Article Management using SQL and Python

Anupam Kumar

December 10, 2020

Contents

1 Project Report

1.1 Project Proposal

1.1.1 Proposal Attributes

Topic Article Database System

Database MySQL

Front End Python

Project Description People often want to keep track of articles that they read, are reading, and would like to read later. They want to take notes about the articles, attach topic and tag to article, rate & review the article and check the article rating/review provided by other users. Our idea is to build an Article Database System, where users can log on to, check rating/review of articles, put articles in read/unread/reading category, attach topic/tag/notes to the article for later access.

1.1.2 Why SQL Database (MySQL)

- MySQL database can support complex queries (which would be helpful to access and provide rating, tags, topic, notes).
- MySQL database can be easily connected with Python.
- MySQL database can be hosted on multiple servers.
- Our project involves only text and numbers data, so NoSQL database is not needed.
- SQL database provides faster response than NoSQL database.

1.1.3 Description of the Application

- We want to make a website to keep track of articles that we read or want to read for self-reference later or to check rating of articles by other users if available. There will be three types of users - a registered user, a nonregistered user, and administrator. The administrator will have access to database and can create/remove additional columns/attributes. The registered user can CREATE their account for login, to build their personal database of articles, to search and READ data from the database. The registered user can UPDATE and DELETE their personal data. The administrator can UPDATE and DELETE any information from the database thus created.
- A nonregistered user can only search for corresponding info about articles.

1.1.4 Why this project

- I often read articles (often from different website resources) and want to keep track of articles I read, want to read later, or currently reading.

I also want to rate, to add topic, tag and notes to the articles for later access. Although there are various articles, I am not sure how effective it would be to be able to access rating/review of articles by other people; the project is more oriented towards personal collection of article database.

1.1.5 User Interaction

User can search for the article. User can create partition all articles into read/unread/reading category. User can add topic, tag, reading time, notes to the article. User can rate and review the article. User can query articles based on topic/tag search,based on read/unread/reading category, based on rating, and combination of these.

1.2 Project Final Report

1.2.1 README section for creating and running the project

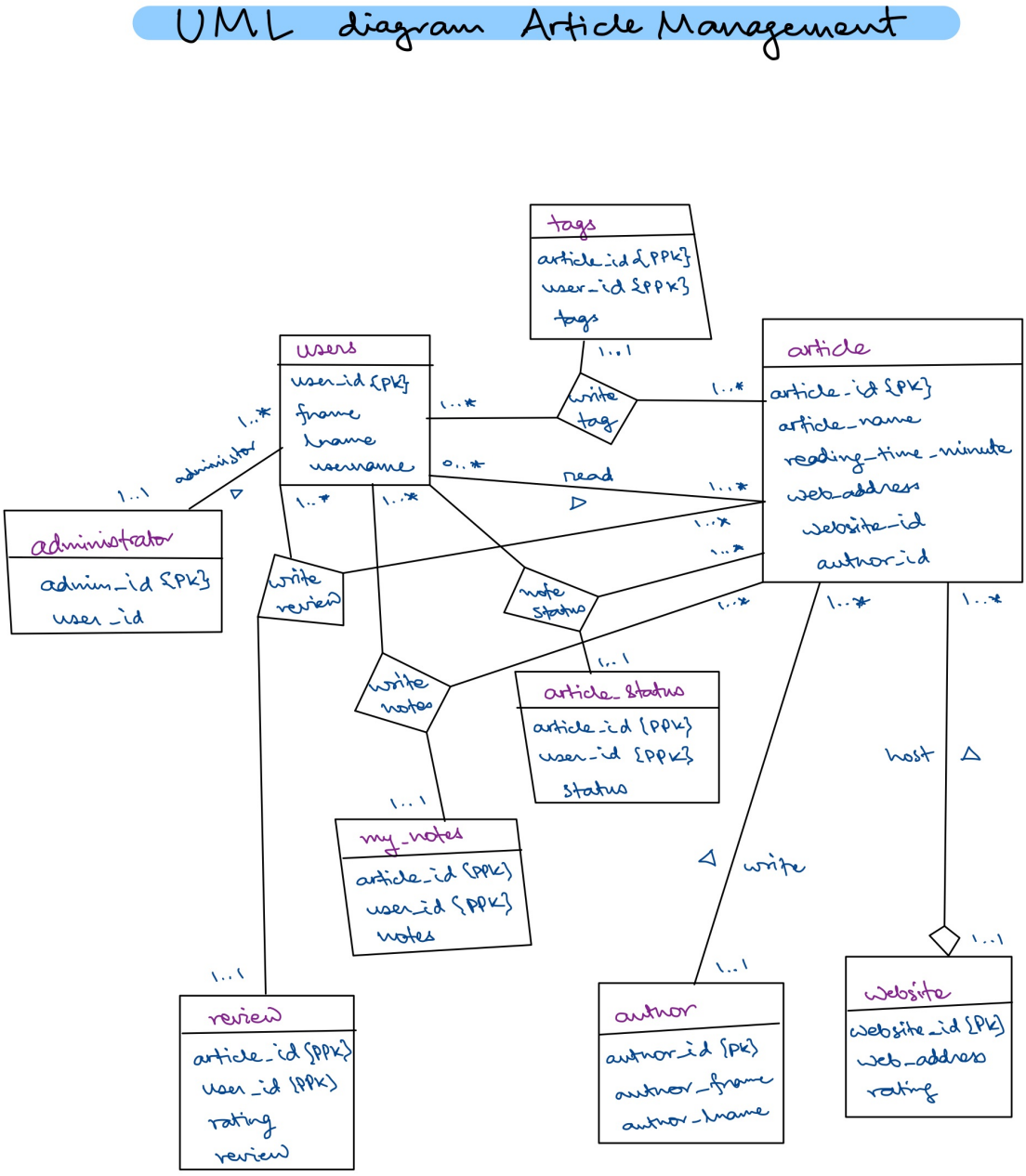
- run the file `kumar_final_project.sql` on MySQLWorkbench
  - It will create schema for database `kumar_articles`.
  - It will insert values in various tables of the generated database.
  - It will create some `Procedures` and `Functions`.
- `kumar_main.py` is the main python file, which will be needed to run on Terminal.
  - the code will be `python kumar_main.py`
  - `kumar_main.py` file is importing functions from other files: `crude_create.py`, `crud_delete.py`, `crud_update.py`, `crud_read.py` and `miscellaneous_fun.py`. The `miscellaneous_fun.py` file contains some try-except functions [to catch some errors] and `open_article_website` function [to open the website of an article].
  - running `python kumar_main.py` will give several options for executing the functions.
  - Selecting a function will provide further options.
  - Interesting procedures and queries are used to execute `"6 -> complex queries"`.
- We don't need any other software or libraries.

