**Apache HBase theNoSQL Database for Hadoop**

Apache Hbase is a NoSql database that runs on top of Hadoop distributed File System (HDFS) and scalable big data store i.e. Hbase can leverage the distributed processing paradigm of HDFS for storage and Map reduce programming for computation.

Prior to Nasal database we had Relational Database Management System (RDBMS) which helped companies for the storage of information in databases used for financial records, manufacturing and logistical information, personnel data, and other applications since the 1980s. but in today’s era data is getting generated in big Volume (Peta byte or Zeta byte) with different Variety and Velocity from sources like social media,e-commerce etc. hence Apache HBase was introduced as nosql database to handle this big data.

**History:**

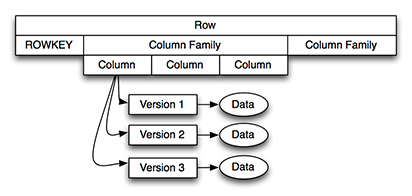
Hadoop was mainly introduced for batch processing but companies also needed a database which could be used for real-time responses. Google was faced with a challenging problem: How could it provide timely search results across the entire Internet? The answer was that it essentially needed to cache the Internet and define a new way to search that enormous cache quickly. It defined the following technologies for this purpose:

* [***Google File System***](http://research.google.com/archive/gfs.html)***:*** A scalable distributed file system for large distributed data-intensive applications
* [***Bitable***](http://research.google.com/archive/bigtable.html)***:*** A distributed storage system for managing structured data that is designed to scale to a large size: petabytes of data across thousands of commodity servers
* [***Map Reduce***](http://research.google.com/archive/mapreduce.html): A programming model and an associated implementation for processing and generating large data sets

**Introduction Apache HBase:**

***Apache HBase***is the open source implementation of Google’s Big Table, with slight modifications.  It was created in 2007, it was initially a contributions to Hadoop, but beyond its Hadoop roots, HBase is a powerful database in its own right that blends real-time query capabilities with the speed of a key/value store and offline or batch processing via MapReduce.It is meant to host large tables with billions of rows with potentially millions of columns and run across a cluster of commodity hardware. In short, HBase allows you to query for individual records as well as derive aggregate analytic reports across a massive amount of data. HBase is a top-level Apache project that runs in Facebook, Twitter, and Adobe, just to name a few.

HBase actually defines a four-dimensional data model as shown below:



Following four coordinates define each cell:

1. ***Row Key***: Each row has a unique row key; the row key does not have a data type and is treated internally as a byte array.
2. ***Column Family***: Data inside a row is organized into column families; each row has the same set of column families, but across rows, the same column families do not need the same column qualifiers. Under-the-hood, HBase stores column families in their own data files, so they need to be defined upfront, and changes to column families are difficult to make.
3. ***Column Qualifier***: Column families define actual columns, which are called column qualifiers. You can think of column qualifiers as the columns themselves.
4. ***Version***: Each column can have a configurable number of versions, and you can access the data for a specific version of a column qualifier.

When designing an HBase data model, it is helpful to think about how the data is going to be accessed. You can access HBase data in two ways:

* Through their row key or via a table scan for a range of row keys
* In a batch manner using map-reduce

### Setting Up an HBase Environment

Refer below steps to install and run Hbase:

1. You need to have Java installed to launch HBase, so if you have not already done so, download and install Java from Oracle’s website.

2. Download HBase from Apache website url mentioned below latest version is 1.4.0:

<http://www.apache.org/dyn/closer.cgi/hbase/>

3. HBase can be installed on installed on a UNIX/Linux environment and for Windows download and install Cygwin and run HBase from there.

3. Decompress it to your hard drive.

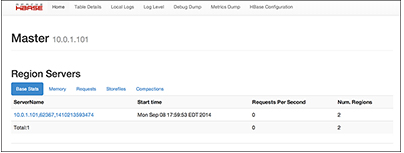
4. Define an environment variable named “HBASE\_HOME” that points to the root directory of where you decompressed HBase

5. Execute the **start-hbase.sh** script from HBase’s bin directory.

6. You can test your installation by opening a browser to the following URL:

<http://localhost:60010>

7. You should see a screen similar to [Figure 3](javascript:popUp('/content/images/art_haines_hbases1_1/elementLinks/haines_hbase_1_1-03.jpg')).



We can interact with HBase in two different ways.

