MongoDB Features

Learn more about MongoDB Atlas

MongoDB is a scalable, flexible NoSQL document database platform designed to overcome the relational databases approach and the limitations of other NoSQL solutions. MongoDB is well known for its horizontal scaling and load balancing capabilities, which has given application developers an unprecedented level of flexibility and scalability.

MongoDB Atlas is the leading global cloud database service for modern applications. Using Atlas, developers can deploy fully managed cloud databases across AWS, Azure, and Google Cloud. Best-in-class data security and privacy standards practices mean that developers can rest easy knowing that they have instant access to the availability, scalability, and compliance they require for enterprise-level application development.

The MongoDB database platform has been downloaded over 200 million times with over 1.8 million MongoDB University registrations. There are drivers for 10+ languages, with dozens more added by the community. Best of all, MongoDB is completely free to use.

MongoDB provides developers with a number of useful out-of-the-box capabilities, whether you need to run privately on site or in the public cloud.

Let's take a look at some of the most essential technical features of MongoDB.

Document Model

MongoDB has been designed with developer productivity and flexibility in mind. It is a document-oriented database, which means that data is stored as documents, and documents are grouped in collections. The document model is a lot more natural for developers to work with because documents are self-contained and can be treated as objects. This means that developers can focus on the data they need to store and process, rather than worrying about how to split the data across different rigid tables.

Documents are grouped together in collections. However, the documents in a single collection don't necessarily need to have exactly the same set of fields. This is what we call a "flexible schema." This flexibility allows developers to iterate faster and migrate data between different schemas without any downtime. However, if you want to lock down your schema at a certain point, you can do so by applying validation rules to your collections.

Documents in MongoDB are stored in the BSON format, which is a binary-encoded JSON format. This means that the data is stored in a binary format, which is much faster than JSON. This also allows for the storage of binary data, which is useful for storing images, videos, and other binary data. Even though BSON is a binary-encoded format, it's easy to work with it using the MongoDB driver for your programming language.

To learn a lot more about the document model, check out the Document Database article.

Sharding

Sharding is the process of splitting larger datasets across multiple distributed instances, or "shards." When applied to particularly large datasets, sharding helps the database distribute and better execute what might otherwise be problematic and cumbersome queries. Without sharding, scaling a growing web application with millions of daily users is nearly impossible.

Sharding in MongoDB allows for much greater horizontal scalability. Horizontal scaling means that each shard in every cluster houses a portion of the dataset in question, essentially functioning as a separate database. Combining the data of the distributed shards forms a single, comprehensive database much better suited to handling the needs of a popular, growing application with zero downtime.

All client operations in a sharding environment are handled through a lightweight process called `mongos`. The `mongos` can direct queries to the correct shard based on the shard key. Proper sharding also contributes significantly to better load balancing.

Check out the dedicated Database Sharding article to learn more about the different sharding architectures and what problems they solve.

Replication

When your data only resides in a single server, it is exposed to multiple potential points of failure, such as a server crash, service interruptions, or even good old hardware failure. Any of these events would make accessing your data nearly impossible.

Replication allows you to sidestep these vulnerabilities by deploying multiple servers for disaster recovery and backup. Horizontal scaling across multiple servers greatly increases data availability, reliability, and fault tolerance. Potentially, replication can help spread the read load to the secondary members of the replica set with the use of read preference.

In MongoDB, replica sets are employed for this purpose. A primary server or node accepts all write operations and applies those same operations across secondary servers, replicating the data. If the primary server should ever experience a critical failure, any one of the secondary servers can be elected to become the new primary node. And if the former primary node comes back online, it does so as a secondary server for the new primary node.

MongoDB Atlas, MongoDB's DBaaS (Database-as-a-Service) platform, has a minimum of three member replica sets. They can span across multiple regions or even multiple cloud providers of your choice.

Check out the Replication article to learn more about how replication works in MongoDB.

Authentication

Authentication is a critical security feature in MongoDB. Authentication ensures that only authorized users can access the database. Without authentication, anyone can access your data.

MongoDB provides a number of authentication mechanisms for users to access the database. The most common is the Salted Challenge Response Authentication Mechanism (SCRAM), which is the default. When used, SCRAM requires the user to provide an authentication database, username, and password.

To learn more about SCRAM and the other available authentication mechanisms, check out the MongoDB Authentication article.

Database Triggers

Database triggers in MongoDB Atlas are a powerful feature that allow you to execute code when certain events occur in your database. For example, you can use triggers to execute a script when a document is inserted, updated, or deleted. Triggers can also be scheduled to execute at specific times.

