Database and SQL standards

How Exponential relates to the SQL standard (ISO/IEC 9075) and the dialects of SQLite, PostgreSQL, MySQL, MariaDB, Oracle and MongoDB: the eZDB abstraction, dialect handling, schemas, character sets and CI coverage.

Source file: `doc/eng-us/standards/databases-and-sql.md` Last changed 2026-11-04
https://doc.exponential.earth/exphelp/viewer/standards/databases-and-sql.md

This page is for developers who write SQL for Exponential or review extension code that does, and for administrators who need to know how the database layer relates to the SQL standard and how well each database is covered. It answers three questions: which SQL Exponential writes, how the differences between the supported databases are handled, and which of them the project's CI tests.

Which databases are supported, in which order, with which versions, settings, installers and backups is the subject of Database support. Read that page first; this one does not repeat it.

Summary

Standard	Version	Status	Evidence (short)
SQL, ISO/IEC 9075	no edition targeted	**Partly complies**	the kernel writes a portable subset and asks the driver for what differs (`lib/ezdb/classes/ezdbinterface.php`); no conformance claim
SQLite 3 dialect	SQLite 3.33 or later	**Complies**	`eZSQLite3DB` (`lib/ezdb/classes/ezsqlite3db.php`); CI: driver unit tests on PHP 8.1 to 8.5
PostgreSQL dialect	PostgreSQL 13 or later	**Complies**	`eZPostgreSQLDB`; CI: database tests on `postgres:15`
MySQL dialect	8.0 tested	**Complies**	`eZMySQLiDB`; CI: database tests on `mysql:8.0`
MariaDB dialect	10.5 seen working	**Unverified**	same driver as MySQL; not in CI
Oracle dialect	19c or later	**Unverified**	`eZOracleDB` in the `ezoracle` extension 2.3.7; not in the kernel's CI
MongoDB (not SQL)	server 6.0 or later	**Partly complies**	`expMongoDB` translates SQL-shaped calls; documented limits; adapter unit tests only
Unicode storage (UTF-8)	per engine	**Partly complies**	`utf8mb4`, `UTF8`, `AL32UTF8`; the MySQL and MariaDB connection uses three-byte `utf8`

The SQL standard (ISO/IEC 9075)

What it is

ISO/IEC 9075 is the international standard for SQL, published by ISO and IEC in several parts; the current edition is SQL:2023. No database implements all of it, and every engine adds its own syntax.

How Exponential complies

Exponential does not target an edition of the standard. Its kernel SQL is written in a subset that the supported engines share, and everything that differs goes through the driver:

- **One interface.** Every database is reached through `eZDBInterface` (`lib/ezdb/classes/ezdbinterface.php`); `eZDB::instance()` (`lib/ezdb/classes/ezdb.php`) returns the driver that `settings/site.ini [DatabaseSettings] DatabaseImplementation` names through `ImplementationAlias[]`.
- **Helpers for what differs:** `subString()`, `concatString()`, `md5()`, `bitAnd()`, `bitOr()`, `escapeString()`, `lastSerialID()`, `generateSQLINStatement()`, `eZTableList()` and `databaseName()` (`mysql`, `postgresql`, `sqlite`, `mongo` or `oracle`).
- **Row limits as parameters.** `arrayQuery( $sql, array( 'offset' => ..., 'limit' => ... ) )` lets the driver append its own syntax; the drivers accept only numeric values (`is_numeric()` in each `arrayQuery()`).
- **An engine-neutral schema.** The schema and initial data are kept in the generic `.dba` format (`share/db_schema.dba`, `share/db_data.dba`), which a schema handler turns into each engine's DDL (see Schemas below).
- **Portable identifiers.** `eZLintSchema` (`lib/ezdbschema/classes/ezlintschema.php`) checks the schema against `settings/dbschema.ini [LintChecker]` (`TableLimit=26`, `FieldLimit=30`, `IndexLimit=30`), limits that keep names usable on every engine.

Status

Partly complies. Kernel SQL is standard where the engines agree and adapted where they do not. Parts of the code base still use MySQL syntax that the SQLite driver rewrites (see below), and extension code is not covered.

