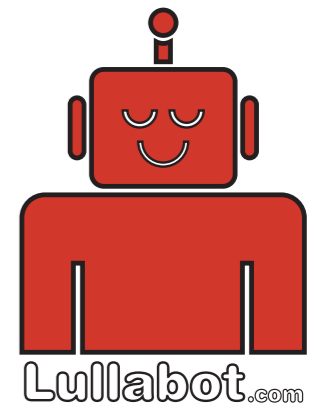


Demystifying Multisite Architecture

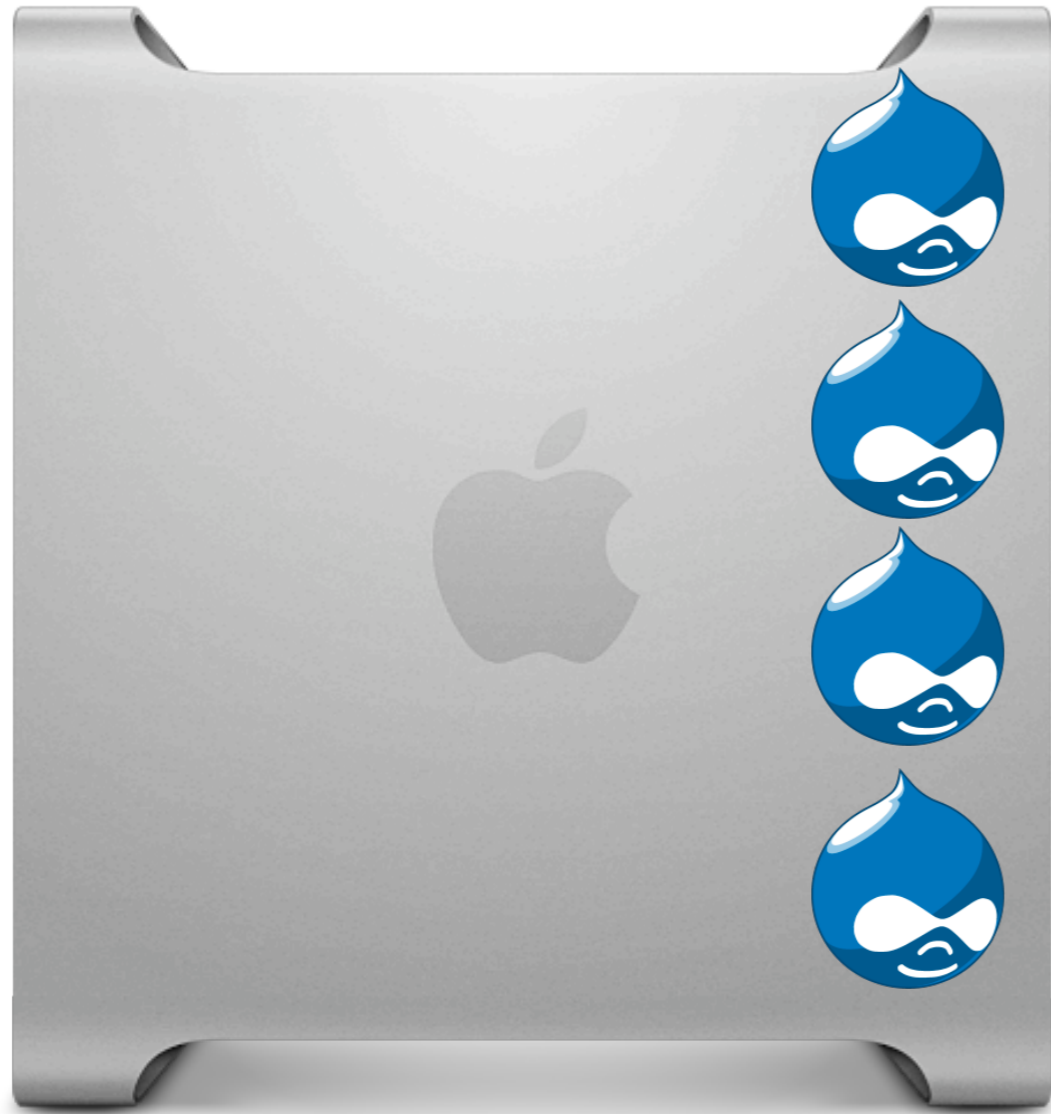
John VanDyk and James Walker
Do It With Drupal, December 2008



What is Multisite?

How Multisite Works

Sharing Information Between Sites



<http://www.example.com>

<http://banana.example.com>

<http://www.example.com/apple>

<http://pancakeblog.net>



<http://www.example.com>

<http://banana.example.com>

<http://www.example.com/apple>

<http://pancakeblog.net>

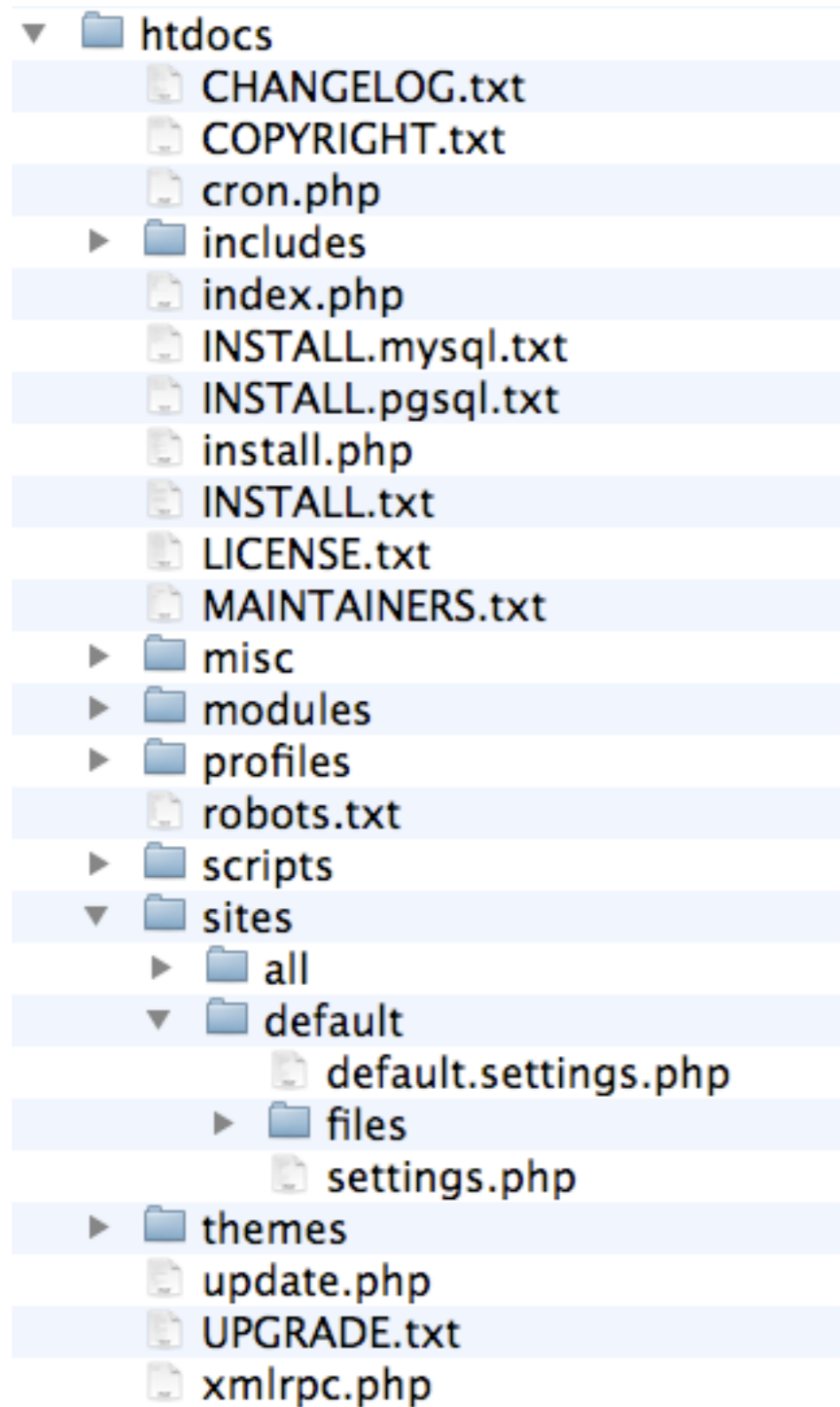
Why?

- Only one copy of the codebase

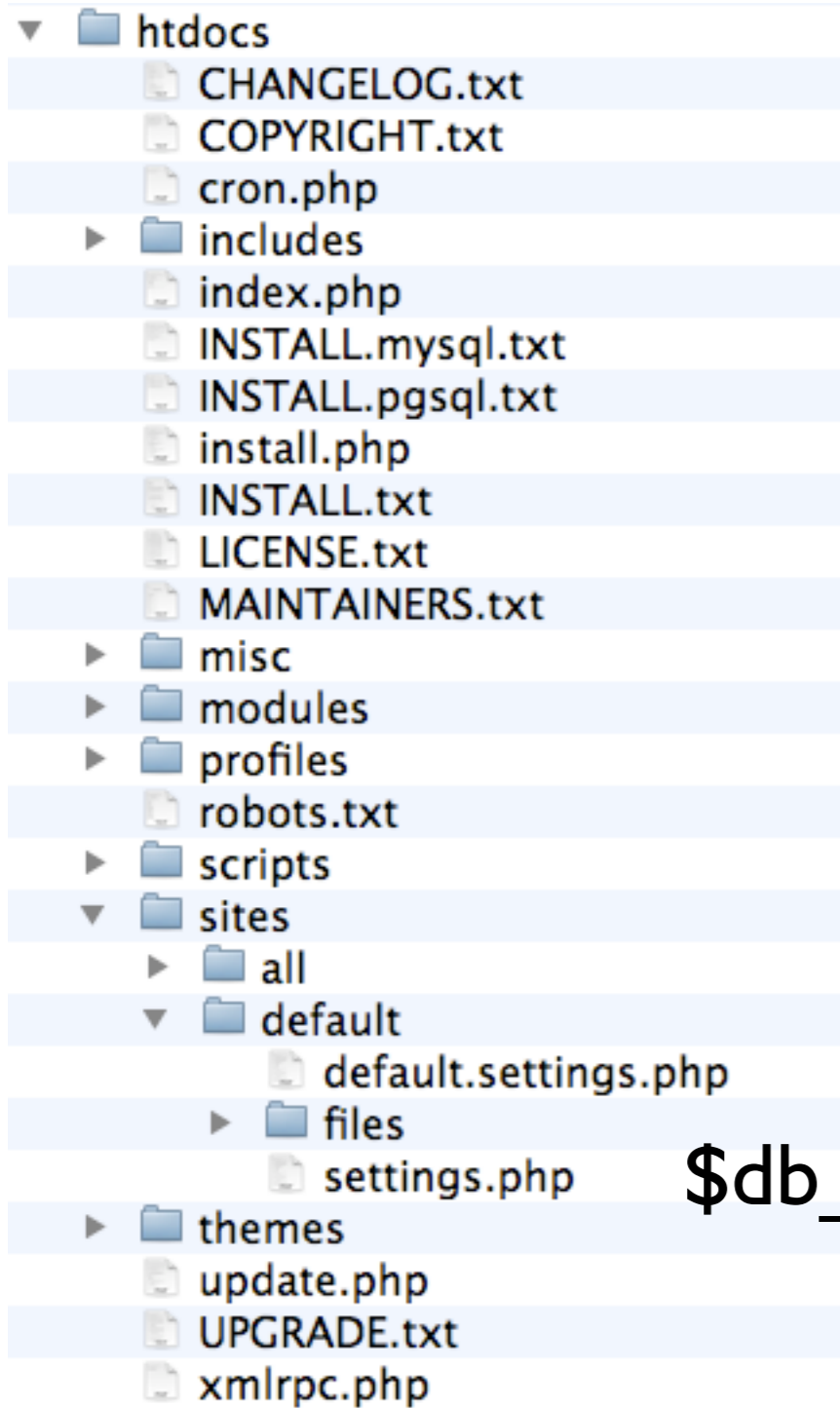
- Only one copy of the codebase
- Only one codebase to upgrade

- Only one copy of the codebase
- Only one codebase to upgrade
- Only one set of files to debug

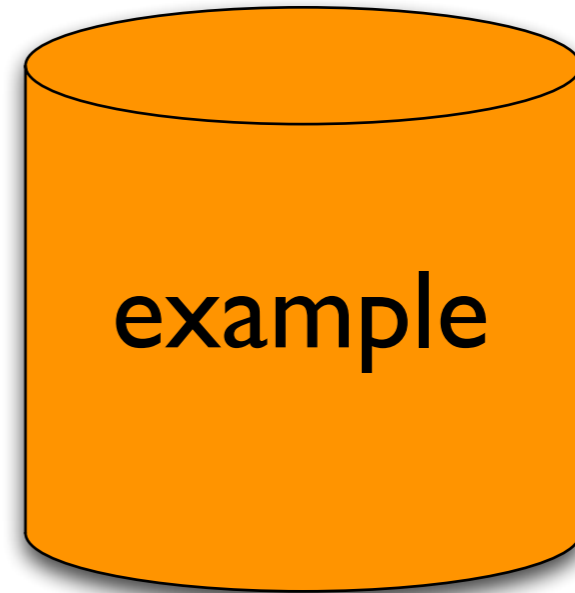
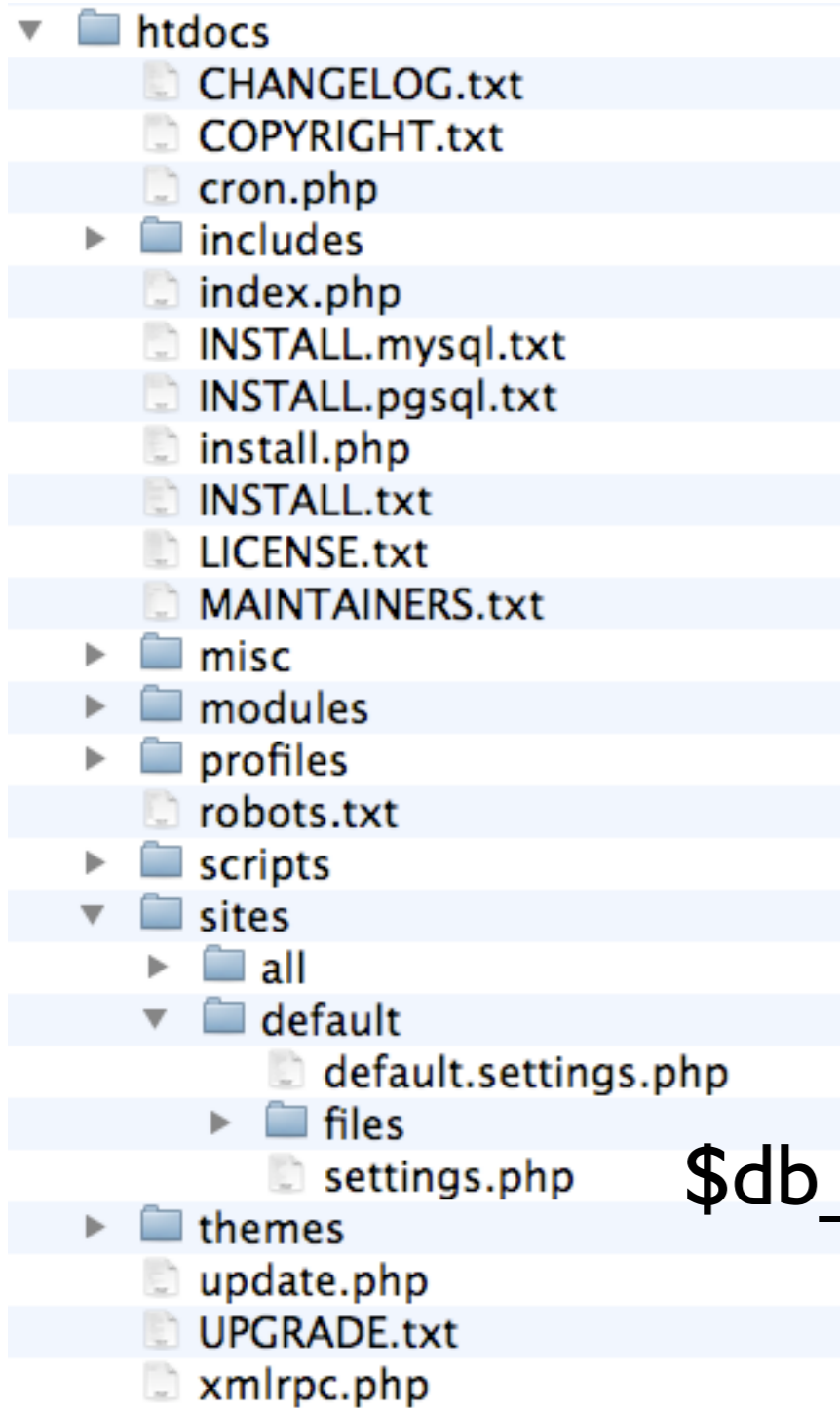
- Only one copy of the codebase
- Only one codebase to upgrade
- Only one set of files to debug
- Still have flexibility for site-specific modules



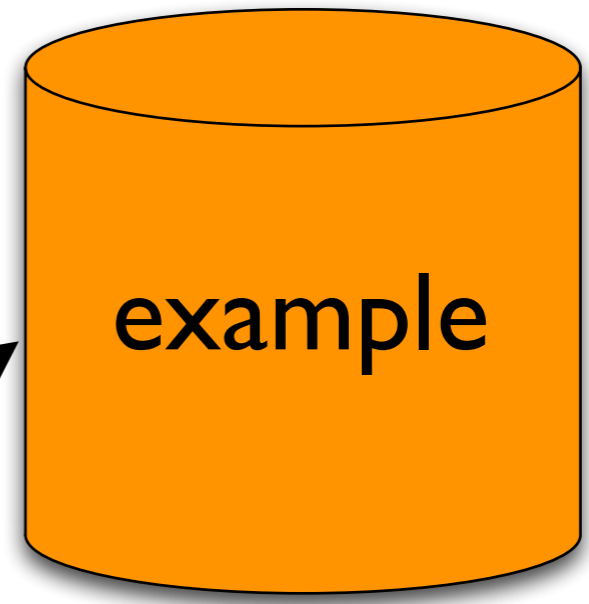
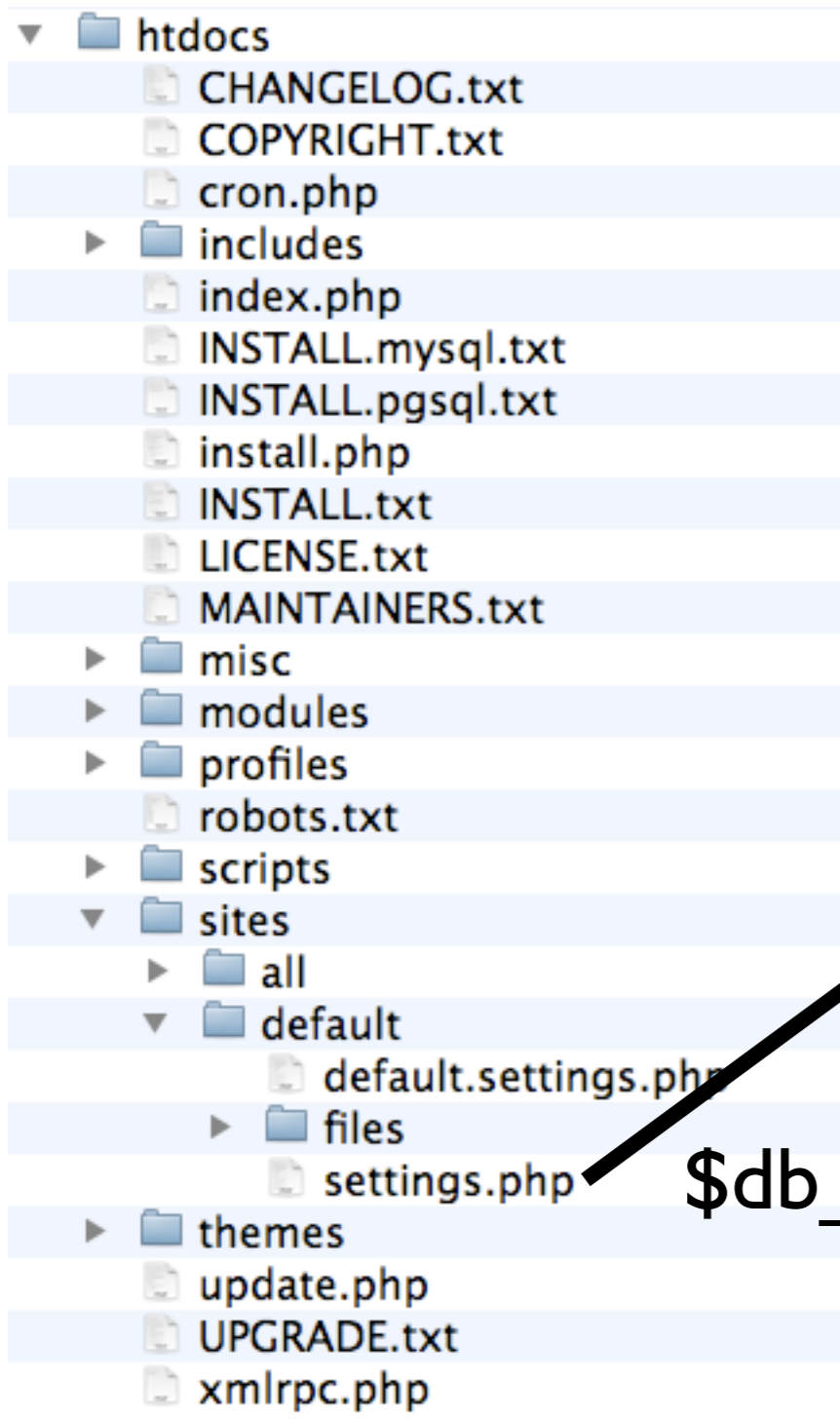
Drupal has three essential components.
The filesystem contains the files that make up the Drupal codebase.



```
$db_url = 'mysql://joe:secret@localhost/example';
```

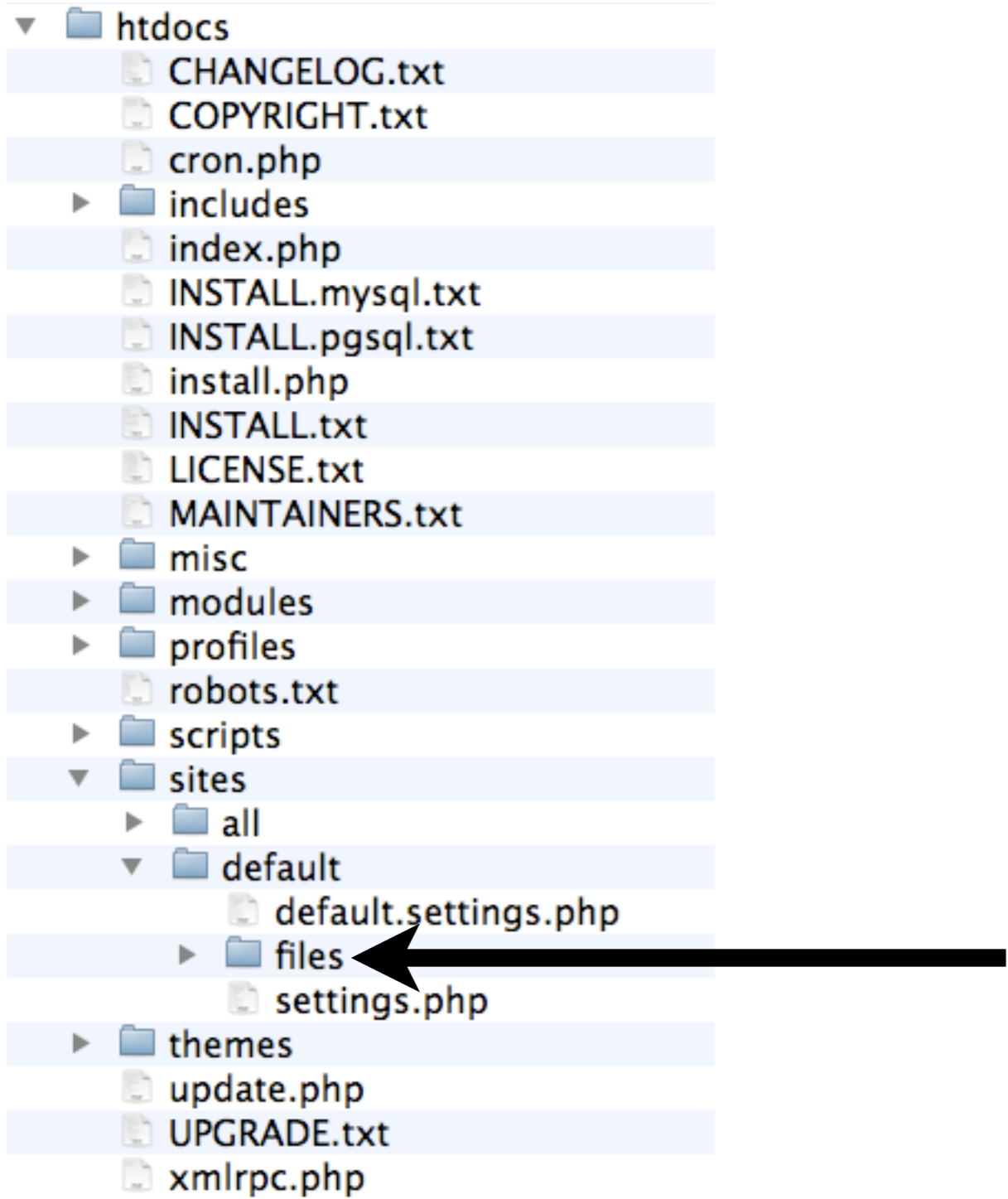


```
$db_url = 'mysql://joe:secret@localhost/example';
```

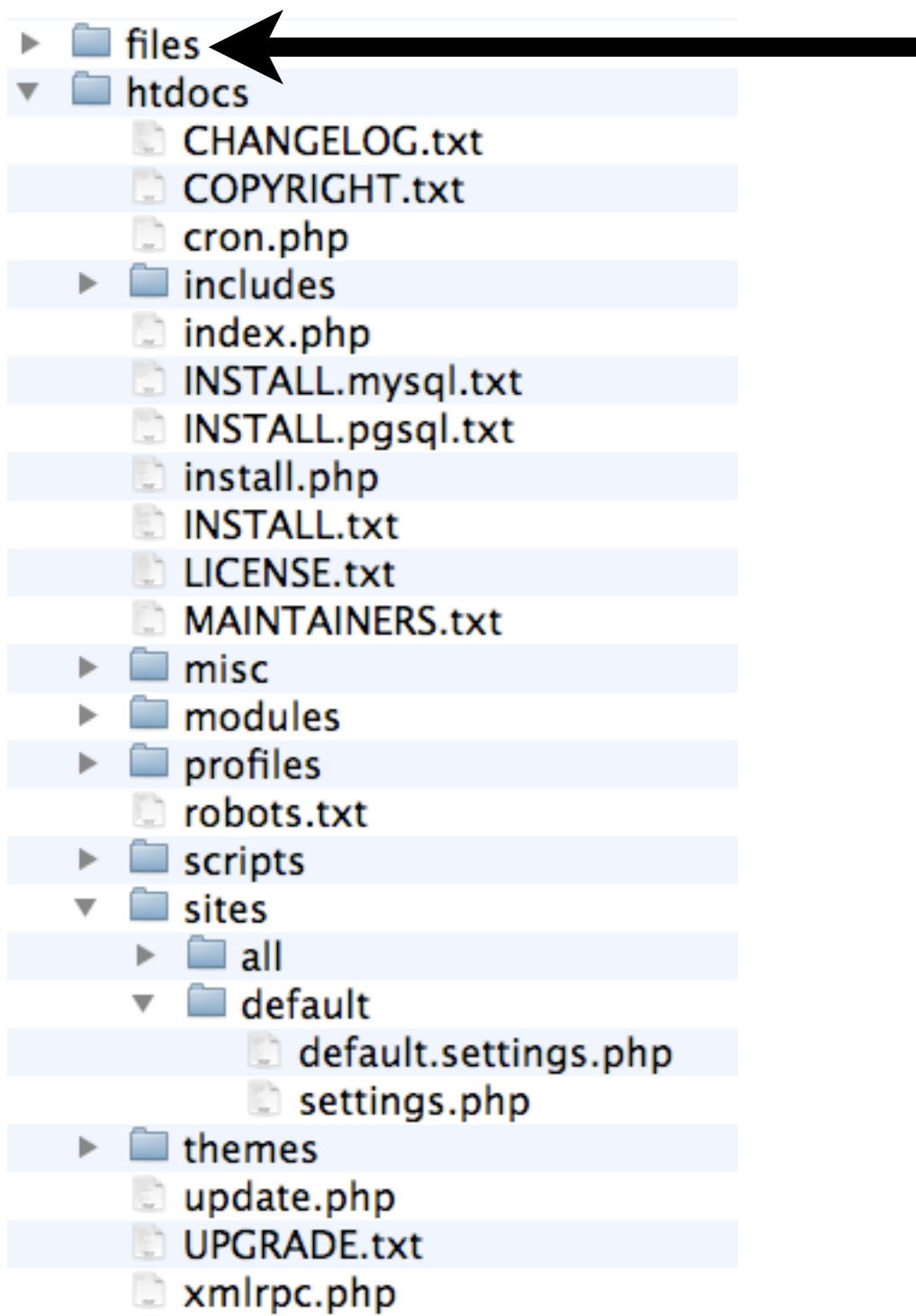


`$db_url = 'mysql://joe:secret@localhost/example';`

The database contains content and most configuration settings.

