DATABASE

CSE3103 : Database

Nazmus Sakib
Assistant Professor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Basic Concepts

- Indexing mechanisms used to speed up access to desired data.
  - E.g., author catalog in library

- **Search Key** - attribute to set of attributes used to look up records in a file.

- An **index file** consists of records (called **index entries**) of the form

search-key	pointer

- Index files are typically much smaller than the original file

- Two basic kinds of indices:
  - **Ordered indices:** search keys are stored in sorted order
  - **Hash indices:** search keys are distributed uniformly across "buckets" using a "hash function".

Index Evaluation Metrics

- Access types supported efficiently. E.g.,
  - records with a specified value in the attribute
  - or records with an attribute value falling in a specified range of values.

- Access time

- Insertion time
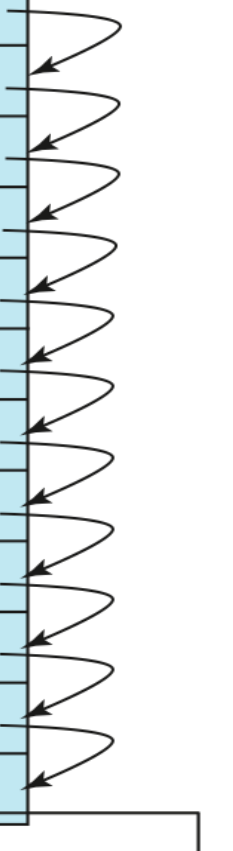
- Deletion time

- Space overhead

Ordered Indices

- In an **ordered index**, index entries are stored sorted on the search key value. E.g., author catalog in library.

- **Primary index:** in a sequentially ordered file, the index whose search key specifies the sequential order of the file.
  - Also called **clustering index**
  - The search key of a primary index is usually but not necessarily the primary key.

- **Secondary index**: an index whose search key specifies an order different from the sequential order of the file. Also called non-clustering index.

- Index-sequential file: ordered sequential file with a primary index.

Dense Index Files

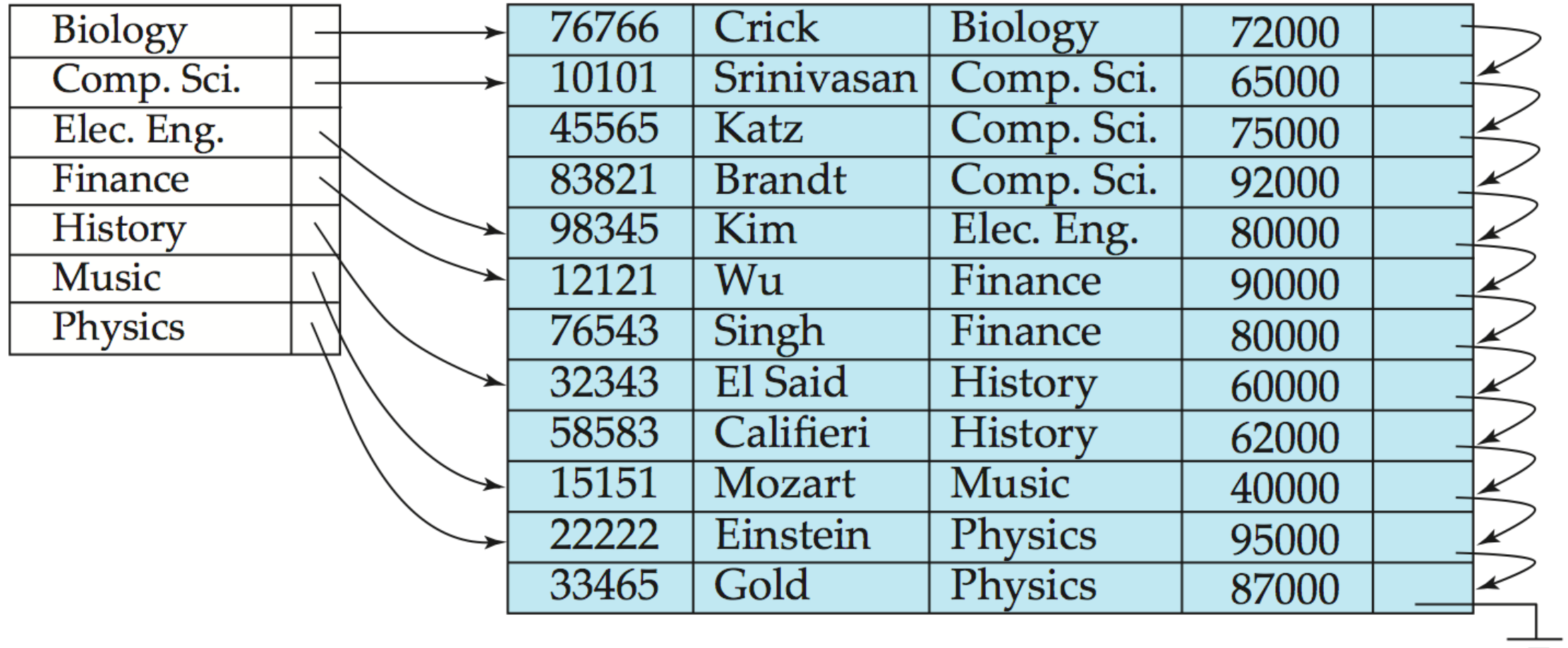- **Dense index** — Index record appears for every search-key value in the file.

