I.J. Information Technology and Computer Science, **2016, 12, 59-66**
Published Online December 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2016.12.07

SQL Versus NoSQL Movement with Big Data Analytics

Sitalakshmi Venkatraman
School of Engineering, Construction and Design (IT), Melbourne Polytechnic, VIC 3072, Australia
E-mail: SitaVenkat@melbournepolytechnic.edu.au

Kiran Fahd, Samuel Kaspi
School of Engineering, Construction and Design (IT), Melbourne Polytechnic, VIC 3072, Australia
E-mail: Kiran.Fahd@hotmail.com, SamKaspi@melbournepolytechnic.edu.au

Ramanathan Venkatraman
National University of Singapore, Singapore
E-mail: rvenkat@nus.edu.sg

***Abstract*—**Two main revolutions in data management have occurred recently, namely Big Data analytics and NoSQL databases. Even though they have evolved with different purposes, their independent developments complement each other and their convergence would benefit businesses tremendously in making real-time decisions using volumes of complex data sets that could be both structured and unstructured. While on one hand many software solutions have emerged in supporting Big Data analytics, on the other, many NoSQL database packages have arrived in the market. However, they lack an independent benchmarking and comparative evaluation. The aim of this paper is to provide an understanding of their contexts and an in-depth study to compare the features of four main NoSQL data models that have evolved. The performance comparison of traditional SQL with NoSQL databases for Big Data analytics shows that NoSQL database poses to be a better option for business situations that require simplicity, adaptability, high performance analytics and distributed scalability of large data. This paper concludes that the NoSQL movement should be leveraged for Big Data analytics and would coexist with relational (SQL) databases.

***Index Terms*—**Structured Query Language (SQL), Non SQL (NoSQL), Big Data, Big Data Analytics, Relational Database, SQL Database, NoSQL Database.

I. INTRODUCTION

As the technology environment transforms and faces new challenges, businesses increasingly realize the need to evaluate new approaches and databases to manage their data to support changing business requirements and growing complexity and expansion of their applications [1]. Relational database has been the default choice for data model adoption in businesses worldwide over the past thirty years with Structured Query Language (SQL) as the standard language designed to perform the basic data operations. However, with the explosion of data volume, SQL-based data querying lose efficiency, and in particular, managing larger databases has become a major challenge [2]. In addition, relational databases exhibit a variety of limitations in meeting the recent Big Data analytics requirement in businesses. While clusters-based architecture has emerged as a solution for large databases, SQL is not designed to suit clusters and this mismatch has led to think of alternate solutions. There are mismatches between persistent data model and in-memory data structures, and servers based on SQL standards are now prone to memory footprint, security risks and performance issues.

NoSQL (Non SQL) databases with a set of new data management features, on the other hand, are more flexible and horizontally scalable. They are considered as alternatives to overcome the limitations of the current SQL-dominated persistence landscape and hence they are also known as non-relational databases [3]. The main goal for the NoSQL movement is to allow easy storage and retrieval of data, regardless of its structure and content, which is possible due to the non-existence of a rigid data structure in non-relational databases. NoSQL databases exhibit horizontal scalability by taking advantage of new clusters and several low-cost servers. In addition, they are envisaged to automatically manage data administration including fault recovery and these capabilities would result in huge cost savings. Though non-relational databases are providing different features and advantages, they were initially characterised by lack of data consistency and non-ability to query stored records using SQL. With the emergence of NoSQL databases new features and optimisation characteristics are evolving to overcome these limitations as well. However, their total capabilities are still not disclosed [4]. Also, due to the increasing differences in NoSQL database offerings and their non-standard features, businesses are not clear on what is the stand to take.

In this paper, we first provide an overview of the present context of Big Data analytics and NoSQL databases. Next, we discuss the four main data models of non-relational databases and compare them with SQL databases. There are a variety of NoSQL databases and which one is more appropriate for which business operation remains an unanswered question so far. We compare the different data models of NoSQL in terms of their features and the NoSQL databases available in the market that support those features. The different data manipulation mechanisms and optimisation techniques adopted by NoSQL databases could result in their difference in performance. We discuss how these factors play a major role in Big Data analytics and identify the associated challenges. We also consider the coexistence of NoSQL databases with relational databases and discuss their relevance in different business contexts.

