

Theory Topics

1. ACID Properties in DBMS

<https://www.geeksforgeeks.org/acid-properties-in-dbms/>

2. What is OOPS?

OBJECT ORIENTED PROGRAMMING (OOP) is a programming concept that works on the principles of abstraction, encapsulation, inheritance, and polymorphism. It allows users to create the objects that they want and then, create methods to handle those objects. OOP stands for Object-Oriented Programming. **Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.**

Object-oriented programming has several advantages over procedural programming:

- a. OOP is faster and easier to execute
- b. OOP provides a clear structure for the programs
- c. OOP helps to keep the C++ code **DRY "Don't Repeat Yourself"**, and makes the code easier to maintain, modify and debug
- d. OOP makes it possible to create full reusable applications with less code and shorter development time

Classes and objects are the two main aspects of object-oriented programming. So, a class is a template for objects, and an object is an instance of a class. When the individual objects are created, they inherit all the variables and functions from the class. Everything in C++ is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is

an object. The car has attributes, such as weight and color, and methods, such as drive and brake.

Attributes and methods are basically **variables and functions** respectively that belong to the class. These are often referred to as "**class members**". A class is a user-defined data type that we can use in our program, and it works as an object constructor, or a "blueprint" for creating objects.

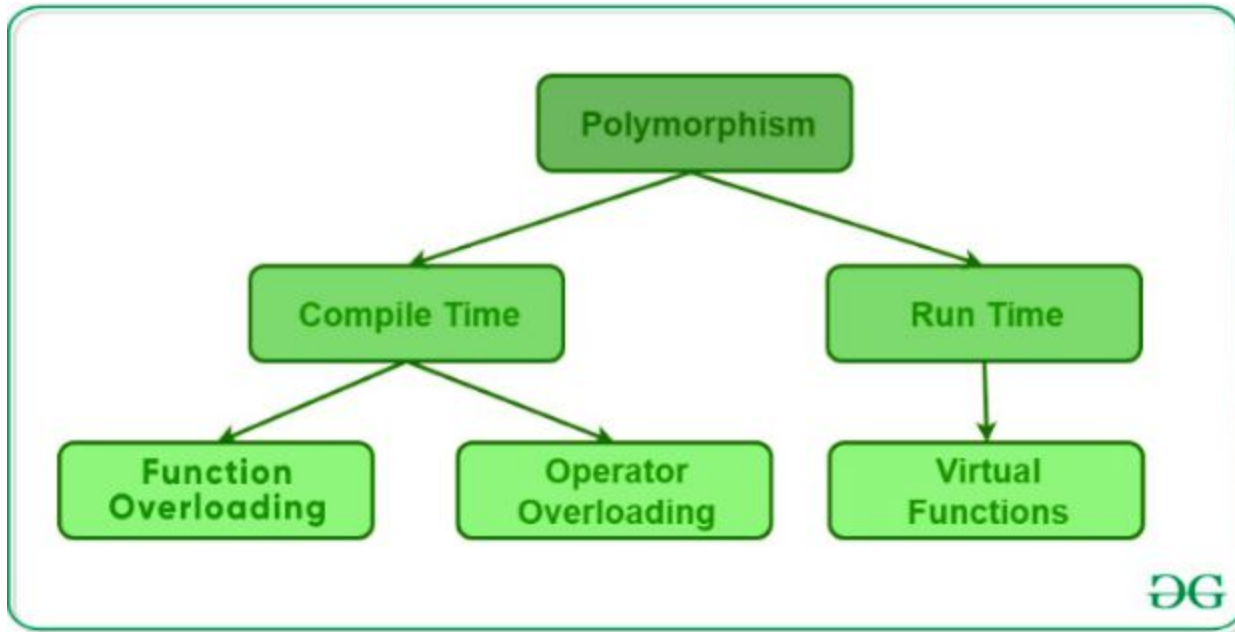
3. Main parts of OOPs

<https://www.javatpoint.com/java-oops-concepts>

<https://www.freecodecamp.org/news/object-oriented-programming-concepts-21bb035f7260/#:~:text=The%20four%20principles%20of%20object,abstraction%2C%20inheritance%2C%20and%20polymorphism.>

4. Polymorphism Concept

The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form. A real-life example of polymorphism, a person at the same time can have different characteristics. Like a man at the same time is a father, a husband, an employee. So the same person possesses different behavior in different situations. This is called polymorphism. Polymorphism is considered as one of the important features of Object Oriented Programming.



In C++ polymorphism is mainly divided into two types:

- a. **Compile time Polymorphism** - This type of polymorphism is achieved by function overloading or operator overloading.
 - **Function Overloading**: When there are multiple functions with the same name but different parameters then these functions are said to be **overloaded**. Functions can be overloaded by **change in number of arguments** or/and **change in type of arguments**.
 - **Operator Overloading**: C++ also provides options to overload operators. For example, we can make the operator ('+') for string class to concatenate two strings. We know that this is the addition operator whose task is to add two operands. So a single operator '+' when placed between integer operands, adds them and when placed between string operands, concatenates them.
- b. **Runtime Polymorphism** - This type of polymorphism is achieved by Function Overriding.
 - **Function overriding** on the other hand occurs when a derived class has a definition for one of the member functions of the base class. That base function is said to be **overridden**.

```

// C++ program for function overriding
#include <bits/stdc++.h>
using namespace std;

class base
{
public:
    virtual void print ()
    { cout<< "print base class" <<endl; }

    void show ()
    { cout<< "show base class" <<endl; }
};

class derived:public base
{
public:
    void print () //print () is already virtual function in derived class, we could also decl
    { cout<< "print derived class" <<endl; }

    void show ()
    { cout<< "show derived class" <<endl; }
};

//main function
int main()
{
    base *bptr;
    derived d;
    bptr = &d;

    //virtual function, binded at runtime (Runtime polymorphism)
    bptr->print();

    // Non-virtual function, binded at compile time
    bptr->show();

    return 0;
}

```

Output:

```

print derived class
show base class

```

5. DBMS vs R-DBMS

<https://www.geeksforgeeks.org/difference-between-rdbms-and-dbms/>

DBMS represents a Database Management System. It is used to create/update/delete and maintain a database and it provides controlled access to data. RDBMS, Relational Database Management System, is an enhanced version of DBMS.

Following are the important differences between DBMS and RDBMS.

Sr. No.	Key	DBMS	RDBMS
1	Definition	DBMS stands for DataBase Management System.	RDBMS stands for Relational DataBase Management System.
2	Data Storage	Data is stored as file.	Data is stored as tables.
3	Data Access	In DBMS, each data elements are to be accessed individually.	In RDBMS, multiple data elements can be accessed at same time.
4	Relationship	There is no relationship between data in DBMS.	Data is present in multiple tables which can be related to each other.
5	Normalization	Normalization cannot be achieved.	Normalization can be achieved.
6	Distributed database	DBMS has no support for distributed databases.	RDBMS supports distributed databases.
7	Data Quantity	DBMS deals with small quantity of data.	RDBMS deals with large quantity of data.
8	Data Redundancy	Data Redundancy is common in DBMS.	Data Redundancy can be reduced using key and indexes in RDBMS.
9	User	DBMS supports single user at a time.	RDBMS supports multiple users at a time.
10	Security	DBMS provides low security during data manipulation.	RDBMS has multilayer security during data manipulation.
11	Example	MSAccess.	Oracle, SQL Server.

