Project 1

Due: 5PM 20 October 2020 (Tuesday)

1. **Introduction.** This project is to design and implement the following two components of a database management system, storage and indexing.
   (1) For the storage component, the following settings are assumed.
   - a fraction of main memory is allocated to be used as disk storage for simplicity and the disk capacity could be 100 - 500 MB (depending on your machine's main memory configuration);
   - the block size is 100 B;

   (2) For the indexing component, the following settings are assumed.
   - a B+ tree is used;
   - the B+ tree should follow the definitions introduced in the lectures;

2. **Implementation and Experiments.**
   (1) Design and implement the storage and indexing components based on the settings described in Part 1. C/C++ is recommended for this project, but other programming languages including Java and C# are also acceptable.
   (2) Experiment 1: store the data (which is about IMDb movives and described in Part 4) on the disk and report the following statistics:
   - the number of blocks;
   - the size of database;

   (3) Experiment 2: build a B+ tree on the attribute "averageRating" by inserting the records sequentially and report the following statistics:
   - the parameter n of the B+ tree;
   - the number of nodes of the B+ tree;
   - the height of the B+ tree, i.e., the number of levels of the B+ tree;
   - the root node and its child nodes (actual content);

   (4) Experiment 3: retrieve the attribute "tconst" of those movies with the "averageRating" equal to 8 and report the following statistics:
   - the number and the content of index nodes the process accesses;
   - the number and the content of data blocks the process accesses;
   - the attribute "tconst" of the records that are returned;

   (5) Experiment 4: retrieve the attribute "tconst" of those movies with the attribute "averageRating" from 7 to 9, both inclusively and report the following statistics:
   - the number and the content of index nodes the process accesses;
   - the number and the content of data blocks the process accesses;
   - the attribute "tconst" of the records that are returned;

(6) Experiment 5: delete those movies with the attribute "averageRating" equal to 7, update the B+ tree accordingly, and report the following statistics:
- the number of times that a node is deleted (or two nodes are merged) during the process of the updating the B+ tree;
- the number nodes of the updated B+ tree;
- the height of the updated B+ tree;
- the root node and its child nodes of the updated B+ tree;

(7) Re-set the block size to be 500 B and re-do Experiment 1, 2, 3, 4, and 5.

3. Materials to submit including:

A report including:

(1) Design of the storage component, including: how each data item is stored as a field, how fields are packed into a record, and how records are packed into a block. It is suggested to use some figures to illustrate the designs and include the size information of fields and records.

(2) Design of the B+ tree component, including the data structure of a node and the maximum number of keys a node maintains.

(3) Results of the experiments in Part 2;

(4) The contribution of each group member; and

Source code (You must attach an installation guide to ensure that your code can be run successfully. You will not receive any credit if your code fails to execute.)

4. Data. The data contains the IMDb rating and votes information for movies
- tconst (string) - alphanumeric unique identifier of the title
- averageRating – weighted average of all the individual user ratings
- numVotes - number of votes the title has received

The first line in each file contains headers that describe what is in each column. The data could be downloaded via this link:
https://www.dropbox.com/s/c04kfatnd9lrtx9/data.tsv?dl=0

5. Submission policy.

(1) All submissions should be uploaded to NTULearn (a submission slot shall be created later on).

(2) Late submissions will be penalized by 5% deduction per day for at most 7 days. Beyond 7 days after the deadline, no submissions will be accepted.

(3) It is not allowed to copy or refer to public code repositories. Strict plagiarism will be conducted. Any found plagiarism will mean a failing grade and be subject to further disciplinary actions. Some groups may be asked to demonstrate/explain their codes.