

Disk Storage and Architectures, File Structures, and Hashing

CS 301

TAs

- Mappuri Siva Jagadesh
- Rachapudi Maruthi Sriram
- Naga Srihith Penjarla
- Kautuk Raj

“Goal of the chapter”

“Learn about the organization of databases in storage and the techniques for accessing them efficiently.”

1. Introduce the concept of computer storage hierarchies and their usage in DB systems.
2. Discuss different storage technologies and their characteristics, methods for physically organizing data on disks, and double buffering.
3. Describe buffer management and buffer replacement strategies.
4. Discuss formatting, storing and methods for organizing file records on disk, and various operations applied to the file records.
5. Introduce RAID and some modern developments in the storage architectures such as SANs, NAS, iSCSI.

Background


- The collection of data in a DB is done physically on some computer storage medium. The DBMS software then retrieves, updates and processes the data.
- Computer storage media form a storage hierarchy that include:
 1. Primary storage – This includes storage media that can be operated on directly by CPU. These usually provide fast access to data but it is of limited storage capacity. The contents of main memory are lost in case of power failure or a system crash.
 2. Secondary storage – Examples are magnetic disk, flash memory and SSD (solid-state-drive).
 3. Tertiary storage – Examples are CD-ROMs, DVDs and tapes that have larger storage capacity and are used as offline storage for archiving DBs.

Background

- Tertiary storage media cost less and provide slower access to data than do primary storage devices.
 - Data in secondary or tertiary storage cannot be processed directly by the CPU as it must be first copied into the primary storage and then processed by the CPU.

Memory hierarchies and storage devices

- In general, the highest-speed memory is the most expensive and is therefore available with the least capacity whereas the lowest-speed memory is offline tape storage, which is essentially available in indefinite storage capacity.
- Currently, data resides and is transported throughout a hierarchy of storage media.
- At the *primary storage level*, cache memory which is a static RAM (random access memory) is used by the CPU to speed up execution of program instructions.
- At next level, DRAM (dynamic RAM) also called main memory provides the main work area for the CPU for keeping program instructions and data.

- 
- At the *secondary* and *tertiary* storage level, the hierarchy includes magnetic disks, CD-ROM, DVD and tapes with storage capacity ranging from kilobytes (KB) to megabytes (MB) to gigabytes (GB) to terabytes (TB) to petabytes (1000 terabytes) and Exabyte (1000 petabyte or million TB).

- Flash memory – Between DRAM and magnetic disk storage, flash memory is popular which is non-volatile. They are high-density, high-performance memories with fast access speed. They are used in cameras, MP3/MP4 players, cell phones, PDAs, etc.
- USB (universal serial bus) flash drives or USB sticks have become the most portable medium for carrying data between personal computers as they have a flash memory storage device integrated with a USB interface.
- Optical drives – Example, CDs and DVDs. Optical Jukebox memories use an array of CD-ROM platters, which are loaded onto drives on demand.
- Magnetic tapes – They are used for archiving and backup storage of terabytes of data as tertiary storage devices.

- Tape jukeboxes – contain a bank of tapes that are catalogued and can be automatically loaded onto tape drives. They are popular tertiary storage used in NASA's EOS (Earth Observation Satellite) system to store archived DBs.
- Table below summarises the current state of the storage devices and systems showing the range of capacities, average access times, bandwidths and costs.

Type	Capacity*	Access		Commodity Prices (2014)**
		Time	Max Bandwidth	
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–1.2PB/sec	\$3K–\$1M+

Table: Types of storage with capacity, access time, transfer speed and cost.

Storage organisation of databases

- **Persistent data** – DBs typically store large amounts of data that must persist over long periods of time, they are referred to as persistent data.
- **Transient data** – data that persists for only a limited time during program execution are referred to as transient data.
- **Physical database design** – This involves choosing the particular data organisation techniques that best suit the given application requirements from among the options.
- **Primary file organisation** – It determines how the file records are physically placed on the disk and how the records can be accessed.
 - A heap file (or unordered file) places the records on disk in no particular order by appending new records at the end of the file.

Storage organisation of databases

- Sorted file (also known as sequential file) on the other hand keeps the records ordered by the value of a particular field (sort key).
- A hashed file uses a hash function applied to a particular field (called the hash key) to determine a record's placement on disk.
- **Secondary file organisation** – They allow efficient access to file records based on alternate fields than those that have been used for the primary file organisation.

Secondary (disk) storage devices

- Magnetic disks are used for storing large amounts of data. The device that holds the disks is called **hard disk drive (HDD)**.
- By magnetizing an area on a disk in certain ways, one can make that area represent a bit value of “0” or “1”.
- **Capacity** of a disk is the number of bytes it can store. Example, modern hard disk hold a few TBs.
- Preferred secondary storage devices facilitate high storage capacity with low cost.
- All disks are made of magnetic material shaped as a thin circular disk (see next slide) protected by a plastic or acrylic cover.
- A disk is **single-sided** if it stores information on one of its surfaces only and **double-sided** if both surfaces are used.

