# Core CS Fundamentals Placement Guide: OS, DBMS & OOPs

Author: Placements Simplified Editorial Team

## Class Introduction: Foundations of Software Engineering

Dear student, many candidates write excellent algorithms but fail technical interviews because they cannot explain how systems manage memory, execute queries, or enforce encapsulation rules. This guide is built to teach you these concepts step-by-step. Each of the following pages covers a high-frequency topic asked in interviews at companies like Google, Microsoft, Adobe, and top startup engineering teams. Read through each lesson carefully.

# Operating Systems (OS) Q&A Lecture Series - Session 1

### ■ What is a process and process table?

A process is an instance of a program in execution. For example, a Web Browser is a process, and a shell (or command prompt) is a process. The operating system is responsible for managing all the processes that are running on a computer and allocates each process a certain amount of time to use the processor. In addition, the operating system also allocates various other resources that processes will need, such as computer memory or disks. To keep track of the state of all the processes, the operating system maintains a table known as the process table. Inside this table, every process is listed along with the resources the process is using and the current state of the process.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the different states of the process?

Processes can be in one of three states: running, ready, or waiting. The running state means that the process has all the resources it needs for execution and it has been given permission by the operating system to use the processor. Only one process can be in the running state at any given time. The remaining processes are either in a waiting state (i.e., waiting for some external event to occur such as user input or disk access) or a ready state (i.e., waiting for permission to use the processor). In a real operating system, the waiting and ready states are implemented as queues that hold the processes in these states.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 2

### ■ What is a Thread?

A thread is a single sequence stream within a process. Because threads have some of the properties of processes, they are sometimes called lightweight processes. Threads are a popular way to improve the application through parallelism. For example, in a browser, multiple tabs can be different threads. MS Word uses multiple threads, one thread to format the text, another thread to process inputs, etc.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the differences between process and thread?

Aspects comparison between Process and Thread:

• **Execution Unit:** Process is an independent program under execution managed by the OS. Thread is the smallest unit of CPU scheduling within a process.
• **Memory Usage:** Process has its own address space (code, data, stack, heap). Thread shares code, data, and heap of the parent process; has its own stack, registers, and program counter.
• **Communication:** Processes use Inter-Process Communication (IPC) which is slower. Threads communicate through shared memory which is much faster.
• **Context Switching:** Process context switching is heavy (requires saving/restoring complete process state). Thread context switching is lightweight (only thread-specific registers and stack are switched).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 3

### ■ What are the benefits of multithreaded programming?

Multithreaded programming makes the system more responsive and enables resource sharing. It leads to the use of multiprocess architecture. It is more economical and preferred for concurrent applications.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is Thrashing?

Thrashing is a situation when the performance of a computer degrades or collapses. Thrashing occurs when a system spends more time processing page faults than executing transactions. While processing page faults is necessary in order to appreciate the benefits of virtual memory, thrashing has a negative effect on the system. As the page fault rate increases, more transactions need processing from the paging device. The queue at the paging device increases, resulting in increased service time for a page fault.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 4

### ■ What is Buffer?

A buffer is a memory area that stores data being transferred between two devices or between a device and an application. Buffering helps bridge speed gaps and format differences between hardware units.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is virtual memory?

Virtual memory creates an illusion that each user has one or more contiguous address spaces, each beginning at address zero. The sizes of such virtual address spaces are generally very high. The idea of virtual memory is to use disk space to extend the RAM. Running processes don't need to care whether the memory is from RAM or disk. The illusion of such a large amount of memory is created by subdividing the virtual memory into smaller pieces, which can be loaded into physical memory whenever they are needed by a process.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 5

### ■ Explain the main purpose of an operating system?

An operating system acts as an intermediary between the user of a computer and computer hardware. The purpose of an operating system is to provide an environment in which a user can execute programs conveniently and efficiently. An operating system is a software that manages computer hardware. The hardware must provide appropriate mechanisms to ensure the correct operation of the computer system and to prevent user programs from interfering with the proper operation of the system.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is demand paging and how does it work?

The process of loading the page into memory on demand (whenever a page fault occurs) is known as demand paging.
Working of Demand paging:
1. Process: Only the required memory pages of a process are loaded into RAM when needed, not the entire process.
2. Page Table: Keeps track of which pages are in memory and which are on disk.
3. Page Fault: Occurs when a referenced page is not in RAM, the OS loads it from secondary storage.
4. Loading: The missing page is brought into an empty frame in memory, and the page table is updated.
5. Execution Resumes: After the page is loaded, the process continues execution from where it was interrupted.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 6

### ■ What is a kernel?

A kernel is the central component of an operating system that manages the operations of computers and hardware. It basically manages operations of memory and CPU time. It is a core component of an operating system. Kernel acts as a bridge between applications and data processing performed at the hardware level using inter-process communication and system calls.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the different CPU scheduling algorithms?

1. First-Come, First-Served (FCFS) Scheduling
2. Shortest-Job-Next (SJN) / Shortest Job First (SJF) Scheduling
3. Priority Scheduling
4. Shortest Remaining Time (SRT) Scheduling
5. Round Robin (RR) Scheduling
6. Multiple-Level Queues Scheduling

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 7

### ■ Describe the objective of multi-programming?

Multi-programming increases CPU utilization by organizing jobs (code and data) so that the CPU always has one to execute. The main objective of multi-programming is to keep multiple jobs in the main memory. If one job gets occupied with IO, the CPU can be assigned to other jobs.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What problem do we face in a computer system without an OS?

