SOFTWARE ENGINEERING

# INTERVIEW QUESTIONS WITH ANSWERS

# Interview Questions with Answers

***Note**: These questions are based on my past experiences and research*

# Chapter 1. Object Oriented Programming

## Basic Level:

### 1. What is Object Oriented Programming?

Object-Oriented Programming(OOPs) is a type of programming that is based on objects rather than just functions and procedures. Individual objects are grouped into classes. OOPs implement real-world entities like inheritance, polymorphism, hiding, etc into programming. It also allows binding data and code together.

### 2. Why use OOPs? OR Advantages of OOPs

- OOPs allows clarity in programming thereby allowing simplicity in solving complex problems
- Code can be reused through inheritance thereby reducing redundancy
- Data and code are bound together by encapsulation
- OOPs allows data hiding, therefore, private data is kept confidential
- Problems can be divided into different parts making it simple to solve
- The concept of polymorphism gives flexibility to the program by allowing the entities to have multiple forms

### 3. What are the main features/pillars of OOPs?

- Inheritance
- Encapsulation
- Polymorphism
- Data Abstraction

### 4. What are the main features of OOPs?

- Inheritance
- Encapsulation
- Polymorphism
- Data Abstraction

## 5. What is a class?

A class is a prototype that consists of objects in different states and with different behaviors. It has a number of methods that are common to the objects present within that class.

## 6. What is an object?

An object is a real-world entity which is the basic unit of OOPs for example chair, cat, dog, etc. Different objects have different states or attributes, and behaviors.

**Note**: Class is a concept and to map this concept in real life is an object.

## 7. What is the difference between a class and a structure?

**Class:** User-defined blueprint from which objects are created. It consists of methods or sets of instructions that are to be performed on the objects.

**Structure:** A structure is basically a user-defined collection of variables which are of different data types.

## 8. Can you call the base class method without creating an instance?

Yes, you can call the base class without instantiating it if:

- It is a static method
- The base class is inherited by some other subclass

| Object | Class |
|---|---|
| A real-world entity which is an instance of a class | A class is basically a template or a blueprint within which objects can be created |
| An object acts like a variable of the class | Binds methods and data together into a single unit |
| An object is a physical entity | A class is a logical entity |
| Objects take memory space when they are created | A class does not take memory space when created |
| Objects can be declared as and when required | Classes are declared just once |

### 9. What is inheritance?

Inheritance is a feature of OOPs which allows classes to inherit common properties from other classes. For example, if there is a class such as ‘vehicle’, other classes like ‘car’, ‘bike’, etc can inherit common properties from the vehicle class. This property helps you get rid of redundant code thereby reducing the overall size of the code.

**9. What are the different types of inheritance?**

- Single inheritance
- Multiple inheritance
- Multilevel inheritance
- Hierarchical inheritance
- Hybrid inheritance

**10. What is the difference between multiple and multilevel inheritance?**

| Multiple Inheritance | Multilevel Inheritance |
|---|---|
| Multiple inheritance comes into picture when a class inherits more than one base class | Multilevel inheritance means a class inherits from another class which itself is a subclass of some other base class |
| Example: A class defining a child inherits from two base classes Mother and Father | Example: A class describing a sports car will inherit from a base class Car which inturn inherits another class Vehicle |

**.11. What is hierarchical inheritance?**

Hierarchical inheritance refers to inheritance where one base class has more than one subclass. For example, the vehicle class can have 'car', 'bike', etc as its subclasses.

**12. What are the limitations of inheritance?**

- Increases the time and effort required to execute a program as it requires jumping back and forth between different classes
- The parent class and the child class get tightly coupled
- Any modifications to the program would require changes both in the parent as well as the child class
- Needs careful implementation else would lead to incorrect results.

**13. What is a superclass?**

A superclass or base class is a class that acts as a parent to some other class or classes. For example, the Vehicle class is a superclass of class Car.

**14. What is a subclass?**

A class that inherits from another class is called the subclass. For example, the class Car is a subclass or a derived of Vehicle class.