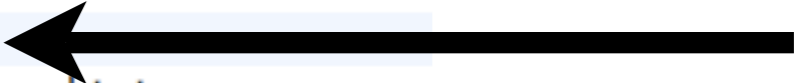


The files directory contains uploaded files.



If using private files, the files directory can be outside of the web root.

- ▼ htdocs
 - CHANGELOG.txt
 - COPYRIGHT.txt
 - cron.php
 - ▶ includes
 - index.php
 - INSTALL.mysql.txt
 - INSTALL.pgsql.txt
 - install.php
 - INSTALL.txt
 - LICENSE.txt
 - MAINTAINERS.txt
 - ▶ misc
 - ▶ modules
 - ▶ profiles
 - robots.txt
 - ▶ scripts
 - ▼ sites
 - ▶ all
 - ▼ default
 - default.settings.php
 - ▶ files
 - settings.php
 - ▶ themes
 - update.php
 - UPGRADE.txt
 - xmlrpc.php



```
<?php  
require_once './includes/bootstrap.inc';  
drupal_bootstrap(FULL);
```

CONFIGURATION: initialize configuration.

EARLY PAGE CACHE

DATABASE: initialize database layer.

ACCESS: identify and reject banned hosts.

SESSION: initialize session handling.

LATE PAGE CACHE

LANGUAGE: identify the language

PATH: path handling

FULL: Drupal is fully loaded

CONFIGURATION: initialize configuration.

EARLY PAGE CACHE

DATABASE: initialize database layer.

ACCESS: identify and reject banned hosts.

SESSION: initialize session handling.

LATE PAGE CACHE

LANGUAGE: identify the language

PATH: path handling

FULL: Drupal is fully loaded

We've got a request for <http://www.example.com/>
That's from host www.example.com, path /
Let's look for a settings file at

We've got a request for <http://www.example.com/>
That's from host www.example.com, path /
Let's look for a settings file at

[sites/www.example.com/settings.php](http://www.example.com/sites/www.example.com/settings.php)

We've got a request for <http://www.example.com/>
That's from host www.example.com, path /
Let's look for a settings file at

[sites/www.example.com/settings.php](http://www.example.com/sites/www.example.com/settings.php)

[sites/example.com/settings.php](http://www.example.com/sites/example.com/settings.php)

We've got a request for <http://www.example.com/>
That's from host www.example.com, path /
Let's look for a settings file at

[sites/www.example.com/settings.php](http://www.example.com/settings.php)

[sites/example.com/settings.php](http://example.com/settings.php)

[sites/com/settings.php](http://com/settings.php)

We've got a request for <http://www.example.com/>
That's from host www.example.com, path /
Let's look for a settings file at

[sites/www.example.com/settings.php](#)

[sites/example.com/settings.php](#)

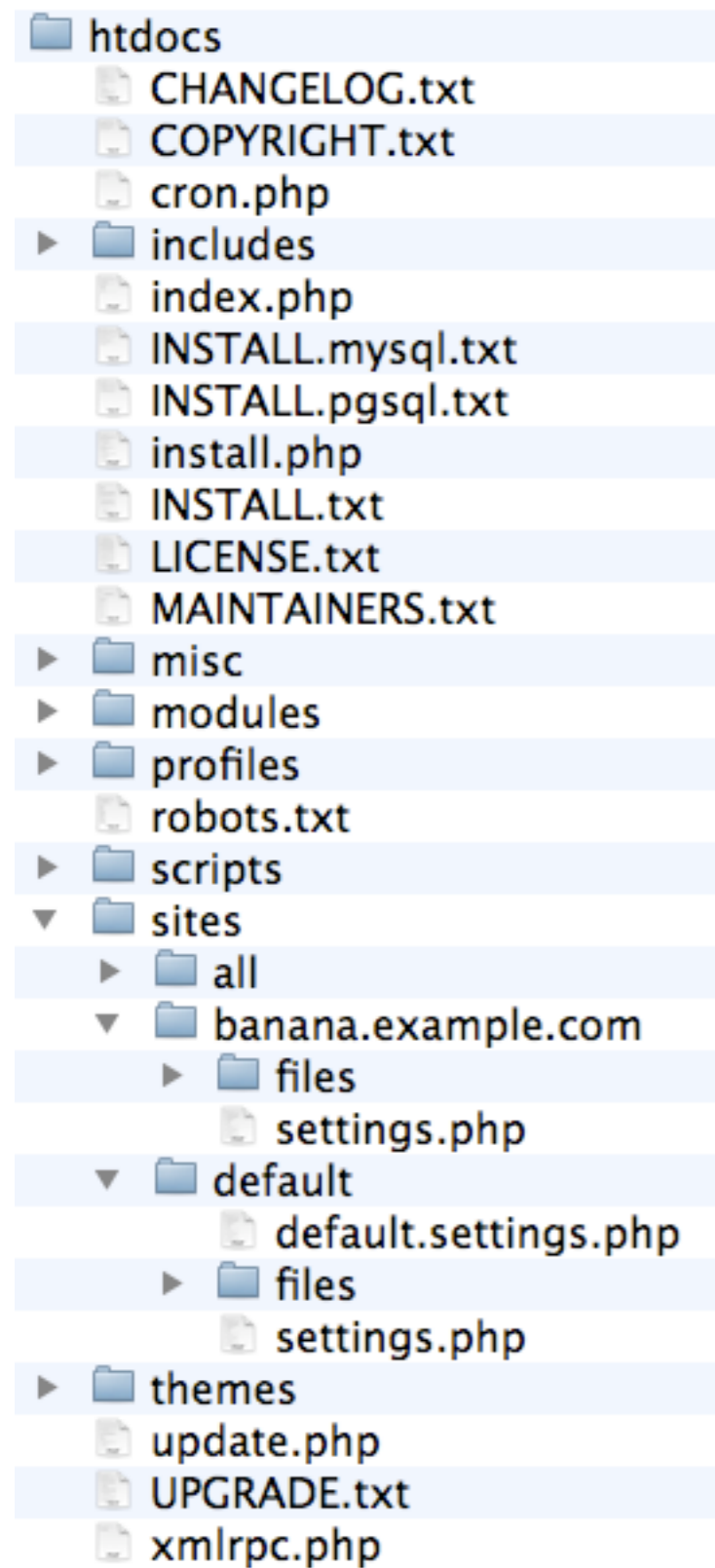
[sites/com/settings.php](#)

[sites/default/settings.php](#)



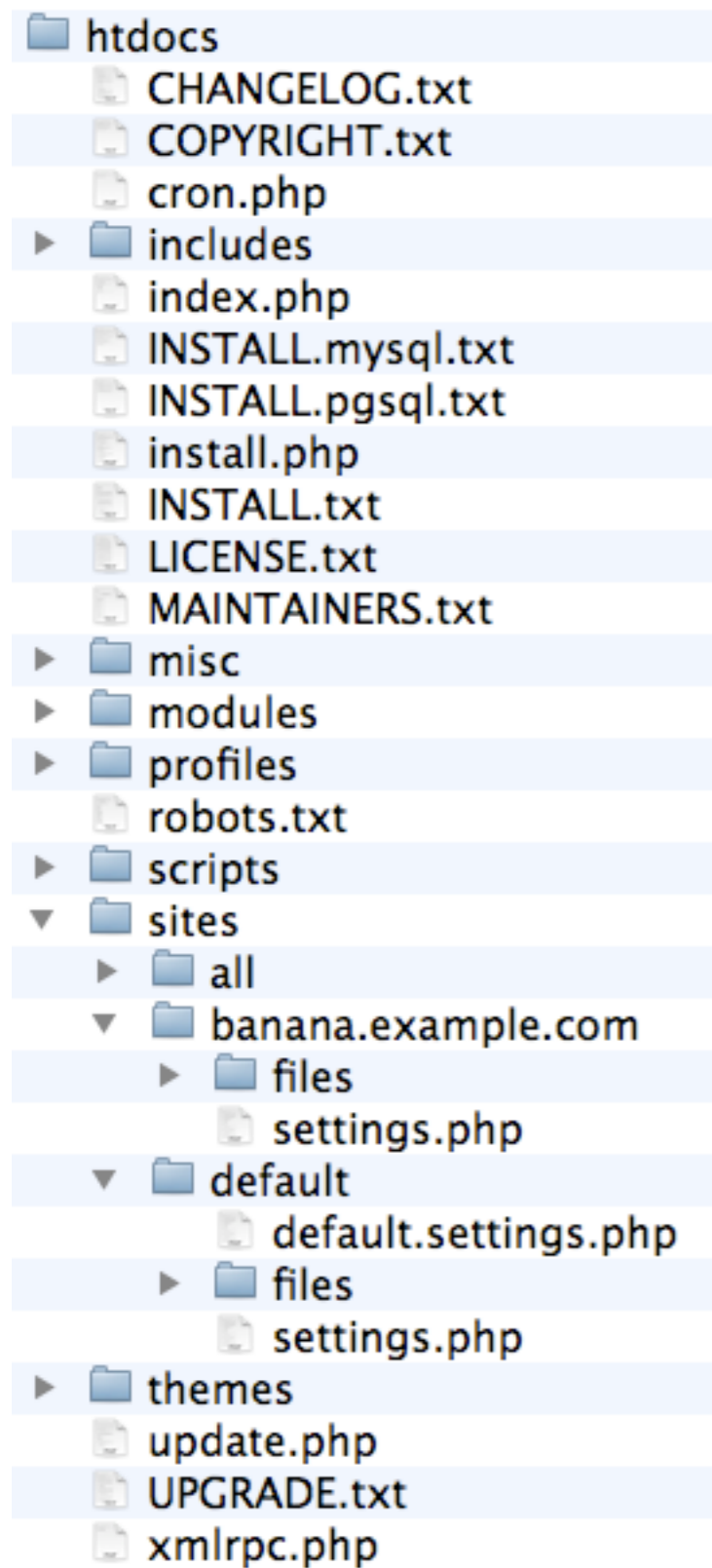
<http://www.example.com>

<http://banana.example.com>

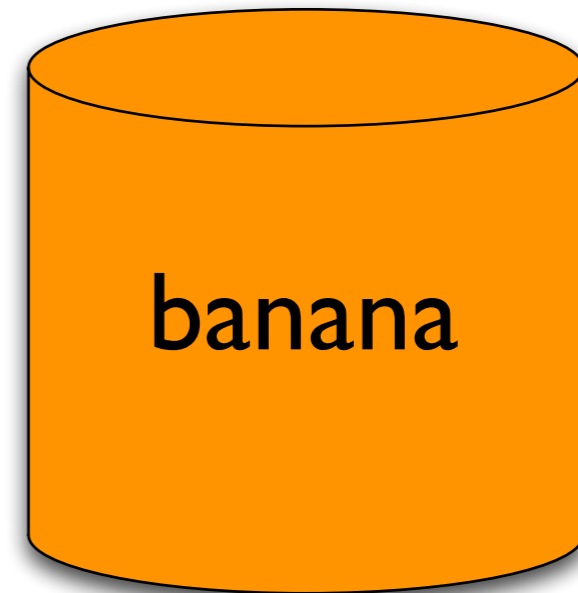


```
$db_url = 'mysql://jdoe:secret@localhost/banana';
```

```
$db_url = 'mysql://moe:shhhh@localhost/www';
```

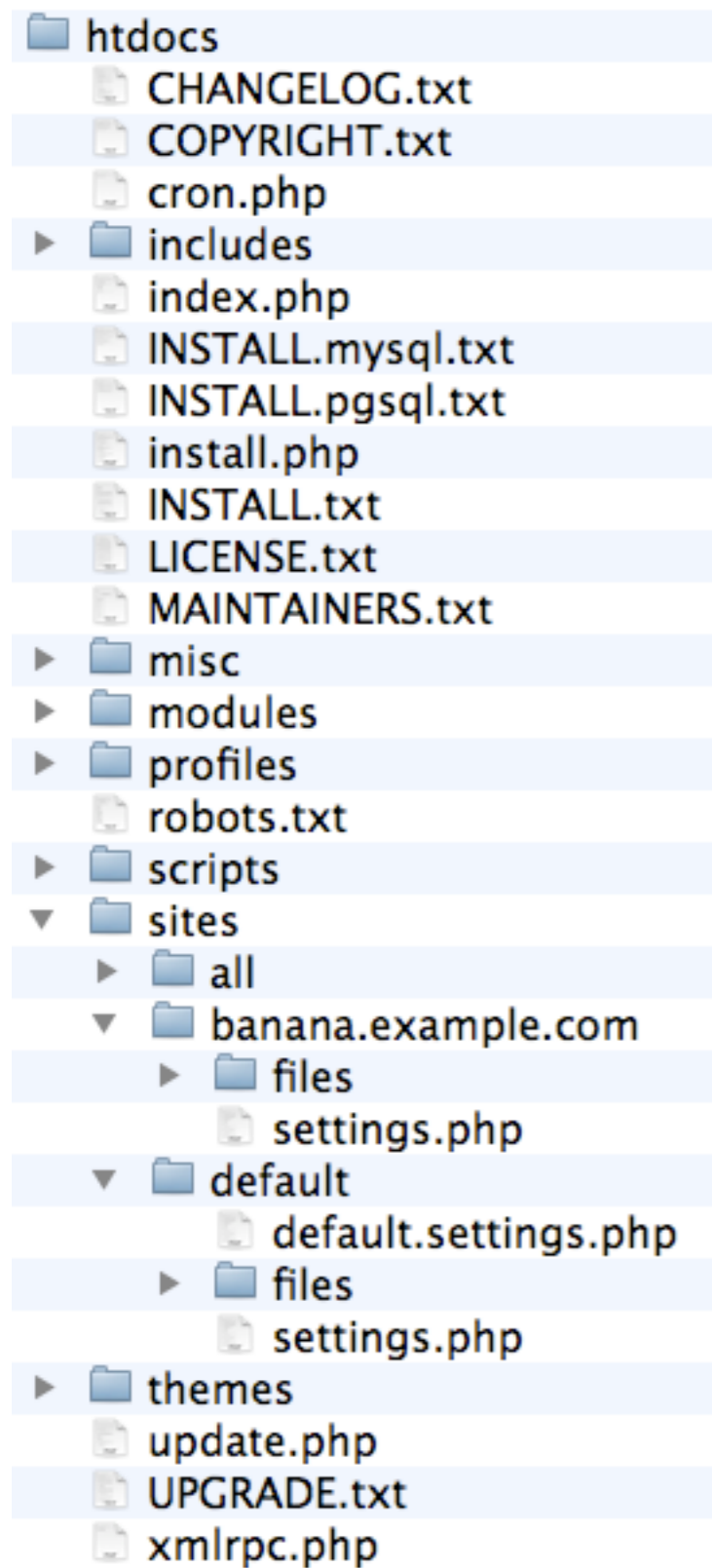


```
$db_url = 'mysql://jdoe:secret@localhost/banana';
```

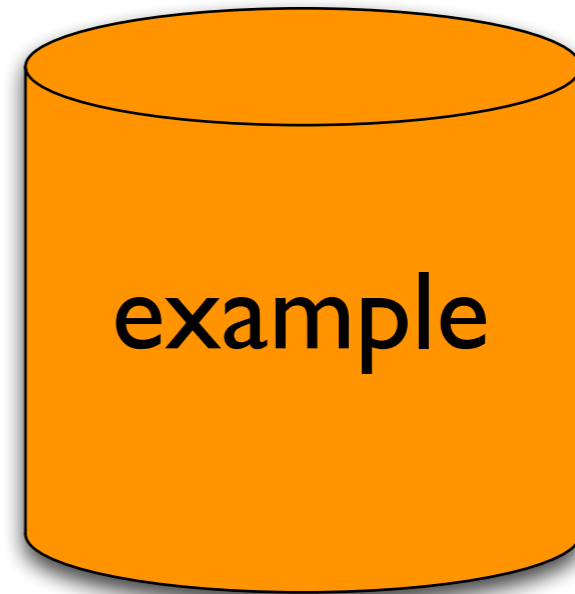
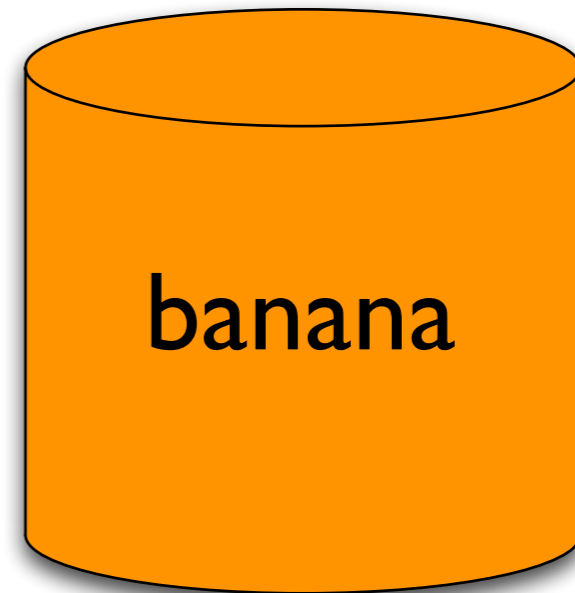


```
$db_url = 'mysql://moe:shhhh@localhost/www';
```

Now banana.example.com will get a site using the banana database; everything else gets the www database.

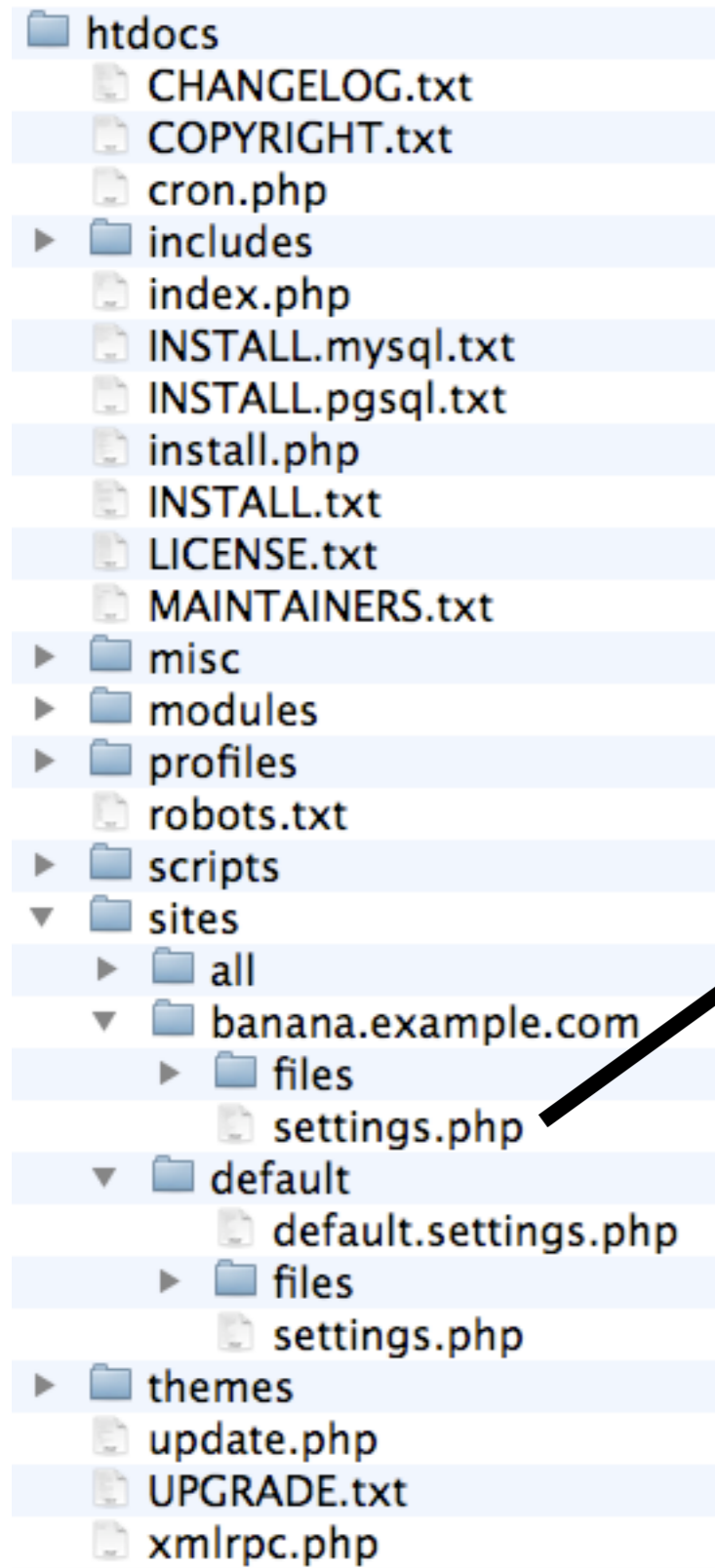


```
$db_url = 'mysql://jdoe:secret@localhost/banana';
```

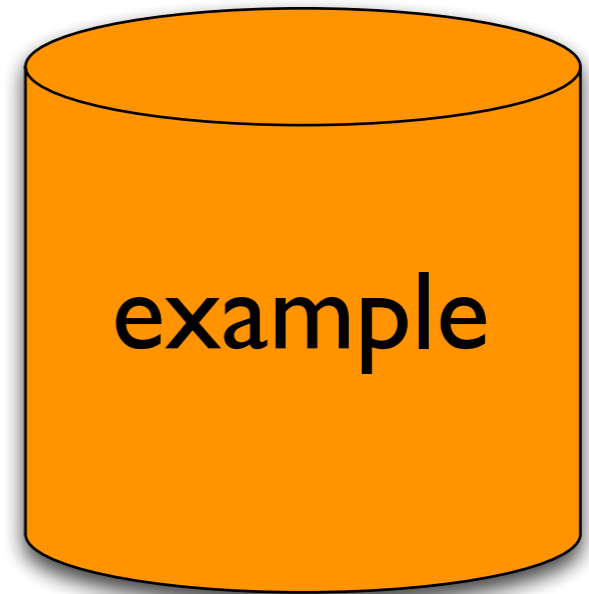
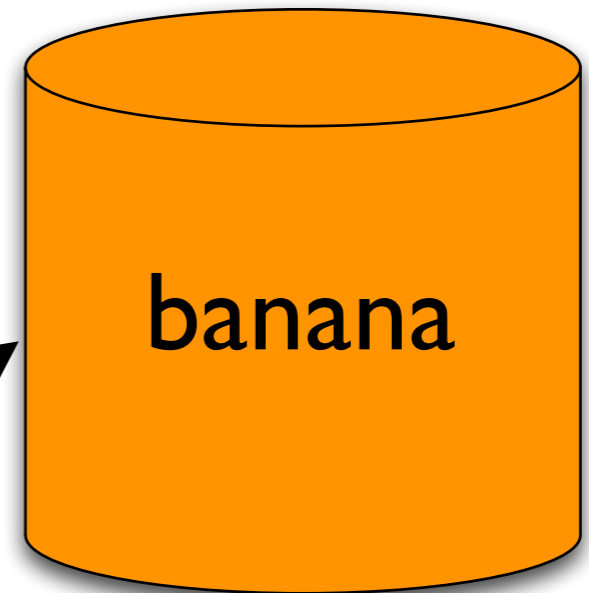


```
$db_url = 'mysql://moe:shhhh@localhost/www';
```

Now banana.example.com will get a site using the banana database; everything else gets the www database.

