
MONGO DB

➤ What is mongo db:

MongoDB is a popular NoSQL (non-relational) database management system. Unlike traditional relational databases like MySQL or PostgreSQL, MongoDB uses a document-oriented data model, which means data is stored in flexible, JSON-like documents instead of rigid tables with rows and columns.

➤ Why it is used:

- **Flexible Data Model:** MongoDB's document-oriented data model allows for flexible and dynamic schemas. This flexibility is especially useful for applications with evolving data requirements or where the data structure is not predetermined.
- **Scalability:** MongoDB is designed to scale horizontally, allowing it to handle large volumes of data and high throughput by distributing data across multiple servers.
- **High Performance:** MongoDB's architecture and indexing mechanisms contribute to high performance for both read and write operations. It can efficiently handle complex queries and large datasets.
- **Developer Productivity:** MongoDB's JSON-like documents make it easy for developers to work with data, as it closely aligns with modern programming languages and data formats. Developers can store data in its natural form without the need for complex mappings.
- **Rich Query Language:** MongoDB provides a powerful and expressive query language that supports a wide range of operations, including filtering, sorting, aggregating, and geospatial queries.
- **Ad Hoc Queries:** MongoDB supports ad hoc queries, allowing developers to quickly query and analyze data without the need to predefine schema or create complex joins.

-
- **Horizontal Scalability:** MongoDB supports horizontal scaling through sharding, allowing organizations to distribute data across multiple servers to handle growing datasets and traffic.
 - **High Availability:** MongoDB offers features like replica sets, which provide automated failover and data redundancy, ensuring high availability and data durability.
 - **Real-Time Analytics:** MongoDB's aggregation framework enables real-time analytics by allowing developers to perform complex data aggregation operations across large datasets.
 - **Community and Ecosystem:** MongoDB has a vibrant community and extensive ecosystem, with support for various programming languages, frameworks, and tools. This makes it easier for developers to integrate MongoDB into their applications and leverage additional features and functionalities.

➤ **Advantages:**

1. Full cloud-based developer data platform.
2. Flexible document schemas.
3. Widely supported and code-native data access.
4. Change-friendly design.
5. Powerful querying and analytics.
6. Easy horizontal scale-out with sharding.
7. Simple installation.
8. Cost-effective.

➤ **Limitations of mongo db:**

- **Transaction Support:** Historically, MongoDB lacked support for multi-document transactions across multiple collections. Although support for multi-document transactions was introduced in later versions, they are limited to replica sets and not available in sharded clusters.
- **Memory Usage:** MongoDB's memory usage can be relatively high, especially if your dataset exceeds available RAM. This can lead to performance issues if not managed properly.

-
- **Indexing Complexity:** While MongoDB supports indexing, creating and managing indexes requires careful consideration to ensure optimal query performance. Inadequate or improper indexing can lead to slow query performance.
 - **Data Size Limitations:** MongoDB has a maximum document size of 16 MB. While this limit is quite large for most applications, it can be a limitation for certain use cases with extremely large documents.
 - **Atomicity Across Documents:** MongoDB provides atomic operations at the document level, but transactions across multiple documents are limited. This can lead to consistency challenges in certain scenarios.
 - **Query Performance with Complex Joins:** MongoDB is not optimized for complex join operations like relational databases. While the aggregation framework provides powerful capabilities, performing complex joins can be less efficient compared to traditional SQL databases for certain use cases.
 - **Schema Design Challenges:** While MongoDB's flexible schema is advantageous in many scenarios, it can also introduce challenges in schema design and data consistency, especially in applications with evolving schemas.
 - **Backup and Restore:** MongoDB's backup and restore mechanisms are not as mature as some traditional relational databases. Implementing a robust backup strategy requires careful planning and consideration.