

PHP and MySQL for dynamic content

Content Management

PHP and MySQL for dynamic content

PART 1: INTRODUCING MYSQL

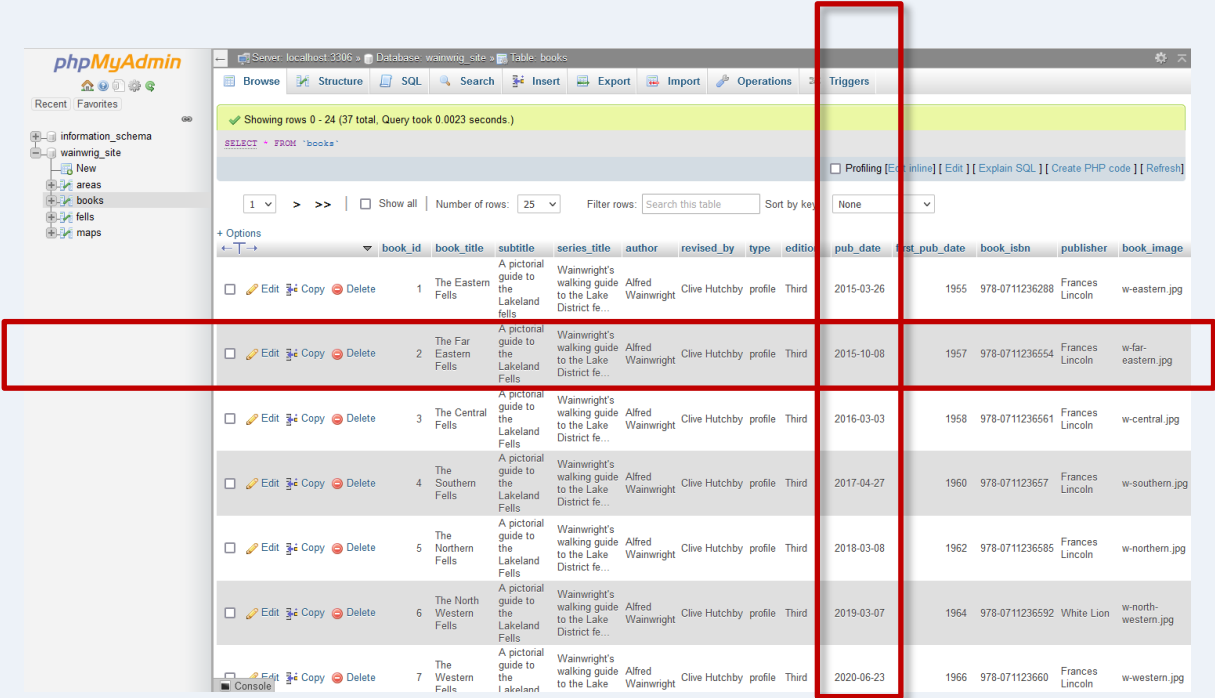
Storing data for web applications

- We know that PHP can store data in variables. For example, a variable called `$date` could be used to hold a date.
- We can even create complex variables called *arrays* that can store multiple bits of data. For example, an array called `$dates` could be used to hold all your family birthdays.
- The problem with variables and arrays is that they can only hold data while PHP scripts are running. Once the page is parsed by the server, the data in the variables is lost.

Permanent data

column

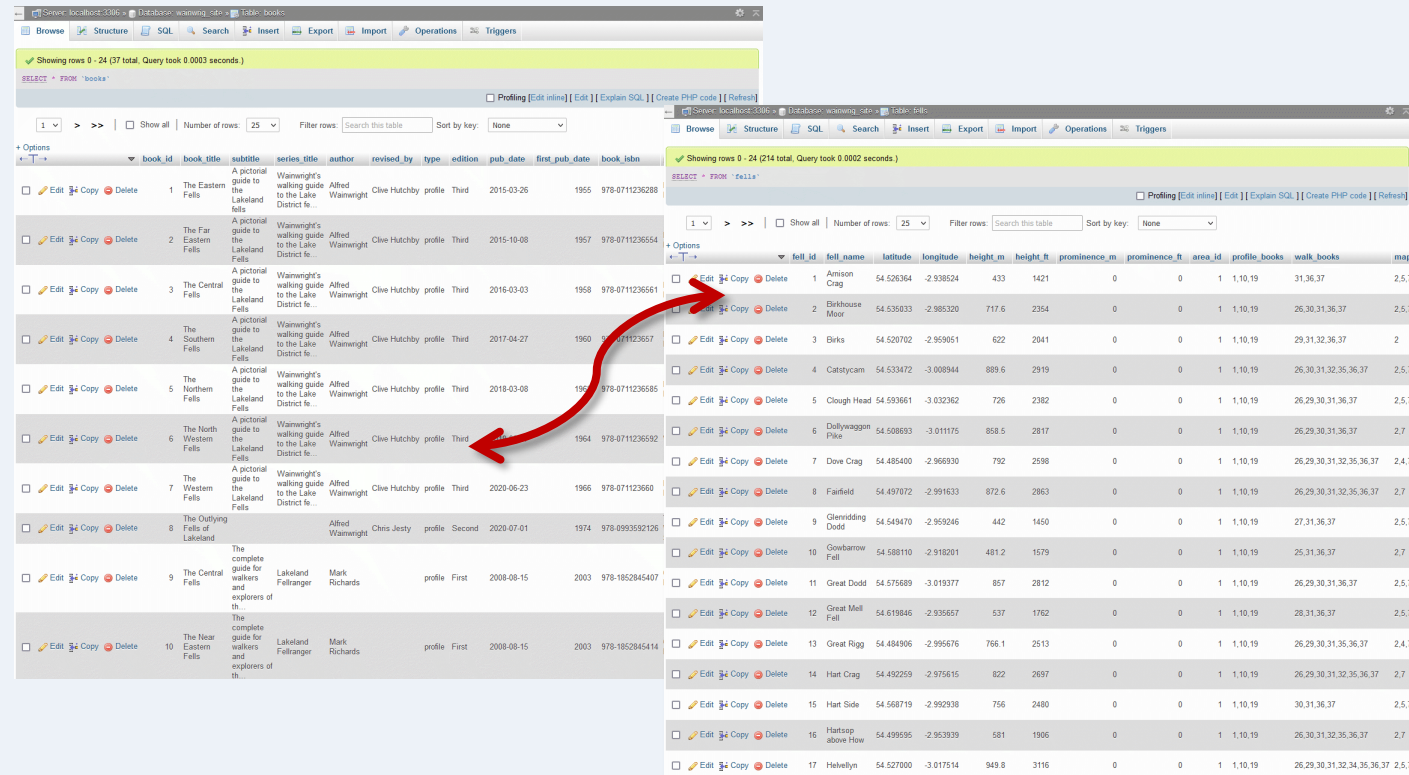
row



	book_id	book_title	subtitle	series_title	author	revised_by	type	edition	pub_date	first_pub_date	book_isbn	publisher	book_image
	1	The Eastern Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fe...	Alfred Wainwright	Clive Hutchby	profile	Third	2015-03-26	1955	978-0711236288	Frances Lincoln	w-eastern.jpg
	2	The Far Eastern Fells	A pictorial guide to the Lakeland Fells	Wainwright's walking guide to the Lake District fe...	Alfred Wainwright	Clive Hutchby	profile	Third	2015-10-08	1957	978-0711236554	Frances Lincoln	w-far-eastern.jpg
	3	The Central Fells	A pictorial guide to the Lakeland Fells	Wainwright's walking guide to the Lake District fe...	Alfred Wainwright	Clive Hutchby	profile	Third	2016-03-03	1958	978-0711236561	Frances Lincoln	w-central.jpg
	4	The Southern Fells	A pictorial guide to the Lakeland Fells	Wainwright's walking guide to the Lake District fe...	Alfred Wainwright	Clive Hutchby	profile	Third	2017-04-27	1960	978-0711236567	Frances Lincoln	w-southern.jpg
	5	The Northern Fells	A pictorial guide to the Lakeland Fells	Wainwright's walking guide to the Lake District fe...	Alfred Wainwright	Clive Hutchby	profile	Third	2018-03-08	1962	978-0711236585	Frances Lincoln	w-northern.jpg
	6	The North Western Fells	A pictorial guide to the Lakeland Fells	Wainwright's walking guide to the Lake District fe...	Alfred Wainwright	Clive Hutchby	profile	Third	2019-03-07	1964	978-0711236592	White Lion	w-north-western.jpg
	7	The Western Fells	A pictorial guide to the Lakeland Fells	Wainwright's walking guide to the Lake District fe...	Alfred Wainwright	Clive Hutchby	profile	Third	2020-06-23	1966	978-0711236600	Frances Lincoln	w-western.jpg

- A database is a way to store data permanently on a web server where it can be retrieved by PHP at any time.
- Data is stored in a database *table*. A table is very much like a spreadsheet and is composed of *columns* and *rows*.

What is a database?



The image shows two database management tool windows. The left window displays a table named 'books' with 37 rows. The right window displays a table named 'fells' with 17 rows. A red arrow points from the 6th row of the 'books' table to the 6th row of the 'fells' table, illustrating a relationship between the two tables.