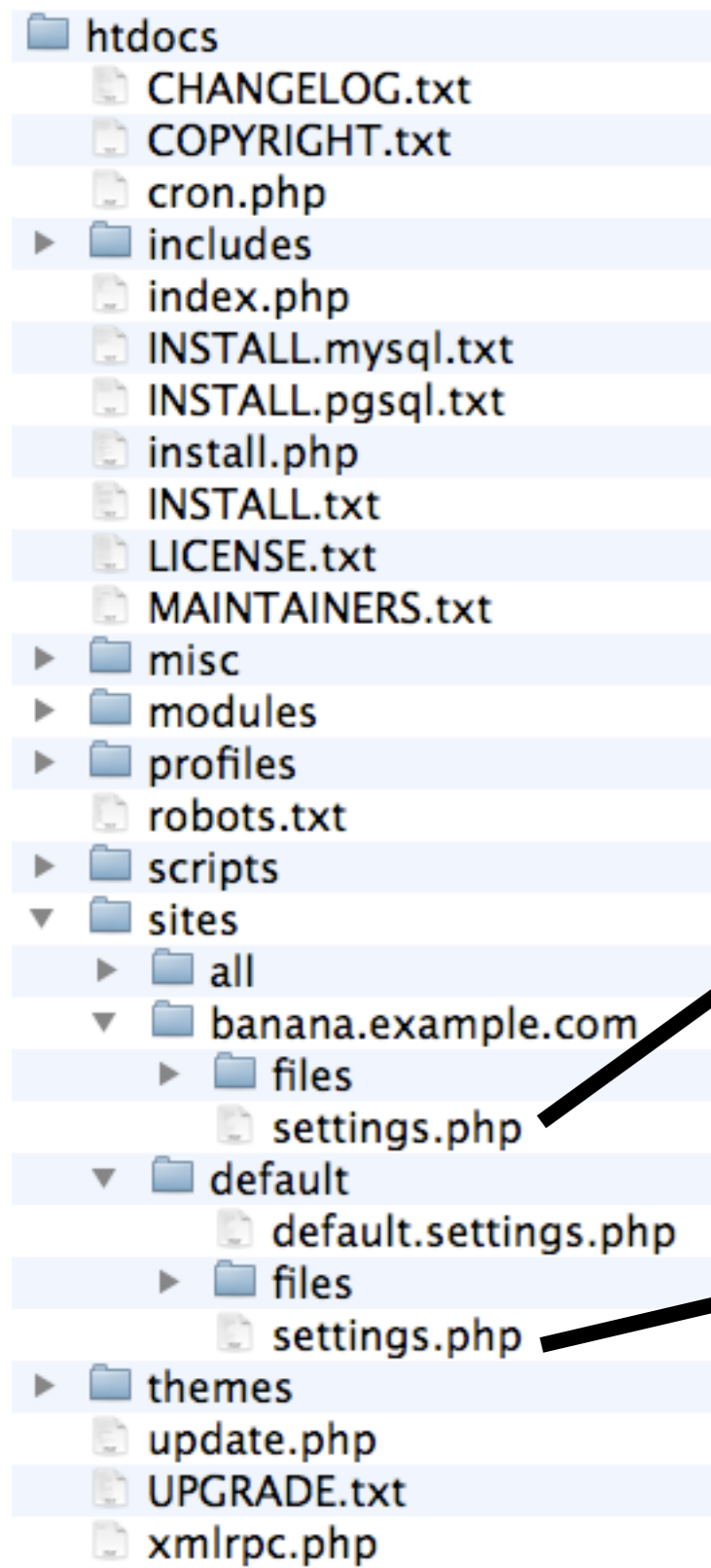


```
$db_url = 'mysql://jdoe:secret@localhost/banana';
```

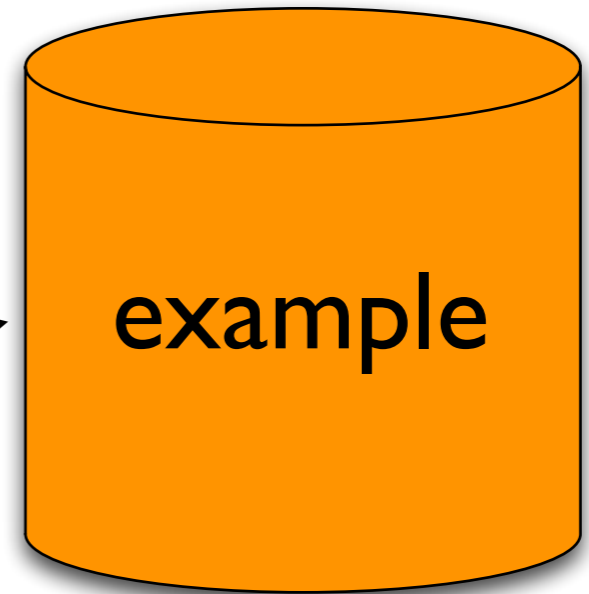
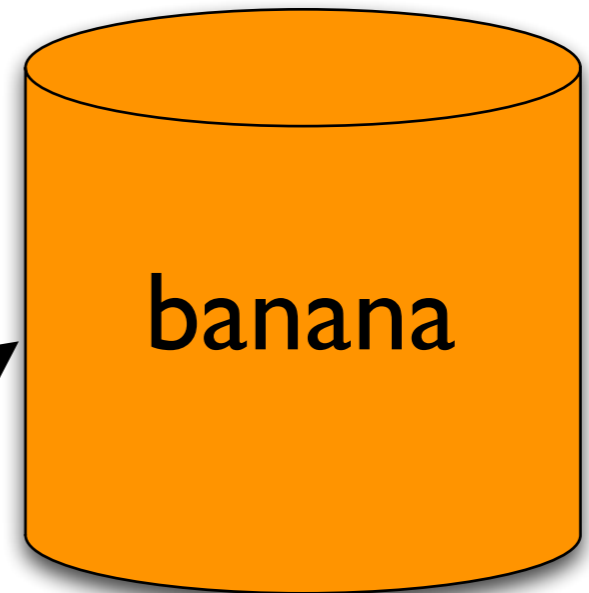


```
$db_url = 'mysql://moe:shhhh@localhost/www';
```

Now banana.example.com will get a site using the banana database; everything else gets the www database.



```
$db_url = 'mysql://jdoe:secret@localhost/banana';
```

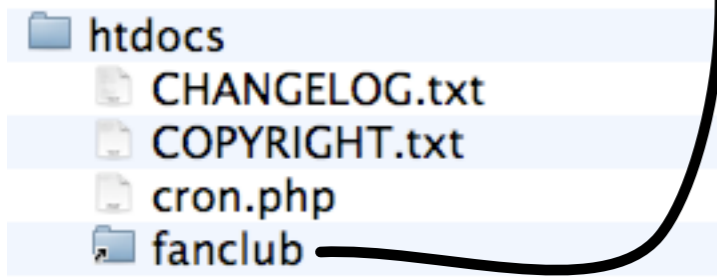


```
$db_url = 'mysql://moe:shhhh@localhost/www';
```

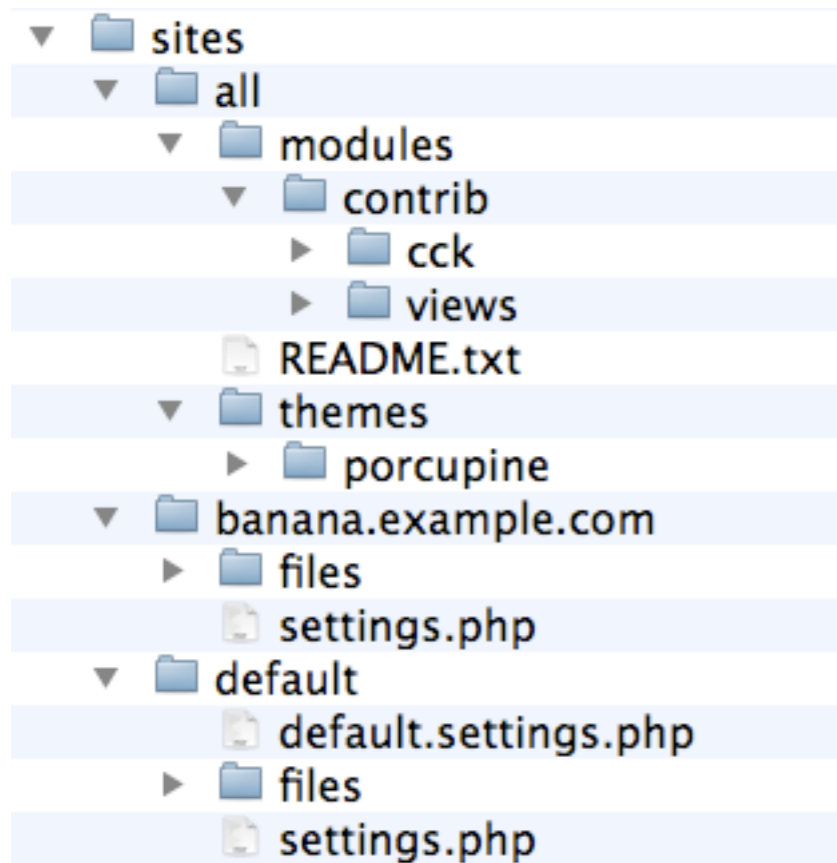
Now banana.example.com will get a site using the banana database; everything else gets the www database.

Edge case

<http://www.example.com/something>

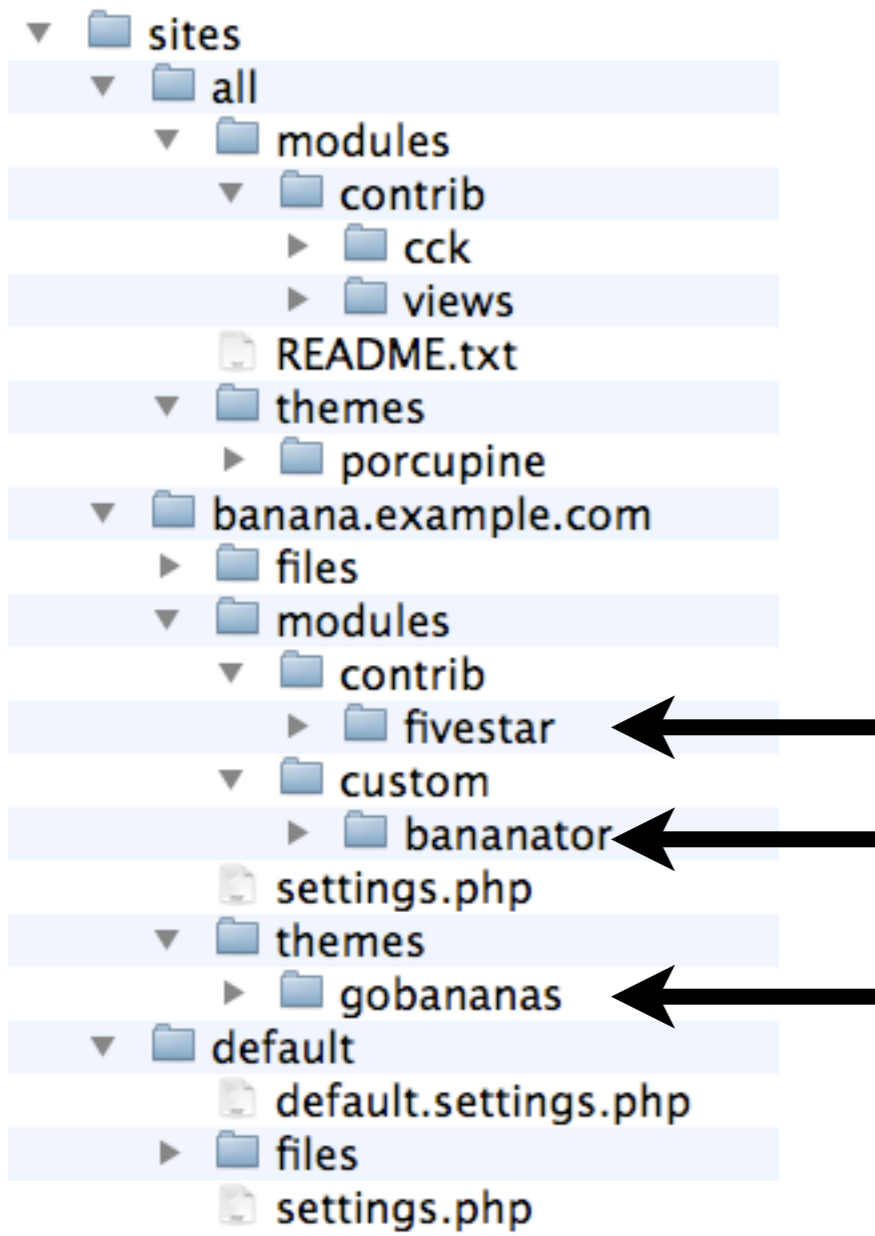
- 
- htdocs
 - CHANGELOG.txt
 - COPYRIGHT.txt
 - cron.php
 - fanclub
 - includes
 - index.php
 - INSTALL.mysql.txt
 - INSTALL.pgsql.txt
 - install.php
 - INSTALL.txt
 - LICENSE.txt
 - MAINTAINERS.txt
 - misc
 - modules
 - profiles
 - robots.txt
 - scripts
 - sites
 - all
 - banana.example.com
 - files
 - modules
 - settings.php
 - themes
 - banana.example.com.fanclub
 - files
 - modules
 - settings.php
 - themes

<http://banana.example.com/fanclub>

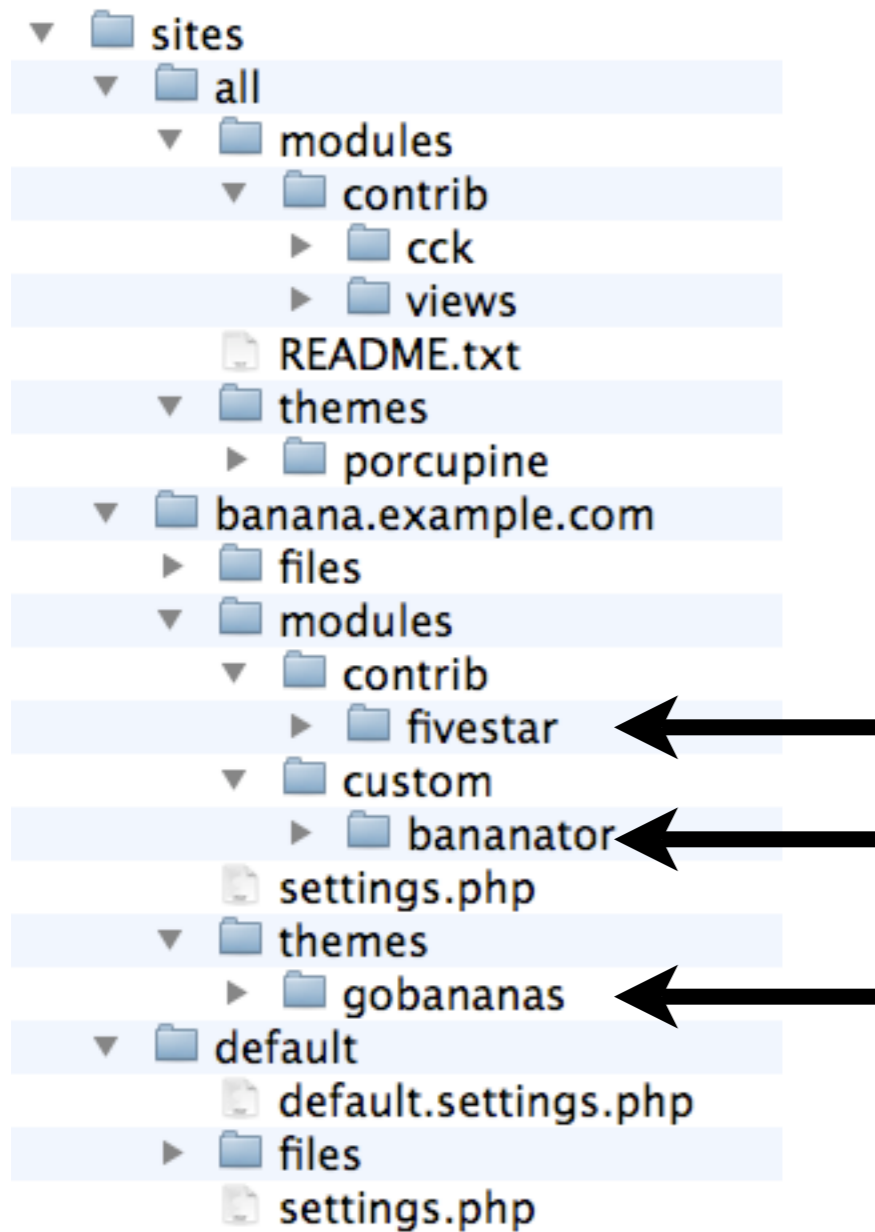


**CCK and Views
available for all sites**

**porcupine theme
available for all sites**

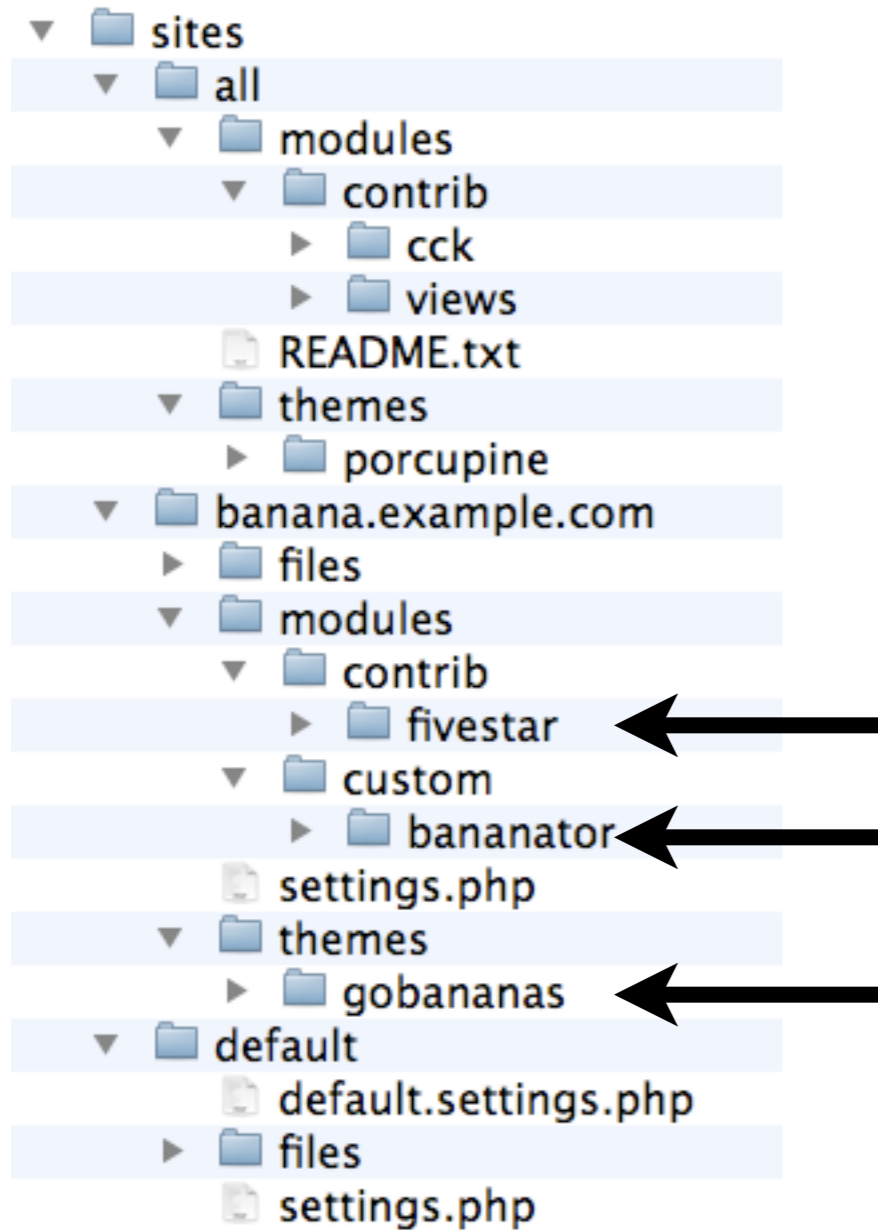


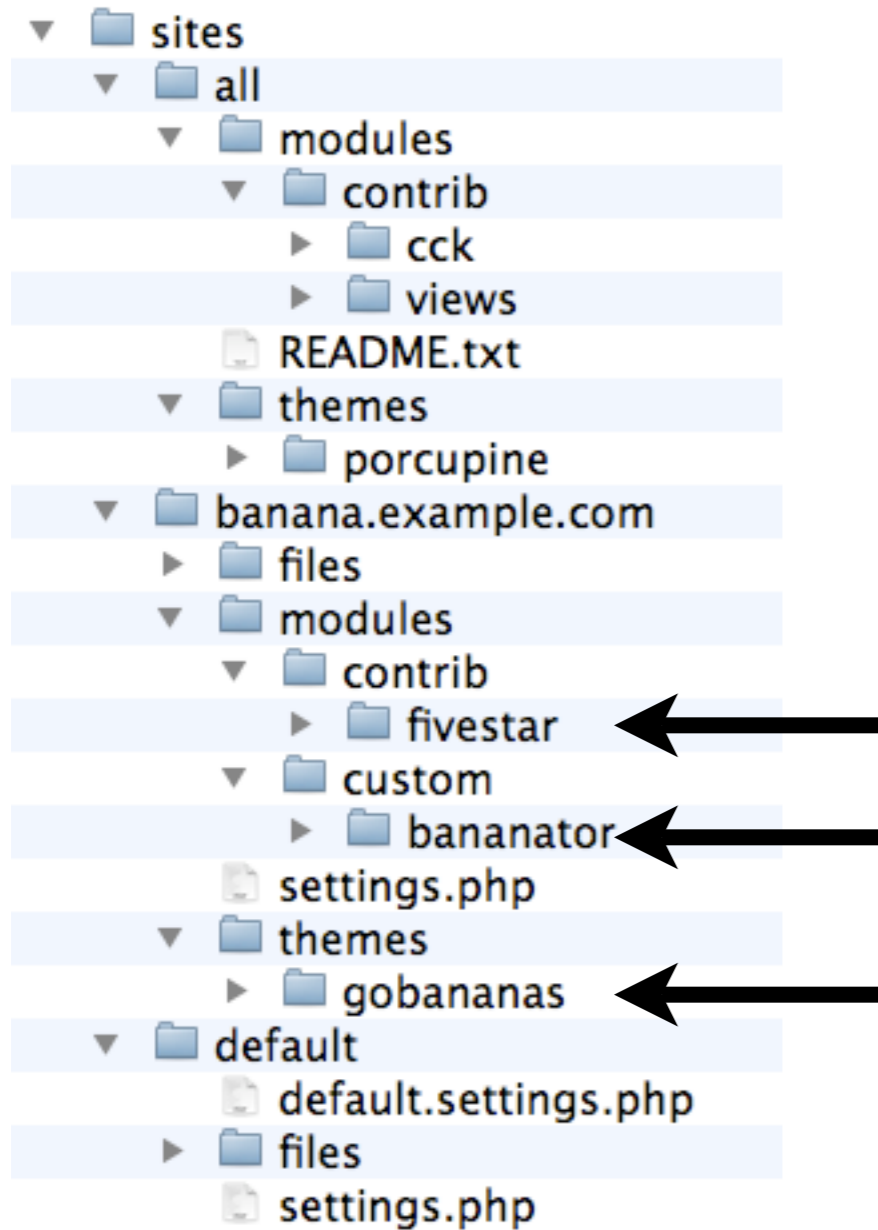
fivestar module only available on banana.example.com



fivestar module
only available
on banana.example.com

bananator module
only available
on banana.example.com

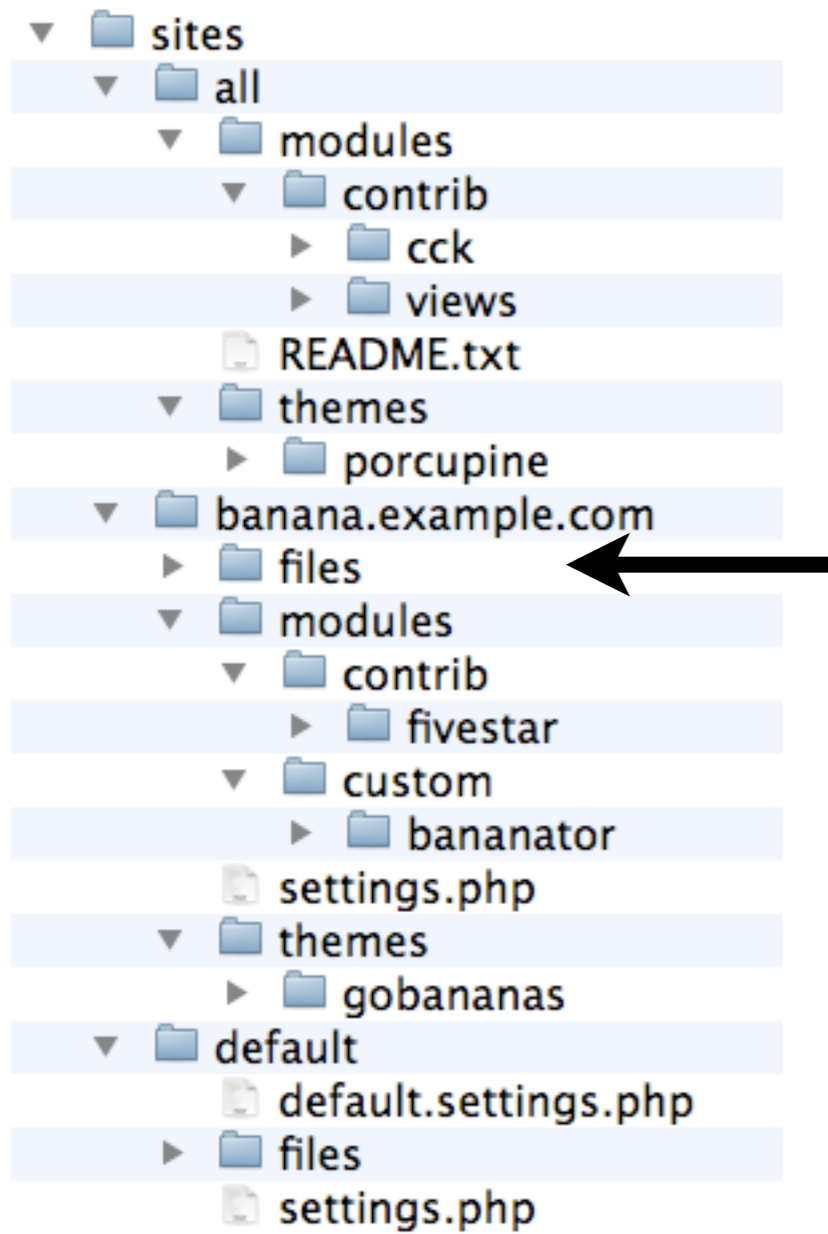




**fivestar module
only available
on banana.example.com**

**bananator module
only available
on banana.example.com**

**gobananas theme
only available
on banana.example.com**



Each site has its own place to put uploaded files.