MongoDB Atlas allows you to create and manage triggers in a simple, intuitive way. You can control your triggers through the Atlas UI.

Database triggers are a great way to perform audits, ensure data consistency and data integrity, and to perform complex event processing. Check out the dedicated Database Triggers article to learn more about the different types of triggers and how to use them.

Time Series Data

Time series data is most commonly generated by a device, such as a sensor, that records data over time. The data is stored in a collection of documents, each of which contains a timestamp and a value. MongoDB provides a number of features to help you manage time series data.

The native time series collections in MongoDB are designed to be storage-efficient and perform well with sequences of measurements. You have a number of parameters to control the storage of time series data, including the granularity (the time span between measurements) and the expiration threshold of old data.

To learn more about the native time series collections and other MongoDB features that make working with time series data easier, check out the Time Series Data article.

Ad-Hoc Queries

When designing the schema of a database, it is impossible to know in advance all the queries that will be performed by end users. An ad-hoc query is a short-lived command whose value depends on a variable. Each time an ad-hoc query is executed, the result may be different, depending on the variables in question.

Optimizing the way in which ad-hoc queries are handled can make a significant difference at scale, when thousands to millions of variables may need to be considered. This is why

MongoDB, a document-oriented, flexible schema database, stands apart as the cloud database platform of choice for enterprise applications that require real-time analytics. With ad-hoc query support that allows developers to update ad-hoc queries in real time, the improvement in performance can be game-changing.

MongoDB supports field queries, geo queries, and regular expression searches. Queries can return specific fields and also account for user-defined functions. This is made possible with MongoDB indexes, BSON documents, and the MongoDB Query Language (MQL). MongoDB also supports aggregations via the Aggregation Framework.

To learn more about the analytics features of MongoDB, check out the dedicated Real-Time Analytics article.

Indexing

In our experience, the number one issue that many technical support teams fail to address with their users is indexing. Done right, indexes are intended to improve search speed and performance. A failure to properly define appropriate indexes can and usually will lead to a myriad of accessibility issues, such as problems with query execution and load balancing.

Without the right indexes, a database is forced to scan documents one by one to identify the ones that match the query statement. But if an appropriate index exists for each query, user requests can be optimally executed by the server. MongoDB offers a broad range of indexes and features with language-specific sort orders that support complex access patterns to datasets.

Notably, MongoDB indexes can be created on demand to accommodate real-time, ever-changing query patterns and application requirements. They can also be declared on any field within any of your documents, including those nested within arrays.

Check out the Performance Best Practices: Indexing article to learn more about the different types of indexes and how to use them.

Load Balancing

At the end of the day, optimal load balancing remains one of the holy grails of large-scale database management for growing enterprise applications. Properly distributing millions of client requests to hundreds or thousands of servers can lead to a noticeable (and much appreciated) difference in performance.

Fortunately, via horizontal scaling features like replication and sharding, MongoDB supports large-scale load balancing. The platform can handle multiple concurrent read and write requests for the same data with best-in-class concurrency control and locking protocols that ensure data consistency. There's no need to add an external load balancer–MongoDB ensures that each and every user has a consistent view and quality experience with the data they need to access.

If you're curious how load balancing works in a sharded cluster, check out the Sharded Cluster Balancer page in the MongoDB Documentation.

Conclusion

MongoDB is a flexible, document-oriented database platform that is designed to be the cloud database of choice for enterprise applications. MongoDB provides a number of features that make it a great choice for a wide variety of applications.

MongoDB Atlas is MongoDB's DBaaS (Database-as-a-Service) platform offering that provides a fully managed MongoDB cluster with a dedicated MongoDB instance for each user. Atlas has all the features of a MongoDB Enterprise instance, plus the ability to scale horizontally and vertically in a click of a button. You can get started with MongoDB Atlas today by creating a free cluster.

FAQ

What are the features of NoSQL?

NoSQL databases are a new generation of database technologies that are designed to provide a flexible, scalable, and reliable solution for storing and retrieving data. Among the most notable features of NoSQL databases are:

- Horizontal scalability.
- Flexible schema.
- Fast queries.
- Ease of use.

What are the most important features of MongoDB?

MongoDB is a document-oriented NoSQL database platform which provides numerous important features. Among them are:

- The document model with a flexible schema to best store data for your application needs.
- Replication for better data availability and stability.
- Sharding for horizontal scalability.
- Ad-hoc queries for optimized, real-time analytics.

What is MongoDB good for?

MongoDB is a general-purpose document database platform with a wide variety of use cases. Because of the flexibility of the document model and its scale-out architecture, MongoDB is the preferred database for developers across multiple industries. For more information, check out MongoDB Use Cases.