Chapter 10: Storage and File Structure

Database System Concepts, 6th Ed.

Chapter 10: Storage and File Structure

- Overview of Physical Storage Media
- Magnetic Disks
- File Organization
- Organization of Records in Files
- Data-Dictionary Storage
- Storage Access

Classification of Physical Storage Media

- **Speed** with which data can be accessed
- **Cost** per unit of data
- **Reliability**
  - data loss on power failure or system crash
  - physical failure of the storage device
- Can differentiate storage into:
  - **volatile storage:** loses contents when power is switched off
  - **non-volatile storage**:
    - Contents persist even when power is switched off.
    - Includes secondary and tertiary storage, as well as batter-backed up main-memory.

Physical Storage Media

- **Cache** – fastest and most costly form of storage; volatile; managed by the computer system hardware.
- **Main memory**:
  - fast access (10s to 100s of **nanoseconds**; 1 nanosecond = 10^{-9} seconds)
  - generally too small (or too expensive) to store the entire database
    - capacities of up to a few Gigabytes widely used currently
    - Capacities have gone up steadily and rapidly (roughly factor of 2 every 2 to 3 years)
  - **Volatile** — contents of main memory are usually lost if a power failure or system crash occurs.

Physical Storage Media (Cont.)

- **Flash memory**
  - Data survives power failure
  - Data can be written at a location only once, but *location can be erased and written to again*
    - Can support only a limited number (10K – 1M) of write/erase cycles.
  - Reads are roughly as fast as main memory
  - But writes are slow (few **microseconds**), erase is slower

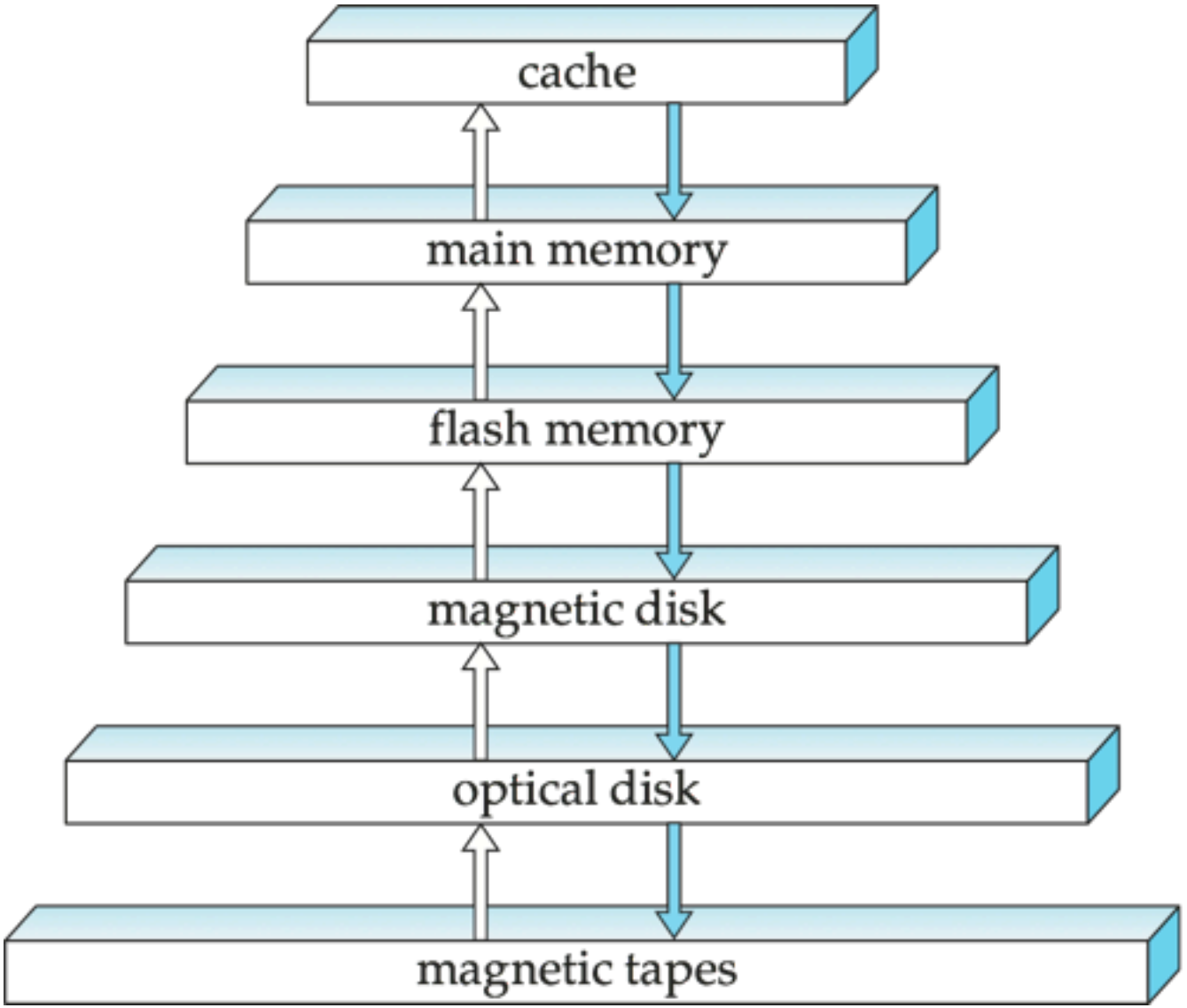
Physical Storage Media (Cont.)

