

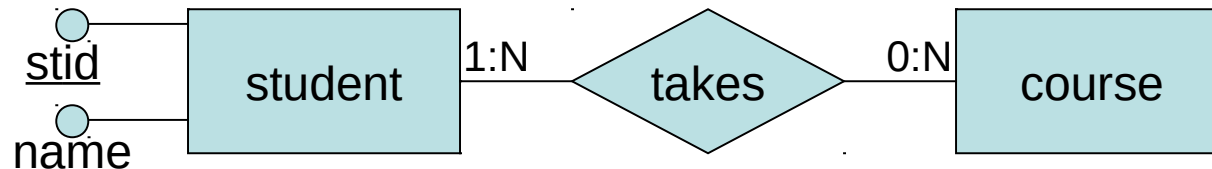
Physical Storage Organization

Outline

- Where and How data are stored?
 - physical level
 - logical level

Building a Database: High-Level

- Design conceptual schema using a data model, e.g. ER, UML, etc.



Building a Database: Logical-Level

- Design logical schema, e.g. relational, network, hierarchical, object-relational, XML, etc schemas
- Data Definition Language (DDL)

```
CREATE TABLE student  
(cid char(8) primary key,name varchar(32))
```

student	
<u>cid</u>	name

Populating a Database

- Data Manipulation Language (DML)

INSERT INTO student VALUES ('00112233', 'Paul')

student	
<u>cid</u>	name
00112233	Paul

Transaction operations

- Transaction: a collection of operations performing a single logical function

```
BEGIN TRANSACTION transfer
```

```
UPDATE bank-account SET balance = balance - 100 WHERE account=1
```

```
UPDATE bank-account SET balance = balance + 100 WHERE account=2
```

```
COMMIT TRANSACTION transfer
```

- A failure during a transaction can leave system in an inconsistent state, eg transfers between bank accounts.

Where and How all this information is stored?

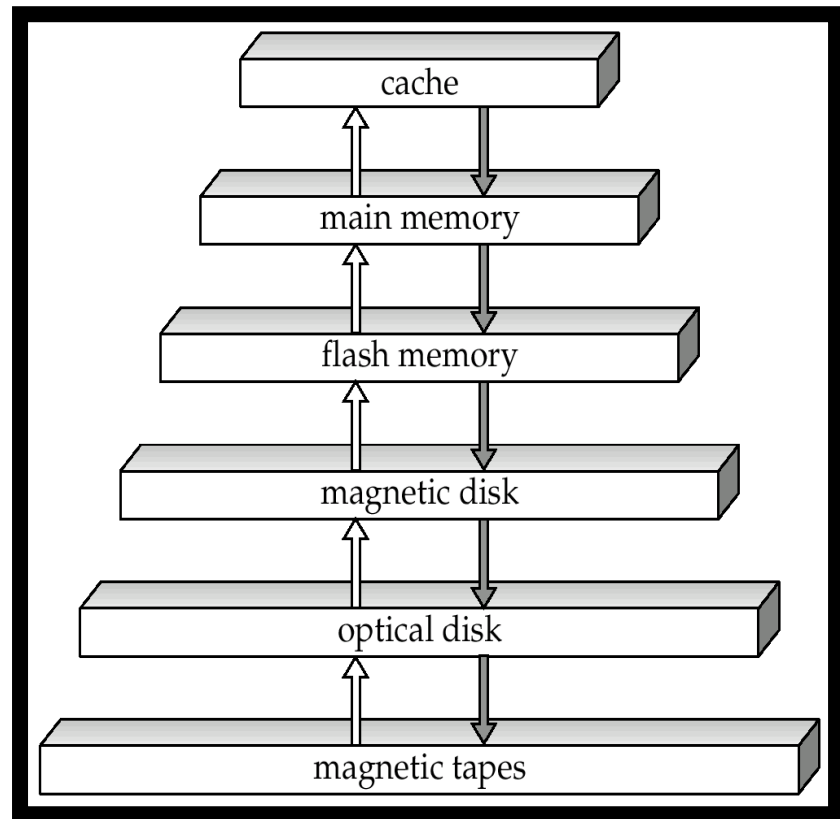
- Metadata: tables, attributes, data types, constraints, etc
- Data: records
- Transaction logs, indices, etc

Where: In Main Memory?