[http://www.example.com/sites/www.example.com/files/
image.jpg](http://www.example.com/sites/www.example.com/files/image.jpg)

[http://www.example.com/sites/www.example.com/files/
image.jpg](http://www.example.com/sites/www.example.com/files/image.jpg)

```
RewriteRule ^files/(.*)$ /sites/{HTTP_HOST}/files/$1 [L]
```

[http://www.example.com/sites/www.example.com/files/
image.jpg](http://www.example.com/sites/www.example.com/files/image.jpg)

```
RewriteRule ^files/(.*)$ /sites/{HTTP_HOST}/files/$1 [L]
```

<http://www.example.com/files/image.jpg>

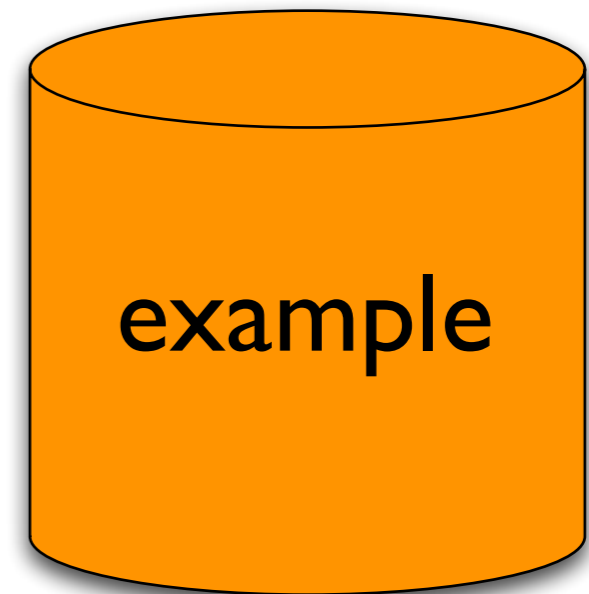
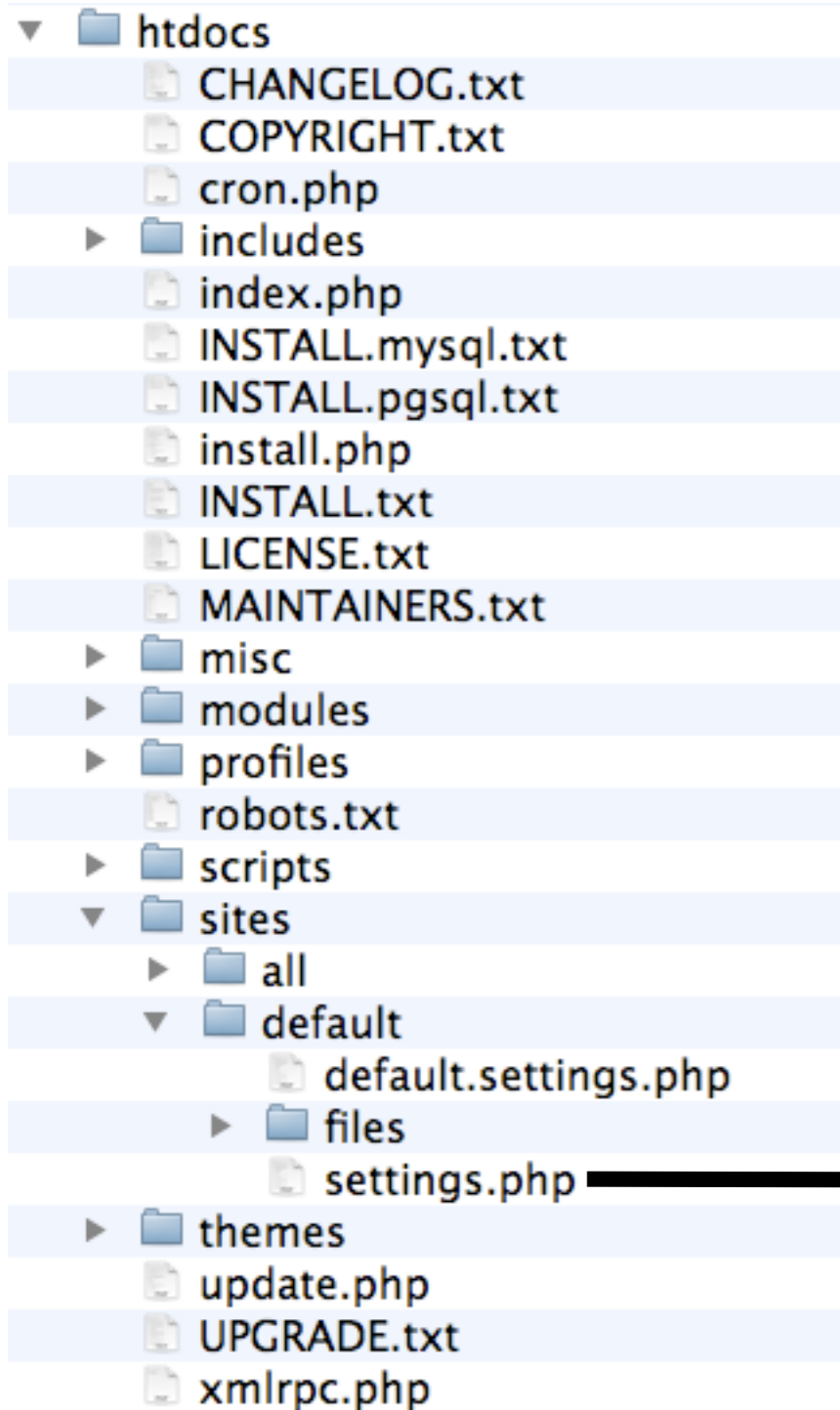
Summary So Far

- Single codebase
- Everything in sites/all is shared: modules, themes
- Site-specific modules and themes go in *sites/sitename/*
- Each site has its own file uploads directory

Cron!

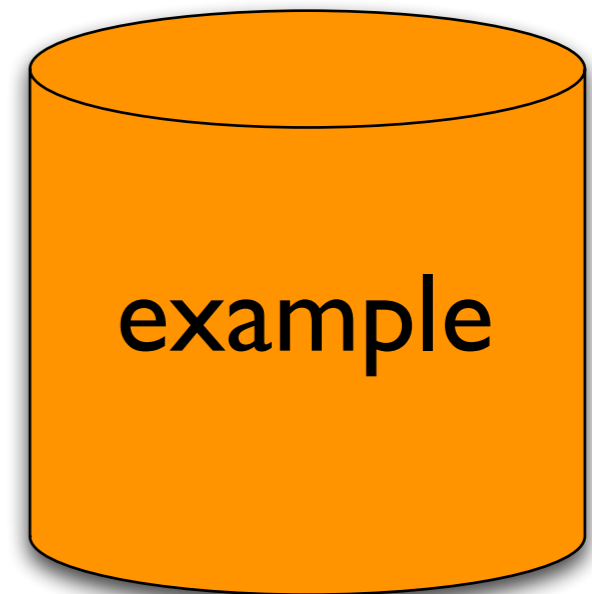
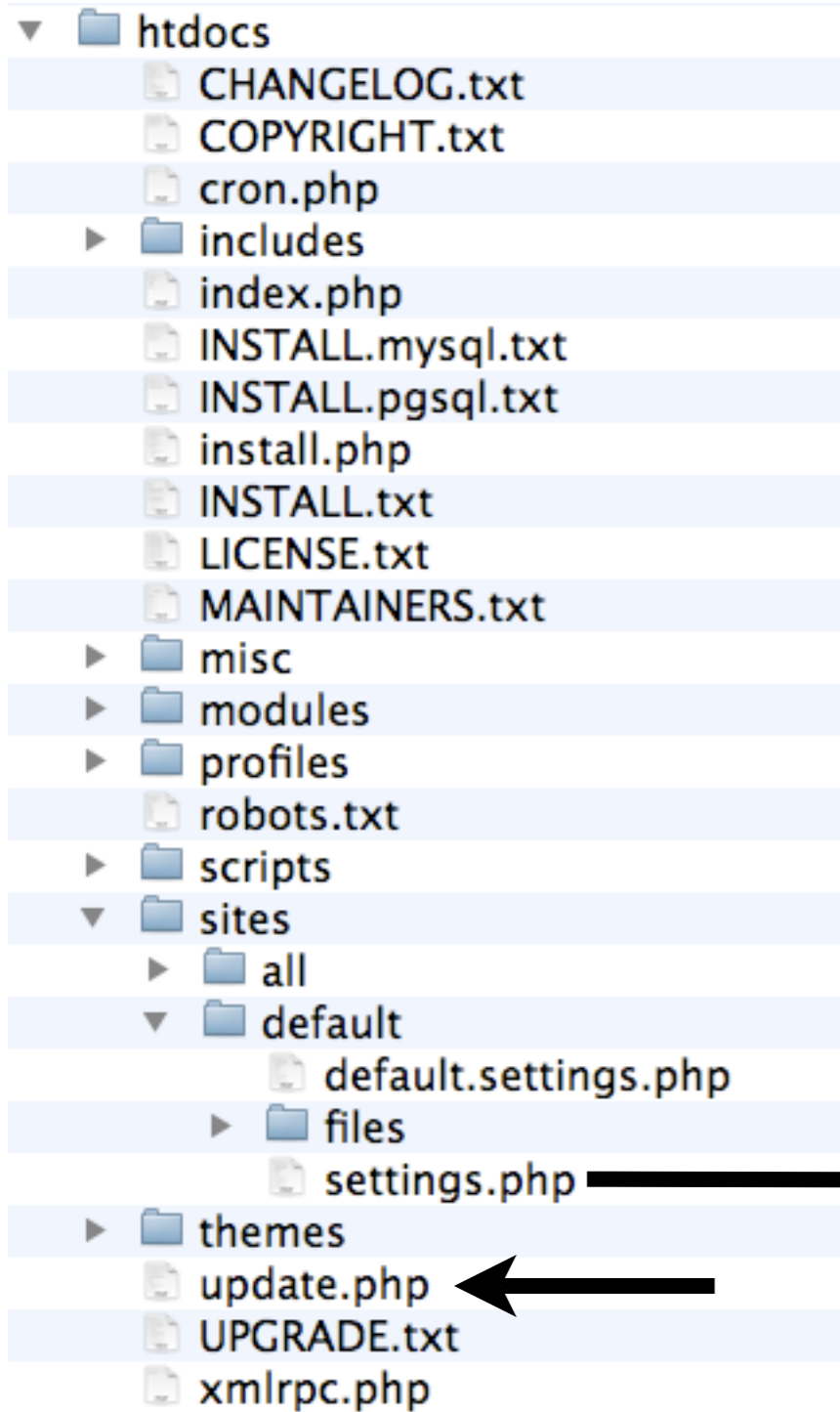
Updating Multisite

Updating a Drupal Site



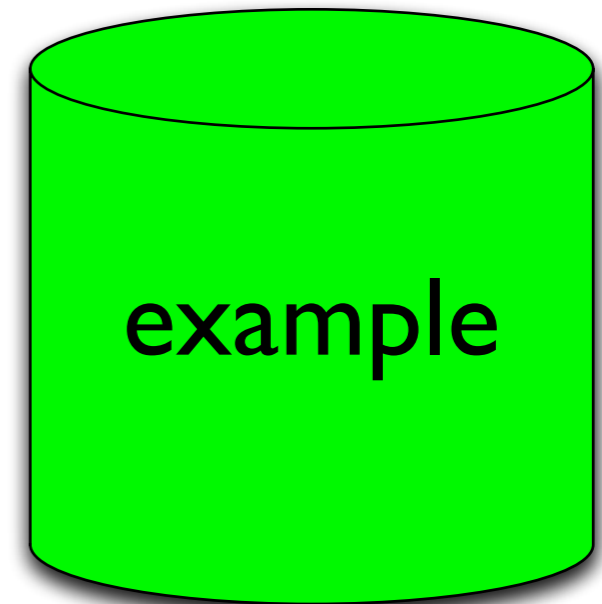
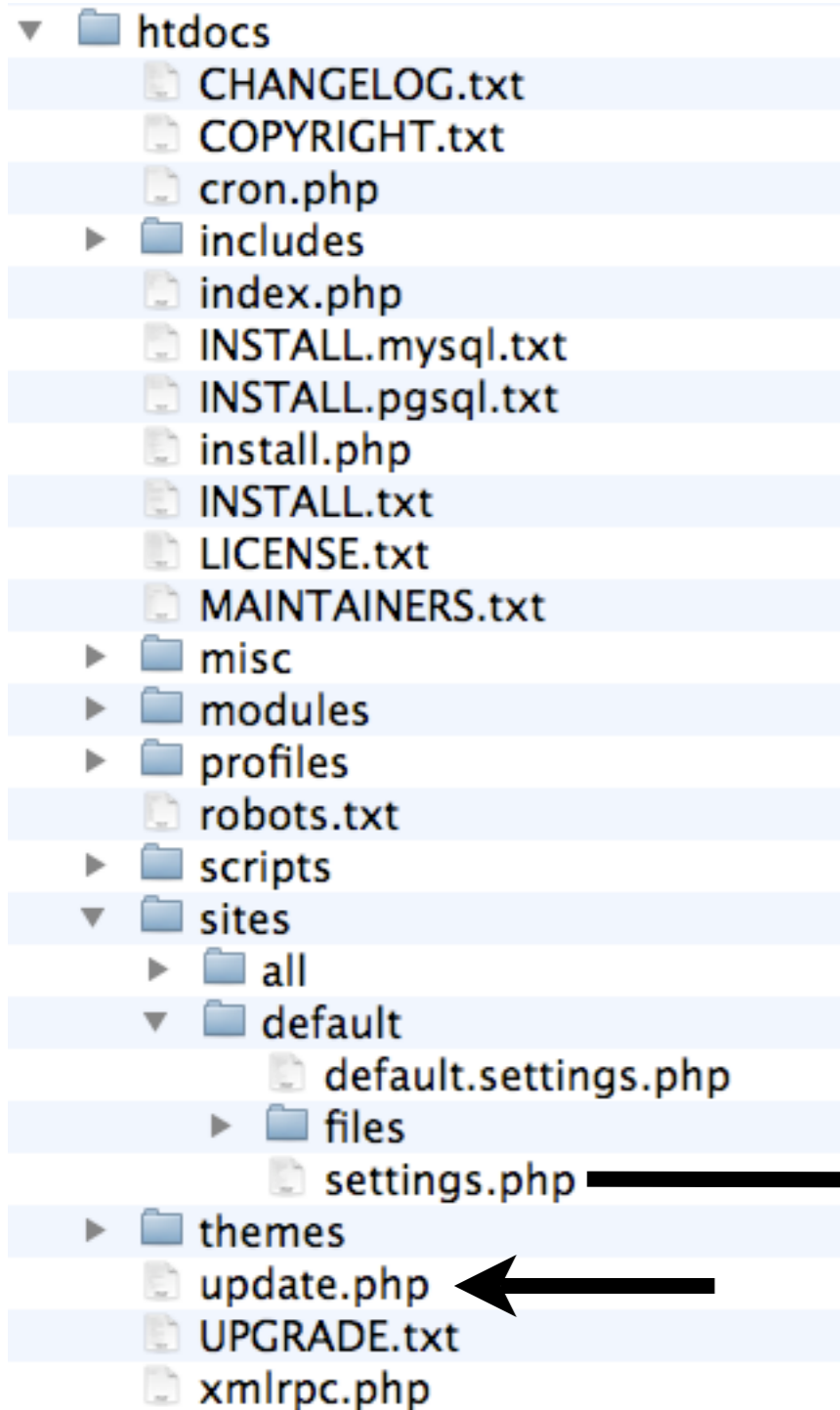
The database contains content and most configuration settings.

Updating a Drupal Site



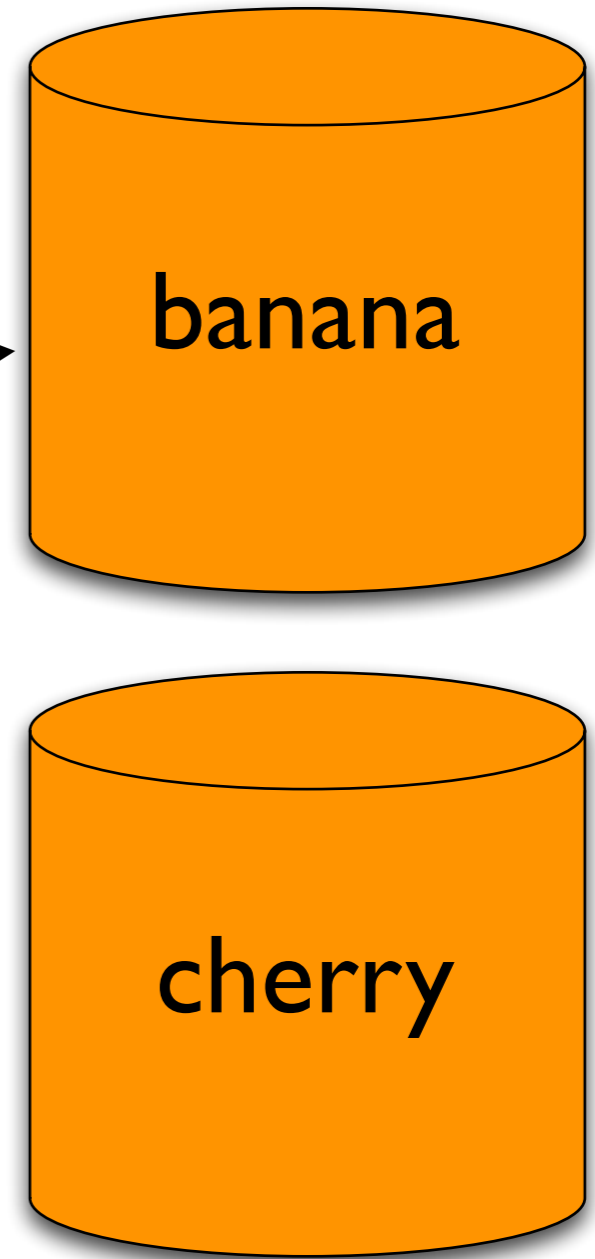
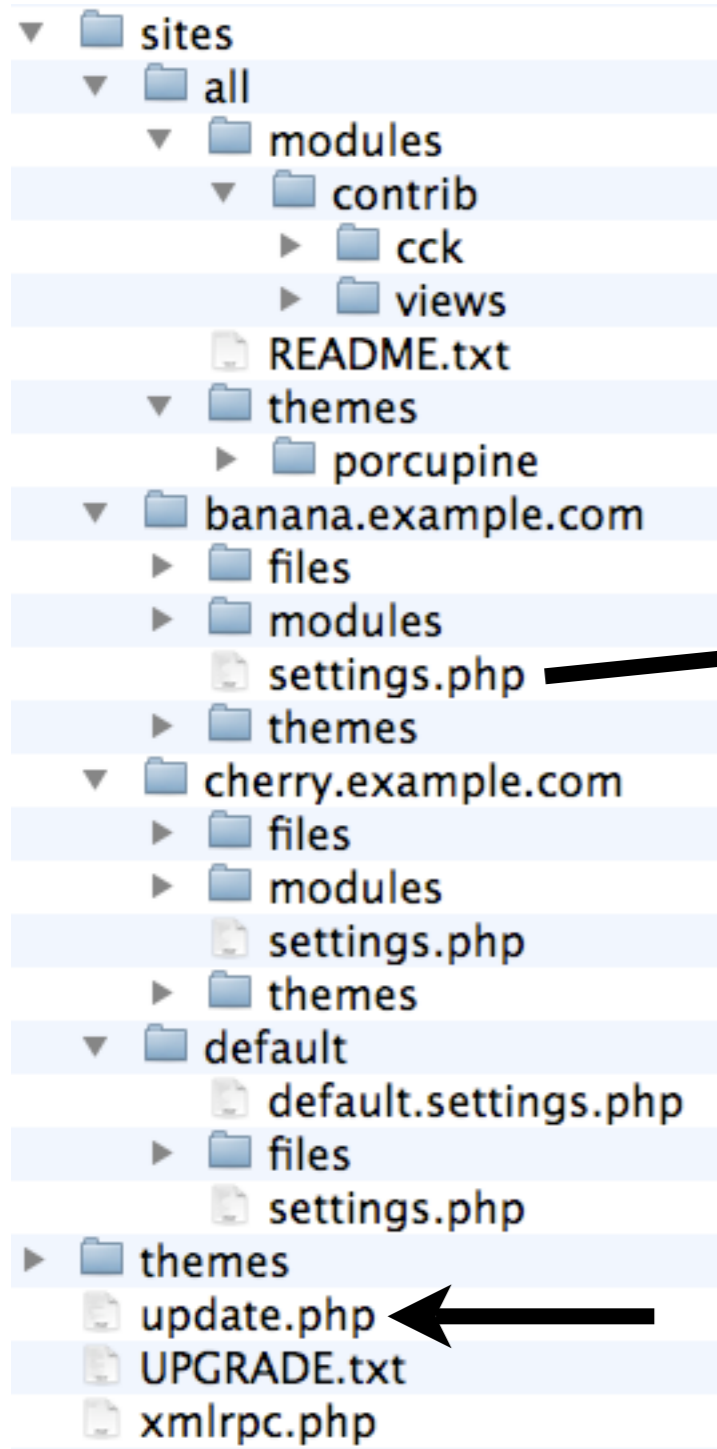
The database contains content and most configuration settings.

