北风网 IBEIFENG.COM

大数据Hadoop高薪直通车课程

HBase高级使用

www.ibeifeng.com

讲师：轩宇（北风网版权所有)

课程大纲

1. HBase 表的设计
2. HBase 表属性
3. HBase 表管理
4. 集成 Hive使用
5. HBase 实战案例

www.ibeifeng.com

课程大纲

1. HBase 表的设计
2. HBase 表属性
3. HBase 表管理
4. 集成 Hive使用
5. HBase 实战案例

www.ibeifeng.com

Create Table

```
hbase(main):005:0> create 't1', 'cf'
0 row(s) in 0.4350 seconds

=> Hbase::Table - t1
hbase(main):006:0> describe 't1'
DESCRIPTION                                                          ENABLED
 't1', {NAME => 'cf', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW true
 ', REPLICATION_SCOPE => '0', VERSIONS => '1', COMPRESSION => 'NONE', MI
 N_VERSIONS => '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'false', BLO
 CKSIZE => '65536', IN_MEMORY => 'false', BLOCKCACHE => 'true'}
1 row(s) in 0.0490 seconds
```

www.1beifeng.com

Table Property

't1',
{
NAME => 'cf',
DATA_BLOCK_ENCODING => 'NONE',
BLOOMFILTER => 'ROW',
REPLICATION_SCOPE => '0',
VERSIONS => '1',
COMPRESSION => 'NONE',
MIN_VERSIONS => '0',
TTL => 'FOREVER',
KEEP_DELETED_CELLS => false',
BLOCKSIZE => '65536',
IN_MEMORY => 'false',
BLOCKCACHE => 'true'
}

```
't1',
{
    NAME => 'cf',
    DATA_BLOCK_ENCODING => 'NONE',
    BLOOMFILTER => 'ROW',
    REPLICATION_SCOPE => '0',
    VERSIONS => '1',
    COMPRESSION => 'NONE',
    MIN_VERSIONS => '0',
    TTL => 'FOREVER',
    KEEP_DELETED_CELLS => 'false',
    BLOCKSIZE => '65536',
    IN_MEMORY => 'false',
    BLOCKCACHE => 'true'
}
```

www.ibeifeng.com

Hadoop Native Libraries in HBase

If you see the following in your HBase logs, you know that HBase was unable to locate the Hadoop native libraries:

```
2014-08-07 09:26:20,139 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
builtin-java classes where applicable
```

If the libraries loaded successfully, the WARN message does not show.

Lets presume your Hadoop shipped with a native library that suits the platform you are running HBase on. To check if the Hadoop native library is available to HBase, run the following tool (available in Hadoop 2.1 and greater):

```
$ ./bin/hbase --config ~/conf_hbase org.apache.hadoop.util.NativeLibraryChecker
2014-08-26 13:15:38,717 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
builtin-java classes where applicable
Native library checking:
hadoop: false
zlib:   false
snappy: false
lz4:    false
bzip2:  false
2014-08-26 13:15:38,863 INFO  [main] util.ExitUtil: Exiting with status 1
```

www.ibeifeng.com

HFile Compression

```
$ export HBASE_HOME=/opt/modules/hbase-0.98.6-cdh5.3.3
$ export HADOOP_SNAPPY_HOME=/opt/modules/hadoop-snappy-0.0.1-SNAPSHOT
$ cp $HADOOP_SNAPPY_HOME/lib/hadoop-snappy-0.0.1-SNAPSHOT.jar $HBASE_HOME/lib
$ mkdir $HBASE_HOME/lib/native
$ cp -r $HADOOP_SNAPPY_HOME/lib/native/Linux-amd64-64/* $HBASE_HOME/lib/native
```

$ export HBASE_HOME=/opt/modules/hbase-0.98.6-cdh5.3.6

$ export HADOOP_SNAPPY_HOME=/opt/modules/hadoop-snappy-0.0.1-SNAPSHOT

$ export HADOOP_HOME=/opt/modules/hadoop-2.5.0-cdh5.3.6

$ cp $HADOOP_SNAPPY_HOME/lib/hadoop-snappy-0.0.1-SNAPSHOT.jar $HBASE_HOME/lib

$ mkdir $HBASE_HOME/lib/native

$ ln -s $HADOOP_HOME/lib/native $HBASE_HOME/lib/native/Linux-amd64-64

Restart HBase Cluster

www.ibeifeng.com

HFile Compression

You can use the CompressionTest tool to verify that your compressor is available to HBase:

```
$ hbase org.apache.hadoop.hbase.util.CompressionTest
hdfs://host/path/to/hbase snappy
```

Enforce Compression Settings On a RegionServer

You can configure a RegionServer so that it will fail to restart if compression is configured incorrectly, by adding the option hbase.regionserver.codecs to the *hbase-site.xml*, and setting its value to a comma-separated list of codecs that need to be available. For example, if you set this property to `lzo,gz`, the RegionServer would fail to start if both compressors were not available. This would prevent a new server from being added to the cluster without having codecs configured properly.

