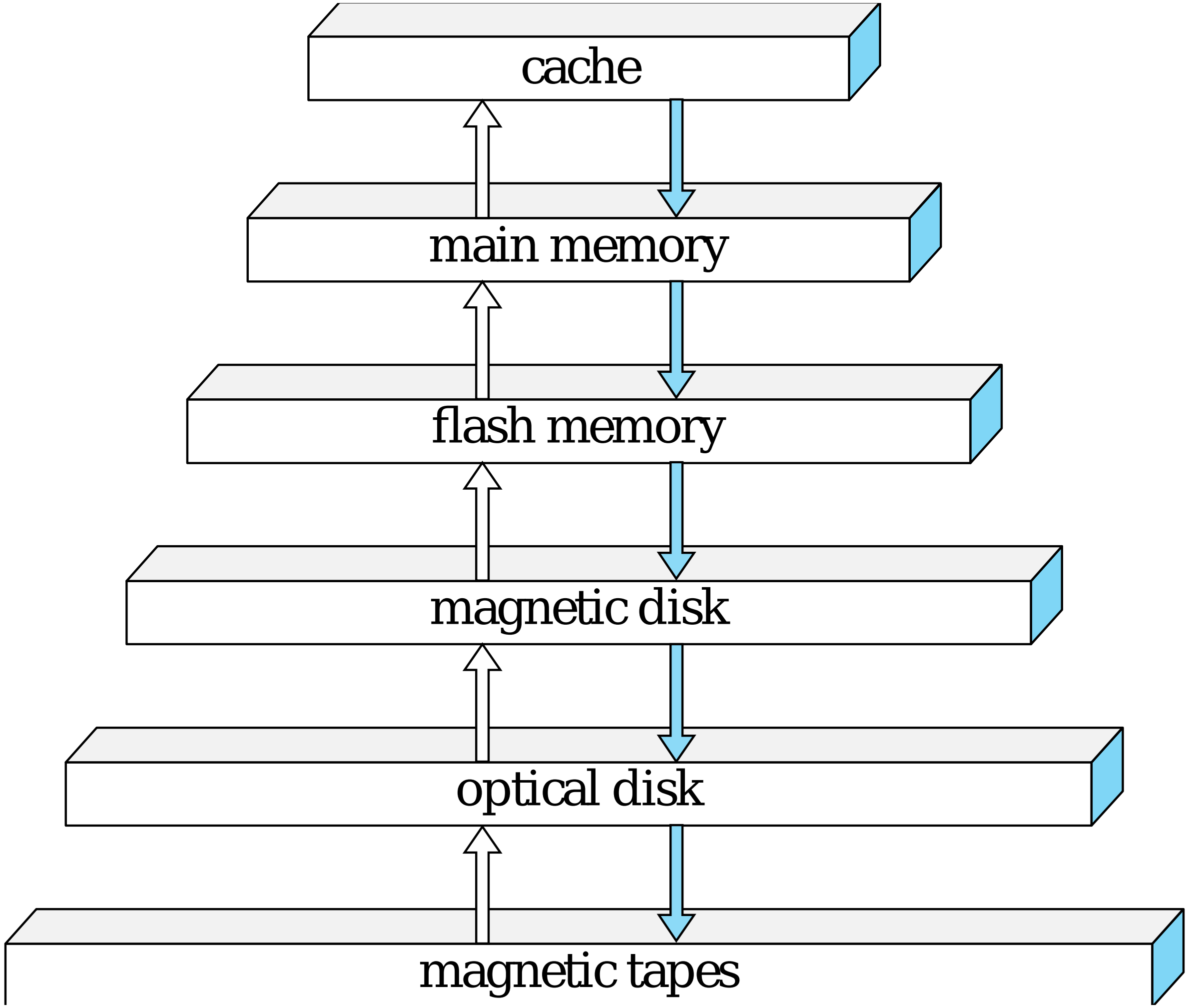
# Module 10:  Physical Storage Systems

Database System Concepts, 7th Ed.

# Storage Hierarchy

- **Primary storage:** Fastest media but volatile (cache, main memory).
- **Secondary storage:** next level in hierarchy, non-volatile, moderately fast access time
  - Also called **on-line storage**
  - Examples: flash memory, magnetic disks
- **Tertiary storage:** lowest level in hierarchy, non-volatile, slow access time
  - Also called **off-line storage**
  - Examples: magnetic tape, optical storage

# Storage Hierarchy



# Storage Interfaces

- Magnetic disks and flash-based solid-state disks are connected to a computer system through a high-speed interconnection.
- Disks typically support either the Serial  ATA  (SATA) interface, or the Serial Attached SCSI  (SA) interface.
- Storage area network (SAN)  architecture, allows  a large numbers of disks  to be appear as a single large disk and these are connected by a high-speed network to a number of server computers.
- Network attached storage (NAS) is similar to SAN except that provides a file system interface using networked file system protocols (e.g., CIFS)
- Cloud storage allows data  to be is stored in the cloud and accessed via an API.

