# 第 5 章

# Hive

Hive 是一个基于 Hadoop 的开源数据仓库工具，旨在提供数据查询和分析的能力。它采用类似于 SQL 的查询语言，称为 HiveQL(HQL)，使非技术人员能够以类似于关系数据库的方式查询和分析大规模数据。Hive 的核心思想是将数据存储在 Hadoop 分布式文件系统(HDFS)中，并使用 Hive 提供的元数据管理和查询引擎来执行查询操作。Hive 将查询转换为一系列 MapReduce 任务或更高级的计算引擎(如 Apache Tez)，以实现高效的数据处理。Hive 提供了丰富的数据模型和查询功能，并支持表和分区的概念，可以将数据组织为表，并利用分区进行数据分割和优化。

## 5.1 任务目的和要求

(1) 掌握 Hive 的安装和部署方法。
(2) 掌握 Hive 的数据库管理方法。
(3) 掌握 Hive 的基本表、视图、索引以及查询方法。
(4) 掌握 Hive 的编程方法。

## 5.2 Hive 的安装

### 5.2.1 Hive 的配置

安装 Hive 前需要安装 Hadoop，方法可参考 Hadoop 的安装教程。

(1) 下载并解压 Hive 源程序，本实验以 Ubuntu18 作为示例，将 hive-1.2.1 安装包 apache-hive-1.2.1-bin.tar.gz 下载至/opt/software：

```
nbu@ecs:/opt/software$ sudo tar -zxvf \
./apache-hive-1.2.1-bin.tar.gz -C /usr/local
#解压到/usr/local 中
nbu@ecs:/opt/software$ cd /usr/local/
nbu@ecs:/usr/local$ sudo mv apache-hive-1.2.1-bin hive
#将文件夹名改为 hive
```

```
nbu@ecs:/usr/local$ sudo chown -R nbu:nbu hive
#修改文件权限
```

(2) 配置环境变量。为了方便使用,可以把 hive 命令加入环境变量中,编辑~/.bashrc 文件 vim ~/.bashrc,在最前面一行添加:

```
export HIVE_HOME=/usr/local/hive
export PATH=$PATH:$HIVE_HOME/bin
export CLASSPATH=$CLASSPATH:/usr/local/hive/lib/*
```

保存退出后,运行 source ~/.bashrc 使配置立即生效。

(3) 修改/usr/local/hive/conf 下的 hive-site.xml:

```
nbu@ecs:/usr/local/hive$ cd conf/
#将 hive-default.xml.template 重命名为 hive-default.xml
nbu@ecs:/usr/local/hive/conf$ cp hive-default.xml.template hive-default.xml
#新建一个文件 touch hive-site.xml
nbu@ecs:/usr/local/hive/conf$ touch hive-site.xml
```

在 hive-site.xml 中粘贴如下配置信息:

```
<?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&amp;useUnicode=true&amp;characterEncoding=utf8&amp;useSSL=false</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>
```

### 5.2.2 MySQL 的安装

(1) Ubuntu 下 MySQL 的安装,先将 mysql-connector-java-5.1.40.tar.gz 下载至/opt/software/:

```
nbu@ecs:~$ cd /opt/software
nbu@ecs:/opt/software$ tar -zxvf mysql-connector-java-5.1.40.tar.gz  #解压
#将 mysql-connector-java-5.1.40-bin.jar 复制到/usr/local/hive/lib 目录下
nbu@ecs:/opt/software$ cp \
mysql-connector-java-5.1.40/mysql-connector-java-5.1.40-bin.jar \
/usr/local/hive/lib
```

(2) 启动并登录 MySQL Shell:

```
nbu@ecs:/opt/software$ sudo service mysql start  #启动 MySQL 服务
nbu@ecs:/opt/software$ mysql -u root -p          #登录 Shell 界面
```

(3) 新建 Hive 数据库:

```
mysql> create database hive;
Query OK, 1 row affected (0.00 sec)
#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
```

(4) 配置 MySQL 允许 Hive 接入,建立账号名为 hive,密码为 hive 的账号。

```
mysql> grant all on *.* to hive@localhost identified by 'hive';
Query OK, 0 rows affected, 1 warning (0.00 sec)
#刷新 mysql 系统权限关系表
mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)
mysql> exit
Bye
```

**注意**:若出现 The MySQL server is running with the --skip-grant-tables option so it cannot execute this statement 问题,需要执行 flush privileges 刷新权限表。

(5) 启动 Hive,启动 Hive 之前,请先启动 Hadoop 集群。

```
#启动 Hadoop
nbu@ecs:~$ /usr/local/hadoop/sbin/start-all.sh
#启动 Hive
nbu@ecs:~$ /usr/local/hive/bin/hive
#开启 metastore
nbu@ecs:~$ /usr/local/hive/bin/hive --service metastore &
#开启 hiveserver2
nbu@ecs:~$ /usr/local/hive/bin/hive --service hiveserver2 &
```

若在 Hive 启动时出现 Hive metastore database is not initialized 的错误,出错原因是重

新安装 Hive 和 MySQL，导致版本、配置不一致。在终端执行如下命令：

```
$ schematool -dbType mysql -initSchema
```

