

A WordPress site can fail in surprisingly ordinary ways when its server runs out of disk space. The homepage may show a server error, the dashboard may refuse to load, images may stop uploading, or updates may fail. You might even see the message no space left on device when the disk appears to have room. That can happen if the server has run out of inodes—the capacity to create files—not just storage space.

This guide walks through fast, reversible checks first, then explains how to investigate disk space, logs, database credentials, hosting health, and wp-config.php. The goal is to identify the cause without making an outage worse. Before changing or deleting anything, record what you see and make a backup where possible.

Start With Fast, Reversible Outage Triage

Before changing WordPress files or deleting data, check whether the issue is widespread and capture the evidence. A disk-full outage can overlap with a database outage, a failed deployment, or a hosting incident, so avoid assuming that disk space is the cause based on one error alone.

1. **Record the time and exact error.** Note the time and time zone, the affected URL, what you were doing, and the complete message. Take a screenshot if the error appears in a browser or hosting panel.
2. **Check more than one page.** Try the homepage, a few internal pages, and /wp-admin/. Note whether the whole site is down or only a feature such as image uploads.
3. **Check your hosting provider's status page and control panel.** Look for maintenance notices, resource alerts, quota warnings, and service health indicators.
4. **Pause risky changes.** Don't install cleanup plugins, run updates, or repeatedly retry uploads while the server may be out of space.

Keep a small incident scratchpad. For example: "2026-10-15, 14:22 UTC — Media upload failed; message: 'The uploaded file could not be moved to wp-content/uploads.' Homepage still loads." Exact wording helps distinguish storage problems from permissions, database errors, and PHP failures.

Check Disk Space With the df -h Command

If your hosting account provides SSH, connect using the access details from your provider. Run `df -h` to see mounted filesystems, their size, used space, available space, and usage percentage:

```
df -h
```

Find the filesystem that contains your WordPress installation. If its Use% is close to or at 100%, the server or account may not have enough room to write files. Pay attention to the Avail column as well as the percentage. Hosting plans can also impose account quotas that differ from the filesystem-wide figures.

To check the filesystem for a particular path, run:

```
df -h /path/to/wordpress
```

Replace the example path with the actual path to your site. If you don't know it, your host's file manager or SSH session may show the current directory. Use the hosting panel's quota page too, because some managed WordPress plans don't expose the server's full disk information.

Check for Inode Exhaustion

Servers use an inode for each file and directory. A site can run out of inodes when it contains an enormous number of small files—even if there are gigabytes of disk space left. Check inode usage with:

```
df -i
```

If the relevant filesystem shows 100% inode use, you may see no space left on device while `df -h` still reports free [wordpress white screen database](#) storage. Don't try to fix this by deleting random WordPress files. First identify which directory contains the large file count, and ask your host for help if you can't safely inspect it.

Find What Is Using Space Without Deleting It

Once you confirm a space problem, inspect usage before removing anything. The following command reports the size of top-level directories under a path:

```
du -h --max-depth=1 /path/to/wordpress 2>/dev/null | sort -h
```

Some systems use a different version of `du`. If that command isn't supported, try `du -sh /path/to/wordpress/*`, or use your provider's file manager to review folder sizes. The `2>/dev/null` part hides permission warnings; it does not delete files.

Common space users include backups, old site archives, cache files, large debug logs, and uploads. Check whether a backup is the only copy before removing it. A backup can look expendable during an outage but may be your best route to recovery.

Clear Logs Carefully

Logs can grow very large, especially when an error repeats continuously. Look for log files in your hosting panel, the site directory, or a designated logs folder. Check their size and the recent entries before taking action. A log can reveal the cause of the outage, so don't erase your only diagnostic evidence.

If a log is clearly consuming space, download or copy it first if possible. Then use your host's log-management tool, or ask the provider for the safest way to rotate or truncate it. Avoid deleting active system logs or files outside your account. After any change, verify that the site works and that logging continues normally.

Finding What it may mean Safer next step Disk usage is near 100% The filesystem or account may have no room for writes Identify large files and check account quotas Inode usage is 100% There may be too many files, even with storage available Locate the file-heavy directory; don't delete blindly Disk appears available, but writes fail A hosting quota, permissions issue, or separate full mount may be involved Check the correct filesystem and provider limits

Check Hosting and Server Health

Disk space is only one part of server health. Review the hosting dashboard for account quotas, CPU or memory limits, PHP status, database service status, and recent incidents. If SSH is available, basic commands such as `uptime` and `free -h` can provide context about load and memory. These checks don't change server settings.

