COL 362 & COL 632

Database Storage Organization

22 Feb 2023

Buffer Replacement Policies (2/2)

B1, B2, B3, B1, B4, B3, B2

	M1	M2	M3
B1	B1	-	-
B2	B1	B2	-
B3	B1	B2	B3
B1	B1	B2	B3
B4	B1	B4	B3
B3	B1	B4	B3
B2	B2	B4	B3

LRU – Least Recently Used

Nested-loop join

```
for each tuple i of instructor do
  for each tuple d of department do
    if i[dept_name] == d[dept_name] then
          <generate join tuple>
    end
  end
end
```

- For `department` blocks B1, B2, B3, B4 (and loop over for next block of instructor tuples)

LRU

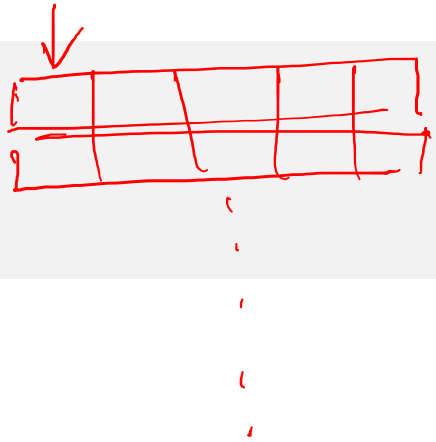
	M1	M2	M3
B1	B1	-	-
B2	B1	B2	-
B3	B1	B2	B3
B4	B4	B2	B3

MRU – Most Recently Used

	M1	M2	M3
B1	B1	-	-
B2	B1	B2	-
B3	B1	B2	B3
B4	B1	B2	B4

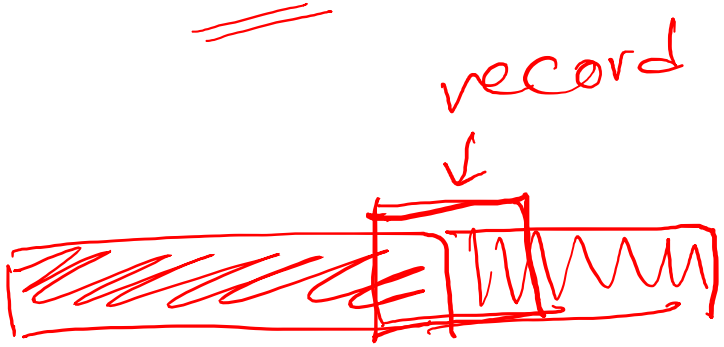
record = tuple
relation = table

File Organization



- The database is stored as a collection of *files*
  - Each file is a sequence of *records*
  - A record is a sequence of fields.
- A table is mapped to a file
- Records mapped to ~~blocks~~ pages
  - A block contains multiple records
  - A single record could span multiple blocks

4K



Operations on Records

- Inserting a new record
- Updating a record
- Deleting a record

Alternative organizations could be efficient for different ”workloads”

Fixed-Length Records

- Simple approach:
  - Store record *i* starting from byte $n * (i - 1)$, where n is the size of each record.
  - Record access is simple but records may cross blocks
    - Modification: do not allow records to cross block boundaries

```
instructor (
        ID varchar(5),
        name varchar(20),
        deptname varchar(20),
        salary numeric(8,2)
)
```

