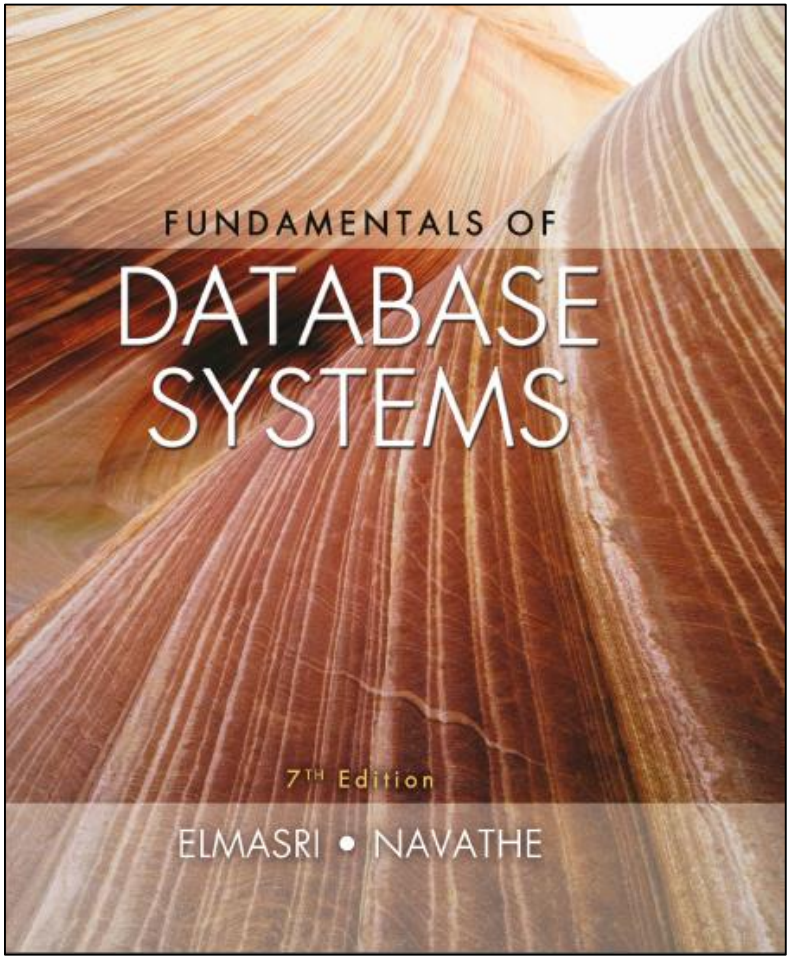
Fundamentals of Database Systems



Chapter 16

Disk Storage, Basic File Structures, Hashing, and Modern Storage Architectures

Disk Storage, Basic File Structures, Hashing, and Modern Storage Architectures

1- Introduction

2- Secondary Storage Devices

3- Buffering of Blocks

4- Placing File Records on Disk

5- Operations on Files

16.1 Introduction

- Databases typically stored on magnetic disks
  - Accessed using physical database file structures
- Storage hierarchy
  - Primary storage
    - CPU main memory, cache memory
  - Secondary storage
    - Magnetic disks, flash memory, solid-state drives
  - Tertiary storage
    - Optical disks (CD-ROMs, DVDs, and other similar storage media) and tapes

Memory Hierarchies and Storage Devices

- Cache memory
  - Static RAM (random access memory)
  - DRAM (dynamic random access memory, main memory)
- Mass storage
  - Magnetic disks (CD-ROM, DVD, tape drives)
- Flash memory
  - Nonvolatile

