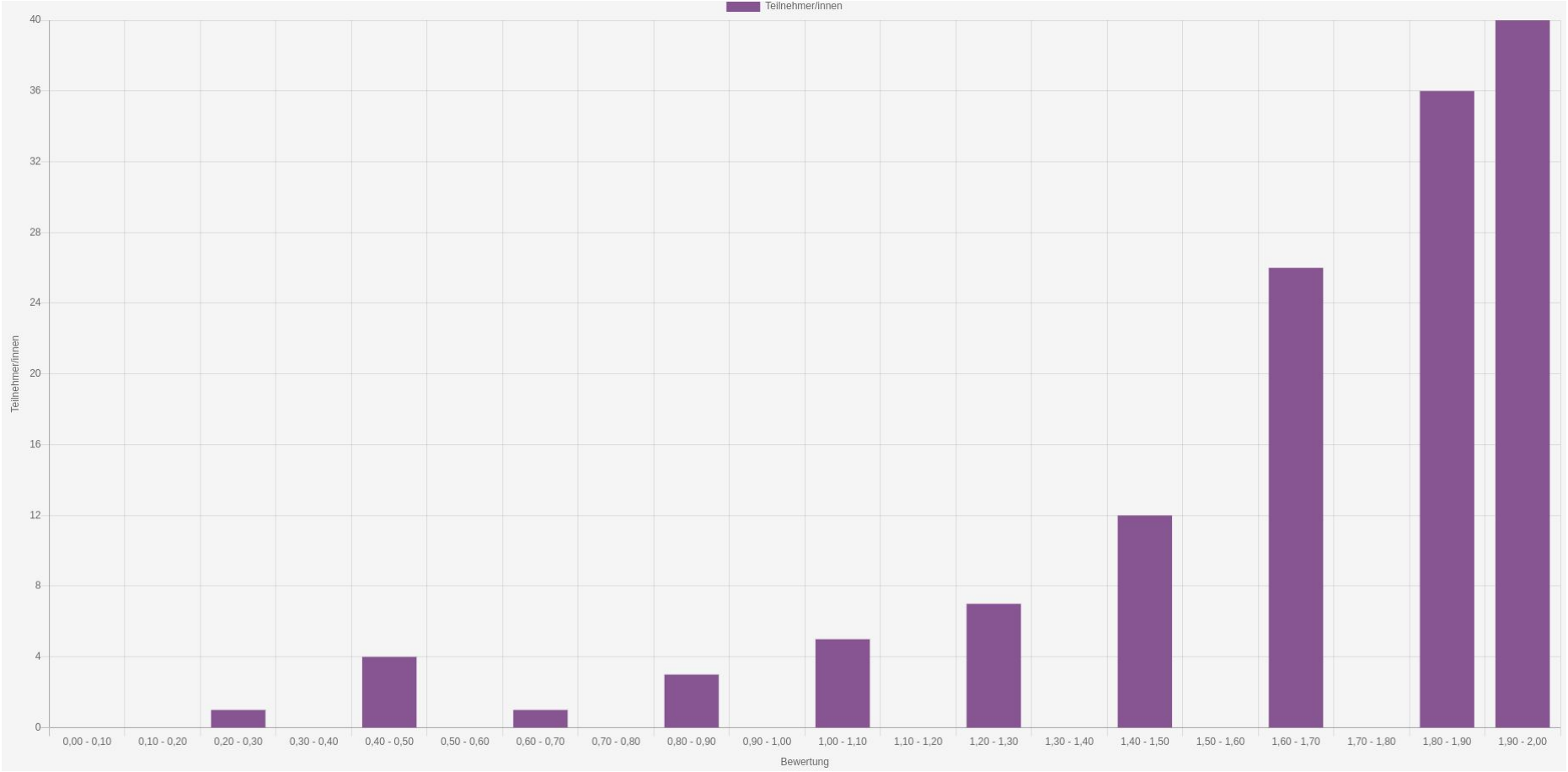
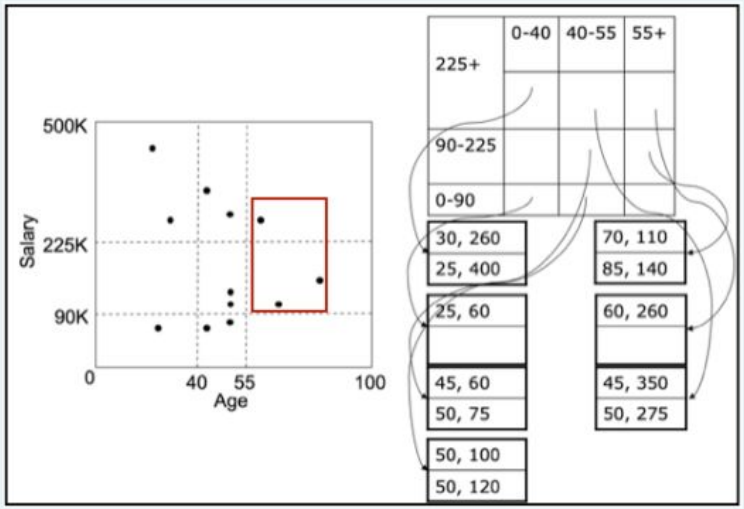
Database Technology

