CSCI 6315 Applied Database Systems
ASSIGNMENT 4: Data Storage and Querying, Transaction Management
Due is 04/28/2020 00:00

Ulvi Bajarani
Student ID 20539914
E-mail: ulvi.bajarani01@utrgv.edu

Questions and Answers:

Problem 1. This problem has two parts:

a. Construct a B^+-tree for the following key values:

2, 3, 5, 7, 11, 17, 19, 23, 29, 31.

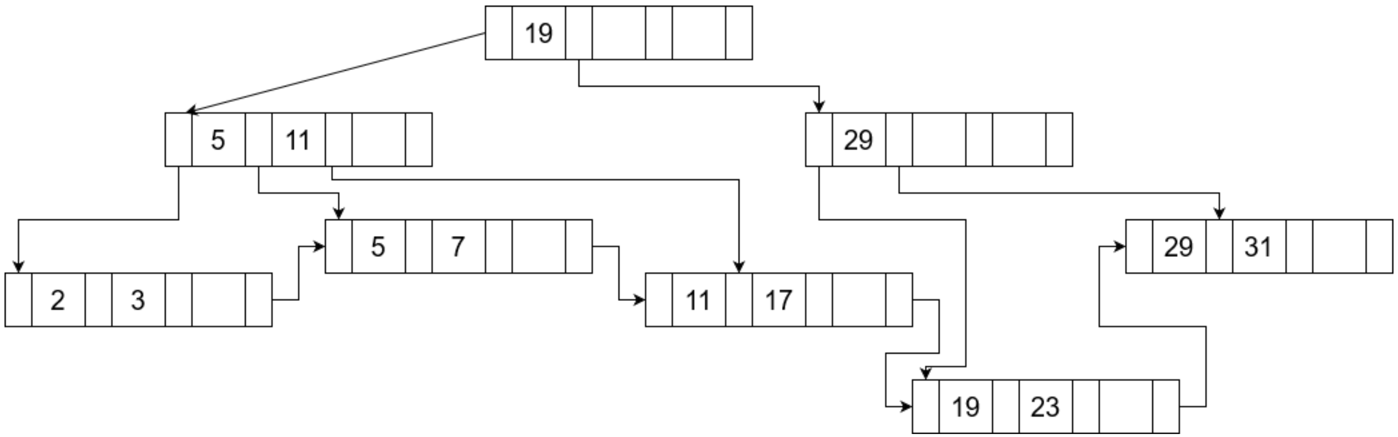
Assume that the number of pointers that will fit in one internal node is 4 and each leaf node can store 3 key values.

b. After the B^+-tree is constructed for Part a, show the final tree after the following operations:

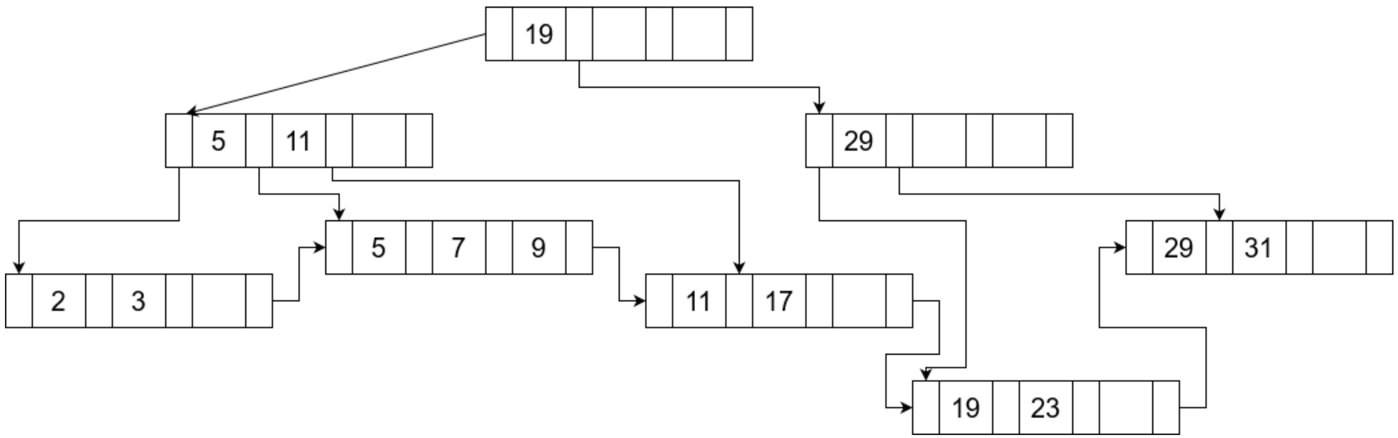
1. Insert 9
2. Insert 10
3. Insert 8
4. Delete 23
5. Delete 19

Answer 1.