www.ibeifeng.com

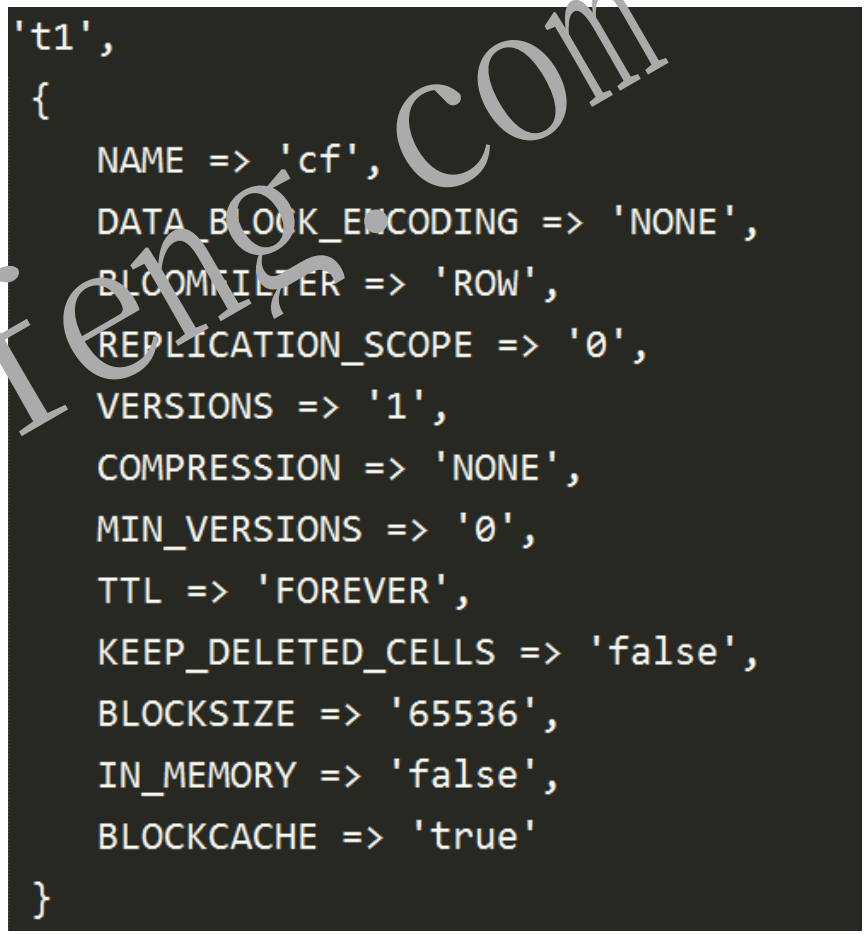
HFile Compression

```
hbase(main):001:0> create 'test2', { NAME => 'cf2', COMPRESSION => 'SNAPPY' }
0 row(s) in 3.4030 seconds

=> Hbase::Table - test2
hbase(main):002:0> describe 'test2'
DESCRIPTION                                                                    ENABLED
 'test2', {NAME => 'cf2', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW',  true
 REPLICATION_SCOPE => '0', VERSIONS => '1', COMPRESSION => 'SNAPPY', MIN_VERSIO
 NS => '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'false', BLOCKSIZE => '6553
 6', IN_MEMORY => 'false', BLOCKCACHE => 'true'}
1 row(s) in 0.2140 seconds

hbase(main):003:0> put 'test2', '10001', 'cf2:name', 'zhangsan'
0 row(s) in 0.2830 seconds
```

www.ibeifeng.com

Table Property

```
't1',
 {
        NAME => 'cf',
        DATA_BLOCK_ENCODING => 'NONE',
        BLOOMFILTER => 'ROW',
        REPLICATION_SCOPE => '0',
        VERSIONS => '1',
        COMPRESSION => 'NONE',
        MIN_VERSIONS => '0',
        TTL => 'FOREVER',
        KEEP_DELETED_CELLS => false',
        BLOCKSIZE => '65536',
        IN_MEMORY => 'false',
        BLOCKCACHE => 'true'
 }
```



www.ibeifeng.com

Table Property

```
/**
 * Default number of versions of a record to keep.
 */
public static final int DEFAULT_VERSIONS = HBaseConfiguration.create().getInt(
  "hbase.column.max.version", 1);

/**
 * Default is not to keep a minimum of versions.
 */
public static final int DEFAULT_MIN_VERSIONS = 0;
```

```
hbase(main):022:0> create 't1', {NAME => 'f1', VERSIONS => 5}
0 row(s) in 0.4020 seconds

=> Hbase::Table - t1
hbase(main):023:0> describe 't1'
DESCRIPTION                                                          ENABLED
 't1', {NAME => 'f1', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW', REPL true
 ICATION_SCOPE => '0', VERSIONS => '5', COMPRESSION => 'NONE', MIN_VERSIONS =>
 '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'false', BLOCKSIZE => '65536', IN
 _MEMORY => 'false', BLOCKCACHE => 'true'}
1 row(s) in 0.0440 seconds
```

get 't1', '100001', {COLUMN => 'f1:name', VERSIONS => 3}

www.ibeifeng.com

Table Property

```
't1',
 {
        NAME => 'cf',
        DATA_BLOCK_ENCODING => 'NONE',
        BLOOMFILTER => 'ROW',
        REPLICATION_SCOPE => '0',
        VERSIONS => '1',
        COMPRESSION => 'NONE',
        MIN_VERSIONS => '0',
        TTL => 'FOREVER',
        KEEP_DELETED_CELLS => false',
        BLOCKSIZE => '65536',
        IN_MEMORY => 'false',
        BLOCKCACHE => 'true'
 }
```

```
't1',
{
    NAME => 'cf',
    DATA_BLOCK_ENCODING => 'NONE',
    BLOOMFILTER => 'ROW',
    REPLICATION_SCOPE => '0',
    VERSIONS => '1',
    COMPRESSION => 'NONE',
    MIN_VERSIONS => '0',
    TTL => 'FOREVER',
    KEEP_DELETED_CELLS => 'false',
    BLOCKSIZE => '65536',
    IN_MEMORY => 'false',
    BLOCKCACHE => 'true'
}
```