- **Magnetic-disk**
  - Data is stored on spinning disk, and read/written magnetically
  - Primary medium for the long-term storage of data
  - Data must be moved from disk to main memory for access, and written back for storage
    - Much slower access than main memory
  - **direct-access** – possible to read data on disk in any order, unlike magnetic tape
  - Capacities range up to roughly 3 TB as of 2012
    - Much larger capacity/cost than main memory/flash memory
    - Growing constantly and rapidly (factor of 2 to 3 every 2 years)
  - Survives power failures and system crashes
    - disk failure can destroy data, but is rare

Physical Storage Media (Cont.)

- **Tape storage**
  - non-volatile, used primarily for backup (to recover from disk failure), and for archival data
  - **sequential-access** – much slower than disk
  - tape can be removed from drive ⇒ storage costs much cheaper than disk, but drives are expensive
  - Tape jukeboxes available for storing massive amounts of data
    - hundreds of terabytes (1 terabyte = 10^9 bytes) to even multiple **petabytes** (1 petabyte = 10^{12} bytes)

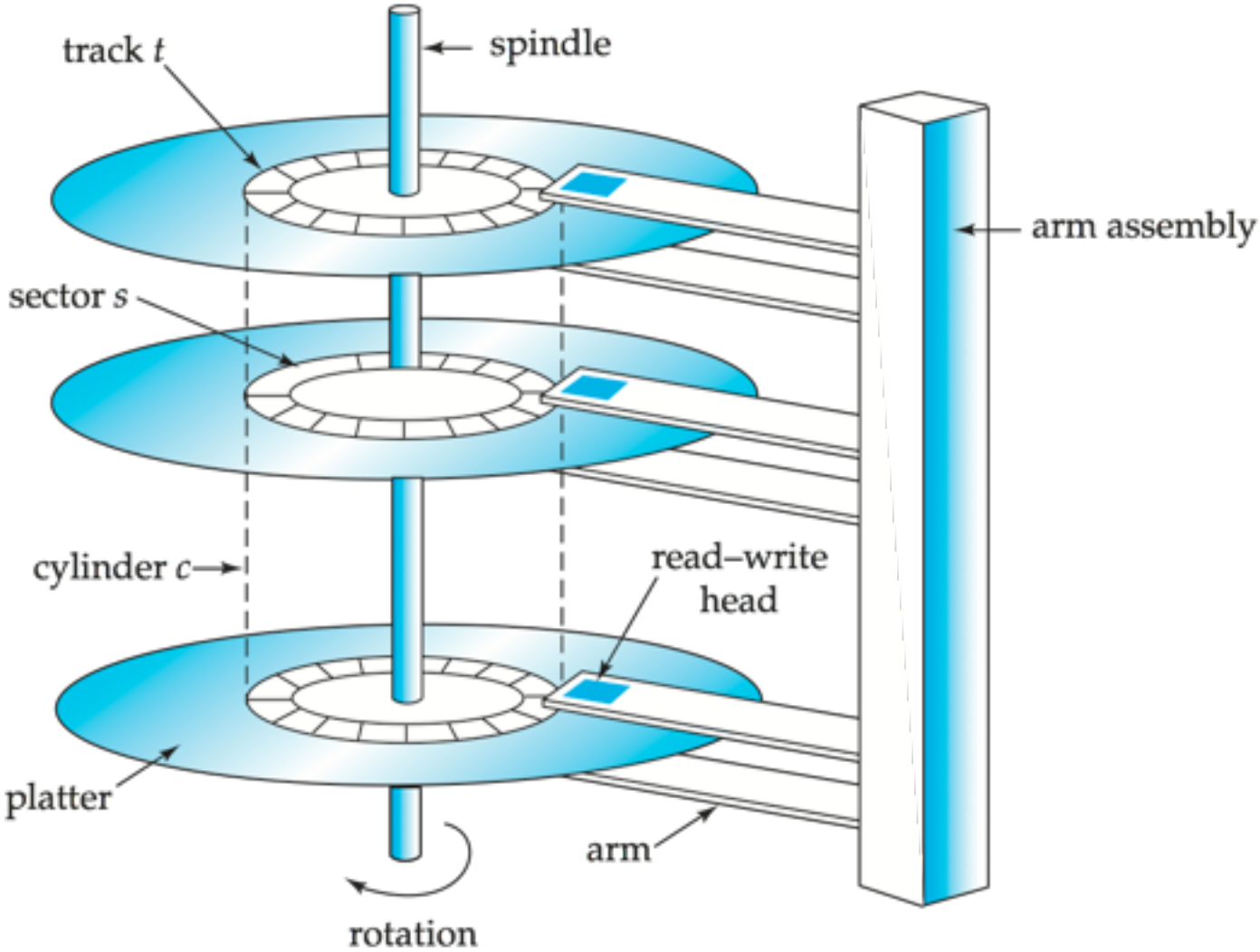
Storage Hierarchy



Chapter 10: Storage and File Structure

- Overview of Physical Storage Media
- **Magnetic Disks**
- File Organization
- Organization of Records in Files
- Data-Dictionary Storage
- Storage Access

Magnetic Hard Disk Mechanism



NOTE: Diagram is schematic, and simplifies the structure of actual disk drives

