Internal Assessment Test 3 – July 2023

QP SCHEME

Sub:	File Structures					Sub Code:	18IS61	Branch:	ISE
Date:	04/06/2023	Duration:	90 min's	Max Marks:	50	Sem/Sec:	VI A, B & C		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1.	Explain What is hashing?.Explain different hashing functions with example.	10	CO3	L2

What is Hashing?

- ✓ A Hash function is like a black box that produces an address every time a key is dropped.
- ✓ The process of producing an address when a key is dropped to hash function is called Hashing
- ✓ Hash function is given by h(K) **--** it transforms a key 'K' into an address.
- ✓ The resulting address is used to store and retrieve the record.

Square the key and take the mid (Mid Square Method):

This method involves:

- ✓ treating the key as single large number
- ✓ squaring the number and
- ✓ extracting whatever number of digits are required from the middle of the result.

For example:

- ✓ Consider the key 453, its square is (453)2 = 205209.
- ✓ Extracting the middle 2 digits yields a number 52 which is between 0 – 99.

Radix Transformation:

This method involves:

- ✓ Converting the key from one base system to another base system.
- ✓ Then dividing the result with maximum address and taking the reminder.

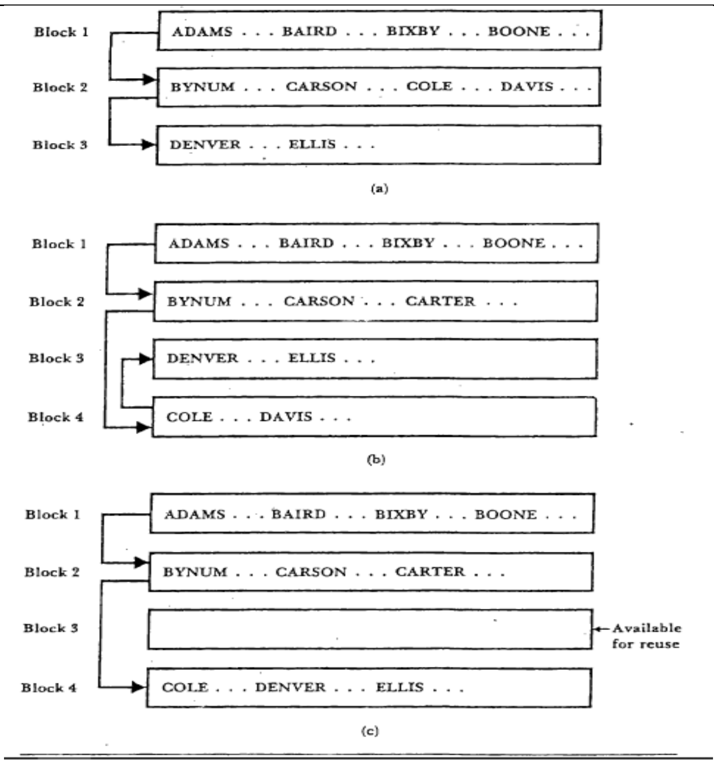
For example:

- ✓ If the hash address range is 0 – 99 and key is (453)11.
- ✓ Converting this number to base 11 system results in (453)11=382.
- ✓ Then 382 mod 99 = 85.
- ✓ So 85 is the hash address.

2.a	Explain Use of Blocks	5	CO3	L2

USE Of BLOCKS

- ✓ Sorting entire file is expensive. So, localize it.
- ✓ One of the best ways is, to collect the records into blocks.
- ✓ Size of buffers used in a program, can hold an entire block.



An example : To show how blocks keep a sequence set in order.

- ➢ Suppose records are keyed on last name and
- ➢ Collected together so there are 4 records in a block, and
- ➢ Also link field is included in each block.
- ✓ Insertion of new records into a block can cause the block to overflow.
- ✓ The overflow condition can be handled by a block-splitting process.
- ✓ Deletion of records can cause a block to be less than half full and therefore to underflow.

