The Database Layer of Yii Framework 2.0

Carsten Brandt

June 16, 2017 - Yiiconf, Москва

Table of Contents

Why do we need DB abstraction?

- code simplification and readablity

Why do we need DB abstraction?

- code simplification and readablity
- reusable code and MVC separation

Why do we need DB abstraction?

- code simplification and readablity
- reusable code and MVC separation
- security enhancements

3 Layers

DAO
Database Access Objects

3 Layers

DAO
Database Access Objects

Query Builder

3 Layers

DAO
Database Access Objects

Query Builder

Active Record

3 Layers

DAO
Database Access Objects

yii\db\Command

Query Builder

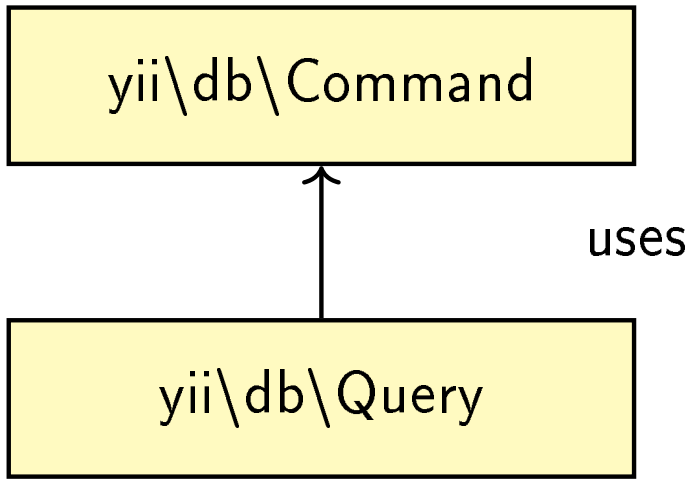
Active Record

3 Layers

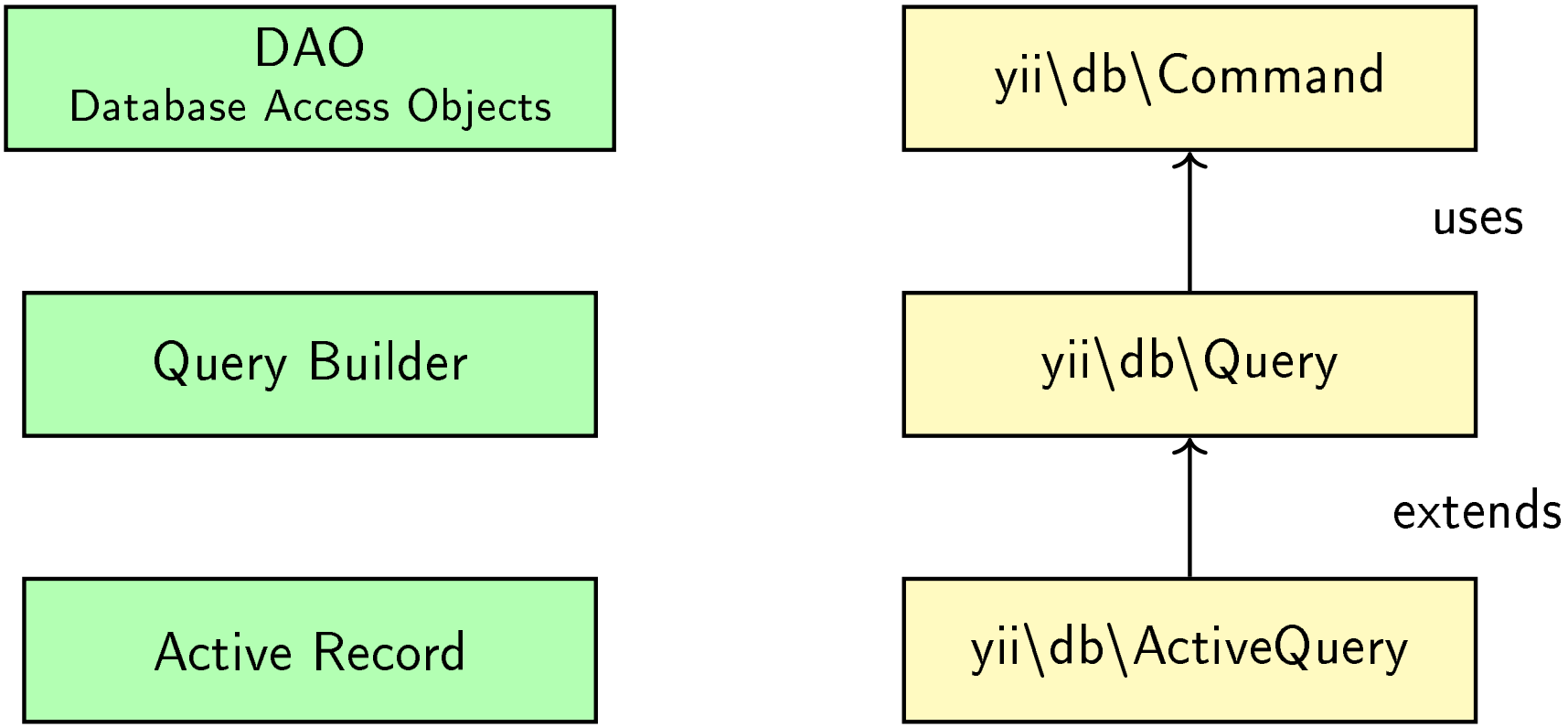
DAO
Database Access Objects

Query Builder

Active Record



3 Layers


DAO
Database Access Objects
Query Builder
Active Record
yii\db\Command
uses
yii\db\Query
extends
yii\db\ActiveQuery