Updating a Drupal Site



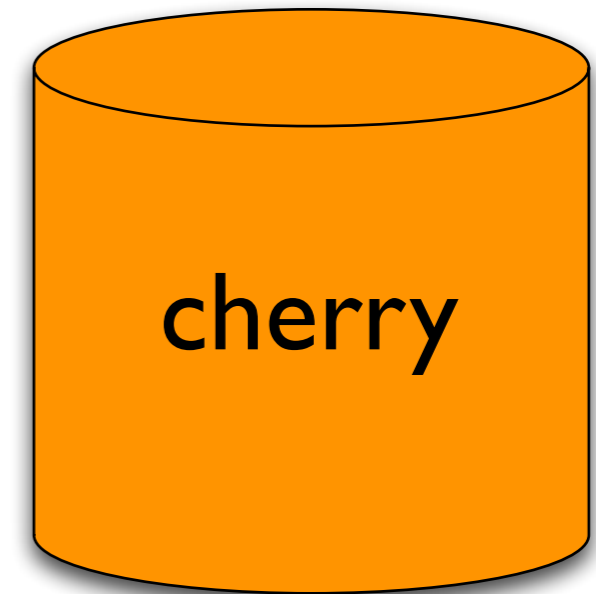
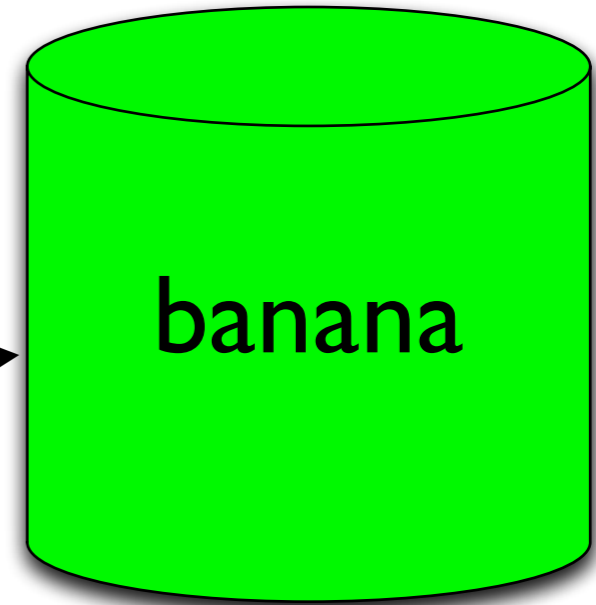
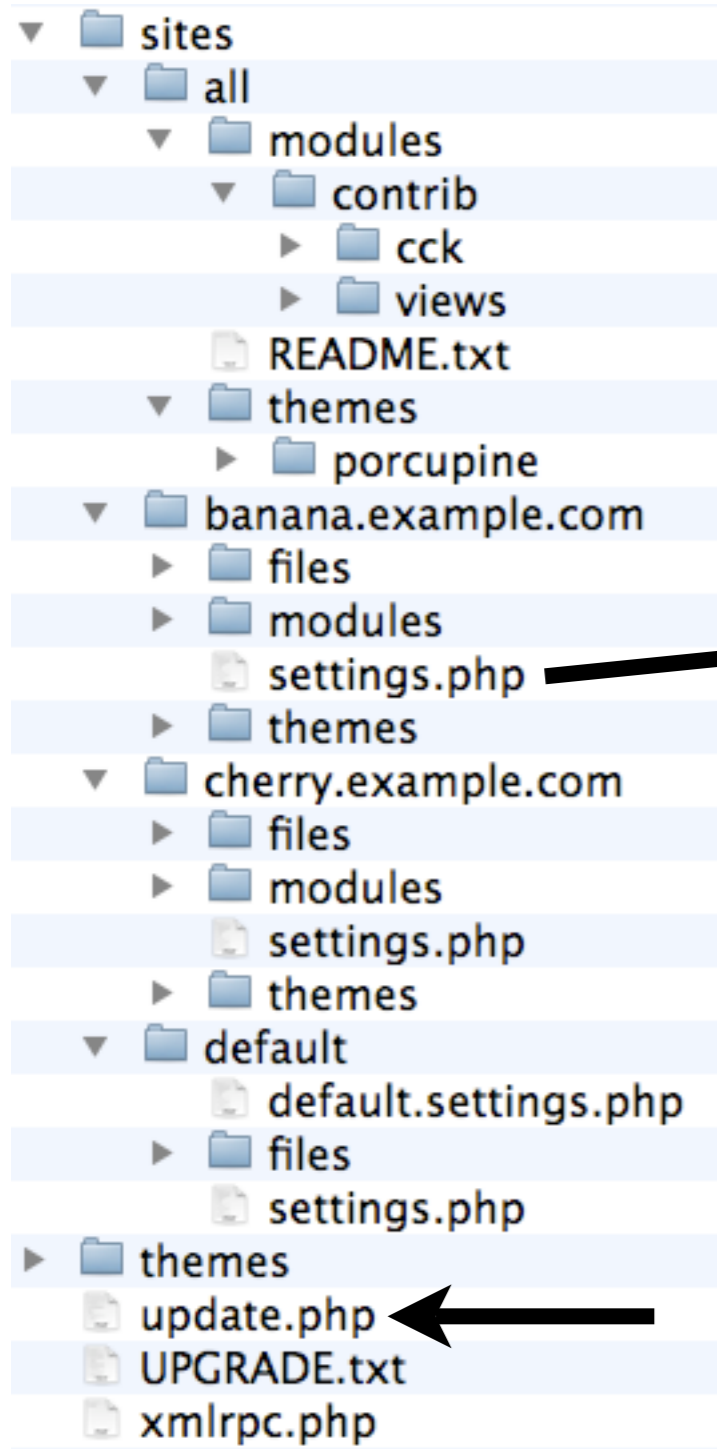
The database contains content and most configuration settings.

Updating Multisite



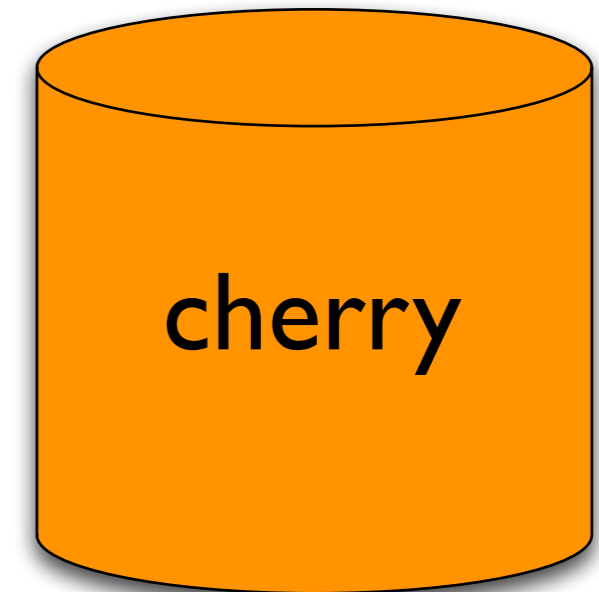
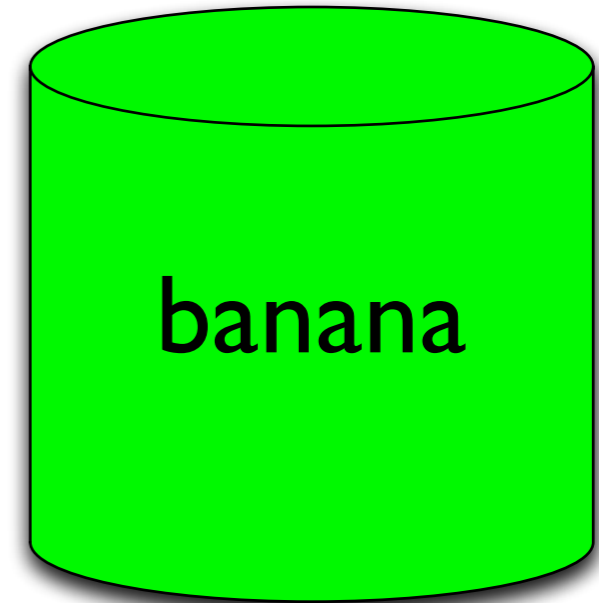
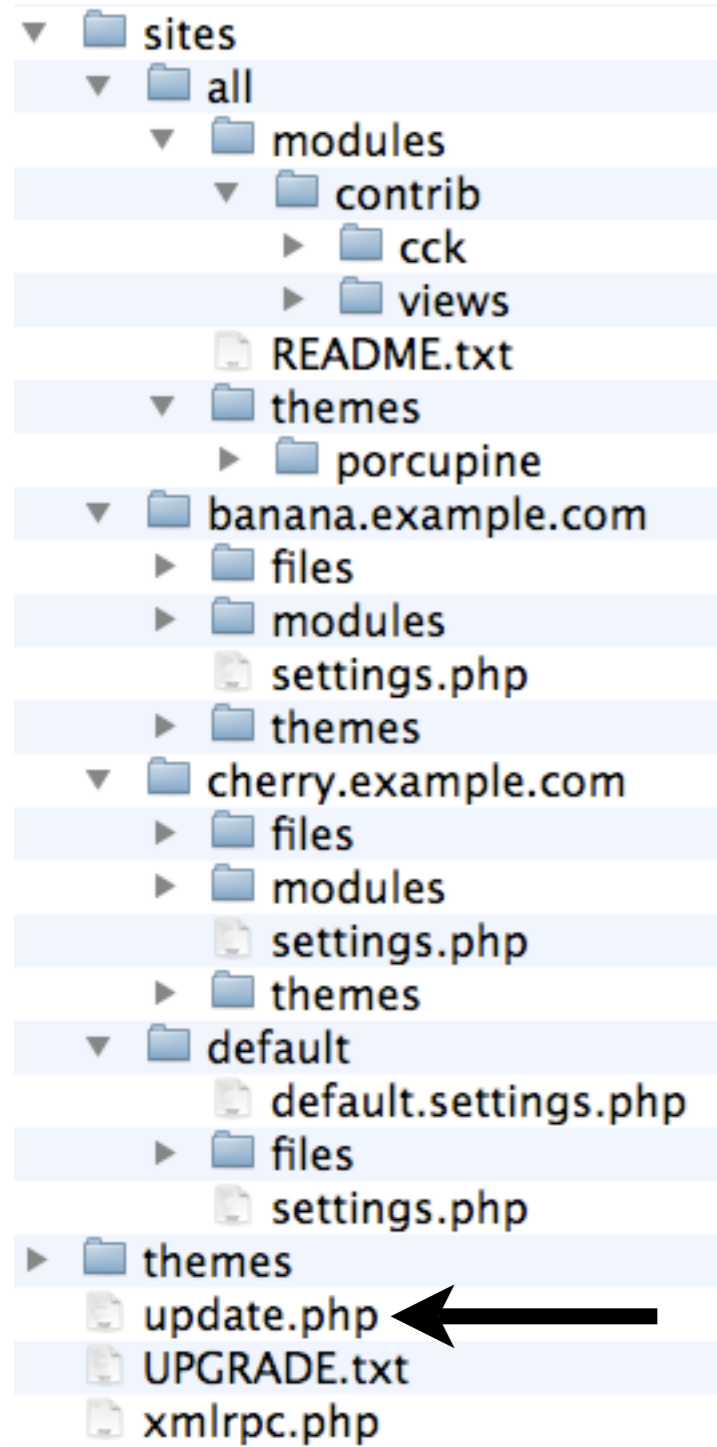
Running update.php once is not enough!

Updating Multisite



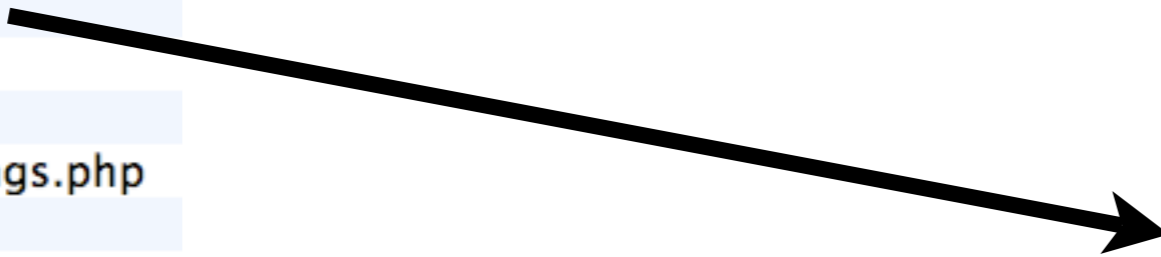
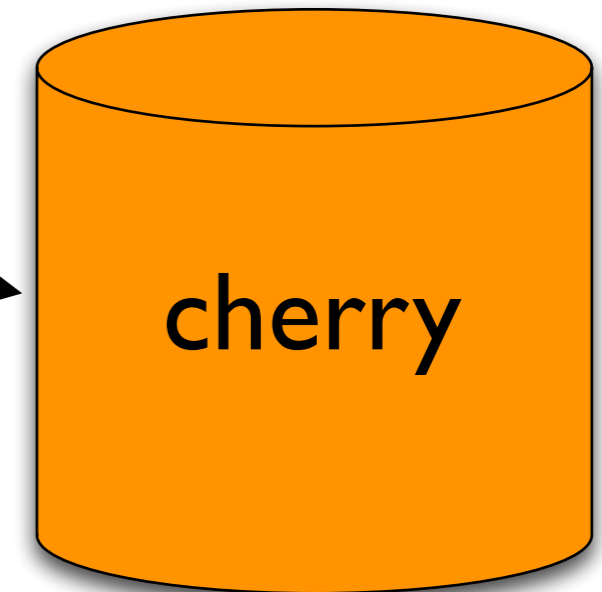
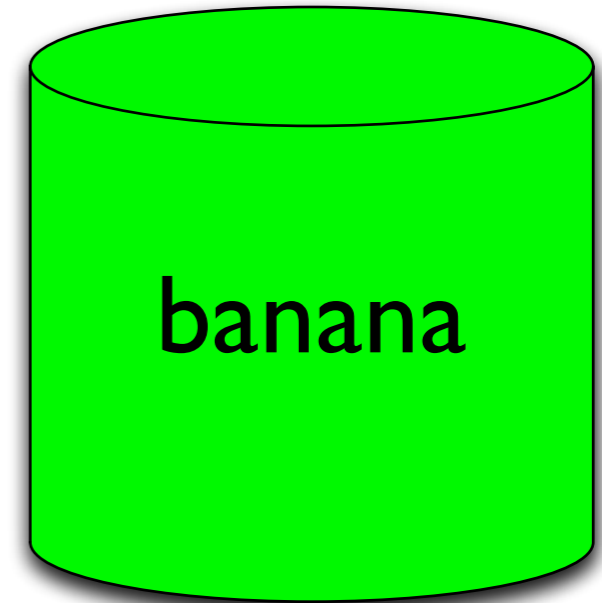
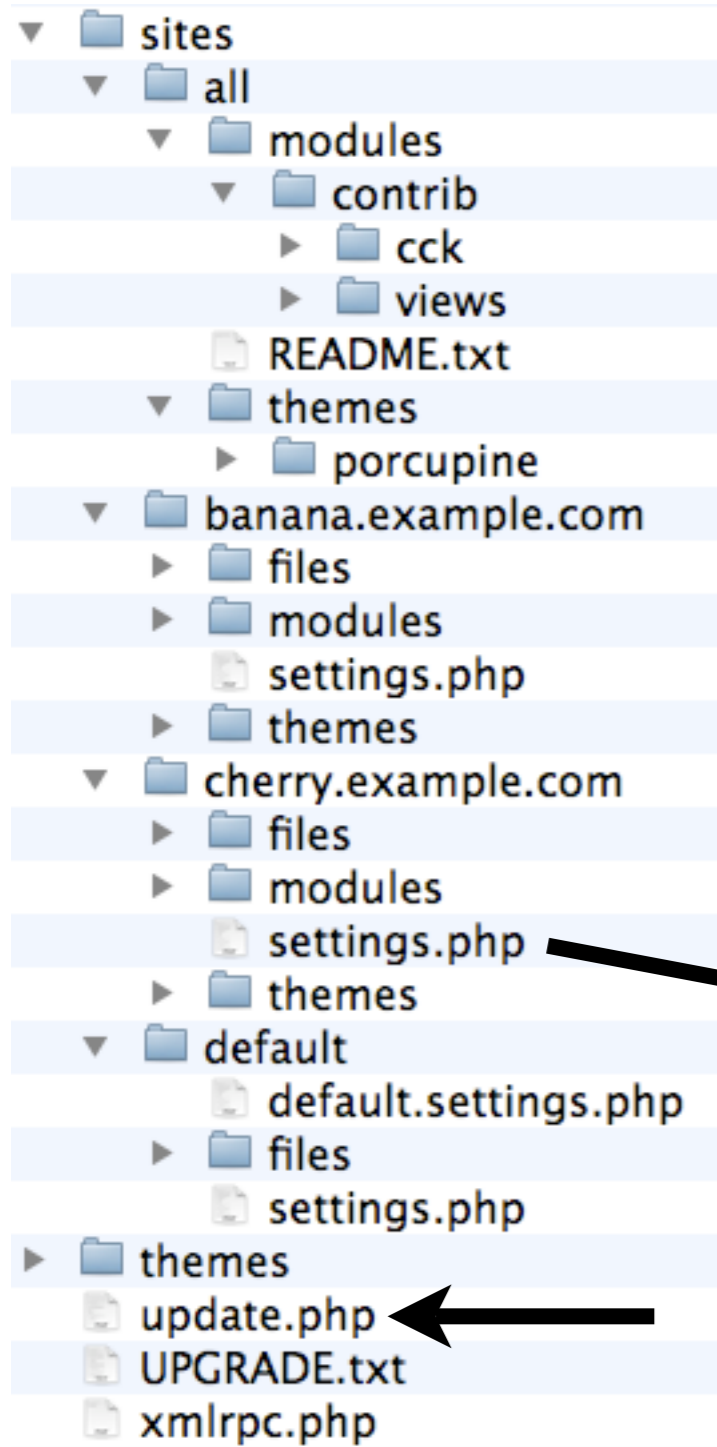
Running update.php once is not enough!

Updating Multisite



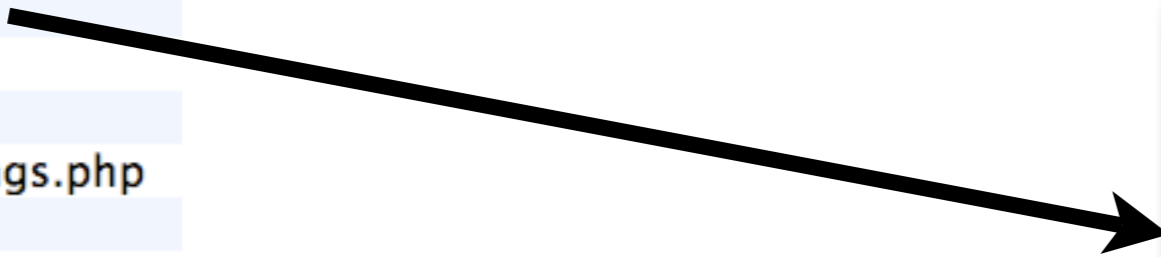
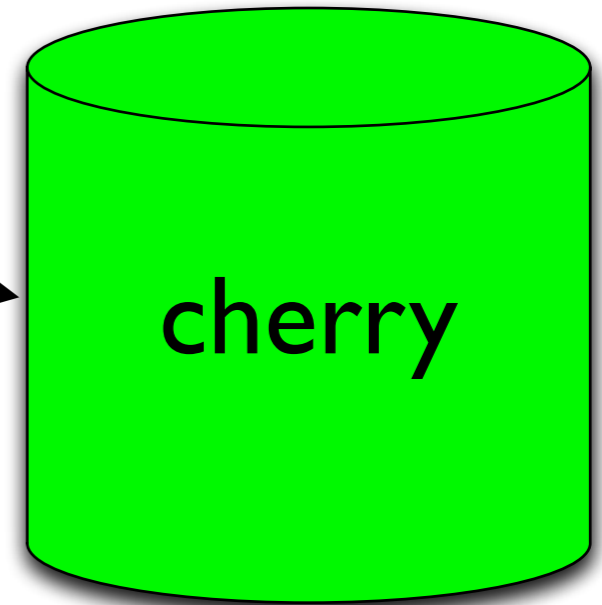
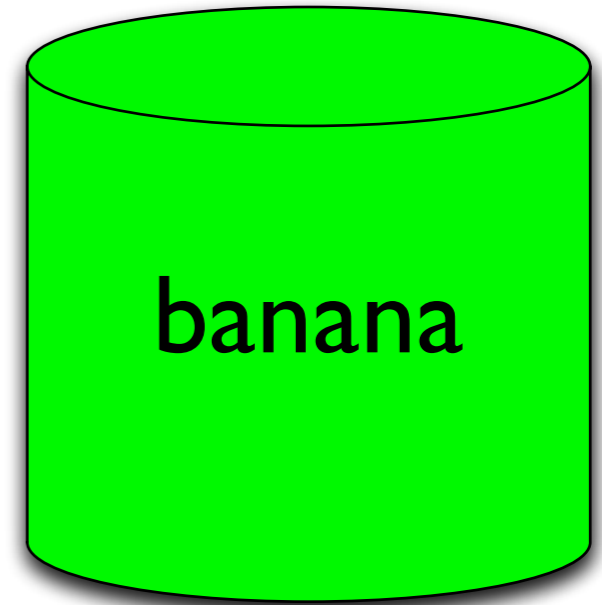
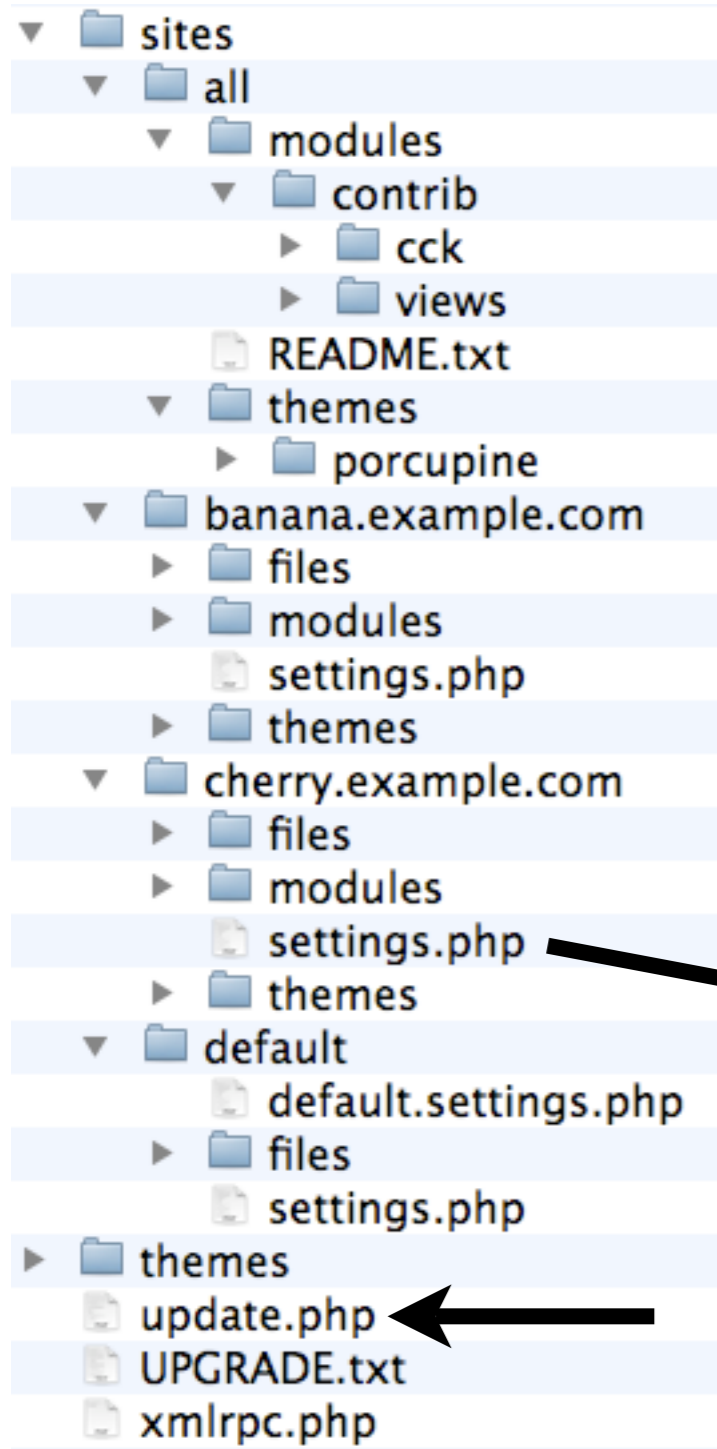
Running update.php once is not enough!

Updating Multisite



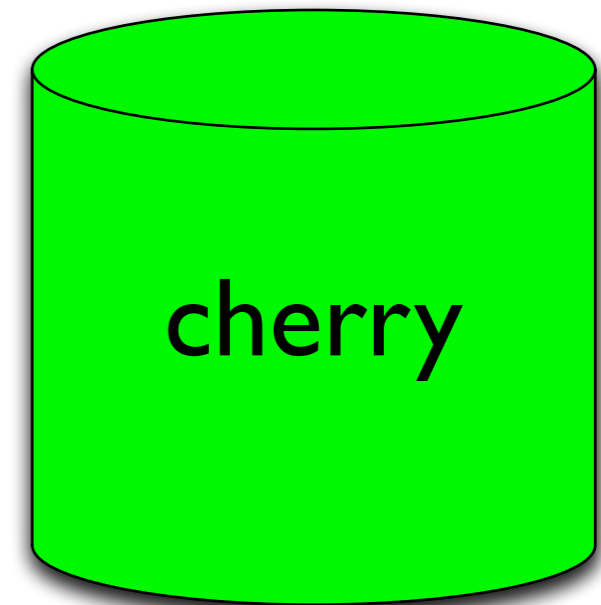
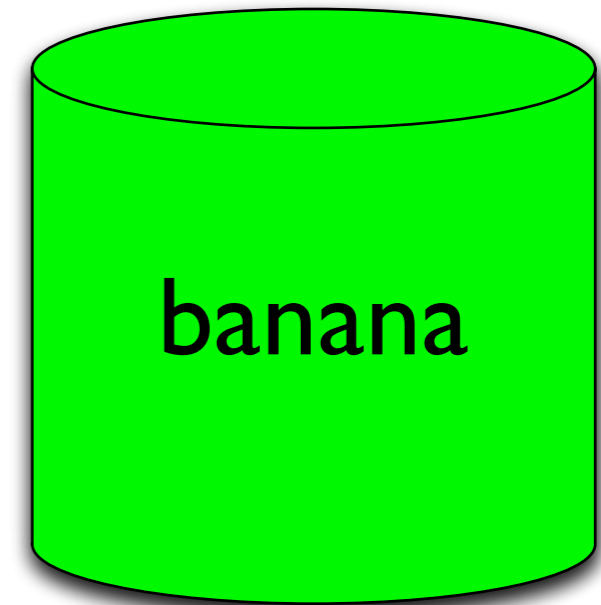
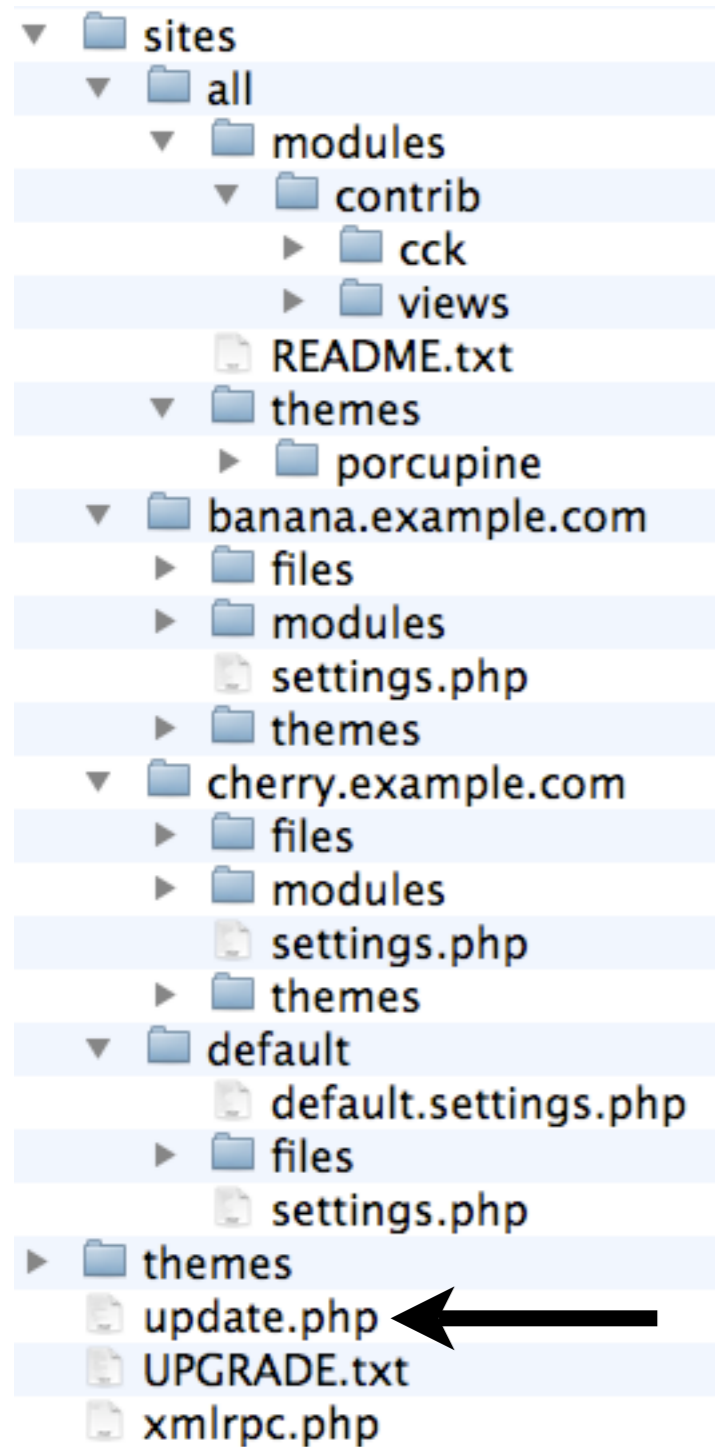
Running update.php once is not enough!

