Datdat Øving 4

Adrian Langseth, Eldar Sandanger, Mathias Chunnoo, Tore Fossland

April 2019

Task 1: Normal forms and splitting of tables

a) Is the table on second normal form? Why or why not?

The table is on second normal form as all non-key attributes are fully functionally dependent on the primary key.

b) Is the table on third normal form? Why or why not?

The table is not on 3NF, as there is a transitive functional dependency. C $\rightarrow$ BA, and A $\rightarrow$ D. This could be solved by making a separate table for the A $\rightarrow$ D.

c) Is the table on Boyce-Codd normal form (BCNF)? Why or why not?

Since the table is not on 3NF, it is not on BCNF.

d) Explain what criterias we require when a table is split into multiple subtables

When we split a table into multiple subtables, we require that

- each table has its own primary-key.
- relations and functional dependencies between tables are preserved. This can be done using junction tables or foreign keys.

e) The table R is now to be decomposed into two tables R1 and R2, which both must be on BCNF. Find a suggestion for R1 and R2, satisfying the criterias.

A suggestion might be $R_1 = [C, B]$ and $R_2 = [A, D]$. In this case, the functional dependencies can be preserved and C and A become primary keys for their respective tables. All non-key attributes are dependent on the primary key, and only the primary key, which implies that it fulfills the requirements of BCNF.

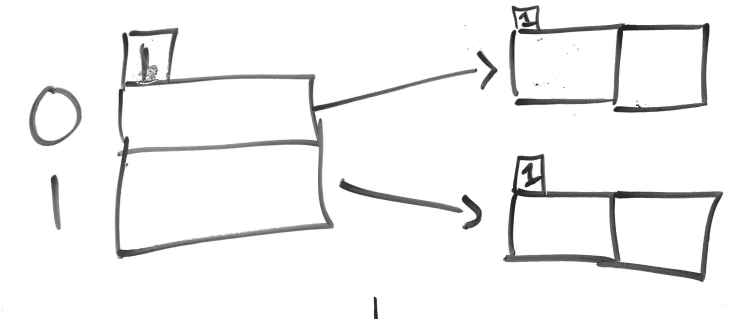

Figure 1: Before extendable hashing

f) Fill in the empty rows, such that it is a valid way of storing the information

Name	Role	Hobby
Chris Martin	Vocals	Elephants
Chris Martin	Piano	Politics
Chris Martin	Guitar	Politics
Chris Martin	Guitar	Elephants
Chris Martin	Piano	Elephants
Chris Martin	Vocals	Politics

Task 2: Extendible hashing

See figure 1 and 2.

Task 3: Construction of B+ tree

See Figure 3 and 4.

Task 4: Storage and queries with a clustered B+ tree

a)

$$\frac{10000}{8} \cdot \frac{3}{2} = 1875 \tag{1}$$

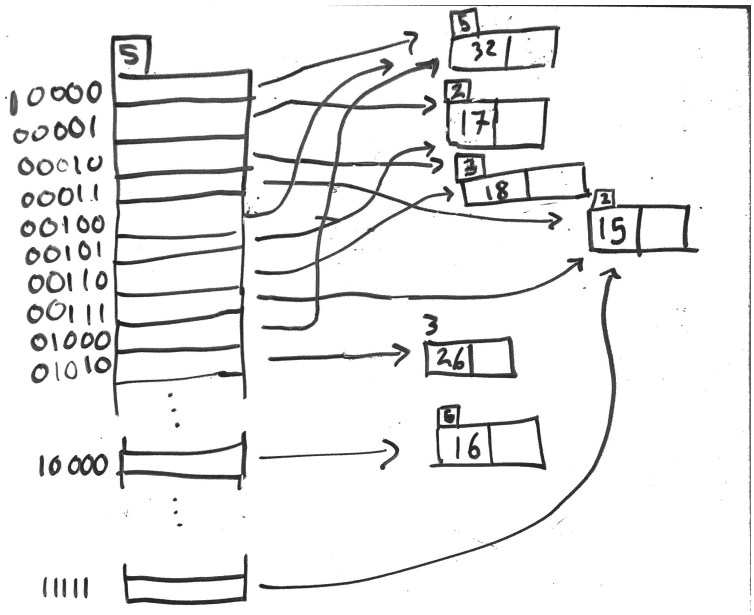

Figure 2: After extendable hashing

b)

