Class 06: PHP and MySQL for dynamic content

What is MySQL?
How to create a MySQL database
Working with cPanel, Pentangle and phpMyAdmin
Creating a database
Adding structure to a database
Adding data to a database
Querying a database
Formatting and printing data with PHP
Building a simple application with PHP and MySQL
Other types of notation (PDO)

References

PHP and MySQL for Dynamic Web Sites 5th Ed. by Larry Ullman
Head First PHP & MySQL by Lynn Beighley and Michael Morrison
PHO & MySQL by Jon Duckett
PHP Solutions: Dynamic Web Design Made Easy 2nd Ed. by David Powers

Homework

Read: Chapters 4, 5 and 9 of PHP and MySQL for Dynamic Web Sites
OR Chapters 10 - 14 of PHP Solutions: Dynamic Web Design Made Easy

You may not need to use a database just yet, but your web hosting account will allow you to create a few, so do experiment. Try to create a simple news system as demonstrated in this week's presentation. Databases can easily be deleted (dropped), so a test database can be removed at a later date.

Note: be very careful not to delete your WordPress database, currently used to store your course blog. You cannot undelete a database!

PHP scripts for connecting to databases and running queries can vary a lot in their syntax – there are many ways of doing it. The *standard* or *procedural* notation in this week's presentation is the same as used in Larry Ullman's book but you may use any method you feel comfortable with (Procedural or PDO) but procedural notation is probably easier to understand for the beginner. It's also worth reiterating that you should avoid the old **mysql** methods and use the newer **mysqli** instead because they are more efficient and more secure (the "i" stands for improved). The standard mysqli notation for connecting to a database is:

```
$conn = new mysqli('localhost', $db_user, $db_password,
$db_name) or die ('Cannot open database');
```

The News Application

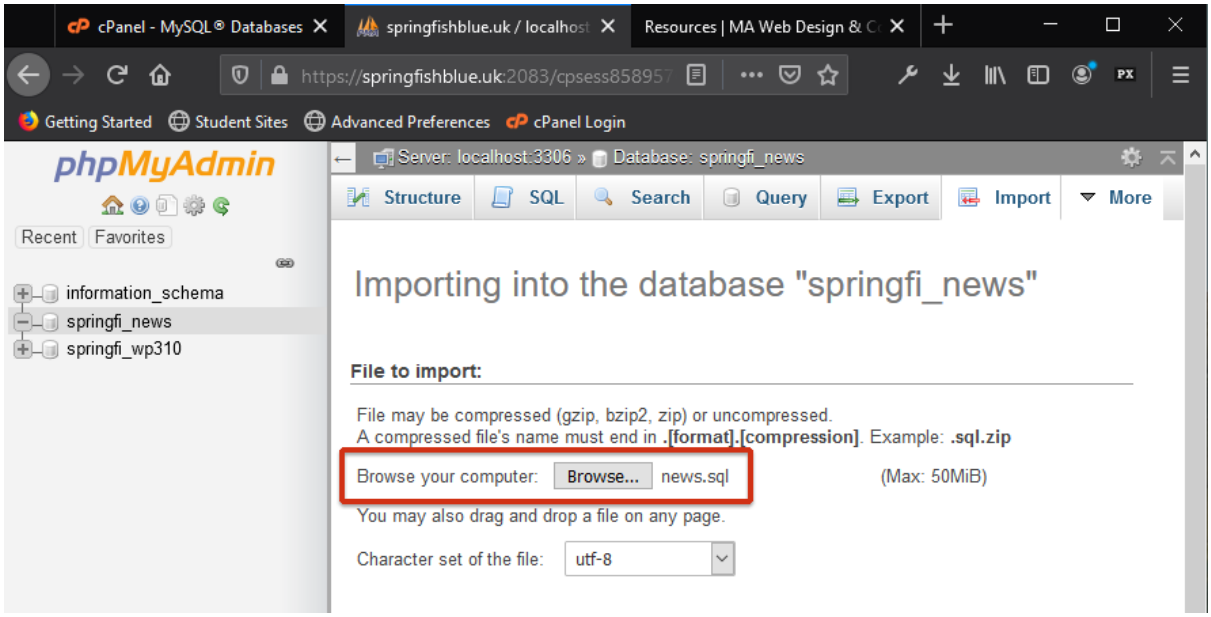
The example news application can be downloaded if you'd like to give it a try.

