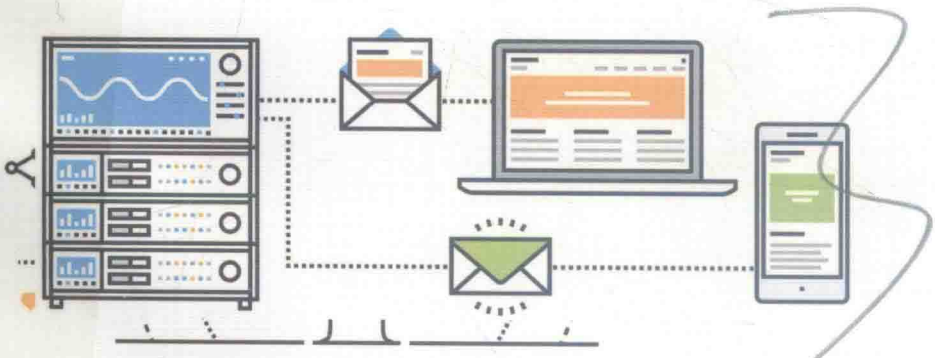
普通高等教育新工科人才培养规划教材

（大数据专业）

Hive 编程技术与应用

主　编　孙　帅　王美佳
副主编　李紫薇　邹先锋
张美娟　臧红久



中国水利水电出版社
www.waterpub.com.cn

普通高等教育新工科人才培养规划教材（大数据专业）

Hive 编程技术与应用

主　编　孙　帅　王美佳

副主编　李紫薇　邹先锋　张美娟　臧红久

中国水利水电出版社
www.waterpub.com.cn

·北京·

内 容 提 要

本书通过原理加案例的方式系统地讲解了 Hive 编程技术，使读者能够全面地了解使用 Hive 的开发流程。书中精心安排了 Hive 的原理分析、架构特点、环境搭建、HiveQL 使用等内容，给出了大量的开发案例及其开发过程，使读者对 Hive 开发有直观的印象。

全书共 10 章：前 6 章系统讲解 Hive 工作原理、特点，Hive 架构，HiveQL 表操作，HiveQL 数据操作，HiveQL 查询，Hive 安装与配置，Hive 自定义函数；第 8～10 章是综合案例部分，通过案例帮助读者掌握整个大数据项目的开发流程，包括数据清洗、数据处理、数据导入导出。本书知识结构简单明了，案例生动具体，内容设计新颖，思路清晰。

本书不仅可作为普通高校大数据相关专业的教材，也可以作为想继续深入了解大数据编程的读者的参考书，还可作为各类相关培训班的培训教材。

图书在版编目（CIP）数据

Hive编程技术与应用 / 孙帅，王美佳主编. -- 北京：中国水利水电出版社，2018.9
普通高等教育新工科人才培养规划教材. 大数据专业
ISBN 978-7-5170-6914-0

Ⅰ. ①H… Ⅱ. ①孙… ②王… Ⅲ. ①数据库系统－程序设计－高等学校－教材 Ⅳ. ①TP311.13

中国版本图书馆CIP数据核字(2018)第218707号

策划编辑：石永峰　责任编辑：张玉玲　加工编辑：张青月　封面设计：梁　燕

书　名	普通高等教育新工科人才培养规划教材（大数据专业） Hive 编程技术与应用 Hive BIANCHENG JISHU YU YINGYONG
作　者	主　编　孙　帅　王美佳 副主编　李紫薇　邹先锋　张美娟　臧红久
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（营销中心）、82562819（万水）
经　售	全国各地新华书店和相关出版物销售网点
排　版	北京万水电子信息有限公司
印　刷	三河市鑫金马印装有限公司
规　格	184mm×260mm　16 开本　10 印张　242 千字
版　次	2018 年 9 月第 1 版　2018 年 9 月第 1 次印刷
印　数	0001—3000 册
定　价	28.00 元

前　言

现在是大数据时代，我们正以前所未有的速度和规模产生数据。数据资产正在成为与土地、资本、人力并驾齐驱的关键生产要素，并在社会、经济、科学研究等方面颠覆人们探索世界的方法、驱动产业间的融合与分立。大数据是用来描述数据规模巨大、数据类型复杂的数据集，它本身蕴含着丰富的价值。对这些数据的分析处理促进了许多优秀的海量数据分析平台的产生，Hadoop 平台就是当前最为主流的一款。

Hive 是 Hadoop 生态系统中必不可少的一个工具，它提供了一种 SQL 语言，可以查询存储在 HDFS 中的数据或者其他 Hadoop 支持的文件系统，如 MapR-FS、Amazon S3、HBase 和 Cassandra。Hive 降低了应用程序迁移到 Hadoop 集群的复杂度，掌握 SQL 语句的开发人员可以轻松地学习并使用 Hive。

