INTERVIEW QUESTIONS

C INTERVIEW QUESTIONS:

Q1. What are the basic Datatypes supported in C Programming Language?

Ans: The Datatypes in C Language are broadly classified into 4 categories. They are as follows:

- Basic Datatypes
- Derived Datatypes
- Enumerated Datatypes
- Void Datatypes

Q2. What do you mean by Dangling Pointer Variable in C Programming?

Ans: A **Pointer** in C Programming is used to point the memory location of an existing variable. In case if that particular variable is deleted and the Pointer is still pointing to the same memory location, then that particular pointer variable is called as a **Dangling Pointer Variable.**

Q3. What do you mean by the Scope of the variable? What is the scope of the variables in C?

Ans: **Scope** of the variable can be defined as the part of the code area where the variables declared in the program can be accessed directly. In C, all identifiers are lexically (or statically) scoped.

Q4. What are static variables and functions?

Ans: The variables and functions that are declared using the keyword **Static** are considered as Static Variable and Static Functions. The variables declared using Static keyword will have their scope restricted to the function in which they are declared.

Q5. Differentiate between calloc() and malloc()

Ans: **calloc()** and **malloc()** are memory dynamic memory allocating functions. The only difference between them is that calloc() will load all the assigned memory locations with value 0 but malloc() will not.

Q6. What are the valid places where the programmer can apply Break Control Statement?

Ans: Break Control statement is valid to be used inside a loop and **Switch control statements.**

Q7. How can we store a negative integer?

Ans: To store a negative integer, we need to follow the following steps. Calculate the two's complement of the same positive integer.

Eg: 1011 (-5)

Step-1 − One's complement of 5: 1010

Step-2 − Add 1 to above, giving 1011, which is -5

Q8. Differentiate between Actual Parameters and Formal Parameters.

Ans: The Parameters which are sent from main function to the subdivided function are called as **Actual Parameters** and the parameters which are declared a the Subdivided function end are called as **Formal Parameters.**

Q9. Can a C program be compiled or executed in the absence of a main()?

Ans: The program will be compiled but will not be executed. To execute any C program, main() is required.

Q10. What do you mean by a Nested Structure?

Ans: When a data member of one structure is referred by the data member of another function, then the structure is called a **Nested Structure.**

Q11. What is a C Token?

Ans: *Keywords, Constants, Special Symbols, Strings, Operators, Identifiers* used in C program are referred to as **C Tokens.**

Q12. What is Preprocessor?

Ans: A Preprocessor Directive is considered as a built-in predefined function or macro that acts as a directive to the compiler and it gets executed before the actual C Program is executed.

Q13. What is the purpose of printf() and scanf() in C Program?

Ans: printf() is used to print the values on the screen. To print certain values, and on the other hand, **scanf()** is used to scan the values. We need an appropriate datatype format specifier for both printing and scanning purposes. For example,

- **%d**: It is a datatype format specifier used to print and scan an **integer** value.
- **%s**: It is a datatype format specifier used to print and **scan** a string.
- **%c**: It is a datatype format specifier used to display and scan a **character** value.
- **%f**: It is a datatype format specifier used to display and scan a **float** value.

Q14. What is an array?

Ans. The array is a simple data structure that stores multiple elements of the same datatype in a reserved and sequential manner. There are three types of arrays, namely,

- One Dimensional Array
- Two Dimensional Array
- Multi-Dimensional Array

Q15. What is /0 character?

Ans: The Symbol mentioned is called a **Null Character**. It is considered as the terminating character used in strings to notify the end of the string to the compiler.

Q16. What is the main difference between the Compiler and the Interpreter?

Ans: Compiler is used in C Language and it translates the complete code into the Machine Code in one shot. On the other hand, Interpreter is used in Java Programming Langauge and other high-end programming languages. It is designed to compile code in line by line fashion.

Q17. Can I use int datatype to store 32768 value?

Ans: No, Integer datatype will support the range between **-32768 and 32767.** Any value exceeding that will not be stored. We can either use **float** or **long int.**

Q18. What is Dynamic Memory allocation? Mention the syntax.