Underflow in B-Tree can lead to either of 2 solutions:

- ✓ If a neighbouring node is also half full,
- ✓ Merge the two nodes, freeing one for reuse.
- ✓ If the neighbouring nodes are more than half full,
- ✓ Redistribute records between the nodes to make distribution even

2. b	Explain Choice of block size.	5	CO3	L2

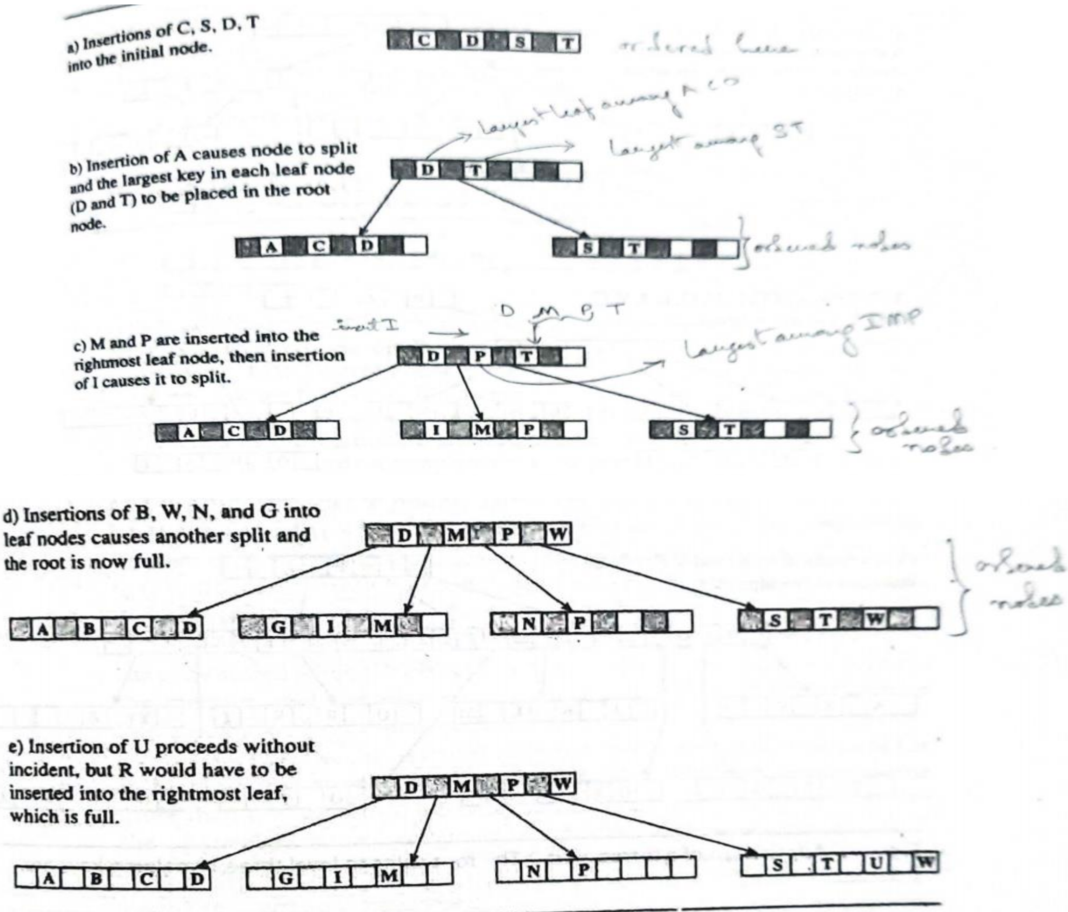
Block is the basic for I/O

Then, What is the block size?

Consideration regarding an upper bound for block size are as follows:

Consideration 1: The block size should be such that we can hold severe blocks in memory at once.

For example, in performing a block split or merging, we should be able to hold at least 2 blocks in memory at a time.

	Consideration 2: Reading in or writing out a block should not take very long. Upper limit is placed on the block size so we would not end up reading entire file just to get at a single record.			
3.	Explain a B-Tree, the creation with examples. ✓ B-trees are balanced search tree. ✓ More than 2 children are possible. ✓ B-Tree, stores all information in the leaves and stores only keys and Child pointer. ✓ If an internal B-tree node x contains n[x] keys then x has n[x]+1 children. Statement of the problem ✓ Searching an index must be faster than binary searching. ✓ Inserting and deleting must be as fast as searching. Construct a B – Tree of order 4, for the following set of keys **C S D T A M P I B W N G U R K E H O L J Y Q Z F X V**  **Figure 9.14** Growth of a B-tree, part 1. The tree grows to a point at which the root needs to be split the second time.	10	CO3	L2
4.	Using an example explain the limitations of chained progressive overflow. ✓ It forms a linked list, or chain, of synonyms.	10	CO3	L3

- ✓ Each home address contains a number indicating the location of the next record with the same home address.
- ✓ The next record in turn contains a pointer to the other record with the same home address.
- ✓ This is shown in the figure below: In the figure below Admans contain the pointer to Cole which is synonym.
- ✓ Then Bates contain pointer to Dean which are again synonym. (Consider the below given Table)
- ✓ The figure below represents the chained progressive overflow technique.

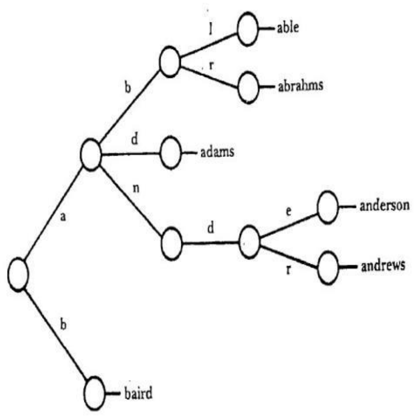
Home address	Actual Address	Records	Address of next Synonym	Search Length
	19	.	.	.
20	20	Adams	22	1
21	21	Bates	23	1
20	22	Cole	25	2
21	23	Dean	-1	2
24	24	Evans	-1	1
20	25	Flint	-1	3
	26		.	.

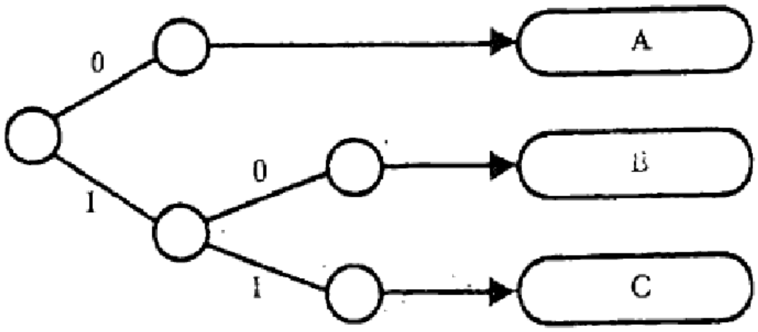
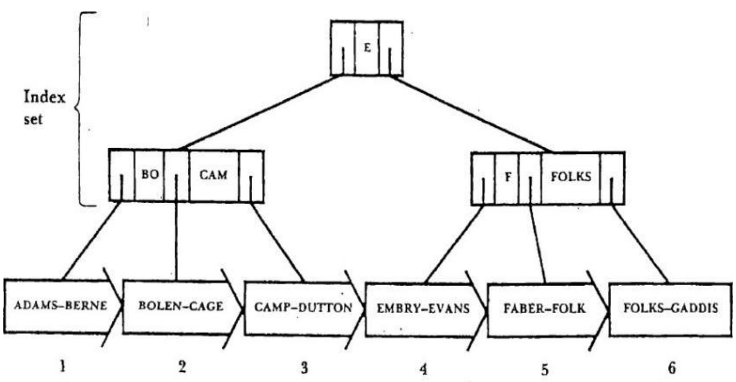
5. Explain how Extendable hashing works. — 10 | CO3 | L2

- ✓ It combines conventional hashing with another retrieval approach called the trie.
- ✓ Tries are also sometimes referred to as radix searching.

