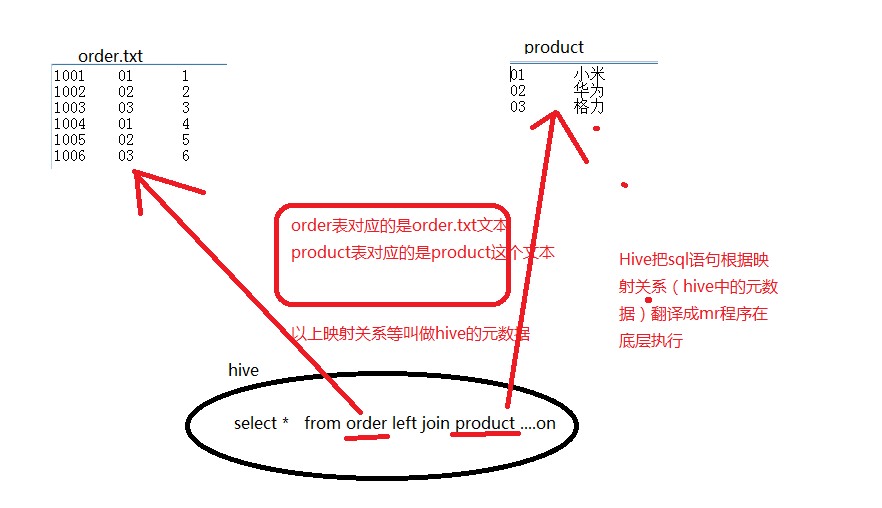
# 第1章 Hive基本概念

## 1.1 什么是Hive

Hive：由Facebook开源用于解决海量结构化日志的数据统计。Hive基于Hadoop来完成工作。

Hive是基于Hadoop的一个数据仓库工具，可以将结构化的数据文件映射为一张表，并提供类SQL查询功能。



**本质是：将HQL转化成MapReduce程序**



1）Hive处理的数据存储在HDFS

2）Hive分析（翻译工作）数据底层的实现是MapReduce

3）执行程序运行在Yarn上

Hadoop = HDFS + MR + YARN;

综上所述，可以认为Hive这个软件工具是Hadoop的一个客户端

要想使用hive，要考虑存储元数据信息、还要把yarn集群开启

## 1.2 Hive的优缺点

### 1.2.1 优点

1. 操作接口采用类SQL语法，提供快速开发的能力（简单、容易上手）。
2. 避免了去写MapReduce，减少开发人员的学习成本。
3. Hive的执行延迟比较高，因此Hive常用于数据分析，对实时性要求不高的场合。
4. Hive优势在于处理大数据，对于处理小数据没有优势，因为Hive的执行延迟比较高。
5. Hive支持用户自定义函数，用户可以根据自己的需求来实现自己的函数。

### 1.2.2 缺点

1．Hive的HQL表达能力有限

（1）迭代式算法无法表达

（2）数据挖掘方面不擅长

2．Hive的效率比较低

（1）Hive自动生成的MapReduce作业，通常情况下不够智能化

（2）Hive调优比较困难，粒度较粗

## 1.3 Hive架构原理



图1 Hive架构原理

1．用户接口：Client

CLI（hive shell）、JDBC/ODBC(java访问hive)、WEBUI（浏览器访问hive）

2．元数据：Metastore

元数据包括：表名、表所属的数据库（默认是default）、表的拥有者、列/分区字段、表的类型（是否是外部表）、表的数据所在目录等；

默认存储在自带的derby数据库中，推荐使用MySQL存储Metastore

3．Hadoop

使用HDFS进行存储，使用MapReduce进行计算。

4．驱动器：Driver

（1）解析器（SQL Parser）：将SQL字符串转换成抽象语法树AST，这一步一般都用第三方工具库完成，比如antlr；对AST进行语法分析，比如表是否存在、字段是否存在、SQL语义是否有误。

（2）编译器（Physical Plan）：将AST编译生成逻辑执行计划。

（3）优化器（Query Optimizer）：对逻辑执行计划进行优化。

（4）执行器（Execution）：把逻辑执行计划转换成可以运行的物理计划。对于Hive来说，就是MR/Spark。



Hive通过给用户提供的一系列交互接口，接收到用户的指令(SQL)，使用自己的Driver，结合元数据(MetaStore)，将这些指令翻译成MapReduce，提交到Hadoop中执行，最后，将执行返回的结果输出到用户交互接口。

## 1.4 Hive和数据库比较

注意：Hive是一个翻译工具，将sql翻译为底层mr程序的，它不是数据库，只不过在表现形式上和数据库有很多类似而已（比如表、database、字段等）

由于 Hive 采用了类似SQL 的查询语言 HQL(Hive Query Language)，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。本文将从多个方面来阐述 Hive 和数据库的差异。数据库可以用在 Online（在线） 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。