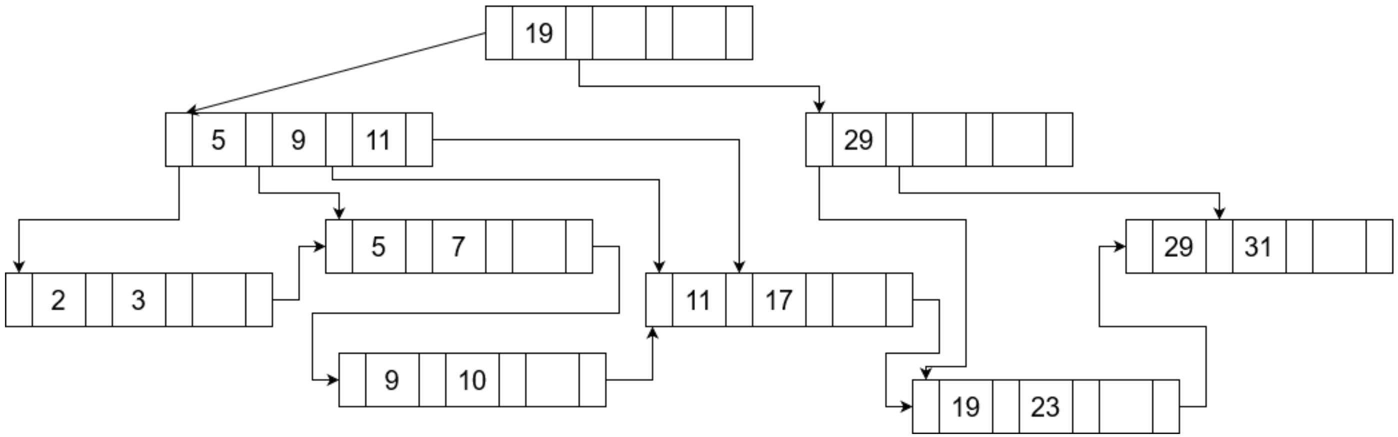
a. ***Assuming that nodes might contain 4 pointers (3 keys), the*** B^+ ***Tree is:***