Command Usage

```
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```

Command Usage

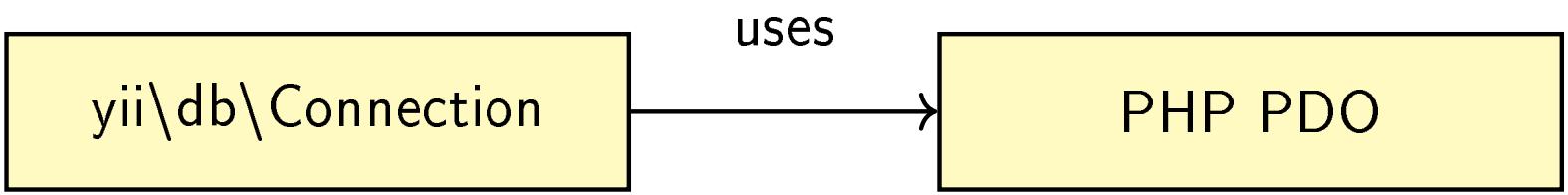
```
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```

Command Usage

```
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```

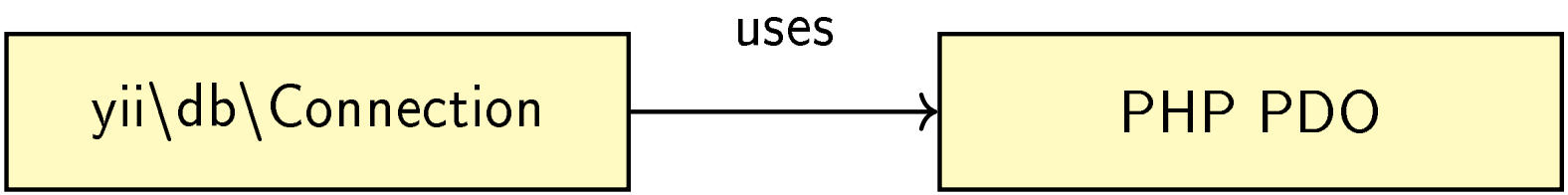
Command Usage

```
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```



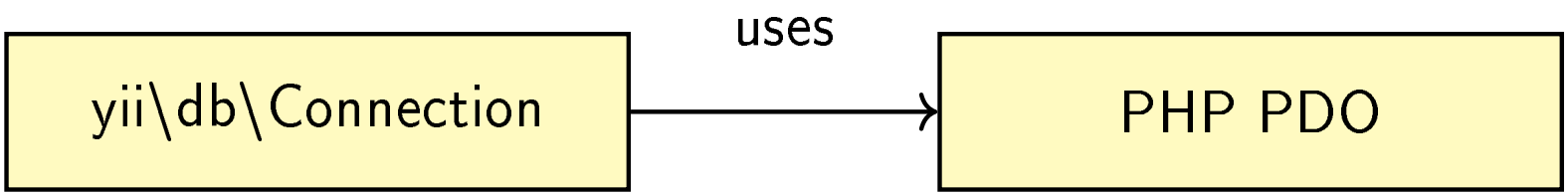
Command Usage

```php
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```



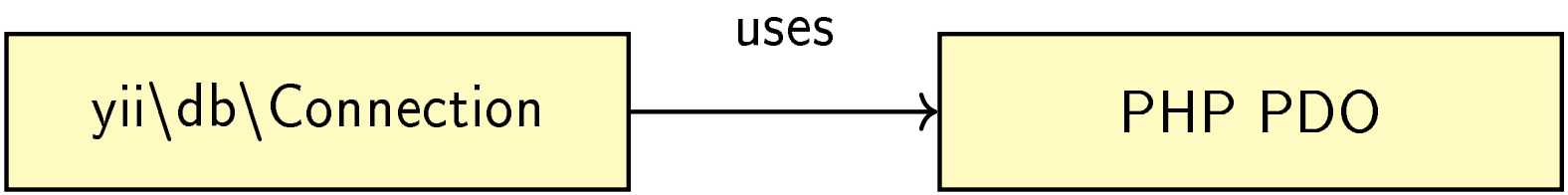
Command Usage

