

Hive 简介及安装部署

目录

1 HIVE介绍	3
1.1 HIVE介绍.....	3
1.2 HIVE运行架构.....	4
1.3 HIVE数据模型.....	6
1.4 HIVE数据类型.....	8
1.5 HIVE与关系数据库的区别.....	9
2 HIVE搭建过程	10
2.1 安装MySQL数据库.....	10
2.1.1 下载mysql安装文件.....	10
2.1.2 上传mysql安装文件.....	10
2.1.3 卸载旧的mysql.....	11
2.1.4 安装mysql.....	12
2.1.5 设置root密码.....	13
2.1.6 设置hive用户.....	15
2.1.7 创建hive数据库.....	15
2.2 安装HIVE.....	16
2.2.1 下载hive安装文件.....	16
2.2.2 下载mysql驱动.....	16
2.2.3 上传hive安装文件和mysql驱动.....	17
2.2.4 解压缩.....	17
2.2.5 把mysql驱动放到hive的lib目录下.....	18
2.2.6 配置/etc/profile环境变量.....	18
2.2.7 设置hive-env.sh配置文件.....	19
2.2.8 设置hive-site.xml配置文件.....	20
2.3 启动并验证HIVE.....	22
2.3.1 启动Hive.....	22
2.3.2 在hive中操作.....	23
3 问题解决	24
3.1 设置MySQL数据库ROOT用户密码报错.....	24
3.2 HIVE启动,报COMMANDNEEDRETRYEXCEPTION异常.....	26
3.3 在HIVE中使用操作语言.....	27

Hive 简介及安装部署

1 Hive 介绍

1.1 Hive 介绍

Hive 是一个基于 Hadoop 的开源数据仓库工具，用于存储和处理海量结构化数据。它是 Facebook 2008 年 8 月开源的一个数据仓库框架，提供了类似于 SQL 语法的 HQL 语句作为数据访问接口，Hive 有如下优缺点：

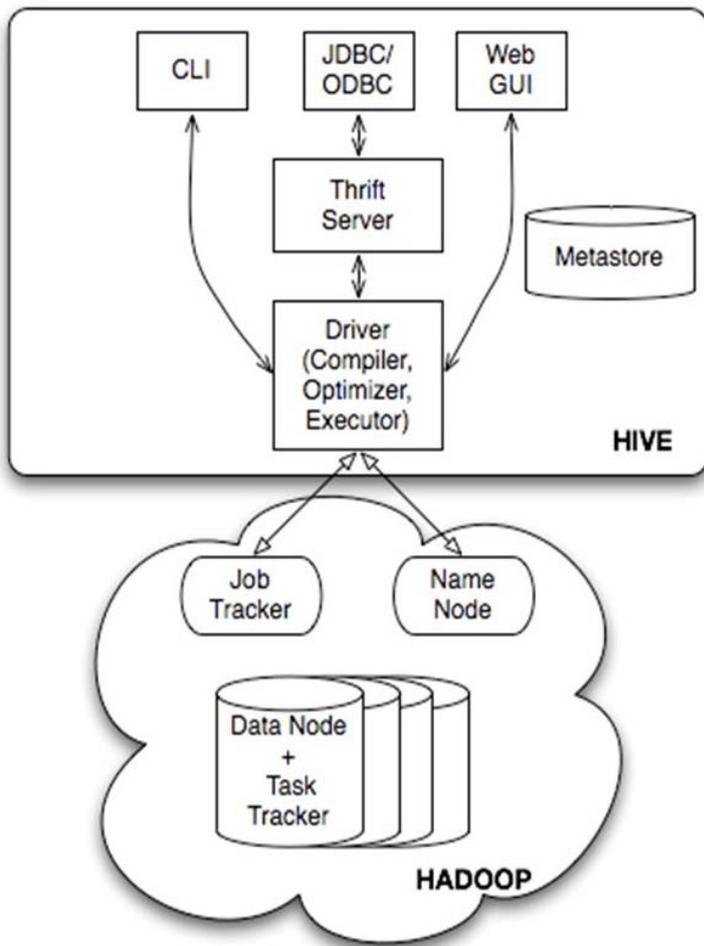
- **优点：**

1. Hive 使用类 SQL 查询语法，最大限度的实现了和 SQL 标准的兼容，大大降低了传统数据分析人员学习的曲线；
2. 使用 JDBC 接口/ODBC 接口，开发人员更易开发应用；
3. 以 MR 作为计算引擎、HDFS 作为存储系统，为超大数据集设计的计算/扩展能力；
4. 统一的元数据管理（Derby、MySql 等），并可与 Pig、Presto 等共享；

- **缺点：**

1. Hive 的 HQL 表达的能力有限，有些复杂运算用 HQL 不易表达；
2. 由于 Hive 自动生成 MapReduce 作业，HQL 调优困难；
3. 粒度较粗，可控性差

1.2 Hive 运行架构



由上图可知，Hadoop 的 MapReduce 是 Hive 架构的根基。Hive 架构包括如下组件：CLI (Command Line Interface)、JDBC/ODBC、Thrift Server、WEB GUI、Metastore 和 Driver(Compiler、Optimizer 和 Executor)，这些组件分为两大类：服务端组件和客户端组件。

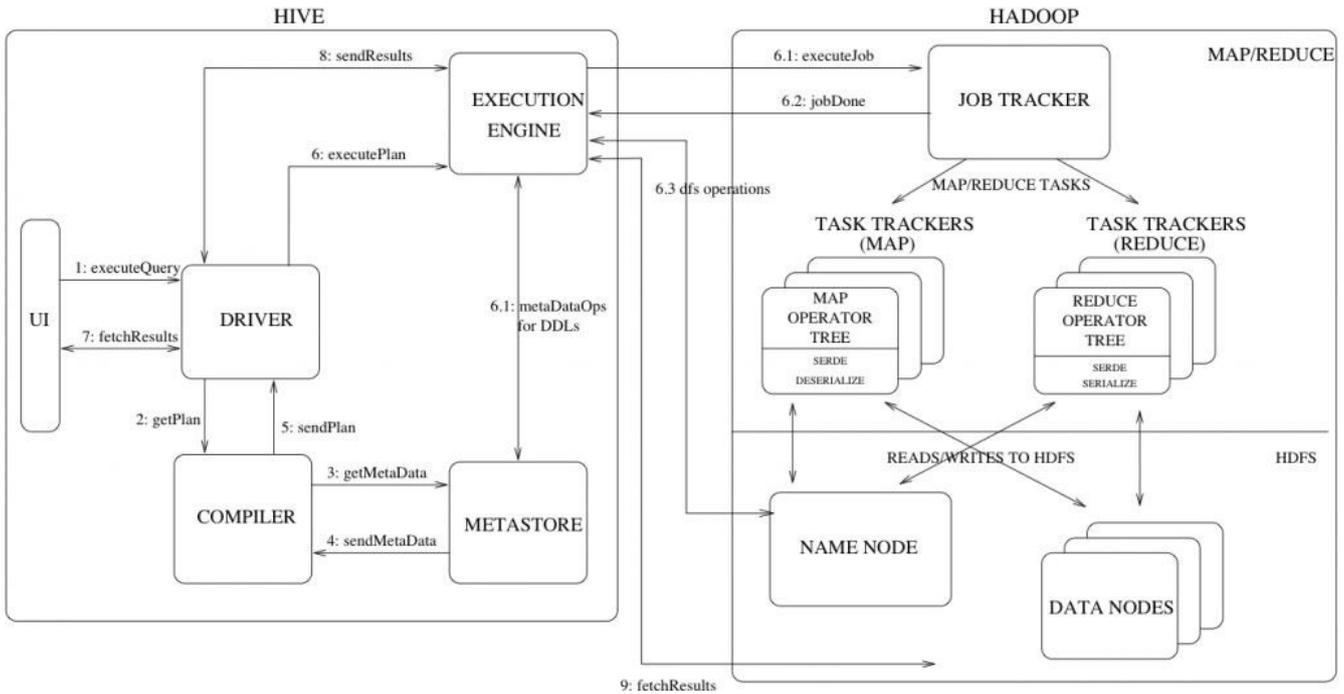
服务端组件：

- Driver 组件：该组件包括 Compiler、Optimizer 和 Executor，它的作用是将 HiveQL (类 SQL) 语句进行解析、编译优化，生成执行计划，然后调用底层的 MapReduce 计算框架；
- Metastore 组件：元数据服务组件，这个组件存储 Hive 的元数据，Hive 的元数据存储在关系数据库里，Hive 支持的关系数据库有 Derby 和 Mysql。元数据对于 Hive 十分重要，因此 Hive 支持把 Metastore 服务独立出来，安装到远程的服务器集群里，从而解耦 Hive 服务和 Metastore 服务，保证 Hive 运行的健壮性；
- Thrift 服务：Thrift 是 Facebook 开发的一个软件框架，它用来进行可扩展且跨语言的服务的开发，Hive 集成了该服务，能让不同的编程语言调用 Hive 的接口。

客户端组件：

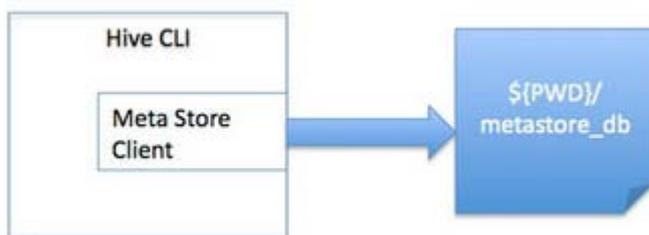
- CLI : Command Line Interface , 命令行接口。
- Thrift 客户端：上面的架构图里没有写上 Thrift 客户端，但是 Hive 架构的许多客户端接口是建立在 Thrift 客户端之上，包括 JDBC 和 ODBC 接口。
- WEBGUI :Hive 客户端提供了一种通过网页的方式访问 Hive 所提供的服务。这个接口对应 Hive 的 HWI 组件 (Hive Web Interface)，使用前要启动 HWI 服务。

