

Indexing and Hashing



**CHAPTER 8: FILE STRUCTURES
COMPUTER SCIENCE: AN OVERVIEW**

**LECTURE PREPARED AND DELIVERED BY
DR SYED KHALDOON KHURSHID**

8.3: Quick File Access

- Disadvantage of sequential files:
 - no quick access to particular file data
- Two techniques to overcome this problem:
 - (1) *Indexing* or (2) *Hashing*
- Indexing:

Indexed File

12N67	John Smith	23-Jul-71	17,000.00	New York	...
13C08	Andrew White	27-Jun-70	24,500.00	Boston	...
23G19	Mary Jackson	5-Mar-39	41,000.00	San Francisco	...
24X17	Eleanor Tracy	17-Sep-63	9,635.00	Fort Lauderdale	...
26X28	Michael Flanagan	1-Nov-44	18,800.00	Washington	...
32E76	Glenn White	29-Feb-68	17,000.00	Detroit	...
36Z05	Virginia Moore	27-Jun-70	32,000.00	San Francisco	...
:	:	:	:	:	...
:	:	:	:	:	...
:	:	:	:	:	...

keys

loaded into main
memory when opened

Index

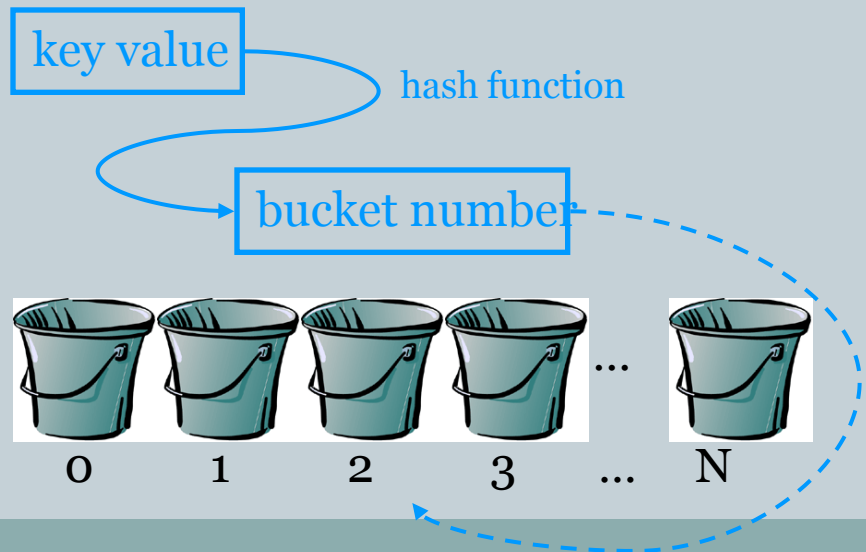
12N67	location
13C08	location
23G19	location
24X17	location
26X28	location
32E76	location
36Z05	location
:	:
:	:
:	:

8.4: Hashing



- Solution: '*hashing*'
 - finds position in file using a key value (as in indexing)...
 - ... simply by identifying location *directly from the key*
- Disadvantage of indexing is... the index
 - requires extra space

- How?
 - define set of '*buckets*' & '*hash function*' that converts keys to bucket numbers



8.4: Hash Function: Example

- If storage space divided into *40 buckets* and hash function is *division*:
 - key values 14, 54, & 94 all map onto same bucket (collision)