- Fast!
- But:
 - Too small
 - Too expensive
 - Volatile

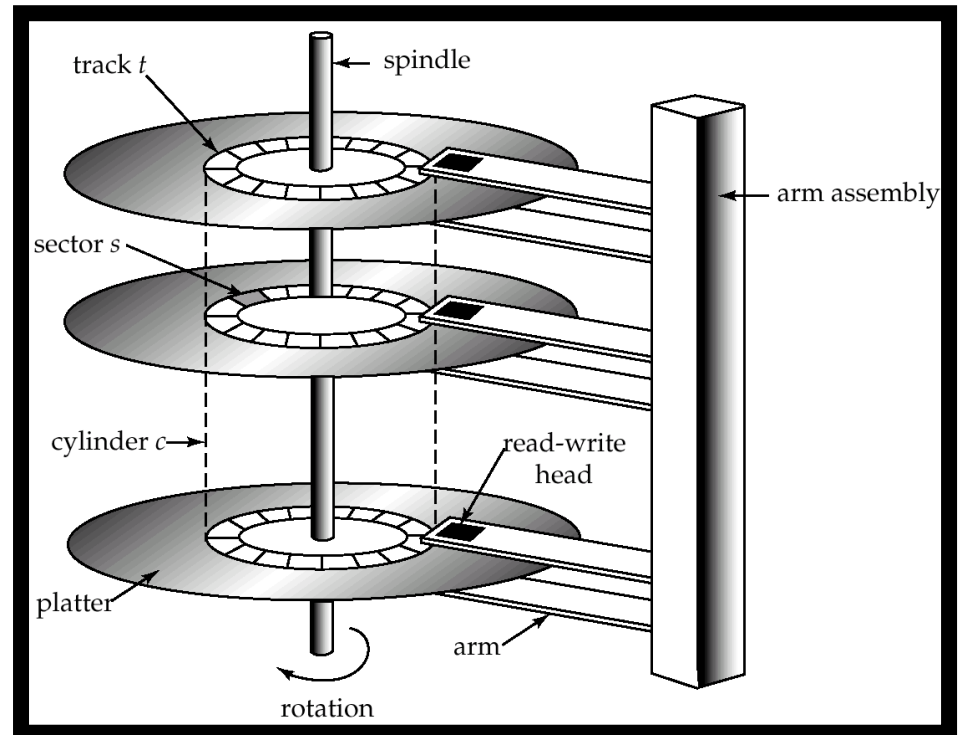
Physical Storage Media

- Primary Storage
 - Cache
 - Main memory
- Secondary Storage
 - Flash memory
 - Magnetic disk
- Offline Storage
 - Optical disk
 - Magnetic tape



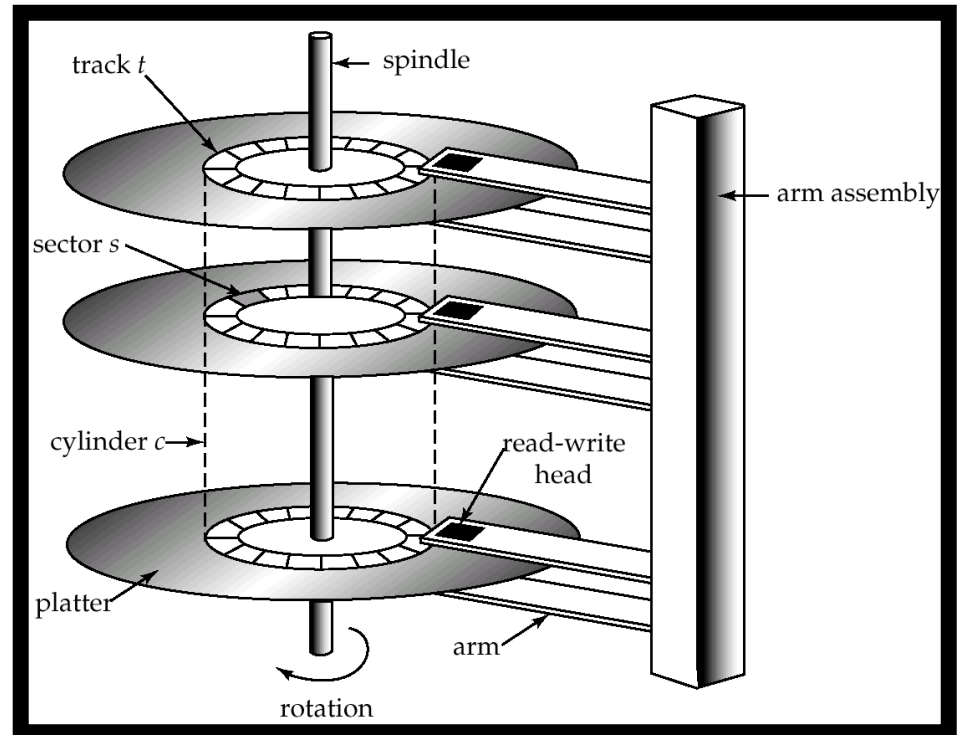
Magnetic Disks

- Random Access
- Inexpensive
- Non-volatile



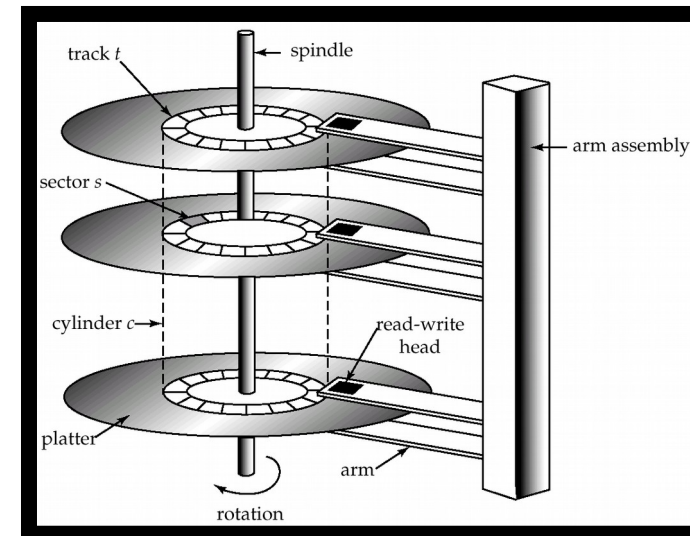
How do disks work?

