**Central SQL Database of Solid Waste Management System Report**

Project participants:

Lin Lyu; Rui He; Mitchell Small

**1 Basic Projects Introduction**

This database intends to document Solid Waste Management System ( SWMS) related data around the world with a focus in the U.S. and EU for the Bee2Waste Crypto project. This database collects data of waste characterization, technologies, best management practices, and system configurations for solid wastes generated by the public sector (e.g. municipal solid waste (MSW), construction and demolition (C&D) debris, electronic wastes, disaster debris, etc.). Besides, this database attempts to capture the socioeconomic contexts as well as performance metrics of SWMS, including economic benefits, environmental footprints, and societal elements.

**1.1 Project goals**

The project goal is to build the structure of a relational database from Solid Waste Management System spreadsheets, so that we could have a central database that is useful for all partners at any model developed and achieve using SQL in Python.

**Diagram

Description automatically generated**

Figure 1.Basic sturcture of the whole database

Figure 1. represents a basic Structure of the whole SWMS relational database for the Bee2waste project.

We could see there are three main parts in the whole database: Socioeconomic System, Waste System and Performance Metrics. And the some objects in the database would impact or give feedback to other objects.

**1.2 Brief introduction of the tables and their relationships in the excel spreadsheet**

In the excel spreadsheets, table W1 is about EU List of Wastes, excluding hazardous wastes and. It has four variables, ‘code’ is the codes of all of the different waste types and is the primary key of this table. Primary key is unique for each record in table W1 and is used as a unique identifier to quickly parse data within the table. ‘description’ is the description of its corresponding code using the waste names, ‘Waste type’ is the type of the waste and etc..

Table ‘W1-W2’ shows the relationship of waste type and waste properties. And in this table code variable is the primary key and it is also a foreign key of the primary key ‘code’ in table ‘W1’, which means that the code variable in ‘W1-W2’ is a subset and references the primary key ‘code’ in table ‘W1’. The function of the foreign key provides a link between data in these two tables. It acts as a cross-reference between ‘W1-W2’ and ‘W1’ tables because it references the primary key of table ‘W1’, thereby establishing a link between them.

Table ‘W4-P’ shows the relationship of SWMS component, technology and Performance metrics (societal, environmental and financial) and here technology code is the primary key.

Table ‘W2-W4’ shows the relationship between waste properties and technology and code variable is the primary key in this table.

Table ‘W1-W4’ shows the relationship between waste types and technology and code variable is the primary key and it’s also a foreign key of the primary key ‘code’ in table ‘W1’, referencing the primary key ‘code’ in table ‘W1’.

Table ‘W3’ contains information about waste generation and W3.ID is the primary key.

Table ‘S-W3’ shows the relationship between Socioeconomic System and waste generation. W3.ID is the primary key and is also a foreign key of the primary key ‘W3.ID’ in table ‘W3’, referencing the primary key ‘W3.ID’ in table ‘W3’.

Table ‘W3-W4’ shows the relationship between waste generation and technology. W3.ID is the primary key and is also a foreign key of the primary key ‘W3.ID’ in table ‘W3’, referencing the primary key ‘W3.ID’ in table ‘W3’.

Table ‘W3-W5’ shows relationship between waste generation and waste management. W3.ID is the primary key and is also a foreign key of the primary key ‘W3.ID’ in table ‘W3’, referencing the primary key ‘W3.ID’ in table ‘W3’.

**2 Methods**

The main tool used is SQLite Software and SQLite is a relational database management system, which is really useful and easy to learn when deal with database less than 10 GB. The main language used is SQL and Python.

As for a method Overview,

1) At first do numerous data cleaning steps and format work in Excel for the whole database;

2) Use SQLite to establish a relational database from spreadsheets;

3) Use Python to map between relational database and Python code;

4) Deal with the relational database in Python; two solutions: one is using Peewee package in Python, still write python code to deal with SQL database; another is using sqlite3 package in Python, write SQL command and execute SQL command in Python.