Storage Types and Characteristics

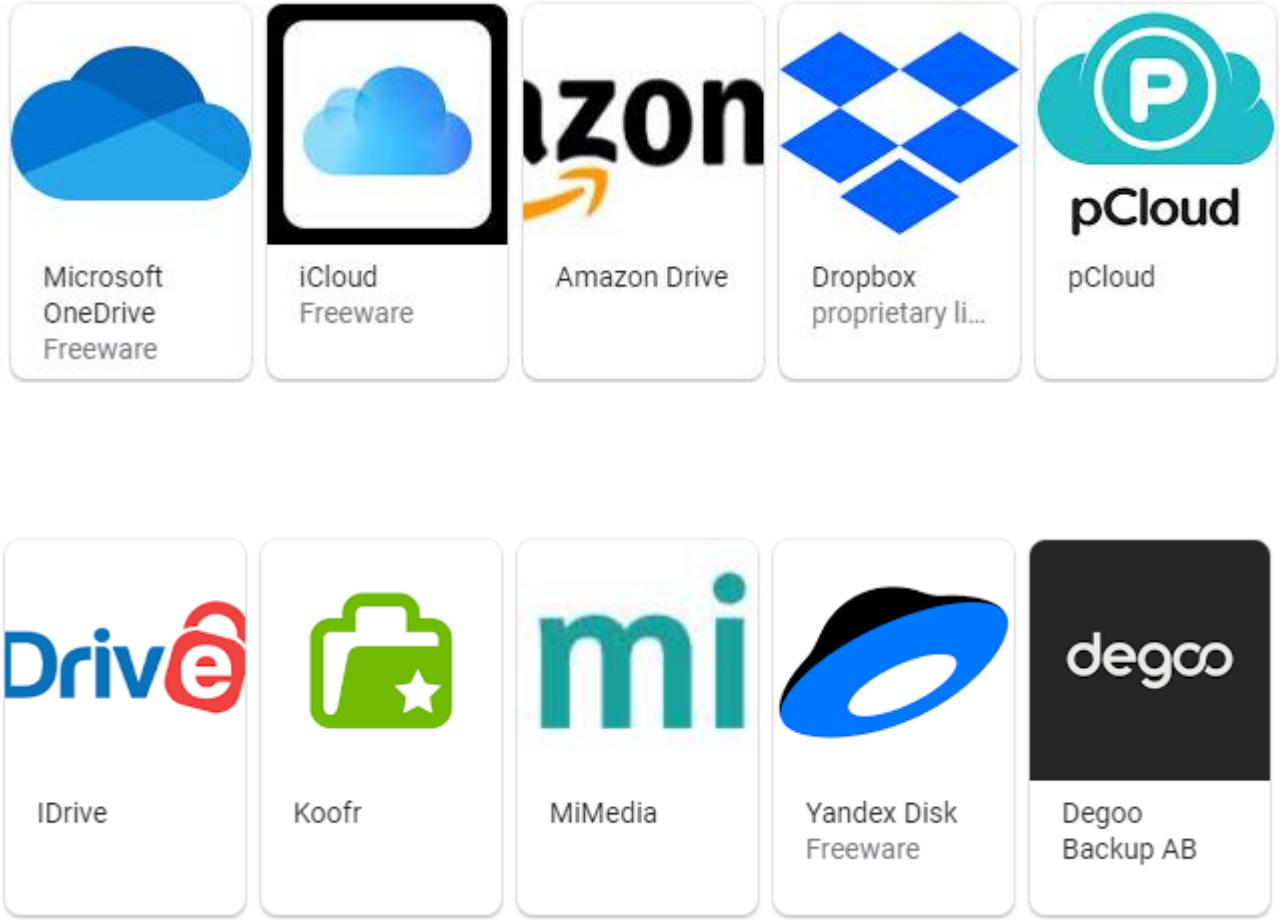
Table 16.1 Types of Storage with Capacity, Access Time, Max Bandwidth (Transfer Speed), and Commodity Cost

Type	Capacity*	Access Time	Max Bandwidth	Commodity Prices (2014)**
Main Memory- RAM	4GB–1TB	30ns	35GB/sec	$100–$20K
Flash Memory- SSD	64 GB–1TB	50μs	750MB/sec	$50–$600
Flash Memory- USB stick	4GB–512GB	100μs	50MB/sec	$2–$200
Magnetic Disk	400 GB–8TB	10ms	200MB/sec	$70–$500
Optical Storage	50GB–100GB	180ms	72MB/sec	$100
Magnetic Tape	2.5TB–8.5TB	10s–80s	40–250MB/sec	$2.5K–$30K
Tape jukebox	25TB–2,100,000TB	10s–80s	250MB/sec–1.2P B/sec	$3K–$1M+

*Capacities are based on commercially available popular units in 2014.