```
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```



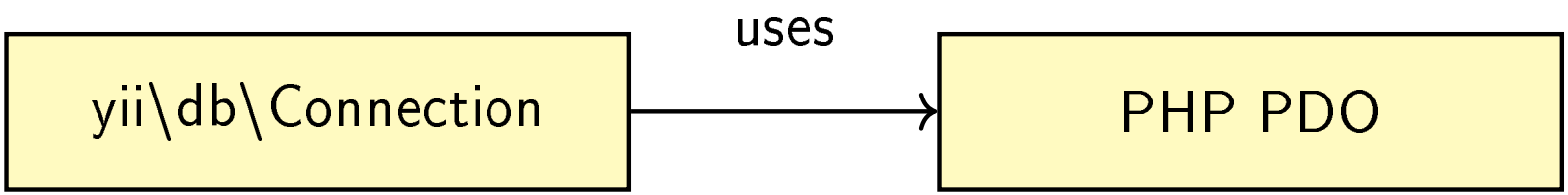
Command Usage

```
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```



Command Usage

```php
$db = new Connection([
    'dsn' => 'mysql:host=localhost;dbname=example',
    'username' => 'cebe',
    'password' => '1337',
]);

$db->open();
$command = $db->createCommand("SELECT * FROM user");
$user = $command->queryOne();
$users = $command->queryAll();
```



Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Command Usage

```
$db->createCommand()->insert('users', [
        'name' => 'CeBe',
        'email' => 'mail@cebe.cc',
        'created_at' => time(),
        'updated_at' => time(),
])->execute();
$db->createCommand()->update(
        'users', // table
        ['name' => 'Carsten'], // SET
        "name = 'Carsten'" // WHERE
])->execute();
$db->createCommand()->delete(
        'users', // table
        "name = 'Carsten'" // WHERE
])->execute();
```

Other DAO features

- Transactions

Other DAO features

- Transactions
- Read/Write splitting for Master-Slave setup

Other DAO features

- Transactions
- Read/Write splitting for Master-Slave setup
- Connection Failover

Other DAO features

- Transactions
- Read/Write splitting for Master-Slave setup
- Connection Failover
- Yii features: Events,

Other DAO features

- Transactions
- Read/Write splitting for Master-Slave setup
- Connection Failover
- Yii features: Events, Configuration,

Other DAO features

- Transactions
- Read/Write splitting for Master-Slave setup
- Connection Failover
- Yii features: Events, Configuration, DI

Query Builder

```
$query = (new yii\db\Query)
        ->select(['name', 'email'])
        ->from('users')
        ->where(['id' => 42]);
$user = $query->one();
```

Query Builder

```
SELECT name, email FROM users WHERE id = 42
```

```
$query = (new yii\db\Query)
        ->select(['name', 'email'])
        ->from('users')
        ->where(['id' => 42]);
$user = $query->one();
```

Query Builder

```
SELECT name, email FROM users WHERE id = 42
```

```
$query = (new yii\db\Query)
        ->select(['name', 'email'])
        ->from('users')
        ->where(['id' => 42]);
$user = $query->one();
```

Query Builder

```
SELECT name, email FROM users WHERE id = 42
```

```
$query = (new yii\db\Query)
        ->select(['name', 'email'])
        ->from('users')
        ->where(['id' => $_GET['id']]);
$user = $query->one();
```

Query Builder

```
SELECT name, email FROM users WHERE id = 42
```

```
$query = (new yii\db\Query)
        ->select(['name', 'email'])
        ->from('users')
        ->where('id = :id', [':id' => $_GET['id']]);
$user = $query->one();
```

Query Builder

```
SELECT name, email FROM users WHERE id = 42
```

```
$query = (new yii\db\Query)
        ->select(['name', 'email'])
        ->from('users')
        ->where('id = :id', [':id' => $_GET['id']]);
$user = $query->one();
```

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

```
SELECT *
FROM article
WHERE category_id IN (
  SELECT id FROM categories WHERE active=1
);
```

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

```
SELECT *
FROM article
WHERE category_id IN (
  SELECT id FROM categories WHERE active=1
);
```

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

```
SELECT *
FROM article
WHERE category_id IN (
  SELECT id FROM categories WHERE active=1
);
```

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

```
SELECT *
FROM article
WHERE category_id IN (
  SELECT id FROM categories WHERE active=1
);
```

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

- select()

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

- select()
- from()

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

- select()
- from()
- join()

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

- select()
- from()
- join()
- where()

Query Builder - Subqueries

```
$activeQuery = (new Query)
        ->select('id')
        ->from('categories')
        ->where(['active' => 1]);
$query = (new Query)
        ->from('article')
        ->where(['category_id' => $activeQuery]);
$articlesInActiveCategories = $query->all();
```

- select()
- from()
- join()
- where()
- union()

Query Builder

- SQL as string, vs. real code

Query Builder

- SQL as string, vs. real code
- Easier to create queries

Query Builder

- SQL as string, vs. real code
- Easier to create queries
- More secure

Query Builder

- SQL as string, vs. real code
- Easier to create queries
- More secure
- Separation of code

Active Record

