动  docker run -d -p 3306:3306 -e MYSQL\_USER="woniu" -e MYSQL\_PASSWORD="123456" -e MYSQL\_ROOT\_PASSWORD="123456" --name mysqltest1 mysql:5.7 --character-set-server=utf8 --collation-server=utf8\_general\_ci  参数说明：  -e MYSQL\_USER="woniu"  ：添加woniu用户  -e MYSQL\_PASSWORD="123456"：设置添加的用户密码  -e MYSQL\_ROOT\_PASSWORD="123456"：设置root用户密码  --character-set-server=utf8：设置字符集为utf8  --collation-server=utf8\_general\_cli：设置字符比较规则为utf8\_general\_cli  挂载外部配置和数据安装  1.创建目录和配置文件my.cnf  mkdir /docker  mkdir /docker/mysql  mkdir /docker/mysql/conf  mkdir /docker/mysql/data    创建my.cnf配置文件  touch /docker/mysql/conf/my.cnf    my.cnf添加如下内容：  [mysqld]  user=mysql  character-set-server=utf8  default\_authentication\_plugin=mysql\_native\_password  [client]  default-character-set=utf8  [mysql]  default-character-set=utf8  2.创建容器，并后台启动  docker run -d -p 3306:3306 --privileged=true -v /docker/mysql/conf/my.cnf:/etc/mysql/my.cnf -v /docker/mysql/data:/var/lib/mysql -e MYSQL\_ROOT\_PASSWORD=123456 --name mysqltest2 mysql:5.7  参数说明：  --privileged=true：容器内的root拥有真正root权限，否则容器内root只是外部普通用户权限  -v /docker/mysql/conf/my.cnf:/etc/my.cnf：映射配置文件  -v /docker/mysql/data:/var/lib/mysql：映射数据目录  ---------------------  作者：码农code之路  来源：CSDN  原文：https://blog.csdn.net/woniu211111/article/details/80968154  版权声明：本文为博主原创文章，转载请附上博文链接！ |

关于挂载有一下几个地方需要注意

1.挂载之前my.cnf 必须要配置一个默认用户[mysqld] user=mysql

2.挂载之前本地/docker/mysql/data这个路径下不能有其他文件或者文件夹