- Platter: covered with magnetic recording material
- Track: logical division of platter surface
- Sector: hardware division of tracks
- Block: OS division of tracks
 - Typical block sizes:
512 B, 2KB, 4KB
- Read/write head



Disk I/O

- Disk I/O := block I/O
 - Hardware address is converted to Cylinder, Surface and Sector number
 - Modern disks: Logical Sector Address 0...n
- Access time: time from read/write request to when data transfer begins
 - Seek time: the head reaches correct track
 - Average seek time 5-10 msec
 - Rotation latency time: correct block rotated under head
 - 5400 RPM, 15K RPM
 - On average 4-11 msec
- Block Transfer Time



Optimize I/O

- Database system performance I/O bound
- Improve the speed of access to disk:
 - Scheduling algorithms
 - File Organization
- Introduce disk redundancy
 - Redundant Array of Independent Disks (RAID)
- Reduce number of I/Os
 - Query optimization, indices

~~Where~~ and **How** all this information is stored?

- Metadata: tables, attributes, data types, constraints, etc
- Data: records
- Transaction logs, indices, etc
- A collection of **files (or tables)**
 - **Physically** partitioned into **pages or data blocks**
 - **Logically** partitioned into **records**

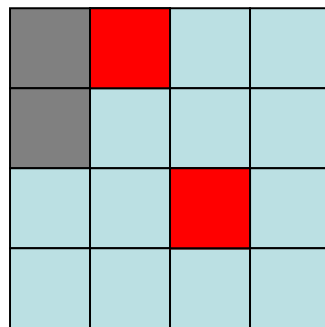
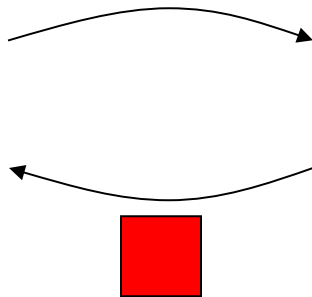
Storage Access

- A collection of files
 - Physically partitioned into pages
 - Typical database page sizes: 2KB, 4KB, 8KB
 - Reduce number of block I/Os := reduce number of page I/Os
 - How?
- Buffer Manager

Buffer Management (1/2)

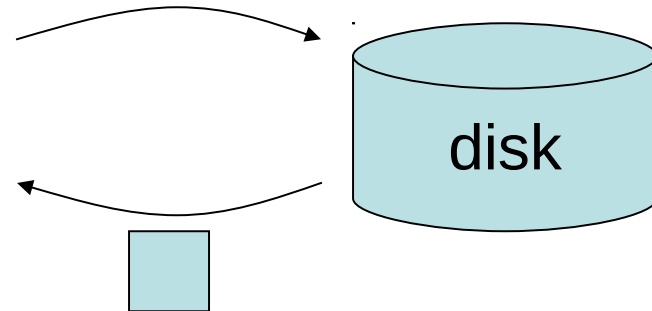
- Buffer: storing a page copy
- Buffer manager: manages a pool of buffers
 - Requested page in pool: hit!
 - Requested page in disk:
 - Allocate page frame
 - Read page and pin
- Problems?

Page request



buffer pool

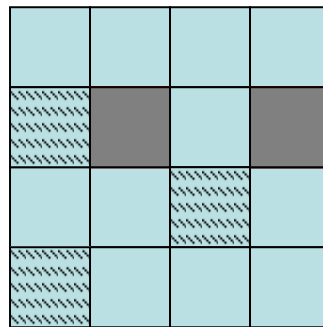
Page request



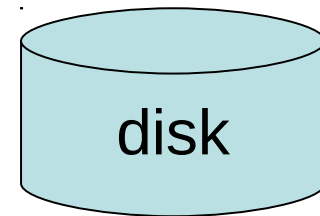
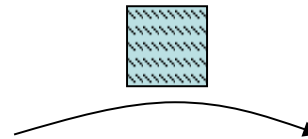
Buffer Management (2/2)

- What if no empty page frame exists:
 - Select victim page
 - Each page associated with dirty flag
 - If page selected dirty, then write it back to disk
- Which page to select?
 - Replacement policies (LRU, MRU)