### 5.2.3 MRS 平台使用 Hive

本实验以华为 MRS 官方示例“Hive 加载 HDFS 数据并分析图书评分情况”(https://support.huaweicloud.com/bestpractice-mrs/mrs_05_0021.html)的数据 bookstore.txt 作为分析数据，将 bookstore.txt 上传至 OBS，并从 OBS 上传数据文件到 HDFS。bookstore.txt 的内容如下：

```
202001,242,3,Good!
202002,302,3,Test.
202003,377,1,Bad!
220204,51,2,Bad!
202005,346,1,aaa
202006,474,4,None
202007,265,2,Bad!
202008,465,5,Good!
202009,451,3,Bad!
202010,86,3,Bad!
202011,257,2,Bad!
202012,465,4,Good!
202013,465,4,Good!
202014,465,4,Good!
202015,302,5,Good!
202016,302,3,Good!
......
```

**1. 创建 OBS 服务**

对象存储服务(Object Storage Service，OBS)是一个基于对象的海量存储服务，为客户提供数据存储能力，OBS 系统和单个桶都没有总数据容量和对象/文件数量的限制，适合存放任意类型的文件，适合普通用户、网站、企业和开发者使用。华为官网 OBS 服务地址为 https://www.huaweicloud.com/product/obs.html。

首先需要购买 OBS 资源包，配置如图 5-1 所示。

创建 obs-nbu 桶，如图 5-2 所示，用于存放 bookstore.txt 文件。

上传 bookstore.txt 文件对象到 obs-nbu，选择桶名称，选择“对象”→“上传对象”，如图 5-3 所示，将数据文件上传至 obs-nbu 桶内。

切换回“MRS 控制台”，如图 5-4 所示，选择创建好的 MRS 集群名称，进入“概览”，选择“IAM 用户同步”所在行的“同步”，等待几分钟即可同步完成。

将 bookstore.txt 文件上传 HDFS，返回 MRS 控制台的管理界面，在“文件管理”页签，选择“HDFS 文件列表”，创建数据存储目录如“/tmp”。

**注意**：“/tmp”目录仅为示例，可以是界面上的任何目录，也可以通过“新建”创建新的文件夹。


图 5-1 OBS 资源配置界面


图 5-2 创建 obs-nbu 桶


图 5-3 向 obs 上传对象


图 5-4 MRS 集群运维管理

选择“导入数据”，选择从 OBS 导入，如图 5-5 所示，选择创建好的 obs_nbu 桶，找到“book_score.txt”文件，勾选“我确认所选脚本安全，了解可能存在的风险，并接受对集群可能造成的异常或影响。”复选框，单击“确定”按钮。


图 5-5 选择 OBS 文件

等待数据导入成功，此时数据文件已上传至 MRS 集群的 HDFS 文件系统内。

**2. 创建 Hive 表**

使用 MRS 的 Hive 客户端，可参考 MRS 安装教程安装 MRS 集群客户端。进入客户端所在目录并加载变量：

```
[root@nbu-node-master1SdHR client]#cd /opt/client
[root@nbu-node-master1SdHR client]#source bigdata_env
```

执行 beeline -n 'hdfs'命令进入 Hive Beeline 命令行界面。

执行以下命令创建一个与原始数据字段匹配的 Hive 表：

```
create table bookscore (userid int,bookid int,score int,remarks string) row
format delimited fields terminated by ','stored as textfile;
```

查看表是否创建成功：

```
show tables;
```

**3. 将原始数据导入 Hive 并进行分析**

继续在 Hive Beeline 命令行中执行以下命令，将已导入 HDFS 的原始数据导入 Hive 表中：

```
load data inpath '/user/tmp/book_score.txt' into table bookscore;
```

数据导入完成后，执行如下命令，查看 Hive 表的内容：

```
select * from bookscore;
```

执行以下命令统计表行数：

```
select count(*) from bookscore;
```

**注意**：由于 count(*)函数需要通过 mapreduce 实现，可以看到该查询相对上一查询耗时较多。

执行以下命令，筛选原始数据中累计评分最高的图书 top3：

```
select bookid,sum(score) as sumscore from bookscore group by bookid order by
sumscore desc limit 3;
```

以上内容表示，ID 为 456、302、474 的 3 本书籍，为累计评分最高的 Top3 图书。

## 5.3 Hive 操作案例

### 5.3.1 数据库管理

**1. 数据库的创建**

在 Hive 中利用命令语句创建一个数据库，所创建的数据库是一个命令空间或表的集

合，语法如下：

```
CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database name>
```

其中，IF NOT EXISTS 是一个可选子句，判断是否存在相同名称的数据库。可采用下面的查询执行创建一个名为 nbubd 数据库：

```
hive> CREATE DATABASE [IF NOT EXISTS] nbudb;
```

下面的查询用于验证数据库列表：

```
hive> SHOW DATABASES;
OK
default
nbudb
Time taken: 0.013 seconds, Fetched: 2 row(s)
```

