Journal of Theoretical and Applied Information Technology
31st March 2014. Vol. 61 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

THE NEXT GENERATION DATABASE LANGUAGE WITH BASE PROPERTIES

[1]G.SEKAR, [2]DR. N.M. ELANGO

[1]Registered Scholar In Computer Science, Bharathiar University, Tamilnadu, India

[2]Director, Department Of Computer Applications, R.M.K Engineering College, Tamilnadu, India

E-mail: [1]gsekarg@yahoo.co.in, [2]nmeoxford@yahoo.com

ABSTRACT

"Not Only SQL" is popularly known as NOSQL, the new buzz word in the world of database technology. It really means that the database is not a relational database. Early, it is called as "No SQL-NoSQL", (i.e.) a system represents totally an alternative to SQL. Since it is inevitable to use SQL, later it is renamed as NOSQL. This paper brings out the properties, various flavours, examples, advantages and also some limitations of NOSQL database technology.

Keywords: *NOSQL, ACID, BASE, Key-Value store, Eventual Consistency*

1. INTRODUCTION

The NOSQL is a revolutionary event that brings new innovative technology and experts under single window. For any organization it is difficult to answer that which should be used NOSQL or traditional relational data? It is still too early to predict the success of NoSQL; but it is sure that it will produce a great impact in the stream of databases' storage and retrieval. The key merits are in the parameters of performance and scalability[1]. Unlike SQL, there is no fixed schema, but the schema has to be defined by the developer at the runtime. NOSQL uses an API to get the data and does not use mere SQL statements.

2. BIRTH OF NOSQL

In 1998, Carlo Strozzi used the term "NoSQL" to name his *lightweight, open-source relational database* that did not expose an SQL interface. Strozzi said that, the NoSQL "departs from the relational model altogether; it should therefore have been called more appropriately 'NoREL', or something to that effect".

A Rackspace employee, Eric Evans in 2009 reintroduced the term NoSQL. NoSQL actually emerges due to some transactions that don't exhibit ACID (Atomicity, Consistency, Isolation, and Durability) at all times. Instead of ACID, the NoSQL databases exhibit BASE (Basically Available, Soft State and Eventually Consistent) properties. The problem is if a database scales across multiple servers it is very hard to maintain ACID properties.

Actually NoSQL supports the concept of post-write replication which is similar to master-slave replication [3]. Write operation propagates on a primary replica and if commits only later they reflect in all other replicas.

BASE is just opposite to ACID (ie) ACID is pessimistic and focuses on concept of consistency at the end of every operation within the transaction. But BASE is optimistic and focuses on consistency only at the end of transaction. Therefore NoSQL means a database with "No ACID".

3. DEVIATION FROM ACID

BASE properties are mainly seeded by the CAP theorem. The properties of a distributed system can be described by the **CAP Theorem**[2] formulated by Eric Brewer (Figure 1). Of the three properties we can only have at most two:

- **C**onsistency
- **A**vailability
- tolerance to network **P**artitioning

Journal of Theoretical and Applied Information Technology
31st March 2014. Vol. 61 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

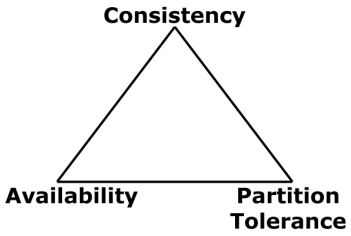

Figure 1: CAP Theorem

Amazon Dynamo uses Eventual Consistency to come close to get all three properties. Amazon Dynamo has the A and P properties. Google take a different approach with Big Table, which has the C and A properties.
Most of the NOSQL data stores (not all) will obey the properties like:

1. Permanence in data storage
2. Easy to implement in clusters
3. Scalability according to available memory
4. Allows schema migration
5. Owns individual query system and don't use SQL
6. Maintains ACID properties within a node and eventually consistent across the clusters

4. FLOURISH OF NOSQL

There are three main reasons which add to the reputation of NOSQL[4]. They are:

1. Certain user-generated content sites like twitter, face book often have more read-writes which will produce more traffic rather than static-content sites. So the first parameter is **traffic profile driver**.
2. The second parameter is that **the data changes over time**. The business model evolves concepts and the data models often try to evolve those concepts.
3. NoSQL technology is an Open Source Software and it can be downloaded easily anywhere in the world and so now it becomes **a commodity**.

One common question that is asked about NOSQL is whether people are really using it or whether it is just hype. The answer is that if we have entered Amazon, Yahoo or Google then we are served with data via a NOSQL solution. If we have used eBay or Twitter we have indirectly used data stores that bear little resemblance to traditional databases (i e) eBay does not use transactions and Twitter uses a custom graph database to track who follows whom. Are people really using it? The answer is that if we are facing issues dealing with certain types of data then there is potential competitive advantage to be gained by looking at a NOSQL solution.

It is obvious that there are a lot of advantages enjoyed by people when using SQL. It takes even years for NoSQL to overcomes or equalize the benefits of SQL.NOSQL products=SQL +Comprehensive standards [10]. The advantages of NOSQL are Matured and Refined. There are some standards already available for querying such as SparQL. The following is the list of various cases where RDBMS concepts are not recommended:

1. The relational databases will not scale to the traffic at an acceptable cost.
2. The number of tables required to maintain a normal form has being increasing.
3. The business model generates a lot of temporary data that are unable to store as a field in the table Eg:- Shopping carts, incomplete questionnaires.
4. The database model has already been denormalized for reasons of performance and convenience.
5. The dataset consists of large quantities of text or images and the column definition is simply a Large Object (CLOB or BLOB).
6. To run queries against the data that does not involve simple hierarchical relations.
7. There is a local data transaction that does not have to be very durable.

There needs a clear study of application to suggest what data can be stored as relational database.

- A particular example is the use of a graph database instead of very complex relational tables. It is possible to create sets of many to many relations in relational data and then query the intersections of these relationships.
- Web content can be maintained in terms of document and key-value data stores.
- Look-up with lots of reference data, maps, lists and sets.
-

Journal of Theoretical and Applied Information Technology
31st March 2014. Vol. 61 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5. NOSQL-MERITS

For a quarter of a century, the master of the database industry is RDBMS. Nowadays, NOSQL has been gaining the importance as an alternative model for RDBMS. The following are the advantages of using NOSQL:

1. **Excellent Scaling**: So far DBAs have believed on scale up and not on scale out. Scale Up means buying bigger servers as database load increases. Scale out (Figure 2) means distribution of database across various host when traffic and load increases. One of the key features of NoSQL is shared nothing horizontal scaling which means replication of data over multiple servers and it helps to support various read and writes transactions. This is called as **OLTP** (On Line Transaction Processing)

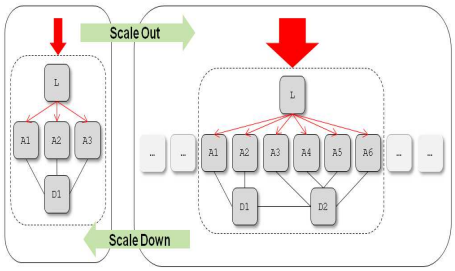

Figure 2: Varieties of Scaling

2. **Huge volume of data:** Nowadays the quantity of data becomes very large and it requires heavy storage. O'Reilly has called this as the "industrial revolution of data". But as the transaction increases, the parameters to maintain the database are not tolerable. But NOSQL systems can manage even largest biggest RDBMS very easily.
3. **No requirement of DBA's:** We all know that a RDBMS needs trained, experienced and intelligent Data Base Administrators for its design, installation, and ongoing tuning. But the NoSQL databases don't need such DBAs since they are tuned from the scratch itself to support automatic repair, data distribution
4. **Cost:** NOSQL databases use cheap commodity servers to maintain the data and transactions, while RDBMS uses expensive servers and storage systems. So the cost to store data becomes increases when using RDBMS than NoSQL.
5. **Data Models:** Change management is a big task for large RDBMS.RDBMS have to be carefully managed even to update very small changes. NoSQL databases are more relaxed in the view of data models [5]. But there is a virtual structure for every data element to store in a NoSQL key-value stores. Even the more rigidly defined Big Table-based NOSQL databases typically allow the creation of new columns with less effort.

6. NOSQL-LIMITATIONS

On the other side of the coin, there are also many challenges that stood behind the NOSQL as listed below:

1. **Maturity:** It is obvious that the age and experience of RDBMS is high than NoSQL (ie) the maturity of the RDBMS is reassuring. RDBMS systems are stable and richly functional. Still, some NOSQL alternatives are under live versions with many of the key features are yet to be implemented.
2. **Support**: There is a high-level of support for every RDBMS from enterprise level, in which point NoSQL is lack. But most NOSQL systems are open source projects with one or more firms offering a little support for each NOSQL database.
3. **Analytics and Business Intelligence**: Surely every NoSQL has to meet the issues while scaling the database. As a result, most of their features are concerned with demands of application to be implemented. But the data stored in the application has value apart from usual the insert-read-update-delete cycle of a typical web application. But NoSQL provides few facilities for ad-hoc query and analysis. A very simple query needs significant programming expertise, and commonly used BI (Business Intelligence) tools do not provide connectivity to NOSQL.
4. **Administration:** NOSQL requires a lot of skills to install and a lot of effort to maintain.
5. **Expertise:** While millions of developers are familiar with RDBMS concepts and programming while a people of NoSQL is only in the learning phase (ie) it is now far easier to find experienced RDBMS experts than a NOSQL expert.

Journal of Theoretical and Applied Information Technology
31st March 2014. Vol. 61 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

7. COMPARISON BETWEEN RELATIONAL AND NON-RELATIONAL DATABASES

The column-oriented database stores data as a group of columns instead of group of rows. A file is created for each column and grouping of identical values take place within that column. So there is a possibility of storing several records in a single page of a disk. Comparing the features of row store and column store, it is found that the column store suited well for massive database with read transactions.

