$$R \bowtie \underbrace{\underbrace{\sigma_{B=5 \text{ and } C=10}(S)}_{U}}_{V}$$

$$T(U) = \frac{T(S)}{V(S,C)} = \frac{200}{10} = 20$$

as B is not involved in sealection condition:

$$V(U, b) = V(S, b) = 40$$

$$T(V) = \frac{T(U)}{V(U,b)} = \frac{20}{40} = \frac{1}{2} \rightsquigarrow T(Z)$$

final tuples: $R \bowtie Z$: assume as Z, rightside

$$\frac{T(R) \times T(Z)}{\max(\underbrace{V(R,b)}_{30}, \underbrace{V(Z,b)}_{40})}$$

$$= \frac{1000 * \frac{1}{2}}{40}$$

$$= \frac{50}{4}$$

$$= \boxed{\frac{25}{2}} \text{ tuples}$$

$$= \text{ceil}[12.5]$$

$$= \boxed{13} \text{ tuples}$$

7) (10 points)

We have three relations with the following statistics:

- B(Order) = 3000 blocks
- B(Cust) = 1000 blocks
- B(Book) = 100 blocks

Assume that the relations are stored contiguously, i.e., they are clustered relations. You should further assume that each operation uses memory efficiently. You can ignore final output I/O cost.

a) We want to perform a selection of "Price<10" over Book. We have 10 blocks of main-memory. What is the required number of I/Os?
b) We want to perform a one pass join of Order and Cust. How many main-memory-blocks do you need?
c) We want to perform a one pass join of Order and Cust and have sufficient main-memory. What is the required number of I/Os?
d) We want to perform a hash join (without the "hybrid" optimization) of Order and Book. How many main-memory-blocks do you need?
e) We want to perform a hash join (without the "hybrid" optimization) of Order and Book and have sufficient main-memory. What is the required number of I/Os?

6)

Let R(A, B) and S(B, C) be relations with the following statistics:

- T(R) = 50
- T(S) = 100
- V(R, A) = 5
- V(R, B) = 5
- V(S, B) = 10
- V(S, C) = 20

Estimate the number of tuples of the following expression:

$\sigma_{A=1}(R) \bowtie S$

R(A, B), S(B, C), T(D, E)

T(R) = 1000   V(R,A) = 1   V(R,B) = 30

T(S) = 200   V(S,B) = 40   V(S,E) = 10

T(T) = 10000   V(T,D) = 10   V(T,E) = 20

Estimate

$R \bowtie \sigma_{B=5 \text{ and } C=10}(S)$

(6 points)

4)
Suppose we store relation R(a, b, c) in a partitioned hash table with 1024 buckets. That is, the hash function produces 10 bits. Queries about R all specify exactly one of the attributes a, b, or c, and each of the attributes is equally likely to be specified. The hash function produces 5 bits based only on the value of a, 3 bits based only on the value on b and 2 bits based only on c. What is the average number of buckets that must be searched to find matching tuples?

(6 points)

3)
Consider B+-trees of order 2. Give an example of a B+-tree with three levels whose set of keys could alternatively be represented in a B+-tree with two levels. Your example should consist of two trees, one with three levels and the equivalent one with two levels. Your trees should show all keys and pointers.

(6 points)

2)
Consider the following situation: Blocks are 1000 bytes long. There is no need for a block header. Records are 100 bytes long, of which 12 bytes are the key field. Pointers take 8 bytes. A sequential file (sorted by the key field) consists of 10,000 records. Each block of the file contains as many records as possible.

a) What is the minimum number of blocks required for a dense index on this file?

b) What is the minimum number of blocks required for a sparse index on this file?

Note: *Carefully justify your answers!* Answers can only be accepted if the approach to the solution is explained!

