International Journal of Management, IT & Engineering
Vol. 8 Issue 6, June 2018,
ISSN: 2249-0558 Impact Factor: 7.119
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

DIFFERENCE BETWEEN SQL AND NOSQL DATABASES

Kalyan Sudhakar *

Abstract

This article explores the various differences between SQL and NoSQL databases. It begins by giving brief descriptions of both SQL and NoSQL databases and then discussing the various pros and cos of each database type. This article also examines the applicability differences between the two database types. Structured Query Language (SQL) refers to a domain-specific or standardized programming language used by database designers in designing and managing relational SQL databases or controlling data stored in RDSMS (relational database management system). The scope of SQL database encompasses data query, data definition (creation and modification of schema), data manipulation (inserting, deleting, and updating), as well as control of data access. NoSQL databases are non-relational and can exist together with relational databases. NoSQL databases are most suitable for use in applications that involve large amounts of data, and the data can either be unstructured, structured, or semi-structured. SQL databases have been the most widely used databases in the world over the years. However, it is the requirements that drive data models, which in turns, influence the choice between SQL and NoSQL databases.

.

* **Data Engineer**

1. Introduction to SQL Database

Structured Query Language (SQL) refers to a domain-specific or standardized programming language used by database designers in designing and managing relational SQL databases or controlling data stored in RDSMS (relational database management system) [10]. Some of the common SQL databases include MySQL, Oracle, Sybase, and IBM DM2 [7]. In such SQL databases, SQL helps in executing queries, retrieving data, and editing data by deleting, updating, and creating new data records [7]. SQL is a lightweight and declarative language that performs several heavy liftings for the relational database by acting as a version of the server-side script of the database [1].

The SQL concept got introduced in the 1970s, and it is often used by database developers and administrators in performing various activities, such as setting up and running analytical queries, as well as writing data integration scripts [10]. SQL databases are built around tuple relational calculus and relational algebra, and they provide mechanisms for data storage and retrieval through the use of tables [10]. The tables used for data storage in SQL databases exist in the form of Excel spreadsheets with columns and rows. The only difference is that a database table has a much stricter structure [7]. A row in such data tables represents a single record, whereas the columns form the attributes for that particular record. SQL databases are specifically used for handling structured data whose different variables or entities have relations, and they allow users to access several records using a single command [7].

SQL databases consist of a broad range of statements, often referred to as sublanguages and include DQL (data query language), DCL (data control language), DDL (data definition language), and DML (data manipulation language) [3]. The scope of SQL database encompasses data query, data definition (creation and modification of schema), data manipulation (inserting, deleting, and updating), as well as control of data access [10]. Despite being described largely as a 4GL (a declarative language), SQL also has various procedural elements. SQL became the ANSI (American National Standards Institute)'s standard in the year 1986 and a standard of the ISO (International Organization for Standardization) in the year 1987 [3], [10]. The SQL database has, since then, been revised to incorporate other features. However, even with such

additional features, most SQL codes are not entirely compatible with other databases without further adjustments [7], [10].

2. Introduction to NoSQL Database

NoSQL implies or stands for "Not Only SQL. NoSQL databases are non-relational and can exist together with relational databases. NoSQL databases are most suitable for use in applications that involve large amounts of data, and the data can either be unstructured, structured, or semi-structured [2], [6]. NoSQL databases exist in different forms or types, which include Apache Cassandra, Coach DB, Mongo DB, HBase, Infinite Graph, and Redis among others. All the NoSQL databases differ in the context of their usage and structure [2], [6]. Additionally, NoSQL databases are different from the relational SQL databases in the sense that while relational databases require the creation of tables, the definition of a schema, and setting of data types before inserting the data, NoSQL databases allow for direct insertion and updating of data [4], [9].

In NoSQL document stores, the data is often stored or kept in collections, which consist of objects known as documents. The collections act as or are the equivalent of tables in relational databases, while the documents resemble the records [8], [9]. Therefore, in NoSQL databases, a collection consists of documents, and each document represents a different user. NoSQL is the next generation database, and most NoSQL databases are open source with the capability of horizontal scalability [8], [9]. Despite the advanced nature of NoSQL database, there exist certain situations where one would prefer SQL database over NoSQL database. However, when handling amounts of data, NoSQL database is often the best choice [5].

NoSQL database is increasingly gaining popularity with its most significant implementations being MongoDB and Apache Cassandra among others [6]. NoSQL database has primarily been developed to address the scalability issue or problem that characterizes SQL database. Besides, NoSQL database has emerged as a schema-free database build on distributed systems, which gives it the required scalability and makes it easy to shard [6], [9]. Despite presenting new challenges, NoSQL database provides great rewards to users who can effectively incorporate or

integrate it into their solution portfolio [4]. The primary benefits of NoSQL database revolve around improved data comprehension, productivity, and flexible scaling solutions [4], [8].