www.ibeifeng.com

65.4. Block Cache

Memstore & BlockCache

- HBase上Regionserver的内存分为两个部分，一部分作为Memstore，主要用来写；另外一部分作为BlockCache，主要用于读。
- 写请求会先写入Memstore，Regionserver会给每个region提供一个Memstore，当Memstore满64MB以后，会启动 flush刷新到磁盘。当Memstore的总大小超过限制时（heapsize * hbase.regionserver.global.memstore.upperLimit * 0.9），会强行启动flush进程，从最大的Memstore开始flush直到低于限制。
- 读请求先到Memstore中查数据，查不到就到BlockCache中查，再查不到就会到磁盘上读，并把读的结果放入BlockCache。由于BlockCache采用的是LRU策略，因此BlockCache达到上限(heapsize * hfile.block.cache.size * 0.85)后，会启动淘汰机制，淘汰掉最老的一批数据。
- 在注重读响应时间的应用场景下，可以将 BlockCache设置大些，Memstore设置小些，以加大缓存的命中率。

www.ibeifeng.com

Memstore & BlockCache

属性	值	说明
RegionServer 中所有 Memstore 的最大大小 hbase.regionserver.global.memstore.upperLimit	0.4 默认值	阻止新更新和强迫刷新前，RegionServer 中所有 memstore 的最大大小。
Memstore 刷新的低水位线 hbase.regionserver.global.memstore.lowerLimit	0.38 默认值	当 memstores 被迫刷新以节省内存时，请一直刷新直到达到此数量。如此数量等于“hbase.regionserver.global.memstore.upperLimit”，则由于 memstore 限制阻止更新时，可能会最低限度地进行刷新。
HBase Memstore 刷新大小 hbase.hregion.memstore.flush.size	128 兆字节 默认值	凡 memstore 大小超过此值（字节数），Memstore 将刷新到磁盘。通过运行由 hbase.server.thread.wakefrequency 指定的频率的线程检查此值。

属性	值	说明
HFile 块缓存大小 hfile.block.cache.size	0.4 默认值	用于阻止 HFile/StoreFile 使用的缓存所分配的最大堆（-Xmx 设置）的百分比。要禁用，请将此值设置为 0。

www.ibeifeng.com

BlockCache

- 将Cache分级思想的好处在于：
  - 首先，通过inMemory类型Cache，可以有选择地将in-memory的column families放到RegionServer内存中，例如Meta元数据信息；
  - 通过区分Single和Multi类型Cache，可以防止由于Scan操作带来的Cache频繁颠簸，将最少使用的Block加入到淘汰算法中。
- 默认配置下，对于整个BlockCache的内存，又按照以下百分比分配给Single、Multi、InMemory使用：0.25、0.50和0.25。

```
public class BucketCache implements BlockCache, HeapSize {
  static final Log LOG = LogFactory.getLog(BucketCache.class);

  /** Priority buckets */
  private static final float DEFAULT_SINGLE_FACTOR = 0.25f;
  private static final float DEFAULT_MULTI_FACTOR = 0.50f;
  private static final float DEFAULT_MEMORY_FACTOR = 0.25f;
  private static final float DEFAULT_EXTRA_FREE_FACTOR = 0.10f;
```

其中InMemory队列用于保存HBase Meta表元数据信息，因此如果将数据量很大的用户表设置为InMemory的话，可能会导致Meta表缓存失效，进而对整个集群的性能产生影响。

www.ibeifeng.com

课程大纲

1. HBase 表的设计
2. HBase 表属性
3. HBase 表管理
4. 集成 Hive使用
5. HBase 实战案例

www.ibeifeng.com

HBase Admin

随着 memstore 中的数据不断刷写到磁盘中，会产生越来越多的 HFile 文件，HBase 内部有一个解决这个问题的管家机制，即用合并将多个文件合并成一个较大的文件。合并有两种类型：minor 合并（minor compaction）和 major 压缩合并（majar compaction）。minor 合并将多个小文件重写为数量较少的大文件，减少存储文件的数量，这个过程实际上是个多路归并的过程。因为 HFile 的每个文件都是经过归类的，所以合并速度很快，只受到磁盘 I/O 性能的影响。

major 合并将一个 region 中一个列族的若干个 HFile 重写为一个新 HFile，与 minor 合并相比，还有更独特的功能：major 合并能扫描所有的键/值对，顺序重写全部的数据，重写数据的过程中会略过做了删除标记的数据。断言删除此时生效，例如，对于那些超过版本号限制的数据以及生存时间到期的数据，在重写数据时就不再写入磁盘了。

www.iberfeng.com

HBase Admin

- HRegoin Server上的storefile文件是被后台线程监控的，以确保这些文件保持在可控状态。磁盘上的storefile的数量会随着越来越多的memstore被刷新而变等于越来越多——每次刷新都会生成一个storefile文件。当storefile数量满足一定条件时（可以通过配置参数类调整），会触发文件合并操作——minor compaction，将多个比较小的storefile合并成一个大的storefile文件，直到合并的文件大到超过单个文件配置允许的最大值时会触发一次region的自动分割，即region split操作，将一个region平分成2个。

www.ibeifeng.com

HBase Admin

- **minor compaction，轻量级**

  将符合条件的最早生成的几个storefile合并生成一个大的storefile文件，它不会删除被标记为“删除”的数据和以过期的数据，并且执行过一次minor合并操作后还会有多个storefile文件。

- **major compaction，重量级**

  把所有的storefile合并成一个单一的storefile文件，在文件合并期间系统会删除标记为"删除"标记的数据和过期失效的数据，同时会block所有客户端对该操作所属的region的请求直到合并完毕，最后删除已合并的storefile文件。

www.ibeifeng.com

HBase Admin

- HBase Master Web UI
- Using HBase Shell to manage tables
- Using HBase Shell to access data in HBase
- Using HBase Shell to manage the cluster
- Executing Java methods from HBase Shell
- Row counter
- WAL tool—manually splitting and dumping WALs
- HFile tool—viewing textualized HFile content
- HBase `hbck`—checking the health of an HBase cluster

www.iberfeng.com

Manage the Cluster

1. Flush all regions in the table using the `flush` command:

```
hbase> flush 'hly_temp'
```

2. You can also flush an individual region by passing the region name to the `flush` command:

```
hbase> flush 'hly_temp,,1324174482248.
e3d9b9952973964f3d8e61e191924698.'
```

