**Chap9 Hive数据定义**

**实验步骤**

1  **启动Hadoop集群**

在进行实验之前，首先需要确保Hadoop集群已经正常启动。如果Hadoop没有启动，需要通过下列步骤启动Hadoop集群。

**在模板中，我们已经配置好了Hadoop伪分布式环境，同学们不需要再次配置，可以直接启动使用。**

**步骤1.**  **启动Hadoop**

打开一个终端模拟器，通过下列命令启动Hadoop：

start-all.sh

命令运行后的输出内容如下所示：

[root@master ~]# start-all.sh

This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh

18/11/08 14:42:12 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Starting namenodes on [master]

master: starting namenode, logging to /usr/cx/hadoop-2.7.1/logs/hadoop-root-namenode-master.out

localhost: starting datanode, logging to /usr/cx/hadoop-2.7.1/logs/hadoop-root-datanode-master.out

Starting secondary namenodes [master]

master: starting secondarynamenode, logging to /usr/cx/hadoop-2.7.1/logs/hadoop-root-secondarynamenode-master.out

18/11/08 14:42:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

starting yarn daemons

starting resourcemanager, logging to /usr/cx/hadoop-2.7.1/logs/yarn-root-resourcemanager-master.out

localhost: starting nodemanager, logging to /usr/cx/hadoop-2.7.1/logs/yarn-root-nodemanager-master.out

[root@master ~]#

由输出内容可以看到，每个进程在启动以及运行过程中，会将相关的日志信息记录到日志文件中，因此当进程出现问题时，一个有效的办法就是查看日志文件进行排错处理。

**步骤2.**  **验证Hadoop是否启动成功**

通过下列命令，查看相应的JVM进程确定Hadoop是否启动成功：

jps

当有以下5个进程启动时，则说明Hadoop已经成功启动，如果某个进程没有启动，则需要查看日志文件确定错误发生的原因：

[root@master ~]# jps

3631 NameNode                                     （注：HDFS元数据管理进程）

1679 VmServer.jar

3757 DataNode                                     （注：HDFS数据存储进程）

4488 Jps

3922 SecondaryNameNode                           （注：HDFS数据检查点进程）

4174 NodeManager                                  （注：任务执行进程）

4073 ResourceManager                             （注：资源管理进程）

[root@master ~]#

2  **进入HiveCLI命令行界面**

我们需要在HiveCLI命令行界面中进行相关的实验操作。Hive默认使用的Derby数据库进行元数据存储，而此时从哪一个目录进入HiveCLI，相应的元数据就会存储到所进入的目录下。因此在后文的实验中，我们约定在/root目录下进入HiveCLI命令行界面。

通过下列命令进入/root目录：

cd /root/

进入/root目录后，通过下列命令进入HiveCLI命令行界面：

hive

命令运行后的返回结果如下所示：

[root@master ~]# hive

Logging initialized using configuration in jar:file:/usr/cx/apache-hive-1.2.1-bin/lib/hive-common-1.2.1.jar!/hive-log4j.properties

hive>

3  **数据库操作**

Hive中数据库的概念本质上仅仅是表的一个目录或者命名空间。然而，对于具有很多组和用户的大集群来说，这是非常有用的，因为这样可以避免表命名冲突。通常会使用数据库来将生产表组织成逻辑组。

**步骤1.**  **创建数据库**

1.   在HiveCLI命令行界面中通过下列命令创建数据库financials：

CREATE DATABASE financials;

命令运行后的返回结果如下所示：

hive> CREATE DATABASE financials;

OK

Time taken: 1.131 seconds

hive>

如果我们再次运行命令创建数据库financials，由于数据库financials已经存在，因此会出现错误提示，错误信息如下所示：

hive> CREATE DATABASE financials;

FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask. Database financials already exists

hive>

2.   通过下列命令指定当数据库financials不存在时进行创建，如果已经存在则忽略：

CREATE DATABASE IF NOT EXISTS financials;

命令运行后的返回结果如下所示：

hive> CREATE DATABASE IF NOT EXISTS financials;

OK

Time taken: 0.025 seconds

hive>

3.   通过如下命令可以查看Hive中所包含的数据库信息：

SHOW DATABASES;

命令运行后的返回结果如下所示（可见上文中创建的financials数据库已经成功显示）：

hive> SHOW DATABASES;

OK

default

financials

Time taken: 0.344 seconds, Fetched: 2 row(s)

hive>

4.   可以使用正则表达式匹配来筛选出需要的数据库名。通过下面的命令列举出所有以字母fi开头，以其他字符结尾（即.\*部分含义）的数据库名：

SHOW DATABASES LIKE 'fi.\*';

命令运行后的返回结果如下所示：

hive> SHOW DATABASES LIKE 'fi.\*';

OK

financials

Time taken: 0.055 seconds, Fetched: 1 row(s)

hive>

默认情况下，当我们创建数据库financials时，Hive将会在HDFS的/user/hive/warehouse/目录下创建financials.db子目录，我们可以访问HDFS进行查看。

5.   打开一个新的终端模拟器，通过下面的命令查看/user/hive/warehouse/目录下的内容：

hadoop fs -ls /user/hive/warehouse/

命令运行后的返回结果如下所示（由返回结果可以看出，financials.db子目录已经被成功创建）：

[root@master ~]# hadoop fs -ls /user/hive/warehouse/

18/11/20 15:56:39 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Found 1 items

drwxr-xr-x   - root supergroup          0 2018-11-20 15:41 /user/hive/warehouse/financials.db

[root@master ~]#

**步骤2.**  **为数据库增加描述信息**

1.   在HiveCLI命令行界面中通过下面的命令可以在创建数据库的同时为数据库增加描述信息：

CREATE DATABASE mydatabase02 COMMENT 'This is my second database.';

命令运行后的返回结果如下所示：

hive> CREATE DATABASE mydatabase02 COMMENT 'This is my second database.';

OK

Time taken: 0.066 seconds

hive>

2.   通过下面的命令可以查看数据库的相关信息：

DESCRIBE DATABASE mydatabase02;

命令运行后的返回结果如下所示：

hive> DESCRIBE DATABASE mydatabase02;

OK

mydatabase02    This is my second database.    hdfs://master:9000/user/hive/warehouse/mydatabase02.db   root USER

Time taken: 0.074 seconds, Fetched: 1 row(s)

hive>

DESCRIEB DATABASE语句可以显示出这个数据库描述信息以及所在的文件目录位置路径等信息。

3.   在HiveCLI命令行界面中通过下面的命令可以在创建数据库的同时为数据库增加相关的键-值对属性信息：

CREATE DATABASE mydatabase03 WITH DBPROPERTIES('creator'='root','date'='2015-8-11');