### 1.4.1 查询语言

由于SQL被广泛的应用在数据仓库中，因此，专门针对Hive的特性设计了类SQL的查询语言HQL。熟悉SQL开发的开发者可以很方便的使用Hive进行开发。

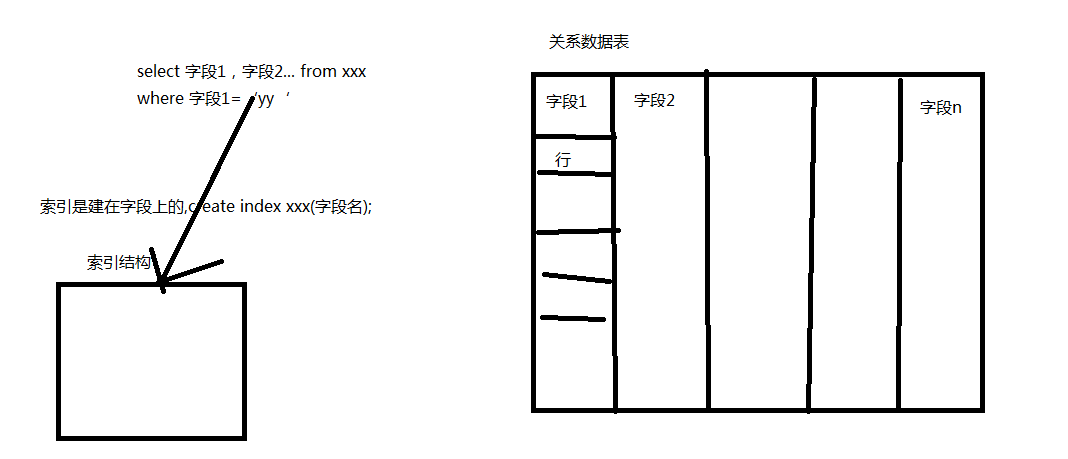
### 1.4.2 数据存储位置

Hive 是建立在 Hadoop 之上的，所有 Hive 的数据都是存储在 HDFS 中的。而数据库则可以将数据保存在块设备或者本地文件系统中。

### 1.4.3 数据更新

由于Hive是针对数据仓库应用设计的，而数据仓库的内容是读多写少的。因此，Hive中不建议对数据的改写，所有的数据都是在加载的时候确定好的。而数据库中的数据通常是需要经常进行修改的，因此可以使用 INSERT INTO …  VALUES 添加数据，使用 UPDATE … SET修改数据。

### 1.4.4 索引



Hive在加载数据的过程中不会对数据进行任何处理，甚至不会对数据进行扫描，因此也没有对数据中的某些Key建立索引。Hive要访问数据中满足条件的特定值时，需要暴力扫描整个数据，因此访问延迟较高。由于 MapReduce 的引入， Hive 可以并行访问数据，因此即使没有索引，对于[大数据](http://lib.csdn.net/base/hadoop" \t "_blank" \o "Hadoop知识库)量的访问，Hive 仍然可以体现出优势。数据库中，通常会针对一个或者几个列建立索引，因此对于少量的特定条件的数据的访问，数据库可以有很高的效率，较低的延迟。由于数据的访问延迟较高，决定了 Hive 不适合在线数据查询。

### 1.4.5 执行

Hive中大多数查询的执行是通过 Hadoop 提供的 MapReduce 来实现的。而数据库通常有自己的执行引擎。innoDb引擎

### 1.4.6 执行延迟

Hive 在查询数据的时候，由于没有索引，需要扫描整个表，因此延迟较高。另外一个导致 Hive 执行延迟高的因素是 MapReduce框架。由于MapReduce 本身具有较高的延迟，因此在利用MapReduce 执行Hive查询时，也会有较高的延迟。相对的，数据库的执行延迟较低。当然，这个低是有条件的，即数据规模较小，当数据规模大到超过数据库的处理能力的时候，Hive的并行计算显然能体现出优势。

### 1.4.7 可扩展性

由于Hive是建立在Hadoop之上的，因此Hive的可扩展性是和Hadoop的可扩展性是一致的（世界上最大的Hadoop 集群在 百度!，2015年的规模在1.3万 台节点左右）。而数据库由于 ACID 语义的严格限制，扩展行非常有限。目前最先进的并行数据库 [Oracle](http://lib.csdn.net/base/oracle" \t "_blank" \o "Oracle知识库) 在理论上的扩展能力也只有100台左右。

### 1.4.8 数据规模

由于Hive建立在集群上并可以利用MapReduce进行并行计算，因此可以支持很大规模的数据；对应的，数据库可以支持的数据规模较小。

# 第2章 Hive安装

## 2.1 Hive安装地址

1．Hive官网地址

http://hive.apache.org/

2．文档查看地址

https://cwiki.apache.org/confluence/display/Hive/GettingStarted

3．下载地址

http://archive.apache.org/dist/hive/

4．github地址

https://github.com/apache/hive

## 2.2 Hive安装部署

1．Hive安装及配置

（1）把apache-hive-1.2.1-bin.tar.gz上传到linux的/opt/software目录下

（2）解压apache-hive-1.2.1-bin.tar.gz到/opt/module/目录下面

[root@hadoop101 software]# tar -zxvf apache-hive-1.2.1-bin.tar.gz -C /opt/module/