You can find the region's name under the **Table Regions** section of the table's administration page:

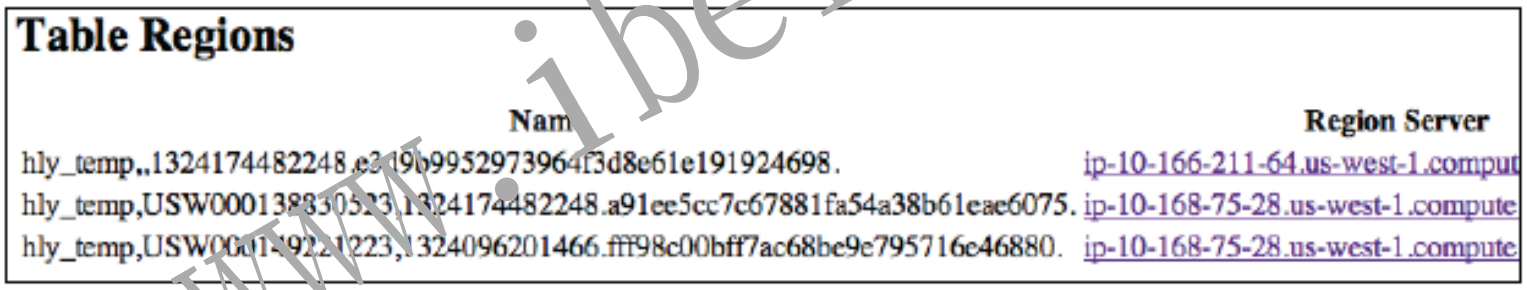


Table Regions

Name	Region Server
hly_temp,,1324174482248.e3d9b9952973964f3d8e61e191924698.	ip-10-166-211-64.us-west-1.comput
hly_temp,USW00013830523,1324174482248.a91ee5cc7c67881fa54a38b61eae6075.	ip-10-168-75-28.us-west-1.compute
hly_temp,USW00013922223,1324096201466.fff98c00bff7ac68be9e795716e46880.	ip-10-168-75-28.us-west-1.compute

3. Compact all the regions in a table by running the `compact` command:

```
hbase> compact 'hly_temp'
```

4. Run a major compaction on a table by running the `major_compact` command:

```
hbase> major_compact 'hly_temp'
```

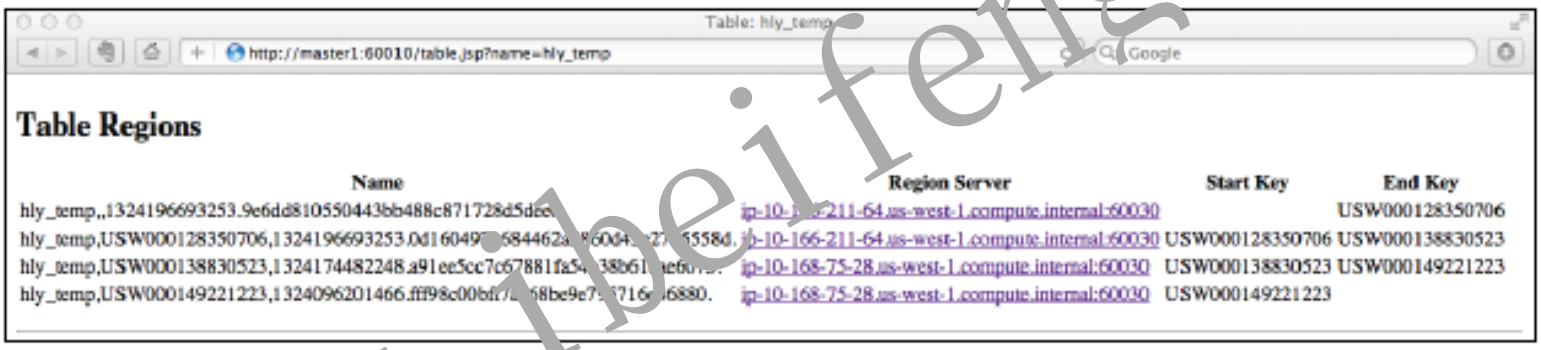
www.ibeifeng.com

Manage the Cluster

5. Split a region in a table by running the `split` command:

```
hbase> split 'hly_temp,,1324174482248.
e3d9b9952973964f3d8e61e191924698.'
```

In the table's administration page, you will find that the region has been split into two regions, so the total region count becomes four:



6. Use the `balance_switch` command to enable the balancer:

```
hbase> balance_switch true
false
```

The output (`false`) is the previous balancer state.

www.ibeifeng.com

Manage the Cluster

7. Balance the load of the cluster by using the `balancer` command:

```
hbase> balancer
true
```

The output `true` indicates that a balancing call has been triggered successfully. It will run in the background on the master server.

8. Move a region to a specific region server by using the `move` command:

```
hbase> move 'e3d91995973964f3d8e61e191924698', 'ip-10-168-75-28.
us-west-1.compute.internal:60030:1324190304563'
```

www.beifeng.com

HBase hbck

1. Check the health of the cluster with the default `hbck` command option:

   ```
   $ $HBASE_HOME/bin/hbase hbck
   ```

   You will get the following output:

   At the end of the command's output it prints **Status: OK**, which indicates the cluster is in consistent status.

www.ibeifeng.com

HBase hbck

1. Enter HBase Shell by typing the following command:

```
$ $HBASE_HOME/bin/hbase shell
```

2. Manually close a region using the `close_region` command:

```
hbase> close_region 'hly_temp,,1324196693253.9e6dd810550443bb488c871728d5dee0.'
```

Replace the last parameter with your region name, which can be found on the HBase web UI.

