# STUDY OF VARIOUS NoSQL DATABASES

Amandeep Kaur*[1] and Kanwalvir Singh[2]

*[1]CSE, BBSBEC, Fatehgarh Sahib, Punjab, India
Amandeepkaur3farwaha@gmail.com[1]
[2]CSE, BBSBEC, Fatehgarh Sahib, Punjab, India
kanwalvir.singh@bbsbec.ac.in[2]

***Abstract*—**A relational database is a table-based system where there's no scalability, lowest data duplication, computationally overpriced table joins and issue in addressing complicated data. The matter with relations in relationaldatabase is that advancedoperations with massive data sets quickly become prohibitively resource intense. Relational databases don't lend themselves well to the type of horizontal scalability that is needed for large - scale social networking or cloud applications. NoSQL has emerged as a results of the demand for relational database alternatives. The most important motivation behind NoSQL is scalability. NoSQL is supposed for the present growing breed of netapplications that require to scale effectively. This paper analyzes the NoSQL databases that is the demand of the present large-scale social networking or cloud applications. We tend toadditionally analyze the capabilities of assorted NoSQL models like BigTable, Cassandra, CouchDB, MongoDB and Couchbase.

**Keywords**—NoSQL,Scalability

## INTRODUCTION

A relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as introduced by E. F. Codd. It supports a tabular structure for the data, with implemented relationships between the tables. Preferred business and open source databases presently in use are based on the relative model. The problem with RDBMS is not that they do not scale, it's that they're incredibly hard to scale. The foremost common RDBMS are Microsoft SQL Server, DB2, Oracle, MYSQL etc.

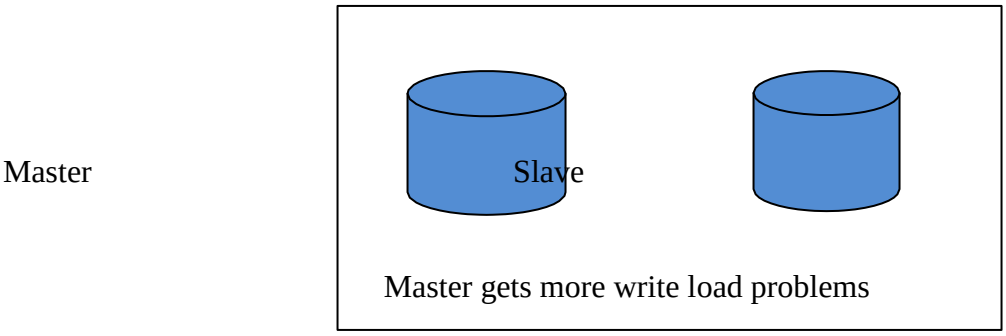
Many web applications merely don't need to represent data as a group of connected tables which means all applications need not to be a traditional relational database management system (RDBMS) that uses SQL to perform operations on data. Rather, data can be stored in the form of objects, graphs, documents and retrieved using a key. For instance, a user profile will be drawn as associate object graph (such as pojo) with one key being the user id. Another example: documents or media files can be stored with a single key with indexing of metadata handling by a separate search engine.

These types of data storage are not relational and lack SQL, however they may be quicker than RDBMS as a result of they do not have to maintain indexes, relationships, constraints and parse SQL. Technology like that has existed since the 1960s (consider, as an example, IBM's VSAM file system).

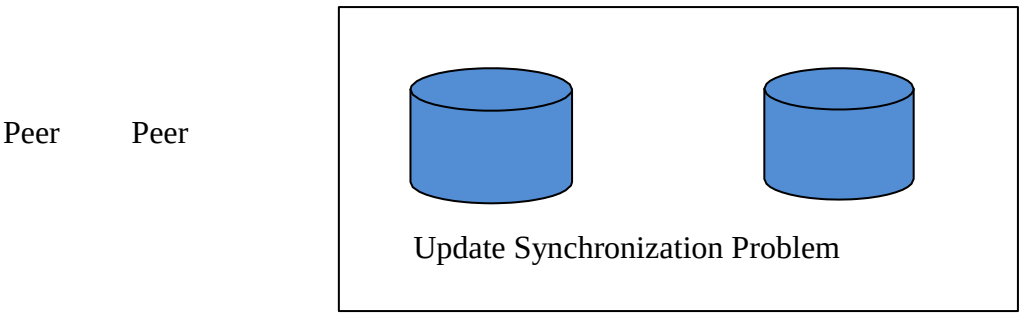
Relational databases are able to handle countless products and service very large sites. However, it is difficult to create redundancy and parallelism with relative databases, so that they become one purpose of failure. Especially, replication isn't trivial. To understand why, take into account the matter of getting 2 database servers that require to own identical data. Having both servers for reading and writing knowledge makes it difficult to synchronize changes. Having one master server and another slave is unhealthy too, as a result of the master has got to take all the warmth once users square measure writing information. So as arelational database grows, it becomes a bottleneck and therefore the purpose of failure for the complete system. As mega e-commerce sites grew over the past decade