（3）修改apache-hive-1.2.1-bin.tar.gz的名称为hive

[root@hadoop101 module]# mv apache-hive-1.2.1-bin/ hive

（4）修改/opt/module/hive/conf目录下的hive-env.sh.template名称为hive-env.sh

[root@hadoop101 conf]# mv hive-env.sh.template hive-env.sh

（5）配置hive-env.sh文件

（a）配置HADOOP\_HOME路径

export HADOOP\_HOME=/opt/module/hadoop-2.7.2

（b）配置HIVE\_CONF\_DIR路径

export HIVE\_CONF\_DIR=/opt/module/hive/conf

2．Hadoop集群配置

（1）必须启动hdfs和yarn

[root@hadoop101 hadoop-2.7.2]# sbin/start-dfs.sh

[root@hadoop102 hadoop-2.7.2]# sbin/start-yarn.sh

1. 在HDFS上创建/tmp和/user/hive/warehouse两个目录并修改他们的同组权限可写

[root@hadoop101 hadoop-2.7.2]# bin/hadoop fs -mkdir /tmp

[root@hadoop101 hadoop-2.7.2]# bin/hadoop fs -mkdir -p /user/hive/warehouse

[root@hadoop101 hadoop-2.7.2]# bin/hadoop fs -chmod g+w /tmp

[root@hadoop101 hadoop-2.7.2]# bin/hadoop fs -chmod g+w /user/hive/warehouse

或者在配置文件中关闭权限检查 在hadoop 的hdfs-site.xml 中

<property>

<name>dfs.permissions.enable</name>

<value>false</value>

</property>

3．Hive基本操作

（1）启动hive

[root@hadoop101 hive]# bin/hive

（2）查看数据库

hive> show databases;

（3）打开默认数据库

hive> use default;

（4）显示default数据库中的表

hive> show tables;

（5）创建一张表

hive> create table student(id int, name string);

（6）显示数据库中有几张表

hive> show tables;

（7）查看表的结构

hive> desc student;

（8）向表中插入数据

hive> insert into student values(1000,"ss");

（9）查询表中数据

hive> select \* from student;

（10）退出hive

hive> quit;exit;

## 2.3 将本地文件导入Hive案例

需求

将本地/opt/module/datas/student.txt这个目录下的数据导入到hive的student(id int, name string)表中。

1．数据准备

在/opt/module/datas这个目录下准备数据

（1）在/opt/module/目录下创建datas

[root@hadoop101 module]# mkdir datas

（2）在/opt/module/datas/目录下创建student.txt文件并添加数据

[root@hadoop101 datas]# touch student.txt

[root@hadoop101 datas]# vim student.txt

1001 zhangshan

1002 lishi

1003 zhaoliu

注意以tab键间隔。

2．Hive实际操作

（1）启动hive

[root@hadoop101 hive]# bin/hive

（2）显示数据库

hive> show databases;

（3）使用default数据库

hive> use default;

（4）显示default数据库中的表

hive> show tables;

（5）删除已创建的student表

hive> drop table student;

（6）创建student表, 并声明文件分隔符’\t’

hive> create table student(id int, name string) ROW FORMAT DELIMITED FIELDS TERMINATED

BY '\t';

（7）加载/opt/module/datas/student.txt 文件到student数据库表中。 双引号也可以

hive> load data local inpath '/opt/module/datas/student.txt' into table student;

（8）Hive查询结果

hive> select \* from student;

OK

1001 zhangshan

1002 lishi

1003 zhaoliu

Time taken: 0.266 seconds, Fetched: 3 row(s)

3．遇到的问题

再打开一个客户端窗口启动hive，会产生java.sql.SQLException异常。

Exception in thread "main" java.lang.RuntimeException: java.lang.RuntimeException:

Unable to instantiate

org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClient

at org.apache.hadoop.hive.ql.session.SessionState.start(SessionState.java:522)

at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:677)

at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:621)

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.run(RunJar.java:221)

at org.apache.hadoop.util.RunJar.main(RunJar.java:136)

Caused by: java.lang.RuntimeException: Unable to instantiate org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClient

at org.apache.hadoop.hive.metastore.MetaStoreUtils.newInstance(MetaStoreUtils.java:1523)

at org.apache.hadoop.hive.metastore.RetryingMetaStoreClient.<init>(RetryingMetaStoreClient.java:86)

at org.apache.hadoop.hive.metastore.RetryingMetaStoreClient.getProxy(RetryingMetaStoreClient.java:132)

at org.apache.hadoop.hive.metastore.RetryingMetaStoreClient.getProxy(RetryingMetaStoreClient.java:104)

at org.apache.hadoop.hive.ql.metadata.Hive.createMetaStoreClient(Hive.java:3005)

at org.apache.hadoop.hive.ql.metadata.Hive.getMSC(Hive.java:3024)

at org.apache.hadoop.hive.ql.session.SessionState.start(SessionState.java:503)

... 8 more

原因是，**Metastore默认存储在自带的derby数据库中，推荐使用MySQL存储Metastore;**

## 2.4 MySql安装（使用root用户）

### 2.4.1 安装包准备

1．查看mysql是否安装，如果安装了，卸载mysql

（1）查看

[root@hadoop101 桌面]# rpm -qa|grep mysql

mysql-libs-5.1.73-7.el6.x86\_64

（2）卸载

[root@hadoop101 桌面]# rpm -e --nodeps mysql-libs-5.1.73-7.el6.x86\_64

2．解压mysql-libs.zip文件到当前目录

[root@hadoop101 software]# unzip mysql-libs.zip

[root@hadoop101 software]# ls

mysql-libs.zip

mysql-libs

3．进入到mysql-libs文件夹下

[root@hadoop101 mysql-libs]# ll

总用量 76048

-rw-r--r--. 1 root root 18509960 3月 26 2015 MySQL-client-5.6.24-1.el6.x86\_64.rpm