6. Distributed vs Centralised Databases

<https://www.geeksforgeeks.org/difference-between-centralized-database-and-distributed-database/>

7. Generalization vs Specialization in DBMS

<https://www.geeksforgeeks.org/difference-between-generalization-and-specialization-in-dbm-s/?ref=rp>

8. Primary Key vs Foreign Key

In the relational database key is the most important element to maintain the relationship between two tables or to uniquely identify data from the table. **Primary key is used to identify data uniquely therefore two rows can't have the same primary key. It can't be null.**

On the other hand, foreign key is used to maintain relationship between two tables. Primary of a table act as foreign key in the other table. Foreign key in a table enforces Referential Integrity constraint. It can be more than one in the table.

Sr. No.	Key	Primary Key	Foreign Key
1	Basic	It is used to uniquely identify data in the table	It is used to maintain relationship between tables
2	Null	It can't be null	It can accept the null values
3	Duplicate	Two or more rows can't have same primary key	It can carry duplicate value for a foreign key attribute
4	Index	Primary has clustered index	By default, It is not clustered index
5	Tables	Primary key constraint can be defined on temporary table	It can't be defined on temporary tables

9. Quick Sort vs Merge Sort

<https://www.geeksforgeeks.org/quick-sort-vs-merge-sort/>

10. Postfix and Prefix Expression

<https://runestone.academy/runestone/books/published/pythonds/BasicDS/InfixPrefixandPostfixExpressions.html>

11. Linked list vs Array

<https://www.geeksforgeeks.org/linked-list-vs-array/>

12. Thrashing

To know what is thrashing, you must first be aware of swapping and page fault. So let's start with those concepts :

First we need to know what actually happens inside your OS that makes your system behave as if it has a very large memory. In operating systems that implement a virtual memory space, the programs allocate memory from an address space that may be much larger than the actual amount of RAM the system possesses. OS decides which program's memory is actually in RAM at any specific instant of time.

Page Fault and Swapping: A page fault occurs when the memory access requested (from the virtual address space) does not map to something that is in RAM. A page must then be sent from RAM to swap, so that the requested new page can be brought from swap to RAM. This results in 2 disk I/Os. Now you might know that disk I/Os are very slow as compared to memory access.

Thrashing: Now if it happens that your system has to swap pages at such a higher rate that a major **chunk of CPU time is spent in swapping** then this state is known as thrashing. So effectively during thrashing, the CPU spends less time in some actual productive work and more time in swapping i.e. **Thrashing** is a condition or a situation when the system is spending a major portion of its time in servicing the page faults, but the actual processing done is very negligible.



Now the effects of thrashing and also the extent to which thrashing occurs will be decided by the type of page replacement policy.

1. **Global Page Replacement:** The paging algorithm is applied to all the pages of the memory regardless of which process "owns" them. A page fault in one process may cause a replacement from any process in memory. Thus, the size of a partition may vary randomly.
2. **Local Page Replacement:** The memory is divided into partitions of a predetermined size for each process and the paging algorithm is applied independently for each region. A process can only use pages in its partition.

What happens after Thrashing starts?

If **global** page replacement is used, situations worsen very quickly. CPU thinks that CPU utilization is decreasing, so it tries to increase the degree of multiprogramming. Hence bringing more processes inside memory, which in effect increases the thrashing and brings down CPU utilization further down. The CPU notices that utilization is going further down, so it increases the degree of multiprogramming further and the cycle continues.

The solution can be **local** page replacement where a process can only be allocated pages in its own region in memory. If the swaps of a process increase also, the overall CPU utilization does not decrease much. If other transactions have enough page frames in the partitions they occupy, they will continue to be processed efficiently.

But local page replacement has its own disadvantages which you can now explore further.

<https://www.geeksforgeeks.org/techniques-to-handle-thrashing/>





13. Deadlock

<https://www.geeksforgeeks.org/introduction-of-deadlock-in-operating-system/#:~:text=Deadlock%20is%20a%20situation%20where,acquired%20by%20some%20other%20process.>

14. Orphan Processes vs Zombie Processes

Zombie Process - A process which has finished the execution but still has entry in the process table to report to its parent process is known as a zombie process. A child process always first becomes a zombie before being removed from the process table. The parent process reads the exit status of the child process which reaps off the child process entry from the process table.

In the following code, the child finishes its execution using `exit()` system call while the parent sleeps for 50 seconds, hence doesn't call `wait()` and the child process's entry still exists in the process table.

```

// A C program to demonstrate Zombie Process.
// Child becomes Zombie as parent is sleeping
// when child process exits.
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
int main()
{
    // Fork returns process id
    // in parent process
    pid_t child_pid = fork();

    // Parent process
    if (child_pid > 0)
        sleep(50);


    // Child process
    else
        exit(0);

    return 0;
}

```

Orphan Process - A process whose parent process no longer exists i.e. either finished or terminated without waiting for its child process to terminate is called an orphan process.

In the following code, the parent finishes execution and exits while the child process is still executing and is called an orphan process now. **However, the orphan process is soon adopted by the init process, once its parent process dies.**



```
// A C program to demonstrate Orphan Process.
// Parent process finishes execution while the
// child process is running. The child process
// becomes orphan.
#include<stdio.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
    // Create a child process
    int pid = fork();

    if (pid > 0)
        printf("in parent process");

    // Note that pid is 0 in child process
    // and negative if fork() fails
    else if (pid == 0)
    {
        sleep(30);
        printf("in child process");
    }

    return 0;
}
```

15. Cache

<https://www.geeksforgeeks.org/cache-memory-in-computer-organization/>

- Buffer between RAM and CPU. (also known as fast memory)
- Store frequently requested data and instructions so that they are immediately available to the CPU when needed.

