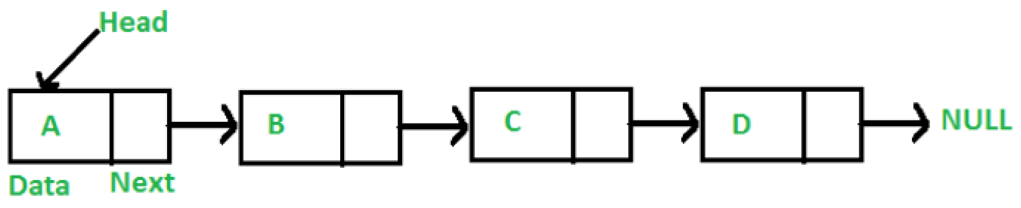
In computer science, a **linked list** is a linear collection of data elements, whose order is not given by their physical placement in memory. Instead, each element points to the next:



The elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers.

In simple words, a linked list consists of nodes where each node contains a data field and a reference(link) to the next node in the list.

A linked list is a data structure consisting of a collection of nodes which together represent a sequence.

In its most basic form, each node contains:

- data
- reference (in other words, a link) to the next node in the sequence.

The structure of linked list allows for efficient insertion or removal of elements from any position in the sequence during iteration. More complex variants add additional links, allowing more efficient insertion or removal of nodes at arbitrary positions.

The principal benefit of a linked list over a conventional array is that the list elements can be easily inserted or removed without reallocation or reorganization of the entire structure because the data items need not be stored contiguously in memory or on disk, while restructuring an array at run-time is a much more expensive operation.

Linked lists allow insertion and removal of nodes at any point in the list, and allow doing so with a constant number of operations by keeping the link previous to the link being added or removed in memory during list traversal.

On the other hand, since simple linked lists by themselves do not allow random access to the data or any form of efficient indexing, many basic operations—such as obtaining the last node of the list, finding a node that contains a given datum, or locating the place where a new node should be inserted—may require iterating through most or all of the list elements.

Linked list are dynamic, so the length of list can increase or decrease as necessary. Each node does not necessarily follow the previous one physically in the memory.

A drawback of linked lists is that access time is linear (and difficult to pipeline). Faster access, such as random access, is not feasible. Arrays have better cache locality compared to linked lists.

Linked lists require more memory than arrays because of the storage used by their pointers.

Nodes in a linked list must be read in order from the beginning as linked lists are inherently sequential access.

Nodes are stored incontiguously, greatly increasing the time periods required to access individual elements within the list, especially with a CPU cache.

Difficulties arise in linked lists when it comes to reverse traversing. For instance, singly linked lists are cumbersome to navigate backwards and while doubly-linked lists are somewhat easier to read, memory is consumed in allocating space for a back-pointer.