一.HBASE

2017年4月10日

10:08

**一.HBASE架构**

**二.深入浅出HBASE -吴超**

**(一)第一次课**

**1.简介**

hbase是面向列的列式存储,列存放在一个段中;

hbase使用HDFS作为文件存储系统;

***(1)逻辑模型:***

表:

行:

(columns family)列:不能脱离列簇而独立存在.

(row keys)行键:

时间戳:数据插入数据库的时间,在HBASE中是天然存在的.

***(2)物理模型***

ab列放在一个文件中;

cde列放在一个文件中;

(columns family)列簇物理上单独存储一系列列的文件.

数据在存储的时候,是按照行键的存储的, 存储的时候是经过大小比较.

不是按照插入顺序存储的.

***(3)存储模型***

Cloums family(列簇)存放在一个文件:

region多个行,每个region放在不同的region server上;

(负载均衡功能)

存放一个行键的多列数据:

**(二)HBASE的体系结构**

**1.HMaster和HRegionServer**

HRegionServer存放region行簇

HMaster处理请求,找到region在哪个HRegionServer

***(1)两张特殊的table***

..meta存储了region的起始row key行键;

..meta表的数据量比较大的时候,也将..meta划分region存放不同region上;

..root记录了..meta表的region的信息.

**(三)hbase shell访问hbase**

**1.简单语句**

***(1)建表***

语句:表名称,列簇名称;

hbase(main):003:0> create 'stu','bas'

0 row(s) in 2.7590 seconds

=> Hbase::Table - stu

***(2)查看表:***

hbase(main):006:0> exists 'stu'

Table stu does exist

0 row(s) in 0.1350 seconds

hbase(main):007:0> list 'stu'

TABLE

stu

1 row(s) in 0.0380 seconds

=> ["stu"]

***(3)插入数据***

Put table\_name,Row key,columns family:column,

put 'stu','1','bas:id','1'

put 'stu','1','bas:name','clark'

put 'stu','1','bas:age','23'

put 'stu','2','bas:id','2'

put 'stu','2','bas:name','saber'

put 'stu','2','bas:age','23'

***(4)查询一条记录:列是一行查看***

get 'stu','1'

get 'stu','1','bas:name'

hbase(main):006:0> get 'stu','1'

COLUMN CELL

bas:age timestamp=1491794621821, value=23

bas:id timestamp=1491794516290, value=1

bas:name timestamp=1491794647517, value=clark

1 row(s) in 0.0750 seconds

hbase(main):007:0> get 'stu','1','bas:name'

COLUMN CELL

bas:name timestamp=1491794647517, value=clark

hbase(main):009:0> get 'stu','2'

COLUMN CELL

bas:age timestamp=1491794641807, value=23

bas:id timestamp=1491794631942, value=2

bas:name timestamp=1491794637264, value=saber

***(5)查看所有数据***

scan 'stu'

hbase(main):010:0> scan 'stu'

ROW COLUMN+CELL

1 column=bas:age, timestamp=1491794621821, value=23

1 column=bas:id, timestamp=1491794516290, value=1

1 column=bas:name, timestamp=1491794647517, value=clark

2 column=bas:age, timestamp=1491794641807, value=23

2 column=bas:id, timestamp=1491794631942, value=2

2 column=bas:name, timestamp=1491794637264, value=saber

***(6)删除列***

delete 'stu','2','bas:age'

hbase(main):011:0> delete 'stu','2','bas:age'

0 row(s) in 0.0440 seconds

hbase(main):012:0> get 'stu','2','bas:age'

COLUMN CELL

0 row(s) in 0.0300 seconds

***(7)删除行键(删除行键的所有列)***

deleteall 'stu','2'

***(8)修改***

Put 就是新增

get 'stu','1','bas:name'

put 'stu','1','bas:name','saber'

get 'stu','1','bas:name'

hbase(main):017:0> get 'stu','1','bas:name'

COLUMN CELL

bas:name timestamp=1491794647517, value=clark

1 row(s) in 0.0450 seconds

hbase(main):018:0> get 'stu','1','bas:name'

COLUMN CELL

bas:name timestamp=1491794647517, value=clark

1 row(s) in 0.0240 seconds

hbase(main):019:0> put 'stu','1','bas:name','saber'