Hive 的执行流程

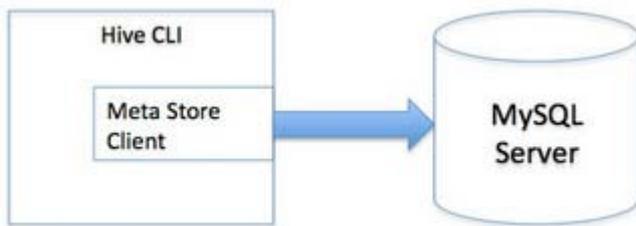


三种部署模式

1. **单用户模式** 此模式连接到一个 In-Memory 的数据库 Derby，一般用于 Unit Test。



2. **多用户模式** 通过网络连接到一个数据库中，是最经常使用到的模式。



3. **远程服务器模式** 用于非 Java 客户端访问元数据库，在服务器端启动 MetaStoreServer，客户端利用 Thrift 协议通过 MetaStoreServer 访问元数据库。



1.3 Hive 数据模型

Hive 没有专门的数据存储格式，用户可以自由的组织 Hive 中的表，只需要在创建表的时候告诉 Hive 数据中的列分隔符和行分隔符，Hive 就可以解析数据。Hive 中所有的数据都存储在 HDFS 中，存储结构主要包括数据库、文件、表和视图。Hive 中包含以下数据模型：Table 内部表，External Table 外部表，Partition 分区，Bucket 桶。Hive 默认可以直接加载文本文件，还支持 sequence file、RCFile。

1. Hive 数据库

类似传统数据库的 DataBase，在第三方数据库里实际是一张表
简单示例命令行：*create database test_database;*

2. 内部表

Hive 的内部表与数据库中的 Table 在概念上是类似。每一个 Table 在 Hive 中都有一个相应的目录存储数据。例如一个表 tbInner，它在 HDFS 中的路径为/user/hive/warehouse/tbInner，其中/user/hive/warehouse 是在 hive-site.xml 中由\${hive.metastore.warehouse.dir} 指定的数据仓库的目录，所有的 Table 数据（不包括 External Table）都保存在这个目录中。内部表删除时，元数据与数据都会被删除。

内部表简单示例：

创建数据文件：*test_inner_table.txt*

创建表：*create table test_inner_table (key string);*

加载数据：*LOAD DATA LOCAL INPATH 'filepath' INTO TABLE test_inner_table;*

查看数据：*select * from test_inner_table;*

删除表：*drop table test_inner_table;*

3. 外部表

外部表指向已经在 HDFS 中存在的数据库，并可以创建 Partition。它和内部表在元数据的组织上是相同的，而实际数据的存储则有较大的差异。内部表的创建过程和数据加载过程这两个过程可以分别独立完成，也可以在同一语句中完成，在加载数据的过程中，实际数据会被移动到数据仓库目录中；之后对数据的访问将会直接在数据仓库目录中完成。删除表时，表中的数据和元数据将会被同时删除。而外部表只有一个过程，加载数据和创建表同时完成 (CREATE EXTERNAL TABLELOCATION)，实际数据是存储在 LOCATION 后面指定的 HDFS 路径中，并不会移动到数据仓库目录中。当删除一个 External Table 时，仅删除该链接。

外部表简单示例：

创建数据文件：*test_external_table.txt*

创建表：*create external table test_external_table (key string);*

加载数据：*LOAD DATA INPATH 'filepath' INTO TABLE test_inner_table;*

查看数据：*select * from test_external_table;*

删除表：*drop table test_external_table;*

4. 分区

Partition 对应于数据库中的 Partition 列的密集索引，但是 Hive 中 Partition 的组织方式和数据库中的很不相同。在 Hive 中，表中的一个 Partition 对应于表下的一个目录，所有的 Partition 的数据都存储在对应的目录中。例如 pvs 表中包含 ds 和 city 两个 Partition，则对应于 ds = 20090801, ctry = US 的 HDFS 子目录为 /user/hive/warehouse/pvs/ds=20090801/ctry=US；对应于 ds = 20090801, ctry = CA 的 HDFS 子目录为 /user/hive/warehouse/pvs/ds=20090801/ctry=CA。

分区表简单示例：

创建数据文件：*test_partition_table.txt*

创建表：*create table test_partition_table (key string) partitioned by (dt string);*

加载数据：*LOAD DATA INPATH 'filepath' INTO TABLE test_partition_table partition (dt= '2006');*

查看数据：*select * from test_partition_table;*

删除表：*drop table test_partition_table;*

5. 桶

Buckets 是将表的列通过 Hash 算法进一步分解成不同的文件存储。它对指定列计算 Hash，

根据 Hash 值切分数据，目的是为了并行，每一个 Bucket 对应一个文件。例如将 user 列分散至 32 个 bucket，首先对 user 列的值计算 Hash，对应 Hash 值为 0 的 HDFS 目录为 /user/hive/warehouse/pvs/ds=20090801/ctry=US/part-00000；Hash 值为 20 的 HDFS 目录为/user/hive/warehouse/pvs/ds=20090801/ctry=US/part-00020。如果想应用很多的 Map 任务这样是不错的选择。

桶的简单示例：

创建数据文件：*test_bucket_table.txt*

创建表：*create table test_bucket_table (key string) clustered by (key) into 20 buckets;*

加载数据：*LOAD DATA INPATH 'filepath' INTO TABLE test_bucket_table;*

查看数据：*select * from test_bucket_table; set hive.enforce.bucketing = true;*

6. Hive 的视图

视图与传统数据库的视图类似。视图是只读的，它基于的基本表，如果改变，数据增加不会影响视图的呈现；如果删除，会出现问题。如果不指定视图的列，会根据 select 语句后的生成。

示例：*create view test_view as select * from test;*

1.4 Hive 数据类型

Hive 支持两种数据类型，一类叫原子数据类型，一类叫复杂数据类型。

- 原子数据类型包括数值型、布尔型和字符串类型，具体如下表所示：

基本数据类型		
类型	描述	示例
TINYINT	1 个字节 (8 位) 有符号整数	1
SMALLINT	2 字节 (16 位) 有符号整数	1
INT	4 字节 (32 位) 有符号整数	1
BIGINT	8 字节 (64 位) 有符号整数	1
FLOAT	4 字节 (32 位) 单精度浮点数	1.0
DOUBLE	8 字节 (64 位) 双精度浮点数	1.0
BOOLEAN	true/false	true
STRING	字符串	'xia' , " xia"

由上表我们看到 Hive 不支持日期类型，在 Hive 里日期都是用字符串来表示的，而常用的

日期格式转化操作则是通过自定义函数进行操作。

Hive 是用 Java 开发的，Hive 里的基本数据类型和 java 的基本数据类型也是一一对应的，除了 String 类型。有符号的整数类型：TINYINT、SMALLINT、INT 和 BIGINT 分别等价于 Java 的 Byte、Short、Int 和 Long 原子类型，它们分别为 1 字节、2 字节、4 字节和 8 字节有符号整数。Hive 的浮点数据类型 FLOAT 和 DOUBLE 对应于 Java 的基本类型 Float 和 Double 类型。而 Hive 的 BOOLEAN 类型相当于 Java 的基本数据类型 Boolean。对于 Hive 的 String 类型相当于数据库的 Varchar 类型，该类型是一个可变的字符串，不过它不能声明其中最多能存储多少个字符，理论上它可以存储 2GB 的字符数。

- 复杂数据类型包括数组 (ARRAY)、映射 (MAP) 和结构体 (STRUCT)，具体如下所示：

类型	解释	举例
STRUCT	与C/C++中的结构体类似，可通过“.”访问每个域的值，比如STRUCT {first STRING; last STRING}，可通过name.first访问第一个成员。	struct('John', 'Doe')
MAP	存储key/value对，可通过['key']获取每个key的值，比如'first'→'John' and 'last'→'Doe'，可通过name['last']获取last name。	map('first', 'John', 'last', 'Doe')
ARRAY	同种类型的数据集合，从0开始索引，比如['John', 'Doe']，可通过name[1]获取“Doe”。	array('John', 'Doe')

1.5 Hive 与关系数据库的区别

由于 Hive 采用了 SQL 的查询语言 HQL，因此很容易将 Hive 理解为数据库。其实从结构上来看，Hive 和数据库除了拥有类似的查询语言，再无类似之处。数据库可以用在 Online 的应用中，但是 Hive 是为数据仓库而设计的，清楚这一点，有助于从应用角度理解 Hive 的特性。

Hive 和数据库的比较如下表：

	Hive	RDBMS
查询语言	HQL	SQL
数据存储	HDFS	Raw Device or Local FS
数据格式	用户定义	系统决定
数据更新	不支持	支持
索引	无	有

执行	MapReduce	Executor
执行延迟	高	低
处理数据规模	大	小
可扩展性	高	低

2 Hive 搭建过程

2.1 安装 MySql 数据库

2.1.1 下载 MySql 安装文件

下载地址：<http://dev.mysql.com/downloads/mysql/#downloads>，使用系统为CentOS选择 Red Hat Enterprise Linux/Oracle系列：



操作系统为 64 位，选择对应安装包进行下载：

Red Hat Enterprise Linux 6 / Oracle Linux 6 (x86, 64-bit), RPM Package Development Libraries (MySQL-devel-5.6.21-1.el6.x86_64.rpm)	5.6.21	3.2M	Download
Red Hat Enterprise Linux 6 / Oracle Linux 6 (x86, 64-bit), RPM Package Client Utilities (MySQL-client-5.6.21-1.el6.x86_64.rpm)	5.6.21	17.7M	Download
Red Hat Enterprise Linux 6 / Oracle Linux 6 (x86, 64-bit), RPM Package MySQL Server (MySQL-server-5.6.21-1.el6.x86_64.rpm)	5.6.21	53.1M	Download

