HBase: A NoSQL Database

Technical Report · May 2017

DOI: 10.13140/RG.2.2.22974.28480

Hiren Patel
Western Sydney University
3 PUBLICATIONS 0 CITATIONS

HBase: A NoSQL database

Abstract

In past decade we have witnessed the explosion of data and it has been always challenging for us to store and retrieve the data. Until the 1970s we were using RDBMS but that was not enough to handle a large amount of data. The rise of growing data gave us the NoSQL databases and HBase is one of the NoSQL database built on top of Hadoop. This paper illustrates the HBase database its structure, use cases and challenges for HBase. HBase is suitable for the applications which require a real-time read/write access to huge datasets.

Introduction

Big data has proven itself a huge attraction point for many researchers and academics across the world. Due to vast usage of social media applications data is growing rapidly nowadays. This data is often formless, disorganized and unpredictable. Storing and analysing this data is not an easy task. However, NoSQL databases are the databases by which we can handle and extract this data with ease. There are many NoSQL databases available for the data scientists. This paper illustrates HBase the NoSQL database type that is being used widely in many big companies. HBase is an open source NoSQL database which is a java implementation of Google's Big Table. HBase is a part of Apache and also sometimes referred as the Apache HBase.

HBase and History

HBase is abbreviated as the Hadoop Database and it runs on top of the Hadoop as a scalable big data store. It is the Hadoop database that means it has the advantages of Hadoop's distributed file system and MapReduce model by default. It is referred as the columnar database because in contrast to a relational database which stores data in rows, HBase stores data in columns (Egan, 2017).

HBase is modeled after the Google's BigTable so it provides distributed data storage capabilities like the BigTable on HDFS. HBase has capabilities of allowing access to sparse data. Sparse data is defined as the data which is small but valuable data within the gigantic volume of unstructured data used for Big Data analytics. It has the abilities to report failures automatically, data reproduction throughout clusters and coherent read and write. There are several advantages of HBase over relational databases like the later one is hard to scale and have to have a schema for them.

Some of the key features of HBase are listed below (Devmanuals.com, 2016):

- Horizontally scalable
- Fault tolerant storage capability for sparse data
- Supports parallel processing, HDFS and MapReduce
- High adaptable data model
- Ability to host large tables
- Real-time lookups
- Automatic load balancing of tables
- Supports block cache and bloom filters for big amount of query optimization
- Easy JAVA API for clients

There is a small history behind the HBase. Back in 2004, when Google was facing problem on how they could provide efficient search results they developed BigTable technology. In 2007 Mike Cafarella released code for open source implementation of BigTable known as HBase. At the start, the initial model of HBase was developed as a contributing data model for Hadoop. Moreover, between the year 2008 and 2010 HBase became the top level project under the Apache Hadoop (Hbase.apache.org, 2017 & Haines, 2017).