3. Run `hbck` again; you will find that, at the end of its output it reports the status of the cluster as inconsistent:

```
$ $HBASE_HOME/bin/hbase hbck

ERROR: Region hly_temp,,1324196693253.9e6dd810550443bb488c871728d5dee0. not deployed on any region server.

ERROR: (region hly_temp,U...00128350706,1324196693253.0d1604971684462a2860d43e2715...58d. First region should start with an empty key.

ERROR: Found inconsistency in table hly_temp

...

... inconsistencies detected.

Status: INCONSISTENT
```

4. Use `hbck` with the `-fix` option, to fix the inconsistencies:

```
$ $HBASE_HOME/bin/hbase hbck -fix

ERROR: Region hly_temp,,1324196693253.9e6dd810550443bb488c871728d5dee0. not deployed on any region server.

Trying to fix unassigned region...
```

www.ibeifeng.com

课程大纲

1. HBase 表的设计
2. HBase 表属性
3. HBase 表管理
4. 集成 Hive使用
5. HBase 实战案例

www.ibeifeng.com

Hive HBase Integration

- Hive HBase Integration
  - Introduction
  - Storage Handlers
  - Usage
  - Column Mapping
    - Multiple Columns and Families
    - Hive MAP to HBase Column Family
    - Illegal: Hive Primitive to HBase Column Family
    - Example with Binary Columns
    - Simple Composite Row Keys
    - Complex Composite Row Keys and HBaseKeyFactory
  - Put Timestamps
  - Key Uniqueness
  - Overwrite
  - Potential Followups
  - Build
  - Tests
  - Links
  - Acknowledgements
  - Open Issues (JIRA)

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

www.ibeifeng.com

Version information
As of Hive 0.9.0 the HBase integration requires at least HBase 0.92, earlier versions of Hive were working with HBase 0.89/0.90

Usage

- The storage handler is built as an independent module, **hive-hbase-handler-x.y.z.jar**, which must be available on the Hive client auxpath, along with **HBase, Guava and ZooKeeper jars**.
- It also requires the correct configuration property to be set in order to **connect to the right HBase master**.

www.ibeifeng.com

Usage

```
// Here's an example using CLI , targeting a single-node HBase server.
$HIVE_SRC/build/dist/bin/hive \
    --auxpath \
        $HIVE_SRC/build/dist/lib/hive-hbase-handler-0.9.0.jar,\
        $HIVE_SRC/build/dist/lib/hbase-0.92.0.jar,\
        $HIVE_SRC/build/dist/lib/zookeeper-3.3.4.jar,\
        $HIVE_SRC/build/dist/lib/guava-r09.jar \
    --hiveconf hbase.master=hbase.yoyodyne.com:60000

// Here's an example which instead targets a distributed HBase cluster
// where a quorum of 3 zookeepers is used to elect the HBase master:
$HIVE_SRC/build/dist/bin/hive \
    --auxpath \
        $HIVE_SRC/build/dist/lib/hive-hbase-handler-0.9.0.jar,\
        $HIVE_SRC/build/dist/lib/hbase-0.92.0.jar,\
        $HIVE_SRC/build/dist/lib/zookeeper-3.3.4.jar,\
        $HIVE_SRC/build/dist/lib/guava-r09.jar \
    --hiveconf hbase.zookeeper.quorum=zk1.yoyodyne.com,zk2.yoyodyne.com,zk3.yoyodyne.com
```

www.ibeifeng.com

Usage

```
ln -s $HBASE_HOME/lib/hbase-common-0.98.6-cdh5.3.3.jar $HIVE_HOME/lib/hbase-common-0.98.6-cdh5.3.3.jar
ln -s $HBASE_HOME/lib/hbase-server-0.98.6-cdh5.3.3.jar $HIVE_HOME/lib/base-server-0.98.6-cdh5.3.3.jar
ln -s $HBASE_HOME/lib/hbase-client-0.98.6-cdh5.3.3.jar $HIVE_HOME/lib/hbase-client-0.98.6-cdh5.3.3.jar
ln -s $HBASE_HOME/lib/hbase-protocol-0.98.6-cdh5.3.3.jar $HIVE_HOME/lib/hbase-protocol-0.98.6-cdh5.3.3.jar
ln -s $HBASE_HOME/lib/hbase-it-0.98.6-cdh5.3.3.jar $HIVE_HOME/lib/hbase-it-0.98.6-cdh5.3.3.jar
ln -s $HBASE_HOME/lib/htrace-core-2.04.jar $HIVE_HOME/lib/htrace-core-2.04.jar
```

ln -s $HBASE_HOME/lib/hbase-common-0.98.6-cdh5.3.6.jar $HIVE_HOME/lib/hbase-common-0.98.6-cdh5.3.6.jar
ln -s $HBASE_HOME/lib/hbase-server-0.98.6-cdh5.3.6.jar $HIVE_HOME/lib/base-server-0.98.6-cdh5.3.6.jar
ln -s $HBASE_HOME/lib/hbase-client-0.98.6-cdh5.3.6.jar $HIVE_HOME/lib/hbase-client-0.98.6-cdh5.3.6.jar
ln -s $HBASE_HOME/lib/hbase-protocol-0.98.6-cdh5.3.6.jar $HIVE_HOME/lib/hbase-protocol-0.98.6-cdh5.3.6.jar
ln -s $HBASE_HOME/lib/hbase-it-0.98.6-cdh5.3.6.jar $HIVE_HOME/lib/hbase-it-0.98.6-cdh5.3.6.jar
ln -s $HBASE_HOME/lib/htrace-core-2.04.jar $HIVE_HOME/lib/htrace-core-2.04.jar