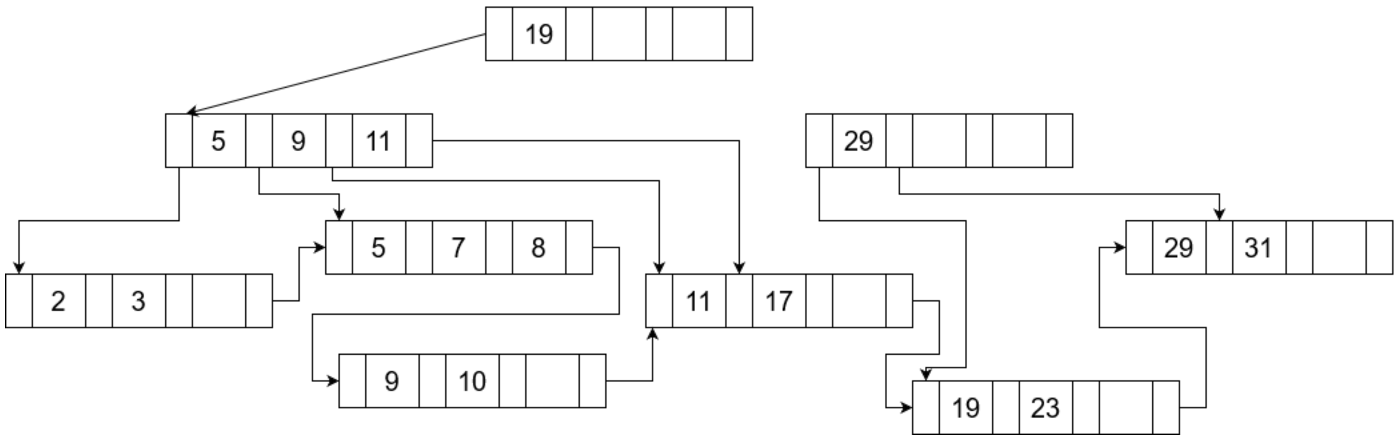
b.1. **After 9 insertion:**



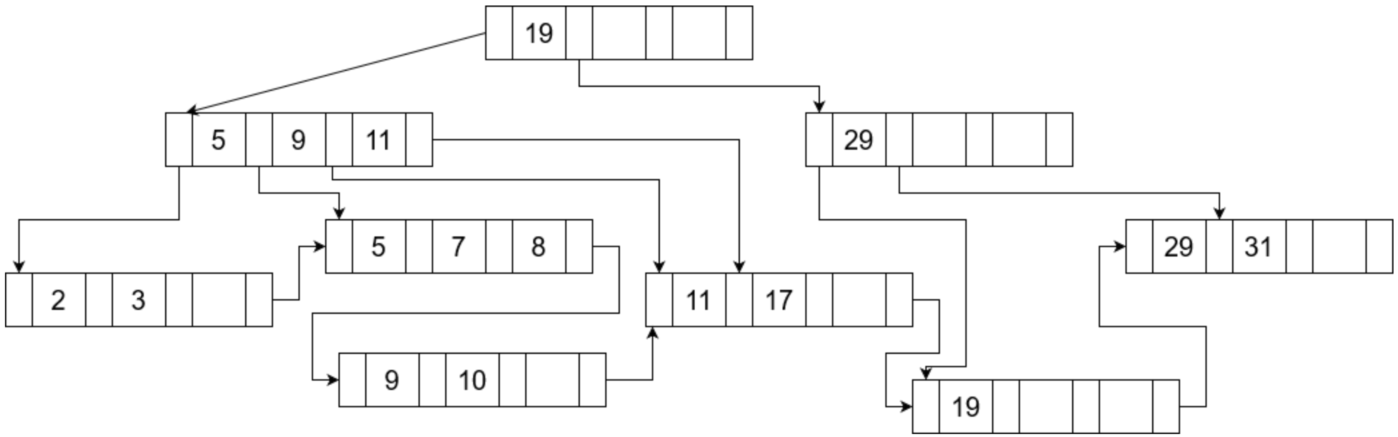
b.2. **After 10 insertion:**



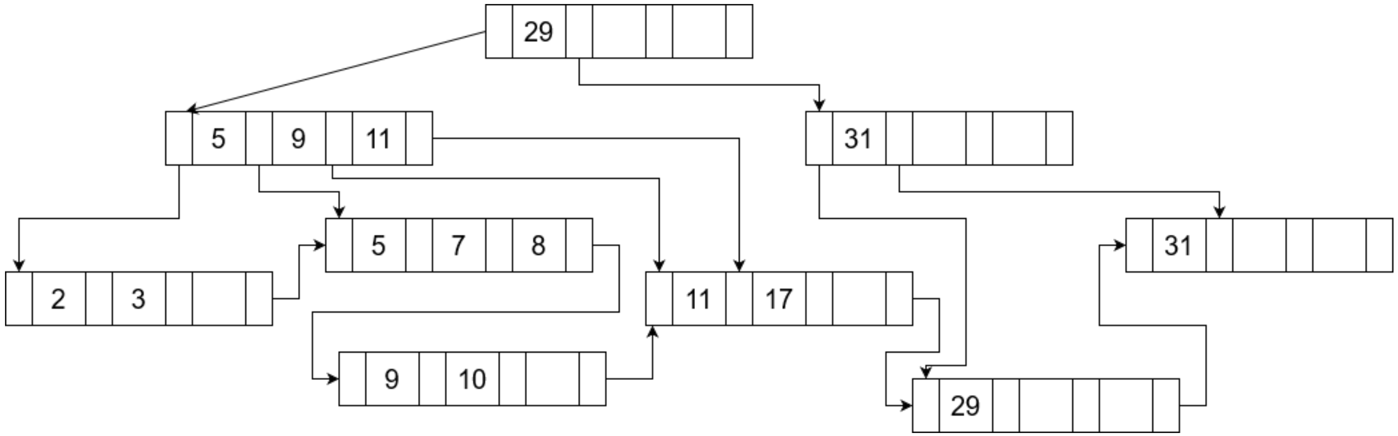
b.3. **After 8 insertion:**



b.4. **After 23 deletion:**



b.5. **After 19 deletion. The final version:**



Problem 2. This problem has two parts:

a. Suppose that we are using extendable hashing on a file that contains records with the following search key values:

2, 3, 5, 7, 11, 17, 19, 23, 29, 31.