Page request



buffer pool

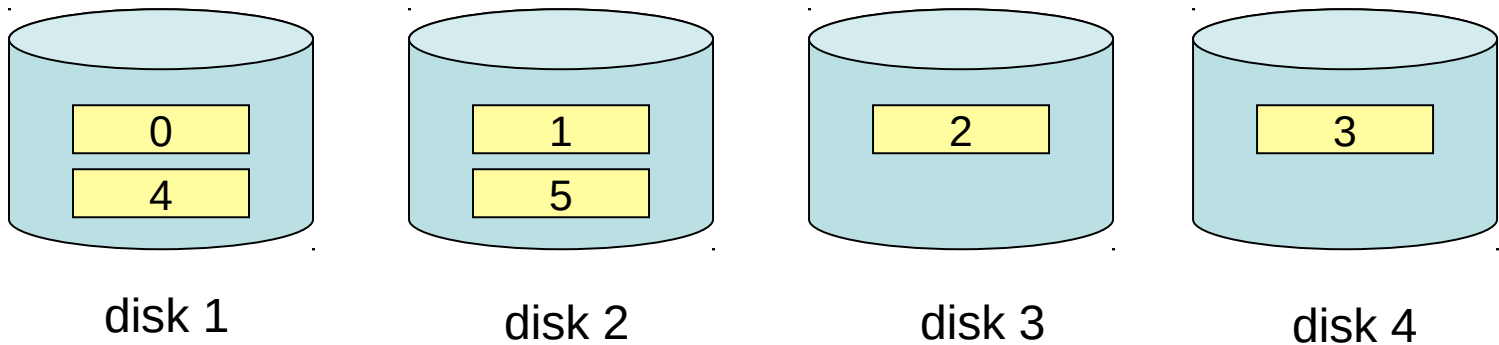


Disk Arrays

- Single disk becomes bottleneck
- Disk arrays
 - instead of single large disk
 - many small parallel disks
 - read N blocks in a single access time
 - concurrent queries
 - tables spanning among disks
- Redundant Arrays of Independent Disks (RAID)
 - 7 levels (0-6)
 - reliability
 - redundancy
 - parallelism

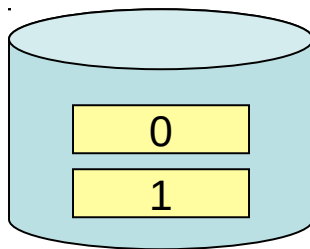
RAID level 0

- Block level striping
- No redundancy
- maximum bandwidth
- automatic load balancing
- best write performance
- but, no reliability

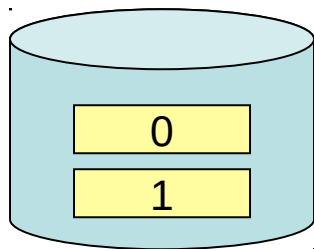


Raid level 1

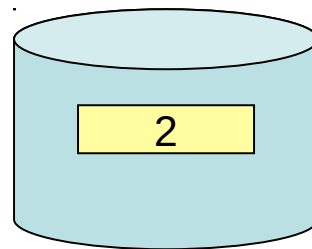
- Mirroring
 - Two identical copies stored in two different disks
- Parallel reads
- Sequential writes
- transfer rate comparable to single disk rate
- most expensive solution



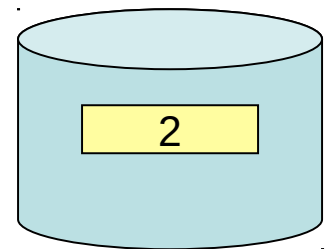
disk 1



disk 2
mirror of disk 1



disk 3



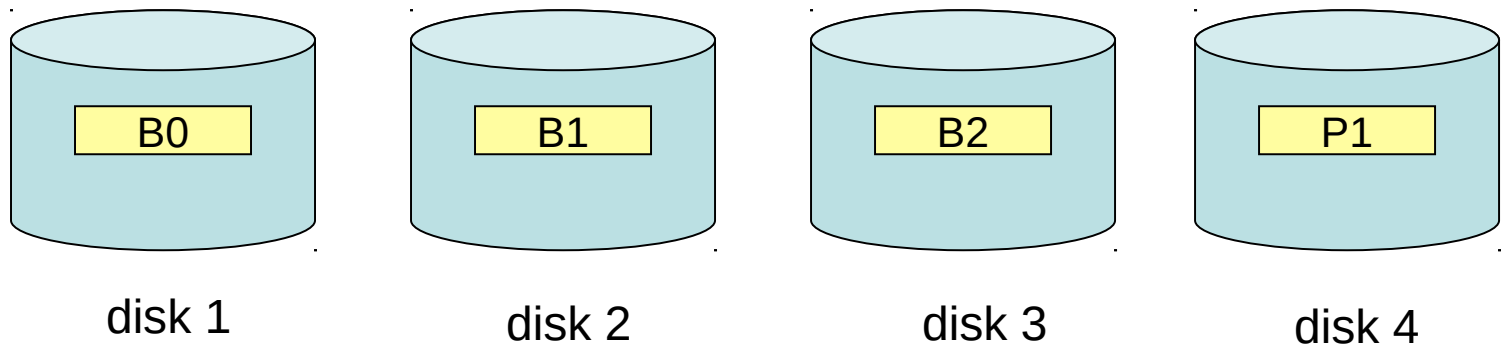
disk 4
mirror of disk 3

RAID levels 2 and 3

- bit level striping (next bit on a separate disk)
- error detection and correction
- RAID 2
 - ECC error correction codes (Hamming code)
 - Bit level striping, several parity bits
- RAID 3
 - Byte level striping, single parity bit
 - error detection by disk controllers (hardware)
- RAID 4
 - Block level striping, single parity bit