books table:

book_id	book_title	subtitle	series_title	author	revised_by	type	edition	pub_date	first_pub_date	book_isbn
1	The Eastern Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Clive Hutchby	profile	Third	2015-03-26	1955	978-0711236288
2	The Far Eastern Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Clive Hutchby	profile	Third	2015-10-08	1957	978-0711236554
3	The Central Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Clive Hutchby	profile	Third	2016-03-03	1958	978-0711236561
4	The Southern Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Clive Hutchby	profile	Third	2017-04-27	1960	978-071123657
5	The Northern Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Clive Hutchby	profile	Third	2018-03-08	1961	978-0711236585
6	The North Western Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Clive Hutchby	profile	Third	2019-03-08	1964	978-0711236592
7	The Western Fells	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Clive Hutchby	profile	Third	2020-06-23	1966	978-071123660
8	The Outlying Fells of Lakeland	A pictorial guide to the Lakeland fells	Wainwright's walking guide to the Lake District fells	Alfred Wainwright	Chris Jesty	profile	Second	2020-07-01	1974	978-0993592126
9	The Central Fells	The complete guide for walkers and explorers of th...	Lakeland Fellranger	Mark Richards		profile	First	2008-08-15	2003	978-1852845407
10	The Near Eastern Fells	The complete guide for walkers and explorers of th...	Lakeland Fellranger	Mark Richards		profile	First	2008-08-15	2003	978-1852845414

fells table:

fell_id	fell_name	latitude	longitude	height_m	height_ft	prominence_m	prominence_ft	area_id	profile_books	walk_books	map
1	Amison Crag	54.526364	-2.938524	433	1421	0	0	1	1,10,19	31,36,37	2,5,7
2	Birkhouse Moor	54.535033	-2.985320	717	2354	0	0	1	1,10,19	26,30,31,36,37	2,5,7
3	Birks	54.520702	-2.959051	622	2041	0	0	1	1,10,19	29,31,32,36,37	2
4	Catstycam	54.533472	-3.008944	889	2919	0	0	1	1,10,19	26,30,31,32,35,36,37	2,5,7
5	Clough Head	54.593661	-3.032362	726	2382	0	0	1	1,10,19	26,29,30,31,36,37	2,5,7
6	Dalbywasgill Pike	54.508893	-3.011175	858	2817	0	0	1	1,10,19	26,29,30,31,36,37	2,7
7	Dove Crag	54.484400	-2.966930	792	2598	0	0	1	1,10,19	26,29,30,31,32,35,36,37	2,4,7
8	Fairfield	54.497072	-2.991633	872	2863	0	0	1	1,10,19	26,29,30,31,32,35,36,37	2,7
9	Glenridding Dodd	54.549470	-2.959246	442	1450	0	0	1	1,10,19	27,31,36,37	2,5,7
10	Gowbarrow Fell	54.588110	-2.918201	481	1579	0	0	1	1,10,19	25,31,36,37	2,7
11	Great Dodd	54.575689	-3.019377	857	2812	0	0	1	1,10,19	26,29,30,31,36,37	2,5,7
12	Great Mell Fell	54.619846	-2.935657	537	1762	0	0	1	1,10,19	28,31,36,37	2,5,7
13	Great Rigg	54.484906	-2.995676	766	2513	0	0	1	1,10,19	26,29,30,31,35,36,37	2,4,7
14	Hart Crag	54.492259	-2.975615	822	2697	0	0	1	1,10,19	26,29,30,31,32,35,36,37	2,7
15	Hart Side	54.568719	-2.992938	756	2480	0	0	1	1,10,19	30,31,36,37	2,5,7
16	Harrop above How	54.499595	-2.953939	581	1906	0	0	1	1,10,19	26,30,31,32,35,36,37	2,7
17	Helvellyn	54.527000	-3.017514	943	3116	0	0	1	1,10,19	26,29,30,31,32,34,35,36,37	2,5,7

- A *database* may contain one table or many tables.
- Data in one table can have a specific **relationship** with data in another table.
- Hence the term *relational database*.

See: [relational database](#)

What is MySQL?

- MySQL is a relational database management system that supports the SQL language.
- SQL stands for “Structured Query Language”.
- MySQL has a close relationship with PHP.
- PHP has several functions specifically designed to work with MySQL.
- Like PHP, it is open source and therefore free.
- MySQL is the “M” in LAMP*.



*Linux, Apache (web server), MySQL, PHP – see [LAMP](#)

LAMP

Linux
Apache
MySQL
PHP

operating system
web server
database
scripting

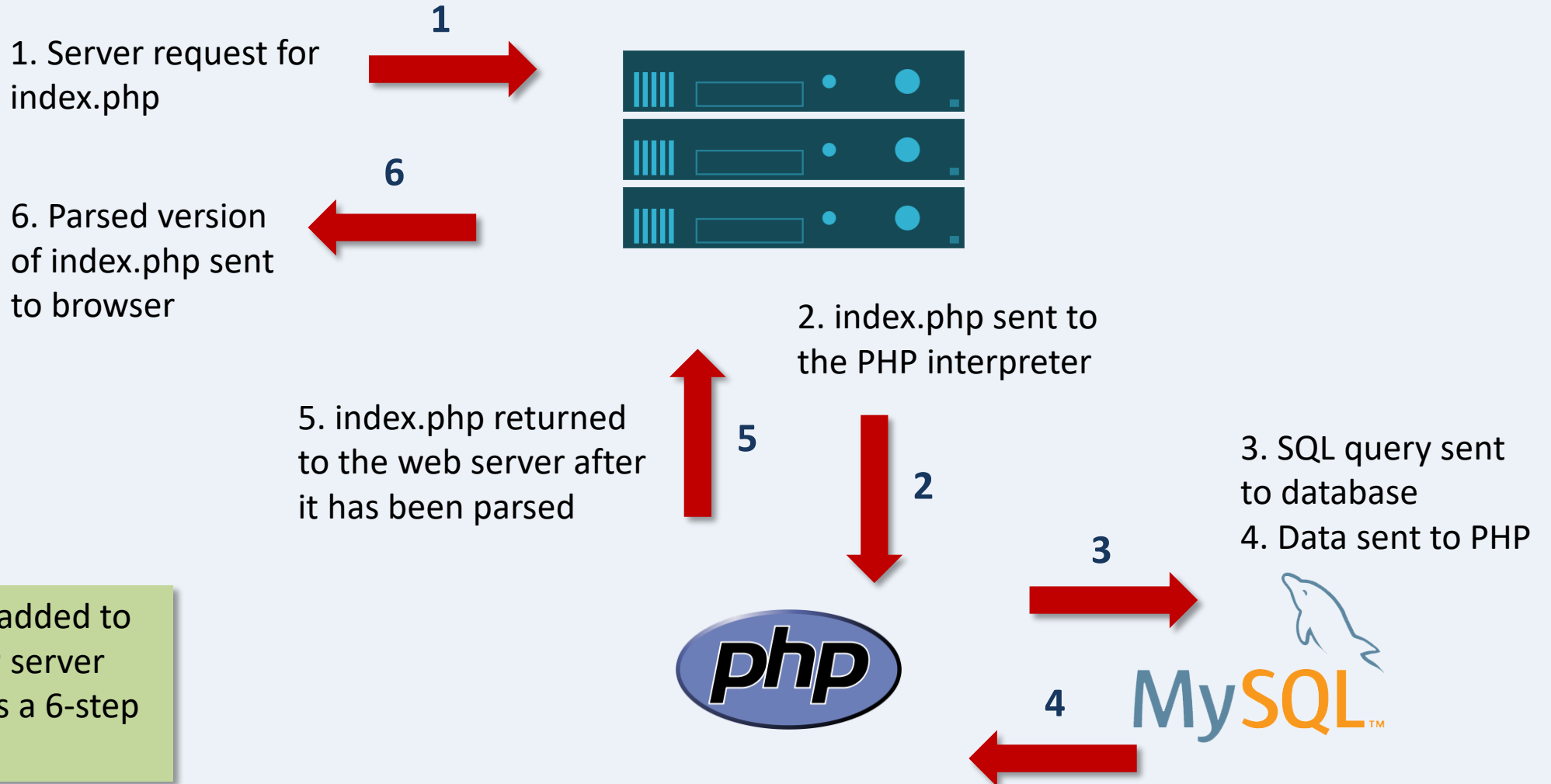
How do I work with MySQL?

- Like PHP, MySQL can be downloaded and installed on a desktop computer*.
- However, like PHP it is designed to run on a web server where it is used to store data.
- Most web hosts who provide Linux hosting also provide PHP and MySQL as part of their offering.
- PHP and MySQL have grown up together and each has been developed to work well with the other.

*see [WAMP](#) or [MAMP](#)



The PHP and MySQL server request

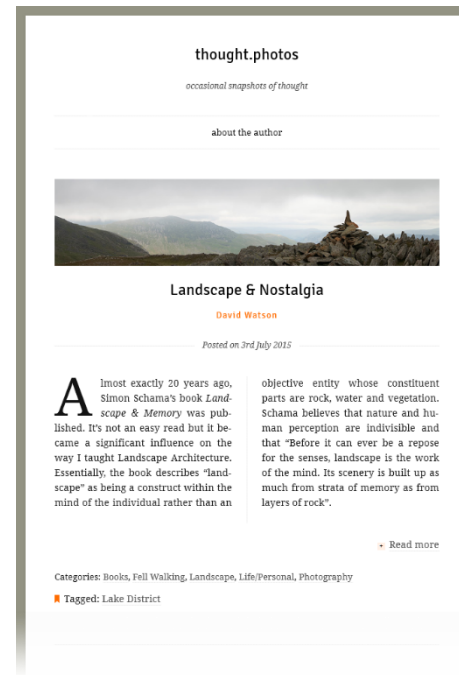


When MySQL is added to the mix, the PHP server request becomes a 6-step process.

How can MySQL be used?

- Almost all the web applications that we are familiar with use MySQL databases. Content management platforms like Wordpress, Drupal, Perch, and Craft, all use MySQL databases. Community forums like vBulletin, and Invision Community Suite, and wiki collaborative publishing platforms like MediaWiki all use MySQL to store their data.
- However, PHP is required to select and extract data from MySQL databases and to construct dynamic web pages from it using PHP/HTML template pages.
- For example, in a content management system, all the content of articles (text and links to images), the date, the name of the author etc. are stored in the database and are extracted and compiled “on the fly” using PHP each time the page is requested by a browser.

Template-driven websites



On template driven sites, the template file is a mixture of HTML for structure and PHP to pull in content dynamically from a database. It's called a template because one file can be used to create many pages.

PHP and MySQL for dynamic content

PART 2: BUILDING A DATABASE

3 steps to building a database

1. Create the database and the user*

usually using your web hosting control panel

2. Add structure to the database

tables and fields, usually using PHPMyAdmin

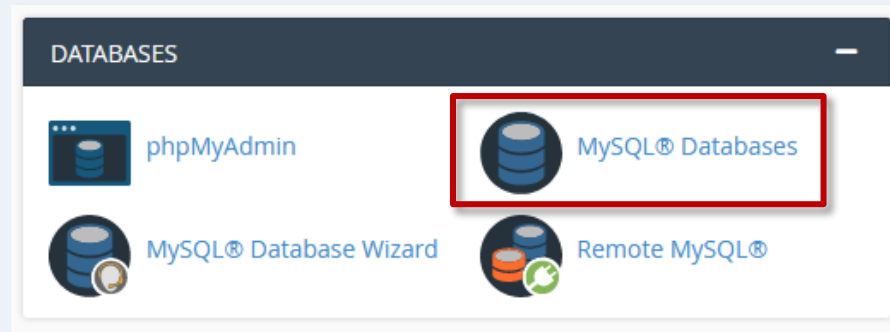
3. Add data to the database

website content, usually using PHPMyAdmin

*If you are manually installing a web application, like WordPress, you still need to create the database and user, but the installation procedure will automatically structure and populate the database for you. If you're building your own bespoke applications, you'll need to follow all 3 steps in the procedure above.