Without an Operating System, users face several major issues: Poor resource management, Lack of User Interface (everything must be coded in binary/machine level), No File System (no standard storage directories), No Networking capabilities, and no standard Error handling interfaces.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 8

### ■ Give some benefits of multithreaded programming?

A thread is also known as a lightweight process. The idea is to achieve parallelism by dividing a process into multiple threads. Threads within the same process run in shared memory space, resulting in minimal communication overhead and faster execution.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Briefly explain FCFS scheduling?

FCFS stands for First Come First Served. In the FCFS scheduling algorithm, the job that arrived first in the ready queue is allocated to the CPU, and then the job that came second, and so on. FCFS is a non-preemptive scheduling algorithm as a process holds the CPU until it either terminates or performs I/O. Thus, if a longer job has been assigned to the CPU, then many shorter jobs after it will have to wait.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 9

## ■ What is the RR scheduling algorithm?

A round-robin scheduling algorithm is used to schedule the process fairly for each job in a time slot or quantum. Features:

• Round-robin is cyclic in nature, so starvation doesn't occur.
• Round-robin is a variant of first-come, first-served scheduling.
• No priority or special importance is given to any process or task.
• RR scheduling is also known as Time-slicing scheduling.

## ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

## ■ Enumerate the different RAID levels?

A redundant array of independent disks (RAID) is a set of several physical disk drives that the operating system sees as a single logical unit. It plays a significant role in narrowing the gap between fast processors and slow disk drives. RAID levels include Level-0 (striping), Level-1 (mirroring), Level-2, Level-3, Level-4, Level-5 (distributed parity), and Level-6.

## ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 10

### ■ What is Banker's algorithm?

The banker's algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for the predetermined maximum possible amounts of all resources, then makes an 's-state' check to test for possible activities, before deciding whether allocation should be allowed to continue.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ How does dynamic loading aid in better memory space utilization?

With dynamic loading, a routine is not loaded until it is called. This method is especially useful when large amounts of code are needed in order to handle infrequently occurring cases such as error routines.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 11

### ■ What are overlays in memory management?

The concept of overlays is that whenever a process is running it will not use the complete program at the same time; it will use only some part of it. The overlay concept says that whatever part you require, you load it, and once that part is done, you unload it. Formally: 'The process of transferring a block of program code or other data into internal memory, replacing what is already stored'.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is fragmentation in memory management?

Processes are stored and removed from memory, which makes free memory space, which is too little to even consider utilizing by other processes. Suppose that a process is not ready to dispense to memory blocks since its little size and memory hinder consistently staying unused is called fragmentation. This occurs during dynamic memory allotment.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 12

### ■ What is the basic function of paging?

Paging is a method or technique which is used for non-contiguous memory allocation. It is a fixed-size partitioning scheme. In paging, both main memory and secondary memory are divided into equal fixed-size partitions. Secondary memory is divided into pages and main memory into frames. Each process is split into fixed-size pages that are loaded into available frames in main memory. This enables efficient and flexible use of memory.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ How does swapping result in better memory management?

Swapping is a simple memory/process management technique used by the operating system to increase the utilization of the processor by moving some blocked processes from the main memory to the secondary memory, thus forming a queue of the temporarily suspended processes. Swapping allows more processes to be run than can fit into memory at one time.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 13

### ■ Write the name of classic synchronization problems?

Classic synchronization problems include: 1. Bounded-buffer, 2. Readers-writers, 3. Dining philosophers, and 4. Sleeping barber.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is the Direct Access Method & DMA?

The direct Access method is based on a disk model of a file, such that it is viewed as a numbered sequence of blocks or records. It allows arbitrary blocks to be read or written. Direct memory access (DMA) is a method that allows an input/output (I/O) device to send or receive data directly to or from the main memory, bypassing the CPU to speed up memory operations. The process is managed by a chip known as a DMA controller (DMAC).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 14

### ■ When does thrashing occur in operating systems?

Thrashing occurs when processes on the system frequently access pages that are not available in physical memory (causing continuous page faults).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is the best page size when designing an operating system?

The best paging size varies from system to system, so there is no single best when it comes to page size. Factors to consider: page table overhead, paging transfer time, and overall efficiency of the operating system.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 15

### ■ What is multitasking in operating systems?

Multitasking is a logical extension of a multiprogramming system that supports multiple programs to run concurrently. In multitasking, more than one task is executed at the same time, sharing common processing resources such as CPU cycles.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is caching?

The cache is a smaller and faster memory that stores copies of the data from frequently used main memory locations. Cache memory is used to reduce the average time to access data from the main memory.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 16

### ■ What is spooling?

Spooling is a process where data or jobs are temporarily stored in a buffer (usually in memory or on disk) before being sent to a peripheral device. It allows the CPU to continue working while slower devices (like printers or disk drives) access the data at their own pace.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is the functionality of an Assembler?

The Assembler is used to translate the program written in Assembly language into machine code. The source program is an input of an assembler that contains assembly language instructions. The output generated is the object code or machine code.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 17

### ■ What are interrupts?

Interrupts are signals emitted by hardware or software when a process or an event needs immediate attention. It alerts the processor to suspend the current working process to execute an Interrupt Service Routine (ISR).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is GUI?