命令运行后的返回结果如下所示：

hive> CREATE DATABASE mydatabase03 WITH DBPROPERTIES('creator'='root','date'='2015-8-11');

OK

Time taken: 0.121 seconds

hive>

4.   通过下面的命令可以查看数据库的相关信息：

DESCRIBE DATABASE mydatabase03;

命令运行后的返回结果如下所示：

hive> DESCRIBE DATABASE mydatabase03;

OK

mydatabase03         hdfs://master:9000/user/hive/warehouse/mydatabase03.db   root USER

Time taken: 0.071 seconds, Fetched: 1 row(s)

hive>

5.   通过EXTENDED关键词可以更加清晰的查看数据库的键-值对信息：

DESCRIBE DATABASE EXTENDED mydatabase03;

命令运行后的返回结果如下所示：

hive> DESCRIBE DATABASE EXTENDED mydatabase03;

OK

mydatabase03         hdfs://master:9000/user/hive/warehouse/mydatabase03.db   root USER   {date=2015-8-11, creator=root}

Time taken: 0.051 seconds, Fetched: 1 row(s)

hive>

**步骤3.**  **显示当前数据库**

1.   可以通过下面的命令设置一个属性值来在提示符里面显示当前所在的数据库信息：

set hive.cli.print.current.db=true;

命令运行后的返回结果如下所示：

hive> set hive.cli.print.current.db=true;

hive (default)>

2.   通过下面的命令切换到数据库mydatabase03，提示符中的数据库信息也会随之改变：

use mydatabase03;

命令运行后的返回结果如下所示：

hive (default)> use mydatabase03;

OK

Time taken: 0.036 seconds

hive (mydatabase03)>

**步骤4.**  **删除数据库**

默认情况下，Hive是不允许用户删除一个包含有表的数据库的。因此当数据库中存在数据表，可以先删除数据库中的表，然后再删除数据库，或者在删除命令的后面加上关键字CASCADE，这样可以使Hive自行先删除数据库中的表。

1.   通过下面的命令可以删除一个空数据库：

DROP DATABASE IF EXISTS mydatabase02;

命令运行后的返回结果如下所示：

hive (mydatabase03)> DROP DATABASE IF EXISTS mydatabase02;

OK

Time taken: 0.052 seconds

hive (mydatabase03)>

2.   通过下面的命令在数据库mydatabase03中创建表test：

CREATE TABLE mydatabase03.test(name STRING);

命令运行后的返回结果如下所示：

hive (mydatabase03)> CREATE TABLE mydatabase03.test(name STRING);

OK

Time taken: 0.404 seconds

hive (mydatabase03)>

3.   通过下面的命令可以查看mydatabase03数据库中的表信息：

show tables;

命令运行后的返回结果如下所示：

hive (mydatabase03)> show tables;

OK

test

Time taken: 0.069 seconds, Fetched: 1 row(s)

hive (mydatabase03)>

4.   通过下面的命令删除数据库mydatabase03，将会出现错误提示：

DROP DATABASE IF EXISTS mydatabase03;

命令运行后的返回结果如下所示：

hive (mydatabase03)> DROP DATABASE IF EXISTS mydatabase03;

FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask. InvalidOperationException(message:Database mydatabase03 is not empty. One or more tables exist.)

hive (mydatabase03)>

5.   通过下面的命令可以强制删除非空数据库mydatabase03：

DROP DATABASE IF EXISTS mydatabase03 CASCADE;

命令运行后的返回结果如下所示：

hive (mydatabase03)> DROP DATABASE IF EXISTS mydatabase03 CASCADE;

OK

Time taken: 1.03 seconds

hive (mydatabase03)>

6.   通过下面的命令查看数据库：

show databases;

命令运行后的返回结果如下所示，可见数据库mydatabase03已经成功被删除：

hive (mydatabase03)> show databases;

OK

default

financials

Time taken: 0.027 seconds, Fetched: 2 row(s)

hive (mydatabase03)>

7.   通过下面的命令可以切换回default数据库：

use default;

命令运行后的返回结果如下所示：

hive (mydatabase03)> use default;

OK

Time taken: 0.039 seconds

hive (default)>

**步骤5.**  **修改数据库**

在Hive中数据库的元数据信息都是不可更改的，包括数据库名和数据库所在的目录位置。可以使用ALTER DATABASE <database>命令更改某个数据库的DBPROPERTIES键-值对属性信息，但是没有办法可以删除或者“重置”数据库属性。

1.   在HiveCLI命令行界面通过下面的命令新建一个带有键-值对属性信息的数据库mydatabase04：

CREATE DATABASE mydatabase04 WITH DBPROPERTIES('data'='2012-01-01');

命令运行后的返回结果如下所示：

hive (default)> CREATE DATABASE mydatabase04 WITH DBPROPERTIES('data'='2012-01-01');

OK

Time taken: 0.055 seconds

hive (default)>

2.   通过下面的命令查看数据库的键-值对属性信息：

DESCRIBE DATABASE EXTENDED mydatabase04;

命令运行后的返回结果如下所示：

hive (default)> DESCRIBE DATABASE EXTENDED mydatabase04;

OK

mydatabase04         hdfs://master:9000/user/hive/warehouse/mydatabase04.db   root USER   {data=2012-01-01}

Time taken: 0.053 seconds, Fetched: 1 row(s)

hive (default)>

3.   通过下面的命令可以修改键-值对属性信息：

ALTER DATABASE mydatabase04 SET DBPROPERTIES('data'='2015-08-12');

命令运行后的返回结果如下所示：

hive (default)> ALTER DATABASE mydatabase04 SET DBPROPERTIES('data'='2015-08-12');

OK

Time taken: 0.068 seconds

hive (default)>

4.   通过下面的命令查看数据库的键-值对属性信息：

DESCRIBE DATABASE EXTENDED mydatabase04;

命令运行后的返回结果如下所示，可见属性信息已经成功被修改：

hive (default)> DESCRIBE DATABASE EXTENDED mydatabase04;

OK

mydatabase04         hdfs://master:9000/user/hive/warehouse/mydatabase04.db   root USER   {data=2015-08-12}

Time taken: 0.039 seconds, Fetched: 1 row(s)

hive (default)>

4  **管理表操作**

**步骤1.**  **管理表创建**

1.   通过下面的命令创建测试数据库：

CREATE DATABASE mydatabase;

命令运行后的返回结果如下所示：

hive (default)> CREATE DATABASE mydatabase;

OK

Time taken: 0.061 seconds

hive (default)>

2.   通过下面的命令在数据库mydatabase中创建表employees\_01（在Hive中，会将遇到输入的“；”作为命令终止符，因此在输入代码的时候，为了看起来整齐直观，我们可以借助回车键来进行多行输入）：

