**一、安装Hive3.1.2**

**1. 下载并解压Hive安装包**  
首先需要下载Hive安装包文件， [Hive官网下载地址](http://www.apache.org/dyn/closer.cgi/hive/)  
也可以直接[点击这里从百度云盘下载软件](https://pan.baidu.com/s/1lLjW9cmS1tkBHkrqtpkjWw)（提取码：ziyu）。进入百度网盘后，进入“软件”目录，找到apache-hive-3.1.2-bin.tar.gz文件，下载到本地。

1. sudo tar -zxvf ./apache-hive-3.1.2-bin.tar.gz -C /usr/local *# 解压到/usr/local中*
2. cd /usr/local/
3. sudo mv apache-hive-3.1.2-bin hive *# 将文件夹名改为hive*
4. sudo chown -R dblab:dblab hive *# 修改文件权限*

Shell 命令

注意，上面的dblab:dblab是用户组和用户名，如果你当前使用用户名hadoop登录了Linux系统，则把dblab替换成hadoop。  
**2. 配置环境变量**  
为了方便使用，我们把hive命令加入到环境变量中去，  
请使用vim编辑器打开.bashrc文件，命令如下：

1. vim ~/.bashrc

Shell 命令

在该文件最前面一行添加如下内容：

export HIVE\_HOME=/usr/local/hive

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

export HADOOP\_HOME=/usr/local/hadoop

HADOOP\_HOME需要被配置成你机器上Hadoop的安装路径，比如这里是安装在/usr/local./hadoop目录。  
保存退出后，运行如下命令使配置立即生效：

1. source ~/.bashrc

Shell 命令

**3. 修改/usr/local/hive/conf下的hive-site.xml**  
执行如下命令：

1. cd /usr/local/hive/conf
2. mv hive-default.xml.template hive-default.xml

Shell 命令

上面命令是将hive-default.xml.template重命名为hive-default.xml；  
然后，使用vim编辑器新建一个配置文件hive-site.xml，命令如下：

1. cd /usr/local/hive/conf
2. vim hive-site.xml

Shell 命令

在hive-site.xml中添加如下配置信息：

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>

<property>

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

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

<description>JDBC connect string for a JDBC metastore</description>

</property>

<property>

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

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

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

</property>

<property>

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

<value>hive</value>

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

</property>

<property>

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

<value>hive</value>

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

</property>

</configuration>

然后，按键盘上的“ESC”键退出vim编辑状态，再输入:wq，保存并退出vim编辑器。

**二、安装并配置mysql**

这里我们采用MySQL数据库保存Hive的元数据，而不是采用Hive自带的derby来存储元数据。  
**1.Ubuntu下mysql的安装请参考**：[Ubuntu安装MySQL](http://dblab.xmu.edu.cn/blog/install-mysql/)  
**2.下载mysql jdbc 包**,[下载地址](http://www.mysql.com/downloads/connector/j/)

1. tar -zxvf mysql-connector-java-5.1.40.tar.gz *#解压*
2. cp mysql-connector-java-5.1.40/mysql-connector-java-5.1.40-bin.jar /usr/local/hive/lib *#将mysql-connector-java-5.1.40-bin.jar拷贝到/usr/local/hive/lib目录下*

Shell 命令

**3. 启动并登陆mysql shell**

1. service mysql start *#启动mysql服务*
2. mysql -u root -p *#登陆shell界面*

Shell 命令

**4. 新建hive数据库**。

1. mysql> create database hive; *#这个hive数据库与hive-site.xml中localhost:3306/hive的hive对应，用来保存hive元数据*

mysql

**5. 配置mysql允许hive接入：**

1. mysql> grant all on \*.\* to hive@localhost identified by 'hive'; *#将所有数据库的所有表的所有权限赋给hive用户，后面的hive是配置hive-site.xml中配置的连接密码*
2. mysql> flush privileges; *#刷新mysql系统权限关系表*

