Práctica Profesionalizante I

Unidad 9

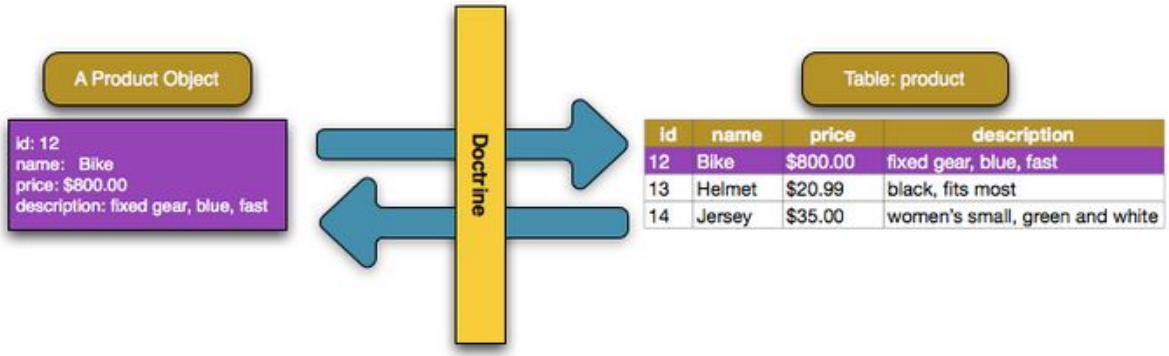
Entidades

Agenda

- Entidades
- Concepto de ORM
- Creación de entidades
- Manejo de entidades
- Relaciones entre entidades

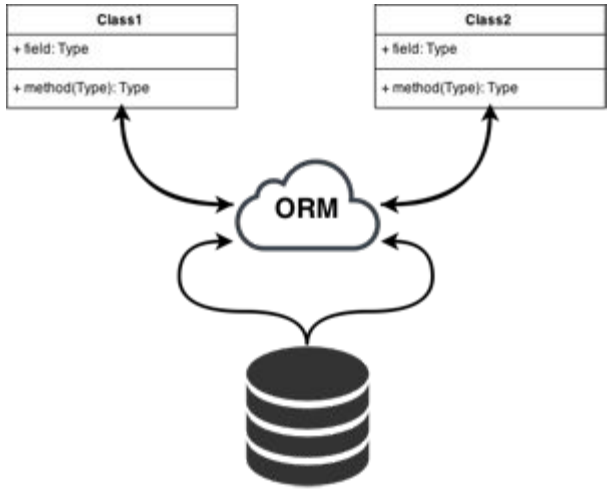
Entidades

- Objetos PHP para manipular información de la bd
- Cada tabla se representa con una entidad



ORM

- Permite mapear
  - Tablas de base de datos
  - Objetos
- Independencia del motor de db
- No utiliza SQL
- Symfony utiliza Doctrine



Configuración de Doctrine

- Debemos indicarle a Doctrine la base de datos con la que vamos a trabajar
- Se configura en el archivo .env
- Variable de entorno DATABASE_URL

```
DATABASE_URL="mysql://app:!ChangeMe!@127.0.0.1:3306/app?serverVersion=8&charset=utf8mb4"
```

Configuración de Doctrine

- Una vez configurada la conexión podemos crear la base

```
php bin/console doctrine:database:create
```

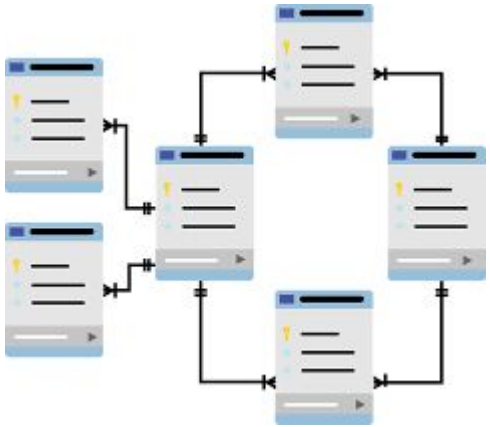
Práctica

Realizar el ejercicio 1 de la práctica

Creación de entidades

- Symfony provee una utilidad para crear entidades
- Usamos el comando make:entity

```
php bin/console make:entity
```



Práctica

Realizar el ejercicio 2 de la práctica

Creación de entidades

- Una vez creada la entidad debemos impactarla en la base de datos
- Debemos crear la tabla correspondiente
- Symfony provee dos utilidades

Creación de entidades

```
php bin/console make:migration
```

Genera un script con la definición de las tablas

```
php bin/console doctrine:migrations:migrate
```