**2. 数据库的删除**

DROP DATABASE 是删除所有的表并删除数据库的语句，语法如下：

```
DROP DATABASE StatementDROP (DATABASE | SCHEMA) [IF EXISTS] database_name
[RESTRICT|CASCADE];
```

下面的查询用于删除数据库 userdb：

```
hive> DROP DATABASE IF EXISTS nbudb;
OK
Time taken: 0.135 seconds
```

**3. 数据导入导出**

这里将 stu.txt 作为实验样例数据，存储在/home/nbu/路径下，stu.txt 的具体内容如下：

```
student_id|name|gender|birth_date|class|university
1|李梓涵|男|Sep-98|通信工程(2)班|nbu
2|刘芳|女|Nov-00|大数据(2)班|nbu
3|罗雨婷|男|Dec-00|大数据(2)班|nbu
4|董俊杰|男|Nov-98|大数据(2)班|nbu
5|乃宇轩|男|Sep-99|通信工程(1)班|nbu
6|刘思琪|女|Jun-98|网络安全(3)班|nbu
7|林芳|男|Mar-98|通信工程(3)班|nbu
8|董思琪|男|1-Sep|网络安全(4)班|nbu
……
```

使用 load 语句导入：

```
#选择数据库
use nbu_db;
```

```
#创建表
CREATE TABLE IF NOT EXISTS students (  #external 关键字表示外部表
  student_id INT,
  name STRING,
  gender INT,
birth_date STRING,
class STRING,
university STRING
)
COMMENT 'Student Info'                          #表信息备注
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|';                       #自定义数据分隔符
```

加载数据：

```
#从 Linux 本地加载数据文件到 hive 表中
load data local inpath '/home/nbu/stu.txt' [overwrtie] into table nbu_db.
students;
#从 HDFS 上加载数据文件到 hive 表中(注意,基于 HDFS 进行加载会导致源文件消失,本质是将数
#据文件移动到表所在的目录)
load data inpath '/tmp/exp' [overwrtie] into table nbu_db.students; #overwrtie
#表示对已有数据进行覆盖,否则为追加数据
```

再次加载数据：

```
load data inpath '/tmp/exp' into table nbu_db.students1;
```

使用 insert 语句导入,其中 overwrite 表示对已有数据进行覆盖,否则为追加数据：

```
INSERT [overwrite] INTO nbu_db.students SELECT * FROM nbu_db.studentss;
```

INSERT OVERWRITE 方式数据导出语法如下：

```
INSERT OVERWRITE [LOCAL] DIRECTORY 'path'
-- LOCAL 可选,带 LOCAL 表示导出到 Linux 本地,不带 LOCAL 表示导出到 HDFS
    [ROW FORMAT DELIMITED FIELDS TERMINATED BY '']
-- 可选,自定义列分隔符
    SELECT ... FROM ...;
```

将查询结果导出到 HDFS,结果如图 5-6 所示。

```
0: jdbc:hive2://localhost:10000> insert overwrite directory '/tmp/export' row format delimited fields terminated by
'|' select * from students;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a
different execution engine(i.e.spark,tez) or using Hive 1.X releases.
No rows affected (6.944 seconds)
```

**图 5-6 将查询结果导出到 HDFS**

将查询结果导出到本地/home/export 路径下,结果如图 5-7 所示。

```
0: jdbc:hive2://localhost:10000> insert overwrite local directory '/home/export' row format delimited fields
terminated by '|' select * from students;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a
different execution engine(i.e.spark,tez) or using Hive 1.X releases.
No rows affected (7.48 seconds)
```

图 5-7　将查询结果导出到本地

### 5.3.2　查询语言

Hive 查询语言(HiveQL)是一种查询语言,用于处理和分析结构化数据。

**1. SELECT 语句**

SELECT 语句用来从表中检索的数据,其中 WHERE 子句是对数据进行条件过滤,使用这个条件过滤数据,并返回给出一个有限的结果。内置运算符和函数产生一个表达式,满足以下条件。

下面是 SELECT 查询的语法:

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...
FROM table_reference
[WHERE where_condition]
[GROUP BY col_list]
[HAVING having_condition]
[CLUSTER BY col_list | [DISTRIBUTE BY col_list] [SORT BY col_list]]
[LIMIT number];
```

假设有一个学生绩点表 students_score,有 student_id、name、class、gpa 等字段,生成一个查询检索 gpa 超过(包括)3.5 的学生的详细信息。

```
1|李梓涵|通信工程(2)班|3
2|刘芳|大数据(2)班|2.2
3|罗雨婷|大数据(2)班|2.9
4|董俊杰|大数据(2)班|2.1
5|乃宇轩|通信工程(1)班|2.9
6|刘思琪|网络安全(3)班|3.9
7|林芳|通信工程(3)班|4
8|董思琪|网络安全(4)班|3.1
…

#数据库创建语句如下
CREATE TABLE IF NOT EXISTS students_score (
  student_id INT,
  name STRING,
  class STRING,
gpa INT
)
COMMENT 'Student Score'                          #表信息备注
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|';                        #自定义数据分隔符
#导入数据
```