Docker 安装 mysql

> docker hub仓库：https://hub.docker.com/_/mysql/

什么是MySQL?

> MySQL是世界上最流行的开源数据库。凭借其久经考验的性能，可靠性和易用性，MySQL已经成为基于Web的应用程序的领先数据库选择，涵盖从个人项目和网站，电子商务的信息服务到高知名度的整个范围。网站属性包括Facebook, Twitter, YouTube, Yahoo! 还有很多。

安装步骤

1. 获取镜像：
   - docker pull mysql
   - docker pull mysql:5.7
2. 查看本地所有镜像：
   - docker images
3. 运行部署命令：
   - docker run -d -p 3306:3306 [mysql:tag]

- 启动MySQL服务器实例

```
# 启动MySQL服务器实例
docker run --name some-mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:ta

docker run -d -p 3306:3306 [mysql:tag]

# --name 分配给容器的名称
--name mysql
# -P -e mysqld
# 创建容器时初始化数据库,详情查看环境变量
-e MYSQL_DATABASE=testDb
# 创建窗口时初始化root用户密码
-e MYSQL_ROOT_PASSWORD=123456
# -v 宿主机命令映射其目录到容器或主机，用于存储数据
-v /home/inface/mysql_data:/var/lib/mysql
```

从另一个Docker容器中的应用程序连接到MySQL

> 由于镜像公共标准MySQL端口（3306），该容器连接使MySQL容器可用于其他应用程序容器。

```
# 像这样启动你的应用程序容器，以便将它连接到MySQL容器
docker run --name some-app --link some-mysql:mysql -d application-that-uses-
```

从MySQL命令行客户端连接到MySQL

> 该容器启动另一个 mysql 容器实例并针对 mysql 原始容器运行命令行客户端，允许对数据库实执行SQL语句

```
docker run -it --link some-mysql:mysql --rm mysql sh -c 'exec mysql -h"$MYSQ

# some-mysql 原始 mysql 容器的名称在哪里
```

- 该镜像还可以用作非Docker或远程MySQL实例的客户端：

```
docker run -it --rm mysql mysql -hsome.mysql.host -usome-mysql-user -p
```

容器shell访问和查看MySQL日志

> 这个docker exec 命令允许你在Docker容器内运行命令。以下命令行将为你提供 mysql 容器内的bash shell:

```
docker exec -it --some-mysql bash
```

MySQL Server 日志可通过Docker的容器日志获得：

```
docker logs some-mysql
```

使用自定义MySQL配置文件

可以在 /etc/mysql/my.cnf 中找到MySQL的默认配置，这可以是 !includedir 其他目录，如 /etc/mysql/conf.d 或 /etc/mysql/mysql.conf.d 。请检查mysql镜像本身内的相关文件和目录以获取更多信息。

如果 /my/custome/config-file.cnf 是自定义配置文件的路径和名称，则可以 mysql 像这样启动容器（请注意，此命令中仅使用自定义配置文件的目录路径）：

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

这将启动一个新容器some-mysql，其中MySQL实例使用组合的启动设置，/etc/mysql/my.cnf并且/etc/mysql/conf.d/config-file.cnf后者的设置优先。

没有cnf文件的配置

> 许多配置选项可以作为标志传递给mysqld。这将使你可以灵活地自定义容器而无需cnf文件。例如，如果要更改所有表的默认编码和排序规则以使用UTF-8(utf8mb4)，只需运行以下命：

```
docker run --name some-mmysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:t
```

- 如果你想查看可用选项的完整列表，请运行

```
$ docker run -it --rm mysql:tag --verbose --help
```

环境变量

> 启动 mysql 映像时，可以通过在 docker run 命令行上传递一个或多个环境变量来调整MySQL实例的配置。请注意，如果使用已包含数据库的数据目录启动容器，则以下任何变量都不会产生任何影响：任何预先存在的数据库在容器启动时始终保持不变。
>
> 另请参阅https://dev.mysql.com/doc/refman/5.7/en/enviroment-variables.html以获取MySQL本身所关注的环境变量的文档（尤其是变量，例如 MYSQL_HOST ，与此镜像一起使用时会出现问题)。

MYSQL_ROOT_PASSWORD

> 此变量是必需的，并指定将为MySQL root 超级用户账户设置的密码。在上面的例子中，它被设置为 my-secret-pw 。

MYSQL_DATABASE

> 此变量是可选的，允许你指定要在镜像启动时创建数据库的名称。如果提供了用户/密码（见下文），则该用户将被授予对该数据库的超级用户访问权限（对应 GRANT ALL）

MYSQL_USER, MYSQL_PASSWORD

> 这些变量是可选的，可以结合使用来创建新用户并设置该用户的密码。此用户将被授予MYSQL_DATABASE变量指定的数据库的超级用户权限（参见上文）。这两个变量都是创建用户所必需的。
>
> 请注意，不需要使用此机制来创建超级超级用户，默认情况下会使用 MYSQL_ROOT_PASSWORD 变量指定的密码创建该用户。

MYSQL_ALLOW_EMPTY_PASSWORD

> 这是一个可选变量。设置为yes允许以root用户的空密码启动容器。注意：建议不要将此变量设置为 yes ，除非你确实知道自己在做什么，因为这会使你的MySQL实例完全不受保护，从而允许任何人获得完整的超级用户访问权限。

