

# Own Your WordPress: How to Optimize



A dark, moody illustration featuring a central figure wrapped in thick, green, vine-like tendrils. The figure's face is visible, showing closed eyes and a small mouth. The background is a dark, textured grey. The overall style is artistic and somewhat somber.

**Amy Kamala**



**What is  
WordPress?**

# WordPress:

Software/App/Content Management System used for creating  
websites and blogs.

Core files/folders ---->

Made with PHP/HTML/CSS/JS

Uses SQL Database

Lives in LAMP/LEMP Environment

Extensible through plugins and themes

Open Source & free

```
.'
..
favicon.gif
favicon.ico
.htaccess
index.php
license.txt
phpinfo.php
readme.html
wp-activate.php
wp-admin
wp-blog-header.php
wp-comments-post.php
wp-config.php
wp-config-sample.php
wp-content
wp-cron.php
wp-includes
wp-links-opml.php
wp-load.php
wp-login.php
wp-mail.php
wp-settings.php
wp-signup.php
wp-trackback.php
xmlrpc.php
```

# <3 Your Database

A database is a system for storing data so applications and websites can retrieve that data efficiently and on demand.

Databases use 'tables', 'rows' and 'columns' as the pillars of its structure, much like a spreadsheet.

Most WordPress site content is stored in a SQL database.

- post content
- page content
- comments
- user data
- permalinks / slugs
- attachments

## Do

- Keep core, plugins, themes up to date.
- Database < 2GB and \_options < 5MB
- Home page < 3 MB
- Size images to-scale, 72 DPI and < 1 MB
- Use a CDN (Content Delivery Network)
- Use cache and minifying
- Use a crawl delay and CAPTCHA
- Use php.ini, akismet and lazy load
- Place Javascript just before `</body>` tag
- Use a DNS firewall service, like CloudFlare.
- Hide wp-admin.
- Use a reliable hosting platform.

## Don't

- Keep plugins, themes or media you are not using.
- Use forms, logins, ads or sessions on home page.
- Use real-time backups or page view counters.

# Types of Hosting

**Shared:** Great to start out on but has limited resources and noisy neighbors.

**Cloud:** Highly scalable, global distribution, redundant.

**Private/Dedicated:** More reliable hardware, less scalable.

# Hosting Features

- **Managed hosting** - Hosting company will maintain server hardware and software for you.
- **Server-side cache** - Hosting company will configure exceptions and debug for you
- **Current Linux and software versions.**
- **Stellar service** is a must.
- Data center should be **close to your audience.**
- SQL location should also be close.



# PHP Environment & php.ini

- Use PHP 7.1 or higher. [PHP 7.3](#) is fastest.
- Use [https/SSL](#) and [HTTP/2](#)
- Use a [php.ini](#) file to maintain control over your hosting environment. Look up your host's documentation on how to implement this. Some hosts don't use a standard php.ini.
- Recommended php.ini settings:

```
GNU nano 2.2.6                               File: php.ini

memory_limit = 200M
max_execution_time = 180
█
```

- These settings are generous! The limit is per php spawn, so you want to keep them low to prevent the potential of rogue spawns. A php spawn is a process created on the server to execute a function. Like when buttons are clicked.

# PHP Environment & wp-config.php

- You can set php environment limits in the wp-config.php file. wp-config.php is used for a secure database connection and a number of primary settings. It can be customized. Yay!

\*make sure you use straight ' and not curly '

```
/**
 * WordPress Database Table prefix.
 *
 * You can have multiple installations in one database if you give each
 * a unique prefix. Only numbers, letters, and underscores please!
 */
$table_prefix = 'wp_gmrig8_';

define( 'WP_MAX_MEMORY_LIMIT', '128M' );
define( 'WP_MEMORY_LIMIT', '128M' );

/* That's all, stop editing! Happy blogging. */

/** Absolute path to the WordPress directory. */
if ( ! defined( 'ABSPATH' ) )
    define( 'ABSPATH', dirname( __FILE__ ) . '/' );

/** Sets up WordPress vars and included files. */
require_once ABSPATH . 'wp-settings.php';
```

Check out: <https://wordpress.org/support/article/editing-wp-config-php/#increasing-memory-allocated-to-php> for more info

# Front End Tools: .htaccess

An .htaccess file interacts with the Apache service to determine how the site is delivered to visitors. In WordPress, the .htaccess file regulates pretty permalinks (links with the post name in them). WordPress Generates the .htaccess file for you when permalinks are set.

```
# BEGIN WordPress

<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [L]
</IfModule>

# END WordPress
```

# Browser-side Cache with .htaccess

Browser-side caching in .htaccess using mod\_expires or mod\_headers.

```
# BEGIN Caching
<ifModule mod_headers.c>
<filesMatch "\\. (ico|pdf|flv|jpg|jpeg|png|gif|swf|ttf|otf|woff|woff2|eot|svg)$">
Header set Cache-Control "max-age=2592000, public"
</filesMatch>
<filesMatch "\\. (css)$">
Header set Cache-Control "max-age=604800, public"
</filesMatch>
<filesMatch "\\. (js)$">
Header set Cache-Control "max-age=216000, private"
</filesMatch>
<filesMatch "\\. (xml|txt)$">
Header set Cache-Control "max-age=216000, public, must-revalidate"
</filesMatch>
<filesMatch "\\. (html|htm|php)$">
Header set Cache-Control "max-age=1, private, must-revalidate"
</filesMatch>
</ifModule>
# END Caching
```

```
# BEGIN Expire headers
<ifModule mod_expires.c>
ExpiresActive On
ExpiresDefault "access plus 5 seconds"
ExpiresByType image/x-icon "access plus 2592000 seconds"
ExpiresByType image/jpeg "access plus 2592000 seconds"
ExpiresByType image/png "access plus 2592000 seconds"
ExpiresByType image/gif "access plus 2592000 seconds"
ExpiresByType image/svg+xml "access plus 2592000 seconds"
ExpiresByType application/x-font-ttf "access plus 2592000 seconds"
ExpiresByType application/x-font-truetype "access plus 2592000 seconds"
ExpiresByType application/x-font-opentype "access plus 2592000 seconds"
ExpiresByType application/x-font-woff "access plus 2592000 seconds"
ExpiresByType application/font-woff2 "access plus 2592000 seconds"
ExpiresByType application/vnd.ms-fontobject "access plus 2592000 seconds"
ExpiresByType application/font-sfnt "access plus 2592000 seconds"
ExpiresByType application/x-shockwave-flash "access plus 2592000 seconds"
ExpiresByType text/css "access plus 604800 seconds"
ExpiresByType text/javascript "access plus 216000 seconds"
ExpiresByType application/javascript "access plus 216000 seconds"
ExpiresByType application/x-javascript "access plus 216000 seconds"
ExpiresByType text/html "access plus 600 seconds"
ExpiresByType application/xhtml+xml "access plus 600 seconds"
</ifModule>
# END Expire headers
```

Check this link for more info! Thanks, DreamHost!