**Costs are based on commodity online marketplaces.

List of cloud storage



Storage Organization of Databases

- Persistent data
  - Most databases
- Transient data
  - Exists only during program execution
- File organization
  - Determines how records are physically placed on the disk
  - Determines how records are accessed

Disk Storage, Basic File Structures, Hashing, and Modern Storage Architectures

1- Introduction

2- Secondary Storage Devices

3- Buffering of Blocks

4- Placing File Records on Disk

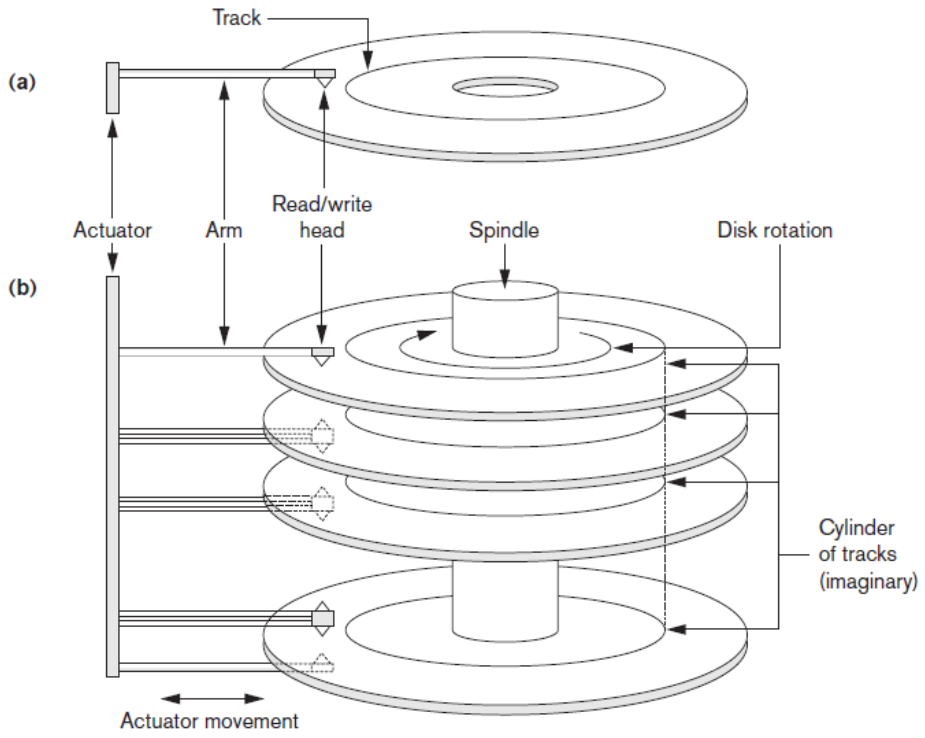
5- Operations on Files

16.2 Secondary Storage Devices (1 of 4)

- Hard disk drive
- Bits (ones and zeros)
  - Grouped into bytes or characters
- Disk capacity measures storage size
- Disks may be single or double-sided
- Concentric circles called tracks
  - Tracks divided into blocks or sectors
- Disk packs
  - Cylinder

