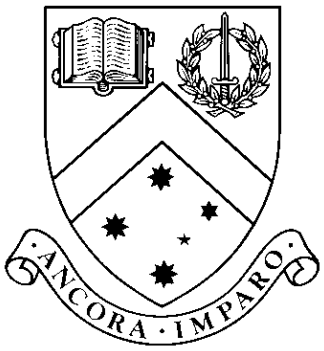
Clayton School of Information Technology
Monash University



Honours Reading Unit — Semester 1, 2015

A brief overview of NoSQL databases

Jonathan Poltak Samosir
[2271 3603]

Supervisors: Dr Maria Indrawan-Santiago
Dr Pari Delir Haghighi

Contents

1 Introduction

The use of relational database management systems (RDBMS) have for a considerably long time been almost treated as if they were a pseudo-standard practise when it comes to the storage and management of data in data sciences, industry, and application development in general. Relatively recently, we have seen a surge of a new, or different type, of data storage and management systems, currently which are commonly known and referred to as "NoSQL" databases. Through the use of the "NoSQL" label, what is actually meant is generally that the database does not use the traditional relational data structure. The "NoSQL" label that has been given to these types of database was rather controversial in itself, and was a topic of hot debate given the confusion and misconceptions that ensue from this name. The origin and subsequent use of the "NoSQL" label will be a topic that will be delved into further in this essay.

This essay will provide an explanation of what NoSQL databases actually are, and how they differ from the more traditional relational database model. This will be covered in §2. An explanation will be given of the different categories of NoSQL databases, along with how they differ, in §3. Finally examples of NoSQL database, and actual use cases for each example, will be given in §4 before concluding in §5.

2 "NoSQL" vs "SQL"

The label "NoSQL" has arguably been a major source of confusion for many people unfamiliar with the various database technologies. SQL, or more descriptively Structured Query Language, has been in use since being first introduced in the 1970s by Chamberlin and Boyce at IBM [4], referred to then simply as "SEQUEL". Chamberlin and Boyce, upon working on "SEQUEL", cited the works of E. F. Codd on the relational model of data, of which all modern relational databases are based upon, as their underlying data structure. Codd presented the relational model of data as a very simple view of data based upon relations as defined and used in algebraic set theory [6]. Hence, SQL was developed with working with the relational data model in mind.

Since then, both relational databases along with SQL, with slight variants in the language depending on certain features vendors may allow in their databases, have been used almost exclusively for general purpose data storage and manipulation for a number of varying use cases which share the same requirement: the accommodation of data available on an on-demand basis for a large number of users. More recently, within the last decade, a rise in the use of non-relational databases to accommodate similar use cases has been noted [24]. As was stated in §1, these non-relational databases are currently what we refer to as NoSQL databases. This label is of much confusion, but in this document, by the use of the NoSQL label, what is meant is fundamentally databases which are non-relational.

What should be noted is that while there is a lot of talk and activity in

the world of NoSQL databases at present, NoSQL databases have existed since the 1960s, whether it be as conceptual databases or in active use. However, it has not been until recently where we have seen such as surge in activity and popularity surrounding them [14].

NoSQL databases are not a homogeneous group within themselves. NoSQL databases vary within their own group considerably, in terms of underlying data structures in which the data is stored and manipulated. These variations present in NoSQL databases will be further elaborated on in §3. however the shared characteristics of databases falling under the NoSQL label will be touched upon here.

2.1 The CAP theorem

In 2000, Brewer put forward the Consistency Availability Partition (CAP) theorem [2], which in its most simple form states that, in the case of NoSQL databases, a database may only have two of the following features [1]:

- **Consistency (C)**: having all managed data up-to-date, or consistent with one another.
- **High availability (A)**: having all managed data available for use in updates.
- **Tolerance to network partitions (P)**: tolerance to message loss between different data partitions on a network.

As Brewer states, only two of the three features can be properly adhered to, given the tension between each of them, hence different NoSQL databases may differ in the way of which CAP theorem properties they adhere to.

2.2 Suitable use-cases for NoSQL

Both relational and NoSQL databases are intended to coexist, both generally fitting into specific use-cases. According to Vicknair, et al. [27] and Kleppmann [13], if a use-case exhibits the following signs, a NoSQL database may be more appropriate for use than a relational database:

1. Data tables containing many columns, of which are scarcely used by all rows.
2. Having separate tables for attributes.
3. Data exhibiting large amount of many-to-many relationships.
4. Data exhibiting tree-like attributes.
5. Data exhibiting dynamically changing schemas.

Of course, whether or not the use-case is suitable for non-relational database then also leads to the problem of the category of non-relational database which would be most suitable. These categories of NoSQL databases are touched upon in §3.