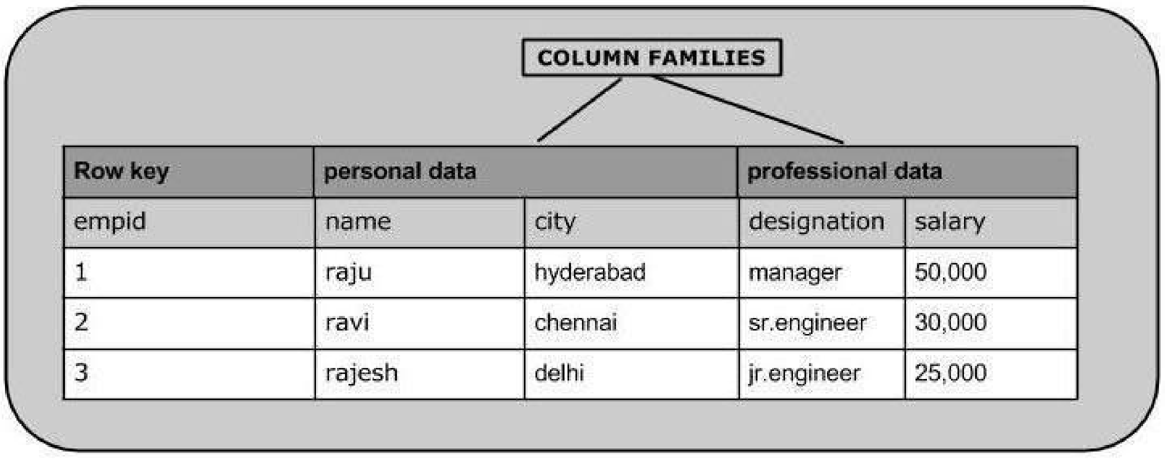
BigTable

BigTable is a distributed storage system developed by Google designed to store the gigantic size of data across several severs. Many projects such as Google Earth, Google analytics, personalized search, all web indexing and financial data stored in BigTable. All these applications have various demands regarding the size and latency but BigTable provides a flexible and high-performance solutions for them. It provides a data model that supports dynamic control over data format rather than a relational data model. The API of BigTable delivers functionalities for creating and deleting tables and columns, changing metadata, cluster and access control. All these capabilities have adopted by HBase, however, there are many features that differentiate both these technologies (Chang et al., 2008).

HBase structure and architecture

HBase enables the low expectancy read-write on top of HDFS. Tables in HBase stored as a multidimensional sparse map with rows and columns enabling the random real-time read-write access. Each cell has the timestamp and uniquely identified by the table, row, column family, and timestamp. As HBase has Java client API, the tables in HBase can be used as an input and output target for MapReduce job. HBase uses the Zookeeper which is an open source project of Apache especially used for the management of partial failures in databases. Additionally, it also provides the maintenance of configuration information and distributed synchronization. Zookeeper has fleeting nodes which represent the region servers which are used to track the failures and network partitions (Taylor, 2010).

As mentioned earlier HBase is the column-oriented database so the tables in HBase are organized by the rows. The HBase table schema only defines the column families with key value pairs. The tables are the collection of rows, rows are the collection of column families, a column family is the collection of columns and these columns are the collection of key-value pairs. The main benefit of the column-oriented database over the row-oriented database is it can be used for the huge amount of data which requires online analytical processing. Figure 1 illustrates the basic structure of HBase table. As shown the columns has the key value pairs collections.


COLUMN FAMILIES

Row key	personal data		professional data	
empid	name	city	designation	salary
1	raju	hyderabad	manager	50,000
2	ravi	chennai	sr.engineer	30,000
3	rajesh	delhi	jr.engineer	25,000

Fig 1: HBase column-oriented database example (www.tutorialspoint.com, 2017)

The tables in HBase are divided into different regions which are vertically divided by column families and these regions are served by the region servers. Typically, the HBase architecture has three main features a master server (HMaster), region servers and zookeeper.

I. **Master Server or HMaster**

The main task of master server is to the assignment of regions to the region servers with the help of Zookeeper, controlling of load balancing of region servers. Load balancing is a key feature of the master server in which it unloads the busy servers and moves it to the unoccupied servers maintaining the state of the cluster (www.tutorialspoint.com, 2017). Master server also liable for schema changes and a creation of tables and column families. Other responsibilities of Master Servers are managing and monitoring of Hadoop clusters and operating supervision on them, failover handling and DDL operations (DeZyre, 2016).

II. Region server

Regions are the tables spread within the region servers. The regions of the region servers communicate with the client and control data related processes. The region server contains memory stores and HFiles. Region servers can be said as the worker nodes which handles the CRUD operations from clients. It runs on the HDFS data nodes and it has four main components. Block cache, MemStore, WAL, HFile. A block cache is basically the read cache work similar as the other cache systems. Here intermittent data is stored and when the block cache is full most recent data will be removed. In contrast to block cache, the MemStore is a write cache which stores new data that has not been stored. WAL (Write ahead log) stores new data that is not stored in the permanent storage. HFiles is the actual storing files which store the sorted key values of rows (DeZyre, 2016).