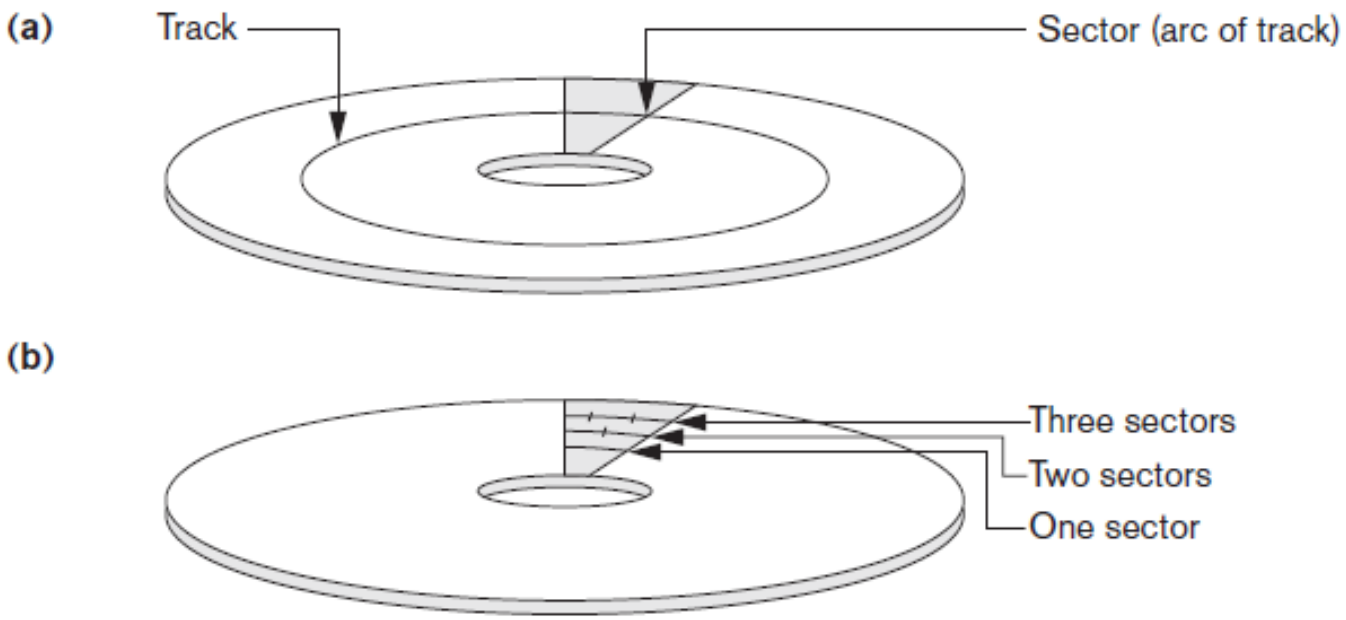
Single-Sided Disk and Disk Pack

Figure 16.1 (a) A single-sided disk with read/write hardware



Sectors on a Disk

(b) Sectors maintaining a uniform recording density



16.2 Secondary Storage Devices (2 of 4)

- Formatting
  - Divides tracks into equal-sized disk blocks
  - Blocks separated by interblock gaps
- Data transfer in units of disk blocks
  - Hardware address supplied to disk I/O hardware
- Buffer
  - Used in read and write operations
- Read/write head
  - Hardware mechanism for read and write operations

16.2 Secondary Storage Devices (3 of 4)

- Disk controller
  - Interfaces disk drive to computer system
  - Standard interfaces
    - SCS I(small computer system interface)
    - SATA (serial advanced technology attachment)
    - SAS (serial attached SCSI)

16.2 Secondary Storage Devices (4 of 4)

- Techniques for efficient data access
  - Data buffering
  - Proper organization of data on disk
  - Reading data ahead of request
  - Proper scheduling of I/O requests
  - Use of log disks to temporarily hold writes
  - Use of SSDs or flash memory for recovery purposes

Solid State Device Storage

- Sometimes called flash storage
- Main component: controller
- Set of interconnected flash memory cards
- No moving parts
- Data less likely to be fragmented
- More costly than HDDs
- DRAM-based SSDs available
  - Faster access times compared with flash

Magnetic Tape Storage Devices

- Sequential access
  - Must scan preceding blocks
- Tape is mounted and scanned until required block is under read/write head
- Important functions
  - Backup
  - Archive

