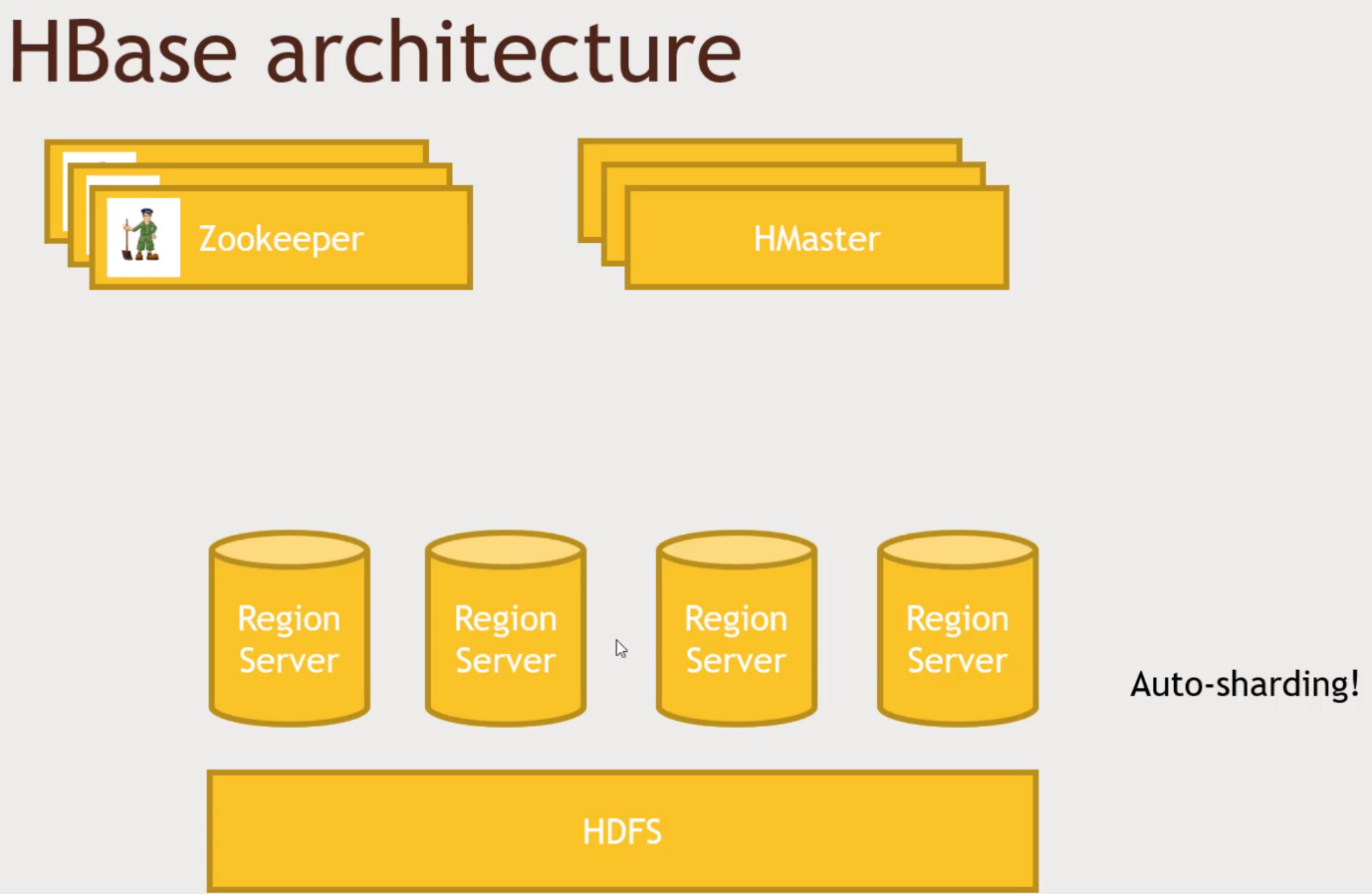
HBASE Intuition – Non-relational, scalable database built on HDFS.

CRUD

* Create
* Read
* Update
* Delete
* There is no query language, only CRUD API’s !

HBase architecture

If your data grow, it can do Auto-Sharding.

Web application – it will talk to the Region Server, HMaster oversees the operation. Zookeeper, who watches the watchers. Who is the current Master is? If One master goes down who will take over and etc.

