**Hive安装与配置**

# 上载和解压缩

解压

tar -zxvf apache-hive-2.1.1-bin.tar.gz

把解压好的移到/user/local/下

mv apache-hive-2.1.1-bin /usr/local/hive

# 配置环境变量

#编辑/etc/profile，添加hive相关的环境变量配置

vim /etc/profile

在文件结尾添加内容如下：

export HIVE\_HOME=/usr/local/hive

export PATH=$PATH:$HIVE\_HOME/bin

修改完文件后，执行如下命令，让配置生效：

source /etc/profile

# Hive-site.xml配置

## 配置Hadoop HDFS

进入目录$HIVE\_HOME/conf，将hive-default.xml.template文件复制一份并改名为hive-site.xml

使用hadoop新建hdfs目录,因为在hive-site.xml中有如下配置：

<property>

<name>hive.metastore.warehouse.dir</name>

<value>/user/hive/warehouse</value>

<description>location of default database for the warehouse</description>

</property>

<property>

执行hadoop命令新建/user/hive/warehouse目录：

新建目录/user/hive/warehouse

$HADOOP\_HOME/bin/hadoop dfs -mkdir -p /user/hive/warehouse

给新建的目录赋予读写权限

sh $HADOOP\_HOME/bin/hdfs dfs -chmod 777 /user/hive/warehouse

查看修改后的权限

sh $HADOOP\_HOME/bin/hdfs dfs -ls /user/hive

Found 1 items

drwxrwxrwx - impala supergroup 0 2017-06-06 01:46 /user/hive/warehouse

运用hadoop命令新建/tmp/hive目录

$HADOOP\_HOME/bin/hdfs dfs -mkdir -p /tmp/hive

给目录/tmp/hive赋予读写权限

$HADOOP\_HOME/bin/hdfs dfs -chmod 777 /tmp/hive

检查创建好的目录

$HADOOP\_HOME/bin/hdfs dfs -ls /tmp

Found 1 items

drwxrwxrwx - hadoop supergroup 0 2017-06-06 05:06 /tmp/hive

## 配置中的临时目录

将hive-site.xml文件中的${system:java.io.tmpdir}替换为hive的临时目录，例如我替换为$HIVE\_HOME/tmp，该目录如果不存在则要自己手工创建，并且赋予读写权限。

cd $HIVE\_HOME

mkdir tmp

配置文件hive-site.xml:

将文件中的所有 ${system:java.io.tmpdir}替换成/usr/local/hive/tmp

将文件中所有的${system:user.name}替换为root

## 配置mysql

上传并把mysql的驱动包上传到Hive的lib目录下：

cp mysql-connector-java-5.1.36.jar $HIVE\_HOME/lib

编辑 hive-site.xml

搜索javax.jdo.option.connectionURL,将该name对应的value修改为MySQL的地址:

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://192.168.56.181:3306/hive?createDatabaseIfNotExist=true</value>

<description>

JDBC connect string for a JDBC metastore.

To use SSL to encrypt/authenticate the connection, provide database-specific SSL flag in the connection URL.

For example, jdbc:postgresql://myhost/db?ssl=true for postgres database.

</description>

</property>

搜索javax.jdo.option.ConnectionDriverName，将该name对应的value修改为MySQL驱动类路径:

<property>

<name>javax.jdo.option.ConnectionDriverName</name>

<value>com.mysql.jdbc.Driver</value>

<description>Driver class name for a JDBC metastore</description>

</property>

<property>

搜索javax.jdo.option.ConnectionUserName，将对应的value修改为MySQL数据库登录名:

<property>

<name>javax.jdo.option.ConnectionUserName</name>

<value>root</value>

<description>Username to use against metastore database</description>

</property>

搜索javax.jdo.option.ConnectionPassword，将对应的value修改为MySQL数据库的登录密码:

<property>

<name>javax.jdo.option.ConnectionPassword</name>

<value>Love88me</value>

<description>password to use against metastore database</description>

</property>

搜索hive.metastore.schema.verification，将对应的value修改为false：

<property>

<name>hive.metastore.schema.verification</name>

<value>false</value>

<description>

Enforce metastore schema version consistency.