II. Related Work: the Context of NoSQL Databases with Big Data Analytics

From the recent trends reported in literature [5][6], it is evident that in today's context, there is an exponential growth of data volume that are structured as well as unstructured (Big Data) from a variety of data sources, such as social media, e-mails, text documents, GPS data, sensor data, surveillance data, etc. with increasing Internet usage. Hence, we can say that Big Data is characterised by structured, semi-structured, and unstructured data collected from digital and non-digital resources. The main challenge is the effective use of this Big Data that represents the data source for efficient decision-making by adopting suitable data mining techniques [7][8].

Based on our literature survey, we have identified that the current challenges presented by Big Data are due to the following general characteristics experienced by businesses:

- High data Velocity – rapidly and continuously updated data streams from different sources and locations.
- Data Variety – structured, semi-structured and unstructured data storage.
- Data Volume – huge number of datasets with sizes of several terabytes or petabytes.
- Data Complexity – data organized in several different locations or data centres.

It is important for businesses to perform Big Data analytics, which is the process of examining large data sets containing a variety of data types. Using Big Data Analytics, businesses are able to arrive at more accurate analysis of huge amounts of data to uncover hidden patterns, unknown correlations, market trends, customer preferences and other useful business information [2][9]. In order to support timely and effective decision making, Big Data analytics relies on large volumes of data that requires clusters for data storage. However, since relational databases are not designed for clusters, and exhibit performance issues with regard to Big Data analytics, businesses are considering the need for the NoSQL movement [10].

The schema of NoSQL is not fixed. It uses varied interfaces to store and analyse sheer volume of user-generated content, personal data and spatial data being generated by modern applications, cloud computing and smart devices. [1][11].

In this context, NoSQL database presents a preferred solution than SQL database primarily for its ability to cater to the horizontal partitioning of data, flexible data processing and improved performance. Large Internet companies (Facebook, LinkedIn, Amazon and Google), which cannot process services by using existing relational databases, had researched and led to the advent of NoSQL to solve their problem of dealing with continuously increasing data, optimised data utilization and horizontal scalability of large data. NoSQL databases are a better option for the information systems that require high performance and dynamic scalability more than the requirements of reliability, highly distributed nature of the three-tier Internet architecture systems and cloud computing [1][3][11]. Therefore, it is necessary to investigate further and compare SQL versus NoSQL as well as the salient differences in the performance of NoSQL data models in supporting the necessary features for Big Data analytics. This paper presents these investigations and findings in today's Big Data context.

III. NoSQL Data Models

There are many NoSQL databases available, however, they fall under four data models described below [3][11][12]. Each category has its own specific attributes but there are crossovers between the different data models. Generally, all NoSQL databases are built to be distributed and scaled horizontally.

Key-Value Store Database – Key-Value store is a simple but efficient and powerful NoSQL database. The data is stored in two parts, a string that represents the key and the actual data that represents the value, thus creating a “key-value” pair. This results in values being indexed by keys for retrieval, a concept similar to hash tables. In other words, the store allows the user to request the values according to the key specified. It can handle structured or unstructured data. It offers high concurrency and scalability as well as rapid lookups, but little consistency.

Such Key-Value store databases can be used to develop forums and online shopping carts and websites where user sessions are required to be stored. Some notable examples are Amazon’s DyanmoDB, Apache’s Cassandra, Azure Table Storage (ATS), Oracle Berkeley DB, and Basho Technologies’ Riak. Amazon offers fully managed NoSQL store service DynamoDB for the purpose of internet scale applications. It is a distributed key-value storage system which provides fast, reliable and cost-effective data access and high availability and durability due to its replica feature.

One of the advantages of Key-Value store database is its high insert/read rates compared to traditional SQL database. This is achieved by saving more than one entry to the store as shown in the example below:

```
@db.bulk_save([
{"hot" => "and spicy"},
{"cold" => "yet loving"},
{"other" => ["set","of","keys"]}
])
```

Column Oriented (or wide-column) Store Databases – In column store databases, columns are defined for each row instead of being predefined by the table structure having uniform sized columns for each row. Such stores have a two-level aggregate structure, a key and a row aggregate, which is a group of columns. Any column can be added to any row, and rows can have very different columns. In other words, each row has different number of columns that are stored. It can also store data tables as sections of columns of data. Data can be viewed as either row-oriented where each row is an aggregate, or column-oriented where each column family defines a record type. Each key is associated with one or more columns and a key for each column family is used for rapid data retrieval with less I/O activity thereby offering very high performance. These databases provide high scalability as they store data in highly distributed architectures.

