

SPJRUD to SQL

Purpose

The purpose of this project is to build a simple library to convert Relational Algebra operations (SPJRUD) to SQL queries. The project presents itself as a simple python library to import in a project or in an interactive python console. See Usage for more information.

Used operations (SPJRUD)

Below a list of the operations used in this project, with an explanation of what they are.

Selection

The selection selects tuples that satisfy the given predicate from a relation.

Projection

The projection projects column(s) that satisfy a given predicate.

Join

Combine two relations into one with a cartesian product.

Rename

The rename allows us to rename column(s) in the output relation.

Union

The result of the union operation is tuples, which are present in at least one of both relation.

Difference

The result of set difference operation is tuples, which are present in one relation but are not in the second relation.

Usage

Here an example of how the project is supposed to be used in interactive mode :

Import the module and set the database.

A function is included to set a database cursor. By default, the extension of the databases files is '.db' but you can set a custom extension as a second parameter for the function.

```
>>> from spjrud_to_sql import *
>>> c = set_db('database')
```

A file called "database.db" will then be created.

Fill or create your database

If your database is empty, you can fill it with the given cursor.

```
>>> c.execute('' CREATE TABLE Rouge (A TEXT, B TEXT, C NUMERIC)'' )
>>> c.execute("INSERT INTO Rouge VALUES ('abc', 'klm', 1)")
>>> c.execute("INSERT INTO Rouge VALUES ('def', 'nop', 2)")
>>> c.execute("INSERT INTO Rouge VALUES ('hij', 'qrs', 3)")
```

Additionally, you can use the `create_table()` function to create a table based on another table which already exist by using the `spjrud` functions.

```
>>> create_table('Magenta', s(Attr('A'), Cst('abc'), Rel('Rouge')))
[('abc', 'klm', 1)]
Table Magenta successfully created.
```

Start using SPJRUD operations.

From now you can use the provided relational algebra operations. Each operations can be used with a provided function which simply is the first letter of the operation. So, for example, if you want to project the attribute A in the relation called 'Rouge':

```
>>> p([Attr('A')], Rel('Rouge'))
[('abc',), ('def',), ('hij',)]
```

It will automatically print the output of the query. If you want to get the last output, you can get it with the `Database.current.output` variable.

Operation's arguments

| Operation | Arguments |
|-----------|---------------------------------------------|
| Select | An attribute, an attribute or a constant. |
| Project | A list of attributes, a relation/subquery. |
| Join | A relation/subquery, a relation/subquery |
| Rename | An attribute, a string, a relation/subquery |
| Union | A relation/subquery, a relation/subquery |
| Diff | A relation/subquery, a relation/subquery |