Big Data Database Systems : A Comparative Study

Adarsh Kumar
Research Scholar
Department of Computer Science, Himachal Pradesh University, Shimla-5
E-mail: adarshsharma008@gmail.com
Anita Ganpati
Associate Professor
Department of Computer Science, Himachal Pradesh University, Shimla-5
E-mail: anitaganpati@gmail.com

----------Abstract----------

We are living in 21st century digital world where large amount of data generated from various sources in bulk and in different format at high rate. In Current time data is more important than information and bulk data help in making correlation and help in decision making which is not possible with small data set. Different formats are not fit in one system. In this paper, a comparative study on different big data database systems in which discuss about classification, architectural properties and functionality.

Keywords-**BASE**, **Big Data, MVCC, NoSQL, Open Source, Scalable SQL, Scalability**

1. Introduction

In the early days of data generation, almost the data is generated in relational form typically tables, flat files and records which stored and analyzed by traditional system such as MySQL, IBM DB2 etc. Digital Revolution start almost two decade ago, by the Internet or WWW and HTTP Protocol which becomes the standard for information sharing in 1991[3]. Data was generated in diverse forms like blog posts, tweets, social network interactions, scientists detailed measurements etc. About 30,000 gigabytes of data are generated every second, and the rate of data creation is accelerating [12].

A Study by International Data Corporation on Digital Universe in 2011 estimates the volume of data created and stored grow at 45.2% to 7,910 exabytes is projected in 2015 on the basis of 130 exabytes in 2005 and 1,227 exabytes in 2010[7] .

Big Data define as the amount of data just beyond the traditional database methods and tools capability to store, manage and process efficiently. Big data generally explained according to three V's namely Velocity, Variety and Volume. **According to Gratner, " Bigdata** is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation"[13].

2. Overview of Big Data Database Systems

Large volumes of data are coming at a much faster velocity in varieties of formats which is not handled by traditional systems. The term "NoSQL" is emerging to handle this type of data and originally develop by Carlo Strozzi in 1998.**The definition of "NoSQL" stands for "Not Only SQL" or "Not Relational" [1].** The big data database systems are inspired by BigTable [2]- Persistent record Storage, memcached (in memory indexes), and Amazon's Dynamo [4] is the idea of eventual consistency . A key feature of NoSQL systems is "shared nothing" horizontal scaling and replicating and partitioning data over many servers. The proponents of NoSQL often cite Eric Brewers's CAP Theorem [8] :A distributed storage system must choose to sacrifice either consistency or availability while having partition tolerance . The term "BASE: Basically Available, Soft state, Eventually consistent" coined by Eric brewer [8] for these systems to handle the needs of internet and cloud based models of storage which abandon the ACID properties as a tradeoff for their increased performance and Scalability. The databases are grouped according to their strengths and CAP Theorem compromise by Big Data Working

Group [14] are relational, document –oriented, key-value, BigTable-inspired, Dynamo-inspired, graph and NewSQL. The Big data database systems are

Key-value Database Systems

The data model used by key-value systems is similar to memcached distributed in memory cache[5] in which a single key value for all the data. These systems generally provide a persistence mechanism, replication, versioning, locking, transactions, sorting, and client interface provides inserts, deletes and index lookup.

Project Voldemort

Project Voldemort[15] is an open source, advanced key- value database system written in Java with substantial contributions from LinkedIn and license under Apache 2.0. It provides MVCC for updates replicas asynchronously and guarantees an up-to-date view for read a majority of replicas. It supports optimistic locking for consistent multi record update by Vector clocks provide an ordering on version and automatic sharding of data by consistent hashing. Nodes can be added or removed from a database cluster, and the system adapts automatically and automatically detects and recovers failed nodes. It store data in RAM and also permits plugging in a storage engine (Berkeley DB and Random Access File storage).

Redis

Redis[16] is an open source BSD licensed), in-memory **data structure system**, used as database, cache and message broker written in ANSI C. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs and geo-spatial indexes with radius queries. A Redis server is accessed by a wire protocol implemented in various client libraries and the distributed hashing over servers by client side. The servers store data in RAM and copied to disk for backup. System shutdown may be needed to add more nodes. Redis implements insert, delete and lookup operations. It does atomic updates by locking and asynchronous replication, on disk persistence, high availability via Redis Sentinel, automatic partitioning by Redis cluster.

Document Database Systems

These systems store documents in form of articles, Microsoft Word files, etc. and a document is any kind of "pointer less object". Most of systems support secondary indexes and multiple types of documents (objects) per database, nested documents or lists and do not provide ACID transactional properties.