本书共分 10 章，其中不仅有详细的理论讲解，还有大量的实战操作。具体内容如下：

第 1 章首先介绍了 Hive 的基本工作原理及 HiveQL 语句在 Hive 中执行的具体流程；其次介绍了 Hive 中的数据类型，主要包括原子数据类型和复杂数据类型；最后给出了 Hive 的设计特点。

第 2 章详细介绍了 Hive 的基本架构，主要包括 Hive 的相关用户接口、Hive 元数据库中的表结构和三种存储方式、Hive 数据存储中的相关概念、Hive 中文件格式的不同特性和区别。

第 3 章讲解了 HiveQL 的相关表操作。

第 4 章描述了 HiveQL 的相关数据操作，主要包括数据的导入和导出。

第 5 章讲解了 HiveQL 的查询语句中的不同语法和使用方式。

第 6 章讲解了 Hive 的完整安装过程。在此基础上给出 Hive 的不同访问方式，并基于 Hive CLI 方式给出相关操作的介绍，同时给出 Hive 数据定义的相关操作。

第 7 章介绍了 Hive 的自定义函数，给出了 UDF、UDTF、UDAF 各自的函数实现方式，并给出了具体的实现源码。

第 8～10 章给出了 Hive 的相关综合案例，将之前章节的内容通过实际案例串联起来，达到最终应用的目的。

本书的编写得到北京百知教育科技有限公司的大力支持，在此表示感谢。

由于时间仓促，加之编者水平有限，本书难免存在不足之处，恳请读者对本书提出宝贵的意见和建议。

编　者

2018 年 5 月

目　　录

第 1 章　Hive 介绍

Hive 是基于 Hadoop 的一个数据仓库工具。它可以将结构化的数据文件映射为一张数据库表，并提供完整的 SQL 查询功能，可以将 SQL 语句转换为 MapReduce 任务进行运行。其优点是学习成本低，可以通过类 SQL 语句快速实现简单的 MapReduce 统计，不必开发专门的 MapReduce 应用，十分适合数据仓库的统计分析。同时，Hive 也是建立在 Hadoop 上的数据仓库基础构架。它提供了一系列的工具用来进行数据提取、转化、加载（ETL），并且提供了存储、查询和分析 Hadoop 中的大规模数据的机制。Hive 定义了简单的类 SQL 查询语言，称为 HiveQL，它允许熟悉 SQL 的用户查询数据。这个语言也允许熟悉 MapReduce 的开发者设计自定义的 Mapper 和 Reducer 来处理内建的 Mapper 和 Reducer 无法完成的复杂的分析工作。

Hive 的命令行接口和关系数据库的命令行接口类似。但是 Hive 和关系数据库还是有很大的不同，主要体现在以下几点：

（1）Hive 和关系数据库存储文件的系统不同。Hive 使用的是 Hadoop 的 HDFS，关系数据库使用的则是服务器本地的文件系统。

（2）Hive 使用的计算模型是 MapReduce，而关系数据库使用的则是自己设计的计算模型。

（3）关系数据库都是为实时查询的业务进行设计的，而 Hive 则是为海量数据做数据挖掘设计的。Hive 的实时性很差，实时性的差别导致 Hive 的应用场景和关系数据库有很大的不同。

（4）Hive 很容易扩展自己的存储能力和计算能力，这是继承了 Hadoop 的特性，而关系数据库在这方面要比 Hive 逊色很多。

1.1　Hive 工作原理

Hive 的工作原理简单来说就是一个查询引擎。当 Hive 接收到一条 SQL 语句后会执行如下的操作：

（1）词法分析和语法分析。使用 antlr 将 SQL 语句解析成抽象语法树。

（2）语义分析。从 MetaStore 中获取元数据信息，验证 SQL 语句中的表名、列名、数据类型。

（3）逻辑计划生成。生成逻辑计划得到算子树。

（4）逻辑计划优化。对算子树进行优化，包括列剪枝、分区剪枝、谓词下推等。

（5）物理计划生成。将逻辑计划生产出包含由 MapReduce 任务组成的 DAG 的物理计划。

（6）物理计划执行。将 DAG 发送到 Hadoop 集群进行执行。

（7）将查询结果返回。

Hive 的工作流程图如图 1-1 所示。

如图 1-1 所示，HiveQL 通过 Hive CLI、Thrift、jdbc 服务接口提交，经过 Hive 的 SQL 解析引擎解析，并且结合 MetaStore 中的元数据进行类型检测和语法分析，生成一个逻辑方案，同时进行内部的简单的优化处理，产生一个以有向无环图 DAG 数据结构形式展现的 MapReduce 任务，其中涉及的组件有：

