11/4/2021

Trifonov,Dobri D.V.

iNDIVIDUAL PROJECT

Research Document

Suitable Database Architecture

Table of Contents

[What is the most appropriate way to store data inside of my application 2](#_Toc87044093)

[1. What are the structural differences 2](#_Toc87044094)

[2. Advantages and Disatvantages of both databases 3](#_Toc87044095)

[3. Scalability 4](#_Toc87044096)

[Conclusion 5](#_Toc87044097)

[When to choose SQL 5](#_Toc87044098)

[When to choose NoSQL 5](#_Toc87044099)

[Available choices 5](#_Toc87044100)

[My Personal Choice 5](#_Toc87044101)

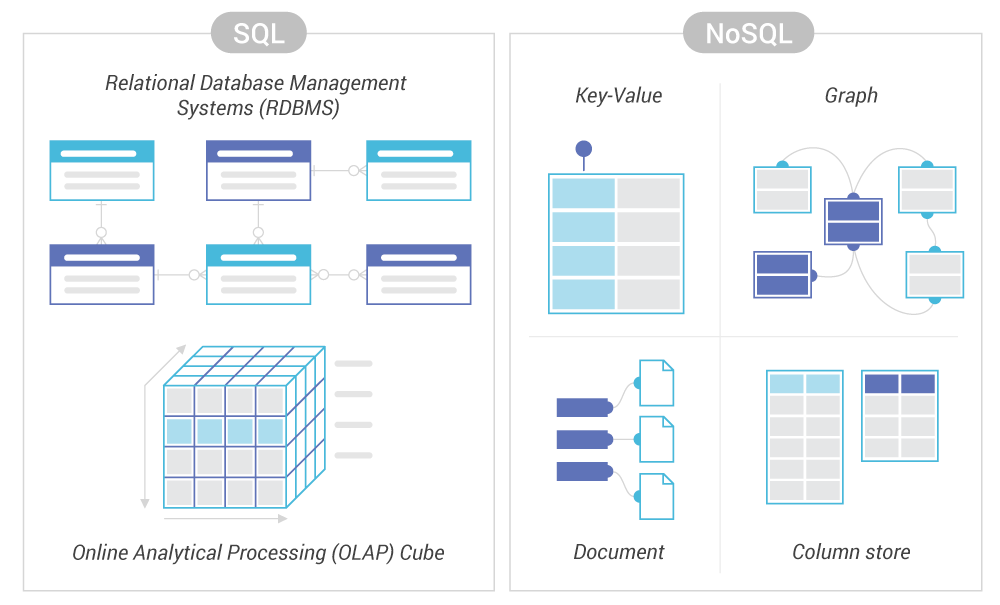
[References 6](#_Toc87044102)

# What is the most appropriate way to store data inside of my application

## What are the structural differences

**SQL** - All SQL database share the same SQL language. Some database have additional features on top of the default ones, but they more or less have the same common functionality. What keeps them all so similar is the structure. The data is layed out in a table with columns and rows, the data is connected with keys (Primary, Foreign Keys, etc). This makes the SQL data structure very relational and have a very rigid structure.

**NoSQL** - databases on the other hand do not have keys that create relationships between data tables. Because of that we lose the convenience of SQL's ability to join tables together using simple queries. The data in a NoSQL database is unstructured, meaning that every data row does not need to match the set columns. This allows for a slightly better flexibility because we can create data without structure, then add to it as we go on with the project. Data can be stored in plethora of ways such as Document, Column, Graph or even Key-Value store.



[Scylladb]. NoSQL vs SQL

* [Key-value store — Stores data with simple indexed keys and values.](#_Turner,_A._(2020,)
* [Wide column store — Uses tables, rows and columns. But the format and naming of the columns can vary in different rows within the same table.](#_Turner,_A._(2020,)
* [Document database — A more complex and structured version of the key-value model, which gives each document its own retrieval key.](#_Turner,_A._(2020,)
* [Graph database — Presents interconnected data as a logical graph.](#_Turner,_A._(2020,)

## Advantages and Disatvantages of both databases

SQL - SQL databases are generally used when we have planned out, structured data and we need to keep that data in strict check. The set columns are unchangeable, tables relate to each other using different keys allowing for easy joins between tables. This allows for the creation of quicker and more complex queries. In addition to that SQL databases also examine higher read speeds than most NoSQL databases. This makes SQL databases ideal for applications that work with complex data and applications that make large read requests for data.

