Initial Post: Shan Swanlow

Although SQL has proven to be reliable, modern software products encounter challenges with scaling it to meet large user bases. SQL databases implement ACID principles, which guarantee that database operations behave consistently and data integrity is maintained. Although a good approach for specific scenarios, it comes at the cost of efficiency due to the safeguards applied to implement the principles, such as database locking (Callison, N.D.). For certain use cases, this tradeoff is not beneficial, as data integrity is not a strict requirement. Rather, scalability may be a larger concern, and for this reason, NoSQL is used in these cases. NoSQL databases implement the opposite of ACID, which is known as BASE. BASE principles make it possible to scale databases for large numbers of users, and have added resilience due to this, however, this comes at the cost of data consistency and integrity. Databases built on BASE guarantee that updates will apply fully after some time has passed, and not immediately, as opposed to ACID databases, which guarantee immediate updates (Neo4j, 2018).

A form of a NoSQL database- a graph database- has experienced an increase in adoption because of the shortcomings that traditional relational databases have when it comes to examining the relationships between entries. Relational databases do not directly store relationships between columns, instead, these relationships often have to be "reconstructed" by performing joins on foreign keys (Neo4j, 2021). This makes it harder to analyze data in the context of its relationship with other data entries; for example, joins and nested queries would first need to be made in order to establish relationships in such a way that SQL can analyze it. Graph databases make this type of task easier, because relationships are explicitly included in each entry and therefore there is no need to write complex join statements- this approach also has the benefit of increased performance at query time (Oracle, N.D.).

References

Callison, J. (N.D.) Database Locking: What it is, Why it Matters and What to do About it. Available from: https://www.methodsandtools.com/archive/archive.php?id=83 [Accessed 5 July 2021].

Oracle. (N.D.) Graph database defined. Available from: https://www.oracle.com/za/big-data/what-is-graph-database/ [Accessed 4 July 2021].

Neo4j. (2021) Concepts: Relational to Graph. Available from: https://neo4j.com/developer/graph-db-vs-rdbms/ [Accessed 5 July 2021].

Sasaki, B. (2018) Graph Databases for Beginners: ACID vs. BASE Explained. Available from: https://neo4j.com/blog/acid-vs-base-consistency-models-explained/ [Accessed 5 July 2021].

Reply: Sergio Rafael Zavarce Caldera

Hi Shan, I found your post very interesting. There is still a long debate about when to implement NoSQL technologies over relational databases. There is an increase adoption of NoSQL databases by cloud providers to use them as cache services, logs, and other fast-dependant storage services. It can also be argued than some database systems can offer the same speed at the cost of reliability, same as NoSQL databases, but in these frameworks the user can select how to approach the consistency in data storage. In the future this might dilute the speed advantage of NoSQL technologies over relational databases.

I also found the graph database description instructive. For someone who uses relational databases in a daily base, trying to get the best of key relationships, it is very interesting to know that this relationships can be explicitly included in each entry.

Peer Response: Shan Swanlow

Hi Sergio,

Thanks for the insights. NoSQL is still a fairly new technology and based on some case studies I've seen on highscalability.com, a lot of large-scale tech companies have used a hybrid of SQL and NoSQL. I didn't know that SQL databases are now supporting different consistency models, I'll definitely be looking into that. As you say, in the future, this may reduce the popularity of NoSQL, and that might mean that NoSQL would only need to be used if flexible schema are required for a project.

What has your experience been with updating and scaling relational databases? At my current place of work, we initially used PostgreSQL for device logs, and we recently switched to MongoDB for scalability, however, one challenge we've come across is that the MongoDB cluster is consistently under high CPU loads due to the amount of write operations. There is a cost associated with this, so we're working on ways of optimizing that.

Reply: Sergio Rafael Zavarce Caldera

Hi Shan,

In the last few years, I have been working for consultancy companies providing services for big customers. The projects I have been involved with are designed over Oracle or MSSQL, and they have not considered changing to a NoSQL technology. They use NoSQL only for logs or documents, such as elasticsearch, but even changing many services to the cloud, they keep on-premises relational databases, and when they feel there could be a bottleneck, they chose queues to persist data in relational databases if necessary. I would like to participate in a migration project to NoSQL but I am pretty sure it must be demanding.

Peer Response: Shan Swanlow

Hi Sergio,

Thanks for your response.

I think you're right about the difficulty involved in migrating to NoSQL- it can be a challenge if the data is complex. A MongoDB case study (2015) mentioned how migrating to NoSQL from SQL was a challenge because the company in question had to redesign their data structures and relationships between data, since everything was made in a relational way. They also encountered some issues with how unstructured schema can become extremely large, so it took a few tries to find the best document structure to use. From personal experience and from what I could understand about this case study, I think that the difficulty appears if you're migrating a database is highly normalized and has many such tables. When I worked on migrated data that wasn't complex (i.e. log data), migration was fairly straightforward.

References

MongoDB. (2015) Migration from SQL to MongoDB - A Case Study at TheKnot.com. Available from: https://www.mongodb.com/presentations/migration-from-sql-to-mongodb-a-case-study-at-theknot-com [Accessed 26 July 2021].