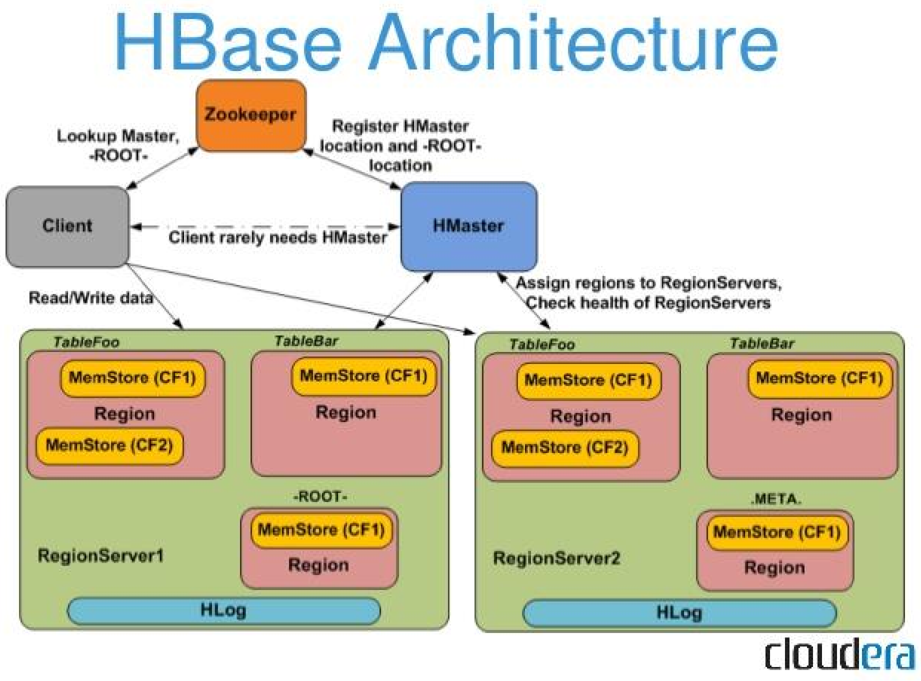


Fig 2: HBase architecture (DeZyre, 2016)

III. Zookeeper

Zookeeper is also an essential part of HBase architecture as it stands between the client and HMaster. It is used as a scattered coordination facility to recover the any crashes happened to the region servers by facilitating them on other region severs which are fully functional. As it is the center between the client and master server it maintains the structure of information and management. Whenever there is a request from a client to access the regions their first contact point is the zookeeper because master and region servers are registered with the zookeeper. It is also referred as the coordinator. Zookeeper keeps information of all the region servers such as how many region servers are available and each of them holding which data nodes. Moreover, it facilitates the tracking of server failures and divisions, maintenance of composition information and so on.

The table in HBase is built of several regions and each of the regions is definite by the startkey and endkey. They are made up of HDFS files each of them is simulated by Hadoop. In the schema, only the parent column families are fixed and other columns are added on the fly to tables. Here, each cell is associated with particular column family and name by which we can identify what type of data each cell comprises (Taylor, 2010).

HBase and Big Data

Since the last decade, we have created a massive amount of data per second and it is increasing every day. It is not only the volume and the velocity but the variety of the data is also of different nature due to the vast usage of smart phones. Data warehousing, analyzing and summarizing this data in competently and cost efficient manner is the biggest challenge. For operating big data, the platforms such as Hadoop can be useful because it enables the use of HDFS and MapReduce by which is able to store and process these data in terabytes and petabytes size

(Garg, 2014). As HBase runs on top of Hadoop it is a suitable option for Big Data and additionally for the huge volume of data relational database is not capable.

There are several benefits of HBase over other NoSQL databases. It gives developers a direct access to Hadoop scale storage by which they can read-write specific data quickly. It also gives the transactional platform for running high-scale and constant applications by which it can handle the volume, complexity, and variety of Hadoop database easily (Henschen, 2012). Moreover, HBase is strongly consistent, high-quality performer and have the Hadoop's ecosystem make it the choice over other NoSQL databases (Purtell, 2016). HBase's region servers provide several data nodes which provide the suitable environment for the large data volume. It is also schema independent so it can be used for a variety of applications. Recently the integration between YARN and HBase makes it a strong candidate for Big Data applications (Shaikh, 2016).

However, there are other NoSQL competitors such as MongoDB and Cassandra for HBase who are now more popular for use in Big Data operations. At the start, HBase did well but it started losing its popularity a few years back. Nowadays, MongoDB, Cassandra, and Redis are ahead of HBase in terms of use in Big Data applications. Many data analysts found HBase a "pain to deploy and run" and "impossible to run". Back in 2014, Rick Grehan argued that there is a fundamental difficulty in assembling and troubleshooting related to HBase schema (Asay, 2016). Figure 3 shows the comparisons between the various databases used for Big Data applications between 2013 to 2016. As illustrated MongoDB is most used NoSQL database followed by Cassandra and Redis ahead of the HBase.