Performance Measures of Disks

- **Access time** – the time it takes from when a read or write request is issued to when data transfer begins. Consists of:
  - **Seek time** – time it takes to reposition the arm over the correct track.
    - 4 to 10 **milliseconds** on typical disks
  - **Rotational latency** – time it takes for the sector to be accessed to appear under the head.
    - 4 to 11 **milliseconds** on typical disks (5400 to 15000 r.p.m.)
- **Data-transfer rate** – the rate at which data can be retrieved from or stored to the disk.
  - 25 to 100 **MB per second** max rate, lower for inner tracks

Optimization of Disk-Block Access

- **Block** – a contiguous sequence of sectors from a single track
  - data is transferred between disk and main memory in blocks
  - Typical block sizes today range from 4 to 16 kilobytes

Optimization of Disk-Block Access

- **Block** – a contiguous sequence of sectors from a single track
  - data is transferred between disk and main memory in blocks
  - Typical block sizes today range from 4 to 16 kilobytes
    - Smaller blocks: more transfers from disk
    - Larger blocks: more space wasted due to partially filled blocks

Chapter 10: Storage and File Structure

- Overview of Physical Storage Media
- Magnetic Disks
- File Organization
- Organization of Records in Files
- Data-Dictionary Storage
- Storage Access

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.