Wide-column databases is ideal to be used for data mining and analytic applications with Big Data. Examples of some column-oriented store providers are Facebook's high-performance Cassandra, Apache Hbase, Google's Big Table and HyperTable. Google's Big Table is high performance wide-column database that can deal with vast amount of data. It is developed on Google File System GFS using C/C++. It is used by multiple Google applications like YouTube and Gmail that have varied latency demand of the database. It is not distributed outside Google besides the usage inside Google's App Engine. Big Table is designed for easy scalability across thousands of machines, thus, it is tolerant to hardware failures.

Document Store Databases – Document database extends the basic key-value database concept and stores complex data in document form such as XML, PDF or JSON documents. A document store is typically schema-less where each document can contain different fields of any length. Documents are accessed or identified by using a unique key which may be simple string, URI string or path string. Document databases are more complex databases but offer high performance, horizontal scalability and schema flexibility which allow storing virtually any structure required by any application.

Document oriented databases are suitable for content management systems and blog applications. Some examples of providers using document oriented databases are 10gen's MongoDB, Apache CouchDB, Basho Technologies' Riak, Azure's DocumentDB and AWS DynamoDB. MongoDB is developed by 10gen using C++ and is a structure free, cross-platform document oriented database. It uses Grid File System to store large files such as images and videos in BSON (Binary JSON) format. It provides efficient performance, high consistency and high persistence but it is not very reliable and is resource hungry.

Graph Store – Graph database focuses on relationships between data. It uses the graph theory approach to store the data and optimises the search by using index free adjacency technique. It is designed for data whose relationships are well represented by graph structures consisting of nodes, edges and properties. A node represents an object (an entity in the database), an edge describes the relationship between the objects and the property is the node on the other end of the relationship. In index free adjacency technique, each node consists of a pointer which directly points to the adjacent node as shown in Fig. 1.

These stores provide fast performance, ACID compliance and rollback support. These databases are suitable to develop social-networking applications, bioinformatics applications, content management systems and cloud management services. Examples of notable Graph databases are Neo Technology's Neo4j , Orient DB, Apache Giraph and Titan.

Apache Giraph is an open source large-scale graph processing system and implementation of Google Pregel (a graph processing architecture which has vertex-centric approach). It is designed for high scalability to overcome the crucial need for scalable platforms and parallel architectures that can process the bulk data produced by modern applications such as social networks and knowledge bases. For example, it is currently used at Facebook, LinkedIn and Twitter to analyse the graph formed by users and their connections. Giraph is a distributed and fault-tolerant system and offers features such as, master computation, sharded aggregators, edge-oriented input and out-of-core computation.

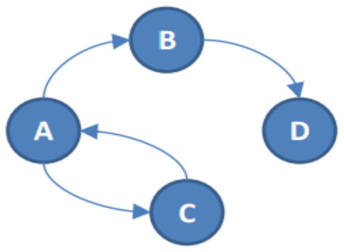

Fig.1. Graph algorithm.

IV. High Level Comparison Between NoSQL and SQL Databases

Based on the features of each type of database recently reported in the literature [1][3][11][13], we performed a high level comparison between SQL (relational) and NoSQL (non-relational) databases and the summary of findings is given in Table 1.

We considered aspects such as database type, schema, data model used, scaling model available, transactional capabilities, data manipulation method used, and popular database software available in the market in order to compare SQL databases versus NoSQL databases. Some examples are also given in Table 1 for a better understanding of their differences. Overall, Table 1 provides the high level differences in the key features and properties exhibited by relational and non-relational databases, which would support businesses in making decisions about using SQL or NoSQL database options in various Big Data application scenarios.