<https://help.dreamhost.com/hc/en-us/articles/216363157-How-can-I-cache-my-site-with-an-htaccess-file->

# Cache

Method of storing data temporarily so it can be retrieved quickly.

\*User Sessions (logins, forms, my-account pages) interfere with cache.

\*Cache exceptions can be configured. How to do so varies among different types of cache.

\*Debugging techniques also vary among cache types.

\*When debugging, cache hits and misses help identify problem areas.

- Browser-side: htaccess
- Front End: Plugins
- Server-side: Ngxcache, Varnish
- Persistent Object: Memcached, Redis
- 3rd party like CloudFlare, Sucuri; cache layer outside of the hosting system.

# Minifying

Minification is the removal of white spaces, comments and unused code, shortening variables, combining scripts and other optimizations. You can use a plugin for minification.

- [Fast Velocity Minify](#)
- [WP Smush](#)
- [Autoptimize](#)

# Front End Tools:

# JetPack

Has Command Line!

!!!!!!!

- Image CDN Module
- Lazy-loading Images Module
- Image Performance Module
- Site Accelerator
- Option to speed up static file load times

This can be done in wp-admin under  
JetPack -> Settings -> Performance

..or using JetPack Command Line

```
wp jetpack
```

Usage:

- wp jetpack status [<full>]
- wp jetpack module <list|activate|deactivate|toggle> [<module\_name>]
- wp jetpack options <list|get|delete|update> [<option\_name>] [<option\_value>]
- wp jetpack protect <whitelist> [<ip|ip\_low-ip\_high|list|clear>]
- wp jetpack reset <modules|options>
- wp jetpack disconnect <blog|user> [<user\_identifier>]

Check it out: <https://jetpack.com/support/jetpack-cli/>

# File and Data Management

You can connect to your server via SSH using a terminal client, to manage your files with command line.

You can edit files in your browser with <https://yourdomain.com/wp-admin>

You can upload, download, edit files with an S/FTP client like FileZilla, Cyberduck or similar

You can manage your database content using a client such as PHPmyadmin or MySQL WorkBench

cPanel does all of the above (except SSH)



# Intro to SSH / Command Line Basics

Login with the command `ssh username@yourserver.com` (contact your host for credentials and instructions)

Make new file: `pico <filename>` Exit pico: `ctrl + o` to save and `ctrl + x` to exit

Edit a file: `nano <filename>` Exit nano: `ctrl + o` to save and `ctrl + x` to exit

Edit a file: `vim <filename>` Exit vim: press `esc + shift + ;` then `rq + enter` (to save)

Move a file: `mv <filename> /new/location/`

Copy a file: `cp -r <filename> <newfilename>` Delete a file: `rm -R <filename>`

Navigate between folders: `cd /path/to/folder` Navigate one folder backwards: `cd ..`

Check CPU usage: `top -c` Check memory usage: `free -m`

# SSH / Command Line Basics

```
[kittens-book:~ kitten$ ssh kamala@[REDACTED].dreamhostps.com
[kamala@[REDACTED].dreamhostps.com's password:
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 3.14.52-vs2.3.6.15-1 x86_64)

 * Documentation:  https://help.ubuntu.com/
Last login: Wed Sep  4 12:28:59 2019 from cpe-[REDACTED]com
[ps:[REDACTED]]$ pico newfile.txt
[ps:[REDACTED]]$ nano newfile.txt
[ps:[REDACTED]]$ vim newfile.txt
[ps:[REDACTED]]$ cd aopentertainment.com
[ps:[REDACTED]]$ cd ..
[ps:[REDACTED]]$ [REDACTED]
```

# Working with WordPress Command Line

Make sure WordPress is up to date! Check WP Core version with 'wp core version'

```
[[dp- ]$ wp core version  
5.2.3
```

Check Installed Themes and Their Status with 'wp theme list' and 'wp theme status'

```
[[dp- ]$ wp theme list
```

name	status	update	version
button	inactive	none	1.0.4
illustratr	inactive	none	1.3.3
nataraja-child	inactive	none	1.0.0
nataraja	active	none	1.0.1
scrawl	inactive	none	1.0.13
sela	inactive	none	1.0.17
sketch	inactive	none	1.2.4
twentyineteen	inactive	none	1.4
twentyseventeen	inactive	none	2.2
vw-bakery	inactive	available	0.4

Check installed plugins and their status with 'wp plugin list' and 'wp plugin status'

```
[[dp- ]$ wp plugin list
```

name	status	update	version
akismet	active	none	4.1.2
contact-form-7	active	available	5.1.3
dreamhost-panel-login	active	none	1.0.0
dreamspeed-cdn	active	none	0.7.4
everest-gallery-lite	active	none	1.0.3
hello-dolly	inactive	none	1.7.2
jetpack	active	available	7.4.1
KittenIA/KittenIA	active	none	1.0.0
varnish-http-purge	active	none	4.8.1
responsive-lightbox	active	none	2.1.0
sucuri-cloudproxy-waf	inactive	none	1.4
whatever	inactive	none	1
kitten-image-audit	inactive	none	1
KittenIA/kitten-image-audit	inactive	none	1
vaultpress	inactive	available	1.9.10
woocommerce	active	available	3.6.4
wp-dbmanager	inactive	none	2.79.2
wpforms-lite	inactive	available	1.5.3.1

\*Find more commands on: <https://developer.wordpress.org/cli/commands/>

# WP Cli Updates, Re-Installs, Un-Installs

## Update Core, Force Core Download, Flush Cache

```
[dp-██████]$ wp core update  
Success: WordPress is up to date.  
[dp-██████]$ wp core download --force  
Downloading WordPress 5.2.3 (en_US)...  
md5 hash verified: ██████████  
Success: WordPress downloaded.  
[dp-██████]$ wp cache flush  
Success: The cache was flushed.
```

## Update Plugins and Themes

```
[dp-3]$ wp plugin update --all
Enabling Maintenance mode...
Fetching pre-update site response...
-> HTTP status code: 503
-> Correctly detected closing </body> tag.
-> No uncaught fatal error detected.
Downloading update from https://downloads.wordpress.org/plugin/contact-form-7.5.1.4.zip...
Unpacking the update...
Installing the latest version...
Removing the old version of the plugin...
Plugin updated successfully.
```