Also verify that the site's files and database are on the expected hosting account. Some setups use separate storage for the website, temporary files, backups, and the database. A full temporary directory or database volume may cause errors even when the WordPress directory looks fine.

If you use a managed host, share your recorded timestamps and error text with support. Ask them to check the account quota, disk and inode usage, database service health, and relevant server logs for the affected time. This

gives support a concrete investigation request rather than a vague report.



Check Database Credentials Before Editing Them

A database connection error is not proof that the disk is full. WordPress may show “Error establishing a database connection” because the database service is down, credentials are incorrect, the database name or host changed, or the database server itself has a storage problem. Check your host’s database status and account details before editing configuration.

In wp-config.php, WordPress stores the database connection settings. The main values usually look like this:



```
define( 'DB_NAME', 'database_name' );
```

```
define( 'DB_USER', 'database_user' );
```

```
define( 'DB_PASSWORD', 'database_password' );
```

```
define( 'DB_HOST', 'localhost' );
```

Your actual DB_HOST may not be localhost; use the value supplied by your host. Check for accidental edits, missing characters, extra spaces, or a recent password change. Don't paste database passwords into public tickets or chat. If you change a value, make sure the new value comes from the hosting provider or database administrator.

Understand the Relevant wp-config.php Basics

The wp-config.php file is usually in the WordPress root directory, alongside folders such as wp-admin and wp-content. It contains important site settings and may include secrets. Don't publish it, email it without protection, or overwrite it with a generic sample file.

Before editing, create a copy in a secure location and confirm you can restore it. Check file permissions through your host's recommended settings; don't make the file broadly writable as a troubleshooting shortcut. One setting that may affect disk usage is WordPress debugging:

```
define( 'WP_DEBUG', true );
```

Debugging may be useful when diagnosing an issue, but leaving verbose logging enabled can create large files. The setting WP_DEBUG_LOG can direct messages to a debug log, often wp-content/debug.log. Don't assume that file is safe to delete until you've reviewed or saved it, and don't enable debugging on a live site without a reason. A host's PHP or error logs may be more appropriate for diagnosing server-level failures.

Make Changes With a Rollback Plan

When you find a likely cause, change one thing at a time. Prefer a hosting control panel's documented cleanup or log-rotation tools over manually removing unfamiliar files. Don't delete WordPress core files, database files, or uploads simply because they look large.

- **What changed:** Write down the exact file, setting, or hosting action, plus the time.
- **How to roll back:** Record the backup or original value needed to restore it.
- **Verify:** Recheck the site, dashboard, uploads, and disk or inode usage.
- **Preserve evidence:** Keep relevant error text and logs until the cause is understood.

If you remove a verified redundant archive or rotate a log, note its location and whether it was saved first. If a change makes the site worse, restore the previous file or setting rather than trying several unrelated fixes in a row.

When to Escalate

Stop and involve your host or server administrator if you can't identify the affected filesystem, the database volume is full, the account quota disagrees with `df -h`, inode usage is exhausted, or you would need to delete unfamiliar system files. Also escalate if the site remains down after space is available—another issue may have been triggered or revealed by the outage.

Send a concise report with the site name, affected URL, time and time zone, exact error message, output or screenshots from `df -h` and `df -i` if available, and any changes already made. Ask the provider to confirm the account's disk and inode limits, inspect server and database health, and advise on safe cleanup. Keep passwords and private configuration values out of the report.

Quick Checklist

- Record the timestamp, affected pages, and exact error message.
- Check the hosting status page, account quota, and database health.
- Run `df -h` and `df -i` if SSH is available.
- Inspect large directories and logs before removing anything.
- Verify database settings in `wp-config.php` without exposing credentials.
- Back up before edits; document what changed and how to roll back.
- After recovery, monitor usage and arrange a safe log, backup, or storage plan.

A full disk is often fixable, but rushed cleanup can turn a temporary outage into lost data. Start with read-only checks, confirm whether storage or inodes are exhausted, preserve useful evidence, and make only changes you can reverse.