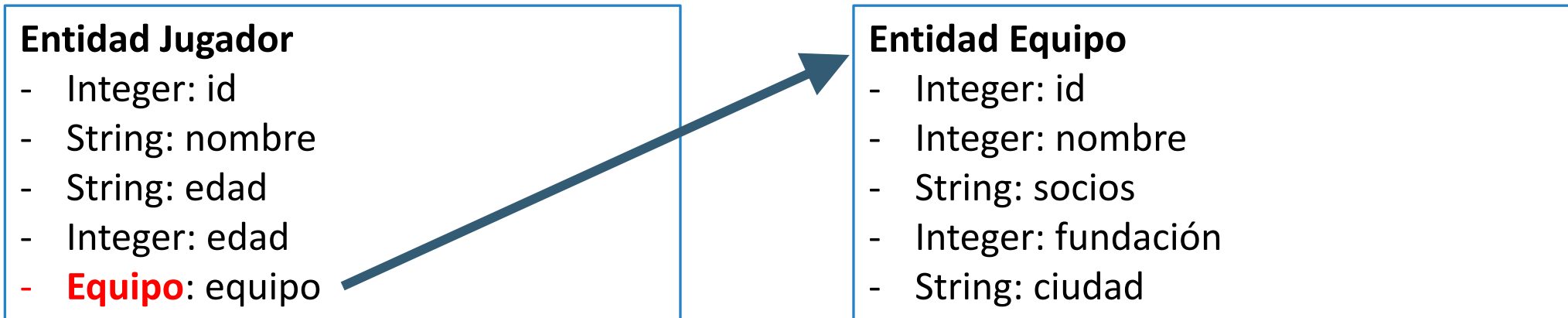


Asociaciones muchos a uno **bidireccionales**

Ahora queremos que desde el objeto-entidad Equipo podamos recuperar los jugadores:



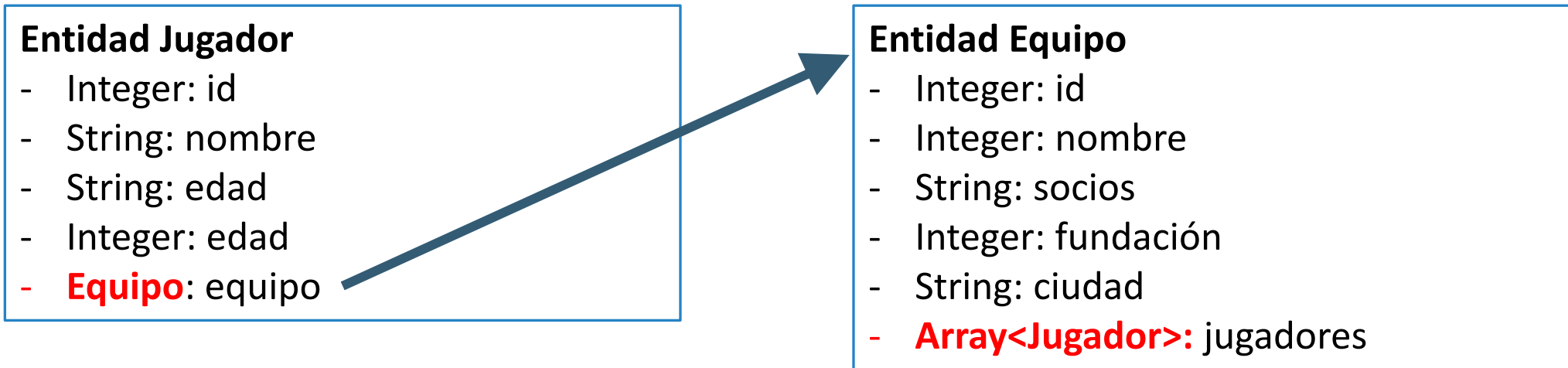
Al hacer esto, tenemos una relación bidireccional.

NO hay que cambiar la base de datos, solo configurar los atributos.



Asociaciones muchos a uno bidireccionales

Ahora queremos que desde el objeto-entidad Equipo podamos recuperar los jugadores:



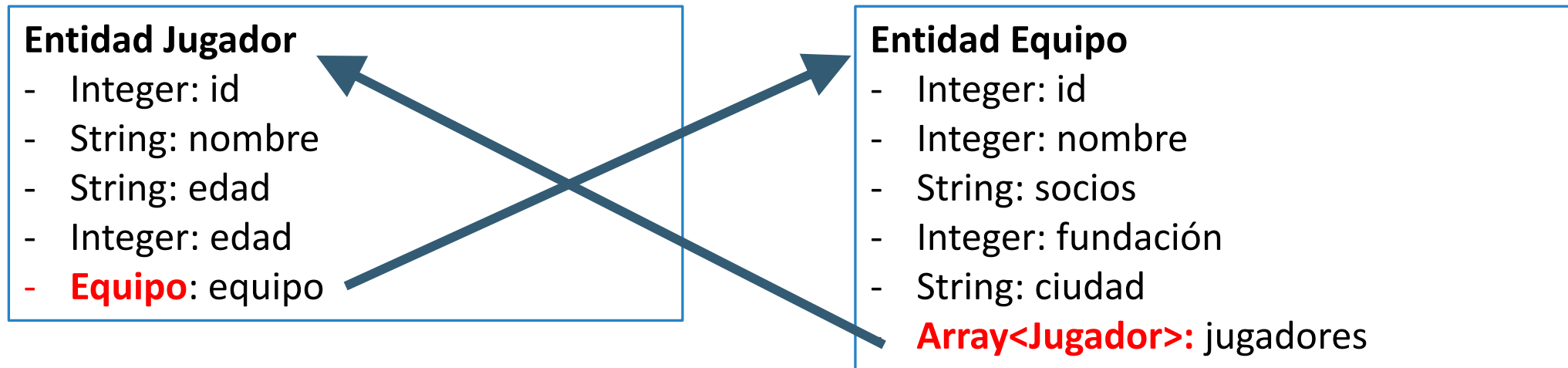
Al hacer esto, tenemos una relación bidireccional.

