Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg

January 16, 2020

Databases IIB: DBMS-Implementation — Exercise Sheet 10 —

Please read Part a) and b), think about the answers, and mark questions which you want to discuss in class. You only have to submit Part g) to i). Please upload your solution into the StudIP file folder called "Hausaufgabe_10" in the StudIP entry of the lecture. The deadline is January 21 (the day before the next lecture).

Repetition Questions

a) What would you answer to the following questions in an oral exam? (These questions are still about the physical storage of relations.)

- What are advantages and disadvantages of fixed-length rows compared to rows of variable length? Consider also the case of an `ALTER TABLE` that adds a column. You may also think about data compression techniques.
- Which blocks does Oracle read in a full table scan? Give an example where this looks very inefficient. What are the reasons for this problem? If you would program your own DBMS, would you use the same technique as Oracle?
- Consider a block format for fixed size rows and a linked list of blocks which have free space. How is this linked list managed (i.e. where do you find the start of the linked list and what happens on insertions and deletions of rows)? When a block was full, and a single row was deleted, why might it be good to put the block not immediately again at the start of the linked list (but wait instead until it has space for several rows)? Discuss advantages and disadvantages of this technique.
- Explain the row format of Oracle. How does one find the value of a given column? Discuss alternatives. What are advantages and disadvantages of the Oracle row format? If you develop your own DBMS, which row format would you choose?
- Why do columns at the end of the row that contain a null value need no storage space in Oracle?
- How can one calculate the storage size of a `NUMERIC` column value in Oracle? What are the components of the value in the internal representation? How much space would the value `123` need? And what about the value `-12.34`?
- Explain the `ANALYZE TABLE` command. Why is it no problem if the statistics in the data dictionary are not 100% current? When should the DBA run the `ANALYZE TABLE` command? Why should this not be done during the main office hours?

b) And how would you answer the following questions about B-trees?

- What is the difference between a binary search tree and a B-tree?
- Why does one store many search key values in a node of a B-tree? How is the maximal number determined?
- Why is a B-tree balanced? What are the exact conditions? What is the worst possible case, and what sequence of insertions gives this B-tree? Why does one not require perfect balancing?
- What is the complexity of searching in a B-tree? What is the complexity of insertion?
- What is the difference between a B-tree and a B$^+$-tree? Why do most database systems use B$^+$-trees?
- Explain the algorithm for insertion in a B$^+$-tree (for simplicity, assume that it is a unique index, and the search key values have all the same size).
- Give an algorithm for creating a B$^+$-tree on an attribute for a given table (of course, one can do this as many insertions on an empty B$^+$-tree, but there is a better way).
- For what operations in relational algebra can a B$^+$-tree be used? You can also give typical SQL queries and explain how they are evaluated with a B$^+$-tree index.
- For which selections will a B$^+$-tree index return the ROWIDs in sorted order, and for which not? Why is it useful to get ROWIDs sorted by position on disk?
- What is the difference between an index on the column combination (A, B) and an index on the column combination (B, A)?
- Suppose you have two indexes, one on A, and one on B, and the query contains equality conditions for them with known constants, e.g. $A = 1234 \wedge B = 56789$. How can you use both indexes?
- Compare a full table scan with an access to a table via an index and looking up the tuples for the ROWIDs. What does one have to know about the relation to decide which query evaluation plan is better? Given an example where using the index is worse than doing a full table scan.
- What is an index-only query evaluation plan? Given an example for a query that can be evaluated in this way.
- In which cases is an index useful for evaluating an ORDER BY clause? Discuss also possible disadvantages.
- What are disadvantages of indexes? In which cases you should not define an index on a column of a relation?

In-Class Exercises

c) Create the relation for which you calculated the storage size in the last homework:

```
R(A: numeric(5), B: varchar(10), C: varchar(50))
```

Make sure that `PCTFREE` is 10. For the example studied in the homework, we need to insert 10 000 rows of the form:

```
(99999, null, 'abcdefghijklmnopqrstuvwxyz')
```

It is also possible to use unique numbers instead of `99999`, but please make sure that they need five digits.