3 Categories of NoSQL Databases

As explained earlier, NoSQL databases are heterogeneous in terms of their underlying data structures, and vary considerably amongst themselves. Here, the most common differently structured categories of NoSQL databases will be listed and elaborated on. Examples of such databases will be given for each category, however these databases will not be elaborated on here; this will be given in §4.

There are four commonly accepted distinct categories of NoSQL databases [26]:

- Document-oriented databases
- Key-value stores
- Column-oriented databases
- Graph databases

3.1 Document-oriented databases

Document-oriented databases, or simply document databases, are arguably the most commonly used form of NoSQL database, with popular open-source databases, such as MongoDB [18] and CouchDB [7], falling into this category [26].

Essentially what is meant by the term "document-oriented" is the use of structured document types, such as JSON or XML, for example, as the underlying data structure for storing data in [10].

This style of document-oriented database is often popular with programmers who may have prior experience with the underlying document type. Especially so for popular document types, such as JSON and XML.

3.2 Key-value stores

In key-value databases, the data is stored in single data sets made up of a unique key element and a corresponding value elements. Keys are unique and generally of a single data type, for example an integer, while values are accepted as being of multiple types, including key-value pairs themselves, allowing nested data sets [28]. These key-value pairs making up an individual data point in the database should be fairly familiar to most general computer programmers, which are often used as built-in data structures in common programming languages, known as hashes or associative arrays.

Note that key-value store based databases can be very similar in internal data storage structure to document-orientated databases which make use of the

JSON document structure. This is because the JSON document structure is essentially based upon the idea of keys and values.

Popular examples of key-value store based databases are Cassandra [3], and Berkeley DB [23].

3.3 Column-oriented databases

Column-oriented databases stem from Google's original BigTable database [5], which has been used numerous times within many different Google projects. In general, column-oriented databases use an underlying table-like structure for the storing of data, much like in a general relational database. In fact, the main difference between column-oriented databases and relational databases is in their handling of null values [11]. In relational databases, a null value is generally permitted where an attribute in a relation has no value. In column-oriented databases, a value is only stored in a row if it is really needed, hence allowing what would be considered "relations", in a relational database, to be missing attributes rather than having an explicit null value [11]. This allows for convenient storage of heterogeneous datasets, which is a key characteristic of NoSQL databases (or the lack of strict schemas).

Examples of column-oriented databases include Google's BigTable [5], and the open-source BigTable implementations, HBase and Hypertable [12].

3.4 Graph based databases

Graph based databases are a much more obscure structured type of database, where the underlying data storage structure reflects a graph data structure; thus being made up of nodes and edges connecting those nodes. Nodes represent sets of data, while the edges represent relationships between the data sets [27].

A common example use-case of a graph based database would be that of a social network. Each node would represent the data about a member of that network, while the edges would represent the connections between that member and other members of the network.

Given the grounding and possibilities that come from graph theory, a graph based database is a very flexible structured database, and can be molded into use with a number of non-obvious use-cases. However, compared to other categories of NoSQL databases, the graph based database is arguably the least commonly used style of NoSQL database. Commonly used examples of a graph based database would be Neo4j [8] and GraphDB [21].

4 Examples of NoSQL Databases

In terms of databases in general, usage of relational databases, such as MySQL [19], Microsoft SQL Server [16], MariaDB [15], and Oracle Database [22], is still very commonplace in a lot of applications. Most of these databases are considered to be quite mature, built upon well tested and proved concepts, and have been

used in production for a considerable amount of time. In contrast, most of the new NoSQL databases that are currently being used in production are relatively new, have not been used in production as extensively as their relational counterparts, and as such are often considered to be less mature solutions. However with the success of big companies who use NoSQL databases in production, such as BigTable at Google [5] and the thousands of organisations using MongoDB in production [17], this stereotype is slowly changing.

Some examples of NoSQL databases that will be briefly looked at include MongoDB, Neo4j, and BigTable.

4.1 MongoDB

MongoDB is a document-oriented NoSQL database, that is currently gaining fast user adoption. MongoDB, first developed in 2007 at 10gen, now MongoDB Inc., is open-sourced under the GNU Affero General Public License v3.0, currently at version 3.0.1 [18] MongoDB uses a document structure based upon JSON which makes use of binary form for data storage, called BSON, or Binary JSON [9].

MongoDB provides a console interface, Javascript as its query language, along with allowing a simple REST interface for interactions. MongoDB uses write-ahead journaling for its durability [25].

4.2 Neo4j

Neo4j is a graph based NoSQL database, developed at Neo Technology, Inc., dual licenced under both the open-source GNU Affero General Public License v3.0 licence and a commercial licence. Neo4j is currently at version 2.2.0 [8]. In Neo4j, all data is stored either as an edge or a node, or an attribute on either an edge or a node [20].

