**MongoDB**

* **MongoDB is an open source** NoSQL database management program. **NoSQL (Not only SQL)** is used as an alternative to traditional relational databases.
* NoSQL databases are quite useful for working with large sets of distributed data. MongoDB is a tool that can manage document-oriented information, store or retrieve information.
* MongoDB is used for high-volume data storage, helping organizations store large amounts of data while still performing rapidly.
* Organizations also use MongoDB for its ad-hoc queries, indexing, load balancing, aggregation, server-side JavaScript execution and other features.

Structured Query Language (SQL) is a standardized programming language that is used to manage relational databases. SQL normalizes data as schemas and tables, and every table has a fixed structure.

* Instead of using **tables and rows as in relational databases**, as a NoSQL database, the **MongoDB architecture is made up of collections and documents**.
* **Collections** are equivalent of **SQL tables**, contain document sets. **Documents** are made up of **key-value pairs** -- MongoDB's basic unit of data.

**Difference between SQL and NoSQL**

| **SQL** | **NoSQL** |
| --- | --- |
| **RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)** | **Non-relational or distributed database system.** |
| These databases have fixed or static or predefined schema | They have a dynamic schema |
| These databases are not suited for hierarchical data storage. | These databases are best suited for hierarchical data storage. |
| These databases are best suited for complex queries | These databases are not so good for complex queries |
| Vertically Scalable | Horizontally scalable |
| **Examples:**[MySQL](https://www.geeksforgeeks.org/mysql-common-mysql-queries/), [PostgreSQL](https://www.geeksforgeeks.org/what-is-postgresql-introduction/), Oracle, MS-SQL Server, etc | **Examples:**[MongoDB](https://www.geeksforgeeks.org/mongodb-tutorial/), [GraphQL](https://www.geeksforgeeks.org/graphql-query/), [HBase](https://www.geeksforgeeks.org/architecture-of-hbase/), [Neo4j](https://www.geeksforgeeks.org/neo4j-introduction/),[Cassandra,](https://www.geeksforgeeks.org/apache-cassandra-nosql-database/) etc |

**MongoDB vs. RDBMS: What are the differences?**

* A relational database management system (RDBMS) is a collection of programs and capabilities that let IT teams and others create, update, administer and otherwise interact with a relational database.
* RDBMS store data in the form of tables and rows. RDBMS most commonly uses SQL.
* One of the main differences between MongoDB and RDBMS is that RDBMS is a relational database while MongoDB is nonrelational.
* Likewise, while most RDBMS systems use SQL to manage stored data, MongoDB uses BSON for data storage -- a type of NoSQL database.
* While RDBMS uses tables and rows, MongoDB uses documents and collections. In RDBMS a table -- the equivalent to a MongoDB collection -- stores data as columns and rows. Likewise, a row in RDBMS is the equivalent of a MongoDB document but stores data as structured data items in a table.
* A column denotes sets of data values, which is the equivalent to a field in MongoDB.

|  |  |
| --- | --- |
| RDBMS | **MongoDB** |
| Database | **Database** |
| Table | **Collection** - stores data as columns and rows. |
| Row | **Document** - stores data as structured data items |
| Column | **Field** - denotes sets of data values |
| Primary Key | Primary Key (Default key \_id provided by MongoDB itself) |

**MongoDB is also better suited for hierarchical storage.**

**Why is MongoDB used?**

An organization might want to use MongoDB for the following:

* **Storage**. MongoDB can store large structured and unstructured data volumes and is scalable vertically and horizontally. Indexes are used to improve search performance. Searches are also done by field, range and expression queries.
* **Data integration.** This integrates data for applications, including for hybrid and multi-cloud applications.
* **Complex data structures descriptions.** Document databases enable the embedding of documents to describe nested structures (a structure within a structure) and can tolerate variations in data.
* **Load balancing.** MongoDB can be used to run over multiple servers.

**Features of MongoDB**

Features of MongoDB include the following:

* **Replication.** A replica set is two or more MongoDB instances used to provide high availability. Replica sets are made of primary and secondary servers. The primary MongoDB server performs all the read and write operations, while the secondary replica keeps a copy of the data. If a primary replica fails, the secondary replica is then used.
* **Scalability.** MongoDB supports vertical and horizontal scaling. Vertical scaling works by adding more power to an existing machine, while horizontal scaling works by adding more machines to a user's resources.
* **Load balancing.** MongoDB handles load balancing without the need for a separate, dedicated load balancer, through either vertical or horizontal scaling.
* **Schema-less.** MongoDB is a schema-less database, which means the database can manage data without the need for a blueprint.

**Schema-less databases are a type of NoSQL database that do not require a predefined schema to store data. Instead, they allow data to be stored in flexible and dynamic formats, such as JSON documents, key-value pairs, graphs, or columns.**

* **Document**. Data in MongoDB is stored in documents with key-value pairs instead of rows and columns, which makes the data more flexible when compared to SQL databases.