Size of `char`: 1B
Size of `numeric`: 8B

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

Handling Record Deletion - Compaction

- Delete Record 3

Compaction

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

Handling Record Deletion – Move Last Record

- Delete Record 3

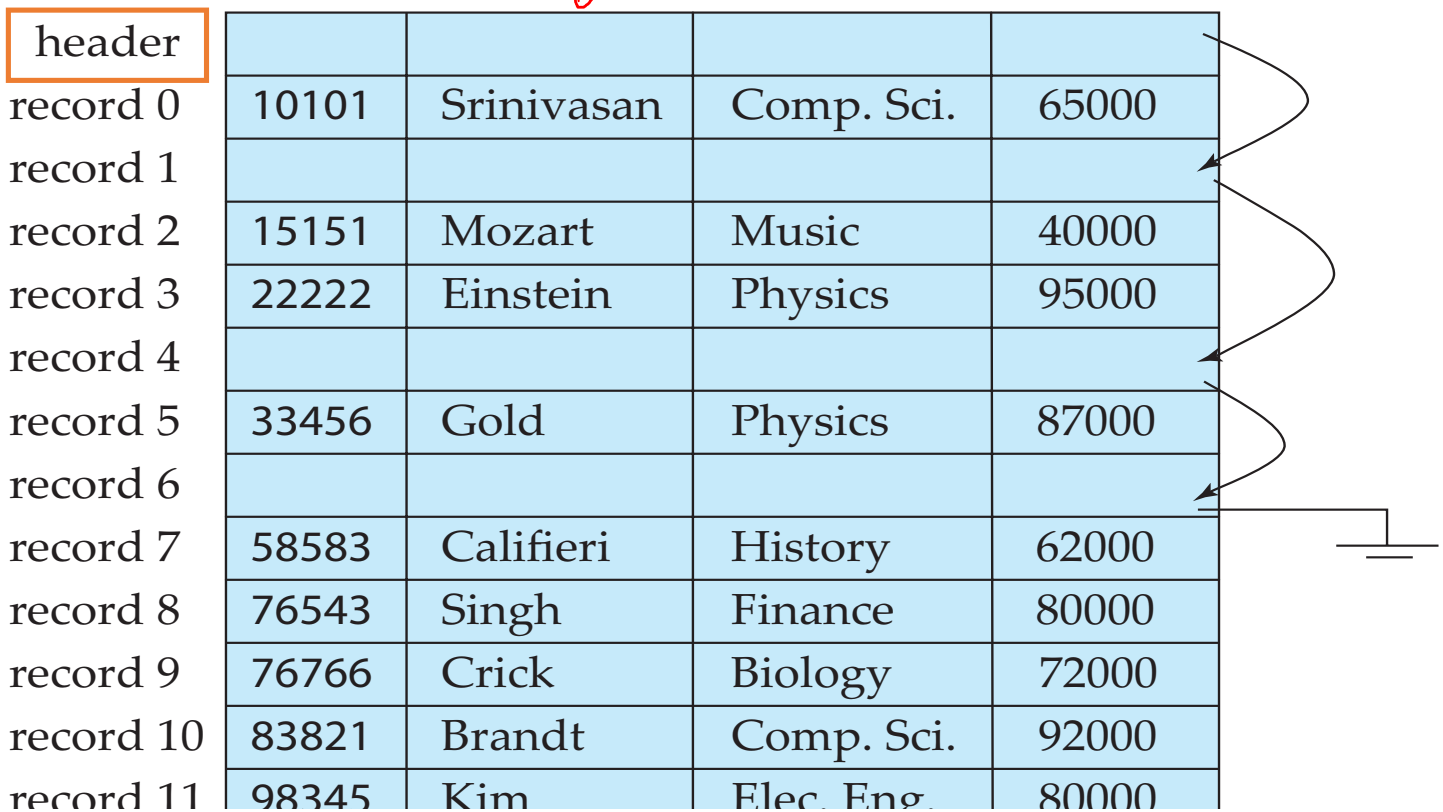
Move last record

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 11	98345	Kim	Elec. Eng.	80000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000

Handling Record Deletion – Free List

- Delete Record 3

Maintain a free list

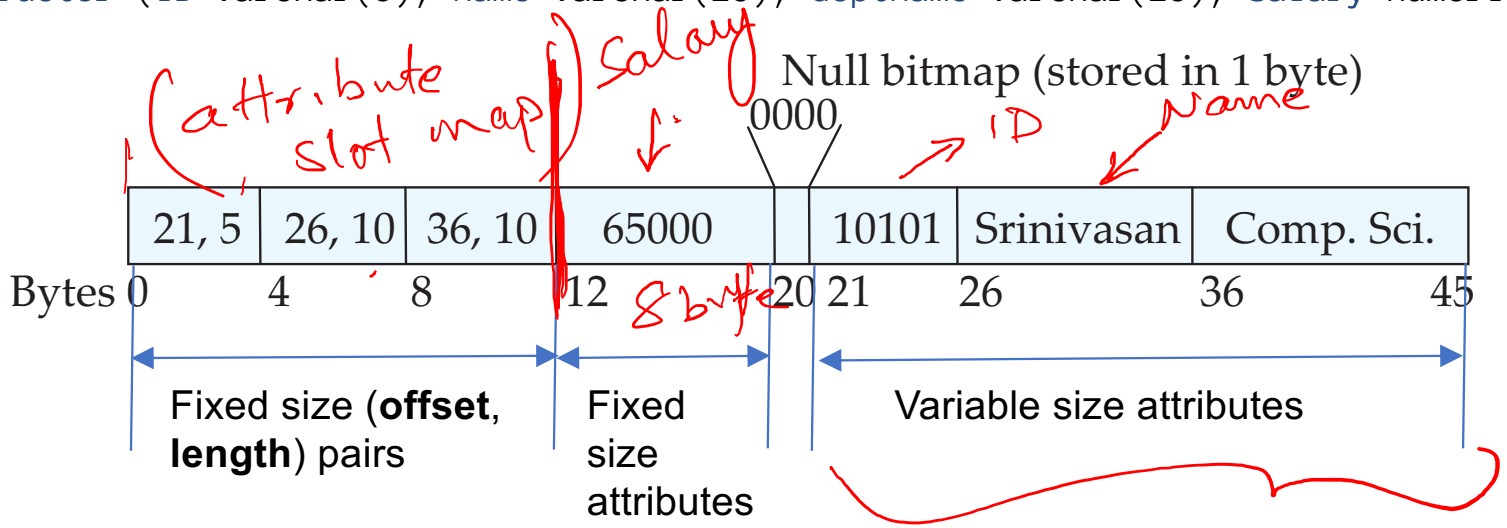


header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1				
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