Table 1. Relational Versus NoSQL Databases - High Level Differences

	Relational Databases	***NoSQL Databases***
Data base Type	One SQL DBMS product (marginal variations)	Four general types: key value, document and wide column and graph stores
Schema	▪ Based on pre-defined foreign-key relationships between tables in an explicit database schema ▪ Strict definition of schema and data types is required before inserting data ▪ Any update alters the entire database.	▪ Dynamic db schema ▪ Do not force schema definition in advance ▪ Different data can be stored together as required ▪ Allows modification of the schema freely with no downtime.
Data Models	▪ Data records are stored as row and columns in different tables joined via relationships ▪ Explicit defined data types of columns to store a specific piece of data ▪ For example, SQL engine joins two separate tables the "employees" and "departments" together to find out the department of an employee.	▪ Supports all types of data – structured, semi-structured, and unstructured ▪ Different products offer different and flexible data models. For example:. Document store type organizes all related data using references and embedded documents tools.
Scaling Model	▪ Vertical Scaling ▪ Data resides on a single node and capacity is added to existing resources(data storage or I/O capacity)	▪ Horizontal Scaling ▪ Modern approach of partitioning of the data across additional servers or cloud instances as required.
Trans-action Capab-ilities	▪ Based on ACID transactional properties, such as atomicity, consistency, isolation, durability to ensure high data reliability and data integrity. ▪ Atomic transactions ▪ Degrade the performance	▪ Supports AID transactions and CAP Theorem of distributed systems supports consistency of data across all nodes of a NoSQL database ▪ there is atomicity at the single document.
Data Manipulation	▪ Structured Query Language – SQL DML Statements are used to manipulate data e.g. SELECT customer_name FROM customers WHERE customer_age>18;	▪ Query data efficiently. ▪ Object- Oriented APIs are used e.g. db.customers.find({customer_age: {$gt : 18 }} { customer_name:1 })
Software	Oracle, MySQL, DB2, SQLServer	▪ Mongodb, Riak, Couchbase, Rethinkdb, Redis, Aerospike, Leveldb, Hbase, Cassandra, Neo4j, Elasticsearch, Lucene

V. Performance Of NoSQL And SQL Databases For Big Data Analytics

The most important reason in moving towards NoSQL from relational database is due to requirements of performance improvements. Choi et al. [1] found that a NoSQL database such as MongoDB provided more stable and faster performance at the expense of data consistency. The tests were done on an internal blog system based on an open source project. It was found that MongoDB stored posts 850% faster than a SQL database. It has been suggested that NoSQL should be used in environments which are concerned with data availability rather than consistency.

Fotache & Cogean [14] describe the use of MongoDB in mobile applications. Certain multiple update operations like Upsert are easier and faster to perform with NoSQL than SQL database. The use of cloud computing along with NoSQL is said to increase the performance especially in the data layer for mobile platforms.

Ullah [15] compared performance of both relational database management system (RDBMS) and NoSQL database where Resource Description Framework (RDF) based Triple store was used as the NoSQL database. It was noted that NoSQL database was slower than the relation database due to the mass amount of memory usage by the NoSQL database. Reading a large amount of data takes toll on the database and because of the unstructured format of NoSQL database the storage of thousand records requires a huge amount of storage whereas the RDBMS uses less amount of storage. For example, searching red berry in the database took 5255 ms in the NoSQL database while it only took 165.43 ms to search it in RDBMS.

Floratou et al. [4] performed the Yahoo Cloud Serving Benchmark (YCSB) test on RDBMS and MongoDB. They tested SQL client sharded database against MongoDB auto and client sharded databases. The tests found that SQL client sharded database was able to attain higher throughput and lower latency in most of the benchmarks. The reason for higher performance is SQL is attributed to the fact that majority of the read requests are made to pages in the buffer pool whereas NoSQL databases tend to read shards located at different nodes. The study has tried to prove that RDBMS still has the processing power to handle larger workloads similar to NoSQL.

There are many advantages of NoSQL databases over SQL databases like easy scalability, flexible schema, lower cost and efficient and high performance. Having said that, there are some weaknesses of NoSQL over SQL databases to [12][16]. These are summarised below:

- NoSQL is new and immature; therefore, there is lack of familiarity and limited expertise.
- NoSQL databases scale horizontally by giving up either consistency or availability.
- There is no standard query and manipulation language in all NoSQL databases.
- There is no standard interface for NoSQL databases

- It is difficult to export all data in distributed ones (Cassandra) compared to non-distributed ones (MongoDB).
- NoSQL databases are challenging to install and difficult to maintain.