1)

(6 points)

Consider a sorted data file of initially n records. Suppose that we handle insertions into the file by creating overflow blocks as needed. Also, suppose that the data blocks are currently all half full, i.e., we have not yet used overflow blocks. If we insert new records at random, how many records do we have to insert before the average number of data blocks (including overflow blocks if necessary) that we need to examine to find a rec with a given key reaches 3? Assume that on a lookup, we search the block pointed to by an index first, and c search overflow blocks, in order, until we find the record, which is definitely in one of the blocks of the cha

5) (6 points)

Relation R(a, b) is stored in 200 blocks, with 100 records (tuples) of R fitting in one block. Relation S(c, d) is stored in 100 blocks with 50 records (tuples) per block. Assume that the relations are clustered i.e., are stored contiguously. We want to compute

$$R \bowtie_{b=c} S$$

and find that each tuple of R joins with five tuples of S. How many blocks does the result of this equi-join require? (Note, that this is not a natural join!) You can ignore overhead resulting from record splits.

ı linear hash table with a capacity threshold of 80%. Suppose that keys are hashed to four-bit
locks can hold two records.
at is the minimum number of records we have to insert to get a hash table with 5 buckets?
vide an example of a hash table with exactly 4 buckets and 2 overflow blocks. Draw the t
: with their hash values.

Use a linear hash table with a capacity threshold of 80%. Suppose that keys are hashed to four-bit sequences and that blocks can hold two records.

a) What is the minimum number of records we have to insert to get a hash table with 5 buckets?

b) Provide an example of a hash table with exactly 4 buckets and 2 overflow blocks. Draw the records along with their hash values.

We have two relations R(A, B) and S(B, C) with the following statistics:

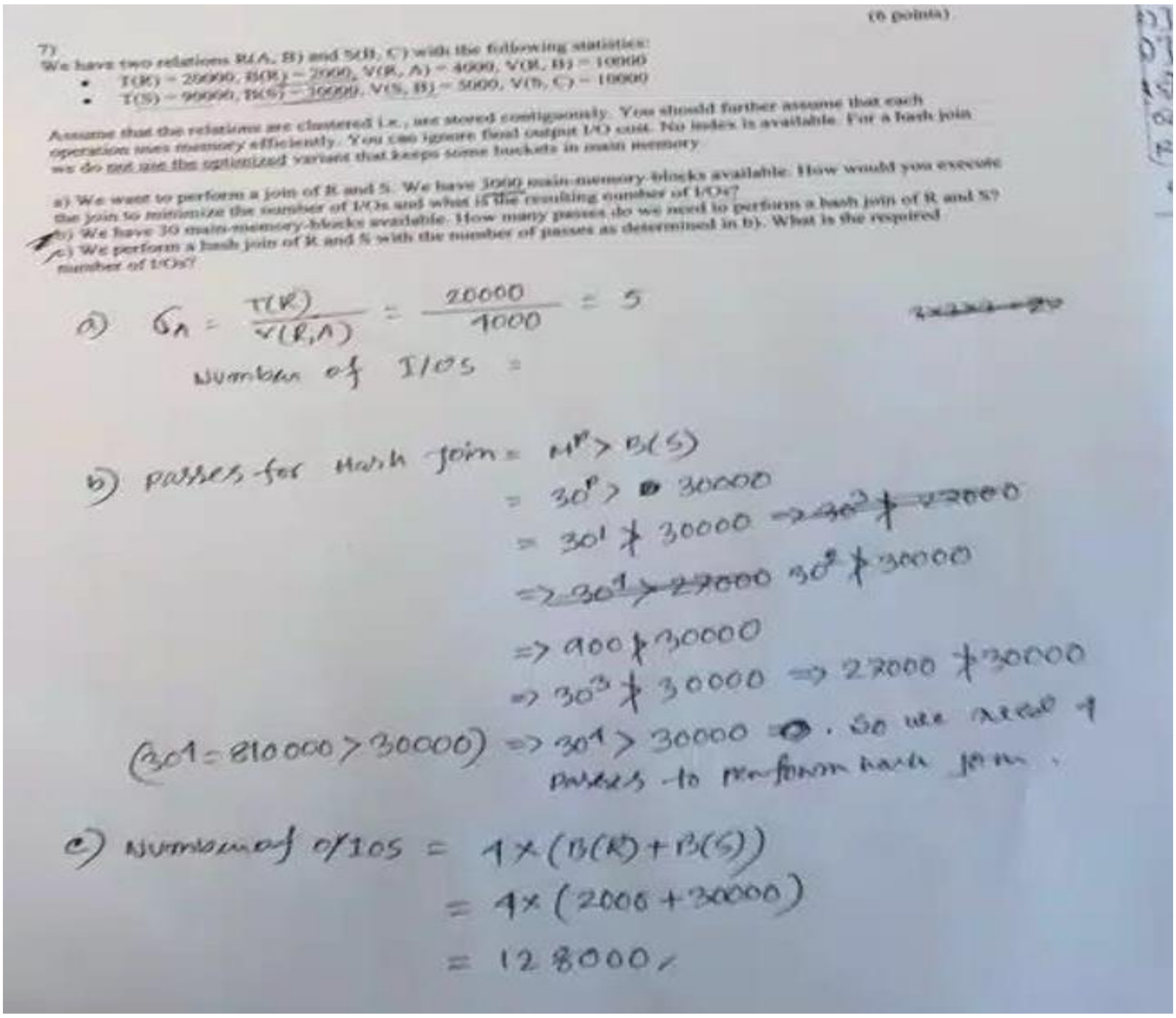
- T(R) = 20000, B(R) = 2000, V(A, R) = 4000, V(B, R) = 10000
- T(S) = 30000, B(S) = 3000, V(B, S) = 5000, V(C, S) = 10000