mysql

**6. 启动hive**  
启动hive之前，请先启动hadoop集群。

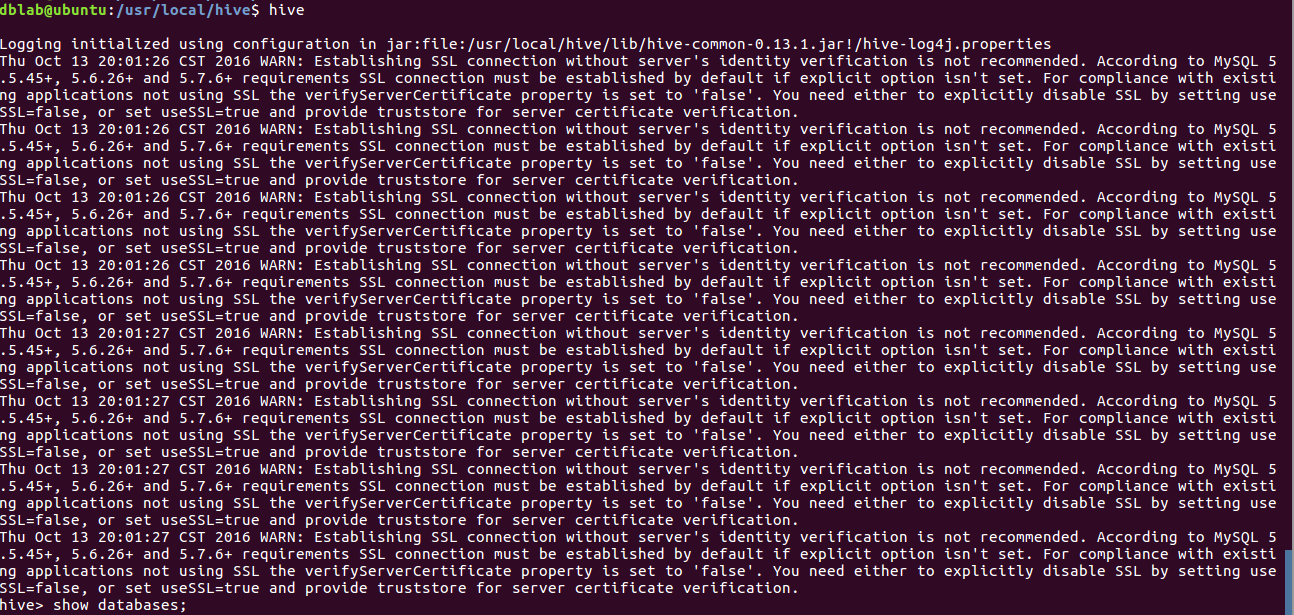
1. start-dfs.sh *#启动Hadoop的HDFS*
2. hive *#启动hive*

Shell 命令

注意，我们这里已经配置了PATH，所以，不要把start-all.sh和hive命令的路径加上。如果没有配置PATH，请加上路径才能运行命令，比如，本教程Hadoop安装目录是“/usr/local/hadoop”，Hive的安装目录是“/usr/local/hive”，因此，启动hadoop和hive，也可以使用下面带路径的方式：

1. cd /usr/local/hadoop *#进入Hadoop安装目录*
2. ./sbin/start-dfs.sh
3. cd /usr/local/hive
4. ./bin/hive

Shell 命令

使用mysql作为元数据库时登陆

启动Hive过程中，可能出现的错误和解决方案如下：

【错误1】  
java.lang.NoSuchMethodError: com.google.common.base.Preconditions.checkArgument  
【原因】  
com.google.common.base.Preconditions.checkArgument 这是因为hive内依赖的guava.jar和hadoop内的版本不一致造成的。  
【解决方法】  
1.查看hadoop安装目录下share/hadoop/common/lib内guava.jar版本  
2.查看hive安装目录下lib内guava.jar的版本 如果两者不一致，删除版本低的，并拷贝高版本的 问题解决！

【错误2】org.datanucleus.store.rdbms.exceptions.MissingTableException: Required table missing : “VERSION” in Catalog “” Schema “”. DataNucleus requires this table to perform its persistence operations.

