# Data Systems Group Assignment 3

**Team Members:** Harsh Sinha (20171010), Rishabh Singhal (20171213)

## Information about Relations

- **Relation R:**
  1. Total number of bytes taken by one row, $\mathbf{BR_R} = 32 + 24 + 28 + 4 + 16 + 192\ bytes =$ **296 bytes**.
  2. Total number of rows $\mathbf{NR_R} = 2^{20}$.
  3. Total number of bytes for whole relation $\mathbf{SZ_R} = 2^{20} \times 296$ bytes.
  4. Number of rows per disk block (512 bytes) $= \lfloor \frac{512}{296} \rfloor = 1$
  5. Number of blocks (512 bytes) $= \lceil \frac{2^{20}}{1} \rceil = 2^{20}$
  6. Number of rows per disk block (4096 bytes) $= \lfloor \frac{4096}{296} \rfloor = 13$
  7. Number of blocks (4096 bytes) $= \lceil \frac{2^{20}}{13} \rceil = 80660$
  8. Number of rows per disk block (32768 bytes) $= \lfloor \frac{32768}{296} \rfloor = 110$
  9. Number of blocks (32768 bytes) $= \lceil \frac{2^{20}}{110} \rceil = 9533$
- **Relation S:**
  1. Total number of bytes taken by one row, $\mathbf{BR_S} = 16+24+28+4+188\ bytes =$ **260 bytes**.
  2. Total number of rows $\mathbf{NR_S} = 2^{8}$.
  3. Total number of bytes for whole relation $\mathbf{SZ_S} = 2^{8} \times 260$ bytes.
  4. Number of rows per disk block (512 bytes) $= \lfloor \frac{512}{260} \rfloor = 1$
  5. Number of blocks (512 bytes) $= \lceil \frac{2^{8}}{1} \rceil = 2^{8}$
  6. Number of rows per disk block (4096 bytes) $= \lfloor \frac{4096}{260} \rfloor = 15$
  7. Number of blocks (4096 bytes) $= \lceil \frac{2^{8}}{15} \rceil = 18$
  8. Number of rows per disk block (32768 bytes) $= \lfloor \frac{32768}{260} \rfloor = 126$
  9. Number of blocks (32768 bytes) $= \lceil \frac{2^{8}}{126} \rceil = 3$

## Questions

**Question a:** R and S are un-ordered files, determine the number of block accesses for the retrieval.

1. $KR = val$
2. $KR > val$
3. $S.A = val$

4. $S.A > val$

**Answer a:**

1. $KR = val$
   As records are unordered this would require sequential scan. So number of block accesses is $\frac{N}{2}$ on an average, where N is total number of blocks.

   - 512 bytes
     $N = 2^{20} => Number\ of\ block\ accesses = \lceil \frac{N}{2} \rceil = \lceil \frac{2^{20}}{2} \rceil = 2^{19} = 524288$
   - 4096 bytes
     $N = 80660 => Number\ of\ block\ accesses = \lceil \frac{N}{2} \rceil = \lceil \frac{80660}{2} \rceil = 40330$
   - 32768 bytes
     $N = 9533 => Number\ of\ block\ accesses = \lceil \frac{N}{2} \rceil = \lceil \frac{9533}{2} \rceil = 4766$

2. $KR > val$
   As records are unordered, entire record needs to be scanned sequentially as rows satisfying this condition can lie anywhere in the record. So number of block accesses is $N$, where $N$ is total number of blocks.

   - 512 bytes
     $N = 2^{20} => Number\ of\ block\ accesses = 2^{20} = 1048576$
   - 4096 bytes
     $N = 80660 => Number\ of\ block\ accesses = 80660$
   - 32768 bytes
     $N = 9533 => Number\ of\ block\ accesses = 9533$

3. $S.A = val$
   As records are unordered, entire record needs to be scanned sequentially as rows satisfying this condition can lie anywhere in the record. So number of block accesses is $N$, where $N$ is total number of blocks.

   - 512 bytes
     $N = 2^{8} => Number\ of\ block\ accesses = 2^{8} = 256$
   - 4096 bytes
     $N = 18 => Number\ of\ block\ accesses = 18$
   - 32768 bytes
     $N = 3 => Number\ of\ block\ accesses = 3$

4. $S.A > val$
   As records are unordered, entire record needs to be scanned sequentially as rows satisfying this condition can lie anywhere in the record. So number of block accesses is $N$, where $N$ is total number of blocks.

- 512 bytes
  $N = 2^8 => Number\ of\ block\ accesses = 2^8 = 256$
- 4096 bytes
  $N = 18 => Number\ of\ block\ accesses = 18$
- 32768 bytes
  $N = 3 => Number\ of\ block\ accesses = 3$

**Question b:** R and S are ordered files, determine the number of block accesses for the retrieval

1. $S.KS = val$
2. $S.KS > val$
3. $R.B = val$
4. $R.B > val$

**Answer b:**
**Assumption:** $R, S$ are ordered on $S.KS, R.B$.

1. $S.KS = val$
   As $S$ is ordered and $KS$ is the key (which means that, $KS$ have distinct values) and hence at max only 1 record exists with $S.KS = val$. So, the number of average block accesses is $\log_2 N$ (using binary search), where $N$ is the total number of blocks.

   - 512 bytes
     $N = 2^8 => Number\ of\ block\ accesses = \lceil \log_2 2^8 \rceil = 8$
   - 4096 bytes
     $N = 18 => Number\ of\ block\ accesses = \lceil \log_2 18 \rceil = 5$
   - 32768 bytes
     $N = 3 => Number\ of\ block\ accesses = \lceil \log_2 3 \rceil = 2$

2. $S.KS > val$
   For this case, one can once find the position of the required record (with value $val$) as $S$ is ordered on $KS$ which is the key and then it will be a linear scan from then till the end (to obtain all the record with $S.KS > val$). **Now assuming the worst case, it can happen that all the records have S.KS > val, in that case it will be a linear scan.**

   Also, note that, let's say number of records where $S.KS > val$ be $N_p$, then the number of disk block access will be $= \lceil \log_2 N \rceil + \lceil \frac{N_p}{bfr} \rceil$.

   - 512 bytes
     $N = 2^8 => Number\ of\ block\ accesses = \lceil log_2 2^8 \rceil + \lceil \frac{N_p}{1} \rceil$
     $=> 8 + \lceil \frac{N_p}{1} \rceil$
     $Average\ value = 8 + \lceil \frac{\sum_{n=0}^{2^8} \lceil \frac{n}{1} \rceil}{1+2^8} \rceil = 136$

- 4787 bytes
  $N = 18 => Number\ of\ block\ accesses = \lceil log_2 18 \rceil + \lceil \frac{N_p}{15} \rceil$
  $=> 5 + \lceil \frac{N_p}{15} \rceil$
  $Average\ value = 5 + \lceil \frac{\sum_{n=0}^{2^8} \lceil \frac{n}{15} \rceil}{1+2^8} \rceil = 14$
- 32768 bytes
  $N = 3 => Number\ of\ block\ accesses = \lceil log_2 3 \rceil + \lceil \frac{N_p}{126} \rceil$
  $=> 2 + \lceil \frac{N_p}{126} \rceil$
  $Average\ value = 2 + \lceil \frac{\sum_{n=0}^{2^8} \lceil \frac{n}{126} \rceil}{1+2^8} \rceil = 4$

But (as we don't know what $N_p$ is, the number of block accesses will be **N** in worst case (where all records are $> val$ and $N$ is total number of blocks.)

