Database Systems Lab

SESSION 5

DELETE Operation

In this lab session, you will implement the DELETE operation to delete existing record in the data file.

Complete the following tasks:

Modify the PDS function as per the following:

A) rollno_pds.c Changes

```
int pds_create(char *repo_name)
// No Change

int pds_open(char* repo_name, int rec_size)
// No Change

int pds_load_ndx()
// No Change

int put_rec_by_key(int key, void*rec)
{
  // Seek to the end of the data file
  // Create an index entry with the current data file location using
ftell
  // (NEW) ENSURE is_deleted is set to 0 when creating index entry
  // Add index entry to BST using offset returned by ftell
  // Write the key at the current data file location
  // Write the record after writing the key
}

int get_rec_by_ndx_key(int key, void*rec)
{
  // Search for index entry in BST
  // (NEW) Check if the entry is deleted and if it is deleted, re-
turn PDS_REC_NOT_FOUND
  // 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 pds_close()
{
```

```
// Open the index file in wb mode (write mode, not append mode)
// Unload the BST into the index file by traversing it in PRE-ORDER (overwrite the entire index file)
// (NEW) Ignore the index entries that have already been deleted.
// Free the BST by calling bst_destroy()
// Close the index file and data file

}

int get_rec_by_non_ndx_key(void *key, void *rec, int (*matcher)(void *rec, void *key), int *io_count)
{
  // Seek to beginning of file
  // Perform a table scan - iterate over all the records
  //   Read the key and the record
  //   Increment io_count by 1 to reflect count no. of records read
  //   Use the function in function pointer to compare the record with required key
  //   (NEW) Check the entry of the record in the BST and see if it is deleted. If so, return PDS_REC_NOT_FOUND
  // Return success when record is found
}

int delete_rec_by_ndx_key( int key) // New Function
{
  // Search for the record in the BST using the key
  // If record not found, return PDS_DELETE_FAILED
  // If record is found, check if it has already been deleted, if so return PDS_DELETE_FAILED
  // Else, set the record to deleted and return PDS_SUCCESS
}
```

B) rollno_contact.c Changes

Add the following functions to contact.c

```
int search_contact_by_phone( struct Contact *c, char *phone );
// No Change

int match_contact_phone( struct Contact *c, char *phone );
// No Change

// Function to delete a record based on ndx_key
int delete_contact ( int contact_id )
{
     // Call the delete_contact_ndx_key function
     // Return CONTACT_SUCCESS or CONTACT_FAILURE based on status of above call
}
```

Testing

a. The following driver program is given to you:
    - pds_tester.c (generic testing with input data file like testcase.in)

b. Test your program thoroughly with the above driver program by creating your own test input files

Commands

- Use the following command for creating pds_tester executable:

```
gcc -o pds_tester bst.c rollno_contact.c roll_pds.c pds_tester.c
```

For testing using pds_tester, use the following command:

```
pds_tester testcase.in
```

Submission

Upload the following to LMS as a zipped file (IMTXXXXXXX.zip):

- rollno_pds.c
- rollno_contact.c

Make sure you only use with the bst.c provided to you.

YOU ARE NOT EXPECTED CHANGE ANY OF THE FILES GIVEN TO YOU