a. General overview of the system & a small user guide

Overview

The program we created simulates a database management system which allows the user to maintain the article database and perform some simple functions such as adding and searching for articles and authors; it also has a function that gives an overview of the database which list the most cited venues.

User guide

Part 1

This part the user must have a server running with a valid port before running the first part of the program. Upon starting the program, the user must provide the filename followed by the port of the server to establish a connection. The user must ensure they are correct.

Part 2

Upon starting the program, a prompt will show asking a user for primary actions, **search for articles, search for authors, list the venues, add an article** and **exit.** Search for articles allows the user to search for article(s) by providing one or more space separated keywords. This will return the articles that match the keywords in either the abstract, title, author, venue or year. Note each valid result must contain **all** keywords to be considered valid. Missing as little as one will unqualify the article for return. User may select from the result to view details of the article by following the prompt. Search for author will ask the user to give one keyword and simply return the author with name containing the keyword. User may select from the result to view all the articles written by the selected author. List the venues shall provide the rankings of the venues in which is sorted by the number of citations the venues receive. This function will ask the user to provide a number and the top n qualified venues will be returned. Add an article will ask the user to provide all necessary information pertaining to the article as input, user will provide one field at a time and the article will be added to the database.

b. Design

The program is divided into four major functions, as such they are organized into their own separated files, each files defines the function behavior and execution mechanism. They will each handle the user inputs and go through series of preprocessing. Afterward, it will be concatenated into a MongoDB command to prepare for execution. The documents retrieved will then again be preprocesses to eliminate unwanted columns and other information. At this stage, the ui.py will be used in which define the style of the output and what it will look like. Add an article will update the database and its indexes. It's worth noting that each entry will have a new field called "reference count" in which stores the number of times that this article

is referenced by other articles in the same collection. Functions are organized into sub directory call “subtask”.

c. Testing strategy

Database construction (Phase 1)

The script load-json.py was being tested in 2 aspects - correctness & efficiency. Correctness of database construction was verified by importing the smallest json file dblp-ref-10.json; the import result was visualized & checked using MongoDB for VScode. Efficiency of database construction was verified by importing 1M json file & timing to see if the execution is within 5 minutes. Statement coverage (cover all lines of code) was used as the coverage criteria.

Search for articles

The searching tasks was tested with the smallest 10-line database. Each matching field (abstract, year, venue, title, authors) was tested with one keyword to verify the correctness of text searching target; a mix of keywords pointing to 1 article was then performed to test the correctness of AND semantic; 1 extra word that is part (but not the full word, e.g., “gen” in “general”) of a word in the target article was then appended to the search keywords to verify the correctness of full text matching. Statement coverage was used as coverage criteria

Search for authors

The searching tasks was tested with the smallest 10-line database. A keyword that matches with one author, a keyword that matches with multiple authors, & a keyword that did not match with any author, were performed & tested to satisfy statement coverage criteria.

List the venues

Dynamically, the testing for list the venue is conducted in a small but controlled database, the initial entries in the database will a list of venues sorted by the number of citations each of them receives. Afterwards, a new entry will be added to the database. This new entry is unique as it is referenced by so many articles such that it will make whatever the venue it’s in the most cited one. As such, by comparing with the previous result, we can check if list the venue has worked as expected since the order is now subject to change. Statically, we also manually calculated the most cited venues by hand as we traverse through the database and check if the function did return the venues in the expected order.

Add an article

This part was tested with the following cases in order: non-unique id provided; unique id + title + 1 author + year; unique id + title + multiple authors + year. The added results were visualized & verified using MongoDB for VScode. Statement coverage was satisfied.

d. Group work break-down strategy

1. Jianxi is in charge of the structure of overall project as well as the searching functionalities. He planned out what the system should look like, which functions are supposed to organized inside the same. He also completed the UI designs for most part of the project & the implementation of search for articles & authors. Jianxi spent approximately 20 hours on working the project.
2. Jiemin did the phase 1 (database construction) & the implementation of add an article (with the UI of it). Jiemin spent approximately 10 hours on working the project.
3. Yihe is in charge of listing the venues and as well as its UI designs. He also wrote the design documents and created data flow diagrams when for users and artists. Yihe spent approximately 15 hours on working the project.

Communications were made through in person & online meetings and Discord group chats, works are distributed at the beginning of the project, later merged for completion using git. All three tested each other's codes.

Data flow diagrams