

User Guide

Introduction	1
Spreadsheet template	1
Relational database requirements	1
Metadata requirements	1
Advices	2
Metadata tables	2
Software Use	3
Command Line Interface	3
Graphical User Interface mode	4

Introduction

Ss2db is a software that allows to convert spreadsheets organized like relational databases into sqlite relational databases. The aim of this software is to enhance data fairness by providing structured, standardized, rich metadata in a more reusable format that facilitates understanding of researchers' work.

Spreadsheet template

Different choices were made to both ensure that the relational database requirements are met and also to guarantee the value of metadata produced. For this, a template for the spreadsheet has been defined. Requirements are split in two categories, those relative to relational databases and those imposed to enhance metadata's value. A third part provides guidelines that are not mandatory but should be considered to produce more valuable documents and take full advantage of the software.

Relational database requirements

Additionally of the requirements listed in the Relational Database Good Sense Guidelines document, some rules must be respected to not break sqlite requirements.

Table and field names must contain only letters or underscores (no parenthesis, no dot, no space, no special character).

Parenthesis in table and field name break SQL statements because they are used to encapsulate fields in table and composite key's fields.

Dot break Relational DataBase Management System because to access a table in a database one syntax is the following: "*SELECT database_name.table_name*" so having a dot in database or table or column name generates error.

Each table must have a Primary Key defined.

This is not an explicit requirement of relational databases, however, each table should have a Primary key because it allows to uniquely identify each record in a table. In that, each table should have a Primary key even if it has not been defined like so.

User Guide

Metadata requirements

Foreign keys must have the same name as the keys to which they are associated.

This rule is not required by the relational database system but has been added to ensure consistency of metadata. Indeed, when metadata is produced each distinct field name is added to the metadata dictionary, this rule prevents having two different names associated with the same definition.

Advices

Choose a naming convention and follow it.

If you decide to call a table in CamelCase like TreeSpecies do not use different underscore naming conventions for other tables like Experiment_methods.

Be descriptive and try to not use abbreviations.

Try to choose names that are descriptive both for tables and columns so that anyone could understand what they contain immediately. For example, use customer for customer table instead of cust ; use customer_id instead of id.

Be careful not to use reserved words for your table and column name.

[SQLite.org - SQL reserved words](https://www.sqlite.org/lang_keywords.html)

Using those words will cause errors when querying the database.

Specify Foreign Keys whenever possible.

If two tables can be associated through one or many attributes, those attributes should be defined as Foreign keys. Indeed, this is a good practice for relational database design because it ensures data consistency and it is a good practice from a metadata quality point of view because it avoids having different names having the same definition and purpose.

Metadata tables

Several tables called metadata tables are added to sqlite to provide rich and standardized metadata.

Below list of metadata tables added with their purposes detailed.

- **metadata_term:** this table contains DataCite schema metadata terms. All mandatory terms from the DataCite must be completed by researchers while optional and recommended ones can be filled or not. This is used both to produce the documentation and enhance machine readability of metadata.
- **ddict_table:** this table contains description of tables in the database. Descriptions are provided by the researcher to enhance future reuse and understanding.
- **ddict_attribute:** this table contains description of attributes present in the database. Descriptions are provided by the researcher to enhance future reuse and understanding.
- **table_info:** this table is like an information_schema, it contains information about table names, their attributes and attributes constraints, that is if there are or not Primary or Foreign keys and potential reference table (parent table for Foreign keys). It is up to the researcher to fill those fields based on the schema he defined for the

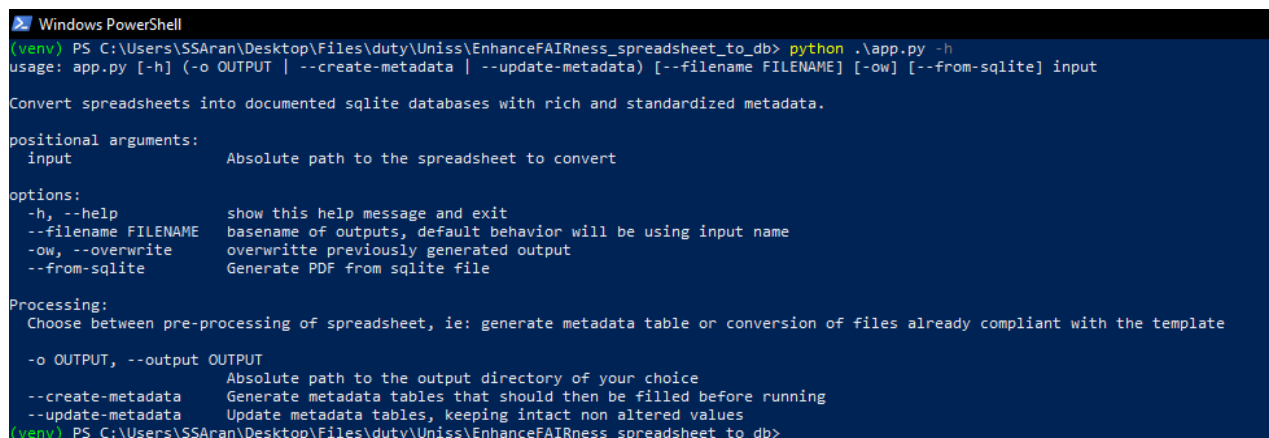
User Guide

database. Foreign keys must have the same name as their associated attributes in the parent table.