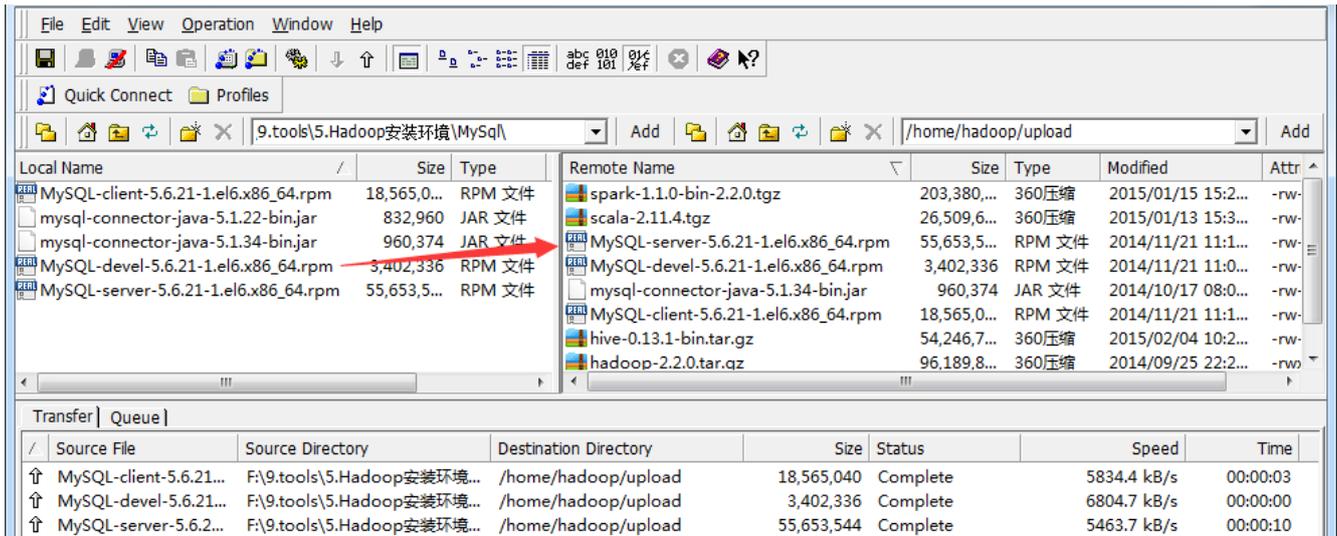
下载在本地目录如下图：

MySQL-client-5.6.21-1.el6.x86_64.rpm	2014/11/21 11:11	360压缩	18,130 KB
MySQL-devel-5.6.21-1.el6.x86_64.rpm	2014/11/21 11:04	360压缩	3,323 KB
MySQL-server-5.6.21-1.el6.x86_64.rpm	2014/11/21 11:14	360压缩	54,350 KB

2.1.2 上传 MySql 安装文件

把下载的 mysql 安装包，使用 SSH Secure File Transfer 工具（参见《Spark 编译与部署（上）》

--基础环境搭建》1.3.1 介绍) 上传到/home/hadoop/upload 目录下, 如下图所示:



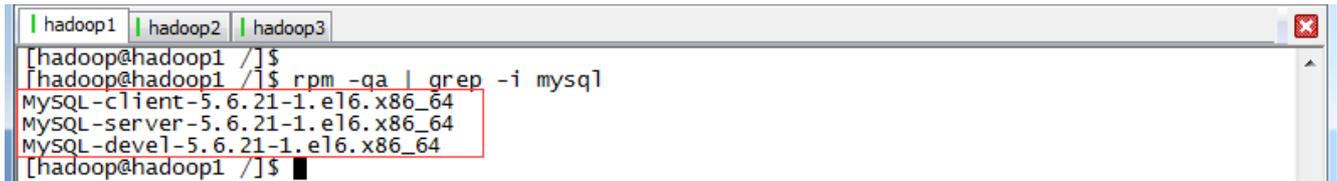
2.1.3 卸载旧的 MySQL

(1) 查找以前是否安装有 mysql

使用命令查看是否已经安装过 mysql:

```
$rpm -qa | grep -i mysql
```

可以看到如下图所示:



说明之前安装了:

```
MySQL-client-5.6.21-1.el6.x86_64
```

```
MySQL-server-5.6.21-1.el6.x86_64
```

```
MySQL-devel-5.6.21-1.el6.x86_64
```

如果没有结果, 可以进行 mysql 数据库安装

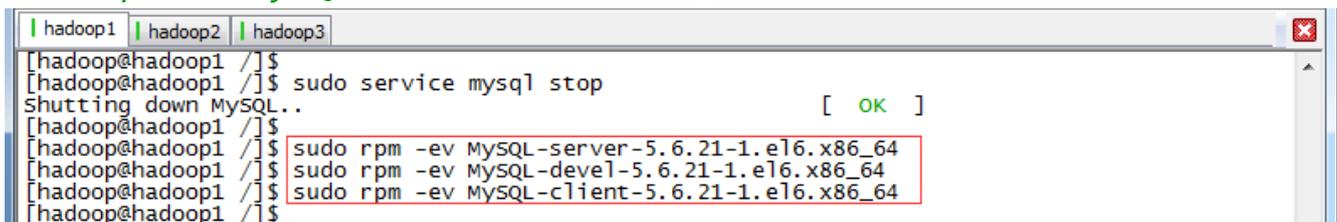
(2) 停止 mysql 服务、删除之前安装的 mysql

停止 mysql 服务、删除之前安装的 mysql 删除命令: rpm -e --nodeps 包名

```
$sudo rpm -ev MySQL-server-5.6.21-1.el6.x86_64
```

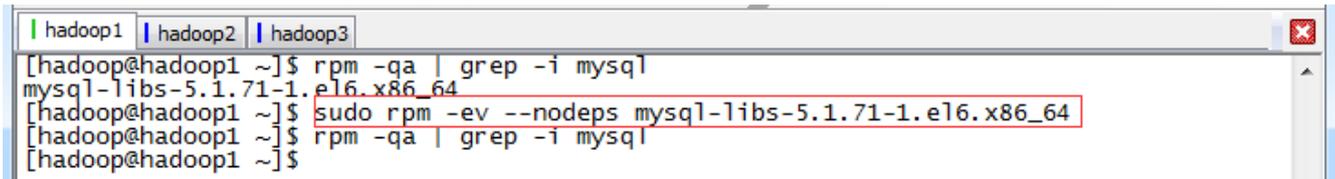
```
$sudo rpm -ev MySQL-devel-5.6.21-1.el6.x86_64
```

```
$sudo rpm -ev MySQL-client-5.6.21-1.el6.x86_64
```



如果存在 CentOS 自带 mysql-libs-5.1.71-1.el6.x86_64 使用下面的命令卸载即可

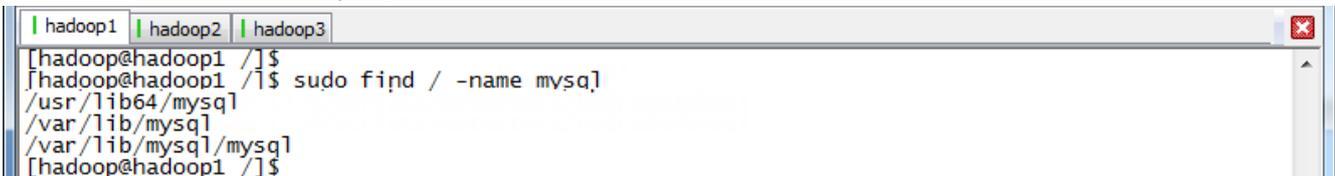
```
$sudo rpm -ev --nodeps mysql-libs-5.1.71-1.el6.x86_64
```



```
[hadoop@hadoop1 ~]$ rpm -qa | grep -i mysql
mysql-libs-5.1.71-1.el6.x86_64
[hadoop@hadoop1 ~]$ sudo rpm -ev --nodeps mysql-libs-5.1.71-1.el6.x86_64
[hadoop@hadoop1 ~]$ rpm -qa | grep -i mysql
[hadoop@hadoop1 ~]$
```

(3) 查找之前老版本 mysql 的目录并且删除老版本 mysql 的文件和库

```
$sudo find / -name mysql
```

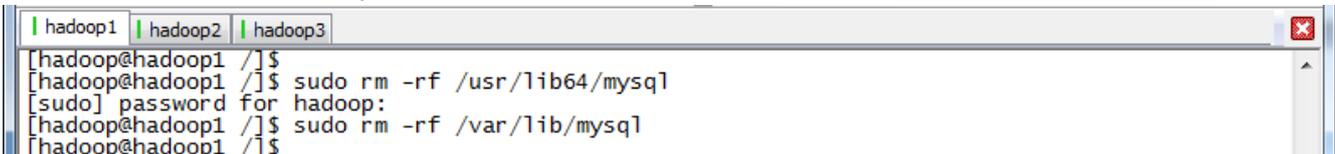


```
[hadoop@hadoop1 ~]$ sudo find / -name mysql
/usr/lib64/mysql
/var/lib/mysql
/var/lib/mysql/mysql
[hadoop@hadoop1 ~]$
```

删除对应的 mysql 目录

```
$sudo rm -rf /usr/lib64/mysql
```

```
$sudo rm -rf /var/lib/mysql
```

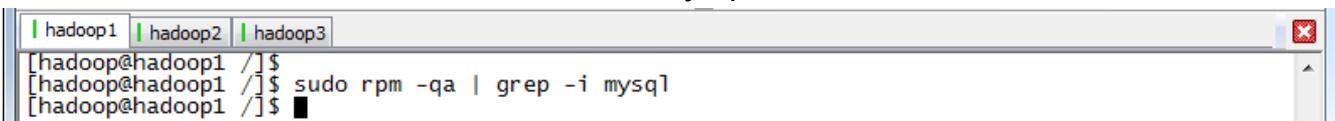


```
[hadoop@hadoop1 ~]$ sudo rm -rf /usr/lib64/mysql
[sudo] password for hadoop:
[hadoop@hadoop1 ~]$ sudo rm -rf /var/lib/mysql
[hadoop@hadoop1 ~]$
```