**15. What is polymorphism?**

Polymorphism refers to the ability to exist in multiple forms. Multiple definitions can be given to a single interface. For example, if you have a class named Vehicle, it can have a method named

speed but you cannot define it because different vehicles have different speeds. This method will be defined in the subclasses with different definitions for different vehicles.

**16. What is static polymorphism?**

Static polymorphism (static binding) is a kind of polymorphism that occurs at compile time. An example of compile-time polymorphism is method overloading.

**17. What is dynamic polymorphism?**

Runtime polymorphism or dynamic polymorphism (dynamic binding) is a type of polymorphism which is resolved during runtime. An example of runtime polymorphism is method overriding.

**18. What is function/method overloading?**

Method overloading is a feature of OOPs which makes it possible to give the same name to more than one method within a class if the arguments passed differ.

**19. What is function/method overriding?**

Method overriding is a feature of OOPs by which the child class or the subclass can redefine methods present in the base class or parent class. Here, the method that is overridden has the same name as well as the signature meaning the arguments passed and the return type.

**20. What is operator overloading?**

Operator overloading refers to implementing operators using user-defined types based on the arguments passed along with it.

**21. Differentiate between overloading and overriding.**

| Overloading | Overriding |
|---|---|
| Two or more methods having the same name but different parameters or signature | Child class redefining methods present in the base class with the same parameters/ signature |
| Resolved during compile-time | Resolved during runtime |

**22. What is encapsulation?**

Encapsulation refers to binding the data and the code that works on that together in a single unit. For example, a class. Encapsulation also allows data-hiding as the data specified in one class is hidden from other classes.

**23. What is data abstraction?**

Data abstraction is a very important feature of OOPs that allows displaying only the important information and hiding the implementation details. For example, while riding a bike, you know that if you raise the accelerator, the speed will increase, but you don't know how it actually happens. This is data abstraction as the implementation details are hidden from the rider.

**24. How to achieve data abstraction?**

Data abstraction can be achieved through:

- Abstract class
- Abstract method

**25. What is an abstract class?**

An abstract class is a class that consists of abstract methods. These methods are basically declared but not defined. If these methods are to be used in some subclass, they need to be exclusively defined in the subclass.

**26. Can you create an instance of an abstract class?**

No. Instances of an abstract class cannot be created because it does not have a complete implementation. However, instances of subclass inheriting the abstract class can be created.

**27. What is an interface?**

It is a concept of OOPs that allows you to declare methods without defining them. Interfaces, unlike classes, are not blueprints because they do not contain detailed instructions or actions to be performed. Any class that implements an interface defines the methods of the interface.

**28. Differentiate between data abstraction and encapsulation.**

| Data abstraction | Encapsulation |
|---|---|
| Solves the problem at the design level | Solves the problem at the implementation level |
| Allows showing important aspects while hiding implementation details | Binds code and data together into a single unit and hides it from the world |

**29. What are virtual functions?**

Virtual functions are functions that are present in the parent class and are overridden by the subclass. These functions are used to achieve runtime polymorphism.

**30. What are pure virtual functions?**

Pure virtual functions or abstract functions are functions that are only declared in the base class. This means that they do not contain any definition in the base class and need to be redefined in the subclass.

**31. What is a constructor?**

A constructor is a special type of method that has the same name as the class and is used to initialize objects of that class.

**32. What is a destructor?**

A destructor is a method that is automatically invoked when an object is destroyed. The destructor also recovers the heap space that was allocated to the destroyed object, closes the files and database connections of the object, etc.

**33. Types of constructors**

Types of constructors differ from language to language. However, all the possible constructors are:

- Default constructor
- Parameterized constructor
- Copy constructor
- Static constructor
- Private constructor

**34. What is a copy constructor?**

A copy constructor creates objects by copying variables from another object of the same class. The main aim of a copy constructor is to create a new object from an existing one.

## 35. What is an exception?

An exception is a kind of notification that interrupts the normal execution of a program. Exceptions provide a pattern to the error and transfer the error to the exception handler to resolve it. The state of the program is saved as soon as an exception is raised.

