A Comparison between SQL & NoSQL Databases

Hanif I. Lumsden

University of Maryland Global Campus

ITEC 626

Dr. Scoggins

March 9, 2021

Introduction

Relational databases (RDB) and structured query language (SQL) were a solution for companies to deal with database management for 40 years until their limit was reached at the turn of the century (Gianina, 2020). The internet post-dotcom recession produced large volumes of data at tremendous speeds and variety (Caffrey, 2019; Gianina, 2020). However, RDBs are not highly scalable and available and cannot efficiently manage big data. Hence, companies developed a database management system that is non-relational, distributed, open-source, and horizontally scalable for structured, schema-less, and heterogeneous data sets (Gianina, 2020). For relational database management systems (RDBMS), DB developers know the exact schema of the DB, the roles of the entities in a query, and the JOIN paths (Li, Pan, & Jagadish, 2014). Not only SQL (NoSQL) DBs are different across different access methods and languages as a function of their data store (Gianina, 2020; Stonebraker, 2010; Schreiner, Duarte, Dos Santos Mello, 2019). This paper highlights the differences and advantages between SQL & NoSQL and asserts that their benefit is relative.

Open Source and Closed Source Juxtaposition

A considerable difference between SQL and NoSQL is that most RDBMSs are closed-source software; however, NoSQL DBs are open-source software (OSS). Users cannot modify CSS code like OSS ("Open source database," n.d.). Lack of ability to change CSS code poses a problem when there exist glitches in the software. One would have to wait for a software update by the software creator(s) rather than changing the code themselves. In addition, CSS databases often come with expensive licenses.

The OSS source code is publicly available for use, code altering, and freely distributable (Horstmann, 2005). NoSQL databases are OSS, so users can alter the code to meet a business's

significant data needs. This shift for NoSQL databases to be open-source fits historically with the post-dotcom recession internet. It is less expensive and beneficial for companies to utilize a freely distributable database system to fit OSS DB license agreements. Below is a Venn diagram of examples of CSS & OSS DB using the Canva web application.

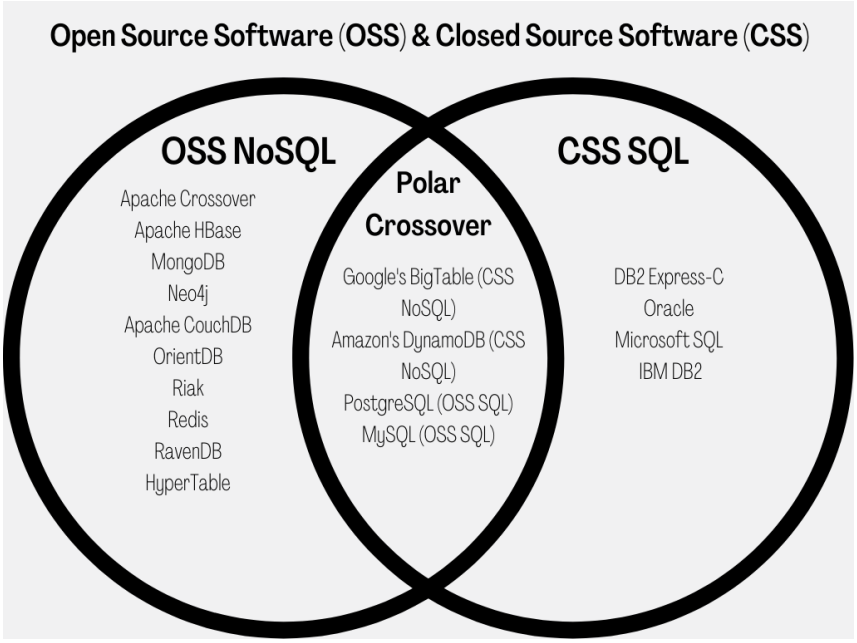

Figure 1:A Venn diagram of OSS and CSS database software. The left circle is open-source NoSQL software; the right circle is closed-source SQL software. The "Polar Crossover" indicates database software that is opposite of the norm, meaning a closed NoSQL or an open SQL software ("21of the best", 2018; Finley, 2015; "Top 10 open source", 2020).

It is important to note that RDB software like MySQL or PostgreSQL is the most popular because of its availability (Finley, 2015). Companies generally produce CSS to fit their needs, but NoSQL is OSS conventionally. A downside to OSS is the lack of support and influx of options to choose from (Krstić & Krstić, 2018).

