Best Practices of MongoDB

In this reading, you'll learn about the definition of MongoDB and describe the best practices for MongoDB.

What is MongoDB?

MongoDB is an open source document-oriented database designed to store large data sets for industries. It enables efficient work with data and falls under not only structured query language (NoSQL) database because data storage and retrieval do not form tables.

MongoDB. Inc., under a server-side public license (SSPL), manages and develops the MongoDB database. It provides driver support for various programming languages, such as C, C++, C #, .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, and Mongoid.

Best practices: MongoDB

MongoDB offers flexibility and scalability to harness its full potential. Let's understand the best practices for optimizing performance and designing efficient schemas in MongoDB.

Data Modelling

MongoDB offers a flexible document-based model for representing complex, hierarchical data structures. Therefore, it is important to thoroughly understand the application's data requirements to design a MongoDB schema. However, various factors such as data relationships, read patterns, and scalability are also important.

Indexing

Indexing the fields that are frequently queried or used for sorting helps to improve query execution times. However, ensuring that the queries support appropriate indexes and avoid creating extra indexes is vital. This helps to impact the right performance and save storage space. Also, keep in mind that indexing is a continuous process; however, queries need to be monitored and updated based on the business requirements.

Aggregation framework

The MongoDB aggregation framework provides powerful tools for performing data transformations, analytics, and computations within the database. Instead of fetching large datasets and processing them in application code, the aggregation pipeline can be used to perform complex operations directly within MongoDB. However, you should familiarize yourself with aggregation operators and stages, such as $match, $group, $project, and $lookup, to manipulate and aggregate data efficiently.

Scaling horizontally

Scaling horizontally is one of the key points of MongoDB. Sharding helps distribute data across servers or shards to handle large volumes of data and high throughput workloads. However, consider the scalability requirements and plans for early sharding when designing MongoDB deployment. Further, select an appropriate shard key to distribute data equally across the shards by regularly monitoring their distribution and performance based on application growth.
Preventing unauthorized access

Databases often contain valuable and sensitive information, such as personal data, financial records, intellectual property, and proprietary business information. To achieve this, follow security best practices such as enabling authentication, using role-based access control (RBAC) to define user permissions, and configuring network encryption with transport layer security/secure sockets layer (TLS/SSL). Regularly update MongoDB to the latest version to patch security vulnerabilities and maintain a robust security posture.

Summary

In this reading, you've learned about what MongoDB is. MongoDB is an open source document-oriented database designed to store large data sets for industries. It supports various programming languages such as C, C++, C#, .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, and Mongoid.

To use the full potential of MongoDB, one should follow several best practices, such as:

- Data Modelling
- Indexing
- Aggregation framework
- Scaling horizontally
- Preventing unauthorized access