**Above Scalability feature of mongoDB is explained in detail.**

**Scaling**

Scaling alters the size of a system. In the scaling process, we either compress or expand the system to meet the expected needs. The scaling operation can be achieved by adding resources to meet the smaller expectation in the current system, by adding a new system to the existing one, or both.

Scaling can be categorized into 2 types:

* **Vertical Scaling:**

When new resources are added to the existing system to meet the expectation, it is known as vertical scaling.

Consider a rack of servers and resources that comprises the existing system. Now when the existing system fails to meet the expected needs, and the expected needs can be met by just adding resources, this is considered vertical scaling. Vertical scaling is based on the idea of adding more power(CPU, RAM) to existing systems, basically adding more resources.

Vertical scaling is not only easy but also cheaper than Horizontal Scaling. It also requires less time to be fixed.

* **Horizontal Scaling:**

When new server racks are added to the existing system to meet the higher expectation, it is known as horizontal scaling.

Now when the existing system fails to meet the expected needs, and the expected needs cannot be met by just adding resources, we need to add completely new servers. This is considered horizontal scaling. Horizontal scaling is based on the idea of adding more machines to our pool of resources. Horizontal scaling is difficult and also costlier than Vertical Scaling. It also requires more time to be fixed.

**Advantages of MongoDB**

MongoDB offers several potential benefits:

* **Schema-less.** Like other NoSQL databases, MongoDB doesn't require predefined schemas. It stores any type of data. This gives users the flexibility to create any number of fields in a document, making it easier to scale MongoDB databases compared to relational databases.
* **Document-oriented.** One of the advantages of using documents is that these objects map to native data types in several programming languages., Having embedded documents also reduces the need for database joins, which can lower costs.
* **Scalability.** A core function of MongoDB is its horizontal scalability, which makes it a useful database for companies running big data applications. In addition, sharding lets the database distribute data across a cluster of machines. MongoDB also supports the creation of zones of data based on a shard key.
* **Third-party support.** MongoDB supports several storage engines and provides pluggable storage engine APIs that let third parties develop their own storage engines for MongoDB.

**Disadvantages of MongoDB**

Though there are some valuable benefits to MongoDB, there are some downsides to it as well.

* **Continuity.** With its automatic failover strategy, a user sets up just one master node in a MongoDB cluster. If the master fails, another node will automatically convert to the new master. This switch promises continuity, but it isn't instantaneous -- it can take up to a minute.
* **Data consistency.** MongoDB doesn't provide full referential integrity through the use of foreign-key constraints, which could affect data consistency.
* **Security**. In addition, user authentication isn't enabled by default in MongoDB databases. However, malicious hackers have targeted large numbers of unsecured MongoDB systems in attacks, which led to the addition of a default setting that blocks networked connections to databases if they haven't been configured by a database administrator.

**How to install and setup MongoDB**

**Step 1:**

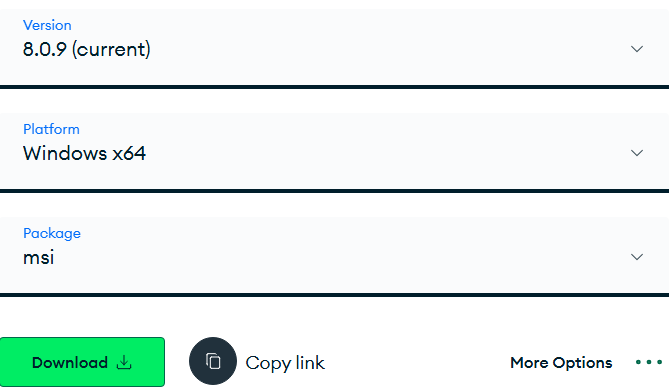
* Go to <https://www.mongodb.com/>
* Under the **Products** tab

**|-**Community Edition

**|-**Download Community

**|-**Download Mongo dB Community server (8.0.9)

**-**Select Version, Platform, Package(msi- **Microsoft Software Installer**) and Download it



* **Run and install downloaded version**

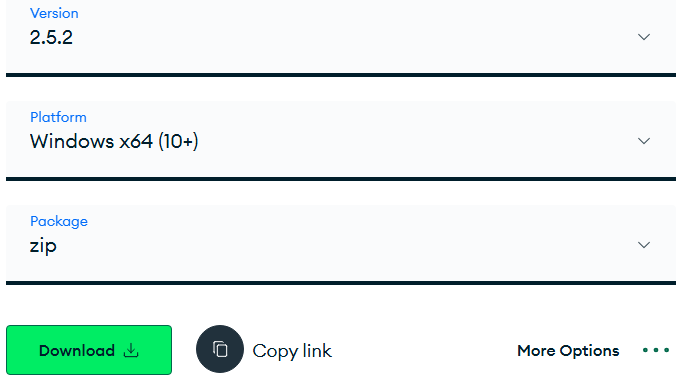
**Step 2:**

**Mongo Shell**

The mongo shell is an interactive JavaScript interface to MongoDB. You can use the mongo shell to query and update data as well as perform administrative operations.

