SQL VS NoSQL

[Mohamed Atef Elsafty]

Introduction

SQL is a decades-old method for accessing relational databases, and most who work with databases are familiar with it. As unstructured data, amounts of storage and processing power and types of analytics have changed over the years, however, we've seen different database technologies become available that are a better fit for newer types of use cases. These databases are commonly called NoSQL.

What is a SQL database?

SQL, which stands for “Structured Query Language,” is the programming language that's been widely used in managing data in relational database management systems (RDBMS) since the 1970s. In the early years, when storage was expensive, SQL databases focused on reducing data duplication.

Fast-forward to today, and SQL is still widely used for querying relational databases, where data is stored in rows and tables that are linked in various ways. One table record may link to one other or to many others, or many table records may be related to many records in another table. These relational databases, which offer fast data storage and recovery, can handle great amounts of data and complex SQL queries.

What is a NoSQL database?

NoSQL is a non-relational database, meaning it allows different structures than a SQL database (not rows and columns) and more flexibility to use a format that best fits the data. The term “NoSQL” was not coined until the early 2000s. It doesn't mean the systems don't use SQL, as NoSQL databases do sometimes support some SQL commands. More accurately, “NoSQL” is sometimes defined as “not only SQL.”

When to use SQL

SQL is a good choice when working with related data. Relational databases are efficient, flexible and easily accessed by any application. A benefit of a relational database is that when one user updates a specific record, every instance of the database automatically refreshes, and that information is provided in real-time.

When to use NoSQL

While SQL is valued for ensuring data validity, NoSQL is good when it’s more important that the availability of big data is fast. It’s also a good choice when a company will need to scale because of changing requirements. NoSQL is easy-to-use, flexible and offers high performance.

[Youssef Ashraf Sabry]

Introduction

Databases are used everywhere; they are the backbone of any application that needs to store and retrieve data. However, different database technologies exist for other cases and knowing which to choose is vital to the success of an application. SQL and NoSQL are among the most popular database technologies for different reasons.

SQL

SQL database is a relational tabular database with a fixed schema and uses Structured Query Language to communicate with the database. The data is spread across multiple tables, which are connected using a common field known as the *primary* key. SQL is used to manage data that have structure and relations.

NoSQL

NoSQL database is a non-relational database and does not require a schema. There are four main types of NoSQL databases: document databases, key-value databases, wide-column databases, and graph databases. While each is used for a specific purpose, document databases are the most common.

Major Differences between SQL and NoSQL

There are numerous differences between SQL and NoSQL, which have to be taken into account when deciding what might be the most suitable database management system. These include differences in:

1. Language
2. Structure
3. Scalability
4. Community

Language

The significant difference between SQL and NoSQL is that they are different languages. SQL databases utilize just one query question language. NoSQL databases use one-of-a-kind and non-general dialects.

SQL is the most standardized data information storage language. This can be helpful on the grounds that it creates consistency and accessibility. However this makes SQL stricter than NoSQL.

Structure

SQL databases are relational because their storage systems use tables and relational algebra to process data information. This is a good fit for data that have a structure. The schema has to be predetermined and structured.

NoSQL databases are non-relational databases. These include document, wide-column, key-value, and graph databases. This structure allows for more flexibility than SQL. It also allows the schema to be updated regularly without any issues.

Scalability

SQL databases are mostly vertically scalable. Therefore, adding power to the database means improving the infrastructure and expanding the server's RAM, CPU, or SSD. On the other hand, NoSQL databases are horizontally scalable. Therefore, organizations use sharding and adding servers to expand data storage capabilities. Sharding takes enormous informational indexes and isolates them into little pieces. These little pieces can be spread out across various servers.

Community

SQL has a lot more developed and a better community than NoSQL. There are more forums where developers can share knowledge and discuss issues. However, NoSQL's community is catching up due to its widespread usage.

[Yousef Alaa Adel]

Introduction

We all hear about the term SQL and NOSQL in the field of programming but what is the difference? Let’s see.
The first SQL language was developed by IBM researchers Raymound Boyce and Donald Chamberlin in 1970 named as SEQUEL, on the other side the term NOSQL first used 1998 by Carlo Strozzi while naming his lightweight open-source relational database that don’t use structured query language**(SQL)** ,and then the name came up again in 2009 when it used to describe non-relational database by Eric Evans and Johan Oskarsson.

Why NOSQL invented

So you ask yourself right now if SQL is that old and doing the job till today why they invented non-relational databases anyway. In the beginning at least non-relational databases (NoSQL) were developed as a response to web data, the need for processing unstructured data, and the need for faster processing.

Main differences

Talking about SQL and NOSQL there is five practical differences:

1: Language **2:** scalability
3: structure **4:** properties
5:support

Language

As we mentioned SQL has been around for about 50 years, so it's recognizable, documented, and widely used. Its safe and versatile language ,and well suited for complex queries. However SQL restricts users to working with predefined schemas and more care must be taken to organize and understand the data before it is used.
On the other hand the dynamic schema of NoSQL databases allow representation of alternative structures, often alongside each other, encouraging greater flexibility; giving greater freedom when adding new

attributes or fields. However, as a group, NoSQL languages lack the standard interface which SQL provides, so more complex queries can be difficult to execute.

Scalability

Most SQL databases can be scaled vertically, by increasing the processing power of existing hardware.

NoSQL databases use a master-slave architecture which scales better horizontally, with additional servers or nodes.

Structure

SQL databases schemata always represent relational, tabular data, with rules about consistency and integrity. They contain tables with columns and rows, and keys have constrained logical relationships.

NoSQL databases need not to stick to this format, but generally fit into one of four broad categories:

Column-oriented: databases transpose row-oriented RDBMSs, allowing efficient storage of high-dimensional data and individual records with varying attributes.

Key-Value: stores are dictionaries which access diverse objects with a key unique to each.

Document: stores hold semi-structured data: objects which contain all of their own relevant information, and which can be completely different from each other.

Graph: databases add the concept of relationships to documents, allowing rapid traversal of greatly connected data sets.

Properties

At a high level, SQl and NoSQL comply with separate rules for resolving transactions. RDBMs must exhibit four “ACID” props:

- **Atomicity** means all transactions must succeed or fail completely. They cannot be partially-complete, even in the case of system failure.
- **Consistency** means that at each step the database follows invariants: rules which validate and prevent corruption.
- **Isolation** prevents concurrent transactions from affecting each other. Transactions must result in the same final state as if they were run sequentially, even if they were run in parallel.

- **Durability** makes transactions final. Even system failure cannot roll-back the effects of a successful transaction.

NoSQL technologies adhare to the “CAP” theorem, which says that in any distributed database, only two of the following properties can be guaranteed at once:

- **Consistency:** Every request receives the most recent result, or an error.
- **Availability:** Every request has a non-error result, regardless of how recent that result is.
- **Partition tolerance:** Any delays or losses between nodes will not interrupt the system’s operation.

Support

SQL databases represent massive communities, stable codebases, and proven standards.

NoSQL technologies are being adopted quickly, but communities remain smaller and more fractured. However, many SQL languages are proprietary or associated with large single-vendors, while NoSQL benefit from open systems and concerted commitment to onboarding users.