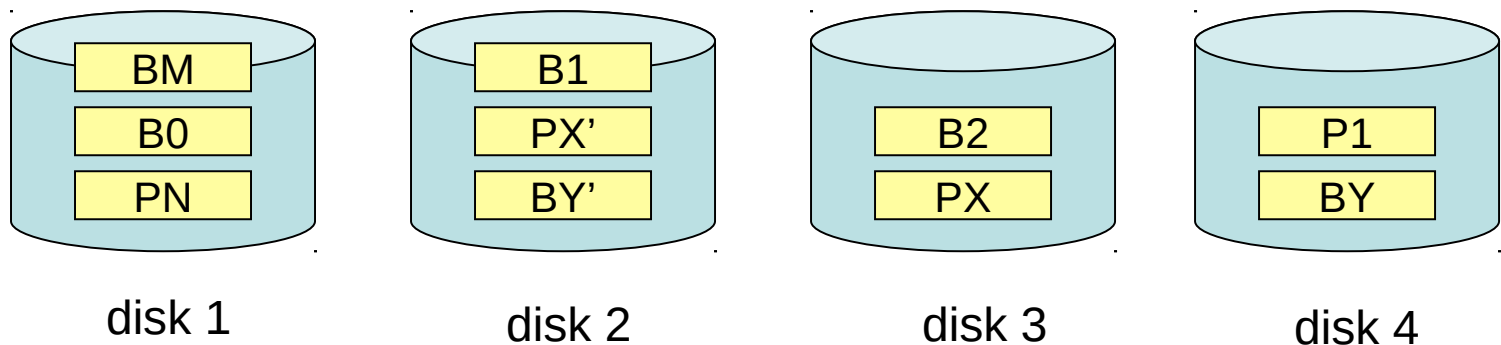
RAID level 4

- block level striping
- parity block for each block in data disks
 - $P1 = B0 \text{ XOR } B1 \text{ XOR } B2$
 - $B2 = B0 \text{ XOR } B1 \text{ XOR } P1$
- an update:
 - $P1' = B0' \text{ XOR } B0 \text{ XOR } P1$ (every update -> must write parity disk)



RAID level 5 and 6

- subsumes RAID 4
- parity disk not a bottleneck
 - parity blocks distributed on all disks
- RAID 6
 - tolerates two disk failures
 - P+Q redundancy scheme
 - 2 bits of redundant data for each 4 bits of data
 - more expensive writes



What pages contain logically?

- Files:
 - Physically partitioned into pages
 - Logically partitioned into **records**
- Each file is a sequence of records
- Each record is a sequence of fields

student	
<u>cid</u>	name
00112233	Paul

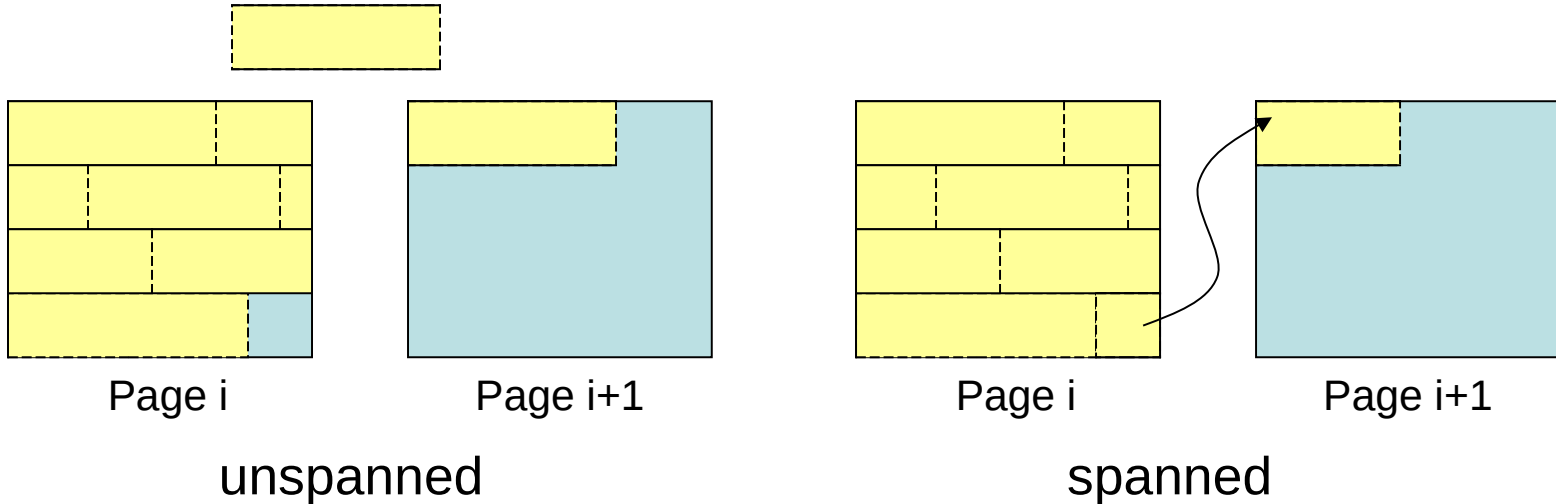
student record:

00112233	Paul
----------	------

$8 + 4 = 12$ Bytes

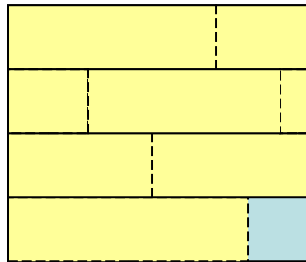
Page Organization

- Student record size: 12 Bytes
- Typical page size: 2 KB
- Record identifiers: <Page identifier, offset>
- How records are distributed into pages:
 - Unspanned organization
 - Blocking factor = $\left\lfloor \frac{pagesize}{recordsize} \right\rfloor$
 - Spanned organization