GUI is short for Graphical User Interface. It provides users with an interface wherein actions can be performed by interacting with icons and graphical symbols rather than text command prompts.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 18

### ■ What is preemptive multitasking?

Preemptive multitasking is a type of multitasking that allows computer programs to share operating systems (OS) and underlying hardware resources. It divides the overall operating and computing time between processes, and the switching of resources occurs through predefined criteria enforced by the OS timer interrupt.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is a pipe and when is it used?

A Pipe is a technique used for inter-process communication (IPC). A pipe is a mechanism by which the output of one process is directed into the input of another process. Thus it provides a one-way flow of data between two related processes.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 19

### ■ What are the advantages of semaphores?

Advantages of semaphores include: They are machine-independent, easy to implement, correctness is easy to determine, can have many different critical sections protected with different semaphores, and they avoid wasted CPU cycles as there is no busy waiting.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is a bootstrap program in the OS?

Bootstrapping is the process of loading a set of instructions when a computer is first turned on or booted. During startup, diagnostic tests (POST) check configurations, and then the bootloader or bootstrap program is loaded to initialize the OS.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 20

### ■ What is IPC?

Inter-process communication (IPC) is a mechanism that allows processes to communicate with each other and synchronize their actions, allowing cooperation between active tasks.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the different IPC mechanisms?

IPC mechanisms include:

- **Pipes:** One-way data flow between related processes.
- **Named Pipes:** FIFO pipes accessible by unrelated processes.
- **Message Queuing:** Messages passed via a queue managed by the kernel.
- **Semaphores:** Synchronization values protecting critical sections.
- **Shared Memory:** Direct memory space access shared by multiple processes.
- **Sockets:** Network-based communication, computer and OS independent.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 21

### ■ What is a zombie process?

A process that has finished execution but still has an entry in the process table to report to its parent process is known as a zombie process. The entry is removed once the parent process reads the exit status.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are orphan processes?

A process whose parent process no longer exists (either finished or terminated without waiting for its child process to terminate) is called an orphan process. They are reaped by the init process (PID 1).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 22

### ■ What are starvation and aging in OS?

• **Starvation:** A resource management problem where a process does not get resources for a long time as they keep getting allocated to other tasks.
• **Aging:** A technique to avoid starvation by gradually increasing the priority of processes waiting in the queue over time.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Write about monolithic kernel?

A Monolithic Kernel manages system resources between application and hardware. User services and kernel services are implemented under the same address space. This increases execution speeds but makes the kernel size larger.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 23

### ■ What is Context Switching?

Switching of CPU to another process means saving the state of the old process and loading the saved state for the new process. The state of the process is saved in the Process Control Block (PCB).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is PCB?

The process control block (PCB) is a block that is used to track the process's execution status. It contains information about registers, priority, state, and memory allocations. The process table is an array of PCBs.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 24

### ■ When is a distributed system in a safe state?

A state is safe if there exists at least one temporal order in which all processes can run to completion without resulting in a deadlock.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is Cycle Stealing?

Cycle stealing is a method of accessing computer memory (RAM) or bus without interfering with the CPU, allowing DMA controllers to transfer data during unused bus cycles.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 25

### ■ What are a Trap and Trapdoor?

• **Trap:** A software-generated interrupt, usually the result of an error condition.
• **Trapdoor:** A secret undocumented entry point into a program used to bypass normal authentication access.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is a dispatcher?

The dispatcher is the module that gives process control over the CPU after selection by the short-term scheduler. It switches context, switches to user mode, and jumps to the program location to restart execution.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 26

### ■ Define the term dispatch latency?

Dispatch latency is the amount of time it takes for a system to stop one process and start another one running in the CPU.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the goals of CPU scheduling?

Goals include: Max CPU utilization, max throughput, min turnaround time, min waiting time, and min response time.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 27

### ■ What is a critical section?

A critical section is a code segment containing shared variables or resources that must be accessed atomically to maintain data consistency.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Write the name of synchronization techniques?

Synchronization techniques include: Mutexes, Semaphores, Condition variables, and File locks.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 28

### ■ What are the advantages of multithreading?

Advantages include: Improved throughput with concurrent CPU and I/O operations, efficient resource utilization, better application responsiveness, and simplified program structures.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the drawbacks of semaphores?

Drawbacks include: Priority inversion, lack of compiler enforcement (convention only), programmer overhead to track wait/signal calls, and deadlock risks.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 29

### ■ What is Peterson's approach?

It is a concurrent programming algorithm that synchronizes two processes to maintain mutual exclusion for shared resources using flags and a turn variable.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Define the term Bounded waiting?

Bounded waiting means that there is a limit on the number of times other processes can enter their critical sections after a process has requested entry, guaranteeing entry in finite time.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 30

### ■ What are the solutions to the critical section problem?

Solutions must satisfy: Software solutions (e.g. Peterson's), Hardware solutions (e.g. TestAndSet instructions), and Semaphore controls.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is concurrency?

Concurrency is the execution of multiple instruction sequences at the same time (either via multi-core parallelism or single-core time-slicing interleaving).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 31

### ■ Write a drawback of concurrency?

Drawbacks include: Required protection overhead, coordinate execution complexity, and performance overhead for context switching.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the necessary conditions which can lead to a deadlock?

