KASU JOURNAL OF MATHEMATICAL SCIENCES (KJMS) VOL. 1, ISSUE 2, December 2020
ISSN 2734-3839 (Print), ISSN 2735-962X (E-Copy)
http:/www.journal.kasu.edu.ng/index.php/kjms

KASU JOURNAL OF MATHEMATICAL SCIENCES

A Comparison of NoSQL and Relational Database Management Systems (RDBMS)

[1]Musa Garba and [2]HassanAbubakar

[1]Computer Science Department, Kaduna State University
[2]Mathematics Department, UDU Sokoto,
[1]musa.garba@kasu.edu.ng, [2]hassan.abubakar@udusok.edu.ng

Abstract

Relational databases (RDBMS) have led the database market since its development by Edgar Codd in 1970 (Shuxin & Indrakshi, 2005). A more recent model that has appeared in the database market is known as NoSQL and it is quickly gaining ground. The new entrant is a non-relational data store which is being deployed massively in the scaled website settings where relational database features are less needed and better-quality performance in the area of data retrieval matters. NoSQL also known as non-relational databases compliments relational databases and are now used by the world's largest organizations such as Facebook, Amazon, and Google. Both models are good in specific areas and for specific applications. Depending on what issues are to be solved by the company, it will determine the choice of a database model to be adopted. Some organizations, however, prefer to make use of a hybrid database which is the combination of both NoSQL and relational databases otherwise referred to as multi-model database. The essence of this paper is to bring to the fore the importance of this relatively new technology and clearly show its architecture. NOSQL security features will be compared with the better-known relational databases for better understanding. This review will enable ease of choice for those who have the need for such databases or facilitate the choice to embrace the trending practice of combining the two in the same application to form a hybrid database.

Keywords: Databases, NOSQL, Relational Database, Integration, Scalability

1. Introduction

Big companies such as Facebook, Amazon, and Google discovered that relational database technology has serious limitations in the area of supporting huge amounts of data. They were able to as a result come up with various data management techniques which resulted in the creation of interest from companies with similar issues. This brings about the birth of NoSQL and its growing popularity today. (Zaki, A. K., 2014). NoSQL is a non-relational database considered as the future of databases and they provide important features such as huge storage data storage, dynamic schema, scale-out architecture, flexible data model and access requirements. NoSQL is nowadays used for its scalability and performance characteristics which were not a problem ten years ago. (Zaki, A. K., 2014).

2. Background Review

For more than thirty (30) years relational databases have been in use for business data processing as the best for storing personal data and financial records. (Mohamed, M. A. et al., 2014). Software developers decided that their data is not fitting for the relational database model when data started getting bigger and hence developed the NoSQL architecture for storing data. (Zaki. A. K. 2014). NoSQL databases have a simple to comprehend data model and are of high scalability. They have very easy query language, no instrument for data consistency and integrity and no need for database security support. (Okman, L.; Gal-Oz, N.; Gonen, Y.; Gudes, E.; Abramov, J., 2011). Perhaps the biggest advantage of the NoSQL database system is its ability to accommodate unstructured data which include video, audio, word doc, emails etc. It offers excellent scalability. The claim that NoSQL is of higher performance has also been shared by others which is vital for large establishments with a huge amount of data (Leavitt, N.2010). As data generated by social networks, real-time systems, and global users grow exponentially and storage for this huge data on distributed system seek for alternatives it becomes expedient that major establishments with huge data turn to the use of NoSQL for its storage. (Zaki. A. K. 2014). Atomicity, Consistency, Isolation and Durability (ACID) restraints that relational database uses are not adhered to in NoSQL databases. Providers of NoSQL listed this as advancement in terms of performance, but we know that this is also undesirable. A good example of NoSQL database performance is Cassandra which can manage over one hundred million users concurrently. (Okman, L.; Gal-Oz, N.; Gonen, Y.; Gudes, E.; Abramov, J., 2011). Kunda and Phiri (2017) also did an extensive review of some past literature on Relational Databases and NoSQL database model descriptions and features, some of the challenges of NoSQL were highlighted. Their features were compared by the authors to ascertain which one is better among the two in supporting modern database needs taking into consideration the challenges of NoSQL and whether NoSQL can replace the Relational Databases. Apart from the security challenge of NoSQL, lack of a standard query language was mentioned by the authors due to numerous ways of implementing NoSQL since each has its own language and interface which led to NoSQL having fewer users than the relational database. The Authors conclude NoSQL's popularity will be on the increase due to its Big Data, IoF and Social Network capabilities but many applications will still rely on a relational database hence both Databases must continue to co-exist to solve the shortcomings of one another. As it has been shown relational database features are best at handling a limited volume of structured data while NoSQL features are targeted at scalability and performance over a thin layer of security. Priyanka and AmitPal (2016) describe NoSQL in relation to the relational database as an alternative, not a replacement. NoSQL databases adapt BASE properties in place of ACID properties of relational databases which led to achieving better performance and scalability. They therefore conclude that although NoSQL adoption is at its peak right now with Facebook, Google, and Amazon due to its advantages, it remains true that the situation at hand determines the type of database to be used and hence a definitive decision on which one is better can't be reached. The combination of those database types in one application can also be considered for better performance, a process referred to as Polyglot persistence. Pore and Pawar (2015) also discuss that the limitation of SQL database

