29 HBase

HBase
– Basic Concepts
– HBase Scheme Design
– Other Issues

目标：能够根据大尺寸非结构化和半结构化数据存储与分析的要求，设计并实现基于HBase的数据存储与分析方案

基本概念

- **HBase – Hadoop Database**，是一个高可靠性、高性能、面向列、可伸缩的**分布式存储系统**。

  HBase是一个分布式的、面向列的开源数据库，该技术来源于 Fay Chang 所撰写的Google论文“Bigtable：一个结构化数据的分布式存储系统”。就像Bigtable利用了Google文件系统（File System）所提供的分布式数据存储一样，HBase在Hadoop之上提供了类似于Bigtable的能力。

  HBase是Apache的Hadoop项目的子项目。HBase不同于一般的关系数据库，它是一个适合于非结构化数据存储的数据库。

- 列族的概念：列族有几个关联比较大的列组成，里面的数据是一起存储的，不同的列族可以分开来存

  列的命名： station : identifier

  每一行都有一个独特的id，按id顺序存储

- 另一个不同的是HBase基于列的而不是基于行的模式

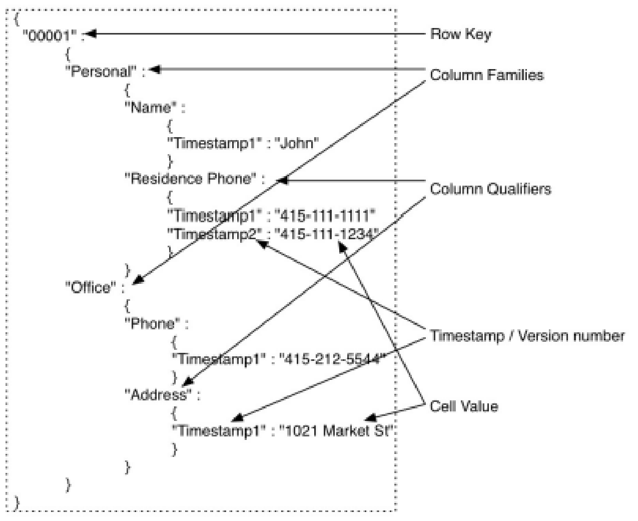
HBase is a distributed column-oriented database built on top of HDFS.

- HBase is the Hadoop application to use when you require real-time read/write random-access to very large datasets.
- HBase comes at the scaling problem from the opposite direction.
  - It is built from the ground-up to scale linearly just by adding nodes.
- HBase is not relational and does not support SQL, but given the proper problem space,
  - it is able to do what an RDBMS cannot: host very large, sparsely populated tables on clusters made from commodity hardware.
- The canonical HBase use case is the webtable, a table of crawled web pages and their attributes (such as language and MIME type) keyed by the web page URL.
  - The webtable is large, with row counts that run into the billions.

- 按列存：考虑到一个列的额数据相似，编码机制和压缩机制会更高效。
- 非结构化存储：会有空的地方——稀疏矩阵，但在实际存储的时候这些地方不会空着

The following represents the same information as a multi-dimensional map.

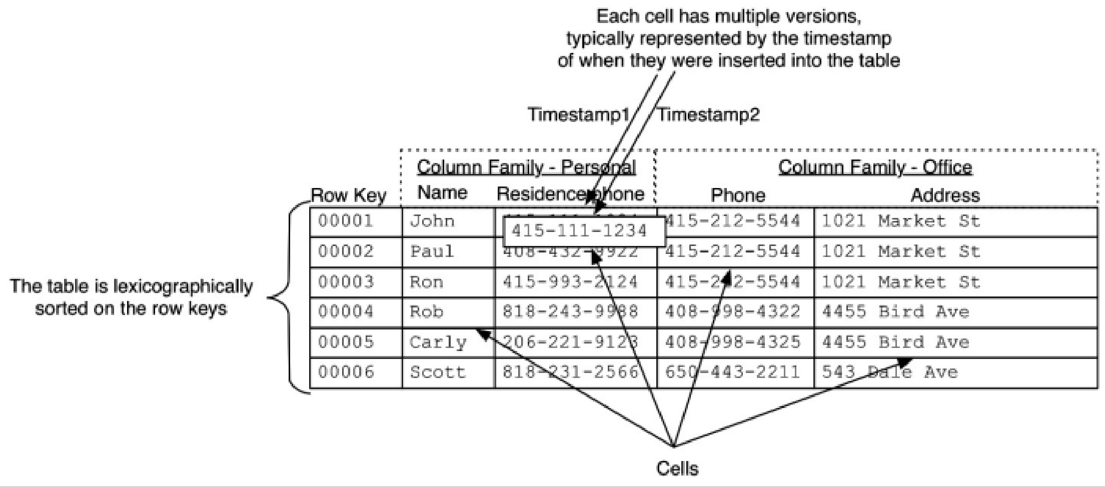
```
{
 "com.cnn.www": {
   contents: {
     t6: contents:html: "<html>..."
     t5: contents:html: "<html>..."
     t3: contents:html: "<html>..."
   }
   anchor: {
     t9: anchor:cnnsi.com = "CNN"
     t8: anchor:my.look.ca = "CNN.com"
   }
   people: {}
 }
 "com.example.www": {
   contents: {
     t5: contents:html: "<html>..."
   }
   anchor: {}
   people: {
     t5: people:author: "John Doe"
   }
  }
}
```