(4) 再次查找机器是否安装 mysql

```
$sudo rpm -qa | grep -i mysql
```

无结果，说明已经卸载彻底、接下来直接安装 mysql 即可



```
[hadoop@hadoop1 ~]$ sudo rpm -qa | grep -i mysql
```

2.1.4 安装 MySQL

进入安装文件的目录，安装 mysql 服务端

```
$cd /home/hadoop/upload
```

```
$sudo rpm -ivh MySQL-server-5.6.21-1.el6.x86_64.rpm
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /home/hadoop/upload
[hadoop@hadoop1 upload]$ ls
hadoop-2.2.0.tar.gz
hive-0.13.1-bin.tar.gz
MySQL-client-5.6.21-1.el6.x86_64.rpm
mysql-connector-java-5.1.34-bin.jar
MySQL-devel-5.6.21-1.el6.x86_64.rpm
MySQL-server-5.6.21-1.el6.x86_64.rpm
scala-2.11.4.tgz
spark-1.1.0-bin-2.2.0.tgz
[hadoop@hadoop1 upload]$ sudo rpm -ivh MySQL-server-5.6.21-1.el6.x86_64.rpm
Preparing... ##### [100%]
 1:MySQL-server ##### [100%]
warning: user mysql does not exist - using root
warning: group mysql does not exist - using root
2015-02-04 11:10:07 0 [warning] TIMESTAMP with implicit DEFAULT value is deprecated.
Please use --explicit_defaults_for_timestamp server option (see documentation for mor
e details).
2015-02-04 11:10:07 14718 [Note] InnoDB: Using atomics to ref count buffer pool pages
2015-02-04 11:10:07 14718 [Note] InnoDB: The InnoDB memory heap is disabled
2015-02-04 11:10:07 14718 [Note] InnoDB: Mutexes and rw_locks use GCC atomic builtins
2015-02-04 11:10:07 14718 [Note] InnoDB: Memory barrier is not used
2015-02-04 11:10:07 14718 [Note] InnoDB: Compressed tables use zlib 1.2.3
2015-02-04 11:10:07 14718 [Note] InnoDB: Using Linux native AIO
2015-02-04 11:10:07 14718 [Note] InnoDB: Using CPU crc32 instructions
2015-02-04 11:10:07 14718 [Note] InnoDB: Initializing buffer pool, size = 128.0M
2015-02-04 11:10:08 14718 [Note] InnoDB: Completed initialization of buffer pool
2015-02-04 11:10:08 14718 [Note] InnoDB: The first specified data file ./ibdata1 did
not exist: a new database to be created!
```

安装 mysql 客户端、mysql-devel

```
$sudo rpm -ivh MySQL-client-5.6.21-1.el6.x86_64.rpm
```

```
$sudo rpm -ivh MySQL-devel-5.6.21-1.el6.x86_64.rpm
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 upload]$ sudo rpm -ivh MySQL-client-5.6.21-1.el6.x86_64.rpm
Preparing... ##### [100%]
 1:MySQL-client ##### [100%]
[hadoop@hadoop1 upload]$ sudo rpm -ivh MySQL-devel-5.6.21-1.el6.x86_64.rpm
Preparing... ##### [100%]
 1:MySQL-devel ##### [100%]
[hadoop@hadoop1 upload]$
```

2.1.5 设置 root 密码

在 CentOS6.5 下安装 mysql 设置 root 密码时，出现如下错误：

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 /]$
[hadoop@hadoop1 /]$ /usr/bin/mysqladmin -u root password 'root';
/usr/bin/mysqladmin: connect to server at 'localhost' failed
error: 'Access denied for user 'root'@'localhost' (using password: NO)'
[hadoop@hadoop1 /]$
```

/usr/bin/mysqladmin: connect to server at 'localhost' failed

error: 'Access denied for user 'root'@'localhost' (using password: NO)'

可以进入安全模式进行设置 root 密码

(1) 停止 mysql 服务

使用如下命令停止 mysql 服务：

```
$sudo service mysql stop
```

```
$sudo service mysql status
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ sudo service mysql stop
[sudo] password for hadoop:
Shutting down MySQL.. [ OK ]
[hadoop@hadoop1 ~]$ sudo service mysql status
MySQL is not running [FAILED]
[hadoop@hadoop1 ~]$
```

(2) 跳过验证启动 mysql

使用如下命令验证启动 mysql, 由于 & 结尾是后台运行进程, 运行该命令可以再打开命令窗口或者 Ctrl+C 继续进行下步操作, 由于 mysql 启动时间会长点, 需要等待几分钟再查看启动状态:

```
$sudo mysqld_safe --skip-grant-tables &
$sudo service mysql status
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ sudo mysqld_safe --skip-grant-tables &
[1] 4817
[hadoop@hadoop1 ~]$ 141124 15:49:58 mysqld_safe Logging to '/var/lib/mysql/hadoop1.err'.
141124 15:49:58 mysqld_safe Starting mysqld daemon with databases from /var/lib/mysql
^C
[hadoop@hadoop1 ~]$ sudo service mysql status
MySQL running (4909) [ OK ]
[hadoop@hadoop1 ~]$
```

(3) 跳过验证启动 MySQL

验证 mysql 服务已经在后台运行后, 执行如下语句, 其中后面三条命令是在 mysql 语句:

```
mysql -u root
mysql>use mysql;
mysql>update user set password = password('root') where user = 'root';
mysql>flush privileges;
mysql>quit;
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ mysql -u root
welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.6.21 MySQL Community Server (GPL)

copyright (c) 2000, 2014, oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

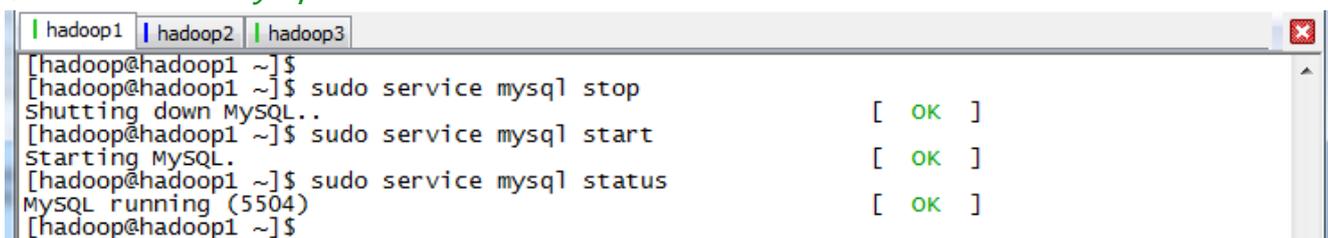
Database changed
mysql> update user set password = password('root') where user = 'root';
Query OK, 4 rows affected (0.00 sec)
Rows matched: 4 Changed: 4 Warnings: 0

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

(4) 跳过验证启动 MySQL

重启 mysql 服务并查看状态

```
$sudo service mysql stop  
$sudo service mysql start  
$sudo service mysql status
```

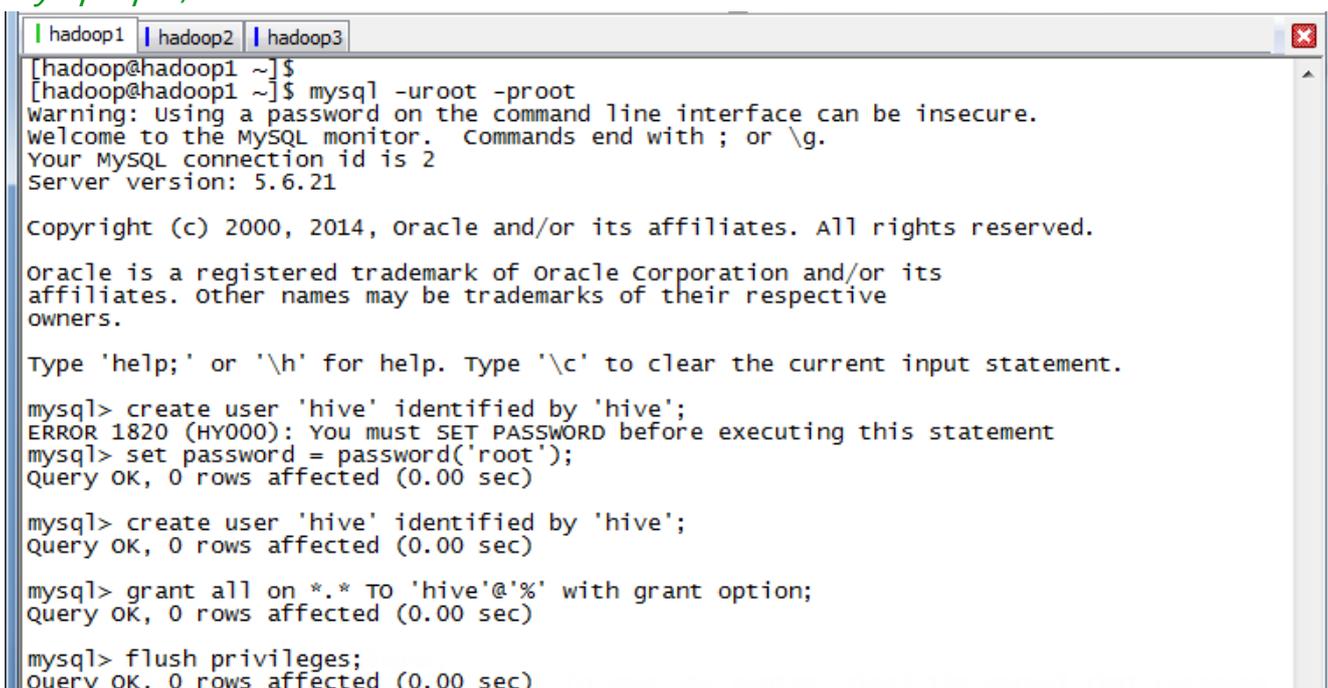


```
hadoop1 | hadoop2 | hadoop3  
[hadoop@hadoop1 ~]$  
[hadoop@hadoop1 ~]$ sudo service mysql stop  
Shutting down MySQL.. [ OK ]  
[hadoop@hadoop1 ~]$ sudo service mysql start  
Starting MySQL. [ OK ]  
[hadoop@hadoop1 ~]$ sudo service mysql status  
MySQL running (5504) [ OK ]  
[hadoop@hadoop1 ~]$
```