CouchDB

CouchDB [17] commonly refer as Apache CouchBD is an open source database with an architecture that completely embraces the web written in Erlang and license under Apache2.0. CouchDB store documents with metadata and maintain its own schema. It provides a RESTful HTTP for reading and updating documents with lockless and optimistic. It distributes data with incremental replication and supports automatic conflict detection. It is highly available and partition tolerance with eventually consistency. It provides scalability through asynchronous replication and not guarantees consistency through implementation of MVCC on individual documents.

MongoDB

MongoDB[18] is a open source document database written in C, C++and Java Script which license under GPL v3.0 open source and Apache 2.0 . Documents similar to JSON object as a data structure composed of field and value pairs. It provides high performance and data persistence by support of embedded data models and faster queries with indexes support and atomic operation on fields. It provides high availability by asynchronous replication facility, automatic failover and data redundancy. It provides scalability by sharding. It stores the data on disk and supports multiple storage engines WiredTiger and MMAPv1.

Extensible Record Database Systems

These systems are motivated by Google's BigTable [2] and Amazon's Dynamo [4]. The basic model is rows and columns in which rows are split across nodes by sharding and columns by columns grouping and their scalability model is spilting columns over nodes.

Hbase

Hbase[19] is a open source distributed hadoop database system written in Java which License under Apache 2.0. it provide Google's BigTable like capabilities on top of Hadoop and HDFS. It hosts very large tables, supports row operations with row level locking and transactions and updates into memory and periodically writes to files on disk by optimistic concurrency control with other updates. The partitioning and distribution are automatic and configurable sharding of tables with multiple master supports. It provides automatic fail over support, easy

to use Java API for client's access and linear and modular scalability.

Cassandra

Cassandra [20] is a open source distributed database system written in Java, License under Apache 2.0, supports by DataStax and originally open source by Facebook in 2008. It handles very large tables on commodity hardware and cloud infrastructure with no single point failure. It has column groups, updates are cached in memory and flushes to disk after writing log on disk, disk representation is compacted periodically. It provides no locking mechanism and replication asynchronously by MVCC, automatic failure detection and recovery. It uses phi accrual algorithm to detect a node failure and to determine cluster membership by a gossip style algorithms. It uses hash indexes and support versioning and conflict resolution.

Scalable Relational Systems

Relational DBMSs have a pre-defined schema, a query interface and ACID properties and not achieve scalability. By introduction of scalable relational system which provide scalability and avaliabilty and good per-node performance.

MySQL Cluster

MySQL Cluster [21] is a distributed shared nothing system written in C and C++ has a part of the MySQL and license under GNU General Public License and a standard commercial License from Oracle. It integrates MySQL server with in- memory cluster storage engine called NDB (Network Database). It refers to a part of the setup that is specific to storage engine. It provides real-time response time, high throughput, auto sharding for write scalability by partitions tables across nodes and 99.999% availability. It provides Synchronous Replication, automatic failover, Self healing and Geographical Replication with no single point failure.

VoltDB

VoltDB [22] is a distributed shared nothing system written in Java and C++, license under GNU affero Public License v3 and voltDB proprietary License. It is a in-memory NewSQL System in which tables are partitioned over multiple nodes and replicated synchronously. It provides good per-node performance and scalability and availability, and Query processing is single threaded for each shard with transaction consistency (ACID) of traditional relational systems.

3. Literature Review

Rick Cattell [1] has done a comprehensive survey on Scalable SQL and NoSQL databases. In this paper, He classify these system on their data model, consistency control, data storage, durability, availability, query support, and other dimensions into key-value, document, extended record and relational.

Tudorica, Bogdan George, and Cristian Bucur[10] has done a comparison between several NoSQL databases with comments and notes. This paper is trying to comment on the various NoSQL systems and to make a comparison based on qualitative and quatitative point of view between Cassandra, Hbase and MySQL. The qualitative view or criteria compare features are persistence, replication, high availability, transactions, rack-locality awareness, implementation language, influences / sponsors and license type . The quantitative evaluation criteria or view based on two sets, one related to size(number of records/rows/document store, number of node in an installation) and other related to performance(Read and write latency in both write and read intensive environment). The conclusion shows the SQL and the NoSQL databases are having some shared features are not similar in a given instances. These systems cannot be used interchangeable for solving any type of problem, but choose between the two types of databases for a given instance.