CREATE TABLE IF NOT EXISTS mydatabase.employees\_01(

name STRING COMMENT 'Employee name',

salary FLOAT COMMENT 'Employee salary',

subordinates ARRAY<STRING> COMMENT 'Names of subordinates',

deductions MAP<STRING,FLOAT> COMMENT 'Keys are deductions are percentges',

address STRUCT<street:STRING,city:STRING,state:STRING,zip:INT> COMMENT 'Home address')

COMMENT  ' Description of the table '

LOCATION '/user/hive/warehouse/mydatabase.db/employees\_01';

命令运行后的返回结果如下所示：

hive (default)> CREATE DATABASE mydatabase;

OK

Time taken: 0.061 seconds

hive (default)> CREATE TABLE IF NOT EXISTS mydatabase.employees\_01(

              > name STRING COMMENT 'Employee name',

              > salary FLOAT COMMENT 'Employee salary',

              > subordinates ARRAY<STRING> COMMENT 'Names of subordinates',

              > deductions MAP<STRING,FLOAT> COMMENT 'Keys are deductions are percentges',

              > address STRUCT<street:STRING,city:STRING,state:STRING,zip:INT> COMMENT 'Home address')

              > COMMENT  ' Description of the table '

              > LOCATION '/user/hive/warehouse/mydatabase.db/employees\_01';

OK

Time taken: 0.115 seconds

hive (default)>

3.   通过下列命令可以进行表属性信息查看：

DESCRIBE EXTENDED mydatabase.employees\_01;

命令运行后的返回结果如下所示：

hive (default)> DESCRIBE EXTENDED mydatabase.employees\_01;

OK

name                   string                  Employee name

salary                 float                   Employee salary

subordinates          array<string>          Names of subordinates

deductions            map<string,float>     Keys are deductions are percentges

address               struct<street:string,city:string,state:string,zip:int>   Home address

Detailed Table Information    Table(tableName:employees\_01, dbName:mydatabase, owner:root, createTime:1542705235, lastAccessTime:0, retention:0, sd:StorageDescriptor(cols:[FieldSchema(name:name, type:string, comment:Employee name), FieldSchema(name:salary, type:float, comment:Employee salary), FieldSchema(name:subordinates, type:array<string>, comment:Names of subordinates), FieldSchema(name:deductions, type:map<string,float>, comment:Keys are deductions are percentges), FieldSchema(name:address, type:struct<street:string,city:string,state:string,zip:int>, comment:Home address)], location:hdfs://master:9000/usr/hive/warehouse/mydatabase.db/employees\_01, inputFormat:org.apache.hadoop.mapred.TextInputFormat, outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1, serdeInfo:SerDeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{serialization.format=1}), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[], skewedColValueLocationMaps:{}), storedAsSubDirectories:false), partitionKeys:[], parameters:{transient\_lastDdlTime=1542705235, comment= Description of the table }, viewOriginalText:null, viewExpandedText:null, tableType:MANAGED\_TABLE)

Time taken: 0.138 seconds, Fetched: 7 row(s)

hive (default)>

4.   使用下面的命令可以通过拷贝一张已经存在的表的表模式创建新表：

CREATE TABLE IF NOT EXISTS mydatabase.employees\_02 LIKE mydatabase.employees\_01;

命令运行后的返回结果如下所示：

hive (default)> CREATE TABLE IF NOT EXISTS mydatabase.employees\_02 LIKE mydatabase.employees\_01;

OK

Time taken: 0.123 seconds

hive (default)>

5.   通过下列命令可以进行表属性信息查看：

DESCRIBE EXTENDED mydatabase.employees\_02;

命令运行后的返回结果如下所示，可见employees\_02表与employees\_01表属性信息是一致的：

hive (default)> DESCRIBE EXTENDED mydatabase.employees\_02;

OK

name                   string                  Employee name

salary                 float                   Employee salary

subordinates          array<string>          Names of subordinates

deductions            map<string,float>     Keys are deductions are percentges

address               struct<street:string,city:string,state:string,zip:int>   Home address

Detailed Table Information    Table(tableName:employees\_02, dbName:mydatabase, owner:root, createTime:1542705497, lastAccessTime:0, retention:0, sd:StorageDescriptor(cols:[FieldSchema(name:name, type:string, comment:Employee name), FieldSchema(name:salary, type:float, comment:Employee salary), FieldSchema(name:subordinates, type:array<string>, comment:Names of subordinates), FieldSchema(name:deductions, type:map<string,float>, comment:Keys are deductions are percentges), FieldSchema(name:address, type:struct<street:string,city:string,state:string,zip:int>, comment:Home address)], location:hdfs://master:9000/user/hive/warehouse/mydatabase.db/employees\_02, inputFormat:org.apache.hadoop.mapred.TextInputFormat, outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1, serdeInfo:SerDeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{serialization.format=1}), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[], skewedColValueLocationMaps:{}), storedAsSubDirectories:false), partitionKeys:[], parameters:{transient\_lastDdlTime=1542705497}, viewOriginalText:null, viewExpandedText:null, tableType:MANAGED\_TABLE)

Time taken: 0.122 seconds, Fetched: 7 row(s)

hive (default)>

**步骤2.**  **数据表查询**

1.   通过下面的命令可以查看指定数据库下的表信息：

SHOW TABLES IN mydatabase;

命令运行后的返回结果如下所示：

hive (default)> SHOW TABLES IN mydatabase;

OK

employees\_01

employees\_02

Time taken: 0.04 seconds, Fetched: 2 row(s)

hive (default)>

2.   通过下面的命令切换到mydatabase数据库下：

use mydatabase;

命令运行后的返回结果如下所示：

hive (default)> use mydatabase;

OK

Time taken: 0.054 seconds

hive (mydatabase)>

3.   通过下面的命令，可以使用正则表达式查看mydatabase数据库下所有以‘\_02’结尾的数据表信息：

SHOW TABLES '.\*02';

命令运行后的返回结果如下所示：

hive (mydatabase)> SHOW TABLES '.\*02';

OK

employees\_02

Time taken: 0.399 seconds, Fetched: 1 row(s)

hive (mydatabase)>

4.   可以使用FORMATTED关键字替代EXTENDED关键字进行表属性查询，此时可以提供更加可读的和冗长的输出信息，输入命令如下所示：

DESCRIBE FORMATTED mydatabase.employees\_02;

命令运行后的返回结果如下所示：

hive (mydatabase)> DESCRIBE FORMATTED mydatabase.employees\_02;

OK

# col\_name            data\_type              comment

name                   string                  Employee name

salary                 float                   Employee salary

subordinates          array<string>          Names of subordinates

deductions            map<string,float>     Keys are deductions are percentges

address               struct<street:string,city:string,state:string,zip:int>   Home address