0 row(s) in 0.0130 seconds

hbase(main):020:0> get 'stu','1','bas:name'

COLUMN CELL

bas:name timestamp=1491795194623, value=saber

1 row(s) in 0.0310 seconds

***(9)删除表***

hbase(main):021:0> disable 'test'

0 row(s) in 2.4880 seconds

hbase(main):022:0> drop 'test'

0 row(s) in 1.3100 seconds

**(四)使用java来访问HBASE数据库**

**三.Python通过thrift操作Hbase**

<http://blog.sina.com.cn/s/blog_9f48885501018g03.html>

**(一)Hbase安装thrift**

<https://mirrors.tuna.tsinghua.edu.cn/apache/thrift/0.10.0/>[thrift-0.10.0.tar.gz](https://mirrors.tuna.tsinghua.edu.cn/apache/thrift/0.10.0/thrift-0.10.0.tar.gz)

## Requirements for building from source

\* GNU build tools:

\* autoconf 2.65

\* automake 1.13

\* libtool 1.5.24

\* pkg-config autoconf macros (pkg.m4)

\* lex and yacc (developed primarily with flex and bison)

\* libssl-dev

### Install C++ Lib Dependencies

sudo yum -y install libevent-devel zlib-devel openssl-devel

### Upgrade Boost >= 1.53

wget <http://sourceforge.net/projects/boost/files/boost/1.53.0/boost_1_53_0.tar.gz>

tar xvf boost\_1\_53\_0.tar.gz

cd boost\_1\_53\_0

./bootstrap.sh

sudo ./b2 install

(error: no command provided, default command 'g++' not found

**yum install gcc-c++**

...failed updating 58 targets...

...skipped 12 targets...

...updated 898 targets...

)

## Build and Install the Apache Thrift IDL Compiler

git clone <https://git-wip-us.apache.org/repos/asf/thrift.git>

cd thrift

./bootstrap.sh

./configure --with-lua=no

make

sudo make install

This will build the compiler (thrift/compiler/cpp/thrift --version) and any language libraries supported. The make install step installs the compiler on the path: /usr/local/bin/thrift

You can use the ./configure --enable-libs=no switch to build the Apache Thrift IDL Compiler only without lib builds. To run tests use "make check".

(

[root@node01 thrift-0.10.0]# find -name bootstrap.sh

./contrib/fb303/bootstrap.sh

[root@node01 thrift-0.10.0]# ./contrib/fb303/bootstrap.sh

Building Python Library ...... : yes

Building Perl Library ........ : no

Building NodeJS Library ...... : no

Python Library:

Using Python .............. : /usr/bin/python

If something is missing that you think should be present,

please skim the output of configure to find the missing

component. Details are present in config.log.

[root@node01 ~]# yum -y install libboost-dev libboost-test-dev libboost-program-options-dev libevent-dev automake libtool flex bison pkg-config g++ libssl-dev

Installing : bison-2.4.1-5.el6.x86\_64 Installing : flex-2.5.35-9.el6.x86\_64

./configure --with-boost=/usr/local

Make

Make install

checking for bison version >= 2.5... no

configure: error: Bison version 2.5 or higher must be installed on the system!

wget <http://ftp.gnu.org/gnu/bison/>bison-2.5.1.tar.gz

configure

make

make install

)

Installation

============

If you are building from the first time out of the source repository, you will

need to generate the configure scripts. (This is not necessary if you

downloaded a tarball.) From the top directory, do:

./bootstrap.sh

Once the configure scripts are generated, thrift can be configured.

From the top directory, do:

./configure

You may need to specify the location of the boost files explicitly.

If you installed boost in /usr/local, you would run configure as follows:

./configure --with-boost=/usr/local

Note that by default the thrift C++ library is typically built with debugging

symbols included. If you want to customize these options you should use the

CXXFLAGS option in configure, as such:

./configure CXXFLAGS='-g -O2'

./configure CFLAGS='-g -O2'

./configure CPPFLAGS='-DDEBUG\_MY\_FEATURE'

To enable gcov required options -fprofile-arcs -ftest-coverage enable them:

./configure --enable-coverage

Run ./configure --help to see other configuration options

Please be aware that the Python library will ignore the --prefix option

and just install wherever Python's distutils puts it (usually along

the lines of /usr/lib/pythonX.Y/site-packages/). If you need to control

where the Python modules are installed, set the PY\_PREFIX variable.