-rw-r--r--. 1 root root 3575135 12月 1 2013 mysql-connector-java-5.1.27.tar.gz

-rw-r--r--. 1 root root 55782196 3月 26 2015 MySQL-server-5.6.24-1.el6.x86\_64.rpm

### 2.4.2 安装MySql服务器

1．安装mysql服务端

[root@hadoop101 mysql-libs]# rpm -ivh MySQL-server-5.6.24-1.el6.x86\_64.rpm，

报错需要安装mysql的依赖

安装mysql的依赖库

yum install perl

yum install net-tools

2．查看产生的随机密码

[root@hadoop101 mysql-libs]# cat /root/.mysql\_secret

OEXaQuS8IWkG19Xs

3．查看mysql状态

[root@hadoop101 mysql-libs]# service mysql status

4．启动mysql

[root@hadoop101 mysql-libs]# service mysql start

### 2.4.3 安装MySql客户端（使用root用户）

1．安装mysql客户端

[root@hadoop101 mysql-libs]# rpm -ivh MySQL-client-5.6.24-1.el6.x86\_64.rpm

2．连接mysql

[root@hadoop101 mysql-libs]# mysql -uroot -pOEXaQuS8IWkG19Xs

3．修改密码

mysql>SET PASSWORD=PASSWORD('000000'); **(大小写均可)**

4．退出mysql

mysql>exit;

### 2.4.4 MySql中user表中主机配置

配置：只要是root用户+密码，在任何主机上都能登录MySQL数据库（可远程访问）。

1．进入mysql

[root@hadoop101 mysql-libs]# mysql -uroot -p000000

2．显示数据库

mysql>show databases;

3．使用mysql数据库

mysql>use mysql;

4．展示mysql数据库中的所有表

mysql>show tables;

5．展示user表的结构

mysql>desc user;

6．查询user表

mysql>select user, host, password from user; **(大小写均可)**

7．修改user表，把Host字段内容修改为%

mysql>update user set host='%' where host='localhost';

8．删除root用户的其他host

mysql>delete from user where Host='hadoop101';

mysql>delete from user where Host='127.0.0.1';

mysql>delete from user where Host='::1';

9．刷新

mysql>flush privileges;

10．退出

mysql>quit;

## 2.5 Hive元数据配置到MySql

### 2.5.1 驱动拷贝

1．在/opt/software/mysql-libs目录下解压mysql-connector-java-5.1.27.tar.gz驱动包

[root@hadoop101 mysql-libs]# tar -zxvf mysql-connector-java-5.1.27.tar.gz

2.拷贝/opt/software/mysql-libs/mysql-connector-java-5.1.27目录下的mysql-connector-java-5.1.27-bin.jar到/opt/module/hive/lib/

[root@hadoop101 mysql-connector-java-5.1.27]# cp mysql-connector-java-5.1.27-bin.jar

/opt/module/hive/lib/

### 2.5.2 配置Metastore到MySql

1．在/opt/module/hive/conf目录下创建一个hive-site.xml

[root@hadoop101 conf]# touch hive-site.xml

[root@hadoop101 conf]# vi hive-site.xml

2．根据官方文档配置参数，拷贝数据到hive-site.xml文件中

<https://cwiki.apache.org/confluence/display/Hive/AdminManual+MetastoreAdmin>

<?xml version="1.0"?>

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

<configuration>

<property>

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

<value>jdbc:mysql://hadoop101:3306/metastore?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>root</value>

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

</property>

<property>

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

<value>000000</value>

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

</property>

</configuration>

3．配置完毕后，如果启动hive异常，可以重新启动虚拟机。（重启后，别忘了启动hadoop集群）

### 2.5.3 多窗口启动Hive测试

1．先启动MySQL

[root@hadoop101 mysql-libs]# mysql -uroot -p000000

查看有几个数据库

mysql> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| test |

+--------------------+

2．再次打开多个窗口，分别启动hive

[root@hadoop101 hive]# bin/hive

3．启动hive后，回到MySQL窗口查看数据库，显示增加了metastore数据库

mysql> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| metastore |

| mysql |

| performance\_schema |

| test |

+--------------------+

## 2.6 Hive常用交互命令

|  |
| --- |
| [root@hadoop101 hive]# bin/hive -help  usage: hive  -d,--define <key=value> Variable subsitution to apply to hive  commands. e.g. -d A=B or --define A=B  --database <databasename> Specify the database to use  -e <quoted-query-string> SQL from command line  -f <filename> SQL from files  -H,--help Print help information  --hiveconf <property=value> Use value for given property  --hivevar <key=value> Variable subsitution to apply to hive  commands. e.g. --hivevar A=B  -i <filename> Initialization SQL file  -S,--silent Silent mode in interactive shell  -v,--verbose Verbose mode (echo executed SQL to the console) |

1．“-e”不进入hive的交互窗口执行sql语句

[root@hadoop101 hive]# bin/hive -e "select id from student;"

2．“-f”执行脚本中sql语句

（1）在/opt/module/datas目录下创建hivef.sql文件

[root@hadoop101 datas]# touch hivef.sql

文件中写入正确的sql语句

select \*from student;

