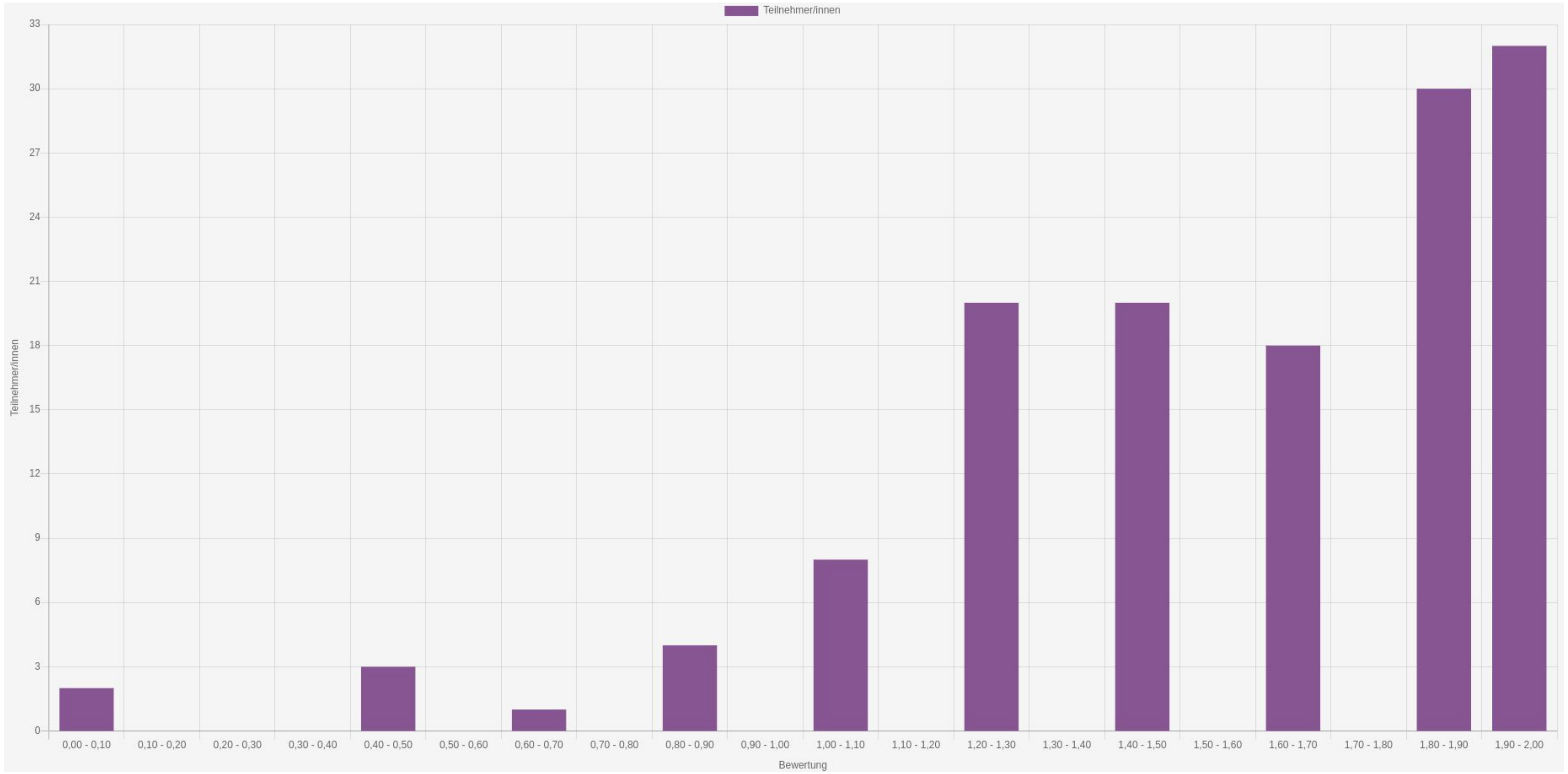
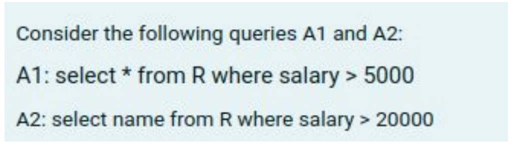
Database Technology

Exercise 3: Review


Teilnehmer/innen
33
30
27
24
21
18
15
12
9
6
3
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: Can A2 be answered from the resulting tuples from A1?



Consider the following queries A1 and A2:

A1: select * from R where salary > 5000

A2: select name from R where salary > 20000

True

Q2: How many cache misses will occur if the cache replacement policy is LRU?

The cache in the system can hold 4 pages in total and is initially empty. Consider the following sequence of page requests to the cache:

15, 4, 6, 15, 9, 11, 6, 9, 12, 1, 9

Additional information:

Cache Hit - A cache hit occurs when a requested page is found in the cache.

Cache Miss - A cache miss occurs when a requested page is not found in the cache.

7

Q3: How many cache hits will occur if the cache replacement policy is FIFO?

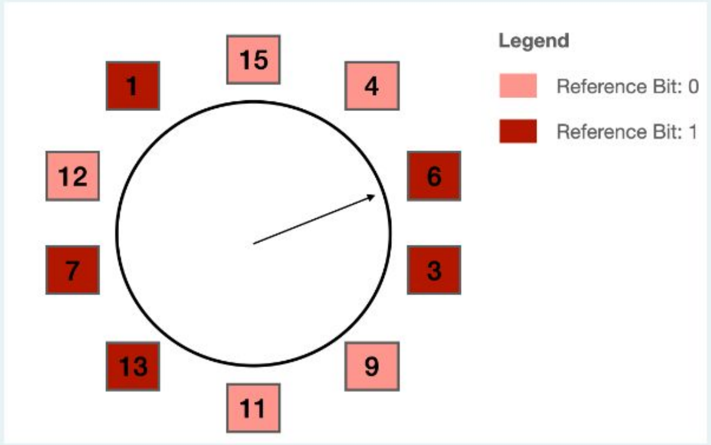
The cache in the system can hold 5 pages in total and is initially empty. Consider the following sequence of page requests to the cache:
15, 4, 6, 15, 9, 11, 9, 7, 12, 1

Additional information:
Cache Hit - A cache hit occurs when a requested page is found in the cache.
Cache Miss - A cache miss occurs when a requested page is not found in the cache.

2

Q4: Which page will be evicted when the last request in the sequence (Page 61) is served?

Consider the CLOCK algorithm. The system cache has space for 10 pages, and the content of the cache (at time t) is shown in the figure below. The clock head points to Page 6.



Assume that when a page is accessed and it is already in the cache, its reference bit is set to 1 (if it is 0 it is changed to 1, if it is 1 then it is left as 1). The clock hand does not move in such a case (i.e., it stays where it was before this access). The clock hand only moves when the accessed page is not in the cache and the cache is looking for a page to evict. Also assume that when a new page is brought into the cache its reference bit is set to 1.

Consider the following sequence of page requests to the cache: 20, 31, 35, 42, 61

4

Q5: A dense index has exactly four entries for each row in any table.

False

Q6: A sparse index has an entry for every row in the table, and therefore it needs less space compared to the dense index.

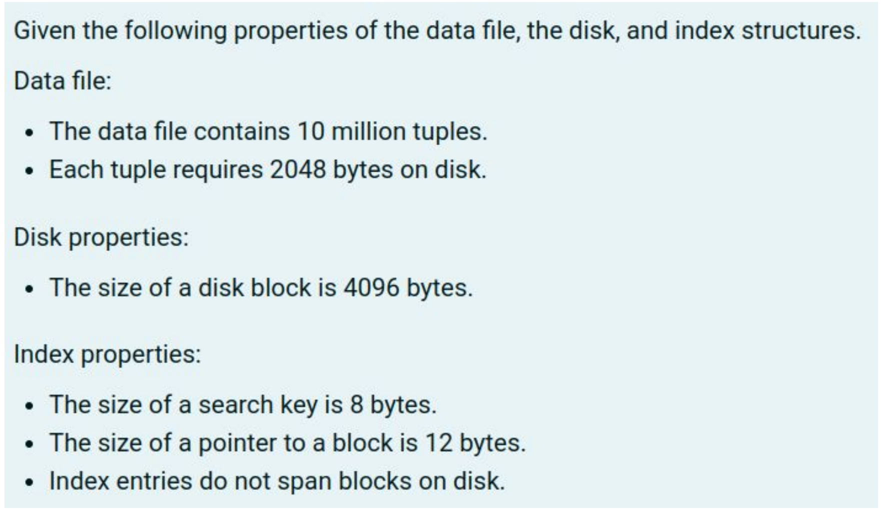
False

Q7: Low selectivity means that the ratio of selected tuples and the total number of tuples is very big.

Additional Information: Here selectivity refers to as the property of the index or query.

True

Q8: How many blocks are required for a dense index on the data file?


Given the following properties of the data file, the disk, and index structures.

Data file:

- The data file contains 10 million tuples.
- Each tuple requires 2048 bytes on disk.