One block has 2048 bytes of space. When dividing into pointers and id's of each 4 byte, we get 254 posts for id's and 255 pointers. Because the fillgrade is $\frac{2}{3}$, we get

$$\frac{2}{3} \cdot 255 = 170 \tag{2}$$

170 pointers. Then we can just divide the 1875 blocks we want to point to with the amount of pointers per block on level 2 to get the amount of blocks we need on level 2.

$$\frac{1875}{170} = 11.03 \tag{3}$$

The conclusion is then that we need 12 blocks on level 2 and only one block on the root level.

c)

1. If you index using personId, you can find the correct block at first try by going through the root level and second level sequentially. Therefore you only need to access three blocks in total.

2. We have to access every leafnode + one node from each level $\rightarrow$ we have to access $2 + N$ blocks.

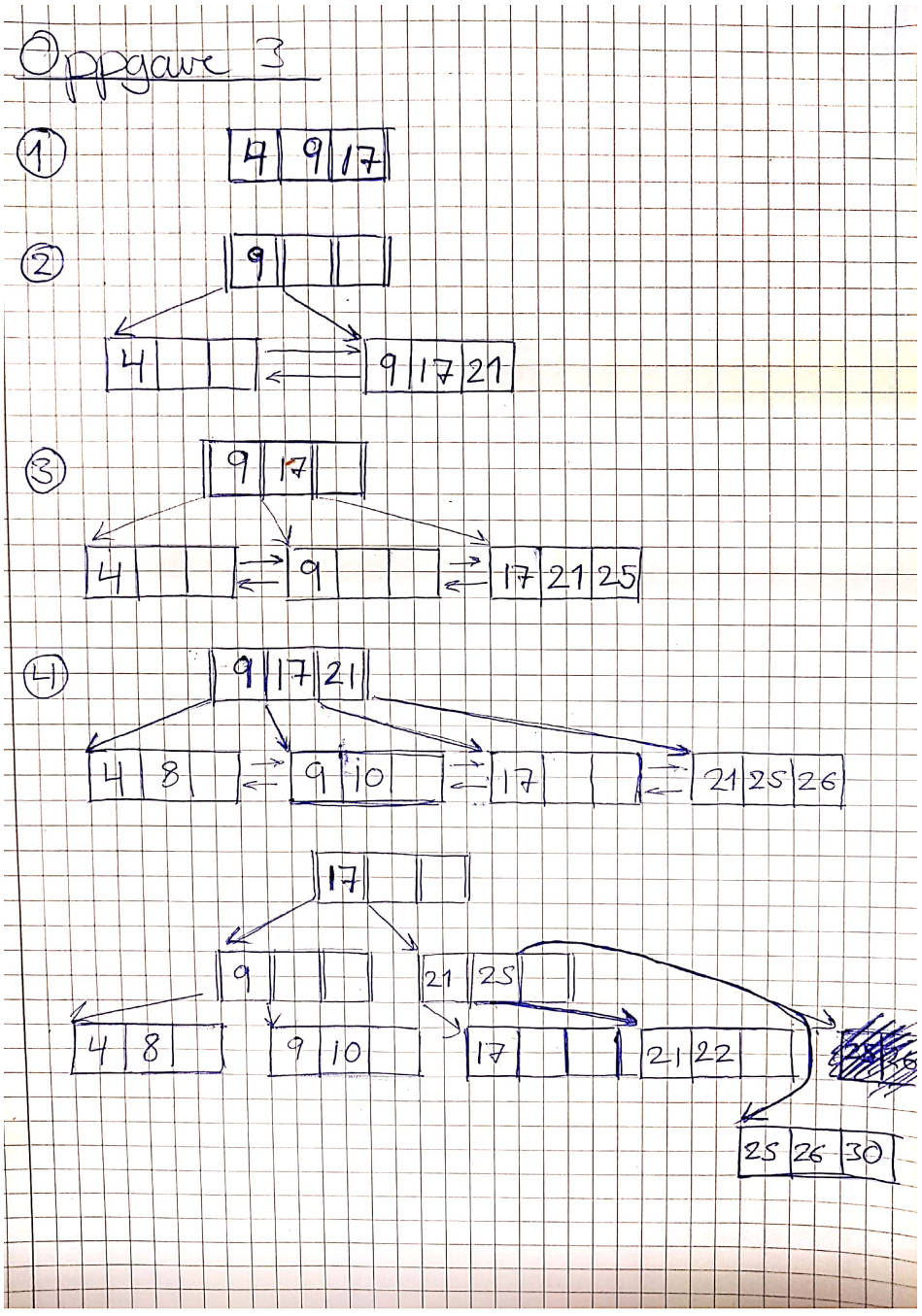

Figure 3: Part 1 of the construction of B+-tree

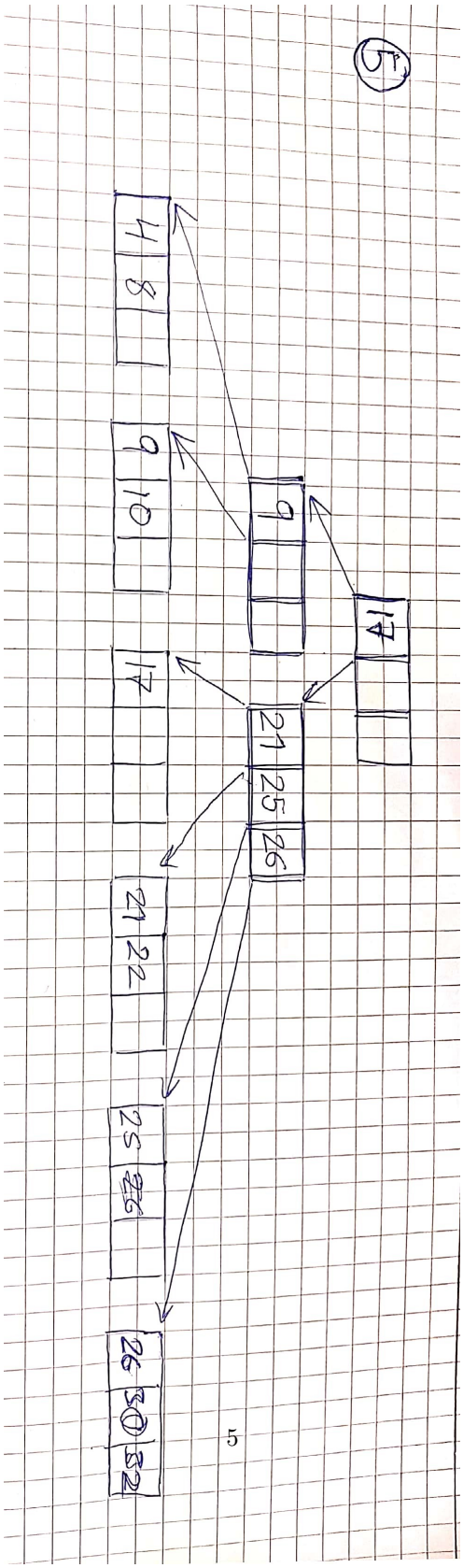

Figure 4: Part 2 of the construction of B+-tree