they became conscious of this issue - adding a lot of web servers doesn't facilitate as a result of it is the database that finishes up being a problem.

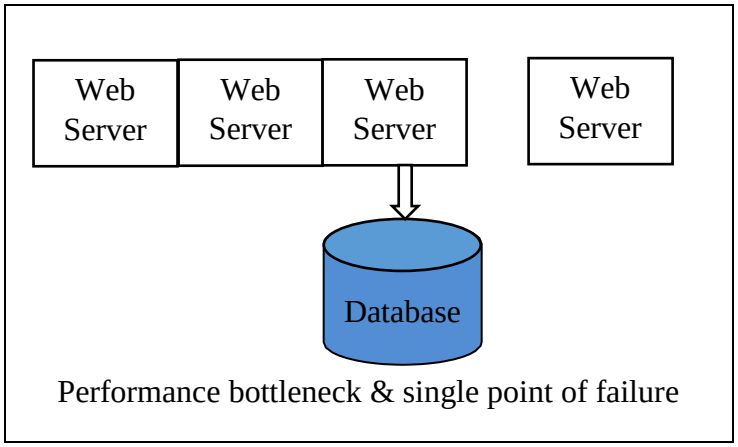
*A. MasterSlaveSetup*




*B. Multiple peers setup*




*C. Single Database Server Setup*




## LITRATURE SURVEY

Moniruzzaman and Hossain [1] give - classification, characteristics and analysis of NoSQL databases in massive data Analytics. The report is meant to assist users, particularly to the organizations to get an independent understanding of the strengths and weaknesses of assorted NoSQL information approaches to supporting

applications those method Brobdingnagian volumes of information. The study report motivated to offer an independent understanding of the strengths and weaknesses of varied NoSQL database approaches to supporting applications that method vast volumes of data; similarly on provide a worldwide summary of the non-relational NoSQL databases.

Lourenço et al. [8] highlighted the performance comparison of various NoSQL databases. In the paper, author had gathered a brief and up-to-date comparison of NoSQL engines, their most useful use case situations from the programmer viewpoint, their benefits and disadvantages by measuring the presently on the market literature. In the paper author concluded that though there are a spread of studies and evaluations of NoSQL technology, there's still not enough data to verify however suited every non-relational database is during a specific state of affairs or system. Moreover, every operating system differs from one another and the required functionalities and mechanisms extremely have an effect on the database selection. Typically there's no chance of clearly stating the simplest information answer.

Chitra and Jeevarani [10] focused primarily on the market, scalable and Eventually Consistent NoSQL Databases. The paper analyses the requirement of following generation data storage that is that the need of the present large-scale social networking or cloud applications additionally author analyze the capabilities of assorted NoSQL models like BigTable, Cassandra, CouchDB, generator and MongoDB. During the paper author conclude that NoSQL databases usually method data faster than relative databases. Developers typically don't have their NoSQL databases supporting ACID properties, so as to extend performance, however this will cause issues once used for applications that need nice exactness.

Sharma and Dave [19] discussed about NoSQL, its background, fundamentals like ACID, BASE and CAP theorem. The main aim of the paper is to give a summary of NoSQL databases, regarding however it's declined the dominance of SQL, with its background and characteristics. It also describes its fundamentals that type the bottom of the NoSQL databases like ACID, BASE and CAP theorem. ACID property isn't utilized in the NoSQL database thanks to data consistency therefore we get to understand however SQL lags data consistency.

Kaisler et al. [15] presented an Introduction to massive Data: Challenges, Opportunities and Realities. Big data remains a maturing and evolving discipline. Big data databases and files have scaled on the far side the capacities and capabilities of business direction systems. Structured representations become a bottleneck to economic data storage and retrieval. Author concludes that increasing variety of disciplines and drawback domains wherever big data has a sway and one sees a rise within the variety of challenges and opportunities for big data to own a serious impact on business, science, and government.