We have identified the following situations when NoSQL should be more suitable than SQL in the context of Big Data analytics:

1. Simplicity of use – current Big Data technologies are complex requiring highly skilled technical expertise, while NoSQL offers simplicity that would improve the productivity of both developers and users. The simple, small, intuitive and easy to learn NoSQL stacks can suit businesses that require Big Data analytics to adopt a clean NoSQL-like APIs.
2. Adaptability to change – when business requirements and data models change warranting flexible Big Data analytics, NoSQL that supports flexible data schemas are ideal to integrate siloed and disparate backend systems.
3. Efficiency for analytics functionality – The foundation data structure of majority of NoSQL technology is the Javascript Object Notation (JSON) data format that caters to both schema-on-read and schema-on-write efficiently for data warehousing functionality. For example, NoSQL Big Data Warehouse, SonarW for JSON makes analytics functionality efficient for Big Data applications.
4. Distributed scalability – with more and more distributed nature of systems and transactions, flexible data becomes the norm and strict schema approach is unsuitable. With schema evolution, NoSQL provides the necessary scalability for Big Data platforms to perform distributed queries faster.

Table 2. Comparison of NoSQL Data Models

NoSQL Data Models	**NoSQL Databases**	**Performance**	**Scalability**	**Flexibility**	**Complexity**	**Functionality**
Key-Value	DyanmoDB, Cassandra, ATS, Riak Berkeley DB,	High	High	High	None	Variable (None)
Wide-Column	Cassandra, Hbase, Big Table, HyperTable	High	High	Moderate	Low	Minimal
Document	MongoD, CouchDB, Riak, DynamoDB	High	Variable (High)	High	Low	Variable (Low)
Graph	Neo4j, Orient DB, Giraph, Titan.	Vari-able	Variable	High	High	Graph Theory

VI. Comparison of NoSQL Data Models

NoSQL databases vary in their performance depending on their data model [17]. We compare the key attributes of the four types of NoSQL data models and summarise them in Table 2.

As shown in Table 2, we have considered key attributes such as, performance, scalability, flexibility, complexity and functionality for comparing the four data models supported by the popular NoSQL database software that are available in the market.

Fig. 2 shows CAP theorem that forms a visual guide to NoSQL databases under each NoSQL data model [16], which is based on consistency, availability and partition tolerance features. With NoSQL databases, there are now other options for storing different kinds of data where typically distributed set of servers have to fit two of the three requirements of the CAP theorem, which is usually a deciding factor in what technology could be used.

Bazar & Losif [3] compared the performance of MongoDB, Cassandra and Couchbase databases, each possessing different features and functionalities. The tests were conducted using the YCSB tool.

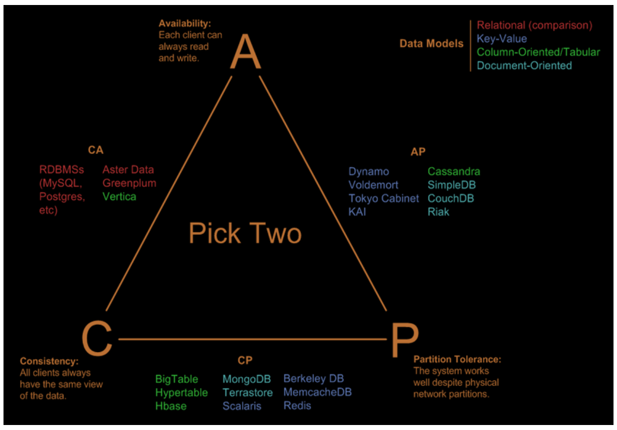

Fig.2. CAP theorem for NoSQL databases.

VII. Results

The benchmark tests found that Couchbase produced the lowest latencies for interactive database applications. Couchbase is able to process more operations per second with a lower average latency in reading and writing data than both MongoDb and Cassandra. Document level locking in Couchbase database is the primary reason for faster read and write operations. Cassandra is faster in writing than MongoDb but both of them have almost equal reading speed. It is also mentioned that each NoSQL database is suitable to specific application environments and cannot be considered a complete solution for every workload and use case.

Another case study by Klein et al. [18] looked at the use of NoSQL database MongoDB, Riak and Couchbase in a distributed healthcare organisation. These databases use different NoSQL data models including key-value (Riak), column (Cassandra) and document (MongoDB).

Cassandra produced the overall best performance for all types of database operations (Reading, Writing, and Updating). Riak's performance was degraded due to its internal thread pool creating a pool for each client session instead of creating a shared pool for all client sessions. Cassandra had the highest average latencies but also produced the best throughput results. This was firstly due to the indexing features that allowed Cassandra to retrieve the most recent written records efficiently, especially compared to Riak. Secondly, the hash-based sharding allowed Cassandra to distribute the request for storage to be load better than MongoDB.

