Introduction to HBase

Concept, Data Model and Architecture

趨勢科技研發實驗室

Public 2009/5/13

Outline

- Introduction to HBase
- HBase Data Model
- HBase Architecture
- Reference

Copyright 2009 - Trend Micro Inc.

Introduction to HBase

- HBase is …
  - Distributed storage
  - Table-like in data structure (multi-dimensional map)
  - High scalability
  - High availability
  - High performance
- HBase provides Google BigTable-like capabilities
- A distributed data store that can scale horizontally to 1,000s of commodity servers and petabytes of indexed storage.
- Designed to operate on top of the Hadoop distributed file system (HDFS) or Kosmos File System (KFS, aka Cloudstore) for scalability, fault tolerance, and high availability.
- Integrated into the Hadoop map-reduce platform and paradigm.

Copyright 2009 - Trend Micro Inc.

History of HBase

- 2006.11
  - Google releases paper on BigTable
- 2007.2
  - Initial HBase prototype created as Hadoop contrib.
- 2007.10
  - First useable HBase
- 2008.1
  - Hadoop become Apache top-level project and HBase becomes subproject
- 2008.10~
  - HBase 0.18, 0.19 released

Copyright 2009 - Trend Micro Inc.

HBase Is (Not) …

- Not a relational data storage system
  - Tables have one primary index, the *row key*.
  - No join operators.
  - Scans and queries can select a subset of available columns, perhaps by using a wildcard.
  - There are three types of lookups:
    - Fast lookup using row key and optional timestamp.
    - Full table scan
    - Range scan from region start to end.
  - Limited atomicity and transaction support.
    - HBase supports multiple batched mutations of single rows only.
    - Data is unstructured and untyped.
  - No accessed or manipulated via SQL.
    - Programmatic access via Java, REST, or Thrift APIs.
    - Scripting via JRuby.

Copyright 2009 - Trend Micro Inc.

Why Bigtable?

- Performance of RDBMS system is good for transaction processing but for very large scale analytic processing, the solutions are commercial, expensive, and specialized.
- Very large scale analytic processing
  - Big queries – typically range or table scans.
  - Big databases (100s of TB)
- Map reduce on Bigtable with optionally Cascading on top to support some relational algebras may be a cost effective solution.
- Sharding is not a solution to scale open source RDBMS platforms
  - Application specific
  - Labor intensive (re)partitionaing

Copyright 2009 - Trend Micro Inc.

Why HBase?

- HBase is a Bigtable clone.
- It is open source
- It has a good community and promise for the future
- It is developed on top of and has good integration for the Hadoop platform, if you are using Hadoop already.
- It has a Cascading connector.

Copyright 2009 - Trend Micro Inc.

Data Model – Conceptual View

- Tables are sorted by Row
- Table schema only define it's *column families* .
  - Each family consists of any number of columns
  - Each column consists of any number of versions
  - Columns only exist when inserted, NULLs are free.
  - Columns within a family are sorted and stored together
  - Everything except table names are byte[]
- (Row, Family: Column, Timestamp) → Value

Row Key	Time Stamp	Column (Family) "content:"	Column (Family) "anchor:"	
com.cnn.www	t9	"<html>…"	"anchor:cnnsi.com"	"CNN"
	t8		"anchor:cnnsi.com" "anchor:my.lock.ca"	"CNN" "MyLook"
	t6	"<html>…"		

Copyright 2009 - Trend Micro Inc.

Data Model – Conceptual View

- Conceptually a table may be thought of a collection of rows that are located by a row key (and optional timestamp) and where any column may not have a value for particular row key (sparse)

Row Key	Time Stamp	Column (Family) "content:"	Column (Family) "anchor:"	
com.cnn.www	t9	"<html>…"	"anchor:cnnsi.com"	"CNN"
	t8		"anchor:cnnsi.com" "anchor:my.lock.ca"	"CNN" "MyLook"
	t6	"<html>…"		

Copyright 2009 - Trend Micro Inc.

Data Model – Physical Storage

- Although, at a conceptual level, tables may be viewed as a sparse set of rows, physically they are stored as column oriented data sets.

