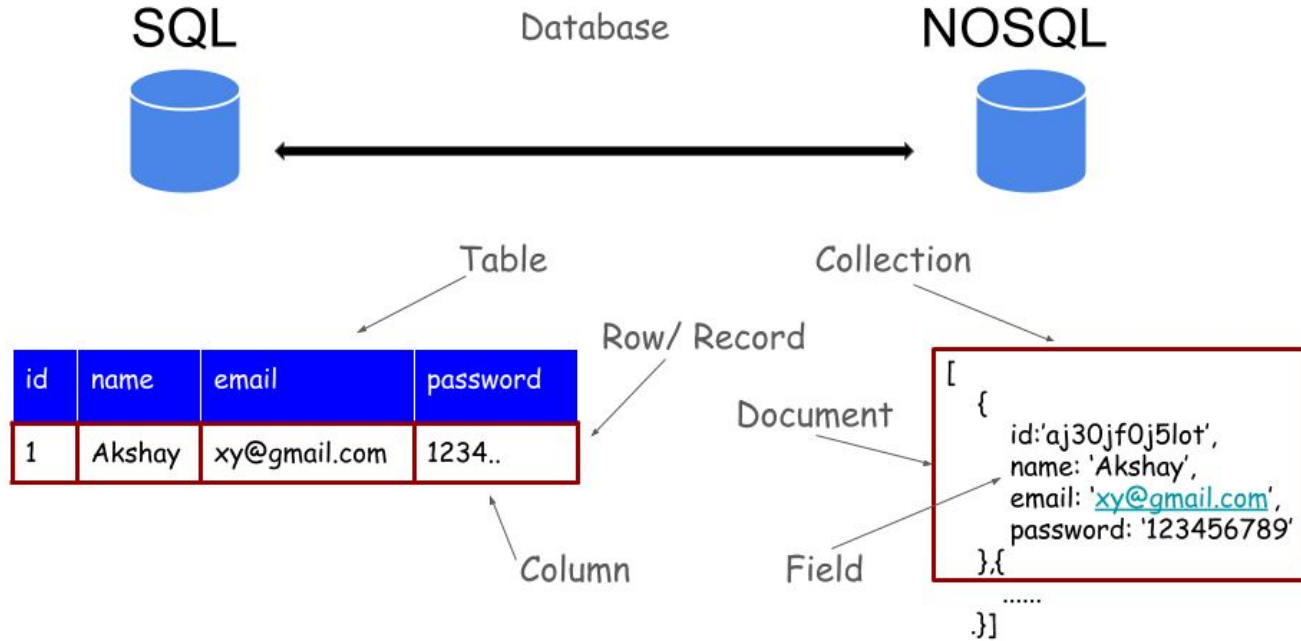

Nodejs

— Databases: mongodb —

Introduction

- MongoDB est une base de données de documents NoSQL (sans schéma).
- Cela signifie que
 - vous pouvez y stocker des documents JSON,
 - La **structure** de ces documents peut **varier** Ce qui rend les bases de données MongoDB très faciles à modifier et à mettre à jour à l'avenir
- C'est l'un des avantages de l'utilisation de NoSQL car il
 - accélère le développement d'applications
 - réduit la complexité des déploiements.

SQL vs NOSQL DB



Introduction

- Pour pouvoir expérimenter les exemples de code il faut:
 - Télécharger et installer MongoDB sur votre machine locale
 - Ou, utiliser un service cloud MongoDB à l' adresse <https://www.mongodb.com/cloud/atlas>
 - .
 - MongoDB compass
- Pour utiliser mongodb avec nodejs il faut installer le pilote MongoDB
 - ***npm install mongodb***

Créer un base de donnée

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/upmdb";

MongoClient.connect(url, function(err, db) {
  //si la base de donnée upmdb est introuvable elle sera
  automatiquement créer
  if (err) throw err;
  console.log("Database created!");
  db.close();
});
console.log("End !!!")
```

Créer une collection

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  var dbo = db.db("upmdb");
  dbo.createCollection("users", function(err, res) {
    if (err) throw err;
    console.log("Collection created!");
    db.close();
  });
});
```

Inserer un document

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  let dbo = db.db("upmdb");
  let user = { name: "reda", password: "1234" };
  dbo.collection("users").insertOne(user, function(err, res) {
    if (err) throw err;
    console.log("1 document inserted");
    db.close();
  });
});
```

Insérer plusieurs documents

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  let dbo = db.db("upmdb");
  let users=[{ name: "nadia", password: "1234" },{ name: "Khadija", password:
"1234" }];
  dbo.collection("users").insertMany(users, function(err, res) {
    if (err) throw err;
    console.log("2 documents inserted");
    db.close();
  });
});
```


