Database Programming Project

Project 1: Slotted Page Structure

Due: **May 27. 5 pm**

1. Introduction

In this programming project, you will implement a slotted page structure that contains the records of the form (variable string key (type char*), and 8byte integer value). You will implement three functions, insert(key, val), find(key), and is_full(sizeof(record)).

Please note that this programming project does not consider the case of overflow (i.e., the case when the page is full), but the functions that are implemented in this project will be used for the next project.

이번 프로젝트에서는 slotted page structure를 구현합니다. slotted page structure는 variable length를 가지는 데이터를 저장하는데 적합한 자료구조로 널리 사용되고 있는 페이지 자료구조 중 하나입니다. 본 프로젝트에서 구현하는 slotted page structure는 16byte record size, variable length key, 그리고 8byte integer value를 가지는 record를 포함합니다. 이번 프로젝트에서는 insert, find, 그리고 is_full 함수를 구현합니다.

이번 프로젝트에서는 페이지가 가득 찬 경우를 체크하거나, 그에 대한 고려를 하지 않지만, 다음 프로젝트에서 이번에 작성한 코드가 사용 된다는 점을 유의하시기 바랍니다.

2. Getting Started.

Please unzip the source code of this project on your Linux virtual machine.

Makefile: A simple Makefile for you to compile the project. To compile your project, type “make page”. If you want to delete object files, you can delete them by typing “make clean”.

slot_header.hpp/slot_header.cpp: In the codes, you can find the definition and implementations of the class for the slot_header structure. Please do not modify these codes.

page.hpp/page.cpp: You will implement insert(), find(), and is_full fuction() in page.cpp. Please do not modify page.hpp file.

main_slotted_page.cpp: This file includes the main function that you can test your implementation.

If you want to test, you can execute ./page file after compiling the source codes.

코드를 받으신 후, 리눅스 VM 등에 소스코드의 압축을 푸시면 다음과 같은 파일들을 보실 수 있습니다.

Makefile: 컴파일 하실 때 사용하실 파일입니다. 이번 프로젝트를 위해서는 “make page”라고 명령어를 치시면 됩니다. 만약 중간에 생성되는 object 파일을 지우시고 싶으실 경우, “make clean” 명령어를 치시면 됩니다.

slot_header.hpp/slot_header.cpp: 이 파일에서는 slotted_header와 관련된 구현과 자료구조 정의등을 확인하실 수 있습니다. 이 파일을 고치지 마세요.

page.hpp/page.cpp: page.cpp내의 insert, find, is_full함수를 이번 프로젝트에서 구현하시게 됩니다. page.hpp파일을 수정하지 마세요.

main_slotted_page.cpp: 이 파일에는 main 함수가 포함되어 있습니다. 따라서, 이 파일을 사용해서 이번 프로젝트에서 작성한 코드를 테스트할 수 있습니다.

구현한 부분을 테스트 해보고 싶은 경우, 컴파일 후 생성되는 ./page 파일을 실행하시면 됩니다.

3. Design Overview

The slotted page structure is a widely used page structure for managing variable-length records. When you insert or find a record from the slotted page structure, you should first traverse sorted offset_array, and get an offset of the records. After getting offset, you should get an address for the record to read the key and value.

The records stored in the slotted page structure consist of 2bytes record size, variable-length key, and 8bytes value.

The following functions are provided (Please do not modify them).

- constructor

There is a default constructor for this class

-print()

The function prints the information of the slotted page structure, such as the number of data, offset array, and records.

- put2byte(dest,data), get2byte(dest).

These functions are used for storing and getting 2bytes of information from the page structure. You should provide the address where you want to store or retrieve the data. In the put2byte function, you also need to provide the data to store.

- get_record_size(record_addr), get_key(record_addr), get_val(key_addr)

Each record contains size_of_record, key, and value. You can use these functions to get the information of each record.

The following functions that you will be responsible for implementing.

- insert(key, val)
In this function, add the record (record size, key and value pair) to the slotted page structure. You also need to manage the offset array in this function.

- find(key)
Implement the codes for fiding the key from the given slotted structure. If the key does not exist, then return 0.

- is_full(record_size)
If the slotted_page is full, then return true. If it is not, then return false.

Slotted page 구조는 variable length record를 관리하기 위해 널리 사용되는 자료구조 중 하나입니다. record를 slotted page structure에 넣는 경우, 먼저 정렬된 offset_array를 통해, offset 정보를 얻은 뒤, offset 정보를 활용하여 실제 레코드의 주소 값을 구해야 합니다. 주소 값을 얻은 뒤에, record_size, variable-length key 그리고 8bytes value를 구할 수 있습니다.

다음의 함수는 제공되는 함수들입니다. 고치지 마세요.
- constructor
이 클래스의 기본적인 생성자입니다.

- print()
이 함수는 slotted page structure 내부 정보를 출력할 수 있는 print 함수입니다.

- put2byte(dest,data), get2byte(dest).
이 함수들은 slotted page structure 내부에, 2bytes 데이터를 저장하거나, 데이터를 구할 때 쓰입니다.

- get_record_size(record_addr), get_key(record_addr), get_val(key_addr)
이 함수들은 레코드들의 정보를 구하는 데 쓰입니다. 앞의 2개의 함수는 레코드의 주소 값을 활용하여, 레코드의 크기와 키를 구하고, 마지막 함수는 키의 주소를 통해 밸류의 값을 구합니다.

아래의 함수는 구현 해야하는 함수들입니다.

- insert(key, val)
이 함수에서는 제공된 key와 value, 그리고 이들의 크기를 slotted page structure에 저장해야 합니다. 또한, 이 경우, offset_array 도 이들 정보를 포함하고 정렬되어 있어야 합니다.

- find(key)
이 함수에서는 주어진 slotted page 내에서 key를 찾습니다. key가 있는 경우 연관된 value를 반환하고, 없는 경우 0을 반환합니다.

- is_full(record_size)
이 함수에서는 slotted page구조 내에 현재 충분한 공간이 있는지 확인합니다.

4. Submission Guidelines

Please submit your page.cpp file to the e-Campus.
If you have any questions, please upload your question to the e-campus.

page.cpp파일을 e-Campus를 통해 제출해주시고, 질문 사항 또한 e-Campus를 통해 올려주시기 바랍니다.