链滴

(运维篇)- 使用 docker 搭建 hadoop-hive-spark 集群 (一)

作者：sixleaves
原文链接：https://ld246.com/article/1585493212704
来源网站：链滴

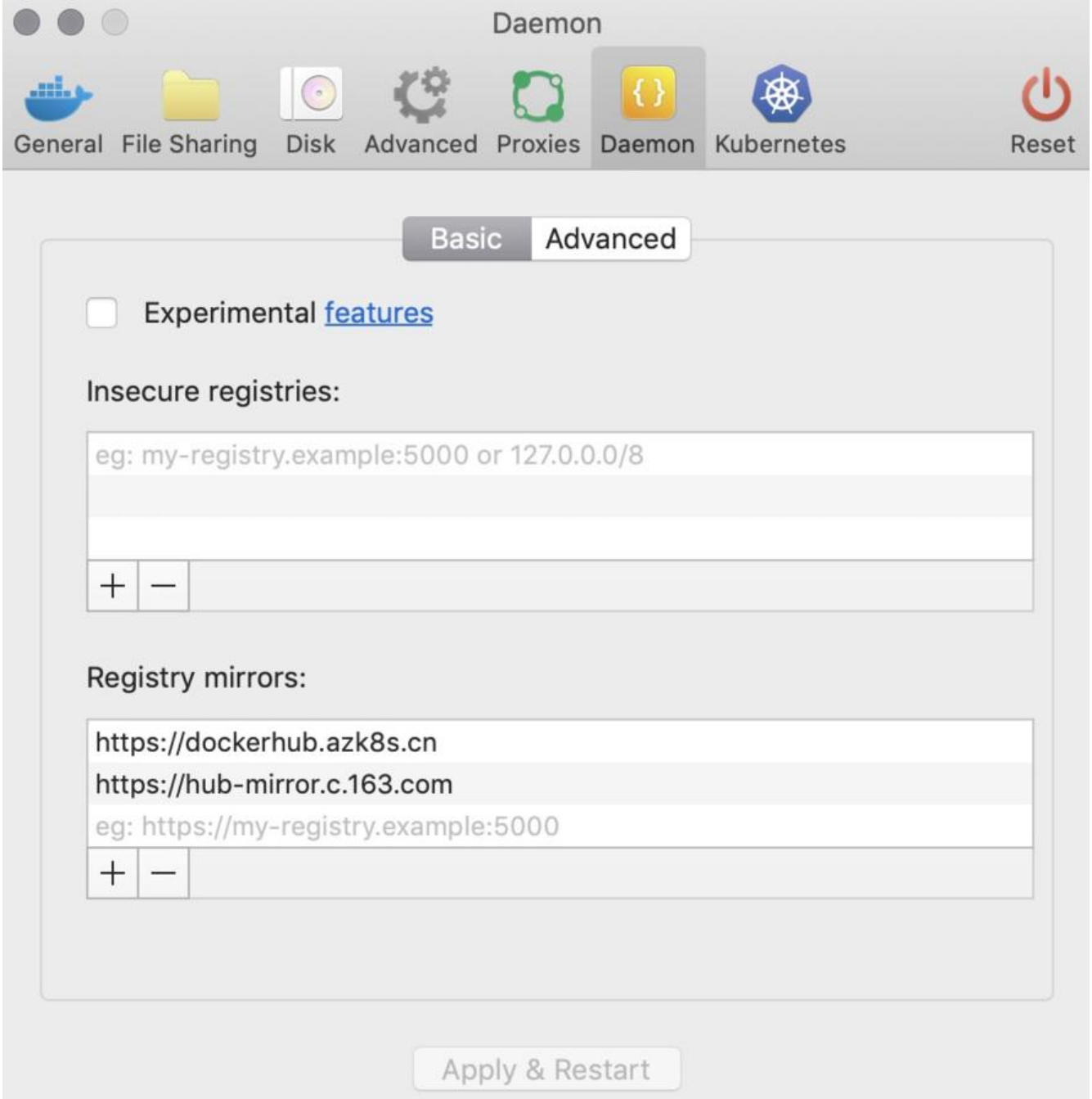
一、安装 docker

1.1 使用 brew cask 安装

由于是 Mac 系统，直接使用 brew 最为方便

brew cask install docker

1.2 镜像替换为加速镜像



如上图得两个镜像地址，填入后，点击 apply&restart 重启 docker

https://dockerhub.azk8s.cn

https://hub-mirror.c.163.com

二、构建 CentOS 镜像

找一个目录分别创建以下两个文件

2.1 创建 run.sh

```
!/bin/bash
/usr/sbin/sshd -D
```

2.2 创建 Dockerfile

```
#生成的新镜像以centos镜像为基础
FROM centos
MAINTAINER by kongtrio(kongtrio@sina.com)
#升级系统
RUN yum -y update
#安装openssh-server、client
RUN yum -y install openssh-server openssh-clients.x86_64 vim less wget
#修改/etc/ssh/sshd_config
#RUN sed -i 's/UsePAM yes/UsePAM no/g' /etc/ssh/sshd_config

#将密钥文件复制到/etc/ssh/目录中。这里要用root的权限生成key
RUN mkdir -p /root/.ssh
#生成秘钥、公钥
RUN ssh-keygen -t rsa -b 2048 -P '' -f /root/.ssh/id_rsa
RUN cat /root/.ssh/id_rsa.pub > /root/.ssh/authorized_keys
RUN cp /root/.ssh/id_rsa /etc/ssh/ssh_host_rsa_key
RUN cp /root/.ssh/id_rsa.pub /etc/ssh/ssh_host_rsa_key.pub

# 安装 jre 1.8
RUN yum -y install java-1.8.0-openjdk.x86_64
ENV JAVA_HOME=/etc/alternatives/jre_1.8.0

#定义时区参数
ENV TZ=Asia/Shanghai
#设置时区
RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo '$TZ' > /etc/timezone

#将ssh服务启动脚本复制到/usr/local/sbin目录中，并改变权限为755
ADD run.sh /usr/local/sbin/run.sh
RUN chmod 755 /usr/local/sbin/run.sh

#变更root密码为root
RUN echo "root:root"| chpasswd
#开放窗口的22端口
EXPOSE 22
#运行脚本，启动sshd服务
CMD ["/usr/local/sbin/run.sh"]
```

1.3 构建自己的 docker 镜像

docker build -t my_centos:v1 .

```
➜ hadoop docker build -t my_centos:v1 .
Sending build context to Docker daemon  4.096kB
Step 1/18 : FROM centos
latest: Pulling from library/centos
8a29a15cefae: Downloading [==========>                          ]  42.17MB/73.23MB
```

1.4 启动容器