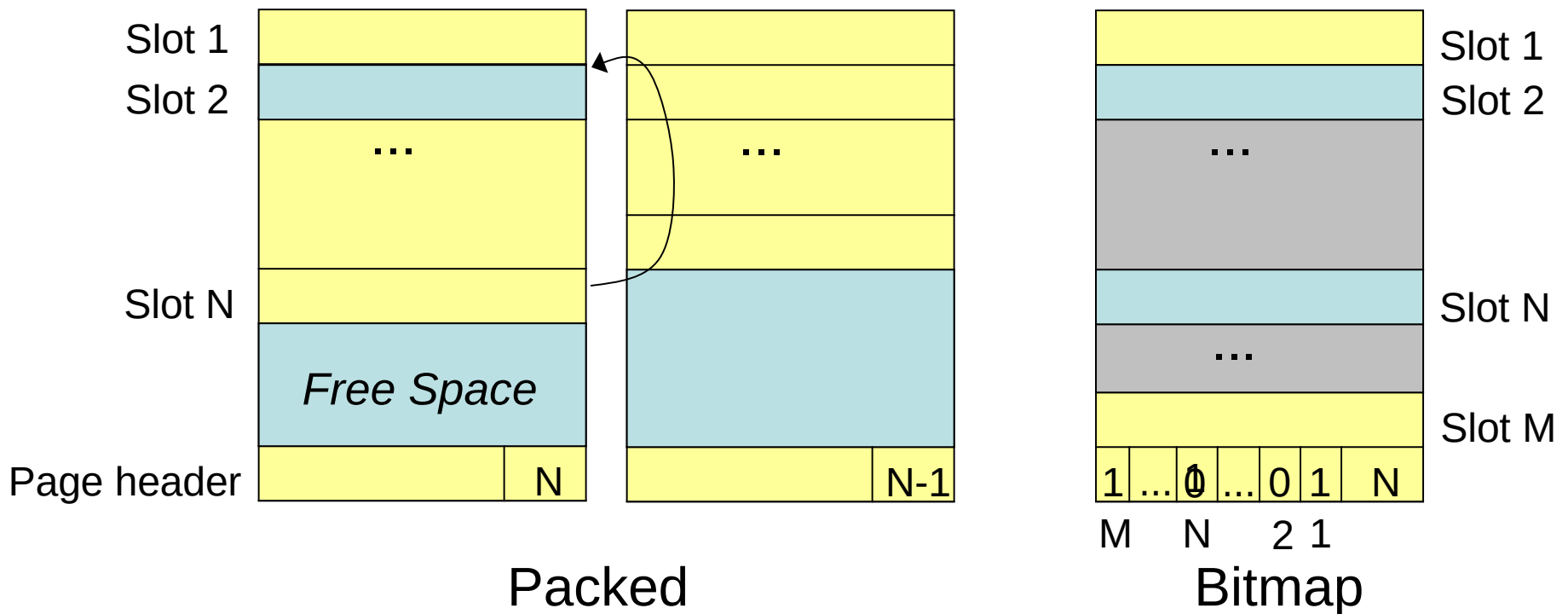
What if a record is deleted?

- Depending on the type of records:
 - Fixed-length records
 - Variable-length records



