**Homework Assignment 4**

Submission due: 11:59PM June 16, 2020   
*(We will solve the problems together in class on June 12)*

|  |
| --- |
| * This is an individual work; Please be clear with HGU CSEE Standard:   + Submitting assignments or program codes written by others or acquired from the internet without explicit approval of the professor is regarded as cheating.   + Showing or lending one’s own homework to other student is also considered cheating that disturbs fair evaluation and hinders the academic achievement of the other student.   + It is regarded as cheating if two or more students conduct their homework together and submit it individually when the homework is not a group assignment. * Read the assignment carefully. *In this assignment, you will need to* ***write and execute several SQL queries****; and* ***submit the results of your queries***. * You are **allowed to re-use any of the queries from the lecture slides** while developing solutions to the problems. * When finished, submit your work to *LMS.* |

1. Read, understand Slides 24-26, and answer the following questions.

(a) What is the worst-case time complexity of a hash table? Give an example case where this happens.

Average time complexity: O(1)

-> Worst case time complexity: O(n) n = hash table capacity

Hash function returns a constant

-> Hash table with a perfect hash function: if key1 != key2, then hash(key1) != hash(key2). Always O(1)

-> Hash table's capacity is infinity: lower the time complexity

-> #collisions \propto (proportional to) time complexity

->Hashing schemes -> aims to reduce down the distance between the hash(key) position and the actual position the data is located(stored)

(b) Explain the differences between static hash tables and dynamic hash tables.

A static hash table is: a hash table for key-value objects or its chains whose size does not change once the store is created.

Dynamic hash tables can grow the store dynamically as insertions or deletions happen while in use

(c) What is an M-way search tree?

Multi-way tree that is a generalized version of binary search trees

Each node contains a maximum of M-1 elements and M children

-> Does not include any features related to balanced trees

-> M-way search tree + balanced tree = b+tree

(d) What is self-balancing tree? Why it is necessary?

Tree that automatically keeps the height as small as possible when the values that it contains change.

The height must always be at most the ceiling of log n, where b is the degree of the tree.

**Consider the tree below:**

A picture containing clock

Description automatically generated

(e) What is the degree of node B?

2

(f) What is the degree of node C?

1

(g) What is the degree of the tree?

3

(h) What is the height of the tree?

3

2. Consider a hash table with the Cuckoo hashing scheme. Let us assume that this data structure uses two internal hash tables, which are based on the following hash functions *h*1(*key*) and *h*2(*key*).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *key* | *Value* | *h*1(*key*) | *h*2(*key*) |
| *hi*: *key* → *hi*(*key*) | A | 48 | 2 | 3 |
| B | 29 | 1 | 0 |
| C | 91 | 3 | 1 |
| D | 43 | 2 | 0 |
| E | 26 | 1 | 0 |
| F | 57 | 0 | 3 |

Assuming that each of the internal hash tables is of capacity 4, draw the final state of the hash table (two internal hash tables) below. For all coin flips, you may assume that the algorithm always chooses the hash table with the lower number (*i.e.*, lower hash table number). Note that each entry of the hash table stores both a key and an associated value.

Internal Hash Table #1 Internal Hash Table #2

E

B

[0] B|29

[1]

[2]

[3] A|48

[0] F|57

[1] E|26

[2] D|43

[3] C|91

D

C

F

A

3. Consider the B+tree of maximum degree 4 with a single node given below.

11

15

19

(a) Show the tree after inserting the key of value 7.

7

11

15

19

15

15

19

7

11

(b) Show the tree after inserting the key of value 17 (start from the original depth-1 tree).

11

15

17

19

17

17

19

11

15

(c) Show the tree after inserting the key of value 20 (start from the original depth-1 tree).

20

11

15

19

19

19

20

11

15

4. Consider the B+tree of maximum degree 4. Insert the keys 11, 29, 17, 31, 47, 14, 21, 30, and 39. Show the progression.

Insert 11

11

Insert 29

11

29

Insert 17

11

17

29

Insert 31

29

29

31

11

17

Insert 47

29

29

31

47

11

17

Insert 14

29

29

31

47

11

14

17

29

29

31

47

11

14

17

Insert 21

29

29

31

47

11

14

17

11 14 | 17 21

17

29

29

31

47

17

21

11

14

Insert 30

17

29

31

29

30

17

21

31

47

11

14

Insert 39

17

29

31

29

30

17

21

31

39

47

11

14