

Schema Model

Schema Model

Anonymous · 08/10/2026

Schema Model

Derafu ETL describes a database structure with its own model, independent from Doctrine DBAL and from any spreadsheet format: a `Schema` has tables, a table has columns, indexes and foreign keys. Every schema source reads into this model and every schema target writes from it.

Indexes

An index (`Derafu\ETL\Schema\Index`, `IndexInterface`) has:

Property	Type	Meaning
<code>name</code>	<code>string</code>	The index name.
<code>columns</code>	<code>string[]</code>	The indexed columns, in order.
<code>type</code>	<code>IndexType</code>	The kind of index.
<code>clustered</code>	<code>bool</code>	Whether the index is clustered.

`Derafu\ETL\Schema\Enum\IndexType` is a string-backed enum:

Case	Value
<code>IndexType::REGULAR</code>	<code>regular</code>
<code>IndexType::UNIQUE</code>	<code>unique</code>
<code>IndexType::FULLTEXT</code>	<code>fulltext</code>
<code>IndexType::SPATIAL</code>	<code>spatial</code>

```
use Derafu\ETL\Schema\Enum\IndexType;
use Derafu\ETL\Schema\Index;

$regular = new Index('idx_name', ['name']);
$unique = new Index('idx_email', ['email'], IndexType::UNIQUE);
$fulltext = new Index('idx_body', ['body'], IndexType::FULLTEXT);
$clustered = new Index('idx_order', ['order_id', 'line'], IndexType::REGULAR, true);

$unique->isUnique();      // true (it is `type === IndexType::UNIQUE`).
```

```
$clustered->isClustered(); // true.
```

`isUnique()` is derived from the type, so the two can never contradict each other. To change an index use `setType()` and `setClustered()`.

The enum values are the ones stored in a spreadsheet (see below), so they are part of the file format and do not change.

What each target does with them

Target	Type	Clustered
Doctrine	Mapped to Doctrine's own <code>IndexType</code> .	Passed to the index.
Spreadsheet	Stored as <code>type</code> .	Stored as <code>clustered</code> .
Text	<code>INDEX</code> , <code>UNIQUE INDEX</code> , <code>FULLTEXT INDEX</code> , <code>SPATIAL INDEX</code> .	A <code>CLUSTERED</code> line under the index.
Markdown	<code>INDEX</code> , <code>UNIQUE</code> , <code>FULLTEXT</code> , <code>SPATIAL</code> .	<code>CLUSTERED</code> in the <i>Clustered</i> column.
D2	<code>INDEX</code> , <code>UNIQUE</code> , <code>FULLTEXT</code> , <code>SPATIAL</code> .	Not shown.
SQL (SQLite)	<code>UNIQUE</code> is honored.	Ignored.

The SQLite target has no `FULLTEXT` or `SPATIAL` index through `CREATE INDEX`, and no clustered indexes, so those are created as regular indexes there. Whether `clustered` has any effect depends on the platform of the database that receives the Doctrine schema (for example SQL Server).

Schema in a Spreadsheet

`SpreadsheetSchemaTarget` writes the schema into a sheet (`__schema` by default) with one row per element, and `SpreadsheetSchemaSource` reads it back. Each row has a `type`, a `name` and its properties. An index row looks like this:

```
{
  "type": "index",
  "name": "invoice.idx_number",
  "properties": {
    "columns": ["invoice_number"],
    "type": "unique",
    "clustered": false
  }
}
```

Files from previous versions

Older versions wrote `unique` and `flags` instead of `type` and `clustered`:

```
{ "columns": ["invoice_number"], "unique": true, "flags": ["clustered"] }
```

Those spreadsheets are still read:

- `unique: true` becomes `IndexType::UNIQUE`.
- The flag `clustered` becomes `clustered: true`.
- The flags `fulltext` and `spatial` become the matching `IndexType`.
- Any other flag throws an exception, instead of being dropped silently.

An index row with a `type` that is not one of the four values also throws. When the file is written again it uses the current format.

Last updated on 08/10/2026