Show the extendable hash structure for this file if the hash function is $h(x) = x \% 8$ and each bucket can hold three records.

b. After Completing Part a, show the extendable hash structures after the following operations:

b.1. Delete 11

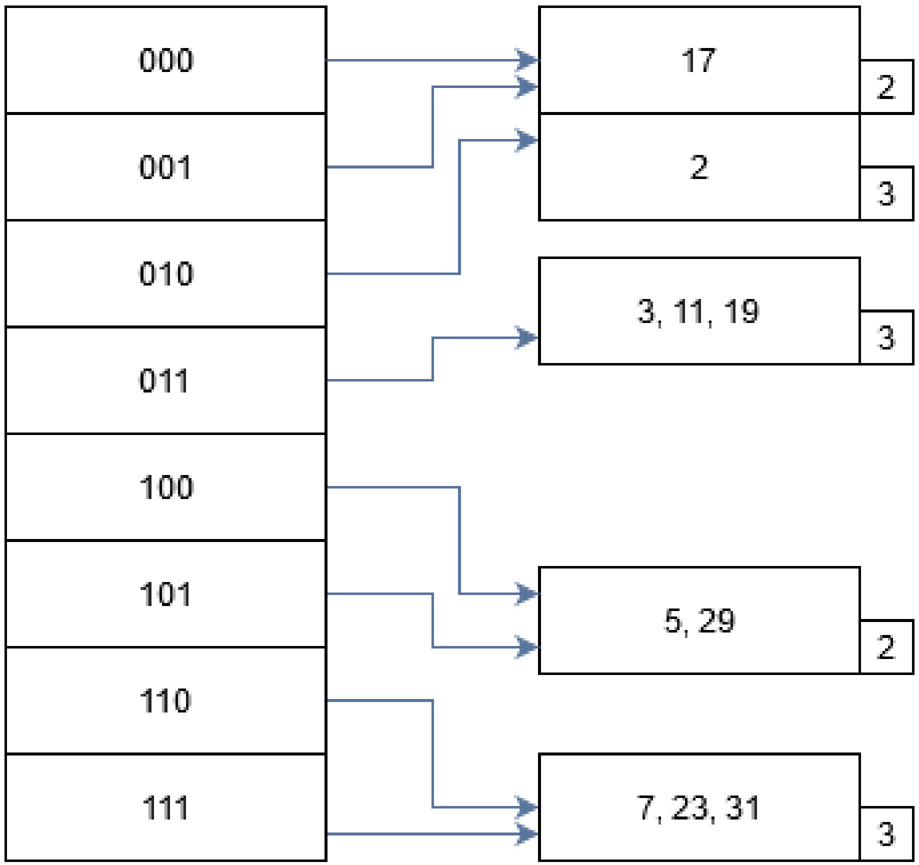
b.2. Delete 31

b.3. Insert 1