16. Temporal vs Spatial Locality

<https://www.geeksforgeeks.org/difference-between-spatial-locality-and-temporal-locality/>

17. Show NAND and NOR are universal gates and to convert an expression using one of the universal gates.

18. Malloc vs Calloc

<https://www.geeksforgeeks.org/difference-between-malloc-and-calloc-with-examples/>

19. <https://www.geeksforgeeks.org/last-minute-notes-dbms/>

20. <https://www.geeksforgeeks.org/last-minute-notes-computer-network/>

21. <https://www.geeksforgeeks.org/last-minute-notes-computer-organization/?ref=rp>

22. <https://docs.microsoft.com/en-us/cpp/cpp/virtual-functions?view=vs-2019#:~:text=A%20virtual%20function%20is%20a,class's%20version%20of%20the%20function.>

23. Virtual Functions

<https://www.geeksforgeeks.org/virtual-function-cpp/>

24. <https://www.geeksforgeeks.org/commonly-asked-operating-systems-interview-questions-set-1/>

25. <https://career.guru99.com/top-50-operating-system-interview-questions/>

26. <https://www.geeksforgeeks.org/commonly-asked-dbms-interview-questions/>

27. <https://www.softwaretestinghelp.com/top-dbms-interview-questions/>

28. <https://www.javatpoint.com/dbms-interview-questions>

29. Integrity Constraints in DBMS

<https://www.javatpoint.com/dbms-integrity-constraints#:~:text=Integrity%20constraints%20are%20a%20set,data%20integrity%20is%20not%20affected.>

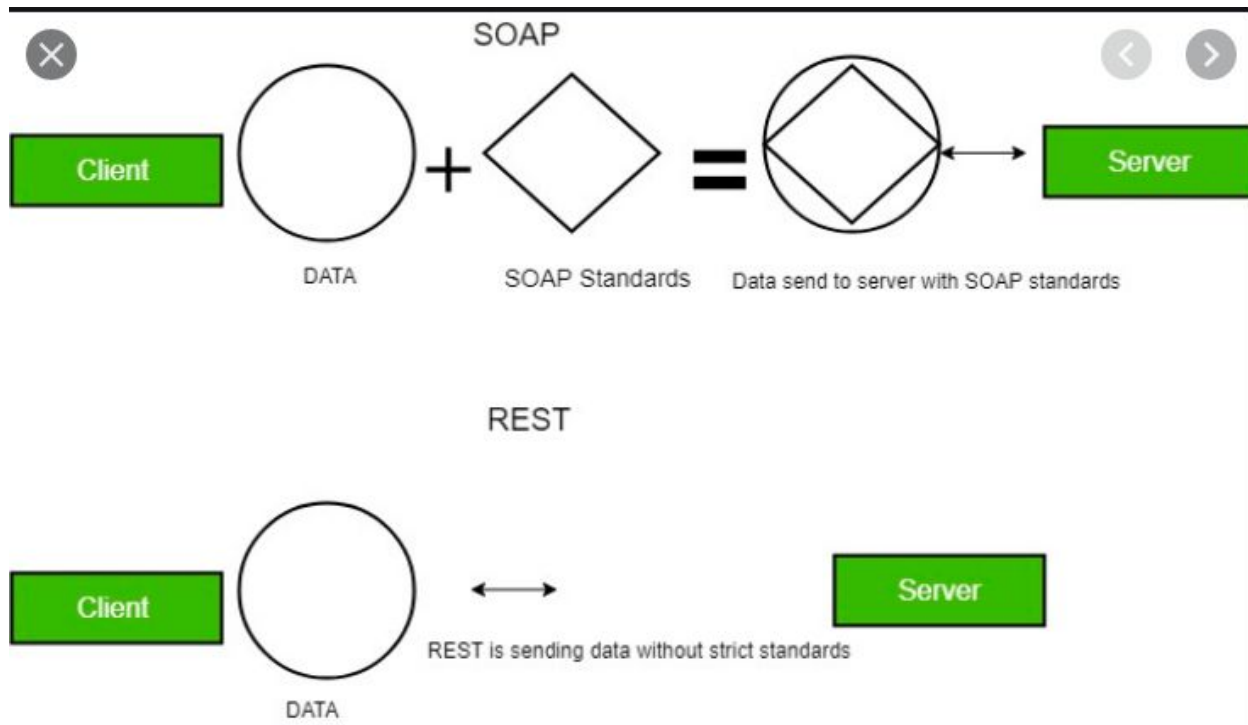
30. <https://www.geeksforgeeks.org/commonly-asked-computer-networks-interview-questions-set-1/>

31. <https://learning.naukri.com/articles/networking-interview-questions-answers/>

32. REST API vs SOAP API

REST API - Representational State Transfer

SOAP API - Simple Object Access Protocol



What are Rest and Soap APIs?

The communication/transfer/exchange of data that takes place between the clients and servers on the basis of some rules and regulations (known as web services) which basically defines the type and format of data that can be exchanged that the two ends abide by, kind of forms a communication system. Such communication systems are called REST or SOAP APIs.

Sr. No.	Key	REST API	SOAP API
1	Implementation	Rest API is implemented as it has no official standard at all because it is an architectural style.	On other hand SOAP API has an official standard because it is a protocol.
2	Internal communication	REST APIs uses multiple standards like HTTP, JSON, URL, and XML for data communication and transfer.	SOAP APIs is largely based and uses only HTTP and XML.
3	Resource requirement	As REST API deploys and uses multiple standards as stated above, so it takes fewer resources and bandwidth as compared to SOAP API.	On other hand Soap API requires more resource and bandwidth as it needs to convert the data in XML which increases its payload and results in the large sized file.
4	Description	REST API uses Web Application Description Language for describing the functionalities being offered by web services.	On other hand SOAP API used Web Services Description language for the same.
5	Security	REST has SSL and HTTPS for security.	On other hand SOAP has SSL(Secure Socket Layer) and WS-security due to which in the cases like Bank Account Password, Card Number, etc. SOAP is preferred over REST.
6	Abbreviation	REST stands for Representational State Transfer.	On other hand SOAP stands for Simple Object Access Protocol
7	Interchange	REST can make use of SOAP as the underlying protocol for web services, because in the end it is just an architectural pattern.	On other hand SOAP cannot make use of REST since SOAP is a protocol and REST is an architectural pattern.

33. React JS

React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It's 'V' in MVC. ReactJS is an open-source, component-based front end library responsible only for the view layer of the application. It designs simple views for each state in your application, and React will efficiently update and render just the right component when your

data changes. A react application is made of multiple components, each responsible for rendering a small, reusable piece of HTML. Components can be nested within other components to allow complex applications to be built out of simple building blocks. React is not a framework. It is just a library. React **Hot** Loader is a plugin that allows React components to be live reloaded without the loss of state. It works with Webpack and other bundlers that support both **Hot** Module Replacement (HMR) and Babel plugins.

34. Node JS

Node.js is an open source and cross-platform runtime environment for executing JavaScript code outside of a browser. You need to remember that **NodeJS is not a framework and it's not a programming language**. Most of the people are confused and understand it's a framework or a programming language. We often use Node.js for building back-end services like APIs like Web App or Mobile App.

35. Express JS

Express is a small framework that sits on top of Node.js's web server functionality to simplify its APIs and add helpful new features. It makes it easier to organize your application's functionality with middle ware and routing; it adds helpful utilities to Node.js's HTTP objects; it facilitates the rendering of dynamic HTTP objects.

Express is a lightweight web application framework for node.js used to build the back-end of web applications relatively fast and easily.

36. MongoDB

MongoDB, the most popular NoSQL database, is an open-source document-oriented database. The term 'NoSQL' means 'non-relational'. It means that MongoDB isn't based on the table-like relational database structure but provides an altogether different mechanism

for storage and retrieval of data. This format of storage is called BSON (similar to JSON format).