## 36. What is exception handling?

Exception handling in Object-Oriented Programming is a very important concept that is used to manage errors. An exception handler allows errors to be thrown and caught and implements a centralized mechanism to resolve them.

## 37. What is the difference between an error and an exception?

| Error | Exception |
| --- | --- |
| Errors are problems that should not be encountered by applications | Conditions that an application might try to catch |

### 38. What is a try/ catch block?

A try/ catch block is used to handle exceptions. The try block defines a set of statements that may lead to an error. The catch block basically catches the exception.

### 39. What are the limitations of OOPs?

- Usually not suitable for small problems
- Requires intensive testing
- Takes more time to solve the problem
- Requires proper planning
- The programmer should think of solving a problem in terms of objects.

#### 40.. What are 'access specifiers'?

Access specifiers or access modifiers are keywords that determine the accessibility of methods, classes, etc in OOPs. These access specifiers allow the implementation of encapsulation. The most common access specifiers are public, private and protected. However, there are a few more which are specific to the programming languages.

# Chapter 2. Linked Lists

#### 1. Mention what Linked lists?

A linked list is a data structure that can store a collection of items. In other words, linked lists can be utilized to store several objects of the same type. Each unit or element of the list is referred to as a node. Each node has its own data and the address of the next node. It is like a chain. Linked Lists are used to create graphs and trees.

#### 2. What type of memory allocation is referred to for Linked lists?

Dynamic memory allocation is referred for Linked lists.

#### 3. Mention what is traversal in linked lists?

Term Traversal is used to refer to the operation of processing each element in the list.

**4. Describe what is Node in the link list? And name the types of Linked Lists?**

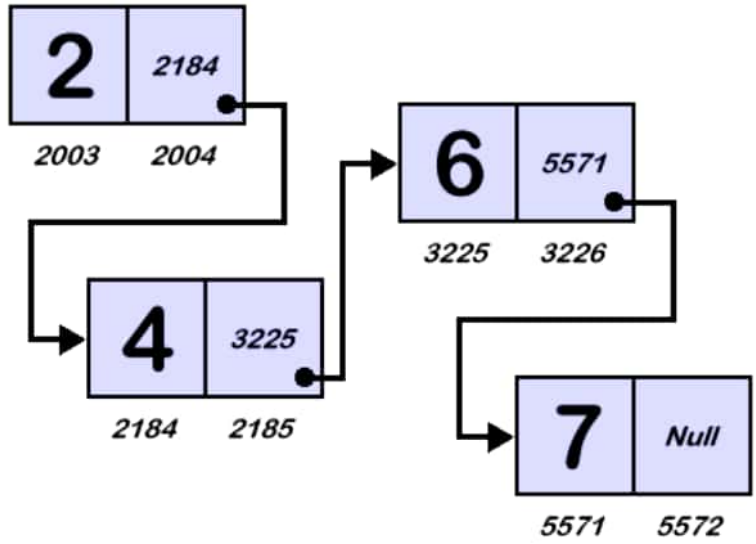
Together (data + link) is referred to as the Node.

Types of Linked Lists are,

- Singly Linked List
- Doubly Linked List
- Multiply Linked List
- Circular Linked List

**5. What is a Singly Linked list?**

Singly Linked lists are a type of data structure. In a singly linked list, each node in the list stores the contents of the node and a reference or pointer to the next node in the list. It does not store any reference or pointer to the previous node.




**6. What is the difference between Linear Array and Linked List?**

| Arrays | Linked List |
|---|---|
| Deletion and Insertions are difficult. | Deletion and Insertions can be done easily. |
| For insertion and deletion, it needs movements | or insertion and deletion, it does not require movement of nodes |
| In it space is wasted | In it space is not wasted |
| It is expensive | It is not expensive |

| | |
|---|---|
| It cannot be reduced or extended according to requirements | It can be reduced or extended according to requirements |
| To avail each element the same amount of time is required. | To avail each element a different amount of time is required. |
| In consecutive memory locations elements are stored. | Elements may or may not be stored in consecutive memory locations |
| We can reach there directly if we have to go to a particular element | To reach a particular node, you need to go through all those nodes that come before that node. |

**7. Mention what are the applications of Linked Lists?**

Applications of Linked Lists are,

- Linked lists are used to implement queues, stacks, graphs, etc.
- In Linked Lists you don't need to know the size in advance.
- Linked lists let you insert elements at the beginning and end of the list.