Find

- **`db.collection(collection).findOne({}, (err,res)=>{})`**: renvoi le premier documents dans la collection
- **`db.collection(collection).find({}).toArray((err,res)=>{})`**: renvoi tous les documents dans la collection
- **`db.collection(collection).find({query}).toArray((err,res)=>{})`**: renvoi tous les documents correspondants à “query” dans la collection
- **`db.collection(collection).find({query}, { projection: { _id: 0, name: 1, password: 1 } }).toArray((err,res)=>{})`**: renvoi tous les documents correspondants à “query” dans la collection sans le champ ***id***

findOne

```
const MongoClient = require('mongodb').MongoClient;
const url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  let dbo = db.db("upmdb");
  dbo.collection("users").findOne({}, function(err, result) {
    if (err) throw err;
    console.log(result);
    db.close();
  });
});
```

findAll

```
const MongoClient = require('mongodb').MongoClient;
const url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  let dbo = db.db("upmdb");
  dbo.collection("users").find().toArray(function(err, result) {
    if (err) throw err;
    console.log(result);
    db.close();
  });
});
```

Find query

```
const MongoClient = require('mongodb').MongoClient;
const url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  let dbo = db.db("upmdb");
  //let query={name:"said"};
  let query={$and:[{name:"said"},{password:"1234"}]};
  dbo.collection("users").find(query).toArray(function(err, result) {
    if (err) throw err;
    console.log(result);
    db.close();
  });
});
```

Update one/many

```
const MongoClient = require('mongodb').MongoClient;
const url = "mongodb://127.0.0.1:27017/";

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  const dbo = db.db("upmdb");
  let query = { name: "nadia" };
  let values = { $set: {password: "2345" } };
  //dbo.collection("users").updateMany(query, values, function(err, res) {
  dbo.collection("users").updateOne(query, values, function(err, res) {
    if (err) throw err;
    console.log(res);
    db.close();
  });
});
```

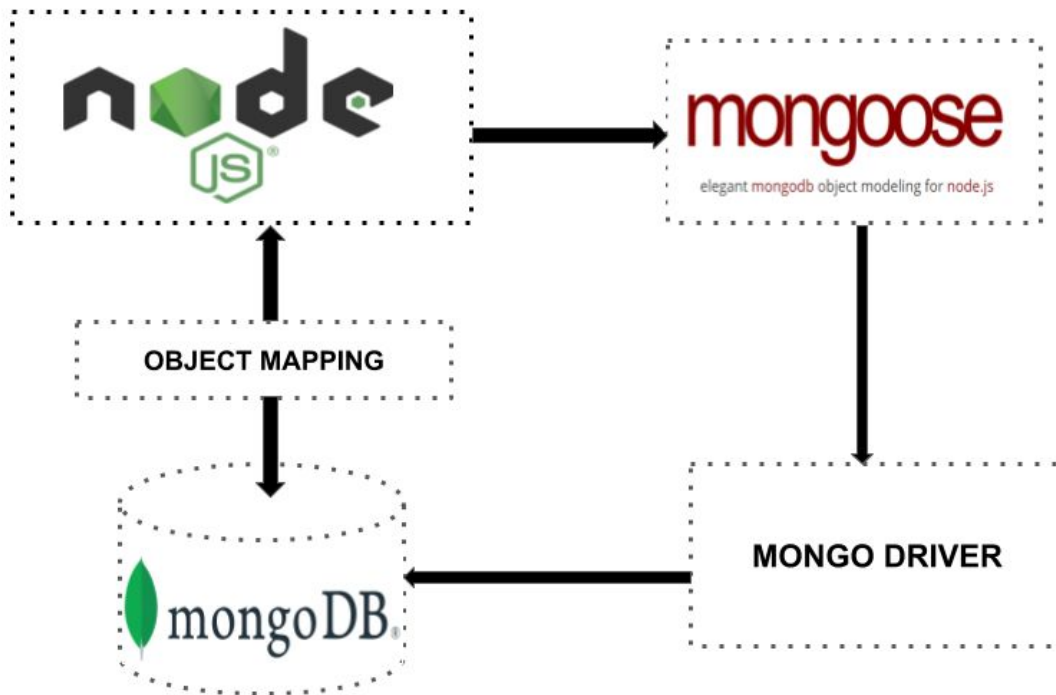
Delete one/many

```
let MongoClient = require('mongodb').MongoClient;
let url = "mongodb://localhost:27017/";
MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  let dbo = db.db("upmdb");
  //let query = { name: 'said' };
  let query = { name: /^s/ }; //starts with "s"
  //dbo.collection("users").deleteMany(query, function(err, obj) {
  dbo.collection("users").deleteOne(query, function(err, obj) {
    if (err) throw err;
    console.log("1 document deleted");
    db.close();
  });
});
```

Mongodb async

```
const MongoClient = require('mongodb').MongoClient;
const url = 'mongodb://localhost:27017';
(async function() {
  let client;
  try {
    client = await MongoClient.connect(url);
    console.log("Connected correctly to server");
    const db = client.db("upmdb");
    let r = await db.collection('users').insertOne({name:"said"});
    console.log(r)
    r = await db.collection('users').insertMany([{name:"said"}, {name:"amale"}]);
    console.log(r)
    r = await db.collection('users').find({name:"said"}).limit(2).sort({password:1}).toArray();
    console.log(r)
  } catch (err) {
    console.log(err.stack);
  }
  client.close();
})();
```

Object Data Modeling : Mongoose



C'est quoi mongoose

- Mongoose est une bibliothèque ODM (Object Data Modeling) pour MongoDB et node.js.
- gère les relations entre les données,
- fournit une validation de schéma,
- utilisé pour traduire entre les objets dans le code et la représentation de ces objets dans MongoDB.