Deadlocks require 4 simultaneous conditions: 1. Mutual Exclusion, 2. Hold and Wait, 3. No Preemption, and 4. Circular Wait.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 32

### ■ What are the issues related to concurrency?

Issues include: Non-atomic operations, race conditions, blocking delays, starvation, and deadlock states.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Why do we use precedence graphs?

A precedence graph is a directed acyclic graph (DAG) used to represent the execution order of statements or processes in operating systems.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 33

### ■ Explain the resource allocation graph (RAG)?

RAG is a directed graph depicting the state of a system in terms of processes (nodes) and resources (edges). It detects deadlocks if cycles are present.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is a deadlock?

Deadlock is a situation where two or more processes wait endlessly for resources held by each other, preventing any from completing execution.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 34

### ■ What is the goal and functionality of memory management?

Goals include: Relocation (load programs anywhere), Protection (prevent memory interference), Sharing (shared libraries), Logical organization, and Physical organization.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Explain address binding?

Address binding is the mapping of program instructions and data references to actual physical memory addresses.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 35

### ■ Write different types of address binding?

Binding can occur at: Compile-time, Load-time, or Execution-time.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Write an advantage of dynamic allocation algorithms?

Dynamic memory allocation is flexible when the memory size is unknown beforehand, enables dynamic lists/trees, and optimizes runtime memory footprint.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 36

### ■ Define Compaction?

Compaction is the process of collecting scattered memory fragments into a contiguous block by relocating active processes in RAM.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Write about the advantages and disadvantages of a hashed-page table?

• **Advantages:** Efficient page table searches for large address spaces.
• **Disadvantages:** Hash collisions slow down lookups; does not support null values.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 37

### ■ What is Locality of Reference?

Locality of reference is the tendency of a program to access the same or nearby memory locations over a short duration of time (temporal and spatial locality).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Write down the advantages of virtual memory?

Advantages include: High degree of multiprogramming, eliminates external fragmentation, simplifies programming as address space is virtual, and requires less physical RAM footprint.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 38

### ■ How do you calculate performance in virtual memory?

Calculated using Effective Access Time (EAT):
Effective Access Time = (1-p) * Memory Access Time + p * Page Fault Time
Where p is the page fault rate.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Write down the basic concept of the file system?

A file system is a collection of related files recorded on secondary storage. It represents the smallest allocation of logical storage for users.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 39

### ■ Write the names of different operations on files?

File operations include: Create, Open, Read, Write, Rename, Delete, Append, Truncate, and Close.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ Define the term Bit-Vector?

A Bit-Vector (or Bitmap) is an array of bits where each bit represents a disk block (0 indicates allocated, 1 indicates a free block).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 40

### ■ What is a File Allocation Table (FAT)?

FAT is a legacy table-based file system mapping clusters on a hard disk to determine where file segments are stored.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is rotational latency?

Rotational latency is the time taken by the desired sector of the disk platter to rotate under the read/write heads.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 41

### ■ What is seek time?

Seek time is the time taken to locate the disk arm to the specified track containing the target data sector.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What is Belady's Anomaly?

Belady's anomaly is the phenomenon where increasing the number of physical memory frames results in an increase in page faults (seen in FIFO page replacement).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 42

### ■ What happens if a non-recursive mutex is locked twice by the same thread?

It triggers a self-deadlock, because the thread waits forever for the mutex lock to be released, which it holds itself.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ What are the advantages of a multiprocessor system?

Advantages include: Enhanced system performance, concurrency inside applications, high throughput, and shared hardware overheads.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# Operating Systems (OS) Q&A Lecture Series - Session 43

### ■ What are real-time systems?

Real-time systems are computing systems where the correctness of operations depends not only on logical results but also on meeting strict execution deadlines.

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

### ■ How do you recover from a deadlock?

Deadlock recovery options: 1. Process termination (abort all or one-by-one), or 2. Resource preemption (take resources back and rollback process states).

### ■ Teacher's Placement Tip & Analogy:

When asked this in interviews, walk through a real-world parallel. For instance, processes vs threads is like separate corporate offices (isolated memory) vs employees sharing the same room (shared memory space). Discussing execution overheads shows depth of implementation knowledge.

# DBMS Study Block - Lesson 1: Introduction to DBMS & Key Features

A Database Management System (DBMS) is a software system designed to manage and organize data in a structured manner. It allows users to create, modify, and query a database, as well as manage security and access controls. Key Features include Data modeling, Data storage and retrieval, Concurrency control, Data integrity & security, and Backup & recovery. DBMS is classified into Relational DBMS (RDBMS - uses tables with keys) and Non-Relational DBMS (NoSQL - flexible schemas like documents, key-values, graphs).

## ■ High-Yield Interview Q&A:

**Q: Why is Introduction to DBMS & Key Features critical for databases?**

A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 2: Database Languages: DDL, DML, DCL, and TCL

Database languages are categorized into 4 main groups:
1. **Data Definition Language (DDL):** Deals with schemas and descriptions. Commands: CREATE (make objects), ALTER (alter structure), DROP (delete objects), TRUNCATE (remove all records, freeing space), COMMENT, and RENAME.
2. **Data Manipulation Language (DML):** Deals with data manipulation. Commands: SELECT (retrieve), INSERT (insert), UPDATE (update), DELETE (remove), MERGE (upsert), CALL, EXPLAIN PLAN, and LOCK TABLE.
3. **Data Control Language (DCL):** Acts as access specifiers. Commands: GRANT (give permissions) and REVOKE (withdraw permissions).
4. **Transactional Control Language (TCL):** Manages transactions. Commands: COMMIT (save changes), ROLLBACK (undo changes), and SAVEPOINT (temporary save points).