（2）执行文件中的sql语句

[root@hadoop101 hive]# bin/hive -f /opt/module/datas/hivef.sql

（3）执行文件中的sql语句并将结果写入文件中

[root@hadoop101 hive]# bin/hive -f /opt/module/datas/hivef.sql > /opt/module/datas/hive\_result.txt

## 2.7 Hive其他命令操作

1．退出hive窗口：

hive(default)>exit;

或者

hive(default)>quit;

2．查看在hive中输入的所有历史命令

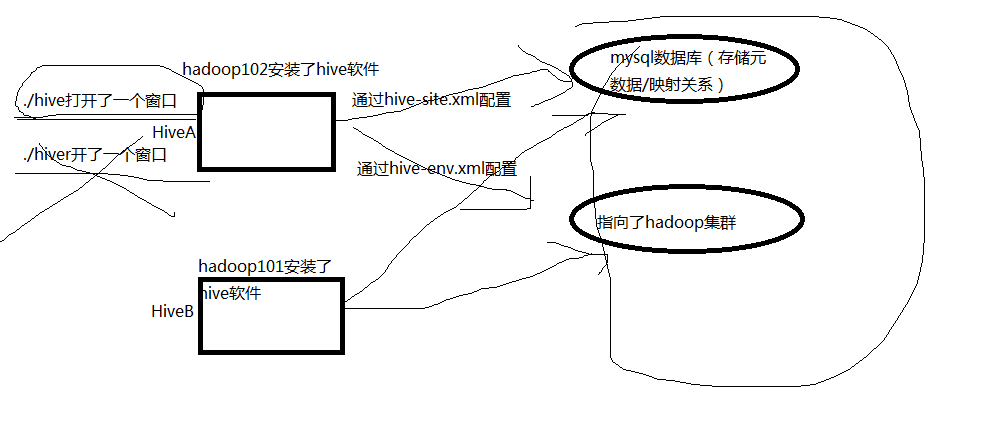
（1）进入到当前用户的根目录/root

（2）查看. hivehistory文件

[bigdata@hadoop102 ~]# cat .hivehistory

## 2.8 Hive常见属性配置

Hive工具的关系：



### 2.8.1 Hive数据仓库位置配置

1）Default数据仓库的最原始位置是在hdfs上的：/user/hive/warehouse路径下。

2）在仓库目录下，没有对默认的数据库default创建文件夹。如果某张表属于default数据库，直接在数据仓库目录下创建一个文件夹。

3）修改default数据仓库原始位置（将hive-default.xml.template如下配置信息拷贝到hive-site.xml文件中）。

|  |
| --- |
| <property>  <name>hive.metastore.warehouse.dir</name>  <value>/user/hive/warehouse</value>  <description>location of default database for the warehouse</description>  </property> |

配置同组用户有执行权限

bin/hdfs dfs -chmod g+w /user/hive/warehouse

### 2.8.2 查询后信息显示配置

1）在hive-site.xml文件中添加如下配置信息，就可以实现显示当前数据库，以及查询表的头信息配置。

<property>

<name>hive.cli.print.header</name>

<value>true</value>

</property>

<property>

<name>hive.cli.print.current.db</name>

<value>true</value>

</property>

2）重新启动hive，对比配置前后差异。

（1）配置前，如图2所示

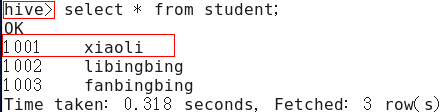


图2 配置前

（2）配置后，如图3所示

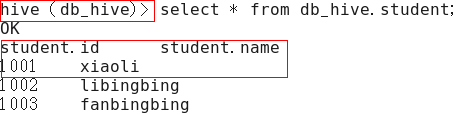


图3 配置后

### 2.8.3 Hive运行日志信息配置

1．Hive的log默认存放在/tmp/root/hive.log目录下（当前用户名下）

2．修改hive的log存放日志到/opt/module/hive/logs

（1）修改/opt/module/hive/conf/hive-log4j.properties.template文件名称为

hive-log4j.properties

[root@hadoop101 conf]# pwd

/opt/module/hive/conf

[root@hadoop101 conf]# mv hive-log4j.properties.template hive-log4j.properties

（2）在hive-log4j.properties文件中修改log存放位置

hive.log.dir=/opt/module/hive/logs

# 第3章 Hive数据类型

## 3.1 基本数据类型

表1

|  |  |  |  |
| --- | --- | --- | --- |
| Hive数据类型 | Java数据类型 | 长度 | 例子 |
| TINYINT | byte | 1byte有符号整数 | 20 |
| SMALINT | short | 2byte有符号整数 | 20 |
| INT | int | 4byte有符号整数 | 20 |
| BIGINT | long | 8byte有符号整数 | 20 |
| BOOLEAN | boolean | 布尔类型，true或者false | TRUE FALSE |
| FLOAT | float | 单精度浮点数 | 3.14159 |
| DOUBLE | double | 双精度浮点数 | 3.14159 |
| STRING | string | 字符系列。可以指定字符集。可以使用单引号或者双引号。 | ‘now is the time’ “for all good men” |
| TIMESTAMP |  | 时间类型 |  |
| BINARY |  | 字节数组 |  |

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。

## 3.3 类型转化

Hive的原子数据类型是可以进行隐式转换的，类似于Java的类型转换，例如某表达式使用INT类型，TINYINT会自动转换为INT类型，但是Hive不会进行反向转化，例如，某表达式使用TINYINT类型，INT不会自动转换为TINYINT类型，它会返回错误