Disk properties:

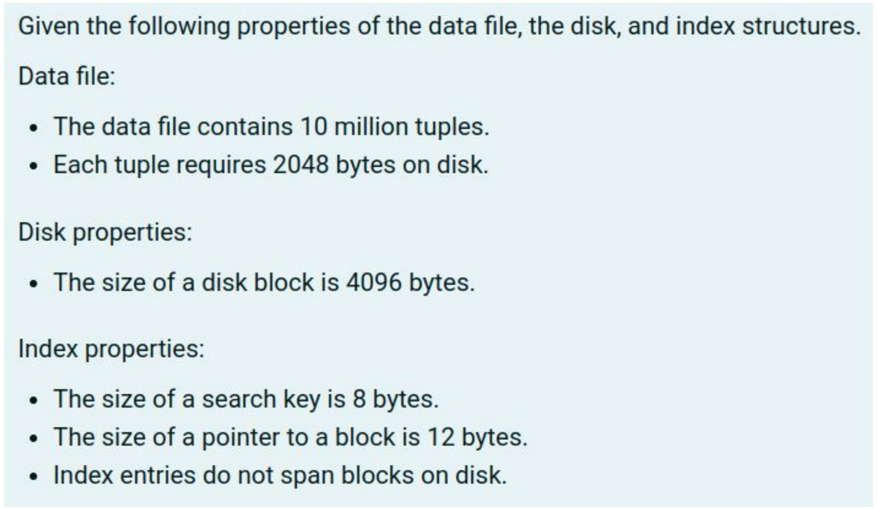
- The size of a disk block is 4096 bytes.

Index properties:

- The size of a search key is 8 bytes.
- The size of a pointer to a block is 12 bytes.
- Index entries do not span blocks on disk.

TotalIndexTuplePerBlock = Floor(4096/(8+12))
Total Block = Ceil(Total Tuple/TotalIndexTuplePerBlock)
= Ceil(10^7/204)
= 49020

Q9: How many blocks are required for a sparse index on the data file?



Given the following properties of the data file, the disk, and index structures.

Data file:

- The data file contains 10 million tuples.
- Each tuple requires 2048 bytes on disk.

Disk properties:

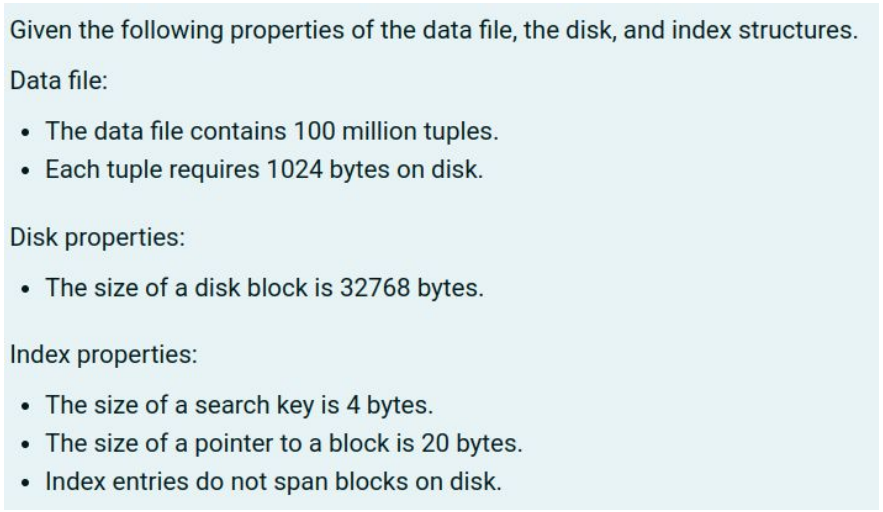
- The size of a disk block is 4096 bytes.

Index properties:

- The size of a search key is 8 bytes.
- The size of a pointer to a block is 12 bytes.
- Index entries do not span blocks on disk.

TotalIndexTuplePerBlock = **Floor**(4096/(8+12))=204
TuplePerBlock = **Floor**(BlockSize/TupleSize)
NumOfBlockToStoreTuples = **Ceil**(TotalTuple/TuplePerBlock)
TotalIndexBlock =
Ceil(NumOfBlockToStoreTuples/TotalIndexTuplePerBlock)
= **Ceil**($5*10^6$/204)
= 24510

Q10: How many blocks are required for a second-level sparse index on the data file, if the first-level index is also sparse?


Given the following properties of the data file, the disk, and index structures.

Data file:

- The data file contains 100 million tuples.
- Each tuple requires 1024 bytes on disk.

Disk properties:

- The size of a disk block is 32768 bytes.

Index properties:

- The size of a search key is 4 bytes.
- The size of a pointer to a block is 20 bytes.
- Index entries do not span blocks on disk.

TotalIndexTuplePerBlock = **Floor**(32768/(4+24))=1365
TuplePerBlock = **Floor**(BlockSize/TupleSize)=32
NumOfBlockToStoreTuples = **Ceil**(TotalTuple/TuplePerBlock)
=3125000
TotalFirstLvlIndexBlock =
Ceil(NumOfBlockToStoreTuples/TotalIndexTuplePerBlock)
= **Ceil**(3125000/1365)
= 2290

TotalSecondLvlIndexBlock=
Ceil(TotalFirstLvlIndexBlock/TotalIndexTuplePerBlock)
=Ceil(2290/1365)
=2