A Technical Insight on the New Generation Databases: NoSQL

Nikhila T Bhuvan
Rajagiri School of Engineering & Technology
Rajagiri Valley, Kakkanad,
Kochi - 682 039, Kerala, INDIA

M Sudheep Elayidom
Cochin University of Science and Technology (CUSAT)
Thrikkakara, South Kalamassery, Kochi, Kerala 682022

ABSTRACT
The rapid acceptance of social networking sites like Facebook, Twitter etc and with the huge use of mobile phones, the data volume is increasing at the rate of terabytes. As the data size and variety is increasing dramatically, databases intend to be more schema-less and less strict, in order to achieve higher scalability. Their queries tend to be quick responding, which grants them very fast writes and even faster reads. All these could not be managed with our Relational Databases or SQL. The ease of scalability and improved security has increased the popularity of Not-Only SQL. But they are not as reliable or effective as Relational Databases. Here, in this paper, an extensive study of the pros and cons of SQL and NoSQL, their relevance in cloud and at last, an evaluation of some NoSQL databases is also done.

General Terms
NoSQL databases, Cloud and Big data.

Keywords
HBase, Cassandra, Dynamo, BigTable, MongoDB

1. INTRODUCTION
Big data, data varying in size, volume and type is generated as formerly non-digitized processes become digitized. The services like online banking, online shopping, e-learning, e-mails, instead of the traditional methods is increasing day by day[1]. Every one of us carry a mobile phone obviously with cameras, laptops, takes pictures and videos ,uploads it to social networks which keeps on adding to the huge bulk of data. We generate data as we willingly share information with the world through social media in the form of tweets, comments, likes and many more.

Decreases in the cost of both storage and data capture technology is the one that make up this new era of data revolution. It would be a tedious task to move these terabytes of data generated to a centralized location for processing or for storage, so better to organize it in a distributed environment, where an amount of data will stay where it was originally created and be transparently accessed from a data warehouse. The ”one size fits all“-data stores cannot be adopted for this distributed processing as the data produced will vary in Volume, Variety and Velocity. [3]This has lead to the emergence of a great variety of alternative databases. This drift towards the new data stores is commonly subsumed under the term NoSQL.

Google’s BigTable, based on distributed Google File System(GFS) implemented in 2006 is the pioneer among these series which increased the popularity of NoSQL databases. The highly distributed ever-growing collection of semi-structured data all around the world was an issue for Google, which got fixed with the help of BigTable[2]. Google’s BigTable provides a simple data model which gives client the flexibility over the data layout and format.

2. CLOUD AND NoSQL
Large amounts of data can be handled by traditional or conventional databases as long as a structuring process is undergone by the data. The analysis of this large volume of data requires on-demand compute power and distributed storage to crunch the 3Vs(Volume, Velocity, Variety) data problem and Cloud seamlessly provides this elastic on-demand computing required for the same. Cloud has glorified the "As-a-Service" model by concealing the complexity and challenges involved in building a scalable elastic self-service application which could solve at least some problems associated with Big Data[1].

The rise of cloud computing in the business and other social networking fields also valued NoSQL (Not-Only SQL) based implementations rather than traditional databases[2]. The use of infrastructure-as-a service model increased the reliability, scalability and accessibility of data over dedicated or shared hosting solutions. The challenge for database administrators has been to choose which among the series of ever emerging databases to implement when hosting in the cloud. Documentation, simplicity, familiarity, data integrity and reliability are some of the advantages of relational databases compared to the non relational. Handling Complex query handling and aggregations works well for SQL implementation than other solutions but the drawbacks come when a developer wants to scale the database across multiple servers. SQL injection is one of the loopholes in Relational databases in cloud, whereas the NoSQL is able to avoid it to an extend.

The NoSQL systems generally do not provide ACID (Atomicity, Consistency, Isolation and Durability) transactional properties, rather follows a BASE (Basically Available, Soft state, Eventually consistent) property, that is the updates are eventually propagated, but there is limited guarantee on the consistency of read. By giving up ACID constraints, much higher performance and scalability can be achieved. However, the systems differ in how much they give up.

It is often said that NoSQL systems can have only one out of three properties (CAP): Availability, Consistency and Partition tolerance[4]. A NoSQL system usually compromises on its consistency. They could be said as eventually consistent, that is, the nearest document would be updated first and saved, then progressively updating the other copies in the cloud. That means, two people trying to access the same copy of the data at the same time would be provided with different images. This would be a disadvantage for the real time applications but for the systems like social media it would be acceptable.

Availability is considered as the important factor for the NoSQL implementation. The redundant copies of the document across the cloud would make it always available and helps to protect against data loss. The property of horizontal scalability allows distributing data and operations over many servers anywhere in the world. The relational databases allow vertical scalability where they utilizes many cores or CPU that shares RAM and disk. In horizontal scalability these multiple cores are effectively utilized, as number of core that share RAM is limited.

Table1. The SQL and NoSQL database difference summarized

Properties	Relational Database	NoSQL Database
Availability	Less	High
Scalability	Vertically scalable	Horizontally scalable
Consistency	Always consistent	Eventually consistent
Security	Much prone to attacks	Less prone to attacks
Data loss	High	Multiple copies of the same data kept
Data Storage Model	Individual records stored as rows	Varies based on the database type. Explained in the next section
Type of data	Structured Data	Semi Structured or unstructured data
Time required	Need much time to design and normalize a database	Minimal investment in design and maintenance
Queries	Complex	Simple
Usage	Small frequent Read/Write	Intensive Read/Write

3. NoSQL CLASSIFICATION

NoSQL databases can be classified into 4 types[5]:

Key-Value store: This type can be considered as the mother of all NoSQL databases. Data here is stored as a pair of key and a value. The Key is the unique identifier. The database querying is done with the help of a key. All the keys in any data objects will be alpha-numeric. The simple nature of Key-Value Stores makes them best suited for lightning-fast and highly-scalable application tasks like session managing or user profile managing or retrieving product names etc. Dynamo, the K-V system of Amazon is used in their shopping cart. Dynamo is used by some of Amazon‘s core services use to provide highly available and scalable distributed data store. Examples: Key-Value Stores- Dynamo (Amazon); Voldemort (LinkedIn); Redis; BerkeleyDB; Riak.

Document store: This type of database store unstructured (text) or semi-structured (XML or JSON) document which are schema free and are not fixed in nature. This type of storage is more complex when you compare with that of Key value pair. A single column can store number of attributes, but the number and type of attributes recorded may vary from row to row. In document databases any attribute can be used for searching, both keys and values. These databases are good for storing and managing large data collections of documents, like text files, emails, WebPages and XML documents etc which can have large variety of data within it. As you know a webpage can have video files, audio files, some text fields, images and so on. They are also good for storing sparse data, that is semi structured data, which makes us use to define those fields as “nulls” in relational databases Some of the examples are : MongoDB ,CouchDB, [2]Google BigTable etc.

Wide- Column Family : Column-oriented data structure accommodates multiple attributes per key. Distributed data storage, Large-scale batch-oriented data storing and processing can be handled by this type of data model. Exploratory and predictive Big Data analytics could also make use of the wide columnar architecture as such kind of works would pour in different kind of data. MapReduce is the processing model used here. Facebook. When Facebook expanded its messaging services infrastructure to include email, text, instant messages, and more, they created Cassandra, combining the elements of Google’s BigTable and Amazon’s Dynamo. Apache HBase is another NoSQL database with strong consistency, load balancing and automatic failover.

Graph database: Even though we store multiple data in different data models, the relationship between the data are unknown. The graph databases could be used to model the network of relationship between specific data element. It facilitates the efficient management of densely linked data. This is most widely used in social network analysis. Some examples are Twitter FlockDB, Neo4j, HyperGrph DB.[3]

4. EVALUATION OF NoSQL DATABASE

The Yahoo! Cloud Serving Benchmark (YCSB), is a benchmarking framework developed by Yahoo! It provides an environment to stress-test multiple databases and compares them. MongoDB, HBase and Cassandra are compared using it[5]. Cassandra, providing an extended key-value store belongs to wide-column store NoSQL family. HBase, based on the Hadoop map-reduce framework and Hadoop Distributed File System (HDFS) also belongs to the wide column store. MongoDB is basically a document-store

NoSQL database which can be used as Key-value store or wide columnar store when ever required. Graph databases cannot be evaluated using YCSB as usage of links between data records requires a different approach. There are specific benchmarks like XGDBench[7] to evaluate the performance of Graph databases.

