File Structure: Assignment #2

100 points

Cairo University, Faculty of Computers and Information

Notes:

1. *Cheaters will be graded by –ve points , Don't copy any code from anywhere ..*
2. *Submit your code to through Acadox only.*
3. *Submit one compressed file with you ID and Group Name*
4. *Due Date **17/4/2018 10:30 PM***
5. *Team = max 2 students, team must be from the same lab*

We want to store data about **books** and **authors**. (for simplicity consider any book has only one author, but any author may have more than one book, i.e. author has at least one book.)

Book attributes
Char [13]: Book_ID //primary key
Char [30]: Author_ID //sec. key
Char[50]: Book _Title
Float : Book_Price

author attributes
Char [30]: Author_ID //primary key
Char[50]: Author_Name //sec. key
Char[50]: Author_Address
Char[11]: Author_Mobile

- Consider we want to save 10 books and 10 authors.
- Save the data for books and authors in the following format: **delimited fields, length indicator records.**
- You should develop the following indexes
  1. **Primary index using the Book_ID (for Books datafile)**
  2. **Primary index using the Author_ID (for Authors datafile)**
  3. **Secondary index using Author_ID (for Books datafile) //Author_ID is sec. key in Books datafile**
  4. **Secondary index using Name (for Authors datafile)**
- The user can write a **query** that contains **fixed** key words (formatted in **red** below)
- Examples for queries that user can write
  - **select all from Books where Author_ID = 'xxxx'** // this query will use sec. index to get results
  - **select all from Authors where Author_ID = 'xxxx'** // this query will use primary. index to get results
  - **select Book_Tile from Books where Book_ID = 'xxxx'** // this query will use sec. index to get results
  - **select all from Books and Authors** // (check snapshot below as an example)

Books

Book_ID	Author_ID	Book_Title	Book_Price
123456789	1	C#	100
321456987	2	Java	150
565587954	3	C++	130
569955665	1	Python	200

Authors

Author_ID	Author_Name	Author_Address	Author_Mobile
1	Ahmed Aly	Giza	01001111111
2	Moh Sayed	Cairo	01001111112
3	Aly Samt	Alex	01001111113

Author_ID	Author_Name	Author_Address	Author_Mobile	Book_ID	Book_Title	Book_Price
1	Ahmed Aly	Giza	01001111111	123456789	C#	100
1	Ahmed Aly	Giza	01001111111	569955665	Python	200
2	Moh Sayed	Cairo	01001111112	321456987	Java	150
3	Aly Samt	Alex	01001111113	565587954	C++	130

select all from Books and Authors

File Structure: Assignment #2

100 points

Cairo University, Faculty of Computers and Information

the main welcome screen is below.

1) Add New Book
2) Add New Author
3) Delete Book (ID)
4) Delete Author (ID)
5) Print Book(ID)
6) Print Book(Author_ID)
7) Print Author(ID)
8) Print Author(name)
9) Write a Query
10) Exit

Important notes:

- All indexes are sorted ascending
- *No need to use a status flag to check that indexes are up-to-date.*
- *But, you MUST implement secondary indexes using* ***inverted list*** *technique.*
- Searching in indexes is performed using ***binary search***.
- To delete a record just put an * in the beginning of that record. (no need for avail list implementation)
- All operations (add, delete) will affect indexes *as explained in lecture 13*.
- Search operations will use indexes (primary or secondary)
- Bind all secondary indexes with the primary index, don't bind them by addresses directly.

Assume any other information you need.