Design Document

By:

Jacob Hopkins

Misky Abshir

Tyler Willard

Anuska Shrestha

Sequence Set Group Project

Objective: Generate a sequence set as a file composed of (doubly) linked blocks of fixed-sized records comprising fixed-sized fields.

Introduction

The objective of this project is to create a sequence set class for the text file provided. This text file includes strings named: Zip Code, Place Name, State, County, Latitude, and Longitude. We placed these fields in records, followed by placing them into b-trees.

This sequence class will store the data we are given efficiently, along with adding a method to verify the records stored. We will also display records, allowing user input to get specific records from the data, along with modifying fields, inserting, and deleting records.

**Sequence Class Design**

**Methods**

**Utility Methods**

Split\_string

· This function will take apart a string and split it by some char delimiter

· Std::vector<std::string>

· std::string str, char delimiter

string\_to\_vector

· This function will put a string into a vector

· std::vector<char>

· std::string s, int n

add\_c\_to\_a\_til\_size\_of\_b

· Check to see if records are spaced correctly

· std::string

· std::string a,std::string b,std::string c, bool front = true

**Class Methods**

SequenceSet::SequenceSet

· This is the default constructor

· Uses default values for our particular case

· Block size: 512

· 512 records in each block

· Default cap 0.5 meaning we have 50% fill or 256 records

· Primary key index is 0 – zipcode

· First – block null

· End of file header

· Record size of 1 record because we use strings

· In\_filename: "us\_postal\_codes\_formatted.txt"

· Out\_filename: "us\_postal\_codes\_sequence\_set\_file.txt"

SequenceSet::SequenceSet

SequenceSet(int b\_size, int r\_size, float d\_cap, std::string i\_filename, std::string o\_filename); //! copy constructor

· The constructor for the SequenceSet that takes in inputto assign all the values relivant to the header, rather than our case specific example inputs (default constructor)

· int b\_size, int r\_size, float d\_cap, std::string i\_filename, std::string o\_filename

SequenceSet::~SequenceSet

· First destructor for the SequenceSet, which removes our variables

· &field\_count, &block\_size, &record\_size, &default\_cap, &in\_filename, &out\_filename

Sequenceset::Create

void create(); /\*! function prototype for create() that creates empty file for the header any it contains \*/

This makes the output file and writes the header to the file, defining all the header data

· Field\_tuple

· A Struct of Array (SoA) is made to hold the record information, using this, we write the header

· std::string a,std::string b,std::string c

fields

· array of structures

· std::vector<field\_tuple>

Sequenceset::load

void load(); /\*! function prototype for load() that load block of sequence set file into ram \*/

· Creates a local file for loading in that data

Sequenceset::close

void close(); /\*! function prototype for close() that is called when file needs to be closed \*/

· Closes files if needed, checking each file to see if it was left open. If a file is still open, this closes it

Sequenceset::is\_empty

bool is\_empty(int flag, int block, int record, int field); /\*! prototype for is\_empty() to know the state of the structure \*/

· if nothing is given but the flag then it will do input/output on command line for user and will take in the index's of requested whatever structure and tell you its status

· bool true if empty and false if populated

· purpose: to know the state of a structure

· int flag (0 – file, 1 – block, 2 – record, 3 – field)

Sequenceset::search

int search\_file(int primKey); /\*! function prototype search\_file(int) searches for a file \*/

· searching the block for a specific term

· std::vector<int>

· std::string search\_term

Sequenceset::populate

void populate(); /\*! function prototype for populate() that creates an empty node for a btree \*/

· This function reads in the lines after the end of header in the input file

o Each line is read as a record

o A b+ tree is created with the index struct

o A block is created

§ A block is a doubly linked list with a vector of strings

§ Each block is of size block\_size

§ The first block is assigned to first

§ Each successive block is the b - > next of the prev and the b -> prevous of the current+ Records are pushed to the blocks vector until its reached default fill size

o While the file has data each block is filled

o Starts to grab primary keys to be put into the b treesequenceset::insert

· This inserts a new record into the last block, making sure it has room. If it does not, a new block is created for the new record

Sequenceset::delete\_record

void delete\_record(int block, int record); /\*! function prototype for delete\_record(int, int) that deletes specific record from user input \*/

· This deletes a record, given which block the record is in, along with which record to delete

Sequenceset::update

void update(int block, int record, int field, std::string new\_field); /\*! function prototype for update(int, int, string) that updates a record, field or adds new field \*/

· This updates a field in a record in a specific block, given a block number, record number, and a field number (which will update the field)

· int block, int record, int filed, std::string new\_field

Sequenceset::display\_record

void display\_record(int record, int block); /\*! function prototype display\_record(int, int) displays specific record request by user input \*/

· This displays a specific record, given the record number, and in which block to view

· int record = -1, int block = -1

Sequenceset::display\_field

void display\_field(int field, int record, int block); /\*! function prototype display\_field(int, int, int) displays specific field request by user input \*/

· This displays a specific field, given the field number, record number, and in which block to view

· int field = -1, int record = -1, int block = -1

sequenceset::display\_file

· This displays the entire sequence set in records, one by one

· int limit = -1

void display\_file(int limit); /\*! function prototype display\_file(int) displays file request by user input \*/

sequenceset::display\_SS

void display\_SS(); /\*! function prototype display\_SS() to display the sequence set \*/

· This displays the entire sequence set in blocks, one by one

Sequenceset::validate

void validate(); /\*! function prototype validate() to validate a record in the file \*/

· Validates the blocks to make sure they are correct

sequenceset::addIndex

· Adds an index to the block

· Int primKey, Block \*b

Sequenceset::delIndex

· Deletes an index, given user input

· Int primKey

sequenceset::developer\_show

· Shows specifics about the sequence set (field count, labels, sizes, and types)

Sequenceset::get\_field\_range\_tuple

std::vector<int> get\_field\_range\_tuple(int field\_index); /\*! function prototype get\_field\_range\_tuple(int) for extracting the range of character index in a record\*/

· extracting range to get the where the characters index is

· std::vector<int>

· int field\_index

|  |  |
| --- | --- |
| Field | Field Description |
| zip | Zip Code of this city. |
| city | City Name |
| state | State Abbreviation of this city. |
| county | County this city is in. |
| longitude | The Longitude Coordinates of this city. |
| latitude | The Latitude Coordinates of this city. |

void state\_and\_place\_from\_zip(std::string zip);

void nsew\_most(std::string state);

**Test Driver Program Design**

In the tester.cpp file, you will find a main that allows a user to input a character to test specific things to validate the classes we have made.

Choosing an option (options are listed on the testing document), you will be able to see our tests for each of the following:

|  |  |
| --- | --- |
| Option | Test |
| d | Deletes record |
| m | Displays menu |
| r | Displays record |
| f | Displays field |
| s | Displays ss |
| i | Inserts record |
| b | Prints record |
| c | Creates a record |
| u | Updates a record |
| x | Exits program |