MongoDB

...

Mongodb v8.0
Mongosh v2.3.8

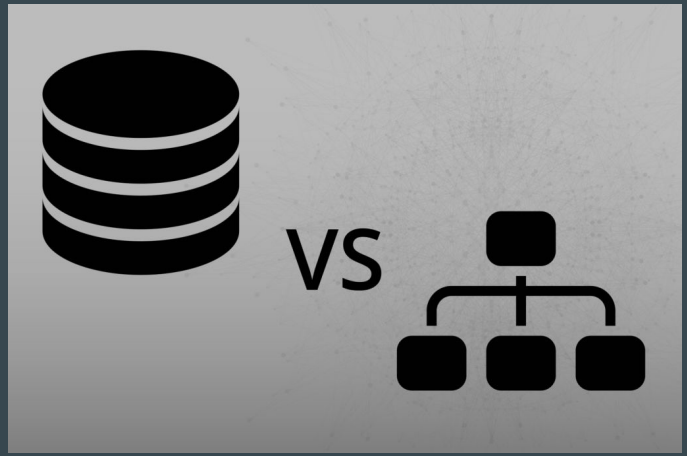
Alzahraa Eid
Full-Stack developer @fixed solutions

Agenda (Day 1)

- Overview
- Relational Database
- Non-Relational Database(Nosql)
- CAP theorem
- What is mongodb?
- Why mongodb?
- Installation
- Shell vs Drivers
- Exploring the server & shell
- Document
- JSON vs BSON
- Embedded document
- CRUD

Overview

- Data redundancy and inconsistency
- Integrity problems
- Atomicy problems
- Difficulty in accessing the data
- Security problems



File-Based vs Database

❌ File systems are not convenient and efficient as a data Model

Relational Database

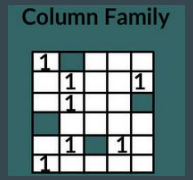
- **Data** → Structured data stored in tables
- **Schema** → Rigid
- **Language** → SQL
- **Scalability** → Vertical scaling
- **Transactions** → ACID
  - **A**tomicity: All or Nothing
  - **C**onsistency: DB must be consistent before and after
  - **I**solation: Transactions are independent
  - **D**urability: Committed data mustn't be lose even in the case of a system crash.

- Not efficient for large data and cost

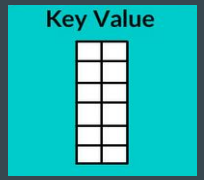
Non-Relational Database(Nosql)

- **Data** → Unstructured data
- **Schema** → Flexible (Schema–less)
- **Language** → No specific query lang, depends on the DB type
- **Scalability** → Horizontal scaling
- **Transactions** → BASE
  - **B**asically **A**vailable: guarantees availability, no isolation
  - **S**oft state: state of the system may change over time
  - **E**ventual consistency: Over time, all replicas of the database will become consistent

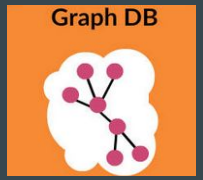
- Efficient for large data and cost
- Most NoSQL databases are open source
- Types:



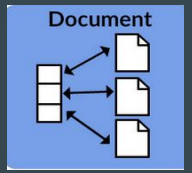
Ex: HBASE, Cassandra



Ex: Redis, riak



Ex: Neo4j



Ex: MongoDB, CouchDB

CAP theorem

Consistency

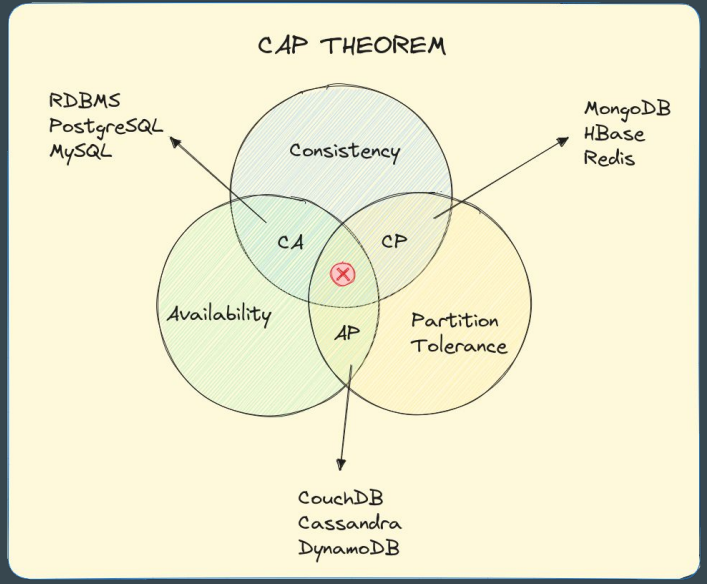
↳ All nodes in the database return the same data at any given time

Availability

↳ The system guarantees a response to every request, even if some nodes are down.