37. Real Life examples of Starvation, Deadlock, Race Condition.

Deadlock : You are building a house. To get a building permit, you need the loan from the bank. To get a loan from the bank, you need a building permit. To resolve it, either the bank or the local municipality has to realize that the procedure is flawed and make an exception. Or, you could get a “letter of intent” from one side which would prove the other side that the procedure is started, but pending.

Starvation : Airport has a single check-in counter. There are two queues, one for business and one for economy class. However, there is a business convention nearby, and the business queue is always full. The economy queue doesn't move at all. If you happen to be in economy class, you'll miss your plane.

Race Condition : A scholarship grant should be awarded to the first 100 applicants. A lot of applications have arrived and the envelopes are sorted by submission date. There are 3 people opening the envelopes and reviewing the applications. If their actions are not synchronized, it can happen that the 101th envelope is processed before 99th.

38. DFS vs BFS

Following are the important differences between BFS and DFS.

Sr. No.	Key	BFS	DFS
1	Definition	BFS, stands for Breadth First Search.	DFS, stands for Depth First Search.
2	Data structure	BFS uses Queue to find the shortest path.	DFS uses Stack to find the shortest path.
3	Source	BFS is better when target is closer to Source.	DFS is better when target is far from source.
4	Suitability for decision tree	As BFS considers all neighbour so it is not suitable for decision tree used in puzzle games.	DFS is more suitable for decision tree. As with one decision, we need to traverse further to augment the decision. If we reach the conclusion, we won.
5	Speed	BFS is slower than DFS.	DFS is faster than BFS.
6	Time Complexity	Time Complexity of BFS = $O(V+E)$ where V is vertices and E is edges.	Time Complexity of DFS is also $O(V+E)$ where V is vertices and E is edges.

39.

DFS visits all children nodes before visiting neighbours. For implementation, **BFS uses** a queue data structure, while **DFS uses** a stack. **BFS uses** a **larger** amount of **memory** because it expands all children of a vertex and keeps them in **memory**.

40. Get vs Post Requests

https://www.w3schools.com/tags/ref_httpmethods.asp

41. Process vs Threads

The major differences between a process and a thread are given as follows –

Comparison Basis	Process	Thread
Definition	A process is a program under execution i.e an active program.	A thread is a lightweight process that can be managed independently by a scheduler.
Context switching time	Processes require more time for context switching as they are more heavy.	Threads require less time for context switching as they are lighter than processes.
Memory Sharing	Processes are totally independent and don't share memory.	A thread may share some memory with its peer threads.
Communication	Communication between processes requires more time than between threads.	Communication between threads requires less time than between processes .
Blocked	If a process gets blocked, remaining processes can continue execution.	If a user level thread gets blocked, all of its peer threads also get blocked.
Resource Consumption	Processes require more resources than threads.	Threads generally need less resources than processes.
Dependency	Individual processes are independent of each other.	Threads are parts of a process and so are dependent.
Data and Code sharing	Processes have independent data and code segments.	A thread shares the data segment, code segment, files etc. with its peer threads.
Treatment by OS	All the different processes are treated separately by the operating system.	All user level peer threads are treated as a single task by the operating system.
Time for creation	Processes require more time for creation.	Threads require less time for creation.
Time for termination	Processes require more time for termination.	Threads require less time for termination.

42. What is race condition?

A **race condition** is a situation that may occur inside a critical section. This happens when the result of multiple thread execution in the critical section differs according to the order in which the threads execute.

43. What is the Virtual Memory concept in OS?

https://www.tutorialspoint.com/operating_system/os_virtual_memory.htm#:~:text=A%20computer%20can%20address%20more,to%20emulate%20the%20computer's%20RAM.

44. Paging vs Segmentation

Difference between Paging and Segmentation:

S.NO	PAGING	SEGMENTATION
1.	In paging, program is divided into fixed or mounted size pages.	In segmentation, program is divided into variable size sections.
2.	For paging operating system is accountable.	For segmentation compiler is accountable.
3.	Page size is determined by hardware.	Here, the section size is given by the user.
4.	It is faster in the comparison of segmentation.	Segmentation is slow.
5.	Paging could result in internal fragmentation.	Segmentation could result in external fragmentation.
6.	In paging, logical address is split into page number and page offset.	Here, logical address is split into section number and section offset.
7.	Paging comprises a page table which encloses the base address of every page.	While segmentation also comprises the segment table which encloses segment number and segment offset.
8.	Page table is employed to keep up the page data.	Section Table maintains the section data.
9.	In paging, operating system must maintain a free frame list.	In segmentation, operating system maintain a list of holes in main memory.
10.	Paging is invisible to the user.	Segmentation is visible to the user.
11.	In paging, processor needs page number, offset to calculate absolute address.	In segmentation, processor uses segment number, offset to calculate full address.

45. Semaphore vs Mutex

<https://www.geeksforgeeks.org/mutex-vs-semaphore/>

46. What are the types of Access Specifiers?

<https://www.geeksforgeeks.org/access-modifiers-in-c/>

47. What is the problem with Multiple Inheritance?

Multiple inheritance has been a sensitive **issue** for many years, with opponents pointing to its increased complexity and ambiguity in situations such as the "**diamond problem**", where it may be ambiguous as to which parent class a particular feature is **inherited** from if more than one parent class implements said feature.

48. What are the types of Inheritance?

49. <https://www.edureka.co/blog/interview-questions/oops-interview-questions/>

50. Call by Value vs Call by Reference

<https://www.geeksforgeeks.org/difference-between-call-by-value-and-call-by-reference/>

51. TCP vs UDP

<https://www.tutorialspoint.com/difference-between-tcp-and-udp#:~:text=As%20we%20know%20that%20both,a%20simpler%2C%20connectionless%20Internet%20protocol.>

52. Vertical vs Horizontal Scaling

Vertical Scaling : To increase the capacity if we increase resources in the same logical unit/server then it is vertical scaling. E.g. Add more CPUs to the existing server. If the system is not handled by one CPU then increase the CPU to 3 or 4.

Horizontal Scaling : when your business grows at the same time hits also grows so the responsibility of your server/node grows. So to reduce this responsibility what we can do is we can add one more server with the same capacity along with the existing server. Now these two servers can handle the traffic effectively.

53. LRU Cache Implementation

<https://www.geeksforgeeks.org/lru-cache-implementation/>

54. HTTP, https, TCP/IP, different layers(7) in networking

<https://www.geeksforgeeks.org/difference-between-http-and-https/>

<https://www.cloudflare.com/en-gb/learning/ssl/why-is-http-not-secure/#:~:text=HTTPS%20is%20HTTP%20with%20encryption,far%20more%20secure%20than%20HTTP>

<https://www.geeksforgeeks.org/tcp-ip-model/>

<https://www.geeksforgeeks.org/layers-of-osi-model/>

55. Write down all the data structures which I know. And then, to think of real-time applications of each of them.

- Linked Lists
- Queues
- Stacks
- Heap
- Arrays
- Trees
- Graphs

56. What is a pure virtual function and abstract class?

A pure virtual function is a virtual function in C++ for which we need not to write any function definition and only we have to declare it. It is declared by assigning 0 in the declaration.