# Install, Un-Install, Deactivate, Activate

```
[dp-3] wp plugin install wordfence
Installing Wordfence Security - Firewall & Malware Scan (7.4.0)
Fetching pre-update site response...
-> HTTP status code: 200
-> Correctly detected closing </body> tag.
-> No uncaught fatal error detected.
Downloading installation package from https://downloads.wordpress.org/plugin/wordfence.7.4.0.zip...
Unpacking the package...
Installing the plugin...
Plugin installed successfully.
Fetching post-update site response...
-> HTTP status code: 200
-> Correctly detected closing </body> tag.
-> No uncaught fatal error detected.
Success: Installed 1 of 1 plugins.
[dp-3] wp plugin uninstall wordfence
Uninstalled and deleted 'wordfence' plugin.
Success: Uninstalled 1 of 1 plugins.
[dp-3] wp cache flush
Success: The cache was flushed.
[dp-3] wp plugin deactivate KittenIA
Plugin 'KittenIA' deactivated.
Success: Deactivated 1 of 1 plugins.
[dp-3] wp plugin activate KittenIA
Plugin 'KittenIA' activated.
Success: Activated 1 of 1 plugins.
```

# Checksums & Scans

Verify Core health with 'wp core verify-checksums'

```
[[dp-██████████]$ wp core verify-checksums  
Success: WordPress installation verifies against checksums.
```

Verify plugin health with 'wp plugin verify-checksums --all'

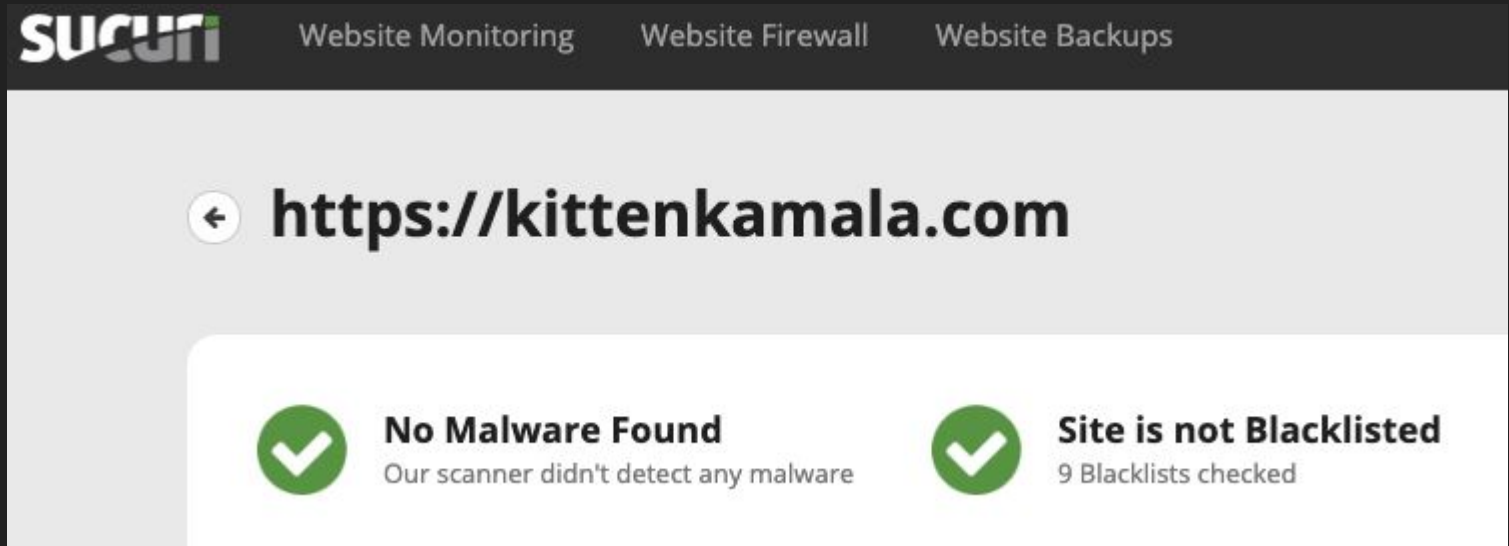
```
[[dp-██████████] wp plugin verify-checksums --all  
Warning: Could not retrieve the checksums for version 1.0.0 of plugin dreamhost-panel-login, skipping.  
Warning: Could not retrieve the checksums for version 1.0.0 of plugin KittenIA, skipping.  
Warning: Could not retrieve the checksums for version 1 of plugin whatever, skipping.  
Warning: Could not retrieve the checksums for version 1 of plugin kitten-image-audit, skipping.  
Warning: Could not retrieve the checksums for version 1.0.0 of plugin KittenIA, skipping.
```

plugin_name	file	message
responsive-lightbox	assets/select2/js/select2.js	File is missing
responsive-lightbox	assets/select2/js/select2.min.js	File is missing

Responsive Lightbox needs to be reinstalled!

# Checksums & Scans

Sucuri offers a free malware scanner at: <https://sitecheck.sucuri.net>



# Fix a Hacked WordPress

- 1) Reset all passwords (``wp user reset-password admin editor``)
- 2) Export Database (``wp db export``)
- 3) Rename public\_html directory and create new, empty one using a `mv` command.
- 4) Install a fresh WordPress core (``wp core download``) in new public\_html
- 5) Update wp-config with new salts and new db credentials
- 6) Re-install Plugins and Theme (``wp plugin install <plugin>`` ``wp theme install <theme>``)
- 7) Import database (``wp db import <filename>``)
- 8) Reset permalinks or make new .htaccess  
(`wp rewrite structure '%year%/%monthnum%/%postname%/'`)
- 9) Copy over /wp-content/uploads if not containing hacked files using a `cp -r` command
- 10) If the database is hacked, export the hacked table, use `SED` to clean it, drop the live table and re-import the clean table.

# Debugging



Pinpointing the Source of Error



# Debugging Tools

Browser Developer Tools

php.ini

wp-config.php

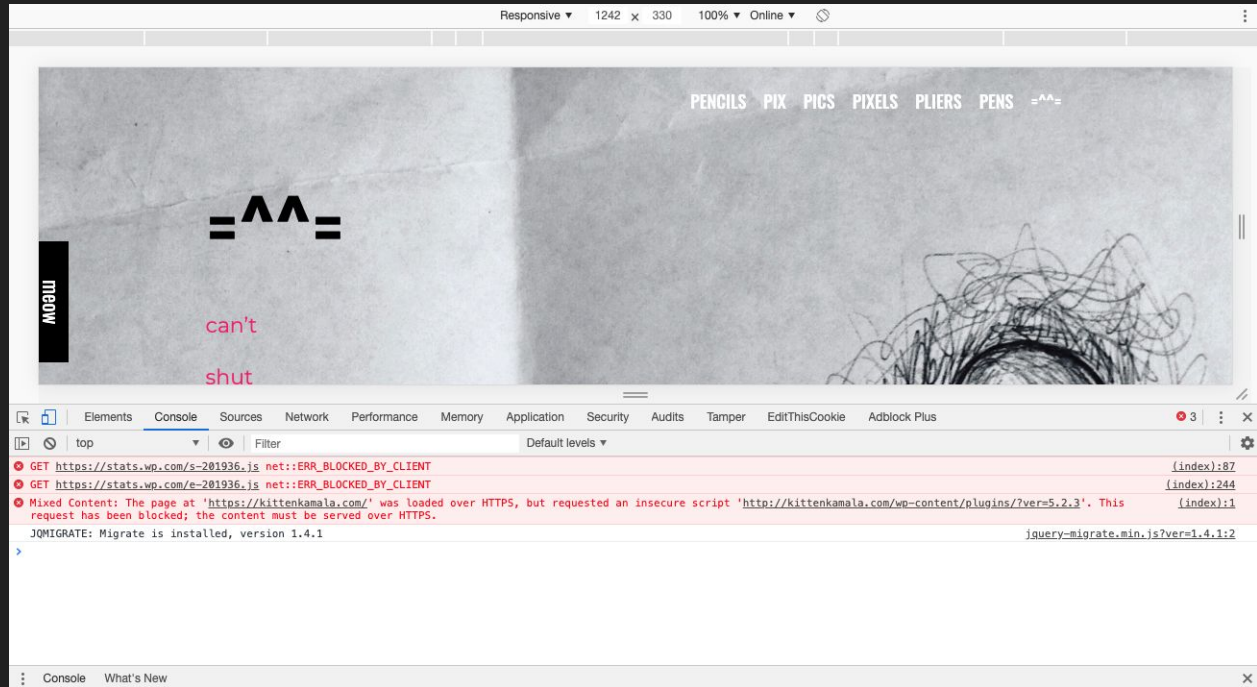
Site Health Module

WordPress Command Line

Error logs

Software such as Xdebug, New Relic, Zabbix,  
Nagios, Grafana...

# Debug With Browser Developer Tools



Found under 'View -> Developer -> Developer Tools' in Chrome  
and 'Tools -> Web Developer' in FireFox

# Error reporting with php.ini and wp-config.php

Add code to your php.ini file to enable error reporting. Syntax will vary from host to host.

```
display_errors = 1
error_reporting = 1
```

Displays errors on website or use an error log...

```
log_errors = on
error_reporting = 32767
error_log = /path/to/file
```

Add code to your wp-config.php file to enable error reporting.

```
// Enable WP_DEBUG mode
define( 'WP_DEBUG', true );

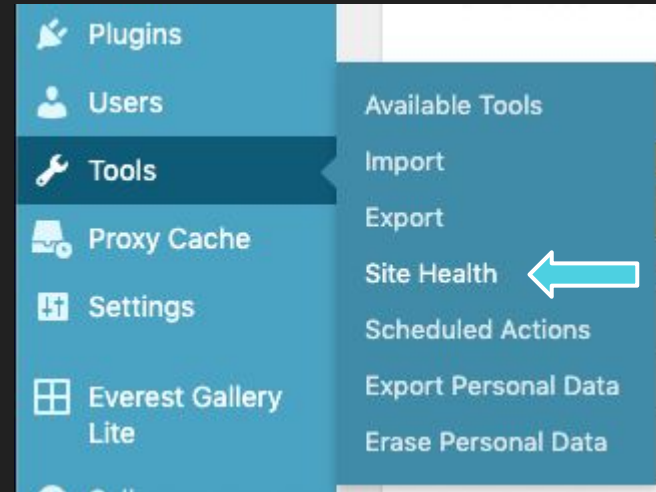
// Enable Debug logging to the /wp-content/debug.log file
define( 'WP_DEBUG_LOG', true );

// Disable display of errors and warnings
define( 'WP_DEBUG_DISPLAY', false );
@ini_set( 'display_errors', 0 );
```

\*This creates an error.log in your wp-content folder.

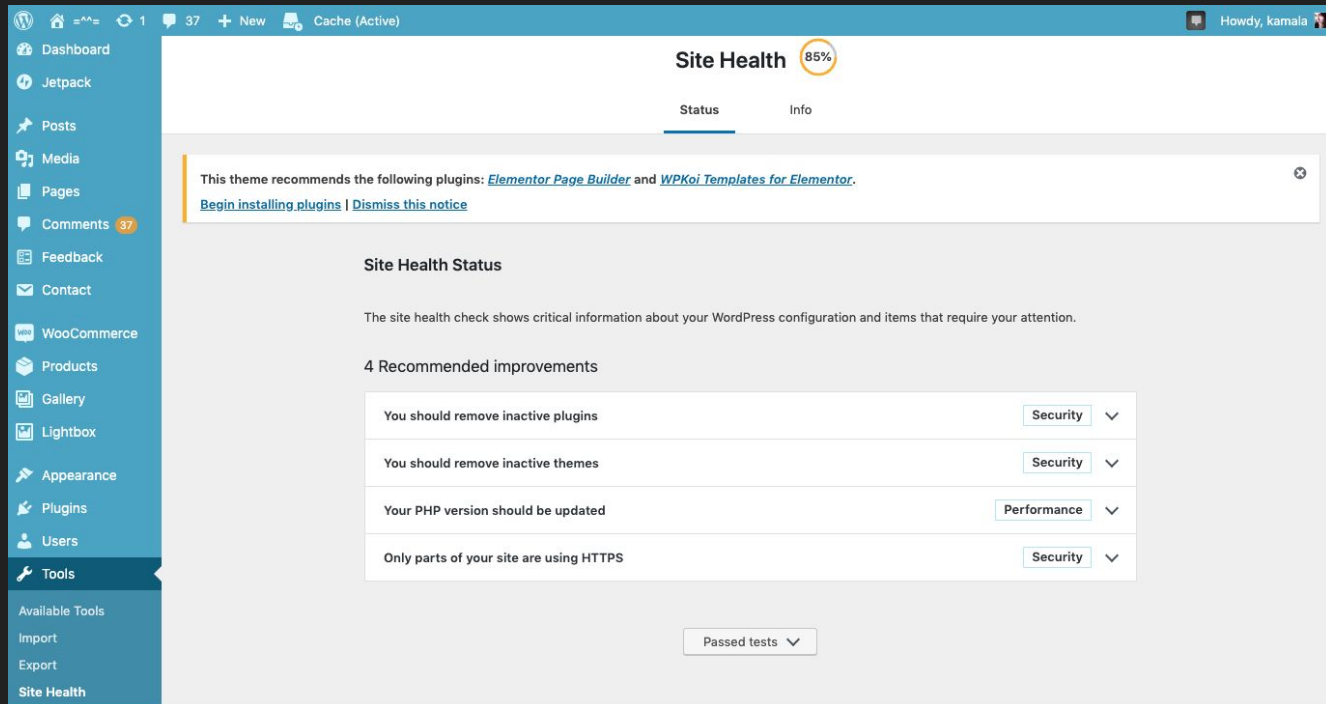
# Site Health Check Module in Core

Since WordPress 5.2 the Site Health Module has been integrated into WordPress Core and can be found under Tools -> Site Health in the wp-admin dashboard.



