NoSQL Databases in support of Big Data and Analytics

Uma M Kugan
Indiana University
711 N Park Avue
Bloomington, IN 47408, USA
umakugan@iu.edu

ABSTRACT

This paper will help us identify how NoSQL is efficient and cost effective in handling big data and also will highlight on why Big Data can't be handled in traditional RDBMS.

KEYWORDS

i523, hid323, NoSQL

1 INTRODUCTION

RDBMS have always been the preferred method of storage for many years and its powerful Query language made it very user friendly.Data has grown exponentially in a past decade due to the growth of social media, e-commerce and web applications which posed a big challenge for the traditional databases.Need of the hour is not just to limit the data within the structure, but also ability and flexibility to read and store data from all sources and types, with or without structure.Organizations that collect large amounts of unstructured data are increasingly turning to non relational databases, now frequently called NoSQL databases.[**?**] There are lot of limiting factors in these databases for Big Data especially Structured schema which was one of the main reason for RDBMS to scale it for larger databases[6].

2 WHY NOSQL

The term NoSQL was first used by Carlo Strozzi to name a database management system (DBMS) he developed.This system explicitly avoided SQL as querying language, while it was still based on a relational model[3]. The term NoSQL means that the database doesn't follow the relational model espoused by E.F Codd in his 1970 paper A Relational Model of Data for Large Shared Data Banks[5] which would become the basis for all modern RDBMS. NoSQL doesn't mean NO to SQL. It means Not Only SQL. NoSQL means storage is just nonvolatile object store with no maintenance concerns. Most NOSQL DB's are open source which allows everyone to evaluate the tool of their choice at low cost.

3 NOSQL TYPES

In [1] Edlich et al. identify four classes of NoSQL systems as 'Core-NoSQL' systems: Key-Value stores, Wide column stores, Graph databases and Document stores.

3.1 Key-Value Stores:

It is a very basic type of non-relational database where every item (value) is stored as an attribute name (key), with its value. e.g. Redis

3.2 Wide Column Stores:

Every record in the stores may differ in the number of columns. This is very important factor for analytics because it needs very low I/O and also reduces the volume of data that are read to the disk. e.g. HBase and Cassandra

3.3 Graph Database:

As the name indicates, it uses graph structures nodes and edges to represent the data. This is very useful in depicting social relationship, network topology. e.g. Neo4J

3.4 Document Stores:

It stores the data as document typically in Jason or XML format.It is widely used due to its flexibility and ability to query the data. e.g. MongoDB and CouchDB.

4 NOSQL FOR BIGDATA

Following factors have to be considered while evaluating NoSQL for Big Data Projects:

Solution Based on the project Requirements :

Real time Updates for Data Analytics - NoSQL is the solution for applications that receives large volume of data in a real time and where data insights are generated using real time data that was fed.

Publish/Subscribe - NoSQL is the best fit where the enterprise doesn't require complex messaging features for publishing/subscribing.

Document based - Application where data structure is not restricted by schema, NoSQL comes in hand in such places.

Limitation of traditional Databases :

Scalability - RDBMS are designed for scaling up meaning if storage needs to be increased, we need to upgrade other resources in the existing machine whereas in NoSQL we just have add additional nodes in the existing cluster.

Acid compliance - RDBMS are always acid compliant i.e. Atomicity, Consistency, Integrity and Durability and which of course is its strength to process transactional data while the drawback is it can't handle larger volume of data without impacting the performance. If there are use cases where we don't require ACID compliance and where it has to handle huge volume of data in significantly very less time, then NoSQL is the solution.

Complexity - RDBMS stores the data in defined, structured schema in tables and columns. If the data can't be converted to store in tables, it becomes cumbersome to handle such situations.

5 HOW TO HANDLE RELATIONAL DATA IN NOSQL

NoSQL database in general can't perform joins between data structures and hence the schema has to be designed in such a way so that it can support joins. Below are the key things that needs to be considered to handle relational data in a NoSQL.

Avoid Sub Queries : Instead of using complex sub queries or nested joins to retrieve the data, break into multiple queries. NoSQL performances are very high when compared to traditional RDBMS Queries.