# Magnetic Disks

- Magnetic disks provide the bulk of secondary storage for modern computer systems.
- A disk consists of a number of disk platters
  - Each disk platter has a flat, circular shape.  Its two surfaces are covered with a magnetic material, and information is recorded on the surfaces.
  - Platters are made from rigid metal or glass.
- When the disk is in use, a drive motor spins it at a constant high speed, typically 5,400 to 10,000 revolutions per minute (RPM).

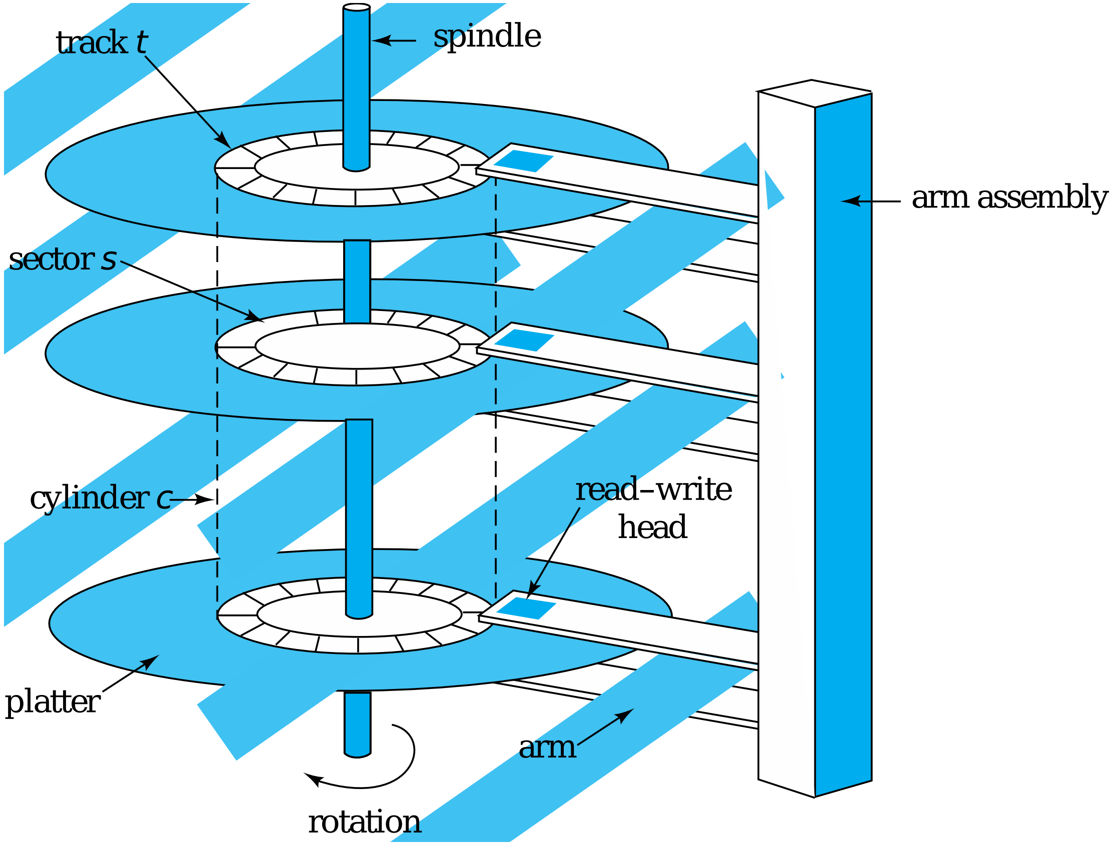
# Magnetic Disks (Cont.)

- Surface of platter divided into circular **tracks**
  - Over 50K-100K tracks per platter on typical hard disks
- Each track is divided into **sectors.**
  - A sector is the smallest unit of data that can be read or written.
  - Sector size typically 512 bytes
  - Current generation disks have between 2 billion and 24 billion sectors.
  - The inner tracks (closer to the spindle) are of smaller length than the outer tracks, and the outer tracks contain more sectors than the inner tracks.
- **Read-write head**
  - Positioned very close to the platter surface (almost touching it)
  - Reads or writes magnetically encoded information.