An abstract class is a class in C++ which has at least one pure virtual function.

- a. Abstract class can have normal functions and variables along with a pure virtual function.
- b. Abstract class cannot be instantiated, but pointers and references of Abstract class type can be created.
- c. Abstract classes are mainly used for Upcasting, so that its derived classes can use its interface.
- d. If an Abstract Class has derived class, they must implement all pure virtual functions, or else they will become Abstract too.
- e. We can't create an object of abstract class as we reserve a slot for a pure virtual function in Vtable, but we don't put any address, so Vtable will remain incomplete.

Example Code

Live Demo

```
#include<iostream>
using namespace std;
class B {
public:
    virtual void s() = 0; // Pure Virtual Function
};

class D:public B {
public:
    void s() {
        cout << "Virtual Function in Derived class\n";
    }
};

int main() {
    B *b;
    D dobj;
    b = &dobj;
    b->s();
}
```

Output

```
Virtual Function in Derived class
```

<https://www.tutorialspoint.com/pure-virtual-functions-and-abstract-classes-in-cplusplus#:~:text=A%20pure%20virtual%20function%20is,least%20one%20pure%20virtual%20function.>

57. Internal vs External Fragmentation

Internal Fragmentation

Memory block assigned to process is bigger. Some portion of memory is left unused, as it cannot be used by another process. The internal fragmentation can be reduced by effectively assigning the smallest partition but large enough for the process.

External Fragmentation

Total memory space is enough to satisfy a request or to reside a process in it, but it is not contiguous, so it cannot be used. External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic.

Following are the important differences between Internal Fragmentation and External Fragmentation.

Sr. No.	Key	Internal Fragmentation	External Fragmentation
1	Definition	When there is a difference between required memory space vs allotted memory space, problem is termed as Internal Fragmentation.	When there are small and non-contiguous memory blocks which cannot be assigned to any process, the problem is termed as External Fragmentation.
2	Memory Block Size	Internal Fragmentation occurs when allotted memory blocks are of fixed size.	External Fragmentation occurs when allotted memory blocks are of varying size.
3	Occurrence	Internal Fragmentation occurs when a process needs more space than the size of allotted memory block or use less space.	External Fragmentation occurs when a process is removed from the main memory.
4	Solution	Best Fit Block Search is the solution for internal fragmentation.	Compaction is the solution for external fragmentation.
5	Process	Internal Fragmentation occurs when Paging is employed.	External Fragmentation occurs when Segmentation is employed.

58. Paging vs Demand Paging

Demand paging is identical to the paging system with swapping. In Demand Paging, a page is delivered into the memory on demand i.e., only when a reference is made to a location on that page. Demand paging combines the features of simple paging and implements virtual memory as it has a large virtual memory. Lazy swapper concept is

implemented in demand paging in which a page is not swapped into the memory unless it is required.

59. Demand Paging vs Segmentation

The difference between Demand Paging and Segmentation are as follows:

S.NO.	DEMAND PAGING	SEGMENTATION
1.	In demand paging, the pages are of equal size.	While in segmentation, segments can be of different size.
2.	Page size is fixed in the demand paging.	Segment size may vary in segmentation as it grants dynamic increase of segments.
3.	It does not allows sharing of the pages.	While segments can be shared in segmentation.
4.	In demand paging, on demand pages are loaded in the memory.	In segmentation, during compilation segments are allocated to the program.
5.	Page map table in demand paging manages record of pages in memory.	Segment map table in segmentation demonstrates every segment address in the memory.
6.	It provides large virtual memory and have more efficient use of memory.	It provides virtual memory and maximum size of segment is defined by the size of memory.

60. Virtual Functions vs Friend Functions

[https://www.quora.com/What-is-virtual-function-and-friend-function-in-C++-What-are-the-uses-of-both#:~:text=func\(\)%20th-,A%20virtual%20function%20is%20a%20member%20function%20that%20you,be%20redefined%20in%20derived%20classes.&text=A%20friend%20function%20of%20a,protected%20members%20of%20the%20class.](https://www.quora.com/What-is-virtual-function-and-friend-function-in-C++-What-are-the-uses-of-both#:~:text=func()%20th-,A%20virtual%20function%20is%20a%20member%20function%20that%20you,be%20redefined%20in%20derived%20classes.&text=A%20friend%20function%20of%20a,protected%20members%20of%20the%20class.)

61. Virtual vs Pure Virtual Functions

<https://www.geeksforgeeks.org/difference-between-virtual-function-and-pure-virtual-function-in-c/>

62. Concepts of Kernels

<https://afteracademy.com/blog/what-is-kernel-in-operating-system-and-what-are-the-various-types-of-kernel>

63. What is DMA?

- a. DMA also known as **Direct Memory access**.
- b. It 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**).
- c. It is a method of transferring data from the computer's RAM to another part of the computer without processing it using the CPU. While most data that is input or output from your computer is processed by the CPU, some data does not require processing, or can be processed by another device.
- d. Example a sound card may need to access data stored in the computer's RAM, but since it can process the data itself, it may use DMA to bypass the CPU.
- e. Types of DMA transfer:-

1) Memory to Memory Transfer 2)

<https://www.tutorialspoint.com/concept-of-direct-memory-access-dma>

[https://www.techopedia.com/definition/2767/direct-memory-access-dma#:~:text=Direct%20memory%20access%20\(DMA\)%20is,a%20DMA%20controller%20\(DMAC\).](https://www.techopedia.com/definition/2767/direct-memory-access-dma#:~:text=Direct%20memory%20access%20(DMA)%20is,a%20DMA%20controller%20(DMAC).)

<https://www.quora.com/What-is-the-function-of-DMA-in-a-computer>

64. Master's Theorem

<https://www.programiz.com/dsa/master-theorem#:~:text=In%20this%20tutorial%2C%20you%20will,b%20%3D%20size%20of%20each%20subproblem.>

65. System Calls

[https://www.geeksforgeeks.org/introduction-of-system-call/#:~:text=A%20system%20call%20is%20a,Application%20Program%20Interface\(API\).](https://www.geeksforgeeks.org/introduction-of-system-call/#:~:text=A%20system%20call%20is%20a,Application%20Program%20Interface(API).)

66. Constructor vs Destructor

<https://www.geeksforgeeks.org/difference-between-constructor-and-destructor-in-c/#:~:text=Constructor%20helps%20to%20initialize%20the,used%20to%20destroy%20the%20instances.>

67. Operating System vs Kernel

<https://www.tutorialspoint.com/difference-between-operating-system-and-kernel/#:~:text=Operating%20system%20is%20a%20system,a%20part%20of%20operating%20system.&text=Operating%20system%20acts%20as%20an,interface%20between%20applications%20and%20hardware.>

68. <https://www.educba.com/computer-science-interview-questions/>

69. <https://www.geeksforgeeks.org/most-asked-computer-science-subjects-interview-questions-in-a-mazon-microsoft-flipkart/>

70. Constructor vs Method

Constructor	Method
Constructor is used for initializing the instance of any class.	Method is used to perform some operation or function.
It does not have any return type	It has a return type.
The constructor name must be the same as a class name.	The name of the method can be the same or different as per need.
It calls automatically when you create a class object.	You need to call the method explicitly.
There is a default constructor which is provided by the compiler.	There is no method provided by the compiler.