【解决方案】  
进入hive安装目录（比如/usr/local/hive），执行如下命令：./bin/schematool -dbType mysql -initSchema

【错误3】在启动Hive时，有可能会出现Hive metastore database is not initialized的错误，这里给出解决方案。  
【解决方案】  
解决Hive启动，Hive metastore database is not initialized的错误。出错原因：以前曾经安装了Hive或MySQL，重新安装Hive和MySQL以后，导致版本、配置不一致。解决方法是，使用schematool工具。Hive现在包含一个用于 Hive Metastore 架构操控的脱机工具，名为 schematool.此工具可用于初始化当前 Hive 版本的 Metastore 架构。此外，其还可处理从较旧版本到新版本的架构升级。所以，解决上述错误，你可以在终端执行如下命令:

1. cd /usr/local/hive
2. ./bin/schematool -dbType mysql -initSchema

Shell 命令

执行后，再启动Hive，应该就正常了。

启动进入Hive的交互式执行环境以后，会出现如下命令提示符：

1. hive>

hive

可以在里面输入SQL语句，如果要退出Hive交互式执行环境，可以输入如下命令：

1. hive>exit;

hive

**三、Hive的常用HiveQL操作**

1）Hive基本数据类型

首先，我们简单叙述一下HiveQL的基本数据类型。

Hive支持基本数据类型和复杂类型, 基本数据类型主要有数值类型(INT、FLOAT、DOUBLE ) 、布尔型和字符串, 复杂类型有三种:ARRAY、MAP 和 STRUCT。

a.基本数据类型

* TINYINT: 1个字节
* SMALLINT: 2个字节
* INT: 4个字节
* BIGINT: 8个字节
* BOOLEAN: TRUE/FALSE
* FLOAT: 4个字节，单精度浮点型
* DOUBLE: 8个字节，双精度浮点型STRING 字符串

b.复杂数据类型

* ARRAY: 有序字段
* MAP: 无序字段
* STRUCT: 一组命名的字段

2）常用的HiveQL操作命令

Hive常用的HiveQL操作命令主要包括：数据定义、数据操作。接下来详细介绍一下这些命令即用法（想要了解更多请参照《Hive编程指南》一书）。

a.数据定义：主要用于创建修改和删除数据库、表、视图、函数和索引。

**创建、修改和删除数据库**

1. create database if not exists hive; *#创建数据库*
2. show databases; *#查看Hive中包含数据库*
3. show databases like 'h.\*'; *#查看Hive中以h开头数据库*
4. describe databases; *#查看hive数据库位置等信息*
5. alter database hive set dbproperties; *#为hive设置键值对属性*
6. use hive; *#切换到hive数据库下*
7. drop database if exists hive; *#删除不含表的数据库*
8. drop database if exists hive cascade; *#删除数据库和它中的表*