- 元存储（MetaStore）。该组件存储了 Hive 中表的描述信息，其中包含表、表的分区、模式、列及其类型、表数据映射关系等。通常在实际的应用中会考虑将 MetaStore 中的数据存储到 RDBMS，比如 MySQL。
- 驱动（Driver）。控制 HiveQL 生命周期的组件，当 HiveQL 查询提交到 Hive 时，该驱动管理着会话句柄以及任何会话的统计。
- 查询编译器（Query Compiler）。该组件将 HiveQL 编译成有向无环图形式的 MapReduce 任务。
- 执行引擎（Execution Engine）。该组件按照依赖性顺序执行由编译器产生的任务。
- Hive 服务器（Hive Server）。目前该组件提供 Thrift、JDBC 远程语句接口等。
- 客户端组件。提供命令行接口 Hive CLI、Web UI 以及 JDBC 驱动。

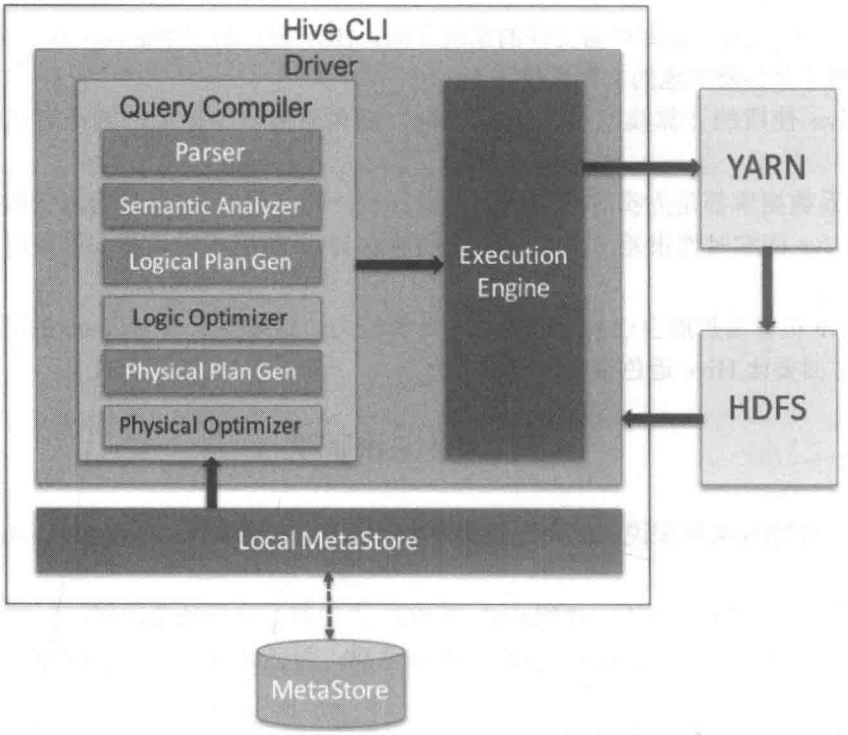


图 1-1　Hive 的工作流程图

1.2　Hive 的数据类型

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，具体见表 1-1。

Hive 是用 Java 开发的，除了 STRING 类型，Hive 中的基本数据类型和 Java 的基本数据类型也是一一对应的。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 的基本数据类型 float 和 double；Hive 的 BOOLEAN 数据类型相当于 Java 的基本数据类型 boolean。

表 1-1 Hive 原子数据类型

基本数据类型		
类型	描述	示例
TINYINT	1 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	'hive'或者"hive"

Hive 的 STRING 数据类型相当于数据库的 VARCHAR 数据类型。该类型是一个可变的字符串，它不能声明其中最多能存储多少个字符，理论上它可以存储长度为 2GB 的字符串。

Hive 支持基本数据类型的转换，占用字节少的基本数据类型可以转化为占用字节多的数据类型。例如 TINYINT、SMALLINT、INT 可以转化为 FLOAT，而所有的整数类型、FLOAT 以及 STRING 类型可以转化为 DOUBLE 类型。这些转化可以从 Java 语言的类型转化考虑，因为 Hive 就是用 Java 语言编写的。当然也支持占用字节多的数据类型转化为占用字节少的数据类型，这就需要使用 Hive 的自定义函数 CAST 了。

