DS230 Data-Centric Computing **Milestone**
Project 3 **11/30/23**

OBJECTIVES

- Refine experience with programmatic database interactions.

ASSIGNMENT

I. Database Creation for Class Schedule

A. The file "Course Schedule.csv" contains the data associated with the courses that were offered in a previous semester. Open this file and study its structure.
B. Create some code that creates a database table – SCHEDULE – that is consistent with the data contained in this file.
    a. Note that some fields in the file should be segmented into multiple columns for the database.
    b. You do not need to programmatically infer the structure by reading the file (i.e. don't use the headers in your program) – it is ok for you to have a literal create query based on your observations of the data.
    c. You should also include fields for the YEAR and TERM. For now, you can put in dummy data: Year = 2021, Term = Fall.
C. Create some code that is able to read this CSV file and populate your Schedule table.
    a. You may find it beneficial to "pre-process" the CSV file. You are allowed to make any changes that you want/need to this file; however, those changes should be made programmatically. (For example, it may be beneficial to remove some the "Headings" such as "Computer Science,,,,,,," --- do not do this by hand). The idea is that this file will come to you next semester in the same format --- your program should be able to take this file and populate the DB

II. Simple queries with Schedule.

A. Provide functions that are able to ask the following queries:
    a. What courses are available from a given department?
    (department is provided by an input parameter to your function)
    b. What courses are available at a given time-block?
    (How flexible can you be with time block?)
    c. What courses are available to satisfy my Social Sciences general education requirement (ANT, ECO, POL, PSY, SOC).
        i. Generalize this to other general education requirements – e.g. Humanities, Fine Arts and Natural Sciences.
    d. What courses are available to satisfy my DCP (ends with 6, 7, 8). You should group the courses according to which category they meet. (Note - you may need to use the SQL 'LEFT', 'SUBSTRING', 'LIKE' or 'REGEX' commands)
    e. Which course are offered by my favorite professor
    (No – you can't use the literal 'Hughes' in your query).

III. Course Registration

A. Expand your database to include a STUDENT table and an ENROLLMENT table.
    a. The Student table should be defined by (STUDENT ID, Student Number, Fname, Lname, ClassYear, Major1, Major2, Minor1, Advisor)
        i. STUDENT ID is the primary key
    b. The Enrollment table should be defined by (STUDENT ID, COURSE ID, status, classSize);
        i. STUDENT ID and COURSE ID are the primary keys.
        ii. Status is either 'Active', 'WaitList', or 'Complete'
        iii. For now, fix classSize at 100 for all courses.

These tables should be created programmatically.

B. Create at least 5 students and enroll these students into 4 courses each. (They should not all have the same schedule.

IV. Anonymize and load student data

A. The file "Registration.csv" contains the data associated with the course registrations your Course Schedule. Open this file and study its structure.
B. This file contains sensitive information that you don't want to become part of your development-level database. Namely, it is associating real Student Numbers with real student names. We need to create aliases for these names and numbers --- however, you should preserve the relationships. This means that if student: (1837273, Beth, Travers) was taking (CS125, MTH125, ARTH 100 and POL 107), her alias (1872822, Carol, Ubben) should also be taking those same courses.
    a. Tip: The IDs can be a random number, the names can be drawn randomly from a collection of first names and last names.
C. After your data has been anonymized, create a function that is capable of adding this enrollment data (and student data) to the database.

V. (more) Complex queries

A. Produce code to answer 6 additional questions; at least 3 of these queries must involve multiple tables.
    a. Clearly state the question in English.
    b. Provide the code to answer your question. Most of the "heavy lifting" should be done by the SQL query (including any need to JOIN multiple tables). If you do any post-SQL python processing, you should also show the intermediate output from the SQL query.
    c. Clearly output the answer to your question.

VI. New Functionality

Now that we have a development-level database to work with, we need to explore adding some additional functionality. You should accomplish **two** of the following:

A. Add the ability to enforce pre-requisites while before enrolling in a class. Your demonstration should provide a proof-of-concept: Pick 2 or three departments, add pre-requisite information to the database and show that it is required for the student to have completed any pre-requisites before they can enroll in a new course.
B. General Education requirements recently changed (starting this fall). Instead of belonging to a prefix, you now need to complete courses in 11 different categories. These categories are outlined in the file "New Gen Ed.pdf). Add data to the database and some create functions that allows the user to query which courses are being offered that meet a given requirement (Proof of concept- show this for at least 2 of the categories).
C. Add some functionality that allows students to be added to a waitlist if the cap is full.
    a. Demonstrate this by adjusting the cap size to something reasonable (i.e. less than the default of 100) and then showing that students are added to the waitlist instead of an active enrollment.
    b. Explain how you would alter the database to move a student from the waitlist to the active status when room becomes available You do not need to implement this, but you should provide a thorough accounting of what needs to change in the database; what conditions would trigger this action; and what steps your program would take.