www.ibeifeng.com

Usage

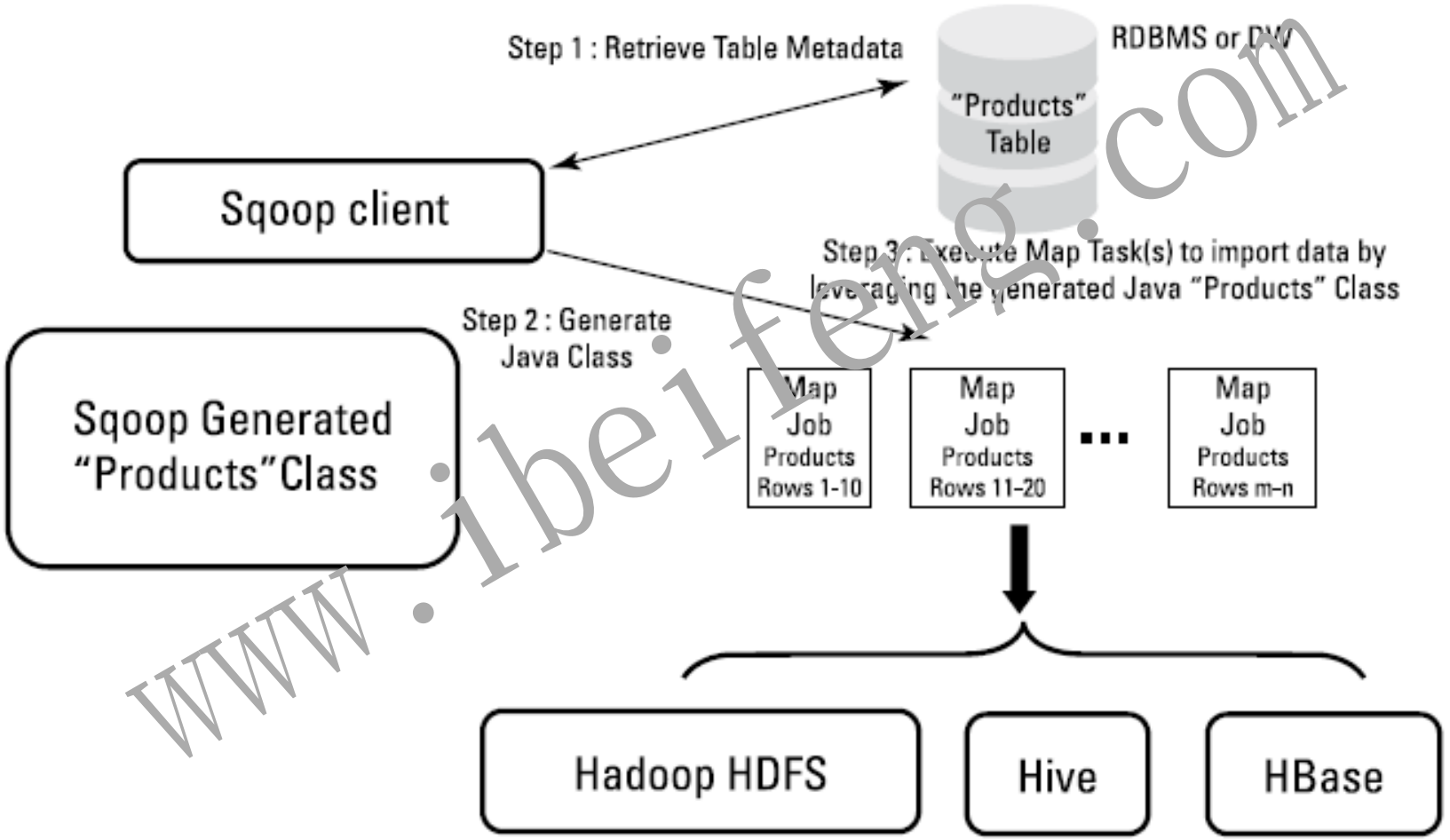
In order to create a new HBase table which is to be managed by Hive, use the `STORED BY` clause on `CREATE TABLE`:

```
CREATE TABLE hbase_table_1(key int, value string)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,cf1:val")
TBLPROPERTIES ("hbase.table.name" = "xyz");
```

The `hbase.columns.mapping` property is required and will be explained in the next section. The `hbase.table.name` property is optional; it controls the name of the table as known by HBase, and allows the Hive table to have a different name. In this example, the table is known as `hbase_table_1` within Hive, and as `xyz` within HBase. If not specified, then the Hive and HBase table names will be identical.

www.ibeifeng.com

The Sqoop import flow of execution



www.ibeifeng.com

HBase & Sqoop

```
age: sqoop import [GENERIC-ARGS] [TOOL-ARGS]
Common arguments:
   --connect <jdbc-uri>         Specify JDBC connect string
   --driver <class-name>        Manually specify JDBC driver class to use
   --password <password>        Set authentication password
   --username <username>        Set authentication username

Import control arguments:
   --table <table-name>         Table to read

HBase arguments:
   --column-family <family>     Sets the target column family for the import
   --hbase-create-table         If specified, create missing HBase tables
   --hbase-table <table>        Import to <table> in HBase
   --hbase-row-key <col>        Specifies which input column to use as the row key
   --hbase-bulkload             Enables HBase bulk loading
```