NO hay que cambiar la base de datos, solo configurar los atributos.



Asociaciones muchos a uno bidireccionales

Ahora queremos que desde el objeto-entidad Equipo podamos recuperar los jugadores:



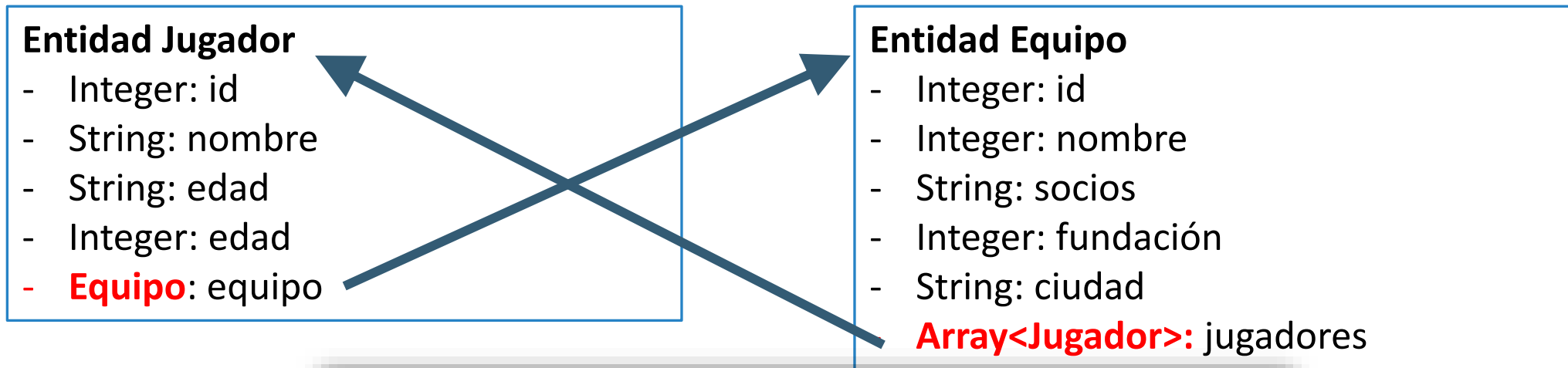
Al hacer esto, tenemos una relación bidireccional.

NO hay que cambiar la base de datos, solo configurar los atributos.



Asociaciones muchos a uno **bidireccionales**

Ahora queremos que desde el objeto-entidad Equipo podamos recuperar los jugadores:



Al hacer esto, te

Equipo(Id, Nombre, Socios, Fundación, Ciudad)

NO hay que can

Jugador(Id, Nombre, Apellidos, Edad, **Equipo**)



Asociaciones muchos a uno bidireccionales

```
src/JugadorBidireccional.php  
src/EquipoBidireccional.php  
probar_bideccional.php
```



Asociaciones muchos a uno bidireccionales

```
#[ORM\Entity]
#[ORM\Table(name: 'equipo')]
class EquipoBidireccional
{
    #[ORM\OneToMany(targetEntity:'JugadorBidireccional', mappedBy:'equipo')]
    private $jugadores;
```

```
#[ORM\Entity]
#[ORM\Table(name: 'jugador')]
class JugadorBidireccional
{
    #[ORM\ManyToOne(targetEntity:'EquipoBidireccional', inversedBy:'jugadores')]
    #[ORM\JoinColumn(name:'equipo', referencedColumnName:'id')]
    private $equipo;
```

1º Nuevo atributo



Asociaciones muchos a uno bidireccionales

```
#[ORM\Entity]
#[ORM\Table(name: 'equipo')]
class EquipoBidireccional
{
    #[ORM\OneToMany(targetEntity:'JugadorBidireccional', mappedBy:'equipo')]
    private $jugadores;
```