Partition Tolerance

↳ The system continues to function even if there is a network partition (communication failure between nodes)



database can guarantee at most two out of them

What is mongodb?

- NoSQL database -> Document store
- stores data in a flexible, JSON-like format called BSON (Binary JSON)

SQL

Database
- Tables
  - Rows
    - Columns

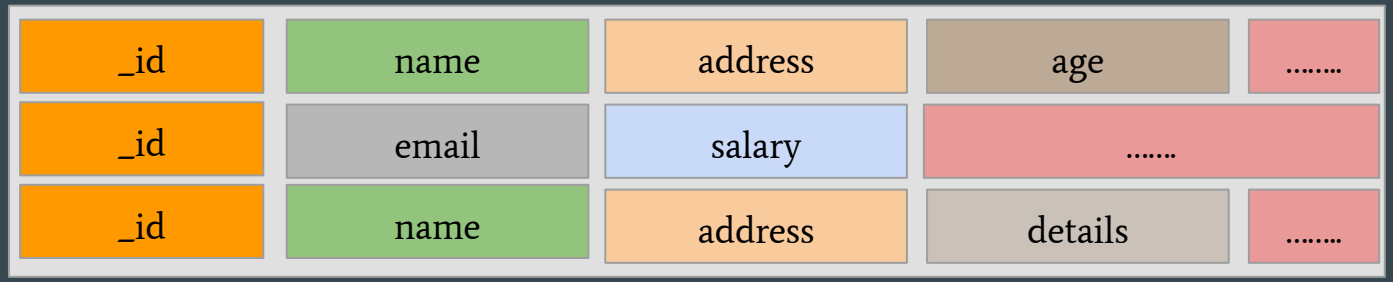
MongoDB

Database
- Collections
  - Documents
    - Fields

Why mongodb?

- Flexibility (schema-less or schema-flexible)