Disk Storage, Basic File Structures, Hashing, and Modern Storage Architectures

1- Introduction

2- Secondary Storage Devices

3- Buffering of Blocks

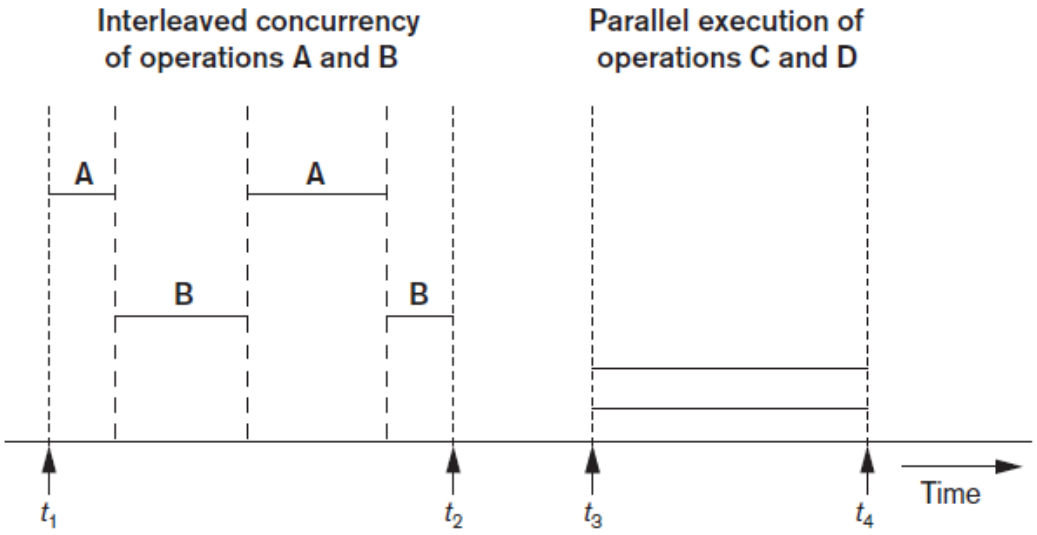
4- Placing File Records on Disk

5- Operations on Files

16.3 Buffering of Blocks (1 of 2)

- Buffering most useful when processes can run concurrently in parallel

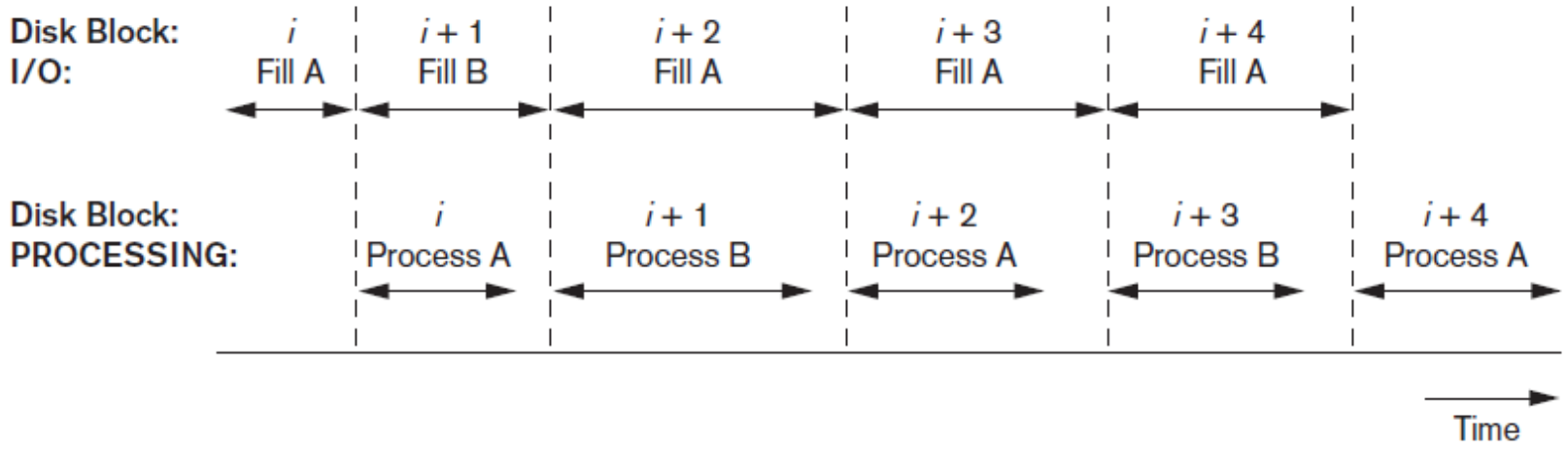
Figure 16.3 Interleaved concurrency versus parallel execution



16.3 Buffering of Blocks (2 of 2)

- Double buffering can be used to read continuous stream of blocks

Figure 16.4 Use of two buffers, A and B, for reading from disk



Buffer Management Strategies

Buffer management information

Pin count: The number of times that page has been requested, or the number of current users of that page. If this count falls to zero, the page is considered **unpinned**. Initially the pin-count for every page is set to zero. Incrementing the pin-count is called **pinning**.

Dirty bit: Initially set to zero for all pages but is set to 1 whenever that page is updated by any application program.

Buffer Replacement Strategies

Buffer management must make sure that the number of buffers fits in main memory.

- Buffer replacement strategies
  - Least recently used (LRU): throw out the page that has not been used (read or written) for the longest time.
  - Clock policy: the buffers are arranged like a circle similar to a clock. Each buffer has a flag with a 0 or 1 value.
  - First-in-first-out (FIFO) : the one that has been occupied the longest by a page is used for replacement.