Hive 的复杂数据类型包括数组（ARRAY）、映射（MAP）和结构体（STRUCT），具体见表 1-2。

表 1-2 Hive 复杂数据类型

类型	描述	示例
ARRAY	一组有序字段。字段的类型必须相同	ARRAY(1,2)
MAP	一组无序的键值对。键的类型必须是原子的，值可以是任何类型，同一个映射的键的类型必须相同，值的类型也必须相同	MAP('a',1,'b',2)
STRUCT	一组命名的字段。字段类型可以不同	STRUCT('a',1,1,0)

下面给出使用复杂数据类型建表的实例。

```
Create table complex(Col1 ARRAY<INT>,
Col2 MAP<STRING,INT>,
Col3 STRUCT<a:STRING,b:INT,c:DOUBLE>);
```

对应此表的查询语句：

```
Select Col1[0],Col2['b'],Col3.c
from complex;
```

1.3 Hive 的特点

Hive 是一种底层封装了 Hadoop 的数据仓库处理工具，使用类 SQL 的 HiveQL 语言实现数据查询，所有 Hive 的数据都存储在 Hadoop 兼容的文件系统(例如 Amazon S3、HDFS)中。Hive 在加载数据过程中不会对数据进行任何的修改，只是将数据移动到 HDFS 中 Hive 设定的目录下。因此 Hive 不支持对数据的改写和添加，所有的数据都是在加载的时候确定的。Hive 的设计特点如下：

（1）支持索引，加快数据查询。

（2）支持不同的文件存储类型，例如，纯文本文件、HBase 中的文件。

（3）将元数据保存在关系数据库中，大大减少了在查询过程中执行语义检查的时间。

（4）可以直接使用存储在 Hadoop 文件系统中的数据。

（5）内置大量用户函数 UDF 来操作时间、字符串和其他的数据挖掘工具；支持用户扩展 UDF 函数来完成内置函数无法实现的操作。

（6)类 SQL 的查询方式，将 SQL 查询转换为 MapReduce 的 Job 在 Hadoop 集群上执行。

1.4 本章小结

本章首先介绍了 Hive 的基本工作原理，HiveQL 语句在 Hive 中执行的具体流程；其次介绍了 Hive 中的数据类型，主要包括原子数据类型和复杂数据类型；最后给出了 Hive 的设计特点。

第 2 章　Hive 架构

Hive 是为了简化用户编写 MapReduce 程序而生成的一种框架。在 Hive 架构中主要包括 Hive 用户接口、Hive 元数据库等。本章将给出 Hive 架构的详细介绍。

2.1　Hive 用户接口

Hive 提供了以下三种客户端用户访问接口。

（1）Hive CLI（Hive Command Line，Hive 命令行）。客户端可以直接在命令行模式下进行操作。通过命令行，用户可以定义表、执行查询等。如果没有指定其他服务，这个就是默认的服务。

（2）HWI（Hive Web Interface，Hive Web 接口）。Hive 提供了更直观的 Web 界面，可以执行查询语句和其他命令，这样可以不用登录到集群中的某台机器上使用 CLI 来进行查询。

（3）Hive 提供了 Thrift 服务，即 Hiveserver。它是监听来自于其他进程的 Thrift 连接的一个守护进程。Thrift 客户端目前支持 C++/Java/PHP/Python/Ruby 语言。

2.1.1　Hive CLI

Hive CLI 提供了执行 HiveQL、设置参数等功能。要启用 CLI 只需要在命令行下执行 $HIVE_HOME/bin/hive 命令。在命令下执行 hive -H 可以查看 CLI 的参数选项，如图 2-1 所示。

```
-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)
```

图 2-1　Hive CLI 的参数选项

其中 Hive CLI 每一个对应的参数选项的具体解释见表 2-1。

表 2-1　Hive CLI 参数选项详解

参数选项	说明
-d,--define <key=value>	应用于 Hive 命令的变量替换，如-d　A=B 或者--define A=B
--database <databasename>	指定所使用的数据库

续表

参数选项	说明
-e <quoted-query-string>	执行命令行指定的 SQL
-f <filename>	执行文件中的 SQL
-H,--help	打印帮助信息
-h <hostname>	连接远程主机上的 Hive 服务器

下面介绍几个常用的 Hive 命令行操作实例。

（1）执行一个查询：

```
$HIVE_HOME/bin/hive -e 'select a.col from a'
```

命令执行过程中会在终端上显示 MapReduce 的进度。执行完毕后，把查询结果输出到终端上，接着 Hive 进程退出，不会进入交互模式。

（2）静音模式执行一个查询：

```
$HIVE_HOME/bin/hive -S -e 'select a.col from a'
```

命令中加入-S 则终端上的输出不会有 MapReduce 的进度。执行完毕只会把查询结果输出到终端上。这个静音模式很实用，通过第三方程序调用，第三方程序通过 Hive 的标准输出获取结果集。