```
class User extends yii\db\ActiveRecord
{
    public static function tableName()
    {
        return 'users';
    }
}
```

users

id	name
1	alice
2	bob
3	...
⋮	

Active Record

```
class User extends yii\db\ActiveRecord
{
    public static function tableName()
    {
        return 'users';
    }
}
```

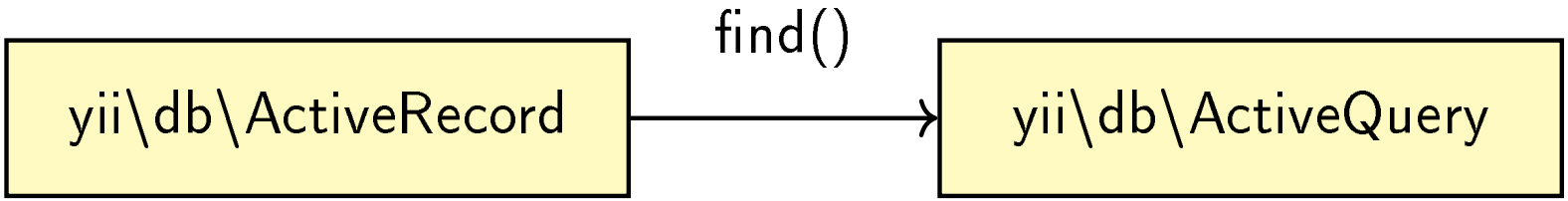
users

id	name
1	alice
2	bob
3	...
⋮	

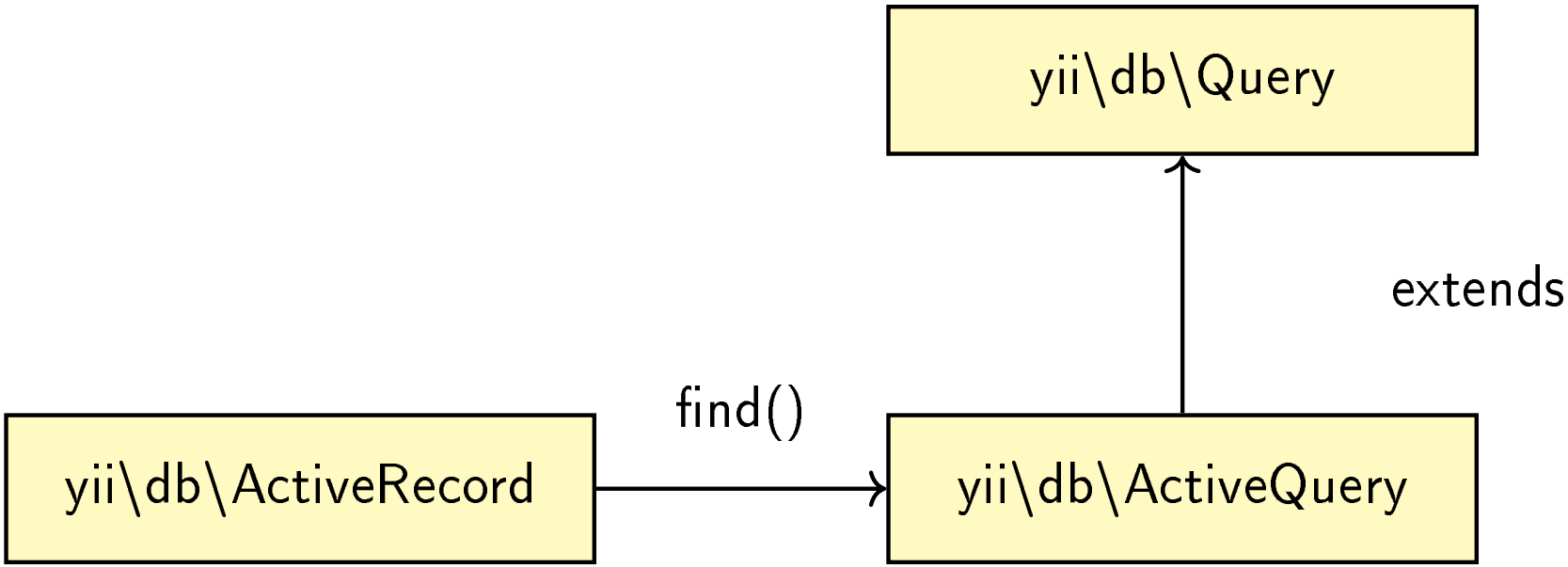
Active Record

yii\db\ActiveRecord