Jing han et al. [9] has done a survey on NoSQL database. This paper describes the background, basic characteristics, data model of NoSQL, classifies NoSQL databases according to the CAP theorem, describe the mainstream NoSQL databases and on the basis of properties to help enterprises to choose NoSQL. Based on the above knowledge of the mainstream NoSQL databases companies decide whether to use NoSQL. In their study observed that companies need to consider the following options when deciding which properties NoSQL are Data Model, CAP Support, Multi Data Center Support, Capacity, Performance, Query API, Reliability, Data Persistence, Rebalancing and Business Support.

Santhosh Kumar Gajendran [6] has done a survey on nosql database. The goal of this survey is to understand the current needs that have led to the evolution of NoSQL databases, why relational database systems were not able to meet these requirements and a brief discussion of some of the successful NoSQL data stores. In their study,

common concepts underlying these databases and how they compromise on ACID properties to achieve high scalability and availability. The NoSQL databases Dynamo, voldemort, CouchDB, MongoDB, BigTable, HBase and Cassandra based on License type, concurrency control, data storage and replication are surveyed . Each database and its implementation has strengths at addressing specific enterprise or cloud concerns such as being easy to operate, providing a flexible data model, high availability, high scalability and fault tolerance.

Manoj V [11] has done a comparative study on NoSQL databases are Cassandra, MongoDB and Hbase on basis of architecture and working. The parameter of study are classification, architecture, availability, data model, partitioning and evaluation of Cassandra as industry use case. In their study that MongoDB fits for use cases with document, document search and aggregation functions are mandate. HBase suits the scenarios in which hadoop map reduce is useful for bulk read and load operations and offers optimized read performance with hadoop platform. Cassandra can be used for applications requiring faster writes and high availability.

4. Objective and Scope of the Study

In this paper, a comparative study on different database systems specially NoSQL and Scalable SQL in which discuss about classification, architectural properties and functionality. The parameters of study are License type, Implementation Language, Influences/ sponsors, persistence, Concurrency control, Data storage, Replication and Transaction Mechanism. The Graph and object oriented database systems are beyond our scope.

5. Research Methodology

The research methodology followed a theoretical approach that comprises literature survey, articles, books, research papers and internet.

6. Analysis and results

The study emphasis on the comparative study of NoSQL and Scalable SQL systems are Project Voldemort, Redis, CouchDB, MongoDB, HBase, Cassandra, MySQL Cluster and VoltDB. The comparative study or criteria compare parameters are License type, Implementation Language, Influences/ sponsors, persistence, Concurrency control, Data storage, Replication and Transaction Mechanism.

Table 1: Comparison of Big Data Database Systems on Basic Parameters

DATA BASE MODEL	DATA BASE SYSTEM	PARAMETERS			
		License type	Implementation Language	Influences/ Sponsors	Persistence
Key-value	Project voldemort	Apache 2.0	Java	LinkedIn	Yes
	Redis	BSD open Source	C	Eredis, redox, carmine	Yes
Document	CouchDB	Apache 2.0	Erlang	Apache	Yes
	MongoDB	GNU General Pubic Licence v3.0 open source, Apache 2.0	C++, C, Java Script	10gen, linkedIn, FourSqure	Yes
Extended Record	HBase	Apache2.0	Java	BigTable	Yes
	Cassandra	Apache2.0	Java	BigTable, Amazon's Dynamo, facebook	Yes

Scalable MySQL/ Relational	MySQL Cluster	GNU General Public License v3.0, a standard commercial License from Oracle	C/C++	Oracle	Yes
	VoltDB	GNU General Public License v3.0	Java	SQL / Hstore LMAX	Yes

The Apache 2.0 license type Systems are Project Voldemort, CouchDB, MongoDB, HBase, Cassandra, BSD open source is Redis, GNU General Public License v3.0 are MongoDB, MySQL Cluster and VoltDB, and a standard commercial License for Oracle is MySQL Cluster.The systems are implemented in different languages, in Java implemented systems are Project Voldemort, HBase, Cassandra and VoltDB, C and C++ implemented systems are Redis, MongoDB and MySQL Cluster, Erlang implemented is CouchDB and Java Script Implemented System is MongoDB. Persistence in term of durability, failure detection and recovery is provided by all systems.