**2.2 Steps to create the relational database and edit it**

To be more detailed about data cleaning steps, for example we need to delete the blank in the variable ‘code’, and then transfer it from text type to numeric type.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 2. Result of one table in SQLite

After finishing data cleaning steps, we could use SQLite to import the csv files and create a table in the database.

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

After creating the table, right click the table, we could modify some table attributes, such as defining primary key and etc.

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

Figure 3. UI of SQLite about how to create

Above is a series of figures about how to create tables in the database and how to modify the table after it has been created, such as naming a table name, define the variable types through modifying table.

Above is the steps to create one table in the database, we could repeat these steps to create other tables in the database and use primary key and foreign key to connect the tables.

**2.3 Ways to use it**

After finishing creating the relational database in sqlite, we have three different ways to edit and use the database.

The first one is using a package called Peewee in Python.

**2.3.1 solution one**

Graphical user interface, text, application

Description automatically generated

First of all, we need to use one command to install one package: !pip install peewee;



Then the next step is to transfer the established db to a .py file in terminal using this command: python -m pwiz -e sqlite /Users/linlyu/Desktop/Bee2waste.db > db.py

After execute the command above, we could find a produced file called db.py in the prescribed route above.

In order to deal with the database in python, we need to open the ‘translated’ database in python, part of the code shown in the following figure:

Graphical user interface, text

Description automatically generated

Followings are some easy example code about editing the db in Python:

get db table name:

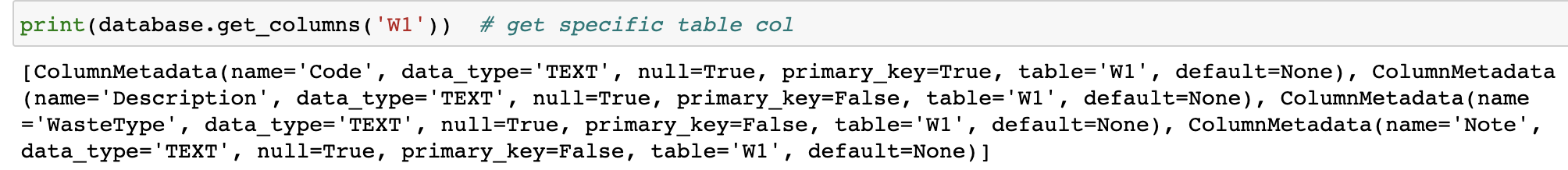
print(database.get\_tables())

Graphical user interface, text, application

Description automatically generated

get specific table col:

print(database.get\_columns('W1'))



create table W1:



Query a single data/one row:

Text

Description automatically generated

Query multiple data:

Text

Description automatically generated

Read the data in the database:

Graphical user interface, text

Description automatically generated with medium confidence

There is also a package called palyhouse in peewee, have some very useful functions, such as change table(model) to dict, etc. And for more details about code, please see in the jupyter notebook.

**2.3.2 Solution two**

The second method to edit the database is using SQL language to edit in Python. The following picture shows an example of execute SQL in python.

Text

Description automatically generated

**2.3.3 Solution three**

The third method is using sqlite to edit it. Users could just click the Bee2waste.db file created before, and the they could edit the database in sqlite. There is a section called Execute SQL where we could write sql language to edit the database. And we could also use Browse Date section to Browse data and do some basic filter or add data, delete data stuff.

More details could be seen in the demo video.

**4 Results and Future Plans**

1) Established a template relational database in SQLite;

2) Map between relational database and Python code so that we could edit the database in Python;

3) Realized three solutions to deal with SQL database in Python based on the template database;

4) Made an instructions and demo video about how to use this established relational database in Python, including some basic code about how to use peewee package and sqlite3 package to read, query and etc. the database.

As for future plans, if necessary, we could import our local database to Amazon RDS, building a cloud database, so that people could operate it at the same time.