

# Acceso a bases de datos en Yii 2

Ricardo Pérez López

IES Doñana, curso 2019/2020



1. DAO
2. Query Builder
3. Active Record
4. Metadatos

# 1. DAO

1.1 `yii\db\Connection`

1.2 `yii\db\Connection::createCommand()`

1.3 Consultas SQL

1.4 Sentencias no `SELECT`

## 1.1. yii\db\Connection

## 1.2. yii\db\Connection::createCommand()

## 1.3. Consultas SQL

1.3.1 `queryAll()`

1.3.2 `queryOne()`

1.3.3 `queryColumn()`

1.3.4 `queryScalar()`

## 1.4. Sentencias no SELECT

1.4.1 `execute()`

1.4.2 `insert()`

1.4.3 `update()`

1.4.4 `delete()`

## 2. Query Builder

### 2.1 `yii\db\Query`

### 2.2 Creación de consultas

### 2.3 Recogida de resultados

### 2.4 Consultas por lotes



## 2.1. yii\db\Query

## 2.2. Creación de consultas

2.2.1 `select()`

2.2.2 `from()`

2.2.3 Condiciones y filtrado de filas

2.2.4 `orderBy()`

2.2.5 `groupBy()`

2.2.6 Condiciones y filtrado de grupos

2.2.7 `limit()`

2.2.8 `offset()`

2.2.9 Combinaciones

2.2.10 `union()`

`where()`

## Formatos de condiciones

De cadena

De array

De operadores

andWhere()

orWhere()

```
filterWhere()
```

`andWhereWhere()`



```
orWhereWhere()
```

# having()

# filterHaving()

`andFilterHaving()`

`orFilterHaving()`

`join()`

`innerJoin()`

```
leftJoin()
```



```
rightJoin()
```

## 2.3. Recogida de resultados

2.3.1 `all()`

2.3.2 `one()`

2.3.3 `column()`

2.3.4 `scalar()`

2.3.5 `exists()`

2.3.6 `count()`

2.3.7 Funciones de grupo

2.3.8 `indexBy()`

sum()

average()

max()

`min()`

## 2.4. Consultas por lotes

2.4.1 `batch()`

2.4.2 `each()`

## 3. Active Record

3.1 `findOne()`

3.2 `findAll()`

3.3 `save()`

3.4 `ActiveQuery`

3.5 Atributos sucios

3.6 Relaciones

3.7 `joinWith()`

3.8 Atributos virtuales



### 3.1. findOne()

## 3.2. findAll()

### 3.3. save()

## 3.4. ActiveQuery

### 3.4.1 `find()`

## 3.5. Atributos sucios

## 3.6. Relaciones

### 3.6.1 Encadenamiento de relaciones

## 3.7. `joinWith()`

## 3.8. Atributos virtuales

### 3.8.1 Siete técnicas



## Calcular a mano cuando/donde haga falta

## Usar vistas SQL

Sobreescribir el método `find()` del modelo para que se use siempre en lugar del heredado de `ActiveRecord`

Sobreescribir el método `afterFind()` para rellenar el atributo a mano cada vez que se hace un `find()`

## Capturar el evento `EVENT_AFTER_FIND` del modelo

## Usar una propiedad con *getter* y *setter*

Crear un método `findEspecial()` que se usará en lugar de `find()` cuando haga falta

La mejor opción, en la mayoría de los casos: combinar las dos anteriores

Ejemplo



## 4. Metadatos

4.1 Objetivos de la unidad

4.2 Resultados de aprendizaje y criterios de evaluación asociados

## 4.1. Objetivos de la unidad

## 4.2. Resultados de aprendizaje y criterios de evaluación asociados

4.2.1 RA2

4.2.2 RA3

4.2.3 RA4

4.2.4 RA5

4.2.5 RA6

4.2.6 RA9

## CE4.g

## CE5.f

## CE5.g

## CE5.h

## CE6.a



## CE6.b

## CE6.c

## CE6.d

## CE6.e

## CE6.f

## CE6.g

## CE6.h

## CE9.e



## CE9.f

## CE9.g