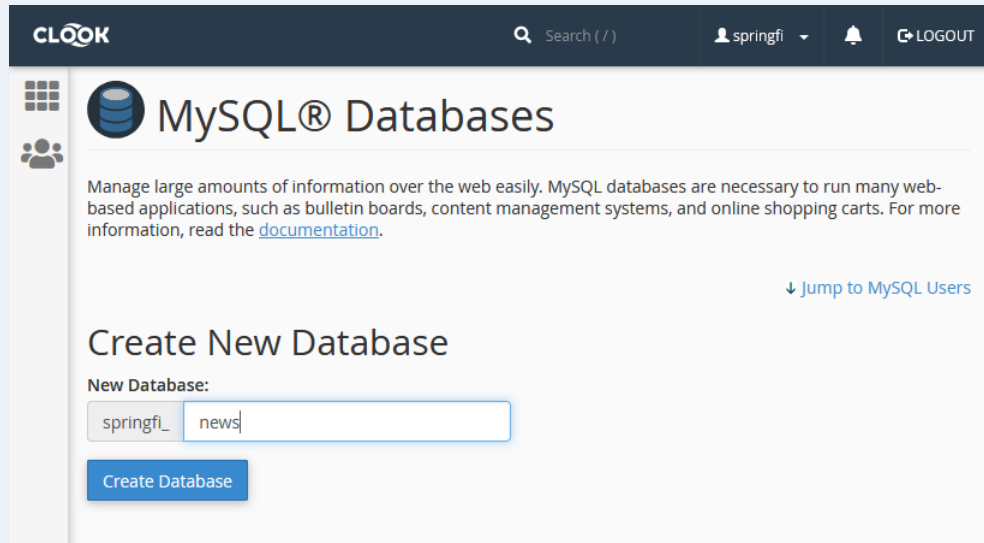
Step 1: How do I create a database?

- Databases are usually created using the control panel provided by your web host.
- In most cases, on a Linux server, that will be cPanel.
- You can use either the main *MySQL Databases* page or the *MySQL Database Wizard* to create a new database.
- Some hosts may use different control panels.



The Databases card in cPanel

Create database and user in cPanel



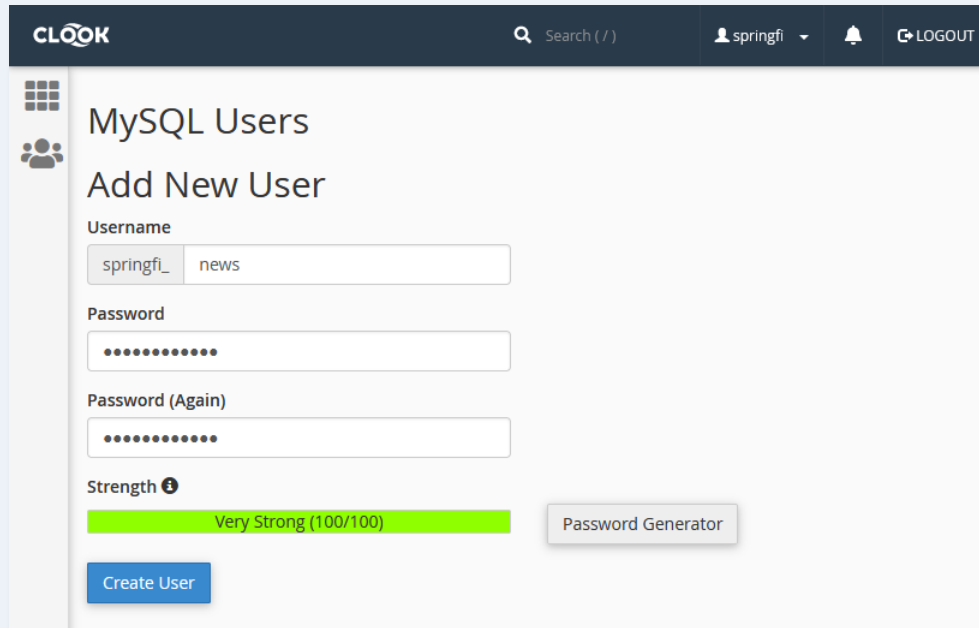
Creating a new database is a simple, 3-step process...

STEP 1: Just enter a name for your database and click “Create Database”.

In most cases, there will be a name prefix which is fixed and specific to the domain you are working on. This ensures that the name you use is not the same as the name of another database on the same server.

During the 3-step process, you will need to make a note of three key bits of information: the *database name*, the *user name* and the *user password*. You will need this information to install applications such as WordPress.

Create database and user in cPanel



The screenshot shows the cPanel interface for managing MySQL users. The top navigation bar includes the cPanel logo, a search bar, a user profile dropdown for 'springfi', a notification bell, and a 'LOGOUT' button. The main content area is titled 'MySQL Users' and 'Add New User'. It contains a form with the following fields: 'Username' (split into 'springfi_' and 'news'), 'Password' (masked with dots), and 'Password (Again)' (also masked). Below the password fields is a 'Strength' indicator showing a green bar and the text 'Very Strong (100/100)'. To the right of the strength indicator is a 'Password Generator' button. At the bottom of the form is a blue 'Create User' button.

STEP 2: Create a database user.

To connect to a database, you need to specify a *User* who is allowed to access the database and that user must have a password.

So, to create a user, enter a name and a password and click “Create User”.

The password should be strong. Use the password generator. You never need to remember this password so it can simply be a random selection of letters, numbers and symbols.

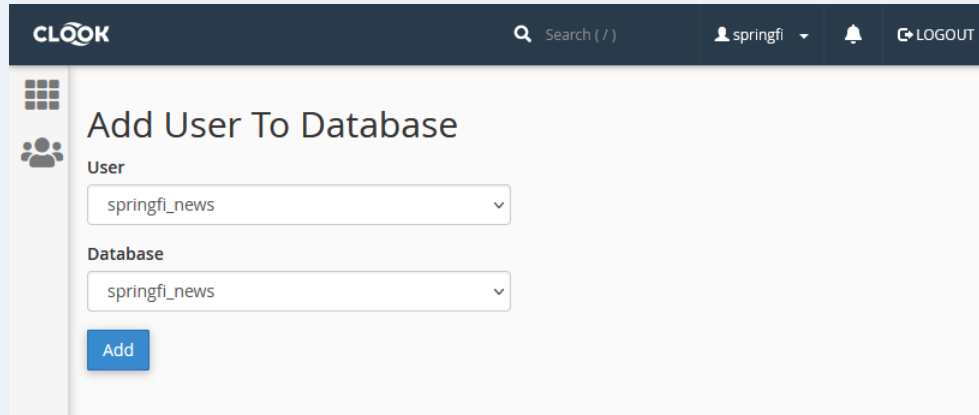
Create database and user in cPanel

STEP 3: Add the user to the database.

Note: if you are using the MySQL Database Wizard, you can omit this step, the wizard adds the user to the database for you.

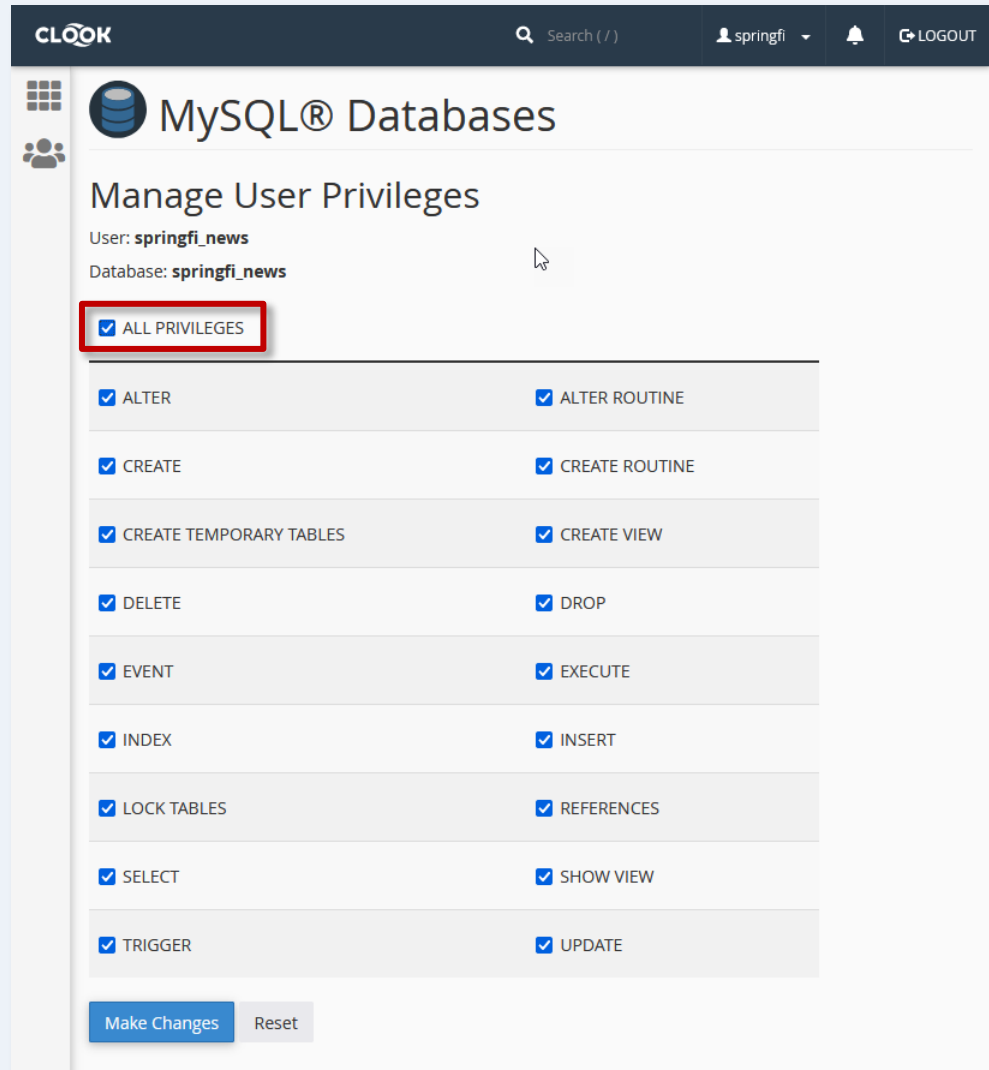
Select the user name and the database name from the two drop-down lists and then click “Add”.

You will then be taken to a screen where you can specify the access privileges that this user will have...