2.1.6 设置 Hive 用户

进入 mysql 命令行，创建 Hive 用户并赋予所有权限：

```
mysql -uroot -proot  
mysql>set password=password('root');  
mysql>create user 'hive' identified by 'hive';  
mysql>grant all on *.* TO 'hive'@'%' with grant option;  
mysql>flush privileges;  
mysql>quit;
```



```
hadoop1 | hadoop2 | hadoop3  
[hadoop@hadoop1 ~]$  
[hadoop@hadoop1 ~]$ mysql -uroot -proot  
warning: using a password on the command line interface can be insecure.  
welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 2  
Server version: 5.6.21  
  
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
mysql> create user 'hive' identified by 'hive';  
ERROR 1820 (HY000): You must SET PASSWORD before executing this statement  
mysql> set password = password('root');  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> create user 'hive' identified by 'hive';  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> grant all on *.* TO 'hive'@'%' with grant option;  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> flush privileges;  
Query OK, 0 rows affected (0.00 sec)
```

(注意：如果是 root 第一次登录数据库，需要重新设置一下密码，所报异常信息如下：**ERROR 1820 (HY000): You must SET PASSWORD before executing this statement**)

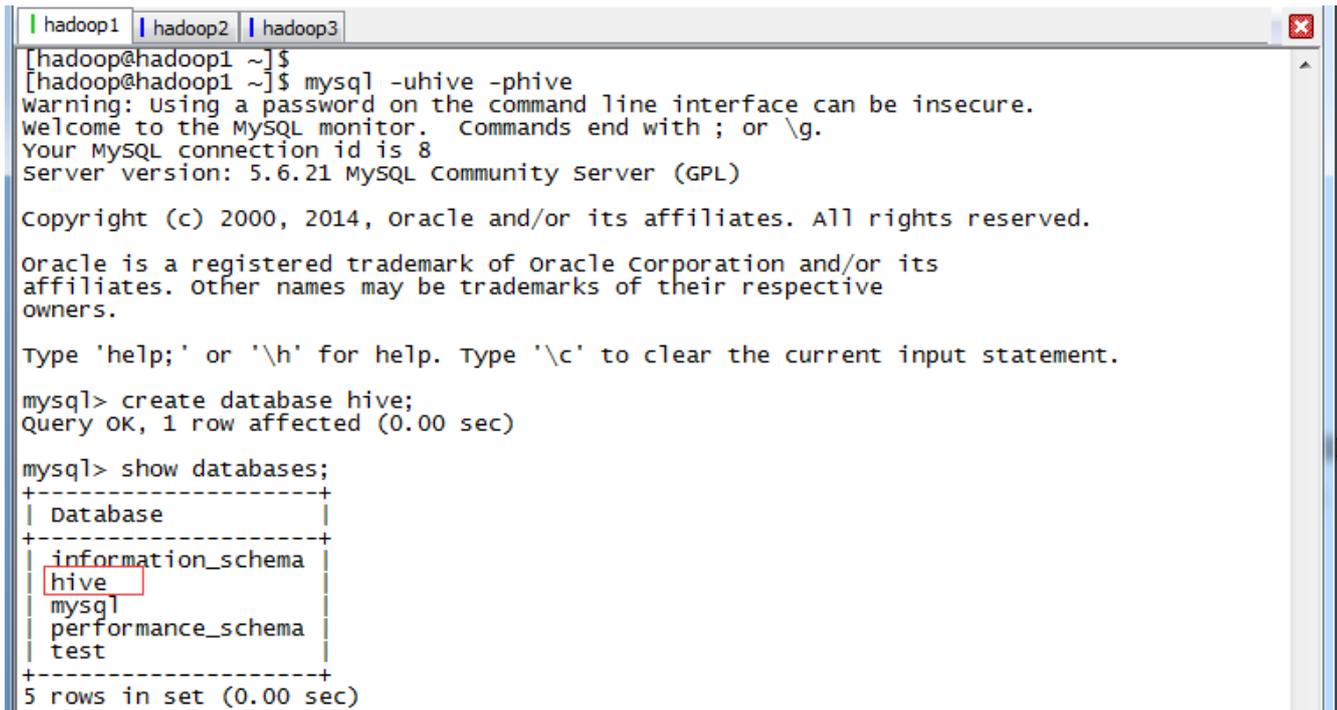
2.1.7 创建 Hive 数据库

使用 hive 用户登录，创建 Hive 数据库：

```
mysql -uhive -phive
```

```
mysql> create database hive;
```

```
mysql> show databases;
```

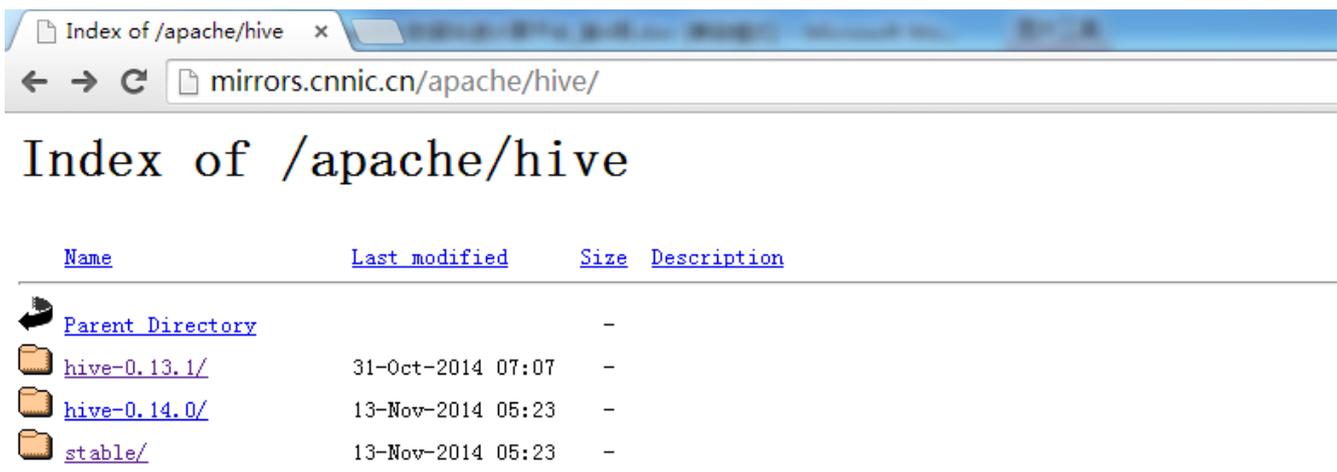


```
[hadoop@hadoop1 ~]$  
[hadoop@hadoop1 ~]$ mysql -uhive -phive  
Warning: Using a password on the command line interface can be insecure.  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 8  
Server version: 5.6.21 MySQL Community Server (GPL)  
  
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql> create database hive;  
Query OK, 1 row affected (0.00 sec)  
  
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| hive |  
| mysql |  
| performance_schema |  
| test |  
+-----+  
5 rows in set (0.00 sec)
```

2.2 安装 Hive

2.2.1 下载 Hive 安装文件

可以到 Apache 基金 hive 官网 <http://hive.apache.org/downloads.html>，选择镜像下载地址：<http://mirrors.cnnic.cn/apache/hive/> 下载一个稳定版本，这里选择下载 `apache-hive-0.13.1-bin.tar.gz` 文件，如下图所示：



Apache/2.0.64 (Unix) Server at mirrors.cnnic.cn Port 80

2.2.2 下载 MySQL 驱动

到 mysql 官网进入下载页面：<http://dev.mysql.com/downloads/connector/j/>，默认情

况下是 Windows 安装包，这里需要选择 Platform Independent 版本下载 zip 格式的文件

Generally Available (GA) Releases

Connector/J 5.1.34

Select Platform: Platform Independent

Looking for previous GA versions?

Platform Independent (Architecture Independent), Compressed TAR Archive	5.1.34	3.6M	Download
(mysql-connector-java-5.1.34.tar.gz)			
MD5: ad4c875c719247f8e8c41cd7f0609e00 Signature			
Platform Independent (Architecture Independent), ZIP Archive	5.1.34	3.9M	Download
(mysql-connector-java-5.1.34.zip)			
MD5: 51a49bfdeb963d1be3ac7bd41ac7908f Signature			

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

2.2.3 上传 Hive 安装文件和 MySQL 驱动

把下载的 hive 安装包和 mysql 驱动包，使用 SSH Secure File Transfer 工具（参见《Spark 编译与部署（上）--基础环境搭建》1.3.1 介绍）上传到 /home/hadoop/upload 目录下，如下图所示：

Local Name	Size	Type	Mod	Remote Name	Size	Type	Modified
hive-0.12.0-bin.tar.gz	65,662,4...	360压缩	201	hadoop-2.2.0.tar.gz	96,189,8...	360压缩	2014/09/25 22:2...
hive-0.13.0-bin.tar.gz	54,107,8...	360压缩	201	hive-0.13.1-bin.tar.gz	54,246,7...	360压缩	2015/02/04 10:2...
hive-0.13.1-bin.tar.gz	54,246,7...	360压缩	201	mysql-connector-java-5...	960,374	JAR 文件	2014/10/17 08:0...
hive-0.14.0-bin.tar.gz	80,288,7...	360压缩	201	scala-2.11.4.tgz	26,509,6...	360压缩	2015/01/13 15:3...
httpd-2.2.25-win32-x86-no_ssl.msi	5,754,604	Window...	201	spark-1.1.0-bin-2.2.0.tgz	203,380,...	360压缩	2015/01/15 15:2...
jdk-7u9-linux-i586.tar.gz	97,361,2...	360压缩	201				
mahout-distribution-0.6.tar.gz	60,072,2...	360压缩	201				
mahout-distribution-0.9.tar.gz	69,248,3...	360压缩	201				