Pore and Pawar [17] presented comparative Study of SQL & NoSQL Databases. The main aim of analysis the paper is to judge the fundamentals of SQL and NoSQL databases and therefore the comparative analysis of those 2 databases. The paper additionally describes the Axiomatics of SQL and NoSQL databases. ACID property isn't utilized in the NoSQL databases. The paper additionally describes samples of SQL databases and kinds of NoSQL databases on the premise of CAP Theorem. Databases are horizontally ascendible just {in case} of NoSQL databases and vertically ascendible in case of SQL databases. Performance of each the information is counting on the database size and therefore the variety of queries which is able to be performed by the applications.

Truica et al. [4] describes performance analysis for CRUD operations in asynchronously replicated document oriented database. The paper examines asynchronous replication, one among the key options for an ascendible and flexible system. Three of the most in style Document-Oriented Databases, MongoDB, CouchDB, and Couchbase, are examined. Author concludes that though CouchDB performs alright for the insert, update and delete, MongoDB is that the quickest once it involves attractive knowledge. Overall, the NoSQL databases perform higher than the relational ones.

V et al. [18] highlighted comparative study of NoSQL database. The aim of the paper is to explore NoSQL technologies and present a comparative study of document and column store NoSQL databases like Cassandra, MongoDB and Hbase in varied attributes of relative and distributed information system principles. Author concludes that mongodb fits to be used cases with document storage, document search and wherever aggregation functions are mandate. Hbase suits the eventualities wherever Hadoop map scale back is helpful for bulk read and load operations Hbase offers optimized scan performance with hadoop platform.

## VARIOUS NoSQL DATABASES

Organizations that collect large amounts of unstructured data are increasingly turning to non-relational databases, now frequently called NoSQL databases:

**1. *Bigtable:*** *A Distributed Storage System for Structured Data*

Bigtable is Google's internal database system. Bigtable is a distributed storage system for managing structured data that's designed to scale to a really massive size i.e. computer memory units (1 petabyte = one.12589991 × 1015 bytes) of data across thousands of commodity servers. Several projects at Google store data in Bigtable, including web indexing, Google Earth, and Google Finance. These applications place terribly different demands on Bigtable, each in terms of data size (from URLs to web pages to satellite images) and latency requirements (from backend bulk processing to real-time data serving). Despite these varied demands, Bigtable has successfully provided a versatile, high -performance answer for all of those Google products.

A Bigtable is a distributed and persistent multidimensional sorted map. The map is indexed by a row key, column key, and a timestamp; every value within the map is an uninterpreted array of bytes. Every cell in a Bigtable (like field in DBMS) can contain multiple versions of a similar data; these versions are indexed by timestamp (Microseconds). Bigtable timestamps are 64-bit integers. Different versions of a cell are stored in decreasing timestamp order, so the foremost recent versions are often read first. Bigtable depends on a cluster management system for scheduling jobs, managing resources on shared machines, addressing machine failures, and observing machine status.

**2. *Cassandra:*** *A Decentralized, highly scalable, eventually consistent DB*

A decentralized, extremely scalable, eventually consistent database Cassandra is extremely reliable second-generation distributed database. Cassandra was open sourced by Facebook in 2008 and is presently being developed as an Apache incubator project. The system offers a fault tolerant, high availableness, decentralized store for information which may be scaled up by adding hardware nodes to the system. Cassandra implements an "eventually consistent" model that trades-off consistency of data stores within the system for availableness. Information is automatically replicated to multiple nodes for fault-tolerance. Replication across multiple information centres is supported. Unsuccessful nodes may be replaced with no period. Cassandra is in use at Rackspace, Digg, Facebook, Twitter, Cisco, Mahalo, Ooyala, and additional corporations that have massive, active information sets. The biggest production cluster has over a hundred TB of knowledge in over a hundred and fifty machines.

***3. CouchDB:*** *Extremely scalable, highly available and reliable storage system*

CouchDB, is a free and open source document-oriented database written within the erlang programing language for its emphasis on fault tolerance, accessible using a restful JavaScript Object Notation (JSON) API. The term "Couch" is an acronym for "Cluster Of Unreliable commodity Hardware", reflective the goal of CouchDB being extremely scalable, providing high availability and reliability, even whereas running on hardware that's usually vulnerable to failure.

