EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

DB2
Sommersemester 2022
Prof. Torsten Grust, Denis Hirn
WSI — Database Systems Research Group

Assignment 7

Handin until: Friday, 01.07.2022, 09:00

1. [8 Points] **B⁺Tree - Insert**

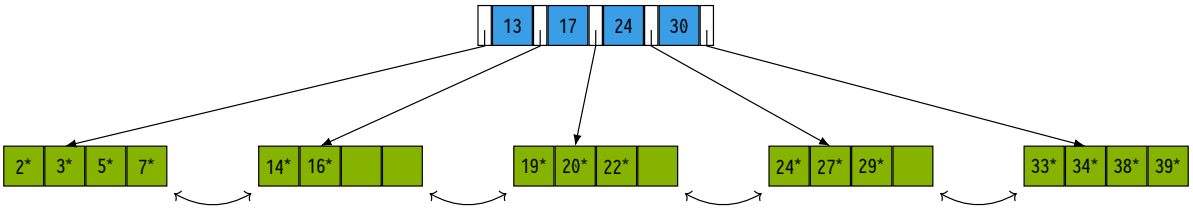

Figure 1: A B⁺Tree

Assume that all key values inserted into the B+Tree are unique and that the `Insert`-operation of the index does **not implement *redistribution***.

Answer the following questions, each based on the *unmodified* B⁺Tree of Figure 1:

(a) Identify *four* leaf node entries ($a, \ldots, d$) which, when inserted in sequence (`Insert` $a, \ldots,$ `Insert` d), completely fill the leaf pages.

(b) What is the minimum number of `Insert` operations required to increase the size of the tree by two levels?

(c) Take all entries of the leaf level sequence set (`2,3,5,...`) and `Insert` them, one by one, into a new B⁺Tree of order $o = 1$. Submit sketches of the B⁺Tree instances after the first three `Insert` steps, along with the final resulting B⁺Tree.

Note: You can submit your solutions as plain text, PDF or PNG image file. However, handwritten solutions are **not** allowed and will not be graded.

2. [12 Points] **B⁺Tree - Maintenance**

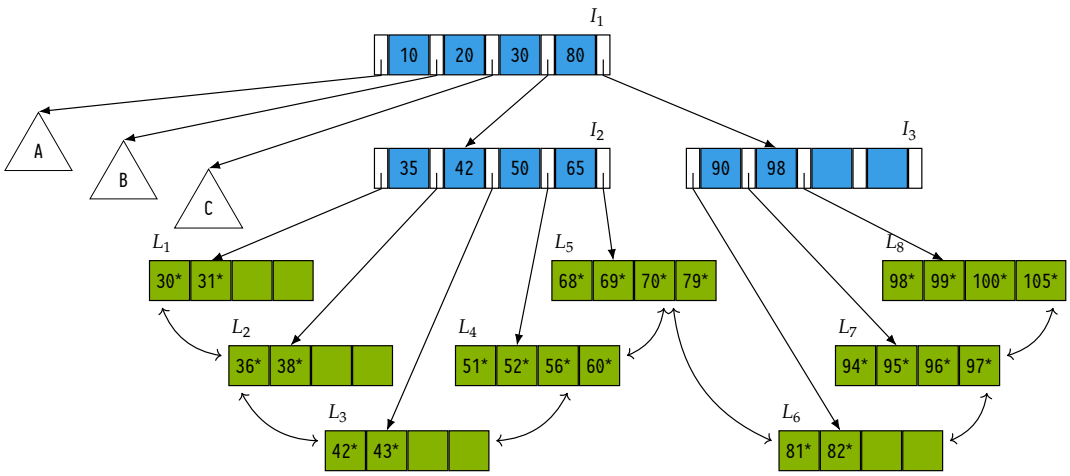

Figure 2: Another B⁺Tree

Assume that all key values inserted into the B+Tree are unique. However, assume both index operations, `Insert` and `Delete`, to implement ***redistribution*** **at leaf level**. Note that redistribution occurs only between direct siblings, that is, between directly connected leaf nodes that share the same parent node.

Answer the following questions, each based on the *unmodified* B⁺Tree of Figure 2:

(a) Note all nodes (I_j or L_k) that need to be read by the RDBMS to answer the following queries:
   i. "Find all records with a key value greater than **38**"
   ii. "Delete the record with key value **81**"

(b) Add a record with the key value **104** to the tree. Sketch the resulting tree.

(c) Name any key values that directly result in an increase in tree depth when inserted.

(d) Delete the entry with key value **36** from the tree. Sketch the resulting tree.

(e) The subtrees **A**, **B** and **C** have not been fully specified. Describe all characteristics that can still be inferred about these subtrees.

Note: You may hand in all tree sketches as plain text, PDF or PNG image file. Again, handwritten solutions are **not** allowed and will not be graded.

3. [10 Points] **B⁺Tree - PostgreSQL**

The file **`btree.sql`** creates and populates table

```
indexed (a INT PRIMARY KEY, b TEXT, c NUMERIC(3,2)).
```

Load the file with PostgreSQL and answer the following questions. Hand in all SQL queries and intermediate results used to find the final answer.

Note: The tasks require you to use functions **`bt_metap(relname TEXT)`**, **`bt_page_stats(relname TEXT, blkno INT)`** and **`bt_page_items(relname TEXT, blkno INT)`** previously mentioned in the lectures. For more information about these functions, read the documentation at

https://www.postgresql.org/docs/current/pageinspect.html#id-1.11.7.31.6.

(a) How many pages were created for the index that PostgreSQL automatically created based on the primary key?

(b) Write a query to find the *root node* of the B⁺Tree. What is its page number and what is its and fan-out?

(c) What is the *average* fan-out of all non-leaf nodes?

(d) Use the functions `bt_page_stats(...)` and `bt_page_items(...)` to manually traverse the B$^+$Tree from the root to the index leaf page containing the key entry for `a = 150 000`. What are the **minimum** and **maximum key values** found on this leaf node? To which pages of relation `indexed` do the entries of that leaf node point?

Note: Function `bt_page_items(...)` returns a column `data` which represents the key value of an index entry. The `data` value is encoded as a hexadecimal string. For example, the value `'77 97 01 00 00 00 00 00'` is read as the hexadecimal number `0x19777` which can be converted to the decimal number `104311`.

For convenience, we have provided function `data_to_numeric(TEXT)` in `btree.sql`, to convert these `data` values to decimal numbers. It can be used as follows:

```
1 SELECT data_to_numeric('77 97 01 00 00 00 00 00');

 data_to_numeric
-----------------
          104311
(1 row)
```