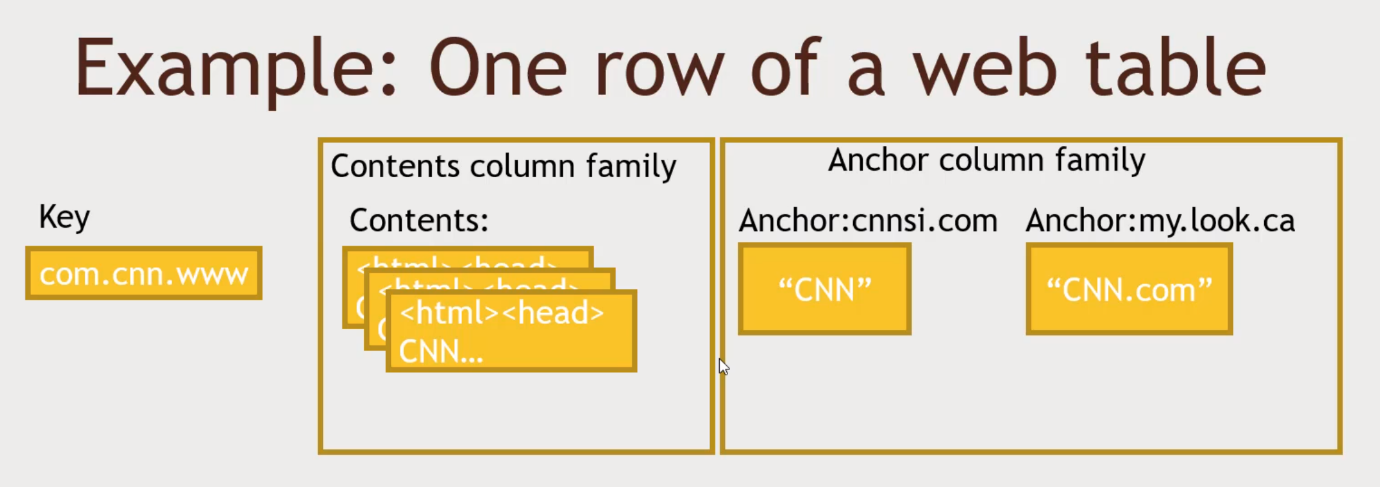
HBase data model

* Fast access to any given ROW
* A ROW is referenced by a unique KEY (CustomerID, columns maps to customers)
* Each ROW has some small number of COLUMN FAMILIES
* A COLUMN FAMILIES may contain arbitrary COLUMNs (Contains a large number of columns that is mapped)
* You can have a very large number of COLUMNs in a COLUMN FAMILY.
* Each CELL can have many VERSIONS with given timestamps
* Sparse data is A-OK – missing columns in a row consume no storage.

How can I have a very SMALL Number of COLUMN FAMILY associated with each row of my database.

CELL have many versions of the cell, based on timestamps. Retain many versions.

Google wants to track all the links in a given webpage.

Key is stored in reverse order. To deal with scalability issues.