- **metadata_extra**: this table allows adding an abstract and a description for the dataset that will appear in the documentation automatically generated by the software.
- **ddict_schema**:

Software Use

Command Line Interface



```
Windows PowerShell
(venv) PS C:\Users\SSAran\Desktop\Files\duty\Uniss\EnhanceFAIRness_spreadsheet_to_db> python .\app.py -h
usage: app.py [-h] (-o OUTPUT | --create-metadata | --update-metadata) [--filename FILENAME] [-ow] [--from-sqlite] input

Convert spreadsheets into documented sqlite databases with rich and standardized metadata.

positional arguments:
  input                Absolute path to the spreadsheet to convert

options:
  -h, --help            show this help message and exit
  --filename FILENAME   basename of outputs, default behavior will be using input name
  -ow, --overwrite       overwrite previously generated output
  --from-sqlite         Generate PDF from sqlite file

Processing:
  Choose between pre-processing of spreadsheet, ie: generate metadata table or conversion of files already compliant with the template

  -o OUTPUT, --output OUTPUT
                        Absolute path to the output directory of your choice
  --create-metadata     Generate metadata tables that should then be filled before running
  --update-metadata     Update metadata tables, keeping intact non altered values
(venv) PS C:\Users\SSAran\Desktop\Files\duty\Uniss\EnhanceFAIRness_spreadsheet_to_db>
```

REPLACE IMAGE AFTER BUILD

Mandatory arguments to provide are:

- **input** corresponds to the file path to the spreadsheet to convert.
example:
/Users/username/my_folder/spreadsheet_name.xlsx on UNIX
C:\username\my_folder\spreadsheet_name.xlsx on windows
- **Processing option**: one option must be chosen
 - **output** corresponds to the folder where generated documents will be stored.
The folder must exist, it will not be created.
example:
/Users/username/my_existing_output_folder/
Example of minimal command:
`Ss2db -i /Users/username/my_folder/spreadsheet_name.xlsx -o /Users/username/my_existing_output_folder/`
 - **create-metadata** is a flag, when called metadata tables are created so that the user only has to fill it. By default, the original spreadsheet is not overwritten, except if the overwrite flag is called. The name of the output spreadsheet with metadata tables can be specified with the filename option.
Example of minimal command:
`Ss2db -i /Users/username/my_folder/spreadsheet_name.xlsx -create-metadata --ow`

User Guide

- **update-metadata** is a flag, when called the software checks for changes in table and column names and update metadata tables. The behavior is the following: unmodified items remain unchanged and their info already completed remain unchanged too ; deleted items and their info are deleted from metadata tables too ; new items are added.¹

```
Ss2db -i /Users/username/my_folder/spreadsheet_name.xlsx -update-metadata --ow
```

Optional arguments:

- **overwrite** allows to overwrite existing outputs having the name of the spreadsheet or having the same name as the specified `--filename`.

example:

```
Ss2db -i /Users/username/my_folder/spreadsheet_name.xlsx -o /Users/username/my_existing_output_folder/ -ow
```

- **filename** allows to choose output name's.

example:

```
Ss2db -i Users/username/my_folder/spreadsheet_name.xlsx -o /Users/username/my_existing_output_folder/ --filename test
```

This command will create *test.sqlite*, *test.svg* and *test.pdf* instead of *spreadsheet_name.sqlite*, etc.

- **from-sqlite** option should be used carefully, it allows to generate the pdf documentation directly from a sqlite file that still needs to respect the original template.

example:

```
Ss2db -i Users/username/my_folder/my_database.sqlite -o /Users/username/my_existing_output_folder/ --filename test --from-sqlite
```

This will generate a *test.pdf* documentation from *my_database.sqlite*.

!!WARNING!!: the overwrite flag even when set on false (default value) does not prevent from overwriting previous output in pdf format. This means that if pdf documentation has already been generated with its entity relation diagram and you run the command with "from-sqlite" set on true, even having `-ow` set on False, the already existing pdf will be overwritten.

Graphical User Interface mode

screenshots incoming

¹ Future behavior could be trying to detect similar names after changes to keep their info anyway.