Ejecuta el script generado y crea las tablas

Práctica

Realizar el ejercicio 3 de la práctica

Agregando un objeto

```
class ProductController extends AbstractController
{
    #[Route('/product', name: 'create_product')]
    public function createProduct(EntityManagerInterface $entityManager): Response
    {
        // creo un objeto y seteo sus atributos
        $product = new Product();
        $product->setName('Keyboard');
        $product->setPrice(1999);
        $product->setDescription('Ergonomic and stylish!');

        // persistir el objeto en la base de datos
        $entityManager->persist($product);
        $entityManager->flush();

        return new Response('Saved new product with id '.$product->getId());
    }
}
```

Generar datos de prueba

- Podemos generar datos de prueba
- Debemos instalar
  - `composer require --dev doctrine/doctrine-fixtures-bundle`
- Creamos el esqueleto del programa
  - `php bin/console make:fixtures ProductoFixtures`

Generar datos de prueba

```
// src/DataFixtures/ProductoFixture.php
<?php

namespace App\DataFixtures;

use Doctrine\Bundle\FixturesBundle\Fixture;
use Doctrine\Persistence\ObjectManager;

class ProductoFixtures extends Fixture
{
    public function load(ObjectManager $manager): void
    {
        // $producto = new Producto();
        // $manager->persist($producto);

        $manager->flush();
    }
}
```

Generar datos de prueba

Modificamos el método load

```
public function load(ObjectManager $manager)
{
    // creamos 10 productos
    for ($i = 0; $i < 10; $i++) {
        $producto = new Producto();
        $producto->setName('product '.$i);
        $producto->setPrice(mt_rand(10, 100));
        $manager->persist($product);
    }

    $manager->flush();
}
```

Cargar datos de prueba

Una vez escrito el fixture lo cargamos

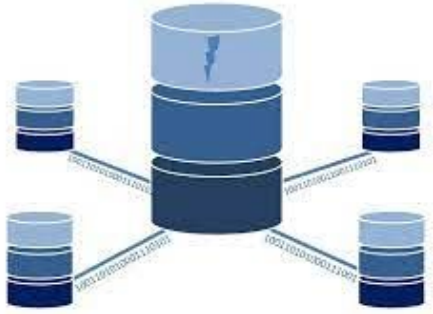
```
php bin/console doctrine:fixtures:load
```

Práctica

Realizar el ejercicio 4 de la práctica

Recuperar un objeto

- Doctrine utiliza repositorios
- Clase encargada de extraer entidades
- Se genera con la entidad
- Cada entidad tiene un repositorio



Recuperar un objeto

```
class ProductController extends AbstractController
{
    #[Route('/product/{id}', name: 'product_show')]

    public function show(ProductRepository $repository, int $id): Response
    {
        $product = $repository->find($id);

        if (!$product) {
            throw $this->createNotFoundException('No existe');
        }
    }
}
```

Recuperar un objeto

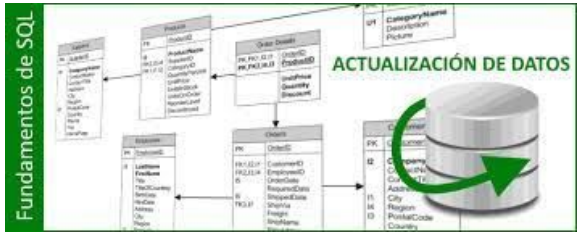
```
// busco todos los objetos productos
$productos = $repository->findAll();

// busco un producto por algún atributo
$producto = $repository->findOneBy(['name' => 'Keyboard']);
$producto = $repository->findOneBy(['name' => 'Keyboard','price' => 1999]);

// busco varios productos por un atributo y los ordeno
$productos = $repository->findBy(['name' => 'Keyboard'],['price' =>
'ASC']);
```

Actualizar un objeto

- Consiste de tres pasos:
  - Obtener el objeto desde doctrine
  - Modificar el objeto
  - Llamar a flush()



Actualizar un objeto

```
public function update(
    EntityManagerInterface $entityManager,
    int $id
): Response
{
    $repository = $entityManager->getRepository(Product::class);
    $product = $repository->find($id);

    if (!$product) {
        throw $this->createNotFoundException('No found for id '.$id);
    }

    $product->setName('New product name!');
    $entityManager->flush();
}
```

Borrar un objeto

```
$repository = $doctrine->getRepository(Product::class);
$product = $repository->find($id);

if (!$product) {
    throw $this->createNotFoundException(
        'No product found for id '.$id
    );
}

$entityManager->remove($product);
$entityManager->flush();
```

Práctica

Realizar el ejercicio 5 de la práctica