

Self-contained native builds

Written by Vikrant Rathore with assistance from Ronak Rathore.

`./build.sh` builds the CLI on macOS or Linux with **statically linked SQLite, sqlite-vec and OpenSSL**. Deploy the resulting `bin/sqlodin` with its license notices and your configuration/certificate files. No shared SQLite, sqlite-vec or OpenSSL installation is required. Normal platform libraries such as macOS `libSystem` or Linux `libc` remain dependencies; this is not a fully static `libc` or cross-platform binary.

Build

Install Odin, Python 3, a C compiler and platform SDK/headers, `ar`, `Make` and `Perl`. Odin also needs its platform link driver (normally `Clang` on Linux). No Homebrew/apt database or TLS development package is required. The tested targets are macOS `arm64` and Linux `x86_64`; native target mappings also exist for macOS `x86_64` and Linux `aarch64` but have not been tested here. Cross-compilation is not implemented.

```
git submodule update --init --recursive
python3 tools/build_cli.py --jobs 4
# Equivalent bootstrap, including the submodule initialization:
./build.sh
```

The complete `paxos-odin` gitlink remains pinned. `tools/native_sources.json` pins SQLite **3.51.3**, sqlite-vec **0.1.9**, and OpenSSL **3.5.8 LTS**. SQLite and vector C sources/headers are SHA-256 checked before compilation; the OpenSSL release tarball is checked against its pinned official SHA-256. No floating branch, checksum fetched at build time, or system-library fallback selects application dependencies.

SQLite enables FTS5, disables extension loading, and compiles JSON support as provided by the pinned release. sqlite-vec uses static registration with filesystem helpers disabled. OpenSSL builds with `no-shared`, `no-dso`, `no-module` and `no-engine`. The service disables automatic OpenSSL configuration loading and uses explicit `trust/certificate/key` paths. It does not depend on external provider modules, `openssl.cnf`, system CA configuration or a runtime OpenSSL executable. This build does not claim FIPS certification.

The bundled `build/native/openssl` utility is available for development certificate fixtures. It is not needed to run the SQL service. Supply `OPENSSL_CONF=/dev/null` when using this utility without an installed OpenSSL configuration file.

The CLI also links `shell.c` from the same pinned SQLite archive, verified by the `sqlite_shell` source entry. `tools/build_shell.py` produces its static archive and `source/build` manifest. No installed `sqlite3`, `readline` or `editline` library is required. The Odin cluster editor supplies its own bounded in-memory history. The local shell and client usage are documented in the CLI guide.

Embedded Odin applications

```
python3 tools/build_native.py --sqlite-only
odin run examples/multimaster_search.odin -file
```

Import `src` as before. The engine uses the same generated static SQLite/vector archives; it does not import TLS or link OpenSSL. There is no new C ABI or embedded Python binding in this change. Build the native dependencies within the `SQLodin` checkout before compiling an application that imports it.

Cache and provenance

Downloads stay in build/downloads/; compiled archives and configuration metadata stay in build/native/. File locking prevents concurrent native builders from interleaving writes. Reuse checks compiler/platform identity, build flags, pins, builder source and output hashes. Changing an archive invalidates reuse.

```
python3 tools/build_cli.py --offline          # uses verified local caches only
python3 tools/build_native.py --force        # rebuilds pinned sources
python3 tools/check_native_linkage.py bin/sqlodin --output build/linkage.json
```

The CLI builder automatically rejects shared database/TLS dependencies using `otool -L` on macOS or `readelf -d` on Linux. `bin/sqlodin.build.json` records binary and archive hashes, toolchain information and linkage output. Keep this manifest and `bin/licenses/` beside release artifacts. Builds are source-pinned; the host compiler/SDK and Odin installation are recorded rather than downloaded/pinned, so bit-for-bit reproducibility across arbitrary machines is not claimed.

A security update requires updating the pin/checksum, rebuilding and retesting the binary. System package updates cannot update libraries already linked into a binary. An SQLite source/compile-option change also changes the durable engine fingerprint; there is no automatic migration of older incompatible stores. Keep previous data and binaries until a separately validated migration is available.

The build tools use OpenSSL's official release and checksum and documented build options.

Recorded checks

The validation manifest links the saved macOS arm64 and Linux x86_64 evidence: 91 engine tests per platform, 21 disk-backed service/search/SQLAlchemy checks per platform, 15 TLS checks per platform, and 19 direct mTLS checks across the three authorized Linux instances. Offline archive reuse and four negative source/cache checks passed locally; offline reuse also passed on Linux. Python's 23 tests passed on both platforms. The embedded Odin search example and isolated Python wheel installation passed locally. These checks establish the tested build and feature paths, not production qualification.