## ■ High-Yield Interview Q&A:

**Q: Why is Database Languages: DDL, DML, DCL, and TCL critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 3: Paradigm Shift from File System to DBMS

A File System manages data using flat files on a hard disk. Widespread issues in file-based systems forced the paradigm shift to DBMS:

- **Redundancy of Data:** Wasted space due to identical data copied in multiple files.
- **Inconsistency of Data:** Mismatched records across files due to incomplete updates.
- **Difficult Data Access:** Accessing a record is cumbersome and requires custom parsing code.
- **Unauthorized Access:** File permissions are weak, allowing unauthorized modifications.
- **No Concurrent Access:** Files can only be written by one user at a time.
- **No Backup and Recovery:** Risk of full data loss on system crashes.

## ■ High-Yield Interview Q&A:

**Q: Why is Paradigm Shift from File System to DBMS critical for databases?**

A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 4: DBMS Architecture: 1-Tier, 2-Tier, and 3-Tier Structures

Database topologies are structured depending on system size, scale, and concurrent user counts:
• **1-Tier Architecture:** The client, server, and database exist on the same local machine (e.g. SQL server setup locally for learning). Simple and cost-effective, but not used in production.
• **2-Tier Architecture:** A basic client-server model where client applications directly communicate with the database on the server using APIs (ODBC/JDBC). Fast retrieval for small scales, but poor performance under high traffic.
• **3-Tier Architecture:** An intermediate application server layer exists between the client and database. Clients talk to the app server, which handles business logic and communicates with the DB. Highly secure, scalable, and standard for web applications.

## ■ High-Yield Interview Q&A:

**Q: Why is DBMS Architecture: 1-Tier, 2-Tier, and 3-Tier Structures critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 5: Data Abstraction Levels & Data Independence

To manage complex database schemas and reduce developer cognitive load, DBMS provides 3 levels of abstraction:
1. **Physical or Internal Level:** The lowest level explaining how data is physically stored in memory blocks, files, B+ Trees, or hash indexes.
2. **Logical or Conceptual Level:** Explains what data is stored in the form of tables and the relationships among those tables. Hides physical storage blocks.
3. **View or External Level:** The highest level where users see customized views (rows and columns) of the database, hiding other tables and columns.

**Data Independence:** The ability to modify schema at one level without changing schema at the next level: Physical Data Independence (changing physical storage/disk devices without affecting conceptual tables) and Logical Data Independence (altering conceptual tables without affecting external user views).

## ■ High-Yield Interview Q&A:

**Q: Why is Data Abstraction Levels & Data Independence critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 6: Entity-Relationship (ER) Model symbols

The ER model is a graphical blueprint representing the logical structure of a database:
• **Rectangles:** Represent Entities (real-world objects like Student, Course).
• **Ellipses:** Represent Attributes (properties defining an entity, e.g. age, name).
• **Diamonds:** Represent Relationships among entities (e.g., Enrolls).
• **Double Ellipse:** Represent Multi-Valued Attributes (can have multiple values, e.g. Phone Number).
• **Double Rectangle:** Represents a Weak Entity (depends on a strong identifying entity for its key, e.g. Dependents).
• **Key Attribute:** Represented by an oval with underlined text (uniquely identifies an entity, e.g. Roll No).
• **Composite Attribute:** Composed of multiple attributes (represented by an oval branching into other ovals, e.g., Address branching into Street, City).
• **Derived Attribute:** Can be derived from other attributes (represented by a dashed oval, e.g., Age derived from DOB).

## ■ High-Yield Interview Q&A:

**Q: Why is Entity-Relationship (ER) Model symbols critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 7: Recursive Relationships and Self-Joins

A recursive relationship occurs when the same entity type participates more than once in a relationship type with different roles (e.g., in an Employee table, a manager is also an employee). This is represented as a self-join. For example: `CREATE TABLE employee (id INT PRIMARY KEY, name VARCHAR(50), manager_id INT, FOREIGN KEY (manager_id) REFERENCES employee(id));`. The foreign key references the primary key of the same table, establishing a hierarchy.

## ■ High-Yield Interview Q&A:

**Q: Why is Recursive Relationships and Self-Joins critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 8: ER-to-Relational Schema Mapping rules

Mapping rules translate ER diagrams into relational tables based on cardinality constraints:
• **1:1 Cardinality (Total Participation):** Merge all entities and the relationship into a single table. The primary key of the totally participating entity becomes the table key.
• **1:1 Cardinality (Partial Participation):** Convert into 2 tables. The primary key of one table is added as a foreign key to the other table, allowing NULL values.
• **1:N / N:1 Cardinality:** Convert into 2 tables. Add the primary key of the '1' side table as a foreign key in the 'N' side table (no new table is needed for the relationship).
• **M:N Cardinality:** Convert into 3 tables. Two tables represent the entities, and the third table represents the relationship itself, containing the primary keys of both entities as a composite key.
• **Weak Entity Mapping:** The weak entity becomes a separate table. Its primary key is a composite key combining the primary key of the strong identifying entity and its own partial key.

