Curso CFP

CFP Programador full-stack

CRUD Multitabla

Agenda

- Repaso
  - Create, Read, Update, Delete, ReadAll
- Mapeando tablas con Foreign Key
- One to One
- One to Many
- Many to One
- Many to Many
- Demo en Vivo
- Ejercicios

Repaso - CRUD

- Create → POST
  - Dar de alta en la DB
  - Request: Contenido en el body
- Read → GET
  - Consultar en la DB
  - Request: URL Params, Query Params
- Update → PUT
  - Actualizar en la DB
  - Request: Contenido en el body
- Delete → DELETE
  - Eliminar en la DB
  - Request: URL Params, Query Params

TypeORM - Tablas con FK (1)

- Hasta el momento vimos como asociar una clase con una tabla
- Cuando queremos trabajar con mapeos a tablas con relaciones entre si, usamos algunas anotaciones específicas sobre las entities
- Las anotaciones que vamos a usar son
  - @OneToOne
  - @OneToMany
  - @ManyToOne
  - @ManyToMany
  - @JoinColumn

TypeORM - Tablas con FK (2)

- Vamos a usar una entidad como ejemplo para las relaciones → PhotoAlbum

```
import { PrimaryGeneratedColumn, Column, Entity } from "typeorm";

@Entity('photo_album')
export class PhotoAlbum {

    @PrimaryGeneratedColumn()
    id: number;

    @Column({
        length: 255
    })
    name: string;
}
```

TypeORM - Relación One to One (1)

- Son las relaciones en donde A contiene solo una instancia de B
- Por ejemplo User puede tener solamente un Profile, y viceversa
- Se usan dos anotaciones
  - @OneToOne
  - @JoinColumn

TypeORM - Relación One to One (2)

```
@Entity()
export class Profile {

    @PrimaryGeneratedColumn()
    id: number;

    @Column()
    gender: string;

    @Column()
    photo: string;
}
```

```
@Entity()
export class User {

    @PrimaryGeneratedColumn()
    id: number;

    @Column()
    name: string;

    @OneToOne(type => Profile)
    @JoinColumn()
    profile: Profile;
}
```

La variable “profile” sería la columna que referencia a Profile

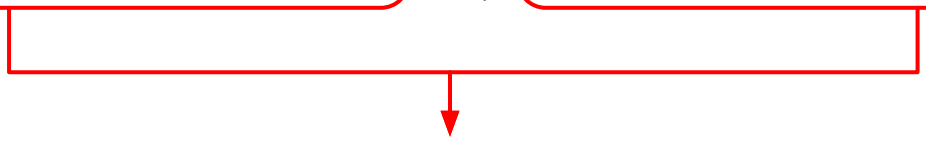
TypeORM - Relaciones One to Many y Many to One (1)

- Son relaciones en donde A contiene muchas instancias de B
- Por ejemplo User puede tener muchas fotos (Photo)
- Un usuario puede tener múltiples fotos, pero cada foto pertenece a solamente un usuario
- Las anotaciones que se usan son
  - @OneToMany
  - @ManyToOne

TypeORM - Relaciones One to Many y Many to One (2)

```typescript
@Entity()
export class Photo {

    @PrimaryGeneratedColumn()
    id: number;

    @Column()
    url: string;

    @ManyToOne(type => User, user => user.photos)
    user: User;
}
```

```typescript
@Entity()
export class User {

    @PrimaryGeneratedColumn()
    id: number;

    @Column()
    name: string;

    @OneToMany(type => Photo, photo => photo.user)
    photos: Photo[];
}
```



Un usuario puede tener muchas fotos, pero una foto tiene solamente un usuario

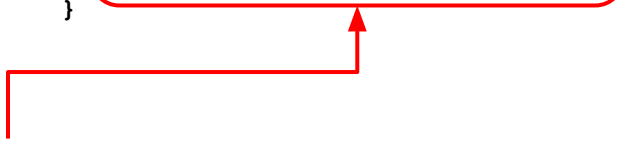
TypeORM - Relación Many to Many (1)

- Son relaciones en donde A tiene muchas instancias de B, y a la vez B tiene muchas de A
- Por ejemplo una pregunta puede tener muchas categorías, pero a la vez cada categoría puede tener muchas preguntas
- Las anotaciones que se usan son
  - @ManyToMany
  - @JoinTable
- Tener en cuenta que este tipo de relaciones se manifiestan con tres tablas
  - Dos para cada entidad
  - Una tercera con dos FK que asocian cada entity

TypeORM - Relación Many to Many (2)

```typescript
@Entity()
export class Category {

    @PrimaryGeneratedColumn()
    id: number;

    @Column()
    name: string;
}
```

```typescript
@Entity()
export class Question {

    @PrimaryGeneratedColumn()
    id: number;

    @Column()
    title: string;

    @Column()
    text: string;

    @ManyToMany(type => Category)
    @JoinTable()
    categories: Category[];
}
```



@Join puede ponerse en cualquiera de las dos entities

TypeORM - Relación Many to Many (3)

- El ejemplo genera estas tres tablas

```
+-------------+--------------+-----------------------------+
|                         category                         |
+-------------+--------------+-----------------------------+
| id          | int(11)      | PRIMARY KEY AUTO INCREMENT  |
| name        | varchar(255) |                             |
+-------------+--------------+-----------------------------+

+-------------+--------------+-----------------------------+
|                         question                         |
+-------------+--------------+-----------------------------+
| id          | int(11)      | PRIMARY KEY AUTO INCREMENT  |
| title       | varchar(255) |                             |
+-------------+--------------+-----------------------------+

+-------------+--------------+-----------------------------+
|              question categories category               |
+-------------+--------------+-----------------------------+
| questionId  | int(11)      | PRIMARY KEY FOREIGN KEY     |
| categoryId  | int(11)      | PRIMARY KEY FOREIGN KEY     |
+-------------+--------------+-----------------------------+
```

Demo en Vivo

- Ir modificando las relaciones entre Photo y PhotoAlbum
  - Para regenerar el esquema, borrar las tablas
  - Bajar la API y levantarla de vuelta
- Ir haciendo un describe de las tablas

Curso CFP

CFP Programador full-stack

Ejercicios

Ejercicios

- Sobre los endpoints existentes agregar relaciones ManyToOne
  - Tener en cuenta el body de los requests que estamos mandando