（3）静音模式执行一个查询把结果集导出：

```
$HIVE_HOME/bin/hive -S -e 'select a.col from a' > a.csv
```

（4）不进入交互模式执行一个 Hive Script：

```
$HIVE_HOME/bin/hive -f /home/hive/hive-script.sql
```

hive-script.sql 是使用 HiveSQL 语法编写的脚本文件，执行的过程和用-e 参数选项类似，区别是从文件加载 SQL。但是 HiveSQL 文件对于 bash 来说是不能使用变量，而使用-e 的方式，可以在 bash 里使用变量。这里也可以和静音模式-S 联合使用，通过第三方程序调用，第三方程序通过 Hive 的标准输出获取结果集。

上述实例中都是在终端直接执行 Hive CLI 命令行操作，并没有进入 Hive 交互式 Shell 模式。当执行$HIVE_HOME/bin/hive 时，没有-e 或者-f 选项则会进入交互式 Shell 模式。

2.1.2 HWI

HWI 是 Hive CLI 命令行接口的一个 Web 替换方案。HWI 的特点是相对于命令行方式界面友好，适合不太熟悉 Linux 命令行操作方式的人员。

1. *配置和启动* HWI

这里以 Hive 的 1.2.1 版本为例。HWI 的运行需要依赖两个包：hive-hwi-1.2.1.jar 和 hive-hwi-1.2.1.war，这两个包应该都部署在$HIVE_HOME/lib 目录下。但是在 apache-hive-1.2.1-bin.tar.gz 的安装包 lib 目录下没有提供 war 包，解决方法是下载对应版本的 Hive 源码。进入到源码包的/hwi/web/目录下，将该目录下的文件夹和文件压缩成 war 包，并且命名为 hive-hwi-1.2.1.war，放到$HIVE_HOME/lib 目录下即可。

配置$HIVE_HOME/conf/hive-site.xml，如图 2-2 所示。

```
<property>
  <name>hive.hwi.listen.host</name>
  <value>0.0.0.0</value>
  <description>This is the host address the Hive Web Interface will listen on</description>
</property>
<property>
  <name>hive.hwi.listen.port</name>
  <value>9999</value>
  <description>This is the port the Hive Web Interface will listen on</description>
</property>
<property>
  <name>hive.hwi.war.file</name>
  <value>lib/hive-hwi-1.2.1.war</value>
  <description>This sets the path to the HWI war file, relative to ${HIVE_HOME}. </description>
</property>
```

图 2-2 hive-site.xml 中关于 HWI 的配置内容

启动 HWI 服务，$HIVE_HOME/bin/hive -service hwi，输出如图 2-3 所示。

```
root@master:~/apache-hive-1.2.1-bin/bin# ./hive --service hwi
18/01/26 16:14:59 INFO hwi.HWIServer: HWI is starting up
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.min.split.size is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.reduce.tasks.speculative.execution is deprecated. Instead, use mapreduce.reduce.speculative
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.min.split.size.per.node is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize.pe
r.node
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.input.dir.recursive is deprecated. Instead, use mapreduce.input.fileinputformat.input.dir.recursive
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.min.split.size.per.rack is deprecated. Instead, use mapreduce.input.fileinputformat.split.minsize.pe
r.rack
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.max.split.size is deprecated. Instead, use mapreduce.input.fileinputformat.split.maxsize
18/01/26 16:14:59 INFO Configuration.deprecation: mapred.committer.job.setup.cleanup.needed is deprecated. Instead, use mapreduce.job.committer.setup.cleanup
.needed
18/01/26 16:15:00 INFO mortbay.log: Logging to org.slf4j.impl.Log4jLoggerAdapter(org.mortbay.log) via org.mortbay.log.Slf4jLog
18/01/26 16:15:00 INFO mortbay.log: jetty-6.1.26
18/01/26 16:15:00 INFO mortbay.log: Extract /root/apache-hive-1.2.1-bin/lib/hive-hwi-1.2.1.war to /tmp/Jetty_0_0_0_0_9999_hive.hwi.1.2.1.war__hwi__vrqy3t/web
app
18/01/26 16:15:00 INFO mortbay.log: Started SocketConnector@0.0.0.0:9999
```

图 2-3 HWI 服务启动过程

2. HWI 的使用

（1）用户认证信息

使用浏览器打开http://Hive 安装包所在服务器的 IP:9999/hwi/，首先会提示填入用户的信息，即用户名和用户组，填入之后单击 Submit 会提示认证完成，如图 2-4 所示。

图 2-4 HWI 的用户认证

（2）创建会话

单击左侧的 Create Session（创建会话）项，创建一个 Hive 的会话（Session）。填入会话名之后单击 Submit 进入会话管理页面，如图 2-5 所示。