Updating Multisite



Running update.php once is not enough!

Updating Multisite



Running update.php once is not enough!

What is Multisite?

How Multisite Works

Sharing Information Between Sites

Table Prefixing

Table Prefixing

GOLD \$3.99/mo

- ↘ 300 GB Webspace
- ↘ 1 Free Domain for Life
- ↘ Free Unlimited Templates
- ↘ Unlimited Emails
- ↘ Ecommerce Enabled
- ↘ Free SSL
- ↘ MS-SQL (add-on), MySQL
- ↘ Windows/Linux Choice

[Details](#)

[Buy](#)

SILVER \$2.39/mo

- ↘ 1000 MB Webspace
- ↘ 1 Free Domain for Life
- ↘ Free \$10,000 Templates
- ↘ Unlimited Emails
- ↘ ASP, ASP.NET, PHP
- ↘ MS-SQL (add-on), MySQL
- ↘ PLESK Control Panel
- ↘ Windows/Linux Choice

[Details](#)

[Buy](#)

BRONZE \$1.99/mo

- ↘ 100 MB Webspace
- ↘ 1 Free Domain for Life
- ↘ Free \$500 Templates
- ↘ Unlimited Emails
- ↘ ASP, ASP.NET, PHP
- ↘ MS-SQL (add-on), MySQL
- ↘ PLESK Control Panel
- ↘ Windows/Linux Choice

[Details](#)

[Buy](#)

Table Prefixing

GOLD \$3.99/mo

- ↘ 300 GB Webspace
- ↘ 1 Free Domain for Life
- ↘ Free Unlimited Templates
- ↘ Unlimited Emails
- ↘ Ecommerce Enabled
- ↘ Free SSL
- ↘ MS-SQL (add-on), MySQL
- ↘ Windows/Linux Choice

[Details](#)

[Buy](#)

SILVER \$2.39/mo

- ↘ 1000 MB Webspace
- ↘ 1 Free Domain for Life
- ↘ Free \$10,000 Templates
- ↘ Unlimited Emails
- ↘ ASP, ASP.NET, PHP
- ↘ MS-SQL (add-on), MySQL
- ↘ PLESK Control Panel
- ↘ Windows/Linux Choice

[Details](#)

[Buy](#)

BRONZE \$1.99/mo

- ↘ 100 MB Webspace
- ↘ 1 Free Domain for Life
- ↘ Free \$500 Templates
- ↘ Unlimited Emails
- ↘ ASP, ASP.NET, PHP
- ↘ MS-SQL (add-on), MySQL
- ↘ PLESK Control Panel
- ↘ Windows/Linux Choice

[Details](#)

[Buy](#)

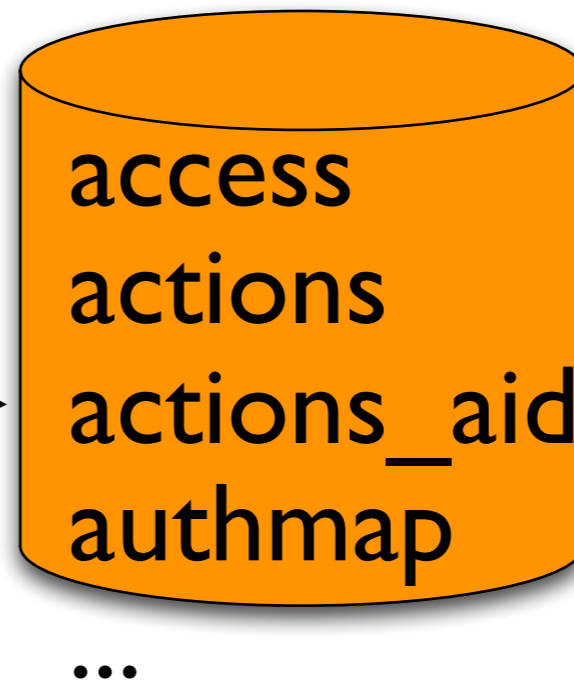
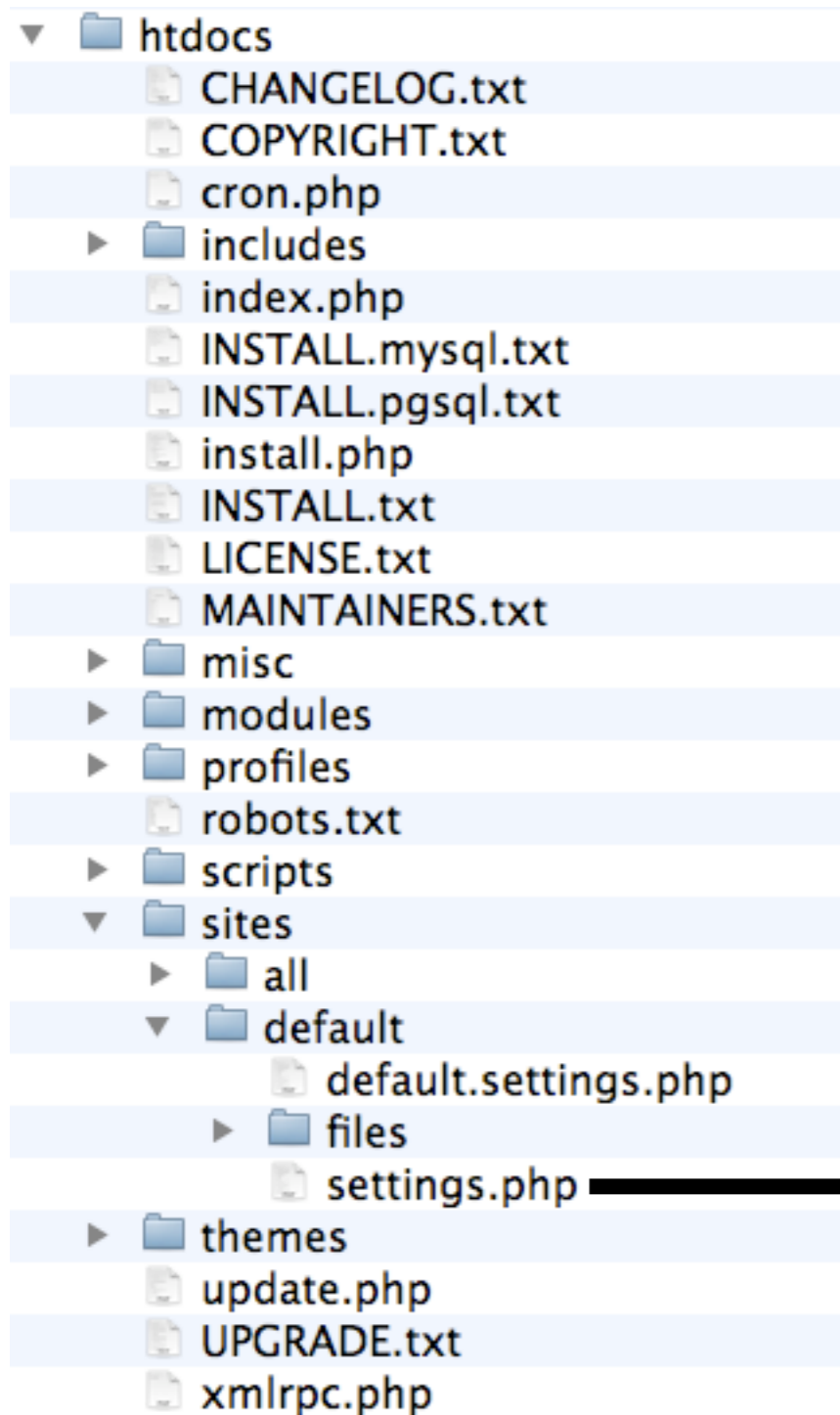
```
$db_prefix = 'zoinks_';
```

```
$db_prefix = array(  
    'default' => 'main_',  
    'users'   => 'shared_',  
    'sessions' => 'shared_',  
    'role'    => 'shared_',  
    'authmap' => 'shared_',  
);
```

```
$db_prefix = array(  
    'default' => "",  
    'users'   => 'shared.',  
    'sessions' => 'shared.',  
    'role'    => 'shared.',  
    'authmap' => 'shared.',  
);
```

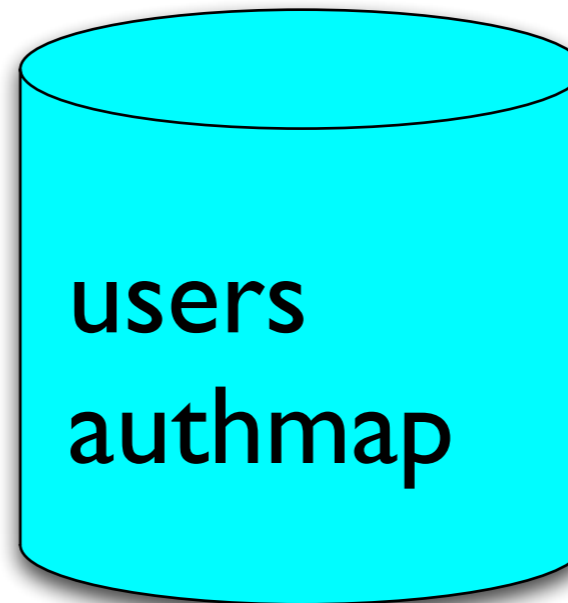
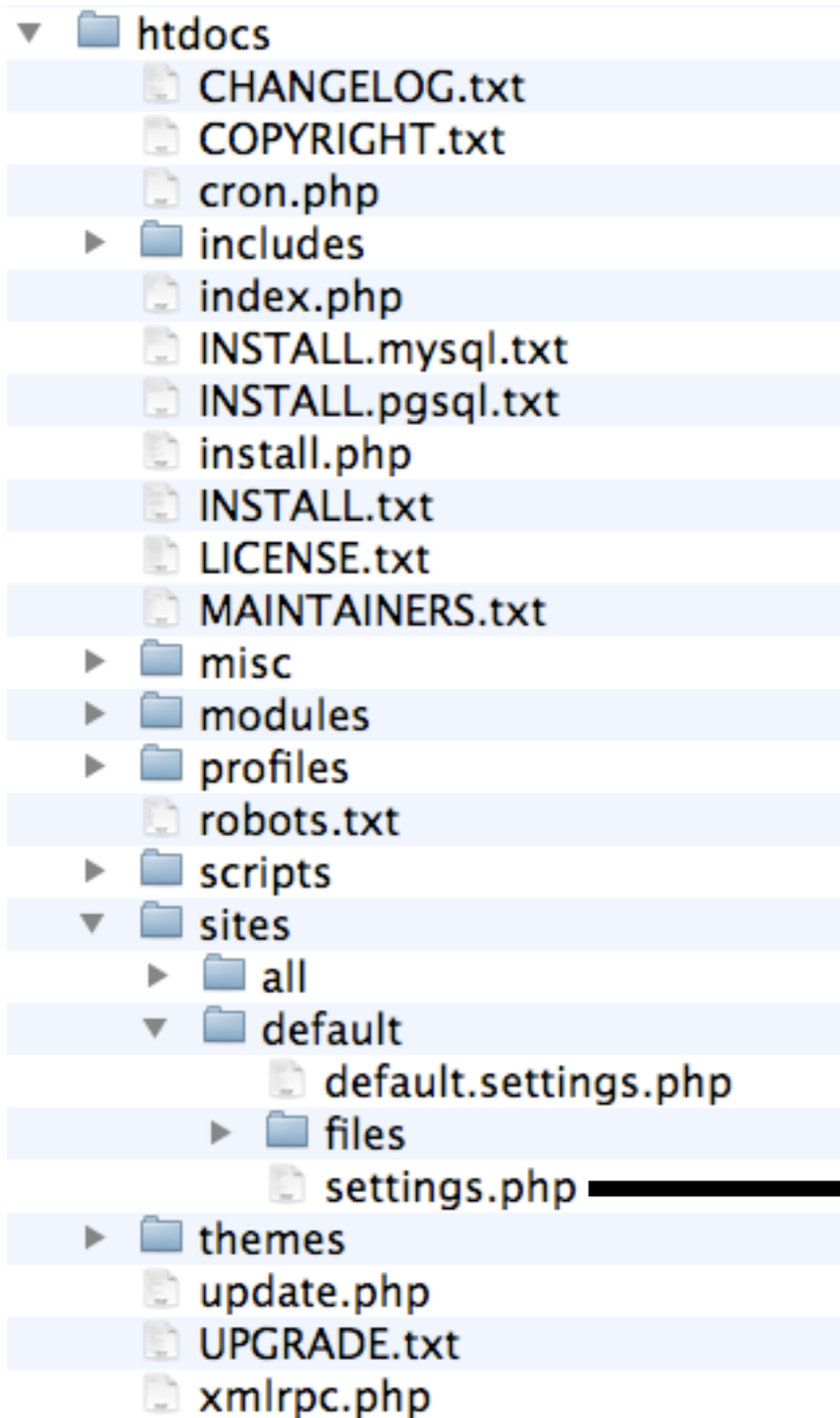
shared.users

databaseName.tableName

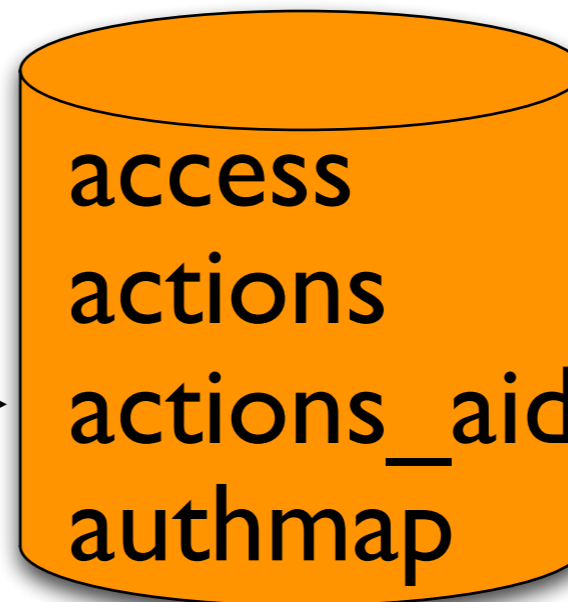


example

What will happen with this arrangement? Nothing different! The location of the two tables has just changed (users and authmap tables in the example database are unused).



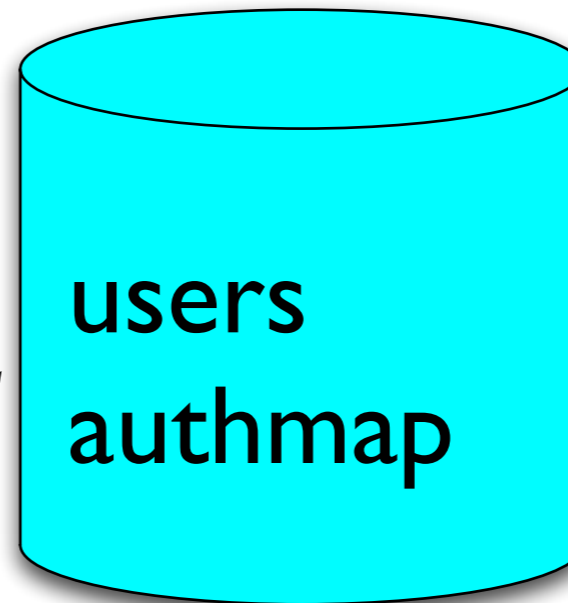
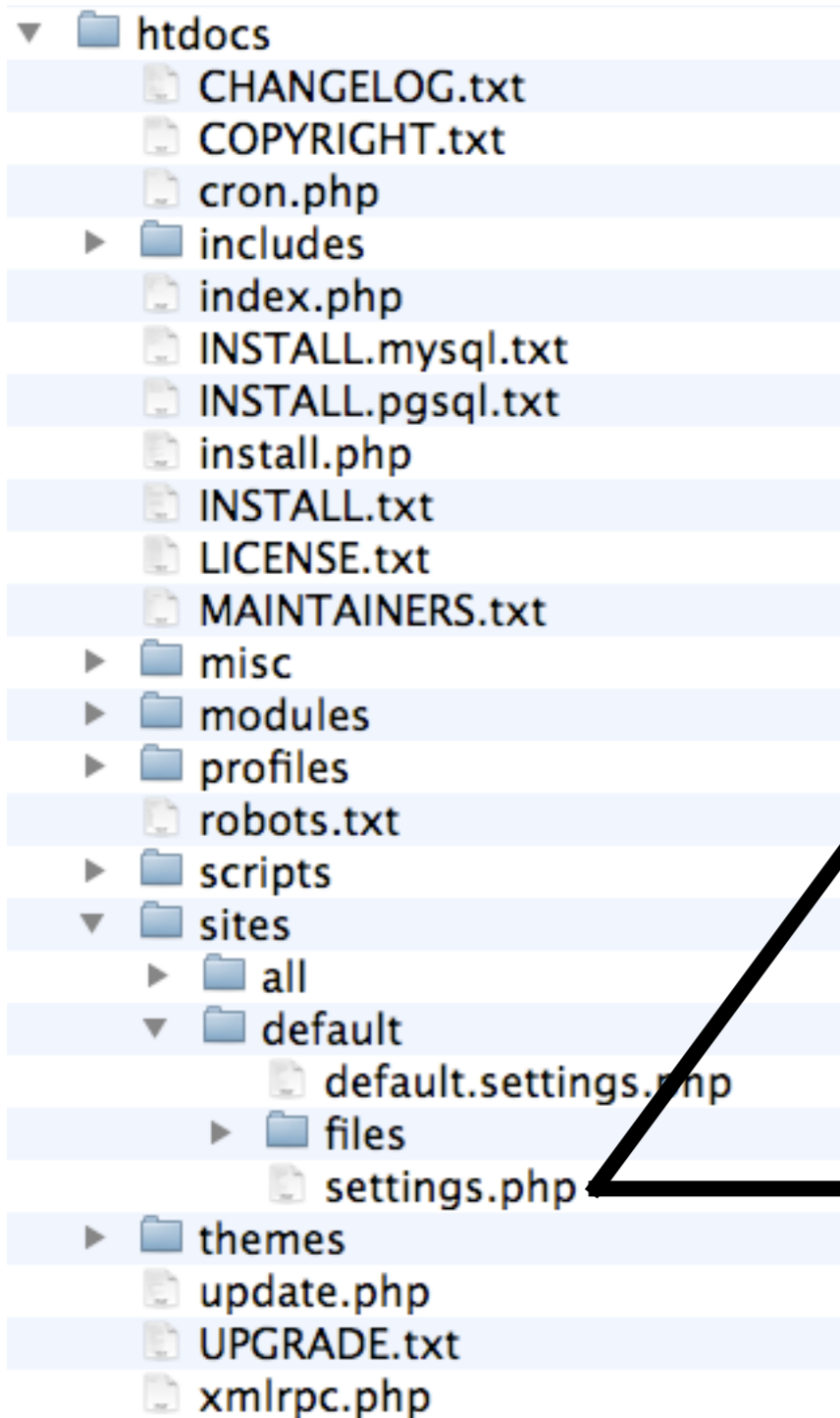
shared



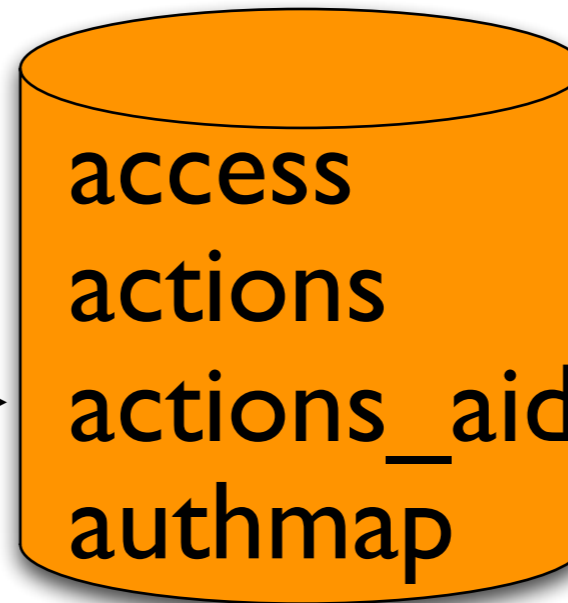
example

...

What will happen with this arrangement? Nothing different! The location of the two tables has just changed (users and authmap tables in the example database are unused).



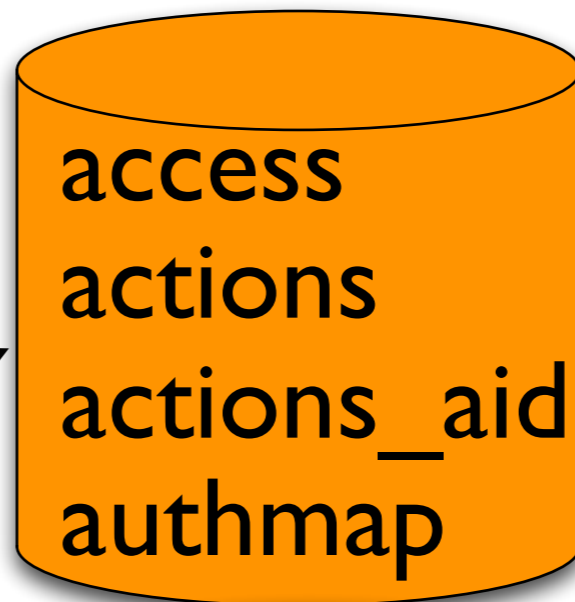
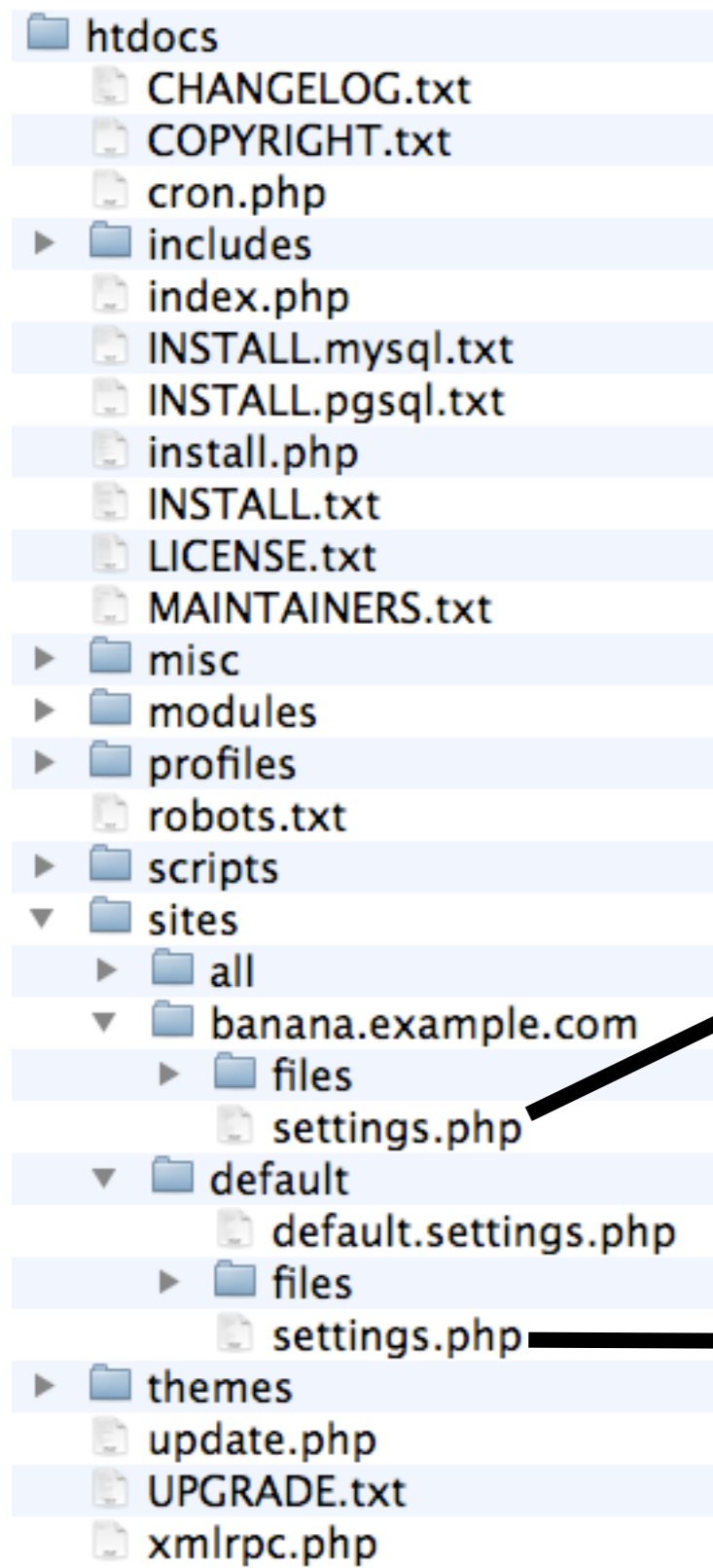
shared



example

...

