Computer Science and Information Technology 5(5): 149-153, 2017 http://www.hrpub.org
DOI: 10.13189/csit.2017.050501

A Comparison between Characteristics of NoSQL Databases and Traditional Databases

Mitko Radoev

Department of Information Technologies and Communications, Faculty of Applied Informatics and Statistics, University of National and World Economy, Sofia, Bulgaria

Abstract With the increasing popularity of NoSQL databases, the question arises if they have all characteristics of databases and can they be a real alternative to the relational databases in all application domains. This paper makes an attempt to systematize the most important characteristics of the traditional databases and then to analyze whether NoSQL databases have these characteristics. On this basis it is possible to draw a conclusion whether NoSQL databases have the necessary qualities to be called databases, or rather they are data stores with limited capabilities. The results of the comparison shows that none of the NoSQL DBMS under consideration covers more than 50% of the characteristics of the traditional databases so the use of the term "database" in respect of any one of them is not fully correct.

Keywords Databases, Relational Databases, NoSQL Databases

1. Introduction

In recent years, so-called NoSQL databases have become increasingly popular. The term NoSQL databases are interpreted in different ways. It should literally refer to databases that do not use the SQL language. There is also an interpretation that it means Not Only SQL. In fact, they are databases that do not use SQL, but also they are not based on the relational data model at all, and therefore the more precise term would be Non-relational databases. Moreover, they are not hierarchical nor network databases (the predecessors of the relational databases). They are an entirely new type of databases based on different data models. Further on, the article will look at the different NoSQL databases according to the models on which they are based.

What is the growing popularity of these new databases based on? The reasons for this are to be analyzed, but it is a fact that in most database management system rankings, NoSQL databases are already among the Top 10. According to DB-engines [1], the Top 10 most popular DBMSs in September 2017 are:

1. Oracle
2. MySQL
3. Microsoft SQL Server
4. PostgreSQL
5. MongoDB
6. DB2
7. Microsoft Access
8. Cassandra
9. Redis
10. Elasticsearch

According to a survey, conducted by StackOverflow [2] among software developers, the most popular 8 DBMS for 2017 are:

1. MySQL
2. Microsoft SQL Server
3. PostgreSQL
4. SQLite
5. MongoDB
6. Oracle
7. Redis
8. Cassandra

In the both rankings NoSQL database management systems like MongoDB, Cassandra and Redis are among the first 10 places, and they are constantly improving their positions compared to the previous periods. The use of NoSQL databases is increasing not only in developing new systems, but some of the relational database users have decided to replace them by these new alternative databases. Are all users aware of what they choose and what consequences of their choice would be? Because the term database is used, although NoSQL, users often expect to get everything they used to get from databases, along with the benefits that NoSQL DBMS undoubtedly have. Unfortunately, everything has a price, and each of the benefits of NoSQL databases is paid for by the loss of traditional database capabilities - control of data

redundancy, ensuring data integrity, maintaining transaction, and so on. Ironically, what is most missing from NoSQL databases is SQL [3]. There are developers who have already experienced in practice the deficiencies of the new data stores and some of them decided to return back to the relational databases [4].

To be able to make informed choice, users must be fully aware of the characteristics of the database systems. Last but not least, they have to be aware of whether the NoSQL databases have the necessary qualities to be called databases, or they are rather data stores with limited capabilities compared to relational databases. The main purpose of this article is to analyze the characteristics of NoSQL databases in comparison to the characteristics of traditional databases and on this basis to answer the question whether the NoSQL databases are real databases.

2. Traditional Databases

The first databases that appeared used hierarchical (tree-like) or network structures to store data. Thereafter, relational databases emerged and rapidly gained popularity, and to date they are still the most common databases. These three types of databases will be called traditional.

2.1. Genesis of the Databases

It is necessary to review briefly the appearance and development of databases in order to recall the problems that seem to be forgotten today.

Databases originated in the 1960s in order to replace the existing file organization of data. What are the disadvantages of organizing data in separate files? Here are some of the main drawbacks:

1. Redundant data

Storing the data required by each application program results in serious duplication of data. Data duplication is undesirable, not only because it leads to extra storage, but above all because it can lead to inconsistent data.