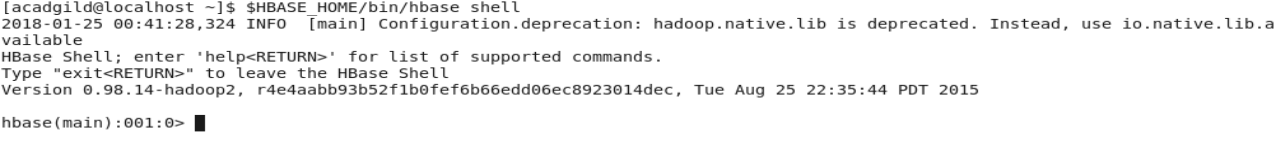
1. Through HBase interactive shell
2. HBase Java Client API

Let’s begin by interacting with HBase using its command shell.

1. **We can login to hbase shell using below command**

**$HBASE\_HOME/bin/hbase shell**

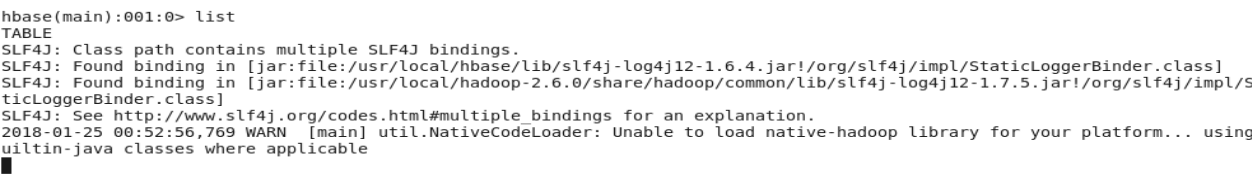
You should see output similar to the following:



2. **Version command will display the version of HBase**



3. **List command will list all the tables present HBase**



### Creating a table in HBase :

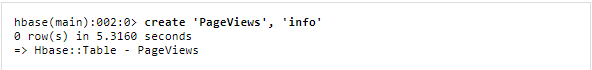
HBase data model, columns are grouped into column families, which must be defined during table creation. We must have at least one column family. HBase currently does not do well with above three column families so keep the number of column families in your schema low.

Syntax:

create ‘<table-name>’,’<column-family1>’ ,’<column-family2>’ …….

Let’s create a new table named **'**PageViews**'** with a column family named **'**info**'**:

**create ‘Pageviews’,’info’**

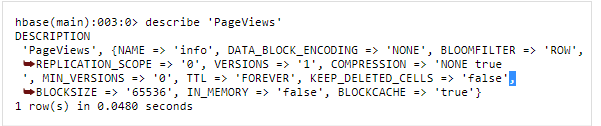
****

Every table needs to have at least one column family, so we created one named **'**info**'**.

Execute the following command to **list** all tables:



We can get more information about the table by executing the describe command:



### Inserting the data into HBase

We can insert the data using PUT command

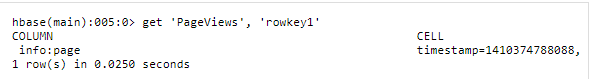
**Syntax:** put ‘<table-name>’,’row-key’,’columnfamily:columnname’,’value’

Row-key is a mandatory field which serves as the unique identifier for every record.

The following command inserts a new row into the info column family:



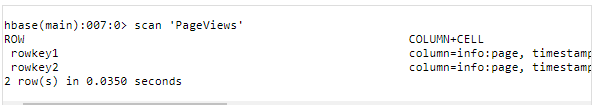
The **put** command inserts a new record into the row with row key “**rowkey1**”. We specify that we are adding the value “**/mypage**” into the **page** column in the **info** column family. We can then retrieve the column families and their values for **rowkey1** with the **get** command:



You can see that column “info:page”, or more specifically the page column qualifier in the info column family, has the value “/mypage” and the specified time stamp for when it was inserted. Let's add another row before we do a table scan:



Now, If we execute a **scan** command with the table name, it returns all rows in the table



**Summary**

HBase is a Nasal database commonly referred to as the Hadoop Database, which is open-source and is based on Google's Big Table white paper. HBase runs on top of the Hadoop Distributed File System (HDFS), which allows it to be highly scalable, and it supports Hadoop's map-reduce programming model. HBase permits two types of access: random access of rows through their row keys and offline or batch access through map-reduce queries.