

Implementation of DBMS

Exercise Sheet 10, Solutions

Klingemann, WS 2024 / 2025

1) Suppose we have a file of 1,000,000 records that we want to hash into a table with 1000 buckets. 100 records will fit in a block, and we wish to keep blocks as full as possible, but not allow two buckets to share a block. Empty buckets do not consume a block. What is the minimum and maximum number of blocks that we need to store this hash table?

Solution:

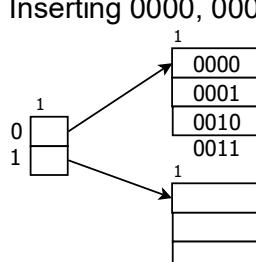
In the best case, each bucket contains a number of records which is a multiple of hundred. In this case each block is completely full. For example, each bucket could contain 1000 records and therefore requires $\lceil 1000 \text{ records} / 100 \text{ records per block} \rceil = 10$ blocks. In total, we therefore need $10 \text{ blocks per bucket} * 1000 \text{ buckets} = 10000$ blocks.

In the worst case, we have in each bucket a block which contains just a single record. For example, we can have 999 buckets with one record, each. Every bucket of this kind consumes one block. The last bucket contains the remaining records and needs $\lceil 999001 \text{ records} / 100 \text{ records per block} \rceil = 9991$ blocks. In total, we therefore need $999 \text{ blocks} + 9991 \text{ blocks} = 10990$ blocks.

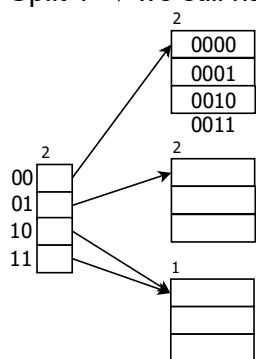
2) Suppose that keys are hashed to four-bit sequences and that blocks can hold three records. If we start with a hash table with two empty blocks (corresponding to 0 and 1), show how the hash table evolves if we insert records with the following hash values: 0000, 0001, ..., 1111, and the method of hashing is extensible hashing.

Solution:

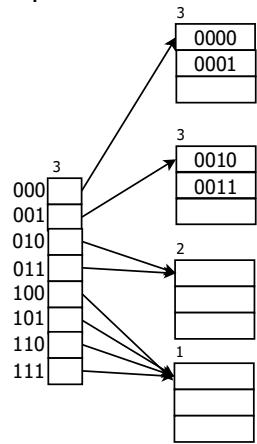
Inserting 0000, 0001, 0010 and 0011 → overflow:



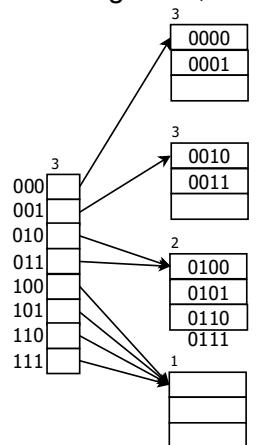
Split 1 → we still have an overflow:



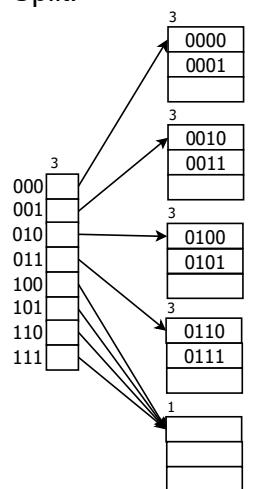
Split 2:



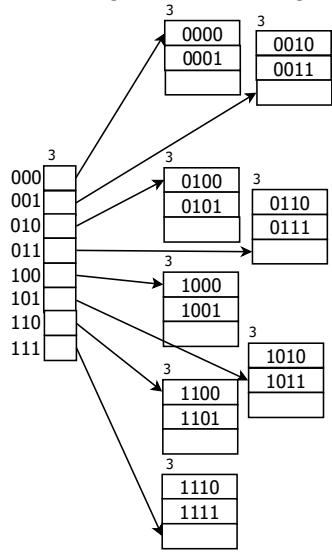
Inserting 0100, 0101, 0110 and 0111 → overflow:



Split:



Inserting the remaining records → the lower part of the table evolves in a similar way:



3) In an extensible hash table with n records per block, what is the probability that an overflow will have to be handled recursively, i.e., all members of the block will go into the same one of the two blocks created in the split?

Solution:

We split a block when we have $n+1$ records for this block. When we split a block, we consider an additional bit to decide to which of the two blocks a record will be assigned. When all $n+1$ records go into the same block, they have the same value for this bit. The value of this bit can be for all records either 0 (with a probability $(\frac{1}{2})^{n+1}$) or 1 (also with a probability $(\frac{1}{2})^{n+1}$). In total, the probability is $(\frac{1}{2})^n$.