Disk Storage, Basic File Structures, Hashing, and Modern Storage Architectures

1- Introduction

2- Secondary Storage Devices

3- Buffering of Blocks

4- Placing File Records on Disk

5- Operations on Files

16.4 Placing File Records on Disk

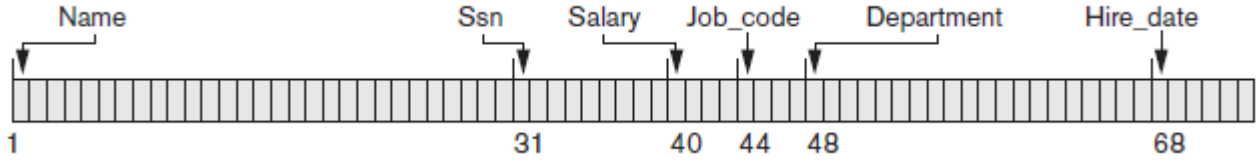
- Record: collection of related data values or items
  - Values correspond to record field
- Data types
  - Numeric
  - String
  - Boolean
  - Date/time
- Binary large objects (BLOBs)
  - Unstructured objects

16.4 Placing File Records on Disk

A **file** is a *sequence* of records.

In many cases, all records in a file are of the same record type.

If every record in the file has exactly the same size (in bytes), the file is said to be made up of **fixed-length records**.



If different records in the file have different sizes, the file is said to be made up of **variable-length records**.

A file may have variable-length records for several reasons:

16.4 Placing File Records on Disk

- Reasons for variable-length records

1. One or more fields have variable length

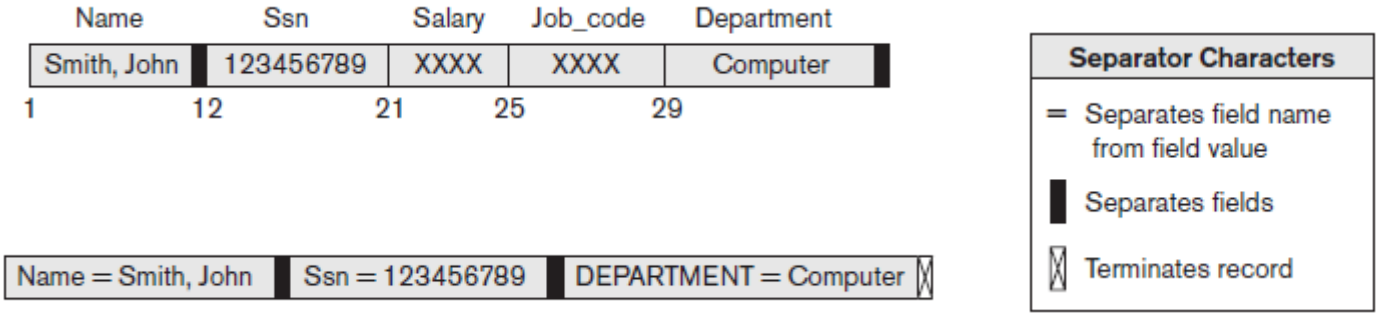
The Name field of EMPLOYEE can be a variable-length field.