Ans: Dynamic Memory Allocation is the process of allocating memory to the program and its variables in runtime. Dynamic Memory Allocation process involves three functions for allocating memory and one function to free the used memory.

Q19. Where can we not use &(address operator in C)?

Ans: We cannot use **&** on **constants** and on a variable which is declared using the **register storage** class.

Q20. Differentiate between call by value and call by reference

Ans:

Factor	Call by Value	Call by Reference
Safety	Actual arguments cannot be changed and remain safe	Operations are performed on actual arguments, hence not safe
Memory Location	Separate memory locations are created for actual and formal arguments	Actual and Formal arguments share the memory space.
Arguments	Copy of actual arguments are sent	Actual arguments are passed

Q21. Differentiate between getch() and getche()

Ans: Both the functions are designed to read characters from the keyboard and the only difference is that

getch(): reads characters from the keyboard but it does not use any buffers. Hence, data is not displayed on the screen.

getche(): reads characters from the keyboard and it uses a buffer. Hence, data is displayed on the screen.

Q22. Explain toupper() with an example.

Ans. toupper() is a function designed to convert lowercase words/characters into upper case.

Q23. Explain Local Static Variables and what is their use?

Ans: A local static variable is a variable whose life doesn't end with a function call where it is declared. It extends for the lifetime of the complete program. All calls to the function share the same copy of local static variables.

Q24. When should we use the register storage specifier?

Ans: We use Register Storage Specifier if a certain variable is used very frequently. This helps the compiler to locate the variable as the variable will be declared in one of the CPU registers.

Q25. Which statement is efficient and why? x=x+1; or x++;

Ans: x++; is the most efficient statement as it just a single instruction to the compiler while the other is not.

Q26. Which variable can be used to access Union data members if the Union variable is declared as a pointer variable?

Ans: Arrow Operator(->) can be used to access the data members of a Union if the Union Variable is declared as a pointer variable.

Q27. Mention File operations in C Language.

Ans: Basic File Handling Techniques in C, provide the basic functionalities that user can perform against files in the system.

Function	Operation
fopen()	**To Open a File**
fclose()	**To Close a File**
fgets()	**To Read a File**
fprint()	**To Write into a File**

Q28. What are the different storage class specifiers in C?

Ans: The different storage specifiers available in C Language are as follows:

- **auto**
- **register**
- **static**
- **extern**

Q29. What is typecasting?

Ans: Typecasting is a process of converting one data type into another is known as typecasting. If we want to store the floating type value to an int type, then we will convert the data type into another data type explicitly.

JAVA INTERVIEW QUESTIONS:

Q1. Explain JDK, JRE and JVM?

JDK	JRE	JVM
It stands for Java Development Kit.	It stands for Java Runtime Environment.	It stands for Java Virtual Machine.
It is the tool necessary to compile, document and package Java programs.	JRE refers to a runtime environment in which Java bytecode can be executed.	It is an abstract machine. It is a specification that provides a run-time environment in which Java bytecode can be executed.
It contains JRE + development tools.	It's an implementation of the JVM which physically exists.	JVM follows three notations: Specification, **Implementation,** and **Runtime Instance**.

Q2. Explain public static void main(String args[]) in Java.

Ans: main() in Java is the entry point for any Java program. It is always written as **public static void main(String[] args)**.

- **public**: Public is an access modifier, which is used to specify who can access this method. Public means that this Method will be accessible by any Class.
- **static**: It is a keyword in java which identifies it is class-based. main() is made static in Java so that it can be accessed without creating the instance of a Class. In case, main is not made static then the compiler will throw an error as **main**() is called by the JVM before any objects are made and only static methods can be directly invoked via the class.
- **void**: It is the return type of the method. Void defines the method which will not return any value.
- **main**: It is the name of the method which is searched by JVM as a starting point for an application with a particular signature only. It is the method where the main execution occurs.
- **String args[]**: It is the parameter passed to the main method.

Q3. Why Java is platform independent?

Ans: Java is called platform independent because of its byte codes which can run on any system irrespective of its underlying operating system.

Q4. Why Java is not 100% Object-oriented?