What will happen with this arrangement? Nothing different! The location of the two tables has just changed (users and authmap tables in the example database are unused).



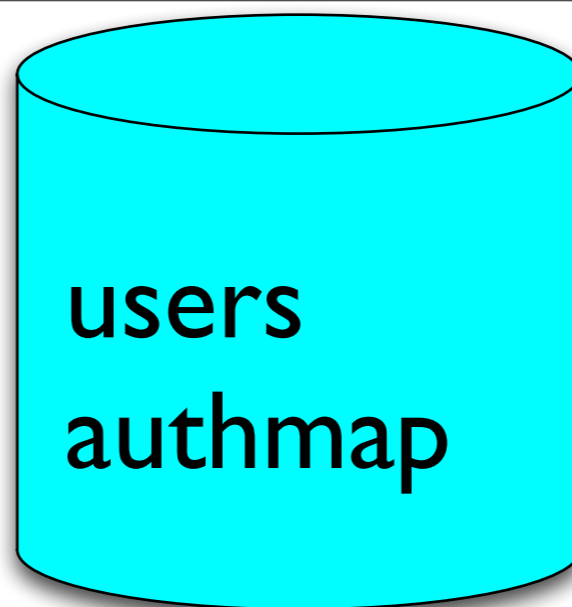
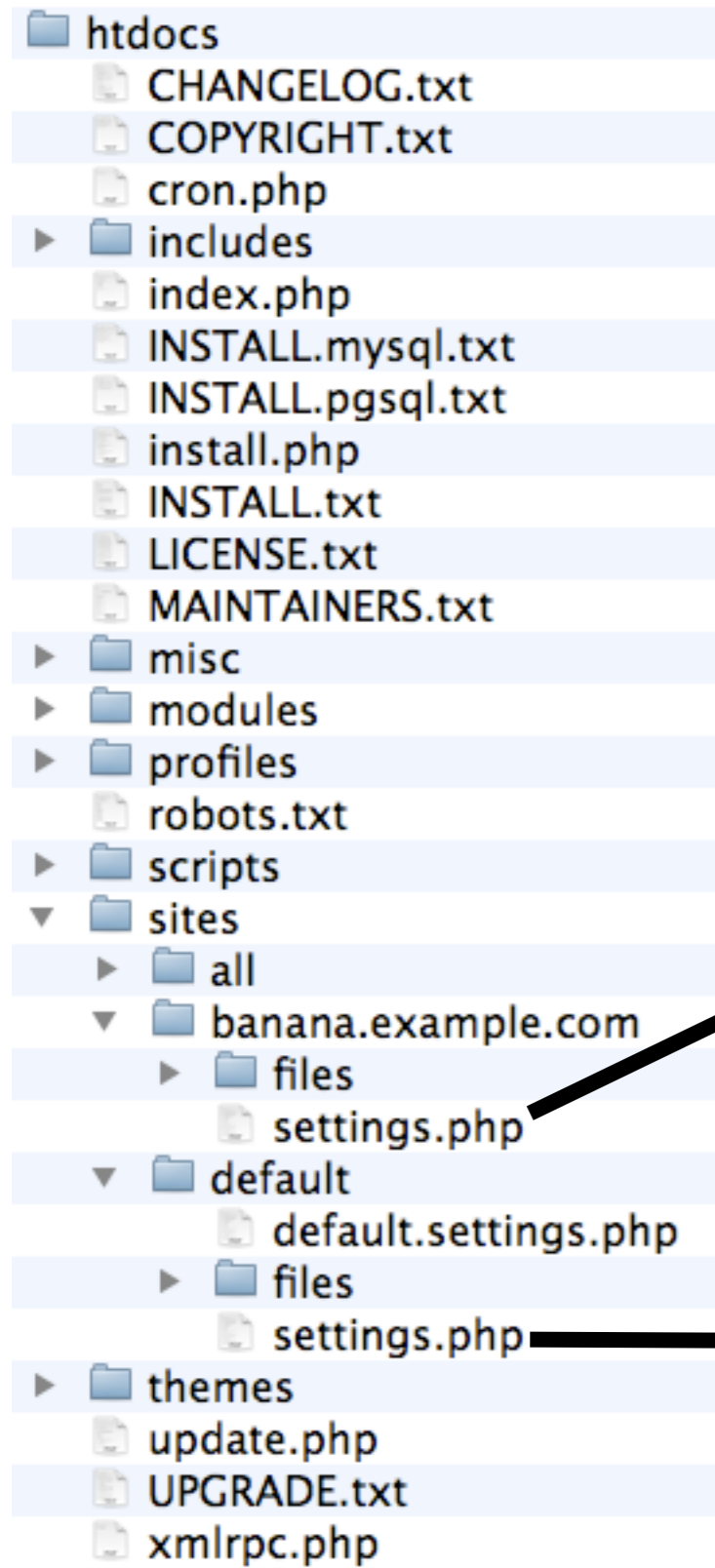
banana



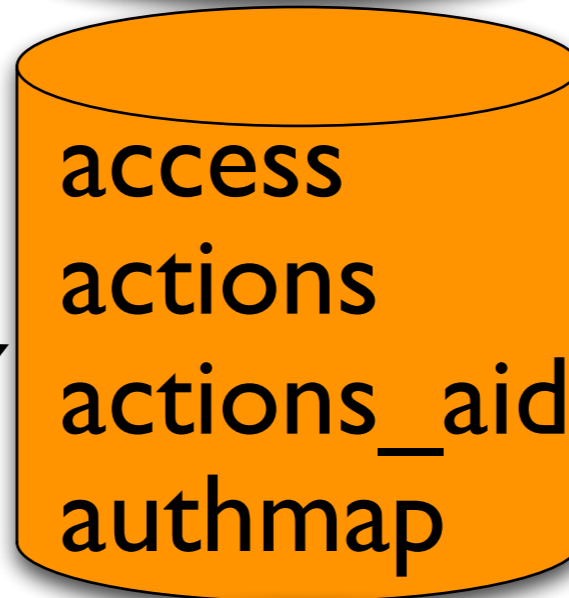
example

...

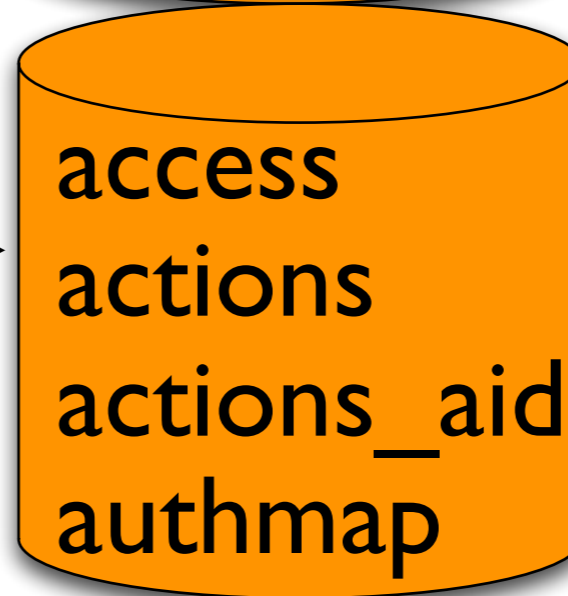
Still no change in behavior!



shared

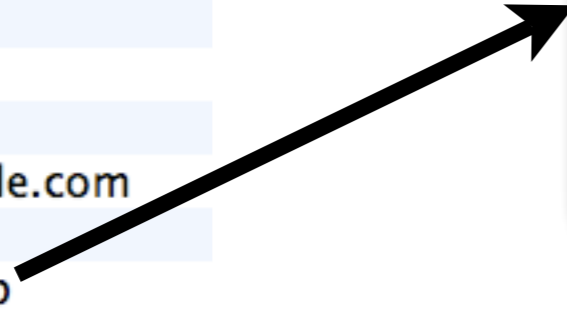


banana

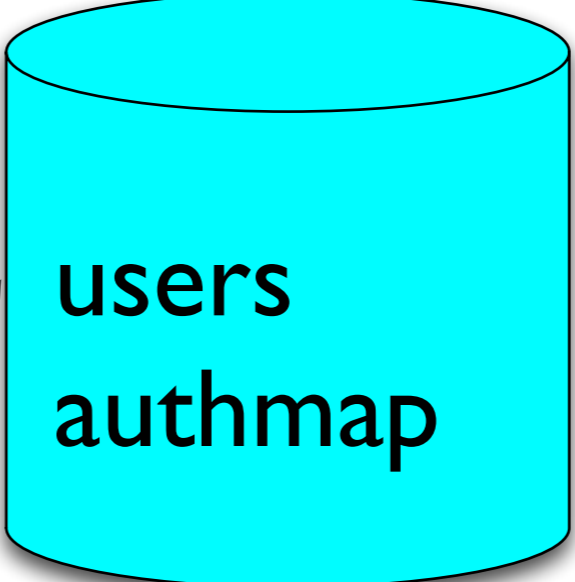
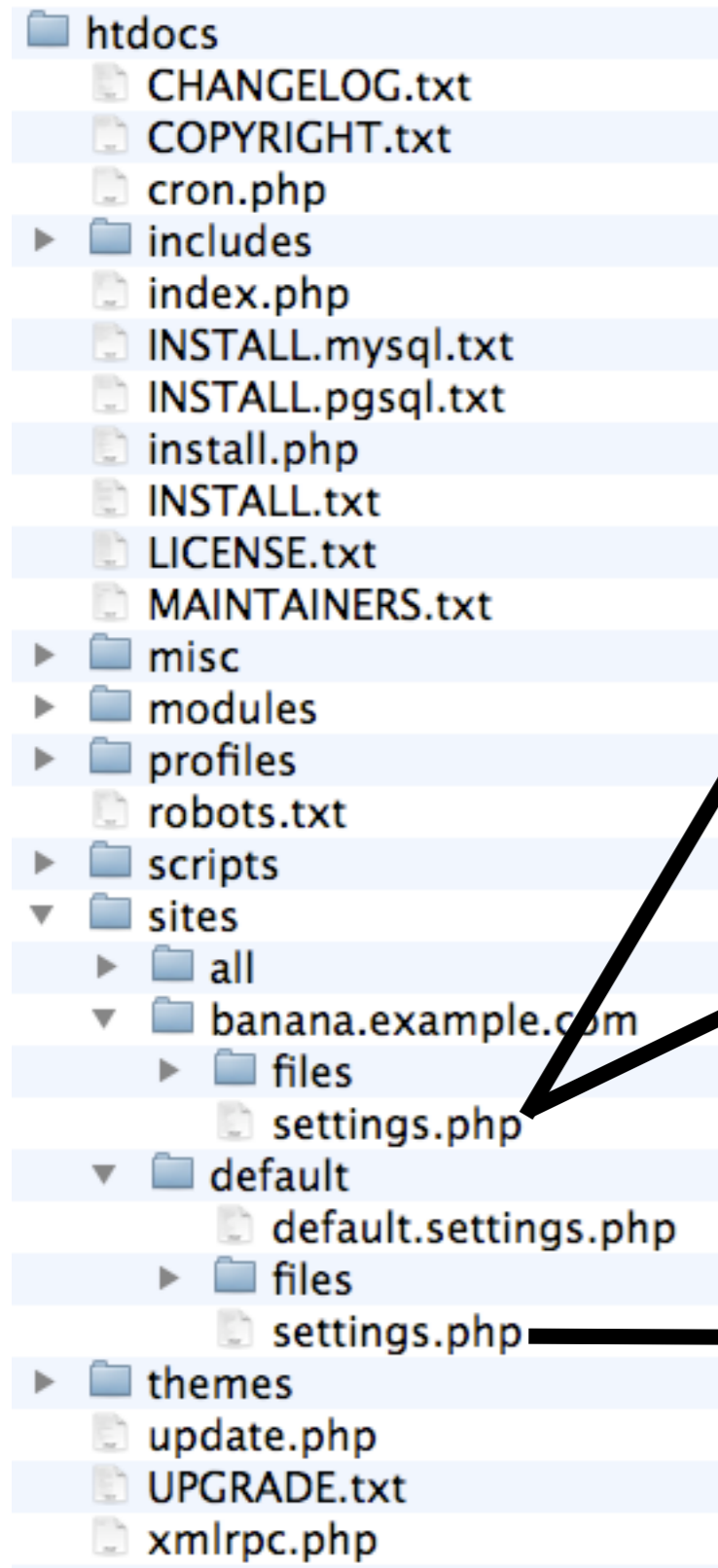


example

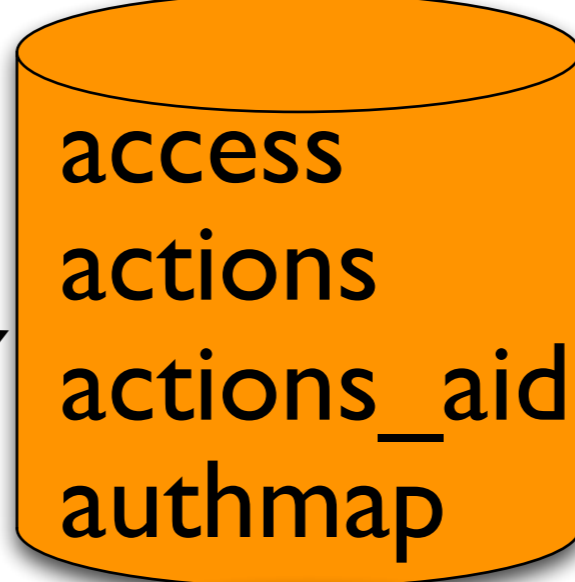
...



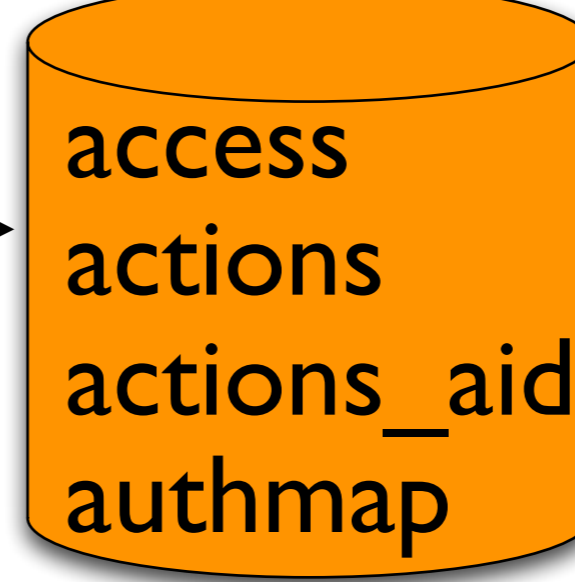
Still no change in behavior!



shared



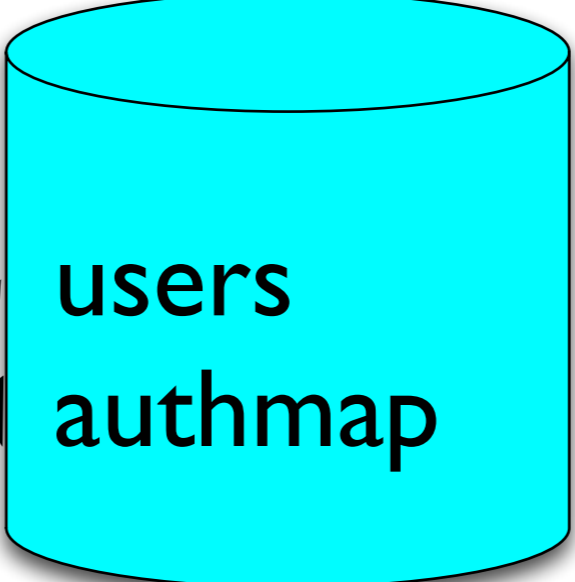
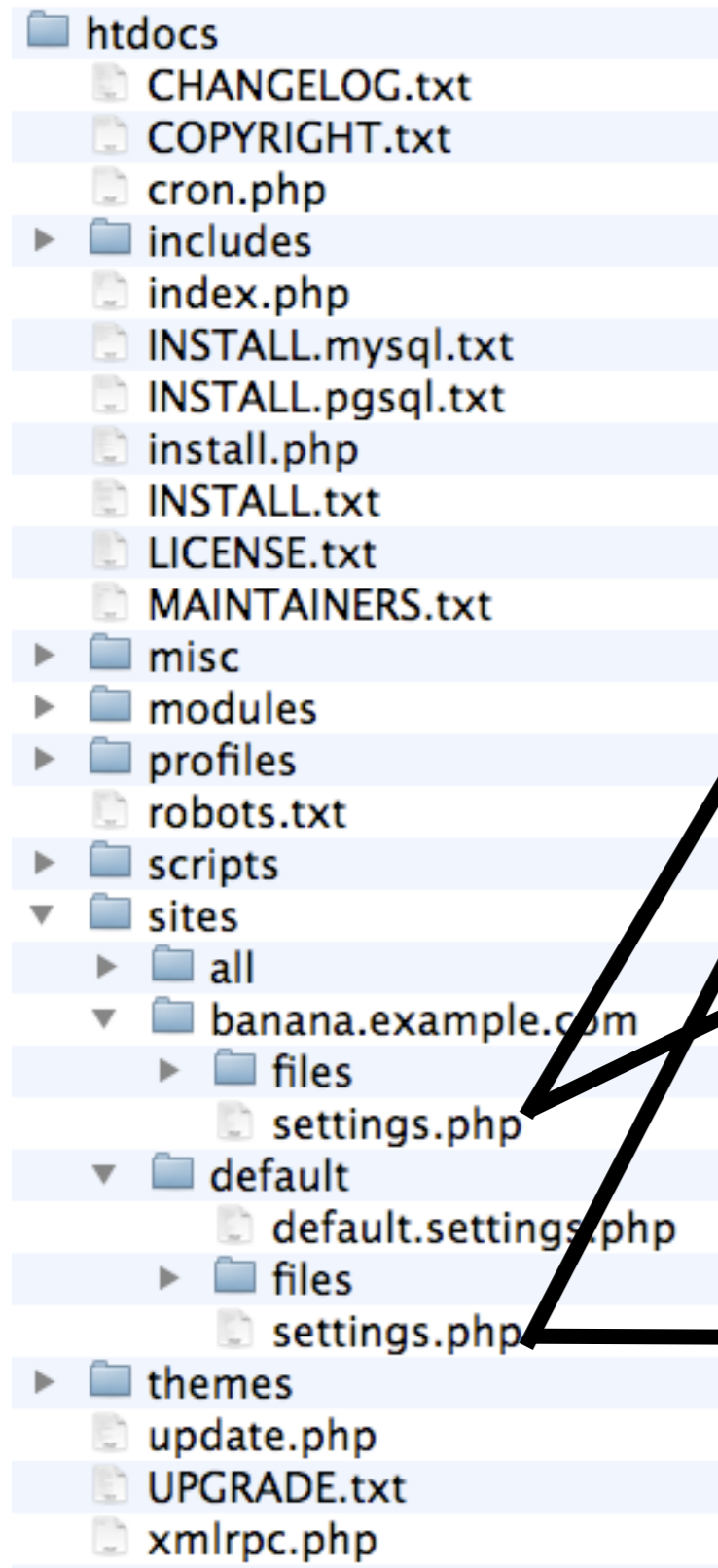
banana



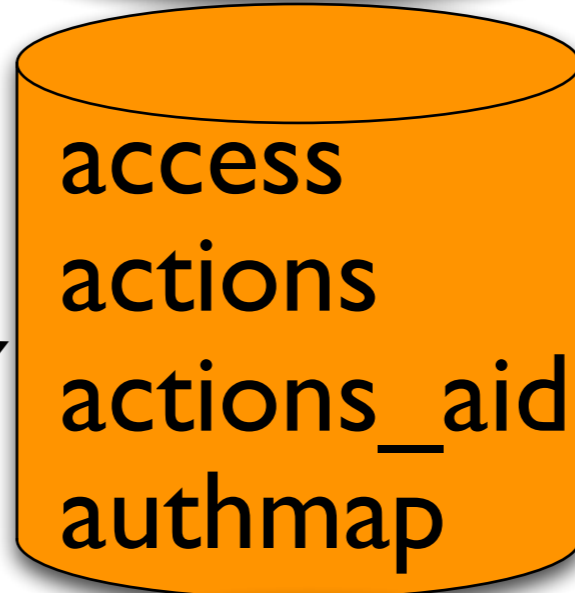
example

...

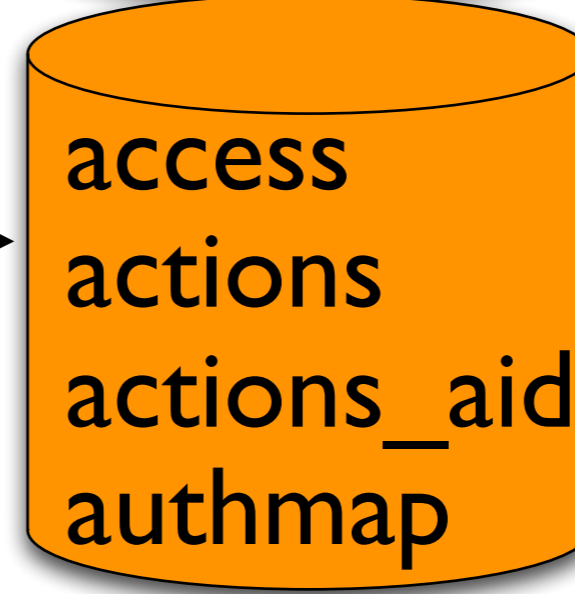
Still no change in behavior!