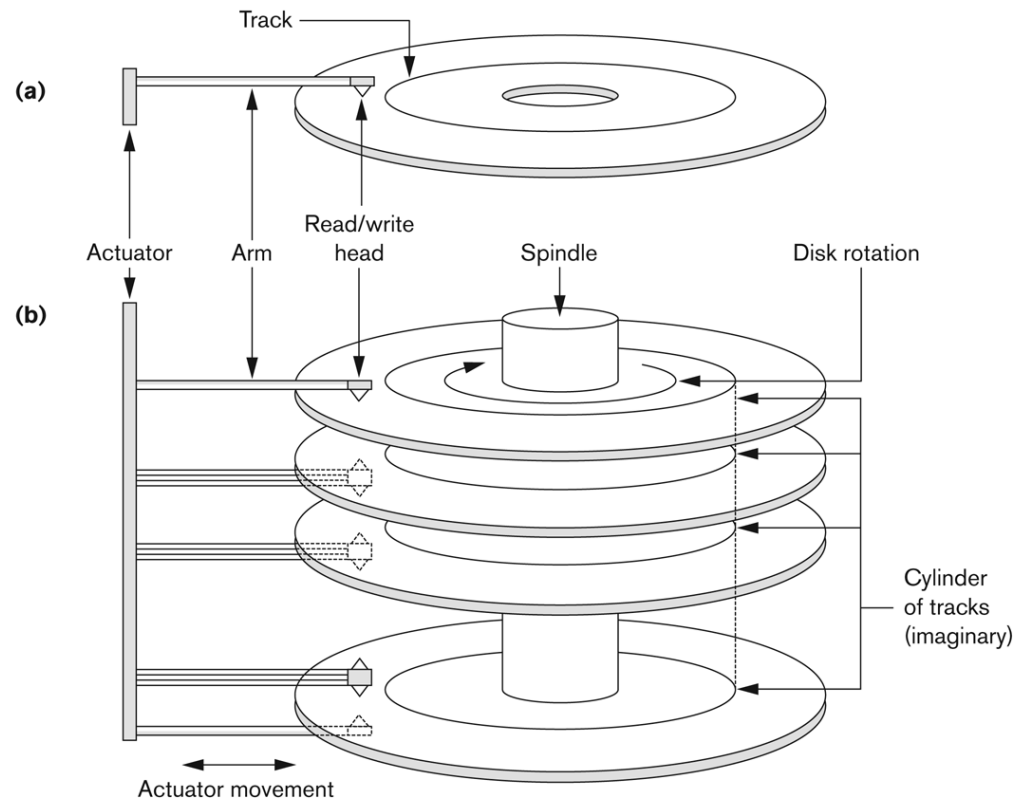


Figure: (a) A single-sided disk with read/write hardware. (b) A disk pack with read/write hardware.

A cylinder is **any set of all of tracks of equal diameter in a** hard disk drive (HDD). It can be visualized as a single, imaginary, circle that cuts through all of the platters (and both sides of each platter) in the drive. The magnetic media on each side of each platter is divided into a series of **tracks**.

Secondary (disk) storage devices

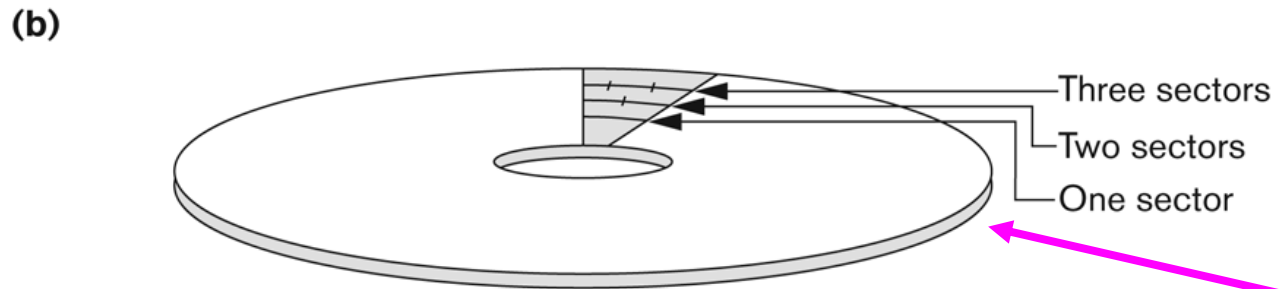
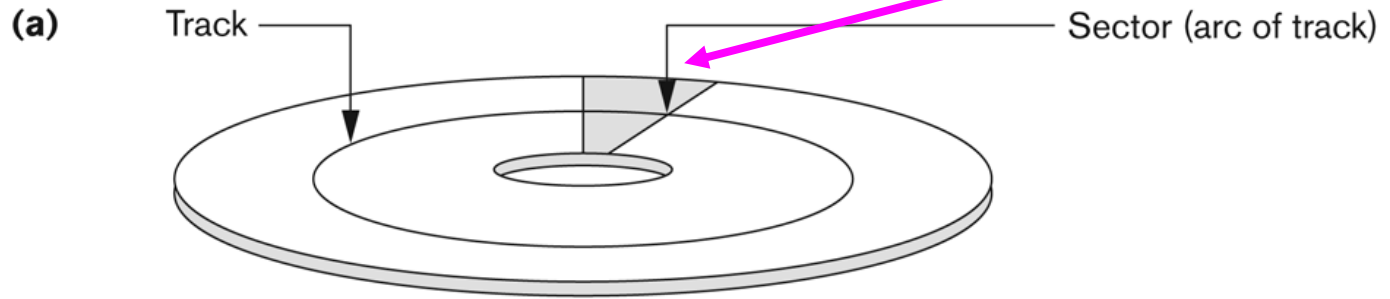
- To increase storage capacity, disks are assembled into **disk pack** that may contain several magnetic disks and therefore many surfaces.
- Information is stored on a disk surface in concentric circles of distinct diameter and each circle is called a **track**. Track capacities vary typically from 4 to 150 Kbytes or more.
- In disk packs, tracks with the same diameter are called a cylinder (see figure in previous slide) where data stored on “one” cylinder can be retrieved much faster than if data were distributed among “different” cylinders. (Typically, number of tracks on a disk ranges from a few thousand to 152000 on the disk. See table in next slide)
 - Because a track usually contains a large amount of information, it is further divided into smaller blocks or sectors which is hard coded on the disk surface.