Prasad & Gohil [11] discussed the use of different NoSQL databases for different work environments. It is reported that the performance of NoSQL databases is increased because of the use of a collection of processors in the distributed system. MongoDB and Cassandra are considered the best databases to be used in cases where data is frequently written but rarely read. The NoSQL databases are mentioned to be victims of Consistency, Availability and Partitioning (CAP) theorem. This means that a trade-off is always made e.g. the database can either be consistent with low performance or offers high availability and low consistency with fast performance [11][17][19].

Zhikun et al. [20] suggested the use of a new database allocation strategy based on load (DASB) in order to increase performance of the NoSQL database. However, the DASBL only works when it satisfies four conditions and is unable to cater to an unbalanced system load. Prasad et al. [11] compared different attributes such as Replication, Sharding, Consistency and Failure handling. We summarise all these findings in Table 3, which provides a list of the best NoSQL databases for each of the features reported in literature.

Several doubts arise on the NoSQL promises and studies have been conducted to explore the strengths and weaknesses of NoSQL [21][22]. A recent study reviews the trends of storage and computing tools with their relative capabilities, limitations and environment they are suitable to work with [23]. While high-end platforms like IBM Netezza AMPP could cater to Big Data, due to economic considerations, choices such as Hadoop have proliferated world-wide resulting in the rise of NoSQL database adoption that can integrate easily with Hadoop. Even though HBase supports strong integration with Hadoop using Apache Hive, it could provide a better choice for application development only but not for real-time queries and OLTP applications due to very high latency. On the other hand, graph-based platforms such as Neo4j and Giraph form better options for storage and computation due to their capability to model vertex-edge scenarios in businesses that involve data environments such as social networks and geospatial paths.

Overall, Big Data has led to the requirement of new generation data analytics tools [24][25] and hence it is realistic to believe that both SQL and NoSQL databases will coexist. With cloud environments that support SQL databases, fast processing of data is warranted to enable efficient elasticity [26] and Big Data analytics that involve current and past data as well as future predictions. New solutions are being proposed for cloud monitoring with the use of NoSQL databases back-end to achieve very quick response time.

Table 3. NoSQL Databases mapped to their features

Features	**Best NoSQL Databases**
High availability	Riak, Cassandra, Google Big Table, Couch DB
Partition Tolerance	MongoDB, Cassandra, Google Big table, CouchDB, Riak, Hbase
High Scalability	Google Big table
Consistency	MongoDB, Google Big Table, Redis, Hbase
Auto-Sharding	MongoDB
Write Frequently, Read Less	MongoDB, Redis, Cassandra
Fault Tolerant (No Single Point Of Failure)	Riak
Concurrency Control (MVCC)	Riak, Dynamo, CouchDB, Cassandra, Google Big Table
Concurrency Control (Locks)	MongoDB, Redis, Google Big Table

VIII. Conclusions

The industry has been dominated by relational or SQL databases for several years. However, with business situations recently having the need to store and process large datasets for business analytics, NoSQL database provides the answer to overcome such challenges. NoSQL offers schemaless data store and transactions that allow businesses to freely add fields to records without the structured requirement of defining the schema a priori which is a prime constraint in SQL databases. With the growing need to manage large data and unstructured business transactions via avenues such as social networks, NoSQL graphs are well suited for data that has complex relationship structures and at the same time simplicity is achieved through key-value stores. NoSQL data models provide options for storing unstructured data to be document-oriented, key-value pairs, column-oriented or graphs. These NoSQL storage models are easy to understand and implement and do not require complex

SQL optimization techniques to perform Big Data analytics. This paper has compared SQL versus NoSQL databases as well as the four data models of NoSQL in the context of Big Data analytics for business situations. We conclude that the flexible data modelling of NoSQL is well suited to support dynamic scalability and improved performance for Big Data analytics and could be leveraged as new categories of data architectures coexisting with traditional SQL databases.

