Introduction to MongoDB

WHAT IS MONGODB?

MongoDB is a NoSQL, document-oriented database that uses JSON-like documents to store data

Key Features:

- Schema-less (flexible data model).
- High scalability and performance.
- Open-source, with a rich ecosystem.

Why MongoDB?

Challenges with Relational Databases:

- Fixed schemas.
- Complex joins for nested data.
- Limited scalability in distributed systems

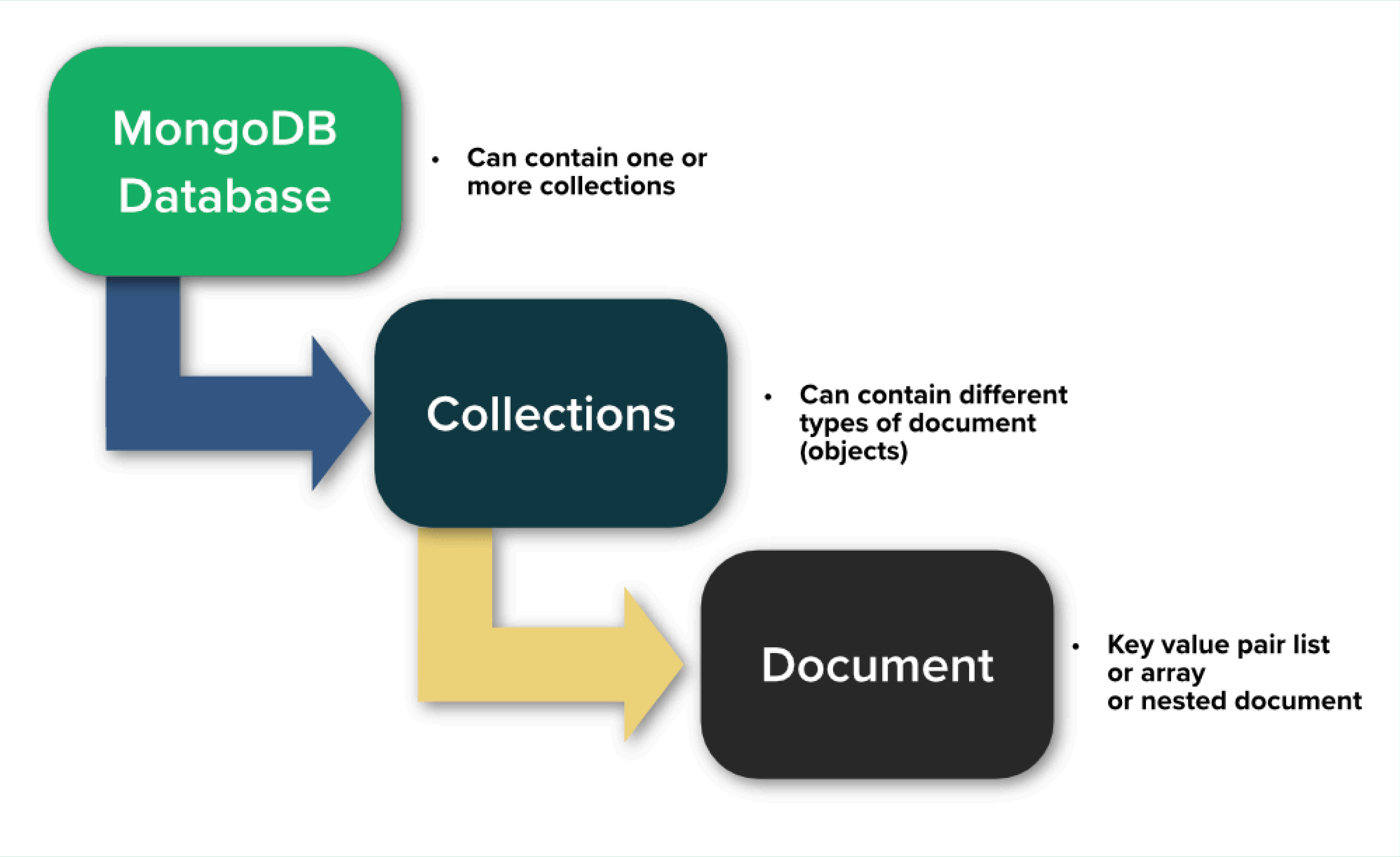
MongoDB Benefits:

- Flexible document structure.
- Horizontal scaling (sharding).
- Optimized for modern applications.

