7-2 利用Docker部署程序（一）

一、安装Docker程序

这节课我们要利用Docker环境部署很多东西，有数据库、人脸识别程序、Emos项目等等，所以首先要安装Docker环境。

```
#安装Docker
yum install docker -y
#启动Docker服务
service docker start
```

因为Docker在线安装镜像是从国外的DockerHub网站下载文件，所以速度超级慢，所以我们要给Docker设置加速器。我们用的是腾讯云主机，当然设置腾讯云加速器是最快的。

打开 `/etc/docker/daemon.json` 文件，然后设置成如下内容：

```
{
    "registry-mirrors": ["https://mirror.ccs.tencentyun.com"]
}
```

重新启动Docker服务，加速器才能生效

```
service docker restart
```

二、安装MySQL

利用刚才设置的加速器，我们可以在线安装MySQL镜像，这里我下载8.0.23版本的MySQL数据库

```
docker pull mysql:8.0.23
```

创建容器的时候，我们需要把MySQL容器内的数据目录映射到CentOS系统上面。如果MySQL容器挂掉了，数据库文件不会丢失。我们新建一个MySQL容器，挂载上这个数据目录就又能正常使用MySQL了。

以前我说过Docker会给每个容器创建一个 `虚拟的网卡` ，然后分配一个Docker `内网IP地址` 。假设A容器部署了MySQL，B容器中的Java程序想要访问A容器的MySQL，JDBC路径就要写A容器的Docker内网IP地址，略显麻烦。

这次部署项目，我打算把容器设置成 `host模式` ，就是不让Docker为容器虚拟网卡，Docker容器直接使用CentOS的网卡。A容器和B容器中使用的都是CentOS的网卡，所以A容器中localhost代表CentOS，B容器中的localhost也代表CentOS，两个容器相互访问，URL地址写localhost即可。例如B容器中JDBC路径的host写localhost，就能访问到A容器里面的MySQL。这样我们部署的若干容器，相互访问就简单多了。

MySQL容器我分配内存空间是500M，如果将来觉得不够用，删除容器，再创建新容器的时候分配更大的内存。而且只要挂载上那些文件目录，MySQL的数据就不会丢失。

```
docker run -it -d --name mysql --net=host \
-m 500m -v /root/mysql/data:/var/lib/mysql \
-v /root/mysql/config:/etc/mysql/conf.d  \
-e MYSQL_ROOT_PASSWORD=abc123456 \
-e TZ=Asia/Shanghai mysql:8.0.23
```

用Navicat连接MySQL数据库，创建emos逻辑库，然后导入数据。

三、安装MongoDB

执行命令，下载最新版本的MongoDB镜像

```
docker pull mongo
```

创建 `/root/mongo/mongod.conf` 文件，然后在文件中添加如下内容：

```
mkdir -p /root/mongo
vim /root/mongo/mongod.conf
```

```
net:
    port: 27017
    bindIp: "0.0.0.0"
storage:
    dbPath: "/data/db"
security:
   authorization: enabled
```


创建容器，为MongoDB分配500M内存

```
docker run -it -d --name mongo --net=host \
-v /root/mongo:/etc/mongo -m 500m \
-e MONGO_INITDB_ROOT_USERNAME=admin \
-e MONGO_INITDB_ROOT_PASSWORD=abc123456 \
mongo --config /etc/mongo/mongod.conf
```

用Navicat连接MongoDB，检测是否可用。

四、安装Redis

执行命令，在线安装Redis镜像

```
docker pull redis:6.0.10
```

创建 `/root/redis/conf/redis.conf` 文件，然后添加如下内容：

```
mkdir -p /root/redis/conf/
vim /root/redis/conf/redis.conf
```

```
bind 0.0.0.0
protected-mode yes
port 6379
tcp-backlog 511
timeout 0
tcp-keepalive 0
loglevel notice
logfile ""
databases 4
save 900 1
save 300 10
save 60 10000
stop-writes-on-bgsave-error yes
rdbcompression yes
rdbchecksum yes
dbfilename dump.rdb
dir ./
requirepass abc123456
```

执行命令，创建Redis容器，分配300M内存

```
docker run -it -d --name redis -m 300m --net=host \
-v /root/redis/conf:/usr/local/etc/redis redis:6.0.10 \
redis-server /usr/local/etc/redis/redis.conf
```

五、安装RabbitMQ

把镜像上传到云主机，然后导入Docker

```
docker load < rabbitmq.tar.gz
```

执行命令，创建RabbitMQ容器，分配300M内存

```
docker run -it -d --name mq -m 300m --net=host rabbitmq
```

六、部署人脸识别程序

把人脸识别镜像文件上传到云主机（2M带宽大概需要40分钟），导入Docker环境

```
docker load < face.tar.gz
```

把demo.tar文件上传到Linux根目录，然后解压缩

```
tar -xvf demo.tar
```

执行命令，创建人脸识别容器，分配1GB内存

```
docker run -d -it -v /demo:/demo --net=host -m 1g --name node face
#进入到node容器
docker exec -it node bash
cd /demo
#把Python程序挂起到后台运行
nohup python3 -c "from app import app;" > log.out 2>&1 &
```