# Magnetic Disks (Cont.)

- To read/write a sector
  - Disk arm swings to position head on right track
  - Platter spins continually; data is read/written as sector passes under head
- Head-disk assemblies
  - Multiple disk platters on a single spindle (1 to 5 usually)
  - One head per platter, mounted on a common arm.
- **Cylinder** *i* consists of *i*th track of all the platters

# 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.
- **Data-transfer rate** – the rate at which data can be retrieved from or stored to the disk
- **Mean time to failure (MTTF)** – the average time the disk is expected to run continuously without any failure

# Access Time

- **Seek time** – time it takes to reposition the arm over the correct track.
  - Average seek time is 1/2 the worst case seek time.
    - Would be 1/3 if all tracks had the same number of sectors, and we ignore the time to start and stop arm movement
  - 4 to 10 milliseconds on typical disks
- **Rotational latency** – time it takes for the sector to be accessed to appear under the head.
  - Average latency is 1/2 of the worst case latency.
  - 4 to 11 milliseconds on typical disks (5,400 to 15,000 RPM.)

# Data-transfer rate

- 25 to 100 MB per second max rate, lower for inner tracks
- Multiple disks may share a controller, so rate that controller can handle is also important
  - For Example:
    - SATA: 150 MB/sec, SATA-II 3Gb (300 MB/sec)
    - Ultra 320 SCSI: 320 MB/s, SAS (3 to 6 Gb/sec)
    - Fiber Channel (FC2Gb or 4Gb): 256 to 512 MB/s

# Mean time to failure (MTTF)

- Typically 3 to 5 years
- Probability of failure of new disks is quite low, corresponding to a “theoretical MTTF” of 500,000 to 1,200,000 hours for a new disk
  - An MTTF of 1,200,000 hours for a new disk means that given 1000 relatively new disks, on an average one will fail every 1200 hours
- MTTF decreases as disk ages

# Flash Memory

- There are two types of flash memory -- NOR flash and NAND flash.
- NAND flash is the variant that is predominantly used for data storage, since it is much cheaper than NOR flash
- Requires page-at-a-time read (page: 512 bytes to 4 KB)
- Transfer rate around 20 MB/sec
- Erase is very slow (1 to 2 millisecs)
  - Erase block contains multiple pages
  - **Remapp**ing of logical page addresses to physical page addresses avoids waiting for erase
    - **Translation table** tracks mapping
      - also stored in a label field of flash page
    - remapping carried out by **flash translation layer**
  - After 100,000 to 1,000,000 erases, erase block becomes unreliable and cannot be use

# Solid-state disks

- Solid-state disks SSDs are built using NAND flash and provide the same block-oriented interface as disk storage.
- Compared to magnetic disks, SSDs can provide much faster random access:
  - Transfer rate of 100 to 200 MB/sec
  - The latency to retrieve a page of data ranges from 20 to 100 microseconds for SSDs,
    - A random access on disk would take 5 to 10 milliseconds.
- The data transfer rate of SSDs is higher than that of magnetic disks and is usually limited by the interconnect technology;
  - Transfer rates range from around 500 megabytes per second up to 3 gigabytes per second.
- The power consumption of SSDs is also significantly lower than that of magnetic disks.

# RAID

# Redundant Arrays of Independent Disks (RAID)

- Disk organization techniques that manage a large numbers of disks, providing a view of a single disk of
  - High capacity and high speed  by using multiple disks in parallel,
  - High reliability by storing data redundantly, so that data can be recovered even if  a disk fails
- Originally a cost-effective alternative to large, expensive disks
  - I in RAID originally stood for  “inexpensive”
  - Today RAIDs are used for their higher reliability and bandwidth and the “ I” is interpreted as “independent”

# RAID (Cont.)

- The chance that some disk out of a set of *N* disks will fail is much higher than the chance that a specific single disk will fail.
  - For example, a system with 100 disks, each with MTTF of 100,000 hours (approx.  11 years), will have a system MTTF of 1000 hours (approx. 41 days)
  - Techniques for using redundancy to avoid data loss are critical with large numbers of disks
- **Redundancy** – store extra information that can be used to rebuild information lost in a disk failure