隐式类型转换规则如下

（1）任何整数类型都可以隐式地转换为一个范围更广的类型，如TINYINT可以转换成INT，INT可以转换成BIGINT。

（2）所有整数类型、FLOAT和STRING类型都可以隐式地转换成DOUBLE。

（3）TINYINT、SMALLINT、INT都可以转换为FLOAT。

（4）BOOLEAN类型不可以转换为任何其它的类型。

# 第4章 DDL数据定义

## 4.1 创建数据库

1）创建一个数据库，数据库在HDFS上的默认存储路径是/user/hive/warehouse/\*.db。

hive (default)> create database db\_hive;

2）避免要创建的数据库已经存在错误，增加if not exists判断。（标准写法）

|  |
| --- |
| hive (default)> create database db\_hive;  FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask. Database db\_hive already exists  hive (default)> create database if not exists db\_hive; |

3）创建一个数据库，指定数据库在HDFS上存放的位置

hive (default)> create database db\_hive2 location '/db\_hive2.db';

## 4.2 查询数据库

### 4.2.1 显示数据库

1．显示数据库

hive> show databases;

2．过滤显示查询的数据库

hive> show databases like 'db\_hive\*';

OK

db\_hive

db\_hive\_1

### 4.2.2 查看数据库详情

1．显示数据库信息

hive> desc database db\_hive;

OK

db\_hive hdfs://hadoop102:9000/user/hive/warehouse/db\_hive.db bigdataUSER

### 4.3.3 切换当前数据库

hive (default)> use db\_hive;

## 4.4 删除数据库

1．删除空数据库

hive>drop database db\_hive2;

2．如果删除的数据库不存在，最好采用 if exists判断数据库是否存在

hive> drop database db\_hive;

FAILED: SemanticException [Error 10072]: Database does not exist: db\_hive

hive> drop database if exists db\_hive2;

3．如果数据库不为空，可以采用cascade命令，强制删除

hive> drop database db\_hive;

FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask. InvalidOperationException(message:Database db\_hive is not empty. One or more tables exist.)

hive> drop database db\_hive cascade;

## 4.5 创建表

1．建表语法

CREATE [EXTERNAL] TABLE [IF NOT EXISTS] table\_name

[(col\_name data\_type [COMMENT col\_comment], ...)]

[COMMENT table\_comment]

[ROW FORMAT row\_format]

[STORED AS file\_format]

[LOCATION hdfs\_path]

LIKE tablename

2．字段解释说明

（1）CREATE TABLE 创建一个指定名字的表。如果相同名字的表已经存在，则抛出异常；用户可以用 IF NOT EXISTS 选项来忽略这个异常。

（2）EXTERNAL关键字可以让用户创建一个外部表，在建表的同时指定一个指向实际数据的路径（LOCATION），Hive创建内部表时，会将数据移动到数据仓库指向的路径；若创建外部表，仅记录数据所在的路径，不对数据的位置做任何改变。在删除表的时候，内部表的元数据和数据会被一起删除，而外部表只删除元数据，不删除数据。

（3）COMMENT：为表和列添加注释。

（4）ROW FORMAT

DELIMITED FIELDS TERMINATED BY char

（5）STORED AS指定存储文件类型

常用的存储文件类型：SEQUENCEFILE（二进制序列文件）、TEXTFILE（文本）、RCFILE（列式存储格式文件）

如果文件数据是纯文本，可以使用STORED AS TEXTFILE。如果数据需要压缩，使用 STORED AS SEQUENCEFILE。

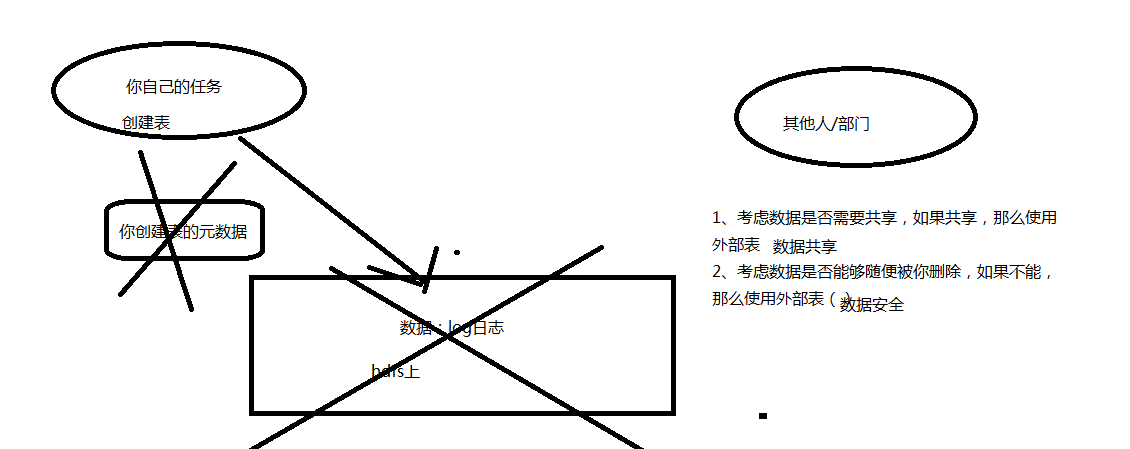
（6）LOCATION ：指定表在HDFS上的存储位置。

（7）LIKE允许用户复制现有的表结构，但是不复制数据。

如果当前创建的表create table table1，希望table1和已经存在的table0的表结构相同，就可以使用like指令直接创建，create table table1 like table0;