Typical disk parameters

Specifications	1200GB
SED Model Number	ST1200MM0017
SED FIPS 140-2 Model Number	ST1200MM0027
Model Name	Enterprise Performance 10K HDD v7
Interface	6Gb/s SAS
Capacity	
Formatted 512 Bytes/Sector (GB)	1200
External Transfer Rate (MB/s)	600
Performance	
Spindle Speed (RPM)	10K
Average Latency (ms)	2.9
Sustained Transfer Rate Outer to Inner Diameter (MB/s)	204 to 125
Cache, Multisegmented (MB)	64
Configuration/Reliability	
Disks	4
Heads	8
Nonrecoverable Read Errors per Bits Read	1 per 10E16
Annualized Failure Rate (AFR)	0.44%
Physical	
Height (in/mm, max)	0.591/15.00
Width (in/mm, max)	2.760/70.10
Depth (in/mm, max)	3.955/100.45
Weight (lb/kg)	0.450/0.204

Disk storage devices

- One type of “sector organization” calls a portion of a track that subtends a fixed angle at the center as a sector.



- Several other “sector organizations” are possible, one of which is to have the sectors subtend “smaller angles” at the center as one moves away, thus maintaining a uniform density of recordings.

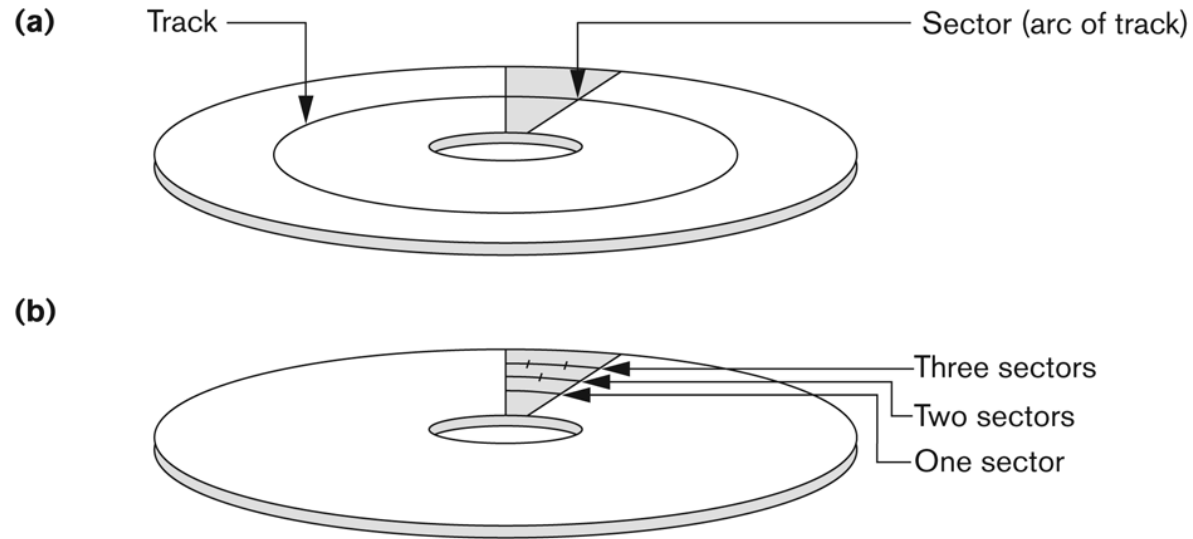


Figure: Different sector organisations on disk. (a) Sectors subtending a fixed angle. (b) Sectors maintaining a uniform recording density.

- ZBR (zone bit recording) is an algorithm that allows a range of cylinders to have the same number of sectors per arc.
- Example: cylinders 0-99 may have one sector per track, 100-199 may have two tracks, etc.

- A track is divided into **blocks**.
 - The block size B is fixed for each system.
 - Typical block sizes range from $B=512$ bytes to $B=4096$ bytes.
 - Whole blocks are transferred between disk and main memory for processing.
 - The division of a track into equal-sized **disk blocks** (or **pages**) is set by the OS during disk **formatting** which is fixed, and cannot be changed dynamically.
- Blocks are separated by fixed size **interblock gaps** which include specially coded control information that is used to determine which block on the track follows each interblock gap.

- A **read-write head** which is part of a disk drive reads or writes a block.
- So, a disk or disk pack is mounted in the disk drive that has a motor that rotates the disk and the read/write head is attached to a **mechanical arm**.
- For several **disk packs** with multiple surfaces, all arms are connected to an **actuator** attached to another electrical motor, which moves the read/write head and positions them precisely over the cylinder of tracks specified in a block address.

- So, **we understood** that a physical disk block (hardware) address consists of:
 - a cylinder number (imaginary collection of tracks of same radius from all recorded surfaces),
 - the track number or surface number (within the cylinder), and
 - block number (within track).
- Reading or writing a disk block is time consuming because of the **seek time s** and **rotational delay (latency) rd**.
- “Double buffering” (we will learn about this term in the later part of the slides) can be used to speed up the transfer of contiguous disk blocks.
- Note: Disk drives for HDD rotate the disk pack at a speed ranging from 5400 to 15000 rpm.

- There are also some disk units that have fixed read/write heads, with many heads as the number of tracks. These are called fixed-head disks, whereas disk units with an actuator are called movable-head disks.
- For fixed-head disks, a track or cylinder is selected by electronically switching to the appropriate read/write head rather than mechanical movement. So they are much faster.

How to make “disk data access” more efficient?

- Some of the commonly used techniques to access data more efficiently on HDDs are:
 - **Buffering * of data:** In order to deal with the incompatibility of speeds between a CPU and the HDD (which is slower), *buffering* of data is done in memory so that the new data can be held in a buffer while old data is processed.

* Note: Buffer refers to a part of the main memory that is available to receive blocks or pages of data from disk.

- **Proper organisation of data on disk:** Related data are stored on contiguous blocks when multiple cylinders are needed by a relation. This avoids unnecessary movement of read/write arm and seek time.
- **Reading data ahead of request:** To minimise seek times, whenever a block is read into the buffer, blocks from the rest of the track can also be read even though they may not have been requested. This works well for applications that are likely to need consecutive blocks.
- **Proper scheduling of I/O requests:** When necessary to read several blocks from disks, total access time can be minimised by scheduling them to pick up blocks in one direction only. A popular “elevator algorithm” mimics the behaviour of an elevator that schedules requests on multiple floors in a sequence. Therefore, the arm can service requests along its outward and inward movements without disruption.