docker run -d -P --name hadoop_centos my_centos:v1 /usr/local/sbin/run.sh

```
➜ hadoop docker run -d -P --name hadoop_centos my_centos:v1 /usr/local/sbin/run.sh
54fcd61968d3dd567cb52cb60457d85547056596f71d27e0dc069d2c84bfa21
```

1.5 进入容器

docker exec -it hadoop_centos /bin/bash

```
➜ hadoop docker exec -it hadoop_centos /bin/bash
[root@654fcd61968d /]#
```

三、安装 hadoop

3.1 安装 Git

yum -y install git

3.2 安装 hadoop

下载官网二进制包 hadoop-2.7.0.tar.gz

https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-2.7.0/hadoop-2.7.0.tar.gz

```
# 拷贝安装包
docker cp hadoop-2.7.0.tar hadoop_centos:/usr/local
# 进入容器

docker exec -it hadoop_centos /bin/bash
cd /usr/local/

# 解压安装包
tar xvf hadoop-2.7.0.tar
```

```
➜ hadoop docker cp hadoop-2.7.0.tar hadoop_centos:/usr/local
[root@654fcd61968d soft]# cd /usr/local/
[root@654fcd61968d local]# tar xvf hadoop-2.7.0.tar
hadoop-2.7.0/
hadoop-2.7.0/bin/
```

3.3 配置 hadoop

core-site.xml

vim /usr/local/hadoop-2.7.0/etc/hadoop/core-site.xml

```
<property>
    <name>fs.defaultFS</name>
    <value>hdfs://127.0.0.1:9000</value>
</property>
```

```
<configuration>
<property>
        <name>fs.defaultFS</name>
        <value>hdfs://127.0.0.1:9000</value>
</property>
</configuration>
```

hdfs-site.xml

```
vim /usr/local/hadoop-2.7.0/etc/hadoop/hdfs-site.xml

<property>
    <name>dfs.replication</name>
    <value>1</value>
</property>
```

mapred-site.xml

```
cp /usr/local/hadoop-2.7.0/etc/hadoop/mapred-site.xml.template /usr/local/hadoop-2.7.0/et
/hadoop/mapred-site.xml
vim /usr/local/hadoop-2.7.0/etc/hadoop/mapred-site.xml

<property>
   <name>mapreduce.framework.name</name>
    <value>yarn</value>
</property>
```

hadoop-env.sh

```
vim /usr/local/hadoop-2.7.0/etc/hadoop/hadoop-env.sh

# 将原来的 export JAVA_HOME=${JAVA_HOME} 改成下面这个
export JAVA_HOME="/etc/alternatives/jre_1.8.0"
```