www.ibeifeng.com

HBase & Sqoop

```
$ sqoop import \
    --connect jdbc:mysql://localhost/serviceorderdb \
    --username root -P \
    --table customercontactinfo \
    --columns "customernum,contactinfo" \
    --hbase-table customercontactinfo \
    --column-family ContactInfo \
    --hbase-row-key customernum -m 1
```

```
$ sqoop import \
    --connect jdbc:mysql://localhost/serviceorderdb \
    --username root -P \
    --table customercontactinfo \
    --columns "customernum,customername" \
    --hbase-table customercontactinfo \
    --column-family CustomerName \
    --hbase-row-key customernum -m 1
```

```
$ sqoop import \
    --connect jdbc:mysql://localhost/serviceorderdb \
    --username root -P \
    --table customercontactinfo \
    --columns "customernum,productnums" \
    --hbase-table customercontactinfo \
    --column-family ProductNums \
    --hbase-row-key customernum -m 1
```

www.iberfeng.com

HBase & Hue

The Thrift framework is provided by a Thrift server, which provides a way for scalable flexibility and interoperability across computer languages and services development. It builds an engine with the help of code generation, which works efficiently between HBase and C++, Java, Python, PHP, Ruby, Perl, and so on.

This service is provided by HBase using the gthrough package:

```
org.apache.hadoop.hbase.thrift;
```

The Thrift service can be started like this:

```
bin/hbase-daemon.sh start thrift
```

And can be stopped like this:

```
bin/hbase-daemon.sh stop thrift
```

www.ibeifeng.com

HBase & Hue

```
[hbase]
  # Comma-separated list of HBase Thrift servers for clusters in the format of '(name|host:port)'
  # Use full hostname with security.
  hbase_clusters=(Cluster|hadoop-ehp01.cloudyhadoop.com:9090)

  # HBase configuration directory, where hbase-site.xml is located.
  hbase_conf_dir=/opt/modules/hbase-0.98.6-cdh5.3.3/conf

  # Hard limit of rows or columns per row fetched before truncating.
  ## truncate_limit = 500

  # 'buffered' is the default of the HBase Thrift Server and supports security.
  # 'framed' can be used to chunk up responses,
  # which is useful when used in conjunction with the nonblocking server in Thrift.
  ## thrift_transport=buffered
```

www.ibeifeng.com

课程大纲

1. HBase 表的设计
2. HBase 表属性
3. HBase 表管理
4. 集成 Hive使用
5. HBase 实战案例

www.ibeifeng.com

Fully-distributed

Distributed mode can be subdivided into distributed but all daemons run on a single node—a.k.a *pseudo-distributed*—and *fully-distributed* where the daemons are spread across all nodes in the cluster. The *pseudo-distributed* vs. *fully-distributed* nomenclature comes from Hadoop.

Pseudo-distributed mode can run against the local filesystem or it can run against an instance of the *Hadoop Distributed File System* (HDFS). Fully-distributed mode can ONLY run on HDFS.

www.ibeifeng.com

Fully-distributed

By default, HBase runs in standalone mode. Both standalone mode and pseudo-distributed mode are provided for the purposes of small-scale testing. For a production environment, distributed mode is appropriate. In distributed mode, multiple instances of HBase daemons run on multiple servers in the cluster.

Just as in pseudo-distributed mode, a fully distributed configuration requires that you set the `hbase.cluster.distributed` property to `true`. Typically, the `hbase.rootdir` is configured to point to a highly-available HDFS filesystem.

In addition, the cluster is configured so that multiple cluster nodes enlist as RegionServers, ZooKeeper QuorumPeers, and backup HMaster servers. These configuration basics are all demonstrated in quickstart-fully-distributed.

www.ibeifeng.com

Fully-distributed

Table 1. Distributed Cluster Demo Architecture

Node Name	Master	ZooKeeper	RegionServer
node-a.example.com	yes	yes	no
node-b.example.com	backup	yes	yes
node-c.example.com	no	yes	yes

www.ibeifeng.com

Distributed RegionServers

Distributed RegionServers

Typically, your cluster will contain multiple RegionServers all running on different servers, as well as primary and backup Master and Zookeeper daemons. The *conf/regionservers* file on the master server contains a list of hosts whose RegionServers are associated with this cluster. Each host is on a separate line. All hosts listed in this file will have their RegionServer processes started and stopped when the master server starts or stops.

www.ibeifeng.com

ZooKeeper and HBase

This is a bare-bones *conf/hbase-site.xml* for a distributed HBase cluster. A cluster that is used for real-world work would contain more custom configuration parameters. Most HBase configuration directives have default values, which are used unless the value is overridden in the *hbase-site.xml*. See "Configuration Files" for more information.

```
<configuration>
  <property>
    <name>hbase.rootdir</name>
    <value>hdfs://namenode.example.org:8020/hbase</value>
  </property>
  <property>
    <name>hbase.cluster.distributed</name>
    <value>true</value>
  </property>
  <property>
    <name>hbase.zookeeper.quorum</name>
    <value>node-a.example.com,node-b.example.com,node-c.example.com</value>
  </property>
</configuration>
```

www.ibeifeng.com

ZooKeeper and HBase

This is an example *conf/regionservers* file, which contains a list of nodes that should run a RegionServer in the cluster. These nodes need HBase installed and they need to use the same contents of the *conf/* directory as the Master server

```
node-a.example.com
node-b.example.com
node-c.example.com
```

This is an example *conf/backup-masters* file, which contains a list of each node that should run a backup Master instance. The backup Master instances will sit idle unless the main Master becomes unavailable.

```
node-b.example.com
node-c.example.com
```

www.ibeifeng.com

项目背景

- 各大电商，订单查询
  - 订单数据量大
  - 客户实时查询
- 数据库
  - RDBMS，无法满足海量数据实时查询
  - NoSQL，海量数据存储准实时查询 — **HBase**

www.ibeifeng.com

【京东】订单查询

www.ibeifeng.com

【一号店】订单查询

订单号4107131766952

包裹1　包裹2 已完成

包裹状态：已完成

包裹金额：¥22.63

配送方式：普通快递(1个快递箱)

