11. Indexing and Hashing

Student Table

StudID	StudName	Birth	DeptName	TotCredits
00128	Zhang	1992-04-18	Comp. Sci.	102
12345	Shankar	1995-12-06	Comp. Sci.	32
19991	Brandt	1993-05-24	History	80
23121	Chavez	1992-04-18	Finance	110
44553	Peltier	1995-10-18	Physics	56
45678	Levy	1995-08-01	Physics	46
54321	Williams	1995-02-28	Comp. Sci.	54
55739	Sanchez	1995-06-04	Music	38
70557	Snow	1995-11-22	Physics	0
76543	Brown	1994-03-05	Comp. Sci.	58

11.6 Ordered, Secondary, Dense Index

Make a Sequentially Ordered, Secondary and Dense Index for the Student table with Search Key DeptName. Assume that the data file has 4 records in each block.

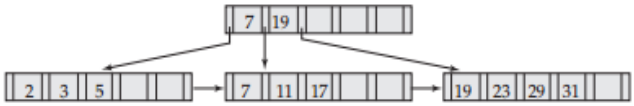
Ordered: Index is a sorted list
Secondary: Index and file are sorted on different Search Keys
Dense: All file records have an index record.

Search Key	Bucket#
Comp. Sci.	1
Finance	2
History	3
Music	4
Physics	5

Bucket#	Block#	Record#
1	1	1
1	1	2
1	2	3
1	3	2
2	1	4
3	1	3
4	2	4
5	2	1
5	2	2
5	3	1

11.7 B+-Tree

A file has the following set of search key values: (2, 3, 5, 7, 11, 17, 19, 23, 29, 31). Consider the following B+-Tree used as an index containing these search key values.



a) What is the order n of the B+ Tree?

N = 6.

b) What is the allowed number of search keys in the root and in a non-root node in a B+ tree of that order?

In the root: min 0, max n-1 = 5.

In a non-root node:

If it is internal: Min $\lceil n/2 \rceil - 1 = 2$. Max n-1=5

If it is a leaf: Min $\lceil (n-1)/2 \rceil = 3$. Max n-1=5

c) Show the steps involved in the two following queries:

a. Find record with a search-key value of 11.

i. Search top level; follow second pointer.

ii. Search next level; follow the second pointer to record with search-key value 11.

b. Find records with a search-key value between 7 and 17, inclusive.

i. Search top level; follow second pointer.

ii. Search next level; follow first pointer to record with search-key value 7, then return.

iii. Follow second pointer to record with search-key value 11, then return.

iv. Follow third pointer to record with search-key value 17.

11.8 Hashed Data File

Make a hash file organization for the Student table with Search Key StudID, assuming that records are inserted in the order shown in the table. There should be 4 buckets, which each can hold max 5 records.

Choose a hash function: h(x) = x % 4 (= MOD(x,4)). We first calculate h(StudID) for all StudID values:

StudID	h(StudID)
00128	0
12345	1
19991	3
23121	1
44553	1
45678	2
54321	1
55739	3
70557	1
76543	3

Then we are ready to place the records in the buckets:

Bucket 0:

00128	Zhang	1992-04-18	Comp. Sci.	102

Bucket 1:

12345	Shankar	1995-12-06	Comp. Sci.	32
23121	Chavez	1992-04-18	Finance	110
44553	Peltier	1995-10-18	Physics	56
54321	Williams	1995-02-28	Comp. Sci.	54
70557	Snow	1995-11-22	Physics	0

Bucket 2:

45678	Levy	1995-08-01	Physics	46

Bucket 3:

19991	Brandt	1993-05-24	History	80
55739	Sanchez	1995-06-04	Music	38
76543	Brown	1994-03-05	Comp. Sci.	58

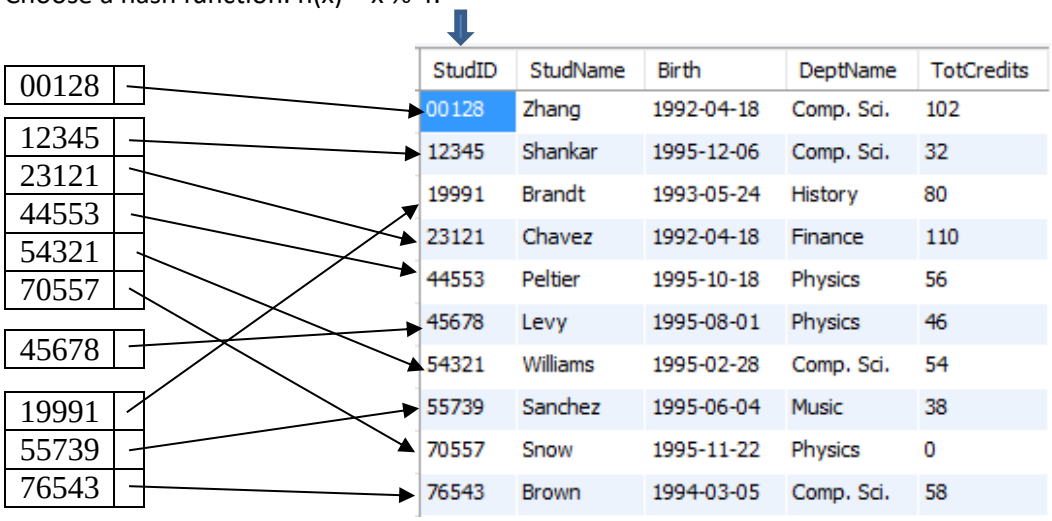
What would the result be, if there could only be 4 records in a bucket?

The record with StudID = 70557 would be placed in an overflow bucket of bucket 1.

11.9 Hash Index (Hashed Index File)

Make a hash index for the Student table with Search Key StudID, assuming that records are inserted in the order shown in the table. There should be 4 buckets, which each can hold max 5 records.

Choose a hash function: h(x) = x % 4.



StudID	StudName	Birth	DeptName	TotCredits
00128	Zhang	1992-04-18	Comp. Sci.	102
12345	Shankar	1995-12-06	Comp. Sci.	32
19991	Brandt	1993-05-24	History	80
23121	Chavez	1992-04-18	Finance	110
44553	Peltier	1995-10-18	Physics	56
45678	Levy	1995-08-01	Physics	46
54321	Williams	1995-02-28	Comp. Sci.	54
55739	Sanchez	1995-06-04	Music	38
70557	Snow	1995-11-22	Physics	0
76543	Brown	1994-03-05	Comp. Sci.	58