Source File	Source Directory	Destination Directory	Size	Status	Speed	Time
hive-0.13.1-bin.ta...	F:\9.tools\5.Hadoop安装...	/home/hadoop/upload	54,246,778	Complete	8163.5 kB/s	00:00:06
mysql-connector-...	F:\9.tools\5.Hadoop安装...	/home/hadoop/upload	960,374	Complete	7683.0 kB/s	00:00:00

2.2.4 解压缩

到上传目录下，用如下命令解压缩 hive 安装文件：

```
cd /home/hadoop/upload  
tar -zxf hive-0.13.1-bin.tar.gz
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /home/hadoop/upload
[hadoop@hadoop1 upload]$ ls
hadoop-2.2.0.tar.gz          MySQL-devel-5.6.21-1.el6.x86_64.rpm
hive-0.13.1-bin.tar.gz      MySQL-server-5.6.21-1.el6.x86_64.rpm
MySQL-client-5.6.21-1.el6.x86_64.rpm  scala-2.11.4.tgz
mysql-connector-java-5.1.34-bin.jar  spark-1.1.0-bin-2.2.0.tgz
[hadoop@hadoop1 upload]$ tar -zxf hive-0.13.1-bin.tar.gz
[hadoop@hadoop1 upload]$ ls
apache-hive-0.13.1-bin      MySQL-devel-5.6.21-1.el6.x86_64.rpm
hadoop-2.2.0.tar.gz         MySQL-server-5.6.21-1.el6.x86_64.rpm
hive-0.13.1-bin.tar.gz     scala-2.11.4.tgz
MySQL-client-5.6.21-1.el6.x86_64.rpm  spark-1.1.0-bin-2.2.0.tgz
mysql-connector-java-5.1.34-bin.jar
[hadoop@hadoop1 upload]$
```

改名并迁移到/app/hadoop 目录下：

```
sudo mv apache-hive-0.13.1-bin /app/hadoop/hive-0.13.1
```

```
ll /app/hadoop
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 upload]$ sudo mv apache-hive-0.13.1-bin /app/hadoop/hive-0.13.1
[hadoop@hadoop1 upload]$ ll /app/hadoop
total 12
drwxr-xr-x 13 hadoop hadoop 4096 Jan 15 00:34 hadoop-2.2.0
drwxrwxr-x  8 hadoop hadoop 4096 Feb  4 12:02 hive-0.13.1
drwxrwxr-x 11 hadoop hadoop 4096 Jan 15 16:07 spark-1.1.0
[hadoop@hadoop1 upload]$
```

2.2.5 把 MySQL 驱动放到 Hive 的 lib 目录下

把下载的 hive 安装包和 mysql 驱动包，使用

```
cd /home/hadoop/upload
```

```
cp mysql-connector-java-5.1.34-bin.jar /app/hadoop/hive-0.13.1/lib
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /home/hadoop/upload
[hadoop@hadoop1 upload]$ ls
hadoop-2.2.0.tar.gz          MySQL-devel-5.6.21-1.el6.x86_64.rpm
hive-0.13.1-bin.tar.gz      MySQL-server-5.6.21-1.el6.x86_64.rpm
MySQL-client-5.6.21-1.el6.x86_64.rpm  scala-2.11.4.tgz
mysql-connector-java-5.1.34-bin.jar  spark-1.1.0-bin-2.2.0.tgz
[hadoop@hadoop1 upload]$ cp mysql-connector-java-5.1.34-bin.jar /app/hadoop/hive-0.13.1/lib
[hadoop@hadoop1 upload]$ ll /app/hadoop/hive-0.13.1/lib/mysql*
-rw-r--r-- 1 hadoop hadoop 960374 Feb  4 12:08 /app/hadoop/hive-0.13.1/lib/mysql-connector-java-5.1.34-bin.jar
[hadoop@hadoop1 upload]$
```

2.2.6 配置/etc/profile 环境变量

使用如下命令打开/etc/profile 文件：

```
sudo vi /etc/profile
```

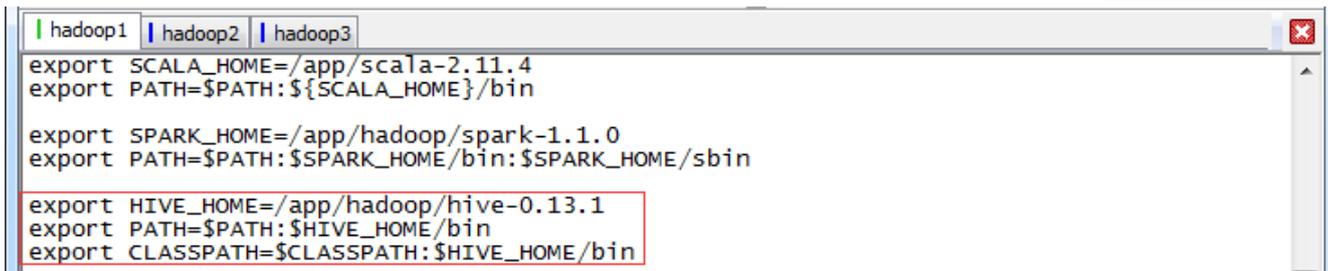
```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ sudo vi /etc/profile
```

设置如下参数：

```
export HIVE_HOME=/app/hadoop/hive-0.13.1
```

```
export PATH=$PATH:$HIVE_HOME/bin
```

```
export CLASSPATH=$CLASSPATH:$HIVE_HOME/bin
```



```
hadoop1 | hadoop2 | hadoop3
export SCALA_HOME=/app/scala-2.11.4
export PATH=$PATH:${SCALA_HOME}/bin

export SPARK_HOME=/app/hadoop/spark-1.1.0
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin

export HIVE_HOME=/app/hadoop/hive-0.13.1
export PATH=$PATH:$HIVE_HOME/bin
export CLASSPATH=$CLASSPATH:$HIVE_HOME/bin
```

使配置文件生效：

```
source /etc/profile
```

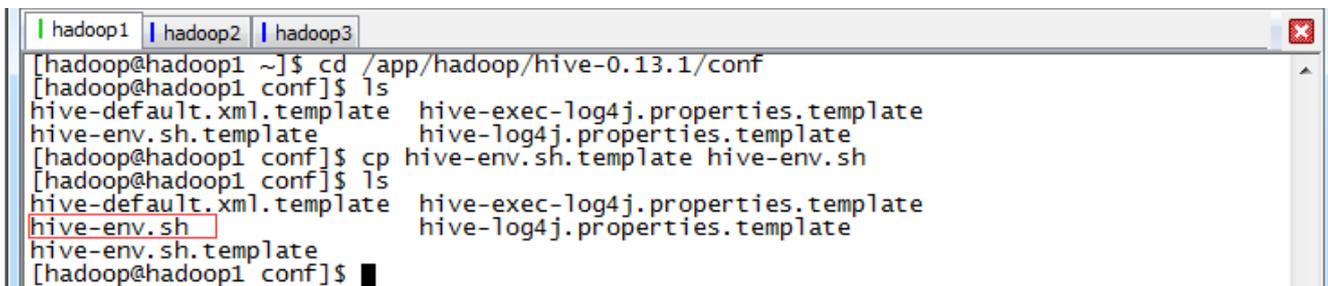


```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ source /etc/profile
[hadoop@hadoop1 ~]$
```

2.2.7 设置 hive-env.sh 配置文件

进入 hive-0.13.1/conf 目录，复制 hive-env.sh.template 为 hive-env.sh：

```
cd /app/hadoop/hive-0.13.1/conf
cp hive-env.sh.template hive-env.sh
ls
```



```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop/hive-0.13.1/conf
[hadoop@hadoop1 conf]$ ls
hive-default.xml.template  hive-exec-log4j.properties.template
hive-env.sh.template      hive-log4j.properties.template
[hadoop@hadoop1 conf]$ cp hive-env.sh.template hive-env.sh
[hadoop@hadoop1 conf]$ ls
hive-default.xml.template  hive-exec-log4j.properties.template
hive-env.sh               hive-log4j.properties.template
hive-env.sh.template
[hadoop@hadoop1 conf]$
```

使用如下命令编辑配置文件

```
sudo vi hive-env.sh
```

分别设置 HADOOP_HOME 和 HIVE_CONF_DIR 两个值：

```
# Set HADOOP_HOME to point to a specific hadoop install directory
export HADOOP_HOME=/app/hadoop/hadoop-2.2.0
```

```
# Hive Configuration Directory can be controlled by:
export HIVE_CONF_DIR=/app/hadoop/hive-0.13.1/conf
```

```
hadoop1 | hadoop2 | hadoop3
# Set HADOOP_HOME to point to a specific hadoop install directory
export HADOOP_HOME=/app/hadoop/hadoop-2.2.0
# Hive Configuration Directory can be controlled by:
export HIVE_CONF_DIR=/app/hadoop/hive-0.13.1/conf
# Folder containing extra libraries required for hive compilation/execution can be controlled by:
# export HIVE_AUX_JARS_PATH=
```

2.2.8 设置 hive-site.xml 配置文件

复制 hive-default.xml.template 为 hive-site.xml

cp hive-default.xml.template hive-site.xml

sudo vi hive-site.xml

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 conf]$ ls
hive-default.xml.template  hive-env.sh.template  hive-log4j.properties.template
hive-env.sh               hive-exec-log4j.properties.template
[hadoop@hadoop1 conf]$ cp hive-default.xml.template hive-site.xml
[hadoop@hadoop1 conf]$ ls
hive-default.xml.template  hive-env.sh.template  hive-log4j.properties.template
hive-env.sh               hive-exec-log4j.properties.template  hive-site.xml
[hadoop@hadoop1 conf]$ sudo vi hive-site.xml
```

(1) 加入配置项

默认 metastore 在本地，添加配置改为非本地

