Radolina Petrova

Applied research document

Data storage

Table of Contents

[What is the best way to store data in your project? 2](#_Toc116078262)

[Why should we use a database? 2](#_Toc116078263)

[What types of databases are there? 2](#_Toc116078264)

[ Relational databases 2](#_Toc116078265)

[ Document databases 3](#_Toc116078266)

[ Key-value databases 3](#_Toc116078267)

[ Column-oriented databases 3](#_Toc116078268)

[ Graph databases 3](#_Toc116078269)

[ Object-oriented databases 3](#_Toc116078270)

[What type of data do we need to store? 4](#_Toc116078271)

[Conclusion 4](#_Toc116078272)

# What is the best way to store data in your project?

Choosing the best approach for data storage is important for the development of the project and can have a big impact on the efficiency of the system, as well as the processing of the data over time. When deciding about the strategy, developers must take into consideration not only the best storage format that currently meets the requirements of the project, but one, that will also continue to do so in the future.

## Why should we use a database?

In time databases have proven to be one of the most organized ways to store our data. They allow us to store information in different formats and in an easily searchable way. Over time technology has improved to the point, where different types of databases are available, each with its strengths and weaknesses, based on their design.

## What types of databases are there?

In order to achieve the most efficient setup, we need to understand the between the different types of databases and how they work. In time a lot of different kinds have been developed, but in general they are divided in NoSQL and SQL databases. First here are all the widely used sub-types.

### Relational databases

One of the major kinds of databases is the SQL-based **relational** one. This type of databases store and analyse data digitally. They are used to identify relationships between key data points, which is why the work well with structured data. The structure of these databases is defined by the unique instance of data, that each record of each table holds. Namely via this unique identifier, the relationships between the tables can be set. Some of the advantages of this type of database are:

* Easy categorization and storage of the data
* Possibility for extendibility without modifications of the existing application
* Lack of data duplication
* Carrying out of complex queries with SQL (the main language, used in relational db)
* Easy collaboration, due to the possibility for multiple users to have access to the database

However, there are some disadvantages to using a relational database.

* Difficulties with the maintenance because it requires a certain level of planning and lacks flexibility and scalability
* Not optimal for largely qualitative, not easily defined, or dynamic data, because the evolving of the schema takes time
* Lack of scalability

It is not easy to handle relational databases across multiple servers, because the bigger distribution of larger data can affect performance and availability.

### Document databases

Another kind of widely used databases are the NoSQL ones. There are several types of them, including **document database**s, in which the data is stored in JSON, BSON or XML documents. Some of their advantages include:

* Less translation due to the format of storage being as close as possible to data object
* Flexibility, which allows reworking of the document’s structure, so it can best suit the application requirements, that can evolve with time
* Usage of API or query language that allows the execution of CRUD operations on the database

### Key-value databases

The name of the NoSQL-based **key-value databases** speak for themselves, as each item contains keys and values. They resemble the relational databases, but in a more minimalistic version. Some of the advantages of these databases are namely:

* The support for defined or semi-structured data they offer, which is connected to optimization in terms of storage and performance
* Easy access to the keys and the possibility for their simple sorting for efficient iterating over them
* Allowing the definition of multiple keys for access to the same information
* Allowing replication of data, which is very important feature in terms of disaster recovery
* Better scalability and performance in terms of reading and writing data

These also have some disadvantages that include:

* The absence of schema can lead to a messy data model in long term
* Processing of useless for the application code data, since the implementation of the database does not provide any information regarding the actual contains of the value

### Column-oriented databases

Another type is a NoSQL-based **column-oriented database**, where the data is stored as a set of columns. They have these advantages:

* Running analytics on a small number of columns, which is efficient, when it comes to reading time
* Quick aggregation of the value of a certain column

A big disadvantage, however, of these databases is:

* All the columns require multiple write events on a disk

### Graph databases

And finally, there are the NoSQL-based **graph databases**, where each element is stored as a node and the connection between them is stored directly. An advantage of these databases is:

* Easy to capture relationships between the elements, which overcomes the joining of tables, used in SQL

A disadvantage, however, is:

* It cannot be solely relied on graph queries to retrieve the requested data

### Object-oriented databases

The NoSQL-based **object-oriented** databases are also often used because they can work with complex data objects, created in the image of those in the object-oriented programming. They store the data directly on the database server’s disk. Their advantages include:

* Relationships between the objects can also be established and can lead to fast retrieval of data across complex associations
* Simplicity and light weight of the code, because of the similarities to the programming objects
* No retrieval and decomposition of the object or the reverse process, everything is simply stored in the database

However, they come with several disadvantages including:

* The sub-optimal relative performance for simple operations
* Dependency of the query syntax on the language

## What type of data do we need to store?

A very important factor regarding the choice of a suitable database is namely the kind of the data we need to store. In general, the purpose of the database is to store data in all kinds of format, including information, texts, images, file media etc. And between SQL- and NoSQL-based there is always a more suitable type of database for a particular data format and depending on its characteristics. When it comes to graph or hierarchal data NoSQL is preferred, especially if the volume of the data is large, it lacks structure and it rapidly changes over time. On the other hand, SQL-based databases are preferred when consistency in the systems is critical, the data is well structured, and the volume is not that big.

## Conclusion

Overall, when we look at the differences between the SQL- and NoSQL-based databases, each one of them has its advantages and disadvantages and depending on the data, that needs to be saved, there surely is a type, which can meet better the requirements than the other.

One of the main characteristics that differentiates the two types of databases is namely the fact, that most of the SQL-based systems require knowledge of the structure of the data in advance and they cannot process unstructured and unpredictable data. Whereas the other type is designed to be able to work with unstructured and semi-structured data, which is perfect for one that undergoes a lot of changes over time. NoSQL-based databases store the data in form of entities and can also process it in real time. They offer a faster pace of development, easier maintenance of the structure of many data forms, easier support of new application paradigms and support of huge volume of data, which is why they are suitable for a range of use cases.

In conclusion, each database type has its advantages and disadvantages, but depending on the use it has some functionalities to offer in order to achieve a better performance of the application.

# References

<https://www.geeksforgeeks.org/advantages-and-disadvantages-of-sql/>

<https://www.talend.com/resources/sql-vs-nosql/>

<https://www.ntchosting.com/encyclopedia/hosting/database-storage/>

<https://aws.amazon.com/nosql/key-value/>

<https://www.projectpro.io/article/nosql-vs-sql-4-reasons-why-nosql-is-better-for-big-data-applications/86#toc-1>

<https://www.mongodb.com/document-databases>

mongodb.com/databases/what-is-an-object-oriented-database

<https://www.techtarget.com/searchdatamanagement/definition/relational-database>

<https://www.indeed.com/career-advice/career-development/relational-database-schema>

<https://www.mongodb.com/nosql-explained>