71. Interfaces

https://www.tutorialspoint.com/cplusplus/cpp_interfaces.htm#:~:text=An%20interface%20describes%20the%20behavior,particular%20implementation%20of%20that%20class.&text=Thus%2C%20if%20a%20subclass%20of,interface%20declared%20by%20the%20ABC.

72. <https://www.algrim.co/74-100-best-computer-science-interview-questions-for-hiring>

73. CPU Cache vs TLB

<https://www.geeksforgeeks.org/whats-difference-between-cpu-cache-and-tlb/>

74. Compiler vs Interpreter

<https://www.geeksforgeeks.org/difference-between-compiler-and-interpreter/>

75. HTML vs CSS vs Javascript

<http://www.web-development-institute.com/the-difference-between-html-css-and-javascript>

76. DOM

"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."

https://www.w3schools.com/js/js_htmlDOM.asp

77. https://www.tutorialspoint.com/operating_system/os_overview.htm

78. DBMS: <https://www.edureka.co/blog/interview-questions/dbms-interview-questions>

79. OS link: <https://career.guru99.com/top-50-operating-system-interview-questions/>

80. OOPS link: <https://career.guru99.com/top-50-oops-interview-questions/>

81. <https://www.javatpoint.com/cpp-interview-questions>

82. Subclass vs Inner Class

<https://stackoverflow.com/questions/33421432/inner-classes-vs-subclasses-in-java#:~:text=inner%20classes%20are%20in%20the,methods%20of%20their%20parent%20class>

83. Primary vs Secondary Memory :

<https://www.geeksforgeeks.org/difference-between-primary-and-secondary-memory/#:~:text=The%20memory%20devices%20used%20for,are%20magnetic%20and%20optical%20memories.&text=Primary%20memory%20is%20also%20known,External%20memory%20or%20Auxiliary%20memory>

84. Static Keyword

<https://www.javatpoint.com/static-keyword-in-java>

85. Normalization in DBMS

Normalization is a technique of organising data into multiple related tables to minimise data redundancy.

- Repetition of data increases the size of the database.
- Other issues like:
 - **Insertion anomaly** - To insert redundant data for every new row is a data insertion problem for anomaly
 - **Deletion anomaly** - Loss of related data set with some other data set is deleted.
 - **Updation anomaly**

STUDENTS TABLE				
rollno	name	branch	hod	office_tel
1	Akon	CSE	Mr. X	53337
2	Bkon	CSE	Mr. X	53337
3	Ckon	CSE	Mr. X	53337
4	Dkon	CSE	Mr. X	

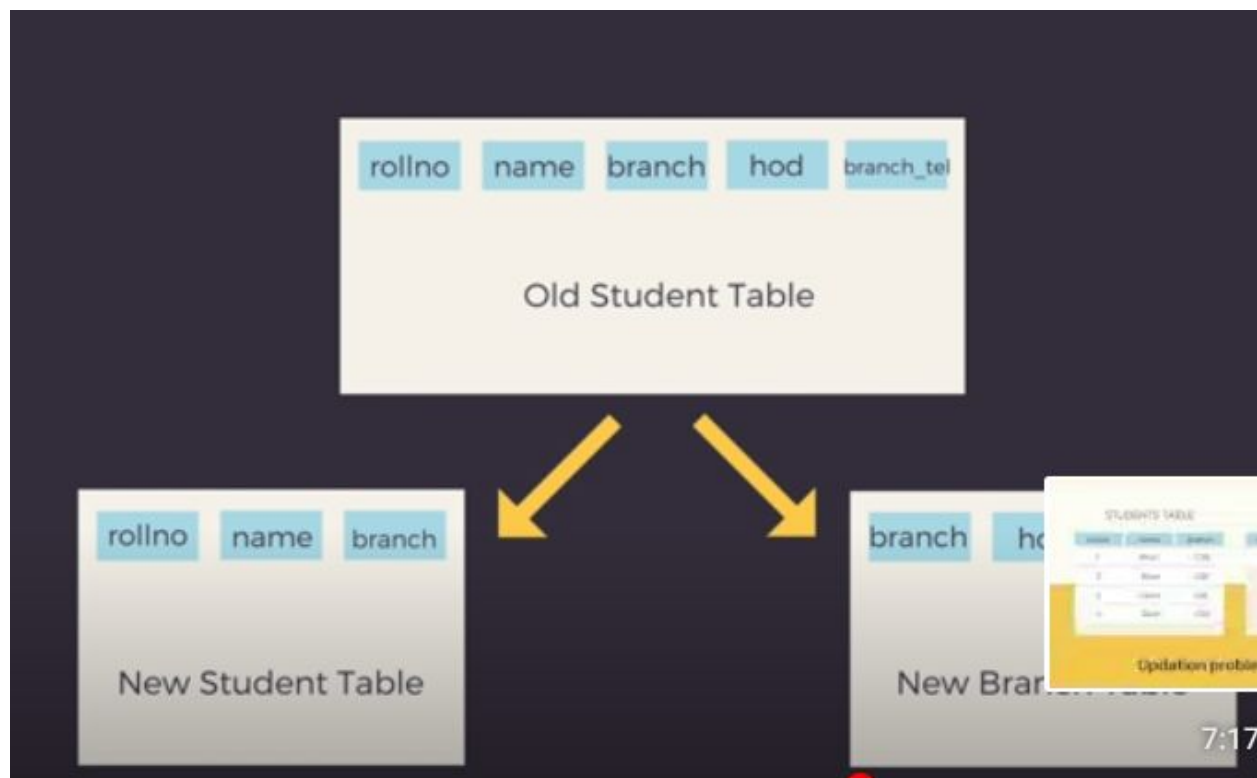
Here the data redundancy is - Different but related data is put in the same table and hence the following anomalies arise.

Here the insertion anomaly is, for each new student we have to add the redundant data of branch HOD and office telephone.

Here the deletion anomaly is, if all the students are deleted from the students table the branch information is lost.

Here the updation anomaly is, if the HOD is changed the HOD information will have to be changed in all the rows.

This is how we can reduce redundancy by normalization.



Types of Normalization :

- **1st Normal Form**

- Each Column should contain atomic values (single values in each cell of the table instead of multiple values).

- A column should contain values that are of the same type/datatype.
- Each column/attribute should have a unique name.
- The order in which we store the data does not matter.

Eg :

STUDENTS TABLE		
rollno	name	subject
101	Akon	OS, CN
103	Ckon	JAVA
102	Bkon	C, C++

Violation of 1 NF

Can be solved as :

STUDENTS TABLE		
rollno	name	subject
101	Akon	OS
101	Akon	CN
103	Ckon	JAVA
102	Bkon	C
102	Bkon	C++

● 2nd Normal Form

- The table should be in 1st Normal Form.
- It should not have any **partial dependencies**.