2. Isolated data

When data is stored in separate files it is difficult to process it together. The situation is complicated by the use of different file formats. Due to the fact that the file structure is defined in the application programs, file formats are different according to used programming tools and developers' decisions.

3. Program–data dependence

The physical structures of files and data records are defined in the program code. This means that changes to existing structures are difficult to make. However, if a data structure changes are necessary, all programs that have access to the changed file must be modified.

4. No ad hoc queries

In order to get new reports, new programs need to be written to implement them, as there are no possibilities for users to ask ad hoc queries.

All of these limitations of the file organization of data are due to two major factors:

- The data definitions are embedded in the application programs instead of being stored together with the data;
- There are no data access and processing capabilities other than those provided by the application programs.

The first types of databases created were based on the hierarchical and network data models. They made an attempt to solve the problems of the existing file organization of data, but they failed to solve all the problems because they did not provide sufficient data independence, and the implementation of queries was rather complicated as they were navigationally oriented.

The crucial moment in database development advancement occurred in 1970 when Edgar Frank Codd published his famous work on the relational model of data [5]. By the late 1970s and early 1980s, there had been already database management systems based on the relational data model.

Putting relational model into practice leads to the creation of extremely powerful and flexible tools such as relational database management systems, which rapidly became prevalent and dominant. The subsequent development of these systems leads to expanding their capabilities, such as the creation of SQL language, which further reinforces their dominant position, which is still the case today.

2.2. Characteristics of the Databases

The databases have a large number of characteristics. In this part was made an attempt to systematize the most important of them, using information from different sources [6-8].

1. Structured data

Data in the database is structured according to the used data model. On the other hand, the structure of the data has to match to the entities of the real world and their essential properties.

2. Related data

In addition to information about the real world entities and their properties, the database also stores information about the essential links between them. In the hierarchical and network models, these links are set explicitly. There is no such requirement in the relational model, links can be realized between any pair of attributes (or sets of attributes) having a common domain. In practice however, almost all relational databases provide some means of

setting the essential relationships, for example through the foreign key constraints.

3. Metadata

Together with the data itself, the database also stores metadata, data about the data. Metadata include information about data structures as well as data integrity constraints, security information etc.

4. Data sharing

The database is a common resource within enterprises or organizations and is used by most (ideally all) application programs serving their business.

5. Restriction on data duplication

Databases are trying to eliminate or minimize data duplication. In the relational model, for example, it is necessary to duplicate the values of the primary keys as foreign keys in order to model the relationships between the entities.

6. Data integrity

Databases usually ensure data integrity by imposing restrictions. These restrictions can apply to the entities and their properties, and to the relationships between entities.

7. Data security

Data security is primarily related to data protection against unauthorized access. This is done via various mechanisms, such as usernames and passwords. User access to data may also be limited by the type of operation (retrieval, insertion, update, deletion).

8. Data reliability

Databases ensure reliable data storage and recovery mechanisms in case the database gets damaged.

9. User views

Databases allow users to define different views to the same data. Each user may have a specific view presented in a form that is familiar to them. The view includes information only about those entities, attributes and relationships from the real world, which the user is interested in.

10. Data independence

Databases are based on a multi-level architecture that ensures the independence of the external structure (user views) from the logical structure of the data, as well as the independence of the logical structure from the physical representation of the data.

11. Data storage, retrieval, and update

The databases provide users and application programs with a mechanism for storing, updating and retrieving data from the database through the database management system (DBMS).

12. Ad hoc queries

Data in the database is directly accessible to end users. Many DBMSs provide query languages or report generators that allow users to get the necessary information without writing a program to retrieve this information from the database.

13. Transaction support

The databases provide a mechanism to ensure execution of transactions, i.e. sequences of actions that are logically related and should be implemented as one action. Transactions must meet the following requirements:

- Atomicity - All actions are executed successfully, or the entire transaction is canceled;
- Consistency - Once the transaction has been executed, the database must be in a consistent state;
- Isolation - The execution of one transaction should not affect the execution of the others transactions;
- Durability - Transaction result must be stored reliably.