technology in the handling of big data as well as big user's requirement led Google, Amazon, Facebook, and LinkedIn to be among the first companies to adopt NoSQL in other to overcome the limitation. Dynamic schema, horizontal scalability and availability are some of the most important features of NoSQL databases that make its adoption attractive despite its numerous drawbacks such as "lack of RDBMS support for end-user querying, limited integrity constraints like foreign key at the structure level and limited support for transaction processing" (Pore and Pawar, 2015). After a detailed discussion on Axiomatics of SQL and NoSQL databases, the authors conclude that the two databases have different behavior, hence the choice of the type of databases to use depends on the application itself as well as the database size and the queries that the application will perform.

3. NoSQL Categories

Key value (KV) Store

This is a very powerful and efficient method for NoSQL database systems. Normally data can be stored as key-value pairs that are key array pairs for the purpose of retrieval using keys. Data are stored in hash tables as alphanumeric with unique keys and values as either JSON or BLOB or a string variable (Moniruzzaman, B. M, & Syed Akhter Hossain, 2013). Standalone tables should have two columns, with one column held as primary key (PK) while the other will serve as the holder of logical values. This procedure is known as Row store also called Tuple store since the data belonging to a single record are all stored together. Some examples of key-value Database Management System (DBMS) includeRiakredis, DynamoDB, Foundation DB and Dynomite.

Column-Oriented (Column Family or Wide Column) Stores

These are also called wide column stores, extensible records stores, column-oriented stores, or columnar database. (Manoj V.,2014). This was created for the purpose of data storing and data processing of a large amount of data which are distributed over different servers. Column family stores are rows containing many columns which store values closely with all columns arranges according to the column family. This is contrasting the relational databases that stores in structured tables, rows and columns with fixed field sizes for every record. This variant of NoSQL data not stored in structured tables but were housed in a large distributed architecture processed across several systems. (Leavitt, N.2010) &(Manoj V. 2014). This will encourage the addition of new columns in rows without inserting values in existing rows. It enables columns in Column family databases to be extended with each key connecting to one or more columns. (Ajayi O.,2014). Fast search and high performance on aggregation queries like Cassandra and Big Table is the main advantage behind storing data in columns.

Document-Based Stores: Document-oriented Database

These are one of the leading categories of NoSQL databases. It is used for Management of data retrieval, and store collection of textual documents e.g. XML, PDF, JSON etc. The schema for document and relational models is quite similar since collections represent the tables. The

documents also represent the row and key-value pairs to represent Columns [6]. The document-based database represents a collection of documents with semi-structured data inside which includes different key-value pairs whose value can be accessed through the PK. Also due to the schema, this type of NoSQL is flexible and can be changed easily which means that both keys and values are searchable in the document. Moreover, the addition of more number of attributes to a document is allowed by the users the data types may be different from the main document (Moniruzzaman, B. M, &Syed Akhter Hossain, 2013).