Hence CouchDB is

- A document database server

- Ad-hoc and schema-free with a flat address space
- Distributed that includes strong, incremental replication with bi-directional conflict detection and management.
- Highly available even if hardware fails
- Query-able and index-able, featuring a table oriented reporting engine that uses JavaScript as a query language

CouchDB is not

- A relational database
- An object-oriented database. Or more specifically, meant to perform as a seamless persistence layer for an OO programming language

**4. *MongoDB***

It is a document oriented database that has high performance, high accessibility, and easy scalability. MongoDB database is simple to introduce that makes reads and writes quick. This database uses the indexes that include keys from documents and arrays. This provides the high availability for higher performance and really simple to scale and easy to manage the operations. MongoDB stores the data into documents and collections rather than storing data in table as rows and columns. Collections enable representation of advanced relationships simply. Is has the potential to handle the big volume of data and may load data across a cluster. It will perform several operations that relational database cannot do.

There are some following features of MongoDB

- Map reduce and Aggregation Tools are supported by MongoDB.
- Java Scripts is used rather than Procedures
- MongoDB is a schema less Document based database.
- MongoDB give the power to use secondary indexes and geospatial indexes.
- Simple to handle the MongoDB in cases of failures.
- MongoDB designed to produce high performance.
- MongoDB stores files of any size without all the way down to failure of memory.

***5. Couchbase***

Couchbase is an open source NoSQL database that can be used as either a document-oriented or pure key-value database and is supported by Couchbase Inc. and authorized under the Apache 2.0 license. It aims for simple scalability, consistent high performance, high reliableness and easy development. When used as a document-oriented database, data is stored in JSON format which can be indexed and queried.

While Couchbase took inspiration from Apache CouchDB and memcached, it's a completely different and separate open source project. It a separate and independent community, provides a very different set of capabilities, and supports very different use cases. More specifically, Couchbase leveraged and changed memcached technology to provide inbuilt caching and leveraged and modified Apache CouchDB technology to enable the document capabilities in recent releases of Couchbase.

In the formulation of Eric Brewer's CAP theorem, Couchbase is a CP kind system that means it provides consistency and partition tolerance. But Couchbase Server are often established as an AP (availability and partition tolerance) system with multiple clusters using XDCR (Cross data Center Replication)

## ANALYSIS OF NoSQL DATABASES

In this section we compare and analyze the NoSQL databases named BigTable, Cassandra, CouchDB, MongoDB and Couchbase.

Table I: Analysis of NoSQL Databases

| NoSQL DB's | Bigtable | Cassandra | CouchDB | MongoDB | Couchbase |
|---|---|---|---|---|---|
| **DB type** | Key-Value store DB | Column Oriented DB | JSON Document Oriented DB | BSON Document Oriented DB | Document, Key-Value |
| **Developer** | Google | Apache | Apache | 10gen | Couchbase |
| **Scalability** | Highly Scalable | Highly Scalable | Easily scalable and readily extensible | Scalable | Elastic scalability |
| **Availability** | Highly Available | High availability is achieved using replication | Highly Available | High write availability | Highly Available |
| **Performance** | High Performance | High Performance at massive scale | Loading speeds are better than retrieval speeds | Excellent solution for short read | Consistent high Performance |
| **Consistency** | Eventual Consistency | Eventual Consistency Immediate Consistency | Eventual Consistency | Eventual Consistency Immediate Consistency | Eventual Consistency Immediate Consistency |
| **Reliability** | Provides reliability at a massive scale | At massive scale is a very big challenge | Excellent solution for short read | Avoid growing documents unsafe writes by default | Better reliability |
| **Flexibility** | High | Moderate | High | High | High |
| **Complexity** | None | Low complexity | Low complexity | Low complexity | Very low complexity |
| **Best Use** | Designed to scale hundreds or thousands of machines | Write often, read less | Accumulating, occasionally changing data with predefined queries | Dynamic queries, frequently written, rarely read statistical data | Session store, user profile store, content store |

## CONCLUSION

NoSQL Databases largely address some of the points: being non-relational, distributed, open-source and horizontal scalable. The original intention has been modern web-scale databases. The main characteristics of NoSQL Databases are: schema-free, replication support, easy API, eventually consistency, and more. Thus the deceptive term "NoSQL" is currently translated to "Not only SQL (NOSQL)". NOSQL databases typically process data quicker than relational databases. Developers usually don't have their NOSQL databases supporting ACID properties, so as to extend performance, however this can cause issues when used for applications that need great precision. NOSQL databases are usually faster as a result of their data models are less complicated. Several leading NOSQL systems are flexible enough to permit developers to use the applications in ways that fulfil their needs.