**8. Mention what is the difference between singly and doubly linked lists?**

A doubly linked list nodes contain three fields:

- An integer value and
- Two links to other nodes
- one to point to the previous node and
- other to point to the next node.

Whereas a singly linked list contains points only to the next node.

**9. Mention what are the applications that use Linked lists?**

Both queues and stacks are often implemented using linked lists. Other applications are list, binary tree, skip, unrolled linked list, hash table, etc.

**10. Mention what is the biggest advantage of linked lists?**

The biggest benefit of linked lists is that you do not specify a fixed size for your list. The more elements you add to the chain, the bigger the chain gets.

# Chapter 3. Database

**1. What do you understand about 'Database'?**

Database is an organized collection of related data where the data is stored and organized to serve some specific purpose.

**2. Define DBMS.**

DBMS stands for Database Management System. It is a collection of application programs which allow the user to organize, restore and retrieve information about data efficiently and as effectively as possible.

Some of the popular DBMS's are MySql, Oracle, Sybase, etc.

**3. Enlist the advantages of DBMS.**

- Data is stored in a structured way and hence redundancy is controlled.
- Validates the data entered and provides restrictions on unauthorized access to the database.
- Provides backup and recovery of the data when required.
- It provides multiple user interfaces.

**4. What do you understand about Data Redundancy?**

Duplication of data in the database is known as data redundancy. As a result of data redundancy, duplicated data is present at multiple locations, hence it leads to wastage of the storage space and the integrity of the database is destroyed.

**5. What are the various types of relationships in Database? Define them.**

- **One-to-one:** One table has a relationship with another table having a similar kind of column. Each primary key relates to only one or no record in the related table.
- **One-to-many:** One table has a relationship with another table that has primary and foreign key relations. The primary key table contains only one record that relates to none, one or many records in the related table.
- **Many-to-many:** Each record in both the tables can relate to many numbers of records in another table.

**6. Explain Normalization and De-Normalization.**

**Normalization** is the process of removing redundant data from the database by splitting the table in a well-defined manner in order to maintain data integrity. This process saves much of the storage space.

**De-normalization** is the process of adding up redundant data on the table in order to speed up the complex queries and thus achieve better performance.

**7. What are the different types of Normalization?**

- **First Normal Form (1NF):** A relation is said to be in 1NF only when all the entities of the table contain unique or atomic values.
- **Second Normal Form (2NF):** A relation is said to be in 2NF only if it is in 1NF and all the non-key attributes of the table are fully dependent on the primary key.
- **Third Normal Form (3NF):** A relation is said to be in 3NF only if it is in 2NF and every non-key attribute of the table is not transitively dependent on the primary key.

**8. What is SQL?**

Structured Query language, SQL is an ANSI(American National Standard Institute) standard programming language that is designed specifically for storing and managing the data in the relational database management system (RDBMS) using all kinds of data operations.

**9. How many SQL statements are used? Define them.**

SQL statements are basically divided into three categories, DDL, DML, and DCL.

- **Data Definition Language (DDL)** commands are used to define the structure that holds the data. These commands are auto-committed i.e. changes done by the DDL commands on the database are saved permanently.
- **Data Manipulation Language (DML)** commands are used to manipulate the data of the database. These commands are not auto-committed and can be rolled back.
- **Data Control Language (DCL)** commands are used to control the visibility of the data in the database like revoke access permission for using data in the database.

**10. Enlist the advantages of SQL.**

- Simple SQL queries can be used to retrieve a large amount of data from the database very quickly and efficiently.
- SQL is easy to learn and almost every DBMS supports SQL.
- It is easier to manage the database using SQL as no large amount of coding is required.

**11. Enlist some commands of DDL, DML, and DCL.**

**Data Definition Language (DDL) commands:**

- CREATE to create a new table or database.
- ALTER for alteration.
- TRUNCATE to delete data from the table.
- DROP to drop a table.
- RENAME to rename a table.