Different Data Stores

SQL has two data stores: the relational database (RDB) and online analytical processing. (OLAP). RDB consists of a rigid data schema, support for JOINs, non-standard inquiry, complex queries, and ACID transactions (Gianina, 2020). Data is represented in tabular rows consisting of a table that stores records and columns composed of an entity's attributes ("What are relational databases," n.d.). The following data store, OLAP, stores business intelligence (BI) databases for complex analysis. OLAP is enabled when a schema is arranged such that dimension tables surround the innermost fact table. (Hamel, 2005). OLAP databases are optimized for intensive read and simple write tasks designed to extract BI from the data in an efficient manner (ZoinerTejada, n.d.). These data stores differ drastically from NoSQL DBs, which are partitioned into four classifications: key-value stores (KVS), document stores (DS), graph stores (GS), and wide-column stores (WCS) (Gianina, 2020). NoSQL offers a more flexible data model.

In KVS DBs, data is loaded into the memory and stored as an alphanumeric identifier pair that acts as an associated value in the standalone hash tables (Gianina, 2020). This database's values can be strings, integers, arrays, or an object (Gianina, 2020). Searches can be initialized via an exact key. This data store increases SQL queries' execution speed, which would otherwise take a long time. In addition, a flexible layout provides add/remove attributes at runtime without operation interruption (Gianina, 2020). KVS is used for single-key apps to access data, such as online shopping carts, user account profiles, and web session information (Gianina, 2020).

DS DB allows data types and a search to return data using attribute names and values (Gianina, 2020). Data is stored in a single document, with some DSs allowing an additional level of doc aggregation, collection, or bucket aggregation, to store document sets containing the same category of information (Gianina, 2020). RDBs have similar table properties to collection aggregation in that each row is a keyed document, but the structure differs. Entities such as resources, replication, persistence, and security are managed better with collection aggregation. Extensible Markup Language DS is used in content management systems, and JavaScript Object Notation DS is used in applications such as social media platforms (Gianina, 2020).

Graph theory has a strong foundation in RDBs and is the following data store type (Gianina, 2020). GS outdoes SQL capabilities in that it is efficient in data management due to eliminating the expensive JOINs through graph traversals. Stored objects are represented as key-value pairs or documents in nodes called ventrices with relationships between them called edges (Gianina, 2020). People or companies can be defined as nodes in this object-oriented programming. GSs are key when relationship analysis between objects is as critical as the objects themselves (Gianina, 2020).

Data is sorted together on a disk, stored, and processed at the column level for wide-column stores (CS). Data aggregated queries thrive better under this data store as the aggregated values are stored in the same disk blocks (Gianina, 2020). Columns can be grouped into families improving data organization. During runtime, users can add rows and columns at will as long as column families are predefined (Gianina, 2020). Wide-column stores support substantial amounts of data (Krstić & Krstić, 2018). There can be a high level of redundancy for CS, and there is no mechanism for preserving integrity (Krstić & Krstić, 2018). CS is suitable for analytical and business intelligence applications (Gianina, 2020).

Transactions: ACID/BASE/CAP & OLTP

Work performed against the database is known as a transaction. RDB follows what is known as the following properties: Atomicity, Consistency, Isolation, and Durability (ACID). The transaction process cannot have indivisible units, meaning all work must be processed in full or not at all (Krstić & Krstić, 2018); Furthermore, data cannot be inconsistent, operations must be mutually isolated, and all work becomes permanent (Krstić & Krstić, 2018). RDBs are built for online transaction processing (OLTP), where valuable records are entered one at a time. OLAP method is utilized for OLTP to reduce analysis costs as OLTP does not have built-in analytical capabilities (ZoinerTejada, n.d.).

NoSQL DBs do not follow the ACID properties as it is unsuitable. Big data applications, like social networks and healthcare databases, do not respect ACID properties. Instead, the Basic Availability, Soft State, and Eventual Consistency (BASE) properties subscribed to the Consistency, Availability, & Partition Tolerance (CAP) theorem is followed (Krstić & Krstić, 2018; Gianina, 2020). NoSQL systems do not require consistency or accuracy to carry out transactions. Inaccuracies are allowed for a short time, and users can manipulate data to reduce the consumption of resources (Krstić & Krstić, 2018). Post-transaction, the system eventually becomes consistent. Following the CAP theorem, in NoSQL, the nodes have identical copies of the data available for the transaction. NoSQL will process requests for information successfully. The system will continuously operate when the link is broken between nodes (Krstić & Krstić, 2018). Only two of the three CAP theorem properties can be guaranteed simultaneously.

