**SI 206 Data-Oriented Programming**

**Project Name:** API data and Databases

**Homework Objective:**

Demonstrate ability to:

* Implement caching
* Create and modify tables in a SQLite Database
* Utilize Twitter API (including researching possible methods)
* Follow proper coding and submission conventions

**Deliverables and Submission Process:**

Submit a modified version of the files 206\_APIsAndDBs.py (provided to the student) as well as your own version of 206\_APIsAndDBs\_cache.json and 206\_APIsAndDBs.sqlite via Canvas. The python code must be executable! Code that does not run will be given a score of 0.

**Supporting Material:**

Files: 206\_APIsAndDBs.py (This code does not run, it is simply a starter for you.)

**Background:**

In this assignment you will be using the skills learned from HW7 and HW8 as well as course material to create a fully-functioning database based on Twitter data. You will need to use the Twitter API to explore the data returned and DB Browser for SQLite to view your tables.

**PART 1 - (40 pts)**

* Caching setup
* Define a function called get\_user\_tweets that gets at least 20 Tweets from a specific Twitter user's timeline and uses caching. The function should return a Python object representing the data that was retrieved from Twitter.
* Defining function get\_user\_tweets(user) that checks the cache for a particular user and returns that data and or retrieves the data using the Twitter API, caches it, and returns it.
* Create a variable called umich\_tweets and store the results of get\_user\_tweets(“@umich”)
* Accessing mentioned users correctly, slash making invocation to user method with caching to get mentioned users

**PART 2 - Creating a database and loading data USING PYTHON (90pts)**

* Creating database file properly with 2 tables, each with the correct columns Making sure both Tweets and Users table have primary keys (no duplicates)
* Updating the tables as needed when get\_user\_tweets() is invoked and saving at least 20 tweets in the Tweets table
* Saving additional user(s) (not just the user passed in the function call get\_user\_tweets) in the users table if there is at least one mentioned user, e.g any users that are being retweeted, replied to, etc.

You will be creating a database file: 206\_APIsAndDBs.sqlite. The database file should have 2 tables, and each should have the following attributes (in this exact order!)

|  |
| --- |
| **Tweets** |
| tweet\_id (PK) |
| text |
| user\_posted (FK) |
| time\_posted |
| retweets |

|  |
| --- |
| **Users** |
| user\_id (PK) |
| screen\_name |
| num\_favs |
| description |

table Tweets, with columns:

* tweet\_id (containing the string id belonging to the Tweet itself, from the data you got from Twitter) -- this column should be the PRIMARY KEY of this table
* text (containing the text of the Tweet)
* user\_posted (an ID string, referencing the Users table, see below)
* time\_posted (the time at which the tweet was created - use **DATETIME**)
* retweets (the integer representing the number of times the tweet has been retweeted)

table Users, with columns:

* user\_id (containing the string id belonging to the user, from twitter data) -- this column should be the PRIMARY KEY of this table
* screen\_name (containing the screen name of the user on Twitter)
* num\_favs (containing the number of tweets that user has favorited)
* description (text containing the description of that user on Twitter, e.g. "Lecturer IV at UMSI focusing on programming" or "I tweet about a lot of things" or "Software engineer, librarian, lover of dogs..." -- whatever it is. OK if an empty string)

***\*\*\*Part Two must be completed by November 10th for full credit in the assignment.***

**PART 3 - Processing data (60 pts)**

* Create a variable called users\_info. Write a query that retrieves all the users and save them in users\_info, which should ultimately be a list of tuples.
* Create a variable called screen\_names. Write a query that retrieves the screen names of all the users. Access all those strings, and save them in screen\_names, which should ultimately be a list of strings.
* Create a variable called retweets and store all the tweets RT'd more than TEN (10) times. retweets should be a list of tuples
* Create a variable called favorites. Write a query that retrieves the descriptions of the users who have fav'd more than 500 (500) tweets. Access all those strings, and save them in favorites, which should ultimately be a list of strings.
* Create a variable called joined\_data and then write a query to get a list of tuples with 2 elements in each tuple: the user screenname and the text of the tweet. The result should be a list of tuples.
* Create a variable called joined\_data2 and store the same data as joined\_data but sorted by descending order of the number of retweets (so the most tweeted texts are first). Do not use python to sort.

ADDITIONAL REQUIREMENTS:

* You must name your database file: 206\_APIsAndDBs.sqlite
* You must follow the database schema requirements
* You must name your cache file 206\_APIsAndDBs\_cache.json
* You must comment each function with the expected inputs and outputs as well as document any code that is "complicated"
* You may NOT share code with fellow students or post your code to Piazza. But you may discuss strategies and share links that you found helpful.

NOTE: There are tests for this project, but the tests are NOT exhaustive -- you should pass them, but ONLY passing them is not necessarily sufficient to get 100% on the project. The caching must work correctly, the queries/manipulations must follow the instructions and work properly. You can ensure they do by testing just the way we always do in class -- try stuff out, print stuff out, use the SQLite DB Browser and see if it looks ok!

**Integrity Policy:**

All materials submitted by students must be their own work - you may not submit material from previous semesters or examples taken from class or the Internet. Students may discuss the homework with others, ***but should not share code***. If you work with others, make sure to indicate their name and the nature of the collaboration. ***Any instances of cheating will receive a 0 on the assignment and one letter grade deduction in the final course grade.*** If you are unsure about the integrity of your submission, you have 48 hours after submission to withdraw your submission.

**Tips:**

Helpful functions: cur.fetchall(), join() for list comprehension

http://www.pythonforbeginners.com/basics/list-comprehensions-in-python