Week 10 - Relational Databases: SQL and SQLite

Overview

This week, we reviewed how relational databases work, using SQLite as an example. For this week's assignment, we will practice using relational databases in two ways: first, you will have to design your own schema for a database to represent the domain of music artists; secondly, you will write a Python script to use an existing database and query against it.

Make sure to create a github repository for this assignment named **IS211_Assignment10**. All development should be done in this repository.

Useful Reminders

1. Read the assignment over a few times. At least twice. It always helps to have a clear picture of the overall assignment when understanding how to build a solution.
2. Think about the problem for a while, and even try writing or drawing a solution using pencil and paper or a whiteboard.
3. Before submitting the assignment, review the "Functional Requirements" section and make sure you hit all the points. This will not guarantee a perfect score, however.

Part I - Music Database

In this part, we will model the domain of music with a relational database. In this domain, we want to model music artists, the albums they create and the songs that appear on those albums. Artists are simple to model, as we only really care about the name of the artist. Albums are also simple to model: every album has a name and an associated artist who created the album (lets keep things simple and assume that every album has one and only one artist). Songs are a little more complicated to model, as we want to record the name of the song, the associated album this song appears on, the track number of the song, and how long the song is (in seconds).

Your tasks is to think about this model, and come up with a set of relational tables that will sufficiently model what was described above. You must submit a file named *music.sql*, with SQL CREATE queries that will create the tables with the proper schema of your model.

Part II - Existing SQL Database

In this part of the assignment, we will give you the database schema, and you will be responsible for writing two Python scripts: one to load data into the database, and another to query and display data. The schema for this part is below:

```
CREATE TABLE person (
    id INTEGER PRIMARY KEY,
    first_name TEXT,
    last_name TEXT,
    age INTEGER
);
```

```
CREATE TABLE pet (
    id INTEGER PRIMARY KEY,
    name TEXT,
    breed TEXT,
    age INTEGER,
    dead INTEGER
);

CREATE TABLE person_pet (
    person_id INTEGER,
    pet_id INTEGER
);
```

Save this database to a new file called *pets.db*. Once that is done, you should do the following:

1. Create a script called *load_pets.py*, which will connect to the database in *pets.db*, and load up the data below by constructing appropriate INSERT commands
    a. **`Table`**: **`Tuple of Column Values`**
    b. `Person: (1, 'James', 'Smith', 41)`
    c. `Person: (2, 'Diana', 'Greene', 23)`
    d. `Person: (3 'Sara', 'White', 27)`
    e. `Person: (4, 'William', 'Gibson', 23)`
    f. `Pet: (1, 'Rusty', 'Dalmation', 4, 1)`
    g. `Pet: (2, 'Bella', 'Alaskan Malamute', 3, 0)`
    h. `Pet: (3, 'Max', 'Cocker Spaniel', 1, 0)`
    i. `Pet: (4, 'Rocky', 'Beagle', 7, 0)`
    j. `Pet: (5, 'Rufus', 'Cocker Spaniel', 1, 0)`
    k. `Pet: (6, 'Spot', 'Bloodhound', 2, 1)`
    l. `Person_Pet: (1, 1)`
    m. `Person_Pet: (1, 2)`
    n. `Person_Pet: (2, 3)`
    o. `Person_Pet: (2, 4)`
    p. `Person_Pet: (3, 5)`
    q. `Person_Pet: (4, 6)`
2. What is the purpose of the `person_pet` table?
3. Create a script called *query_pets.py*, which will
    a. Connect to the database in *pets.db*
    b. Ask the user for a person's ID number
    c. If that user exists:
        i. Print out data on the person (e.g. James Smith, 41 years old)
        ii. Print out all the data on that person's pets (e.g., James Smith owned Rusty, a dalmatian, that was 4 years old)
    d. Otherwise print an error message
    e. Keep doing this until the user enters in a -1, which is an indication to exit the program