The document stores databases support complex data with several indexes per database. Fault tolerance and scalability are the important features of document databases. Though document stores may not be appropriate in a situation where a database got relationships or normalizations. (Nayak, A. et al., 2013). Some popular document stores are listed:

1. Couch DB
2. Terrastore
3. iBoxDB
4. AmisaDB
5. JasDB
6. EJDB
7. MongoDB

Graph Database

This is also known as Graph oriented databases and is a distinct type of NoSQL database. It stores data in graphical form (Ajayi O.,2014) and (Nayak, A. et al.,2013). It has three concepts, the first is called Nodes or Vertexes and they are equivalent to relational database tables. The second deals with relationships between the Nodes called Edges. The third concept is the Properties these are Key value pair called Columns. The keys are strings with value either primitive or array of the primitive type. These are attached to both the nodes and their relationship (Nayak, A., et al., 2013). It is apparent that the structure of graphs databases is constructed around a collection of Nodes and Edges. The method does not include storing network data in Nodes and Edges but in a network of Nodes and Edges. (Tyagi, C.2012). Index free adjacency which means that each Node should contain direct pointer with its close Node is an important quality technique and in this method, nodes do not contain dedicated indexes. This method or technique enables millions of records to be traversed by making a smooth connection with data. With graph databases, ACID properties are available with schema less architecture, rollback support with an effective storage in semi-structured data (Ajayi O. 2014) and (Tyagi, C.2012&Nayak, A. et al., 2013). In graph databases, it is possible for the exploration of a relationship among linked data by using pointers, unlike relational databases. Examples of graph database applications are Social media applications such as Twitter and Facebook. Others are access control, security and bioinformatics these serve as options for managing large relationships of data sets. (Nayak, A. et al., 2013). These graph databases are Graphbase, Trinity, AllegroGraph, Bigdata, Infinite graph and Neo4j.

ISSN 2734-3839 (Print), ISSN 2735-962X (E-Copy)
http:/www.journal.kasu.edu.ng/index.php/kjms

4. Comparing Relational Database & NoSQL Security

Below is a table showing comparisons between the very well-known Relational databases and Non-Relational Databases.

Table1. A Comparison of Relational Databases & NoSQL

Parameters	Relational Databases	Non-Relational Databases
1. Database model	This model database is founded on Relational approach also known as a Relational database (RDBMS).	This database model is founded on the Model less approach is also known as a non-relational or NoSQL database.
2. Data Representation	Stores structured data in tabular form i.e. (columns, rows) with relationships among joins or table.	Stores unstructured data approach with several kinds of stores such as Graph database, Key-value pairs, Document store or Column Family, .
3. Schema	All the data must fit into pre-defined table or structures.	The NoSQL schema is of high flexibility and dynamic.
4. Scaling	Relational databases are vertically scalable i.e. they increase the power of hardware e.g. CPU, RAM, Hard disk etc.	These are horizontally scalable i.e. they increase capacity by increasing machines or database servers.
5. Query Language	Structured Query Language (SQL) is the database language used for the manipulation and definition of data.	There is no dedicated query language for NoSQL. Sometimes Unstructured Query Language (UnQL) is used in NoSQL. The syntax is also different for the databases. Most NoSQL databases producers have created query languages for their own products.
6. Transactional operation	SQL is best with high transactions of delete, insert and update of data.	While selecting data NoSQL databases are better.
7. Transactional properties	Atomicity, Consistency, Isolation, and Durability (ACID) properties are employed by SQL databases.	Consistency, Availability and Partition tolerance (CAP theory) are employed by NoSQL databases.
8. Consistency	In relational databases is that all the users should view the same data after transacting. This enforces better consistency than non-relational databases.	Eventual Consistency guarantees read and write after transactions. All database entities will immediately be consistent.
9. Normalization/ De-normalization	Normalization is used in relational databases. A single table is divided into several smaller ones for performance improvement.	De-normalization is practiced with a non-relational database. A single table is used for record storage. Operations such as Select, Update, Delete and Insert will not be easy.