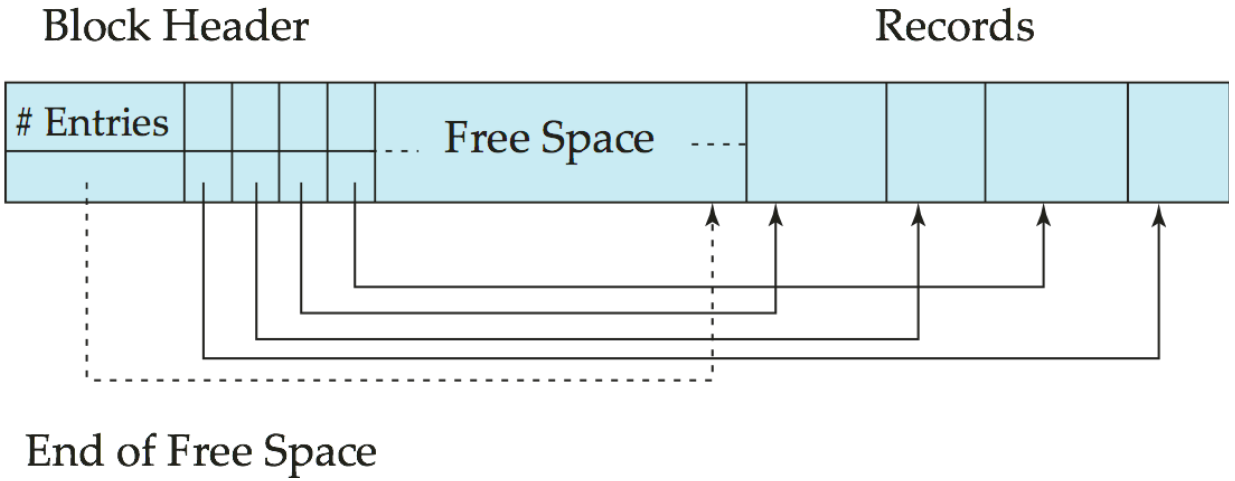
Fixed-Length Records

- Simple approach:
  - Store record *i* starting from byte $n * (i - 1)$, where *n* is the size of each record.

- Deletion of record *i:* alternatives*:*
  - move records $i + 1, \ldots, n$ to $i, \ldots, n - 1$
  - move record *n* to *i*
  - do not move records, but link all free records on a *free list*

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

Variable-Length Records: Slotted Page Structure



- **Slotted page** header contains:
  - number of record entries
  - end of free space in the block
  - size and location of each record
- compaction is possible
  - Records can be moved around within a page to keep them contiguous; entry in the header must be updated.

Chapter 10: Storage and File Structure

- Overview of Physical Storage Media
- Magnetic Disks
- File Organization
- **Organization of Records in Files**
- Data-Dictionary Storage
- Storage Access

Organization of Records in Files

- **Heap** – a record can be placed anywhere in the file where there is space
- **Sequential** – store records in sequential order, based on the value of the **search key** of each record
- **Hashing** – a hash function computed on some attribute of each record; the result specifies in which block of the file the record should be placed

- **multitable clustering file organization** – records of several different relations can be stored in the same file
  - Motivation: store related records on the same block to minimize I/O

Chapter 10: Storage and File Structure

- Overview of Physical Storage Media
- Magnetic Disks
- File Organization
- Organization of Records in Files
- **Data-Dictionary Storage**
- Storage Access

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 (Chapter 11)

Chapter 10: Storage and File Structure

- Overview of Physical Storage Media
- Magnetic Disks
- File Organization
- Organization of Records in Files
- Data-Dictionary Storage
- Storage Access

Storage Access

- A database file is partitioned into fixed-length storage units called **blocks**. Blocks are units of data storage and transfer.
- Database system seeks to minimize the number of block transfers between the disk and memory.
- We can reduce the number of disk accesses by keeping as many blocks as possible in main memory.
- **Buffer** – portion of main memory available to store copies of disk blocks.
- **Buffer manager** – subsystem responsible for allocating buffer space in main memory.

Buffer Manager

- Programs call on the buffer manager when they need a block from disk.
  1. If the block is already in the buffer, buffer manager returns the address of the block in main memory
  2. If the block is not in the buffer, the buffer manager
     1. Allocates space in the buffer for the block
        1. If no more memory space, replace (throw out) some other block to make space for the new block.
        2. Replaced block is written back to disk only if it is **dirty** (modified since the most recent time that it was written to/fetched from the disk).
     2. Reads the desired block from the disk to the buffer, and returns the address of the block in main memory to requester.

Buffer-Replacement Policies

- Most **operating systems** replace the block **least recently used** (LRU strategy)
- Idea behind LRU – unpopular blocks will be unpopular in the future.
- LRU can be a bad strategy for certain access patterns involving repeated scans of data
  - For example: when computing the join of 2 relations r and s by a nested loops
    for each tuple *tr* of *r* do
      for each tuple *ts* of *s* do
        if the tuples *tr* and *ts* match …
- Strategy based on expected data access patterns (often can be known from the query optimizer) is preferable

Buffer-Replacement Policies (Cont.)

- **Pinned block** – memory block that is not allowed to be written back to disk.
- **Toss-immediate** strategy – frees the space occupied by a block as soon as the final tuple of that block has been processed
- Buffer managers also support **forced output** of blocks for the purpose of recovery (more in Chapter 16)

Chapter 10: Storage and File Structure

- Overview of Physical Storage Media
- Magnetic Disks
- File Organization
- Organization of Records in Files
- Data-Dictionary Storage
- Storage Access

End of Chapter 10

Database System Concepts, 6th Ed.