- **Use of log disks to temporarily hold writes:** A single disk may be assigned to just one function called logging of writes. All blocks to be written can go to that disk sequentially and avoid any seek time. This is much faster than doing write to a file at random locations.
 - The log disk can order these writes in (cylinder, track) ordering to minimise arm movement when writing.
 - Although the idea of a “log disk” can improve write performance, it is not feasible for real time applications because the performance is slower.
- **Use of SSDs or flash memory:** This is useful in applications where updates occur with high frequency. Here, a non-volatile SSD buffer is used which can be a flash memory or battery-operated DRAM.

SSD (SolidState device storage)

- It is also known as flash storage because it is based on the “flash memory technology”.
- They are used as an intermediate layer in the form of HDDs between main memory and secondary memory. Since they resemble disks in terms of the ability to store data in secondary storage without continuous power supply, they are called solid-state disks or solid-state drives (SSDs).
- SSD mainly consists of a controller and a set of interconnected flash memory cards. Since there are no moving parts, the unit is more rugged, runs silently, is faster in terms of access time and provides higher transfer rates than HDD. The data is less fragmented. Still they are costly in commercial market.
- Finally, in addition to flash memory, DRAM based SSD offer faster access times of around 10 microseconds as opposed to 100 microseconds for flash but they need an internal battery or an adapter to supply power and are costlier.

Magnetic tape storage devices

- They are sequential access devices to access n th block on tape after scanning the preceding $n-1$ blocks.
- Data is stored on reels of high-capacity magnetic tape. A tape drive is required to read the data from or write the data to a tape reel.
- The main characteristic of a tape is that data are accessed in sequential order. Therefore, tape access can be slow and are not used to store online data.
- They are mostly used for backing up the DB "to keep copies of disk files in case the data is lost due to a disk crash because of mechanical malfunction". Therefore, disk files are copied periodically to tape.
- Tapes are also used for storing, archival or recordkeeping Big data.

Buffering of blocks

- When several blocks need to be transferred from disk to main memory and all the block addresses are known, several buffers can be reserved in main memory to speed up the transfer (This seems logical).
 - While one buffer is being read or written, the CPU can process data in the other buffer because an independent disk I/O processor can transfer a data block between memory and disk independent of and in parallel to CPU processing.
 - Therefore, two processes (say “A” and “B”) can run concurrently in an interleaved fashion whereas other two processes (say “C” and “D”) are running concurrently in parallel (see figure in next slide).
- When a single CPU controls multiple processes, parallel execution is not possible. However, the processes can run concurrently in an interleaved way.

Buffering of blocks

- Buffering is useful when processes can run concurrently in parallel either because a separate disk I/O processor is available or because multiple CPU processors exist.

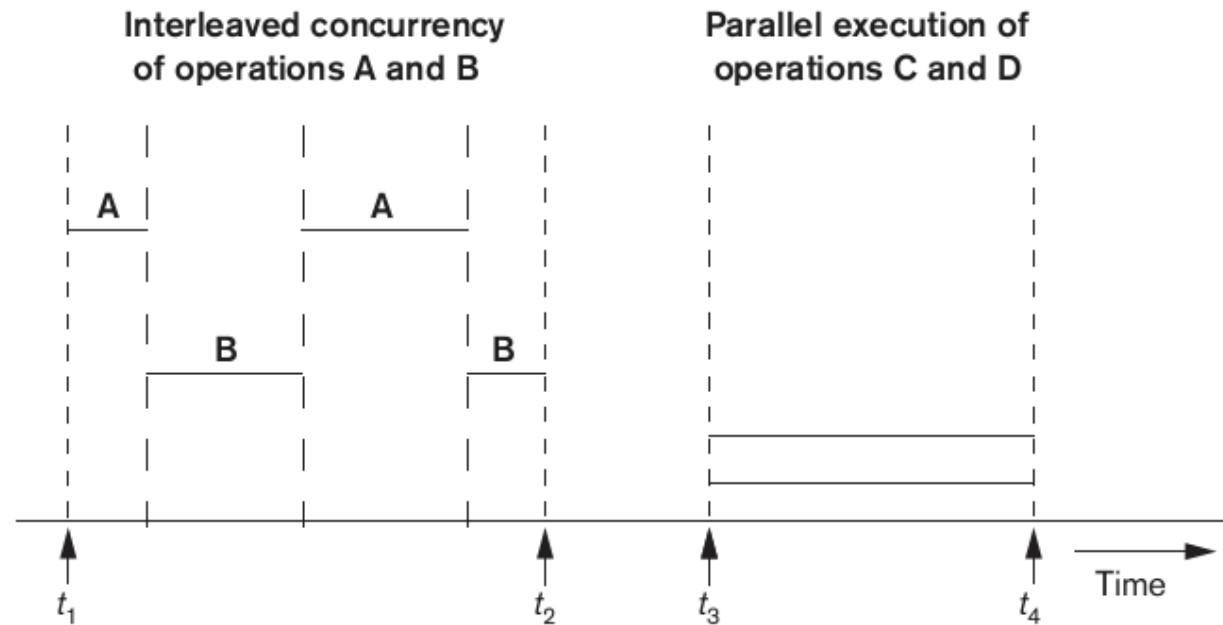


Figure: Interleaved concurrency versus parallel execution.