- 512 bytes
  $N = 2^8 => Number\ of\ block\ accesses = 2^8$
- 4096 bytes
  $N = 18 => Number\ of\ block\ accesses = 18$
- 32768 bytes
  $N = 3 => Number\ of\ block\ accesses = 3$ (note this is less then average value)

3. $R.B = val$
   As the given field is the ordered-field, therefore requires a sequential scan and as it is $2^{10}$-uniform there exist 2 rows having same $val$. So number of block accesses is $\lceil log_2 N \rceil + \lceil \frac{Number\ of\ instances\ of\ R.B=val}{bfr} \rceil$, where N is total number of blocks. Where first binary search is used to find the first record with $R.B = val$, then the consecutive rows are selected which have same $R.B = val$

   - 512 bytes
     $N = 2^{20} => Number\ of\ block\ accesses = \lceil log_2 2^{20} \rceil + \lceil \frac{2}{1} \rceil = 20 + 2 = 22$ (it can be changed to **21 block accesses** because, after searching for 1 block, the next block is adjacent (1 more block access) so $20 + 1 = 21$)
   - 4096 bytes
     $N = 80660 => Number\ of\ block\ accesses = \lceil log_2 80660 \rceil + \lceil \frac{2}{13} \rceil = 17 + 1 = 18$
   - 32768 bytes
     $N = 9533 => Number\ of\ block\ accesses = \lceil log_2 9533 \rceil + \lceil \frac{2}{110} \rceil = 14 + 1 = 15$

4. $R.B > val$
   As the given field is the ordered-field. First, using binary search first instance of $R.B = val$ is found out, then all the subsequent rows are chosen who's $R.B > val$ So number of block accesses is $\lceil log_2 N \rceil + \lceil \frac{Number\ of\ rows\ with\ R.B>val}{bfr} \rceil$, where $N$ is total number of blocks.

   - 512 bytes
     $N = 2^{20} => Number\ of\ block\ accesses = \lceil log_2 2^{20} \rceil + \lceil \frac{Number\ of\ rows\ with\ R.B>val}{1} \rceil$
     $=> 20 + \lceil \frac{Number\ of\ rows\ with\ R.B>val}{1} \rceil$
     $Average\ value = 20 + \lceil \frac{\sum_{n=0}^{2^{20}} \lceil \frac{n}{1} \rceil}{1+2^{20}} \rceil = 524308$

- 4787 bytes
  $N = 80660 => Number\ of\ block\ accesses = \lceil log_2 80660 \rceil + \lceil \frac{Number\ of\ rows\ with\ R.B>val}{13} \rceil$
  $=> 17 + \lceil \frac{Number\ of\ rows\ with\ R.B>val}{13} \rceil$
  $Average\ value = 17 + \lceil \frac{\sum_{n=0}^{2^{20}} \lceil \frac{n}{13} \rceil}{1+2^{20}} \rceil = 40348$
- 32768 bytes
  $N = 9533 => Number\ of\ block\ accesses = \lceil log_2 9533 \rceil + \lceil \frac{Number\ of\ rows\ with\ R.B>val}{110} \rceil$
  $=> 14 + \lceil \frac{Number\ of\ rows\ with\ R.B>val}{110} \rceil$
  $Average\ value = 14 + \lceil \frac{\sum_{n=0}^{2^{20}} \lceil \frac{n}{110} \rceil}{1+2^{20}} \rceil = 4781$

**Question c:** Size in the number of blocks of Primary Index

1. On $R.KR$ and the number of block accesses for $R.KR = val$.
2. On $S.KS$ and the number of block accesses for $S.KS = val$.

**Answer c:**

1. On $R.KR$ and the number of block accesses for $R.KR = val$.

- 512 bytes
  Size of each entry in PI = Length of $KR$ + Block Ptr length = $32 + 8\ bytes = 40 bytes$
  Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{40} \rfloor = 12$
  Number of blocks in PI = $\lceil \frac{Number\ of\ blocks\ in\ R}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^{20}}{12} \rceil = 87382$
  (PI in sparse index, so 1 entry per block of data)
  **Number of block accesses for R.KR = val**
  $=>$ Block accesses in Primary Index $+ 1$
  $=> \lceil \log_2 Number\ of\ blocks\ in\ PI \rceil + 1$
  $=> \lceil \log_2 87382 \rceil + 1 = 17 + 1 = 18$
- 4096 bytes
  Size of each entry in PI = Length of $KR$ + Block Ptr length = $32 + 8\ bytes = 40 bytes$
  Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{40} \rfloor = 102$
  Number of blocks in PI = $\lceil \frac{Number\ of\ blocks\ in\ R}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{80660}{102} \rceil = 791$
  (PI in sparse index, so 1 entry per block of data)
  **Number of block accesses for R.KR = val**
  $=>$ Block accesses in Primary Index $+ 1$
  $=> \lceil \log_2 Number\ of\ blocks\ in\ PI \rceil + 1$
  $=> \lceil \log_2 791 \rceil + 1 = 10 + 1 = 11$
- 32768 bytes
  Size of each entry in PI = Length of $KR$ + Block Ptr length = $32 + 8\ bytes = 40 bytes$
  Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{40} \rfloor = 819$
  Number of blocks in PI = $\lceil \frac{Number\ of\ blocks\ in\ R}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{9533}{819} \rceil = 12$
  (PI in sparse index, so 1 entry per block of data)
  **Number of block accesses for R.KR = val**

=> Block accesses in Primary Index $+ 1$
=> $\lceil \log_2 Number\ of\ blocks\ in\ PI \rceil + 1$
=> $\lceil \log_2 12 \rceil + 1 = 4 + 1 = 5$

2. On $S.KS$ and the number of block accesses for $S.KS = val$

   - 512 bytes
     Size of each entry in PI = Length of $KS$ + Block Ptr length $= 16 + 8\ bytes = 24 bytes$
     Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{24} \rfloor = 21$
     Number of blocks in PI $= \lceil \frac{Number\ of\ blocks\ in\ S}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{21} \rceil = 13$
     (PI in sparse index, so 1 entry per block of data)
     **Number of block accesses for S.KS = val**
     => Block accesses in Primary Index $+ 1$
     => $\lceil \log_2 Number\ of\ blocks\ in\ PI \rceil + 1$
     => $\lceil \log_2 13 \rceil + 1 = 4 + 1 = 5$
   - 4096 bytes
     Size of each entry in PI = Length of $KS$ + Block Ptr length $= 16 + 8\ bytes = 24 bytes$
     Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{24} \rfloor = 170$
     Number of blocks in PI $= \lceil \frac{Number\ of\ blocks\ in\ S}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{18}{170} \rceil = 1$
     (PI in sparse index, so 1 entry per block of data)
     **Number of block accesses for S.KS = val**
     => Block accesses in Primary Index $+ 1$
     => $1 + 1 = 2$(As there is only one block in PI, there is no point of using binary search)
   - 32768 bytes
     Size of each entry in PI = Length of $KS$ + Block Ptr length $= 16 + 8\ bytes = 24 bytes$
     Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{24} \rfloor = 1365$
     Number of blocks in PI $= \lceil \frac{Number\ of\ blocks\ in\ S}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{3}{1365} \rceil = 1$
     (PI in sparse index, so 1 entry per block of data)
     **Number of block accesses for S.KS = val**
     => Block accesses in Primary Index $+ 1$
     => $1 + 1 = 2$(As there is only one block in PI, there is no point of using binary search)