SQL & NoSQL Application in Business & It's Future

Businesses must decide which database system fits their needs as a company. The user must consider the storage type, parallel execution, replication, implementation, and learning curve (Krstić & Krstić, 2018). Different data stores are suitable for various data storage purposes, as displayed in the graphic below:

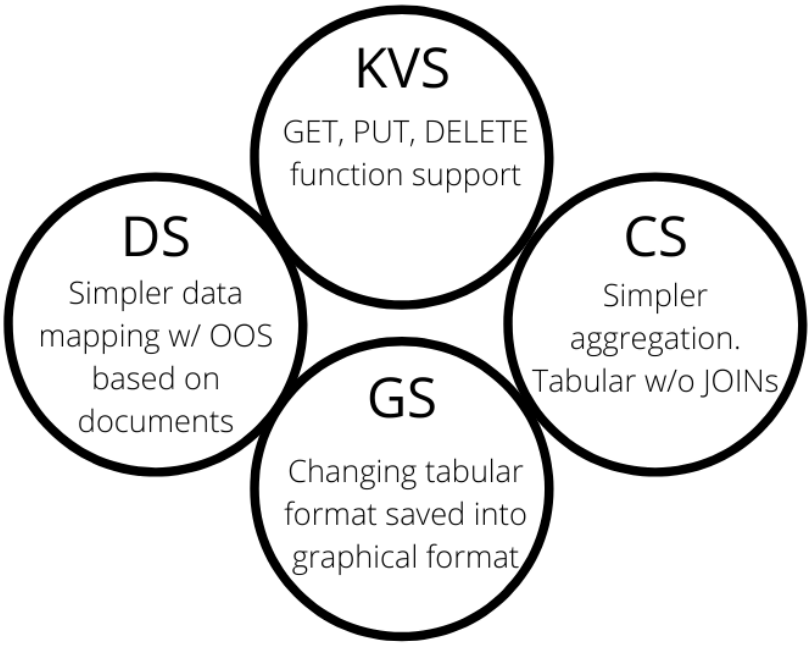

Figure 2: Criteria for selecting a NoSQL database (Krstić & Krstić, 2018).

Parallel execution (PE) defines how two users can modify the same information simultaneously. Typically, a user lose**s** access while another user continues editing (Krstić & Krstić, 2018). Database systems PEs include a lock, preventing more than one user from editing the document. The multi-version concurrency control (MVCC) of PE offers **a** readable overview of the base. Smooth transactions are completed under MVCC. This differs from the lock function in that multiple users can change the document, leading to conflicting versions (Krstić & Krstić, 2018). Reliable transactions are completed under ACID. SQL should be chosen if reliable transactions are needed (Krstić & Krstić, 2018).

Synchronization of backups is another factor in choosing a database system. In the expensive synchronous DB, consistency is ensured; however, for asynchronous DB, inconsistencies are produced due to the DB updating without another DB response. The next factor is implementation, which determines the speed of DB processing. NoSQL DBs "written in low-level programming languages are… the fastest" (Krstić & Krstić, 2018, p. 12), while relational databases are generally higher in programming level and are slowing in processing speed. The last factor is the learning curve. Organizations often implement DaaS for their relational data; However, NoSQL and SQL have little to no support; methodology and syntax differ depending on the management system (Schreiner, Duarte, and Dos Santos Mello, 2019).