# Detailed Table Information

Database:             mydatabase

Owner:                 root

CreateTime:           Tue Nov 20 17:18:17 CST 2018

LastAccessTime:     UNKNOWN

Protect Mode:         None

Retention:            0

Location:             hdfs://master:9000/user/hive/warehouse/mydatabase.db/employees\_02

Table Type:           MANAGED\_TABLE

Table Parameters:

   transient\_lastDdlTime 1542705497

# Storage Information

SerDe Library:        org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe

InputFormat:          org.apache.hadoop.mapred.TextInputFormat

OutputFormat:         org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat

Compressed:           No

Num Buckets:          -1

Bucket Columns:       []

Sort Columns:         []

Storage Desc Params:

   serialization.format  1

Time taken: 0.645 seconds, Fetched: 30 row(s)

hive (mydatabase)>

5.   可以通过下面的命令查看指定列的信息：

DESCRIBE FORMATTED employees\_02.salary;

命令运行后的返回结果如下所示：

hive (mydatabase)> DESCRIBE FORMATTED employees\_02.salary;

OK

# col\_name            data\_type              min                     max                     num\_nulls              distinct\_count         avg\_col\_len            max\_col\_len            num\_trues              num\_falses             comment

salary                 float                                                                                                                                                                                                                   from deserializer

Time taken: 0.262 seconds, Fetched: 3 row(s)

hive (mydatabase)>

5  **外部表操作**

**步骤1.**  **测试数据创建**

1.   打开终端模拟器，通过下列命令创建并打开测试文件/stocks.txt：

vi /stocks.txt

在文件中写入下列内容：

NASDAQ#AAPL#2010-01-01#195.69#197.88#194.00#194.12#17036300#194.12

NYSE#IBM#2010-01-01#129.46#129.86#127.94#128.49#78626000#127.92

NYSE#GE#2010-01-01#37.00#37.00#36.50#36.60#27717300#32.32

文件内容写入完成后，保存并退出。

2.   通过下面的命令在HDFS上新建一个/stocks的文件目录：

hadoop fs -mkdir /stocks

3.   通过下面的命令将本地文件/stocks.txt上传到HDFS的/stocks的目录下:

hadoop fs -put /stocks.txt /stocks

4.   通过下面的命令查看测试文件是否上传成功：

hadoop fs -cat /stocks/stocks.txt

命令运行后的返回结果如下所示：

[root@master ~]# hadoop fs -cat /stocks/stocks.txt

18/11/20 17:45:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

NASDAQ#AAPL#2010-01-01#195.69#197.88#194.00#194.12#17036300#194.12

NYSE#IBM#2010-01-01#129.46#129.86#127.94#128.49#78626000#127.92

NYSE#GE#2010-01-01#37.00#37.00#36.50#36.60#27717300#32.32

[root@master ~]#

**步骤2.**  **外部表管理**

1.   通过下面的命令指定EXTERNAL关键字创建一个外部表，并指定外部表对应的数据源目录为HDFS中的/stocks目录：

CREATE EXTERNAL TABLE IF NOT EXISTS mydatabase.stocks (

bourse STRING,

symbol STRING,

ymd STRING,

price\_open FLOAT,

price\_high FLOAT,

price\_low FLOAT,

price\_close FLOAT,

volume INT,

price\_adj\_close FLOAT)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '#'

LOCATION '/stocks';

命令运行后的返回结果如下所示：

hive (mydatabase)> CREATE EXTERNAL TABLE IF NOT EXISTS mydatabase.stocks (

                 > bourse STRING,

                 > symbol STRING,

                 > ymd STRING,

                 > price\_open FLOAT,

                 > price\_high FLOAT,

                 > price\_low FLOAT,

                 > price\_close FLOAT,

                 > volume INT,

                 > price\_adj\_close FLOAT)

                 > ROW FORMAT DELIMITED FIELDS TERMINATED BY '#'

                 > LOCATION '/stocks';

OK

Time taken: 0.452 seconds

hive (mydatabase)>

2.   通过下面的命令查看外部表数据以确定数据是否加载成功：

SELECT \* FROM mydatabase.stocks;

命令运行后的返回结果如下所示，由返回结果可以看到数据已经成功被加载：

hive (mydatabase)> SELECT \* FROM mydatabase.stocks;

OK

NASDAQ  AAPL 2010-01-01   195.69   197.88   194.0    194.12   17036300 194.12

NYSE    IBM 2010-01-01   129.46   129.86   127.94   128.49   78626000 127.92

NYSE    GE  2010-01-01   37.0 37.0 36.5 36.6 27717300 32.32

Time taken: 0.505 seconds, Fetched: 3 row(s)

hive (mydatabase)>

3.   通过下面的命令查看外部表的相关属性信息：

DESCRIBE FORMATTED mydatabase.stocks;

命令运行后的返回结果如下所示：

hive (mydatabase)> DESCRIBE FORMATTED mydatabase.stocks;

OK

# col\_name            data\_type              comment

bourse                 string

symbol                 string

ymd                    string

price\_open            float

price\_high            float

price\_low             float

price\_close           float

volume                 int

price\_adj\_close     float

# Detailed Table Information

Database:             mydatabase

Owner:                 root

CreateTime:           Tue Nov 20 17:47:53 CST 2018

LastAccessTime:     UNKNOWN

Protect Mode:         None

Retention:            0

Location:             hdfs://master:9000/stocks

Table Type:           EXTERNAL\_TABLE

Table Parameters:

   COLUMN\_STATS\_ACCURATE false

   EXTERNAL               TRUE

   numFiles               0

   numRows                 -1

   rawDataSize            -1

   totalSize              0

   transient\_lastDdlTime 1542707273

# Storage Information

SerDe Library:        org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe

InputFormat:          org.apache.hadoop.mapred.TextInputFormat

OutputFormat:         org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat

Compressed:           No

Num Buckets:          -1

Bucket Columns:       []

Sort Columns:         []

Storage Desc Params:

   field.delim            #

   serialization.format  #

Time taken: 0.147 seconds, Fetched: 41 row(s)

hive (mydatabase)>

由返回结果可以看到，对于外部表所查看到的表类型信息为EXTERNAL\_TABLE，而对于管理表而言，所查看到的表类型信息为MANAGED\_TABLE。

6  **管理分区表操作**

**步骤1.**  **测试数据创建**

1.   打开终端模拟器，通过下列命令创建并打开测试文件/employees01.txt：

vi /employees01.txt

在文件中写入下面的内容：

John Doe#100000#Mary Smith\*Todd Jones#Federal Taxes:0.2\*State Taxes:0.05\*Insurance:0.1#1 Michigan Ave.\*Hilo\*HI\*60500

Boss Man#200000#Fred Finance#Federal Taxes:0.2\*State Taxes:0.05\*Insurance:0.1#1 Pretentious Drive.\*Maui\*HI\*60500

