Creación de relaciones entre documentos de una base de datos MongoDB con Mongoose

¿Qué es mongoose?

Mongoose es una biblioteca que nos permite definir esquemas en nuestra API con la misma estructura que se está utilizando en nuestros documentos de MongoDB.

Instalación y configuración

Usando npm podemos instalarlo:

```
npm i mongoose
```

Una vez instalado procedemos a definir nuestra conexión con la base de datos, en este caso estamos utilizando MongoDB.

Debemos importar en el modelo de nuestro proyecto el módulo de mongoose y asignarle la conexión a nuestra base de datos.

Ahora debemos definir nuestros esquemas, en este caso son dos:

- Alumno
- curso.

El esquema alumno será el encargado de tener la referencia al documento de curso.

- Alumno

```
<i><span style="font-weight: 400;">import</span></i> <span style="font-
weight: 400;">*</span> <i><span style="font-weight: 400;">as</span></i><span style="font-
weight: 400;"> mongoose </span><i><span style="font-
weight: 400;">from</span></i> <span style="font-weight: 400;">'</span><span style="font-
weight: 400;">mongoose</span><span style="font-weight: 400;">';</span>
export const StudentSchema = new mongoose.Schema({
 name: String,
 surname: String,
 courses: [
   {
     type: mongoose.Schema.Types.ObjectId,
     ref: 'Course',
   },
 ],
});
```

En el esquema alumno podemos ver que tiene como atributo un array de cursos debido a que un alumno puede estar inscrito en diferentes cursos. Hacemos referencia a un objeto de tipo Course.

- Curso

```
import * as mongoose from 'mongoose';

export const CourseSchema = new mongoose.Schema({
 name: String,
});
```

En el esquema curso tenemos simplemente su nombre.

No es necesario asignarles una id ya que Moongose se encarga de hacerlo.

Populate

Una vez realizados los esquemas ya podremos agregar en nuestro servicio de alumno, que es el encargado de realizar la consulta en la base de datos, la llamada al método populate() que nos permitirá anidar el objeto alumno con sus respectivos cursos.

```
async getStudents(): Promise<Student[]> {
  return this.studentModel.find()
  .populate({ path: 'courses', model: 'Course' })
  .exec();
    }
```

En el método encargado de recoger todos los alumnos le decimos que queremos poblar esa lista de alumnos con los datos del documento cursos de la base de datos, cuyo id del curso sea el mismo que aparece en el array de cursos de los alumnos. De esta manera es cómo Mongoose nos permite realizar esta relación.

Realizando la petición

Una vez definido nuestro endpoint en el controlador, podremos realizar la petición y ver el resultado.

```
[
  {
    "courses": [
      {
        "_id": "5e454efb7e322f0012fa11b5",
        "name": "curso_java"
      }],
    "_id": "5e454efb7e322f0012fa13h6",
    "name": "Autentia",
    "surname": "Real Business Solutions",
    "__v": 0
    }
]
```

Utilizando mongoose-autopopulate

Otra forma que tenemos de hacer este "populate" es con el plugin mongoose-autopopulate.

```
npm i mongoose-autopopulate
```

Una vez instalada, su uso es muy fácil, basta con agregar en nuestro esquema:

```
import * as mongoose from 'mongoose';

export const StudentSchema = new mongoose.Schema({
 name: String,
 surname: String,
 courses: [
   {
     type: mongoose.Schema.Types.ObjectId,
     ref: 'Course',
     autopopulate: true,
   },
 ],
});

StudentSchema.plugin(require('mongoose-autopopulate'));
```

Con esto ya no tendríamos que hacer nada más.

Para ampliar información, acá los siguientes enlaces:

Populate

https://mongoosejs.com/docs/populate.html

Curso MongoDB - Relación uno a muchos

https://www.youtube.com/watch?v=RX7UyF1rTDw

Nodejs Express MongoDB Relaciones entre colecciones

https://www.youtube.com/watch?v=isYQvMDiJIY

Relaciones entre colecciones con Mongoose. JOIN gracias a la función POPULATE (FullStack Bootcamp)

https://www.youtube.com/watch?v=1joABZS-m8w

Aprender a usar MongoDB: Guía 5

https://blog.findemor.es/2015/06/aprender-a-usar-mongodb-guia-5/