To download

* Go to <https://www.mongodb.com/try/download/shell> (version - 2.5.2)



* Download zip
* Extract files and from **bin** folder **Copy** **exe** and **dll** files to “**C:\Program Files\MongoDB\Server\8.0\bin**”

**Step 3:**

Set path property in environment variable (Under **User variables temporary if you don’t have administrative rights**)

* Select “System Variable” > Path
* Click on “Edit”
* Add “**C:\Program Files\MongoDB\Server\8.0\bin**”

**Step 4:Verify MongoDB Installation**

### C**heck MongoDB Server Version**

### To confirm MongoDB is installed and accessible via command line:

Open **Command Prompt, type:**

mongod --version

This will display the version of the **MongoDB server** (mongod) installed.

S**tart the MongoDB Shell**

To start the **MongoDB Shell (**mongosh**)**:

In **Command Prompt**, type:

mongosh

This connects to the **default MongoDB instance** running on: **localhost:27017**

**What is mongosh?**

The mongosh command starts the **MongoDB Shell**, a command-line interface for interacting with your MongoDB server. It allows you to:

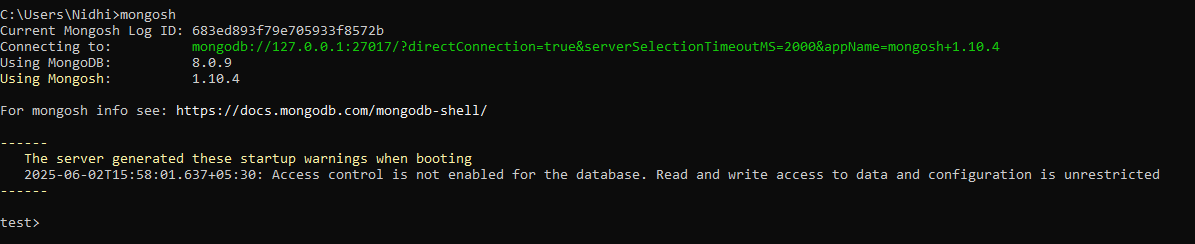
* Perform administrative tasks
* Create and manage databases and collections
* Insert, query, and update documents
* Run MongoDB commands and scripts

**Connect to a Specific MongoDB Instance:**

mongosh "mongodb://hostname:port"

mongosh "mongodb://192.168.1.100:27018"

It will shown as below screenshot.



* **Default database**
  + **Test>**

### **What is MongoDB Compass?**

**MongoDB Compass** is a graphical user interface (GUI) tool provided by MongoDB for interacting with MongoDB databases. It is designed to provide a user-friendly way to visualize, manipulate, and manage your data without needing to use the command line.

# **To download**

* **Go to** <https://www.mongodb.com/try/download/shell>
* **And download MongoDB Compass Download (GUI) (1.46.3 stable)**

**Commands**

|  |  |
| --- | --- |
| **Show Databases:** | show dbs |
| **Switch to a Database:** | use databaseName |
| **Create a Database (Switching to a non-existent database creates it):** | use newDatabase |
| **Drop current database:** | db.dropDatabase() |
| **Show Collections:** | show collections |
| **Create a Collection:** | db.createCollection("collectionName") |
| **Drop a Collection:** | db.collectionName.drop() |
| **Insert a Document:** | db.collectionName.insertOne({ key: "value" }) |
| **Insert Documents:** | db.collectionName.insertMany([{ key: "value" },{ key: "value" }]) |
| **Find All Documents in a Collection:** | db.collectionName.find() |
| **Find Documents with a Query:** | db.collectionName.find({ key: "value" }) |
| **Find Documents’ specific field with a Query:** | db.collectionName.find({ key: "value" },**{key :true}**) |
| **Update a Document:** | db.collectionName.update({ key: "value" }, **{ $set: { key: "new\_value" }}**) |
| **Update a Document and if document not available then insert a document:** | db.collectionName.update({ key: "value" }, { $set: { key: "new\_value" } },**{upsert:true}**) |
| **Delete a first Document:** | db.collectionName.deleteOne({}) |
| **Delete all Documents:** | db.collectionName.deleteMany({}) |
| **Delete a Document with a Query:** | db.collectionName.deleteOne({key: "value"}) |
| **Delete Documents with a Query:** | db.collectionName.deleteMany({key: "value"}) |
| **Show Users:** | show users |