Fred Finance#150000#Stacy Accountant#Federal Taxes:0.2\*State Taxes:0.05\*Insurance:0.1#2 Pretentious Drive\*Kona\*HI\*60500

内容写入完成后保存并退出。

2.   打开终端模拟器，通过下列命令创建并打开测试文件/employees02.txt：

vi /employees02.txt

在文件中写入下面的内容：

Mary Smith#80000#Bill King#Federal Taxes:0.2\*State Taxes:0.05\*Insurance:0.1#100 Ontario St.\*Miami\*FI\*60601

内容写入完成后保存并退出。

3.   打开终端模拟器，通过下列命令创建并打开测试文件/employees03.txt：

vi /employees03.txt

在文件中写入下面的内容：

Todd Jones#70000##Federal Taxes:0.15\*State Taxes:0.03\*Insurance:0.1#200 Chicago AVE.\*Whitehouse\*YT\*60700

内容写入完成后保存并退出。

4.   打开终端模拟器，通过下列命令创建并打开测试文件/employees04.txt：

vi /employees04.txt

在文件中写入下面的内容：

Bill King#60000##Federal Taxes:0.3\*State Taxes:0.03\*Insurance:0.1#300 Obscure Dr.\*Winninpeg\*MB\*60100

内容写入完成后保存并退出。

**步骤2.**  **管理分区表创建及操作**

1.   在HiveCLI命令行操作界面，通过下面的命令在mydatabase数据库下创建分区表employees：

CREATE TABLE mydatabase.employees (

name STRING,

salary INT,

subordinates ARRAY<STRING>,

deductions MAP<STRING, FLOAT>,

address STRUCT<street:STRING,city:STRING,state:STRING,zip:INT>

)

PARTITIONED BY (country STRING,state STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '#'

COLLECTION ITEMS TERMINATED BY '\*'

MAP KEYS TERMINATED BY ':'

LINES TERMINATED BY '\n';

命令运行后的返回结果如下所示：

hive (mydatabase)> CREATE TABLE mydatabase.employees (

                 > name STRING,

                 > salary INT,

                 > subordinates ARRAY<STRING>,

                 > deductions MAP<STRING, FLOAT>,

                 > address STRUCT<street:STRING,city:STRING,state:STRING,zip:INT>

                 > )

                 > PARTITIONED BY (country STRING,state STRING)

                 > ROW FORMAT DELIMITED FIELDS TERMINATED BY '#'

                 > COLLECTION ITEMS TERMINATED BY '\*'

                 > MAP KEYS TERMINATED BY ':'

                 > LINES TERMINATED BY '\n';

OK

Time taken: 0.129 seconds

hive (mydatabase)>

2.   通过下面的命令将本地文件/employees01.txt中的数据装载到我们新建的分区表employees的country='US',state='HI'分区中：

LOAD DATA LOCAL INPATH '/employees01.txt' INTO TABLE employees

PARTITION (country='US',state='HI');

命令运行后的返回结果如下所示：

hive (mydatabase)> LOAD DATA LOCAL INPATH '/employees01.txt' INTO TABLE employees

                 > PARTITION (country='US',state='HI');

Loading data to table mydatabase.employees partition (country=US, state=HI)

Partition mydatabase.employees{country=US, state=HI} stats: [numFiles=1, numRows=0, totalSize=350, rawDataSize=0]

OK

Time taken: 0.98 seconds

hive (mydatabase)>

3.   通过下面的命令将本地文件/employees02.txt中的数据装载到我们新建的分区表employees的country='US',state='FL'分区中：

LOAD DATA LOCAL INPATH '/employees02.txt' INTO TABLE employees

PARTITION (country='US',state='FL');

命令运行后的返回结果如下所示：

hive (mydatabase)> LOAD DATA LOCAL INPATH '/employees02.txt' INTO TABLE employees

                 > PARTITION (country='US',state='FL');

Loading data to table mydatabase.employees partition (country=US, state=FL)

Partition mydatabase.employees{country=US, state=FL} stats: [numFiles=1, numRows=0, totalSize=107, rawDataSize=0]

OK

Time taken: 0.516 seconds

hive (mydatabase)>

4.   通过下面的命令将本地文件/employees03.txt中的数据装载到我们新建的分区表employees的country='US',state='HI'分区中：

LOAD DATA LOCAL INPATH '/employees03.txt' INTO TABLE employees

PARTITION (country='US',state='HI');

命令运行后的返回结果如下所示：

hive (mydatabase)> LOAD DATA LOCAL INPATH '/employees03.txt' INTO TABLE employees

                 > PARTITION (country='US',state='HI');

Loading data to table mydatabase.employees partition (country=US, state=HI)

Partition mydatabase.employees{country=US, state=HI} stats: [numFiles=2, numRows=0, totalSize=455, rawDataSize=0]

OK

Time taken: 0.615 seconds

hive (mydatabase)>

5.   通过下面的命令将本地文件/employees04.txt中的数据装载到我们新建的分区表employees的country='CA',state='MB'分区中：

LOAD DATA LOCAL INPATH '/employees04.txt' INTO TABLE employees

PARTITION (country='CA',state='MB');

命令运行后的返回结果如下所示：

hive (mydatabase)> LOAD DATA LOCAL INPATH '/employees04.txt' INTO TABLE employees

                 > PARTITION (country='CA',state='MB');

Loading data to table mydatabase.employees partition (country=CA, state=MB)

Partition mydatabase.employees{country=CA, state=MB} stats: [numFiles=1, numRows=0, totalSize=101, rawDataSize=0]

OK

Time taken: 0.445 seconds

hive (mydatabase)>

6.   通过下面的命令查询我们刚刚新建的表employees数据：

SELECT \* FROM employees;

命令运行后的返回结果如下所示：

hive (mydatabase)> SELECT \* FROM employees;

OK

Bill King   60000    []  {"Federal Taxes":0.3,"State Taxes":0.03,"Insurance":0.1} {"street":"300 Obscure Dr.","city":"Winninpeg","state":"MB","zip":60100}   CA  MB

Mary Smith  80000    ["Bill King"] {"Federal Taxes":0.2,"State Taxes":0.05,"Insurance":0.1} {"street":"100 Ontario St.","city":"Miami","state":"FI","zip":60601}  US  FL

John Doe    100000   ["Mary Smith","Todd Jones"]    {"Federal Taxes":0.2,"State Taxes":0.05,"Insurance":0.1}    {"street":"1 Michigan Ave.","city":"Hilo","state":"HI","zip":60500}    US  HI

Boss Man    200000   ["Fred Finance"]  {"Federal Taxes":0.2,"State Taxes":0.05,"Insurance":0.1}  {"street":"1 Pretentious Drive.","city":"Maui","state":"HI","zip":60500}   US  HI