3. Pros and Cons of SQL Database

3.1. Pros

One of the advantages of SQL databases is that they are simple and powerful, which makes it possible for the retrieval of related data stored across different tables using a single command in such databases [9]. SQL databases also fit naturally into a broad range of old software stacks, including Ruby and LAMP-based stacks. Besides, SQL databases are well-understood and broadly supported, which makes it easy for solving common problems [4], [9].

SQL databases also have high speeds, and SQL queries can be used for quick and efficient retrieval of large amounts of records from SQL database [1]. SQL databases also have a well-defined standard, which makes it easy for the management of a substantial amount of data without having to write several codes [1], [7]. Also, SQL databases make use of object-oriented DBMS, which allows for the extension of their object storage capabilities [3], [9].

SQL databases have portability since SQL is compatible with a broad range of computer programs [10]. Besides, SQL databases have an interactive language, which can be used for quick communication with other databases [10]. SQL databases also provide platforms for multiple data views with the help of SQL language. Moreover, SQL databases comply with the ACID (atomicity, consistency, isolation, and durability) principles, which makes them stable, secure, and predictable [9], [10].

3.2. Cons

One of the disadvantages of SQL database relates to the difficulty in interfacing. The process of interfacing an SQL database is more sophisticated than adding a few code lines (Severance 9). Additionally, SQL database programmers do not have full control over such databases because of the concealed business rules [7], [10]. Some SQL databases also go for proprietary extensions as a means of ensuring the vendor lock-in [3], [9]. Also, some SQL databases have high operating costs, which makes their access difficult for some programmers.

Moreover, SQL databases are challenging to scale since their performances often deteriorate as they become larger [3], [10].

4. Pros and Cons of NoSQL Database

4.1. Pros

One of the advantages of NoSQL database is that it has elastic scalability. NoSQL databases are designed for transparent scaling or expansion due to their new nodes features, and they are specifically designed for application in low-commodity hardware [2], [4]. NoSQL databases are, therefore, fit in the current world, where outward scalability is replacing upward scalability [2], [4]. Additionally, NoSQL databases are used for handling big data applications, which RDBMSs cannot manage [4].

Another advantage of NoSQL databases is that they require little hands-on management since they have auto repair and data distribution capabilities, as well as simplified data models, less turning, and fewer administration requirements [3], [6]. NoSQL databases are also economical in the context of their installation since they can easily be installed in less expensive commodity hardware clusters as data volumes and transactions increase. In other words, with NoSQL databases, one can store and process more data at significantly lower cost [5], [9].

4.2. Cons

One of the disadvantages of NoSQL databases is that they are less mature compared to RDBMSs, which have been existing for an extended period, thus becoming more stable and richly functional. In contrast, most NoSQL databases are in their pre-production stages, and most of their features have not been implemented, which makes them have less functionality and low stability [3], [5]. NoSQL databases also have little support. All businesses need the assurance that they will get timely, competent support if a major function of their data management system fails. However, NoSQL databases appear to be open-sourced, with just a few firms managing the databases from the support viewpoint [5], [9]. Besides, most of the NoSQL databases have been designed by startups that lack the required resources to fund global support [5], [8].

Another disadvantage of NoSQL databases is the lack of business intelligence and analytics features. NoSQL databases emerged due to the demands of modern-day web applications. Therefore, most NoSQL database features are aligned to meeting such demands, and NoSQL databases usually provide few analysis features when the demands of a given data application go beyond the typical web app cycle characteristic of 'insert-read-update-delete' [6], [9]. Additionally, NoSQL databases still require administration. NoSQL database introduction aimed at providing a solution that would not require administration. However, NoSQL databases still need more technical skills for both their installation and maintenance [6], [8]. Also, since NoSQL databases are still new, they lack advanced expertise, and it is easier to find an RDBMS specialist than a NoSQL expert. Therefore, any organization intending to implement NoSQL solutions has to proceed with caution [4].

Another downside of NoSQL databases relates to their use of the eventual consistency principle. Meaning, if a particular data item fails to get new updates for a given period, all its accesses will eventually return to the last updated value. That is why NoSQL databases are considered to offer BASE (Basically Available, Soft state, Eventual consistency) guarantees [3], [9]. Also, since there exist several types of NoSQL databases with little uniformity, they vary greatly in the context of their performances, flexibility, scalability, and complexity, which makes it challenging to choose one with the best performance [5].

5. Examples of Where SQL Database is used

One of the applications of SQL databases is writing of data integration scripts by database developers and administrators [7], [10]. SQL databases are also used for the retrieval of information for transaction processing and analytics applications [7], [10]. Also, SQL databases are useful for regular setting and running of analytical queries [1], [7]. SQL databases are used by a broad range of organizations, including healthcare organizations (in managing cancer registries), business organizations (in managing inventories and making trends analysis), as well as educational institutions in managing performance data [8], [10].