## REFERENCES

[1] A. B. M. Moniruzzaman and S. A. Hossain, "NoSQL Database : New Era of Databases for Big data Analytics - Classification , Characteristics and Comparison", International Journal of database theory and Application, Vol. 6, No. 4, pp. 1–14, 2013.

[2] A. Ron, B. Sheba, and A. Shulman-peleg, "No SQL , No Injection ? Examining NoSQL Security", 8th International Conference on Databases, IEEE, California, US, 2015.

[3] B. Saraladevi, N. Pazhaniraja, P. V. Paul, M. S. S. Basha, and P. Dhavachelvan, "Big Data and Hadoop-a Study in Security Perspective", *2*nd International Symposium on Big Data and Cloud Computing, Vol. 50, pp. 596–601, 2015.

[4] C.-O. Truica, F. Radulescu, A. Boicea, and I. Bucur, "Performance Evaluation for CRUD Operations in Asynchronously Replicated Document Oriented Database", 20th International Conference on Control Systems and Computer Science, IEEE, pp. 191–196, 2015.

[5] G. Aydin, I. R. Hallac, and B. Karakus, "Architecture and implementation of a scalable sensor data storage and analysis system using cloud computing and big data technologies", Hindawi Journal of Sensors, Vol. 9, No. 02, pp. 1-11, 2015.

[6] I. A. T. Hashem, I. Yaqoob, N. Badrul Anuar, S. Mokhtar, A. Gani, and S. Ullah Khan, "The rise of 'Big Data' on cloud computing: Review and open research issues", Information Systems, Vol. 47, No. 7, pp. 98–115, 2014.

[7] J. Pokorny, "NoSQL Databases : a step to database scalability in Web environment", International Conference on WEB Information Systems, 2015,Vol.9, No.1,pp. 69-82, 2013.

[8] J. R. Lourenço, V. Abramova, M. Vieira, B. Cabral, and J. b. Bernardino, "NOSQL databases: A software engineering perspective", Advances in Intelligent Systems and Computing, Springer, Vol. 353, No. 6, pp. 741–750, 2015.

[9] K. Barmpis and D. S. Kolovos, "Evaluation of Contemporary Graph Databases for Efficient Persistence of Large-Scale Models", Journal of Object Technology, Vol. 13, No. 3, pp. 1-26, 2014.

[10] K. Chitra and B. Jeevarani, "Study on Basically Available, Scalable and Eventually Consistent NOSQL Databases", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, No. 5, pp. 991–996, 2013.

[11] K. Zvarevashe and T. T. Gotora, "A Random Walk through the Dark Side of NoSQL Databases in Big Data Analytics", International Journal of Science and Research, Vol. 3, No. 6, pp. 506-509, 2014.

[12] M. V, "Comparative Study of NOSQL Document , Column Store Databases and Evaluation of Cassandra", International Journal of Database Management Systems, Vol. 6, No. 4, pp. 11–26, 2014.

[13] P. Soni and N. S. Yadav, "Quantitative Analysis of Document Stored Databases", International Journal of Computer Applications, Vol. 118, No. 20, pp. 37–41, 2015.

[14] R. Aniceto and R. Xavier, "Evaluating the Cassandra NoSQL Database Approach for Genomic Data Persistency", Hindawi Publishing Corporation International Journal of Genomics, Vol. 25, No. 03, 2015.

[15] S. Kaisler, F. Armour, and J. A. Espinosa, "Introduction to Big Data: Challenges, Opportunities, and Realities Minitrack", 47th Hawaii International Conference on System Sciences (HICSS), pp. 728–728, 2014.

[16] S. Madden, “From Databases to Big Data”, IEEE Computer Society, Vol. 16, No. 3, pp. 4–6, 2012.

[17] S. S. Pore and S. B. Pawar, “Comparative Study of SQL & NoSQL Databases”, International Journal of Advanced Research in Computer Engineering & Technology, Vol. 4, No. 5, pp. 1747–1753, 2015.

[18] V. A. A, J. B. A, and P. F. B, “Which NoSQL Database? A Performance Overview”, Open Journal of Databases, Vol. 1, No. 2, pp. 17–24, 2014.

[19] V. Sharma and M. Dave, “SQL and NoSQL Databases”, International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 2, No. 8, pp. 20–27, 2012.