Fred Finance     150000   ["Stacy Accountant"]  {"Federal Taxes":0.2,"State Taxes":0.05,"Insurance":0.1} {"street":"2 Pretentious Drive","city":"Kona","state":"HI","zip":60500}   US  HI

Todd Jones  70000    []  {"Federal Taxes":0.15,"State Taxes":0.03,"Insurance":0.1}   {"street":"200 Chicago AVE.","city":"Whitehouse","state":"YT","zip":60700} US  HI

Time taken: 0.279 seconds, Fetched: 6 row(s)

hive (mydatabase)>

7.   通过下面的命令可以查询出country=‘US’，state=‘HI’的雇员信息：

SELECT \* FROM employees WHERE country='US' AND state='HI';

命令运行后的返回结果如下所示：

hive (mydatabase)> SELECT \* FROM employees WHERE country='US' AND state='HI';

OK

John Doe    100000   ["Mary Smith","Todd Jones"]    {"Federal Taxes":0.2,"State Taxes":0.05,"Insurance":0.1}    {"street":"1 Michigan Ave.","city":"Hilo","state":"HI","zip":60500}    US  HI

Boss Man    200000   ["Fred Finance"]  {"Federal Taxes":0.2,"State Taxes":0.05,"Insurance":0.1}  {"street":"1 Pretentious Drive.","city":"Maui","state":"HI","zip":60500}   US  HI

Fred Finance     150000   ["Stacy Accountant"]  {"Federal Taxes":0.2,"State Taxes":0.05,"Insurance":0.1} {"street":"2 Pretentious Drive","city":"Kona","state":"HI","zip":60500}   US  HI

Todd Jones  70000    []  {"Federal Taxes":0.15,"State Taxes":0.03,"Insurance":0.1}   {"street":"200 Chicago AVE.","city":"Whitehouse","state":"YT","zip":60700} US  HI

Time taken: 0.747 seconds, Fetched: 4 row(s)

hive (mydatabase)>

**步骤3.**  **模式设置及分区查询**

1.   通过下面的命令可以将Hive设置为“strict（严格）”模式（此时如果对分区表进行查询而WHERE子句没有加分区过滤条件，将会禁止提交这个任务）：

set hive.mapred.mode=strict;

2.   通过下面命令测试“strict（严格）”模式设置是否生效：

SELECT name,salary FROM employees;

命令运行后的返回结果如下所示，由返回结果可以看出，没有使用WHERE子句增加分区过滤条件，将会禁止提交这个任务：

hive (mydatabase)> SELECT name,salary FROM employees;

FAILED: SemanticException [Error 10041]: No partition predicate found for Alias "employees" Table "employees"

hive (mydatabase)>

3.   可以通过下面的语句将Hive设置为“nostrict（非严格）”模式：

set hive.mapred.mode=nonstrict;

4.   通过下面命令测试“nostrict（非严格）”模式设置是否生效：

SELECT name,salary FROM employees;

命令运行后的返回结果如下所示，由返回结果可以看出，没有使用WHERE子句增加分区过滤条件，依然可以提交任务：

hive (mydatabase)> SELECT name,salary FROM employees;

OK

Bill King   60000

Mary Smith  80000

John Doe    100000

Boss Man    200000

Fred Finance     150000

Todd Jones  70000

Time taken: 0.132 seconds, Fetched: 6 row(s)

hive (mydatabase)>

5.   通过下面的命令可以查看表中存在的所有分区：

SHOW PARTITIONS employees;

命令运行后的返回结果如下所示：

hive (mydatabase)> SHOW PARTITIONS employees;

OK

country=CA/state=MB

country=US/state=FL

country=US/state=HI

Time taken: 0.154 seconds, Fetched: 3 row(s)

hive (mydatabase)>

6.   通过下面的命令可以指定一个或者多个特定分区字段进行过滤查询：

SHOW PARTITIONS employees PARTITION(country='US');

命令运行后的返回结果如下所示：

hive (mydatabase)> SHOW PARTITIONS employees PARTITION(country='US');

OK

country=US/state=HI

country=US/state=FL

Time taken: 0.172 seconds, Fetched: 2 row(s)

hive (mydatabase)>

7.   通过下面的命令可以查看分区表信息：

DESCRIBE EXTENDED employees;

命令运行后的返回结果如下所示：

hive (mydatabase)> DESCRIBE EXTENDED employees;

OK

name                   string

salary                 int

subordinates          array<string>

deductions            map<string,float>

address               struct<street:string,city:string,state:string,zip:int>

country               string

state                  string

# Partition Information

# col\_name            data\_type              comment

country               string

state                  string

Detailed Table Information    Table(tableName:employees, dbName:mydatabase, owner:root, createTime:1542763985, lastAccessTime:0, retention:0, sd:StorageDescriptor(cols:[FieldSchema(name:name, type:string, comment:null), FieldSchema(name:salary, type:int, comment:null), FieldSchema(name:subordinates, type:array<string>, comment:null), FieldSchema(name:deductions, type:map<string,float>, comment:null), FieldSchema(name:address, type:struct<street:string,city:string,state:string,zip:int>, comment:null), FieldSchema(name:country, type:string, comment:null), FieldSchema(name:state, type:string, comment:null)], location:hdfs://master:9000/user/hive/warehouse/mydatabase.db/employees, inputFormat:org.apache.hadoop.mapred.TextInputFormat, outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1, serdeInfo:SerDeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{line.delim=

, field.delim=#, colelction.delim=\*, mapkey.delim=:, serialization.format=#}), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[], skewedColValueLocationMaps:{}), storedAsSubDirectories:false), partitionKeys:[FieldSchema(name:country, type:string, comment:null), FieldSchema(name:state, type:string, comment:null)], parameters:{transient\_lastDdlTime=1542763985}, viewOriginalText:null, viewExpandedText:null, tableType:MANAGED\_TABLE)

Time taken: 0.117 seconds, Fetched: 16 row(s)

hive (mydatabase)>

在输出信息中的模式信息部分会将country和state以及其他字段列在一起，因为就查询而言，它们就是字段，在Detailed Table Information（详细表信息）中将country和state作为分区键处理。