Generally, NoSQL DB is extensive due to the stated differences (Schreiner, Duarte, and Dos Santos Mello, 2019). Choosing SQL or NoSQL depends on scalability, speed, and performance ("Top 10 open-source", 2020). It boils down to what the business need is. Below is a table comparing a few SQL & NoSQL applications as listed in **Fig**. **1** and then some more:

	Relational Database	Online Analytical Processing	Key-Value Store	Document Store	Graph Store	Column Store
Example of DBMS Application	Oracle, DBS Express-C, Microsoft SQL, IBM DB2, Postgre SQL, MySQL	Azure SQL Database, Microsoft SQL Server, Jedox	Riak, Azure Cosmos, Redis, Amazon's DynamoDB	Amazon's DynamoDB, Apache CouchDB	Neo4j, OrientDB, ArangoDB, JanusGraph,	HBase, Azure Cosmose, Datastax Enterprise
Data Structure	Tabular w/ JOIN support	OLAP Cube	Key-Value Pairs	XML/JSON Documents w/ JOIN support	Ventrices & Edges w/ JOIN support	Tabular & Column Pairs
Schema	Rigid	Cube/Star/Snowflake	Schema-Less	Loose	Loose	Loose
Query	Ad-Hoc	Ad-Hoc	Deterministic	Ad-Hoc	Ad-Hoc	Ad-Hoc
Row Aggregation	Available	Available	Not Possible	Not Possible	Not Possible	Available
Scalability	Vertical	Horizontal	Horizontal	Horizontal	Horizontal	Horizontal

Table 1: A table with details about the data structure, schema, query type, aggregation, and scalability of different data stores (Mihai, 2020; Mukherjee, 2019; ”Top 10 open-source”, 2020; ZoinerTejada, n.d.).

Conclusion

SQL and NoSQL databases serve specific purposes and have their strengths and weaknesses. NoSQL was created to expand upon SQL's systems, but modernity does not translate to better quality, as it is a relative scenario. It may be more efficient for a company, perhaps a smaller one, to choose a SQL DB, primarily when the said entity does not deal directly with big data, which NoSQL is innovative in handling. Therefore, one must analyze both database systems' relative advantages or disadvantages for their contextual situation. The reality is that NoSQL has much more room to grow than its counterpart. Data is only growing more with each passing second. Big data entails NoSQL deployment and innovation to increase, as it is a newer database system with much room for expansion.

References

21 of the best free to download closed-source applications. (2018, February 26). LinuxLinks. https://www.linuxlinks.com/closed-source/

Caffrey, C. (2019). Dot-com bubble. Salem Press Encyclopedia.

Finley, K. (2015, January 7). Open source databases keep chipping away at Oracle's empire. Wired. https://www.wired.com/2015/01/open-source-database/

Gianina, M. (2020). A comparison between relational databases and NoSQL databases. Annals of Dunarea de Jos University of Galati Fascicle | Economics and Applied Informatics, XXVI(3), 38-42. https://doi.org/10.35219/eai15840409134

Hamel, L. (2005). A Brief Tutorial on Database Queries, Data Mining, and OLAP. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.120.9494&rep=rep1&type=p

Horstmann, J. (2005). Migration to Open Source Databases. Computation and Information Structures (CIS).

Krstić, L., & Krstić, M. (2018). Testing the performance of NoSQL databases via the database benchmark tool. Vojnotehnicki glasnik, 66(3), 614-639. https://doi.org/10.5937/vojtehg66-15928

Li, F., Pan, T., & Jagadish, H. V. (2014). Schema-free SQL. Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. https://doi.org/10.1145/2588555.2588571

Mukherjee, S. (2019). The battle between NoSQL databases and RDBMS. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3393986

Open source database. (n.d.). Accelerated Analytics Platform | OmniSci. https://www.omnisci.com/technical-glossary/open-source-database

Schreiner, G. A., Duarte, D., & Dos Santos Mello, R. (2019). Bringing SQL databases to key-based NoSQL databases: A canonical approach. Computing, 102(1), 221-246. https://doi.org/10.1007/s00607-019-00736-1

Stonebraker, M. (2010). SQL databases v. NoSQL databases. Communications of the ACM, 53(4), 10–11. https://doi-org.ezproxy.umgc.edu/10.1145/1721654.1721659

Top 10 open-source NoSQL databases in 2020. (2020, September 15). GeeksforGeeks. https://www.geeksforgeeks.org/top-10-open-source-nosql-databases-in-2020/

What are relational databases. (n.d.). C# Corner - Community of Software and Data Developers. https://www.c-sharpcorner.com/article/what-is-a-relational-database/

ZoinerTejada. (n.d.). Online analytical processing (OLAP). Developer tools, technical documentation and coding examples | Microsoft Docs. https://docs.microsoft.com/en-us/azure/architecture/data-guide/relational-data/online-analytical-processing