Pourquoi Mongoose

- Par défaut, MongoDB a un modèle de données flexible.
- Cela rend les bases de données MongoDB très faciles à modifier et à mettre à jour à l'avenir.
- Mais beaucoup de développeurs sont habitués à avoir des schémas rigides.
- Mongoose force un schéma semi-rigide dès le départ.
- Avec Mongoose, les développeurs ***doivent*** définir un ***schéma*** et un ***modèle***.

Terminologies

1. **Schéma:** Un « schéma » Mongoose est une structure de données de document (ou la forme du document) qui est appliquée via la couche d'application.
- **Modèles:** des constructeurs qui prennent un schéma et créent une instance d'un document

Schéma

- Un schéma définit la structure de vos documents de collection.
- Un schéma Mongoose correspond directement à une collection MongoDB.

```
const { Schema, Model } =  
require("mongoose");  
let user=new Schema({  
  name: String,  
  age:Number,  
  email: {  
    type:String,  
    unique:true  
  }  
})
```

Modèle

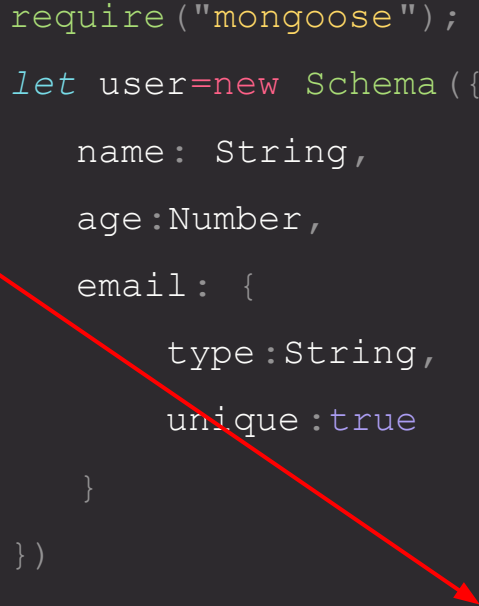
- Les modèles prennent votre schéma et l'appliquent à chaque document de sa collection.
- Les modèles sont responsables de toutes les interactions de document telles que la création, la lecture, la mise à jour et la suppression (CRUD).

```
const { Schema, Model } =  
require("mongoose");  
let user=new Schema({  
  name: String,  
  age:Number,  
  email: {  
    type:String,  
    unique:true  
  }  
})  
  
let User=new Model("User",user)
```

Modèle

- Le premier argument ("**User**") passé au modèle doit être la **forme singulière** du nom de votre collection.
- Mongoose le change automatiquement au pluriel, le transforme en minuscules (**users**) et l'utilise pour le nom de la collection de base de données.

```
const { Schema, Model } =  
require("mongoose");  
let user=new Schema({  
  name: String,  
  age:Number,  
  email: {  
    type:String,  
    unique:true  
  }  
})  
let User=new Model("User",user)
```



Connexion à la BD

```
const mongoose =require('mongoose')
const uri="mongodb+srv://<user>:<password>@<dbhost>/upmdb"
const dbConnect= async ()=>{
  try{
    const conn=await mongoose.connect(uri)
    console.log("connected to db ....")
  }catch(err) {
    console.log(err)
  }
}
dbConnect()
```

Insérer un document

```
const User = require('./models/user')
const user=new User({
  name:"said",
  age:20,
  email:"said@upm.ma"
})
const addone=async ()=> {
  await user.save();
  console.log("user added to users collection..");
}
addone()
```


Insérer un document

```
const User = require('./models/user')
const addone=async ()=> {
  const user = await User.create({
    name:"said",
    age:20,
    email:"said@upm.ma"
  });
  console.log(user);
}
addone()
```

CRUD

```
const user = await User.findOne({});  
//update user  
user.name="Ahmed"  
await user.save()  
//find by id  
const user = await User.findById("62472b6ce09e8b77266d6b1b").exec();  
//projection  
const user = await User.findById("62472b6ce09e8b77266d6b1b","name age").exec();  
console.log(user);  
//suppression  
const user = await User.deleteOne({name:"said"}).exec(); //deleteMany()  
console.log(user);
```

findOne

```
const User = require('./models/user')

const getOneUser=async ()=> {
  const user = await User.findOne({});
  console.log(user);
}

getOneUser()
```