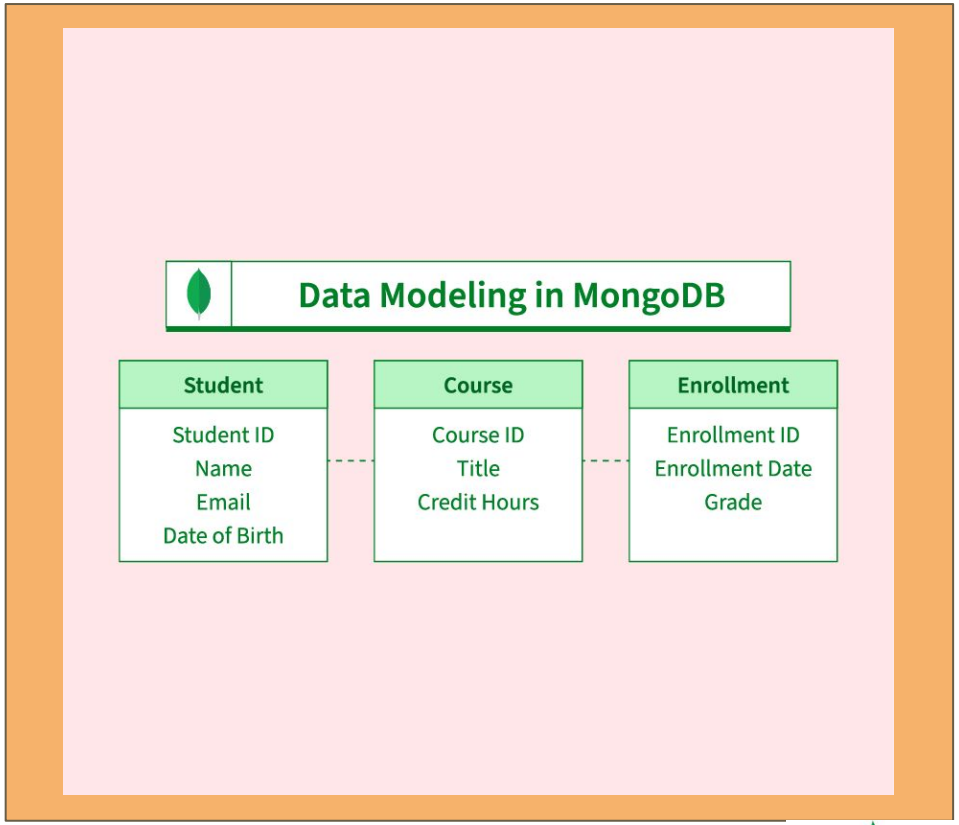
mongo DB

History

- Very successful release in 2009
- Scalability and **document-oriented design** made it popular
- **NoSQL design** allows working with big workloads and rapidly evolving applications
- Adobe, eBay and Coinbase use it for example

Data model and Schema

- **Document-based** data model
- Data stored in **BSON** (schemaless)
- Key-value pairs
- Supports complex data structures (arrays, nested documents)



Consistency and replication

CAP theorem

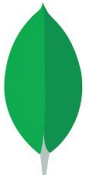
What's the CAP Theorem?

In distributed systems, you can only *guarantee two out of three* things:

- **C: Consistency** – All nodes see the same data at the same time
- **A: Availability** – Every request gets a response, even if it's not the latest data
- **P: Partition Tolerance** – System keeps working even if parts of the network break

How MongoDB uses the CAP theorem

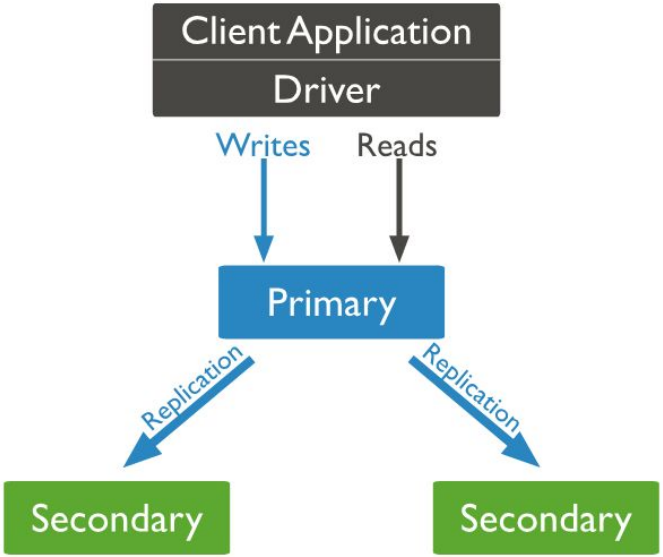
1. Always **Partition Tolerant** (P) – handles network splits
2. Chooses between:
    - **Consistency (C)** when reading from the **primary node**
    - **Availability (A)** when reading from **secondaries**
3. The trade-off is controlled using red/write operations



Replica Set

A ***replica set*** in MongoDB is a group of MongoDB processes that maintain the same data set.

