NoSQL Databases and MongoDB

based on slides by **Tomoko Okochi** and

loosely based on the following book and other sources as noted

Web Development with Node and Express by Ethan Brown Published by O'Reilly Media, Inc., 2014

Databases: NoSQL

- While relational databases remain essential, **NoSQL** databases have become an important part of the database landscape. These databases break the **table/row/column** paradigm that has been established. They're often used to complement relational databases and work alongside them rather than replace them.
- **Document-oriented databases**: this is one type of NoSQL database that stores **objects (called "documents") rather than rows.**
  - Why could this be a good way of structuring databases?
- **MongoDB** is a popular document-oriented database system.

MongoDB

- MongoDB groups documents into **collections** instead of **tables.** Collections often do not have a fixed schema or strict normalization.
- **Document-oriented:** instead of **rows,** we store **documents,** which are like JSON objects. Not every document has to have the same columns, as rows do. Each document can have its own structure.
  - Document properties are not fixed and can be changed.
  - Complex hierarchies can be achieved through nested documents.
  - More intuitive, since we're already used to working with objects in JavaScript.

MongoDB: The Definitive Guide, 3rd Edition by Shannon Bradshaw; Kristina Chodorow Published by O'Reilly Media, Inc., 2019

MongoDB

- MongoDB groups documents into **collections** instead of **tables.** Collections often do not have a fixed schema or strict normalization.
- **Document-oriented:** instead of **rows,** we store **documents,** which are like JSON objects. Not every document has to have the same columns, as rows do. Each document can have its own structure.
  - Document properties are not fixed and can be changed.
  - Complex hierarchies can be achieved through nested documents.
  - More intuitive, since we're already used to working with objects in JavaScript.

Note that we can (and often should) still create a schema for our collections that define property names, data types, and constraints.

MongoDB: The Definitive Guide, 3rd Edition by Shannon Bradshaw; Kristina Chodorow Published by O'Reilly Media, Inc., 2019

MongoDB: Scalability

MongoDB is known for being easily **scalable,** which means the database can be expanded and workload spread out. databases can take advantage of **sharding**, which is the practice of spreading a database over multiple servers.

- Rather than having one super-powerful machine to store the entire database, break it up.
- Locate database resources geographically close to where they will be accessed;
- Place more frequently accessed data on more powerful machines;
- Many shards that make up a database are called a **cluster**.

MongoDB: The Definitive Guide, 3rd Edition by Shannon Bradshaw; Kristina Chodorow Published by O'Reilly Media, Inc., 2019

MongoDB: Replica Sets

MongoDB databases use **replica sets**, which are basically servers that contain copies of the database's data; if one server goes down for some reason, the data is still available.

MongoDB: The Definitive Guide, 3rd Edition by Shannon Bradshaw; Kristina Chodorow Published by O'Reilly Media, Inc., 2019

Use Cases for MongoDB

The following are some potential cases for using MongoDB:

- **Horizontal Scalability:**
  - The database can easily scale across multiple servers when the workload increases;
- **Flexible Schema**
  - The shape of the data can change fluidly as needed; this makes it easier to rapidly develop and iterate your application;
- **Speed, Availability, and Reliability**
  - Data can be quickly queried and has high tolerance against faults due to replica sets;
- **Complex Data**
  - It's easy to create documents that have arbitrary complexity and deep nesting;

From slides created by **Tomoko Okochi**

MongoDB in Node.js

- We will interact with our MongoDB database using their official **Node.js driver** that provides JavaScript functions for this purpose.
- Another popular way to interact with a MongoDB database is using an **Object-Document Model** (not to be confused with Document Object Model) called **Mongoose,** which provides simplified functions and interfaces for working with the database. ODMs are a bit slower because they have to translate objects into the data format of the database, but make it easier to interface with the database.

https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/mongoose