物流信息：配送公司（南京晟邦）　配送单号：1484203613　联系电话：400--6666--066

包裹跟踪

提交订单 2015-05-30 10:54:16

包裹出库 2015-05-30 13:38:29

包裹送达 2015-06-06 18:49:26

2015-06-06 18:49:13　订单付款成功

2015-06-06 18:45:18　配送成功

[订单配送成功]欢迎您下次光临，祝您生活愉快！

手机跟APP

www.ibeifeng.com

业务分析

- 维度查询
  - 用户号（user_id）
  - 时间限制（默认值，create_time）
- 查询页
  - 订单显示页（首页、订单概要信息）
  - 订单详情页（订单相关详细信息）

www.ibeifeng.com

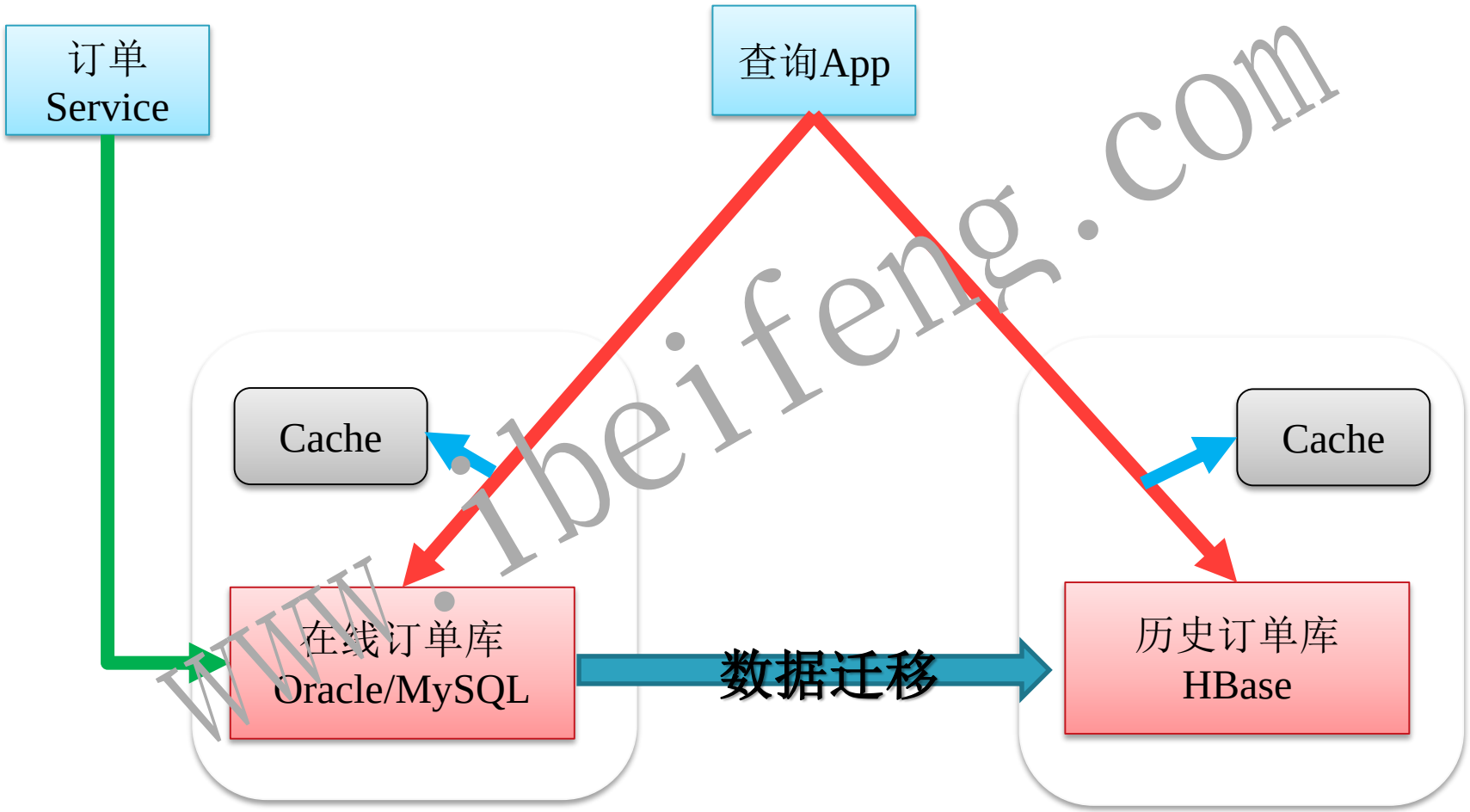
项目分析

- 订单类型
  - 在线订单（近期订单）
  - 历史订单
- 数据分开存储优势
  - 分开存储,减轻在线订单库压力
  - 统一各个系统对历史订单的查询
  - 灵活支撑各种维度历史订单的条件查询
- 存储技术

  RDBMS（Oracle RAC/MySQL RAC + HBase）

www.ibeifeng.com

订单数据技术架构



www.1beifeng.com

订单表设计

- 订单显示表
  - * 表名称：Order_Summary
  - * 列簇名：orderInfo
  - * rowkey：userId_orderCreateTime_orderId
- 订单详情查询
  - * 表名称：Order_Detail
  - * 列簇名：itemInfo
  - * rowkey：orderId_orderItemId
- 订单编码索引表
  - * 表名称：Index_Order_Id_Code
  - * 列簇名：order
  - * rowkey：orderCode
  - * column：orderId

www.1beifeng.com

本课程版权归北风网所有

欢迎访问我们的官方网站

www.ibeifeng.com

www.ibeifeng.com