

# SeSQL installation guide

## Contents

<b>1 Prerequisite</b>	<b>1</b>
<b>2 Installation</b>	<b>1</b>
<b>3 Configuration</b>	<b>1</b>
3.1 ORM choice . . . . .	1
3.2 Text search configuration . . . . .	1
3.3 Clean-up/filters . . . . .	2
3.4 Indexes and fields . . . . .	2
3.5 Types and tables . . . . .	2
3.6 Query parameters . . . . .	3
3.7 Reindexing daemon . . . . .	3
3.8 Search history and statistics . . . . .	3
3.9 Additional features . . . . .	3
<b>4 Constraints</b>	<b>4</b>
<b>5 Upgrading</b>	<b>4</b>

## 1 Prerequisite

- Python 2.5 or 2.6
- Django 1.2 or 1.3 (or SQLAlchemy, see sqlalchemy.txt)
- PostgreSQL 8.4 or 9.x
- GenericCache from <http://pypi.python.org/pypi/GenericCache/1.0.2>

SeSQL may work with later versions of those software, but was not tested with them.

## 2 Installation

SeSQL is a standard Django application. Just drop it in the `apps/` directory, and add it to enabled applications into `settings.py`.

## 3 Configuration

Before using SeSQL you **must** configure it. The configuration file must be named `sesql_config.py` in the python path (usually the project).

For a summary of how to quickly configure SeSQL, please refer to the tutorial. This document contains a detailed list of all recognized options.

### 3.1 ORM choice

Since version 0.10, SeSQL can work with non-django ORM. See `sqlalchemy.txt` for more informations about it.

### 3.2 Text search configuration

PostgreSQL's full text search is based on the concept of *text search configuration* (TSC). Those configuration are detailed on the PostgreSQL manual, and allow to control things like stopwords. SeSQL requires a default TSC, and can support additional TSCs to be used on specific indexes.

SeSQL recognizes the following options related to text search configuration :

**TS\_CONFIG\_NAME** Name of the primary *text search configuration* to create in the PostgreSQL database and use in full text fields where a specific TSC is not specified.

**STOPWORDS\_FILE** Name of the stopwords file. This file **must** be where PostgreSQL will look for (`/usr/share/postgresql/8.4/tsearch_data/`) and **must** only contain plain ascii characters. An example command to generate the file for the french language is :

```
LC_ALL=fr_FR.UTF-8 iconv -f utf-8 -t ascii//TRANSLIT \  
/usr/share/postgresql/8.4/tsearch_data/french.stop > \  
/usr/share/postgresql/8.4/tsearch_data/ascii_french.stop
```

**ADDITIONAL\_TS\_CONFIG**

This should be a list of SQL statements, to define extra TSCs that can be used in specific fields.

### 3.3 Clean-up/filters

**CHARSET** Name of the charset to use. Note that SeSQL was only tested in utf-8. SeSQL will store all data in plain ASCII, the charset will be used for preprocessing, cleanup and conversion.

**ADDITIONAL\_CLEANUP\_FUNCTION** This function (usually a lambda, but not necessarily) will be called to process text both at indexation and search time. It can be used for example to remove html tags or convert entities back to normal letters.

**SKIP\_CONDITION** A function (or lambda) that is called on every object, is not None. If it returns a true value, the object will not be indexed. Useful, for example, to filter on workflow state.

### 3.4 Indexes and fields

**FIELDS** A list or tuple of fields (see `datamodel.txt`), including at least `classname` and `id`.

**CROSS\_INDEXES** This list contains all additional indexes to create in the database. Each index is just a list of column. Indexes that are worth creating depend of the kind of queries you do frequently.

### 3.5 Types and tables

**MASTER\_TABLE\_NAME** The name of the master table, from which all others will inherit. This table should not contain any data, but a query done to it will query all SeSQL tables.

**TYPE\_MAP** This list of (`class`, `table`) couples describes the mapping of Django classes to SeSQL tables. Django classes not present in the list will not be indexed by SeSQL. Subclasses will, by default, be sent to the same table of the superclass.

### 3.6 Query parameters

**DEFAULT\_ORDER** Default sort order for queries, when sort order is not specified. Should be a tuple of index names, with an optional `-` to indicate reverse order.

**DEFAULT\_LIMIT** The default number of items returned by a short query.

**SMART\_QUERY\_INITIAL, SMART\_QUERY\_THRESHOLD, SMART\_QUERY\_RATIO** Control of the smart query heuristic.

**QUERY\_CACHE\_MAX\_SIZE** Maximal number of long query to store in the query cache. Older queries will be discarded first. The cache is used to ensure stability of paginated results, and avoid redoing the search on very page.

**QUERY\_CACHE\_EXPIRY** Maximal time to store long queries in the cache.

### 3.7 Reindexing daemon

**DAEMON\_DEFAULT\_CHUNK** Number of elements to proceed on each iteration of the reindex daemon.

**DAEMON\_DEFAULT\_DELAY** Delay, in seconds, between two chunks.

**DAEMON\_DEFAULT\_PID** Pid file to use for the reindex daemon. The user running the daemon must have write permission to it, and the directory must exist.

### 3.8 Search history and statistics

**HISTORY\_DEFAULT\_FILTER** Queries giving less than this amount of results will be ignored in history.

**HISTORY\_ALPHA = 0.95** Erode factor for time-based decay of recent searches score. The closer to 0, the faster old searches will see their score go down, the closer to 1 the longer they'll remain with high scores.

**HISTORY\_BETA** Weight of the frequency at which the search was performed in the final score. This is on an arbitrary scale, and is only meaningful compared to the HISTORY\_GAMMA parameter.

**HISTORY\_GAMMA** Weight of the number of results given by the query in the final score.

**HISTORY\_BLACKLIST** A list of queries that will be ignored by the history feature.

### 3.9 Additional features

**ENABLE\_SESQL\_ADMIN** If set to yes, you'll be able to use `sesql:<fieldname>` in your admin options classes to search on SeSQL indexes from Django's admin. Please note that this feature relies on a monkey-patch of core Django code, and is therefore disabled by default.

## 4 Constraints

Current version of SeSQL has a few constraints :

- it requires to have a `ClassField` called `classname` and a `IntField` called `id`, referring to the object class and id ;
- since all data is converted to plain ASCII internally, it'll not work out-of-the-box with languages written in non-latin script. Patches are welcomed to handle that.

## 5 Upgrading

If you need to rebuild all SeSQL indexes (because you changed them too heavily for example) you can do :

```
./manage.py createsesqltables | ./manage.py dbshell
./manage.py sesqlreindex
```