True: Verify that version information stored in is compatible with one from Hive jars. Also disable automatic

schema migration attempt. Users are required to manually migrate schema after Hive upgrade which ensures

proper metastore schema migration. (Default)

False: Warn if the version information stored in metastore doesn't match with one from in Hive jars.

</description>

</property>

# hive-env.sh配置

进入目录

cd $HIVE\_HOME/conf

将hive-env.sh.template 复制一份并重命名为hive-env.sh

cp hive-env.sh.template hive-env.sh

打开hive-env.sh并添加如下内容

vi hive-env.sh

export HADOOP\_HOME=/home/hadoop/hadoop2.7.3

export HIVE\_CONF\_DIR=/usr/local/hive/conf

export HIVE\_AUX\_JARS\_PATH=/usr/local/hive/lib

## hive日志路径设置

复制hive-log4j2.properties.template 为hive-log4j2.properties，修改属性property.hive.log.dir

# 启动和测试

## 初始化MySQL

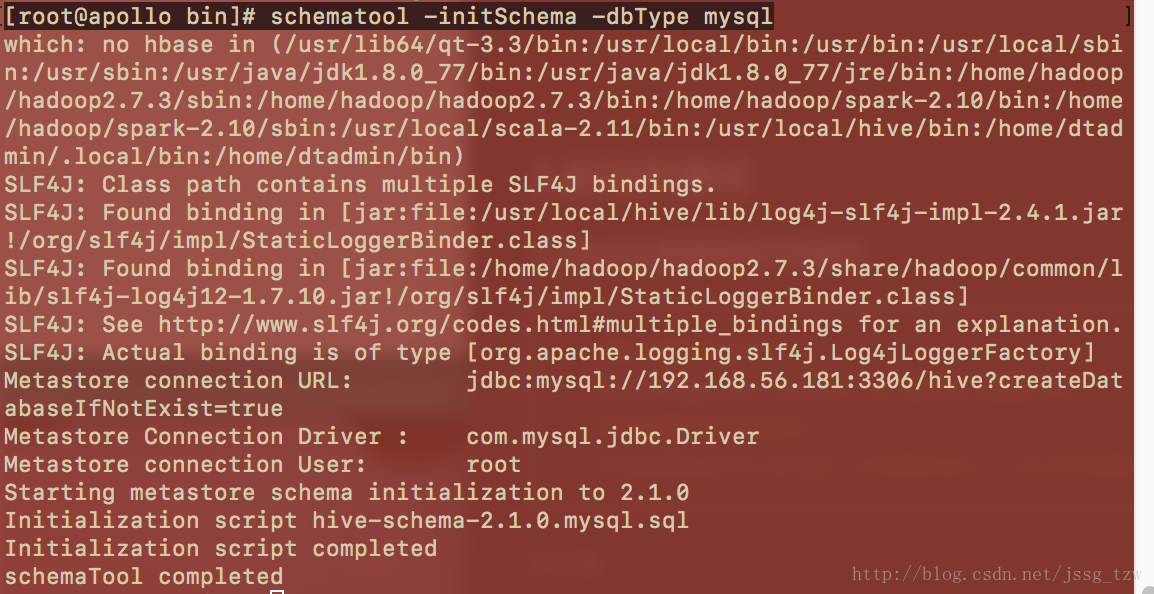
进入$HIVE/bin

cd $HIVE\_HOME/bin

对数据库进行初始化：

schematool -initSchema -dbType mysql

如下图：



执行成功后，查看mysql数据库

## 启动Hive

./hive

这种方式可能会在ssh退出后服务停止，所以加上nohup命令更稳妥。

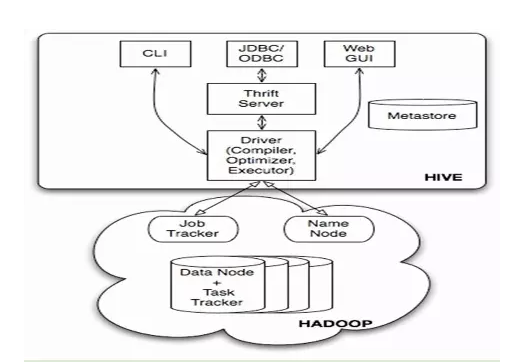
# Metastore与hiveserver2

<https://www.jianshu.com/p/367c2a558d9b>

<https://blog.csdn.net/qq_21383435/article/details/81670355>

在hive中有metaServer与hiveServer2两种服务，看了好多文章说这两个的区别，文章内容有对有错，不够全面，故在这里好好总结一下。

首先，下面这个hive构架图，我们一定不陌生，它反应出hive有哪些组件结构



当然下面的图是hadoop1的部分，现在JobTracker是Yarn了  
上面的部分是访问Hive的三个入口，

1：直接Cli

2:通过JDBC

3：webUI

