Associations with Sequelize

Created	@May 14, 2022 11:26 AM
Class	
Type	
Materials	
Reviewed	☐

What is association ?

In RDBMS, every entity can be related to some other entity by some relation. This relationship is only called association.

Example: Category `has many` products and a product `belongs to` a category.

So in order to build these associations in the database, we use two-three kind of strategies.

belongs to

- A book belongs to an author

So in the table of books we will be having author id as a foreign key.

has many

- An author has many books

So with the foreign key in the books table we can get all the books that belongs to an author.

many to many

- A patient has many doctors and a doctor has many patients

We will be having three tables

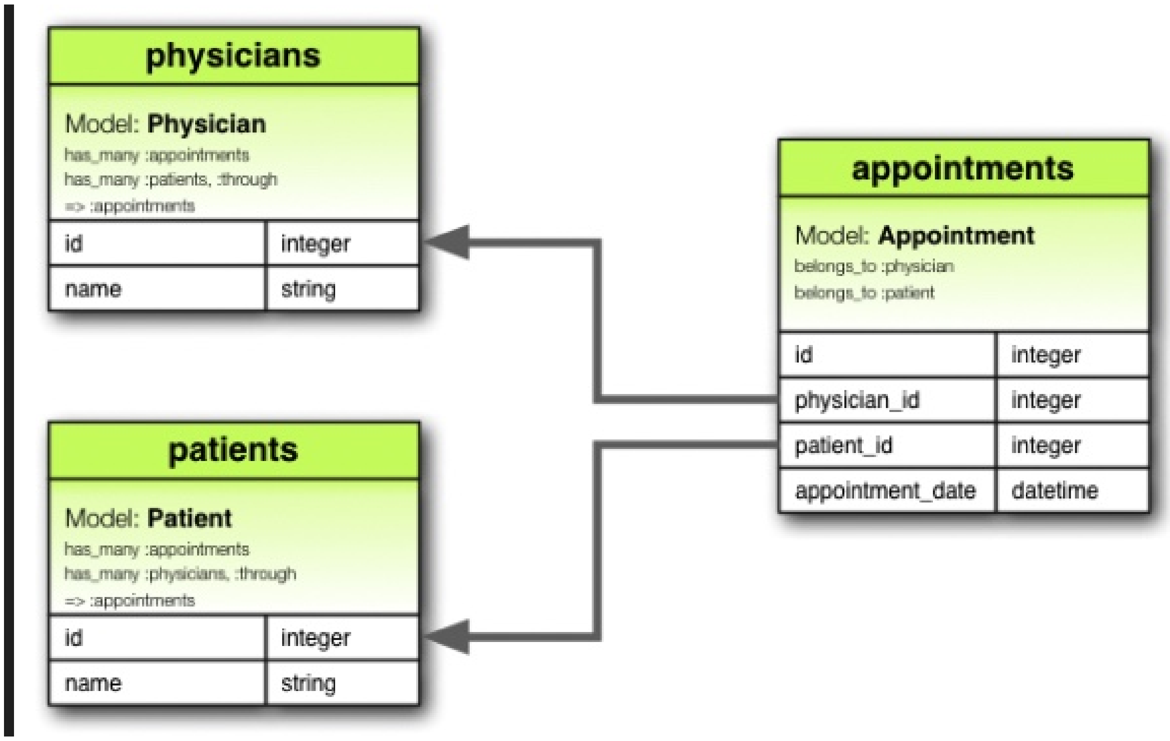
- Doctor table
- Patient table
- Doctor_Patient_Appointment table

Doctor → id, name, qualification etc

Patient → id, name, address, age, etc.

Doctor_Patient_appointment → doctor_id, patient_id, id , appointment_date etc.

So here doctor_id and patient_id are both foreign keys.



So in our ecom app, we already have a `Categories` table. We created a new Model `Products` with `categoryId` as the expected foreign key.

```
npx sequelize model:generate --name Product --attributes name:string,description:string,co
st:integer,categoryId:integer
```

Then we got a migration file as well as a Product Model file.

- In the migration file we made the properties name, cost and category id as NOT NULL by the adding the following propertu `allowNull: false` . We can now run `npx sequelize db:migrate`
- Then in the models file, we added the association between Product and Categories that a Product belongs to a category by using the association function mentioned in the product class of the model file

```
    static associate(models) {
      // define association here
      this.belongsTo(models.Categories, { // added this line of code for association
        foreignKey: {
          name: 'categoryId'
        }
      });
  }
```

The above piece of code only added one way association, i.e. from Product to categories that Product belongs to categories.

Now in order to create a product with categoryid as foreign key

```
const newProduct = await Product.create({
        name: 'Ipad',
        cost: 100000,
        description: 'apple ipad',
        categoryId: 1
    });
```

Now to fetch the product details we can use

```
await Product.findAll();
```

But this method will not return the whole category details associated with the product.

In order to fetch category details also, we can do

```
await Product.findAll({include: Categories});
```