Create a database and a user in cPanel (note the names and the password). Then import **news.sql** using phpMyAdmin. This will add a news table to the database, including the structure and some sample data.

Upload all files (except **news.sql**) to a folder on your server called "news".

Edit **index.php** to add the database name, username and password to the PHP script.

Visit the news folder in your browser and see the application working.



Try inserting some new data using phpMyAdmin to see how the application updates the news page.

Remember, PHP scripts will only work once uploaded to the web server. They will not work on your local computer, so get used to testing your websites live on the server.

A word about collation and character set

Choosing the correct collation and character set for the text fields in your database can be confusing. There are a lot of options and many of them are language-specific. You might wonder why bother using a language-specific collation, if UTF-8 includes all the characters you might need? The answer is to do with the priorities you might have as a database designer. Some collations will be faster at the expense of accuracy, others will be more accurate at the expense of speed. For example, if you know that all your text will only ever be in a particular language, and won't include special characters such as emojis, then a language-specific collation will provide the best balance between speed and accuracy.

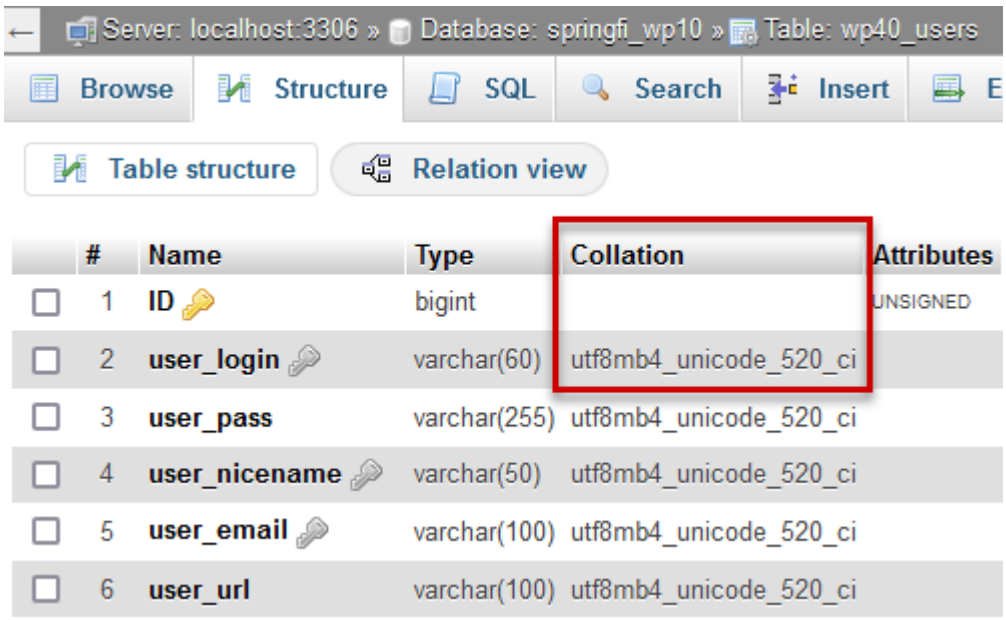
The MySQL documentation describes a collation like this:
A collation is a set of rules that defines how to compare and sort character strings. Each collation in MySQL belongs to a single character set. Every character set has at least one collation, and most have two or more collations.

Course materials: Content Management

In most cases, we will want to use the UTF-8 character set because that is the standard for webpages, but that still leaves a huge choice of collations.

If we know that our text will be mostly in English (but we require some support for other languages) and will contain only the usual alphanumeric characters, then the **UTF8_general_ci** collation is a good, balanced option for both speed and accuracy. However, if we also need full support for supplementary characters, such as emojis, then we should use the **utf8mb4_unicode_520_ci** collation. In doing so we are sacrificing a little speed for greater character set support, but this collation is now the default for MySQL server connections, so it is a sensible option.

WordPress uses the **utf8mb4_unicode_520_ci** collation (if the MySQL server supports it - MySQL 5.6 and above) because the platform is used so widely and in many different language/character contexts (unicode_520 is a newer standard). So, if you want ultimate character set support, you don't mind losing a little speed (which in most cases will be unnoticeable to the user), and your database server supports the latest standard, then that's the one to go for.



An extract from the WordPress *users* table

See MySQL UTF-8 charsets and collations explained for a useful overview, and Chapter 10 of the MySQL documentation for a greater perspective on the topic.