Doctrine Lazy Loading

Did you know?

Did you know that in Doctrine if you don't specifically select a property or join a relationship in your DQL query, you can lazily load that data? Here is a simple example to demonstrate.

```
<?php
$q = Doctrine::getTable('User')
  ->createQuery('u')
  ->select('u.id, u.username')
  ->where('u.username = ?', 'jwage');

$user = $q->fetchOne();
```

Notice how I only selected the id and username, but we have an additional column named email_address. If I were to access that property it would lazily load the data.

```
<?php
echo $user['email_address']

**NOTE** The above code would result in an additional query to the
database that would lazy load all unloaded properties of an object
in proxy state. A record is in proxy state when it has one or more
properties that have not been loaded from the database.
```

Be aware

This can be both a curse and a blessing because if you lazily load many properties and relationships this means you will have a high query count per page. In Doctrine 1.x lazy loading is on by default so it is often overly used. The problem is people aren't even aware that it is happening.

More Examples

Imagine you had a User and Profile model where User has one Profile and Profile has one User, a simple one-to-one relationship. If I were to write a DQL query where I retrieve an individual user record like the following.

```
<?php
$q = Doctrine::getTable('User')
  ->createQuery('u')
  ->where('u.username = ?', 'jwage');

$user = $q->fetchOne();
```

But then I wanted to retrieve the Profile for that User I could do so by simply accessing the Profile property onUser.

```
<?php
$profile = $user->Profile;
```

The above code would result in an additional query to the database that loads the Profile record for the User in question. Now if I were to be rendering a list of twenty User objects and I lazily load the profile of each user that means I would have 21 queries total! That is too many! In that case it would be smart to leftJoin() theProfile object like the following.

```
<?php
$q = Doctrine::getTable('User')
  ->createQuery('u')
  ->leftJoin('u.Profile p');

$users = $q->execute();
```

Now when I access the Profile relationship of an individual object the data would already be present and no lazy loading would be required.

Why?

query counts per page.

I've seen this many times and the blame is often put on Doctrine but really it is the tool not being used properly. So, be aware of when you are lazy loading properties and the total number of queries you execute per page. Make sure you always join required relationships and only select the properties you need to access. It doesn't make sense to load data if you're not going to be using it.

How?

If you need help with keeping track of how many queries you have per page, frameworks like Symfony andZend Framework give you debug tools to show you how many queries you have per page. Or of course you can always use the Profiling tool built in to Doctrine to log the queries in your application and keep track of it that way.