Known gaps

- Exponential makes no claim of conformance to any part or edition of ISO/IEC 9075.
- Prepared statements with bound parameters are not used by the kernel's SQL; `bindingType()` and `bindVariable()` exist on the interface, but kernel queries are built as strings with escaped values.
- Hand-written SQL in extensions often uses MySQL-only syntax (backticks, `LIMIT offset, count`, `SET` statements); the cross-engine notes list what fails where.

The eZDB rules for safe SQL

Because statements are built as strings, two rules are mandatory for any value that comes from a request, a setting or the database:

1. **Escape every string** with the driver: `$db->escapeString( $value )`, inside single quotes. Each driver uses the engine's own function (`mysqli_real_escape_string()`, `pg_escape_string()`, `SQLite3::escapeString()`; the Oracle driver doubles single quotes). `null` becomes the empty string.
2. **Cast every number** before it goes into SQL: `(int)$id`. A cast cannot be escaped around.

PHP

```php
$db = eZDB::instance();
$rows = $db->arrayQuery(
    "SELECT id, name FROM ezsection WHERE identifier = '" . $db->escapeString( $identifier ) . "'
"
    . " AND id <> " . (int)$excludeId,
    array( 'offset' => 0, 'limit' => 10 )
);
```

- Prefer `eZPersistentObject::fetchObjectList()` and the other `eZPersistentObject` methods for tables with a definition; they escape for you (`kernel/classes/ezpersistentobject.php`).
- Transactions nest by counting: `$db->begin()` and `$db->commit()` may be called inside each other, and only the outermost pair sends `BEGIN` and `COMMIT` (`eZDBInterface::begin()`, `commit()`, `TransactionCounter`). `$db->rollback()` rolls back the whole transaction.
- The security fixes of 6.0.13 (SQL injections fixed with integer casts and `escapeString()`) are described in Security hardening of 6.0.13.

Dialect differences, driver by driver

What each driver does where the engines differ. MongoDB is not an SQL engine; its column describes what the adapter does instead ("not checked" marks a cell this page did not verify in the adapter's code).

Topic	**SQLite 3 (** `eZSQLite3DB`**)**	**PostgreSQL (** `eZPostgreSQLDB`**)**	**MySQL, MariaDB (** `eZMySQLiDB`**)**	**Oracle (**`eZOracleDB`**,** `ezoracle`**)**	**MongoDB (** `expMongoDB`**)**
Row limit	`LIMIT offset, count`	`LIMIT count OFFSET offset`	`LIMIT offset, count`	`OFFSET n ROWS FETCH NEXT m ROWS ONLY` (the ISO form)	`limit` and `skip` options of the query
New ids	`lastInsertRowID()`	sequence `<table>_<column>_seq`, `currval()`	`mysqli_insert_id()` (`AUTO_INCREMENT`)	sequence `s_<table>` (`se_<table>` for extension tables), `currval`	a counter document per sequence in the `ezsequence` collection
String concatenation	`||`	`||`	`concat( ... )`	`||`	`concat( ... )`, read by the adapter
Substring	`substr( s, n[, len] )`	`substring( s from n [for len] )`	`substring( s from n [for len] )`	`substr( s, n[, len] )`	not checked

Topic	SQLite 3 (eZSQLite3DB)	PostgreSQL (eZPostgreSQLDB)	MySQL, MariaDB (eZMySQLiDB)	Oracle (eZOracleDB, ezoracle)	MongoDB (expMongoDB)
MD5	`md5()` registered as a user function	`encode( digest( s, 'md5' ), 'hex' )`, needs `pgcrypto`	`MD5()`	`STANDARD_HASH( s, 'MD5' )`, lower-case hex	a quoted literal is hashed in PHP
Bitwise AND, OR	&, |; aggregates `BIT_OR`, `BIT_AND` registered	&, |	&, | cast to `SIGNED`	`BITAND()`; OR as `a + b - BITAND( a, b )`	&, |, evaluated by the adapter
MySQL `UPDATE ... JOIN`	rewritten to a form SQLite accepts (`rewriteJoinedUpdate()`, needs 3.33)	not rewritten	native	not rewritten	not applicable
MySQL session `SET` statements	accepted and ignored	`SET NAMES` only	native	not supported	not applicable
Start of a transaction	`BEGIN IMMEDIATE`, after the writer lock	`BEGIN WORK`	`BEGIN WORK`	no auto commit (`OCI_DEFAULT`)	none: each operation commits on its own
Empty string	stored as `''`	stored as `''`	stored as `''`	stored as NULL; `OracleEmptyStringForNull=enabled` returns `''`	not checked
String comparison	= with case, LIKE without case for ASCII	with case	without case (default collations)	with case; `OracleCaseInsensitive=enabled` for linguistic comparison	with case unless a collation is set