（3）执行查询

进入会话管理页面后，在 Result File 项中填入结果保存文件（注意：这个文件必须存在）；在 Query 项中填入要执行的 HiveQL 语句；在 Start Query 下拉列表中选择 NO；单击 Submit 开始执行 HiveQL 语句，如图 2-6 所示。

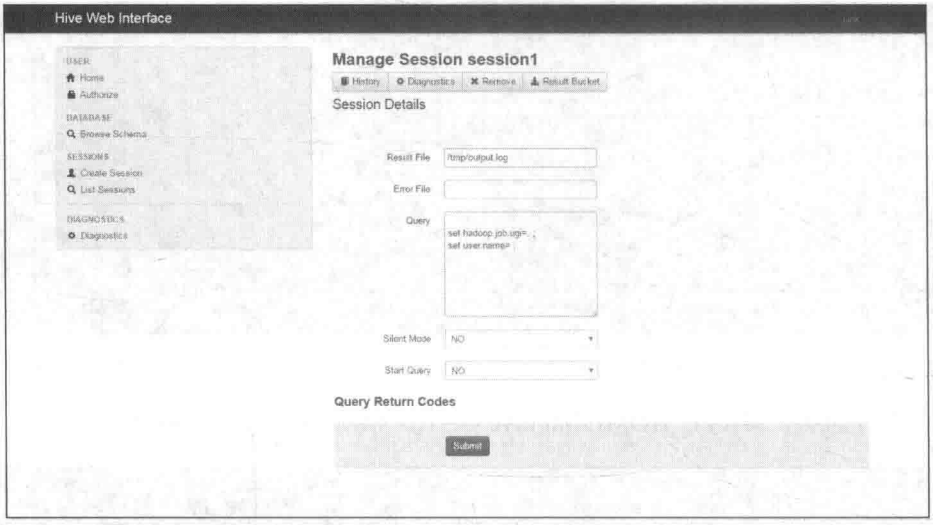


图 2-5　HWI 的会话管理

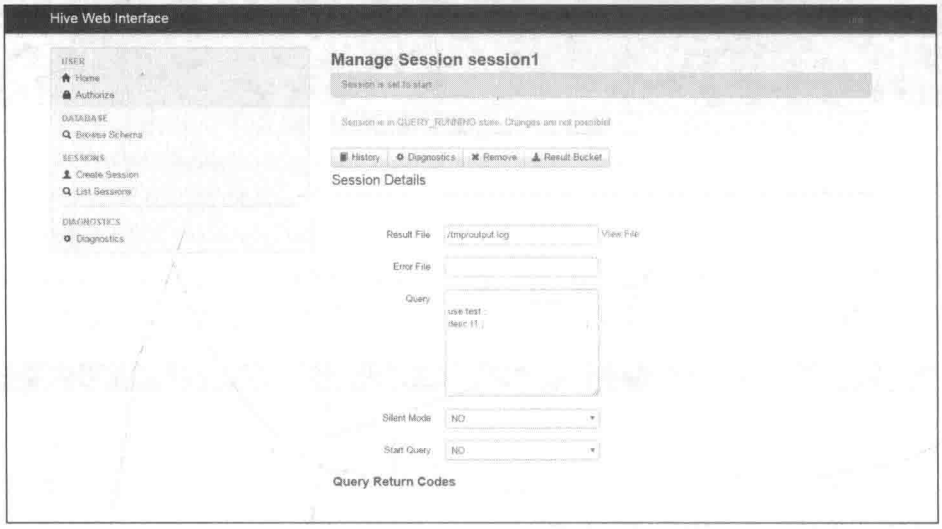


图 2-6　在 HWI 中执行 HiveQL 语句

单击图 2-6 中左侧的 List Sessions 项会显示每个 Session（会话）的当前状态，如图 2-7 所示。

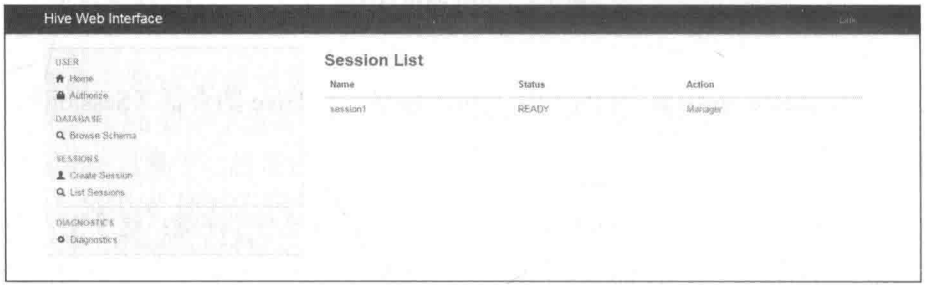


图 2-7　HWI 中的会话状态

当 Status 为 READY 时，表示前面的查询已经执行完。单击 Manager 进入会话管理页面，再单击 Result File 后面的 View File 项查看执行结果，如图 2-8 所示。