Exercise 5: Review


Teilnehmer/innen
40
36
32
28
24
20
16
12
8
4
0
Teilnehmer/innen
0,00 - 0,10
0,10 - 0,20
0,20 - 0,30
0,30 - 0,40
0,40 - 0,50
0,50 - 0,60
0,60 - 0,70
0,70 - 0,80
0,80 - 0,90
0,90 - 1,00
1,00 - 1,10
1,10 - 1,20
1,20 - 1,30
1,30 - 1,40
1,40 - 1,50
1,50 - 1,60
1,60 - 1,70
1,70 - 1,80
1,80 - 1,90
1,90 - 2,00
Bewertung

Q1: What is average salary of the records that are contained in the query rectangle?

Consider the following hypothetical Grid-file index.



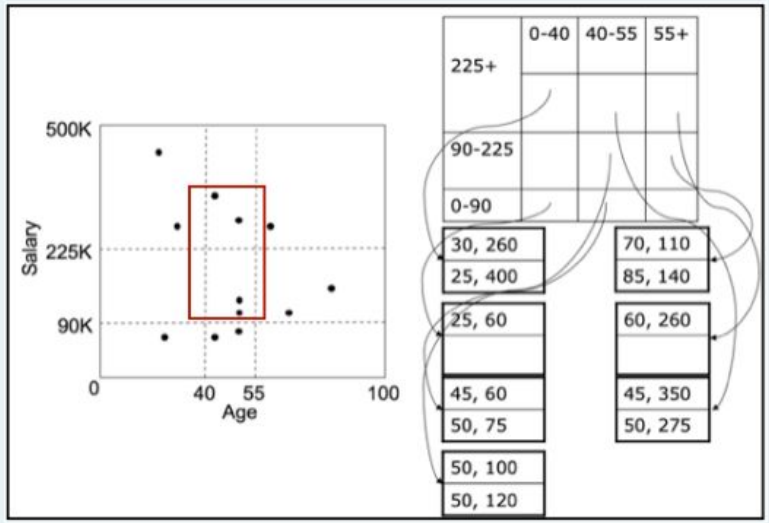
The query is shown as the red rectangle.

Note: Round **up** the answer to the closest decimal number (i.e., if the value you computed is 34.1 the answer will be 35). If the question asks to compute salary the answer will be without K (i.e., if the answer is 225K, the answer you should submit is 225).

(110+140+260)/3 = 170

Q2: Given the linearization order as **horizontal linearization**, the buckets that will be examined to answer the query (shown as the red rectangle in the figure) are:

Consider the following hypothetical Grid-file index.



In this exercise we will linearize the 2D Grid-file index. In Lecture 2, we saw two particular linearization orders: row layout, and column layout. In the row layout, we horizontally visited all the fields of the first record first, and then moved to the next record in the table. In the column layout, we vertically visited the first column of all the records in the table and then moved to second column of the table and so on. For this question, we will refer to these schemes as horizontal linearization (row-layout) and vertical linearization (column-layout). The goal is to number (or enumerate) the buckets as B1, B2, B3 and so on. The linearisation of the grid-file figure will always begin from top-left bucket (i.e., we will first visit the bucket of space salary from 225k to 500k and age from 0 to 40) and this bucket will be numbered B1. The next bucket to be visited will depend on the linearization order. For horizontal linearization we will visit the space [salary -> 225k to 500k and age -> 40 to 55] next (just like in row layout, and this bucket will thus be numbered B2 for horizontal linearization). For vertical linearization we will visit the space [salary 95k to 225k and age 0 to 40] next (and it will be numbered bucket B2). All the buckets will be numbered (or enumerated) even if they are empty.