Buffering of blocks

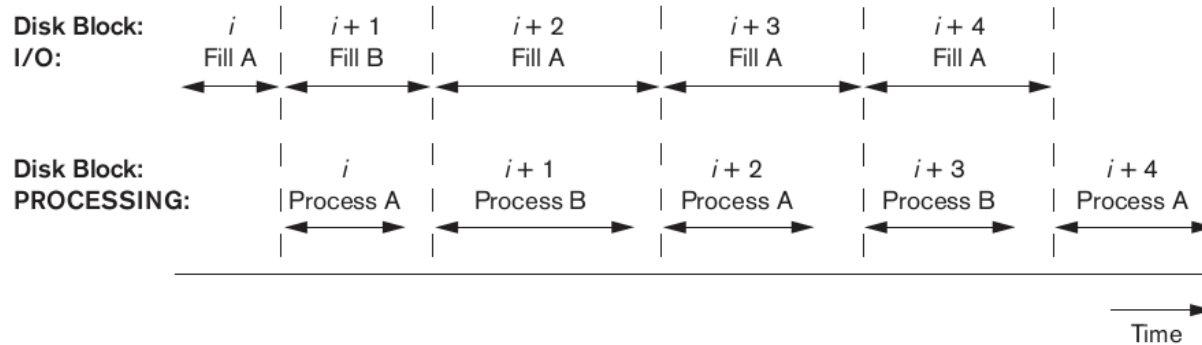


Figure: Use of two buffers, A and B for reading from disk.

- Above figure shows that reading and processing can proceed in parallel when the time required to process a disk block in memory is less than the time required to read the next block and fill a buffer.
- The CPU can start processing a block and at the same time, the disk I/O processor can be reading and transferring the next block into a different buffer – this is called “double buffering” that can be used to read a continuous stream of blocks from disk to memory.

Buffering of blocks

- Advantages of double buffering – It permits continuous reading/writing of data on consecutive disk blocks, which eliminate the seek time and rotational delay for all but the first block transfer. Further, data is kept ready for processing, reducing the waiting time.
- Buffer management: Buffer manager is a software that responds to request for data and decides what buffer to use and what pages to replace in the buffer to accommodate the newly requested blocks.
- Buffer manager:
 1. Maximises the probability that the requested page is found in main memory.
 2. In case of reading a new disk block from disk, a buffer manager finds a page to replace.

- A buffer manager keeps two types of information about each page:
 1. **A pin count**: the number of times that page has been requested or the number of current users of that page. If this count is zero, the page is considered unpinned.
 2. **A dirty bit** which is set to 0 for all pages and is set to 1 whenever that page is updated.
- When a certain page is requested, the buffer manager checks if the requested page is already in a buffer and increments its pin count and releases the page. If the page is not in the buffer pool, then the buffer manager does the following:
 - a. It chooses a page for replacement, using the replacement policy, and increments its pin-count.
 - b. If the dirty bit of the replacement page is on, the buffer manager writes that page to disk by replacing its old copy on disk. If the dirty bit is not on, this page is not modified and the buffer manager is not required to write it back to disk.
 - c. It reads the requested page into the space just freed up.
 - d. The main memory address of the new page is passed to the requesting application.

Buffering of blocks

Buffer replacement strategies: The popular replacement strategies are:

- Least recently used (LRU) – Throw out the page that has not been used for the longest time. This is done by maintaining a table.
- Clock policy – This is a round robin variant of the LRU policy where the buffers are arranged in a circle similar to a “clock” with a flag of 0 or 1. Buffers with 0 may be used for replacement. When a block is read into a buffer, the flag is set to 1. When the buffer manager needs a buffer for a new block, it rotates the hand (like in a clock) until it finds a buffer with 0 and uses that to read and place the new block.
- First-in-first-out (FIFO) – When a buffer is required, the one that has been occupied the longest by a page is used for replacement.

Placing file records on disk

- Record and record types – The data type of a field include numeric (integer, long integer, or floating point), string of characters, Boolean, date and time.
- Files, fixed-length and variable-length records – 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** or else it is a **variable-length record**. A file may have variable length records because
 - One or more of the fields in a file record are of varying size. Example: name of a person.
 - One or more of the fields may have multiple values for individual records.
 - One or more of the fields are optional.
 - The file contains records of different record types and varying size.
- Note: For variable-length fields, each record has a value for each field, but it is not known as to how long is the field value.

Placing file records on disk

- To determine the bytes within a particular record that represent each field, we use a **separator** character (such as ? or % or \$) which do not appear in any field value.

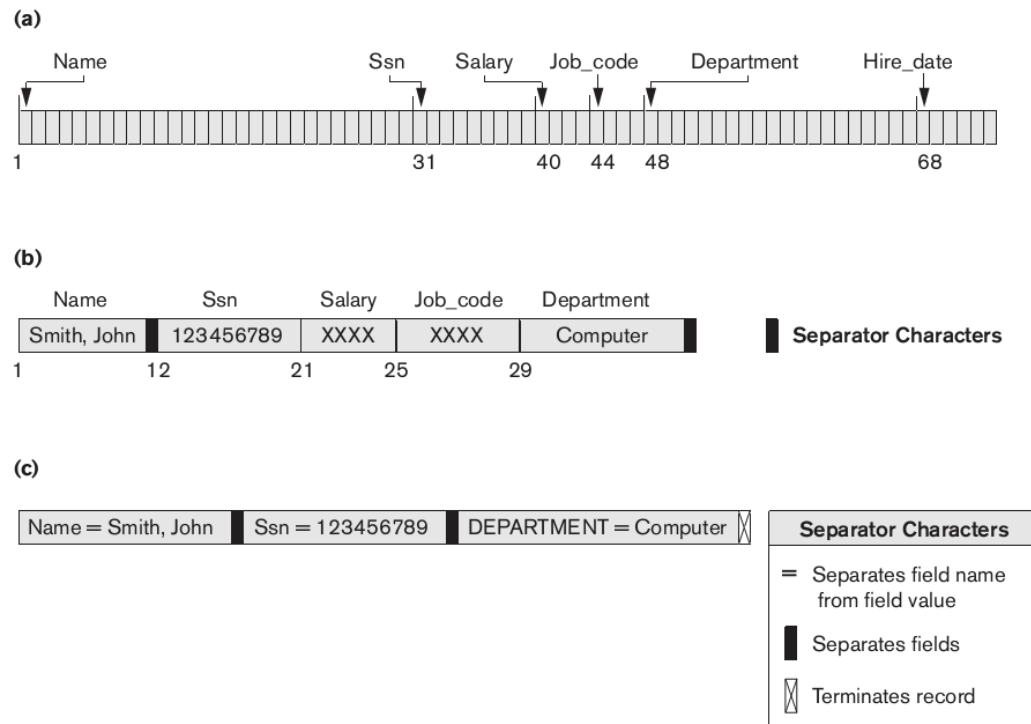


Figure: Two record storage formats. (a) A fixed-length record with six fields and size of 71 bytes. (b) A record with two variable-length fields and three fixed length fields. (c) A variable-field record with three types of separator characters.