26

- 元数据管理简单，表格中数据太大的时候会水平分割成两个region分布式存储，这样也方便version管理

Cells in this table that appear to be empty do not take space, or in fact exist, in HBase. This is what makes HBase "sparse."


Each cell has multiple versions, typically represented by the timestamp of when they were inserted into the table
Timestamp1
Timestamp2
The table is lexicographically sorted on the row keys
Cells

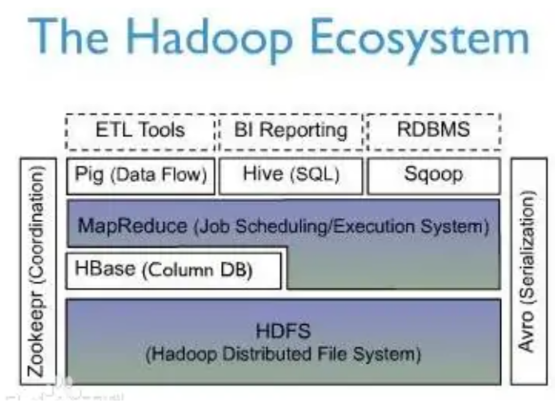
Row Key	Column Family - Personal		Column Family - Office	
	Name	Residence phone	Phone	Address
00001	John	415-111-1234	415-212-5544	1021 Market St
00002	Paul	408-432-9922	415-212-5544	1021 Market St
00003	Ron	415-993-2124	415-212-5544	1021 Market St
00004	Rob	818-243-9988	408-998-4322	4455 Bird Ave
00005	Carly	206-221-9123	408-998-4325	4455 Bird Ave
00006	Scott	818-231-2566	650-443-2211	543 Dale Ave

特点

Features

– Linear and modular scalability.
– Strictly consistent reads and writes.
– Automatic and configurable sharding of tables
– Automatic failover support between RegionServers.
– Convenient base classes for backing Hadoop MapReduce jobs with Apache HBase tables.
– Easy to use Java API for client access.
– Block cache and Bloom Filters for real-time queries.
– Query predicate push down via server side Filters
– Thrift gateway and a REST-ful Web service that supports XML, Protobuf, and binary data encoding options
– Extensible jruby-based (JIRB) shell
– Support for exporting metrics via the Hadoop metrics subsystem to files or Ganglia; or via JMX

左图描述Hadoop EcoSystem中的各层系统。其中,HBase位于结构化存储层，Hadoop HDFS为HBase提供了高可靠性的底层存储支持，Hadoop MapReduce为HBase提供了高性能的计算能力，Zookeeper为HBase提供了稳定服务和failover机制。