a. B5, B7, B9, B8, B4, B6

b. B2, B5, B4, B1

c. B3, B4, B6, B5, B1, B2

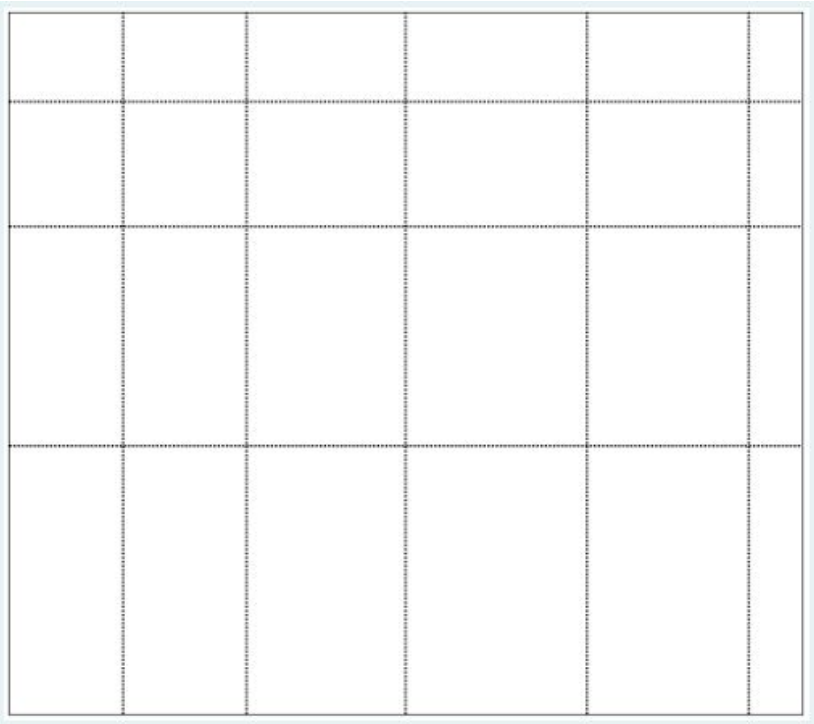
d. B7, B8, B2, B1

Q3: How many buckets have to be checked if the queries only access attribute c?

Consider that we would like to store a relation R(a, b, c, d) using partitioned hashing. The total available buckets are 65536, i.e., the hash function will produce a string of 16 bits. The hash function divides the 16 bits between the three attributes as 4 bits for a, 5 bits for b, 3 bits for c, and 4 bits for d.

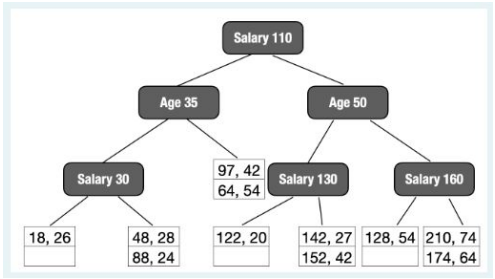
Total number of buckets= 2^16
Number of bits specified = 3
Total buckets to explore = 2^(16-3)

Q4: The index structure shown in the figure below resembles which multi-dimensional index the closest?

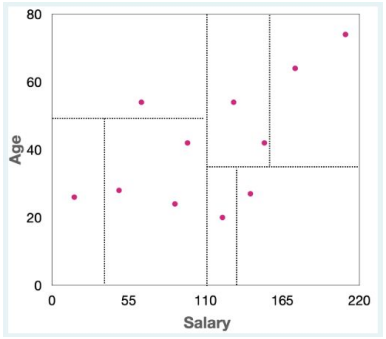


- a. R-tree
- b. kd-tree
- c. Quadtree
- d. Grid-file
- e. B+-tree
- f. ABC+-tree

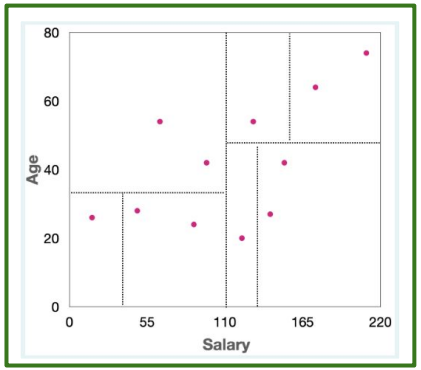
Q5: Given the following classic kd-tree, choose the corresponding implied partitions.



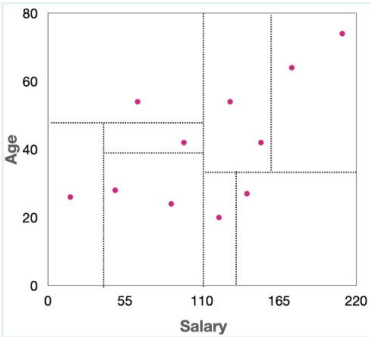
a.



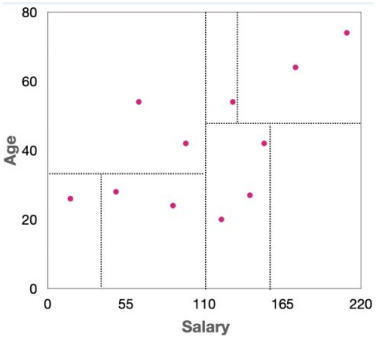
b.



c.