Ans: Java is not 100% Object-oriented because it makes use of eight primitive data types such as boolean, byte, char, int, float, double, long, short which are not objects.

Q5. What are wrapper classes in Java?

Ans: Wrapper classes convert the Java primitives into the reference types (objects). Every primitive data type has a class dedicated to it. These are known as wrapper classes because they "wrap" the primitive data type into an object of that class. Refer to the below image which displays different primitive type, wrapper class and constructor argument.

Q6. What are constructors in Java?

Ans: In Java, constructor refers to a block of code which is used to initialize an object. It must have the same name as that of the class. Also, it has no return type and it is automatically called when an object is created.

There are two types of constructors:

1. **Default Constructor:** In Java, a default constructor is the one which does not take any inputs. In other words, default constructors are the no argument constructors which will be created by default in case you no other constructor is defined by the user. Its main purpose is to initialize the instance variables with the default values. Also, it is majorly used for object creation.
2. **Parameterized Constructor:** The parameterized constructor in Java, is the constructor which is capable of initializing the instance variables with the provided values. In other words, the constructors which take the arguments are called parameterized constructors.

Q7. What is singleton class in Java and how can we make a class singleton?

Ans: Singleton class is a class whose only one instance can be created at any given time, in one JVM. A class can be made singleton by making its constructor private.

Q8. What is the difference between Array list and vector in Java?

ArrayList	Vector
Array List is not synchronized.	Vector is synchronized.
Array List is fast as it's non-synchronized.	Vector is slow as it is thread safe.
If an element is inserted into the Array List, it increases its Array size by 50%.	Vector defaults to doubling size of its array.
Array List does not define the increment size.	Vector defines the increment size.
Array List can only use Iterator for traversing an Array List.	Vector can use both Enumeration and Iterator for traversing.

Q9. What is the difference between equals() and == in Java?

Ans: Equals() method is defined in Object class in Java and used for checking equality of two objects defined by business logic.

"==" or equality operator in Java is a binary operator provided by Java programming language and used to compare primitives and objects. *public boolean equals(Object o)* is the method provided by the Object class. The default implementation uses == operator to compare two objects. For example: method can be overridden like String class. equals() method is used to compare the values of two objects.

Q10. What are the differences between Heap and Stack Memory in Java?

Ans: The major difference between Heap and Stack memory are:

Features	Stack	Heap
Memory	Stack memory is used only by one thread of execution.	Heap memory is used by all the parts of the application.
Access	Stack memory can't be accessed by other threads.	Objects stored in the heap are globally accessible.
Memory Management	Follows LIFO manner to free memory.	Memory management is based on the generation associated with each object.
Lifetime	Exists until the end of execution of the thread.	Heap memory lives from the start till the end of application execution.
Usage	Stack memory only contains local primitive and reference variables to objects in heap space.	Whenever an object is created, it's always stored in the Heap space.

Q11. What is a package in Java? List down various advantages of packages.

Ans: Packages in Java, are the collection of related classes and interfaces which are bundled together. By using packages, developers can easily modularize the code and optimize its reuse. Also, the code within the packages can be imported by other classes and reused. Below I have listed down a few of its advantages:

- Packages help in avoiding name clashes
- They provide easier access control on the code
- Packages can also contain hidden classes which are not visible to the outer classes and only used within the package
- Creates a proper hierarchical structure which makes it easier to locate the related classes

Q12. Why pointers are not used in Java?

Ans: Java doesn't use pointers because they are unsafe and increases the complexity of the program. Since, Java is known for its simplicity of code, adding the concept of pointers will be contradicting. Moreover, since JVM is responsible for implicit memory allocation, thus in order to avoid direct access to memory by the user, pointers are discouraged in Java.

Q13. What are access modifiers in Java?

Ans: In Java, access modifiers are special keywords which are used to restrict the access of a class, constructor, data member and method in another class. Java supports four types of access modifiers:

1. *Default*
2. *Private*
3. *Protected*
4. *Public*

Q14. Define a Java Class.

Ans: A class in Java is a blueprint which includes all your data. A class contains fields (variables) and methods to describe the behavior of an object. Let's have a look at the syntax of a class.