The screenshot shows the 'Add User To Database' interface in a cPanel-like environment. At the top, there is a dark header with the 'CLOOK' logo, a search bar, a user profile dropdown labeled 'springfi', a bell icon, and a 'LOGOUT' button. The main content area has a light beige background. On the left, there is a sidebar with a grid icon and a user icon. The title 'Add User To Database' is prominently displayed. Below the title, there are two dropdown menus: the first is labeled 'User' and contains the text 'springfi_news'; the second is labeled 'Database' and also contains 'springfi_news'. At the bottom left of the form, there is a blue button labeled 'Add'.

Create database and user in cPanel

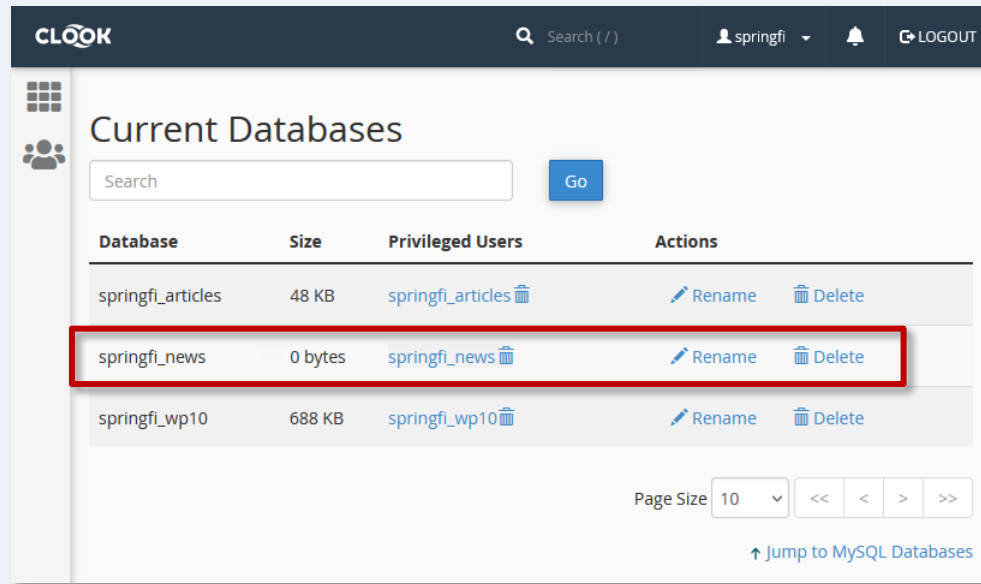


In most cases, you can assign your user to have “All Privileges” but for best security, you may want to restrict privileges to only those required. Click “Make Changes” to complete the action.

Your database is now ready to use and if you are installing a CMS such as WordPress, that’s all you need to do because the install routine will create the structure and add sample content.

However, if you build your own CMS, you’ll need to add the structure yourself before data can be added.

Create database and user in cPanel



Database	Size	Privileged Users	Actions
springfi_articles	48 KB	springfi_articles	Rename Delete
springfi_news	0 bytes	springfi_news	Rename Delete
springfi_wp10	688 KB	springfi_wp10	Rename Delete

Page Size 10 << < > >>

[Jump to MySQL Databases](#)

Now that you've completed the 3-step process, you will see your new database listed along with the user that you created and added to it. Notice that the size of the database is given as "0 bytes". That's because the new database currently has no structure and therefore no data.

Notice that should you ever need to delete a database, you can do it using the Delete option in this list.

If this database is for use with your own project, it's now time to add the structure...

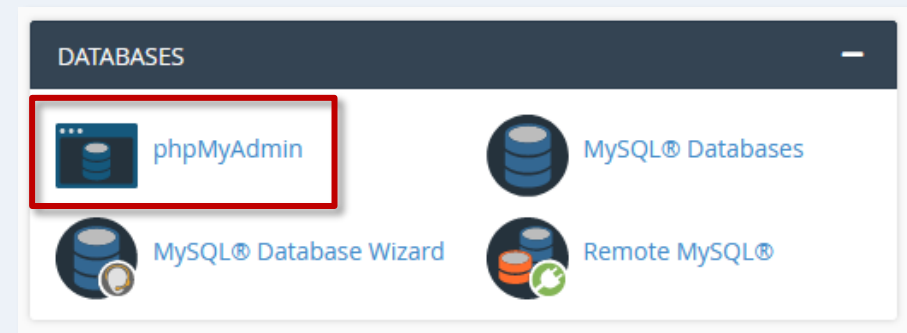
PHP and MySQL for dynamic content

PART 3: ADDING STRUCTURE TO A DATABASE

Normally, you'll be creating databases for use with a CMS or other application but it's useful to know how to structure them manually, using an application called phpMyAdmin...

What is phpMyAdmin?

- Despite its name, phpMyAdmin is an open source administration tool for working with MySQL databases.
- Structure (tables and fields) needs to be added to a database so that it can hold data.
- phpMyAdmin is used to structure your database – to add tables and fields.
- phpMyAdmin can also be used to add data to your database once it is structured.



phpMyAdmin homepage

The screenshot shows the phpMyAdmin interface for a server at localhost:3306. The top navigation bar includes links for Databases, SQL, Status, Export, Import, Settings, Variables, Charsets, Engines, and Plugins. The left sidebar displays a list of databases: information_schema, springfi_articles, springfi_news, and springfi_wp10. A red arrow points from a green callout box to this list. The main content area is divided into several panels: General settings (Server connection collation: utf8mb4_unicode_ci), Appearance settings (Language: English, Theme: pmahomme, Font size: 82%), Database server (Server: Localhost via UNIX socket, Server type: MySQL, Server connection: SSL is not being used, Server version: 8.0.28 - MySQL Community Server - GPL, Protocol version: 10, User: springfi@localhost, Server charset: (utf8mb3)), Web server (cpsrvd 11.94.0.23, Database client version: libmysql - 5.6.43, PHP extension: mysqli, curl, mbstring, PHP version: 7.3.32), and phpMyAdmin (Version information: 4.9.7, Documentation, Official Homepage, Contribute, Get support, List of changes, License).

phpMyAdmin

Recent Favorites

- information_schema
- springfi_articles
- springfi_news
- springfi_wp10

Server: localhost:3306

Databases SQL Status Export Import Settings Variables Charsets Engines Plugins

General settings

Server connection collation: utf8mb4_unicode_ci

Appearance settings

Language: English

Theme: pmahomme

Font size: 82%

More settings

Database server

- Server: Localhost via UNIX socket
- Server type: MySQL
- Server connection: SSL is not being used
- Server version: 8.0.28 - MySQL Community Server - GPL
- Protocol version: 10
- User: springfi@localhost
- Server charset: (utf8mb3)

Web server

- cpsrvd 11.94.0.23
- Database client version: libmysql - 5.6.43
- PHP extension: mysqli curl mbstring
- PHP version: 7.3.32

phpMyAdmin

- Version information: 4.9.7
- Documentation
- Official Homepage
- Contribute
- Get support
- List of changes
- License

All your databases are listed here, click the name to see the tables.

See: [phpMyAdmin](https://www.phpmyadmin.net/)

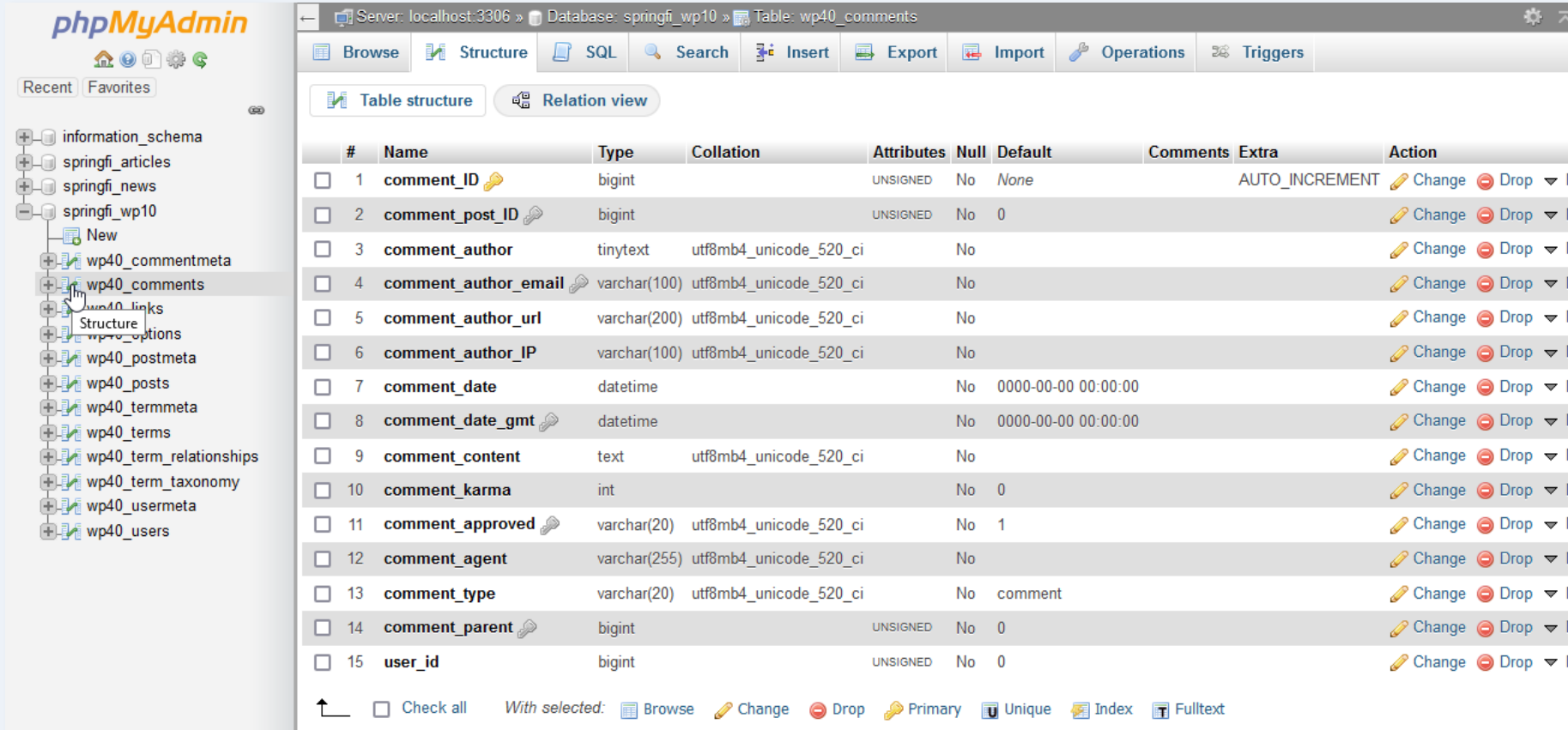
A WordPress database

The screenshot shows the phpMyAdmin interface. On the left, a tree view lists databases: 'information_schema', 'springfi_articles', 'springfi_news', and 'springfi_wp10'. Under 'springfi_wp10', there is a 'New' button and a list of 12 tables: 'wp40_commentmeta', 'wp40_comments', 'wp40_links', 'wp40_options', 'wp40_postmeta', 'wp40_posts', 'wp40_termmeta', 'wp40_terms', 'wp40_term_relationships', 'wp40_term_taxonomy', 'wp40_usermeta', and 'wp40_users'. The main panel shows the 'Structure' tab for the 'springfi_wp10' database. It includes a search filter and a table list with columns: Table, Action, Rows, Type, Collation, Size, and Overhead. The table list shows 12 tables, all using InnoDB engine and utf8mb4_unicode_520_ci collation. The total size of the tables is 688.0 KiB.