# Site Health Check Dashboard

The Site Health Module provides detailed information as well as fatal error recovery. By default Site Health will disable the source of a fatal error and email the admin.



The screenshot displays the WordPress Site Health dashboard. The left sidebar contains navigation links for Dashboard, Jetpack, Posts, Media, Pages, Comments (37), Feedback, Contact, WooCommerce, Products, Gallery, Lightbox, Appearance, Plugins, Users, Tools, Available Tools, Import, Export, and Site Health. The main content area shows the Site Health status with a score of 85%. A notice recommends installing plugins: [Elementor Page Builder](#) and [WPKey Templates for Elementor](#). Below this, the Site Health Status section explains that the check shows critical information about WordPress configuration. It lists 4 Recommended improvements:

You should remove inactive plugins	Security	▼
You should remove inactive themes	Security	▼
Your PHP version should be updated	Performance	▼
Only parts of your site are using HTTPS	Security	▼

At the bottom, a button indicates "Passed tests ▼".

**WHEN YOUR WORDPRESS  
SITE RECOVERS FROM A  
FATAL ERROR AUTOMATICALLY**



# Find Bottlenecks with WP Profile

Name	Description
<a href="#"><u>wp_profile eval</u></a>	Profile arbitrary code execution.
<a href="#"><u>wp_profile eval-file</u></a>	Profile execution of an arbitrary file.
<a href="#"><u>wp_profile hook</u></a>	Profile key metrics for WordPress hooks (actions and filters).
<a href="#"><u>wp_profile stage</u></a>	Profile each stage of the WordPress load process (bootstrap, main_query, template).

\*Thank you to [developer.wordpress.org](https://developer.wordpress.org) for providing an incredible resource!

# WP Profile Stage

```
[dp- ]$ wp profile stage
```

stage	time	query_time	query_count	cache_ratio	cache_hits	cache_misses	hook_time	hook_count	request_time	request_count
bootstrap	1.2321s	0.0166s	50	97.34%	2887	79	0.7865s	31098	0.2671s	1
main_query	0.0035s	0.0009s	2	98%	49	1	0.0012s	129	0s	0
template	0.0945s	0.009s	22	97.74%	1516	35	0.092s	3764	0s	0
total (3)	1.3301s	0.0264s	74	97.69%	4452	115	0.8798s	34991	0.2671s	1

Breaks the site load process in three stages:

- Bootstrap: WordPress getting set up.
- Main\_query: WordPress processing requests.
- Template: WordPress rendering theme.

'wp profile stage' can be used to measure load times for each stage as well as cache misses.



# WP Profile Stage <stage>

Breaks the specified stage down by each hook.

\$ wp profile stage bootstrap									
hook	callback_count	time	query_time	query_count	cache_ratio	cache_hits	cache_misses	request_time	request_count
muplugins_loaded:before		0.1676s	0.0043s	1	25%	1	3	0s	0
muplugins_loaded	2	0.0002s	0s	0	50%	1	1	0s	0
plugins_loaded:before		0.1965s	0.0052s	19	83.12%	197	40	0s	0
plugins_loaded	20	0.1167s	0s	0	100%	90	0	0s	0
setup_theme:before		0.0003s	0s	0	100%	4	0	0s	0
setup_theme	1	0s	0s	0		0	0	0s	0
after_setup_theme:before		0.0175s	0s	0	100%	42	0	0s	0
after_setup_theme	9	0.0087s	0s	0	100%	23	0	0s	0
init:before		0s	0s	0		0	0	0s	0
init	104	0.243s	0.0056s	16	97.38%	780	21	0.1386s	1
wp_loaded:before		0s	0s	0		0	0	0s	0
wp_loaded	13	0.0016s	0s	0		0	0	0s	0
wp_loaded:after		0.4243s	0.0038s	14	99.21%	1749	14	0s	0
total (13)	149	1.1765s	0.019s	50	83.86%	2887	79	0.1386s	1

\*More info found here: <https://developer.wordpress.org/cli/commands/profile/stage/>

# Understanding Errors

## Error Types

- Fatal Error: Website is broken, scripts have stopped running. Code can be read but not executed.
- Parse Error: Code syntax is incorrect. Script execution stops.
- Warnings: Just a heads up. Scripts keep running.
- Notice: Same as Warnings but usually refers to an undefined function or variable.

# Common Errors

## How to troubleshoot WordPress errors:

### Step 1 - Deactivate Plugins.

Step 2 - You're welcome.

- **Allowed memory size of bytes exhausted:** Memory is maxed out. Increase memory or kill processes eating up memory (or both).
- **HTTP Error:** Usually related to php environment. Increase php environment settings; max spawns, max memory, execution time..make sure gd or imagick extensions are working, make sure disk and /tmp are not full.
- **Error Connecting to Database:** Check wp-config for syntax errors (like curly single quotes) and incorrect credentials. Check SQL service and SQL server uptime. Check for database corruption. Repair if needed. Check for database DDOS with a SQL Syntax `'show full processlist'`
- **404 Page Not Found:** Check .htaccess for mistakes. Reset permalinks. Make sure correct theme is in use. Check permissions. Make sure files are in the correct place and DNS is correct.

# Common Errors

- **Stuck in Maintenance Mode:** Delete or rename .maintenance file in public\_html directory.
- **WSOD:** Check for fatal PHP errors. Flush cache. Usually caused by malfunctioning theme or plugin. Delete default.html if present.
- **Internal Server Error 500:** Often from a malfunctioning plugin or theme. Could be from high resource usage, incorrect permissions, full disk, fatal PHP error, Apache/Nginx error or DDOS.
- **Connection Timed Out:** Increase execution time. However you will want to troubleshoot to find the source of the long running processes.
- **Deprecated Code:** Some code in the install is outdated and will expire soon. Update install and code.