## ■ High-Yield Interview Q&A:

**Q: Why is ER-to-Relational Schema Mapping rules critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 9: Relational Model Terminologies

Relational database concepts proposed by E.F. Codd:
• **Attribute:** Properties that define a relation (columns, e.g., ROLL_NO).
• **Relation Schema:** The structure defining the relation name and attributes: `STUDENT (ROLL_NO, NAME, AGE)`.
• **Tuple:** A single row in a table representing a record instance.
• **Relation Instance:** The set of tuples at a specific point in time.
• **Degree:** The number of columns/attributes in a relation.
• **Cardinality:** The number of rows/tuples in a relation.
• **Null Value:** Represents missing, unknown, or not applicable values.

## ■ High-Yield Interview Q&A:

**Q: Why is Relational Model Terminologies critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 10: Key vs. Superkey Concepts

• **Superkey:** A set of one or more attributes that, taken collectively, can uniquely identify each tuple in a table. It can contain extra redundant attributes.
• **Candidate Key:** A minimal superkey. It contains no redundant attributes. It is the smallest set of attributes required for unique identification.
• **Primary Key:** The candidate key selected by the database designer to uniquely identify rows (cannot contain NULL values).
• **Composite Key:** A primary key composed of two or more attributes (used when a single attribute cannot guarantee uniqueness).

## ■ High-Yield Interview Q&A:

**Q: Why is Key vs. Superkey Concepts critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 11: Codd's 12 Rules of Relational Databases

E.F. Codd proposed 13 rules (numbered 0 to 12) to define the characteristics of a true Relational DBMS:
• **Rule 0 (Foundation Rule):** System must manage databases entirely through relational capabilities.
• **Rule 1 (Information Rule):** All information must be represented as cell values in tables.
• **Rule 2 (Guaranteed Access):** Every cell value must be accessible via Table Name + Primary Key + Column Name.
• **Rule 3 (Null Values Treatment):** Systematic support for representing missing/unknown values.
• **Rule 4 (Active Online Catalog):** Database structure must be stored in queryable system tables.
• **Rule 5 (Comprehensive Sub-language):** Must support DDL, DML, DCL, and TCL operations.
• **Rule 6 (View Updating):** All theoretically updatable views must be updatable by the system.
• **Rule 7 (High-level CRUD):** Support set-level insert, update, delete, union, and intersection operations.
• **Rule 8 (Physical Data Independence):** Changes in physical storage do not affect application queries.
• **Rule 9 (Logical Data Independence):** Changes in conceptual tables (e.g. splits) do not affect user views.
• **Rule 10 (Integrity Independence):** Constraints must be stored in the catalog, not hardcoded in apps.
• **Rule 11 (Distribution Independence):** Database queries must remain unchanged even if data is distributed across networks.
• **Rule 12 (Non-subversion Rule):** Low-level access interfaces cannot bypass relational security constraints.

## ■ High-Yield Interview Q&A:

**Q: Why is Codd's 12 Rules of Relational Databases critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 12: Relational Algebra Extended Operators: Joins

Relational algebra defines query operations. Extended operators derived from basic ones include:
• **Inner Join:** Selects rows where common fields match: `SELECT * FROM R INNER JOIN S ON R.id = S.id`. Unmatched records are discarded.
• **Left Outer Join:** Returns all rows from the left table and matched rows from the right. Unmatched right fields return NULL.
• **Right Outer Join:** Returns all rows from the right table and matched rows from the left. Unmatched left fields return NULL.
• **Full Outer Join:** Combines results of Left and Right joins, returning all rows. Unmatched values return NULL.
• **Natural Join:** Performs an inner join automatically matching all columns that share the same name across tables, returning the common column only once.

## ■ High-Yield Interview Q&A:

**Q: Why is Relational Algebra Extended Operators: Joins critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 13: Database Anomalies: Insertion, Deletion, and Update

Anomalies are data inconsistencies caused by poor database design (e.g., storing everything in a single flat file table):
• **Insertion Anomaly:** Occurs when you cannot insert data because certain other attributes are missing (e.g., you cannot insert student info unless they enroll in a course, if the primary key is a composite of student and course).
• **Deletion Anomaly:** Occurs when deleting a record accidentally deletes unrelated critical data (e.g., deleting a student record deletes the entire course details if they were the only student enrolled).
• **Update Anomaly:** Occurs when updating a value requires modifying multiple rows, creating a risk of data inconsistency if any row is missed (e.g., updating a teacher's phone number in every student record).
*To prevent these anomalies, databases are normalized (split into smaller tables).*

## ■ High-Yield Interview Q&A:

**Q: Why is Database Anomalies: Insertion, Deletion, and Update critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 14: Tuple Relational Calculus (TRC)

TRC is a non-procedural (declarative) query language used to describe *what* data is required rather than *how* to retrieve it. Queries are expressed as: `{ t | P(t) }` where `t` is a tuple variable and `P(t)` is a logical formula (predicate). Uses logical operators (AND, OR, NOT) and quantifiers: `∃ t ∈ r (Q(t))` (there exists a tuple satisfying Q(t)) and `∀ t ∈ r (Q(t))` (for all tuples, Q(t) is true). Example: Find loan details for amounts >= 10000: `{ t | t ∈ loan ∧ t[amount] >= 10000 }`.

