NoSQL in Cloud Computing

Ruisheng Fu
21421190
CCNT Lab.

Guanyu Guo
21421189
LIST Lab.

Chen Sui
21421183

January 3, 2015

Contents

Abstract

A lot of changes in database management system has been made since the inception of cloud computing. Such critical and increasing needs within cloud computing as scalability, elasticity and processing a huge amount of data can be fulfilled by the NoSQL databases as opposed to RDBMS[1]. In this report we have dived into several primary NoSQL databases used in leading cloud vendors, summarizing and discussing detailed techniques as well as analysis with comparisons.

[1] Relational Database Management System

1 Introduction

Cloud computing has been a evolving computing terminology that is very much in the public eye. It is its responsibility to manage and group remote servers that allow data storage and online access to various services.

In the field of computing, the various advancements and aspects are key evidences that explain the reason why higher priorities are given to scalability, resource utilization and power savings, with respect to data storage, rather than consistency. The traditional RDBMSs offer functionalities like clustering, synchronization (always consistent), load balancing and structured querying. However, what classical RDBMS could not do so well is to scale[2] to heavy workloads compared to NoSQL databases. As non-relational databases have cropped up both inside and outside the cloud, there comes heated debate around SQL and NoSQL.[1]

The two solutions, manual sharding and caching,(cite) applied to classical SQL databases are not adequate enough to cope up with the modern web applications, thus agility can't be achieved. On the contrary, NoSQL databases is designed to handle such sort of problems. In those applications where high availability, speed, fault tolerance or consistency are needed, NoSQL is the choice, in that it is designed to scale out, to provide elasticity and to be highly available. The misleading term *NoSQL* should be seen as the definition[5] that is "Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontally scalable", and is mostly translated with "Not only SQL".

In this report, we firstly examine several major NoSQL databases implemented and used in cloud vendors like Google, Amazon, and Yahoo, along with description about the main ideas of each design. Afterwards, analysis towards different NoSQL databases and we present benchmarking of top NoSQL as a visualizing comparison. Finally, a brief summarization is included in conclusion and further studies as well as challenges is discussed.

2 Major Stream

Currently there are approximately 150 NoSQL databases categorized by data models into a bunch of classes.[5] We review certain amount of prevailing

[2] Scaling, in a google sense, means that an application runs on small commodity PC hardware, but supports essentially unbounded load as more PC's are added.

NoSQL databases through several aspects, including data model, partitions, availability, and consistency.

2.1 Data Model

The major data models adopted we are going to investigate are Wide Column Store (Column Families), Document Store, and Key Value Store, whereas the minor ones (Graph Databases, Object Databases, etc.) are beyond our scope of discussion in this report.

Column Family Store Columnar databases are almost like tabular databases. The difference is that the data are stored and retrieved column-wise unlike traditional RDBMS. Apache Cassandra: Casandra comes under column family. This database is designed to process the data which are spread across different servers without a single point failure. It is used by many companies such as digg, Cisco , eBay etc,.

Document Store A document-oriented database eschews the table-based relational database structure. MongoDB is one of them. It stores business subjects in the minimal number of documents instead of breaking it up into relational structures[2] in favor of JSON-like formats with dynamic schemas.[4] This flexibility facilitates the mapping of documents to an entity or an object in MongoDB, in which there are two tools to allow applications to represent relationships between data: references and embedded documents.[3]

Key Value Store

2.2 Availability

replication

load balancing

2.3 Partition

2.4 Consistency

3 Performance

4 Conclusion

CAP consistency, availability, partition only 2 of 3

Help

1

Linking words are important. If you are grouping together writers with similar opinions, you would use words or phrases such as:

similarly, in addition, also, again

More importantly, if there is disagreement, you need to indicate clearly that you are aware of this by the use of linkers such as:

however, on the other hand, conversely, nevertheless

At the end of the review you should include a summary of what the literature implies, which again links to your hypothesis or main question.

2

What reporting verbs should I use? Avoid over-using ”states” and ”says”. You may need to use tentative or evaluative verbs. • Tentative verbs are often used to show that findings are incomplete or difficult to generalise from. e.g. Research suggests that a majority of people prefer email to... (Mahlab 1995). Wang (2003) indicates that such results are not necessarily... • Evaluative verbs can pack in extra meaning by incorporating your evaluation of the text. e.g. Jacob concedes that the test is not 100 per cent reliable. This is much stronger than ”Jacob states that... ” since concedes includes the judgement that Jacob was reluctant to make the acknowledgement. Some other strong reporting words are: describe, contend, examine, assert, dispute, claim, purport, persuade, refute, concur, recommend, object, dismiss, contradict, propose, examine, observe

3

It really depends on the needs of your application. For web applications that have light querying, key/value stores are very useful. For enterprise databases where reporting is typically very heavy, relational databases fit better. I cannot really comment on systems like Cache as I have not used them.

For example, instead of storing title and author information in two distinct relational structures, title, author, and other title-related information can all be stored in a single document called Book, which is much more intuitive and usually easier to work with.

References

[1] Tony Bain. Is the Relational Database Doomed?, 2009. URL: http://readwrite.com/2009/02/12/is-the-relational-database-doomed.

[2] Steve Hoberman. *Data Modeling for MongoDB*. Technics Publ., 2014.

[3] MongoDB Inc. Data Modeling Introduction, 2009. URL: http://docs.mongodb.org/manual/core/data-modeling-introduction/.

[4] Rico Suter. MongoDB: An Introduction and performance Analysis. In *Seminar Thesis, Rapperswil*, 2012.

[5] Unknown. NOSQL Databases, 2012. URL: http://nosql-database.org/.