Placing file records on disk

- The record of a file must be allocated to disk blocks. When the block size is larger than the record size, each block will contain numerous records, although some files may have unusually large records that cannot fit in one block.
- To utilise the unused space, part of a record is stored on one block and the rest on another. A “pointer” at the end of the first block points to the block containing the remainder of the record in case it is not the next consecutive block on disk – this is called “spanned” because records can span more than one block (see figure in next slide).
- If records are not allowed to cross block boundaries – it is called “unspanned” records.
- For variable length records, either a spanned or an unspanned organisation can be used. If the record is large, it is reasonable to use spanning to reduce the lost space in each block.

Placing file records on disk

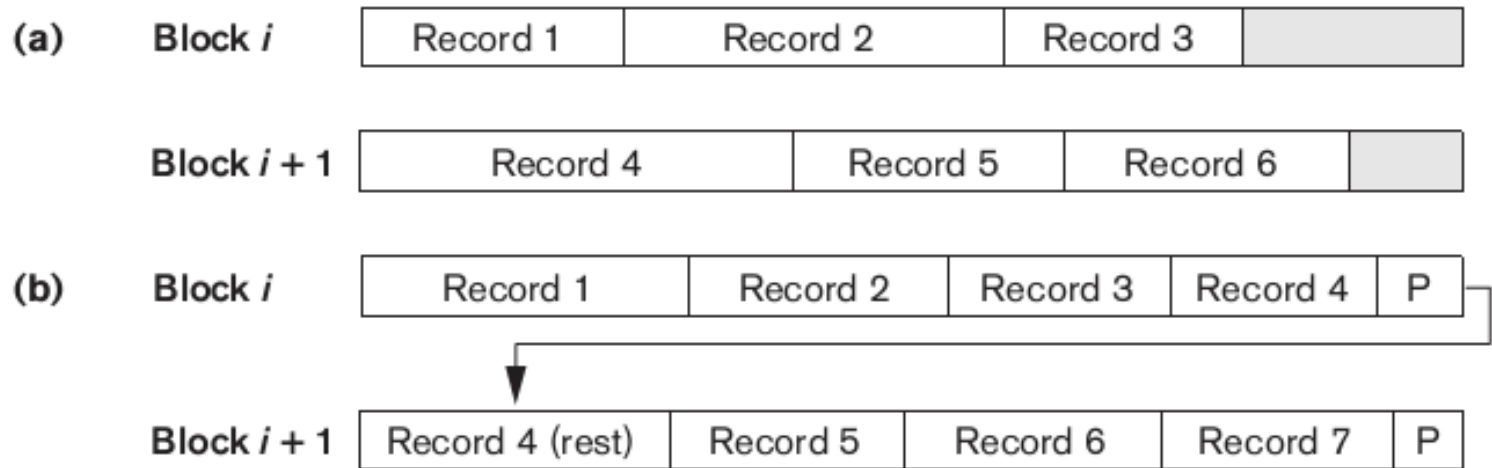


Figure: Types of record organisation – (a) unspanned and spanned.

Placing file blocks on disk

- Allocating file blocks on disk - There are several standard techniques for allocating the blocks of a file on disk.
 1. Contiguous allocation – the file blocks are allocated to consecutive disk blocks which makes reading the whole file very fast using double buffering but it makes expanding the file difficult.
 2. 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.
 3. Clusters – A combination of contiguous (1. above) and linked allocation (2. above) is a cluster and the clusters are linked.
 4. Indexed allocation – uses one or more index blocks that contain “pointers” to the actual file blocks.

File headers

- A **file header** or *file descriptor* contains information about a file that is required by the system programs that access the file records.
- The header includes information to determine the disk address of the file blocks as well as to record format description that may include field lengths, field type codes, separator characters, and record type codes for variable length records.

Operations on files

- Operations on files are either **retrieval** that does not change any data in file but only locate certain records, or **update** operation that can change the file by insertion or deletion of records or by modification of field values.
- A set of **record-at-a-time** file operations for locating and accessing file records are as follows:
 - Open – Prepare the file for reading or writing.
 - Reset – Sets the file pointer of an open file to the beginning of the file.
 - Find (or locate) – Search for the first record that satisfies a search condition.
 - Read (or Get) – Copies the current record from the buffer to a program variable in the user program.
 - FindNext – Searches for the next record in the file that satisfies the research condition.
 - Delete – Delete the current record and update the file.

Operations on files

- Modify – Modifies some field values for the current record and updates the file.
 - Insert – Insert a new record in the file by locating the block where the record is to be inserted.
 - Close – Completes the file access by releasing the buffers.
- A set of **set-at-a-time** file operations are as follows:
 - FindAll – Locates all the records in the file that satisfy a search condition.
 - Find (or Locate) n – Search for the first record that satisfies a search condition and then locate $n-1$ records satisfying the same condition.
 - FindOrdered – Retrieves all the records in the file in some order.
 - Reorganize – Start the organisation process such as reordering based on sorting on a specified field.

Files of unordered records (heap files)

- In the basic type of file organisation, records are placed in the file in the order in which they are inserted, so new records are inserted at the end of the file. Such an organisation is called a **heap or pile file**.
- Inserting a new record is efficient as “the last disk block of the file” is copied into a buffer, the new record is added, and the block is then rewritten back to disk. The address of the last file block is kept in the file header.
- To delete a record:
 - First a block is searched and copied into a buffer and is then deleted from the buffer and then rewritten back to the disk. Disadvantage – deleting a large number of records results in wasted storage space.
 - Use of deletion marker – a record is deleted by setting the deletion marker to a certain value. A different value for the marker indicates a valid (not deleted) record.
 - Both methods require periodic reorganisation of the file to reclaim the unused space of deleted records.