1.面向列：Hbase是面向列的存储和权限控制，并支持独立索引。列式存储，其数据在表中是按照某列存储的，这样在查询只需要少数几个字段时，能大大减少读取的数据量。

2.多版本：Hbase每一个列的存储有多个Version。

3.稀疏性：为空的列不占用存储空间，表可以设计得非常稀疏。

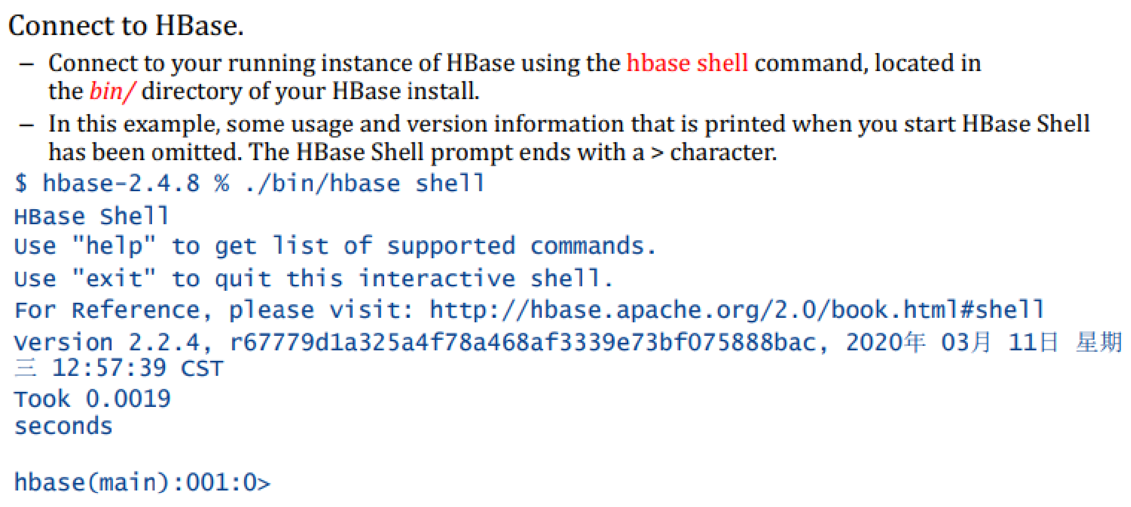
4.扩展性：底层依赖HDFS。

5.高可靠性：WAL机制保证了数据写入时不会因集群异常而导致写入数据丢失，Replication机制保证了在集群出现严重的问题时，数据不会发生丢失或损坏。而且Hbase底层使用HDFS，HDFS本身也有备份。

6.高性能：底层的LSM数据结构和Rowkey有序排列等架构上的独特设计，使得Hbase具有非常高的写入性能。region切分，主键索引和缓存机制使得Hbase在海量数据下具备一定的随机读取性能，该性能真对Rowkey的查询能到达到毫秒级别。

HBase的使用

不同于关系型数据库



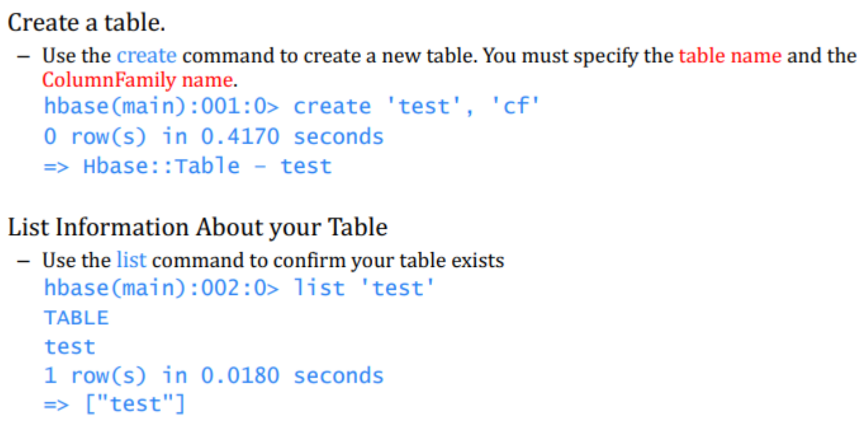
Connect to HBase.

- Connect to your running instance of HBase using the hbase shell command, located in the *bin/* directory of your HBase install.
- In this example, some usage and version information that is printed when you start HBase Shell has been omitted. The HBase Shell prompt ends with a > character.

```
$ hbase-2.4.8 % ./bin/hbase shell
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
For Reference, please visit: http://hbase.apache.org/2.0/book.html#shell
Version 2.2.4, r67779d1a325a4f78a468af3339e73bf075888bac, 2020年 03月 11日 星期
三 12:57:39 CST
Took 0.0019
seconds

hbase(main):001:0>
```

▼ 列族的数量尽量不要超过3个！



Create a table.