10. Data Integrity	Comparing in relational databases will remove all the duplicated records which stop inconsistent data from stocking the database.	Non-relational databases lack data replication. Flat databases update each address manually and ensure consistency.
11. Data Retrieval	SQL is used in relational databases which by its primary key table accesses the requested record.	The use of multiple criteria to access record is inefficient. This can be found in a non-relational database. Several passes are required before record for matches are inspected.
2. Data manipulation	Relational database uses Data Manipulation language (DDL) to manipulate data.	HTTP PUT, DELETE and POST are RESTful interfaces used by Non- relational database with several different formats such as JSON, Thrift and RDF. Also, in use are Data manipulation APIs e.g. Google data store.
13. Features	Maintenance as a key facility of RDBMS makes a repair, backup and tests easy by presenting tools to the database administrator.	Share nothing is a key feature of the NoSQL database. It works by horizontal scaling, partitioning and replication of data between several servers

Table 2. A Comparison of Relational Databases & NoSQL Security Services (Mohamed et al., 2014)

Parameters	**Relational Databases**	**Non-Relational Databases**
Authentication	Relational databases usually have an authentication mechanism which they use as it applies	Most NoSQL databases are defaulted not to possess authentication mechanism but uses external methods to achieve same.
Data Integrity	Data integration is achieved using ACID properties in relational databases.	Data integrity not easily achievable in NoSQL since. BASE properties are the only consistent principle.
Confidentiality	Confidentiality of data is achieved in relational databases using encryption techniques to store data.	Clear storage of data. No data confidentiality.
Auditing	There is provision for audit which allows data to be written to syslog or xml files.	NoSQL databases mostly do not provide an audit. Some provide but with issues since username and passwords are stored within a log file and this is a security compromise.
Client communication	Provision of secure client communication mechanism is done by encryption and SSL protocol.	This is not provided for in most NoSQL databases.

Integration of SQL and NoSQL as a Better Option

The option of integrating SQL and NoSQL has been part of the options discussed by different researchers." NoSQL and relational systems will likely co-exist for some time, and it is valuable to query them simultaneously" Lawrence (2014). According to him a lot of research was carried out towards integration of relational systems using view mediators and schema matching techniques. He further argued that same concepts should work on NoSQL systems and their "prior work on relational integration by Mason and Lawrence (2005) has been extended to support the querying, integration, and virtualization of both relational and NoSQL systems." To this effect, architecture was developed by the author called Unity Architecture. This allow SQL queries execution over both relational and NoSQL systems. The Unity System consist of virtualization and Integration system, the virtualization layer handles translation of SQL queries to NoSQL APIs thereby it executes operations that are not supported by NoSQL systems. Furthermore, the Unity System establishes seamless interaction between NoSQL Systems, relational databases and enterprise reporting applications with minimal overhead in SQL translation process. Similarly, in their paper Gašpar et al (2017) discussed two approaches to data integration between relational and NoSQL databases namely native and hybrid solutions. They use integration transactional data from Oracle databases (a relational database) and data stored in MongoDB (a NoSQL database) as an example. A native solution capitalizes on business layer and standard database drivers as well as how business layer communicates with a specific database. In any integrated system it is expected that data is stored in both, relational and NoSQL, hence for effective utilization of the data, the databases must be integrated. In a native solution, the integration is implemented on the business layer. Data retrieved from the databases (relational and NoSQL) are linked and converted into a format useful to the user at (business) layer likewise, business layer is responsible for the preparation of data to be stored in a specific (relational or NoSQL) database. In a hybrid solution, an additional layer between business and data layer is established to enable SQL communication. This enables developers to use common SQL patterns on the business layer but has to involve a new layer for translating these SQL patterns into the NoSQL programming interface for communication with the NoSQL database. They conclude that users determine the real value of data by how they utilize the data for better understanding of their business, customers and suppliers irrespective of where it is stored either in relational or NoSQL databases. Furthermore the world today is globalized, volatile and dynamic, users like to explore data from everywhere, been it in transactional systems, social networks, web sites etc. users will therefore not like any limitation of data analysis due to where data is stored. It is a well-known fact that efficient response to present and future business challenges rely on data analysis on a huge amount of data which leads to discovery of knowledge hidden in the data. " In that sense, the users see the database technology as a powerful tool that has the task to provide access and use of data wherever it is stored."(Gašpar et al, 2017). Hence this leads to further research on the integration of data stored in both relational and NoSQL database.

5. Conclusion

Relational databases are very important for solving ACID problems where data validity is needed for support in dynamic queries. NoSQL is important for solving data availability issues when faster access to data is needed and when there is a need for scaling based on requirement changes and also useful for fraud detection over relational databases. It is left for you to pick the right tool for the job. Relational database's demand from companies will not go away anytime soon and neither will today's distributed and product-based IT structure. Meanwhile, large applications that are satisfactory to the public will still be served by NoSQL databases.