REFERENCES

[1] Choi, Y., Jeon, W., & Yo, S. (2014), 'Improving Database System Performance by Applying NoSQL', *Journal Of Information Processing Systems, 10(3), 355-364.*

[2] Moniruzzaman, A. B., & Hossain, S. A. (2013), NoSQL database: New era of databases for big data analytics - Classification, characteristics and comparison. *International Journal of Database Theory and Application, 6(4), 1-14.*

[3] Bazar, C., & Losif, C. (2014), 'The Transition from RDBMS to NoSQL. A Comparative Analysis of Three Popular Non-Relational Solutions: Cassandra, MongoDB and Couchbase', *Database Systems Journal, 5(2), 49-59.*

[4] Floratou, A., Teletia, N., Dewitt, D., Patel, J., & Zhang, D. (2012), 'Can the Elephants Handle the NoSQL Onslaught?', *VLDB Endowment, 5(12), 1712-1723.*

[5] Mason, R. T. (2015), 'NoSQL databases and data modeling techniques for a document-oriented NoSQL database', *Proceedings of Informing Science & IT Education Conference (InSITE) 2015, 259-268.*

[6] Pothuganti, A. (2015) 'Big Data Analytics: Hadoop-Map Reduce & NoSQL Databases', *International Journal of Computer Science and Information Technologies, 6(1), 522-527.*

[7] Smolan, R. & Erwit, J. (2012), *The Human face of Big Data*, Against all odds production, O'Reilly, USA.

[8] Sharda, R., Delen, D., & Turban, E. (2015), *Business intelligence and analytics: systems for decision support* (10th ed.). Upper Saddle River, NJ: Pearson.

[9] Ohlhorst, F. (2013), *Big data analytics: Turning big data into big money*. Hoboken, NJ. John Wiley and Sons.

[10] Kaur, P.D., Kaur, A. & Kaur, S. (2015), 'Performance Analysis in Bigdata', *International Journal of Information Technology and Computer Science (IJITCS), 7(11), 55-61.*

[11] Prasad, A, & Gohil, B. (2014), 'A Comparative Study of NoSQL Databases', International Journal Of Advanced Research In Computer Science, 5(5), 170-176.

[12] Nayak, A., Poriya, A. & Poojary, D. (2013), 'Type of NOSQL Databases and its Comparison with Relational Databases', *International Journal of Applied Information Systems (IJAIS), 5(4) Foundation of Computer Science FCS, New York, USA.*

[13] MongoDB (2014), 'Why NoSQL?', https://www.mongodb.com/nosql-explained, [Online: accessed 20-Feb-2016]

[14] Fotache, M., & Cogean, D. (2013), 'NoSQL and SQL Databases for Mobile Applications. Case Study: MongoDB versus PostgreSQL', *Informatica Economica, 17(2), 41-58.*

[15] Ullah, Md A. (2015), 'A Digital Library for Plant Information with Performance Comparison between a Relational Database and a NoSQL Database (RDF Triple Store)', *Technical Library, Paper 205.*

[16] Hurst, N. (2010, *'Visual Guide to NoSQL Systems'*, http://blog.nahurst.com/visual-guide-to-nosql-systems, [Online: accessed 5-Nov-2015]

[17] Planet Cassandra *'NoSQL Databases Defined and Explained'*, http://www.planetcassandra.org/what-is-nosql,[Online: accessed 24-Mar-2016]

[18] Klein, J., Gorton, I., Ernst, N. & Donohoe, P. (2015), 'Performance Evaluation of NoSQL Databases: A Case Study', *Proceedings of the 1st Workshop on Performance Analysis of Big Data Systems, PABS'15, Austin, 5-10.*

[19] MongoDB (2015), *'Top 5 Considerations When Evaluating NoSQL Databases'*, https://s3.amazonaws.com/ info-mongodb-com/10gen_Top_5_NoSQL_Considerations.pdf [Online: accessed 5-Nov-2015]

[20] Zhikun, C., Shuqiang, Y., Shuan, T., Hui, Z., Li., Ge, Z.,& Huiyu, Z. (2014), 'The Data Allocation Strategy Based on Load in NoSQL Database', *Applied Mechanics and Materials, 513-517, 1464-1469.*

[21] Leavitt, N. (2010), 'Will NoSQL Databases Live Up to Their Promise?', *IEEE Computer 43(2) , 12-14.*

[22] Subramanian, S. (2012), *'NoSQL: An Analysis of the Strengths and Weaknesses'*, https://dzone.com/articles/nosql-analysis-strengths-and, [Online: accessed 15-Jan-2016]

[23] Prasad B.R. & Agarwal S. (2016), 'Comparative Study of Big Data Computing and Storage Tools: A Review', *International Journal of Database Theory and Application 9(1), 45-66.*

[24] Warden P. (2012), *Big Data Glossary - A Guide to the New Generation of Data Tools*, O'Reilly, USA.