7  **外部分区表操作**

**步骤1.**  **测试数据创建**

1.   打开终端模拟器，通过下列命令创建并打开测试文件/dividends\_2008.txt：

vi /dividends\_2008.txt

在文件中写入下面的内容：

NASDAQ#AAPL#0.03

NYSE#IBM#0.5

NYSE#GE#0.31

内容写入完成后保存并退出。

2.   打开终端模拟器，通过下列命令创建并打开测试文件/dividends\_2009.txt：

vi /dividends\_2009.txt

在文件中写入下面的内容：

NASDAQ#AAPL#0.02

NYSE#IBM#0.4

NYSE#GE#0.1

内容写入完成后保存并退出。

3.   打开终端模拟器，通过下列命令创建并打开测试文件/dividends\_2010.txt：

vi /dividends\_2010.txt

在文件中写入下面的内容：

NASDAQ#AAPL#0.03

NYSE#IBM#0.3

NYSE#GE#0.31

内容写入完成后保存并退出。

4.   打开终端模拟器，通过下面的命令在HDFS中创建文件夹/dividends：

hadoop fs -mkdir /dividends/

5.   通过下面的命令在/dividends中创建子文件夹/2008：

hadoop fs -mkdir /dividends/2008

6.   通过下面的命令在/dividends中创建子文件夹/2009：

hadoop fs -mkdir /dividends/2009

7.   通过下面的命令在/dividends中创建子文件夹/2010：

hadoop fs -mkdir /dividends/2010

8.   通过下面的命令将本地文件/dividends\_2008.txt上传到HDFS的/dividends/2008目录下：

hadoop fs -put /dividends\_2008.txt  /dividends/2008

9.   通过下面的命令将本地文件/dividends\_2009.txt上传到HDFS的/dividends/2009目录下：

hadoop fs -put /dividends\_2009.txt  /dividends/2009

10.  通过下面的命令将本地文件/dividends\_2010.txt上传到HDFS的/dividends/2010目录下：

hadoop fs -put /dividends\_2010.txt  /dividends/2010

**步骤2.**  **外部分区表创建**

1.   在HiveCLI命令行操作界面中，通过下面的命令在mydatabase数据库中创建一个外部分区表dividends：

CREATE EXTERNAL TABLE mydatabase.dividends (

bourse STRING,

symbol STRING,

dividend FLOAT)

PARTITIONED BY (year STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '#';

命令运行后的返回结果如下所示：

hive (mydatabase)> CREATE EXTERNAL TABLE mydatabase.dividends (

                 > bourse STRING,

                 > symbol STRING,

                 > dividend FLOAT)

                 > PARTITIONED BY (year STRING)

                 > ROW FORMAT DELIMITED FIELDS TERMINATED BY '#';

OK

Time taken: 0.112 seconds

hive (mydatabase)>

2.   通过下面的命令，为dividends表增加一个2008-01-01分区，并指定数据源存放地址对应为HDFS中的/dividends/2008目录：

ALTER TABLE mydatabase.dividends ADD PARTITION (year='2008-01-01') LOCATION 'hdfs://localhost:9000/dividends/2008';

命令运行后的返回结果如下所示：

hive (mydatabase)> ALTER TABLE mydatabase.dividends ADD PARTITION (year='2008-01-01') LOCATION 'hdfs://localhost:9000/dividends/2008';

OK

Time taken: 0.168 seconds

hive (mydatabase)>

3.   通过下面的命令，为dividends表增加一个2009-01-01分区，并指定数据源存放地址对应为HDFS中的/dividends/2009目录：

ALTER TABLE mydatabase.dividends ADD PARTITION (year='2009-01-01') LOCATION 'hdfs://localhost:9000/dividends/2009';

命令运行后的返回结果如下所示：

hive (mydatabase)> ALTER TABLE mydatabase.dividends ADD PARTITION (year='2009-01-01') LOCATION 'hdfs://localhost:9000/dividends/2009';

OK

Time taken: 0.084 seconds

hive (mydatabase)>

4.   通过下面的命令，为dividends表增加一个2010-01-01分区，并指定数据源存放地址对应为HDFS中的/dividends/2010目录：

ALTER TABLE mydatabase.dividends ADD PARTITION (year='2010-01-01') LOCATION 'hdfs://localhost:9000/dividends/2010';

命令运行后的返回结果如下所示：

hive (mydatabase)> ALTER TABLE mydatabase.dividends ADD PARTITION (year='2010-01-01') LOCATION 'hdfs://localhost:9000/dividends/2010';

OK

Time taken: 0.088 seconds

hive (mydatabase)>

**步骤3.**  **外部分区表信息查询**

1.   通过下面命令查看分区表dividends数据：

select \* from mydatabase.dividends;

命令运行后的返回结果如下所示：

hive (mydatabase)> select \* from mydatabase.dividends;

OK

NASDAQ  AAPL 0.03 2008-01-01

NYSE    IBM 0.5 2008-01-01

NYSE    GE  0.31 2008-01-01

NASDAQ  AAPL 0.02 2009-01-01

NYSE    IBM 0.4 2009-01-01

NYSE    GE  0.1 2009-01-01

NASDAQ  AAPL 0.03 2010-01-01

NYSE    IBM 0.3 2010-01-01

NYSE    GE  0.31 2010-01-01

Time taken: 0.161 seconds, Fetched: 9 row(s)

hive (mydatabase)>

2.   和分区管理表一样，通过SHOW PARTITIONS命令可以查看一个外部表的分区信息：

SHOW PARTITIONS mydatabase.dividends;

命令运行后的返回结果如下所示：

hive (mydatabase)> SHOW PARTITIONS mydatabase.dividends;

OK

year=2008-01-01

year=2009-01-01

year=2010-01-01

Time taken: 0.094 seconds, Fetched: 3 row(s)

hive (mydatabase)>

3.   可以通过如下命令可以查看指定分区信息，同时可以查看到分区数据所在的路径信息：

DESCRIBE EXTENDED mydatabase.dividends PARTITION (year='2010-01-01');

命令运行后的返回结果如下所示：

hive (mydatabase)> DESCRIBE EXTENDED mydatabase.dividends PARTITION (year='2010-01-01');

OK

bourse                 string

symbol                 string

dividend              float

year                   string

# Partition Information

# col\_name            data\_type              comment

year                   string

Detailed Partition Information    Partition(values:[2010-01-01], dbName:mydatabase, tableName:dividends, createTime:1542951517, lastAccessTime:0, sd:StorageDescriptor(cols:[FieldSchema(name:bourse, type:string, comment:null), FieldSchema(name:symbol, type:string, comment:null), FieldSchema(name:dividend, type:float, comment:null), FieldSchema(name:year, type:string, comment:null)], location:hdfs://localhost:9000/dividends/2010, inputFormat:org.apache.hadoop.mapred.TextInputFormat, outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1, serdeInfo:SerDeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{field.delim=#, serialization.format=#}), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[], skewedColValueLocationMaps:{}), storedAsSubDirectories:false), parameters:{totalSize=43, numRows=-1, rawDataSize=-1, COLUMN\_STATS\_ACCURATE=false, numFiles=1, transient\_lastDdlTime=1542951517})

