Using NoSQL Database to Persist Complex Data Objects

Maxmiliano Franco BRAGA, Fábio Nogueira de LUCENA
Instituto de Informática
Universidade Federal de Goiás (UFG)
Goiânia – GO – Brasil
{maxmilianobraga, fabio}@inf.ufg.br

***Abstract*—NoSQL Databases are database management systems which use few or no SQL commands to query, store and delete data. They are used for situations on which traditional relational data base managements were not designed for, such as horizontal scaling and storing large amount of complex objects, which are difficult to store on tables.**

The use of this kind of database by some organizations, such as Google and Amazon, which rely on the storage of great amount of data, is a strong evidence that NoSQL has some advantages to be used for large amount of data.

This paper presents the NoSQL database approach and why this kind of database may be a good option applications which deal to large transactions to persist complex data objects.

***Index Terms*—Software Engineering, NoSQL, Database Management Systems, persistence,**

I. Introduction

Organizations collect a large amount of data for future analysis. These data are traditionally stored on databases. Database is a system used to store and retrieve organized data.

A. Relational Database

Relational Database Management Systems, or just Relational Databases, have been the most used storage model for the last decades.

Typical Relational Databases Management Systems work very well on small but frequent read ant write transactions. They were designed for this kind of transactions and handle them very well. For most of the cases, RDBMS are sufficient for applictaions needs. But they present poor performance on applications which store large amount of complex data. Such problems are evidenced on some situations as indexing large number of documents and information, retrieving self-associated complex objects, and streaming stored media files. The relational approach does not suits very well on these situations. [1]

B. NoSQL Database

Although the Relational Model has served well for a long period, there is a growing number of organizations using non relational databases, also known as NoSQL databases. NoSQL is the abbreviation of ”Not only SQL”. [2]

The term NoSQL was first used in 1998 to name an open-source relational database system that did not have a SQL interface explicitly. It was an experimental database system and have not been widely used. The term have no longer been used until 2009, when it emerged on an event about non-relational approaches databases. NoSQL was then used to describe a group of databases which had a different approach of SQL Databases Systems. [3]

Although the term NoSQL and some recently discussions suggest that these databases are alternatives to take place of the common Relational Databases in all situations, they are indeed options for some specific situations for which the relational databases were not designed for. Stonebraker disagree about the adoption of the term NoSQL. He explains that ”not only relational” suits better. Relational databases and NoSQL databases are not competing for the same space. They should coexist, each approach for its suitable situation. In fact, these two approaches have different ranges of problems which they were designed to solve. [4]

Some of the NoSQL databases are: mongoDB, chouchDB, Neo4J, BigTable, Cassandra and SimpleDB.

C. Scalability

Scalability is the capacity of a system to share work load on different resources. If this share is linear and equivalent proportional to each node capacity, the scaling

process is efficient. [5] Vertical scaling, or scaling up, means to add more resources to the same server or node, such as memmory or processor power. Horizontal scaling, or scaling out, means to add more servers to the processing structure.

II. NoSQL Concepts

NoSQL databases were designed to solve some common problems for distributed databases. Scaling and MapReduce are some techlogies most NoSQL Databases use to solve these problems. [6]

A. Scaling

There two technologies NoSQL databases use to scale data: Replication and Sharding.

Replication is, as its names suggest, storing replicate data on different nodes. The trick to acquire high performance is to balance the acces to this replicate data, so the processor parallalizes the work load processing.

Orend explains that replication also makes the database failure tolerant, as if one node fails, there is at least another node which has the same data. Although this advantages, it costs more resources on writing operations.

Sharding is the ability of splitting data to different nodes. Each node is called shard. Some NoSQL can be set up to shard data according to some rules applied to the data primary key.

The result of this is a segregated collection of objects spread on different nodes. The database system must have a service to route the operations through each node, so data can be stored, retrieved and deleted on the right shard.

B. MapReduce

MapReduce is a model created to process large amount of data in a systematic way. It is based on two functions: map and reduce. The main objective is to make possible for the system to use this functions to parallelize computation across large-scale. [7]

Dean explains how Google uses MapReduce to iterate and seek for patterns through large amount of raw data and crawled documents.

The use of MapReduce framework makes it simple for applications to automatically be distributed through different servers. Most NoSQL databases, such as mongoDB, use MapReduce to distribute.