[25] Zareian S., Fokaefs, M., Khazaei H. Litoiu M. & Zhang X. (2016), 'A Big Data Framework for Cloud Monitoring', *Proceedings of the 2nd International Workshop on BIG Data Software Engineering (BIGDSE'16), ACM Digital Library, 58-64.*

[26] Ramanathan, V. & Venkatraman, S. (2015), *Cloud Adoption in Enterprises: Security Issues and Strategies, 96-121*, Book Chapter In Haider A. and Pishdad A. (Eds.), Business Technologies in Contemporary Organizations: Adoption, Assimilation, and Institutionalization, IGI Global Publishers, USA.

Authors' Profiles

Dr. Sitalakshmi Venkatraman obtained doctoral degree in Computer Science, from National Institute of Industrial Engineering, India in 1993 and MEd from University of Sheffield, UK in 2001. Prior to this, she had completed MSc in Mathematics in 1985 and MTech in Computer Science in 1987, both from Indian Institute of Technology, Madras, India. This author is a Senior Member (SM) of IASCIT.

In the past 25 years, Sita's work experience involves both industry and academics - developing turnkey projects for IT industry and teaching a variety of IT courses for tertiary institutions, in India, Singapore, New Zealand, and more recently in Australia since 2007. She currently works as Lecturer (Information Technology) at the School of Engineering, Construction & Design, Melbourne Polytechnic, Australia. She also serves as Member of Register of Experts at Australia's Tertiary Education Quality and Standards Agency (TEQSA).

Sita has published eight book chapters and more than 100 research papers in internationally well-known refereed journals and conferences that include *Information Sciences*, *Journal of Artificial Intelligence in Engineering*, *International Journal of*

Business Information Systems, and *Information Management & Computer Security*. She serves as Program Committee Member of several international conferences and Senior Member of professional societies and editorial board of three international journals.

Kiran Fahd received the B.Eng in software engineering from the National University of Emerging Technologies, Pakistan in 2001, and the Master's degree in Enterprise Planning System - ERP from the Victoria University, Melbourne in 2010. Since 2001, she has worked in the capacity of software engineer and as a teacher.

Kiran has held various lecturing positions in Australian and overseas universities. She currently teaches the subjects of Bachelor of Information Technology under the Software Development major at the School of Engineering, Construction & Design, Melbourne Polytechnic, Australia.

Dr. Samuel Kaspi earned his PhD (Computer Science) from Victoria University, a Masters of Computer Science from Monash University and a Bachelor of Economics and Politics from Monash University. He is a member of Australian Computer Society (ACS) and Association for Computing Machinery (ACM).

Sam is currently the Information Technology Discipline Leader and Senior Lecturer of IT.at the School of Engineering, Construction & Design, Melbourne Polytechnic, Australia. Previously, Dr Kaspi taught at Victoria University, consulted privately and was the CIO of OzMiz Pty Ltd.

Sam has been active in both teaching and private enterprise in the areas of software specification, design and development. As chief information officer (CIO) of a small private company he managed the development and submission of five granted and three pending patents. He also managed the submission of a successful Federal Government Comet grant under the Commercialising Emerging Technologies category. He has also had a number of peer reviewed publications including the Institute of Electrical and Electronics Engineers (IEEE).

Dr. Ramanathan Venkatraman is working as Member, Advanced Technology Application Practice at National University of Singapore. He has served industry and academia for more than 32 years and has a wide spectrum of experience in the fields of IT and business process engineering. His current research focuses in evolving decision models for business problems and more recently, he has been contributing to frontiers of knowledge by devising innovative architectural models for ICT in domains such as Service Oriented Architecture, Big Data and Enterprise Cloud Computing. Dr Venkatraman has a strong practice approach having worked in large scale IT projects across Asia, US, Europe and NZ. He has published more than 20 research papers in leading journals. Apart from research and consulting, he also teaches advanced technical courses for Masters program at NUS and has been a key architect in setting up innovative software engineering and business analytics curriculum in the fast changing IT education scenario.

How to cite this paper: Sitalakshmi Venkatraman, Kiran Fahd, Samuel Kaspi, Ramanathan Venkatraman, "SQL Versus NoSQL Movement with Big Data Analytics", International Journal of Information Technology and Computer Science(IJITCS), Vol.8, No.12, pp.59-66, 2016. DOI: 10.5815/ijitcs.2016.12.07