1.2.2 Technical Specifications for the project

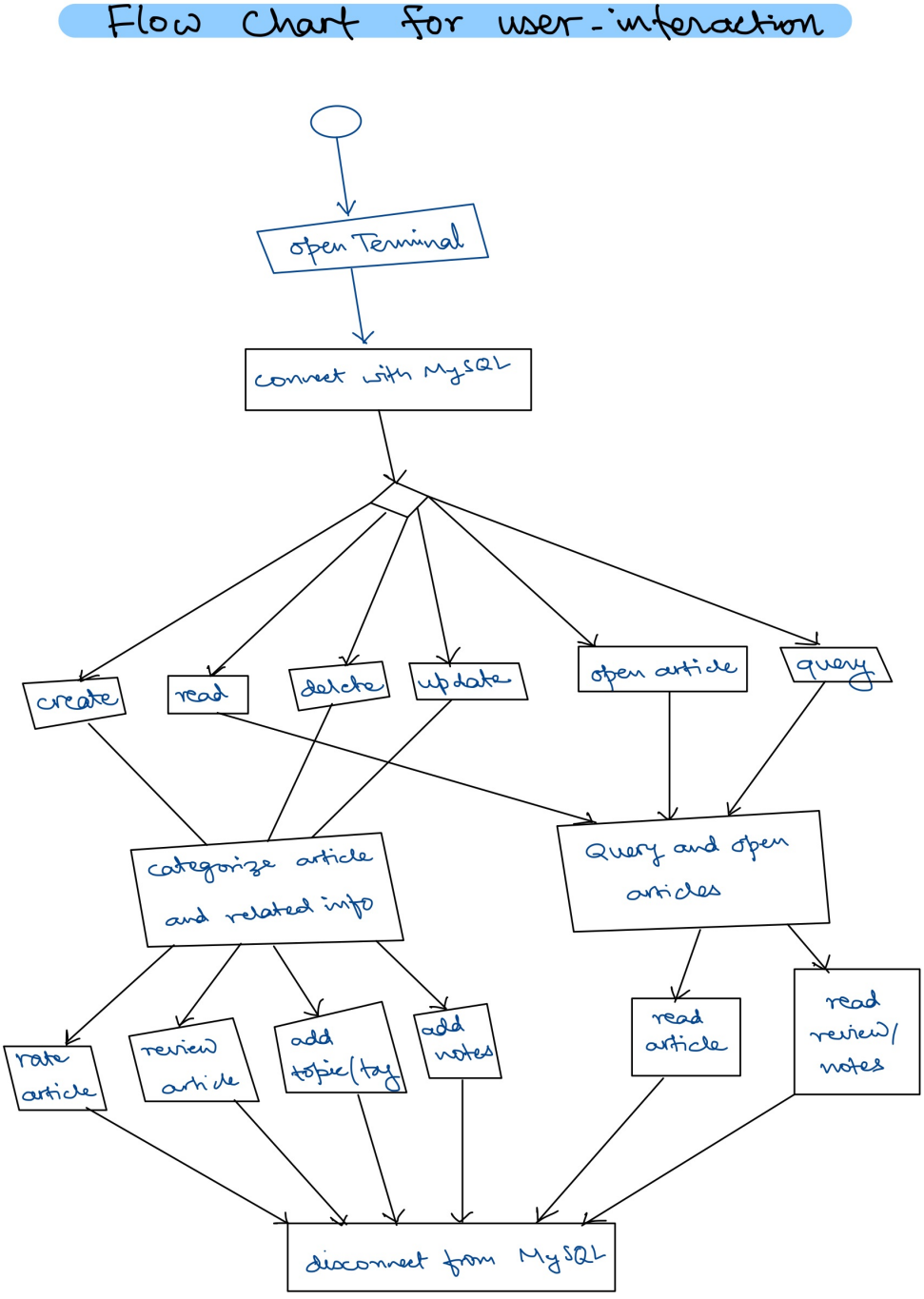
- files
  - The project contains only two types of files: *MySQL* file and *python* file.
  - *Python* file interacts with *MySQL* file to act as front end for users.
- files and functions:
  - `kumar_main.py`
    - It connects python front end to MySQL
    - calls upon other functions which interactively calls upon other functions