Assume that the relations are clustered (i.e., stored contiguously). You should further assume that each operation produces sorted output. You can ignore final output I/O cost. No index is available. For a hash join, assume we use **two hash functions** in main memory.

a) We want to perform a join of R and S. We have 3000 main memory blocks available. How would you execute the join so as to minimize the number of I/Os? What is the resulting number of I/Os?

b) We use 30 main-memory blocks available. How many passes do we need to perform a hash join of R and S?

c) We perform a hash join of R and S with the number of passes as determined in b). What is the required number of I/Os?



(6 points)

7) We have two relations R(A, B) and S(B, C) with the following statistics:

- T(R) = 20000, B(R) = 2000, V(R, A) = 4000, V(R, B) = 10000
- T(S) = 90000, B(S) = 30000, V(S, B) = 5000, V(S, C) = 10000

Assume that the relations are clustered i.e., are stored contiguously. You should further assume that each operation uses memory efficiently. You can ignore final output I/O cost. No index is available. For a hash join we do not use the optimized variant that keeps some buckets in main memory.

a) We want to perform a join of R and S. We have 3000 main-memory blocks available. How would you execute the join to minimize the number of I/Os and what is the resulting number of I/Os?

b) We have 30 main-memory blocks available. How many passes do we need to perform a hash join of R and S?

c) We perform a hash join of R and S with the number of passes as determined in b). What is the required number of I/Os?

a) $\sigma_A = \frac{T(R)}{V(R,A)} = \frac{20000}{4000} = 5$

Number of I/Os =

b) passes for Hash join = $M^P > B(S)$

$= 30^P > 30000$

$= 30^1 \ngtr 30000$

$\Rightarrow 30^2 \ngtr 30000$

$\Rightarrow 900 \ngtr 30000$

$\Rightarrow 30^3 \ngtr 30000 \Rightarrow 27000 \ngtr 30000$

$(30^4 = 810000 > 30000) \Rightarrow 30^4 > 30000$. So we need 4 passes to perform hash join.

c) Number of I/Os $= 4 \times (B(R) + B(S))$

$= 4 \times (2000 + 30000)$

$= 128000$

"**A linear hash table** with a capacity threshold of 80%. Suppose that keys are hashed to four-bit blocks can hold two records.
What is the minimum number of records we have to insert to get a hash table with 5 buckets?
Provide an example of a hash table with exactly 4 buckets and 2 overflow blocks. Draw the records with their hash values."