3. NoSQL Databases

As already mentioned, NoSQL databases are newly emerging non-relational databases. The history of NoSQL databases, the types of NoSQL databases and their characteristics are presents in the following parts.

3.1. Genesis of the NoSQL Databases

This new trend in databases began in the early 21st century with the development of Internet applications and, above all, Google's applications. One of the first publications on the topic dates back to 2003 and is linked to the Google File System [9]. Then, publications related to other Google's systems appeared, such as MapReduce [10], Chubby [11] and Bigtable [12]. Google's Internet applications are followed by Yahoo, Amazon, and later by Facebook, Netflix, EBay and many others, which have led to many new NoSQL databases.

Here are the most significant reasons for the emergence and rapid development of this new direction:

1. Need to store and process huge volumes of data

The amount of photos, videos, geographic data etc. stored and processed by different applications increases every day.

2. Need for real-time access

Stored huge volumes of data must be available in real time from anywhere in the world.

3. Need for flexibility

Ability to easily and quickly change the structure of the

data.

4. Need to store unstructured data

Unstructured or semi-structured data should also be stored along with structured data.

5. Need for scalability

It must be easy to extend the scale of applications and stored data.

Traditional databases cannot meet all the new requirements to the necessary extent. This opens up a niche for creating and developing many new systems based on different models.

3.2. Types of NoSQL Databases and Their Characteristics

The following main types of NoSQL databases can be identified, depending on the data model used:

1. Key-value store

This is one of the simplest models for storing data. Key-value pairs are stored with the keys uniqueness. Data is accessed by searching through the key values. It is suitable for storing large volumes of data and provides quick access by the key. There are different varieties depending on the memory in which the data is stored, the key sorting and data consistency. The most popular systems are Redis, Memcached, Berkeley DB and Oracle NoSQL.

2. Document store

As can be seen from their name, they are designed specifically for document storage. The formats used are XML, JSON, BSON, and others. Data is semi-structured and contains pairs of attribute name-value. Data is accessed by searching both on key values and attribute values. They are suitable for storing text and XML documents and other semi-structured data. The most famous products of this type are MongoDB, Amazon Dynamo DB, Couchbase, and CouchDB.

3. Column-oriented

Unlike relational databases that are row-oriented, this type of data structure is column-oriented. This makes it possible to easily expand the data structure by adding new columns. They are suitable for large volumes of distributed data. Typical representatives are Cassandra, HBase and Google Bigtable.

4. Graph-oriented

The data is represented by a graph-like structure, with the nodes of the graph representing the objects and their set of attributes, and the edges of the graph representing links between the objects. They are suitable for representing data when the links between objects are particularly important for modeling. The most famous products of this type are Neo4j, Titan, Giraph.

There are also products based on two or more different data models. Such are OrientDB, ArangoDB and many others.

According to Brewer's CAP Theorem [13], distributed systems cannot have more than two of the following characteristics at the same time:

- Consistency;
- Availability;
- Partition tolerance.

While traditional databases are focused on ensuring data consistency, the NoSQL databases in most cases prioritize high availability and partitioning, losing the consistency of data. This tendency results in creating systems known as BASE (Basically Available, Soft-state, Eventually consistent). It cannot be ignored that these systems, while trying to solve some problems created other, perhaps more serious, problems.

4. Comparison between NoSQL Databases and Traditional Databases

The main purpose of this article is to compare NoSQL databases with traditional databases, and more specifically, to check which of the characteristics of traditional databases are also characteristics of the NoSQL databases.

The variety of products from each of the types of NoSQL databases leads to significant differences in their characteristics. The characteristics of the most popular products from each category are taken into account when performing the analysis. These are:

1. Redis for key-value stores;
2. MongoDB for document stores;
3. Apache Cassandra for column-oriented stores;
4. Neo4j for graph-oriented stores.

Data from their web sites and other sources have been used to determine product characteristics.

The results of the comparison between characteristics of the traditional and NoSQL databases are summarized in Table 1.

