

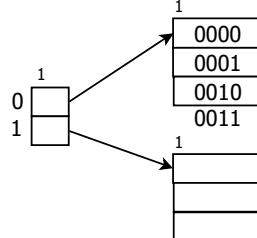
**Implementation of DBMS**  
**Exercise Sheet 10, Solutions**  
**Klingemann, WS 2022 / 2023**

1) 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:

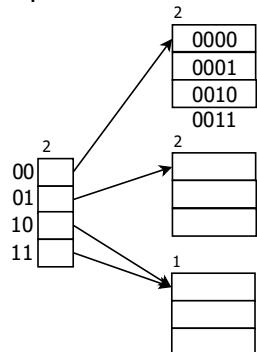
a) 0000, 0001, ..., 1111, and the method of hashing is extensible hashing.

Solution:

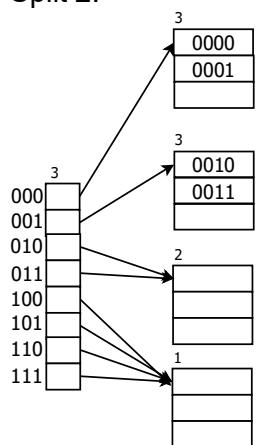
Inserting 0000, 0001, 0010 and 0011 → overflow:



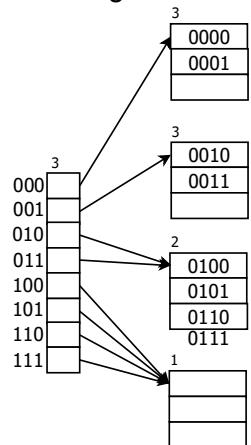
Split 1 → we still have an overflow:



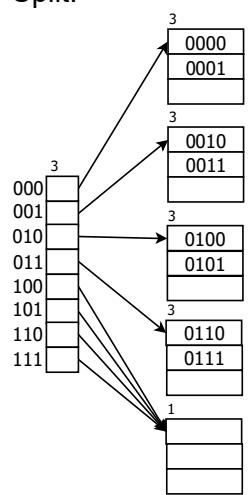
Split 2:



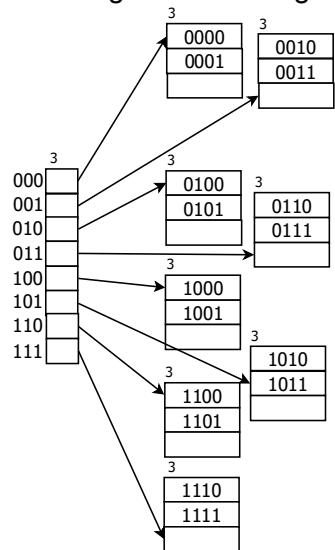
Inserting 0100, 0101, 0110 and 0111 → overflow:



Split:



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



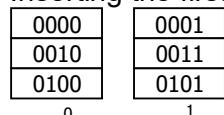
b) 0000, 0001, ..., 1111, and the method of hashing is linear hashing with a capacity threshold of 100%.

Solution:

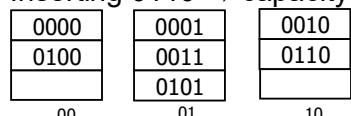
We add buckets as follows:

number of buckets	100% capacity	add bucket when the number of records becomes
2	6	7
3	9	10
4	12	13
5	15	16

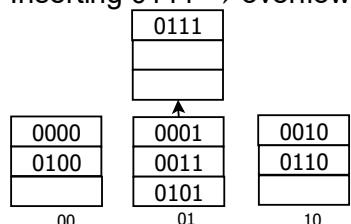
Inserting the first 6 records:



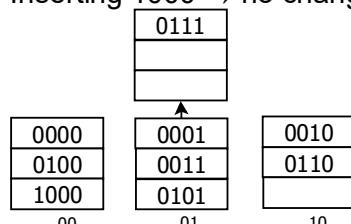
Inserting 0110 → capacity exceeded, new bucket created:



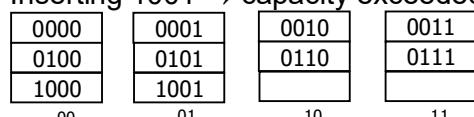
Inserting 0111 → overflow block created:



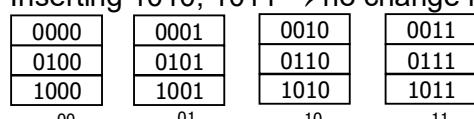
Inserting 1000 → no change in the structure of the table



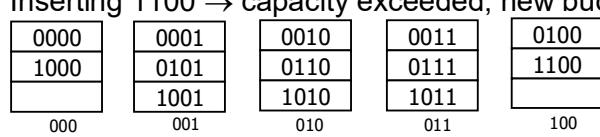
Inserting 1001 → capacity exceeded, new bucket created:



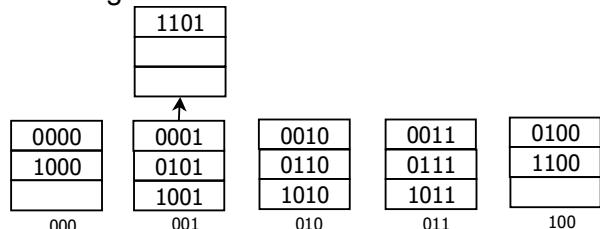
Inserting 1010, 1011 → no change in the structure of the table



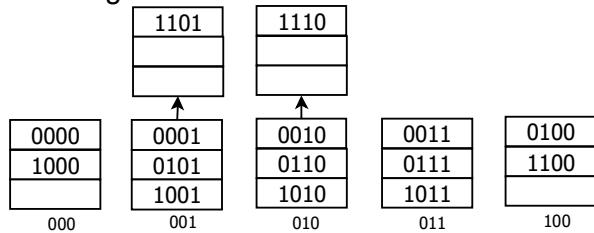
Inserting 1100 → capacity exceeded, new bucket created:



Inserting 1101 → overflow block created:



Inserting 1110 → overflow block created:



Inserting 1111 → capacity exceeded, new bucket created, one overflow block disappears, another one is created:

