Docker 网络

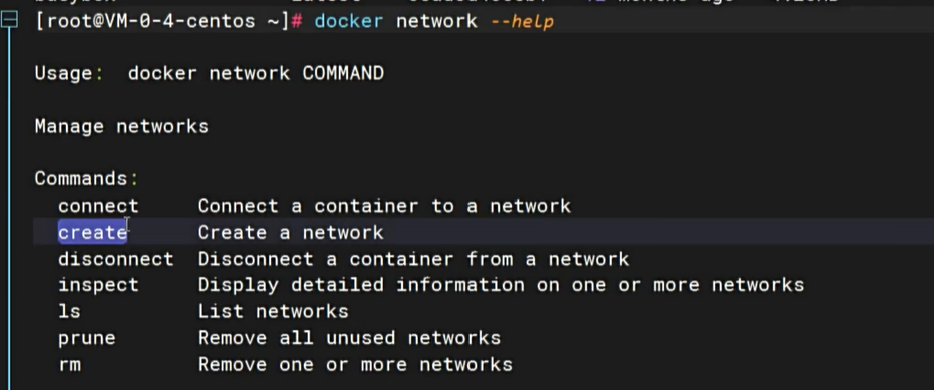
Docker每个容器都有自己的独特ip，容器之间可以互相访问

图26



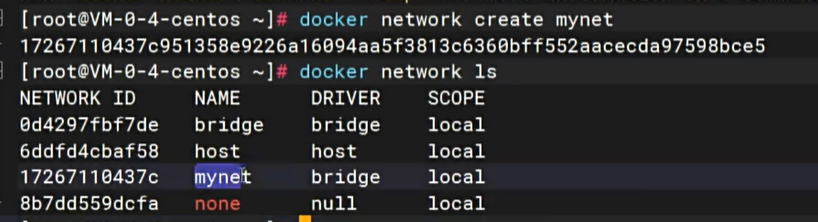
docker network 涉及到所有网络相关命令

图27



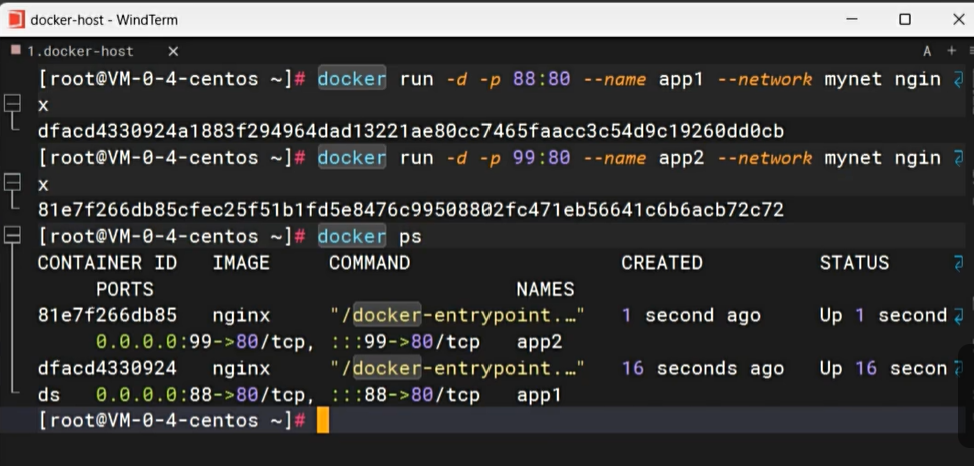
自定义网络

图28



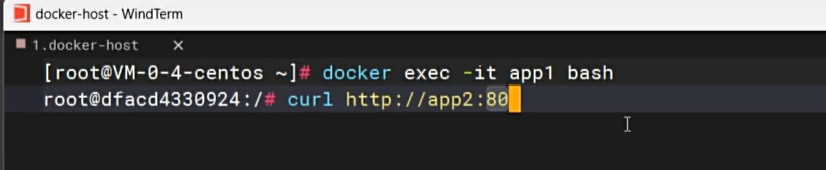
使用自定义网络启动两个容器

图29



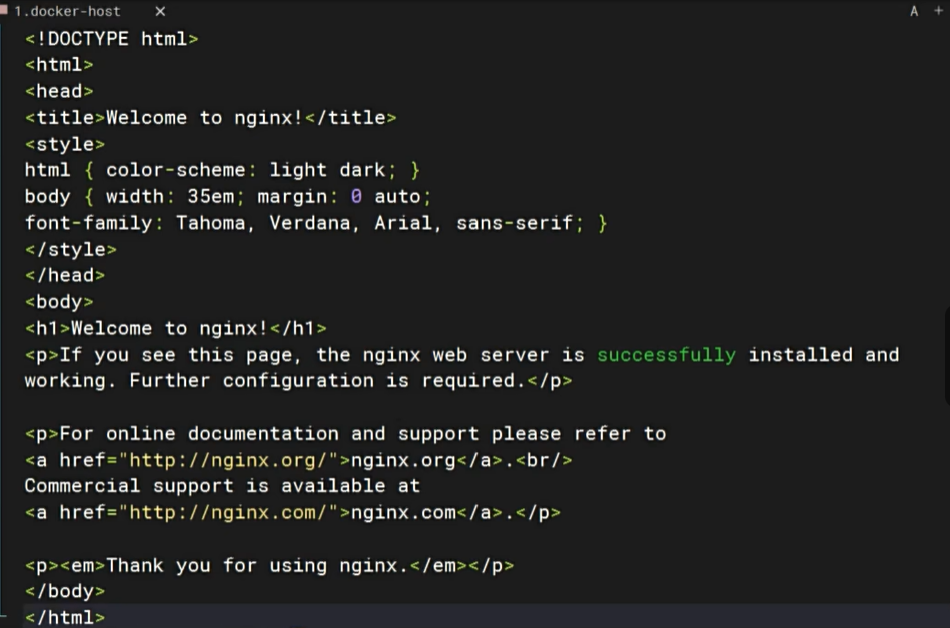
使用容器内部网络访问

图30



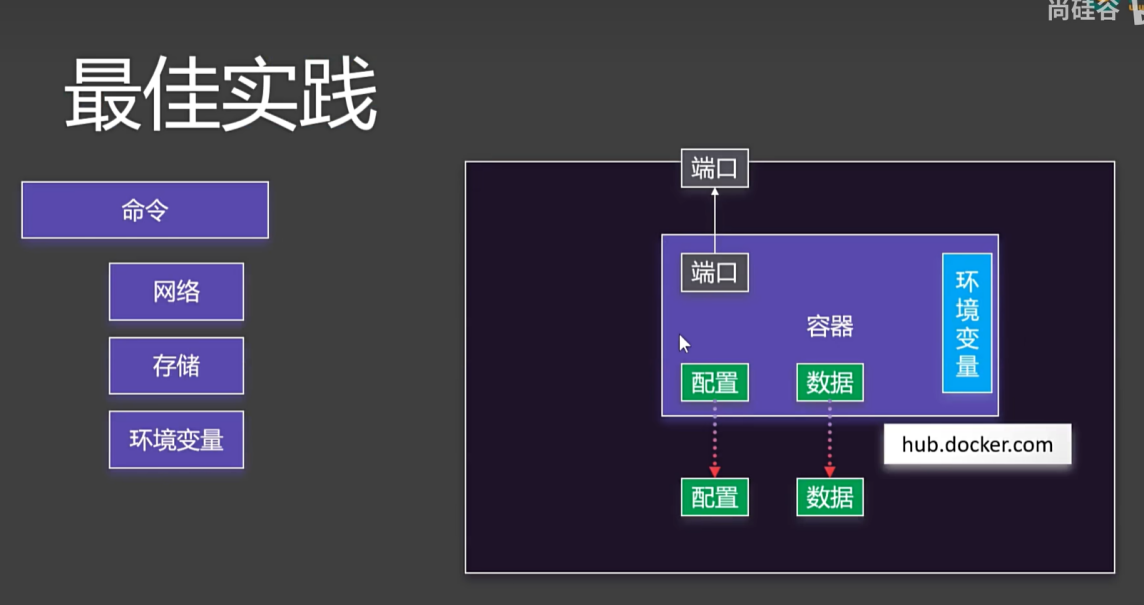
回车后页面数据正常显示

图31



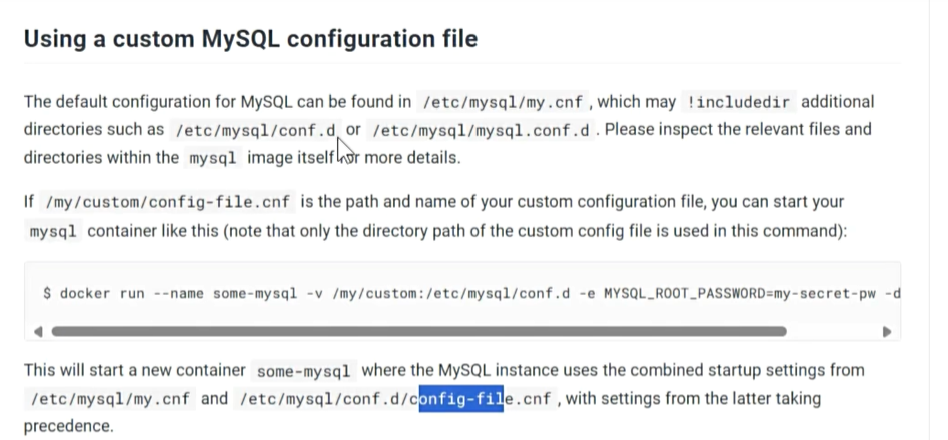
最佳实践，以myaql为例

图32



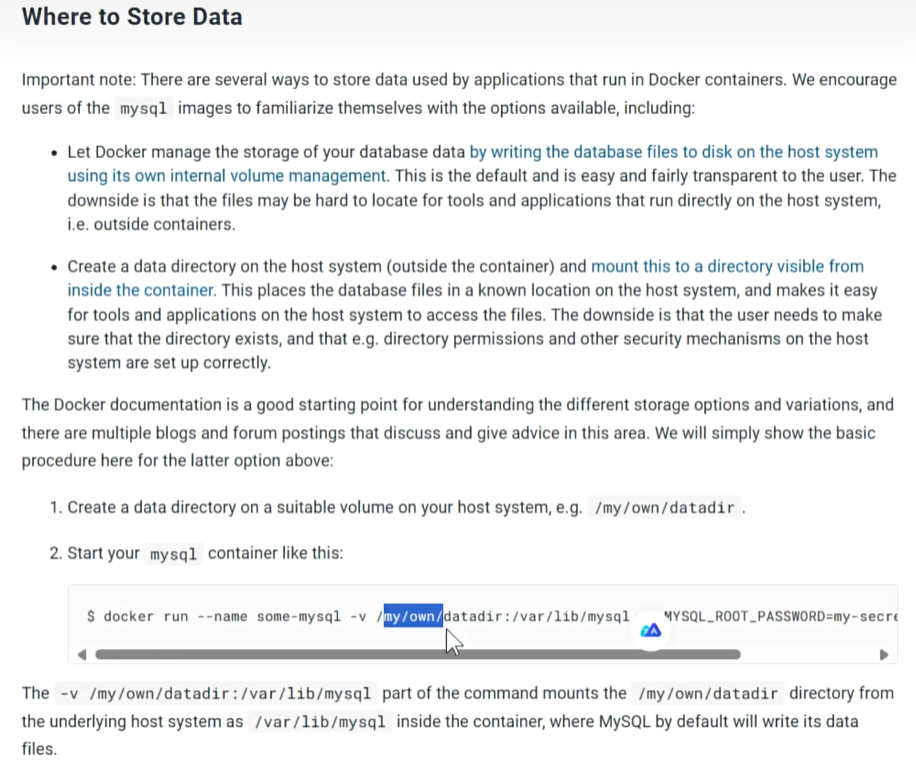
可以在官方文档找到更多信息，比如在这里可以使用.cnf结尾，都会被默认为mysql的配置文件

图33



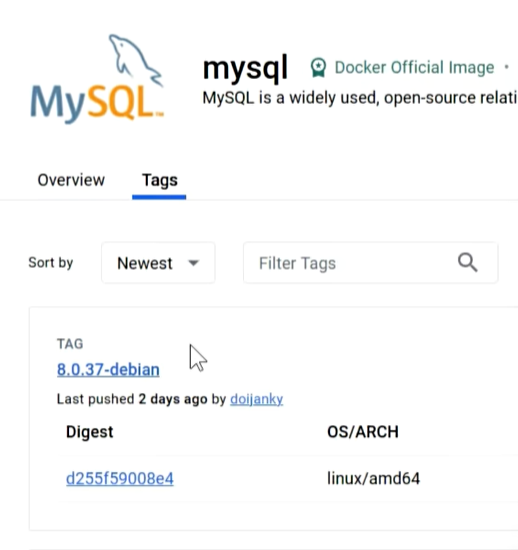
这里告知了mysql的数据存储位置，文档中会有没提到的内容，比如端口，mysql的默认端口是3306

图34



可以在tag这里找到镜像版本标签，默认不用标签的话就是以最新版本开启

图35



\ 为换行符，docker run 命令用于运行一个新的容器实例。

-d 参数表示以分离（detached）模式运行容器。这意味着容器会在后台运行，不会阻塞当前终端。

-p 3306:3306 参数用于端口映射，将主机的端口 3306 映射到容器的端口 3306。这里的3306是MySQL的默认端口，这样你可以通过主机的3306端口访问容器内的MySQL服务。

-v /app/myconf:/etc/mysql/conf.d 参数用于挂载卷，将主机目录 /app/myconf 挂载到容器内的 /etc/mysql/conf.d 目录。这个目录通常用于存放 MySQL 配置文件，因此你可以在主机上修改 MySQL 的配置文件，并立即应用到容器中的 MySQL 服务。

-v /app/mydata:/var/lib/mysql 另一个 -v 参数用于将主机目录 /app/mydata 挂载到容器内的 /var/lib/mysql 目录。这个目录是 MySQL 的数据存储位置，挂载后可以在主机上持久化 MySQL 数据。

-e MYSQL\_ROOT\_PASSWORD=123456 参数用于设置环境变量。在这里，它设置了 MySQL 的 root 用户密码为 123456。

mysql:8.0.37-debian 这是要运行的 Docker 镜像和标签（tag）表示使用 MySQL 8.0.37 版本，并且基于 Debian 操作系统的镜像。如果本地没有这个镜像，Docker 会自动从 Docker Hub 拉取。

图36