Table	Action	Rows	Type	Collation	Size	Overhead
wp40_commentmeta	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_520_ci	48.0 KiB	-
wp40_comments	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_520_ci	96.0 KiB	-
wp40_links	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_520_ci	32.0 KiB	-
wp40_options	Browse Structure Search Insert Empty Drop	132	InnoDB	utf8mb4_unicode_520_ci	96.0 KiB	-
wp40_postmeta	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_unicode_520_ci	48.0 KiB	-
wp40_posts	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_unicode_520_ci	80.0 KiB	-
wp40_termmeta	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_520_ci	48.0 KiB	-
wp40_terms	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_520_ci	48.0 KiB	-
wp40_term_relationships	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_520_ci	32.0 KiB	-
wp40_term_taxonomy	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_520_ci	48.0 KiB	-
wp40_usermeta	Browse Structure Search Insert Empty Drop	18	InnoDB	utf8mb4_unicode_520_ci	48.0 KiB	-
wp40_users	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_520_ci	64.0 KiB	-
12 tables	Sum	162	InnoDB	utf8mb4_unicode_520_ci	688.0 KiB	0 B

This is an example of a WordPress database. It uses 12 tables. In this example, all table names have a “wp40_” prefix to identify them as part of the WordPress dataset. You can click the Structure link to see the fields (or columns) within any table.

Table structure



The screenshot shows the phpMyAdmin interface. On the left is a sidebar with a database tree. The 'springfi_wp10' database is expanded, showing several tables including 'wp40_comments'. The main panel displays the 'Table structure' view for the 'wp40_comments' table. At the top, there's a navigation bar with tabs like 'Browse', 'Structure', 'SQL', 'Search', 'Insert', 'Export', 'Import', 'Operations', and 'Triggers'. Below this, there are tabs for 'Table structure' and 'Relation view'. The table structure is presented as a table with 15 rows, each representing a column. Each row includes a checkbox, an index number, the column name, a primary key icon, the data type, collation, attributes, nullability, default value, comments, extra options, and an 'Action' column with 'Change', 'Drop', and 'More' icons. At the bottom, there are checkboxes for 'Check all' and a 'With selected:' section containing icons for 'Browse', 'Change', 'Drop', 'Primary', 'Unique', 'Index', and 'Fulltext'.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	comment_ID	bigint	UNSIGNED	No	None		AUTO_INCREMENT	
<input type="checkbox"/>	2	comment_post_ID	bigint	UNSIGNED	No	0			
<input type="checkbox"/>	3	comment_author	tinytext		No				
<input type="checkbox"/>	4	comment_author_email	varchar(100)	utf8mb4_unicode_520_ci	No				
<input type="checkbox"/>	5	comment_author_url	varchar(200)	utf8mb4_unicode_520_ci	No				
<input type="checkbox"/>	6	comment_author_IP	varchar(100)	utf8mb4_unicode_520_ci	No				
<input type="checkbox"/>	7	comment_date	datetime		No	0000-00-00 00:00:00			
<input type="checkbox"/>	8	comment_date_gmt	datetime		No	0000-00-00 00:00:00			
<input type="checkbox"/>	9	comment_content	text	utf8mb4_unicode_520_ci	No				
<input type="checkbox"/>	10	comment_karma	int		No	0			
<input type="checkbox"/>	11	comment_approved	varchar(20)	utf8mb4_unicode_520_ci	No	1			
<input type="checkbox"/>	12	comment_agent	varchar(255)	utf8mb4_unicode_520_ci	No				
<input type="checkbox"/>	13	comment_type	varchar(20)	utf8mb4_unicode_520_ci	No	comment			
<input type="checkbox"/>	14	comment_parent	bigint	UNSIGNED	No	0			
<input type="checkbox"/>	15	user_id	bigint	UNSIGNED	No	0			

This is the structure of the **wp40_comments** table, it has 15 columns or fields. Each field has a unique name and is described in terms of its (data) *Type*, *Collation* and a few other things too.

Working with databases
(especially when creating
your own) can initially seem
very complicated, not
because it's intrinsically
difficult but because there's
a lot to remember...

Dynamic Website News

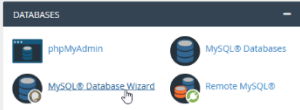
A simple news article application using PHP and MySQL

How should we create databases?

Article #4
21st February 2022

Databases are usually created using the control panel provided by your web host. In most cases, on a LAMP stack server, that will be cPanel.

There are two ways to create a database in cPanel; you can either use the main MySQL Databases option or you can use the MySQL Database Wizard option. The choice is yours; the first option gives you greater control, but the second option is easier.



There are, essentially, three steps to creating a database. First, you give your database a name. Second, you give the database user a name and a strong password. Finally, you add the user to the database and specify what privileges that user has on that database (e.g. select, insert, update, drop). Most often, if we are creating a database for a content management system such as WordPress, we will give the user "All Privileges".

Published: 1:08pm

PHP and MySQL work together perfectly

Article #3
18th February 2022

PHP and MySQL have a very close relationship. They have grown up and matured together. The PHP open source community have developed a range of functions that allow developers to easily access content in MySQL database and use this content to build webpages.

The *mysqli* functions in PHP make it easier, faster and more secure to interface with any MySQL database. The "I" in *mysqli* stands for *Improved*. The improved functions were introduced with PHP5 and should be used with MySQL 4.1.3 or higher.

Published: 6:22pm

What applications use MySQL?

Article #2
15th February 2022

Almost all the web applications that we are familiar with use MySQL databases. Content management platforms like WordPress, Drupal, Perch, and Craft, all use MySQL databases. Community forums like vBulletin, and Invision Community Suite, and wiki collaborative publishing platforms like MediaWiki all use MySQL to store their data.



However, PHP is required to select and extract data from MySQL databases and to construct dynamic web pages from it using PHP/HTML template pages.

For example, in a content management system, all the content of articles (text and links to images), the date, the name of the author etc. are stored in the database and are extracted and compiled "on the fly" using PHP each time the page is requested by a browser.

Published: 4:14pm

A very important announcement

Article #1
12th February 2022

This is the first article in this news feed. Over the next week or so, we'll be bringing you some important news about MySQL and how to work with databases using PHP. We'll be showing you how PHP can be used to pull content out of a database and use it to create dynamic webpages that automatically update when that data changes. Stay tuned!

Published: 4:09pm

Designing database structure

PHP and MySQL work together perfectly

Article #3
18th February 2022

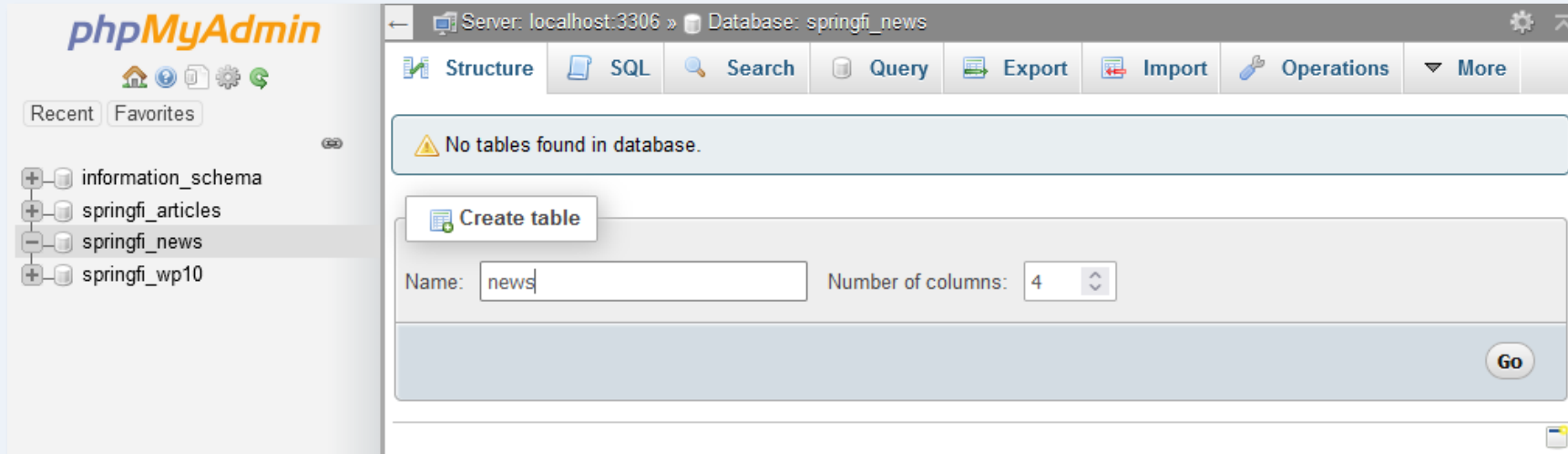
PHP and MySQL have a very close relationship. They have grown up and matured together. The PHP open source community have developed a range of functions that allow developers to easily access content in MySQL database and to use this content to build webpages.

The *mysqli* functions in PHP make it easier, faster and more secure to interface with any MySQL database. The "I" in *mysqli* stands for *Improved*. The improved functions were introduced with PHP5 and should be used with MySQL 4.1.3 or higher.

Published: 6:22pm

The database table structure shown on the following slides is for a simple news application that will show a headline, a date and some content. That's 3 bits of information plus, we'll need a unique index (integer) for each news item. So, four columns in all.

Step 2: Add structure to a new table



Select the name of your new database from the sidebar. In this example, the name is “springfi_news”. The database has no structure, so you are prompted to create a table. To do so, simply enter a name for the table (this one is called **news**) and the number of columns (fields) it should contain (**4** in this case). You can always add more columns later if you need them. Click the “Go” button to create the table.