3. Because the blocks are already ordered, we again have to access $2 + N$ blocks (all of them sequentially from start to finish).

4. In this case we just have to access the two level blocks plus the first blocks containing 5 percent of all the people stored in the tree. With 8 posts per tree, that results in

$$2 + \frac{10000 \cdot 5}{100 \cdot 8} = 2 + 63 = 65 \tag{4}$$

   65 blocks.

Task 5: Queries with heap and unclustered B+ tree

a)

Siden de erlagret i en heapfil på 1250 blokker er det dermed 1250 blokker som akesseres.

b)

Max 302+1250 = 1552 blokker. 2 nedover for å nå løvnivået i treet,en per blokk i løvnodene og en for hver blokk i heapfilen. Lite hensiktsmessig å søke mhp. ID når det er etternavn som er søkenøkkel.

c)

2 nedover i b+ treet for å nå løvnodene + antall blokker med "Søkerud" som søkenøkkel. Dvs. 2+N, hvor N = antall "Søkerud"

d)

Index only query: 2 nedover for å nå løvnodene i b+ treet + 300 bortover. Totalt 300 + 2 = 302

e)

I heapen blir det 1 read + 1 write. I b+ treet blir det 3 read nedover, i tillegg til 1 write. Dette gir en total på 6 blokker.