Q15. What is an object in Java and how is it created?

Ans: An object is a real-world entity that has a state and behavior. An object has three characteristics:

1. State
2. Behavior
3. Identity

An object is created using the 'new' keyword. For example:

ClassName obj = new ClassName();

Q16. What is Object Oriented Programming?

Ans: Object-oriented programming or popularly known as OOPs is a programming model or approach where the programs are organized around objects rather than logic and functions. In other words, OOP mainly focuses on the objects that are required to be manipulated instead of logic. This approach is ideal for the programs large and complex codes and needs to be actively updated or maintained.

Q17. What are the main concepts of OOPs in Java?

Ans: Object-Oriented Programming or OOPs is a programming style that is associated with concepts like:

1. *Inheritance:* Inheritance is a process where one class acquires the properties of another.
2. *Encapsulation:* Encapsulation in Java is a mechanism of wrapping up the data and code together as a single unit.
3. *Abstraction:* Abstraction is the methodology of hiding the implementation details from the user and only providing the functionality to the users.
4. *Polymorphism:* Polymorphism is the ability of a variable, function or object to take multiple forms.

Q18. What is the difference between a local variable and an instance variable?

Ans: In Java, a **local variable** is typically used inside a method, constructor, or a **block** and has only local scope. Thus, this variable can be used only within the scope of a block. The best benefit of having a local variable is that other methods in the class won't be even aware of that variable. Whereas, an **instance variable** in Java, is a variable which is bounded to its object itself. These variables are declared within a **class**, but outside a method. Every object of that class will create it's own copy of the variable while using it. Thus, any changes made to the variable won't reflect in any other instances of that class and will be bound to that particular instance only.

Q19. Differentiate between the constructors and methods in Java?

Methods	Constructors
1. Used to represent the behavior of an object	1. Used to initialize the state of an object
2. Must have a return type	2. Do not have any return type
3. Needs to be invoked explicitly	3. Is invoked implicitly
4. No default method is provided by the compiler	4. A default constructor is provided by the compiler if the class has none
5. Method name may or may not be same as class name	5. Constructor name must always be the same as the class name

Q20. What is the difference between break and continue statements?

break	continue
1. Can be used in switch and loop (for, while, do while) statements	1. Can be only used with loop statements
2. It causes the switch or loop statements to terminate the moment it is executed	2. It doesn't terminate the loop but causes the loop to jump to the next iteration
3. It terminates the innermost enclosing loop or switch immediately	3. A continue within a loop nested with a switch will cause the next loop iteration to execute

Q21. What is the difference between this() and super() in Java?

Ans: In Java, super() and this(), both are special keywords that are used to call the constructor.

this()	super()
1. this() represents the current instance of a class	1. super() represents the current instance of a parent/base class
2. Used to call the default constructor of the same class	2. Used to call the default constructor of the parent/base class
3. Used to access methods of the current class	3. Used to access methods of the base class
4. Used for pointing the current class instance	4. Used for pointing the superclass instance
5. Must be the first line of a block	5. Must be the first line of a block

Q22. Differentiate between static and non-static methods in Java.

Static Method	Non-Static Method
1. *The static* keyword must be used before the method name	1. No need to use the *static* keyword before the method name
2. It is called using the class (className.methodName)	2. It is can be called like any general method
3. They can't access any non-static instance variables or methods	3. It can access any static method and any static variable without creating an instance of the class

Q23. What is the difference between an array and an array list?

Array	ArrayList
Cannot contain values of different data types	Can contain values of different data types.
Size must be defined at the time of declaration	Size can be dynamically changed
Need to specify the index in order to add data	No need to specify the index
Arrays are not type parameterized	Arraylists are type
Arrays can contain primitive data types as well as objects	Arraylists can contain only objects, no primitive data types are allowed

Q24. What is Polymorphism?