Files of ordered records (sorted files)

- The records of a file can be physically ordered on disk based on the value of one of their fields – called the ordering field leading to an ordered or sequential file.
- If the ordering field is a key field, then the field is called the ordering key for the file.
- Example: Consider the EMPLOYEE records with an ordered file with “Name” as the ordering key field (assuming distinct names) [see next slide].
- Advantages: (i) Reading the records in order of the ordering key values are efficient because no sorting is required. (ii) Finding the next record from the current one requires no additional block access. (iii) Using a search condition based on the value of an ordering key field results in faster access.

	Name	Ssn	Birth_date	Job	Salary	Sex
Block 1	Aaron, Ed					
	Abbott, Diane					
	⋮					
	Acosta, Marc					
Block 2	Adams, John					
	Adams, Robin					
	⋮					
	Akers, Jan					
Block 3	Alexander, Ed					
	Alfred, Bob					
	⋮					
	Allen, Sam					
Block 4	Allen, Troy					
	Anders, Keith					
	⋮					
	Anderson, Rob					
Block 5	Anderson, Zach					
	Angeli, Joe					
	⋮					
	Archer, Sue					
Block 6	Arnold, Mack					
	Arnold, Steven					
	⋮					
	Atkins, Timothy					
⋮						
Block $n-1$	Wong, James					
	Wood, Donald					
	⋮					
	Woods, Manny					
Block n	Wright, Pam					
	Wyatt, Charles					
	⋮					
	Zimmer, Byron					

Figure: Blocks of an ordered (sequential) file of EMPLOYEE records with Name as the ordering key field.

Hashing techniques

- Hashing is another type of file organisation providing fast access to records under search conditions and this organisation is called a hash file.
 - Search condition is an equality condition on a single field, called the hash field which is also a *key field* of the file, in which case it is called a hash key.
-
- There are two types of hashing – internal and external.
 - **Internal hashing** – uses a hash table through the use of an array of records.
 - Suppose that the array index range is from 0 to $M-1$, then there are M slots whose addresses correspond to the array indexes.
 - We choose a hash function that transforms the hash field value into an integer between 0 and $M-1$.

	Name	Ssn	Job	Salary
0				
1				
2				
3				
	⋮			
$M - 2$				
$M - 1$				

Figure: Internal hashing data structure. Array of M positions for use in internal hashing.

- Example of a hash function is $h(K) = K \bmod M$ function which returns the remainder of an integer hash field value K after division by M and is then used for the record address.
- Concept of collision in internal hashing – This occurs when the hash field value of a record being inserted hashes to an address that already contains a different record.

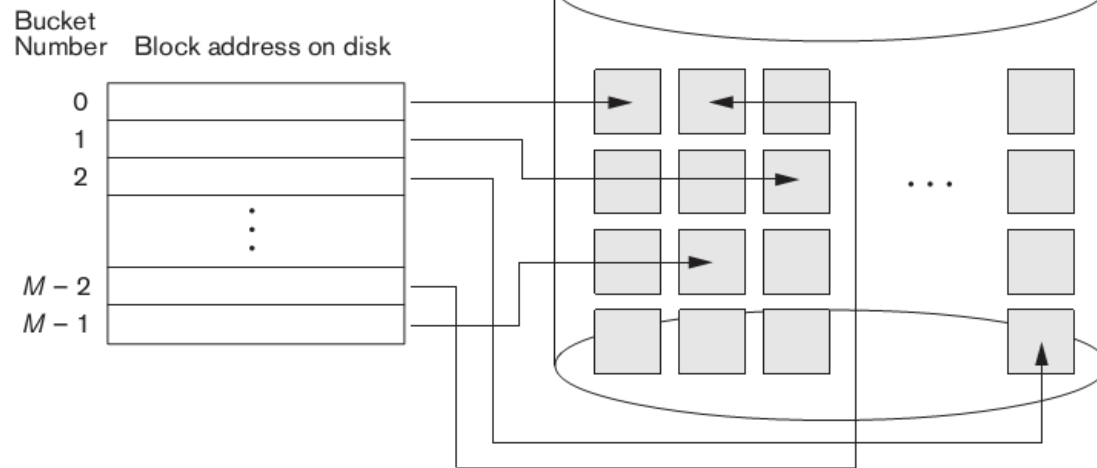


Figure: Matching bucket numbers to disk block addresses.

- **External hashing** – hashing for disk files is called external hashing. To suit the disk characteristics, the target address space is made of buckets, each of which holds multiple records (see figure above).
- A bucket is either one disk block or a cluster of contiguous disk blocks.
- Hashing function maps a key into a relative bucket. A table is maintained in the file header which converts the bucket number into the corresponding disk block address.

Hashing for dynamic file expansion

- A major drawback of the static hashing scheme is that the hash address space is fixed. Therefore it is difficult to expand or shrink the file dynamically. There are three techniques to overcome this problem.
 1. Extendible hashing – It stores an access structure in addition to the file and is similar to indexing.
 2. Dynamic hashing – It uses an access structure based on binary tree data structures.
 3. Linear hashing – It does not require additional access structures.

RAID technology

- The main goal of RAID is to *even out* the widely different rates of performance improvement of disks against those in memory and microprocessors.
- Concept of data striping – RAID utilises parallelism to improve disk performance.
 - It distributes data transparently over multiple disks to make them appear as a single large, fast disk.
- RAID improves reliability and performance.