6. Examples of Where NoSQL Database is used

One of the areas where NoSQL databases are used relates to systems that are used for storing and retrieving vast amounts of data [5], [9]. NoSQL databases are also used when one is not interested in the relationship between the stored data, as well as when the stored data grows continuously [5]. Additionally, NoSQL databases are used when there is a need for personalization since a personalized experience requires large amounts of data. With a distributed NoSQL database, data can be extended elastically to meet the user's workloads, as well as build and update other information on the fly, thereby allowing for real-time engagement with clients [1].

NoSQL database is also used where there is a need for profile management. The management of user profile is key to both mobile and web applications as it allows for user preferences, online transactions, as well as authentication among other functions [4]. With a NoSQL database, it is easy to increase capacity by adding commodity servers, which makes the database less expensive and much easier to scale [4], [8]. NoSQL databases are also used where big real-time data is required. For an agile enterprise, the ability to obtain real-time information from operational data is critical. NoSQL databases help in increasing operational efficiency, reducing costs, and improving revenues for agile enterprises by allowing them to act immediately on real-time data [4].

Additionally, NoSQL databases are used for content management. NoSQL databases enable businesses to select different contents, aggregate them and present the aggregated content to the customer at the time of interaction. Therefore, NoSQL databases allow enterprises to develop and produce new content quickly, as well as incorporate the user-generated content with the same agility and ease [2], [6]. NoSQL databases are also useful in managing catalogues as their flexible data models allow enterprises to easily aggregate catalogue data in a single database [2], [6].

NoSQL databases are also used by organizations that require 360 degrees customer view. Customers always anticipate a consistent experience irrespective of the channel. At the same time, organizations need to capitalize on cross-sell opportunities, as well as offer the best customer service. With a NoSQL database, organizations that require 360 degrees customer view

can access the same customer data using multiple applications and add new information without interfering with other applications [2]. NoSQL databases are also used in mobile applications. With a NoSQL database, mobile apps can begin with a small deployment and increase as the user base expands, as opposed to deploying a large, expensive relational server from the start [2], [9].

Another area where NoSQL databases can be used is a financial service organization, for fraud detection. With NoSQL databases, customers can get immediate confirmation when they make payments with their debit or credit cards [2], [5]. Also, in an enterprise environment, NoSQL databases can be used for digital communications in the form of online interaction through direct messaging. NoSQL databases provide the elastic scalability and sub-millisecond responsiveness required by digital communication applications [2], [5].

7. Conclusion

SQL databases have been the most widely used databases in the world over the years, and SQL language has become the International Organization for Standardization's standard, as well as the standard for the American National Stands Institute. However, a challenging question still exists for any organization or company that has to choose between the traditional SQL database and NoSQL database. Organizations making such choices would benefit from a database system that better matches the amount of data they store and retrieve in practice. Other organizations may also resort to a hybrid mix of database solutions, as opposed to an absolute commitment to one solution model. Additionally, there is a significant change in the way web applications handle data over the past decade due to the increased amount of data and users. Meaning, performance and scalability are becoming a challenge for relational databases than ever. While the SQL databases provide consistency, they are not improved or optimized for best performance in applications that frequently store and process massive data. Fortunately, most NoSQL databases address such challenges by providing new features that boost reliability and scalability. However, the implementation of NoSQL databases under various changes, and their performances may vary. NoSQL database is a better addition or complement to the traditional SQL database standards, although NoSQL databases offer a specialized solution since they work with limited applications. Nevertheless, SQL databases are not becoming obsolete any time soon despite the emergence of NoSQL databases, which provide a different and exciting perspective

of data. It is, therefore, essential to remember that it is the requirements that drive data models, which in turns, influence the choice between SQL and NoSQL databases.

References

1. Bjeladinovic S. A fresh approach for hybrid SQL/NoSQL database design based on data structuredness. Enterprise Information Systems. 2018:1-19.

2. Kanwar R, Trivedi P, Singh K. NoSQL, A solution for distributed database management system. International Journal of Computer Applications. 2013;67(2):6-9.

3. Niyizamwiyitira C, Lundberg L. Performance evaluation of SQL and NoSQL database management systems in a cluster. International Journal of Database Management Systems. 2017;9(6):01-24.

4. Rahman R, Tomar D. Scalable security analytics framework using NoSQL database. International Journal of Database Theory and Application. 2017;10(11):27-46.

5. Rodriguez J, Malgapo A, Quick J, Huang C. Distributed architecture of mobile GIS application using NoSQL database. International Journal of Information and Electronics Engineering. 2017;7(6):156-160.

6. Sattar A, Lorenzen T, Nallamaddi K. Incorporating NoSQL into a database course. ACM Inroads. 2011;4(2):50.

7. Severance C., Fong E. Creating the SQL database standards. Computer. 2014;47(8):7-8.

8. Mpinda SAT, Bungama PA, Maschietto LG. From relational database to column-oriented nosql database: migration process. International Journal of Engineering Research. 2015;4(05).

9. Vathy-Fogarassy Á, Hugyák T. Uniform data access platform for SQL and NoSQL database systems. Information Systems. 2017;69:93-105.

10. Zhang Q. SQL Optimization based on oracle database. Energy Procedia. 2011;11:486-492.