Time taken: 0.196 seconds, Fetched: 11 row(s)

hive (mydatabase)>

8  **表删除**

**步骤1.**  **创建测试数据表**

1.   通过下面命令在数据库mydatabase中创建数据表：

create table mydatabase.test(id int);

命令运行后的返回结果如下所示：

hive (mydatabase)> create table mydatabase.test(id int);

OK

Time taken: 0.079 seconds

hive (mydatabase)>

2.   通过下面命令查看数据库mydatabase下的数据表：

show tables in mydatabase;

命令运行后的返回结果如下所示，由返回结果可以看出，数据表test已经成功被创建：

hive (mydatabase)> show tables in mydatabase;

OK

dividends

employees

employees\_01

employees\_02

stocks

test

Time taken: 0.038 seconds, Fetched: 6 row(s)

hive (mydatabase)>

**步骤2.**  **数据表删除**

1.   通过下面命令删除数据库mydatabase下的数据表test：

DROP TABLE IF EXISTS mydatabase.test;

命令运行后的返回结果如下所示，

hive (mydatabase)> DROP TABLE IF EXISTS mydatabase.test;

OK

Time taken: 0.758 seconds

hive (mydatabase)>

2.   通过下面命令查看数据库mydatabase下的数据表：

show tables in mydatabase;

命令运行后的返回结果如下所示，由返回结果可以看出，数据表test已经成功被删除：

hive (mydatabase)> show tables in mydatabase;

OK

dividends

employees

employees\_01

employees\_02

stocks

Time taken: 0.045 seconds, Fetched: 5 row(s)

hive (mydatabase)>

9  **修改表**

大多数的表属性可以通过ALTER TABLE语句来进行修改。这种操作会修改元数据，但不会修改数据本身。这些语句可用于修改表模式中出现的错误、改变分区路径以及其他一些操作。

**步骤1.**  **表名修改**

1.   通过下面的命令可以对数据库mydatabase中已经存在的表dividends进行重新命名：

ALTER TABLE mydatabase.dividends RENAME TO mydatabase.my\_dividends;

命令运行后的返回结果如下所示：

hive (mydatabase)> ALTER TABLE mydatabase.dividends RENAME TO mydatabase.my\_dividends;

OK

Time taken: 0.138 seconds

hive (mydatabase)>

2.   通过下面的命令查看数据库表信息：

SHOW TABLES IN mydatabase;

命令运行后的返回结果如下所示：

hive (mydatabase)> SHOW TABLES IN mydatabase;

OK

employees

employees\_01

employees\_02

my\_dividends

stocks

Time taken: 0.047 seconds, Fetched: 5 row(s)

hive (mydatabase)>

由返回结果可以看出表dividends已经被重新命名为my\_dividends。

**步骤2.**  **分区修改**

1.   通过下面的命令可以为分区表my\_dividends增加分区（hive不关心一个分区对应的分区目录下是否存在文件，所以location子句后面的路径为空时，也能为表增加分区）：

ALTER TABLE mydatabase.my\_dividends ADD IF NOT EXISTS

PARTITION (year='2011-01-01') LOCATION 'hdfs://localhost:9000/dividends/2011'

PARTITION (year='2012-01-01') LOCATION 'hdfs://localhost:9000/dividends/2012';

命令运行后的返回结果如下所示：

hive (mydatabase)> ALTER TABLE mydatabase.my\_dividends ADD IF NOT EXISTS

                 > PARTITION (year='2011-01-01') LOCATION 'hdfs://localhost:9000/dividends/2011'

                 > PARTITION (year='2012-01-01') LOCATION 'hdfs://localhost:9000/dividends/2012';

OK

Time taken: 0.103 seconds

hive (mydatabase)>

2.   通过下面的命令查询year=2011-01-01分区对应的信息：

SELECT \* FROM mydatabase.my\_dividends WHERE year='2011-01-01';

命令运行后的返回结果如下所示：

hive (mydatabase)> SELECT \* FROM mydatabase.my\_dividends WHERE year='2011-01-01';

OK

Time taken: 0.171 seconds

hive (mydatabase)>

由于year=2011-01-01的分区对应的数据路径为hdfs://localhost:9000/dividends/2011，而此路径下并没有数据信息，因此查询返回的结果为空。

3.   通过下面的命令修改year=2011-01-01分区的数据路径为hdfs://localhost:9000/dividends/2008：

ALTER TABLE mydatabase.my\_dividends PARTITION (year='2011-01-01') SET LOCATION 'hdfs://localhost:9000/dividends/2008';

命令运行后的返回结果如下所示：

hive (mydatabase)> ALTER TABLE mydatabase.my\_dividends PARTITION (year='2011-01-01') SET LOCATION 'hdfs://localhost:9000/dividends/2008';

OK

Time taken: 0.234 seconds

hive (mydatabase)>

4.   通过下面的命令再次查询year=2011-01-01分区对应的信息：

SELECT \* FROM mydatabase.my\_dividends WHERE year='2011-01-01';

命令运行后的返回结果如下所示：

hive (mydatabase)> SELECT \* FROM mydatabase.my\_dividends WHERE year='2011-01-01';

OK

NASDAQ  AAPL 0.03 2011-01-01

NYSE    IBM 0.5 2011-01-01

NYSE    GE  0.31 2011-01-01

Time taken: 0.129 seconds, Fetched: 3 row(s)

hive (mydatabase)>

5.   通过下面的命令可以查询mydatabase数据库下my\_dividends表的分区信息：

SHOW PARTITIONS mydatabase.my\_dividends;

命令运行后的返回结果如下所示：

hive (mydatabase)> SHOW PARTITIONS mydatabase.my\_dividends;

OK

year=2008-01-01

year=2009-01-01

year=2010-01-01

year=2011-01-01

year=2012-01-01

Time taken: 0.067 seconds, Fetched: 5 row(s)

hive (mydatabase)>

6.   通过下面的命令可以删除my\_dividends表中year=2012-01-01对应的分区信息：

ALTER TABLE mydatabase.my\_dividends DROP PARTITION(year='2012-01-01');

命令运行后的返回结果如下所示：

hive (mydatabase)> ALTER TABLE mydatabase.my\_dividends DROP PARTITION(year='2012-01-01');

Dropped the partition year=2012-01-01

OK

Time taken: 0.358 seconds

hive (mydatabase)>

7.   此时我们再次查看my\_dividends表的分区信息：

SHOW PARTITIONS mydatabase.my\_dividends;

命令运行后的返回结果如下所示：

hive (mydatabase)> SHOW PARTITIONS mydatabase.my\_dividends;

OK

year=2008-01-01

year=2009-01-01

year=2010-01-01

year=2011-01-01

Time taken: 0.085 seconds, Fetched: 4 row(s)

hive (mydatabase)>

由返回结果可以看出，year=2012-01-01对应的分区信息已经成功被删除。