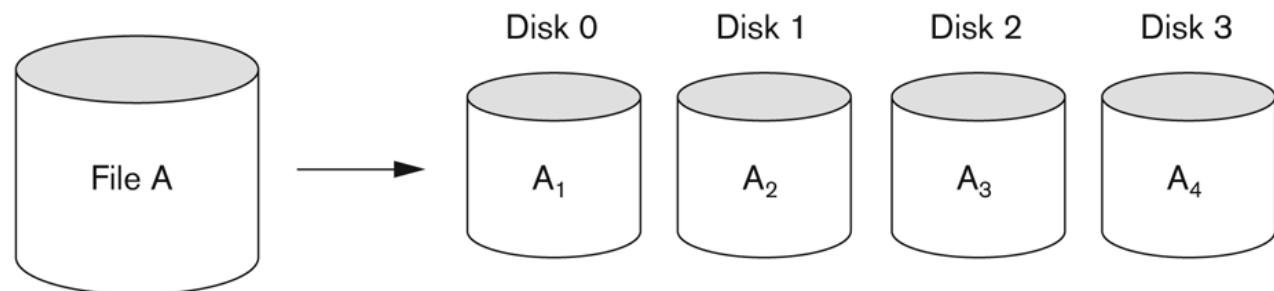
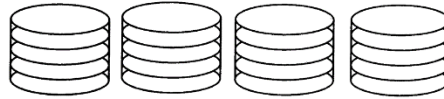


Figure: Data striping. File A is striped across four disks

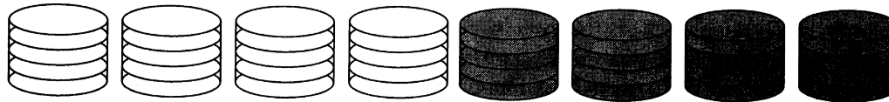
- RAID has levels from 0 to 6 where each level has a specific function.
 - Raid level 0 has no redundant data and hence has the best write performance at the risk of data loss.
 - Raid level 1 uses mirrored disks.
 - Raid level 2 uses memory-style redundancy by using Hamming codes, which contain parity bits for distinct overlapping subsets of components. Level 2 includes both error detection and correction.
 - Raid level 3 uses a single parity disk relying on the disk controller to figure out which disk has failed.
 - Raid levels 4 and 5 use block-level data striping, with level 5 distributing data and parity information across all disks.
 - Raid level 6 applies the so-called $P + Q$ redundancy scheme using Reed-Soloman codes to protect against up to two disk failures by using just two redundant disks.

- Different raid organizations are being used under different situations:
 - Raid level 1 (mirrored disks) is the easiest to rebuild a disk from other disks.
 - It is used for critical applications like logs.
 - Raid level 2 includes both error detection and correction.
 - Raid level 5 (block-level data striping) are preferred for Large volume storage, with Raid level 3 (single parity disks relying on the disk controller to figure out which disk has failed) giving higher transfer rates.

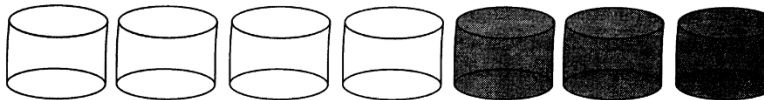
Use of RAID technology



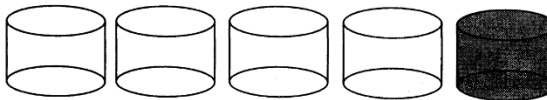
Non-Redundant (RAID Level 0)



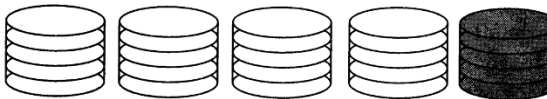
Mirrored (RAID Level 1)



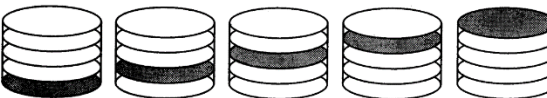
Memory-Style ECC (RAID Level 2)



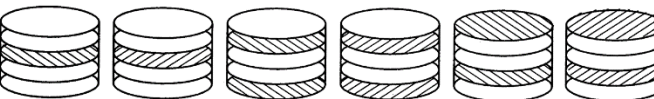
Bit-Interleaved Parity (RAID Level 3)




Block-Interleaved Parity (RAID Level 4)



Block-Interleaved Distribution-Parity (RAID Level 5)



P+Q Redundancy (RAID Level 6)

- 
- Most popular uses of the RAID technology currently are:
 - Level 0 (with striping), Level 1 (with mirroring) and Level 5 with an extra drive for parity.
 - Design decisions for RAID include:
 - level of RAID
 - number of disks
 - choice of parity schemes, and
 - grouping of disks for block-level striping.

New storage architectures

- Storage area networks (SAN) – Here, online storage peripherals are configured as nodes on a high speed network and can be attached or detached from servers in a flexible manner.
- Network-attached storage (NAS) – They are high performance storage devices or servers that do not provide any of the common server services but simply allow the addition of storage for file sharing. They have vast amount of hard-disk storage to be added to the network without shutting them down for maintenance and upgrades. They can reside anywhere on a local area network (LAN) and may be combined in different configurations.
- iSCSI (internet small computer system interface) – It is a block-storage protocol like SAN that allows clients to send commands on remote channels. It does not require Fibre channel and can run over longer distances. It can transfer data over LAN, WAN (wide area networks) or the internet.

Summary

- Discussed characteristics of memory hierarchies.
- Secondary storage devices – magnetic disks.
- Disks, blocks, disk blocks (pages), SSD.
- Buffering of blocks, buffer management, buffer replacement strategies – LRU, clock policy, FIFO.
- Placing file records on disk.
- Operations on files.
- Files of unordered and ordered records.
- Hashing – internal and external.
- RAID, storage area networks (SAN), network-attached storage (NAS) and iSCSI.



Practice questions (check yourself)

1. Define primary, secondary and tertiary storage with examples.
2. What is a flash memory and tape jukebox?
3. Explain structure of a hard disk drive (HDD).
4. What is a solid state device storage (SSD)?
5. Explain double buffering and the buffer replacement strategies.
6. With reference to operations on files, discuss the terms - record-at-a-time and set-at-a-time.
7. Outline the differences between SAN and NAS.