- `crud_create.py` contains functions
  - `insert_author`, `insert_article`, `insert_website`, `insert_article_status`, `insert_review`, `insert_my_notes`
  - Since `article` is the main functioning table for user, and user might not have or feel like entering website-id and author-id for each entry, I have deployed this function such that leaving that entry empty (by pressing *enter* at that entry) will insert `null` for that value.
- `crud_read.py` contains functions
  - `read_author`, `read_article`, `read_status`
- `crud_delete.py` contains functions
  - `delete_author`, `delete_article`, `delete_website`
- `crud_update.py` contains functions
  - `update_author`, `update_article`, `update_website`, `update_status`
- `miscellaneous_fun.py` contains functions
  - `tryexcept_int_val_stmt`, `tryexcept_val_list`, `open_article_website`
- `kumar_final_project.sql` contains following procedures, which are some complex queries (according to procedure names):
  - `articles_greater_than_reading_time`
  - `articles_less_than_reading_time`
  - `filter_articles_by_rating_status`
  - `filter_articles_by_status_readingtime_tag`

1.2.3 Current conceptual UML design and logical design as EER model



1.2.4 Final User Flow of The System



1.2.5 "Lessons Learned" section:

1. Technical expertise gained

    - learned to be able to connect python to MySQL and perform various `CRUD` and other operations and queries.
    - to apply CRUD operations and other procedures in a neatly fashion.
    - learn to add several `try and except` block for Error-Handling.
    - learned that it's difficult to manage a database in its 3rd Normal Form, as several things can break if we are not careful. I was not able to work on several of those points due to time-constraint of the project, but was able to detect some of them and realized that database maintenance is difficult, especially for industry standards.
    - Last point made me realize even more about why we studied several other topics surrounding MySQL query and about forming schema, UML notation, different normal forms, lock and timestamp, etc.

2. Insights, time management insights, data domain insights etc.

    - We need to be very careful about features, tables, data schema that we are implementing. Otherwise, it can get us into trouble of managing and changing it. E.g. in my project, I am keeping track of authors and websites for each article. Keeping track of corresponding parent website is easy but I found that keeping track of authors is difficult since there are so many authors, and we might not need that information.
    - We need good data domain insights especially in case resources are limited.
    - We should focus on objectives and extensibility to decide for data domain, schema and most importantly front end experience.
    - Time management was definitely a constraint. User front could have been made more user-friendly for each function by providing relevant data and error-handling. I implemented it for some but it could have been extended.

3. Realized or contemplated alternative design / approaches to the project

- I wanted to implement website version but figured out it will take a lot of work, since I will have to learn HTML, CSS, Javascript, Node.js, php, etc.
- I should have restricted to fewer number of tables, and probably should have removed authors table. It's making insertion into article table a bit difficult.

4. Documentation of any code not working

   - I think all codes should be working.
   - I have tried to exclude or not implement codes which were not working.
   - I couldn't add many values in `my_notes` table.
   - I also couldn't implement differentiation between admin and user in my project.

1.2.6 "Future work" section:

1. Planned uses of the database

   - My motivation for this project was to keep track of several articles and information around those articles.
   - I think I have met my goals and have some rudimentary version of front end that in theory can work for my need.
   - However, the implementation is at rudimentary level and can not be functionally in its current form. It's not polished enough and maintaining it will take a lot of work.
   - I would like to implement it as a website front end. That will be useful as data can be displayed in a format which would be better accessible and will increase readability.

2. Potential areas for added functionality

   - To implement/differentiate admin and user level access and functionality.
   - Output query in more readable format.
   - Just more function for CRUD operations.
   - To add more `procedures`, `functions`, `triggers` and `events`.