Neo4j provides a console interface, Java programming API, along with a REST API with search indices to allow users and programs to interface and interact with the database [25]. Neo4j can be considered to have ACID durability, something that most NoSQL databases reject [25].

4.3 BigTable

BigTable is a column-oriented database, developed at Google, Inc., in 2004, in use at Google as an inhouse product. It is not licenced for used outside of Google, however it has been written about and documented extensively. This has led to third-party open-source implementations of BigTable, such as HBase. As a column-based database BigTable's underlying data storage structure very much reflects that of a relational database, using tables to store data [5].

BigTable was designed with the use-case in mind of storing mass amounts of data — petabyte scale — for use in Google's mass market products, such as Google Search, Google Maps, and Google Analytics [5]. Thus, this puts

BigTable into the big data category of databases, something which more general purpose databases are not particularly suitable for.

5 Conclusion

In summary, NoSQL databases, while the concept has been around for a number of decades, with recent developments in database technology have presented themselves as a viable option for certain use-cases where relational database solutions are not entirely appropriate. With fast gaining adoption in industry, and further research into NoSQL technology, NoSQL databases are expected to gain further traction and gain even more market share in the previously relational dominated field of consumer databases.

The differences between what makes NoSQL databases and relational databases has been discussed, along with a look at the most commonly accepted categories of NoSQL databases, and what makes each category distinct. A number of popular, different types of NoSQL databases have been given as examples for the concepts discussed in previous sections.

References

[1] Eric Brewer. Pushing the cap: Strategies for consistency and availability. *Computer*, 45(2):23–29, 2012.

[2] Eric A Brewer. Towards robust distributed systems. In *PODC*, volume 7, 2000.

[3] Cassandra.apache.org. The apache cassandra project, 2015.

[4] Donald D Chamberlin and Raymond F Boyce. Sequel: A structured english query language. In *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, access and control*, pages 249–264. ACM, 1974.

[5] Fay Chang, 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. *ACM Transactions on Computer Systems (TOCS)*, 26(2):4, 2008.

[6] Edgar F Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[7] Couchdb.apache.org. Apache couchdb, 2015.

[8] Neo4j Graph Database. Neo4j, the world's leading graph database, 2015.

[9] Docs.mongodb.org. Mongodb manual contents — mongodb manual 3.0.1, 2015.

[10] Jing Han, E Haihong, Guan Le, and Jian Du. Survey on nosql database. In *Pervasive computing and applications (ICPCA), 2011 6th international conference on*, pages 363–366. IEEE, 2011.

[11] Robin Hecht and Stefan Jablonski. Nosql evaluation. In *International Conference on Cloud and Service Computing*, pages 336–41, 2011.

[12] Ankur Khetrapal and Vinay Ganesh. Hbase and hypertable for large scale distributed storage systems. *Dept. of Computer Science, Purdue University*, 2006.

[13] Martin Kleppmann. Should you go beyond relational databases? - treehouse blog, 2009.

[14] Neal Leavitt. Will nosql databases live up to their promise? *Computer*, 43(2):12–14, 2010.

[15] Mariadb.org. Welcome to mariadb! - mariadb, 2015.

[16] Microsoft.com. Sql server 2014 — microsoft, 2015.

[17] Mongodb.com. Organizations creating applications never before possible, 2015.

[18] Mongodb.org. Mongodb, 2015.

[19] Mysql.com. Mysql :: The world's most popular open source database, 2015.

[20] Neo4j.com. The neo4j manual v2.2.0-rc01 -, 2015.

[21] Ontotext. Ontotext graphdbô - powerful graph database - ontotext, 2015.

[22] Oracle.com. Database 12c — oracle, 2015.

[23] Oracle.com. Oracle berkeley db products — overview, 2015.

[24] Rabi Prasad Padhy, Manas Ranjan Patra, and Suresh Chandra Satapathy. Rdbms to nosql: reviewing some next-generation non-relational databases. *International Journal of Advanced Engineering Science and Technologies*, 11(1):15–30, 2011.

[25] Eric Redmond and Jim R Wilson. Seven databases in seven weeks: A guide to modern databases and the nosql movement. 2012.

[26] Clarence JM Tauro, S Aravindh, and AB Shreeharsha. Comparative study of the new generation, agile, scalable, high performance nosql databases. *International Journal of Computer Applications*, 48, 2012.

[27] Chad Vicknair, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen, and Dawn Wilkins. A comparison of a graph database and a relational database: a data provenance perspective. In *Proceedings of the 48th annual Southeast regional conference*, page 42. ACM, 2010.

[28] Silvan Weber. Nosql databases. *University of Applied Sciences HTW Chur, Switzerland*, 2010.