2. One or more fields are repeating

3. One or more fields are optional

They may have values for some but not all of the file records

4. File contains records of different types



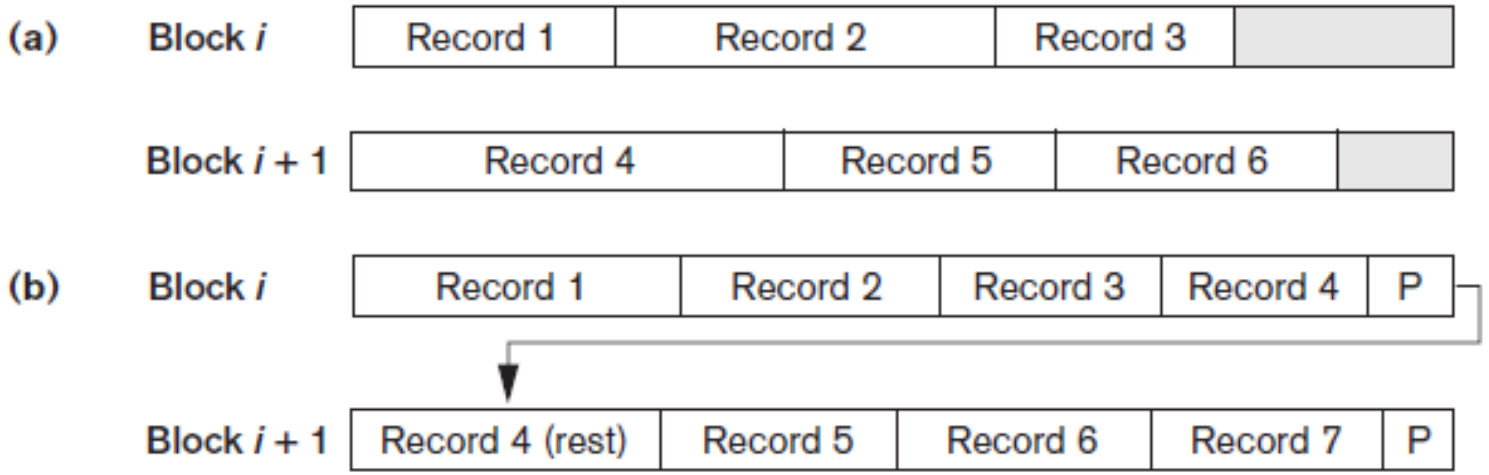
Record Blocking and Spanned Versus Unspanned Records

- File records allocated to disk blocks
- Spanned records
  - Larger than a single block
  - Pointer at end of first block points to block containing remainder of record
- Unspanned
  - Records not allowed to cross block boundaries

Record Blocking and Spanned Versus Unspanned Records

- Blocking factor (*bfr*): Average number of records per block for the file

Suppose that the block size is *B* bytes. For a file of fixed-length records of size *R* bytes, with $B \geq R$, we can fit $bfr = \lfloor B/R \rfloor$ records per block



Types of record organization (a) Unspanned (b) Spanned

Floor and Ceiling
floor(x) is the closest integer less than or equal to x; floor(2.6) = 2, floor(–2.1) = –3
ceiling(x) is the closest integer greater than or equal to x; ceiling(2.6) = 3, and ceiling(–2.1) = –2.

Record Blocking and Spanned Versus Unspanned Records

Allocating file blocks on disk

- Contiguous allocation: The file blocks are allocated to consecutive disk blocks. This makes reading the whole file very fast using double buffering, but it makes expanding the file difficult.

- Linked allocation: Each file block contains a pointer to the next file block. This makes it easy to expand the file but makes it slow to read the whole file.

- Indexed allocation: One or more **index blocks** contain pointers to the actual file blocks.

Record Blocking and Spanned Versus Unspanned Records

- File header (file descriptor)
  - Contains file information needed by system programs
    - Disk addresses
    - Format descriptions

Disk Storage, Basic File Structures, Hashing, and Modern Storage Architectures

1- Introduction

2- Secondary Storage Devices

3- Buffering of Blocks

4- Placing File Records on Disk

5- Operations on Files

16.5 Operations on Files

- Retrieval operations
  - No change to file data
- Update operations
  - File change by insertion, deletion, or modification
- Records selected based on selection condition

When several file records satisfy a search condition, the first record—with respect to the physical sequence of file records—is initially located and designated the current record.

Subsequent search operations commence from this record and locate the next record in the file that satisfies the condition.