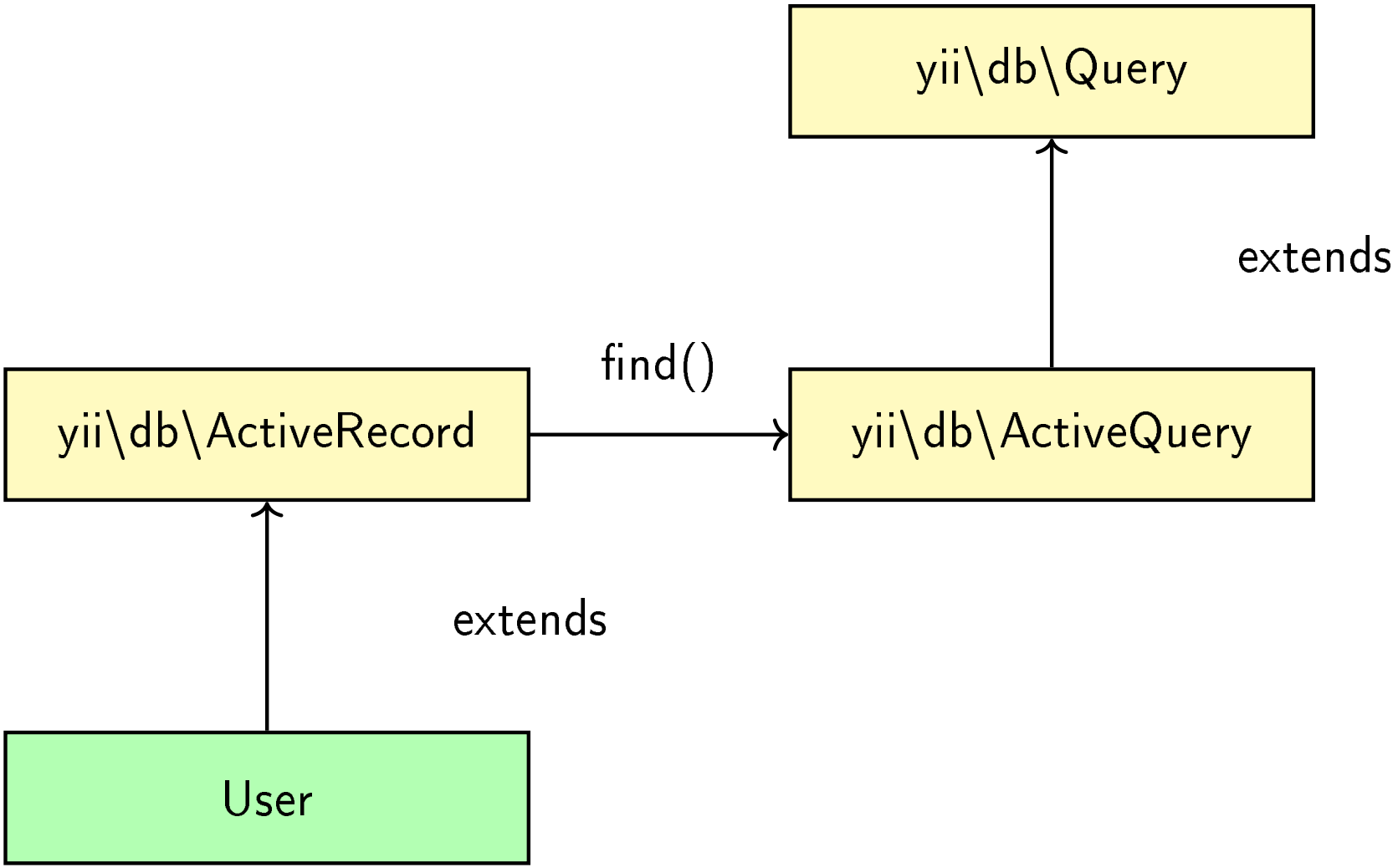
Active Record



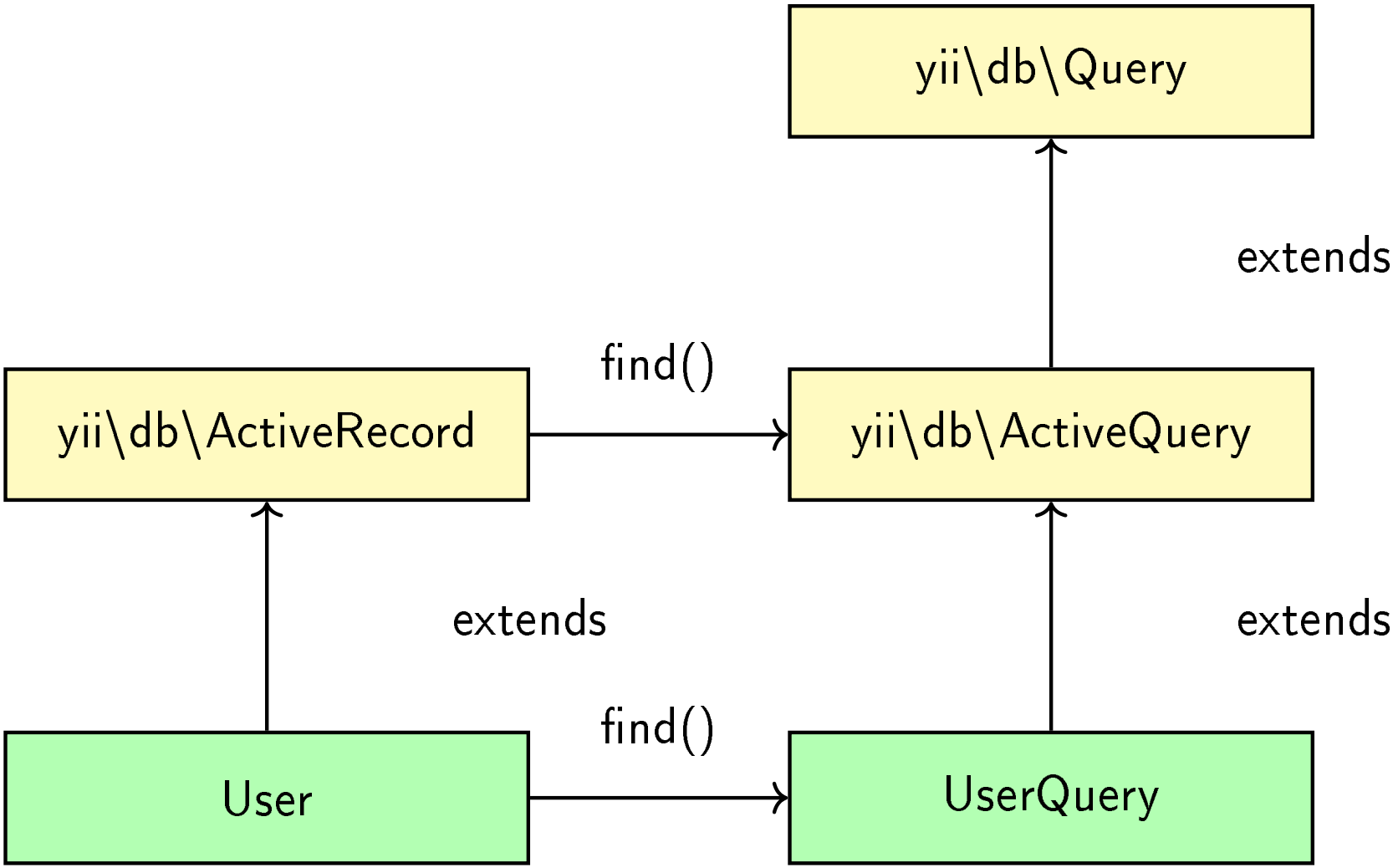
Active Record



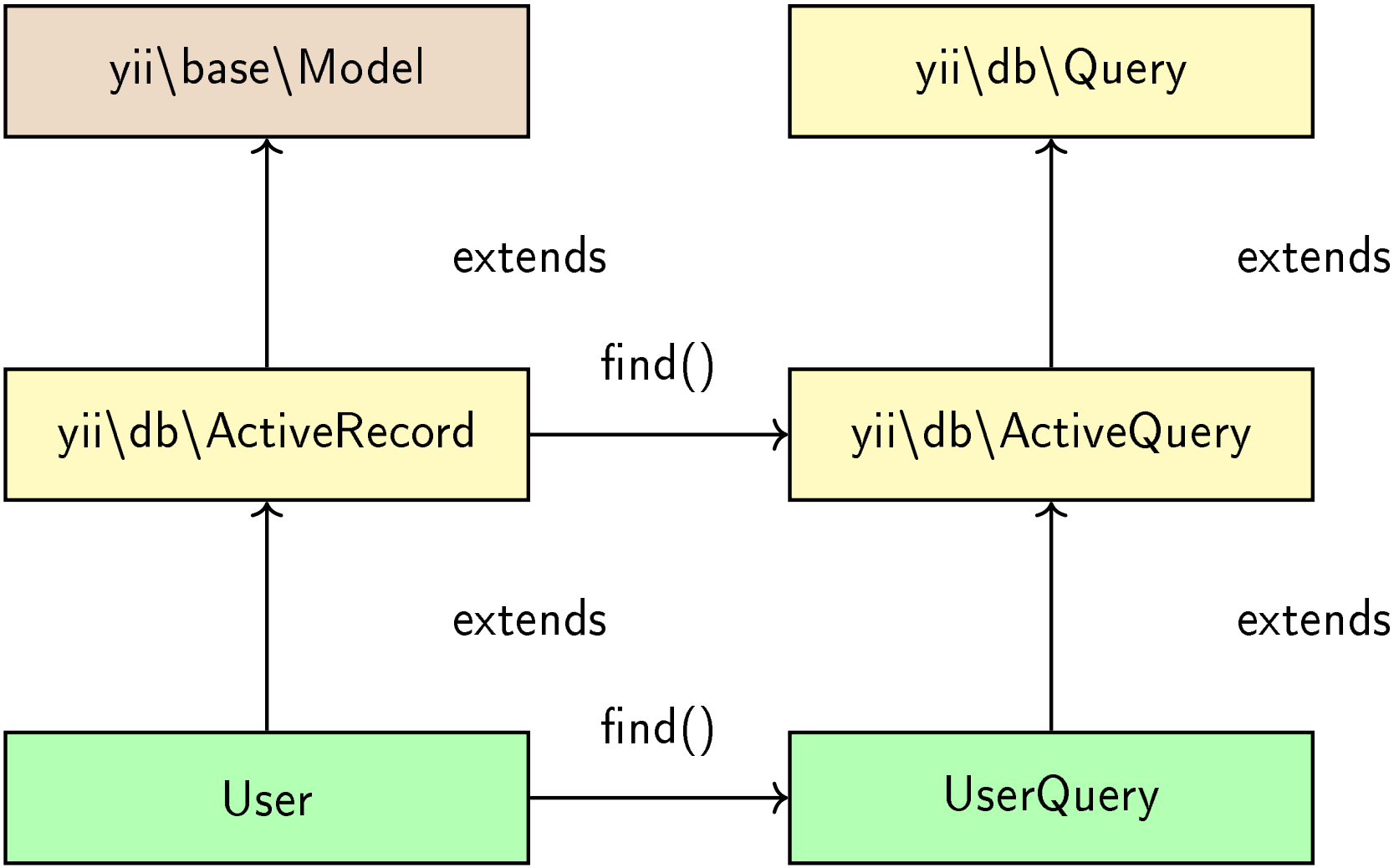
Active Record



Active Record



Active Record



Active Record