Denormalize the data : For faster retrieval of data, it is essential to compromise on denormalizing the data rather than storing only foreign keys.

6 RDBMS TO NOSQL MIGRATION

Database Migrations are always cumbersome and it is better to plan well ahead and take an iterative approach.Based on the need of application, one have to choose which NoSQL DB's we are going to migrate to. [4]

6.1 Planning

The goal of any migration should be better performance at the reduced cost with the newest technology.While migrating from RDBMS, we have to consider volume and source of data that's going to be migrated to NoSQL. All the details should be documented well so that we don't have to face unplanned surprises at the end.

6.2 Data Analysis

This is very critical and will help in understanding the nature of the data and how that data is accessed within the application. Based on the analysis of data usage, we will be able to define how data will be read/written which will help us in building a better data model.

6.3 Data Modeling

When migrating from any RDBMS, depending on the need of application, we may have to sometimes denormalize the data. In this phase, based on the data analysis and the tech-stream, we have to define keys and values.

6.4 Testing

Testing is always very critical and crucial for any migration projects. All aspects of testing from unit testing, functional testing, load testing, integration testing, user acceptance testing etc., have to be carried out and outputs have to be clearly documented.

6.5 Data Migration

Once all the above steps are successfully tested and implemented, next final act is to migrate all data from RDBMS to NoSQL. Post implementation validation has to be carried out to make sure everything went well as per the plan and it has to be monitored for few days until the process is stabilized. If there are any issues with the migration, rollback to original state and root cause analysis have to be performed to identify and fix the issue. Once issue has been fixed, data migration has to be scheduled and this step goes in cyclic unless migration was completely successful.

7 ADVANTAGES AND DISADVANTAGES OF NOSQL

NoSQL databases differ from traditional databases in features and functionality. There is no common query language, high I/O performance, horizontal scalability and don't enforce schema. It is very flexible and let the users to decide to use the data the way they want.

NoSQL databases have the ability to distribute the database across multiple geographic regions to withstand regional failures and enable data localization. Unlike relational databases, NoSQL databases generally have no requirement for separate applications or expensive add-ons to implement replication.[2]

Since NOSQL doesn't enforce atomicity and hence it is not reliable where data accuracy is very critical.RDBMS are much more matured and the best technical support is available.So there is always fear of unknown until the technology gets widely accepted and used.

8 CONCLUSION

With the explosion of the data in the recent years, have paved the big way for the growth of Big Data and everyone wants to move their applications and data into Big Data. Building a big data environment is relatively very cheap when compared to migrating the existing data in RDBMS to NoSQL. We have to carefully weigh in, understand the data and how the data will be used in the use case to enjoy the full benefit of migrating into No SQL.

ACKNOWLEDGMENTS

My sincere thanks to my mentor and leader Vishal Baijal and to my colleague Michael Macal for their support and suggestions to write this paper and also to my fellow classmate Andres Castro Benavides for his support.

REFERENCES

[1] S. Edlich, A. Friedland, J. Hampe, and B.Brauer. Oct 2012. NoSQL: Einstieg in die Welt nichtrelationaler Web 2.0 Datenbanken. Hanser Fachbuchverlag. 6 (Oct 2012).

[2] MongoDB. 2016. *Top 5 Considerations When Evaluating NoSQL Databases.* Technical Report. MongoDB. https://www.mongodb.com/nosql-explained

[3] Editor P. BAXENDALE. June 1970. (June 1970). http:/www.seas.upenn.edu/~zives/03f/cis550/codd.pdf

[4] Nathaniel Slater. March 2015. Best Practices for Migrating from RDBMS to Amazon DynamoDB- Leverage the Power of NoSQL for Suitable Workloads. (March 2015). https://d0.awsstatic.com/whitepapers/migration-best-practices-rdbms-to-dynamodb.pdf

[5] C. Strozzi. July 2012. Nosql relational database management system. (July 2012). http:/www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/NoSQL/HomePage

[6] Aspire System. 2014. *BigData with NoSQL.* Technical Report. Aspire System. http:/www.aspiresys.com/WhitePapers/BigData_with_NoSQL_Whitepaper.pdf?pdf=nosql-whitepaper