Table 2: Comparison of Big Data Database Systems on Functional Parameters

DATA BASE MODEL	DATA BASE SYSTEM	PARAMETERS			
		Concurrency control	Data Storage	Replication	Transaction Mechanism
Key-value	Project voldemort	MVCC	RAM or Plug in a storage engine	Asynchronous	Not
	Redis	Locks	RAM	Asynchronous	Not
Document	CouchDB	MVCC	Disk	Asynchronous	Not
	MongoDB	Locks	Disk	Asynchronous	Not
Extended Record	HBase	Locks	Hadoop	Asynchronous	Local
	Cassandra	MVCC	Disk	Asynchronous	Local
Scalable MySQL / Relational	MySQL Cluster	ACID	Disk	Synchronous	Yes
	VoltDB	ACID, no lock	RAM	Syschronous	Yes, threading on shards

The concurrency control MVCC is provide by Project Voldemort, CouchDB and Cassandra, Locks is provide by Redis, MongoDB and HBase, and ACID is provide by MySQL Cluster and VoltDB. The data storage RAM is use by Project Voltemort, Redis and VoltDB, disk is use by CouchDB, MongoDB, Cassandra and MySQL Cluster. Asynchronous replication is provide by all NoSQL Systems are Project Voltemort, Redis,CouchDB, MongoDB, HBase and Cassandra, Synchronous Replication is provided by MySQL Cluster and VoltDB. Transaction mechanism are supported by HBase, Cassandra, MySQL Cluster and VoltDB, and not in Project Voltemort, Redis, CouchDB and MongoDB.

7. Conclusion and Future Scope

In this Paper, a comparative study of various NoSQL and Scalable Relational Systems based on their classification and architectural properties and features. Each system has its own specific use and no one fit for all requirements. Each of them has been motivated by varying requirements which has led to their development mostly from the industry. The Project Voldemort, Redis, HBase, Cassandra MySQL Cluster and VoltDB is suited well for structured data. CouchDB and MongoDB is well suited for unstructured data. In Future, there will be scope of doing quantitative analysis based on throughput, number of node it handle by systems and on various algorithms used for Concurrency control, failure detection and transaction mechanism.

References

1. Cattell, Rick. "Scalable SQL and NoSQL data stores." ACM SIGMOD Record39.4 (2011): 12-27.

2. Chang, Fay, et al. "Bigtable: A distributed storage system for structured data." ACM Transactions on Computer Systems (TOCS) 26.2 (2008): 4.

3. Dean, Jared. Big data, data mining, and machine learning: value creation for business leaders and practitioners. John Wiley & Sons, 2014.

4. DeCandia, Giuseppe, et al. "Dynamo: amazon's highly available key-value store." ACM SIGOPS Operating Systems Review. Vol. 41. No. 6. ACM, 2007.

5. Fitzpatrick, Brad. "Distributed caching with memcached." Linux journal2004.124 (2004): 5.

6. Gajendran, Santhosh Kumar. "A survey on nosql databases." *University of Illinois* (2012).

7.Gantz, John, and David Reinsel. "Extracting value from chaos." DC iview1142 (2011): 1-12.

8. Gilbert, Seth, and Nancy Lynch. "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services." ACM SIGACT News33.2 (2002): 51-59.

9. Han, Jing, et al. "Survey on NoSQL database." *Pervasive computing and applications (ICPCA), 2011 6th international conference on*. IEEE, 2011.

10 Tudorica, Bogdan George, and Cristian Bucur. "A comparison between several NoSQL databases with comments and notes." Roedunet International Conference (RoEduNet), 2011 10th. IEEE, 2011.Lohr, Steve. The age of big data.New York Times11 (2012).

11.Manoj, V. "Comparative study of nosql document, column store databases and evaluation of cassandra." *International Journal of Database Management Systems* 6.4 (2014): 11.

12. Marz, Nathan, and James Warren.Big Data: Principles and best practices of scalable real time data systems. Manning Publications Co., 2015.

13.“Big Data” http://www.gartner.com/it-glossary/big-data/

14.“BigDataTaxonomy” https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Big_Data_Taxonomy.pdf

15. “ Project Voldemort-a distributed database” http://www.project-voldemort.com/voldemort/

16.“Redis” http://redis.io/

17. “Apache CouchDB 1.6 Documentation” http://docs.couchdb.org/en/1.6.1/

18. “The MongoDB 3.2 Manual ” https://docs.mongodb.org/manual/

19. “Apache HBase ™ Reference Guide” https://hbase.apache.org/book.html

20. “Apache Cassandra 2.2” http://docs.datastax.com/en/cassandra/2.2/cassandra/cassandraAbout.html

21. “ MySQL Cluster” https://dev.mysql.com/doc/index-cluster.html

22.“VOLTDB Documentation” https://docs.voltdb.com/