Table structure

The screenshot shows the phpMyAdmin interface with the 'Table structure' tab selected. The table name is 'news'. The interface includes a sidebar with a database tree showing 'information_schema', 'springfi_articles', 'springfi_news', and 'springfi_wp10'. The main area displays a table structure with columns for Name, Type, Length/Values, Default, Collation, Attributes, Null, Index, A_I, and Comments. There are four empty rows for adding columns, each with a default type of 'INT'. Below the table structure, there are fields for 'Table comments:', 'Collation:', and 'Storage Engine:'. The 'Storage Engine' is set to 'InnoDB'. At the bottom, there is a 'PARTITION definition:' section with a 'Partition by:' dropdown and a text input for '(Expression or column list)', and a 'Partitions:' dropdown. The bottom right corner has 'Preview SQL' and 'Save' buttons.

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	Comments
	INT		None			<input type="checkbox"/>	---	<input type="checkbox"/>	
	INT		None			<input type="checkbox"/>	---	<input type="checkbox"/>	
	INT		None			<input type="checkbox"/>	---	<input type="checkbox"/>	
	INT		None			<input type="checkbox"/>	---	<input type="checkbox"/>	

Table comments: Collation: Storage Engine: InnoDB

PARTITION definition:

Partition by: (Expression or column list)

Partitions:

Preview SQL Save

Table structure is added by giving each column a name and setting a data type. Other settings will be added depending upon data type and how you want the data to be organised (ordered).

All tables must have one primary key, this is one column which is used to order the rows in a table. Typically, this will be a simple integer value (1, 2, 3...) so that rows can be uniquely identified by that number.

Table and column names





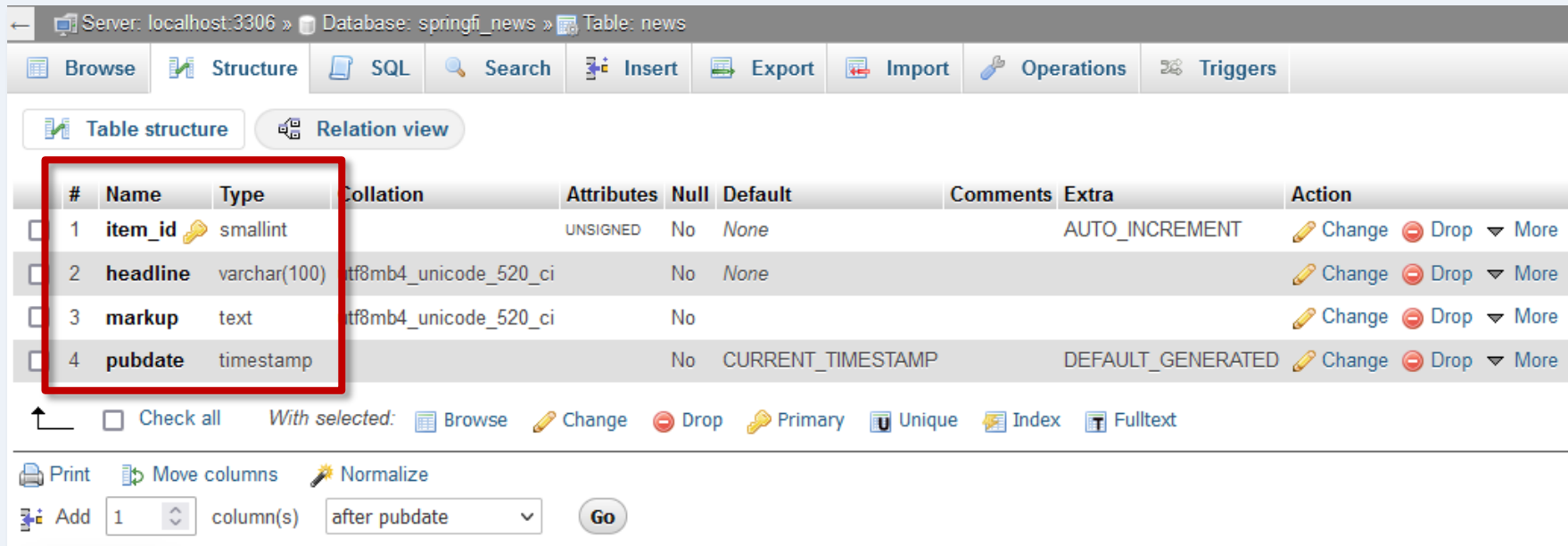
#	Name	Type	Collation	Attrib
<input type="checkbox"/> 1	item_id 	smallint		UNSIGNED
<input type="checkbox"/> 2	headline	varchar(100)	utf8mb4_unicode_520_ci	
<input type="checkbox"/> 3	markup	text	utf8mb4_unicode_520_ci	
<input type="checkbox"/> 4	pubdate	timestamp		
<div> <input type="checkbox"/> Check all With selected:  Browse  Change</div>				

Table and column names can use any combination of letters, numbers and the underscore character. Typically, names use lower-case characters, and the underscore is used as a separator (spaces cannot be used), hence we have simple, identifiable names like **item_id**. When deciding upon names, it is a good idea to avoid using names that could be confused with PHP or MySQL functions. For example, we've called our date column **pubdate** (publication date) because both PHP and MySQL have date functions called "date". Although calling our column **date** wouldn't cause an error, it may lead to confusion.

Data structure with phpMyAdmin



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	item_id	smallint	UNSIGNED	No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	headline	varchar(100)		No	None			Change Drop More
<input type="checkbox"/>	3	markup	text		No				Change Drop More
<input type="checkbox"/>	4	pubdate	timestamp		No	CURRENT_TIMESTAMP		DEFAULT_GENERATED	Change Drop More

With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#)

[Print](#) [Move columns](#) [Normalize](#)

[Add](#) column(s) [Go](#)

This database has just one table with 4 columns (fields), starting with **item_id** (a unique index), then **headline**, **markup** and **pubdate**. Each field is defined as a different data **type** depending on what type of content it will contain. For example, the **pubdate** field is set to display the current MySQL timestamp each time a new row is created. The **item_id** field is an unsigned (positive only) small integer (0-65,535), the **headline** can contain up to 100 variable characters (varchar) and the **markup** is just a text field (up to 65,535 characters).

Data structure with phpMyAdmin

The screenshot shows the phpMyAdmin interface for configuring a table named 'news'. The table has four columns: 'item_id', 'headline', 'markup', and 'pubdate'. The 'item_id' column is configured as a SMALLINT, UNSIGNED, PRIMARY key with the 'A_I' (Auto Increment) option checked. The 'headline' and 'markup' columns are configured as VARCHAR and TEXT respectively, both with a collation of 'utf8mb4_unicode_ci'. The 'pubdate' column is configured as a TIMESTAMP with a default value of 'CURRENT_TIMESTAMP'. The storage engine is set to 'InnoDB'.

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	Comments
item_id	SMALLINT		None		UNSIGNED	<input type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>	
headline	VARCHAR	100	None	utf8mb4_unicode_ci		<input type="checkbox"/>	---	<input type="checkbox"/>	
markup	TEXT		None	utf8mb4_unicode_ci		<input type="checkbox"/>	---	<input type="checkbox"/>	
pubdate	TIMESTAMP		CURRENT_TIMESTAMP			<input type="checkbox"/>	---	<input type="checkbox"/>	

Table comments: Collation: Storage Engine: InnoDB

PARTITION definition:

Preview SQL Save

The settings shown above are appropriate for the 4 columns we are creating. The **item_id** column is our *primary key*, so the Index value is set to "PRIMARY". The attribute for this column is set to "UNSIGNED" because there will be no negative numbers, and the A_I option is checked for "Auto Increment". Note that the **pubdate** column is the only one with a default value (*CURRENT_TIMESTAMP*). All text fields (headline and markup) are given a collation of *utf8mb4_unicode_ci*. When the structure has been configured, click the Save button.

Data structure with phpMyAdmin

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	item_id	smallint	UNSIGNED	No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	headline	varchar(100)		No	None			Change Drop More
<input type="checkbox"/>	3	markup	text		No				Change Drop More
<input type="checkbox"/>	4	pubdate	timestamp		No	CURRENT_TIMESTAMP		DEFAULT_GENERATED	Change Drop More

With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#)

[Print](#) [Move columns](#) [Normalize](#)

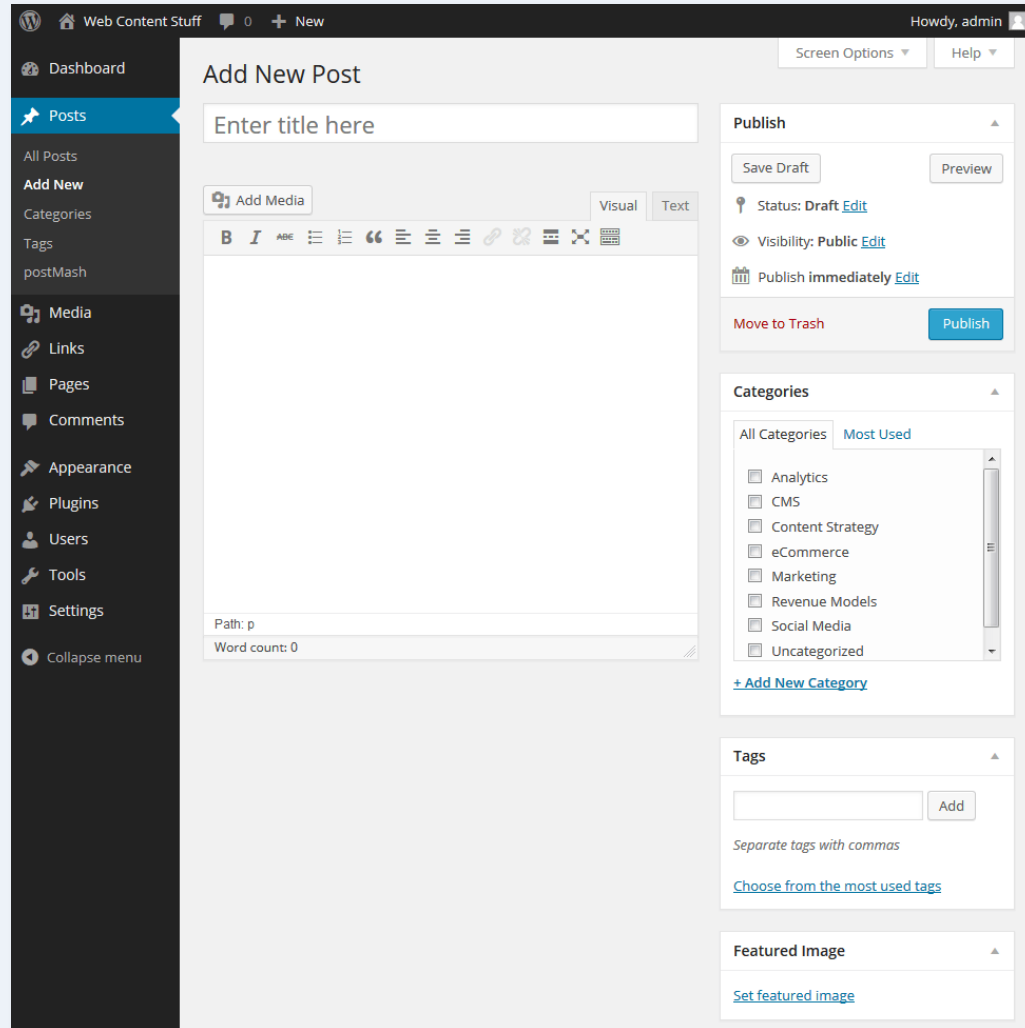
[Add](#) 1 column(s) after pubdate [Go](#)