```
<property>
  <name>hive.metastore.local</name>
  <value>>false</value>
</property>
```

```
hadoop1 | hadoop2 | hadoop3
<property>
  <name>hive.metastore.local</name>
  <value>>false</value>
</property>
<property>
  <name>hive.exec.reducers.bytes.per.reducer</name>
  <value>1000000000</value>
  <description>size per reducer.The default is 1G, i.e if the input size is 10G, it will use 10 reducers.</description>
</property>
```

(2) 修改配置项

hive 默认为 derby 数据库，derby 数据只运行单个用户进行连接，这里需要调整为 mysql 数据库

```
<property>
  <name>hive.metastore.uris</name>
  <value>thrift://hadoop1:9083</value>
  <description>Thrift URI for the remote metastore. ...</description>
```

```

</property>
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://hadoop1:3306/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>

```

```

hadoop1 | hadoop2 | hadoop3
<property>
  <name>hive.metastore.uris</name>
  <value>thrift://hadoop1:9083</value>
  <description>Thrift uri for the remote metastore. Used by metastore client to connect to remote metastore.</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://hadoop1:3306/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>

```

把 hive.metastore.schema.verification 配置项值修改为 false

```
<property>
  <name>hive.metastore.schema.verification</name>
  <value>>false</value>
  <desc....>
</property>
```

```
<property>
  <name>hive.metastore.schema.verification</name>
  <value>>false</value>
  <description>
    Enforce metastore schema version consistency.
    True: Verify that version information stored in metastore matches 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>
```

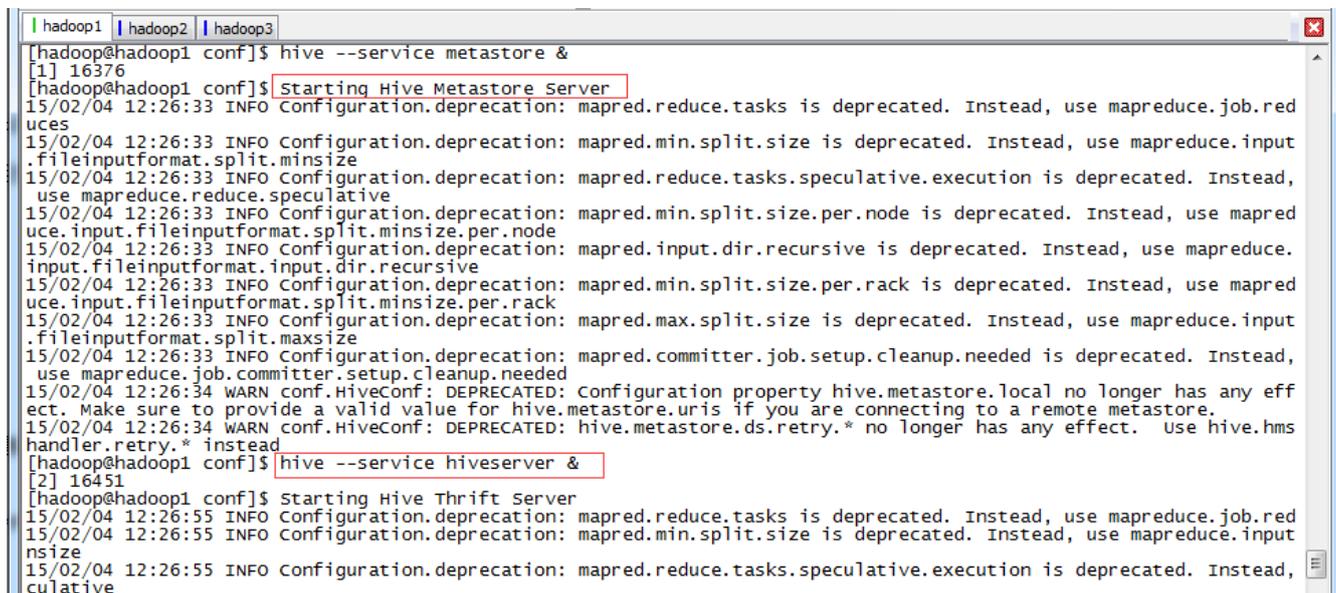
2.3 启动并验证 Hive

2.3.1 启动 Hive

实际使用时，一般通过后台启动 metastore 和 hiveserver 实现服务，命令如下：

```
hive --service metastore &
```

```
hive --service hiveserver &
```



```
[hadoop@hadoop1 conf]$ hive --service metastore &
[1] 16376
[hadoop@hadoop1 conf]$ Starting Hive Metastore Server
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.min.split.size is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.reduce.tasks.speculative.execution is deprecated. Instead, use mapreduce.reduce.speculative
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.min.split.size.per.node is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize.per.node
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.input.dir.recursive is deprecated. Instead, use mapreduce.input.fileinputformat.input.dir.recursive
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.min.split.size.per.rack is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize.per.rack
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.max.split.size is deprecated. Instead, use mapreduce.input.fileinputformat.split.maxsize
15/02/04 12:26:33 INFO Configuration.deprecation: mapred.committer.job.setup.cleanup.needed is deprecated. Instead, use mapreduce.job.committer.setup.cleanup.needed
15/02/04 12:26:34 WARN conf.HiveConf: DEPRECATED: Configuration property hive.metastore.local no longer has any effect. Make sure to provide a valid value for hive.metastore.uris if you are connecting to a remote metastore.
15/02/04 12:26:34 WARN conf.HiveConf: DEPRECATED: hive.metastore.ds.retry.* no longer has any effect. Use hive.hms.handler.retry.* instead
[hadoop@hadoop1 conf]$ hive --service hiveserver &
[2] 16451
[hadoop@hadoop1 conf]$ Starting Hive Thrift Server
15/02/04 12:26:55 INFO Configuration.deprecation: mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
15/02/04 12:26:55 INFO Configuration.deprecation: mapred.min.split.size is deprecated. Instead, use mapreduce.input.size
15/02/04 12:26:55 INFO Configuration.deprecation: mapred.reduce.tasks.speculative.execution is deprecated. Instead, use mapreduce.reduce.speculative
```

启动用通过 jps 命令可以看到两个进行运行在后台

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 conf]$ jps
16451 RunJar
13543 ResourceManager
13124 NameNode
13227 DataNode
16538 Jps
16376 RunJar
13649 NodeManager
13393 SecondaryNameNode
[hadoop@hadoop1 conf]$
```

2.3.2 在 Hive 中操作

登录 hive , 在 hive 创建表并查看该表 , 命令如下 :

```
hive
hive>create table test(a string, b int);
hive>show tables;
hive>desc test;
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ hive
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.min.split.size is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.reduce.tasks.speculative.execution is deprecated. Instead, use mapreduce.reduce.speculative
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.min.split.size.per.node is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize.p
er.node
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.input.dir.recursive is deprecated. Instead, use mapreduce.input.fileinputformat.input.dir.recursive
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.min.split.size.per.rack is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize.p
er.rack
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.max.split.size is deprecated. Instead, use mapreduce.input.fileinputformat.split.maxsize
15/02/04 12:30:31 INFO Configuration.deprecation: mapred.committer.job.setup.cleanup.needed is deprecated. Instead, use mapreduce.job.committer.setup.cleanu
p.needed
15/02/04 12:30:32 WARN conf.HiveConf: DEPRECATED: configuration property hive.metastore.local no longer has any effect. Make sure to provide a valid value f
or hive.metastore.uris if you are connecting to a remote metastore.
15/02/04 12:30:32 WARN conf.HiveConf: DEPRECATED: hive.metastore.ds.retry.* no longer has any effect. Use hive.hmsHandler.retry.* instead
Logging initialized using configuration in jar:file:/app/hadoop/hive-0.13.1/lib/hive-common-0.13.1.jar!/hive-log4j.properties
hive> create table test(a string, b int);
OK
Time taken: 4.284 seconds
hive> show tables;
OK
test
Time taken: 0.325 seconds, Fetched: 1 row(s)
hive> desc test;
OK
a          string
b          int
Time taken: 0.522 seconds, Fetched: 2 row(s)
hive>
```

登录 mysql , 在 TBLS 表中查看新增 test 表 :

```
mysql -uhive -phive
mysql>use hive;
mysql>select TBL_ID, CREATE_TIME, DB_ID, OWNER, TBL_NAME,TBL_TYPE from TBLS;
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ mysql -uhive -phive
warning: Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.6.21 MySQL Community server (GPL)

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use hive;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select TBL_ID, CREATE_TIME, DB_ID, OWNER, TBL_NAME, TBL_TYPE from TBLS;
+-----+-----+-----+-----+-----+-----+
| TBL_ID | CREATE_TIME | DB_ID | OWNER | TBL_NAME | TBL_TYPE |
+-----+-----+-----+-----+-----+-----+
|      1 | 1423024253 |     1 | hadoop | test     | MANAGED_TABLE |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

3 问题解决

3.1 设置 MySQL 数据库 root 用户密码报错

在 CentOS6.5 下安装 mysql 设置 root 密码时，出现如下错误：

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ sudo /usr/bin/mysqladmin -u root password 'root';
/usr/bin/mysqladmin: connect to server at 'localhost' failed
error: 'Access denied for user 'root'@'localhost' (using password: NO)'
[hadoop@hadoop1 ~]$
```

*/usr/bin/mysqladmin: connect to server at 'localhost' failed
error: 'Access denied for user 'root'@'localhost' (using password: NO)'*

(1) 停止 mysql 服务

使用如下命令停止 mysql 服务：

sudo service mysql stop

sudo service mysql status

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ sudo service mysql stop
[sudo] password for hadoop:
Shutting down MySQL..                                     [ OK ]
[hadoop@hadoop1 ~]$ sudo service mysql status
MySQL is not running                                     [ FAILED ]
[hadoop@hadoop1 ~]$
```

(2) 跳过验证启动 mysql

使用如下命令验证启动 mysql，由于 & 结尾是后台运行进程，运行该命令可以再打开命令窗口或者 Ctrl+C 继续进行下一步操作：

mysqld_safe --skip-grant-tables &

sudo service mysql status

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ sudo mysqld_safe --skip-grant-tables &
[1] 4817
[hadoop@hadoop1 ~]$ 141124 15:49:58 mysqld_safe Logging to '/var/lib/mysql/hadoop1.err'.
141124 15:49:58 mysqld_safe Starting mysqld daemon with databases from /var/lib/mysql
^C
[hadoop@hadoop1 ~]$ sudo service mysql status
MySQL running (4909) [ OK ]
[hadoop@hadoop1 ~]$
```