# Taking Care Of Your SQL Database

- In a database, data is stored in tables, rows and columns, like a spreadsheet.
- A WordPress Database should ideally be less than 2 GB total,
- `_options` table should be less than 5 MB. The `_options` table is used by every part of the site. Keep it minimal.
- Keep autoloaded data under 1 MB
- `_posts` tables can become large (2GB+) on news sites and blogs. This should be fine if you keep it well optimized and indexed, and make sure you don't use queries that select the entire table..

# Database and WP Cli (WordPress Command Line)

WordPress Command Line can do everything wp-admin does++ !

Run a “wp db size” command to check database size!.



•Always take a backup before working on a database! You can export a backup with WP Cli ‘wp db export’ command.

```
[dp- ]$ wp db size --size_format=mb
7
[dp- ]$ wp db size --tables --size_format=mb
```

Name	Size
wp_gmrig8_commentmeta	1 MB
wp_gmrig8_comments	1 MB
wp_gmrig8_links	1 MB
wp_gmrig8_options	2 MB
wp_gmrig8_postmeta	2 MB
wp_gmrig8_posts	1 MB
wp_gmrig8_term_relationships	1 MB
wp_gmrig8_term_taxonomy	1 MB
wp_gmrig8_termmeta	1 MB
wp_gmrig8_terms	1 MB
wp_gmrig8_usermeta	1 MB
wp_gmrig8_users	1 MB
wp_gmrig8_wc_product_meta_lookup	1 MB
wp_gmrig8_wc_tax_rate_classes	1 MB
wp_gmrig8_woocommerce_order_itemmeta	1 MB
wp_gmrig8_woocommerce_payment_tokenmeta	1 MB

# Database Health Check and Repair with WP Cli

Check for corruption with 'wp db check'



```
[[ps~]$ wp db check
PHP Warning: Module 'fileinfo' already loaded in Unknown
PHP Warning: Module 'fileinfo' already loaded in Unknown
wp_test_kittenkamala_com.wp_4yti27_commentmeta OK
wp_test_kittenkamala_com.wp_4yti27_comments OK
wp_test_kittenkamala_com.wp_4yti27_links OK
wp_test_kittenkamala_com.wp_4yti27_options OK
wp_test_kittenkamala_com.wp_4yti27_postmeta OK
wp_test_kittenkamala_com.wp_4yti27_posts OK
wp_test_kittenkamala_com.wp_4yti27_term_relationships OK
wp_test_kittenkamala_com.wp_4yti27_term_taxonomy OK
wp_test_kittenkamala_com.wp_4yti27_termmeta OK
wp_test_kittenkamala_com.wp_4yti27_terms OK
wp_test_kittenkamala_com.wp_4yti27_usermeta OK
wp_test_kittenkamala_com.wp_4yti27_users OK
Success: Database checked.
```

\*just a  
warning..  
can  
ignore.

Repair/recreate DB with 'wp db repair'



```
[[ps~]$ wp db repair
PHP Warning: Module 'fileinfo' already loaded in Unknown on line 0
PHP Warning: Module 'fileinfo' already loaded in Unknown on line 0
aopentertainment_com_1.wp_ukv36e_commentmeta OK
aopentertainment_com_1.wp_ukv36e_comments OK
aopentertainment_com_1.wp_ukv36e_huge_itportfolio_images OK
aopentertainment_com_1.wp_ukv36e_huge_itportfolio_portfolios OK
aopentertainment_com_1.wp_ukv36e_links OK
aopentertainment_com_1.wp_ukv36e_options OK
aopentertainment_com_1.wp_ukv36e_postmeta OK
aopentertainment_com_1.wp_ukv36e_posts OK
aopentertainment_com_1.wp_ukv36e_term_relationships OK
aopentertainment_com_1.wp_ukv36e_term_taxonomy OK
aopentertainment_com_1.wp_ukv36e_termmeta OK
note : The storage engine for the table doesn't support repair
aopentertainment_com_1.wp_ukv36e_terms OK
aopentertainment_com_1.wp_ukv36e_usermeta OK
aopentertainment_com_1.wp_ukv36e_users OK
Success: Database repaired.
```

\*More commands found here: <https://developer.wordpress.org/cli/commands/db/>

# Connect directly to Your Database using WP CLI

Connect to your MySQL service directing with a **'wp db cli'** command.



```
[dp- ]$ wp db cli
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 908980
Server version: 5.6.39-log Source distribution

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```



# Talk To Your Database with SQL Syntax

\*SQL Syntax is a type of command line and code that can be used to communicate with a database. To view details of what SQL queries are currently running, use the 'show full processlist;' command. A few more ...

Get table details with 'desc <tablename>;'

```
[mysql> desc wp_4yti27_options;
```

Field	Type	Null	Key	Default	Extra
option_id	bigint(20) unsigned	NO	PRI	NULL	auto_increment
option_name	varchar(191)	NO	UNI		
option_value	longtext	NO		NULL	
autoload	varchar(20)	NO		yes	

```
4 rows in set (0.00 sec)
```

view table names  
'with show tables;':

```
[mysql> show tables;
```

Tables_in_wptest_kittenkamala_com
wp_4yti27_commentmeta
wp_4yti27_comments
wp_4yti27_links
wp_4yti27_options
wp_4yti27_postmeta
wp_4yti27_posts
wp_4yti27_term_relationships
wp_4yti27_term_taxonomy
wp_4yti27_termmeta
wp_4yti27_terms
wp_4yti27_usermeta
wp_4yti27_users

```
12 rows in set (0.00 sec)
```

\*WP Core uses 12 tables

# Talk To Your Database with SQL Syntax

\*This is what a Brute Force attempt on the database looks like...

```
mysql> show processlist;
```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+								
Id	User	Host	db	Command	Time	State	Info	
+-----+-----+-----+-----+-----+-----+-----+-----+-----+								
111	root	localhost	NULL	Query	0	init	show processlist	
241	unauthenticated user				NULL	Connect	NULL   login	NULL
245	unauthenticated user				NULL	Connect	NULL   login	NULL
248	unauthenticated user				NULL	Connect	NULL   login	NULL
251	unauthenticated user				NULL	Connect	NULL   login	NULL

# Database Optimization: Use InnoDB

- InnoDB: can lock on rows, while MyISAM only locks on tables.
- InnoDB: is more recoverable with better logging.
- InnoDB: supports transactions, which means you can test or undo a change.
- Use SQL Syntax to check what Engine is in use:

```
mysql> SHOW TABLE STATUS WHERE Engine = 'MyISAM';  
Empty set (0.00 sec)
```

```
mysql> SHOW TABLE STATUS WHERE Engine = 'InnoDB';
```

- Use SQL Syntax to convert to InnoDB:  
**'ALTER TABLE *table\_name* ENGINE=InnoDB;'**

```
[mysql> ALTER TABLE dev_aopentertainment_com.wp_ukv36e_comments ENGINE=InnoDB;  
Query OK, 22065 rows affected (1.25 sec)  
Records: 22065 Duplicates: 0 Warnings: 0
```

# Database Optimization: Indexing with SQL Syntax

An Index is a type of data structure within a database that makes information more accessible and streamlines performance, requiring less from the server per query. You can check if a table is already indexed using SQL Syntax, with a `'SHOW INDEX FROM tablename;'` command.