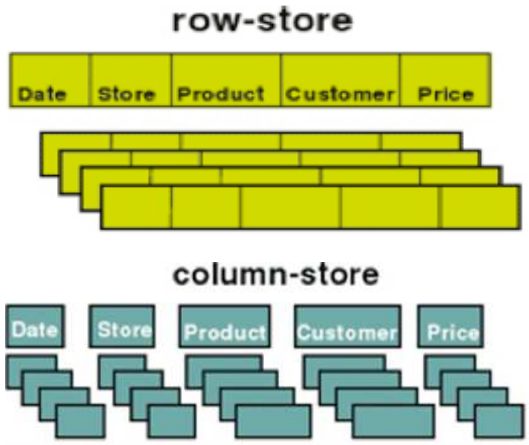

Figure 3: Row VS Column DataStore

But it is not so flexible for the database which is often prone to write operations. It is also not recommended where there are more columns are to be processed as more steps are required here because the column store considers one column at a time [7].

Row Store	Column Store
Suitable for adding or deleting records.	Has an advantage in reading related data.
Reads unnecessary (unused) column values causing disk I/O.	Several entries need to be modified to change a record causing disk I/O.
Relatively low record-level compression rate.	Compression rate is good as the columns of a domain can share the same value.
Has an advantage in performing random access because indexes are configured at record-level	Has an advantage in performing range queries.

Table 1: Properties of Row Store VS Column Store

8. NOSQL DATABASES-CLASSIFICATION

The table reveals the various types of NoSQL databases, their applications and basic differences among them.

NAME	USAGES	MERITS	DEMERITS
Key value stores (Tyrant, Redis)	Content caching	Fast lookups	Data has no schema
Document Databases (CouchDB, MongoDB)	Web applications	Tolerant of incomplete data	Query performance
Graph Databases (Infinite Graph, Neo4J, InfoGrid)	Social networking	Graph algorithms	Not easy to cluster, traverse the entire graph to get the correct answer
XML Databases (Exist, MarkLogic)	Publishing	Mature search algorithms	Hard to update but easier to rewrite
Distributed Peer Stores (Cassandra, HBase)	Distributed File Systems	Fast lookup, good distributed storage of data	Low level API
Object stores (GemStone, Polar)	Finance systems	Mature technology, low latency ACID, matches OO development paradigm	Limited querying options

Table 2: Examples of NOSQL databases

NoSQL solutions fall into two major areas:

- Key/Value or 'the big hash table' [8].
  - Amazon S3 (Dynamo)
  - Voldemort
  - Scalaris
- Schema-less which comes in multiple flavors, column-based, document-based or graph-based.
  - Cassandra (column-based)
  - CouchDB (document-based)
  - Neo4J (graph-based)
  - HBase (column-based)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

If we go for NoSQL solutions, we have to give up:

- joins
- group by
- order by
- ACID transactions
- SQL as a sometimes frustrating but still powerful query language
- easy integration with other applications that support SQL

9. SUMMARY

We can't a world without SQL. But experts start to think about data in other dimension, which is NoSQL. NoSQL databases are becoming a vital part of the database stream, and if used correctly, the people can really enjoy real benefits. However, necessary precautionary measures have to taken before transforming the database from traditional database to new NoSQL.

Therefore the NoSQL systems described here generally do not provide ACID transactional properties: updates are eventually consistent to improve performance and scalability, but there are limited guarantees on the consistency of reads.

NoSQL systems generally have six key features:

- the ability to horizontally scale
- the ability to replicate data
- a simple call level interface
- a weaker consistency model than the ACID transactions
- the ability to dynamically add new attributes to data records
- efficient use of indexes
-

REFERENCES

[1] Vatika Sharma, Meenu Dave, SQL and NoSQL Databases - IJARCSSE, ,Volume 2, Issue 8, August 2012.
[2] Brewer, Eric A. Towards Robust Distributed Systems. Portland, Oregon, July 2000.---CAP theorem.
[3] Amir H. Payberah, — NoSQL Databases April 26, 2012.
[4] WHITE PAPER on —Why NoSQL? — By DataStax Corporation September 2012.
[5] Abhinay B. Angadi Akshata B. Angadi Karuna C. Gull, Growth of New Databases & Analysis of NOSQL Datastores, IJARCSSE,Volume 3, Issue 6, June 2013.
[6] Chang, Fay; Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: A Distributed Storage System for Structured Data. Google. 13 November 2009.
[7] Kellerman, Jim. "HBase: structured storage of sparse data for Hadoop", 13 November 2009.
[8] Aguilera, m. K., merchant, a., shah, m., veitch, a. karamanolis, c. Sinfonia: A new paradigm for building Scalable Distributed Systems, SOSP '07 acm, pp. 159–174.
[9] Zaharia, M., chowdhury, M., franklin, M., Shenker, Stoica, I. Spark: Cluster computing with working sets. In 2nd USENIX workshop on Hot Topics in Cloud Computing (2010).
[10] www.NoSQL-database.org
[11] www.couchbase.com
[12] www.NoSQLdatabases.com