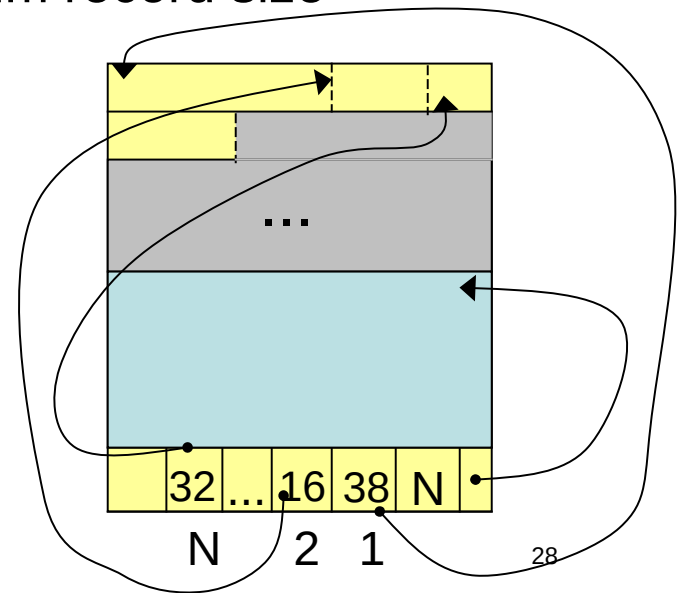
Fixed-length record files

- Upon record deletion:
 - Packed page scheme
 - Bitmap



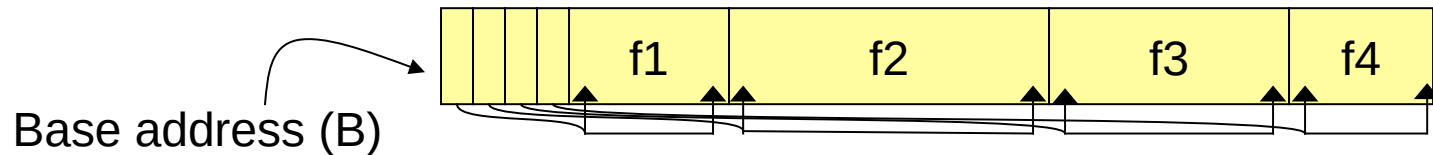
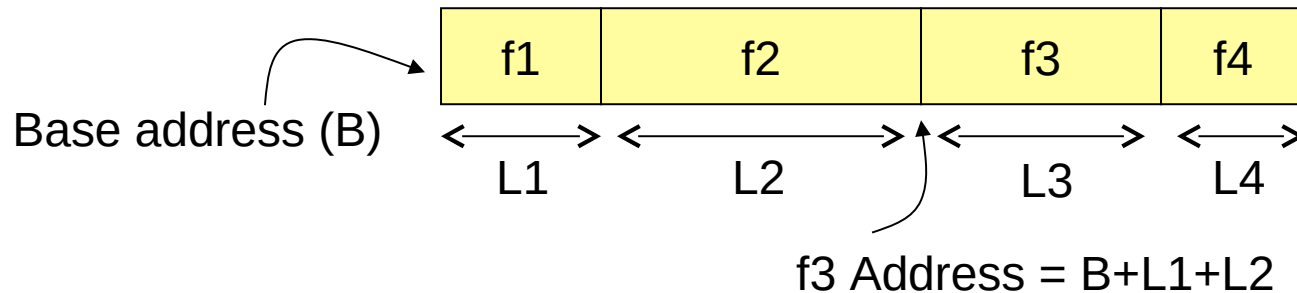
Variable-length record files

- When do we have a file with variable-length records?
 - file contains records of multiple tables
 - create table t (field1 int, field2 varchar2(n))
- Problems:
 - Holes created upon deletion have variable size
 - Find large enough free space for new record
- Could use previous approaches: maximum record size
 - a lot of space wasted
- Use slotted page structure
 - Slot directory
 - Each slot storing offset, size of record
 - Record IDs: page number, slot number



Record Organization

- Fixed-length record formats
 - Fields stored consecutively
- Variable-length record formats
 - Array of offsets
 - NULL values when start offset = end offset



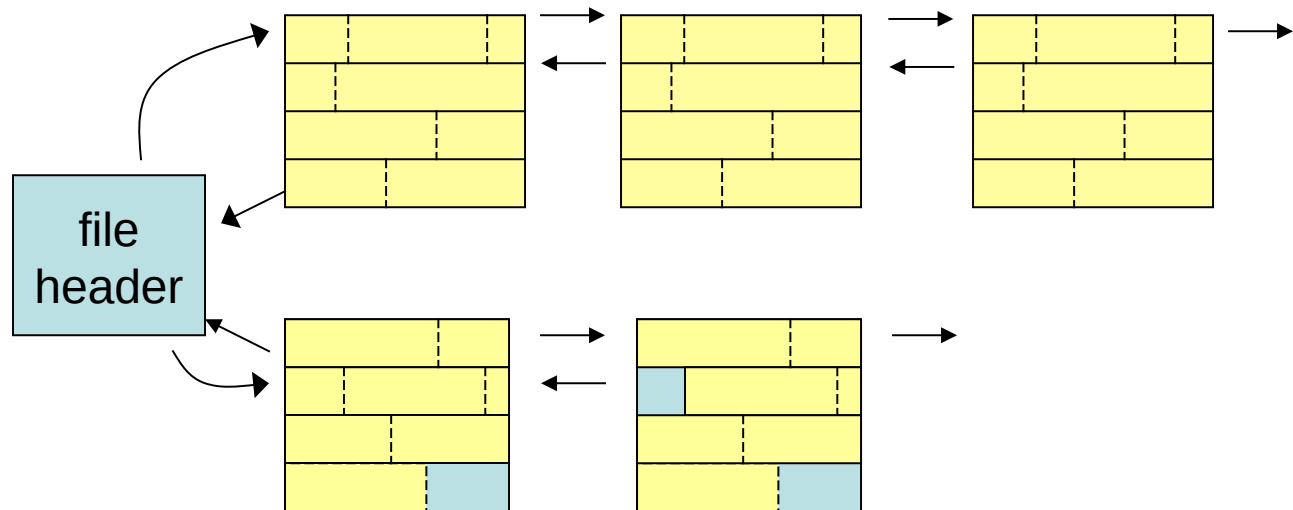
File Organization

(later we study it in a more detailed way)

- Heap files: unordered records
- Sorted files: ordered records
- Hashed files: records partitioned into buckets