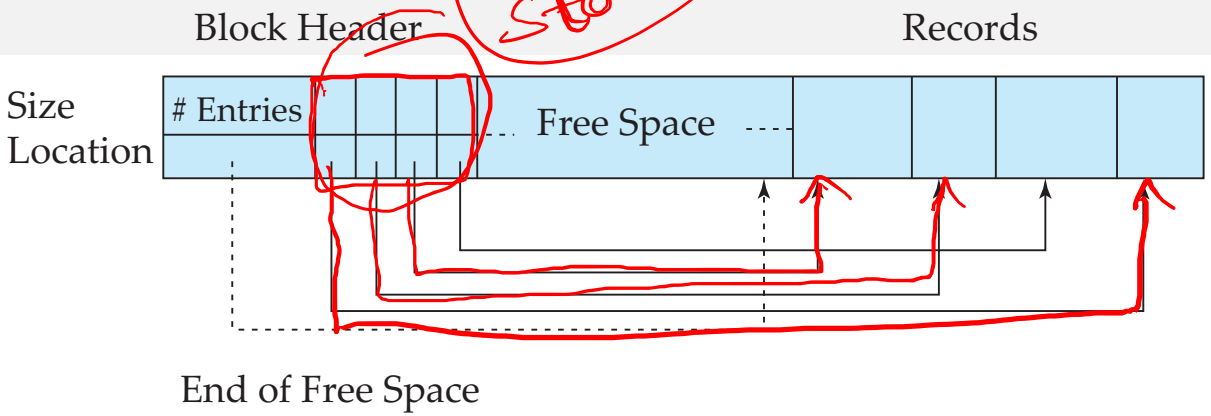
Variable-Length Records (1/2)

- Variable-length records arise in database systems in several ways:
  - Storage of multiple record types in a file.
  - Variable length fields such as strings (varchar)

```
instructor (ID varchar(5), name varchar(20), deptname varchar(20), salary numeric(8,2))
```



Variable-Length Records (2/2)



- **Slotted page** header contains:
  - number of record entries
  - end of free space in the block
  - location and size of each record
- Records can be moved around within the block

Organization of Records in Files

- Heap
  - Store anywhere
- Sequential
  - Maintain ordering based on a search key
- Hashing
  - Easy to locate
- Multitable clustered storage
  - Helps in efficient join processing

Flash storage

H.W.

Sequential File Organization (1/2)

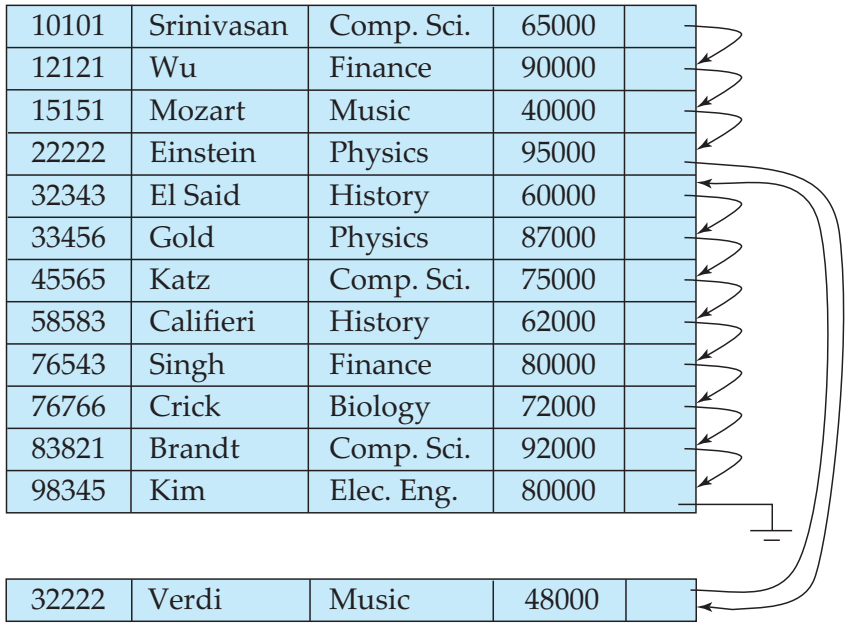
- Suitable for applications that require sequential processing of the entire file
- The records in the file are ordered by a search-key

```
instructor (
      ID varchar(5),
      name varchar(20),
      deptname varchar(20),
      salary numeric(8,2)
)
```

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	

Sequential File Organization (2/2)

- Deletion – use pointer chains
- Insertion – locate the position where the record is to be inserted
  - if there is free space insert there
  - if no free space, insert the record in an overflow block
  - In either case, pointer chain must be updated
- Reorganize the file from time to time



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	

32222	Verdi	Music	48000	

Data Dictionary Storage

The **Data dictionary** (also called **system catalog**) stores **metadata**; that is, data about data, such as

- Information about relations
  - names of relations
  - names, types and lengths of attributes of each relation
  - names and definitions of views
  - integrity constraints
- User and accounting information, including passwords
- Statistical and descriptive data
  - number of tuples in each relation
- Physical file organization information
  - How relation is stored (sequential/hash/…)
  - Physical location of relation
- Information about indices

Relational Representation of System Metadata

- Relational representation on disk
- Specialized data structures designed for efficient access, in memory