MongoDB

CodeWizard

הקסם שלנו ההצלחה שלך

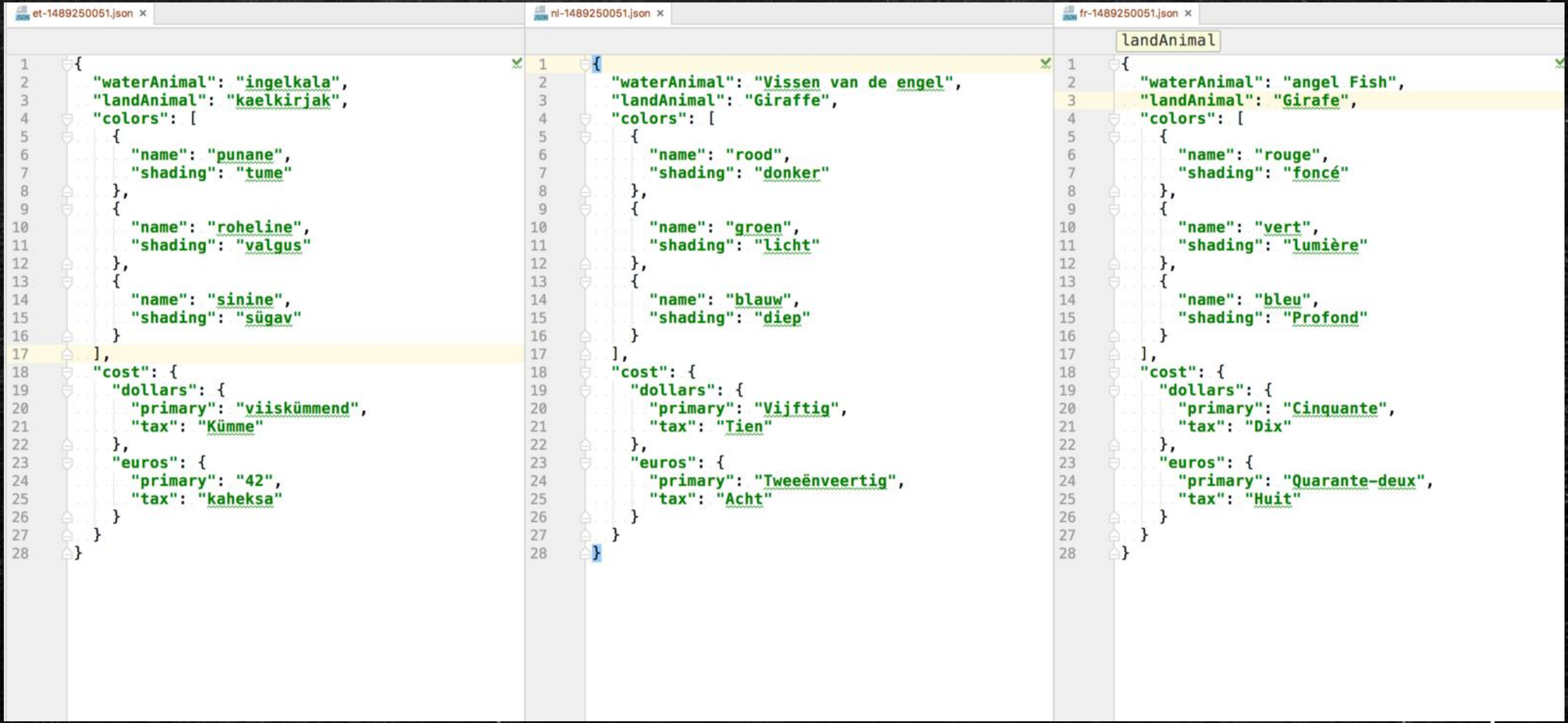
MongoDB

- document oriented NoSQL databse
- Json based
- Dynamic structure & content
- No multi document transactions

Json

- Json are JavaScript based Object
- Json are key/value pairs
- Keys are simple strings
- Values can be any legit language type (string, array, number etc)

Json



```
{
  "waterAnimal": "ingelkala",
  "landAnimal": "kaelkirjak",
  "colors": [
    {
      "name": "punane",
      "shading": "tume"
    },
    {
      "name": "roheline",
      "shading": "valgus"
    },
    {
      "name": "sinine",
      "shading": "sügav"
    }
  ],
  "cost": {
    "dollars": {
      "primary": "viiskümmend",
      "tax": "Kümme"
    },
    "euros": {
      "primary": "42",
      "tax": "kaheksa"
    }
  }
}
```

```
{
  "waterAnimal": "Vissen van de engel",
  "landAnimal": "Giraffe",
  "colors": [
    {
      "name": "rood",
      "shading": "donker"
    },
    {
      "name": "groen",
      "shading": "licht"
    },
    {
      "name": "blauw",
      "shading": "diep"
    }
  ],
  "cost": {
    "dollars": {
      "primary": "Vijftig",
      "tax": "Tien"
    },
    "euros": {
      "primary": "Tweeënveertig",
      "tax": "Acht"
    }
  }
}
```

```
{
  "waterAnimal": "angel Fish",
  "landAnimal": "Girafe",
  "colors": [
    {
      "name": "rouge",
      "shading": "foncé"
    },
    {
      "name": "vert",
      "shading": "lumière"
    },
    {
      "name": "bleu",
      "shading": "Profond"
    }
  ],
  "cost": {
    "dollars": {
      "primary": "Cinquante",
      "tax": "Dix"
    },
    "euros": {
      "primary": "Quarante-deux",
      "tax": "Huit"
    }
  }
}
```

Schema

- Schema maps to a **MongoDB collection** and defines the **shape of the documents** within that collection.

```
var userSchema = new Schema({
     name: String,
     type: String
});
```

CodeWizard
הקסם שלנו ההצלחה שלך

Schema (Types)

- String
- Number
- Date
- BufferString
- Number
- Date
- Buffer
- Boolean
- Mixed
- ObjectId
- Array
- Decimal128
- Map

Schema (Middleware)

- validate
- save
- remove
- init

Schema (additional information)

- required
- enum
- min/max
- unique
- validate

- ***And more***

Model

- Models are fancy constructors compiled from Schema definitions.
- An instance of a model is called a document.
- Models are responsible for creating and reading documents from the underlying MongoDB database

```
var Users = mongoose.model(“User”, schema);
```

CRUD

- Create
- Read
- Update
- Delete

CRUD

- Create
  - Insert
  - Save
  - Create
  - insertOne
  - insertMany
  - bulkWrite

CRUD

- Read (find)

  - Model.find()
  - Model.findOne()
  - Model.findById()
  - Model.findByIdAndDelete()
  - Model.findByIdAndRemove()
  - Model.findByIdAndUpdate()
  - Model.findOneAndDelete()
  - Model.findOneAndRemove()
  - Model.findOneAndUpdate()

CRUD

- Update
  - Model.updateMany()
  - Model.updateOne()
  - upsert boolean
    - Optional. If set to true, **creates a new document** when no document matches the query criteria.
    - **The default value is false**, which does not insert a new document when no match is found.

CRUD

- Delete
  - Model.deleteMany()
  - Model.deleteOne()