### 4.5.1 管理表

内外部表选择原则



1．理论

默认创建的表都是所谓的管理表，有时也被称为内部表。因为这种表，Hive会（或多或少地）控制着数据的生命周期。Hive默认情况下会将这些表的数据存储在由配置项hive.metastore.warehouse.dir(例如，/user/hive/warehouse)所定义的目录的子目录下。 当我们删除一个管理表时，Hive也会删除这个表中数据(元数据和表数据都会删除掉)。管理表不适合和其他工具共享数据。

2．案例实操

（1）普通创建表

|  |
| --- |
| create table if not exists student3(  id int, name string  )  row format delimited fields terminated by '\t'  stored as textfile  location '/user/hive/warehouse/student3'; |

（2）根据查询结果创建表（查询的结果会添加到新创建的表中）

|  |
| --- |
| create table if not exists student3 as select id, name from student; |

（3）根据已经存在的表结构创建表

|  |
| --- |
| create table if not exists student4 like student; |

（4）查询表的类型

hive (default)> desc formatted student3;

Table Type: MANAGED\_TABLE

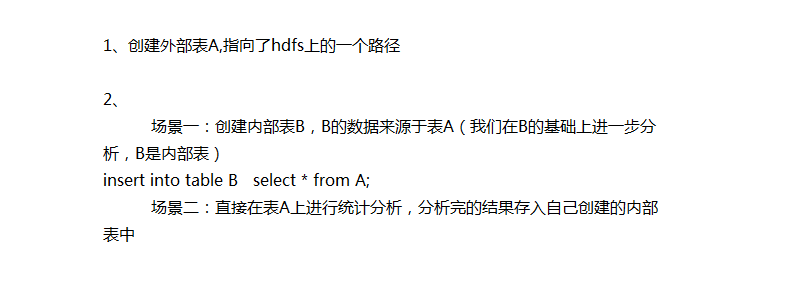
### 4.5.2 外部表

1．理论

因为表是外部表，所以Hive并非认为其完全拥有这份数据。删除该表并不会删除掉这份数据，不过描述表的元数据信息会被删除掉 。

2．管理表和外部表的使用场景

每天将收集到的网站日志定期流入HDFS文本文件。在外部表（原始日志表）的基础上做大量的统计分析，用到的中间表、结果表使用内部表存储，数据通过SELECT+INSERT进入内部表。



3．案例实操

分别创建部门和员工外部表，并向表中导入数据。

（1）原始数据

（2）建表语句

创建部门表

|  |
| --- |
| create external table if not exists dept(  deptno int,  dname string,  loc int  )  row format delimited fields terminated by '\t'; |

创建员工表

|  |
| --- |
| create external table if not exists emp(  empno int,  ename string,  job string,  mgr int,  hiredate string,  sal double,  comm double,  deptno int)  row format delimited fields terminated by '\t'; |

（3）查看创建的表

hive (default)> show tables;

OK

tab\_name

dept

emp

（4）向外部表中导入数据

导入数据

hive (default)> load data local inpath '/opt/module/datas/dept.txt' into table default.dept;

hive (default)> load data local inpath '/opt/module/datas/emp.txt' into table default.emp;

查询结果

hive (default)> select \* from emp;

hive (default)> select \* from dept;

（5）查看表格式化数据

hive (default)> desc formatted dept;

Table Type: EXTERNAL\_TABLE

## 4.7 修改表(了解)

### 4.7.1 重命名表

1．语法

ALTER TABLE table\_name RENAME TO new\_table\_name

2．实操案例

hive (default)> create table dept\_partition(deptno int, dname string, loc string ) row format delimited fields terminated by '\t';

hive (default)> alter table dept\_partition rename to dept\_partition2;

hive (default)> alter table dept\_partition2 rename to dept\_partition;

### 4.7.3 增加/修改列信息

1．语法

更新列

ALTER TABLE table\_name CHANGE [COLUMN] col\_old\_name col\_new\_name column\_type [COMMENT col\_comment]

增加列

ALTER TABLE table\_name ADD COLUMNS (col\_name data\_type [COMMENT col\_comment], ...)

注：ADD是代表新增一字段，字段位置在所有列后面

2．实操案例

（1）查询表结构

hive> desc dept\_partition;

（2）添加列

hive (default)> alter table dept\_partition add columns(deptdesc string);

（3）查询表结构

hive> desc dept\_partition;

（4）更新列

hive (default)> alter table dept\_partition change column deptdesc desc int;

（5）查询表结构

hive> desc dept\_partition;

## 4.8 删除表

hive (default)> drop table dept\_partition;

# 第5章 DML数据操作

## 5.1 数据导入

### 5.1.1 向表中装载数据（Load）

1．语法

hive> load data [local] inpath '/opt/module/datas/student.txt' [overwrite] into table student

（1）load data:表示加载数据

（2）local:表示从本地加载数据到hive表（复制）；否则从HDFS加载数据到hive表（移动）

（3）inpath:表示加载数据的路径

（4）overwrite into:表示覆盖表中已有数据，否则表示追加

（5）into table:表示加载到哪张表

（6）student:表示具体的表

2．实操案例

（0）创建一张表

hive (default)> create table student(id string, name string) row format delimited fields terminated by '\t';