The final results of the comparison are eloquent - none of the products under consideration covers more than 50% of the characteristics of the traditional databases. At the same time, the following should be taken into account before drawing final conclusions:

- Not all database characteristics are equally important to determine their essence;
- The information available for the products is not sufficient to determine categorically possession or non-possession of a particular characteristic;
- Graph-oriented systems differ from the other NoSQL data stores and, although with boundary values, have the potential to be true databases.

Table 1. Match between characteristics of the traditional and NoSQL databases

Characteristics	Key-value (Redis) [14,18]	Document (MongoDB) [15]	Column-oriented (Cassandra) [16]	Graph-oriented (Neo4j) [17]
Structured data	no	partly [1]	yes	yes
Related data	no	no	no	yes
Metadata	no	no	no	no
Data sharing	yes	yes	yes	yes
Restriction on data duplication	no	no	no	no
Data integrity	no	no	no	partly [3]
Data security	no	yes	yes	yes
Data reliability	yes	yes	yes	no
User views	no	no	no	no
Data independence	no	no	no	no
Data storage, retrieval, and update	yes	yes	yes	yes
Ad hoc queries	no	no	no	no
Transaction support	yes	no	partly [2]	yes
Total	**31%**	**35%**	**42%**	**50%**

[1] MongoDB stores semi-structured data.
[2] Cassandra uses batches which are not a full analogue for transactions.
[3] Neo4j allows the imposition of some data constraints.

5. Conclusions

Since none of the products in question have more than 50% of the characteristics of traditional databases, the use of the term "database" in respect of any one of them is not entirely correct.

Instead of NoSQL databases, the products of this category would be more accurately called Non-relational data stores.

Graph-oriented storage systems most closely match characteristics of traditional databases and have the potential to develop into complete databases.

REFERENCES

[1] DB-Engines Ranking, Online available from https://db-engines.com/en/ranking.

[2] Stack Overflow, Online available from https://insights.stackoverflow.com/survey/2017.

[3] S. Tiwari, Professional NoSQL, John Wiley & Sons, Indianapolis, 2011.

[4] S. Mei, Why You Should Never Use MongoDB, Online available from http://www.sarahmei.com/blog/2013/11/11/why-you-should-never-use-mongodb/.

[5] E. F. Codd, A relational model of data for large shared data banks, Communications of the ACM, Vol. 13, № 6, pp. 377 - 387, June 1970. E. F. Codd, A relational model of data for large shared data banks, Communications of the ACM, Vol. 13, № 6, pp. 377 - 387, June 1970.

[6] E. F. Codd, Relational database: A practical foundation for productivity, Communications of the ACM, Vol. 25, № 2, pp. 109-117, February 1982.

[7] C. J. Date, An Introduction to Database Systems, 8th edition, Pearson/Addison Wesley, 2004.

[8] T. Connolly, C. Begg, Database Systems: A Practical Approach to Design, Implementation, and Management, 4th edition, Pearson Education Ltd, 2005.

[9] S. Ghemawat, H. Gobioff, S. Leung, The Google File System, 19th ACM Symposium on Operating Systems Principles, Lake George, 2003.

[10] J. Dean, S. Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, 6th Symposium on Operating System Design and Implementation, San Francisco, 2004.

[11] M. Burrows, The Chubby Lock Service for Loosely-Coupled Distributed Systems, 7th Symposium on Operating System Design and Implementation, Seattle, 2006.

[12] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, R. Gruber, Bigtable: A Distributed Storage System for Structured Data, 7th Symposium on Operating System Design and Implementation, Seattle, 2006.

[13] A. Fox, E. Brewer, Harvest, Yield and Scalable Tolerant Systems, Proc. 7th Workshop Hot Topics in Operating Systems, pp. 174–178, 1999

[14] Redis Documentation, Online available from https://redis.io/documentation/.

[15] The MongoDB 3.4 Manual, Online available from https://docs.mongodb.com/manual/.

[16] Apache Cassandra Documentation v4.0, Online available from http://cassandra.apache.org/doc/latest/.

[17] The Neo4j Developer Manual v3.2, Online available from https://neo4j.com/docs/developer-manual/current/.

[18] A. Moniruzzaman, S. Hossain, NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison, International Journal of Database Theory and Application, Vol. 6, № 4, pp. 1-14, 2013.