III. NoSQL Models

NoSQL databases share some common characteristics, such as the lack of a rigid schemma, no necessity for joins to combine relations, and and easy way to scale horizontally. But they serve for different needs and use different approaches to store data. Three of these approaches are Document Oriented Databases, Key/Value Store and Graph Database.

A. Document Oriented Databases

Document Oriented approach structures data similar to document, as its name indicates. The usually does not use tables. The attributes of the items are dinamically inserted. Such as documments share some common properties, but may have different ones, the items stored on Documento Oriented databases may differ. As no table is declared, there is no rigid struture to store data on it. The application is responsible to treat these dynamic properties.

The mongoDB, which is object of this study, and coachDB use the Document Oriented approach.

B. Key/Value Store

The Key/Value Store is the simplest NoSQL database approach. The data is stored on a hash which has a unique identifier, the Key, and the respective data, the Value. The data is structured similar as an dictionary. Insert, delete and update operations are applied on each given key.

Comparing to a relational model, the Key/Value approach is similar to a table of two columns. The difference is that the value column may store multivalue items.

SimpleDB is provided by Amazon and uses this approach. It is a schema free database. It provides a simple SQL Interface for end-user to query over data. Project Voldermort is anotaher database management systems which adopt the Key/Value approach.

C. Graph Database

Graph is a mathematical abstraction common used on Computer Science. The interest on graphs has grown recently. One of the applications is to represent social networks. [8]

Graph Database are the the most complexes NoSQL databases approach. The data is represented as complex objects which associates to other data similar to graphs structures.

Neo4J is an open source Graph Database implemented in Java.

IV. MONGODB

mongoDB is a NoSQL database management system. We have chosen mongoDB because it is a free open source system and it implements most of the NoSQL technical features presented on this paper.

David Mytton tested most of the NoSQL database systems for a project and explains why they have chosen mongoDB on his article "Choosing a non-relational database; why we migrated from MySQL to MongoDB".

Some of the reasons made Mytton [9] take his choice for mongoDB include its easy way to install, easy replication capability, automated sharding and good documentation.

Mytton explains that performance was not one of the reasons they have chosen mongoDB. Although they discovered mongoDB to be a lot faster than MySQL on huge operations.

A. mongoDB Replication

mongoDB uses replication to outcome failover situations and to acquire data redundancy. Replication on mongoDB are available on two forms: Master/Slave Replication and ReplicaSet.

The mongoDB Documentation [10] recommends Replica Set as the prefered repclication method.

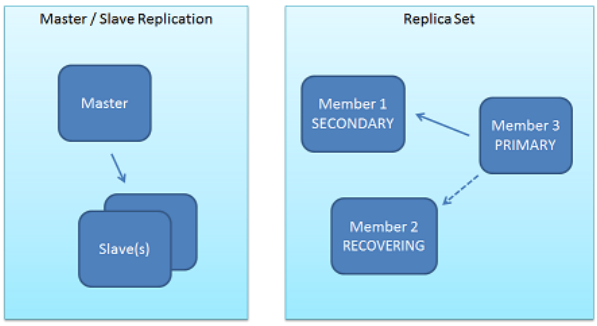

Fig. 1. Two forms of replication on mongoDB.

B. mongoDB Auto-Sharding

[10] Auto-sharding in mongoDB makes it easy to automatic balance data distribution over each shard, addition of more machines and automatic failover.

Sharding means partitioning collections of data and splitting them through machines, called shards. mongoDB offers automated share to balance and handle failover with no user interference. Sharding uses replica set to prevent failover.

mongoDB Documentation suggests a initial sharding architecture as a start point for building a cluster. It

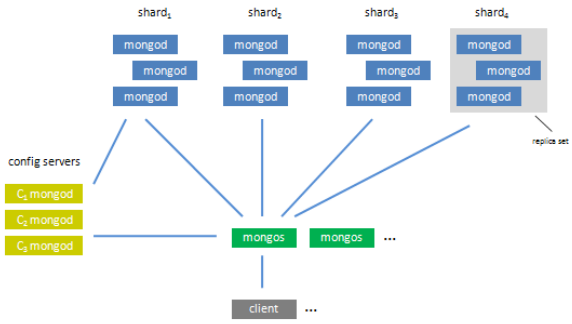

Fig. 2. mongoDB Sharding Architectural Overview

shows how to set up an architecture made of 3 shards for scalability, and 3 copies of each piece of data.