Sources: `lib/ezdb/classes/ezsqlite3db.php`, `ezpostgresqldb.php`, `ezmysqlidb.php`, `expmongodb.php`, and `extension/ezoracle/ezdb/dbms-drivers/ezoracledb.php`; the cross-engine notes of chapter 9, section 9.8; the SQLite3 driver specification.

Isolation levels. No driver sets a transaction isolation level; a search of `lib/` and `kernel/` for `SET TRANSACTION` and `ISOLATION LEVEL` finds nothing. Each engine runs at its own default. SQLite serializes writers itself: the driver takes an exclusive lock on `<database>.writer-lock` and then runs `BEGIN IMMEDIATE`, waiting up to `[DatabaseSettings] SQLiteTransactionWait` seconds (default 25).

Schemas: .dba files and schema handlers

- The kernel's schema and data are kept as PHP arrays in `share/db_schema.dba` and `share/db_data.dba`. Extensions ship their own `share/db_schema.dba` and `share/db_data.dba` the same way.
- `settings/dbschema.ini [SchemaSettings]` maps a database type to a handler: `SchemaHandlerClasses[mysql]` and `[mysqli]=eZMysqlSchema`,

`[postgresql]=eZPgsqlSchema`, `[sqlite]` and `[sqlite3]=eZSQLiteSchema`, `[mongo]=expMongoSchema` (all in `lib/ezdbschema/classes/`). The Oracle handler `eZOracleSchema` comes with `ezoracle` (`extension/ezoracle/ezdb/dbms-schema/`).

- Engine-specific translations live in the same file, for example `[mysql] IndexTranslation[]` (prefix lengths for indexes on long text columns) and `[postgresql] ColumnTypeTranslation[]`.
- Generated SQL per engine: `kernel/sql/mysql/`, `kernel/sql/postgresql/`, `kernel/sql/sqlite/`, plus `kernel/sql/common/`. Upgrades: `update/database/mysql/`, `update/database/postgresql/`, `update/database/sqlite/`; Oracle's are in `ezoracle`; MongoDB has none.
- Tools: `bin/php/ezsqldumpschema.php` (dump to `.dba`) and `bin/php/ezsqlinsertschema.php` (load into an engine with `--type`). Setup > Upgrade check compares a site's tables with the release schema (`lib/ezdbschema/classes/ezdbschemachecker.php`; The upgrade check).

Character sets

Database	Storage	Notes
SQLite 3	UTF-8	nothing to choose
PostgreSQL	`UTF8`	create with `--encoding=UTF8 --template=template0`
MySQL, MariaDB	`utf8mb4` recommended	the connection setting `Charset=utf-8` is sent as `utf8`, which both servers treat as the three-byte `utf8mb3`
Oracle	AL32UTF8	the driver connects in AL32UTF8 and warns when a charset has no Oracle equivalent
MongoDB	UTF-8 (BSON)	nothing to choose

The full table, with collations and the wizard's check, is in Database support, Character sets and collations.

Status: Partly complies. Text is stored as UTF-8 everywhere, but on MySQL and MariaDB characters outside the Basic Multilingual Plane (emoji, for example) cannot pass through the connection, even when the tables are `utf8mb4`.

Per database

Minimum versions below are the practical minimums stated in Database support; the setup wizard's older floors are in `eZSetupDatabaseMap()` (`kernel/setup/ezsetupcommon.php`: MySQL 4.1.1, PostgreSQL 8.0, SQLite 3.0.1, MongoDB 4.0, Oracle 19.0).

SQLite 3

- Driver `eZSQLite3DB`, schema handler `eZSQLiteSchema`, PHP extension `sqlite3`. Practical minimum 3.33.
- Adds what SQLite lacks: `md5()`, the `BIT_OR` and `BIT_AND` aggregates (registered at connection time; the driver's comment says the content engine folds language masks with `BIT_OR`), the rewrite of `UPDATE ... JOIN`, and a write queue.
- Specification: The SQLite3 database driver.