```php
$user = new User();
$user->attributes = $_POST['User'];
if ($user->save()) {
    echo "success!";
} else {
    echo "invalid data: " . print_r($user->getErrors());
}
```

Active Record

```
$user = new User();
$user->attributes = $_POST['User'];
if ($user->save()) {
    echo "success!";
} else {
    echo "invalid data: " . print_r($user->getErrors());
}
```

Active Record

```php
$user = new User();
$user->attributes = $_POST['User'];
if ($user->save()) {
    echo "success!";
} else {
    echo "invalid data: " . print_r($user->getErrors());
}
```

Active Record

```
$user = new User();
$user->attributes = $_POST['User'];
if ($user->save()) {
    echo "success!";
} else {
    echo "invalid data: " . print_r($user->getErrors());
}
```

Active Record

```
$user = new User();
$user->attributes = $_POST['User'];
if ($user->save()) {
    echo "success!";
} else {
    echo "invalid data: " . print_r($user->getErrors());
}
```

Active Record

```
$activeUsers = User::find()
        ->where(['active' => 1])
        ->orderBy('name')
        ->all();
$user = User::find()->where(['id' => 1])->one();
$user = User::findOne(1);
```

Active Record

```
$activeUsers = User::find()
        ->where(['active' => 1])
        ->orderBy('name')
        ->all();
$user = User::find()->where(['id' => 1])->one();
$user = User::findOne(1);
```

Active Record

```
$activeUsers = User::find()
        ->where(['active' => 1])
        ->orderBy('name')
        ->all();
$user = User::find()->where(['id' => 1])->one();
$user = User::findOne(1);
```

Active Record

```
$activeUsers = User::find()
        ->where(['active' => 1])
        ->orderBy('name')
        ->all();
$user = User::find()->where(['id' => 1])->one();
$user = User::findOne(1);
```

Active Record

```
$activeUsers = User::find()
        ->where(['active' => 1])
        ->orderBy('name')
        ->all();
$user = User::find()->where(['id' => 1])->one();
$user = User::findOne(1);
```

Active Record

```
$activeUsers = User::find()
        ->where(['active' => 1])
        ->orderBy('name')
        ->all();
$user = User::find()->where(['id' => 1])->one();
$user = User::findOne(1);
```

Active Record Relations