# Improvement Redundancy

- **Mirroring**
  - Duplicate every disk.  Logical disk consists of two physical disks.
  - Every write is carried out on both disks
    - Reads can take place from either disk
  - If one disk in a pair fails, data still available in the other
    - Data loss would occur only if a disk fails, and its mirror disk also fails before the system is repaired
- **Mean time to data loss**
  - Depends on mean time to failure, and **mean time to repair**
  - E.g. MTTF of 100,000 hours, mean time to repair of 10 hours gives mean time to data loss of $500*10^6$ hours (or 57,000 years) for a mirrored pair of disks (ignoring dependent failure modes)

# Improvement in Performance via Parallelism

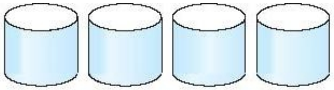
- Two main goals of parallelism in a disk system:
  1. Load balance multiple small accesses to increase throughput
  2. Parallelize large accesses to reduce response time.
- Improve transfer rate by striping data across multiple disks.
- **Bit-level striping** – split the bits of each byte across multiple disks
  - In an array of eight disks, write bit *i* of each byte to disk *i.*
  - Each access can read data at eight times the rate of a single disk.
  - But seek/access time worse than for a single disk
    - Bit level striping is not used much any more
- **Block-level striping** – with *n* disks, block *i* of a file goes to disk (*i* mod *n*) + 1
  - Requests for different blocks can run in parallel if the blocks reside on different disks
  - A request for a long sequence of blocks can utilize all disks in parallel

# RAID Levels

- Schemes to provide redundancy at lower cost by using disk striping combined with parity bits
  - Different RAID organizations, or RAID levels, have differing cost, performance and reliability characteristics
- There are There are 7 different RAID levels, numbered 0 to 6;
  - Levels 2, 3, and 4 are not used in practice anymore and are not covered in the text.
- We concentrate on:
  - Level 0: Block striping; non-redundant
  - Level 1: Mirrored disks with block striping
  - Level 5: Block-Interleaved Distributed Parity
  - Level 6: The P + Q redundancy scheme

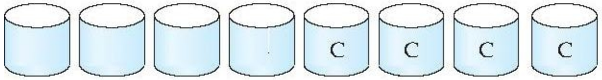
# RAID Levels

- **RAID Level 0**: Block striping; non-redundant.
  - Used in high-performance applications where data loss is not critical



(a) RAID 0: nonredundant striping

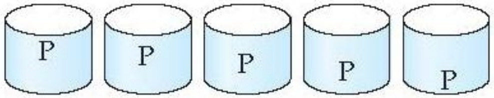
- **RAID Level 1**: Mirrored disks with block striping
  - Offers best write performance.
  - Popular for applications such as storing log files in a database system.



(b) RAID 1: mirrored disks

# RAID Level 5

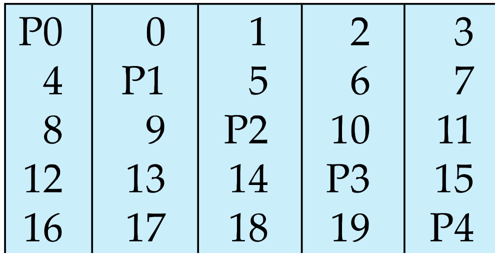
- **RAID Level 5:** Block-Interleaved Distributed Parity.
  - The data and parity are partitioned among all N + 1 disks, rather than storing data in *N* disks and parity in 1 disk.
- Example: with 5 disks, parity block for *n*th set of blocks is stored on disk (*n mod* 5) + 1, with the data blocks stored on the other 4 disks.
- Setup



RAID 5: block-interleaved distributed parity

# RAID Level 5 (Cont.)

- Example. with an array of five disks
  - The parity block, labeled P*k*, for logical blocks 4*k*, 4*k*+1, 4*k*+2, 4*k*+3 is stored in disk k mod 5
  - The corresponding blocks of the other four disks store the four data blocks 4*k* to 4*k*+3.
  - The following table

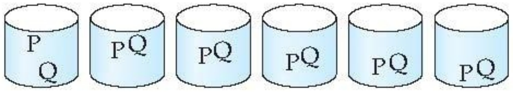


| | | | | |
|---|---|---|---|---|
| P0 | 0 | 1 | 2 | 3 |
| 4 | P1 | 5 | 6 | 7 |
| 8 | 9 | P2 | 10 | 11 |
| 12 | 13 | 14 | P3 | 15 |
| 16 | 17 | 18 | 19 | P4 |

indicates how the first 20 blocks, numbered 0 to 19, and their parity blocks are laid out. The pattern shown gets repeated on further blocks.