**Question d:** Size in the number of blocks for the Clustered Index

1. On $R.B$ and the number of block accesses for $R.B < val$.

2. On $S.A$ and the number of block accesses for $S.A = val$.

**Answer d:**
**Assumption:** The relations $R, S$ are ordered on $R.B, S.A$ (the fields on which Clustered Index is formed).

1. On $R.B$ and the number of block accesses for $R.B < val$.
   As $R.B$ is $2^{10}$-uniform, and the number of rows of R $= 2^{20}$.

- 512 bytes

  Size of each entry in CI (index) = Length of $B$ + Block Ptr length $= 28 + 8\ bytes = 36\ bytes$

  Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{36} \rfloor = 14$

  Number of blocks in CI = $\lceil \frac{Number\ of\ distinct\ entries\ of\ R.B}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^{10}}{14} \rceil = 74$

  **Number of block accesses for R.B < val**

  Using binary search to first find the block ptr to $R.B = val$, then accessing all the records in linear fashion from there for $R.B < val$.

  $=> \lceil log_2(Number\ of\ blocks\ in\ CI) \rceil$ + Number of blocks corresponding to $R.B < val$

  $=> \lceil log_2(74) \rceil\ + \lceil \frac{Number\ of\ records\ with\ R.B<S}{1} \rceil$

  $=> Average\ Accesses = \lceil \frac{\sum_{n=0}^{2^{20}} (\lceil log_2(74) \rceil + \lceil \frac{n}{1} \rceil)}{1+2^{20}} \rceil = 524295$

  But in the worse case (as we don't now the number of records with $R.B < val$), let's say all the records fall under this condition then, disk block accesses will be $N$, where $N$ is the number of blocks corresponding to relation $R$.

  $=> N = 2^{20} => Number\ of\ block\ accesses = 1048576$

- 4096 bytes

  Size of each entry in CI (index) = Length of $B$ + Block Ptr length $= 28 + 8\ bytes = 36 bytes$

  Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{36} \rfloor = 113$

  Number of blocks in CI = $\lceil \frac{Number\ of\ distinct\ entries\ of\ R.B}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^{10}}{113} \rceil = 10$

  **Number of block accesses for R.B < val**

  Using binary search to first find the block ptr to $R.B = val$, then accessing all the records in linear fashion from there for $R.B < val$.

  $=> \lceil log_2(Number\ of\ blocks\ in\ CI) \rceil$ + Number of blocks corresponding to $R.B < val$

  $=> \lceil log_2(10) \rceil\ + \lceil \frac{Number\ of\ records\ with\ R.B<S}{13} \rceil$

  $=> Average\ Accesses = \lceil \frac{\sum_{n=0}^{2^{20}} (\lceil log_2(10) \rceil + \lceil \frac{n}{13} \rceil)}{1+2^{20}} \rceil = 40335$

  But in the worse case (as we don't now the number of records with $R.B < val$), let's say all the records fall under this condition then, disk block accesses will be $N$, where $N$ is the number of blocks corresponding to relation $R$.

  $=> N = 80660 => Number\ of\ block\ accesses = 80660$

- 32768 bytes

  Size of each entry in CI (index) = Length of $B$ + Block Ptr length $= 28 + 8\ bytes = 36 bytes$

  Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{36} \rfloor = 910$

  Number of blocks in CI = $\lceil \frac{Number\ of\ distinct\ entries\ of\ R.B}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^{10}}{910} \rceil = 2$

  **Number of block accesses for R.B < val**

  Using binary search to first find the block ptr to $R.B = val$, then accessing all the records in linear fashion from there for $R.B < val$.

$=> \lceil log_2(Number\ of\ blocks\ in\ CI) \rceil$ + Number of blocks corresponding to $R.B < val$
$=> \lceil log_2(2) \rceil\ + \lceil \frac{Number\ of\ records\ with\ R.B<S}{110} \rceil$
$=> Average\ Accesses = \lceil \frac{\sum_{n=0}^{2^{20}} (\lceil log_2(74) \rceil + \lceil \frac{n}{110} \rceil)}{1+2^{20}} \rceil = 4774$
But in the worse case (as we don't now the number of records with $R.B < val$), let's say all the records fall under this condition then, disk block accesses will be $N$, where $N$ is the number of blocks corresponding to relation $R$.
$=> N = 9533 => Number\ of\ block\ accesses = 9533$

2. On $S.A$ and the number of block accesses for $S.A = val$.
   As $S.A$ is 10-uniform, and the number of rows of S = $2^8$ the maximum number of the occurrence of one value in $S.A$ is $n_{A=val} = \lceil \frac{2^8}{10} \rceil = \lceil 25.8 \rceil = 26$.

   - 512 bytes
     Size of each entry in CI (index) = Length of $B$ + Block Ptr length $= 24 + 8\ bytes = 32\ bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{32} \rfloor = 16$
     Number of blocks in CI = $\lceil \frac{Number\ of\ distinct\ entries\ of\ S.A}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{10}{16} \rceil = 1$
     **Number of block accesses for S.A = val**
     => Block accesses in Clustered Index + number of blocks corresponding to $S.A = val$
     $=> 1 + \lceil \frac{n_{A=val}}{bfr} \rceil$ (As there is only one block in CI, there is no point of using binary search)
     $=> 1 + \lceil \frac{26}{1} \rceil = 27$
   - 4096 bytes
     Size of each entry in CI (index) = Length of $B$ + Block Ptr length $= 24 + 8\ bytes = 32 bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{32} \rfloor = 128$
     Number of blocks in CI = $\lceil \frac{Number\ of\ distinct\ entries\ of\ S.A}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{10}{128} \rceil = 1$
     **Number of block accesses for S.A = val**
     => Block accesses in Clustered Index + number of blocks corresponding to $S.A = val$
     $=> 1 + \lceil \frac{n_{A=val}}{bfr} \rceil$ (As there is only one block in CI, there is no point of using binary search)
     $=> 1 + \lceil \frac{26}{15} \rceil = 1 + 2 = 3$
   - 32768 bytes
     Size of each entry in CI (index) = Length of $B$ + Block Ptr length $= 24 + 8\ bytes = 32 bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{32} \rfloor = 1024$
     Number of blocks in CI = $\lceil \frac{Number\ of\ distinct\ entries\ of\ S.A}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{10}{1024} \rceil = 1$
     **Number of block accesses for S.A = val**
     => Block accesses in Clustered Index + number of blocks corresponding to $S.A = val$
     $=> 1 + \lceil \frac{n_{A=val}}{bfr} \rceil$ (As there is only one block in CI, there is no point of using binary search)
     $=> 1 + \lceil \frac{26}{126} \rceil = 1 + 1 = 2$