16.5 Operations on Files

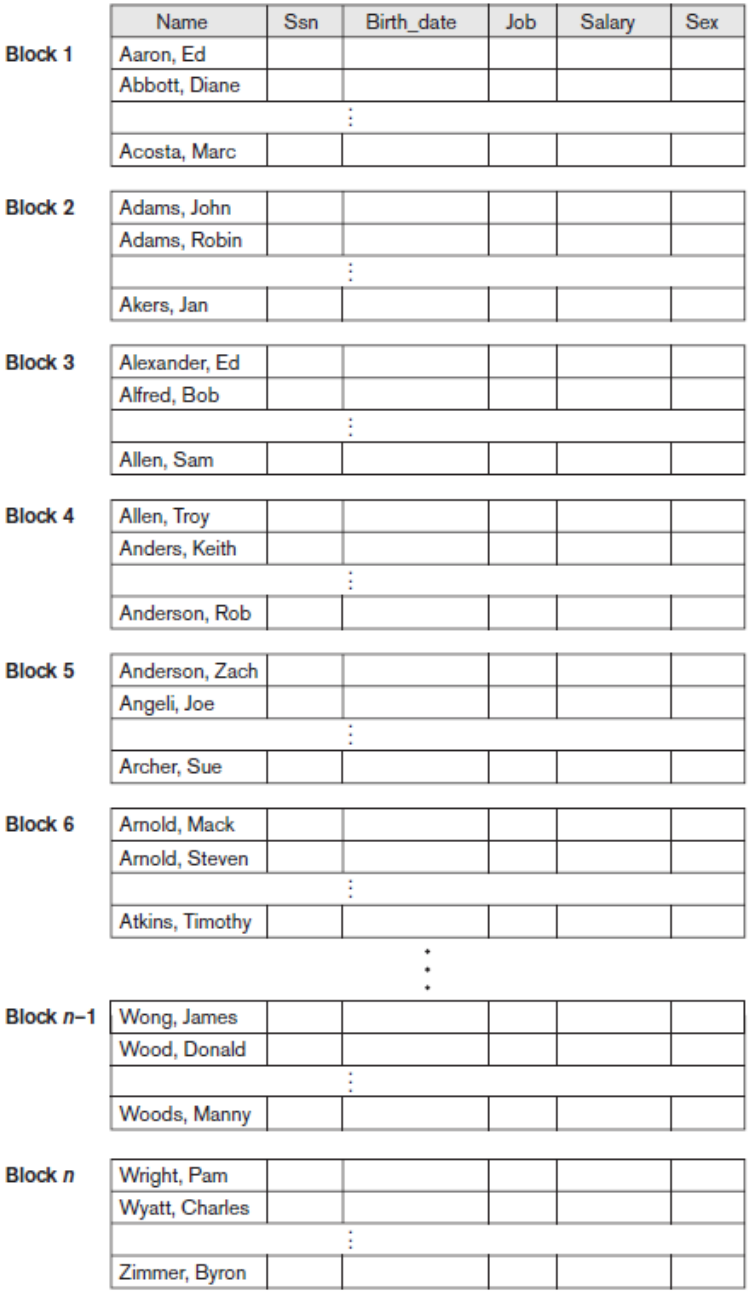
- Examples of operations for accessing file records
  - Open
  - Find
  - Read
  - FindNext
  - Delete
  - Insert
  - Close
  - Scan

16.6 Files of Unordered Records (Heap Files)

- Heap (or pile) file
  - Records placed in file in order of insertion
- Inserting a new record is very efficient
- Searching for a record requires linear search
- Deletion techniques
  - Rewrite the block
  - Use deletion marker

16.7 Files of Ordered Records (Sorted Files)

- Ordered (sequential) file
  - Records sorted by ordering field
    - Called ordering key if ordering field is a key field
- Advantages
  - Reading records in order of ordering key value is extremely efficient
  - Finding next record
  - Binary search technique



Some blocks of an ordered (sequential) file of EMPLOYEE records with Name as the ordering key field.

16.7 Files of Ordered Records (Sorted Files)

A **binary search** for disk files can be done on the blocks rather than on the records.

Suppose that the file has *b* blocks numbered 1, 2, …, *b*; the records are ordered by ascending value of their ordering key field; and we are searching for a record whose ordering key field value is *K*.

Algorithm 16.1. Binary Search on an Ordering Key of a Disk File

```
l ← 1; u ← b; (*b is the number of file blocks*)
while (u ≥ l ) do
    begin i ← (l + u) div 2;
    read block i of the file into the buffer;
    if K < (ordering key field value of the first record in block i )
        then u ← i − 1
    else if K > (ordering key field value of the last record in block i )
        then l ← i + 1
    else if the record with ordering key field value = K is in the buffer
        then goto found
    else goto notfound;
    end;
goto notfound;
```

Access Times for Various File Organizations

Table 16.3 Average access times for a file of *b* blocks under basic file organizations

Type of Organization	Access/Search Method	Average Blocks to Access a Specific Record
Heap (unordered)	Sequential scan (linear search)	$\frac{b}{2}$
Ordered	Sequential scan	$\frac{b}{2}$
Ordered	Binary search	$\log_2 b$

16.12 Summary

- Magnetic disks
  - Accessing a disk block is expensive
- Commands for accessing file records
- File organizations: unordered, ordered, hashed