Above is the Structure view, once the table has been saved. The Collation describes the character set to be used and how text will be sorted. The default value is **latin1_swedish_ci** (for English, Swedish and Finnish) but **utf8mb4_Unicode_520_ci** is the current standard for multi-lingual data (just like setting UTF-8 in our webpages). Notice that **item_id** is set to AUTO_INCREMENT. This means it automatically numbers each row when a new item is added – once set up, we can forget about it. It is also defined as the *primary key*, which just means that the rows in the database will be sorted in that order. We can also forget about **pubdate**, the current date and time will be added automatically when each new item is created.

PHP and MySQL for dynamic content

PART 4: ADDING DATA TO A DATABASE

Web application back-end



Most web applications that use a MySQL database will have a specifically designed back-end that allows for data entry and deals with database transactions. WordPress (shown on the left) is a good example of this. It has a beautifully designed back-end that users can login to and make new posts and other changes.

However, phpMyAdmin can be used as a generic back-end for data entry...

Step 3: Inserting data with phpMyAdmin

Server: localhost:3306 » Database: springfi_news » Table: news

Browse Structure SQL Search Insert Export Import Operations

Column	Type	Function	Null	Value
item_id	smallint unsigned			
headline	varchar(100)			This headline can be up to 100 characters long.
markup	text			<p><p>The markup for this article can be up to 65,535 characters long because we are using the TEXT field type.</p></p> <p><p>Notice that we are adding our HTML markup right in this field. That's OK - it will be interpreted correctly and added to our webpage when we need it.</p></p>
pubdate	timestamp			CURRENT_TIMESTAMP

Go

No value required (auto-increment)

No value required (automatic timestamp)

Clicking the “Insert” tab will display the data entry form defined by the structure we designed. In our example, we need only enter content in the **headline** and the **markup** fields for each new news item. The other data (item_id and pubdate) will be added automatically.

How data is used

The screenshot shows a web page titled "Dynamic Website News" with the subtitle "A simple news article application using PHP and MySQL". The main content area displays a news article with the headline "This headline can be up to 100 characters long.", the sub-header "Article #1", the date "22nd February 2022", the body text "The markup for this article can be up to 65,535 characters long because we are using the TEXT field type." and "Notice that we are adding our HTML markup right in this field. That's OK - it will be interpreted correctly and added to our webpage when we need it.", and the time "Published: 1:15pm". The footer text is "Written for MA Web Design & Content Planning by David Watson".

Annotations with red arrows point to the following elements:

- headline**: Points to the headline text "This headline can be up to 100 characters long."
- date from pubdate**: Points to the date "22nd February 2022"
- markup**: Points to the body text "Notice that we are adding our HTML markup right in this field. That's OK - it will be interpreted correctly and added to our webpage when we need it."
- time from pubdate**: Points to the time "Published: 1:15pm"

Once data has been added to our database, we can use PHP functions to select the data we want and use it to create webpages. The data shown on the previous slide could be formatted and displayed as shown above.

Browse data with phpMyAdmin

Server: localhost:3306 » Database: springfi_news » Table: news

Browse Structure SQL Search Insert Export Import Operations Triggers

✓ Showing rows 0 - 3 (4 total, Query took 0.0009 seconds.)

```
SELECT * FROM `news`
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create P...

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

	item_id	headline	markup	pubdate
<input type="checkbox"/> Edit Copy Delete	1	A very important announcement	<p>This is the first article in this n...	2022-02-12 16:09:18
<input type="checkbox"/> Edit Copy Delete	2	What applications use MySQL?	<p>Almost all the web applications tha...	2022-02-15 16:14:31
<input type="checkbox"/> Edit Copy Delete	3	PHP and MySQL work together perfectly	<p>PHP and MySQL have a very close rel...	2022-02-18 18:22:36
<input type="checkbox"/> Edit Copy Delete	4	How should we create databases?	<p>Databases are usually created using...	2022-02-21 13:08:49

☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Query results operations

Print Copy to clipboard Export Display chart Create view

Each database table is a sort of spreadsheet consisting of rows of data. This table has 4 columns, which define the table structure and there are 4 rows of data, in this case, news articles.

Once data has been entered, it can be extracted by PHP using a *query*...

PHP and MySQL for dynamic content

PART 5: ACCESSING DATA IN A DATABASE

Once we have a database with structured table columns, containing one or more rows of data, we can use it to build a website, starting with a SQL query...

The SQL query

`SELECT headline, markup, pubdate FROM news ORDER BY pubdate DESC`

these three fields this table this value descending

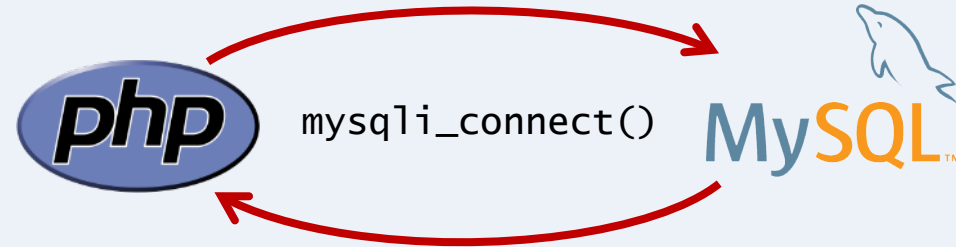
A query is a simple statement that tells MySQL what data we want. This query selects the three fields (columns) `headline`, `markup` and `pubdate` from the `news` table. The words in upper-case are special SQL operators. We could have used `SELECT *` (the wildcard character) to select all columns in the table but more specific queries make for easier interpretation. The query then uses the `pubdate` value to arrange the data in descending order so that the most recent article is always first.

The query, which is just a text string, is assigned to a PHP variable often called `$query` within a PHP script.

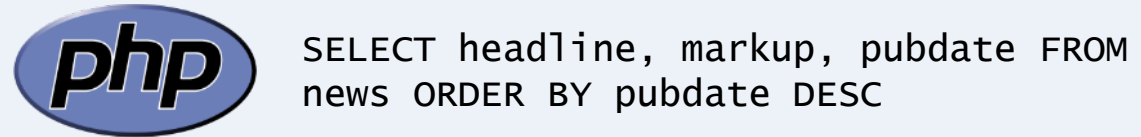
```
$query = "SELECT headline, markup, pubdate FROM news ORDER BY pubdate DESC";
```

How PHP and MySQL work together

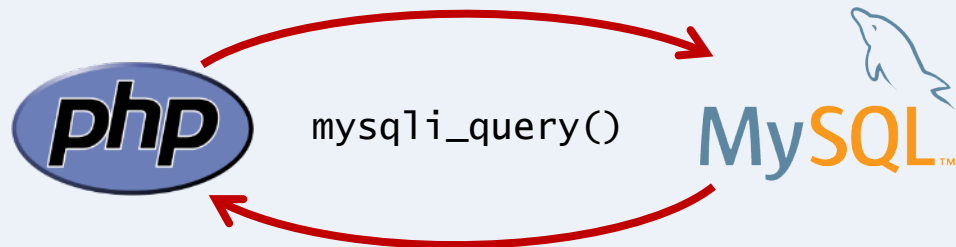
1. Connect to the database. PHP sends access credentials; MySQL returns a connection object.



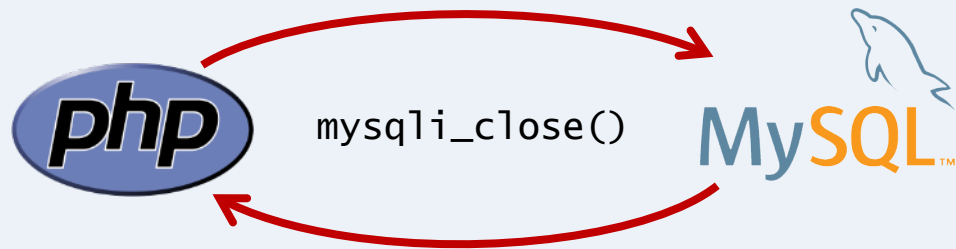
2. Assign query to variable
`$query`



3. Send the query and store the result in another variable
`$result`



4. Close the database connection



Link to database and run query

The credentials you specified when the database was created

```
<?php
```

```
# host, username, password and database name
```

```
$host = "localhost";
```

```
$user = "database_username";
```

```
$password = "+M/*Pqdi1;4Y";
```

```
$name = "database_name";
```

```
# connect to mysql and select database
```

```
$conn = mysqli_connect($host, $username, $password, $name);
```

```
# build a database query
```

```
$query = "SELECT headline, markup, pubdate FROM news ORDER BY pubdate DESC";
```

```
# run the query and assign the result to a variable
```

```
$result= mysqli_query($conn, $query);
```

```
#close the connection
```

```
mysqli_close($conn);
```

```
?>
```

The *host* is always called "localhost" when the database is on the same server as your website (most common scenario) but if not, you need to specify the database server. Ask your web host for details.

This PHP script assigns the database access credentials to variables, connects to the MySQL database, queries the database and stores the resulting data in a variable called **\$result**. It then closes the connection.

What is...

\$result

`$result` is a special kind of array; it contains all the data in all the rows from the database that matched the query we used. Effectively, it's an array of arrays, sometimes referred to as a "two-dimensional array" or "2D array". To access the data, we use a special PHP function that can step through the rows, one at a time.

Fetch and print array data

