CS 3030 Scripting Languages
Lab 5: Database Loader

Introduction

Lab 5 is an opportunity for you to automate a common database administrator activity using Python. You are provided a small file containing student registration records. Each record contains information about the student and the class in which they are registered. You are to take this information and create two tables in a database using the Python scripting language.

The input file is in Comma Separated Values format (CSV) and the output is a SQLite3 database. Python provides support for both file types using plug-in modules.

Requirements

Your Python script shall be named `~/cs3030/lab5/dbload` and be marked executable.

```
   Usage: ./dbload INPUTCSV OUTPUTDB
```

Your Python script shall read CSV file INPUTCV and create a SQLite3 database OUTPUTDB. In that database you will create two tables `classes` and `students` with the following attributes (please use these exact names and data types):

classes
- id (text)
- subjcode (text)
- coursenumber (text)
- termcode (text)

students
- id (text, primary key unique)
- lastname (text)
- firstname (text)
- major (text)
- email (text)
- city (text)
- state (text)
- zip (text)

A sample input CSV file is found at /var/classes/cs3030/lab5/studentregs.csv. The following is true about all input files you will need to process:
- The field separator character is a comma. Double quote marks are used to enclose strings with imbedded spaces.
- There is one record for each class registration and there may be more than one registration per student.
- The student information is duplicated for each class registration by that student.
- The data is complete and in its proper format so (hopefully) no error checking should be required while

reading the data.

- The first record of the CSV file is a typical CSV header record; skip over it.
- There should be no blank records in the CSV file.
- Two fields in the CSV file require explanation:
  - Use the wnumber field in the CSV file as the ID field in both tables
  - The course field in the CSV file is composed of the subject code and the course number. For example, course "CS 3030" is composed of the subject code CS and the course number 3030. Break it up using split(" ") as in this example (adapt this snippet to your code):

```
s = "CS 3030".split(" ")
# s[0] is "CS", s[1] is "3030"
```

Hints

- Suggested logic (and this is not the only way it could be done):
  - Issue an appropriate usage message and `exit(1)` if the user did not specify both the input CSV file and the output database file.
  - Open the CSV file using `try/except` and print the exception and `exit(1)` if an error occurs.
  - Open the database using `try/except` and print the exception and `exit(1)` if an error occurs
  - Drop the two tables if they exist and create the two tables.
  - For each record in the CSV file:
    - skip the header record (the first record in the CSV file)
    - attempt to retrieve the student from the students table using the wnumber
    - if the retrieval fails, insert the student record into the students table
    - insert the class record into the classes table
  - exit (0) after all records have been added to the database
- The students table uses the ID field as a primary key; attempts to add duplicate students will result in an exception in SQLite3.
- Add comments to document your logic.
- Don't forget to import modules `sqlite3`, `csv`, `random` and `os` or you will have errors galore.
- Use `/usr/bin/python` for this lab.

Run cucumber to determine your grade

- When using cucumber, the `tar` command only needs to be run once (unless I update the `cuke.tar` file and notify the class)

```
tar xvf /var/classes/cs3030/lab5/cuke.tar

./cucumber -s
```

CS 3030 Scripting Languages

Lab 5: Database Loader

Files

For this lab you will have created folder `lab5` and the following executable files:

```
dbload
```

CS 3030 Scripting Languages
Lab 5: Database Loader

Grading

Here is how you earn points for this assignment:

FEATURES	**POINTS**
Must-Have Features	
Script is named correctly and found in its proper place on icarus	5
Script is executable	5
Required Features	
Script prints a “Usage:” statement and exits rc=1 if either the INPUTCSV or OUTPUTDB files are not specified on the commandline	10
Script prints an error message containing the word “Error” and exits rc=1 if the INPUTCSV file cannot be opened	15
Script prints an error message containing the word “Error” and exits rc=1 if the OUTPUTDB file cannot be opened	15
Script exits rc=0 on successful completion	10
Script correctly defines the students table in the database	30
Script correctly defines the classes table in the database	30
Script adds the right number of students to the students table	40
Script adds the right number of classes to the classes table	40
Script add the correct student data to the students table	50
Script adds the correct classes data to the classes table	50
Grand Total	300