sql

注意，除 dbproperties属性外，数据库的元数据信息都是不可更改的，包括数据库名和数据库所在的目录位置，没有办法删除或重置数据库属性。

**创建、修改和删除表**

1. *#创建内部表（管理表）*
2. create table if not exists hive.usr(
3. name string comment 'username',
4. pwd string comment 'password',
5. address struct<street:string,city:string,state:string,zip:int>,
6. comment 'home address',
7. identify map<int,tinyint> comment 'number,sex')
8. comment 'description of the table'
9. tblproperties('creator'='me','time'='2016.1.1');
10. *#创建外部表*
11. create external table if not exists usr2(
12. name string,
13. pwd string,
14. address struct<street:string,city:string,state:string,zip:int>,
15. identify map<int,tinyint>)
16. row format delimited fields terminated by ','
17. location '/usr/local/hive/warehouse/hive.db/usr';
18. *#创建分区表*
19. create table if not exists usr3(
20. name string,
21. pwd string,
22. address struct<street:string,city:string,state:string,zip:int>,
23. identify map<int,tinyint>)
24. partitioned by(city string,state string);
25. *#复制usr表的表模式*
26. create table if not exists hive.usr1 like hive.usr;
28. show tables in hive;
29. show tables 'u.\*'; *#查看hive中以u开头的表*
30. describe hive.usr; *#查看usr表相关信息*
31. alter table usr rename to custom; *#重命名表*
33. *#为表增加一个分区*
34. alter table usr2 add if not exists
35. partition(city=”beijing”,state=”China”)
36. location '/usr/local/hive/warehouse/usr2/China/beijing';
37. *#修改分区路径*
38. alter table usr2 partition(city=”beijing”,state=”China”)
39. set location '/usr/local/hive/warehouse/usr2/CH/beijing';
40. *#删除分区*
41. alter table usr2 drop if exists partition(city=”beijing”,state=”China”)
42. *#修改列信息*
43. alter table usr change column pwd password string after address;
45. alter table usr add columns(hobby string); *#增加列*
46. alter table usr replace columns(uname string); *#删除替换列*
47. alter table usr set tblproperties('creator'='liming'); *#修改表属性*
48. alter table usr2 partition(city=”beijing”,state=”China”) *#修改存储属性*
49. set fileformat sequencefile;
50. use hive; *#切换到hive数据库下*
51. drop table if exists usr1; *#删除表*
52. drop database if exists hive cascade; *#删除数据库和它中的表*

sql

**视图和索引的创建、修改和删除**

主要语法如下，用户可自行实现。

1. create view view\_name as....; *#创建视图*
2. alter view view\_name set tblproperties(…); *#修改视图*

sql

因为视图是只读的，所以 对于视图只允许改变元数据中的 tblproperties属性。

1. *#删除视图*
2. drop view if exists view\_name;
3. *#创建索引*
4. create index index\_name on table table\_name(partition\_name/column\_name)
5. as 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler' with deferred rebuild....;

sql

这里’org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler’是一个索引处理器，即一个实现了索引接口的Java类，另外Hive还有其他的索引实现。

1. alter index index\_name on table table\_name partition(...) rebulid; *#重建索引*

sql

如果使用 deferred rebuild，那么新索引成空白状态，任何时候可以进行第一次索引创建或重建。

1. show formatted index on table\_name; *#显示索引*
2. drop index if exists index\_name on table table\_name; *#删除索引*

sql

**用户自定义函数**

在新建用户自定义函数（UDF）方法前，先了解一下Hive自带的那些函数。show functions; 命令会显示Hive中所有的函数名称：



若想要查看具体函数使用方法可使用describe function 函数名：



首先编写自己的UDF前需要继承UDF类并实现evaluate()函数，或是继承GenericUDF类实现initialize()函数、evaluate()函数和getDisplayString()函数，还有其他的实现方法，感兴趣的用户可以自行学习。

另外，如果用户想在Hive中使用该UDF需要将我们编写的Java代码进行编译，然后将编译后的UDF二进制类文件(.class文件)打包成一个JAR文件，然后在Hive会话中将这个JAR文件加入到类路径下，在通过create function语句定义好使用这个Java类的函数。

1. add jar <jar文件的绝对路径>; *#创建函数*
2. create temporary function function\_name;
3. drop temporary function if exists function\_name; *#删除函数*

sql

3）数据操作

主要实现的是将数据装载到表中（或是从表中导出），并进行相应查询操作，对熟悉SQL语言的用户应该不会陌生。

**向表中装载数据**

这里我们以只有两个属性的简单表为例来介绍。首先创建表stu和course，stu有两个属性id与name，course有两个属性cid与sid。

1. create table if not exists hive.stu(id int,name string)
2. row format delimited fields terminated by '\t';
3. create table if not exists hive.course(cid int,sid int)
4. row format delimited fields terminated by '\t';