Ans: Polymorphism is briefly described as "one interface, many implementations". Polymorphism is a characteristic of being able to assign a different meaning or usage to something in different contexts – specifically, to allow an entity such as a variable, a function, or an object to have more than one form. There are two types of polymorphism:

1. Compile time polymorphism
2. Run time polymorphism

Compile time polymorphism is method overloading whereas Runtime time polymorphism is done using inheritance and interface.

Q25. What is runtime polymorphism or dynamic method dispatch?

Ans: In Java, runtime polymorphism or dynamic method dispatch is a process in which a call to an overridden method is resolved at runtime rather than at compile-time. In this process, an overridden method is called through the reference variable of a superclass. Let's take a look at the example below to understand it better.

Q26. What is abstraction in Java?

Ans: Abstraction refers to the quality of dealing with ideas rather than events. It basically deals with hiding the details and showing the essential things to the user. Thus you can say that abstraction in Java is the process of hiding the implementation details from the user and revealing only the functionality to them. Abstraction can be achieved in two ways:

1. **Abstract Classes** (0-100% of abstraction can be achieved)
2. **Interfaces** (100% of abstraction can be achieved)

Q27. What do you mean by an interface in Java?

Ans: An interface in Java is a blueprint of a class or you can say it is a collection of abstract methods and static constants. In an interface, each method is public and abstract but it does not contain any constructor. Thus, interface basically is a group of related methods with empty bodies. Example:

Q28. What is the difference between abstract classes and interfaces?

Abstract Class	Interfaces
An abstract class can provide complete, default code and/or just the details that have to be overridden	An interface cannot provide any code at all, just the signature
In the case of an abstract class, a class may extend only one abstract class	A Class may implement several interfaces
An abstract class can have non-abstract methods	All methods of an Interface are abstract
An abstract class can have instance variables	An Interface cannot have instance variables
An abstract class can have any visibility: public, private, protected	An Interface visibility must be public (or) none
If we add a new method to an abstract class then we have the option of providing default implementation and therefore all the existing code might work properly	If we add a new method to an Interface then we have to track down all the implementations of the interface and define implementation for the new method
An abstract class can contain constructors	An Interface cannot contain constructors
Abstract classes are fast	Interfaces are slow as it requires extra indirection to find the corresponding method in the actual class

Q29. What is inheritance in Java?

Ans: Inheritance in Java is the concept where the properties of one class can be inherited by the other. It helps to reuse the code and establish a relationship between different classes. Inheritance is performed between two types of classes:

1. Parent class (Super or Base class)
2. Child class (Subclass or Derived class)

A class which inherits the properties is known as Child Class whereas a class whose properties are inherited is known as Parent class.

Q30. What are the different types of inheritance in Java?

Ans: Java supports four types of inheritance which are:

1. **Single Inheritance:** In single inheritance, one class inherits the properties of another i.e there will be only one parent as well as one child class.
2. **Multilevel Inheritance:** When a class is derived from a class which is also derived from another class, i.e. a class having more than one parent class but at different levels, such type of inheritance is called Multilevel Inheritance.
3. **Hierarchical Inheritance:** When a class has more than one child classes (subclasses) or in other words, more than one child classes have the same parent class, then such kind of inheritance is known as hierarchical.
4. **Hybrid Inheritance:** Hybrid inheritance is a combination of two *or more types* of inheritance.

Q31. What is method overloading and method overriding?

Ans: *Method Overloading :*

- In Method Overloading, Methods of the same class shares the same name but each method must have a different number of parameters or parameters having different types and order.
- Method Overloading is to "add" or "extend" more to the method's behavior.
- It is a compile-time polymorphism.
- The methods must have a different signature.
- It may or may not need inheritance in Method Overloading.

Method Overriding:

- In Method Overriding, the subclass has the same method with the same name and exactly the same number and type of parameters and same return type as a superclass.
- Method Overriding is to "Change" existing behavior of the method.
- It is a run time polymorphism.
- The methods must have the same signature.
- It always requires inheritance in Method Overriding.

Q32. What is multiple inheritance? Is it supported by Java?

Ans: If a child class inherits the property from multiple classes is known as multiple inheritance. Java does not allow to extend multiple classes.
The problem with multiple inheritance is that if multiple parent classes have the same method name, then at runtime it becomes difficult for the compiler to decide which method to execute from the child class.