3.4 指定 HADOOP 环境变量

```
vim /etc/profile

# 在文本最后加上
export HADOOP_HOME="/usr/local/hadoop-2.7.0"
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin

source /etc/profile
```

3.5 namenode 初始化

```
hadoop namenode -format
```

3.6 启动 hdfs 和 yarn

启动 hdfs

start-dfs.sh

- 启动 hdfs 后会有三个相关进程， NameNode 、 SecondaryNamenode 、 Datanode 。
- 使用 ps -ef | grep hadoop 查看是否有，有表示启动成功。

```
root@654fcd61968d local]# ps -ef | grep hadoop
root        39     8  0 16:11 pts/0    00:00:00 git clone https://github.com/apache/hadoop
root        40    39  0 16:11 pts/0    00:00:02 /usr/libexec/git-core/git-remote-https origin https://github.com/apache/hadoop
root        42    40  0 16:11 pts/0    00:00:00 /usr/libexec/git-core/git fetch-pack --stateless-rpc --stdin --lock-pack --thin --check-self-contained-and-connected --c
ub.com/apache/hadoop/
root       241     1 10 16:32 ?        00:00:03 /etc/alternatives/jre_1.8.0/bin/java -Dproc_namenode -Xmx1000m -Djava.net.preferIPv4Stack=true -Dhadoop.log.dir=/usr/loc
gs -Dhadoop.log.file=hadoop.log -Dhadoop.home.dir= -Dhadoop.id.str=root -Dhadoop.root.logger=INFO,console -Djava.library.path= -Dhadoop.policy.file=hadoop-policy.xml -D
$Stack=true -Djava.net.preferIPv4Stack=true -Djava.net.preferIPv4Stack=true -Dhadoop.log.dir=/usr/local/hadoop-2.7.0/logs -Dhadoop.log.file=hadoop-root-namenode-654fcd6
home.dir=/usr/local/hadoop-2.7.0 -Dhadoop.id.str=root -Dhadoop.root.logger=INFO,RFA -Djava.library.path=/usr/local/hadoop-2.7.0/lib/native -Dhadoop.policy.file=hadoop-
et.preferIPv4Stack=true -Dhadoop.security.logger=INFO,RFAS -Dhdfs.audit.logger=INFO,NullAppender -Dhadoop.security.logger=INFO,RFAS -Dhdfs.audit.logger=INFO,NullAppend
y.logger=INFO,RFAS -Dhdfs.audit.logger=INFO,NullAppender -Dhadoop.security.logger=INFO,RFAS org.apache.hadoop.hdfs.server.namenode.NameNode
root       377     1 13 16:32 ?        00:00:03 /etc/alternatives/jre_1.8.0/bin/java -Dproc_datanode -Xmx1000m -Djava.net.preferIPv4Stack=true -Dhadoop.log.dir=/usr/loc
gs -Dhadoop.log.file=hadoop.log -Dhadoop.home.dir= -Dhadoop.id.str=root -Dhadoop.root.logger=INFO,console -Djava.library.path= -Dhadoop.policy.file=hadoop-policy.xml -D
$Stack=true -Djava.net.preferIPv4Stack=true -Djava.net.preferIPv4Stack=true -Dhadoop.log.dir=/usr/local/hadoop-2.7.0/logs -Dhadoop.log.file=hadoop-root-datanode-654fcd6
home.dir=/usr/local/hadoop-2.7.0 -Dhadoop.id.str=root -Dhadoop.root.logger=INFO,RFA -Djava.library.path=/usr/local/hadoop-2.7.0/lib/native -Dhadoop.policy.file=hadoop-
et.preferIPv4Stack=true -server -Dhadoop.security.logger=ERROR,RFAS -Dhadoop.security.logger=ERROR,RFAS -Dhadoop.security.logger=ERROR,RFAS -Dhadoop.security.logger=IN
hadoop.hdfs.server.datanode.DataNode
root       552     1 41 16:32 ?        00:00:02 /etc/alternatives/jre_1.8.0/bin/java -Dproc_secondarynamenode -Xmx1000m -Djava.net.preferIPv4Stack=true -Dhadoop.log.dir
2.7.0/logs -Dhadoop.log.file=hadoop.log -Dhadoop.home.dir= -Dhadoop.id.str=root -Dhadoop.root.logger=INFO,console -Djava.library.path= -Dhadoop.policy.file=hadoop-poli
preferIPv4Stack=true -Djava.net.preferIPv4Stack=true -Djava.net.preferIPv4Stack=true -Dhadoop.log.dir=/usr/local/hadoop-2.7.0/logs -Dhadoop.log.file=hadoop-root-seconda
968d.log -Dhadoop.home.dir=/usr/local/hadoop-2.7.0 -Dhadoop.id.str=root -Dhadoop.root.logger=INFO,RFA -Djava.library.path=/usr/local/hadoop-2.7.0/lib/native -Dhadoop.p
policy.xml -Djava.net.preferIPv4Stack=true -Dhadoop.security.logger=INFO,RFAS -Dhdfs.audit.logger=INFO,NullAppender -Dhadoop.security.logger=INFO,RFAS -Dhdfs.audit.logg
er -Dhadoop.security.logger=INFO,RFAS -Dhdfs.audit.logger=INFO,NullAppender -Dhadoop.security.logger=INFO,RFAS org.apache.hadoop.hdfs.server.namenode.SecondaryNameNode
```

