Mini-Project Report

Zhengyao Zhang, Bingshen Yang, Siyuan Wu

A) General overview of system:

- The interface of the mini project is based on the functionality of python3, berkeley db and the interaction between these two.

- The interface itself relies on user input for the interaction between user and the program.

- For performing certain actions, try to follow the guidance as much as possible. If the inputs from users is not the acceptable form, the interface will ask for the same input until correct from of input in given.

- Small user guid:
  (1) Run the script file with ‘./buildIndex.sh’
  Here we are assuming the name of the four input file strictly follows project specification
  (2) Run the py file with ‘python3 project2.py’
  (3) Input the query
  E.g “rterm:great sound%”
  (4) Enter ‘Q’ or ‘q’ for exiting the program

B) Algorithm Efficiency:

The design of the source code can be divided into serval parts

- Formulating and evaluating queries:

Function ‘start_query’, ‘formatQuery’ are responsible for formulating and evaluating queries, which means these two functions will recognize the requirement of input query and call related function after finish analysis.

- Implementation of query functionalities:

Function ‘termQuery’, ‘termQuery_start’, ‘dateQuery’, ‘priceQuery’ and ‘scoreQuery’ are corresponding defined functions for implementing the actual functionalities of input queries.

- Retrieve data from data file

Function ‘get_full_data’ and ‘get_brief_data’ are responsible for retrieving related data from data file after the query is evaluated.
‘get_full_data’ is for the user who want to see all the information, while ‘get_brief_data’ is for the user who just want to see the brief information.

- Function ‘main’ is used for combining all the functionalities together.

The idea of the algorithm is that we always check which index file we will use for evaluating each condition of the query, which means we can get the index (review id) for each condition of the query, and then we intersect the result indexes for each condition before we finally retrieving data from reviews.txt.

Overall, we are trying to make the algorithm as efficient as possible by using evaluating the query separately based on its several conditions and use set operations at the end to combine them together

(C) Testing Strategy

We used data from eclass for tasting and completing our source code, and the data is only used for testing purpose, we have **NEVER** repost the data or use it for any bad purpose.

The testing strategy here is try to run as many query as we can, just in order to cover as much edge cases as possible. Overall, our testing strategy is just using as much query as we can to improve our program performance.

At this point, we did not find any existing bugs.

(D) Group work beak-down strategy

This strategy of breaking down work is based on the functions

1. Zhengyao Zhang (about 21 hours):
   a. main b. termQuery c. termQuery_start d. part of start_query

- Zhengyao Zhang is mainly responsible for the functionality of main function and term query.

2. Bingshen Yang (about 27 hours):
   a. dateQuery b. priceQuery c. scoreQuery

- Bingshen Yang is mainly responsible for the functionality of three range type of range queries.

3. Siyuan Wu (about 24 hours):
   a. start_query b. formatQuery c. get_full_data d. get_brief_data

- Siyuan Wu is mainly responsible for formatting queries, that is, he worked for formulating queries before we evaluate them.

- Coordination: After completing all the sub tasks, we combine our code together (using main function) and we did the testing and debugging together.

All functionalities are implemented based on the specifications and requirements of the mini project. We did not make any changes to the requirement.