**Data Manipulation Language (DML) commands:**

- INSERT to insert a new row.
- UPDATE to update an existing row.
- DELETE to delete a row.
- MERGE for merging two rows or two tables.

**Data Control Language (DCL) commands:**

- COMMIT to permanently save.
- ROLLBACK to undo the change.
- SAVEPOINT to save temporarily.

**12. Explain the terms 'Record', 'Field' and 'Table' in terms of database.**

**Record:** Record is a collection of values or fields of a specific entity. **For Example,** An employee, Salary account, etc.

**Field:** A field refers to an area within a record that is reserved for specific data. **For Example,** Employee ID.

**Table:** Table is the collection of records of specific types. **For Example,** the Employee table is a collection of records related to all the employees.

**13. What are the advantages and disadvantages of views in the database?**

View is a virtual table that does not have its data on its own rather the data is defined from one or more underlying base tables.

**Advantages of Views:**

- As there is no physical location where the data in the view is stored, it generates output without wasting resources.
- Data access is restricted as it does not allow commands like insertion, updation, and deletion.

**Disadvantages of Views:**

- The view becomes irrelevant if we drop a table related to that view.
- Much memory space is occupied when the view is created for large tables.

**14. What do you understand about Functional dependency?**

A relation is said to be in functional dependency when one attribute uniquely defines another attribute.

**For Example,** R is a Relation, X and Y are two attributes. T1 and T2 are two tuples. Then,

T1[X]=T2[X] and T1[Y]=T2[Y]

Means, the value of component X uniquely defines the value of component Y.

Also, X->Y means Y is functionally dependent on X.

**15. When is functional dependency said to be fully functional dependent?**

To fulfill the criteria of fully functional dependency, the relation must meet the requirement of functional dependency.

A functional dependency 'A' and 'B' are said to be fully functional dependent when removal of any attribute say 'X' from 'A' means the dependency does not hold anymore.

**16. What do you understand by the E-R model?**

E-R model is an Entity-Relationship model which defines the conceptual view of the database.

The E-R model basically shows the real-world entities and their association/relations. Entities here represent the set of attributes in the database.

**17. Define Entity, Entity type, and Entity set.**

**Entity** can be anything, be it a place, class or object which has an independent existence in the real world.

**Entity Type** represents a set of entities that have similar attributes.

**Entity Set** in the database represents a collection of entities having a particular entity type.

**18. Define a Weak Entity set.**

Weak Entity set is the one whose primary key comprises its partial key as well as the primary key of its parent entity. This is the case because the entity set may not have sufficient attributes to form a primary key.

**19. Explain the terms 'Attribute' and 'Relations'**

**Attribute** is described as the properties or characteristics of an entity. **For Example**, Employee ID, Employee Name, Age, etc., can be attributes of the entity Employee.

**Relation** is a two-dimensional table containing a number of rows and columns where every row represents a record of the relation. Here, rows are also known as 'Tuples' and columns are known as 'Attributes'.

**20. What is the Database transaction?**

Sequence of operation performed which changes the consistent state of the database to another is known as the database transaction. After the completion of the transaction, either the successful completion is reflected in the system or the transaction fails and no change is reflected.

**21. What do you understand by Join?**

Join is the process of deriving the relationship between different tables by combining columns from one or more tables having common values in each. When a table joins with itself, it is known as Self Join.

**22. What are the disadvantages of a Query?**

- Indexes are not present.
- Stored procedures are excessively compiled.
- Difficulty in interfacing.

**23. Define Join types.**

**Answer:** Given below are the types of Join, which are explained with respect to the tables as an **Example.**

**employee table:**

| EmpID | EmpName |
|---|---|
| 1000 | Rohan |
| 1001 | Shruti |
| 1002 | Nikhil |
| 1003 | Naveen |

**employee_info table:**

| EmpID | Address |
|---|---|
| 1000 | Delhi |
| 1001 | Mumbai |
| 1002 | Delhi |
| 1003 | Kolkata |

**a) Inner JOIN:** Inner JOIN is also known as a simple JOIN. This SQL query returns results from both the tables having a common value in rows.