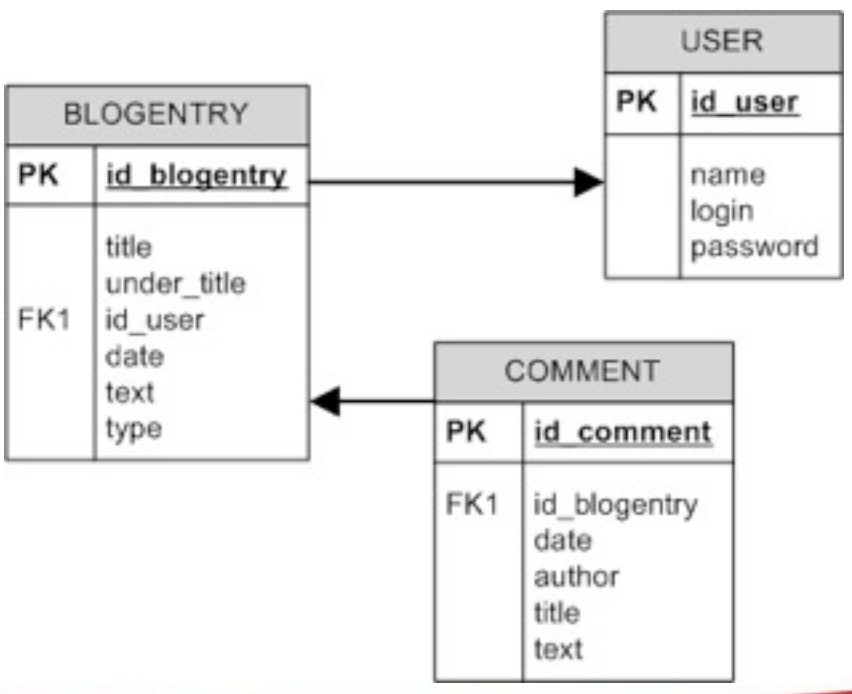
Row Key	Time Stamp	Column (Family) "content:"
com.cnn.www	t9	"<html>…"
	t6	"<html>…"

Row Key	Time Stamp	Column (Family) "anchor:"	
com.cnn.www	t9	"anchor:cnnsi.com"	"CNN"
	t8	"anchor:cnnsi.com" "anchor:my.loc	"CNN" "MyLook"

Copyright 2009 - Trend Micro Inc.

Data Model – Real Life Example

- Consider
  - Blog entries, which consist of a title, an under title, a date, and author, a type (or tag), a text, and comments, can be created and updated by logged in users
  - Users, which consist of a username, a password, a nuame, can log in and log out
  - Comments, which consist of a title, an author, and text
- Source ERD



Copyright 2009 - Trend Micro Inc.

Data Model – Real Life Example

Table	Row Key	Family	Attributs
blogtable	TTYYYYMMDDHHmmss	info:	Always contains the column keys author,title,under_title. Should be IN-MEMORY and have a 1 version
		text:	No column key. 3 versions
		comment_title:	Column keys are written like YYYMMDDHHmmss. Should be IN-MEMORY and have a 1 version
		comment_author:	Same keys. 1 version
		comment_text:	Same keys. 1 version
usertable	login_name	info:	Always contains the column keys password and name. 1 version

- Row key
  - A concatenation of type (shortened to 2 letters) and timestamp
  - Row will be gathered first by type and then by date throughout the cluster
  - More chances of hitting a single region to fetch the needed data
- One-to-many relationship between BLOGENTRY and COMMENT is handled by storing comments as a family in BLOGENTRY and by using the timestamp as column key

Copyright 2009 - Trend Micro Inc.

Data Model – Real Life Example

Table	Row Key	Family	Attributs
blogtable	TTYYYYMMDDHHmmss	info:	Always contains the column keys author,title,under_title. Should be IN-MEMORY and have a 1 version
		text:	No column key. 3 versions
		comment_title:	Column keys are written like YYYMMDDHHmmss. Should be IN-MEMORY and have a 1 version
		comment_author:	Same keys. 1 version
		comment_text:	Same keys. 1 version
usertable	login_name	info:	Always contains the column keys password and name. 1 version

- Target schema advantages
  - Building the front page of the blog requires only a fetch of the “info:” family from BLOGTABLE
  - By using timestamps in the row key, scanners will fetch sequential rows for in order rendering of comments

Copyright 2009 - Trend Micro Inc.

Hbase Archtecture

Region	Row Keys	Column Family "Content:"
Region 1	00000	…
	00001	…
	…	…
	09999	…
Region 2	10000	…
	…	…
	…	…
	29999	…

- Table is made up of any number of regions
  - Region is specified by its *startKey* and *endKey*
- Each region may live on different node and is made up of several HDFS files and blocks, each of which is replicated by Hadoop.