- E.g. index on *ID* attribute of *instructor* relation

10101			10101	Srinivasan	Comp. Sci.	65000	
12121			12121	Wu	Finance	90000	
15151			15151	Mozart	Music	40000	
22222			22222	Einstein	Physics	95000	
32343			32343	El Said	History	60000	
33456			33456	Gold	Physics	87000	
45565			45565	Katz	Comp. Sci.	75000	
58583			58583	Califieri	History	62000	
76543			76543	Singh	Finance	80000	
76766			76766	Crick	Biology	72000	
83821			83821	Brandt	Comp. Sci.	92000	
98345			98345	Kim	Elec. Eng.	80000	

Dense Index Files (Cont.)

- Dense index on *dept_name*, with *instructor* file sorted on *dept_name*

Biology			
Comp. Sci.			
Elec. Eng.			
Finance			
History			
Music			
Physics			

76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
45565	Katz	Comp. Sci.	75000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000
12121	Wu	Finance	90000
76543	Singh	Finance	80000
32343	El Said	History	60000
58583	Califieri	History	62000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
33465	Gold	Physics	87000

Sparse Index Files

- **Sparse Index**: contains index records for only some search-key values.
  - Applicable when records are sequentially ordered on search-key

- To locate a record with search-key value K we:
  - Find index record with largest search-key value < K
  - Search file sequentially starting at the record to which the index record points

10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Index records:
| 10101 |
| 32343 |
| 76766 |
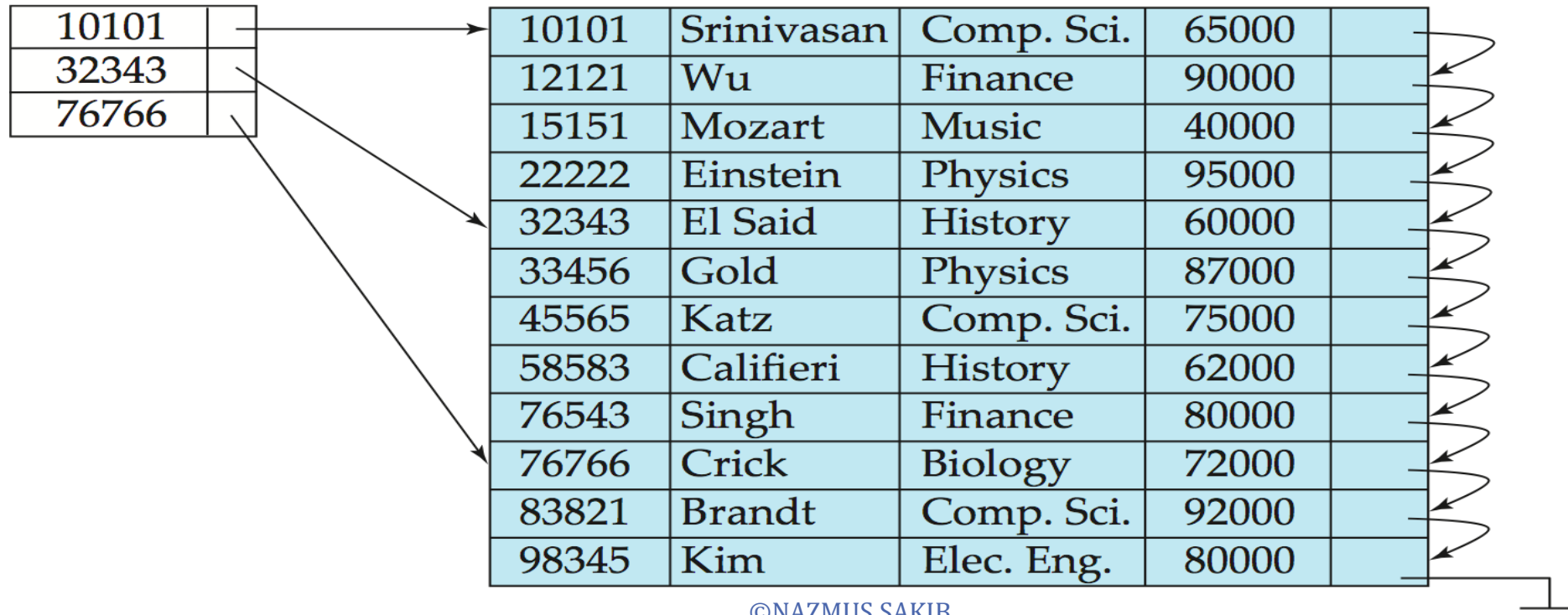
Sparse Index Files (Cont.)

- Compared to dense indices:
  - Less space and less maintenance overhead for insertions and deletions.
  - Generally slower than dense index for locating records.

- **Good tradeoff**: sparse index with an index entry for every block in file, corresponding to least search-key value in the block.