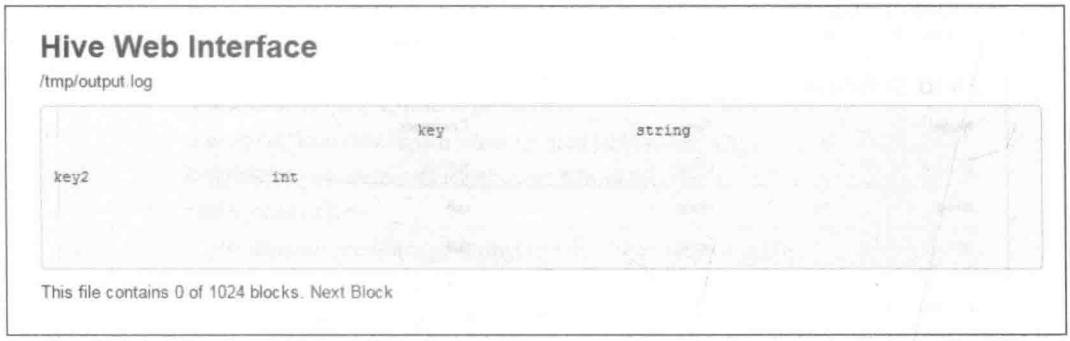


图 2-8 HWI 中的会话执行的结果

（4）浏览数据库

单击会话管理页面左侧的 Browse Schema 项可浏览 Hive 中所有的数据库，如图 2-9 所示。

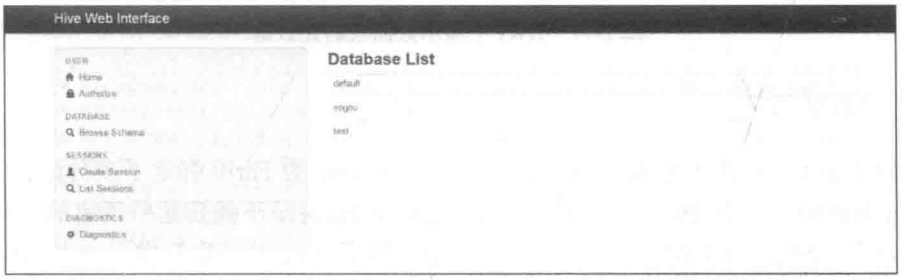


图 2-9 HWI 中显示 Hive 所有的数据库

单击一个数据库可浏览该数据库所有的表，如图 2-10 所示。

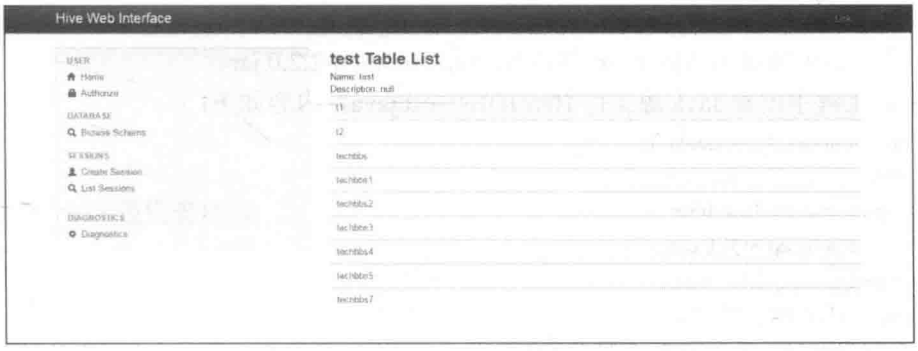
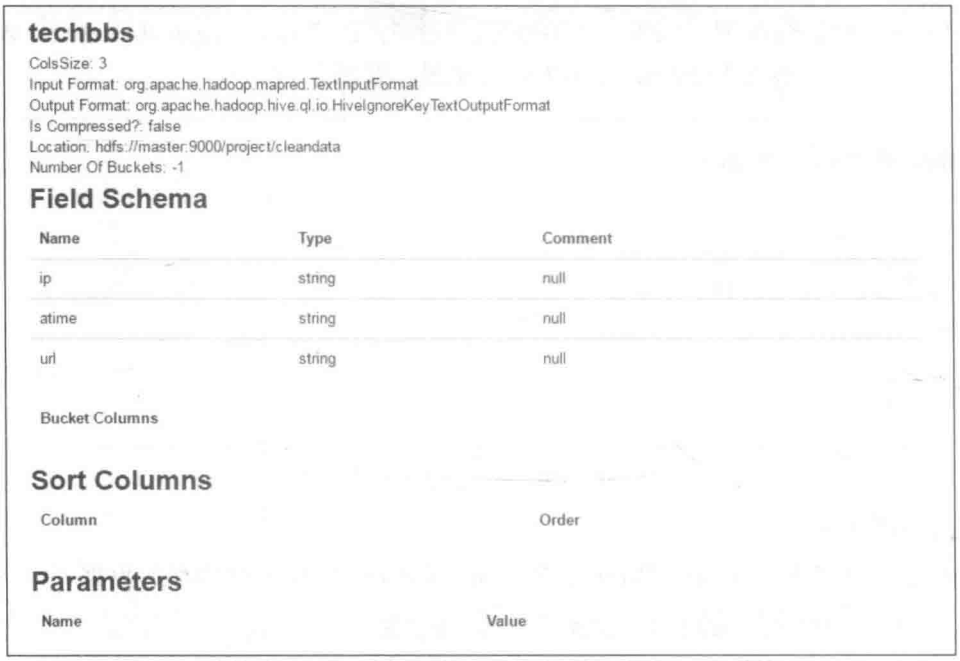