(DESTDIR is respected for Python and C++.)

Make thrift:

make

From the top directory, become superuser and do:

make install

Note that some language packages must be installed manually using build tools

better suited to those languages (at the time of this writing, this applies

to Java, Ruby, PHP).

Look for the README.md file in the lib/<language>/ folder for more details on the

installation of each language library package.

Testing

=======

There are a large number of client library tests that can all be run

from the top-level directory.

make -k check

This will make all of the libraries (as necessary), and run through

the unit tests defined in each of the client libraries. If a single

language fails, the make check will continue on and provide a synopsis

at the end.

To run the cross-language test suite, please run:

make cross

This will run a set of tests that use different language clients and

servers.

**(二)安装 thrift**

 yum install automake libtool flex bison pkgconfig gcc-c++ boost-devel libevent-devel zlib-devel python-devel ruby-devel

重新安装: boost\_1\_53\_0

...on 10200th target...

common.copy /usr/local/lib/libboost\_python.a

common.copy /usr/local/lib/libboost\_random.a

common.copy /usr/local/lib/libboost\_serialization.a

common.copy /usr/local/lib/libboost\_wserialization.a

common.copy /usr/local/lib/libboost\_signals.a

common.copy /usr/local/lib/libboost\_prg\_exec\_monitor.a

common.copy /usr/local/lib/libboost\_unit\_test\_framework.a

common.copy /usr/local/lib/libboost\_thread.a

common.copy /usr/local/lib/libboost\_timer.a

common.copy /usr/local/lib/libboost\_wave.a

...updated 10209 targets...

Building Plugin Support ...... : yes

Building C++ Library ......... : yes

Building C (GLib) Library .... : no

Building Java Library ........ : no

Building C# Library .......... : no

Building Python Library ...... : yes

make[3]: \*\*\* [src/thrift/server/libthriftnb\_la-TNonblockingServer.lo] Error 1

**(四)第四次**

yum -y install automake libtool flex bison pkgconfig gcc-c++ boost-devel libevent-devel zlib-devel [**Python**](http://lib.csdn.net/base/python)-devel ruby-devel crypto-utils  
openssl openssl-devel

yum install glibc\* -y

git clone https://git-wip-us.apache.org/repos/asf/thrift.git

./configure --prefix=/usr --with-libevent=/usr/lib64 --with-boost=/usr/local/include

/usr/local/include/boost

./configure --with-boost=/usr/local/include/因为boost安装在/usr/include里？   
非常重要：可能这里有一个关于libtool的错误Version mismatch error. This is libtool 2.2.4, but the libtool: definition of this LT\_INIT comes from libtool 2.2.6. libtool: You should recreate aclocal.m4 with macros from libtool 2.2.4 libtool: and run autoconf again.

/src/thrift/server/TNonblockingServer.h:42:33: error: event2/event\_struct.h: No such file or directory

./src/thrift/server/TNonblockingServer.h:432:31: warning: use of C99 long long integer constant

src/thrift/server/TNonblockingServer.cpp: In member function 'void apache::thrift::server::TNonblockingServer::listenSocket(int)':

src/thrift/server/TNonblockingServer.cpp:1145: warning: dereferencing pointer 'sin' does break strict-aliasing rules

src/thrift/server/TNonblockingServer.cpp:1144: note: initialized from here

src/thrift/server/TNonblockingServer.cpp:1148: warning: dereferencing pointer 'sin' does break strict-aliasing rules

src/thrift/server/TNonblockingServer.cpp:1147: note: initialized from here

make[3]: \*\*\* [src/thrift/server/libthriftnb\_la-TNonblockingServer.lo] Error 1

make[3]: Leaving directory `/app/thrift-0.10.0/lib/cpp'

make[2]: \*\*\* [all-recursive] Error 1

make[2]: Leaving directory `/app/thrift-0.10.0/lib/cpp'

make[1]: \*\*\* [all-recursive] Error 1

make[1]: Leaving directory `/app/thrift-0.10.0'

make: \*\*\* [all] Error 2