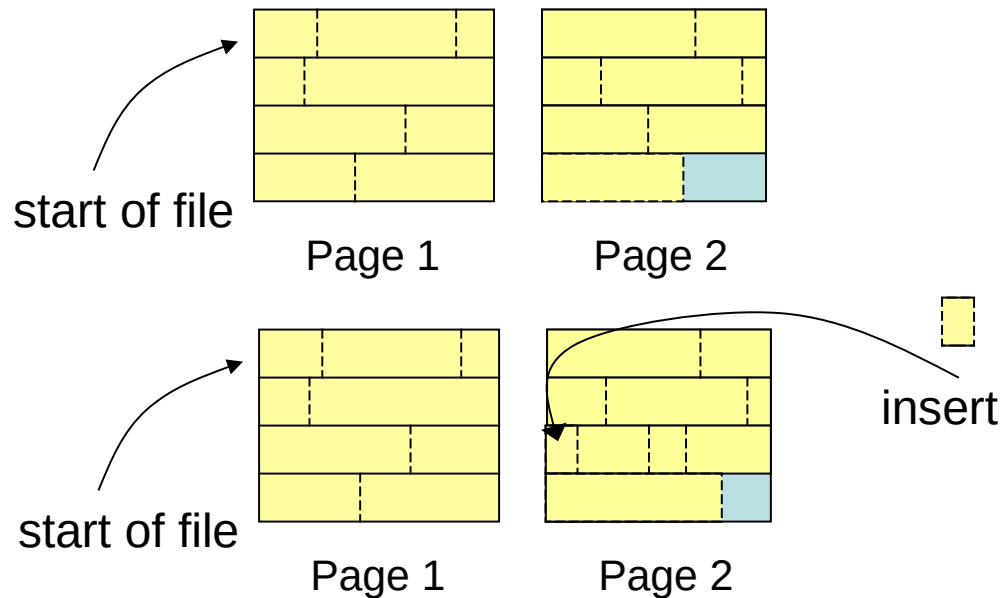
Heap Files

- Simplest file structure
- Efficient insert
- Slow search and delete
 - Equality search: half pages fetched on average
 - Range search: all pages must be fetched



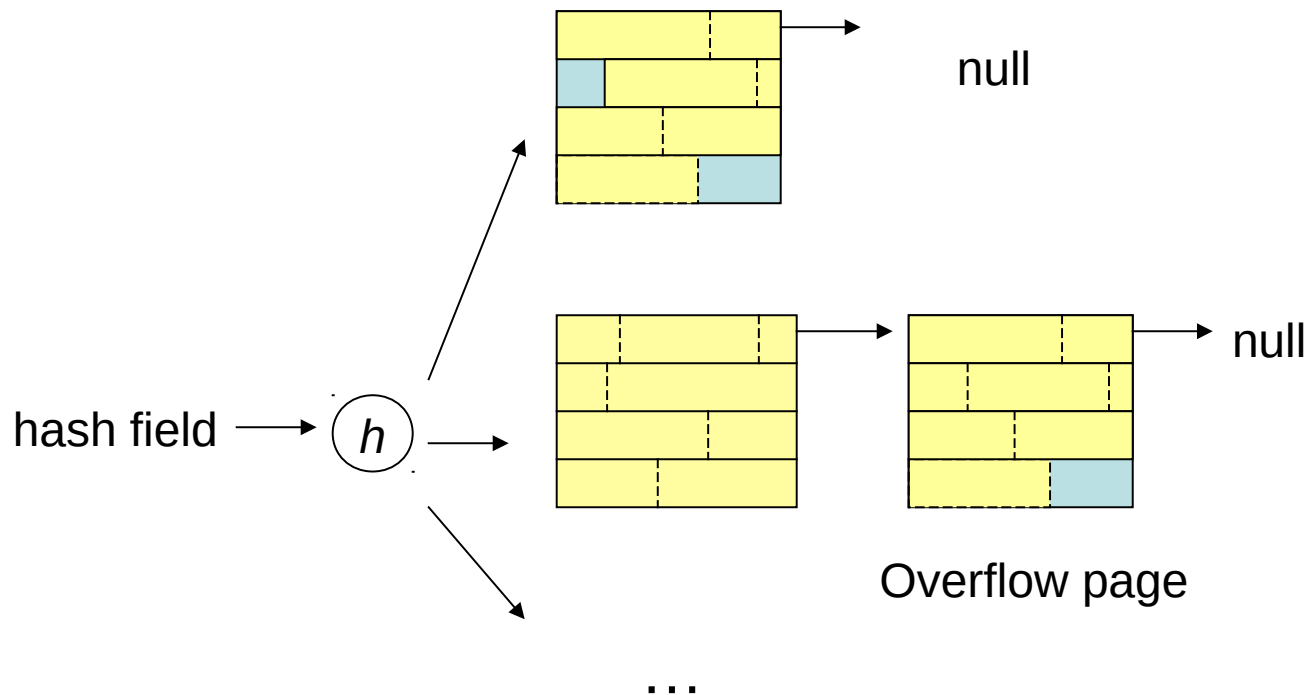
Sorted files

- Sorted records based on **ordering field**
 - If ordering field same as key field, **ordering key field**
- Slow inserts and deletes
- Fast logarithmic search



Hashed Files

- Hash function h on **hash field** distributes pages into buckets
- Efficient equality searches, inserts and deletes
- No support for range searches



Summary (1/2)

- Why Physical Storage Organization?
 - understanding low-level details which affect data access
 - make data access more efficient
- Primary Storage, Secondary Storage
 - memory fast
 - disk slow but non-volatile
- Data stored in files
 - partitioned into pages physically
 - partitioned into records logically
- Optimize I/Os
 - scheduling algorithms
 - RAID
 - page replacement strategies

Summary (2/2)

- File Organization
 - how each file type performs
- Page Organization
 - strategies for record deletion
- Record Organization