In Tries:

- ✓ branching factor of the search tree = the number of alternative symbols that can occur in each position of the key
- ✓ Suppose we want to build a trie that stores the keys able, abrahms, adams, anderson, andrews, and Baird.
- ✓ A schematic form of the trie is shown in Fig.
- ✓ The use-more-as-we-need-more capability is fundamental to the structure of extendible hashing.



	✓ Turning the Trie into a Directory ✓ Tries are used with a radix of 2 in our approach to extendible hashing: ✓ Search decisions are made on a bit-by-bit basis. ✓ Here Tries work in terms of buckets containing keys. Suppose we have: ✓ Bucket A containing keys that, when hashed, have hash addresses that begin with the bits 01. ✓ Bucket B contains keys with hash addresses beginning with 10 and ✓ Bucket C contains keys with addresses that start with 11. Figure shows a trie that allows us to retrieve these.  			
6.a	With a neat sketch, discuss simple prefix B+ Trees and its maintenance ✓ Figure shows how seperators are used to form B-tree index of the sequence set blocks. ✓ The B-tree index is called the index set. ✓ With the sequence set, it forms a file structure called a simple prefix B+ tree. ✓ A node containing N separators branches to N+ 1 children. ✓ Simple Prefix- index set contains shortest Seperators. ✓ Suppose search for record with KEY= EMBRY  Figure 10.7 . A B-tree index set for the sequence set, forming a simple prefix B+ tree.	5	CO3	L2

<table>
<tr><td></td><td>

- ➢ Record insertion and deletion always take place in the sequence set.
- ➢ If splitting, merging, or redistribution is necessary:
- ✓ Perform the operation as if there were no index set at all.
- ✓ If necessary, then make changes in the index set:
- ✓ If blocks are split in the sequence set, a new separator must be inserted into the index set;
- ✓ If blocks are merged in the sequence set, a separator must be removed from the index set; and
- ✓ If records are redistributed between blocks in the sequence set, the value of a separator must be changed.

</td><td></td><td></td><td></td></tr>
<tr><td>6.b</td><td>

Give the structure of Indexed sequential access

- ✓ Indexed sequential file structures provide a choice between 2 alternative views of a file:
- ✓ Indexed: The File can be seen as a set of records that is indexed by key
- ✓ Sequential: The file can be accessed sequentially, returning records in order by key
- ✓ B tree structure provides excellent indexed access to any individual record by key, even as records are added and deleted.

Consider each block contains range of records.

- ✓ If we are looking for a record with the key BURNS, retrieve & inspect the 2nd block.

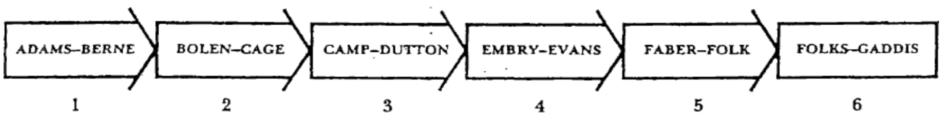


Figure 10.2 Sequence of blocks showing the range of keys in each block.

- ✓ It is easy to construct a simple, single- level index for these blocks.

for example, to build an index of fixed length records that contain the key for the last record in each block, as shown in Fig.

<table>
<tr><th>Key</th><th>Block number</th></tr>
<tr><td>BERNE</td><td>1</td></tr>
<tr><td>CAGE</td><td>2</td></tr>
<tr><td>DUTTON</td><td>3</td></tr>
<tr><td>EVANS</td><td>4</td></tr>
<tr><td>FOLK</td><td>5</td></tr>
<tr><td>GADDIS</td><td>6</td></tr>
</table>

Figure 10.3 Simple index for the sequence set illustrated in Fig. 10.2.

- ✓ The combination of this kind of index with the sequence set of blocks provide **complete Indexed sequential access.**

</td><td>5</td><td>CO3</td><td>L2</td></tr>
</table>