## ■ High-Yield Interview Q&A:

**Q: Why is Tuple Relational Calculus (TRC) critical for databases?**

A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 15: Functional Dependencies & Canonical Cover

• **Functional Dependency (A -> B):** A constraint where the value of attribute A uniquely determines the value of attribute B.
• **Attribute Closure (A+):** The set of all attributes that can be functionally determined from A. Used to identify candidate keys.
• **Canonical Cover (Fc):** A minimal, simplified set of functional dependencies that has the same closure as the original set. It is computed by: 1. Combining dependencies with same left-side attributes using union rules, 2. Eliminating extraneous attributes from left/right sides, and 3. Minimizing redundant dependencies.

## ■ High-Yield Interview Q&A:

**Q: Why is Functional Dependencies & Canonical Cover critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 16: Database Normalization: 1NF, 2NF, and 3NF

Normalization structures tables to eliminate redundancy and anomalies:
• **First Normal Form (1NF):** Requires all column values to be atomic (no multivalued attributes, e.g. comma-separated phone numbers, and no composite attributes).
• **Second Normal Form (2NF):** Must be in 1NF and contain no partial dependencies. No non-prime attribute should depend on a proper subset of any candidate key.
• **Third Normal Form (3NF):** Must be in 2NF and contain no transitive dependencies. For every non-trivial dependency X -> Y, either X is a superkey or Y is a prime attribute.

## ■ High-Yield Interview Q&A:

**Q: Why is Database Normalization: 1NF, 2NF, and 3NF critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 17: Boyce-Codd Normal Form (BCNF)

BCNF is a stricter extension of 3NF. A relation is in BCNF if and only if, for every non-trivial functional dependency X -> Y, X is a superkey of the relation. In BCNF, no prime attribute is allowed to be functionally dependent on a non-superkey. BCNF guarantees lossless join decompositions but does not always preserve all functional dependencies.

## ■ High-Yield Interview Q&A:

**Q: Why is Boyce-Codd Normal Form (BCNF) critical for databases?**

A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 18: Concurrency Control Problems: Dirty Reads & Lost Updates

Concurrent transaction execution can trigger anomalies if read/write conflicts are not controlled:
• **Dirty Read Problem (Write-Read Conflict):** Transaction T1 modifies a row without committing. T2 reads the modified value. T1 aborts and rolls back. T2 is left with invalid 'dirty' data.
• **Lost Update Problem (Write-Write Conflict):** T1 and T2 read the same row. T1 updates and commits. T2 overwrites the row with its update, losing T1's changes.
• **Unrepeatable Read (Read-Write Conflict):** T1 reads a row. T2 modifies/deletes that row and commits. T1 rereads and gets a different value.

## ■ High-Yield Interview Q&A:

**Q: Why is Concurrency Control Problems: Dirty Reads & Lost Updates critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 19: Concurrency Protocols: Two-Phase Locking (2PL)

Locking synchronizes data access: Shared Locks (S - multiple reads, no writes) and Exclusive Locks (X - single write, blocks all others).
**Two-Phase Locking (2PL):** Guarantees serializability by enforcing two execution phases:
1. Growing Phase: Transactions acquire locks but release none.
2. Shrinking Phase: Transactions release locks but acquire no new ones.
• *Strict 2PL:* All Exclusive (X) locks held by a transaction must be held until it commits or aborts.
• *Rigorous 2PL:* All locks (both Shared and Exclusive) must be held until the transaction commits, ensuring easy serializability.
• *Conservative (Static) 2PL:* Transactions lock all required items before execution begins. Prevents deadlocks but has low concurrency.

## ■ High-Yield Interview Q&A:

**Q: Why is Concurrency Protocols: Two-Phase Locking (2PL) critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 20: Concurrency Protocols: Timestamp Ordering

Timestamp-based protocols order transactions using their start times (timestamps) instead of locks. Each database item X maintains two timestamp values: `W_TS(X)` (largest timestamp of successful writes) and `R_TS(X)` (largest timestamp of successful reads). • **Basic TO:** If transaction T issues a read/write that violates the timestamp order, T is aborted and restarted with a new timestamp. Deadlock-free but prone to cascading rollbacks.
• **Strict TO:** Delays read/write operations of T until the transaction that wrote to X commits or aborts, avoiding cascading rollbacks.

## ■ High-Yield Interview Q&A:

**Q: Why is Concurrency Protocols: Timestamp Ordering critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 21: Deadlock Prevention: Wait-Die vs. Wound-Wait

Timestamp-based schemes prevent deadlocks by aborting younger/older transactions when resource conflicts occur:
• **Wait-Die Scheme (Non-preemptive):** If older T1 requests a resource held by younger T2, T1 is allowed to wait. If younger T2 requests a resource held by older T1, T2 dies (aborts and restarts).
• **Wound-Wait Scheme (Preemptive):** If older T1 requests a resource held by younger T2, T1 wounds T2 (forces T2 to abort and release resource). If younger T2 requests a resource held by older T1, T2 is allowed to wait. Wound-wait triggers fewer rollbacks than Wait-Die.