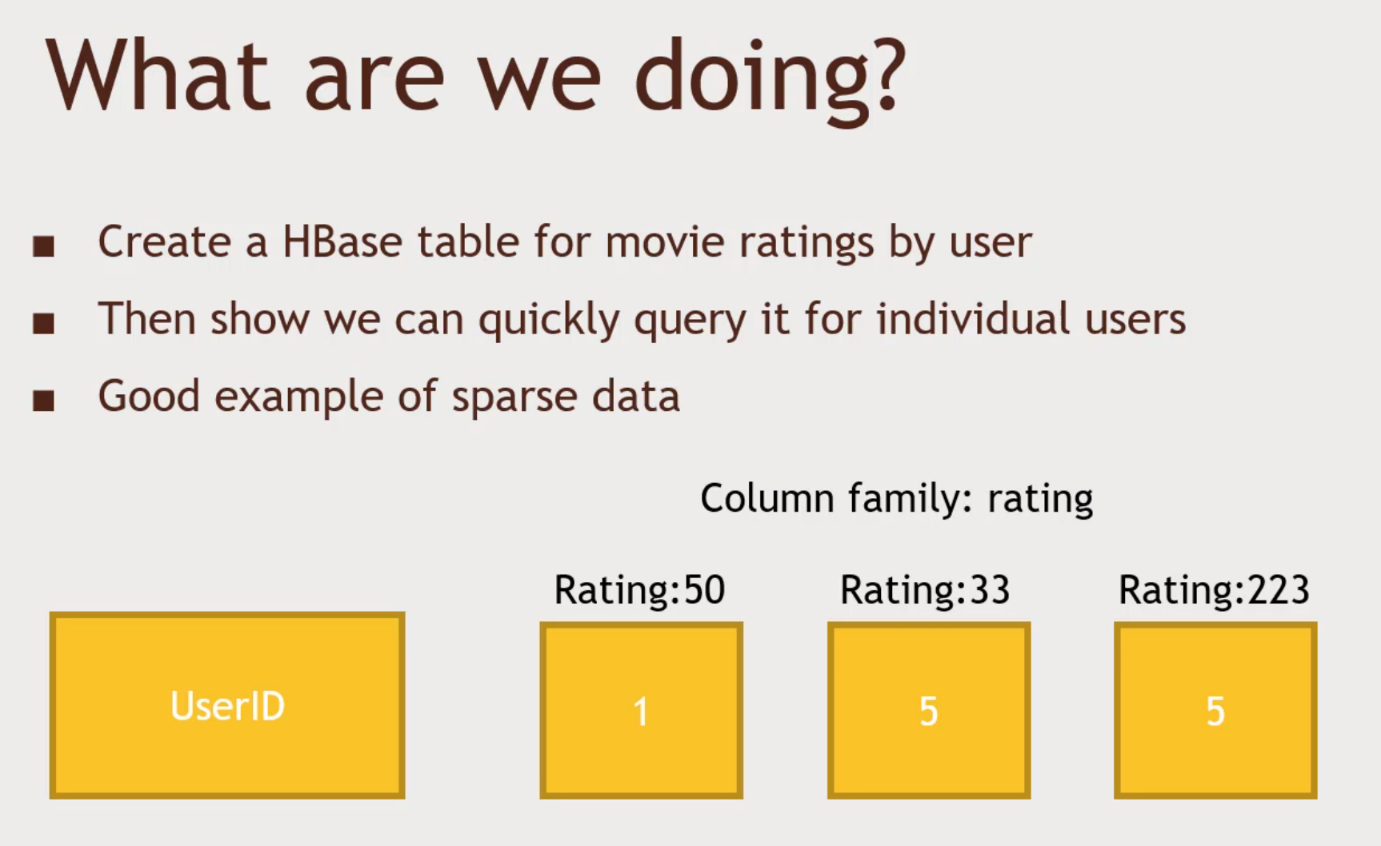
Contents Column Family Contents: 3 copies. The last 3 crawls.

Anchor Column family, all of the pages on the web that link back to cnn. Anchor text, Hyperlink.

Anchor:<website> and the column text there.

Ways to interact with HBASE

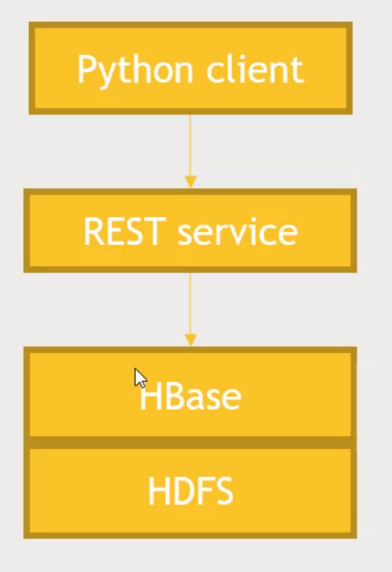
* HBase shell
* Java API (Webserver, webpage, HBase is written in Java)
  + Wrappers for Python, Scala, etc.
* Spark, Hive, Pig (Do all the complicated transformation and Writing results into HBase)
* REST service -> Restful Interface, HTTP requests use Sparks
* Thrift service -> Facebook
* Arvo service

HBase Practice.

Python to populate the table. Gives back all the ratings for a given UserID

This user ID, rated Movie id 50 is 1 star, ID: 33 as 5 stars, ID:223 as 5 stars.

If the customer wants, give me all the Ratings that a particular USERID rated for

We are going to write a client that sits on the REST Service which sits on the HBASE.