MYSQL_RANDOM_ROOT_PASSWORD

> 这是一个可选变量，设置为 yes 为root用户生成随机初始密码（使用 pwgen ）。生成的root密码将打印到 stdout(GENERATED ROOT PASSWORD: .....) 。

MYSQL_ONETIME_PASSWORD

> 初始化完成后，将root(不是 MYSQL_USER ！中指定的用户)设置为过期，在首次登录时强制更改密码。注意：此功能仅在MySQL5.6+上受支持。在MySQL 5.5上使用此选项将在初始化期间抛出适当的错误。

Docker 的秘密

> 作为通过环境变量传递第三信息的替代方法， _FILE 可以将其附加到先前列出的环境变量，从而使用初始化脚本从容器中存在的文件加载这些变量的值。特别是，这可以用于从存储在 /run/secrets/<secret_name> 文件中的Docker机制加载密码。例如：

```
1  docker run --name some-mysql -e MYSQL_ROOT_PASSWORD_FILE=/run/secrets/mysql-
```

目前，这仅支持 `MYSQL_ROOT_PASSWORD` ， `MySQL_ROOT_HOST` ， `MYSQL_DATABASE` ， `MYSQL_USER` 和 `MYSQL_PASSWORD` 。

初始化一个新实例

> 首次启动容器时，将创建具有指定名称的新数据库，并使用提供的配置变量进行初始化。此外，它将执行带扩展名的文件 `.sh` ， `.sql` 并且 `.sql.gz` 可以在其中找到 `/docker-entrypoint-initdb.d` 。文件将按字母顺序执行。你可以通过 `mysql` 将SQL转储装入该目录并使用提供的数据提供自定义映像来轻松填充服务。默认情况下，SQL文件将导入到MYSQL_DATABASE变量指定的数据库中。

注意事项

存储数据的位置

> 重要说明：有几种方法可以存储在Docker容器中运行的应用程序使用数据。我们鼓励mysql镜像用户熟悉可用选项，包括：

- 让 Docker 通过使用自己的内部卷管理将数据库文件写入主机系统上的磁盘来管理数据库数据的存储。这是默认设置，对用户来说简单且相当透明。缺点是文件可能很难找到直接在主机系统上运行的工具和应用程序，即外部容器。
- 在主机系统（容器外部）上创建数据目录，并将其安装到容器内可见的目录中。这会将数据库文件放在主机系统上的已知位置，并使主机系统上的工具和应用程序可以轻松访问这些文件。缺点是用户需要确保目录存在，并且例如主机系统上的目录权限和其他安全机制已正确设置。

Docker文档是了解不同存储选项和变体的一个很好的起点，并且有多个博客和论坛帖子可以在这个领域进行讨论的提供建议。我们将在上面针对后一个项选项简单地显示基本过程:

1. 在主机系统上的适当卷上创建数据目录，例如 `/my/own/datadir` 。
2. mysql像这样启动你的容器

```
1  docker run --name some-mysql -v /my/own/datadir:/var/lib/mysql -e MYSQL_ROOT
```

`-v /my/own/datadir:/var/lib/mysql` 命令的一部分 `/my/own/datadir` 将底层主机系统中的目录安装在 `/var/lib/mysql` 容器内部，默认情况下MySQL将写入其数据文件

在MySQL init 完成之前没有连接

> 如果容器启动时没有初始化数据库，则将创建默认数据库。虽然这是预期的行为，但这意味着在初始化完成之前它不会接受传入的连接。使用自动化工具时可能会出现问题，例如 `docker-compose` ，同时启动多个容器。
>
> 如果你尝试连接到MySQL的应用程序无法处理MySQL停机或等待MySQL正常启动，那么在服务启动之前放置连接重试循环可以是必要的。有关官方镜像中此类实现的示例，请参阅WordPress或Bonita.

针对现有数据库的用法

> 如果 `mysql` 使用已包含数据库（特别是 `mysql` 子目录）的数据目录启动容器实例，`$MYSQL_ROOT_PASSWORD` 则就从运行命令行中省略该变量;在任何情况下都会被忽略，并且不会以任何方式更改预先存在的数据库。

作为任意用户运行

> 如果你知道目录的权限已经适当设置（例如针对现有数据库运行，如上所述），或者您需要mysqld使用特定的UID/GID运行，则可以通过 --user 设置为作为值来调用此镜像（除了 root/ 0)以实现所需的访问/配置:

```
$ mkdir data
$ ls -lnd data
drwxr-xr-x 2 1000 1000 4096 Aug 27 15:54 data

$ docker run -v "$PWD/data":/var/lib/mysql --user 1000:1000 --name some-mysq
```

创建数据库转储

> 大多数常规工具都可以使用，但在某些情况下它们的使用可能有点复杂，以确保它们可以访问 mysqld 服务器。确保这一点的一种简单方法是docker exec 从同一容器中使用和运行该工具，类似于以下内容:

```
$ docker exec some-mysql sh -c 'exec mysqldump --all-databases -uroot -p"$MY
```