The first criterion to be analyzed is the performance of the database as the number of cores on a single node is increased. HBase is able to take advantage of the number of cores better than other databases, it scales almost linearly. MongoDB scales well initially but then shows no improvement as number of cores increases. Such a tendency seems absent for Cassandra which is significantly slower than HBase and MongoDB and scales quite linearly. The mean update latency is a function of the number of cores. The mean response time required for write operations is more or less constant for both Cassandra and HBase, MongoDB shows an improvement in performance when the number of cores increases from one to three, then stabilizes. The read operation also shows its best when performed on MongoDB.

Replication factor is another criterion to be analyzed, higher replication factor more reliable the database system will be. Copies are stored on multiples nodes to avoid the delay and to increase the availability. Maintaining more than one replica does not decrease performance in case of Cassandra and MongoDB. There is a better performance as increase of replication creates more than one data node belonging to a shard, which basically distributes the nodes to more than one master. Neither in HBase behavior, there is no noted decrease in performance, or in throughput, or in response time due to replication.[9]

5. CONCLUSION

The SQL vs. NoSQL debate will continue, and we need not choose NoSQL as it is the latest hype. As the dust settle down, it will become evident that NoSQL solutions will work alongside the SQL solutions, each doing what they do the best. Now, mostly all companies have integrated NoSQL databases into their existing SQL infrastructure as each has its own strengths and weaknesses; neither can entirely displace the other. Here, an analysis of some NoSQL databases are done. The key responsibility of the database designers is to make a good decision in the choice of a database depending on the companies' requirement as choice of databases play an important role and the wrong choice at the beginning may have disastrous effects. The performance and scalability of the databases are the most important factors besides reliability. Comparing different databases can be difficult due to the difference in design, configuration and data access methods. So, to conclude, an application to calculate average income can go for a relational database, for building a shopping cart one can use a key-value Store, for storing structured product information there is document store database and for describing how a user move from point X to Y, a graph database can be used. No one can suggest a best database to work with, it basically depends on the application's requirement.

6. REFERENCES

[1] M. Sudheep Elayidom,"Dataminig and Warehousing", Cengage publishers,2015.

[2] Hiram Medero, Dayne Hammes, Harrison Mitchell, "Comparison of NoSQL and SQL Databases in the Cloud", Proceedings of the Southern Association for Information Systems Conference, Macon, GA, USA March 21st–22nd, 2014.

[3] A B M Moniruzzaman, Syed Akhter Hossain, "NoSQL Database: New Era of databases for Big Data Analytics-Classification, Characteristics and Comparison", International Jounal of Database theory and Application, June 2013.

[4] Greg Burd. 2011. NoSQL. (October 2011), Retrieved April 2, 2015 from https://www.usenix.org/legacy/publications/login/201110/openpdfs/Burd.pdf .

[5] Rick Cattell, "Scalable SQL and NoSQL Data Stores", SIGMOD record, vol.39, No.4, December 2010.

[6] A. Gandini, M. Gribaudo, W.J. Knottenbelt, R.Osman and P. Piazzolla, "Performance Analysis of NoSQL Databases", 11th European Performance Engineering Workshop (EPEW 2014). Sept 11-12, 2014.

[7] Veronika Abramova, Jorge Bernardino, Pedro Furtado, "Which NoSQL Database? A Performance Evaluation", Open Journal of Databases (OJDB) Volume 1, Issue 2, 2014.

[8] Christof Strauch," NoSQL databases: a step to database scalability in web environment", 13th International Conference on Information and Web-based Applications and Services, Dec 2011.

[9] Robin Hecht, Stefan Jablonski, "NoSQL Evaluation: A Use Case Oriented Survey", International Conference on Cloud and Service Computing,2011.