**Question e:** Size in the number of blocks for the secondary index on key.

1. On $S.KS$ and the number of block accesses for $S.KS = val$.

**Answer e:**

1. On $S.KS$ and the number of block accesses for $S.KS = val$.

   - 512 bytes
     Size of each entry in SI = Length of $KS$ + Record Ptr length = 16+16 $bytes$ = 32$bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{32} \rfloor = 16$
     Number of blocks in SI = $\lceil \frac{Number\ of\ rows\ in\ S}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{16} \rceil = 16$
     (As SI is dense, 1 entry per record in SI
     **Number of block accesses for S.KS = val**
     => Block accesses in Secondary Index + 1
     => $\lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + 1$
     => $\lceil \log_2 16 \rceil + 1 = 4 + 1 = 5$
   - 4096 bytes
     Size of each entry in SI = Length of $KS$ + Record Ptr length = 16+16 $bytes$ = 32$bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{32} \rfloor = 128$
     Number of blocks in SI = $\lceil \frac{Number\ of\ rows\ in\ S}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{128} \rceil = 2$
     (As SI is dense, 1 entry per record in SI
     **Number of block accesses for S.KS = val**
     => Block accesses in Secondary Index + 1
     => $\lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + 1$
     => $\lceil \log_2 2 \rceil + 1 = 1 + 1 = 2$
   - 32768 bytes
     Size of each entry in SI = Length of $KS$ + Record Ptr length = 16+16 $bytes$ = 32$bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{32} \rfloor = 1024$
     Number of blocks in SI = $\lceil \frac{Number\ of\ rows\ in\ S}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{1024} \rceil = 1$
     (As SI is dense, 1 entry per record in SI
     **Number of block accesses for S.KS = val**
     => Block accesses in Secondary Index + 1
     => $1 + 1 = 2$(As there is only one block in SI, there is no point of using binary search)

**Question f:** Size in the number of blocks for the secondary index on non-key.

1. On $R.C$ and the number of block accesses for $R.C = val$
2. On $S.B$ and the number of block accesses for $S.B < val$
3. On $R.KS$ and the number of block accesses for $R.KS = val$

**Answer f:**

1. On $R.C$ and the number of block accesses for $R.C = val$.
   As $R.C$ is 2-value uniform, number of distinct $R.C$ values will be 2 hence, the number of entries with $R.C = val$ (some particular value) if $val$ exists in the relation is $n_{A=val} = \frac{Num\ of\ rows\ in\ R}{2} = \frac{2^{20}}{2} = 2^{19}$.

- 512 bytes
  Size of each entry in SI = Length of $C$ + Block Ptr length = $4 + 8\ bytes = 12 bytes$
  Size of each entry in Indirection Block = Record Ptr length = 16 $bytes$
  Number of entries per Indirection block (bfr) $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry\ in\ Indirection\ Block} \rfloor = \lfloor \frac{512}{16} \rfloor = 32$
  Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{12} \rfloor = 42$
  Number of blocks in SI $= \lceil \frac{Number\ of\ distinct\ values\ of\ R.C}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2}{42} \rceil = 1$
  **Number of block accesses for R.C = val**
  => Block accesses in Secondary Index + Number of indirection blocks corresponding to $(R.C = val)$ + Number of rows corresponding to $(R.C = val)$
  $=> \lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + \lceil \frac{n_{A=val}}{bfr_{Indirection\ Block}} \rceil + n_{A=val}$
  $=> \lceil \log_2 1 \rceil + \lceil \frac{n_{A=val}}{bfr_{Indirection\ Block}} \rceil + n_{A=val}$
  $= 1 + \lceil \frac{2^{19}}{32} \rceil + 2^{19}$
  $= 1 + 16384 + 524288 = 540673$

- 4096 bytes
  Size of each entry in SI = Length of $C$ + Block Ptr length = $4 + 8\ bytes = 12 bytes$
  Size of each entry in Indirection Block = Record Ptr length = 16 $bytes$
  Number of entries per Indirection block (bfr) $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry\ in\ Indirection\ Block} \rfloor = \lfloor \frac{4096}{16} \rfloor = 256$
  Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{12} \rfloor = 341$
  Number of blocks in SI $= \lceil \frac{Number\ of\ distinct\ values\ of\ R.C}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2}{341} \rceil = 1$
  **Number of block accesses for R.C = val**
  => Block accesses in Secondary Index + Number of indirection blocks corresponding to $(R.C = val)$ + Number of rows corresponding to $(R.C = val)$
  $=> \lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + \lceil \frac{n_{A=val}}{bfr_{Indirection\ Block}} \rceil + n_{A=val}$
  $=> \lceil \log_2 1 \rceil + \lceil \frac{n_{A=val}}{bfr_{Indirection\ Block}} \rceil + n_{A=val}$
  $= 1 + \lceil \frac{2^{19}}{256} \rceil + 2^{19}$
  $= 1 + 2048 + 524288 = 526337$

- 32768 bytes
  Size of each entry in SI = Length of $C$ + Block Ptr length = $4 + 8\ bytes = 12 bytes$
  Size of each entry in Indirection Block = Record Ptr length = 16 $bytes$
  Number of entries per Indirection block (bfr) $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry\ in\ Indirection\ Block} \rfloor = \lfloor \frac{32768}{16} \rfloor = 2048$
  Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{12} \rfloor = 2730$
  Number of blocks in SI $= \lceil \frac{Number\ of\ distinct\ values\ of\ R.C}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2}{2730} \rceil = 1$
  **Number of block accesses for R.C = val**
  => Block accesses in Secondary Index + Number of indirection blocks corresponding to $(R.C = val)$ + Number of rows corresponding to $(R.C = val)$
  $=> \lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + \lceil \frac{n_{A=val}}{bfr_{Indirection\ Block}} \rceil + n_{A=val}$
  $=> \lceil \log_2 1 \rceil + \lceil \frac{n_{A=val}}{bfr_{Indirection\ Block}} \rceil + n_{A=val}$
  $= 1 + \lceil \frac{2^{19}}{2048} \rceil + 2^{19}$
  $= 1 + 256 + 524288 = 524545$

2. On $S.B$ and the number of block accesses for $S.B < val$
   Here, as $S.B$ is $2^8$-uniform, and the total number of rows in $S$ is $2^8$, therefore all the values in $S.B$ are distinct. Now the number of the block accesses will be traversing all the records in secondary index till $S.B < val$ and accessing corresponding block pointers. Let the number of records where $S.B < val$ be $n_{<val}$, then the number of block accesses will be
   $=> \lceil \frac{n_{<val}}{SI_{bfr}} \rceil + n_{<val}$