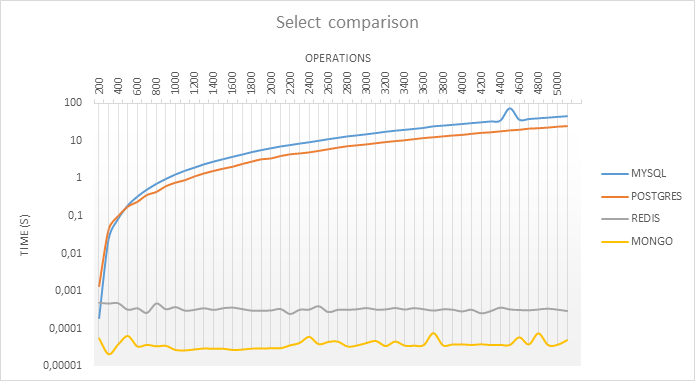
NoSQL - Even though SQL has an advantage in the Read speeds and Data complexity department it cannot compete in the field that NoSQL was designed to be in, writing data. Compared to SQL, NoSQL has an extremely high write speeds contributing to the fact that it does not have as many strict checks as SQL with its structured format. This makes NoSQL the default choice for applications that need to ingest large amounts of data, making it ideal for applications that write in real time. While an advantage this could also be a double edged sword for a developer who doesn't follow a strict data scheme. Since properties can be excluded the data we request may very depending on the place we decide to pull it from.

Table

Description automatically generated

## Scalability

**SQL** – SQL databases in almost all cases scale vertically. That means that to increase the load on a server you need to upgrade the server’s specifications. This could be done by adding more RAM, CPU or SSD capacity. Of course technology has its limits so SQL database will always be bottlenecked by the fastest currently available hardware.

**NoSQL** – NOSQL databases scale horizontally. That means that to upgrade a server you simply need to add more nodes (servers) to it. This makes it great for applications where large amount of users attempt to connect at the same time. It is an infinitely scalable system that not only improves performance, but allows for replica database servers to run in parallel with the original ones. That means that there will be no down time in case a server needs to be swapped out or repaired.

Here we can observe that ,even though MySQL operates faster than Redis, after the 200th request MySQL significantly slows down compared to Redis and MongoDB, which are optimized to handle large amount of requests.

# Conclusion

Between SQL and NoSQL it is impossible to declare one better than the other. They both have their ups and downs and their features vary from database to database. Depending on the application that is being developed one might be considerably better than the other OR there could be almost little to no difference.

### When to choose SQL

* Whenever I high degree of data integrity is required (Financial transactions)
* Complex queries are needed to be performed
* When we do not anticipate large changes
* When we do not expect large volumes of data

### When to choose NoSQL

* When we constantly add new features, functions, data types
* When we have flexible or no predefined schema
* Our data needs to scale up
* We are not concerned about consistency

## Available choices

**MySQL** – One of the most popular SQL databases. It is widely used for its consistency and ability for horizontal partitioning. As a result, it offers high throughput with low latency, making it popular among Web-Scale applications.

**PostgreSQL** – PostgreSQL is known for being one of the most advanced open source databases. It has variety of features and extensions that cut down on developer workload. It too offers horizontal partitioning but also multi model database, meaning it can work with semi-structured, key-value and graph data just like NoSQL could

**MongoDB** – By far the most popular choice when it comes to NoSQL databases. Has many features that make it great for raw data storage. Indexing is one of those big features. MongoDB can not only index rows but also every field, allowing it to rapidly improve the performance of searches within the database.

## My Personal Choice

For my individual project I have decided to use MySQL. The reasoning for that is its more simplistic nature and lower latency compared to PostgreSQL. Even though the performance scales worse than with PostgreSQL I do not expect my application to be in need of hundreds of thousands of operations per second. My application is low scale, containing only small basic information with a planned out data structure. While MongoDB does offer slightly lower latency than MySQL, based on my research, I deemed that data rigidity is more important to me than slightly better performance. So that’s why my choice for the personal project is MySQL.

# References

[Scylladb]. (n.d.). NoSQL vs SQ. https://www.scylladb.com/wp-content/uploads/differences-between-sql-databases-and-nosql-databases.png

Turner, A. (2020, October 27). *ScyllaDB | NoSQL vs SQL*. ScyllaDB. https://www.scylladb.com/resources/nosql-vs-sql/

Olivera, L. (2019, June 5). *Everything you need to know about NoSQL databases*. DEV Community. https://dev.to/lmolivera/everything-you-need-to-know-about-nosql-databases-3o3h

Majerczyk, M. (2021, July 5). *Database Comparison - SQL vs. NoSQL (MySQL vs PostgreSQL vs Redis vs MongoDB)*. Medium. <https://medium.com/profil-software-blog/database-compare-sql-vs-nosql-mysql-vs-postgresql-vs-redis-vs-mongodb-3da5f41c31b5>

*When to Use SQL vs. NoSQL*. (n.d.). INTEGRANT. <https://techblog.integrant.com/when-to-use-sql-vs-nosql>

Taylor, D. (2021, November 1). *What is MongoDB? Introduction, Architecture, Features & Example*. Guru99. https://www.guru99.com/what-is-mongodb.html