```
<?php
```

```
# step through each news article, one at a time
```

```
while ($row = mysqli_fetch_array($result))
```

```
{
```

```
    # assign each array element to a variable
```

```
    $headline = $row['headline'];
```

```
    $date = $row['pubdate'];
```

```
    $markup = $row['markup'];
```

```
    # convert mysql date to php timestamp
```

```
    $timestamp = strtotime( $date );
```

```
    # format php timestamp
```

```
    $display_date = date('jS F Y', $timestamp);
```

```
    # print out the news story
```

```
    echo "<h1>$headline</h1>";
```

```
    echo "<p>$display_date</p>";
```

```
    echo "$markup";
```

```
}
```

```
?>
```

All the selected data is stored as a *2D array* in a variable (**\$result**). The data will be extracted one row (array) at a time using PHP.

This PHP script uses a **while** loop to step through each row in the array using the **mysqli_fetch_array** function. It assigns each row in the array to a new variable (**\$row**). This new variable is also an array because it contains each value in the row (**headline**, **pubdate** and **markup**). The next step is to assign each of those values to a simple variable...

Fetch and print array data

```
<?php
```

```
# step through each news article, one at a time
```

```
while ($row = mysqli_fetch_array($result))
```

```
{
```

```
    # assign each array element to a variable
```

```
    $headline = $row['headline'];
```

```
    $date = $row['pubdate'];
```

```
    $markup = $row['markup'];
```

```
    # convert mysql date to php timestamp
```

```
    $timestamp = strtotime( $date );
```

```
    # format php timestamp
```

```
    $display_date = date('jS F Y', $timestamp);
```

```
    # print out the news story
```

```
    echo "<h1>$headline</h1>";
```

```
    echo "<p>$display_date</p>";
```

```
    echo "$markup";
```

```
}
```

```
?>
```

...We can assign a single value from an array to a variable using the following syntax:

```
$new_variable = $array['index'];
```

Where *index* is the name of the field selected from the database.

This script does that for the 3 fields selected. It then converts the MySQL date to a format ready for printing and then prints out the news articles using the **echo** function and includes any required markup.

The *while* loop will continue until all the rows in **\$result** have been processed.

The PHP file

```
<?php
# assign host, username, password and database name to variables
$host = "localhost"; # usually "localhost" but may be different
$user = "springfi_news-user"; # add your database username here
$password = "X##rKmZ8F6S$"; # add your database user password here
$name = "springfi_news"; # add your database name here
# connect to the database or stop the script and give an error message
$conn = mysqli_connect($host, $user, $password, $name) or die ("Cannot connect to database.");
# build the query and assign it to a variable
$query = "SELECT item_id, headline, markup, pubdate FROM news ORDER BY pubdate DESC LIMIT 5";
# run the query and assign the result to a variable or give an error message
$result = mysqli_query($conn, $query) or die ("Error querying database.");
# close the database connection
mysqli_close($conn);
?>

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Dynamic Website News | PHP and MySQL Example</title>
<link rel="stylesheet" href="style/style.css">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<header>
<h1>Dynamic Website News</h1>
<p>A simple news article application using PHP and MySQL</p>
</header>
<main>
<?php
# step through each news article, one at a time
while ($row = mysqli_fetch_array($result)) {
# assign each field to a variable
$number = $row['item_id'];
$headline = $row['headline'];
$markup = $row['markup'];
$date = $row['pubdate'];
# convert mysql date to php timestamp
$timestamp = strtotime($date);
# format php timestamp
$display_date = date('jS F Y', $timestamp);
$display_time = date('g:ia', $timestamp);
# print out the news article
echo " <article> <!-- start of article card -->\n";
echo " <header>\n";
echo " <h2>$headline</h2>\n";
echo " <p><span class=\"highlight\">Article #<span><span>$number</span></span><br>$display_date</p>\n";
echo " </header>\n";
echo "$markup\n";
echo " <footer>Published: $display_time</footer>\n";
echo " </article>\n\n"; }
?>

</main>
<footer>Written for MA Web Design & Content Planning by David Watson</footer>
</body>
</html>
```

The completed code is shown on the left and the resulting page on the right. We've added a few extra details to the script. For example, the **or die** functions tell the script what to do if there is an error when communicating with the database and **\n** in the echo statements force a new line for good HTML formatting. We have also added a limit for the number of news items (**5**) in the query, this ensures that the page doesn't get too long.

Dynamic Website News

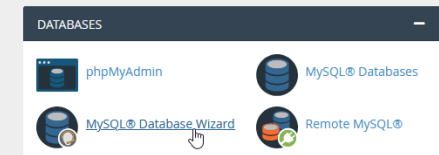
A simple news article application using PHP and MySQL

How should we create databases?

Article #4
21st February 2022

Databases are usually created using the control panel provided by your web host. In most cases, on a LAMP stack server, that will be cPanel.

There are two ways to create a database in cPanel; you can either use the main *MySQL Databases* option or you can use the *MySQL Database Wizard* option. The choice is yours; the first option gives you greater control, but the second option is easier.



There are, essentially, three steps to creating a database. First, you give your database a name. Second, you give the database user a name and a *strong* password. Finally, you add the user to the database and specify what privileges that user has on that database (e.g. select, insert, update, drop). Most often, if we are creating a database for a content management system such as WordPress, we will give the user "All Privileges".

Published: 1:08pm

PHP and MySQL work together perfectly

Article #3
18th February 2022

PHP and MySQL have a very close relationship. They have grown up and matured together. The PHP open source community have developed a range of functions that allow developers to easily access content in MySQL database and to use this content to build webpages.

The *mysqli* functions in PHP make it easier, faster and more secure to interface with any MySQL database. The "I" in *mysqli* stands for *Improved*. The improved functions were introduced with PHP5 and should be used with MySQL 4.1.3 or higher.

Published: 6:22pm

What applications use MySQL?

Article #2
15th February 2022

Formatting HTML with `\n`

print out the news article

```
echo "    <article>
      <header>
        <h2>$headline</h2>
        <p><span class=\"highlight\">Article #<span>$number</span><br>$display_date</p>
      </header>
$markup
      <footer>Published: $display_time</footer>
    </article>\n\n";
```

The new line character `\n` can be added to a string anywhere you need a line break in the generated HTML code. PHP does not add a line break for each echo statement, so you do need them if you want your HTML to read clearly. Similarly, you should consider the indentation of generated markup. The tab character `\t` can be used to add tab spaces.

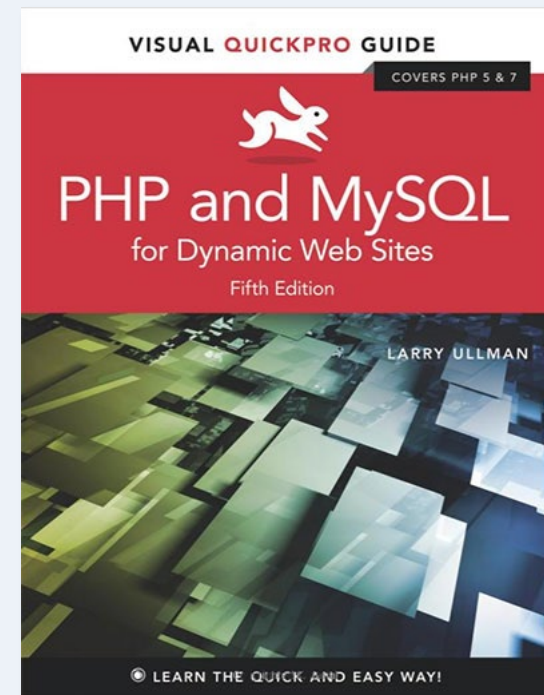
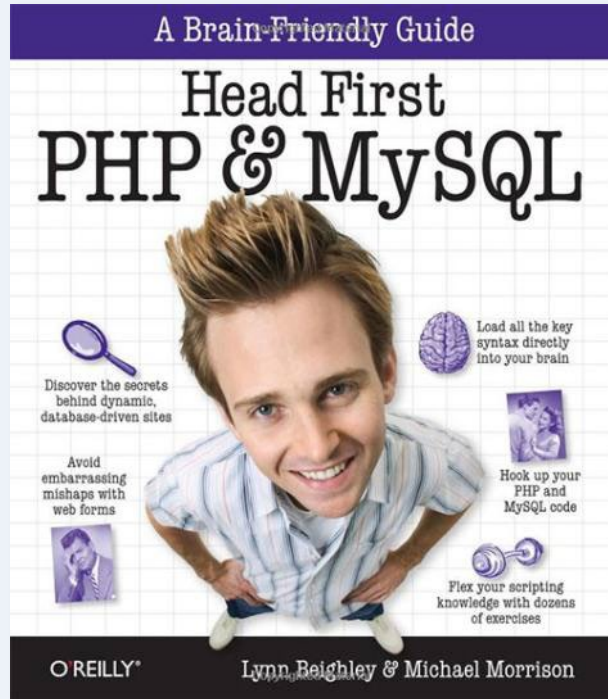
```
<article>
  <header>
    <h1>This is my first news article</h1>
    <p class="date">13th February 2024</p>
  </header>
  <p>This is my first news item. It is very exciting to be working with databases!</p>
  <footer>Published: 3:42pm</footer>
</article>
```


PHP and MySQL for dynamic content

PART 6: LEARNING TO BUILD DYNAMIC PAGES

If you have followed this (initially) complicated sequence, you now know how to build your own simple content management system. As always, the best way to learn is to try it for yourself.

Books



All 3 books provide a good introduction to the subject area. The *Head First* book may be better if you are completely new to programming and *PHP Solutions* may be better if you are a little more confident. Both are available from the library. Larry Ullman's book sits somewhere between the two and it's the book I used when I first learned PHP and MySQL.

The long awaited...

Published 14th February 2022



PDO notation

In this presentation we have used what is called "procedural notation" for connecting to databases. You will likely come across another method, known as PDO, which stands for **PHP Data Objects**. It's an object-oriented method but it is designed to be portable across different types of database (not just MySQL). It has several advanced features and is increasingly popular, but the syntax is less obvious.

```
try {  
    # MySQL with PDO_MYSQL  
    $DBH = new PDO("mysql:host=$host;dbname=$dbname", $user, $pass);  
}  
catch(PDOException $e) {  
    echo $e->getMessage();  
}
```

The code fragment above uses PDO to connect to a MySQL database. PDO is described in Jon Duckett's PHP & MySQL book.

I recommend that you stick with procedural notation while you're learning the basics but consider a move to PDO when you feel comfortable.

See: [Why you Should be using PHP's PDO for Database Access](#)

```
$sql = 'SELECT goodbye FROM lecture';
```