   - 512 bytes
     Size of each entry in SI = Length of $B$ + Record Ptr length = $28 + 16\ bytes = 44 bytes$
     Number of entries per block (bfr) = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{44} \rfloor = 11$
     **Number of block accesses for S.B < val**
     => Block accesses in Secondary Index + $n_{<val}$
     $=> \lceil \frac{n_{<val}}{SI_{bfr}} \rceil + n_{<val} = \lceil \frac{n_{<val}}{11} \rceil + n_{<val}$
     $=> Average\ block\ accesses = \lceil \frac{\sum_{n_{<val}=0}^{2^8} (\lceil \frac{n_{<val}}{11} \rceil + n_{<val})}{2^8+1} \rceil = 141$
   - 4096 bytes
     Size of each entry in SI = Length of $KS$ + Record Ptr length = 28+16 $bytes = 44 bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{54} \rfloor = 93$
     Number of blocks in SI = $\lceil \frac{Number\ of\ rows\ in\ S}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{128} \rceil = 2$
     **Number of block accesses for S.B < val**
     => Block accesses in Secondary Index + $n_{<val}$
     $=> \lceil \frac{n_{<val}}{SI_{bfr}} \rceil + n_{<val} = \lceil \frac{n_{<val}}{93} \rceil + n_{<val}$
     $=> Average\ block\ accesses = \lceil \frac{\sum_{n_{<val}=0}^{2^8} (\lceil \frac{n_{<val}}{93} \rceil + n_{<val})}{2^8+1} \rceil = 130$
   - 32768 bytes
     Size of each entry in SI = Length of $KS$ + Record Ptr length = 28+16 $bytes = 44 bytes$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{44} \rfloor = 744$
     **Number of block accesses for S.B < val**
     => Block accesses in Secondary Index + $n_{<val}$
     $=> \lceil \frac{n_{<val}}{SI_{bfr}} \rceil + n_{<val} = \lceil \frac{n_{<val}}{744} \rceil + n_{<val}$
     $=> Average\ block\ accesses = \lceil \frac{\sum_{n_{<val}=0}^{2^8} (\lceil \frac{n_{<val}}{744} \rceil + n_{<val})}{2^8+1} \rceil = 129$

3. On $R.KS$ and the number of block accesses for $R.KS = val$
   As $R.KS$ is $2^8$-value uniform, number of distinct $R.KS$ values will be $2^8$ hence, the number of entries with $R.KS = val$ (some particular value) if $val$ exists in the relation is $n_{KS=val} = \frac{Num\ of\ rows\ in\ R}{2^8} = \frac{2^{20}}{2^8} = 2^{12}$.

   - 512 bytes
     Size of each entry in SI = Length of $KS$ + Block Ptr length = $16 + 8\ bytes = 24 bytes$
     Size of each entry in Indirection Block = Record Ptr length = 16 $bytes$
     Number of entries per Indirection block (bfr) = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry\ in\ Indirection\ Block} \rfloor = \lfloor \frac{512}{16} \rfloor = 32$
     Number of entries per block = $\lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{512}{24} \rfloor = 21$
     Number of blocks in SI = $\lceil \frac{Number\ of\ distinct\ values\ of\ R.KS}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{21} \rceil = 22$

**Number of block accesses for R.KS = val**
=> Block accesses in Secondary Index + Number of indirection blocks corresponding to $(R.KS = val)$ + Number of rows corresponding to $(R.KS = val)$
=> $\lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + \lceil \frac{n_{KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{KS=val}$
=> $\lceil \log_2 22 \rceil + \lceil \frac{n_{KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{KS=val}$
$= 5 + \lceil \frac{2^{12}}{32} \rceil + 2^{12}$
$= 5 + 128 + 4096 = 4229$

- 4096 bytes
Size of each entry in SI = Length of $KS$ + Block Ptr length $= 16 + 8\ bytes = 24 bytes$
Size of each entry in Indirection Block = Record Ptr length $= 16\ bytes$
Number of entries per Indirection block (bfr) $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry\ in\ Indirection\ Block} \rfloor = \lfloor \frac{4096}{16} \rfloor = 256$
Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{4096}{24} \rfloor = 170$
Number of blocks in SI $= \lceil \frac{Number\ of\ distinct\ values\ of\ R.KS}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{170} \rceil = 2$
**Number of block accesses for R.KS = val**
=> Block accesses in Secondary Index + Number of indirection blocks corresponding to $(R.KS = val)$ + Number of rows corresponding to $(R.KS = val)$
=> $\lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + \lceil \frac{n_{KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{KS=val}$
=> $\lceil \log_2 2 \rceil + \lceil \frac{n_{KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{A=val}$
$= 1 + \lceil \frac{2^{12}}{256} \rceil + 2^{12}$
$= 1 + 16 + 4096 = 4113$

- 32768 bytes
Size of each entry in SI = Length of $KS$ + Block Ptr length $= 16 + 8\ bytes = 24 bytes$
Size of each entry in Indirection Block = Record Ptr length $= 16\ bytes$
Number of entries per Indirection block (bfr) $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry\ in\ Indirection\ Block} \rfloor = \lfloor \frac{32768}{16} \rfloor = 2048$
Number of entries per block $= \lfloor \frac{Block\ size}{Size\ of\ each\ entry} \rfloor = \lfloor \frac{32768}{24} \rfloor = 1365$
Number of blocks in SI $= \lceil \frac{Number\ of\ distinct\ values\ of\ R.KS}{Number\ of\ entries\ per\ block} \rceil = \lceil \frac{2^8}{1365} \rceil = 1$
**Number of block accesses for R.KS = val**
=> Block accesses in Secondary Index + Number of indirection blocks corresponding to $(R.KS = val)$ + Number of rows corresponding to $(R.KS = val)$
=> $\lceil \log_2 Number\ of\ blocks\ in\ SI \rceil + \lceil \frac{n_{KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{KS=val}$
=> $\lceil \log_2 1 \rceil + \lceil \frac{n_{KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{KS=val}$
$= 1 + \lceil \frac{2^{12}}{2048} \rceil + 2^{12}$
$= 1 + 2 + 4096 = 4099$

**Question g:** Consider a B-Tree on $R.KR$ with each block 2/3rd full. Determine the number of block accesses for $R.KR = val$ retrieval.

**Answer g:**
Let the Number of Block pointers per node be $q$
Record Pointers per node $= q - 1$
Key values per node $= q - 1$
For given block size $B$ we can compute $q$ using -

$=> q * length_{blockPtr} + (q-1) * (length_{recordPtr} + length_{key}) \leq B$
$=> 8q + (q-1) * (16+32) \leq B$
$=> 8q + 48q - 48 \leq B$
$=> 56q \leq B + 48$
$=> q = \lfloor \frac{B+48}{56} \rfloor$
As each block is $\frac{2}{3}rd$ full, $q_{actual} = \lfloor \frac{2}{3} * q \rfloor$
Height of B-Tree $= \lceil log_{q_{actual}}(Number\ of\ rows\ in\ relation) \rceil$
In the worst case, Number of block accesses for $(R.KR = val)$ retrieval = Height of B-Tree + 1, as we have to traverse all the way to a leaf node, and 1 block access corresponding to record access using the record pointer stored in tree.