如上图，启动成功。

启动 yarn 的相关进程

start-yarn.sh

yarn 启动后， 正常会有 ResourceManager 和 NodeManager 这两个进程

```
cal/hadoop-2.7.0/logs -Dhadoop.log.file=yarn-root-resourcemanager-654fcd61968d.log -Dyarn.log.file=yarn-root-resourcemanager-6
.root.logger=INFO,RFA -Dyarn.root.logger=INFO,RFA -Djava.library.path=/usr/local/hadoop-2.7.0/lib/native -Dyarn.policy.file=ha
s -Dyarn.log.dir=/usr/local/hadoop-2.7.0/logs -Dhadoop.log.file=yarn-root-resourcemanager-654fcd61968d.log -Dyarn.log.file=yar
/local/hadoop-2.7.0 -Dhadoop.home.dir=/usr/local/hadoop-2.7.0 -Dhadoop.root.logger=INFO,RFA -Dyarn.root.logger=INFO,RFA -Djava
/usr/local/hadoop-2.7.0/etc/hadoop:/usr/local/hadoop-2.7.0/etc/hadoop:/usr/local/hadoop-2.7.0/etc/hadoop:/usr/local/hadoop-2.7
adoop/common/*:/usr/local/hadoop-2.7.0/share/hadoop/hdfs:/usr/local/hadoop-2.7.0/share/hadoop/hdfs/lib/*:/usr/local/hadoop-2.7
yarn/lib/*:/usr/local/hadoop-2.7.0/share/hadoop/yarn/*:/usr/local/hadoop-2.7.0/share/hadoop/mapreduce/lib/*:/usr/local/hadoop-
ib/capacity-scheduler/*.jar:/contrib/capacity-scheduler/*.jar:/contrib/capacity-scheduler/*.jar:/usr/local/hadoop-2.7.0/share/
*:/usr/local/hadoop-2.7.0/etc/hadoop/rm-config/log4j.properties org.apache.hadoop.yarn.server.resourcemanager.ResourceManager
root       903     1 14 16:34 ?        00:00:04 /etc/alternatives/jre_1.8.0/bin/java -Dproc_nodemanager -Xmx1000m -Dhadoop.log
hadoop-2.7.0/logs -Dhadoop.log.file=yarn-root-nodemanager-654fcd61968d.log -Dyarn.log.file=yarn-root-nodemanager-654fcd61968d.
=INFO,RFA -Dyarn.root.logger=INFO,RFA -Djava.library.path=/usr/local/hadoop-2.7.0/lib/native -Dyarn.policy.file=hadoop-policy.
yarn.log.dir=/usr/local/hadoop-2.7.0/logs -Dhadoop.log.file=yarn-root-nodemanager-654fcd61968d.log -Dyarn.log.file=yarn-root-n
p-2.7.0 -Dhadoop.home.dir=/usr/local/hadoop-2.7.0 -Dhadoop.root.logger=INFO,RFA -Dyarn.root.logger=INFO,RFA -Djava.library.pat
adoop-2.7.0/etc/hadoop:/usr/local/hadoop-2.7.0/etc/hadoop:/usr/local/hadoop-2.7.0/etc/hadoop:/usr/local/hadoop-2.7.0/share/had
/*:/usr/local/hadoop-2.7.0/share/hadoop/hdfs:/usr/local/hadoop-2.7.0/share/hadoop/hdfs/lib/*:/usr/local/hadoop-2.7.0/share/had
usr/local/hadoop-2.7.0/share/hadoop/yarn/*:/usr/local/hadoop-2.7.0/share/hadoop/mapreduce/lib/*:/usr/local/hadoop-2.7.0/share/
b/capacity-scheduler/*.jar:/usr/local/hadoop-2.7.0/share/hadoop/yarn/*:/usr/local/hadoop-2.7.0/share/hadoop/yarn/lib/*:/usr/lo
pache.hadoop.yarn.server.nodemanager.NodeManager
```