```
mysql> SHOW INDEX FROM wp_4yti27_options;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
wp_4yti27_options	0	PRIMARY	1	option_id	A	119	NULL	NULL		BTREE		
wp_4yti27_options	0	option_name	1	option_name	A	119	NULL	NULL		BTREE		

```
2 rows in set (0.00 sec)
```

# Database Optimization: Indexing with SQL Syntax

```
[mysql> SHOW INDEX FROM wp_4yti27_options;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
wp_4yti27_options	0	PRIMARY	1	option_id	A	119	NULL	NULL		BTREE		
wp_4yti27_options	0	option_name	1	option_name	A	119	NULL	NULL		BTREE		

2 rows in set (0.00 sec)

This table doesn't need indexing, but if it did you would use the following commands to create or drop (delete) and index:

```
`CREATE INDEX indexname ON tablename(column1, column2);`
```

```
`DROP INDEX indexname ON tablename(column1, column2);`
```

```
[mysql> CREATE INDEX autoloadindex ON wp_4yti27_options(autoload, option_name);  
Query OK, 0 rows affected (0.02 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
[mysql> DROP INDEX autoloadindex ON wp_4yti27_options;  
Query OK, 0 rows affected (0.00 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

# Database Optimization: Clean Up \_options table

Delete Expired Transients from the options table with the WP Cli command '**wp transient delete**'. You will have to leave the SQL environment with an '**exit**' command. Transients are used by themes and plugins to store data temporarily, sort of like cache.

Looks like this:

```
[dp-██████████]$ wp transient delete --expired  
Success: 14 expired transients deleted from the database.
```

Back to SQL with '**wp db cli**'. Find and delete old sessions rows using **SELECT** and **DELETE** (SQL Syntax). Session rows are leftover when crons become out of sync.

```
[mysql> SELECT * FROM kittenkamala_com_4.wp_64qhsd_options WHERE option_name LIKE '_wp_session_%';  
Empty set (0.00 sec)
```

```
[mysql> DELETE FROM kittenkamala_com_4.wp_64qhsd_options WHERE option_name LIKE '_wp_session_%' limit 10;  
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> █ *no sessions rows in this db!
```

More info on wp transient: <https://developer.wordpress.org/cli/commands/transient>

Cron jobs are automated processes that are pre-scheduled and often run at regular intervals. If crons don't execute properly (for instance if memcached is broken) they may get backed up and never clear out. You can clear cron data manually from the `_options` table with SQL Syntax. First **SELECT** the rows and then **UPDATE** with a new (empty) value.

```
-----+-----+
1 row in set (0.01 sec)

mysql> SELECT * FROM kittenkamala_com_4.wp_gmrig8_options where option_name = 'cron';
database_name.prefix_table

[mysql> UPDATE kittenkamala_com_4.wp_gmrig8_options SET option_value = '' WHERE option_name = 'cron';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

Locate automatically loaded data (to delete it) in `_options` table. Autoloaded data is usually leftover from plugins and isn't always cleared out properly especially after uninstalls, causing it to pile up. Look how much autoloaded data is in my options table!

```
mysql> SELECT option_name, length(option_value) AS option_value_length FROM kittenkamala_com_4.wp_gmrig8_options WHERE autoload='yes' ORDER BY option_value_length DESC LIMIT 10;
```

option_name	option_value_length
jetpack_file_data	59168
rewrite_rules	19956
fs_accounts	12426
wp_gmrig8_user_roles	7782
cron	4434
fs_api_cache	3014
theme_mods_pixgraphy	2950
jetpack_active_plan	1719
theme_mods_nataraja	1702
jetpack_constants_sync_checksum	1383

10 rows in set (0.00 sec)

```
mysql> █
```

Shout out to Kinsta for having incredible documentation: <https://kinsta.com/knowledgebase/wp-options-autoloaded-data/>



Let's delete that first one as an example, using an SQL Syntax **DELETE** statement. SQL syntax is not case sensitive but using upper case for SQL Syntax is the standard because it provides more clarity.

```
mysql> DELETE FROM kittenkamala_com_4.wp_gmrig8_options WHERE autoload = 'yes' and option_name LIKE '%jetpack%' limit 10;  
Query OK, 10 rows affected (0.00 sec)
```

```
mysql> DELETE FROM kittenkamala_com_4.wp_gmrig8_options WHERE autoload = 'yes' and option_name LIKE '%jetpack%' limit 10;  
Query OK, 10 rows affected (0.01 sec)
```

```
mysql> DELETE FROM kittenkamala_com_4.wp_gmrig8_options WHERE autoload = 'yes' and option_name LIKE '%jetpack%' limit 10;  
Query OK, 10 rows affected (0.00 sec)
```

```
mysql> DELETE FROM kittenkamala_com_4.wp_gmrig8_options WHERE autoload = 'yes' and option_name LIKE '%jetpack%' limit 100;  
Query OK, 21 rows affected (0.01 sec)
```

```
mysql> DELETE FROM kittenkamala_com_4.wp_gmrig8_options WHERE autoload = 'yes' and option_name LIKE '%jetpack%' limit 100;  
Query OK, 0 rows affected (0.01 sec)
```

```
[  
mysql> █
```

# Database Optimization: Clean Up \_posts Table

Multiple versions of posts are saved during editing. These are called Revisions. Revisions are saved in the database \_posts table. Big sites, especially news sites and blogs, tend to have a lot of revisions cluttering up the db. **Delete old revisions** using SQL Syntax:

```
-----+-----+
4 rows in set (0.00 sec)

mysql> SELECT * FROM wp_ukv36e_posts WHERE post_type = "revision";

mysql> DELETE FROM wp_ukv36e_posts WHERE post_type = "revision";
Query OK, 4 rows affected (0.00 sec)

mysql>
```

Still working on the \_posts table, as part of housekeeping you can **delete trash posts** using **SELECT** and **DELETE**:

```
mysql> SELECT * FROM wp_gmrig8_posts WHERE post_status = "trash";

mysql> DELETE FROM wp_gmrig8_posts WHERE post_status = "trash";
Query OK, 3 rows affected (0.00 sec)
```

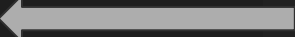
# Database Optimization: Clean Up \_posts Table

Prevent revisions from building up by setting a limit on revision quantity and frequency of automatic saves in your wp-config.php file:

```
$table_prefix = 'wp_gmrig8_';

define( 'WP_MAX_MEMORY_LIMIT', '200M' );
define( 'WP_MEMORY_LIMIT', '200M' );
define( 'WP_POST_REVISIONS', 10 );
define( 'AUTOSAVE_INTERVAL', 600 );

/* That's all, stop editing! Happy blogging. */
```



# Database Optimization: Clean Up term\_relationships

When the content of two (or more) posts are linked together, the links are saved in the database as something called a relationship in the a `_term_relationships` table. In this statement we're working with two tables again, only this time using two **SELECT**s instead of a **JOIN**.



```
mysql> SELECT * FROM wp_gmrig8_term_relationships WHERE term_taxonomy_id=1 AND object_id NOT IN (SELECT id FROM wp_gmrig8_posts);  
Empty set (0.00 sec)
```

```
[mysql> DELETE FROM wp_gmrig8_term_relationships WHERE term_taxonomy_id=1 AND object_id NOT IN (SELECT id FROM wp_gmrig8_posts);  
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> █
```

# Database Optimization: Clean Up \_postmeta

Orphaned (abandoned) postmeta data can be cleaned up using a **JOIN** statement, which connects two tables together to perform a function on both. Meta data is the data associated with the page or element that is not visible. Always **SELECT** the data you want to work with first as a precaution:

```
mysql> SELECT * FROM wp_9vv3wh_postmeta AS m LEFT JOIN wp_gmrig8_posts AS p ON m.post_id = p.ID WHERE p.ID IS NULL;
```

meta_id	post_id	meta_key	meta_value	ID	post_author	post_date	post_date_gmt	post_content	post_title	post_excerpt	post_status	comment_status	ping_status	post_password	post_name	to_ping	pinged	post_modified	post_modified_gmt	post_content_filtered	post_parent	guid	menu_order	post_type	post_mime_type	comment_count
2	3	_wp_page_template	default	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	

```
1 row in set (0.00 sec)
```

```
mysql>
```

# Database Optimization: Clean Up \_postmeta

Once you have viewed the data you want to work with and are ready to move forward you can go ahead and **DELETE**. In this statement m acts as a variable standing in for the postmeta content. NULL is a blank value (not to be confused with no value at all):

```
[mysql> DELETE m FROM wp_9vv3wh_postmeta AS m LEFT JOIN wp_gmrig8_posts AS p ON m.post_id = p.ID WHERE p.ID IS NULL;  
Query OK, 1 row affected (0.00 sec)
```

```
mysql> █
```

This isn't my code! I borrowed it from this awesome article :

<https://metinsaylan.com/8202/12-useful-sql-queries-for-wordpress-database-cleanup/>

# Database Optimization: Clean Up \_comments Table

Delete spam, pingbacks and trash comments from \_comments table. When a 3rd party site links to yours a notification is saved in the DB, called a pingback. Use a **LIMIT** to determine how many rows can be deleted at a time so as not to overwhelm the server:

```
[mysql> DELETE FROM wp_ukv36e_comments WHERE comment_approved = 'spam' limit 10;  
Query OK, 0 rows affected (0.00 sec)
```

```
[mysql> DELETE FROM wp_ukv36e_comments WHERE comment_type = 'pingback' limit 10;  
Query OK, 0 rows affected (0.05 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = 'trash' limit 10;  
Query OK, 0 rows affected (0.00 sec)
```

# Database Optimization: Clean Up \_comments Table

When you DO have something to delete, if you use a **LIMIT** you will have to run the command repeatedly until it says 0 rows affected. Command to delete unapproved comments:

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 10;  
Query OK, 10 rows affected (0.00 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 10;  
Query OK, 10 rows affected (0.00 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 100;  
Query OK, 100 rows affected (0.02 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 100;  
Query OK, 100 rows affected (0.00 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 100;  
Query OK, 100 rows affected (0.01 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 100;  
Query OK, 100 rows affected (0.00 sec)
```

Type **'exit'** to exit out of MySQL.

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 1000;  
Query OK, 1000 rows affected (0.06 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 1000;  
Query OK, 1000 rows affected (0.08 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 1000;  
Query OK, 1000 rows affected (0.05 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 1000;  
Query OK, 1000 rows affected (0.04 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 1000;  
Query OK, 581 rows affected (0.03 sec)
```

```
[mysql> DELETE from wp_ukv36e_comments WHERE comment_approved = '0' limit 1000;  
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> █
```



# Database Optimization: 'wp db optimize'

Always run a '**wp db optimize**' after working on the database. This runs a SQL Optimize and pushes changes through.

```
ldp- [redacted] $ wp db optimize
kittenkamala_com_4.wp_64qhsd_commentmeta
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_comments
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_links
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_options
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_postmeta
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_posts
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_term_relationships
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_term_taxonomy
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_termmeta
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_terms
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_usermeta
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_64qhsd_users
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_9vv3wh_commentmeta
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_9vv3wh_comments
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_9vv3wh_links
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_9vv3wh_options
note : Table does not support optimize, doing recreate + analyze instead
status : OK
```

```
kittenkamala_com_4.wp_gmrig8_woocommerce_order_items
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_payment_tokenmeta
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_payment_tokens
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_sessions
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_shipping_zone_locations
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_shipping_zone_methods
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_shipping_zones
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_tax_rate_locations
note : Table does not support optimize, doing recreate + analyze instead
status : OK
kittenkamala_com_4.wp_gmrig8_woocommerce_tax_rates
note : Table does not support optimize, doing recreate + analyze instead
status : OK
Success: Database optimized.
ldp- [redacted] $
```

More info on wp db optimize: <https://developer.wordpress.org/cli/commands/db/optimize/>

More info on SQL Optimize: <https://dev.mysql.com/doc/refman/5.7/en/optimize-table.html>

# Finish up with a 'wp cache flush'

Always run a '**wp cache flush**' after cleaning up and optimizing the database. This flushes the persistent object cache.



```
[[dp-: ]$ wp cache flush  
Success: The cache was flushed.
```

More info: <https://developer.wordpress.org/cli/commands/cache/flush/>

**AND THATS HOW YOU**

**OPTIMIZE A WORDPRESS INSTALL**