# Database

Class 6 - 10/6/2016

# Database?

Wikipedia will tell us that "a database is an organized collection of data."

Based on that, a database can be a JSON file, HTML, XML, TSV, CSV, or even a simple Array. In IT though, database often refers to a structure that includes the software to manipulate the data — the "Database Management System" (DBMS).

A DBMS includes functions to perform CRUD actions with safety.

# Relational Database

- What:
  - Basically a table with rows and columns.
- Examples:
  - MySQL
  - Postgres
- Choose when:
  - Data is uniform and normal (no redundancies and no empty fields).
  - Aggregation is important.
  - You need to retrieve and aggregate the data in multiple ways.
  - When your data model is unlikely to change.
  - The data is small enough that you don't have to manage it over clusters.
- Data Modeling/Schema:
  - Show tables, metadata, and links.
  - Specify type and constraints ("2 characters," "MM/DD/YYYY date").

# Non-Relational Database (No SQL Database)

- What:
  - A list of *records*.
  - Each record has a set of properties, pretty much like a JavaScript object.
- Examples:
  - MongoDB
  - Redis, Firebase
- When to use:
  - Key/value pairs are a logical model for your structure.
  - Data is not uniform or normal.
  - Aggregation is not important.
  - You need flexibility for the data model to evolve.
  - Impedance mismatch (computational processing required to convert from objects to tables to objects again) is an issue.
  - You have so much data that is need to be handled in clusters.
- Data Modeling/Schema:
  - Show the record and its properties.
  - JSON "pretty" formatting.
  - Specify type and constraints ("2 characters," "MM/DD/YYYY date").

# Why Mongo

You can work with JSON-style documents across your entire development stack!

# MongoDB 101: Getting Started

Starting MongoDB service:

```
$ mongod
```

Connecting to the MongoDB service:

On a new Terminal window, type

```
$ mongo
```

Exit the shell: hit CTRL+C or type

```
$ quit()
```

More here: https://docs.mongodb.com/manual/mongo/

# MongoDB 101: Basic commands

List all database:

```
$ show dbs
```

Show current database:

```
$ db
```

Select a database (this will also create a new database if it doesn’t exist yet)

```
$ use <db_name>
```

Show list of collections in the current database

```
$ show collections
```

Delete current database (BE CAREFUL)

```
$ db.dropDatabase()
```

# MongoDB 101: CRUD commands

CREATE

// create one entry

```
$ db.food.insert({name:'Nutella'});
```

// create one entry

```
$ db.food.save([ {name:'Strawberry'}, {name:'Cheesecake'} ]);
```

# MongoDB 101: CRUD commands

READ

// show all entry

```
$ db.food.find();
```

// format the printed result and make it look pretty

```
$ db.food.find().pretty();
```

// find a specific entry

```
$ db.food.find({name:'Strawberry'});
```

// querying with ‘greater than’ or ‘less than’ attribute

```
$ db.food.find({price: { $lt:5 } })
```

# MongoDB 101: CRUD commands

UPDATE

// updating one name to another

```
$ db.food.update({name: 'Strawberry'}, {name: 'Blueberry'})
```

// adding new property

```
$ db.food.update({name: 'Strawberry'}, { $set {price:'4'} })
```

# MongoDB 101: CRUD commands

DELETE

// remove all

```
$ db.food.remove();
```

// remove one

```
$ db.food.remove({name:'Blueberry'});
```

# GUI Tool

- RoboMongo
- MongoHub
- MongoLab (mLab)
- etc

# Mongoose

# Mongoose

- An object modeling package for Node
- Essentially works like an ORM (Object Role Modelling)
- Enables building Schema

Install:

```
$ npm install mongoose --save
```

# Node.js 101 --- Keep it compact on Github

If you're using git, you should **gitignore** the **node_modules** folder!!!

— you can Google that info