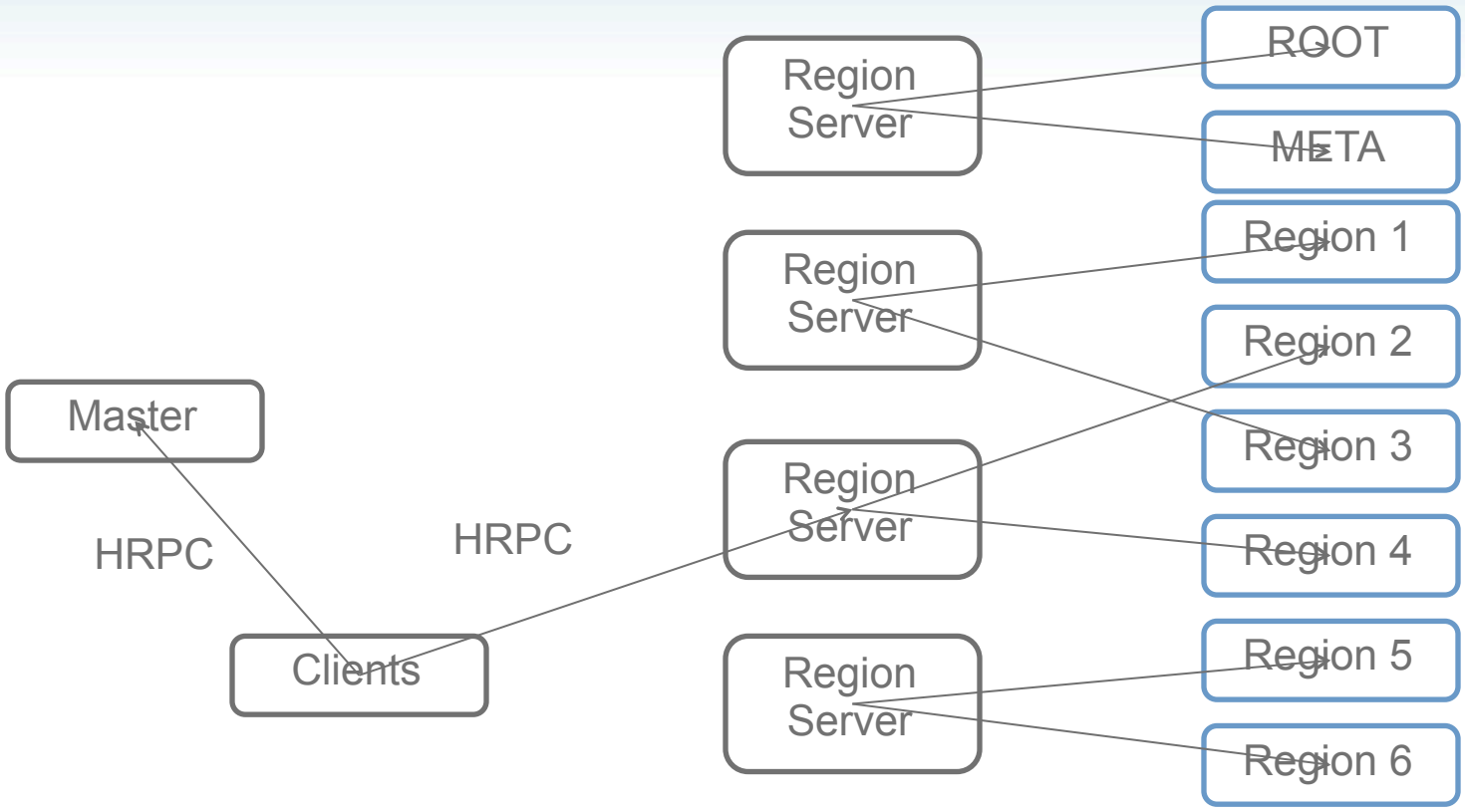
Copyright 2009 - Trend Micro Inc.

HBase Architecture

- Region Servers
  - Serving requests(Write/Read/Scan) of Client
  - Send HeartBeat to Master
  - Throughput and Region numbers are scalable by region servers
- HBase Master
  - Responsible for monitoring region servers
  - Load balancing for regions
  - Redirect client to correct region servers
  - The current SPOF

Copyright 2009 - Trend Micro Inc.

HBase Architecture



Copyright 2009 - Trend Micro Inc.

Connecting to HBase

- Java client
  - *get(byte [] row, byte [] column, long timestamp, int versions);*
- Non-Java clients
  - Thrift server hosting HBase client instance
    - Sample ruby, c++, & java (via thrift) clients
  - REST server hosts HBase client
- TableInput/OutputFormat for MapReduce
  - HBase as MR source or sink
- HBase Shell
  - JRuby IRB with “DSL” to add get, scan, and admin
  - *./bin/hbase shell YOUR_SCRIPT*

Copyright 2009 - Trend Micro Inc.

Reference

- Google BigTable paper
  - http://labs.google.com/papers/bigtable.html
- HBase official site
  - http://hadoop.apache.org/hbase/
- HBase official wiki
  - http://wiki.apache.org/hadoop/Hbase

Copyright 2009 - Trend Micro Inc.