## ■ High-Yield Interview Q&A:

**Q: Why is Deadlock Prevention: Wait-Die vs. Wound-Wait critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 22: Database Recovery: Deferred vs. Immediate Updates

Database recovery managers handle system crashes using transaction logs:
• **Deferred Update (No-Undo/Redo):** Database on disk is not updated until transaction commits. Updates are recorded in a workspace log first. On crash, committed transactions are re-applied (Redo), while active failed transactions require no changes (No-Undo).
• **Immediate Update (Undo/Redo):** Database can be updated while transaction is active. Changes are written to the transaction log before writing to disk. On crash, active uncommitted transactions are rolled back (Undo) and committed ones are re-applied (Redo).
• **Checkpoint Recovery:** Reduces recovery time by periodically saving active transaction states to a file, avoiding parsing the full log history.

## ■ High-Yield Interview Q&A:

**Q: Why is Database Recovery: Deferred vs. Immediate Updates critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 23: File Organizations: Sequential and Heap methods

Files organize physical disk blocks to optimize search times:
• **Sequential File Organization:** Records are stored in linear sequence. Implementing methods: Pile File (records stored in order of arrival, fast inserts but slow searches) and Sorted File (records sorted by a key, fast O(log N) searches but slow updates).
• **Heap File Organization:** Records are placed at the end of the file into data blocks with no ordering rules. Fast writes, but searching requires scanning all blocks (slow).

## ■ High-Yield Interview Q&A:

**Q: Why is File Organizations: Sequential and Heap methods critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# DBMS Study Block - Lesson 24: B-Trees & B+ Trees in DBMS Indexing

Balanced indexing structures optimize disk read operations:
• **B-Tree:** A self-balancing search tree where each node stores both search keys and data block pointers, reducing tree height. Time complexity: O(log N) for search, insert, and delete.
• **B+ Tree:** A variation of B-Tree where keys and data pointers are stored *only* in leaf nodes. Internal nodes contain routing keys only, allowing a high branching factor (shallower tree). Leaf nodes are linked sequentially, enabling fast range queries and sequential scans. B+ Trees are preferred for disk-oriented DBMS storage.

## ■ High-Yield Interview Q&A:

**Q: Why is B-Trees & B+ Trees in DBMS Indexing critical for databases?**
A: Relational structures rely on these consistency constraints, languages, and locks to handle concurrency safely. Without proper normalization or transaction isolations, parallel query execution would corrupt production user data.

# OOPs Study Block - Lesson 1: OOPs: Encapsulation & Data Hiding

Encapsulation groups variables and methods into classes. Data hiding restricts direct access to variables via access modifiers (private, protected).

## ■ High-Yield Interview Q&A:

**Q: What is the main utility of OOPs: Encapsulation & Data Hiding?**
A: Object-oriented models help manage complex codebase bases by grouping related attributes into logical wrappers, avoiding monolithic code structures, and making code components reusable.

# OOPs Study Block - Lesson 2: OOPs: Abstraction & Interfaces

Abstraction hides background complexities, exposing only API routes. Implemented via abstract classes and interface contracts.

## ■ High-Yield Interview Q&A:

**Q: What is the main utility of OOPs: Abstraction & Interfaces?**
A: Object-oriented models help manage complex codebase bases by grouping related attributes into logical wrappers, avoiding monolithic code structures, and making code components reusable.

# OOPs Study Block - Lesson 3: OOPs: Polymorphism (Compile-time vs. Run-time)

Polymorphism enables multiple behaviors: Compile-time (method overloading) and Run-time (virtual method overriding via vtables).

## ■ High-Yield Interview Q&A:

**Q: What is the main utility of OOPs: Polymorphism (Compile-time vs. Run-time)?**
A: Object-oriented models help manage complex codebase bases by grouping related attributes into logical wrappers, avoiding monolithic code structures, and making code components reusable.

# OOPs Study Block - Lesson 4: OOPs: Inheritance Models & Diamond Problem

Inheritance types include single, multiple, and hierarchical. C++ multiple inheritance causes diamond problems, resolved by virtual inheritance.

## ■ High-Yield Interview Q&A:

**Q: What is the main utility of OOPs: Inheritance Models & Diamond Problem?**
A: Object-oriented models help manage complex codebase bases by grouping related attributes into logical wrappers, avoiding monolithic code structures, and making code components reusable.

# OOPs Study Block - Lesson 5: OOPs: Abstract Classes vs. Interfaces

Abstract classes contain instance states and default method logic. Interfaces define contract methods with no instance fields.

## ■ High-Yield Interview Q&A:

**Q: What is the main utility of OOPs: Abstract Classes vs. Interfaces?**

A: Object-oriented models help manage complex codebase bases by grouping related attributes into logical wrappers, avoiding monolithic code structures, and making code components reusable.

# OOPs Study Block - Lesson 6: OOPs: Access Modifiers & Class Visibility

Enforces class accessibility boundaries: Public (accessible everywhere), Protected (accessible inside package/subclasses), Private (isolated to declaring class).

## ■ High-Yield Interview Q&A:

**Q: What is the main utility of OOPs: Access Modifiers & Class Visibility?**
A: Object-oriented models help manage complex codebase bases by grouping related attributes into logical wrappers, avoiding monolithic code structures, and making code components reusable.