SQL vs. NoSQL

Research Report

Marco Kull (3408140)

Fontys University of Applied Sciences

Venlo, January 15, 2019

Contents

1 Introduction

This document is a research report made as part of the module *Software Factory* at Fontys University of Applied Science in the winters semester 2018/19. In this module the students work in groups on a real project for a real customer. The authors task was to create an mobile application which uses a NoSQL database in its backend.

Nearly every modern application has some sort of database to store its data. SQL databases are around for a long time, but NoSQL databases are gaining more ground. This report aims to provide a fair comparison of the two database design approaches.

2 Research Definition

This chapter explains the motivation behind this research and the methods used to conduct the research. After a brief introduction explaining the motivation, the questions are given that this research tries to answer. Afterwards the research method will be described.

2.1 Research Motivation

Like mentioned above this research is part of the Software Factory module given in the seventh semester at the computer science study program at Fonyts University of Applied Science in Venlo. The author was part of a group that developed a extension modules for an already existing mobile application. The application is build with *React Native* and its backend makes use of *MongoDB*, a *NoSQL* database. Since *NoSQL* databases were not covered by any of the authors previous courses taken it seemed an interesting topic to learn about, especially because those kind of databases are widespread these days.

2.2 Research Questions

The following research questions represent the line of march for the execution of the research:

- Which classifications of *NoSQL* databases are there?
- What are the differences of *SQL* and *NoSQL* databases?

2.3 Research Methods

To answer this questions different approaches can be used. For a direct comparison testing different databases and comparing performance differences would have been great, but for a fair comparison this approach would be too resource consuming for being included in the research context. Thus literature research is used to answer the questions.

3 Research Results

This chapter is split in two parts. First the different approaches of the design of different *NoSQL* databases is being looked upon. Secondly a detailed comparison of *SQL* and *NoSQL* database is given.

3.1 *NoSQL* Databases

For a long time *SQL* database have been the only ones on the marked and their steady improvement and optimisation made them even easier and more stable to work with. But with the dawning of the internet the usage of databases got new requirements like dealing with very huge amount of data or a simpler way to query it. This chapter describes the *NoSQL* database designs that are widely used.

3.1.1 Wide Column Store

Wide column stores groups the data of multiple entries by their columns and not by rows like in relational databases. Every entry consist of the name of the column, the data and a timestamp. Related columns for a so called column family which can contain millions of columns. Those column families can be accessed via a specific key.
Wide column stores are very scaleable and have a great reading performance but are inefficient when writing.

3.1.2 Document Store

Unlike traditional databases the data in a document store is not stored in tables but rather in documents. These documents still store their data in key value pairs, but relations between them are not possible. Every document can be accessed through a specific identifier and has its own schema independent of other documents stored in the database.
Document stores are easily scaleable and allow an easy and natural way to store already existing documents, but its lack of a powerful query language can result in more programming work.

3.1.3 Key Value Store

The key value store type of *NoSQL* databases is very similar to rational databases. A specific key points to a value. Every key is only allowed to point at exactly one value. There are on disk and in memory variants of this database type with the latter providing high performance due to the placement in the memory.
Key value stores have efficient data management and are easily scaleable, but because of the simple type schema complex queries are not possible.

3.1.4 Graph Databases

Graph databases organise data in nodes and edges. The nodes represent tuples from rational databases and the edges their relations. The traversal between the nodes is very efficient and thus complex queries have a better performance than on rational databases at the cost of sacrificing a uniformed query language.

3.2 *SQL* vs. *NoSQL*

This chapter draws a comparison between *SQL* and *NoSQL* databases in regard to scaleability, performance, language and consistency.

3.2.1 Scaleability

SQL database are foremost vertically scaleable, which means that performance of a single server can be increased for example by adding more RAM to a machine. *NoSQL* databases on the other hand are horizontally scaleable, which means for example to add more servers to share traffic. In the end this results in a situation that *NoSQL* databases eventually can get larger and more powerful than *SQL* databases.

3.2.2 Performance

Because of the superior scaleability of *NoSQL* databases over *SQL* databases *NoSQL* databases can ultimatly outperform *SQL* databases. This should not suprise much because one reason *NoSQL* databases where invented was because of the need of better performance with huge data sets.

3.2.3 Language

One of the great strengths of a *SQL* database is its unified query language whereas *NoSQL* databases have very different query languages depending on the concrete choice of database implementation.

3.2.4 Consistency

Generally said *SQL* databases enforce consistency more strictly than *NoSQL* databases do - a great part of this is because of the ACID approach. The better performance often correlates with a loss of consistency.

4 Conclusion

Since the amount of data is growing every year *NoSQL* databases have taken a secure seat beside the traditional *SQL* databases. However *SQL* databases will still continue to exist for business purposes that rely strongly on consistency. The key is to understand the different database designs and their advantages and disadvantages so one can select the appropriate database for a specific use case.

5 References

http://nosql-database.org/

https://www.sitepoint.com/sql-vs-nosql-differences/

https://www.bmc.com/blogs/sql-vs-nosql/

https://www.geeksforgeeks.org/difference-between-sql-and-nosql/

https://dbs.uni-leipzig.de/file/seminar_1112_tran_ausarbeitung.pdf

https://medium.com/xplenty-blog/the-sql-vs-nosql-difference-mysql-vs-mongodb-32c9980e67b2