HBase lab

Open Putty and add the following details:

Hostname: [ubuntu@52.91.169.107](mailto:ubuntu@52.91.169.107) (today only)

Click SSH/Auth and in the private key file download the pkk file from the lab folder for today (NoSQL16.pkk) and save it on C:/ and browse to it.

Click Open

When you are logged on I want you to change to be a different user using (your Tnumber). Use the same password and default the details.

sudo useradd <tnumber>

sudo su <tnumber>

cd

pwd

whoami

ls

open a terminal and fire up the HBase shell with the command:

hbase shell

version

status

Creating tables

Use your Tnumber in front of any tables e.g. below I used ‘t00194492wiki’

create 't00194492wiki', 'text'

Input data

put 't00194492wiki', 'Home', 'text:', 'Welcome to the wiki!'

Get data

get 't00194492wiki', 'Home', 'text:'

So far, our wiki schema has pages with titles, text, and an integrated version history but nothing else.

Let’s expand our requirements to include the following:

• In our wiki, a page is uniquely identified by its title.

• A page can have unlimited revisions.

• A revision is identified by its timestamp.

• A revision contains text and optionally a commit comment.

• A revision was made by an author, identified by name.

 To make schema changes, first we have to take the table offline with the disable command.

disable 't00194492wiki'

alter 't00194492wiki', { NAME => 'text', VERSIONS =>org.apache.hadoop.hbase.HConstants::ALL\_VERSIONS }

let’s add the revision column family, again using the alter command:

alter 't00194492wiki', { NAME => 'revision', VERSIONS => org.apache.hadoop.hbase.HConstants::ALL\_VERSIONS }

enable 't00194492wiki'

The put command only allows setting one column value at a time, and in our newly updated schema, we need to add multiple column values simultaneously so they all share the same timestamp.

We’re going to need to start scripting (change the table to be your table below)

import *'org.apache.hadoop.hbase.client.HTable'*

import *'org.apache.hadoop.hbase.client.Put'*

**def** jbytes( \*args )

args.map { |arg| arg.to\_s.to\_java\_bytes }

**end**

table = HTable.new( @hbase.configuration, *"t00194492wiki"* )

p = Put.new( \*jbytes( *"Home"* ) )

p.add( \*jbytes( *"text"*, *""*, *"Hello world"* ) )

p.add( \*jbytes( *"revision"*, *"author"*, *"T00194492"* ) )

p.add( \*jbytes( *"revision"*, *"comment"*, *"my first edit"* ) )

table.put( p )

Our put operation affects several columns and doesn’t specify a timestamp, so all column values will have the same timestamp (the current time in milliseconds).

Let’s verify by invoking get.

get 't00194492wiki', 'Home'

Streaming data from Wikipedia (see Kafka project for more on streaming)

Exit the shell (exit)

Check you are in your own folder (pwd)

Check you have the file import\_from\_wikipedia.rb

You need to edit the file to reference your own table in the script

Use the vi editor

vi import\_from\_wikipedia.rb

Use the arrows to get to the text then use x to delete characters and I to insert new character

Use escape and :w! to write the changes and escape and q! to quit

require *'time'*

import *'org.apache.hadoop.hbase.client.HTable'*

import *'org.apache.hadoop.hbase.client.Put'*

import *'javax.xml.stream.XMLStreamConstants'*

**def** jbytes( \*args )

args.map { |arg| arg.to\_s.to\_java\_bytes }

**end**

factory = javax.xml.stream.XMLInputFactory.newInstance  
reader = factory.createXMLStreamReader(java.lang.System.in)

document = nil  
buffer = nil  
count = 0

table = HTable.new( @hbase.configuration, *'t00194492wiki'* )

table.setAutoFlush( false )

**while** reader.has\_next

type = reader.next

**if** type == XMLStreamConstants::START\_ELEMENT

**case** reader.local\_name

**when** *'page'* **then** document = {}

**when** /title|timestamp|username|comment|text/ **then** buffer = [] **end**

**elsif** type == XMLStreamConstants::CHARACTERS buffer << reader.text **unless** buffer.nil?

**elsif** type == XMLStreamConstants::END\_ELEMENT

**case** reader.local\_name

**when** /title|timestamp|username|comment|text/

document[reader.local\_name] = buffer.join

**when** *'revision'*

key = document[*'title'*].to\_java\_bytes

ts = ( Time.parse document[*'timestamp'*] ).to\_i

p = Put.new( key, ts )

p.add( \*jbytes( *"text"*, *""*, document[*'text'*] ) )

p.add( \*jbytes( *"revision"*, *"author"*, document[*'username'*] ) )

p.add( \*jbytes( *"revision"*, *"comment"*, document[*'comment'*] ) )

table.put( p )

count += 1

table.flushCommits()

**if** count % 10 == 0 **if** count % 500 == 0

puts *"*#{count} *records inserted (*#{document[*'title'*]}*)"*

**end**

**end**

**end**

**end**

table.flushCommits()  
exit

push the escape button then :wq (to write and quit)

start the shell again

hbase shell

The text column family is going to contain big blobs of text content; it would benefit from some compression.

disable 't00194492wiki'

alter 't00194492wiki', {NAME=>'text', COMPRESSION=>'GZ', BLOOMFILTER=>'ROW'}

enable 't00194492wiki'

Through the magic of \*nix pipes, we can download, extract, and feed the XML into the script all at once.

The command looks like this:

curl <dump\_url> | bzcat | ${HBASE\_HOME}/bin/hbase shell import\_from\_wikipedia.rb

Note that you should replace <dump\_url> with the URL of a WikiMedia Foundation dump file of some kind.

Plug in the URL and run it! (note I had to type in the –k as pasting it won’t work in mobaXterm, worked ok on pasting in putty)

Exit the shell and run the following command

curl –k https://dumps.wikimedia.org/enwiktionary/latest/enwiktionary-latest-pages-articles.xml.bz2 | bzcat | hbase shell ./import\_from\_wikipedia.rb

Let it run for a couple of minutes.

Then ctrl-C

Open the shell and look for an entry e.g.

get 't00194492wiki', 'winter’

Exit the shell

So, let’s take a look at your HBase server’s disk usage, which will give us insight into how the data is laid out.

You can inspect HBase’s disk usage by opening a command prompt to the hbase.rootdir location

cd /tmp/hbase-root/hbase

du -h

du is a standard \*nix command-line utility that tells you how much space is used by a directory and its children, recursively. The -h option tells du to report numbers in human-readable form.

This output tells us a lot about how much space HBase is using and how it’s allocated.

The lines starting with /t00194492wiki describe the space usage for the wiki table.

The long-named subdirectory bf60185……..ba represents an individual region (the only region so far).

Under that, the directories /text and /revision correspond to the text and revision column families, respectively.

Finally, the last line sums up all these values, telling us that HBase is using 1.5GB of disk space.

If you leave the process running long enough you will see a split into two regions.