2º Un equipo tiene muchos Jugadores

```
#[ORM\Entity]
#[ORM\Table(name: 'jugador')]
class JugadorBidireccional
{
    #[ORM\ManyToOne(targetEntity:'EquipoBidireccional', inversedBy:'jugadores')]
    #[ORM\JoinColumn(name:'equipo', referencedColumnName:'id')]
    private $equipo;
```



src/JugadorBidireccional.php
src/EquipoBidireccional.php
probar_bideccional.php

Asociaciones muchos a uno bidireccionales

```
#[ORM\Entity]
#[ORM\Table(name: 'equipo')]
class EquipoBidireccional
{
    #[ORM\OneToMany(targetEntity:'JugadorBidireccional', mappedBy:'equipo')]
    private $jugadores;
```

```
#[ORM\Entity]
#[ORM\Table(name: 'jugador')]
class JugadorBidireccional
{
    #[ORM\ManyToOne(targetEntity:'EquipoBidireccional', inversedBy:'jugadores')]
    #[ORM\JoinColumn(name:'equipo', referencedColumnName:'id')]
    private $equipo;
```

3º Devolver un objeto de tipo
JugadorBidireccional



src/JugadorBidireccional.php
src/EquipoBidireccional.php
probar_bidireccional.php

Asociaciones muchos a uno bidireccionales

```
#[ORM\Entity]
#[ORM\Table(name: 'equipo')]
class EquipoBidireccional
{
    #[ORM\OneToMany(targetEntity:'JugadorBidireccional', mappedBy:'equipo')]
    private $jugadores;
```

```
#[ORM\Entity]
#[ORM\Table(name: 'jugador')]
class JugadorBidireccional
{
    #[ORM\ManyToOne(targetEntity:'EquipoBidireccional', inversedBy:'jugadores')]
    #[ORM\JoinColumn(name:'equipo', referencedColumnName:'id')]
    private $equipo;
```

4º Atributo de la entidad JugadorBidireccional
que contiene el equipo



src/JugadorBidireccional.php
src/EquipoBidireccional.php
probar_bideccional.php

Asociaciones muchos a uno bidireccionales

```
#[ORM\Entity]
#[ORM\Table(name: 'equipo')]
class EquipoBidireccional
{
    #[ORM\OneToMany(targetEntity:'JugadorBidireccional', mappedBy:'equipo')]
    private $jugadores;
```

```
#[ORM\Entity]
#[ORM\Table(name: 'jugador')]
class JugadorBidireccional
{
    #[ORM\ManyToOne(targetEntity:'EquipoBidireccional', inversedBy:'jugadores')]
    #[ORM\JoinColumn(name:'equipo', referencedColumnName:'id')]
    private $equipo;
```

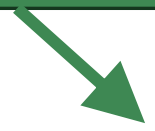
5º Devolver un objeto EquipoBidireccional



Asociaciones muchos a uno bidireccionales

```
#[ORM\Entity]
#[ORM\Table(name: 'equipo')]
class EquipoBidireccional
{
    #[ORM\OneToMany(targetEntity: 'JugadorBidireccional', mappedBy: 'equipo')]
    private $jugadores;
```

6º Atributo de la entidad EquipoBidireccional
que contiene los jugadores



```
#[ORM\Entity]
#[ORM\Table(name: 'jugador')]
class JugadorBidireccional
{
    #[ORM\ManyToOne(targetEntity: 'EquipoBidireccional', inversedBy: 'jugadores')]
    #[ORM\JoinColumn(name: 'equipo', referencedColumnName: 'id')]
    private $equipo;
```



Otras asociaciones

Asociaciones uno a uno unidireccionales:

- `@OneToOne`.

Y todas:

- <https://www.doctrine-project.org/projects/doctrine-orm/en/2.17/reference/association-mapping.html>

Referencia de anotaciones:

- <https://www.doctrine-project.org/projects/doctrine-orm/en/2.17/reference/attributes-reference.html>



Consultas - DQL

DQL

DQL (*Doctrine Query Language*):

- Lenguaje parecido a SQL.
- Permite usar Doctrine de manera alternativa a como hemos visto.
- Estamos operando sobre entidades, **NO** sobre la base de datos.

```
SELECT j FROM jugador j  
SELECT j.nombre FROM jugador j WHERE j.edad > 30
```



DQL

DQL (*Doctrine Query Language*):

- Lenguaje parecido a SQL.
- Permite usar Doctrine de manera alternativa a como hemos visto.
- Estamos operando sobre entidades, **NO** sobre la base de datos.

```
SELECT j FROM jugador j  
SELECT j.nombre FROM jugador j WHERE j.edad > 30
```

Selecciona todos los atributos “nombre”
de las entidades “jugador”
que tengan un atributo “edad” mayor que 30.



DQL: ejemplos

```
SELECT j FROM jugador j ORDER BY j.edad ASC
SELECT j.nombre FROM jugador j WHERE j.edad > 30
SELECT COUNT(p.id) as num FROM jugador p WHERE p.edad > 25
UPDATE jugador j SET j.edad = j.edad
DELETE jugador j WHERE j.edad > 70
```

- Unas devuelven un array y otras objetos completos.



DQL: ejecución

- Ejemplo: obtener el nombre de todos los objetos Jugador.

```
$query = $entityManager->createQuery("SELECT j FROM jugador j");  
$jugadores = $query->getResult();  
foreach($jugadores as $jugador) {  
    echo "Nombre: " . $jugador->getNombre();  
}
```