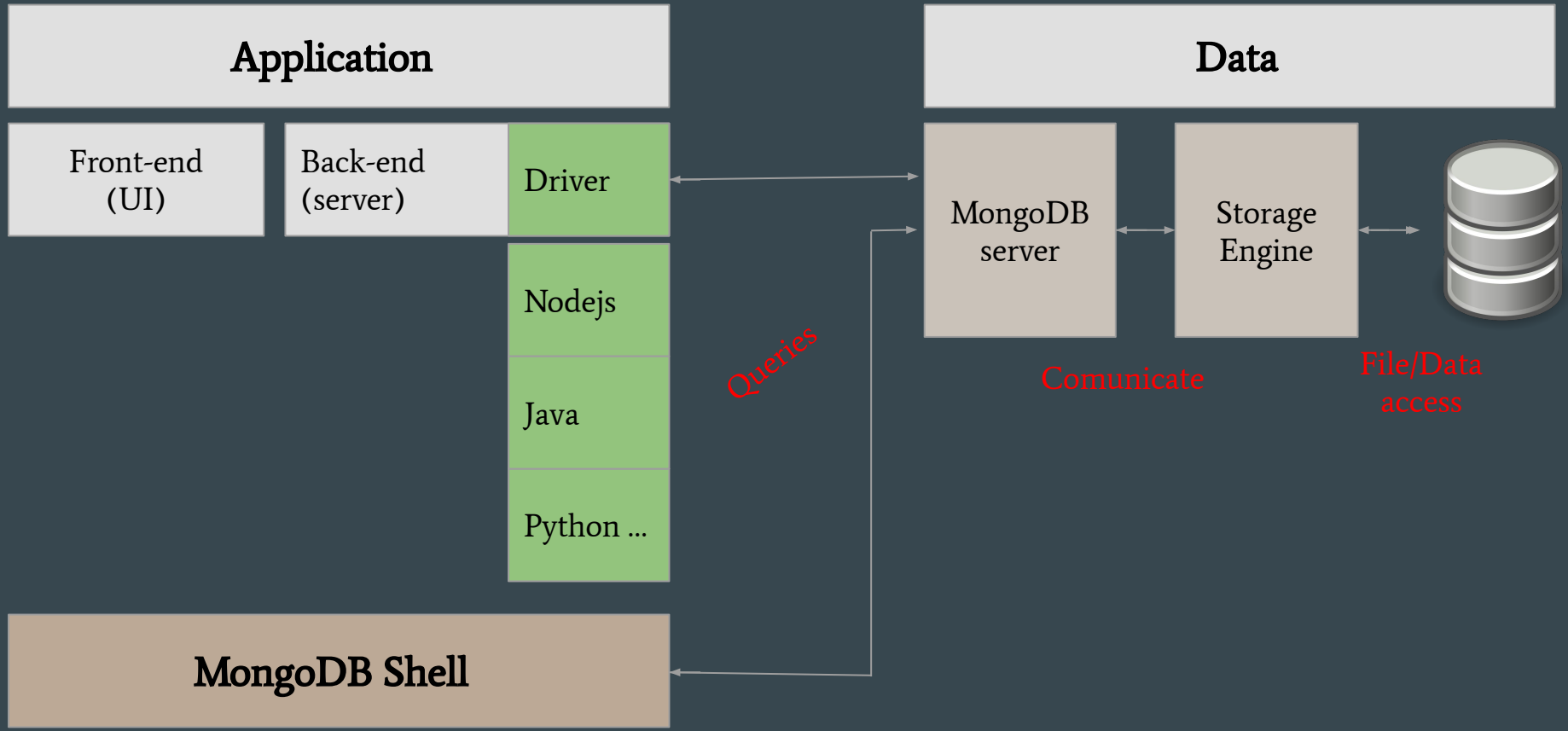
collection



- Relations (few relations)
- Efficiency (store huge data, speed, cost)

Installation

Shell vs Drivers



Exploring the server & shell

Server

The server start with default configuration
Ex: port, dbpath, logpath ....

Default config. stored in: mongod.cfg file

You can change this config:
Explore: mongod –help

Shell

The shell connected by default to : 27017 port
(default port for mongodb server)

You can change this config:
Explore: mongosh –help

Time to start

Document

Inside a document ⟶ use a format called Json to store the data

Max overall document size: 16Mb

```json
{
  "string": "Hi",
  "number": 2.5,
  "boolean": true,
  "null": null,
  "object": { "name": "Kyle", "age": 24 },
  "array": ["Hello", 5, false, null, { "key": "value", "number": 6 }],
  "arrayOfObjects": [
    { "name": "Jerry", "age": 28 },
    { "name": "Sally", "age": 26 }
  ]
}
```

Demo 1

(Explore databases, create first DB)

JSON vs BSON

JSON

- **Structure**
  - Text format
  - Support basic data types like: strings, numbers, booleans, arrays, objects and null
- **Usage in MongoDB**
  - Querying or interacting with MongoDB via drivers or the shell
- **Performance**
  - Slower to parse and less space-efficient due to its text-based nature

BSON

- **Structure**
  - Binary format
  - Supports additional data types like: ObjectId, numbers, Date Timestamp, etc...
- **Usage in MongoDB**
  - Used internally by MongoDB for storage and data transmission between the server and applications.
- **Performance**
  - Faster to parse and more efficient for storage and network transmission.

Embedded document

- document stored as a value inside another document
- Keeps related data together
- often used to represent one-to-one or one-to-few relationships
- making queries simpler and faster
- Nesting up to 100 level (warning: overall doc. Size mustn't exceed 16Mb)

Demo 2

(DataTypes)

CRUD operations

Create

– insertOne(data, options)
– insertMany(data, options)
– insert() → deprecated

Update

– updateOne(filter, data, options)
– updateMany(filter, data, options)
– replaceOne(filter, data, options)
– update() -> deprecated

Read

– find(filter, options)
– findOne(filter, options)

Delete

– deleteOne(filter, options)
– deleteMany(filter, options)

Create

- Methods —> (insertOne vs insertMany vs insert)
- Ordered insert:
  - documents will be inserted one by one in the order they are provided in the array
  - If an error occurs while inserting a document, the insertion process stops immediately, and no further documents are inserted.
  - Any documents that were successfully inserted before the error remain in the collection

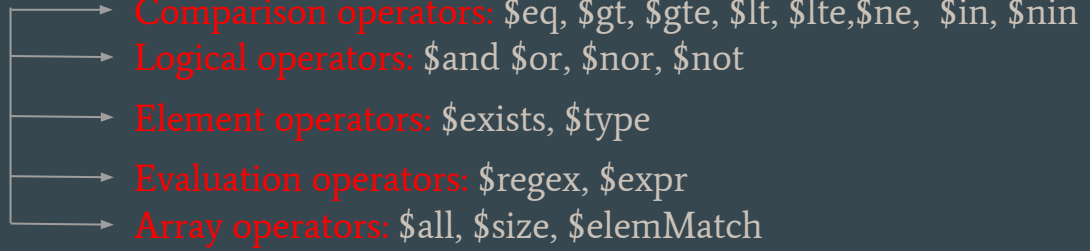
MongoDB CRUD operations are ATOMIC on the document level

Demo 3

(Create)

Read

- Methods —> (findOne vs find)
- Cursor object
- Query selectors



- Projection operators: $elemMatch, $slice

Demo 4

(Read)