DOCUMENTACION PARA EXAMEN

MANY TO ONE (N:1)

✔ UNIDIRECCIONAL ✔

Solo existe la propiedad ManyToOne.

Dónde va la relación? → En el lado MANY

Quién es el dueño? → El lado MANY (porque tiene la FK)

php Copiar código

```php
class Pedido {
    #[ManyToOne(targetEntity: Cliente::class)]
    #[JoinColumn(name:'cliente_id', referencedColumnName:'id')]
    private Cliente $cliente;
}
```

➡️ Cliente **no tiene** `$pedidos`.

✔ BIDIRECCIONAL ✔

ManyToOne + OneToMany

Dueño → ManyToOne

Inverso → OneToMany (mappedBy)

php Copiar código

```php
class Pedido {
    #[ManyToOne(targetEntity: Cliente::class, inversedBy:'pedidos')]
    #[JoinColumn(name:'cliente_id')]
    private Cliente $cliente;
}

class Cliente {
    #[OneToMany(targetEntity: Pedido::class, mappedBy:'cliente')]
    private Collection $pedidos;
}
```

ONE TO ONE (1:1)

✔ UNIDIRECCIONAL ✔

Solo una entidad tiene relación.

Dónde va la relación?

➡️ **En la entidad donde está la FK.**

(Esto es lo que DOCTRINE exige).

Ejemplo: la FK está en Perfil

php — Copiar código

```php
class Perfil {
    #[OneToOne(targetEntity: Usuario::class)]
    #[JoinColumn(name:'id_usuario', referencedColumnName:'id', unique:true)]
    private Usuario $usuario;
}
```

➡️ Usuario NO tiene `$perfil`.

✔ BIDIRECCIONAL ✔

Ambas entidades conocen la relación.

Dueño → el lado con la FK

Inverso → el otro, con `mappedBy`

php — Copiar código

```php
class Usuario {
    #[OneToOne(targetEntity: Perfil::class, mappedBy:'usuario')]
    private Perfil $perfil;
}

class Perfil {
    #[OneToOne(targetEntity: Usuario::class, inversedBy:'perfil')]
    #[JoinColumn(name:'id_usuario')]
    private Usuario $usuario;
}
```

MANY TO MANY (N:N)

✔ UNIDIRECCIONAL ✔

Solo una entidad tiene colección.

Dónde va la relación? → En la entidad que la declares

Doctrine creará la tabla intermedia automáticamente.

php Copiar código

```php
class Estudiante {
    #[ManyToMany(targetEntity: Curso::class)]
    #[JoinTable(name:'estudiante_curso')]
    private Collection $cursos;
}
```

➡️ Curso **no tiene** `$estudiantes`.

✔ BIDIRECCIONAL ✔

Ambas entidades tienen la relación.

Dueño → lado con `JoinTable` + `inversedBy`
Inverso → lado con `mappedBy`

php Copiar código

```php
class Estudiante
{
    #[ManyToMany(targetEntity: Curso::class, inversedBy: 'estudiantes')]
    #[JoinTable(
        name: 'estudiante_curso',
        joinColumns: [
            new JoinColumn(
                name: 'id_estudiante',          // FK → estudiante.id
                referencedColumnName: 'id'
            )
        ],
        inverseJoinColumns: [
            new JoinColumn(
                name: 'id_curso',               // FK → curso.id
                referencedColumnName: 'id'
            )
        ]
    )]
    private Collection $cursos;
}

class Curso
{
    #[ManyToMany(targetEntity: Estudiante::class, mappedBy: 'cursos')]
    private Collection $estudiantes;
}
```