Difference between MongoDB and MySQL

MySQL is a relational database management system (RDBMS) from the Oracle Corporation.
Like other relational systems, MySQL stores data in tables and uses structured query language (SQL) for database access.
When MySQL developers need to access data in an application, they merge data from multiple tables together in a process called a join.
In MySQL, you predefine your database schema and set up rules to govern the relationships between fields in your tables.

MongoDB is a NoSQL database that stores data as JSON-like documents.
Documents store related information together and use the MongoDB query language (MQL) for access.
Fields can vary from document to document - there is no need to declare the structure of documents to the system, as documents are self-describing.
Optionally, schema validation can be used to enforce data governance controls over each collection.

Why is using MongoDB better than using MySQL

Organizations of all sizes are adopting MongoDB, especially as a cloud database, because it enables them to build applications faster, handle highly diverse data types, and manage applications more efficiently at scale.
Development is simplified as MongoDB documents map naturally to modern, object-oriented programming languages.
Using MongoDB removes the complex object-relational mapping (ORM) layer that translates objects in code to relational tables.
MongoDB’s flexible data model also means that your database schema can evolve with business requirements.
MySQL's rigid relational structure adds overhead to applications and slows developers down as they must adapt objects in code to a relational structure.

MongoDB can also be scaled within and across multiple distributed data centers, providing new levels of availability and scalability previously unachievable with relational databases like MySQL.
As your deployments grow in terms of data volume and throughput, MongoDB scales easily with no downtime, and without changing your application.
In contrast, achieving scale with MySQL often requires significant custom engineering work.

Always-On Availability

Replication of data in MongoDB is a first-class citizen - groups of MongoDB nodes that hold the same data set are called replica sets. Replica sets enable high availability of data, with developers able to fine-tune their consistency requirements for even greater performance and availability.

- **Blazing fast failover**

If your database goes down, every second counts. MongoDB can natively detect failures, automatically electing a new primary node in less than five seconds in most cases. Applications can continue to function while the malfunctioning node is replaced.
Failover in MySQL is a manual process - taxing your operations team at the most critical time. If a database node goes down, it can take minutes before a replacement can be brought up.

- **Tuneable consistency guarantees.**

MongoDB's read concern and write concern. As an example, an application requesting a lower read concern would see lower database latency and be able to continue functioning in the event of a serious database outage, in exchange for the possibility of seeing stale data.
MySQL does not support tuneable consistency guarantees, limiting the options developers have to ensure their applications are available even if a several database nodes are down.

Faster Development with JSON Documents

Working with data as flexible JSON documents, rather than as rigid rows and columns, is proven to help developers move faster. It's not hard to find teams who have been able to accelerate development cycles by 3-5x after moving to MongoDB from relational databases. Why is this?

- **Documents are natural**

Documents represent data in the same way that applications do. Unlike the tabular rows and columns of a relational database like MySQL, data can be structured within arrays and subdocuments – in the same way applications represent data, as lists and members / instance variables respectively.

- **Documents are flexible**

Each document can store data with different attributes from other documents. As an example, consider a product catalog where a document storing details for an item of men's clothing will store different attributes from a document storing details of a television. This is a property commonly called "polymorphism". With JSON documents, we can add new attributes when we need to, without having to alter a centralized database schema. Changing schema causes downtime or significant performance overhead in a relational database like MySQL.

- **Documents make applications fast**

With data for an entity stored in a single document, rather than spread across multiple relational tables, the database only needs to read and write to a single place. Having all the data for an object in one place also makes it easier for developers to understand and optimize query performance.

Scale Infinitely and Cheaply

MongoDB includes native support in the database for sharding data across multiple nodes.

- **Scale your applications cheaply**

Sharding is cost-effective, spreading the load on the database across multiple sets of commodity hardware. Buying several low-cost machines is often cheaper than buying a smaller number of machines with significantly beefier specifications - as would be necessary to scale a relational database.

- **No need to make changes to your application to scale**

In most relational systems, scaling the database behind an application requires making application-level changes or enduring downtime while the database is migrated to a new, larger server. Since the relational data model includes frequent JOINs, placing tables across multiple nodes must be done with extreme care.

In MongoDB, a new shard can be added at any time and will automatically begin migrating data. There are no changes to be made in the application. Shards can be geographically distributed around the world with Atlas Global Clusters, providing low latency access to users around the world.

Which query language does each database use?

Both databases support a rich query language.
MySQL, like many relational databases, uses structured query language (SQL) for access.
MongoDB uses the MongoDB Query Language (MQL), designed for easy use by developers.
The documentation compares MQL and SQL syntax for common database operations.

Is MongoDB faster than MySQL?

Database performance can vary widely depending on a number of factors - database design, application query patterns and load on the database being just a few. Since MongoDB's document model stores related data together, it is often faster to retrieve a single document from MongoDB than to JOIN data across multiple tables in MySQL.
Many customers have evaluated and selected MongoDB over MySQL, both because of better performance at scale and for radical improvements to developer productivity.