图 2-10 HWI 中显示数据库的表

单击某一个表可查看该表的元数据，如图 2-11 所示。


techbbs

ColsSize: 3
Input Format: org.apache.hadoop.mapred.TextInputFormat
Output Format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Is Compressed?: false
Location: hdfs://master:9000/project/cleandata
Number Of Buckets: -1

Field Schema

Name	Type	Comment
ip	string	null
atime	string	null
url	string	null

Bucket Columns

Sort Columns

Column	Order

Parameters

Name	Value

图 2-11　HWI 中显示数据表的元数据

2.1.3　Thrift 服务

Hive 以 Thrift 方式作为服务对客户端提供，目前 Hive 的 Thrift 绑定了多种语言（C++/Java/PHP/Python/Ruby），可在 Hive 发行版本的 src/service/src 目录下找到这些语言的 Thrift 绑定。Hive 还提供了 JDBC 和 ODBC 的驱动，大大方便了基于 Hive 的应用开发。本书利用官方的例子对 JDBC 驱动进行了测试。

（1）启动 hiveserver：hive -service hiveserver2。

（2）Eclipse 中新建一个 Java 工程 HiveJDBC。

（3）将工程依赖的 jar 包加入到工程的 buildpath 里。jar 包括 Hive 目录里面的 lib 下的 jar 包，还有目录 share/hadoop/common 下的 hadoop-common-2.2.0.jar。

（4）在工程中创建 Java 源文件 HiveJDBCTest.java，内容如下：

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
public class HiveJDBCTest {
    private static Connection conn = null;
    public static void main(String args[]) {
        try {
            Class.forName("org.apache.hive.jdbc.HiveDriver");
            conn = DriverManager.getConnection(
                    "jdbc:hive2://192.168.51.246:10000/test", "", "");
            Statement st = conn.createStatement();
```

```
            String tableName = "hivetest";
            st.execute("drop table if exists " + tableName);
            st.execute("create table "
                    + tableName
                    + "(key int, value string) row format delimited fields terminated by '\t'");
            System.out.println("Create table success!");
            st.execute("load data inpath '/data.txt' into table hivetest");
            ResultSet rs = st.executeQuery("select * from hivetest");
            while (rs.next()) {
                System.out.println(rs.getString(1) + "    " + rs.getString(2));
            }
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

最后程序运行结果如图 2-12 所示。

```
log4j:WARN No appenders could be found for logger (org.apache.hive.jdbc.Utils).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Create table success!
1  zhangsan
2  lisi
3  wangwu
```

图 2-12　程序运行结果截图

2.2　Hive 元数据库

2.2.1　Hive 元数据表结构

在使用Hive进行开发时，往往需要获得一个已存在 Hive 表的建表语句（Data Definition Language，DDL），然而 Hive 本身并没有提供这样一个工具。要想还原建表 DDL 就必须从元数据入手。我们知道，Hive 的元数据并不存放在 HDFS 上，而是存放在传统的 RDBMS 中，典型的如MySQL、Derby 等。这里我们以MySQL为元数据库，结合 1.2.1 版本的 Hive 进行介绍。

连接上MySQL后可以看到 Hive 元数据对应的表大概有 20 个，其中和表结构信息有关的有 9 个，其余的 10 多个或为空、或只有简单的几条记录。部分主要表的简要说明见表 2-2。