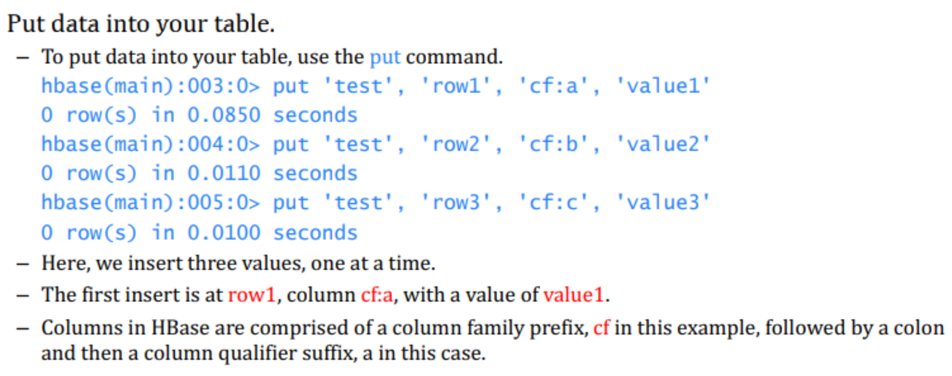
- Use the create command to create a new table. You must specify the table name and the ColumnFamily name.

```
hbase(main):001:0> create 'test', 'cf'
0 row(s) in 0.4170 seconds
=> Hbase::Table - test
```

List Information About your Table

- Use the list command to confirm your table exists

```
hbase(main):002:0> list 'test'
TABLE
test
1 row(s) in 0.0180 seconds
=> ["test"]
```



Put data into your table.

- To put data into your table, use the put command.

```
hbase(main):003:0> put 'test', 'row1', 'cf:a', 'value1'
0 row(s) in 0.0850 seconds
hbase(main):004:0> put 'test', 'row2', 'cf:b', 'value2'
0 row(s) in 0.0110 seconds
hbase(main):005:0> put 'test', 'row3', 'cf:c', 'value3'
0 row(s) in 0.0100 seconds
```

- Here, we insert three values, one at a time.
- The first insert is at row1, column cf:a, with a value of value1.
- Columns in HBase are comprised of a column family prefix, cf in this example, followed by a colon and then a column qualifier suffix, a in this case.

常用操作：

Get

– Get returns attributes for a specified row. Gets are executed via Table.get

Put

– Put either adds new rows to a table (if the key is new) or can update existing rows (if the key already exists). Puts are executed via Table.put (non-writeBuffer) or Table.batch (non-writeBuffer)

Scans

– Scan allow iteration over multiple rows for specified attributes.

Delete

– Delete removes a row from a table. Deletes are executed via Table.delete.

Version管理

- A *{row, column, version}* tuple exactly specifies a cell in HBase.
  - It's possible to have an unbounded number of cells where the row and column are the same but the cell address differs only in its version dimension.
  - The HBase version dimension is stored in decreasing order, so that when reading from a store file, the most recent values are found first.
- Specifying the Number of Versions to Store
  - The maximum number of versions to store for a given column is part of the column schema and is specified at table creation, or via an alter command, via HColumnDescriptor.DEFAULT_VERSIONS.

  - *Modify the Maximum Number of Versions for a Column Family*
  - `hbase> alter ‘t1′, NAME => ‘f1′, VERSIONS => 5`
  - *Modify the Mimimum Number of Versions for a Column Family*
  - `hbase> alter ‘t1′, NAME => ‘f1′, MIN_VERSIONS => 5`

HBase和RDBMS的对比

- **HBase是一种分布式、面向列的数据存储系统。**

– 表模式反映了物理存储，为高效的数据结构序列化、存储和检索创建了一个系统

– 应用程序开发人员有责任以正确的方式使用此存储和检索

- **典型的RDBMS**

– 具有ACID属性和复杂SQL查询引擎的固定模式、面向行的数据库

– 重点在于强大的一致性、引用完整性、物理层的抽象以及通过SQL语言进行的复杂查询

– 您可以轻松创建二级索引，执行复杂的内部和外部联接，跨多个表、行和列对数据进行计数、求和、排序、分组和分页

以下是RDBMS与HBase之间的重要区别。

序号	键	关系数据库管理系统	HBase的
1 个	定义	RDBMS stands for Relational DataBase Management System.	HBase没有完整格式。
2	的SQL	RDBMS requires SQL, Structured Query Language.	HBase不需要SQL。
3	架构图	RDBMS has a fixed schema.	HBase没有固定的架构。
4	方向	RDBMS is row oriented.	HBase是面向列的。
5	可伸缩性	RDBMS faces problems in scalablity.	HBase具有高度可扩展性。
6	性质	DBMS is static in nature.	HBase本质上是动态的。
7	资料检索	RDBMS data retrieval is slow.	HBase数据检索速度很快。
8	规则	RDBMS follws ACID(Atomicity, Consistency, Isolation and Durability) Rule.	HBase遵循CAP（一致性，可用性，分区容忍）规则。
9	数据结构	RDBMS handles structural data.	HBase处理结构，非结构和半结构数据。
10	稀疏数据处理	Sparse data handling is not present.	存在稀疏数据处理。