

Database Systems Lab

SESSION 5

Library Management System with two entities : Book and Student

In this lab session, you will be working with two entities : BOOK and STUDENT. The new functionality being added is ISSUE().

Here, you are expected to implement a issue functionality with an assumption that outnumbered copies of each book are available.

The structure of the code is as follows :

libsys:

- libsys create [create files for book,student,issue]
- libsys open [individual function call to open : book,student,issue]
- libsys close [individual function call to close : book,student,issue]

book:

- get book
- put book
- delete book

student:

- get student
- put student

Complete the following tasks:

```
-----create()-----
int libsys_create( char *repo_name1,char *repo_name2,char *repo_name3);

    // create booksys file pointer to open a file in 'wb' mode
    // handle file pointer error if value is NULL return appropriate error code referring to the error codes
    // close file pointer

    // create studsys file pointer to open a file in 'wb' mode
    // handle file pointer error if value is NULL return appropriate error code referring to the error codes
    // close file pointer

    // create issuesys file pointer to open a file in 'wb' mode
    // handle file pointer error if value is NULL return appropriate error code referring to the error codes
```

```

// close file pointer

// Open the index file in "wb" mode
// Initialize index file by store "0" to indicate there are zero entries in
index file

// return success

```

```

-----open()-----
int libsys_open(char *book_name, char *stud_name, char *issue_name);

//call booksys_open()
//call studsys_open()
//call issuesys_open()

// Open the index file in rb+ mode
// Read number of index entries from index file
// Load index_entries array by reading index entries from the index
file
// Close only the index file

//check status of above functions
// if all of them return success then return SUCCESS else return ER-
ROR

```

```

int booksys_open( char *repo_name );

//1. assign repo handle a file pointer by opening file in 'rb+' mode
//2. handle file pointer error if value is NULL return appropriate
error code referring to the error codes in libsys.h
//3. assign values (repo_name) to booksys_repo_handle
//4. assign value to
repo_handle.book_repo_status=LIB_REPO_OPEN;
//5. return booksys_success

```

```

int issuesys_open( char *repo_name );

//1. assign repo handle a file pointer by opening file in 'rb+' mode
//2. handle file pointer error if value is NULL return appropriate
error code referring to the error codes in libsys.h
//3. assign values (repo_name) to issuesys_repo_handle
//4. assign value to
repo_handle.issue_repo_status=LIB_REPO_OPEN;
//5. return issuesys_success

```

```

int studsys_open( char *repo_name );

```

```

        //1. assign repo handle a file pointer by opening file in 'rb+' mode
        //2. handle file pointer error if value is NULL return appropriate
error code referring to the error codes
        //3. assign values (repo_name) to studsys_repo_handle
            //4. assign value to
repo_handle.stud_repo_status=LIB_REPO_OPEN;
        //5. return studsys_success

```

-----book:NO change -----

```

int get_book_by_isbn( int key, struct Book *rec )
{

// get_rec_by_key
//check repo status
// Search for index entry in index_entries array
//-----use flag to read valid entries
// Seek to the file location based on offset in index entry
// Read the key at the current file location
// Read the record after reading the key

}

int put_book_by_isbn()

//-----check index file for key
//-----if key already present check for flag [flag=1 : entry is valid]
//-----if key is present but flag is 0 then just add entry at same index
i.e update only offset and update flag; return status
//-----if key is present but flag is 1 return failure as data is already
present
//-----if key is not prsent then proceed with following steps:

// Seek to the end of the data file
// Create an index entry with the current data file location using ftell
// Add index entry to array using offset returned by ftell
// Write the key at the current data file location
// Write the record after writing the key
// return status

}

```

```

int delete_book_by_isbn( int key )
{
//-----delete_rec_by_key
//-----check repo status
//-----Search for index entry in index_entries array
//-----if key matches and flag is 1 then reset flag
//-----if key matches but flag is already reset return status

```

```
//-----if key doesn't match then return status
```

```
}
```

```
-----student: additional-----  
int put_student_by_rollno( int rollno_to_write, struct Student *rec );
```

```
    //1. check if repo status is closed then return return appropriate  
error code referring to the error codes  
    //2. else continue with following action sequence  
    //3. set the file pointer to end  
    //4. write rollno_to_write  
    //5. write Student record  
    //6. if both actions are successful then return studsys_success  
    //7. else return studsys_add_failed
```

```
int get_student_by_rollno( int rollno_to_read, struct Student *rec );
```

```
    //1. check if repo status is closed then return appropriate error  
code referring to the error codes  
    //2. else continue with following action sequence  
  
    //3.1 read rollno  
    //3.2 check if rollno is equal to the rollno_to_read  
    //3.3 if yes then read entire record of a Student and return  
studsys_success  
    //3.4 else skip the record and read next rollno of the Student  
    //4. Repeat step 4.1 to 4.4 till end of file  
    //5. Return record not found : appropriate error code referring to  
the error codes
```

```
-----issue()-----  
int issue(int rollno, int isbn);
```

```
    // check if book repo status is closed then return return appropriate  
error code referring to the error codes  
    // else continue with following action sequence
```

```
    // check if student repo status is closed then return return appro-  
priate error code referring to the error codes  
    // else continue with following action sequence
```

```
    // check if issue repo status is closed then return return appropri-  
ate error code referring to the error codes  
    // else continue with following action sequence
```

```

//declare student and book variables

//get book by isbn and store status in status1
//get student by rollno and store status in status1

// if status1 and status2 are successful then continue with following
action sequence else return error

// create Issue object and assign rollno and isbn
// set the file pointer to end
// write issue record

// if both actions are successful then return success
// else return failed

-----close()-----
int libsys_close();

//call booksys_close()
//call studsys_close()
//call issuesys_close()

// Open the index file in wb mode (write mode, not append mode)
// Write number of index entries at the beginning of index file
// Unload the index array into the index file (overwrite the entire
index file)
// Close the index file and data file

//check status of above functions
// if all of them return success then return SUCCESS else return ER-
ROR

int booksys_close();

//1. check if repo status is closed then return appropriate error
code referring to the error codes in libsys.h
//2. else continue with following action sequence
//3. close file pointer
//4. set booksys_name as ""
//5. set book_repo_status=LIB_REPO_CLOSED
//6. return LIB_SUCCESS;

```

```
int studsys_close();
```

```
    //1. check if repo status is closed then return appropriate error  
code referring to the error codes  
    //2. else continue with following action sequence  
    //3. close file pointer  
    //4. set studsys_name as ""  
    //5. set stud_repo_status=LIB_REPO_CLOSED  
    //6. return LIB_SUCCESS;
```

```
int issuesys_close();
```

```
    //1. check if repo status is closed then return appropriate error  
code referring to the error codes  
    //2. else continue with following action sequence  
    //3. close file pointer  
    //4. set issuesys_name as ""  
    //5. set issue_repo_status=LIB_REPO_CLOSED  
    //6. return LIB_SUCCESS;
```

Testing

Two testing programs are given to you.

- a. First test with driver.c

Submission

YOU ARE NOT EXPECTED CHANGE ANY OF THE FILES GIVEN TO YOU. Upload only rollno_lab5.c to LMS.