Arguably, the core line of business applications that supports business shall be best obliged by relational databases. This may even be wholly served by relational databases. Companies will certainly fall within the mid-level spectrum of choosing both NoSQL and relational databases or one of the two depending on the task at hand. It is important that the best approach by extension the best fit application is selected. It will be wise to review the different facets of the several database technologies in existence before deciding on a particular product considering the data store it needs. The integration of relational and NoSQL databases is a major milestone that will handle any type of user, enabling the user to performing data analysis on data from any type of source been it relational or NoSql database.

6. Future Work

Future work from this paper can be conducted to improve integration by eliminating overhead costs as well as optimizing the SQL queries for better performance.

7. Recommendation

The integration of relational and NoSQL databases will lead to a database revolution; hence no institution would like to be left behind. Likewise, organizations and institutions in developing countries need to sit up to meet the challenges, as pointed out by Runde (2017) that current data revolution has a long way to go in developing countries. The following are recommendations that can help in adapting integrated systems:

1. Adapting or utilization of already existing databases with features capable of integrating SQL and NoSQL such as OrientDB by OrientDB ltd, thereby avoiding exhaustive time-consuming process of finding appropriate driver or tool.
2. Integrating systems that will make easy the process of data capturing in remote areas (such as USSD code system) with the applications running an integrated databasefor easy data capture and effective utilization of the data from all sources (transactional data, social media data etc.)

Reference

AmeyaNayak, Anil-Poriya & DikshayPoojary (2013). Types of NOSQL Databases and its Comparison with Relational Databases. *International Journal of Applied Information Systems (IJAIS) Foundation of Computer Science FCS, New York, USA Volume 5– No.4, March 2013.*

Brewer, E. (2000). Towards Robust Distributed Systems.[Online]. Available:http://www.cs.berkeley.edu/ brewer/cs262b-2004/PODCkeynote.Pdf.

Gašpar, D., Mabić, M., & Krtalić, T.(2017,Sept) Integrating Two Worlds: Relational and NoSQL.

Lawrence, R. (2014, March). Integration and Virtualization of Relational SQL and NoSQL Systemsincluding MySQL and MongoDB *In Computational Science and Computational Intelligence(CSCI), 2014 International Conference on (Vol. 1, pp. 285-290). IEEE*

Leavitt, N. (2010). Will NoSQL databases live up to their promise? Computer, 43, 2, 12–14.

Manoj, V. (2014). Comparative Study of NoSQL Document, Column Store Databases and Evaluation of Cassandra. *International Journal of Database Management Systems (IJDMS) Vol.6, No.4, August 2014.*

Mohamed, M. A., et al. (2014). Relational vs. NoSQL Databases: A Survey.

Moniruzzaman, B. M., Syed, A. H.,(2013). NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison, of Database Theory and Application [7].

Okman, L.,Gal-Oz, N., Gonen, Y., Gudes, E., &Abramov, J. (2011). Security issues in NoSQL databases trust, security and privacy in computing and communications (TrustCom), IEEE 10th International Conference, 541-547, 16-18.

Opeyemi Michael Ajayi, a Perspective of NoSQL: User Experience and Scalability of Cassandra and MongoDB.

Pore, S. S., & Pawar, S. B. (2015). Comparative Study of SQL & NoSQL Databases Structure, 4(5)

Priyanka, A. (2016). A review of NOSQL databases, types and comparison with relational database.International Journal of Engineering Science, 4963

Runde, D. (2017). The Data Revolution in Developing Countries Has a Long Way to Go. [Online] Forbes.com. Available at: https://www.forbes.com/sites/danielrunde/2017/02/25/the-data-revolution-in-developing-countries-has-a-long-way-to-go/#25a4176c1bfc [Accessed 20 Jul. 2018].

Shuxin, Y. and Indrakshi, R. (2005) Relational database operations modeling with UML, *Proceedings of the 19th International Conference on Advanced Information Networking and Applications, 927-932.*

T. Mason and R. Lawrence, "Dynamic Database Integration in a JDBC Driver," in ICEIS, 2005, pp. 326–333.

Tyagi, C., (2012). Comparative Analysis of Relational Databases and Graph Databases

Zaki. A., K. (2014). NoSQL Databases: New Millennium Database for Big Data, Big Users, Cloud Computing and Its Security Challenges.