# RAID Level 6

- **RAID Level 6**: P+Q Redundancy scheme;
  - similar to Level 5, but stores extra redundant information to guard against multiple disk failures.
- Better reliability than Level 5 at a higher cost
  - Becoming more important as storage sizes increase
- Setup



(d) RAID 6: P + Q redundancy

# Choice of RAID Level

- Level 1 provides much better write performance than level 5
  - Level 5 requires at least 2 block reads and 2 block writes to write a single block, whereas Level 1 only requires 2 block writes
  - Level 1 preferred for high update environments such as log disks
- Level 1 had higher storage cost than level 5
  - Disk drive capacities increasing rapidly (50% per year) whereas disk access times have decreased much less (x 3 in 10 years)
  - I/O requirements have increased greatly, e.g. for Web servers
  - When enough disks have been bought to satisfy required rate of I/O, they often have spare storage capacity
    - so there is often no extra monetary cost for Level 1!
- Level 5 is preferred for applications with low update rate,
  and large amounts of data
- Level 1 is preferred for all other applications

# Hardware Issues

- **Software RAID**:  RAID implementations done entirely in software, with no special hardware support
- **Hardware RAID**:  RAID implementations with special hardware
  - Use non-volatile RAM to record writes that are being executed
- Power failure during write can result in corrupted disk
  - Example, failure after writing one block but before writing the second in a mirrored system
  - Such corrupted data must be detected when power is restored
  - Recovery from corruption is similar to recovery from failed disk
  - NV-RAM helps to efficiently detected potentially corrupted blocks
    - Otherwise all blocks of disk must be read and compared with mirror/parity block

# Hardware Issues (Cont.)

- **Latent failures**: data successfully written earlier gets damaged
  - can result in data loss even if only one disk fails
- **Data scrubbing:**
  - continually scan for latent failures, and recover from copy/ parity
- **Hot swapping**: replacement of disk while system is running, without power down
  - Supported by some hardware RAID systems,
  - reduces time to recovery, and improves availability greatly

# Hardware Issues (Cont.)

- Many systems maintain spare disks which are kept online, and used as replacements for failed disks immediately on detection of failure
  - Reduces time to recovery greatly
- Many hardware RAID systems ensure that a single point of failure will not stop the functioning of the system by using
  - Redundant power supplies with battery backup
  - Multiple controllers and multiple interconnections to guard against controller/interconnection failures

# DISK-BLOCK ACCESS

# Storage Access

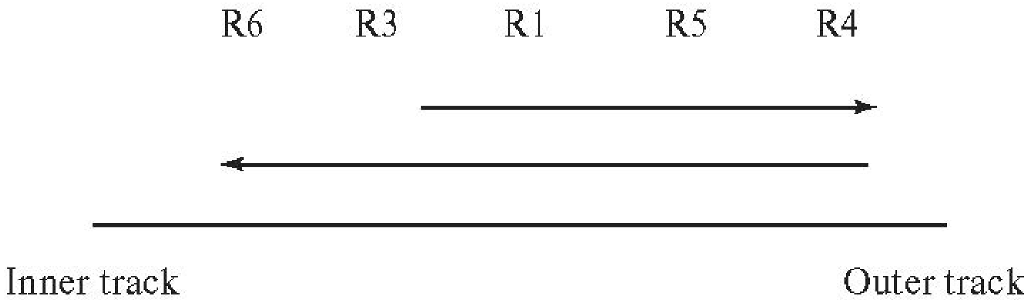
- Requests for disk I/O are generated by the database system, with the query processing sub-system responsible for most of the disk I/O .
- A database file is partitioned into fixed-length storage units called **blocks**.  Blocks are units of both storage allocation and data transfer.
- Database system 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.

# Storage Access (Cont.)

- **Buffering.** Blocks that are read from disk are stored temporarily in an in-memory buffer.
- **Read-ahead**.  When a disk block is accessed, consecutive blocks from the same track are read into an in-memory buffer even if there is no pending request for the blocks.
- **Disk-arm-scheduling** . Order pending accesses to disk tracks so that disk arm movement is minimized
  - **elevator algorithm**

# 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
  - sizes range from 512 bytes to several kilobytes
    - Smaller blocks: more transfers from disk
    - Larger blocks:  more space wasted due to partially filled blocks
    - Typical block sizes today range from 4 to 16 kilobytes
- **Disk-arm-scheduling** algorithms order pending accesses to tracks so that disk arm movement is minimized
  - **Elevator algorithm**:



# End of Module 10

Database System Concepts, 7th Ed.