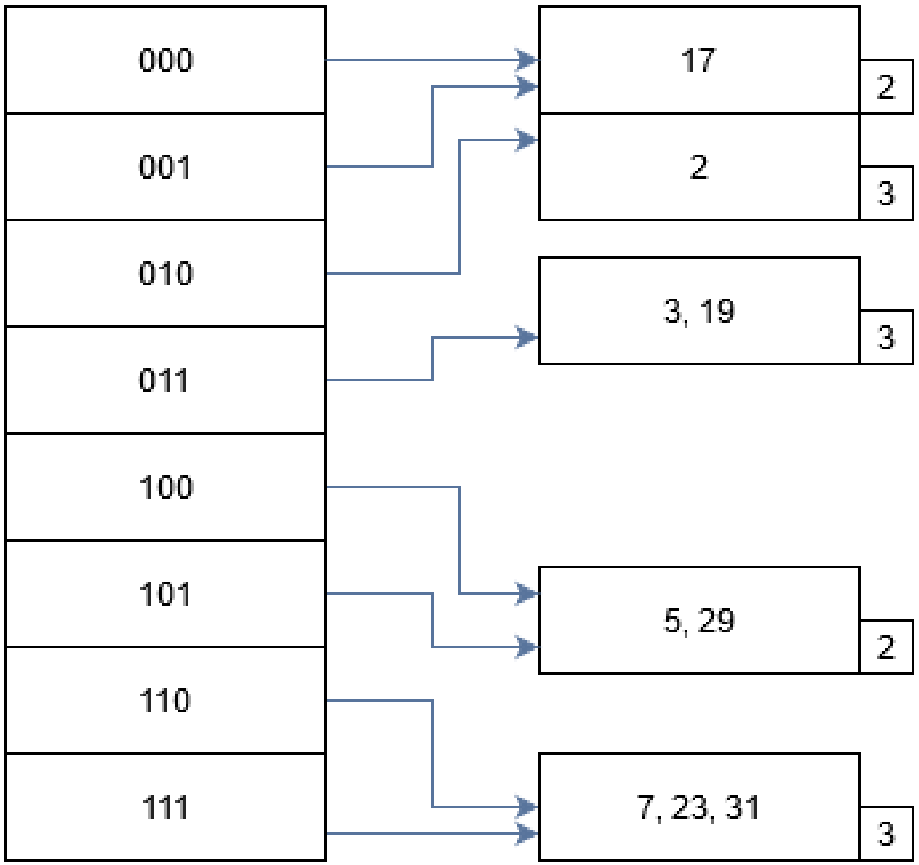
b.4. Insert 15

Answer 2.

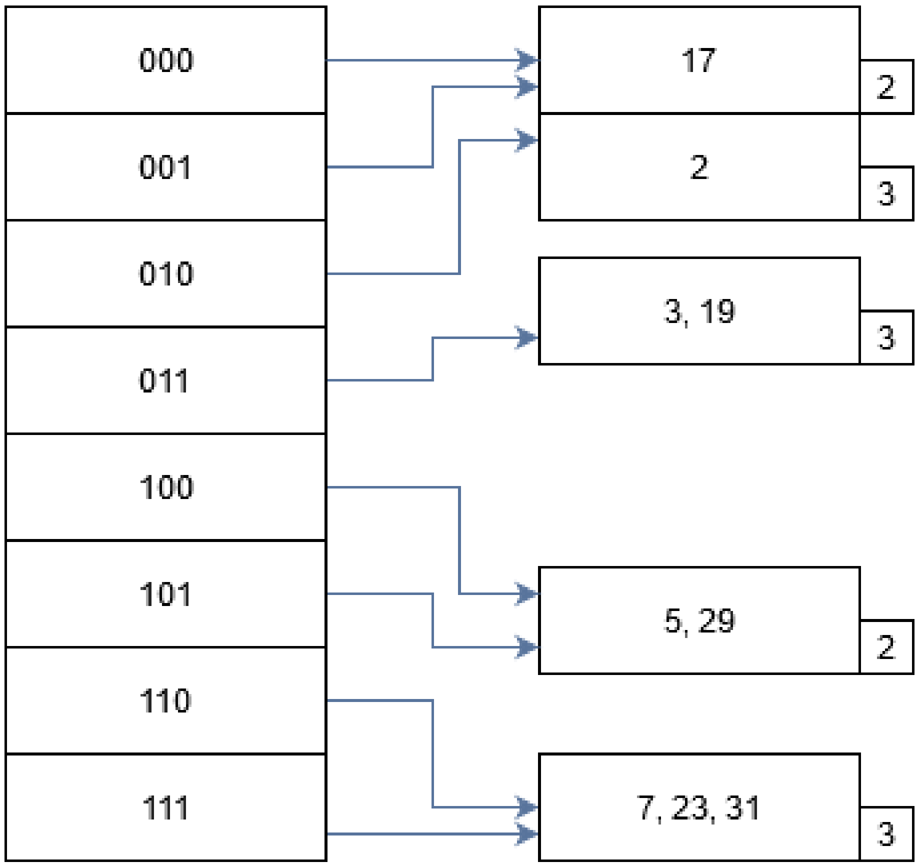
a. Since that we don't have the size in the description, I'll assume that there are two values initially (0 for 0 and even mods and 1 for odd mods). After the extensions, then table should look like this:



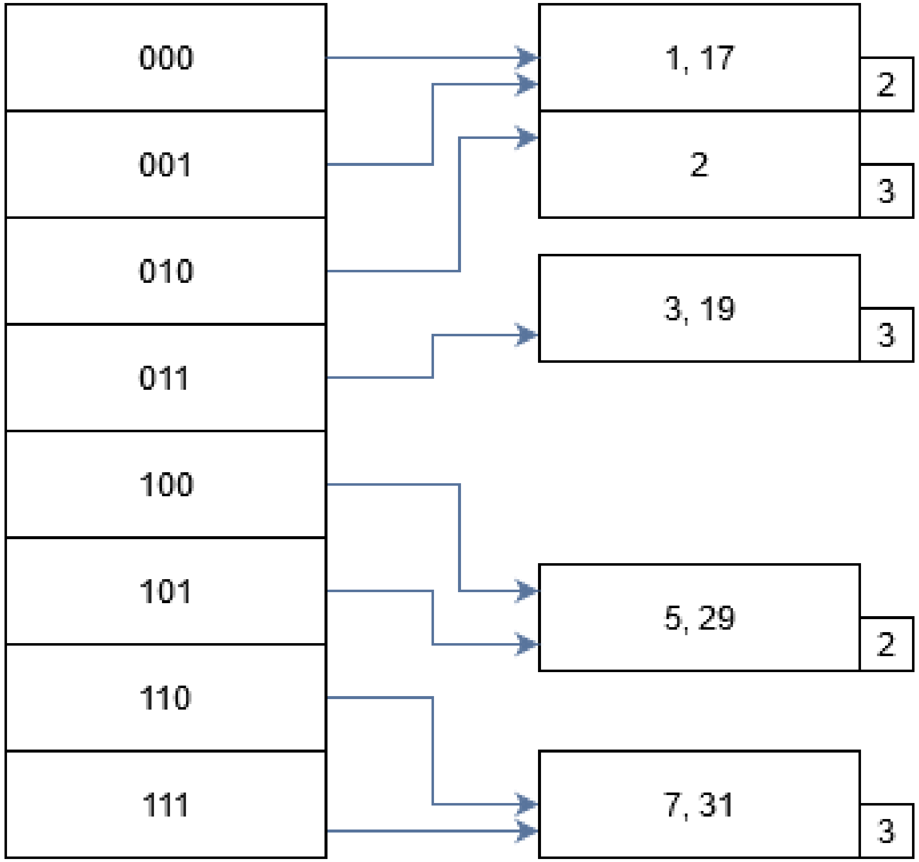
b.1. **After 11 deletion:**



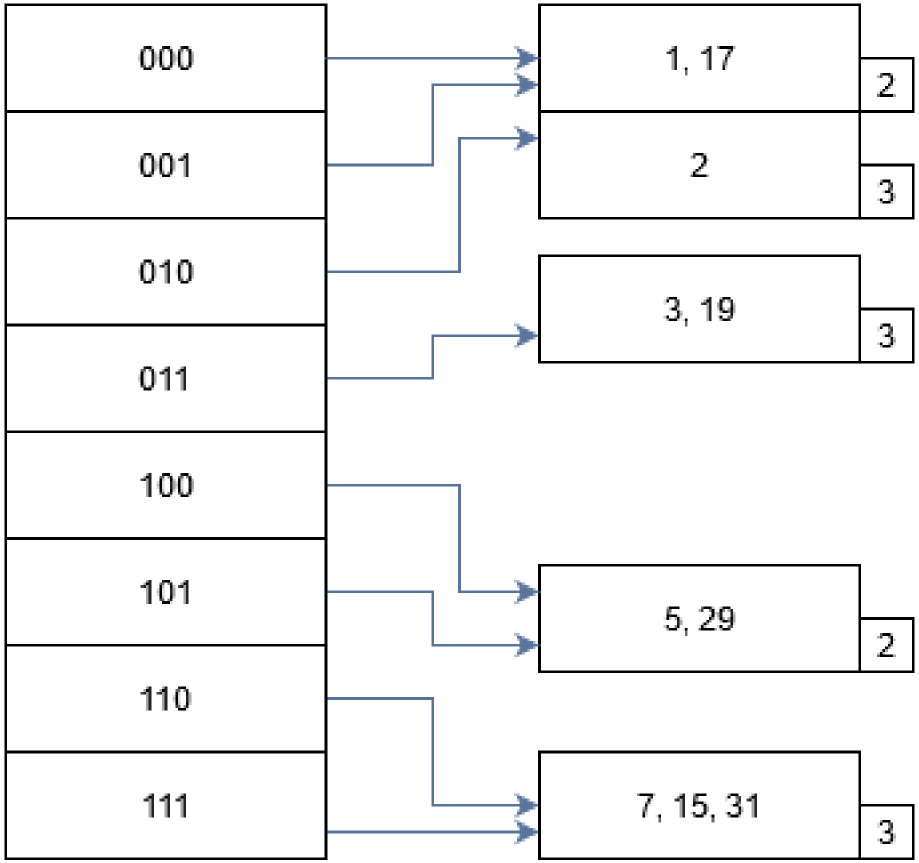
b.2. **After 31 deletion:**



b.3. **After 1 insertion:**



b.4. **After 15 insertion. The final version:**



Problem 3. Consider the following SQL query for our University Database:

select T.dept_name
from department as T, department as S
where T.budget > S.budget and S.building = "MAG"

Write an efficient relational-algebra expression that is equivalent to this query. Justify your choice.

Answer 3.

$$\Pi_{T.dept\_name}((\Pi_{building,\ budget}(\rho_T(department)))) \bowtie_{T.budget\ >\ S.budget} (\Pi_{budget}(\sigma_{(S.building=\text{“}MAG\text{”})}(\rho_S(department))))$$

The expression is efficient due to these reasons:

b.1. The expression performs the theta-join on the smallest possible amount of data, because of the restriction the right-hand side of the join to only the branches in “MAG”.

b.2. The expression eliminates the unnecessary attributes from both sides of the theta-join operation, e.g. from both the operands.

Problem 4. Let the relations $r_1(A,\ B,\ C)$ and $r_2(C,\ D,\ E)$ have the following properties: r_1 has 20,000 tuples, r_2 has 45,000 tuples, 25 tuples of r_1 fit on one block, and 30 tuples of r_2 fit on one block. Estimate the number of block accesses required, using each of the following join strategies for $r_1 \bowtie r_2$

a. Nested-loop join

b. Block nested-loop join

c. Merge join

d. Hash join