```php
class User extends yii\db\ActiveRecord {
    // ...
    public function getProfile() {
        return $this->hasOne(Profile::class, ['user_id' => 'id']);
    }
    public function getPosts() {
        return $this->hasMany(Post::class, ['user_id' => 'id']);
    }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```
class User extends yii\db\ActiveRecord {
  // ...
  public function getProfile() {
    return $this->hasOne(Profile::class, ['user_id' => 'id']);
  }
  public function getPosts() {
    return $this->hasMany(Post::class, ['user_id' => 'id']);
  }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```
class User extends yii\db\ActiveRecord {
    // ...
    public function getProfile() {
        return $this->hasOne(Profile::class, ['user_id' => 'id']);
    }
    public function getPosts() {
        return $this->hasMany(Post::class, ['user_id' => 'id']);
    }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```php
class User extends yii\db\ActiveRecord {
  // ...
  public function getProfile() {
    return $this->hasOne(Profile::class, ['user_id' => 'id']);
  }
  public function getPosts() {
    return $this->hasMany(Post::class, ['user_id' => 'id']);
  }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```
class User extends yii\db\ActiveRecord {
  // ...
  public function getProfile() {
    return $this->hasOne(Profile::class, ['user_id' => 'id']);
  }
  public function getPosts() {
    return $this->hasMany(Post::class, ['user_id' => 'id']);
  }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```
class User extends yii\db\ActiveRecord {
  // ...
  public function getProfile() {
    return $this->hasOne(Profile::class, ['user_id' => 'id']);
  }
  public function getPosts() {
    return $this->hasMany(Post::class, ['user_id' => 'id']);
  }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```
class User extends yii\db\ActiveRecord {
  // ...
  public function getProfile() {
    return $this->hasOne(Profile::class, ['user_id' => 'id']);
  }
  public function getPosts() {
    return $this->hasMany(Post::class, ['user_id' => 'id']);
  }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```
class User extends yii\db\ActiveRecord {
  // ...
  public function getProfile() {
    return $this->hasOne(Profile::class, ['user_id' => 'id']);
  }
  public function getPosts() {
    return $this->hasMany(Post::class, ['user_id' => 'id']);
  }
}
// SELECT * FROM users WHERE id = 1;
$user = User::findOne(1);
// SELECT * FROM profile WHERE user_id = 1;
$profile = $user->profile; // Profile
// SELECT * FROM posts WHERE user_id = 1;
$posts = $user->posts; // Post[]
$postQuery = $user->getPosts()->orderBy('created_at');
// SELECT * FROM posts WHERE user_id = 1 ORDER BY created_at;
$posts = $postQuery->all(); // Post[]
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record Relations

```
// SELECT * FROM users;
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()->with(['profile'])->all();
foreach($users as $user) {
        echo $user->profile->name;
}

// SELECT * FROM users
//   JOIN profiles AS p
//   WHERE p.status = 'featured';
// SELECT * FROM profiles WHERE id IN (...);
$users = User::find()
        ->joinWith(['profile AS p'])
        ->where(['p.status' => 'featured'])
        ->all();
```

Active Record NoSQL

No JOINs?

Active Record NoSQL

No JOINs? → NoSQL!

Active Record NoSQL

No JOINs? → NoSQL!


https://github.com/yiisoft/yii2-elasticsearch
https://github.com/yiisoft/yii2-mongodb
https://github.com/yiisoft/yii2-redis
https://github.com/yiisoft/yii2-sphinx

Active Record NoSQL

yii\db\ActiveRecord**Interface**

Active Record NoSQL

yii**elasticsearch**\ActiveRecord

yii\db\ActiveRecord**Interface**

Active Record NoSQL

yii**elasticsearch**\ActiveRecord

yii**redis**\ActiveRecord

yii\db\ActiveRecord**Interface**

Active Record NoSQL

yii**elasticsearch**\ActiveRecord

yii**redis**\ActiveRecord

yii\db\ActiveRecord**Interface**

yii**mongodb**\ActiveRecord

Active Record NoSQL

yii**elasticsearch**\ActiveRecord

yii**redis**\ActiveRecord

yii\db\ActiveRecord**Interface**

yii**mongodb**\ActiveRecord

yii**sphinx**\ActiveRecord

Active Record NoSQL

- ActiveRecord and ActiveQuery interfaces

Active Record NoSQL

- ActiveRecord and ActiveQuery interfaces
- common usage, select() where() orderBy() limit()

Active Record NoSQL

- ActiveRecord and ActiveQuery interfaces
- common usage, select() where() orderBy() limit()
- used by DataProvider

Active Record NoSQL

- ActiveRecord and ActiveQuery interfaces
- common usage, select() where() orderBy() limit()
- used by DataProvider
- Relation between SQL and NoSQL

Switching between layers

- `findBySql("SELECT ... WHERE ...")`

Switching between layers

- `findBySql("SELECT ... WHERE ...")`
- `find()->asArray()->all()`

Switching between layers

- `findBySql("SELECT ... WHERE ...")`
- `find()->asArray()->all()`
- `yii\db\Expression("plain SQL")`

Switching between layers

- `findBySql("SELECT ... WHERE ...")`
- `find()->asArray()->all()`
- `yii\db\Expression("plain SQL")`

Yii philosophy: maximum flexibilty.

Conclusion / Questions?

- DAO,

Conclusion / Questions?

- DAO, QueryBuilder,

Conclusion / Questions?

- DAO, QueryBuilder, ActiveRecord
- Simple to start with,

Conclusion / Questions?

- DAO, QueryBuilder, ActiveRecord
- Simple to start with, but not limited in complex situations.

Conclusion / Questions?

- DAO, QueryBuilder, ActiveRecord
- Simple to start with, but not limited in complex situations.
- SQL,

Conclusion / Questions?

- DAO, QueryBuilder, ActiveRecord
- Simple to start with, but not limited in complex situations.
- SQL, and NoSQL

Conclusion / Questions?

- DAO, QueryBuilder, ActiveRecord
- Simple to start with, but not limited in complex situations.
- SQL, and NoSQL

Conclusion / Questions?

- DAO, QueryBuilder, ActiveRecord
- Simple to start with, but not limited in complex situations.
- SQL, and NoSQL

Contact:
@cebe on Github
@cebe_cc on Twitter
mail@cebe.cc via Email

https://github.com/yiisoft/yii2/issues