(3) 跳过验证启动 MySQL

验证 mysql 服务已经在后台运行后，执行如下语句，其中后面三条命令是在 mysql 语句：

mysql -u root

mysql>use mysql;

mysql>update user set password = password('root') where user = 'root';

mysql>flush privileges;

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ mysql -u root
welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.6.21 MySQL Community Server (GPL)

Copyright (c) 2000, 2014, oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> update user set password = password('root') where user = 'root';
Query OK, 4 rows affected (0.00 sec)
Rows matched: 4  Changed: 4  warnings: 0

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

(4) 跳过验证启动 MySQL

重启 mysql 服务并查看状态

sudo service mysql stop

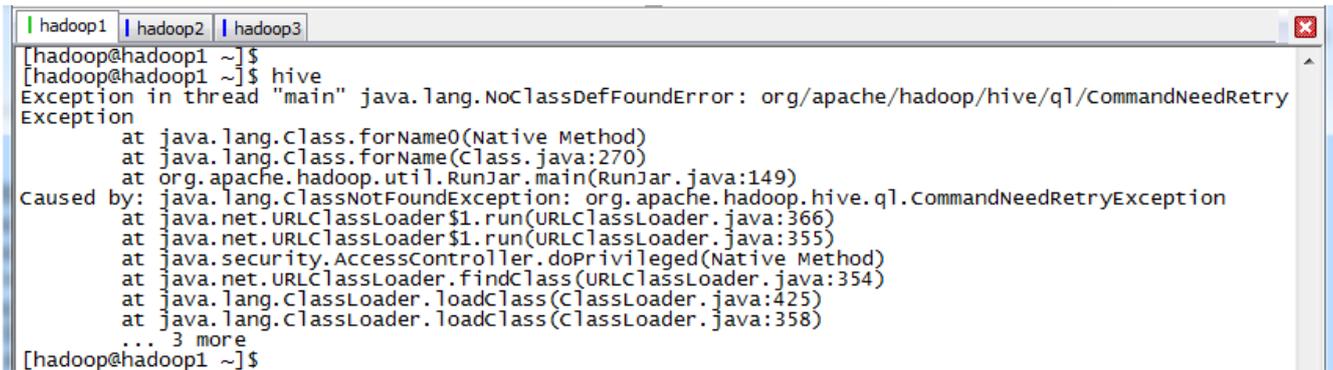
sudo service mysql start

sudo service mysql status

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ sudo service mysql stop
Shutting down MySQL.. [ OK ]
[hadoop@hadoop1 ~]$ sudo service mysql start
Starting MySQL. [ OK ]
[hadoop@hadoop1 ~]$ sudo service mysql status
MySQL running (5504) [ OK ]
[hadoop@hadoop1 ~]$
```

3.2 Hive 启动，报 CommandNeedRetryException 异常

启动 Hive 时，出现 CommandNeedRetryException 异常，具体信息如下：



```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ hive
Exception in thread "main" java.lang.NoClassDefFoundError: org/apache/hadoop/hive/ql/CommandNeedRetryException
    at java.lang.Class.forName0(Native Method)
    at java.lang.Class.forName(Class.java:270)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:149)
Caused by: java.lang.ClassNotFoundException: org.apache.hadoop.hive.ql.CommandNeedRetryException
    at java.net.URLClassLoader$1.run(URLClassLoader.java:366)
    at java.net.URLClassLoader$1.run(URLClassLoader.java:355)
    at java.security.AccessController.doPrivileged(Native Method)
    at java.net.URLClassLoader.findClass(URLClassLoader.java:354)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:425)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:358)
    ... 3 more
[hadoop@hadoop1 ~]$
```

Exception in thread "main"

java.lang.NoClassDefFoundError:org/apache/hadoop/hive/ql/CommandNeedRetryException

at java.lang.Class.forName0(Native Method)

at java.lang.Class.forName(Class.java:270)

at org.apache.hadoop.util.RunJar.main(RunJar.java:149)

Caused by: java.lang.ClassNotFoundException:

org.apache.hadoop.hive.ql.CommandNeedRetryException

at java.net.URLClassLoader\$1.run(URLClassLoader.java:366)

at java.net.URLClassLoader\$1.run(URLClassLoader.java:355)

at java.security.AccessController.doPrivileged(Native Method)

at java.net.URLClassLoader.findClass(URLClassLoader.java:354)

at java.lang.ClassLoader.loadClass(ClassLoader.java:425)

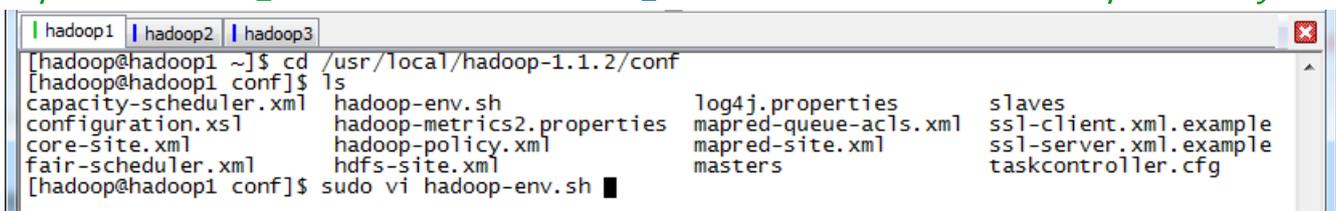
at java.lang.ClassLoader.loadClass(ClassLoader.java:358)

由于以前使用 hadoop 时，修改 hadoop-env.sh 的 HADOOP_CLASSPATH 配置项，由以前的：

export HADOOP_CLASSPATH=/usr/local/hadoop-1.1.2/myclass

修改为：

export HADOOP_CLASSPATH=\$HADOOP_CLASSPATH:/usr/local/hadoop-1.1.2/myclass



```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /usr/local/hadoop-1.1.2/conf
[hadoop@hadoop1 conf]$ ls
capacity-scheduler.xml  hadoop-env.sh          log4j.properties      slaves
configuration.xml      hadoop-metrics2.properties  mapred-queue-acls.xml  ssl-client.xml.example
core-site.xml          hadoop-policy.xml      mapred-site.xml        ssl-server.xml.example
fair-scheduler.xml     hdfs-site.xml          masters                 taskcontroller.cfg
[hadoop@hadoop1 conf]$ sudo vi hadoop-env.sh
```

```
hadoop1 | hadoop2 | hadoop3
# The java implementation to use. Required.
# export JAVA_HOME=/usr/lib/j2sdk1.5-sun
# Extra Java CLASSPATH elements. Optional.
export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:/usr/local/hadoop-1.1.2/myclass
# The maximum amount of heap to use, in MB. Default is 1000.
# export HADOOP_HEAPSIZE=2000
```

3.3 在 Hive 中使用操作语言

启动 Hive 后，使用 HSQL 出现异常，需要启动 metastore 和 hiveserver

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ hive
14/11/24 17:16:06 WARN conf.HiveConf: DEPRECATED: Configuration property hive.metastore.local no longer has any effect. Make sure to provide a valid value for hive.metastore.uris if you are connecting to a remote metastore.
14/11/24 17:16:06 WARN conf.HiveConf: DEPRECATED: hive.metastore.ds.retry.* no longer has any effect. Use hive.hmshandler.retry.* instead
Logging initialized using configuration in jar:file:/usr/local/hive-0.13.0/lib/hive-common-0.13.0.jar!/hive-log4j.properties
Exception in thread "main" java.lang.RuntimeException: java.lang.RuntimeException: Unable to instantiate org.apache.hadoop.hive.metastore.HiveMetaStoreClient
    at org.apache.hadoop.hive.ql.session.SessionState.start(SessionState.java:344)
    at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:681)
    at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:625)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:606)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:156)
```

FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask. java.lang.RuntimeException: Unable to instantiate org.apache.hadoop.hive.metastore.HiveMetaStoreClient

在使用 hive 之前需要启动 metastore 和 hiveserver 服务，通过如下命令启用：

```
hive --service metastore &
hive --service hiveserver &
```

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 bin]$ pwd
/usr/local/hive-0.12.0/bin
[hadoop@hadoop1 bin]$ hive --service metastore &
[1] 5484
[hadoop@hadoop1 bin]$ Starting Hive Metastore Server
14/11/25 10:41:00 WARN conf.HiveConf: DEPRECATED: Configuration property hive.metastore.local no longer has any effect. Make sure to provide a valid value for hive.metastore.uris if you are connecting to a remote metastore.
[hadoop@hadoop1 bin]$ hive --service hiveserver &
[2] 5590
[hadoop@hadoop1 bin]$ Starting Hive Thrift Server
14/11/25 10:41:17 WARN conf.HiveConf: DEPRECATED: Configuration property hive.metastore.local no longer has any effect. Make sure to provide a valid value for hive.metastore.uris if you are connecting to a remote metastore.
[hadoop@hadoop1 bin]$ █
```

启动用通过 jps 命令可以看到两个进行运行在后台

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 bin]$ jps
5484 RunJar
2751 SecondaryNameNode
5590 RunJar
2820 JobTracker
2612 NameNode
5698 Jps
[hadoop@hadoop1 bin]$
```

参考资料：

- (1) 《Hive 体系结构》 <http://blog.csdn.net/zhoudaxia/article/details/8855937>
- (2) 《大数据时代的技术 hive：hive 的数据类型和数据模型》
<http://www.cnblogs.com/sharpxiajun/archive/2013/06/03/3114560.html>