Replica sets provide redundancy and **high availability**, and are the basis for *all* production deployments.



How replica sets work in MongoDB

1. **Primary Node:**
   Handles **all write operations**. It is the authoritative source of truth in the replica set.

2. **Secondary Nodes:**
   Continuously **replicate the primary's oplog** (operations log).
   They apply these operations to their own data sets to **stay in sync** with the primary.

3. This replication is **asynchronous** but typically very fast, ensuring that secondaries closely mirror the primary's state.

4. In case of primary failure, a **secondary is automatically elected** to become the new primary, ensuring uninterrupted operation.

Election process and Oplog

Election process:

Used to select the primary nodes

1. Election is triggered
2. Secondary nodes request votes
3. Majority votes win the election
4. Winner is promoted to primary node

Election can be triggered by:

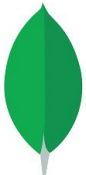
- Startup of the database
- Current primary node failing
- Manual removal of current primary node
- Reconfiguration changes

Oplog (Operations Log):

- stores a rolling log of all operations in a capped collection
- Enables **data-consistency** and **point-in-time recovery**
- It is used to sync secondary nodes with primary ones:

  How it works:

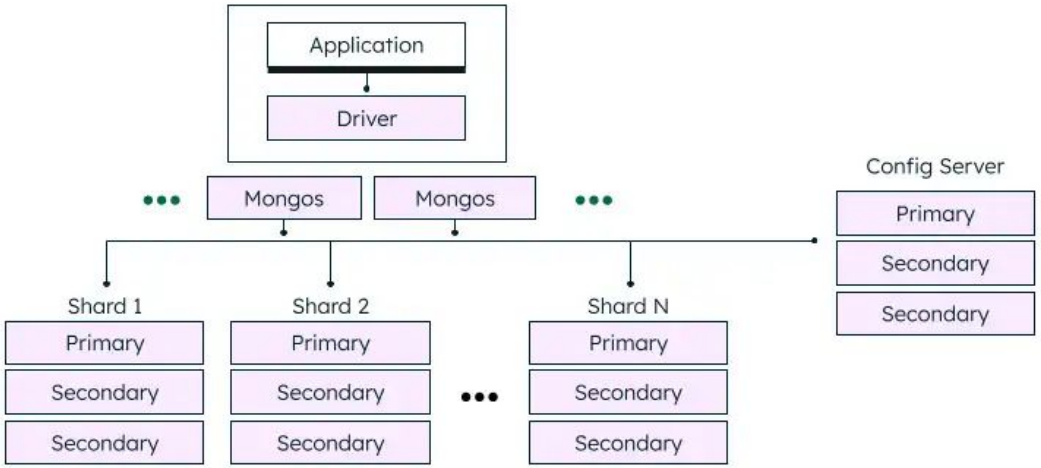
  1. All operations are done on primary node, storing them in the Oplog
  2. Operations are committed when all changes have been made
  3. Secondary nodes tail the primary node, copying all recent operations in the Oplog



Sharded cluster

What is a sharded cluster?

A **sharded cluster** splits your data across **multiple servers (shards)**



A MongoDB **sharded cluster** consists of the following components:

- **Shard**: Each shard contains a subset of the sharded data. Each shard must be deployed as a **replica set.**
- **Routing with mongos**: The mongos acts as a query router, providing an interface between client applications and the sharded cluster.
- **Config servers**: Config servers store metadata and configuration settings for the cluster. Config servers must be deployed as a replica set (CSRS).

Security

MongoDB provides various features, such as authentication, access control, encryption, to secure your MongoDB deployments.

Authentication & Authorization

- Users log in with a **username and password**
- Can connect to external systems like **LDAP** or use **security certificates**
- Uses **roles** to control *who* can do *what* in the database

Auditing & Compliance

- MongoDB provides detailed logging on who did what, so that activity can be tracked

Encryption

- **At Rest:** Data is **encrypted on disk**, so it's safe even if someone gets access to the files

- **In Transit:** Data is **encrypted while moving** between your app and the database using **TLS/SSL**