- CI: the PHPUnit job (PHP 8.1 to 8.5) runs `tests/tests/lib/ezdb/eZSQLite3DBQueryCountTest.php`, `eZSQLite3MirrorTest.php` and `eZSQLite3TransactionWaitTest.php`. The legacy database test suite does not run on SQLite in CI.
- **Status: Complies.**

PostgreSQL

- Driver `eZPostgreSQLDB`, schema handler `eZPgsqlSchema`, PHP extension `pgsql`. Practical minimum 13 (the installer creates the trusted extension `pgcrypto`).
- Sequences named `<table>_<column>_seq` (`DEFAULT nextval(...)` in `ezpgsqlschema.php`).
- CI: job "Database tests (postgresql, PHP 8.3)" in `.github/workflows/phpunit.yml`, service `postgres:15`, `CREATE EXTENSION IF NOT EXISTS pgcrypto`, then `tests/runtests.php --dsn postgresql://... --db-per-test`. Driver tests: `tests/tests/lib/ezdb/eZPostgreSQLDBTest.php`, `eZPostgreSQLDBEscapeStringNullTest.php`.
- **Status: Complies.**

MySQL

- Driver `eZMySQLiDB`, schema handler `eZMysqlSchema`, PHP extension `mysqli`. InnoDB tables (transactions need them).
- The only engine with built-in read replicas (`UseSlaveServer`, `SlaveServerArray[]`).
- CI: job "Database tests (mysql, PHP 8.3)", service `mysql:8.0`, the same `tests/runtests.php` run. Driver test: `tests/tests/lib/ezdb/eZMySQLiDBFKTest.php`.
- **Status: Complies.**

MariaDB

- The MySQL driver and schema handler; everything above applies.
- Not in the project's CI. The MySQL 8.0 job exercises the same driver; MariaDB-specific behavior is not tested.
- **Status: Unverified** (no CI job; this page did not run it).

Oracle

- Driver `eZOracleDB` and schema handler `eZOracleSchema` in the `ezoracle` extension (version 2.3.7, required by the root `composer.json` as `~2.3.7`; activated only for Oracle sites). PHP extension `oci8`. Oracle 19c or later in `AL32UTF8`.
- Uses the ISO row limit syntax `OFFSET ... ROWS FETCH NEXT ... ROWS ONLY`, `STANDARD_HASH` for MD5 and `BITAND`.
- Not in the kernel repository's CI; Oracle-only kernel tests skip on other engines.
- **Status: Unverified** (no CI job in the kernel repository; this page did not run it).

MongoDB

- Adapter `expMongoDB` (`lib/ezdb/classes/expmongodb.php`) on the PHP extension `mongodb` and the library `mongodb/mongodb`; schema handler `expMongoSchema`. MongoDB is not an SQL database: the adapter translates the kernel's SQL-shaped calls into `find`, `aggregate`, insert, upsert and delete operations.
- Transactions are not used: `beginQuery()`, `commitQuery()` and `rollbackQuery()` do nothing, so a group of writes is not atomic and a rollback undoes nothing.
- SQL it cannot translate returns an empty result and logs a warning; see the limits in Database support and MongoDB as the database.
- CI: the `mongodb` suite of `phpunit.xml` (`expMongoDBAdapterTest.php`, `expMongoDBDeclaredDefaultTest.php`, no server). The `mongodb-live` suite needs running servers and is not run in CI.
- **Status: Partly complies.**

Known gaps

- MariaDB and Oracle have no CI job in the kernel repository.
- The legacy database suite runs on MySQL and PostgreSQL only, on PHP 8.3 only; the other PHP versions run the tests that need no database server.
- MongoDB writes are not transactional.
- Kernel SQL is built as strings; safety depends on every caller following the escape and cast rules above.
- No isolation level is chosen; code that depends on one is engine-dependent.
- `tests/tests/lib/ezdb/ezmysqlcharsettest.php` and the other files without the `Test.php` suffix in that directory are not picked up by the suites in `phpunit.xml`.

Related pages

- Database support and chapter 9, Databases
- The SQLite3 database driver, Database drivers and installers, September 2026, SQLite and Oracle drivers
- MongoDB as the database, SQL query cache
- Continuous integration
- API and protocol standards

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.