3.7 验证 hadoop 已经正确启动

```
# 新建一个目录
 hadoop fs -mkdir /test
# 查看是否有对应目录了
 hadoop fs -ls /
```

```
[root@654fcd61968d local]#  hadoop fs -mkdir /test
[root@654fcd61968d local]#  hadoop fs -ls /
Found 1 items
drwxr-xr-x   - root supergroup          0 2020-03-29 16:37 /test
[root@654fcd61968d local]#
```

四、hive 环境安装

4.1 编译 hive

```
# 下载源码
git clone https://github.com/apache/hive.git
# 进入hive目录
cd hive


# 按照maven
wget http://mirrors.hust.edu.cn/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz
tar zxf apache-maven-3.6.3-bin.tar.gz
mv apache-maven-3.6.3 maven3
vim /etc/profile.d/maven.sh
# 输入以下内容
#!/bin/bash
export M2_HOME=/usr/local/maven3
export PATH=$PATH:$M2_HOME/bin

export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.242.b08-0.el8_1.x86_64
export CLASSPATH=.:$JAVA_HOME/jre/lib/rt.jar:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin

# 保存退出后，为该脚本添加可执行权限
chmod 744 /etc/profile.d/maven.sh

# 使环境变量的设置生效：
source /etc/profile.d/maven.sh

# 编译hive
mvn clean package -DskipTests -Pdist -Dmaven.javadoc.skip=true
```

```
[root@654fcd61968d local]# git clone https://github.com/apache/hive.git
Cloning into 'hive'...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 592221 (delta 2), reused 20 (delta 2), pack-reused 592188
Receiving objects: 100% (592221/592221), 485.74 MiB | 253.00 KiB/s, done.
Resolving deltas: 100% (344432/344432), done.
Checking out files: 100% (19459/19459), done.
```

```
# 待解决的错误
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:3.6.1:compile (default-compile) on project hive-exec: Compilation failure
[ERROR] /usr/local/hive/ql/src/java/org/apache/hadoop/hive/ql/io/orc/encoded/EncodedReaerImpl.java:[76,16] sun.misc.Cleaner is internal proprietary API and may be removed in a futur release
```

```
[ERROR]
[ERROR] -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the follow
ng articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MojoFailureException
[ERROR]
[ERROR] After correcting the problems, you can resume the build with the command
[ERROR]   mvn <args> -rf :hive-exec
```

4.2 安装 hive

Tips: 如果老是编译不过，可以用编译好的包，这里我用清华的镜像 2.3.6 版本的 hive

4.2.1 下载解压

```
# 下载hive二进制压缩包
wget https://mirrors.tuna.tsinghua.edu.cn/apache/hive/hive-2.3.6/apache-hive-2.3.6-bin.tar.gz

# 解压
tar -zxvf apache-hive-2.3.6-bin.tar.gz
```

4.2.2 配置

```
cp /usr/local/apache-hive-2.3.6-bin/conf/hive-default.xml.template /usr/local/apache-hive-2.3
6-bin/conf/hive-site.xml
vim /usr/local/apache-hive-2.3.6-bin/conf/hive-site.xml
```

填入如下配置

```
<property>
    <name>system:java.io.tmpdir</name>
    <value>/tmp/hive/java</value>
  </property>
  <property>
    <name>system:user.name</name>
    <value>${user.name}</value>
  </property>
```

4.3 初始化 hive 数据库

4.3.1 配置 MySQL 作为元数据库

```
# 为了让mac可以访问该mysql实例，我们将它的端口映射到3307上
docker run -p 3307:3306 --name mysql5.6 -e MYSQL_ROOT_PASSWORD=root -d mysql:5.6
# 在mac上进入该mysql交互界面，创建一个hive的元数据库
mysql -uroot -proot -P 3307 -h 127.0.0.1
create database hive;
```