Q33. What is encapsulation in Java?

Ans: Encapsulation is a mechanism where you bind your data(variables) and code(methods) together as a single unit. Here, the data is hidden from the outer world and can be accessed only via current class methods. This helps in protecting the data from any unnecessary modification. We can achieve encapsulation in Java by:

- Declaring the variables of a class as private.
- Providing public setter and getter methods to modify and view the values of the variables.

Q34. What is a copy constructor in Java?

Ans: Copy constructor is a member function that is used to initialize an object using another object of the same class. Though there is no need for copy constructor in Java since all objects are passed by reference. Moreover, Java does not even support automatic pass-by-value.

Q35. What is a constructor overloading in Java?

Ans: In Java, constructor overloading is a technique of adding any number of constructors to a class each having a different parameter list. The compiler uses the number of parameters and their types in the list to differentiate the overloaded constructors.

Q36. What is the difference between Error and Exception?

Ans: An error is an irrecoverable condition occurring at runtime. Such as OutOfMemory error. These JVM errors you cannot repair them at runtime. Though error can be caught in the catch block but the execution of application will come to a halt and is not recoverable.
While exceptions are conditions that occur because of bad input or human error etc. e.g. FileNotFoundException will be thrown if the specified file does not exist. Or a NullPointerException will take place if you try using a null reference. In most of the cases it is possible to recover from an exception (probably by giving the user feedback for entering proper values etc.

Q37. How can you handle Java exceptions?

Ans: There are five keywords used to handle exceptions in Java:

1. try
2. catch
3. finally
4. throw
5. throws

Q38. What purpose do the keywords final, finally, and finalize fulfill?

Ans: Final:
Final is used to apply restrictions on class, method, and variable. A final class can’t be inherited, final method can’t be overridden and final variable value can’t be changed.

Finally
Finally is used to place important code, it will be executed whether the exception is handled or not.

Finalize
Finalize is used to perform clean up processing just before the object is garbage collected.

Q39. What are the differences between throw and throws?

throw keyword	throws keyword
Throw is used to explicitly throw an exception.	Throws is used to declare an exception.
Checked exceptions can not be propagated with throw only.	Checked exception can be propagated with throws.
Throw is followed by an instance.	Throws is followed by class.
Throw is used within the method.	Throws is used with the method signature.
You cannot throw multiple exception	You can declare multiple exception e.g. public void method() IOException,SQLException.

Q40. What are the differences between processes and threads?

	Process	Thread
Definition	An executing instance of a program is called a process.	A thread is a subset of the process.
Communication	Processes must use inter-process communication to communicate with sibling processes.	Threads can directly communicate with other threa of its process.
Control	Processes can only exercise control over child processes.	Threads can exercise considerable control over threads of the same process.
Changes	Any change in the parent process does not affect child processes.	Any change in the main thread may affect the behavior of the other threads of the process.
Memory	Run in separate memory spaces.	Run in shared memory spaces.
Controlled by	Process is controlled by the operating system.	Threads are controlled by programmer in a progra
Dependence	Processes are independent.	Threads are dependent.

Q41. What is a finally block? Is there a case when finally will not execute?

Ans: Finally block is a block which always executes a set of statements. It is always associated with a try block regardless of any exception that occurs or not.
Yes, finally will not be executed if the program exits either by calling System.exit() or by causing a fatal error that causes the process to abort.

Q42. What are the important methods of Java Exception Class?

Ans: Methods are defined in the base class Throwable. Some of the important methods of Java exception class are stated below.

1. **String getMessage() –** This method returns the message String about the exception. The message can be provided through its constructor.
2. **public StackTraceElement[] getStackTrace() –** This method returns an array containing each element on the stack trace. The element at index 0 represents the top of the call stack whereas the last element in the array represents the method at the bottom of the call stack.
3. **Synchronized Throwable getCause()** – This method returns the cause of the exception or null id as represented by a Throwable object.
4. **String toString()** – This method returns the information in String format. The returned String contains the name of Throwable class and localized message.
5. **void printStackTrace()** – This method prints the stack trace information to the standard error stream.