Implementing this initial sharding architecture demonstrates hor easy is to set up a mongoDB database that makes use of some of the most important NoSQL features discussed on this paper.

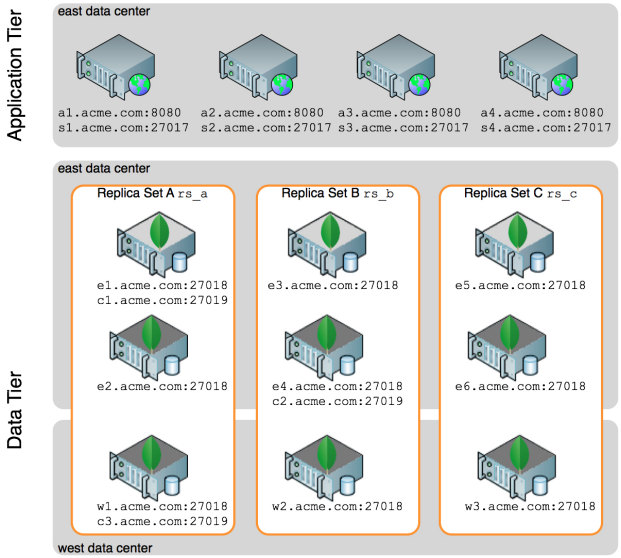

Fig. 3. Sharding start point, suggested by mongoDB documentation

V. CONCLUSION

NoSQL is not an alternative substitute for traditional Relational Database Management Systems. Each kind of database suits for different needs. And that is why each solution must be evaluated for each application.

Although, we believe NoSQL Databases are very good for applications that deal with large amount of data and which has some main entities that associates to many other secondary entities.

Our future efforts will focus on using a NoSQL database to store data for two study case applications: an

Academic Administrative Tool and an Electronic Document Management. This two domain systems should demonstrate environments on which NoSQL databases may be adopted.

ACKNOWLEDGEMENTS

I would like to thank some people who have encouraged me to begin this work, including professor Fabio Nogueira de Lucena and professor Kleber Vieira Cardoso; my colleagues at Instituto de Informática - Universidade Federal de Goiás. Also, my family members and my loved girlfriend who inspire me to work harder and harder every day.

REFERENCES

[1] R. Agrawal, H. Garcia-Molina, J. Gehrke, L. Gruenwald, L. M. Haas, A. Y. Halevy, J. M. Hellerstein, Y. E. Ioannidis, H. F. Korth, D. Kossmann, S. Madden, A. Ailamaki, R. Magoulas, B. C. Ooi, T. O'Reilly, R. Ramakrishnan, S. Sarawagi, M. Stonebraker, A. S. Szalay, G. Weikum, P. a. Bernstein, E. a. Brewer, M. J. Carey, S. Chaudhuri, A. Doan, D. Florescu, and M. J. Franklin, "The Claremont report on database research," *Communications of the ACM*, 2009.

[2] N. Leavitt, "Will NoSQL Databases Live Up to Their Promise?" *Technology News*, 2010.

[3] A. Lith and J. Mattsson, "Investigating storage solutions for large data," Master's thesis, Department of Computer Science and Engineering – Chalmers University of Technology, 2010.

[4] M. Stonebraker, "SQL databases v. NoSQL databases," *Communications of the ACM*, 2010.

[5] D. Nussbaum and A. Agarwal, "Scalability of parallel machines," *Communications of the ACM*, 1991.

[6] K. Orend, "Analysis and Classification of NoSQL Databases and Evaluation of their Ability to Replace an Object-relational Persistence Layer," Master's thesis, Fakultät für Informatik – Technische Universität München, 2010.

[7] J. Dean and S. Ghemawat, "MapReduce : Simplified Data Processing on Large Clusters," *Communications of the ACM*, 2008.

[8] C. Vicknair, X. Nan, Y. Chen, and D. Wilkins, "A Comparison of a Graph Database and a Relational Database," *Access*, 2010.

[9] D. Mytton, "Choosing a non-relational database; why we migrated from MySQL to MongoDB," *Boxed Ice Blog*, 2009.

[10] I. 10gen, "mongoDB web site," http://www.mongodb.org, 2011, [Last visit: june 14th, 2011].