We have 2 tables :

student_id	name	reg_no	branch	address
1	Akon	CSE-18	CSE	TN
2	Akon	IT-18	IT	AP
3	Bkon	CSE-18	CSE	HR
4	Ckon	CSE-18	CSE	MH

subject_id	subject_name

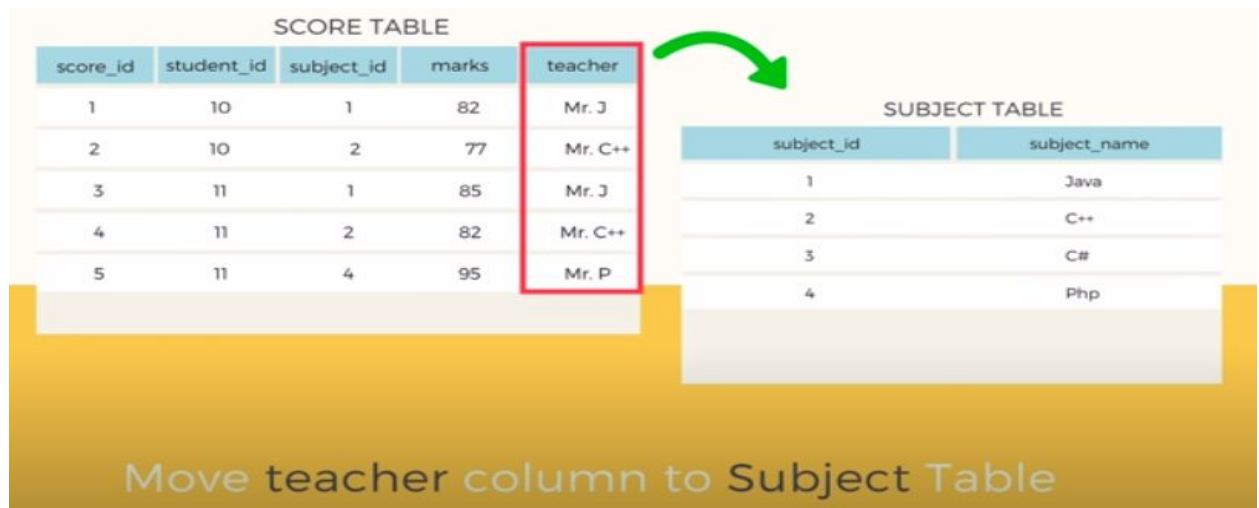
Dependencies in DBMS is a relation between two or more attributes.

Here every other attribute is dependent on the **primary key i.e. student id**.

score_id	student_id	subject_id	marks	teacher
1	1	1	82	Mr. J
2	1	2	77	Mr. C++
3	2	1	85	Mr. J
4	2	2	82	Mr. C++
5	2	4	95	Mr. P

In the above table score ID can be used as a primary key but the more meaningful primary key is student ID + subject ID. The teacher attribute is only dependent on subject ID but not on student ID. This is known as partial dependency. Does this is violating 2NF form.

This can be solved by : Remove Teacher's name from score table.



SUBJECT TABLE		
subject_id	subject_name	teacher
1	Java	Mr. J
2	C++	Mr. C++
3	C#	Mr. C#
4	Php	Mr. P

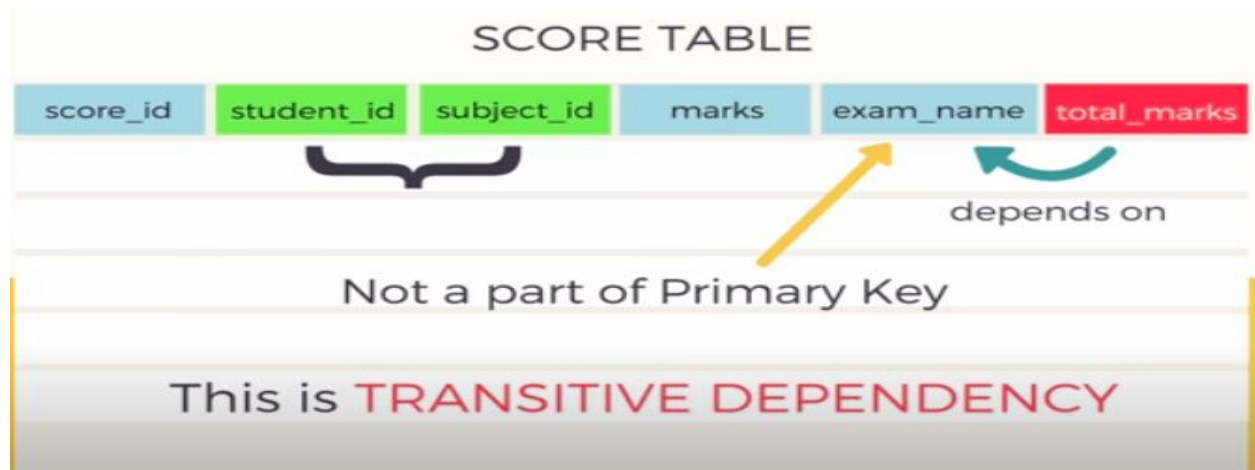
It can also be solved as :

Teacher TABLE	
teacher_id	teacher_name
1	Mr. J
2	Mr. C++
3	Mr. C#
4	Mr. P

- **3rd Normal Form**

- The table must be in the 2nd Normal Form.
- It should not have **transitive dependency**.

Transitive Dependency - When an attribute in the table depends upon some non-prime attribute of the table. (prime attributes are the ones which form the primary key)



This can be solved as :

SCORE TABLE					
score_id	student_id	subject_id	marks	exam_name	total_marks

Diagram illustrating the structure of four database tables:

- Student Table**: Columns: student_id, name, reg_no, branch, address.
- Subject Table**: Columns: subject_id, subject_name, teacher.
- Score Table**: Columns: score_id, student_id, subject_id, marks, exam_name.
- Exam Table**: Columns: exam_name, total_marks.

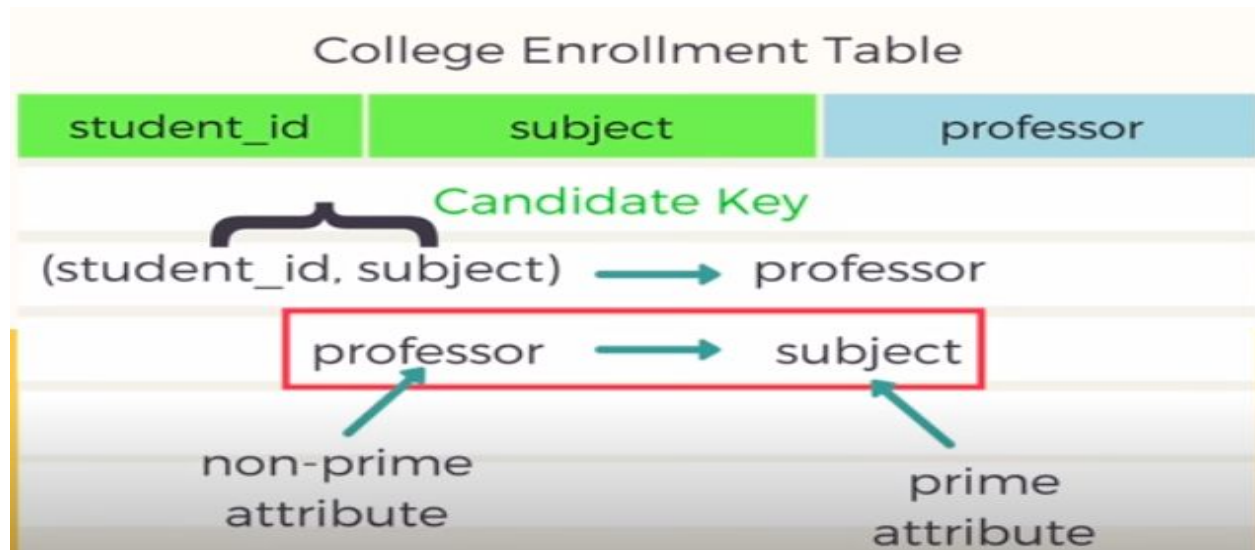
A green diagonal watermark reads "In 3rd Normal Form".

- **BCNF (Boyce-Codd Normal Form)**

- It should be in the 3rd Normal Form.
- For any dependency $A \rightarrow B$, A should be a super key which means, If A cannot be a prime attribute with B being a prime attribute.

Eg

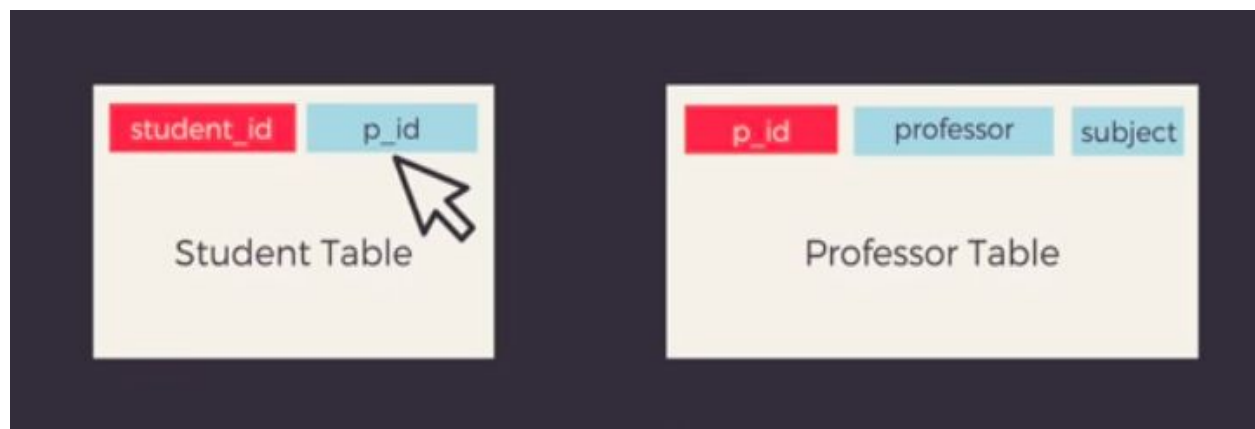
:



Here the subject is dependent upon the Professor but the professor is not the primary key i.e.

Professor can find the subject.

This can be solved as :



- **4th Normal Form**

- The table should satisfy BCNF.
- It should not have Multi-Valued Dependency.

Multi-Valued Dependency

- **5th Normal Form**

As the data requirement increases, database complexity increases. and the need for normalisation too increases.

Machine Learning

Interview Questions

1. <https://www.geeksforgeeks.org/10-basic-machine-learning-interview-questions/>
2. <https://www.analyticsvidhya.com/blog/2016/09/40-interview-questions-asked-at-startups-in-machine-learning-data-science/>
3. <https://elitedatascience.com/machine-learning-interview-questions-answers>
4. <https://www.simplilearn.com/tutorials/machine-learning-tutorial/machine-learning-interview-questions>
5. <https://www.analyticsvidhya.com/blog/2017/04/40-questions-on-probability-for-all-aspiring-data-scientists/>
6. <https://www.indeed.com/career-advice/interviewing/research-interview-questions>
7. <https://www.edureka.co/blog/interview-questions/artificial-intelligence-interview-questions/>
8. <https://towardsdatascience.com/top-30-data-science-interview-questions-7dd9a96d3f5c>
9. <https://www.simplilearn.com/tutorials/data-science-tutorial/data-science-interview-questions>
10. <https://www.guru99.com/machine-learning-interview-questions.html>
11. <https://www.indeed.com/career-advice/interviewing/research-interview-questions>

Linear Algebra Notes

1. https://kaustabpal.github.io/kaustabpal.github.io//linear_algebra#linear-independence
2. <https://www.geeksforgeeks.org/ml-linear-algebra-operations/>

3. <https://www.analyticsvidhya.com/blog/2017/05/comprehensive-guide-to-linear-algebra/>

ML Concepts/Theory

1. <https://www.geeksforgeeks.org/machine-learning/>
2. <https://github.com/kryptc/smai-lecture-notes>
3. <https://www.geeksforgeeks.org/ensemble-classifier-data-mining/>
4. <https://www.geeksforgeeks.org/ml-bagging-classifier/>
5. <https://www.geeksforgeeks.org/boosting-in-machine-learning-boosting-and-adaboost/?ref=rp>
6. <https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205>
7. <https://www.geeksforgeeks.org/comparison-b-w-bagging-and-boosting-data-mining/>
8. <https://stats.stackexchange.com/questions/18891/bagging-boosting-and-stacking-in-machine-learning>
9. <https://www.geeksforgeeks.org/naive-bayes-classifiers/>
10. <https://www.geeksforgeeks.org>
11. </random-forest-regression-in-python/>
12. <https://machinelearningmastery.com/logistic-regression-for-machine-learning/>
13. <https://machinelearningmastery.com/linear-regression-for-machine-learning/>
14. <https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/?ref=rp>
15. <https://www.geeksforgeeks.org/cross-validation-machine-learning/?ref=rp>
16. <https://www.geeksforgeeks.org/confusion-matrix-machine-learning/?ref=rp>
17. <https://www.geeksforgeeks.org/gradient-descent-in-linear-regression/?ref=lbp>
18. <https://www.geeksforgeeks.org/what-is-reinforcement-learning/?ref=lbp>
19. Real Life Examples of PCA, SVD, Eigen Decomposition

20. Logistic Regression, Naive Bayes, PCA, SVD, Gradient Descent, MLE , monty hall problem, Standard Deviation in Gaussian Distribution
21. <https://www.geeksforgeeks.org/difference-between-machine-learning-and-artificial-intelligence/?ref=lbp>