（1）加载本地文件到hive

hive (default)> load data local inpath '/opt/module/datas/student.txt' into table default.student;

（2）加载HDFS文件到hive中(会把hdfs的数据移到hive目录)

上传文件到HDFS

hive (default)> dfs -mkdir -p /user/root/hive;

hive (default)> dfs -put /opt/module/datas/student.txt /user/root/hive;

加载HDFS上数据(注意此处没有local)

hive (default)> load data inpath '/user/root/hive/student.txt' into table default.student;

（3）加载数据覆盖表中已有的数据(会把hdfs的数据移到hive目录)

上传文件到HDFS

hive (default)> dfs -put /opt/module/datas/student.txt /user/root/hive;

加载数据覆盖表中已有的数据

hive (default)> load data inpath '/user/root/hive/student.txt' overwrite into table default.student;

### 5.1.2 通过查询语句向表中插入数据（Insert）

必须提前创建表

1．创建一张表

hive (default)> create table student(id int, name string) row format delimited fields terminated by '\t';

2．基本插入数据

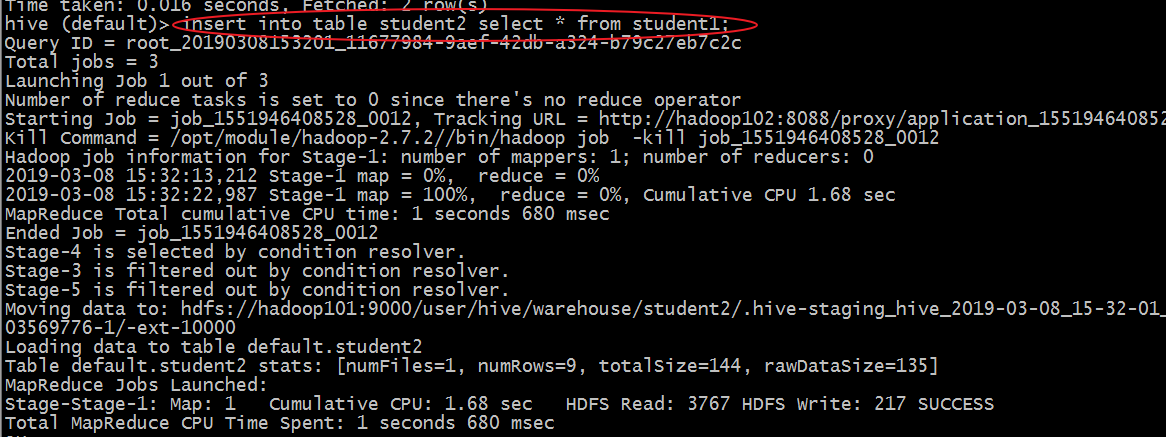
hive (default)> insert into table student values(1,'wangwang');

hive (default)> create table student2(id int, name string) row format delimited fields terminated by '\t';

hive (default)> insert into table student2 select \* from student;

hive (default)> select \* from student2;

快速复制表数据



### 5.1.3 查询语句中创建表并加载数据（As Select）

不必提前创建表，将查询结果插入的同时会自动创建，如果表已经存在，则不会覆盖。

根据查询结果创建表（查询的结果会添加到新创建的表中）

create table if not exists student3

as select id, name from student;

### 5.1.4 创建表时通过Location指定加载数据路径

1．创建表，并指定在hdfs上的位置

hive (default)> create table if not exists student4(

id int, name string

)

row format delimited fields terminated by '\t'

location '/user/hive/warehouse/student4';

2．上传数据到hdfs上

hive (default)> dfs -put /opt/module/datas/student.txt

/user/hive/warehouse/student4;

3．查询数据

hive (default)> select \* from student4;

### 5.1.5 Import数据到指定Hive表中[了解]

注意：先用export导出后，再将数据导入，无需提前创建表。

hive (default)> import table student8 from

'/user/hive/warehouse/export/student';

## 5.2 数据导出

### 5.2.1 Insert导出（会自动创建导出目录）

1．将查询的结果导出到本地

hive (default)> insert overwrite **local** directory '/opt/module/datas/export/student'

select \* from student;

2．将查询的结果格式化导出到本地

hive(default)>insert overwrite local directory '/opt/module/datas/export/student1'

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' select \* from student;

3．将查询的结果导出到HDFS上(没有local)

hive (default)> insert overwrite directory '/user/bigdata/student2'

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

select \* from student;

### 5.2.2 Hadoop命令导出到本地

hive (default)> dfs -get /user/hive/warehouse/student/student.txt

/opt/module/datas/export/student3.txt;

### 5.2.3 Hive Shell 命令导出到本地

基本语法：（hive -f/-e 执行语句或者脚本 > file）

[root@hadoop101 hive]# bin/hive -e 'select \* from default.student;' > /opt/module/datas/export/student4.txt

### 5.2.4 Export导出到HDFS上[了解]

(default)>export table default.student to

'/user/hive/warehouse/export/student';

### 5.2.5 Sqoop导出

后续课程专门讲。

## 5.3 清除表中数据（Truncate）

注意：Truncate只能删除管理表，不能删除外部表中数据

hive (default)> truncate table student;