Answer 4.

r1 = 20000 tuples
r2 = 45000 tuples
25 tuples of r_1 fit on one block
30 tuples of r_2 fit on one block

br_1 = Number of blocks for r_1 is 20000/25=800
br_2 = Number of blocks for r_2 is 45000/30=1500

a. In the worst case,

   $br_1 + r_1 = 800 + 20000 = 20800$ seeks

   $nr_1 * br_2 + br_1 = 20000 * 1500 + 800 = 30000800$ block transfers.

   Total disk accesses $= 30000800 + 20800 = 30021600$

   or

   $br_2 + r_2 = 1500 + 45000 = 46500$ seeks

   $nr_2 * br_1 + br_2 = 45000 * 800 + 1500 = 36001500$ block transfers.

   Total disk accesses $= 36001500 + 46500 = 36048000$

In the best case,

$br_1 + br_2 = 800 + 1500 = 2300$ transfers + 2 seeks = 2302 disk accesses.

b. In the worst case,

$2 * br_1 = 2 * 800 = 1600$ seeks

$br_1 * br_2 + br_1 = 800 * 1500 + 800 = 1200800$ block transfers.

Total disk accesses $= 1200800 + 1600 = 1202400$

or

$2 * br_2 = 2 * 1500 = 3000$ seeks

$br_2 * br_1 + br_2 = 1500 * 800 + 1500 = 1201500$ block transfers.

Total disk accesses $= 1205300$

In the best case,

$br_1 + br_2 = 800 + 1500 = 2300$ transfers + 2 seeks.

c. The block transfers equal to $br_1 + br_2 = 800 + 1500 = 2300$ transfers. In the worst case, $br_1 + br_2 = 800 + 1500 = 2300$ seeks are also required. The case that the data in the blocks might require the sorting. In the worst case, where memory size are 3 blocks (1 for buffer block):

Number of passes for $br_1 = log_{M-1}(br_1/M) = log_2(800/3) = log_2 266.3 \approx 8.05 \approx 9$ passes.