sql

向表中装载数据有两种方法：从文件中导入和通过查询语句插入。

a.从文件中导入

假如这个表中的记录存储于文件stu.txt中，该文件的存储路径为/usr/local/hadoop/examples/stu.txt，内容如下。

stu.txt：

1 xiapi

2 xiaoxue

3 qingqing

下面我们把这个文件中的数据装载到表stu中，操作如下：

1. load data local inpath '/usr/local/hadoop/examples/stu.txt' overwrite into table stu;

sql

如果stu.txt文件存储在HDFS 上，则不需要 local 关键字。

b.通过查询语句插入

使用如下命令，创建stu1表，它和stu表属性相同，我们要把从stu表中查询得到的数据插入到stu1中：

1. create table stu1 as select id,name from stu;

sql

上面是创建表，并直接向新表插入数据；若表已经存在，向表中插入数据需执行以下命令：

1. insert overwrite table stu1 select id,name from stu where（条件）;

sql

这里关键字overwrite的作用是替换掉表（或分区）中原有数据，换成into关键字，直接追加到原有内容后。

**从表中导出数据**

a.可以简单拷贝文件或文件夹

命令如下：

1. hadoop fs -cp source\_path target\_path;

Shell

b.写入临时文件

命令如下：

1. insert overwrite local directory '/usr/local/hadoop/tmp/stu' select id,name from stu;

sql

**查询操作**

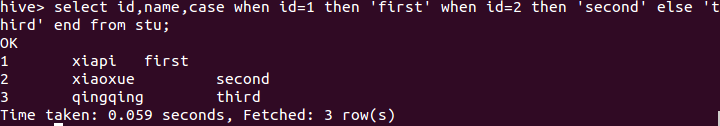
和SQL的查询完全一样，这里不再赘述。主要使用select…from…where…等语句，再结合关键字group by、having、like、rlike等操作。这里我们简单介绍一下SQL中没有的case…when…then…句式、join操作和子查询操作。

case…when…then…句式和if条件语句类似，用于处理单个列的查询结果，语句如下：

1. select id,name,
2. case
3. when id=1 then 'first'
4. when id=2 then 'second'
5. else 'third'

sql

结果如下：



**连接**  
连接（join）是将两个表中在共同数据项上相互匹配的那些行合并起来, HiveQL 的连接分为内连接、左向外连接、右向外连接、全外连接和半连接 5 种。

a. 内连接(等值连接)  
内连接使用比较运算符根据每个表共有的列的值匹配两个表中的行。

首先，我们先把以下内容插入到course表中（自行完成）。

1 3

2 1

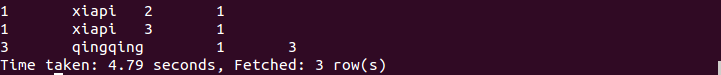
3 1

下面, 查询stu和course表中学号相同的所有行，命令如下：

1. select stu.\*, course.\* from stu join course on(stu .id=course .sid);

sql

执行结果如下：

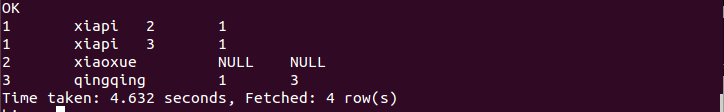


b. 左连接  
左连接的结果集包括“LEFT OUTER”子句中指定的左表的所有行, 而不仅仅是连接列所匹配的行。如果左表的某行在右表中没有匹配行, 则在相关联的结果集中右表的所有选择列均为空值，命令如下：

