**Introduction to Databases, Spring 2021**

**Homework #4 (60 Pts, May 9, 2021)**

**Student ID**

**Name**

Compress ‘main.c’, ‘B+TREE.c’, ‘B+TREE.h’ and ‘your report’ (this current document file) and submit with the filename ‘HW4\_STUDENT ID.zip’

**NOTE**: You can add/modify some functions, but do not use additional libraries.

**(1)** [**40 pts**] Implement **insertion** and **deletion** operations of B+-tree. Also, show the results together for given element sequences. **(Insertion: 20 pts, Deletion: 20 pts)**

Definition of B+-tree

1. Every node has at most m children (m: Max. Degree of B+-tree).

2. Every node that contains data is a leaf node.

3. Every node (except root) has at least ⌈(m+1)/2⌉ children.

4. A non-leaf node with k children contains k−1 keys.

5. All leaves appear in the same level

To implement the B+-tree, please refer to the following sites.

- https://en.wikipedia.org/wiki/B%2B\_tree

- https://www.cs.usfca.edu/~galles/visualization/BPlusTree.html

- https://iq.opengenus.org/b-tree-search-insert-delete-operations/

**(a)** You should fill the implementation code in the B+-tree template.

**Answer: Submit your code to i-campus. Don’t write your code here.**

**(b)** Show the B+-tree for each case.

**Answer: Show your results. (Drawing or Snapshot)**

1. Max degree = 3

Insert(1, 3, 7, 10, 11, 13, 14, 15, 18, 16, 19, 24, 25, 26)

1. Max degree = 4

Insert(1, 3, 7, 10, 11, 13, 14, 15, 18, 16, 19, 24, 25, 26)

**For Deletion :**

1. Max degree = 3

Insert(1, 3, 7, 10, 11, 13, 14, 15, 18, 16, 19, 24, 25, 26) Remove (13)

1. Max degree = 4

Insert(1, 3, 7, 10, 11, 13, 14, 15, 18, 16, 19, 24, 25, 26) Remove (13)

**(2) [20 pts]** Compare the index scan and full table scan using SQL queries on MySQL. The selectivity of a predicate indicates how many rows from a row set will satisfy the predicate.

Compare the running time between index scan and full table scan according to different data selectivity and draw the graph to compare two scan methods depending on the selectivity. (Fix the total number of rows as 20,000,000). You also should explain the experimental results.

**Example code for generating synthetic table.**

**/\* Make a table \*/**

**DROP TABLE** TEST;

**CREATE TABLE** TEST (a **INT**, b **INT**);

DELIMITER $$

**DROP PROCEDURE IF EXISTS** loopInsert $$

**CREATE PROCEDURE** loopInsert()

**BEGIN**

**DECLARE** i **INT** **DEFAULT** 1;

**WHILE** i <= **20000000** **DO**

**INSERT** **INTO** TEST (a, b) **VALUES** (i, i);

**SET** i = i + **1**;

**END** **WHILE**;

**COMMIT**;

**END**$$

DELIMITER ;

**SET** autocommit=**0**;

**CALL** loopInsert;

**COMMIT**;

**SET** autocommit=**1**;

**/\* Make a index \*/**

**ALTER** **TABLE** TEST **ADD** **INDEX**(a);

**/\* Compare the running time between index scan and full table scan at selectivity 50% \*/**

**SELECT** **SUM**(a)

**FROM** TEST

**WHERE** a > **10000000**;

**SELECT** **SUM**(b)

**FROM** TEST

**WHERE** b > **10000000**;

**(a) Comparison graph for running time over selectivity.**

**Please note that this figure is only for example. It is incorrect.**

**Answer: Show the comparison graph.**

**(b) Explain why index scan is faster or slower than full table scan depending on the selectivity in your comparison results.**

**Answer: Explain your comparison graph (= answer of (2)-(a)).**