**SQL Query:**

SELECT * from employee, employee_info WHERE employee.EmpID = employee_info.EmpID ;

**Result:**

| EmpID | EmpName | EmpID | Address |
|---|---|---|---|
| 1000 | Rohan | 1000 | Delhi |
| 1001 | Shruti | 1000 | Delhi |
| 1002 | Nikhil | 1000 | Delhi |
| 1003 | Naveen | 1000 | Delhi |
| 1000 | Rohan | 1001 | Mumbai |
| 1001 | Shruti | 1001 | Mumbai |
| 1002 | Nikhil | 1001 | Mumbai |
| 1003 | Naveen | 1001 | Mumbai |
| 1000 | Rohan | 1002 | Delhi |
| 1001 | Shruti | 1002 | Delhi |
| 1002 | Nikhil | 1002 | Delhi |
| 1003 | Naveen | 1002 | Delhi |
| 1000 | Rohan | 1003 | Kolkata |
| 1001 | Shruti | 1003 | Kolkata |
| 1002 | Nikhil | 1003 | Kolkata |
| 1003 | Naveen | 1003 | Kolkata |

**b) Natural JOIN:** This is a type of Inner JOIN that returns results from both the tables having the same data values in the columns of both the tables to be joined.

**SQL Query:**

SELECT * from employee NATURAL JOIN employee_info;

**Result:**

| EmpID | EmpName | Address |
|---|---|---|
| 1000 | Rohan | Delhi |
| 1001 | Shruti | Mumbai |
| 1002 | Nikhil | Delhi |
| 1003 | Naveen | Kolkata |

**c) Cross JOIN:** Cross JOIN returns the result as all the records where each row from the first table is combined with each row of the second table.

**SQL Query:**

SELECT * from employee CROSS JOIN employee_info;

**Result:**

**Let us do some modification in the above tables to understand Right JOIN, Left JOIN, and Full JOIN.**

**employee table:**

| EmpID | EmpName |
|---|---|
| 1000 | Rohan |
| 1001 | Shruti |
| 1002 | Nikhil |
| 1003 | Naveen |
| 1004 | Shikha |
| 1005 | Shalu |

**employee_info table:**

| EmpID | Address |
|---|---|
| 1000 | Delhi |
| 1001 | Mumbai |
| 1002 | Gurgaon |
| 1003 | Kolkata |
| 1006 | Noida |
| 1007 | Kerala |

**a) Right JOIN:** Right JOIN is also known as Right Outer JOIN. This returns all the rows as a result from the right table even if the JOIN condition does not match any records in the left table.

**SQL Query:**

SELECT * from employee RIGHT OUTER JOIN employee_info on (employee.EmpID = employee_info.EmpID);

**Result:**

| EmpID | EmpName | EmpID | Address |
|---|---|---|---|
| 1000 | Rohan | 1000 | Delhi |
| 1001 | Shruti | 1001 | Mumbai |
| 1002 | Nikhil | 1002 | Delhi |
| 1003 | Naveen | 1003 | Kolkata |
| Null | Null | 1006 | Noida |
| Null | Null | 1007 | Kerala |

**b) Left JOIN:** Left JOIN is also known as Left Outer JOIN. This returns all the rows as a result of the left table even if the JOIN condition does not match any records in the right table. This is exactly the opposite of Right JOIN.

**SQL Query:**

SELECT * from employee LEFT OUTER JOIN employee_info on (employee.EmpID = employee_info.EmpID);

**Result:**

| EmpID | EmpName | EmpID | Address |
|---|---|---|---|
| 1000 | Rohan | 1000 | Delhi |
| 1001 | Shruti | 1001 | Mumbai |
| 1002 | Nikhil | 1002 | Delhi |
| 1003 | Naveen | 1003 | Kolkata |
| 1004 | Shikha | null | Null |
| 1005 | Shalu | null | Null |

**c) Outer/Full JOIN:** Full JOIN return results in combining the result of both the Left JOIN and Right JOIN.

**SQL Query:**

SELECT * from employee FULL OUTER JOIN employee_info on (employee.EmpID = employee_info.EmpID);

