SQLite 3 for database administrators

What a database administrator needs to know about SQLite 3, the default database of Exponential 6.0.x - what the eZSQLite3DB driver adds to the library, the four files of a database, the connection PRAGMAs and why each...

Source file: `doc/eng-us/help/database-administrators/sqlite-for-database-administrators.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/database-administrators/sqlite-for-database-administrators.md

This page is for database administrators who are asked to run an Exponential site on SQLite 3, the installer's default, and who know server databases better than file databases. It answers: what the driver adds to the SQLite library, which files make up a database, which connection settings are applied and why, how readers and the writer coexist, and where SQLite stops fitting. Source: 9.2 SQLite 3 and the driver `lib/ezdb/classes/ezsqlite3db.php`.

Contents

1. Why SQLite is the default
2. Where the database is
3. What the driver does at every connection
4. The PRAGMA defaults
5. WAL: readers and the writer side by side
6. The SQL compatibility layer
7. Where SQLite fits
8. Checklist

Why SQLite is the default

The setup wizard preselects it (`settings/setup.ini [DatabaseSettings] DefaultType=sqlite3`), `exp:install` uses it when no `--db` is given, and the kickstarter template names it first. It needs no server process, no network and no accounts. With the driver Exponential ships it is a full production database for a site served from one machine: most page views are answered by caches before they reach it, and concurrent publishes queue instead of failing.

Where the database is

The `Database` setting is a file name:

- a plain name (`exponential.db`) lives in `var/storage/sqlite3/` (the constant `eZSQLite3DB::STORAGE_DIRECTORY`);
- a name starting with `/` is an absolute path;
- `:memory:` is a throw-away database (tests only).

`exp:install` names the file `exponential.db`, the setup wizard proposes `sqlite.db`. A missing

directory is created with mode `0775`; a missing file is created empty. A wrong `Database` value therefore does not fail at connect: it opens a new, empty database, and the site reports missing tables.

Shell

```
php bin/php/console exp:ini get site.ini/DatabaseSettings/Database --allow-root-user
ls -l var/storage/sqlite3/
```

A database is four files: the database, `-wal`, `-shm` and the driver's `.writer-lock` (SQLite files, ownership and permissions).

What the driver does at every connection

1. **Resolves the file** as above. A file that cannot be opened (after `ConnectRetries` further attempts) raises `eZDBNoConnectionException` with the full path.
2. **Registers functions** SQLite lacks: `md5()` and the aggregates `BIT_OR` and `BIT_AND`, which the content engine uses to fold language masks.
3. **Applies the PRAGMAs**, the busy timeout first.
4. **Switches to WAL**, only when the file is not in WAL mode yet (the mode is stored in the file).

The PRAGMA defaults

PRAGMA	Driver default	SQLite's own default	Why
`synchronous`	`NORMAL`	`FULL`	With WAL still safe against corruption on a crash; only the last commits before a power loss can be lost
`cache_size`	`-65536` (64 MB)	about 2 MB	The page cache of each connection
`mmap_size`	`268435456` (256 MB)	`0`	Reads through memory-mapped I/O, shared through the operating system's file cache
`temp_store`	`MEMORY`	file	Sorts and temporary tables stay in memory
`busy_timeout`	`5000` ms	`0`	A statement that meets a lock waits instead of failing at once

Change or add PRAGMAs with `SQLitePragmas[]`, one `name=value` per entry. The driver accepts only letters and underscores in the name and letters, digits, `_` and `-` in the value, and skips anything else. WAL always wins: a `journal_mode` entry is switched back to WAL.

INI

```
[DatabaseSettings]
SQLitePragmas[]
SQLitePragmas[]=cache_size=-16384
SQLitePragmas[]=journal_size_limit=67108864
```

`cache_size` is per connection: count up to 64 MB for every PHP-FPM worker and every Velocity worker that has opened the database, or lower it as above (16 MB).

WAL: readers and the writer side by side

- **Readers never block the writer, and the writer never blocks readers.** Page views continue while an editor publishes.
- **There is one writer at a time.** The driver makes the next writer wait its turn (SQLite: the write queue and locking).
- **A checkpoint** copies the WAL back into the database, automatically at about 1000 pages; under constant reading the WAL file can grow for a while (SQLite maintenance).
- **All processes must be on one machine.** WAL uses shared memory, so it does not work across a network file system.

The SQL compatibility layer

The kernel's SQL was written for MySQL first. The driver bridges the gaps: it rewrites `UPDATE ... JOIN` into `UPDATE ... FROM` (SQLite 3.33 or later), and accepts and ignores MySQL session statements (`SET NAMES`, `SET FOREIGN_KEY_CHECKS`, `SET CHARACTER SET`, `SET AUTOCOMMIT`, `SET SQL_MODE`, `SET UNIQUE_CHECKS`). SQLite compares with case for `=` and ignores case for ASCII letters in `LIKE`.

Where SQLite fits

It fits one machine, any number of readers and an editorial team. The project's load tests had concurrent readers scale with the number of processes and 16 concurrent writers complete without errors, queued. Choose a database server when more than one machine must write, when many editors publish large changes at the same moment all day, or when you need replication or database-level users and grants (Choosing an engine).

Checklist

- ☐ PHP's SQLite library is 3.33 or later (`php --ri sqlite3`).
- ☐ `PRAGMA journal_mode` prints `wal`.
- ☐ Every process that uses the site can write the directory and the files.
- ☐ The database is not reachable over HTTP.
- ☐ Backups use `.backup` or `VACUUM INTO`, never a plain copy of a busy file (Backups per engine).
- ☐ Memory is sized for `cache_size` per worker.

Related pages

- SQLite: the write queue and locking
- SQLite maintenance: ANALYZE, VACUUM and the WAL file
- SQLite for hosts
- SQLite database support, The SQLite3 database driver
- Free documentation: this page is free to copy, translate and improve under the GNU FDL 1.3.

Become an Exponential Partner

Partners share their information, provide funding, contribute back and develop the products with us, and that is how Exponential grows. A partner website and dashboard are planned for mid-2027: please become a partner today to speed up our growth rate and land new features faster.

Partners: start here | How to apply

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.