shared



banana



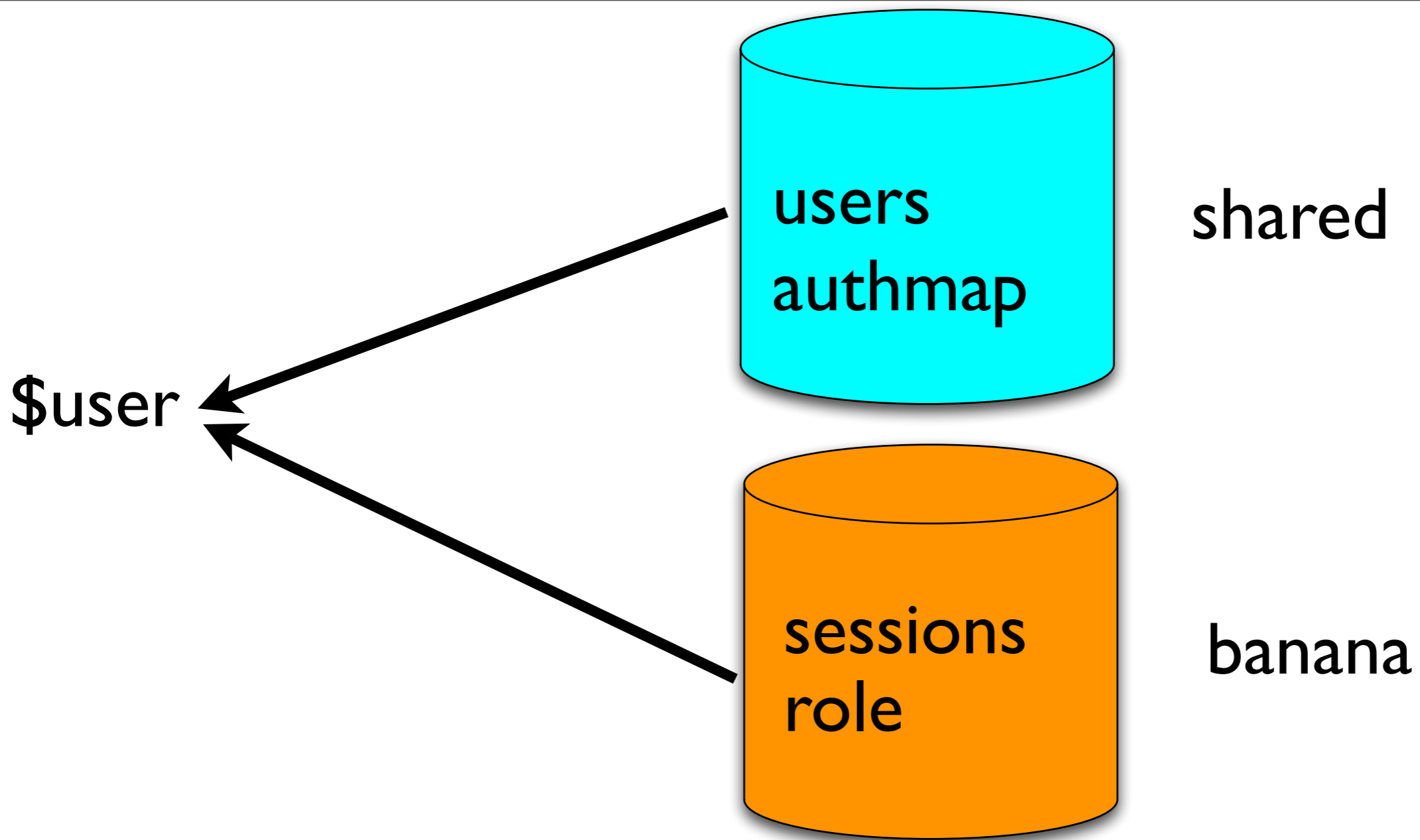
example

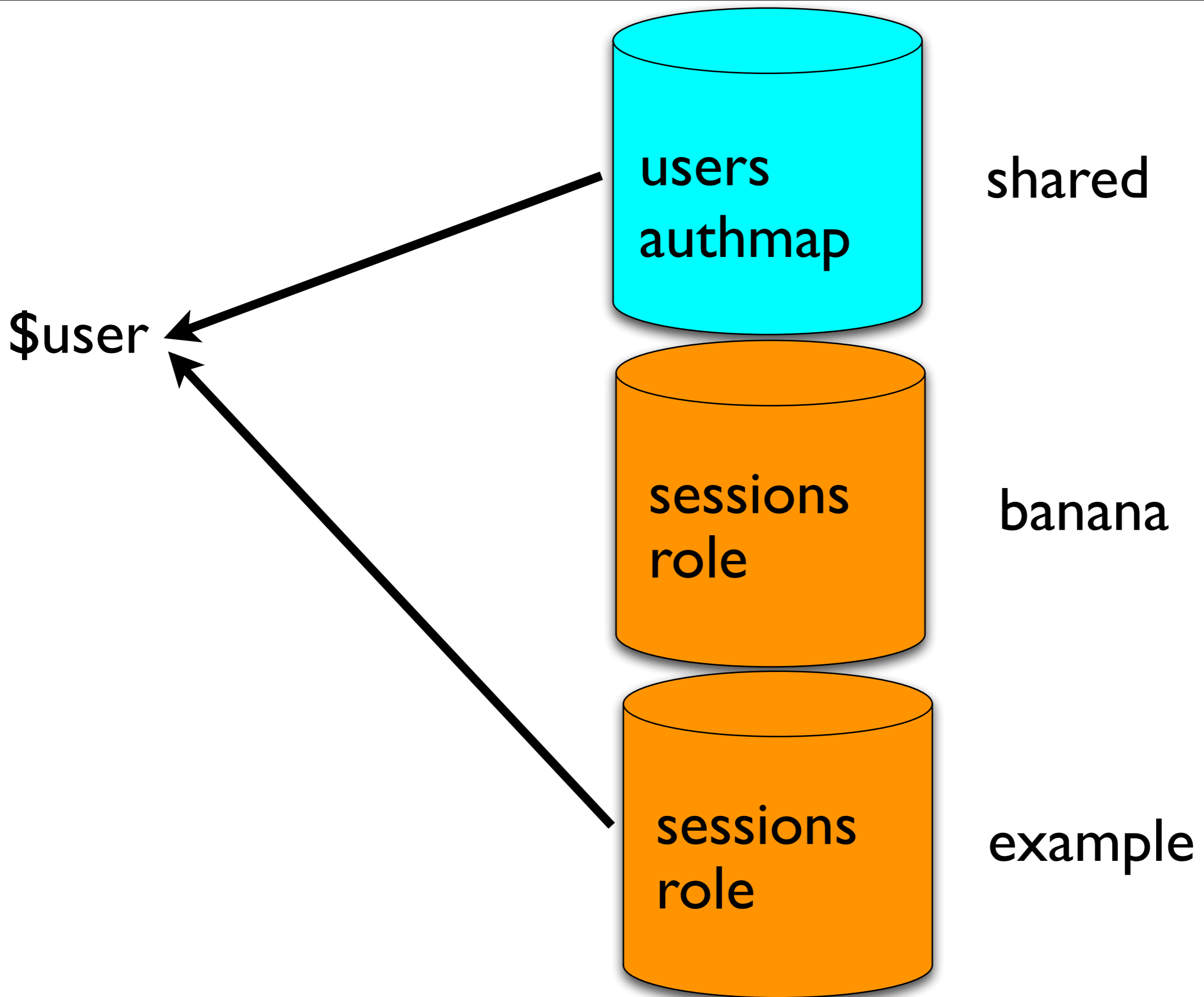
...

Still no change in behavior!

“Logged in” means

- Cookie containing session ID
- Entry in sessions table containing session ID
- Entry in users table





Share user, authmap

Separate login on each site

Simultaneous logins!

User record shared

Roles are *not* shared

Role assignments are *not* shared

Share user, authmap

Separate login on each site

Simultaneous logins!

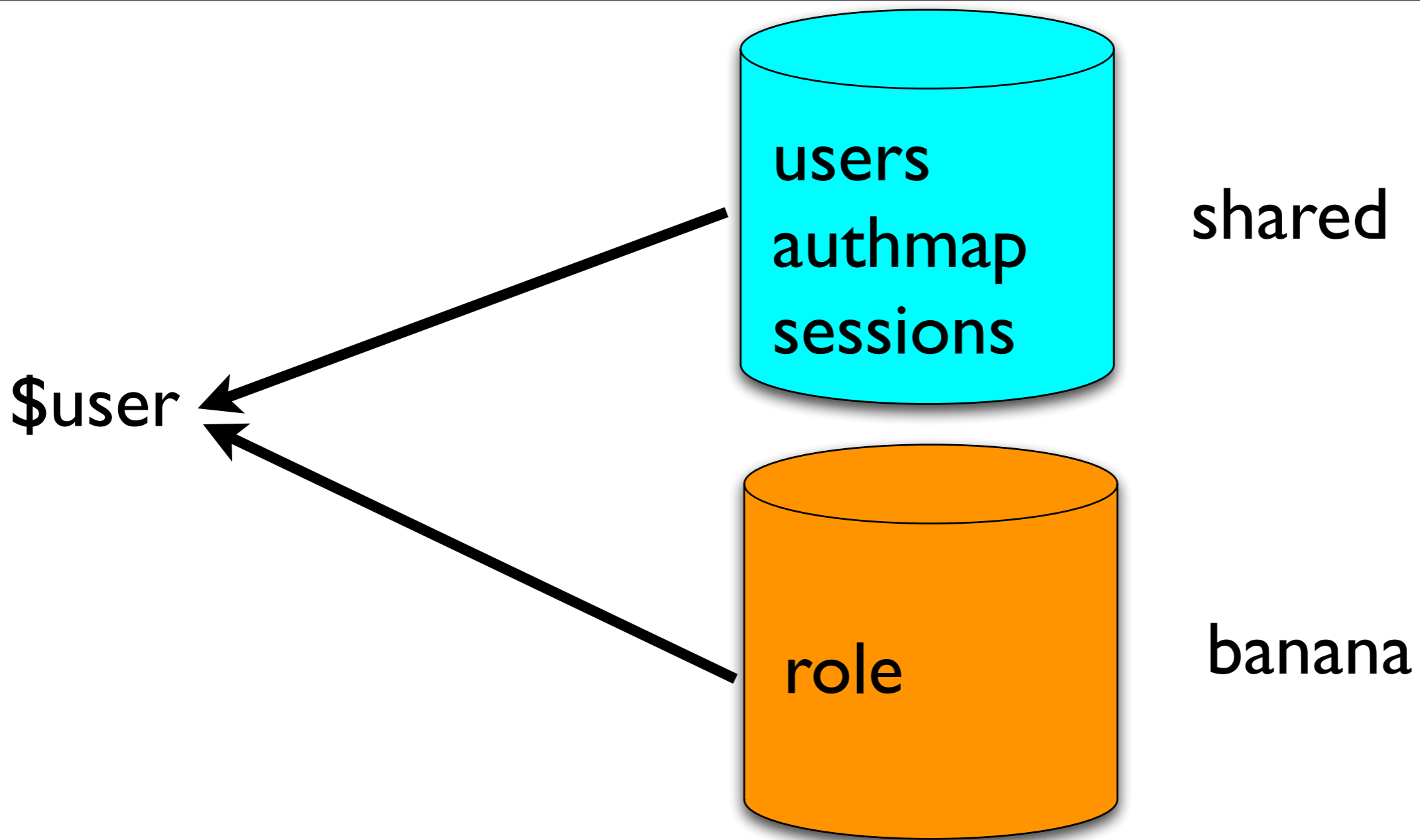
User record shared

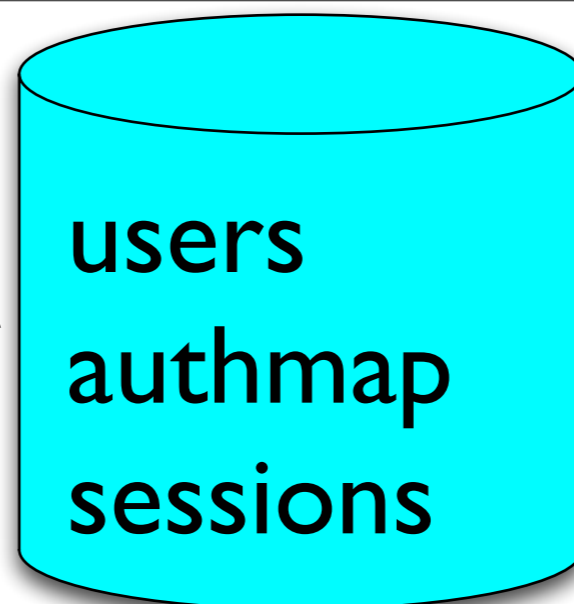
Roles are *not* shared

Role assignments are *not* shared

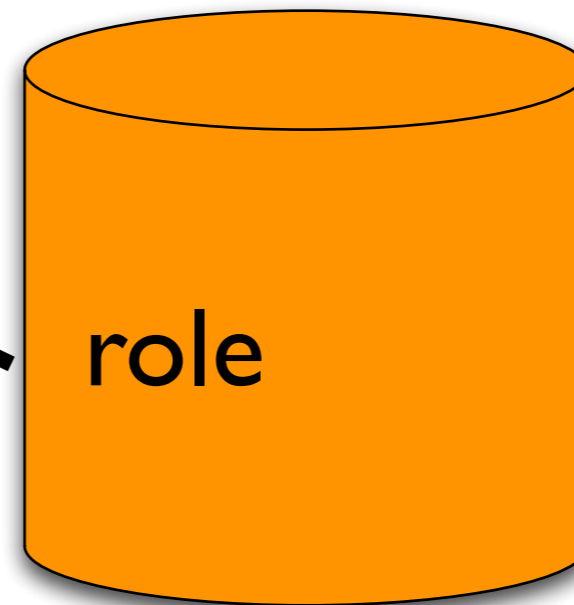
Field
<u>uid</u>
name
pass
mail
mode
sort
threshold
theme
signature
created
access
login
status
timezone
language
picture
init
data

Shared Logins





shared



banana

\$user

```
$db_prefix = array(  
    'default' => '',  
    'users' => 'shared.',  
    'authmap' => 'shared.',  
    'sessions' => 'shared.',  
);
```

What happens?

What happens?

- Log into <http://banana.example.com>

What happens?

- Log into <http://banana.example.com>
- Go to <http://www.example.com>

What happens?

- Log into <http://banana.example.com>
- Go to <http://www.example.com>
- We're not logged in!

What happens?

- Log into <http://banana.example.com>
- Go to <http://www.example.com>
- We're not logged in!
- Huh?

What's wrong with this picture?

SESS0787a3dbcceb3bde85599fd|7a876fa8de:
d977085b070ee|f8def0fa7c2fb26ada

SESS26| |eb8d937302f9383947cd79a86d6888:
26bd9e3|8d607b058|69643|bcfb93b|

```
$cookie_domain = 'example.com'
```

What happens?

What happens?

- Log into <http://banana.example.com>

What happens?

- Log into <http://banana.example.com>
- Go to <http://www.example.com>

What happens?

- Log into <http://banana.example.com>
- Go to <http://www.example.com>
- We're already logged in!

What happens?

- Log into <http://banana.example.com>
- Go to <http://www.example.com>
- We're already logged in!
- Yay!

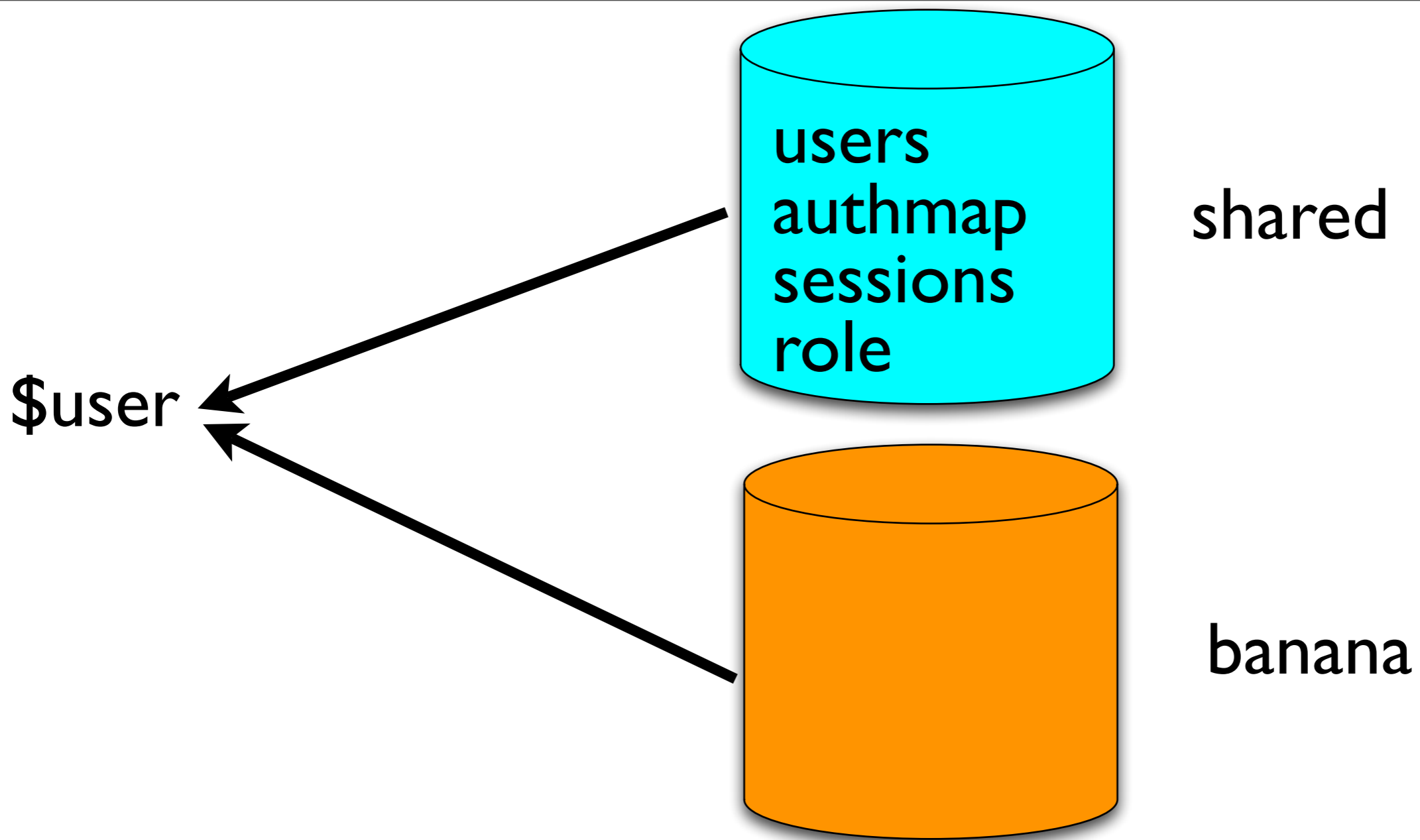
user, authmap, sessions + \$cookie_domain

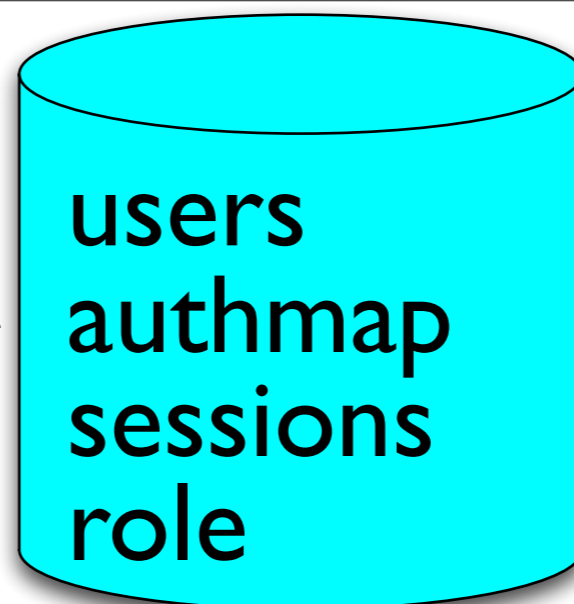
One login for all sites

User record shared

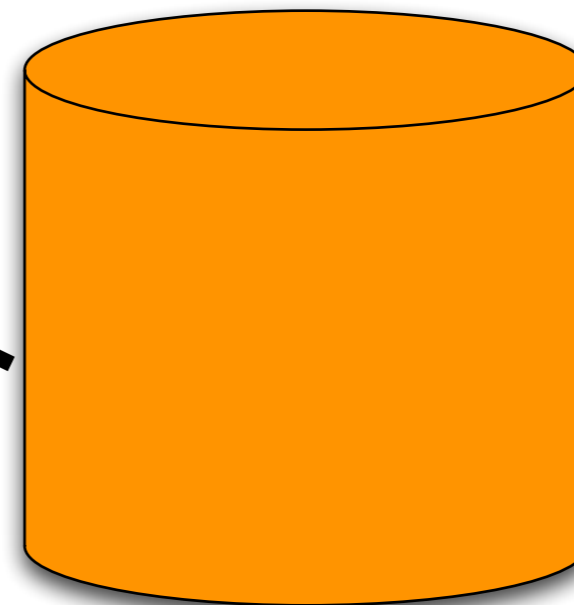
Roles are *not* shared

Role assignments are *not* shared





shared



banana

`$user`

```
$db_prefix = array(  
  'default' => '',  
  'users' => 'shared.',  
  'authmap' => 'shared.',  
  'sessions' => 'shared.',  
  'role' => 'shared.',  
);
```



jvandyk

- My account
- ▷ Create content
- ▽ Administer
 - ▷ Content management
 - ▷ Site building
 - ▷ Site configuration
 - ▽ User management
 - Access rules
 - Permissions
 - Roles
 - User settings
 - Users
 - ▷ Reports
 - Help
- Log out

Roles

Roles allow you to fine tune the security and administration of Drupal. A role defines a group of users that have privileges as defined in [user permissions](#). Examples of roles include: anonymous user, authenticated user, moderator, administrator and so on. In this area you will define the *role names* of the various roles. To delete a role choose

By default, Drupal comes with two user roles:

- Anonymous user: this role is used for users that don't have a user account or that are not authenticated.
- Authenticated user: this role is automatically granted to all logged in users.

Name	Operations	
anonymous user	locked	edit permissions
authenticated user	locked	edit permissions
content reviewer	edit role	edit permissions
devil's advocate	edit role	edit permissions
site editor	edit role	edit permissions

Add role

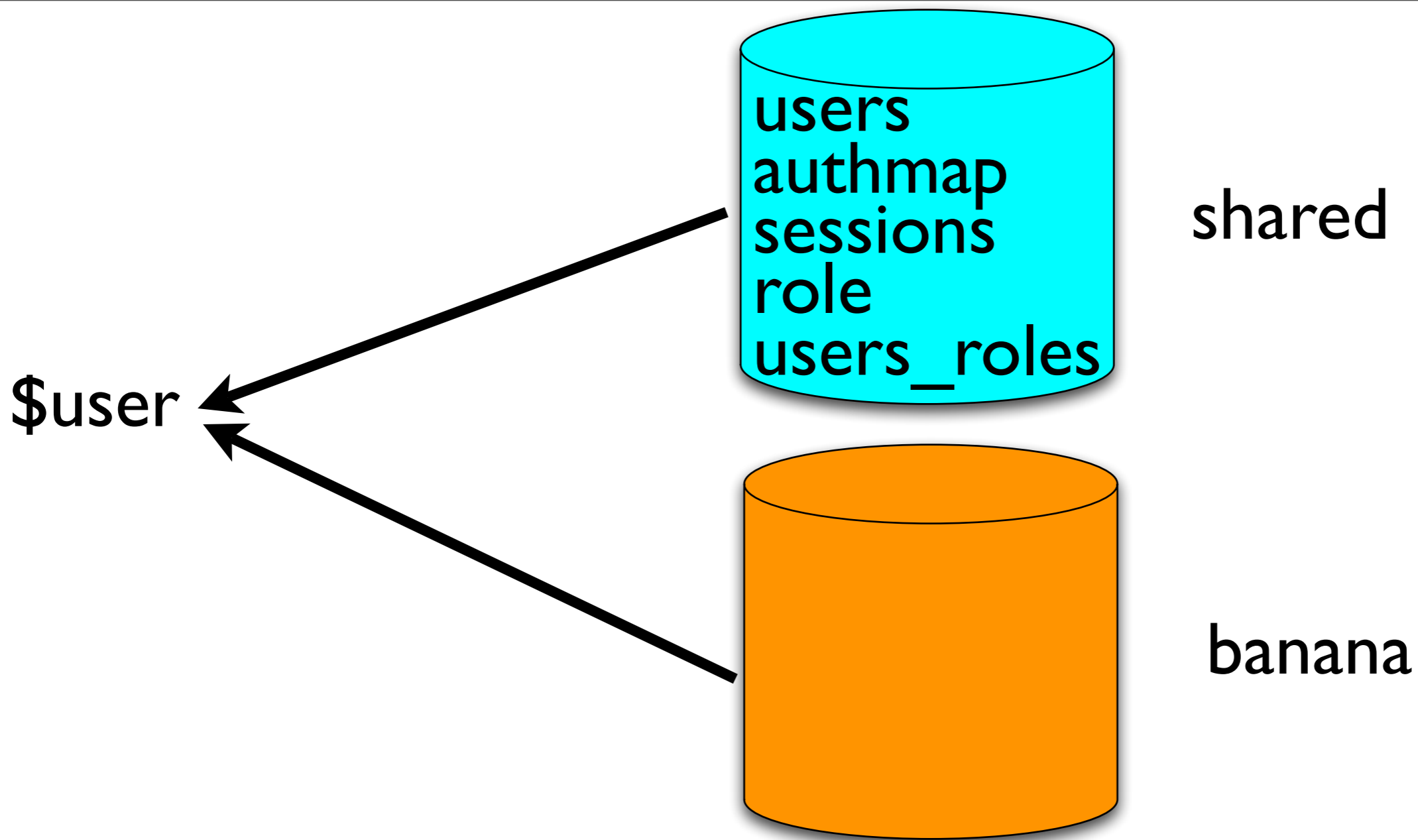
user, authmap, sessions, role +
\$cookie_domain

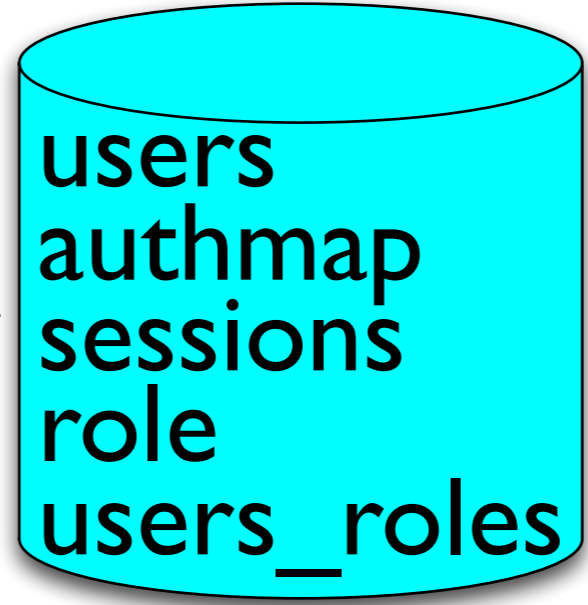
One login for all sites

User record shared

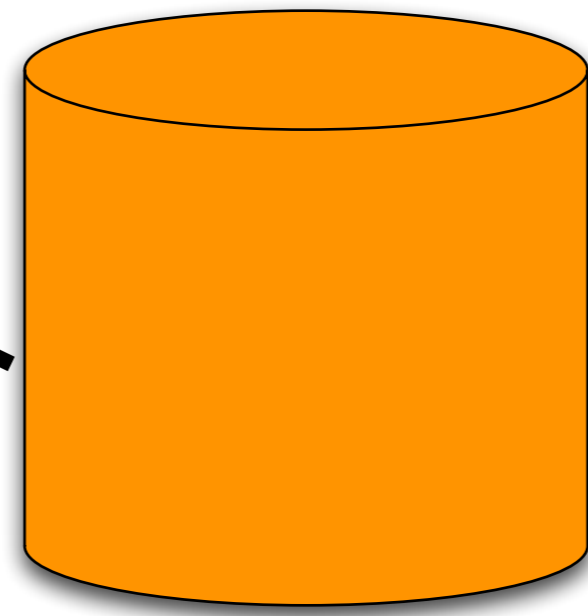
Role names/ids shared

Role assignments are *not* shared





shared



banana

\$user

```
$db_prefix = array(  
    'default' => '',  
    'users' => 'shared.',  
    'authmap' => 'shared.',  
    'sessions' => 'shared.',  
    'role' => 'shared.',  
    'users_roles' => 'shared.',  
);
```

**user, authmap, sessions, role,
users_roles + \$cookie_domain**

One login for all sites

User record shared

Role names/ids shared

Role assignments shared

	User record	Login	Role	Role assignment
user				
+ sessions		*		
+ role				
+ users_role				

Caution!

Caution!

- Table contention

Caution!

- Table contention
- Database updates

Caution!

- Table contention
- Database updates
- Don't share role assignments and not role IDs!

Caution!

- Table contention
- Database updates
- Don't share role assignments and not role IDs!
- Think it out using stories!

What is Multisite?

How Multisite Works

Sharing Information Between Sites