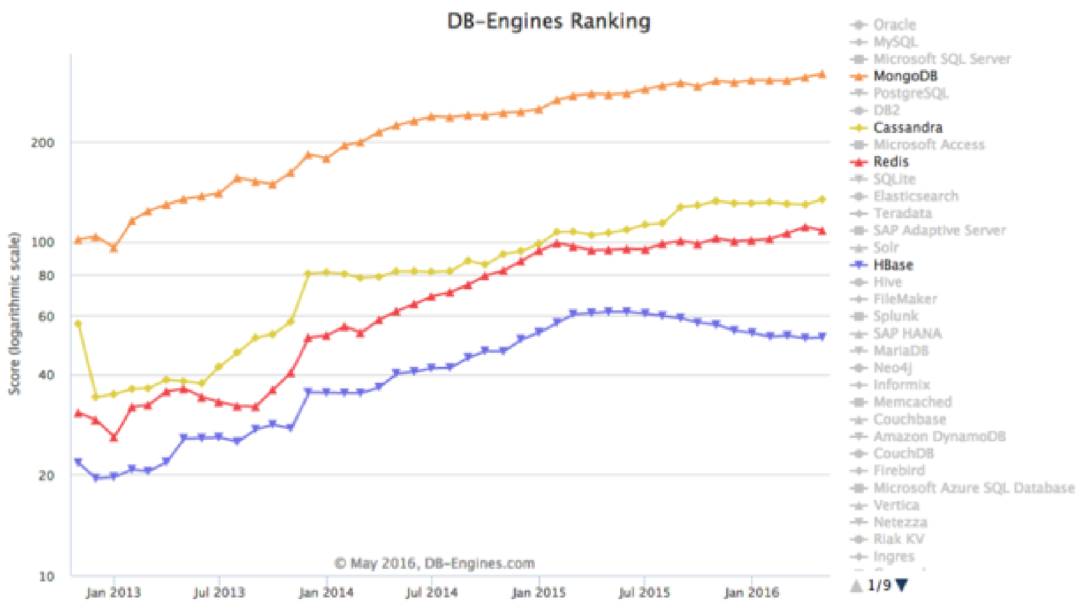


Fig3: DB-Engines rankings (Asay, 2016)

HBase applications and usage

There are many big companies which are using HBase as a storage system for their organization. Several companies have adopted HBase use cases which include handling content and handling incremental data. Facebook was one of the earliest users of HBase which they have used for their messaging platform. HBase supports Facebook's 135 million messages monthly. Based on their need they need a storage which supports a short set of progressive data that inclines to be elusive and an easy access to ever growing data. They use Haystack to store the attachments, zookeeper is used for user discovery (Hoff, 2010). Another major user of HBase is Pinterest which runs around 38 different clusters for accomplishing 5 million operations every 5 seconds. Imgur uses HBase for their notification system (Blog.imgur.com, 2015). There are companies like Adobe, Yahoo, Flurry, Airbnb, Spotify, Netflix and Salesforce.com (DeZyre, 2016).

HBase Challenges and Limitations

Nevertheless, the architecture of HBase is good but it has some problems. As HBase is Java implementation, the non-Java client consigned to the thrift and gateways on the contrary Cassandra offers productive experience in all languages. The architecture is based on region servers and there is usually downtime due to region server failover. When we talk about the data storage application even one minute of downtime is not acceptable (Doug, 2013). HBase's architecture is said to be a single point of failure. Generally, HBase is running with a cluster of large numbers which is managed by only one master server. It also manages all the region servers so if it goes down it took a long time to recover it. The cross data operations and join operations is only possible with the MapReduce which took a lot of time. We need to reconfigure new design when we need to migrate data from external RDBMS sources. In HBase, we cannot have more than one indexing in the table because only row key column acts as a primary key. This affects the performance when a client wants to search more than one field. HBase cannot perform SQL functions and does not support query optimizer. It does not support partial key and consents only one default category per table (Guru99.com, 2017).

In HBase, the indexing is done manually for that a developer has to write the code manually. There is no default indexing like the traditional databases. Moreover, storing of large binary data and normalization is difficult with HBase. The requirements of HBase is expensive which includes at least 5 servers and 2-4 GB minimum per server. Conclusively, the maintenance of HBase is approximate $850 per month excluding the internet service provider charges (Panda, 2013).

My findings

As per my findings, HBase is the strong NoSQL storage system which is inspired by Google's BigTable. It has many capabilities which give advantages over other NoSQL databases. HBase is column based database means the tables schema defines only column families. The column name is denoted by prefix for identification of column family for example, in Fruit: Mango, Fruit: Kiwi fruit is a family name and mango and kiwi are its members. Each row in HBase table is associated with the key and this key is used for access mechanism in a row. Its architecture is very strong and capable of storing sparse data in real time. The three main components the master server, region server and zookeeper are an essential part of its architecture which enables fast operations on data. However, HBase has loose the grip over the years with increasing data and it is not used for Big Data operations. For some feature, it has upper hand over the MongoDB and Cassandra but overall both are performing well than the HBase.