Q43. What is OutOfMemoryError in Java?

Ans: OutOfMemoryError is the subclass of java.lang.Error which generally occurs when our JVM runs out of memory.

Q44. What is a Thread?

Ans: A thread is the smallest piece of programmed instructions which can be executed independently by a scheduler. In Java, all the programs will have at least one thread which is known as the main thread. This main thread is created by the JVM when the program starts its execution. The main thread is used to invoke the main() of the program.

Q45. What are the two ways to create a thread?

Ans: In Java, threads can be created in the following two ways:-

- By implementing the Runnable interface.
- By extending the Thread

Q46. What are the different types of garbage collectors in Java?

Ans: Garbage collection in Java a program which helps in implicit memory management. Since in Java, using the new keyword you can create objects dynamically, which once created will consume some memory. Once the job is done and there are no more references left to the object, Java using garbage collection destroys the object and relieves the memory occupied by it. Java provides four types of garbage collectors:

- Serial Garbage Collector
- Parallel Garbage Collector
- CMS Garbage Collector
- G1 Garbage Collector

DBMS INTERVIEW QUESTIONS:

Q1. What are the advantages of DBMS?

Ans: The advantages of DBMS are as follows:

- **Data Independence:** Allows to change the structure of the data without affecting the structure of any of the running application programs.
- **Sharing of Data:** Multiple users can use data from the same database simultaneously.
- **Integrity constraints:** These constraints allow the data to be stored in a database in a refined manner.
- **Redundancy control:** Supports a mechanism to control the redundancy of data by integrating all the data into a single database.
- **Provide backup and recovery facility:** Provides a feature of 'backup and recovery' to automatically create the data backup and restore the data as and when required.

Q2. Mention the different languages present in DBMS

Ans: The different languages present in DBMS are as follows:

- **Data Definition Language(DDL)** – Consists of commands which are used to define the database.
- **Data Manipulation Language(DML)** – Consists of commands which are used to manipulate the data present in the database.
- **Data Control Language(DCL)** – Consists of commands which deal with the user permissions and controls of the database system.
- **Transaction Control Language(TCL)** – Consist of commands which deal with the transaction of the database.

Q3. What are the different levels of abstraction in the DBMS?

Ans: There are three levels of data abstraction in DBMS. They are:

- **Physical Level:** It is the lowest level of abstraction and describes how the data is stored.
- **Logical Level:** This is the next level of abstraction after the Physical level. This layer determines what data is stored in the database, and what is the relationship between the data points.
- **View Level:** The View Level is the highest level of abstraction and it describes only a part of the entire database.

Q4. What is an entity-relationship model?

Ans: It is a diagrammatic approach to database design, where you represent real-world objects as entities and mention relationships between them. This approach helps the team of DBAs' to understand the schema easily.

Q5. What is concurrency control?

Ans: This is a process managing simultaneous operations in a database so that database integrity is not compromised. The following are the two approaches involved in concurrency control:

- **Optimistic approach** – Involves versioning
- **Pessimistic approach –** Involves locking

Q6. What are the ACID properties in DBMS?

Ans: ACID stands for Atomicity, Consistency, Isolation, Durability. It is used to ensure that the data transactions are processed reliably in a database system.

- **Atomicity:** Atomicity refers to those transactions which are completely successful or failed. Here each transaction refers to a single logical operation of a data. So, even if one part of any transaction fails, the entire transaction fails and the database state is left unchanged.
- **Consistency:** Consistency ensures that the data must meet all the validation rules. In simple words, you can say that your transaction never leaves the database without completing its state.
- **Isolation:** The main goal of isolation is concurrency control.
- **Durability:** Durability means that if a transaction has been committed, it will occur whatever may be the scenario.

Q7. What is normalization and what are the different types of normalization?

Ans: The process of organizing data to avoid any duplication of data and redundancy is known as Normalization. There are many successive levels of normalization which are known as **normal forms**. Each consecutive normal form depends on the previous one. The following are the first three normal forms. Apart from these, you have higher normal forms such as BCNF.