- 512 bytes -
  $q = \lfloor \frac{512+48}{56} \rfloor = \lfloor \frac{560}{56} \rfloor = 10$
  $q_{actual} = \lfloor \frac{2}{3} * 10 \rfloor = 6$
  Height of B-Tree $= \lceil log_6(2^{20}) \rceil = 8$
  **Number of block accesses for** $(R.KR = val)$ **retrieval** = Height of B-Tree + 1 = 8 + 1 = 9

- 4096 bytes -
  $q = \lfloor \frac{4096+48}{56} \rfloor = \lfloor \frac{4144}{56} \rfloor = 74$
  $q_{actual} = \lfloor \frac{2}{3} * 74 \rfloor = 48$
  Height of B-Tree $= \lceil log_{48}(2^{20}) \rceil = 4$
  **Number of block accesses for** $(R.KR = val)$ **retrieval** = Height of B-Tree + 1 = 4 + 1 = 5

- 32768 bytes -
  $q = \lfloor \frac{32768+48}{56} \rfloor = \lfloor \frac{32816}{56} \rfloor = 586$
  $q_{actual} = \lfloor \frac{2}{3} * 586 \rfloor = 390$
  Height of B-Tree $= \lceil log_{390}(2^{20}) \rceil = 3$
  **Number of block accesses for** $(R.KR = val)$ **retrieval** = Height of B-Tree + 1 = 3 + 1 = 4

**Question h:** Consider a B+ Tree on $R.KS$ with each block 2/3rd full. Determine the number of block accesses for $R.KS = val$ retrieval.

**Answer h:**
**NOTE:** As the given field is a non-key, the leaf nodes will point to indirection blocks (rather than records directly) and the table is not ordered according to $R.KS$.

**Indirection Block—**
Size of each entry in Indirection Block = Record Ptr length = 16 *bytes*
Number of entries per Indirection block $Ind_{bfr} = \lfloor \frac{Block\ size}{Size\ of\ each\ entry\ in\ Indirection\ Block} \rfloor = \lfloor \frac{Block\ size}{16} \rfloor$

**Non-Leaf Node —**
Let the Number of Block pointers per node be $q$
Key values per node $= q - 1$

For given block size $B$ we can compute $q$ using -
$=> q * length_{blockPtr} + (q-1) * length_{key} \leq B$
$=> 8q + (q-1) * (16) \leq B$
$=> 8q + 16q - 16 \leq B$
$=> 24q \leq B + 16$
$=> q = \lfloor \frac{B+16}{24} \rfloor$
As each block is $\frac{2}{3}rd$ full, $q_{actual} = \lfloor \frac{2}{3} * q \rfloor$

**Leaf Node —**
Let the Number of Record pointers per node be $q$
Key values per node = $q$
Number of block pointers = 1
For given block size $B$ we can compute $q$ using -
$=> q * (length_{key} + legth_{recordPtr}) + 1 * length_{blockPtr} \leq B$
$=> q * (16 + 16) + 8 \leq B$
$=> 32q + 8 \leq B$
$=> 32q \leq B - 8$
$=> q = \lfloor \frac{B-8}{32} \rfloor$
As each block is $\frac{2}{3}rd$ full, $q_{actual} = \lfloor \frac{2}{3} * q \rfloor$
Height of B+ Tree = $\lceil log_{q_{non-leaf-actual}}(Number\ of\ leaf\ nodes) \rceil$
Number of block accesses for $(R.KS = val)$ retrieval = Height of B+ Tree + Number of indirection blocks corresponding to $(R.KS = val)$ + Number of rows corresponding to $(R.KS = val)$.
$= HeightofB + Tree + \lceil \frac{n_{R.KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{R.KS=val}$
As we have to traverse all the way to a leaf node, and then access all indirection blocks related to $R.KS = val$, then all the blocks (which are un-ordered as the given key is a non-key) corresponding to records.

- 512 bytes -
  $q_{non-leaf} = \lfloor \frac{512+16}{24} \rfloor = \lfloor \frac{528}{24} \rfloor = 22$
  $q_{nonLeafActual} = \lfloor \frac{2}{3} * 22 \rfloor = 14$
  $q_{leaf} = \lfloor \frac{512-8}{32} \rfloor = \lfloor \frac{504}{32} \rfloor = 15$
  $q_{leafActual} = \lfloor \frac{2}{3} * 15 \rfloor = 10$
  Number of entries per Indirection block $Ind_{bfr} = \lfloor \frac{Block\ size}{16} \rfloor = \lfloor \frac{512}{16} \rfloor = 32$
  Number of leaf nodes = $\lceil \frac{Number\ of\ distinct\ values\ of\ R.KS}{q_{leafActual}} \rceil = \lceil \frac{2^8}{10} \rceil = 52$
  $Height\ of B + Tree = \lceil log_{14} 52 \rceil + 1 = 2 + 1 = 3$
  **Number of block accesses for** $(R.KS = val)$ **retrieval** $= HeightofB+Tree+\lceil \frac{n_{R.KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{R.KS=val}$
  $= 3 + \lceil \frac{2^{12}}{32} \rceil + 2^{12} = 4227$

- 4096 bytes -
  $q_{non-leaf} = \lfloor \frac{4096+16}{24} \rfloor = \lfloor \frac{4112}{24} \rfloor = 171$
  $q_{nonLeafActual} = \lfloor \frac{2}{3} * 171 \rfloor = 114$
  $q_{leaf} = \lfloor \frac{4096-8}{32} \rfloor = \lfloor \frac{4088}{32} \rfloor = 127$
  $q_{leafActual} = \lfloor \frac{2}{3} * 127 \rfloor = 84$
  Number of entries per Indirection block $Ind_{bfr} = \lfloor \frac{Block\ size}{16} \rfloor = \lfloor \frac{4096}{16} \rfloor = 256$

Number of leaf nodes $= \lceil \frac{Number\ of\ distinct\ values\ of\ R.KS}{q_{leafActual}} \rceil = \lceil \frac{2^8}{84} \rceil = 7$
$Height of B + Tree = \lceil log_{114} 7 \rceil + 1 = 1 + 1 = 2$
**Number of block accesses for** $(R.KS = val)$ **retrieval** $= Height of B{+}Tree + \lceil \frac{n_{R.KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{R.KS=val}$
$= 2 + \lceil \frac{2^{12}}{256} \rceil + 2^{12} = 4114$

- 32768 bytes -
  $q_{non-leaf} = \lfloor \frac{32768+16}{24} \rfloor = \lfloor \frac{32784}{24} \rfloor = 1366$
  $q_{nonLeafActual} = \lfloor \frac{2}{3} * 1366 \rfloor = 910$
  $q_{leaf} = \lfloor \frac{32768-8}{32} \rfloor = \lfloor \frac{32760}{32} \rfloor = 1023$
  $q_{leafActual} = \lfloor \frac{2}{3} * 1023 \rfloor = 682$
  Number of entries per Indirection block $Ind_{bfr} = \lfloor \frac{Block\ size}{16} \rfloor = \lfloor \frac{32768}{16} \rfloor = 2048$
  Number of leaf nodes $= \lceil \frac{Number\ of\ distinct\ values\ of\ R.KS}{q_{leafActual}} \rceil = \lceil \frac{2^8}{682} \rceil = 1$
  $Height of B + Tree = \lceil log_{910} 1 \rceil + 1 = 2$
  **Number of block accesses for** $(R.KS = val)$ **retrieval** $= Height of B{+}Tree + \lceil \frac{n_{R.KS=val}}{bfr_{Indirection\ Block}} \rceil + n_{R.KS=val}$
  $= 2 + \lceil \frac{2^{12}}{2048} \rceil + 2^{12} = 4100$

**Question i:** Compute the number of block accesses for the block nested loop equijoin between $R$ and $S$ on attributes $R.KS = S.KS$.