- First, insert only a single row.
- Have a look at the storage size of the columns with the function `VSIZE`, e.g. select `VSIZE(A)` (and compare it with the value you calculated in the homework).
- Execute the command

  ```
  ANALYZE TABLE R COMPUTE STATISTICS
  ```
- Then look at the data dicionary table `TABS`, in particular the columns `NUM_ROWS`, `BLOCKS`, `EMPTY_BLOCKS`, `CHAIN_CNT`, `AVG_ROW_LEN`, `AVG_SPACE`. Check whether all values are what you would expect. Of course, `AVG_ROW_LEN` should be the row length that you computed in the homework.

d) A PL/SQL procedure that inserts rows until one block is full is:

```
CREATE OR REPLACE PROCEDURE FILL_BLOCK AS
    N NUMBER;
BEGIN
    N := 1;
    WHILE N < 2 LOOP
    INSERT INTO R VALUES(99999, null, 'abcdefghijklmnopqrstuvwxyz');
        SELECT COUNT(DISTINCT DBMS_ROWID.ROWID_BLOCK_NUMBER(ROWID))
        INTO N FROM R;
    END LOOP;
END;
/
```

You can download a file that contains this procedure definition from

[http://www.informatik.uni-halle.de/˜brass/dbi19/fill_block.sql]

Execute the file in SQL*Plus to declare the procedure. If you should get the message "procedure created with errors", enter the command "`SHOW ERRORS`". Then you run the procedure with: `call fill_block();`

- How many rows does the table now contain? The number is one greater than the number of rows that fit into one block.

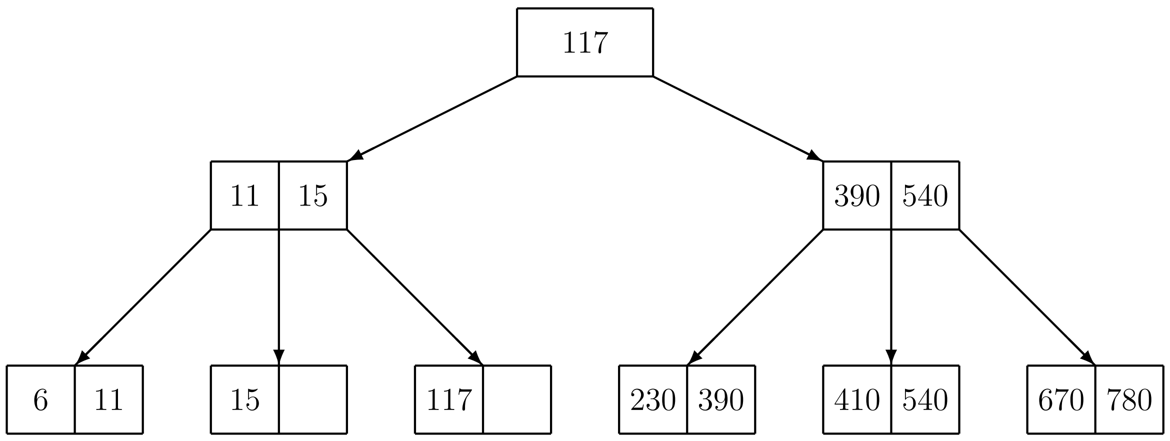
- You can also have a look at the ROWIDs of the actual rows, and count the rows per block (using `GROUP BY`). As a reminder, you can get the block number in which a row is stored with `DBMS_ROWID.ROWID_BLOCK_NUMBER(ROWID)`. The functions for the other components of a ROWID are: `ROWID_OBJECT` (check whether this is the segment number), `ROWID_RELATIVE_FNO` (find the corresponding data file), `ROWID_ROW_NUMBER`.
- Execute the `ANALYZE TABLE` command again.
- Then look at the data dicionary table `TABS` again, and check whether the values of the columns `NUM_ROWS`, `BLOCKS`, `EMPTY_BLOCKS`, `CHAIN_CNT`, `AVG_ROW_LEN`, `AVG_SPACE` are what you expect. You can also look at the table `COLS`, and check `NUM_DISTINCT` and `NUM_NULLS`. The table `USER_TAB_COL_STATISTICS` might also be interesting.

e) Insert the remaining rows so that the table contains 10 000 rows. (You have to modify the procedure again for that or create a new procedure). Do the `ANALYZE TABLE` again and check the size information. In particular, you need the number of used blocks to compare it with your homework result.

f) Do the update that sets column `B` of all rows to `'ABCDEFGHIJ'`. Do the `ANALYZE TABLE` again and check whether there are migrated rows (i.e. the column `CHAIN_CNT` in `TABS`). If there should be migrated rows, do the table reorganisation. If there are no migrated rows, experiment with longer updates.

Homework Exercises

Consider the following B$^+$-tree of height 3 with maximally 2 entries per node:



What is the resulting B$^+$-tree after the following operations? Please use the given B$^+$-tree as input for each operation, and not the result of the previous operation.

g) Insertion of 12.

h) Insertion of 7.

i) Insertion of 300.