Number of transfers for $br_1 = br_1 * (2[log_{M-1}(br_1/M)] + 1) = 800 * (2 * log_2 266.3 + 1) = 15200$ block transfers.

Number of seeks for Number of seeks for $br_1 = 2[br_1/M] + br_1 * (2[log_{_{M-1}}(br_2/M)] - 1) = 2 * 266.3 + 800 * (2 * log_2 266.3 - 1) = 14,132.6 \approx 14133$ seeks.

Number of passes for $br_2 = log_{_{M-1}}(br_2/M) = log_2(1500/3) = log_2 500 \approx 8.96 \approx 9$ passes.

Number of transfers for $br_2 = br_2 * (2[log_{_{M-1}}(br_2/M)] + 1) = 1500 * (2 * log_2 500) + 1) = 27000$ block transfers.

Number of seeks for $br_2 = 2[br_2/M] + br_2 * (2[log_{_{M-1}}(br_2/M)] - 1) = 2 * 500 + 1500 * (2 * log_2 500 - 1) = 26500$ seeks.

So, Total disk accesses in the worst case are $2 * 2300 + 26500 = 31000$ disk accesses.

d. ***If the partitioning is required:***

Block transfers are

$2(br_1 + br_2)[log_{_{M-1}} br_2 - 1] + br_1 + br_2 = 2 * (1500 + 800) * [log_{_{M-1}}(800) - 1)] + 1500 + 800$
or
$2(br_1 + br_2)[log_{_{M-1}} br_2 - 1] + br_1 + br_2 = 2 * (1500 + 800) * [log_{_{M-1}}(1500) - 1)]$ disk accesses, where M is the pages of memory and $M < 800/M$ (the first case) or $M < 1500/M$ (the second case).

If the partitioning is not required $M \geq 800/M$ ***(the first case) or*** $M \geq 1500/M$***:***

Number of disk accesses are
$3 * (br_1) + 4n_h$

Ignoring $4n_h$, we receive almost $3 * (br_1 + br_2) = 6900$ disk accesses.

Problem 5. Show that the following equivalences hold:

a. $E_1 \bowtie_{\Theta} (E_2 - E_3) = (E_1 \bowtie_{\Theta} E_2 - E_1 \bowtie_{\Theta} E_3)$

b. $\sigma_{\Theta_1 \wedge \Theta_2}(E_1 \bowtie_{\Theta_3} E_2) = \sigma_{\Theta_1}(E_1 \bowtie_{\Theta_3} (\sigma_{\Theta_2}(E_2)))$, where Θ_2 involves attributes from E_2.

Answer 5.

a. $E_1 \bowtie_{\Theta} (E_2 - E_3) = (E_1 \bowtie_{\Theta} E_2 - E_1 \bowtie_{\Theta} E_3)$

Assume that $E_1 \bowtie_{\Theta} (E_2 - E_3)$ is R_1, $E_1 \bowtie_{\Theta} E_2$ is R_2, and $E_1 \bowtie_{\Theta} E_3$ is R_3. From this approach might be seen that if a tuple t belongs to R_1, it will also belong to R_2. If tuple t belongs to R_3, $t[E_3$'s attributes$]$ will also belong to E_3, because it cannot belong to R_1. By these two views, we reach to the conclusion that

$\forall t,\, t \in R_1 \implies t \in R_2 - R_3$

If the tuple belongs to $R_2 - R_3$, then $t[R_2$'s attributes$] \in E_2$ and $t[R_2$'s attributes$] \notin E_3$. Therefore,

$\forall t,\, t \in R_2 - R_3 \implies t \in R_1$,

which proves the equality.

b. It is easy to prove it by using various equivalence rules:

$$\sigma_{\Theta_1 \wedge \Theta_2}(E_1 \bowtie_{\Theta_3} E_2) = \sigma_{\Theta_1}(\sigma_{\Theta_2}(E_1 \bowtie_{\Theta_3} E_2)) = \sigma_{\Theta_1}(E_1 \bowtie_{\Theta_3} (\sigma_{\Theta_2}(E_2)))$$