**Answer i:**

- 512 bytes
  Number of blocks in R = $2^{20}$
  Number of blocks in S = $2^{8}$
  As S has less number of blocks, it should be the outer relation in join.
  Main memory buffer = $2^{10} + 2$
  Blocks in Main memory allocated to S = $2^8$
  And 1 block in main memory should be reserved for result.
  $\therefore$ Number of blocks left in main memory for R = $2^{10}+2-1-2^8 = 769 <$ Number of blocks in R
  So it is not possible to store R entirely in Main memory.

$$\begin{aligned}\textbf{Number of blocks accesses} &= \text{Blocks in S} + \frac{\text{Blocks in S}}{\text{Blocks in MM allocated to S}} * \text{Blocks in R} \\ &= 2^8 + \frac{2^8}{2^8} * 2^{20} \\ &= 256 + 1048576 = 1048832\end{aligned}$$

- 4096 bytes
  Number of blocks in R = 80660
  Number of blocks in S = 18
  As S has less number of blocks, it should be the outer relation in join.
  Main memory buffer = $2^{10} + 2$
  Blocks in Main memory allocated to S = 18

And 1 block in main memory should be reserved for result.
$\therefore$ Number of blocks left in main memory for R = $2^{10}+2-1-18 = 1007 <$ Number of blocks in R
So it is not possible to store R entirely in Main memory.

$$\textbf{Number of blocks accesses} = \text{Blocks in S} + \frac{\text{Blocks in S}}{\text{Blocks in MM allocated to S}} * \text{Blocks in R}$$
$$= 18 + \frac{18}{18} * 80660$$
$$= 18 + 80660 = 80678$$

- 32768 bytes
  Number of blocks in R = 9533
  Number of blocks in S = 3
  As S has less number of blocks, it should be the outer relation in join.
  Main memory buffer = $2^{10} + 2$
  Blocks in Main memory allocated to S = 3
  And 1 block in main memory should be reserved for result.
  $\therefore$ Number of blocks left in main memory for R = $2^{10}+2-1-3 = 1022 <$ Number of blocks in R
  So it is not possible to store R entirely in Main memory.

$$\textbf{Number of blocks accesses} = \text{Blocks in S} + \frac{\text{Blocks in S}}{\text{Blocks in MM allocated to S}} * \text{Blocks in R}$$
$$= 3 + \frac{3}{3} * 9533$$
$$= 3 + 9533 = 9536$$

**Question j:** Consider an equijoin of $R$ and $S$, on attributes $R.KS = S.KS$, compute the number of block accesses for nested loop join using B+tree index on $S.KS$.

**Answer j:**
Consider B+tree index on $S.KS$, then the number of block access to retrieve $S.KS = val$ will be height of the tree + 1 (1 for record access).

**Non-Leaf Node —**
Let the Number of Block pointers per node be $q$
Key values per node = $q - 1$
For given block size $B$ we can compute $q$ using -
=> $q * length_{blockPtr} + (q-1) * length_{key} \leq B$
=> $8q + (q-1) * (16) \leq B$
=> $8q + 16q - 16 \leq B$
=> $24q \leq B + 16$
=> $q = \lfloor \frac{B+16}{24} \rfloor$

**Leaf Node —**
Let the Number of Record pointers per node be $q$
Key values per node = $q$
Number of block pointers = 1

For given block size $B$ we can compute $q$ using -
$=> q * (length_{key} + legth_{recordPtr}) + 1 * length_{blockPtr} \leq B$
$=> q * (16 + 16) + 8 \leq B$
$=> 32q + 8 \leq B$
$=> 32q \leq B - 8$
$=> q = \lfloor \frac{B-8}{32} \rfloor$
Height of B+Tree = $\lceil log_{q_{non-leaf}}(Number\ of\ leaf\ nodes) \rceil + 1$ (last level)

- 512 bytes -
  $q_{non-leaf} = \lfloor \frac{512+16}{24} \rfloor = \lfloor \frac{528}{24} \rfloor = 22$
  $q_{leaf} = \lfloor \frac{512-8}{32} \rfloor = \lfloor \frac{504}{32} \rfloor = 15$
  Number of leaf nodes = $\lceil \frac{Number\ of\ distinct\ values\ of\ S.KS}{q_{leaf}} \rceil = \lceil \frac{2^8}{15} \rceil = 18$
  $Height\ of B + Tree = \lceil log_{22} 18 \rceil + 1 = 2$
  Total Number B+ Tree nodes = 1 (root) + 18 (leaf) = 19
  **Number of block access for getting** $S.KS = val$ is,
  $=> Height\ of\ B + Tree\ + 1 = 2 + 1 = 3$

  Number of blocks in R = $2^{20}$
  Number of blocks in S = $2^8$

  As there is a B+ tree index on $S.KS$, therefore the outer loop will be going through $R$, and then finding the row-value in $S$ (using B+ tree index) for the respective value of $R.KS$.

  Main memory buffer = $2^{10} + 2$
  Number of blocks of B+ Tree + Relation S = $19 + 2^8 = 19 + 256 = 275$

  As this number is less than the size of the main memory, we can store complete B+ Tree and Relation S in main memory (using retrieval only once).
  1 block in main memory should be reserved for result.
  Number of blocks left in main memory for R = $2^{10} + 2 - 275 - 1 = 750 <$ Number of blocks in R. So, it is not possible to store R entirely in main memory.