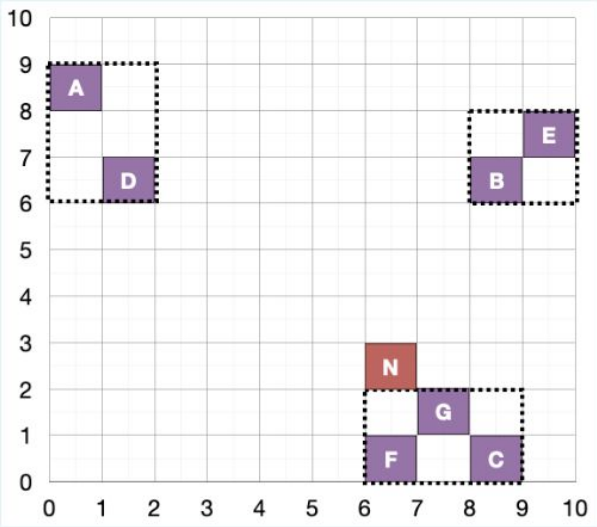


d.



Q6: What is the most viable R-tree after insertion given that the main priority is that the covered area has to be minimized?

Given the following R-tree in the figure below, and a new block N (shown as a red block in the figure) to be inserted in the R-tree:



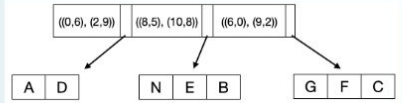
Every node can contain a maximum of three objects.

You may assume that:

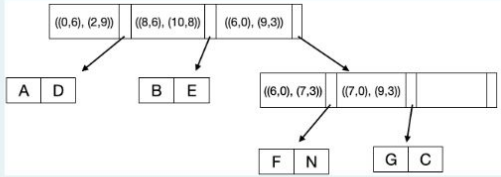
(1) During the insertion process, there is a function available that splits or combines nodes such that the covered area is minimised.

(2) The R-tree can be unbalanced (i.e., not all leaves are at the same level or height)

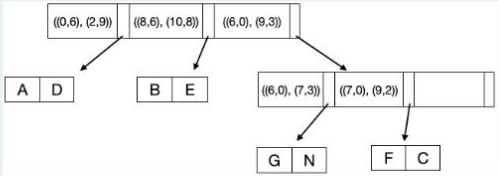
a.



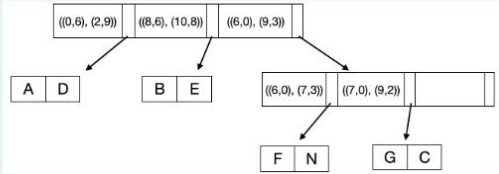
b.



c.



d.



Q7: Quadtrees are always height-balanced.

False

Q8: Following query is the example of _____ .

```
select avg(TotalSpent) from Customer
where Age between 45 and 60 and
     PostalCode between 10629 and 13581
```

Consider the following hypothetical relation
Customer(CustID, Age, Gender, PostalCode, City, TotalSpent).

- a. Partial-Match Query
- b. GIS Query
- c. Point Query
- d. Range Query

Q9: Which of the following options below represents the correct table?

Consider a hypothetical relation:

students(studentID, gender, courseID).

There are two bitmap indexes on the fields gender, and courseID.

The following is the bitmap index on the gender

Male: 010111

Female: 101000

The bitmap index on courseID is as follows:

DBT: 100100

DBT-PRA: 010010

ROC: 000001

DW: 001000

a.

studentID	gender	courseID
1	F	DBT
2	M	DBT-PRA
3	F	ROC
4	M	DBT
5	M	DBT-PRA
6	M	DW

b.

studentID	gender	courseID
1	F	DBT
2	M	DBT-PRA
3	F	DBT
4	M	ROC
5	M	DBT-PRA
6	M	DW

c.

studentID	gender	courseID
1	F	DBT
2	M	DBT-PRA
3	F	DW
4	M	DBT
5	M	DBT-PRA
6	M	ROC

d.

studentID	gender	courseID
1	F	DBT
2	M	DBT-PRA
3	F	DBT
4	M	ROC
5	F	DBT-PRA
6	M	DW

Q10: What will be the size of the bitmap index on Field F in bytes?

Consider a file of 2,500,000 records, with a field *F* that has 3 different values. Assume that the bit vector is always 1-byte aligned (i.e., if bit vector contains 4 bits then the size of bit vector will be 1 byte, if bit vector contains 12 bits then size of the bit vector will be 2 bytes, if the bit vector contains 16 bits then the size of the bit vector will 2 bytes, and so on).

Number of Bits = 2500000
Size of each bit vector = Ceil(2500000/8) = 312500
Number of Distinct values = 3
Total Bitmap index Size = 3*312500 = 937500