- **First Normal Form (1NF)** – No repeating groups within rows
- **Second Normal Form (2NF)** – Every non-key (supporting) column value is dependent on the whole primary key.
- **Third Normal Form (3NF)** – Dependent solely on the primary key and no other non-key (supporting) column value.

Q8. What are the different types of keys in the database?

Ans: There are mainly 4 types of Keys, that can be considered in a database. I am going to consider the below tables to explain to you the various keys.

- **Candidate Key –** This is a set of attributes which can uniquely identify a table. Each table can have more than a candidate key. Apart from this, out of all the candidate keys, one key can be chosen as the Primary key. In the above example, since CustomerID and PanNumber can uniquely identify every tuple, they would be considered as a Candidate Key.
- **Super Key –** This is a set of attributes which can uniquely identify a tuple. So, a candidate key, primary key, and a unique key is a superkey, but vice-versa isn't true.
- **Primary Key –** This is a set of attributes which are used to uniquely identify every tuple. In the above example, since CustomerID and PanNumber are candidate keys, any one of them can be chosen as a Primary Key. Here CustomerID is chosen as the primary key.
- **Foreign Key –** An attribute that can only take the values present as the values of some other attribute, is the foreign key to the attribute to which it refers. in the above example, the CustomerID from the Customers Table is referred to the CustomerID from the Customer_Payment Table.

Q9. What do you understand by functional dependency and transitive dependency in DBMS?

Ans: Functional Dependency: A functional dependency is a constraint which is used in describing the relationship among different attributes in a relation.
Example: Consider a relation "A1" having attributes X and Y. The functional dependency among these two attributes will be **X -> Y,** this implies that Y is functionally dependent on X.

Transitive Dependency: A transitive dependency is a constraint which can only occur in a relation of three or more attributes.

Q10. What are indexes? Mention the differences between the clustered and non-clustered index

Ans: Indexes are data structures responsible for improving the speed of data retrieval operations on a table. This data structure uses more storage space to maintain extra copies of data by using additional writes. So, indexes are mainly used for searching algorithms, where you wish to retrieve data in a quick manner.
The differences between clustered and non-clustered index are as follows:

Clustered Index	Non-clustered Index
A clustered index is faster	Non clustered index is relatively slower
Alters the way records are stored in a database as it sorts out rows by the column which is set to be clustered index	Does not alter the way it was stored but it creates a separate object within a table which points back to the original table rows after searching
One table can only have one clustered index	One table can only have many non clustered indexes

Q11. What do you understand by Data Independence?

Ans: When you say an application has data independence, it implies that the application is independent of the storage structure and data access strategies of data.
Q30. What are the different integrity rules present in the DBMS?
The different integrity rules present in DBMS are as follows:

- **Entity Integrity:** This rule states that the value of the primary key can never be NULL. So, all the tuples in the column identified as the primary key should have a value.
- **Referential Integrity:** This rule states that either the value of the foreign key is NULL or it should be the primary key of any other relation.

Q12. Explain what is a deadlock and mention how it can be resolved?

Ans: Deadlock is a situation which occurs when two transactions wait on a resource which is locked or other transaction holds. Deadlocks can be prevented by making all the transactions acquire all the locks at the same instance of time. So, once deadlock occurs, the only way to cure is to abort one of the transactions and remove the partially completed work.

Q13. What are the differences between DROP, TRUNCATE and DELETE commands?

DROP	TRUNCATE	DELETE
Used to delete a database, table or a view	Used to delete all rows from a table	Used to delete a row in the table
Data cannot be rollbacked	Data cannot be rollbacked	Data can be rollbacked
A DDL command	A DDL command	A DML command.
Slower than TRUNCATE	Faster than DROP and DELETE	Slower than TRUNCATE
Deletes the full structure of the table	Preserves the structure of the table	Deletes the structure of the row from a table

Q14. What are joins in SQL and what are the different types of joins?

Ans: A JOIN clause is used to combine rows from two or more tables, based on a related column between them. It is used to merge two tables or retrieve data from there. There are 4 joins in SQL namely:

- Inner Join
- Right Join
- Left Join
- Full Join