Conclusion

Conclusively, HBase is a distributed column-oriented database with having prebuilt functionalities of Hadoop. It is horizontally scalable and sharable across other databases. Currently, being used by many big ventures across the globe. In this paper, we have talked about the architecture and structure of HBase, its current usage and its limitations. The world of data is growing at a rapid pace and we will need some better solutions for handling and analyzing this data in future.

References

1. Egan, D. (2017). Big Data — What is HBASE | The Sociable Geek. [online] Thesociablegeek.com. Available at: http://thesociablegeek.com/big-data/big-data-what-is-hbase/ [Accessed 21 May 2017].

2. Hbase.apache.org. (2017). *Appendix G. HBase History*. [online] Available at: http://hbase.apache.org/0.94/book/hbase.history.html [Accessed 22 May 2017].

3. Haines, S. (2017). *Introduction to HBase, the NoSQL Database for Hadoop | Introduction to HBase | InformIT*. [online] Informit.com. Available at: http://www.informit.com/articles/article.aspx?p=2253412 [Accessed 22 May 2017].

4. Devmanuals.com. (2016). *History of Apache HBase*. [online] Available at: http://www.devmanuals.com/tutorials/bigdata/hbase/History-of-Apache-HBase.html [Accessed 22 May 2017].

5. Chang, F., Dean, J., Ghemawat, S., Hsieh, W., Wallach, D., Burrows, M., Chandra, T., Fikes, A. and Gruber, R. (2008). Bigtable. *ACM Transactions on Computer Systems*, 26(2), pp.1-26.

6. Taylor, R. (2010). An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC Bioinformatics*, 11(Suppl 12), p.S1.

7. www.tutorialspoint.com. (2017). *HBase Overview*. [online] Available at: https://www.tutorialspoint.com/hbase/hbase_overview.htm [Accessed 23 May 2017].

8. DeZyre. (2016). *Overview of HBase Architecture and its Components*. [online] Available at: https://www.dezyre.com/article/overview-of-hbase-architecture-and-its-components/295 [Accessed 23 May 2017].

9. Garg, N. (2014). *HBase Essentials*. 1st ed. Packt Publishing.

10. Purtell, A. (2016). *Investing In Big Data: Apache HBase – Salesforce + Open Source = ❤ – Medium*. [online] Medium. Available at: https://medium.com/salesforce-open-source/investing-in-big-data-apache-hbase-b9d98661a66b [Accessed 23 May 2017].

11. Henschen, D. (2012). HBase: Hadoop's Next Big Data Act.

12. Shaikh, S. (2016). *How and When should you use HBase NoSQL DB*. [online] Eduonix.com | Blog. Available at: https://www.eduonix.com/blog/bigdata-and-hadoop/use-hbase-nosql-db/ [Accessed 23 May 2017].

13. Asay, M. (2016). *HBase: The database big data left behind*. [online] InfoWorld. Available at: http://www.infoworld.com/article/3066358/analytics/hbase-the-database-big-data-left-behind.html [Accessed 23 May 2017].

14. Hoff, T. (2010). *Facebook's New Real-time Messaging System: HBase to Store 135+ Billion Messages a Month - High Scalability -*. [online] Highscalability.com. Available at:

http://highscalability.com/blog/2010/11/16/facebooks-new-real-time-messaging-system-hbase-to-store-135.html [Accessed 24 May 2017].

15. Blog.imgur.com. (2015). *Tech Tuesday: Imgur Notifications: From MySQL to HBase | The Imgur Blog*. [online] Available at: http://blog.imgur.com/2015/09/15/tech-tuesday-imgur-notifications-from-mysql-to-hbase/ [Accessed 24 May 2017].

16. Doug, H. (2013). Big Data Debate: Will HBase Dominate NoSQL?.

17. Guru99.com. (2017). *Cite a Website - Cite This For Me*. [online] Available at: http://www.guru99.com/hbase-limitations-advantage-problems.html [Accessed 24 May 2017].

18. Panda, B. (2013). *Limitations of Hive and Hbase*. [online] Bigdatawarehouse.blogspot.com.au. Available at: http://bigdatawarehouse.blogspot.com.au/2013/06/limitations-of-hive-and-hbase.html [Accessed 24 May 2017].