之后通过docker inspect检查该容器的ip，我获取到的ip是172.17.0.3
docker inspect mysql5.6 | grep "IPAddress"

```
➜ hadoop docker run -p 3307:3306 --name mysql5.6 -e MYSQL_ROOT_PASSWORD=root -d mysql:5.6
Unable to find image 'mysql:5.6' locally
5.6: Pulling from library/mysql
6d28e14ab8c8: Pull complete
dda15103a86a: Pull complete
55971d75ab8c: Pull complete
f1d4ea32020b: Pull complete
61420072af91: Pull complete
30862a48418b: Pull complete
c6c2ee3a9a57: Pull complete
0f4efadb31df: Pull complete
dd931017b211: Pull complete
488a86083079: Pull complete
921d4bdabca2: Pull complete
Digest: sha256:a72a05bcf3914c902070765a506b1c8c17c06400258e7b574965763099dee9e1
Status: Downloaded newer image for mysql:5.6
d5a2ec5d0d23d0024fb91e3ad9fc98012e622321ac204f3e90a097084793e1f5
```

```
➜ hadoop mysql -uroot -proot -P 3307 -h 127.0.0.1
mysql: [Warning] Using a password on the command li
Welcome to the MySQL monitor.  Commands end with ;
Your MySQL connection id is 1
Server version: 5.6.47 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affilia

Oracle is a registered trademark of Oracle Corporat
affiliates. Other names may be trademarks of their
owners.

Type 'help;' or '\h' for help. Type '\c' to clear t

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
3 rows in set (0.00 sec)

mysql> create database hive;
Query OK, 1 row affected (0.00 sec)
```

```
➜ hadoop docker inspect mysql5.6 | grep "IPAddress"
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.3",
                    "IPAddress": "172.17.0.3",
```

4.3.2 配置 Hive

vim /usr/local/apache-hive-2.3.6-bin/conf/hive-site.xml

```
<property>
      <name>javax.jdo.option.ConnectionUserName</name>
      <value>root</value>
   </property>
   <property>
      <name>javax.jdo.option.ConnectionPassword</name>
      <value>root</value>
   </property>
   <property>
      <name>javax.jdo.option.ConnectionURL</name>
      <value>jdbc:mysql://172.17.0.3:3306/hive</value>
   </property>
   <property>
      <name>javax.jdo.option.ConnectionDriverName</name>
      <value>com.mysql.jdbc.Driver</value>
   </property>

# 配置环境变量
vim /etc/profile
export HIVE_HOME="/usr/local/apache-hive-2.3.6-bin"
export PATH=$PATH:$HIVE_HOME/bin
```

保存退出， 执行 source /etc/profile 让环境变量立即生效

4.3.3 下载 MySQL 驱动包

```
cd /usr/local/apache-hive-2.3.6-bin/lib
wget https://repo1.maven.org/maven2/mysql/mysql-connector-java/5.1.46/mysql-connector-
ava-5.1.46.jar
```

4.3.4 初始化元数据库

```
schematool -initSchema -dbType mysql
```

```
[root@654fcd61968d conf]# schematool -initSchema -dbType mysql
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/apache-hive-2.3.6-b
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.7.0/share/
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4j
Metastore connection URL:        jdbc:mysql://172.17.0.3:3306/hi
Metastore Connection Driver :    com.mysql.jdbc.Driver
Metastore connection User:       root
Starting metastore schema initialization to 2.3.0
Initialization script hive-schema-2.3.0.mysql.sql
Initialization script completed
schemaTool completed
[root@654fcd61968d conf]#
```

4.4 启动 Hiveserver2

4.4.1 需要先往 hdfs 的 core-site.xml 加入以下配置

vim /usr/local/hadoop-2.7.0/etc/hadoop/core-site.xml

```
<property>
   <name>hadoop.proxyuser.root.hosts</name>
   <value>*</value>
</property>
<property>
   <name>hadoop.proxyuser.root.groups</name>
   <value>*</value>
</property>
```

4.4.2 然后重启 hdfs:

```
stop-dfs.sh
start-dfs.sh
```

4.4.3 后台启动 hiveserver2

```
nohup hiveserver2 &
```

4.4.4 验证

```
beeline -u jdbc:hive2://127.0.0.1:10000
show databases;
```