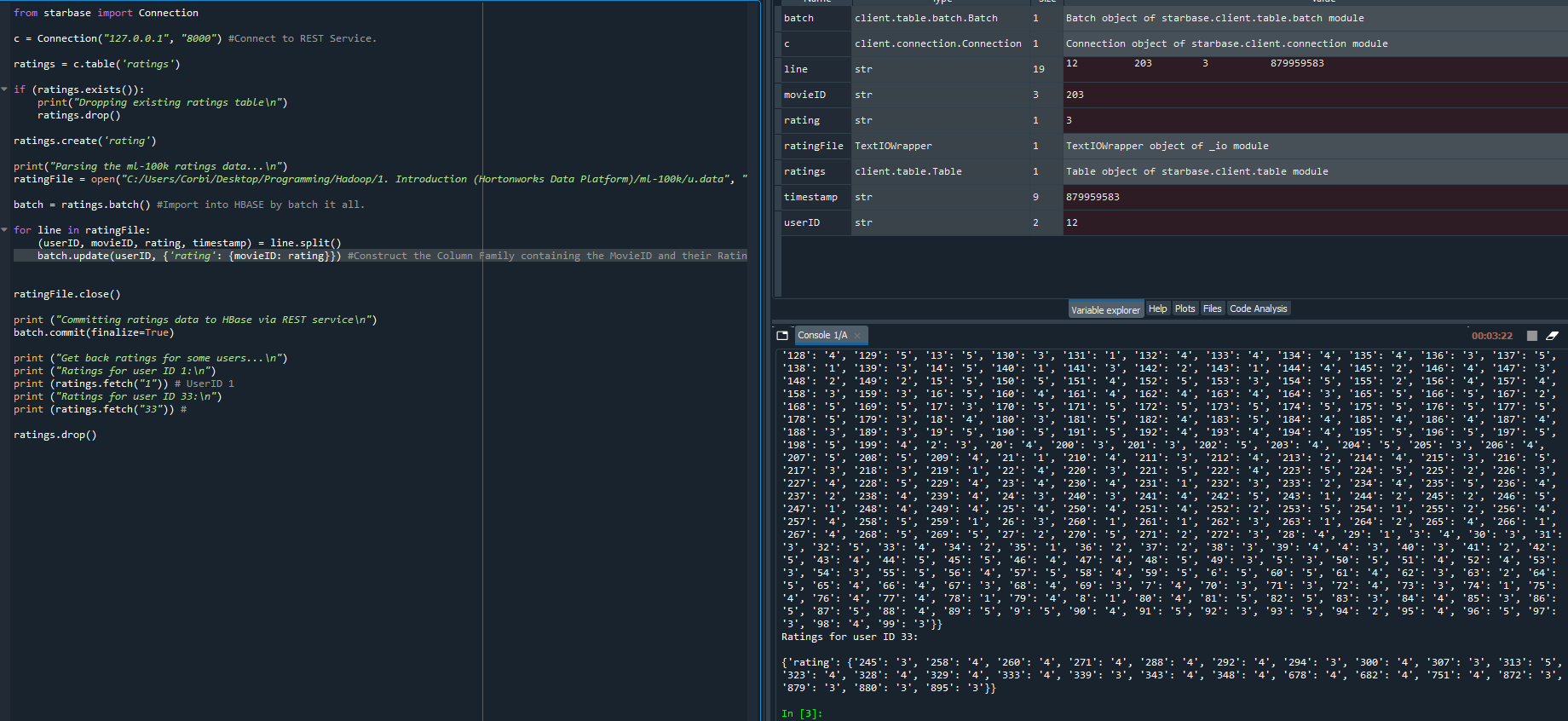
Activity

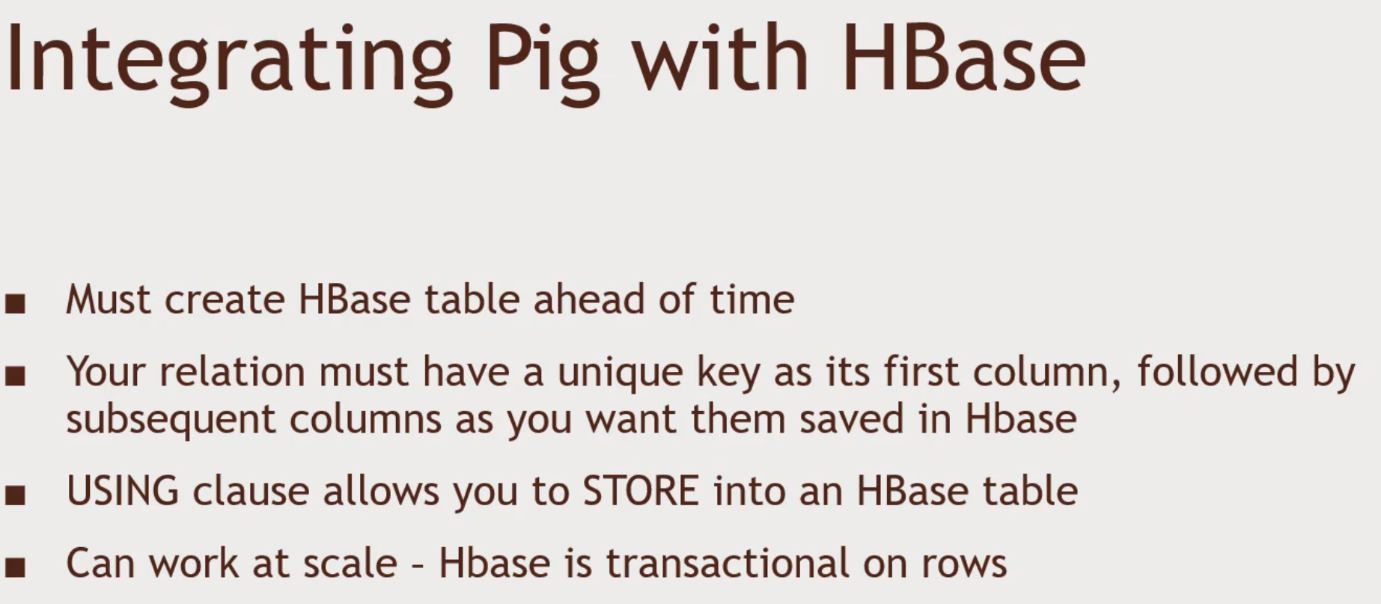
Step1: Get HBase Running. Go to Ambari and Click on HBASE, Services tab and Service Actions -> Start. (Wait for a while to connect to HBase)

Step2: Launch a Rest server. Login to Putty. Login to root by typing su root and password. To start the rest server you type in :”usr/hdp/current/hbase-master/bin/hbase-daemon.sh start rest –p <8001> --infoport 8002”:

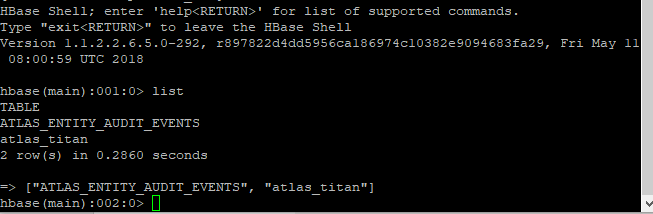
Step3: python script to run on Rest Service, use Starbase. If not installed, use pip install starbase to install starbase and start using the HBaseExample.py script.

See Result Below.

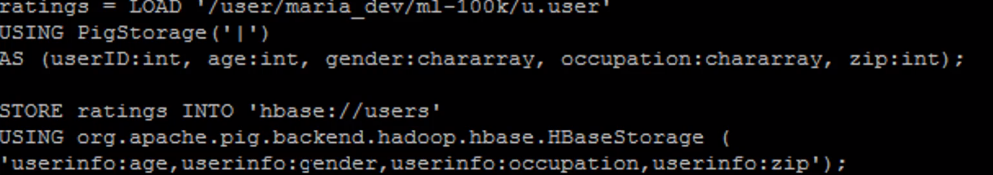
Using HBase with Pig to import data at scale.

Import TSV that does the same thing.

Step1: Use Ambari, HBase service running. Files View in Ambari, and user -> maria\_dev, ml-100k folder, upload the u.user file that will be sitting on HDFS, It is | delimited. Age, Male, Occupaction.

 Step2: Import using pig. Log in using Putty. And access the root. And type hbase shell -> list to see the list of tables created.

Step 3: exit and download the script

Create relation name rating, using PigStorage(|) as the different types of variables. First column is the unique key and the rest is stored in a column family. Look into the hbase.pig. userinfo is the column family name.

Run by typing pig hbase.pig.

Step4: Verify that the data exists. Type hbase shell -> scan ‘users’

Step5: To clean up the mess u must perform -> disable ’users’ then drop ‘users’ then if you do list, the table will be gone.

Step6: Go to Ambari -> Services -> HBase -> Service Actions -> Stop