SQL vs NoSQL

LUCA VENTURA | DVP 2
E: LAVENTURA@FULLSAIL.EDU

MOBILE DEVELOPMENT
FULL SAIL UNIVERSITY

RELATIONAL VS NON-RELATIONAL

TWO WAYS TO STORE DATA

Relational (SQL) and Non-Relational (NoSQL) Databases, although different, both serve the same purpose; to store data. One is not meant as a replacement for the other, but an alternative. The main difference between the two is in the way they store data. SQL Databases are stricter in nature, their schemas are fixed, meaning that the columns must be added before data entr and each row must contain data for each column. NoSQL Databases on the other hand are dynamic, information can be added whenever desired and each row doesn't have to contain data for each column.

Relational databases utilize Structured Query Language (SQL) to create strings that are then parsed by the database. Two of the most commonly used operations in SQL queries are select and join. Select simply selects something based on your string, you can be as specific (example: WHERE and AND operators) or as vague (* to select everything in a table) as you would like. The problem with using queries is that your database is then open to query injection attacks. Another problem comes from joining large tables which ultimately hurts performance, another aspect NoSQL has an advantage over relational databases. What relational databases lack in performance and scalability they make up for in features and functionality. NoSQL databases come with a trade off of reduced functionality in the favorability of speed.

NoSQL stores data in documents in JSON-like field-value pairs compared to relational databases' fixed tables. Relational databases contain tables with data fitted into predefined categories, each table contains one or more data categories in columns. A big advantage NoSQL has over relational databases is NoSQL's scalability. In relational databases, scaling is done vertically meaning that the more data you have the bigger the server. NoSQL scales horizontally across multiple servers making it cheaper and more cost effective.

THREE NOSQL TWITTER FEATURES

HOW-TO, PROS AND CONS

Twitter and other social media sites have millions of users that generate terabytes upon terabytes of content on a daily basis, making them the perfect candidate for NoSQL databases scalability.

Tweets: Every time the user sends a tweet it is stored in Twitter's databases, and given how many times users tweet per day a NoSQL database would be ideal as more and more tweets are published.

Hashtags: Users can search Twitter based on hashtags, meaning that every tweet containing that hashtag is stored in a database. Again this database needs to be ever growing and is perfect for NoSQL's scalability.

Retweets: Similar to normal tweets, this list of growing retweets will need to scale accordingly.

All three of these Twitter features share the same advantage when using a NoSQL database; scalability. Every new tweet, retweet and hashtag can be added to a NoSQL database and scale without any issues. If this were done on a relational database that individual server would reach it's maximum capacity awfully quick and need to be upgraded and upgraded. One con of the tweets, retweets and hashtags being spread out amongst a cluster of NoSQL databases could be a slower search time since multiple databases need to be searched instead of just one large table.

FACEBOOK

ONE RELATIONAL FACEBOOK FEATURE

HOW-TO, PROS AND CONS

One feature of Facebook that would be best suited for a relational database instead of a NoSQL database would be predefined user selections such as Relationship Status. There are a fixed number of statuses that can be set for a user's relationship status and that list will most likely not grow. Since relational databases have fixed schemas this feature of Facebook would fit perfectly into a relational database instead of a NoSQL database. I cannot see any cons of using a relational database for this feature as the options to choose from will most likely not grow and even if they did it would be very minute making a NoSQL database unnecessary in this scenario.

NOSQL

WEATHER APP

NOSQL SOLUTIONS

To create a solution for a weather app from the list of previously discussed NoSQL Databases, I think the two best databases to use would be MongoDB and Cassandra. The weather is constantly changing, and most weather apps give you the ability to view the weather based on hour, day and week. This includes the temperature highs, lows, etc. All of that means everyday massive amounts of data will be written to a database as this information is updated. In this scenario we could make use of Cassandra's large write volume and write speeds. When it comes to displaying all the information to the user, MongoDB's fast query speeds can be put to use. One aspect of a weather app that would be more suited for a relational database would be locations as those are static and don't change. However, all other aspects will gain from a NoSQL database solution.

FIVE NOSQL DATABASES

PROS AND CONS

Hadoop
Pros:
Open source
No license fees
Cons:
Integrating with other existing databases can be difficult.
Requires knowledge of MapReduce instead of SQL.
Lacks security functionality for enterprise.

Cassandra
Pros: Can handle a very large amount of write volume and do so with quick speeds.
Cons: Performance can be unpredictable at times.

MongoDB
Pros: Fast query speed.
Cons: Cannot use joins like in a relational database.

CouchDB
Pros: Simplicity; can store any JSON data.
Cons: Slow write speeds.

Redis
Pros: Ones of the fastest data stores available.
Cons: Uses lots of RAM and can get costly when more RAM needs to be added.

REFERENCES

SQL vs. NoSQL- What You Need to Know - Dataconomy. (2014). Retrieved June 28, 2016, from http://dataconomy.com/sql-vs-nosql-need-know/

Buckler, C. (2015). SQL vs NoSQL: The Differences. Retrieved June 28, 2016, from http://www.sitepoint.com/sql-vs-nosql-differences/

Top Five NoSQL Databases and When to Use Them. (n.d.). Retrieved June 28, 2016, from http://www.itbusinessedge.com/slideshows/top-five-nosql-databases-and-when-to-use-them.html

List Of NoSQL Databases [currently 225]. (n.d.). Retrieved June 28, 2016, from http://nosql-database.org/

MongoDB vs Cassandra - Rubikloud. (2014). Retrieved June 28, 2016, from http://rubikloud.com/mongodb-vs-cassandra/

What is Apache Hadoop? - JAXenter. (2015). Retrieved June 28, 2016, from https://jaxenter.com/apache-hadoop-112821.html

LUCA VENTURA
E: LAVENTURA@FULLSAIL.EDU