MongoDB vs Traditional Databases

Aspect	MongoDB	Relational DB (e.g., MySQL)
Data Model	Document-oriented (JSON/BSON)	Table/Row-based
Schema	Dynamic and flexible	Rigid and predefined
Scalability	Horizontal (sharding)	Vertical (scaling up hardware)
Performance	Optimized for read/write-heavy apps	Best for structured data with joins
Transactions	ACID (since version 4.0)	Fully ACID-compliant

Key Concepts of MongoDB



MongoDB Database

- Can contain one or more collections

Collections

- Can contain different types of document (objects)

Document

- Key value pair list
  or array
  or nested document

Features of MongoDB

- Schema-less design for flexibility.
- Built-in replication and high availability
- Sharding for horizontal scaling.
- Support for geospatial queries and text search.
- Aggregation framework for complex data analysis.
- Document-Oriented Storage

Database Methods

Method	Description	Example
show dbs	Lists all databases.	show dbs
use <db>	Switches to or creates a database.	use myDatabase
db	Displays the current database.	db
db.dropDatabase()	Deletes the current database.	db.dropDatabase()

Collection Methods

Method	Description	Example
show collections	Lists all collections in the current database.	show collections
db.createCollection(name, options)	Creates a new collection with optional configurations.	db.createCollection("users")
db.collection.drop()	Deletes a collection.	db.users.drop()
db.getCollection(name)	Returns a specified collection.	db.getCollection("users")

CRUD Methods

Create

Method	Description	Example
db.collection.insertOne(doc)	Inserts a single document into the collection.	db.users.insertOne({ name: "Alice" })
db.collection.insertMany(docs)	Inserts multiple documents into the collection.	db.users.insertMany([{ name: "Bob" }])

Read

Method	Description	Example
db.collection.find(query, projection)	Retrieves documents matching the query.	db.users.find({ age: { $gt: 25 } })
db.collection.findOne(query)	Retrieves the first document matching the query.	db.users.findOne({ name: "Alice" })

CRUD Methods

Update

Method	Description	Example
db.collection.updateOne(filter, update, options)	Updates a single document matching the filter.	db.users.updateOne({ name: "Alice" }, { $set: { age: 30 } })
db.collection.updateMany(filter, update, options)	Updates multiple documents matching the filter.	db.users.updateMany({ age: { $lt: 20 } }, { $set: { minor: true } })

Delete

Method	Description	Example
db.collection.deleteOne(filter)	Deletes a single document matching the filter.	db.users.deleteOne({ name: "Alice" })
db.collection.deleteMany(filter)	Deletes all documents matching the filter.	db.users.deleteMany({ age: { $lt: 18 } })

Query Modification Methods

Method	Description	Example
cursor.sort(fields)	Sorts query results by specified fields.	db.users.find().sort({ age: -1 })
cursor.limit(number)	Limits the number of query results.	db.users.find().limit(5)
cursor.skip(number)	Skips the specified number of results.	db.users.find().skip(10)
db.collection.countDocuments(query)	Counts documents matching a query.	db.users.countDocuments({ age: { $gt: 25 } })

Indexing Methods

Method	Description	Example
db.collection.createIndex(fields, options)	Creates an index on specified fields.	db.users.createIndex({ name: 1 })
db.collection.getIndexes()	Lists all indexes on a collection.	db.users.getIndexes()
db.collection.dropIndex(index)	Removes a specified index.	db.users.dropIndex("name_1")

Aggregation Methods

Method	Description	Example
db.collection.aggregate(pipeline)	Performs aggregation operations on a collection.	db.sales.aggregate([{ $group: { _id: "$category", total: { $sum: "$amount" } } }])

Admin Commands

Method	Description	Example
db.stats()	Returns statistics about the current database.	db.stats()
db.collection.stats()	Returns statistics about a specific collection.	db.users.stats()
db.serverStatus()	Provides information about the database server.	db.serverStatus()