  Hence, the blocks of R will be loaded one by one and then for each row in the block loaded, the value will be searched in B+ Tree to get the appropriate row of $S$.
  Number of block accesses = Blocks of R + Blocks of S + Blocks of B+ Tree

$$\begin{aligned}\textbf{Number of block accesses} &= \text{Blocks of R} + \text{Blocks in S} + \text{Blocks of B+ Tree} \\ &= 2^{20} + 275 \\ &= 1048576 + 275 = 1048851\end{aligned}$$

- 4096 bytes -
  $q_{non-leaf} = \lfloor \frac{4096+16}{24} \rfloor = \lfloor \frac{4112}{24} \rfloor = 171$
  $q_{leaf} = \lfloor \frac{4096-8}{32} \rfloor = \lfloor \frac{4088}{32} \rfloor = 127$

Number of leaf nodes = $\lceil\frac{Number\ of\ distinct\ values\ of\ S.KS}{q_{leaf}}\rceil = \lceil\frac{2^8}{127}\rceil = 7$
$Height of B+Tree = \lceil log_{171} 7\rceil + 1 = 2$
Total Number B+ Tree nodes (or blocks) = 1 (root) + 7 (leaf) = 8
**Number of block access for getting** $S.KS = val$ is,
$=> Height\ of\ B+Tree\ + 1 = 2 + 1 = 3$

Number of blocks in R = 80660
Number of blocks in S = 18

As there is a B+ tree index on $S.KS$, therefore the outer loop will be going through $R$, and then finding the row-value in $S$ (using B+ tree index) for the respective value of $R.KS$.

Main memory buffer = $2^{10} + 2$
Number of blocks of B+ Tree + Relation S = $8 + 18 = 26$

As this number is less than the size of the main memory, we can store complete B+ Tree and Relation S in main memory (using retrieval only once).
1 block in main memory should be reserved for result.
Number of blocks left in main memory for R = $2^{10} + 2 - 26 - 1 = 999 <$ Number of blocks in R. So, it is not possible to store R entirely in main memory.

Hence, the blocks of R will be loaded one by one and then for each row in the block loaded, the value will be searched in B+ Tree to get the appropriate row of $S$.
Number of block accesses = Blocks of R + Blocks of S + Blocks of B+ Tree

$$\begin{aligned}\textbf{Number of block accesses} &= \text{Blocks of R} + \text{Blocks in S} + \text{Blocks of B+ Tree}\\ &= 80660 + 26\\ &= 80686\end{aligned}$$

- 32768 bytes -
  $q_{non-leaf} = \lfloor\frac{32768+16}{24}\rfloor = \lfloor\frac{32784}{24}\rfloor = 1366$
  $q_{leaf} = \lfloor\frac{32768-8}{32}\rfloor = \lfloor\frac{32760}{32}\rfloor = 1023$
  Number of leaf nodes = $\lceil\frac{Number\ of\ distinct\ values\ of\ S.KS}{q_{leaf}}\rceil = \lceil\frac{2^8}{1023}\rceil = 1$
  $Height of B+Tree = \lceil log_{1366} 1\rceil + 1 = 1$
  Total Number B+ Tree nodes (or blocks) = 1 (root and leaf both are same) = 1
  **Number of block access for getting** $S.KS = val$ is,
  $=> Height\ of\ B+Tree\ + 1 = 1 + 1 = 2$

  Number of blocks in R = 9533
  Number of blocks in S = 3

  As there is a B+ tree index on $S.KS$, therefore the outer loop will be going through $R$, and then finding the row-value in $S$ (using B+ tree index) for the respective value of $R.KS$.

Main memory buffer = $2^{10} + 2$
Number of blocks of B+ Tree + Relation S = $1 + 3 = 4$

As this number is less than the size of the main memory, we can store complete B+ Tree and Relation S in main memory (using retrieval only once).
1 block in main memory should be reserved for result.
Number of blocks left in main memory for R = $2^{10} + 2 - 4 - 1 = 1021 <$ Number of blocks in R. So, it is not possible to store R entirely in main memory.

Hence, the blocks of R will be loaded one by one and then for each row in the block loaded, the value will be searched in B+ Tree to get the appropriate row of $S$.
Number of block accesses = Blocks of R + Blocks of S + Blocks of B+ Tree

$$\begin{aligned}\textbf{Number of block accesses} &= \text{Blocks of R} + \text{Blocks in S} + \text{Blocks of B+ Tree} \\ &= 9533 + 4 \\ &= 9537\end{aligned}$$

**Question k:** Consider B+tree secondary indices on $R.C$ and $S.C$ are available. Which join algorithm will you use, and how many block accesses will you need to perform $R$ and $S's$ equijoin on attributes $R.C = S.C$.

**Answer k:**
C in both relation has 2 unique values and uniform distribution. As we have B+ trees on C for both relations, we'll also need indirection blocks.
**Calculating number of indirection blocks for relation R -**
Number of rows with value (C = k) = $\frac{Number\ of\ rows}{2} = 2^{19}$(As C has only 2 unique values and uniform distribution)
Size of indirection blocks = Number of values * $length_{recordPtr} = 2^{19} * 16 = 2^{23} bytes$

- 512 bytes -
  Number of indirection blocks = $\frac{2^{23}}{512} = 16384$
- 4096 bytes -
  Number of indirection blocks = $\frac{2^{23}}{4096} = 2048$
- 32768 bytes -
  Number of indirection blocks = $\frac{2^{23}}{32768} = 256$

As the number of indirection blocks is very large, it would be better to simply use block nested loop join in this case as for each block in S eventually all the blocks in R will be loaded. So there is no point of adding extra block accesses due to indirection blocks.
From Answer i:

- 512 bytes -
  Number of blocks accesses = 1048832

- 4096 bytes -
  Number of blocks accesses = 80678

- 32768 bytes -
  Number of blocks accesses = 9536

**Question l:** There is a retrieval on a single relation using an indexing method that will take the maximum number of block accesses. What is that retrieval, and how many block accesses will it take?

**Answer l:**
**Retrieval :** Select * from RELATION Order by FIELD
If the relation is not ordered by FIELD and we have a secondary index on this attribute, then this query would need maximum number block accesses. In case of secondary index, we store record pointers. So if we iterate over sorted values in secondary index, it is possible that consecutive record pointers point to different blocks. In the worst case each record pointer would lead to a new block access.

eg -

| Key | Record Pointer | Corresponding Block number |
|---|---|---|
| 1 | ptr1 | 1 |
| 2 | ptr2 | 2 |
| 3 | ptr3 | 3 |
| 4 | ptr4 | 1 |
| 5 | ptr5 | 2 |
| 6 | ptr6 | 3 |

Suppose in this case main memory can only keep upto 2 blocks for given record, then during sequential scan of the keys in secondary index(for sorted retrieval on key), each block would be brought in main memory(due to uniform distribution) and hence we'll have 1 block access per row.
$\therefore$ **Number of block accesses** = Number of rows in relation (Assuming main memory size is small, which leads to large number of page faults)
For relations S -

- 512 bytes -
  Number of block accesses = $2^8$

- 4096 bytes -
  Number of block accesses = $2^8$

- 32768 bytes -
  Number of block accesses = $2^8$

**Question m:** Change the block size so that the complete primary index on S.KS will fit in a single block. Recompute the number of block accesses for the above query (l).

**Answer m:**
If primary index is stored in main memory and we have primary index on $S.KS$ then relation S is ordered by FIELD. Hence the retrieval mentioned in previous case is essentially sequential scan of

entire relation S.
$\therefore$ **Number of block accesses** = Number of blocks in S(There is no overhead because of Primary Index as it lies in main memory.)

- 512 bytes -
  Number of block accesses = $2^8$
- 4096 bytes -
  Number of block accesses = 18
- 32768 bytes -
  Number of block accesses = 3