1. select stu.\*, course.\* from stu left outer join course on(stu .id=course .sid);

sql

执行结果如下：

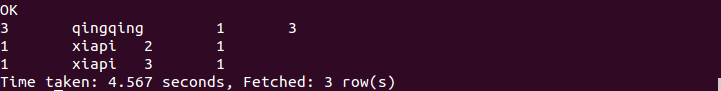


c. 右连接  
右连接是左向外连接的反向连接,将返回右表的所有行。如果右表的某行在左表中没有匹配行,则将为左表返回空值。命令如下：

1. select stu.\*, course.\* from stu right outer join course on(stu .id=course .sid);

sql

执行结果如下：

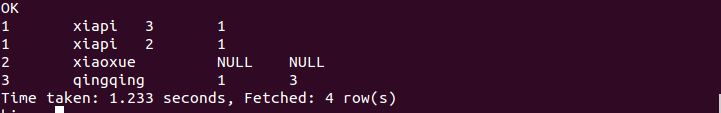


d. 全连接  
全连接返回左表和右表中的所有行。当某行在另一表中没有匹配行时,则另一个表的选择列表包含空值。如果表之间有匹配行,则整个结果集包含基表的数据值。命令如下：

1. select stu.\*, course.\* from stu full outer join course on(stu .id=course .sid);

sql

执行结果如下：

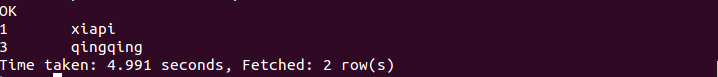


e. 半连接  
半连接是 Hive 所特有的, Hive 不支持 in 操作,但是拥有替代的方案; left semi join, 称为半连接, 需要注意的是连接的表不能在查询的列中,只能出现在 on 子句中。命令如下：

1. select stu.\* from stu left semi join course on(stu .id=course .sid);

sql

执行结果如下：



**子查询**  
标准 SQL 的子查询支持嵌套的 select 子句,HiveQL 对子查询的支持很有限,只能在from 引导的子句中出现子查询。

**注意，在定义或是操作表时，不要忘记指定所需数据库。**

**四、Hive简单编程实践**

下面我们以词频统计算法为例，来介绍怎么在具体应用中使用Hive。词频统计算法又是最能体现MapReduce思想的算法之一，这里我们可以对比它在MapReduce中的实现，来说明使用Hive后的优势。

MapReduce实现词频统计的代码可以通过下载Hadoop源码后，在 $HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar 包中找到(wordcount类)，wordcount类由63行Java代码编写而成。下面首先简单介绍一下怎么使用MapReduce中wordcount类来统计单词出现的次数，具体步骤如下：

1）创建input目录，output目录会自动生成。其中input为输入目录，output目录为输出目录。命令如下：

1. cd /usr/local/hadoop
2. mkdir input

Shell

2）然后，在input文件夹中创建两个测试文件file1.txt和file2.txt，命令如下：

1. cd /usr/local/hadoop/input
2. echo "hello world" > file1.txt
3. echo "hello hadoop" > file2.txt

Shell

3）执行如下hadoop命令：

1. cd ..
2. hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar wordcount input output

Shell

4）我们可以到output文件夹中查看结果，结果如下：

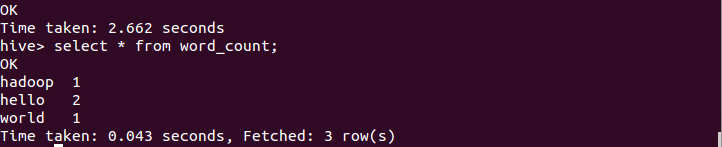


下面我们通过HiveQL实现词频统计功能，此时只要编写下面7行代码，而且不需要进行编译生成jar来执行。HiveQL实现命令如下：

1. create table docs(line string);
2. load data inpath 'file:///usr/local/hadoop/input' overwrite into table docs;
3. create table word\_count as
4. select word, count(1) as count from
5. (select explode(split(line,' '))as word from docs) w
6. group by word
7. order by word;

sql

执行后，用select语句查看，结果如下：



由上可知，采用Hive实现最大的优势是，对于非程序员，不用学习编写Java MapReduce代码了，只需要用户学习使用HiveQL就可以了，而这对于有SQL基础的用户而言是非常容易的。