**Result:**

| EmpID | EmpName | EmpID | Address |
|---|---|---|---|
| 1000 | Rohan | 1000 | Delhi |
| 1001 | Shruti | 1001 | Mumbai |
| 1002 | Nikhil | 1002 | Delhi |
| 1003 | Naveen | 1003 | Kolkata |
| 1004 | Shikha | null | Null |
| 1005 | Shalu | null | Null |
| Null | Null | 1006 | Noida |
| Null | Null | 1007 | Kerala |

**24. Explain the Data Dictionary.**

Data dictionary is a set of information describing the content and structure of the tables and database objects. The job of the information stored in the data dictionary is to control, manipulate and access the relationship between database elements.

**25. Explain the Primary Key and Composite Key.**

Primary Key is that column of the table whose every row data is uniquely identified. Every row in the table must have a primary key and no two rows can have the same primary key. Primary key value can never be null nor can it be modified or updated.

Composite Key is a form of the candidate key where a set of columns will uniquely identify every row in the table.

**26. What do you understand about the Unique key?**

A Unique key is the same as the primary key whose every row data is uniquely identified with a difference of null value i.e. Unique key allows one value as a NULL value.

**27. Name the different data models that are available for database systems.**

**Different data models are:**

- Relational model

- Network model
- Hierarchical model

**28. Differentiate between 'DELETE', 'TRUNCATE' and 'DROP' commands.**

After the execution of **'DELETE'** operation, COMMIT and ROLLBACK statements can be performed to retrieve the lost data.

After the execution of **'TRUNCATE'** operation, COMMIT, and ROLLBACK statements cannot be performed to retrieve the lost data.

**'DROP'** command is used to drop the table or key like the primary key/foreign key

# Chapter 4. Problem Solving

1. Reverse 2 or more consecutive odd numbers containing nodes in a linked list.
2. Find 2nd maximum using 1 loop( O(N)) in an array.
3. Find the largest window sum in an array.
4. Find the largest window sum in an array.
5. Find a reflection BST tree.
6. Find whether a linked list is circular or not.
7. Detect the cycle in a linked list.
8. Find the first non repeated element in a string.
9. Write Down the recursive code for fibonacci series.
10. How to traverse Bubble Sort and Selection Sort?
11. Write down the code for merge sort.
12. Delete a node of linked list without knowing the head of it.
13. You are given a singly linked list e.g 1->2->3->4->5. Suppose that your head is pointing on 4. You have to delete the 5fth Node. Note: You cannot move the pointer backwards.
14. You have two unsorted arrays. You have to merge two arrays in the sorted manner. (Need Best Solution).
15. Check if the given string is palindrome or not?
16. You are given an array. Find the pair of numbers whose sum is equal to 5.
17. You are given a linked list with a head pointer only. Your task is to add a new node to the linked list at the end. (Note: You have only one pointer pointing at head)

    E.g 1->2->3->4->5

    New node will be appended after 5.

# Chapter 5. Some Interview Questions & Tips

**1**. Interviews usually start with the question Tell us something about yourself. This is the very beginning of your interview where you have to be very precise about your answer because interviewers are expecting many things from you out of this question.
You answer should reflect:

- Your Information
- Your past education
- Your FYP (optional)
- Your past industrial experience

If you don't have any industrial experience then mention your FYP and self projects and learnings.

**2.** Try to get to know about the company for which you are applying and before appearing in the interview. Get to know about them by visiting their website or linkedIn profiles. This will help you to answer the question "*What do you know about us?*"

**3.** Why do you want to work here?

**4.** Be confident. Confidence is the key to pass an interview.

**5.** Stick to the point. Try to avoid narrating stories rather than actual answers.

**6.** Don't hesitate to tell interviews you don't know the answer. Just be formal saying "I am not sure about the answer for this question but I'll go through it later".

**7.** Interviewers Usually end the interview by asking you "Do you have any questions?". Try to show them your interest. Come up with the questions and feel free to ask them.

# Thank you

ADARSH CHETAN