Nataliia Zubareva

Essay on the data model design for a fast-food chain internal system

With the growing popularity of the NoSQL approach to data storage it is easy to choose it over a classic SQL database without giving it much thought, however this might not be the best decision for every business model. This essay aims to decide, whether a NoSQL database is a better choice than its SQL counterpart for a fast-food chain.

Firstly, we should cover the main differences in both ways of storing data. A SQL database consists of tables, which consist of rows and columns. Because of that, it requires a fixed schema, i.e. the columns for each table, which allows less flexibility in data types stored. As well as this, a proper SQL system aims to normalize the data, in order to battle the anomalies that might otherwise occur. In this way, the database consists of a big number of linked tables, which store the values and links to other tables.

The main advantages of normalized data are less redundancy in data, as we don’t store any redundant values apart from the connections between parents and children, which leads to the smaller size of the database; fast access to individual values in the tables and fast writes and updates, because of little to no doubling of information.

However, any complicated query will require to have table joins, which can be rather slow, also the human-friendliness of the database decreases.

The speed of reading can be increased if we introduce additional denormalized tables, however there will be redundancy then, and as a consequence, more space required for the data storage. Therefore, it is worth trying to find the balance between the two characteristics based on the specific requirements of the task.

The NoSQL approach is a novel idea of storing data in ways other than tables, for instance in JSON files (like MongoDB) and collections. This allows more flexibility with the datatypes of the stored data, it is easier to manage and very human-readable, since it contains all the information relevant to the object in the fields. Importantly, this means that any sophisticated business information can be stored without having the trouble of normalizing it, and the human resources required for managing the data are less than in the case of SQL database, which will probably need a manager.

Another advantage, probably the biggest one for growing business systems, is scalability. While in SQL it is required that all data is stored together and the only solution to scaling the database is getting a bigger and a more expensive server, a NoSQL database can be sharded and dispersed somewhere in the cloud, which is easier and cheaper. This is very tempting feature for projects where growth can be expected and the decision which plans for this in advance is preferable.

The disadvantages of a NoSQL approach are the very same things that provide its advantages. In contrast to SQL, which supports ACID principles and transactions, keeping the data valid and consistent, NoSQL does not adhere to these norms, instead it follows the CAP theorem, which results in faster simple reads and writes, but can only guarantee eventual consistency, which might not be ideal in situations where accuracy is required.

Also, the problem of JOINS in SQL is solved in NoSQL, however it also causes the fact that complicated queries are simply impossible (since there is no functionality in the language to do the equivalent of join), or have to be implemented on a different level of system, for example in the app that represents the database for the user.

Another occasional weakness of NoSQL lies in its core idea, the lack of specification, structure and standards. This leads to less compatibility with other systems, as pretty much any NoSQL database is unique and unlike any other.

Finally, the NoSQL approach is new, so not a lot of additional solutions and improvements exist, however this is most definitely something that will change in the nearest future. The popularity of this concept encourages developers to work on adjusted technologies and stimulates the education-sector to supply the market with experts on NoSQL. But it will take time for the technology and solutions to mature and become as polished as alternatives in the SQL paradigm.

All in all, SQL is a time-tested model, which guarantees accuracy and allows more complicated queries, with the cost of the effort spent on creating the normalized and maybe somewhat stiff structure. It takes more time to develop and more effort to scale, also might be rather slow to read access in its fully normalized glory, but reliable and fast for update queries, also a lot of optimization solutions exist for its rare flaws. NoSQL, MongoDB, for example, is easier to develop, get running and maintain, it does not require stiff structure, and is therefore more flexible. However, it does not cover the need for strict consistency, sophisticated read queries, and many useful upgrades for it do not exist yet.

Now, regarding the provided business model of fast-food chain, both of these approaches would be beneficial in their own ways. To begin with, the advantageous characteristics of MongoDB, such as lack of table structure and scalability, are huge reasons to decide in favor of NoSQL. Indeed, the ability to store, for example, receipts of dishes in JSON files rather than in tables seems to be desirable, as it appears to be really challenging to create a logical way to put it into the tables. The easier scalability is probably relevant as well, as, presumably, the fast-food chain can grow in popularity and spread, which would require to scale the data model at some point in the future. Also, a NoSQL solution would probably be more understandable, since the enterprise is not specializing in data management and IT, and MongoDB with its readable JSONs is much more understandable than SQL tables.

On the other hand, SQL approach would be fit to fulfill the needs of an enterprise that deals with physical objects and requires a certain level of accuracy. Concurrency, precision and correct-working transactions are something that is crucial for companies that provide physical goods. The exact quantities of goods stored; items sold etc. are something that cannot be guaranteed by the eventually-consistent and somewhat fuzzy NoSQL models. Also, the business would probably need detailed reports to function well, which can be provided by table joins in SQL, but not by NoSQL, unless the information the query looks for fits nicely into one JSON file, it cannot be done easily. Also, the faster updates are beneficial for the ever-changing data that describes the assets of a lively enterprise. Finally, for SQL exists a wide refined system of solutions and frameworks that can compensate for some of its downsides. There are less issues caused by the approach being new, and the number of experts is bigger, which would make it easier for the fast-food chain to outsource the system management. Yes, there would have to be more initial effort in setting up the scheme for the database, but the result will be more reliable.

In conclusion, I would decide in favor of SQL database for this kind of business, since it is more precise in data storage and representation and ensures correctness, which is crucial for trade. As for the disadvantages of relational approach, several steps could be taken to compensate for them. Firstly, some frameworks for optimization could be used. Secondly and rather creatively, it is also possible to build a complicated system that stores some data normalized and some denormalized, for example differentiating between reads and writes. But all in all, the core should, in my opinion, be in normalized relational form.