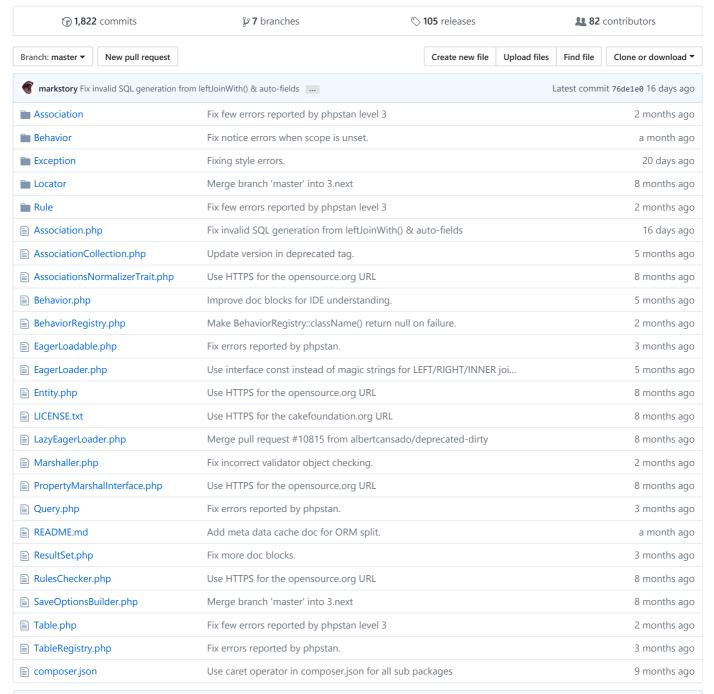
#### □ cakephp / orm

[READ-ONLY] A flexible, lightweight and powerful Object-Relational Mapper for PHP, implemented using the DataMapper pattern. This repo is a split of the main code that can be found in https://github.com/cakephp/cakephp



#### README.md

downloads 27k license MIT

# CakePHP ORM

The CakePHP ORM provides a powerful and flexible way to work with relational databases. Using a datamapper pattern the ORM allows you to manipulate data as entities allowing you to create expressive domain layers in your applications.

# Database engines supported

The CakePHP ORM is compatible with:

- MySQL 5.1+
- Postgres 8+
- SQLite3
- SQLServer 2008+
- Oracle (through a community plugin)

#### Connecting to the Database

The first thing you need to do when using this library is register a connection object. Before performing any operations with the connection, you need to specify a driver to use:

Once a 'default' connection is registered, it will be used by all the Table mappers if no explicit connection is defined.

### **Creating Associations**

In your table classes you can define the relations between your tables. CakePHP's ORM supports 4 association types out of the box:

- belongsTo E.g. Many articles belong to a user.
- hasOne E.g. A user has one profile
- hasMany E.g. A user has many articles
- belongsToMany E.g. An article belongsToMany tags.

You define associations in your table's initialize() method. See the documentation for complete examples.

## **Reading Data**

Once you've defined some table classes you can read existing data in your tables:

```
use Cake\ORM\TableRegistry;

$articles = TableRegistry::get('Articles');
foreach ($articles->find() as $article) {
        echo $article->title;
}
```

You can use the query builder to create complex queries, and a variety of methods to access your data.

#### Saving Data

Table objects provide ways to convert request data into entities, and then persist those entities to the database:

The above shows how you can easily marshal and save an entity and its associations in a simple & powerful way. Consult the ORM documentation for more in-depth examples.

# **Deleting Data**

Once you have a reference to an entity, you can use it to delete data:

```
$articles = TableRegistry::get('Articles');
$article = $articles->get(2);
$articles->delete($article);
```

#### Meta Data Cache

It is recommended to enable meta data cache for production systems to avoid performance issues. For e.g. file system strategy your bootstrap file could look like this:

```
use Cake\Cache\Engine\FileEngine;

$cacheConfig = [
    'className' => FileEngine::class,
    'duration' => '+1 year',
    'serialize' => true,
    'prefix' => 'orm_',
],
Cache::setConfig('_cake_model_', $cacheConfig);
```

#### **Additional Documentation**

Consult the CakePHP ORM documentation for more in-depth documentation.