当我们要连接Hive进行操作时，首先必须是安装了，安装hive很简单，直接在conf/hive-site配置存放Hive元数据的连接信息，通常是用mysql，如下：

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://localhost:3306/hive?createDatabaseIfNotExist=true</value></property><property>

<name>javax.jdo.option.ConnectionDriverName</name>

<value>com.mysql.jdbc.Driver</value></property><property>

<name>javax.jdo.option.ConnectionUserName</name>

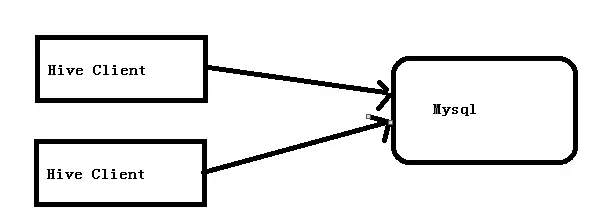
<value>root</value></property><property>

<name>javax.jdo.option.ConnectionPassword</name>

<value>root</value>

</property>

这样配置好后，在hive的lib里加入Mysql JDBC 驱动后，我们就可以通过 bin/hive的方式进行hive客户端，请求数据。如下图：

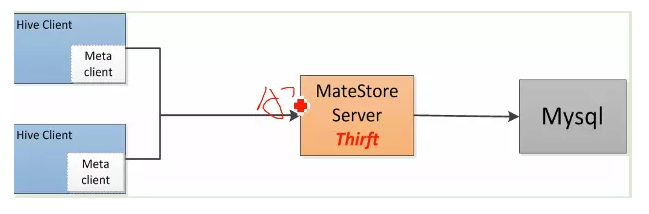


上图所示方式的最大弊端是：如果元数据Mysql库无法启动，要替换从库时，或者我们要移植元数据到另一台机器上时，那么所有的hive cli全要修改一遍！

MetaStoreServer恰好可以解决这一问题。

启动：nohup $HIVE\_HOME/bin/hive –metastore &

如果不加nohup，在终端退出后进程就会被kill掉。



如图，在Hive cli与mysql中间启动一个MetaStoreServer，这样Hive cli就不需要连接Mysql，直接连接这个MetaStoreServer。  
在hive-site.xml配置：

<property>

<name>hive.metastore.uris</name>

<value>thrift://xxxxxx:9083</value>

<description

</property>

这样通过metaserver取得元数据的信息。  
当然上面的图只是一个MetaStoreServer，存在单点问题，但我们完全可以配置两个或者多个MetaStoreServer，就实现了负载均衡与容错的功能了，如下面的配置

<property>

<name>hive.metastore.uris</name>

<value>thrift://dw1:9083,thrift://dw2:9083</value>

<description>A comma separated list of metastore uris on which metastore service is running</description>

</property>

HiveServer2待续

# 问题

https://blog.csdn.net/cjfeii/article/details/49363653