Header Linked List is a modified version of Singly Linked List. In Header linked list, we have a special node, the **Header Node** present at the beginning of the linked list. The Header Node is an extra node at the front of the list storing meaningful information about the list. Such a node is not similar in structure to the other nodes in the list. It does not represent any items of the list like other nodes rather than the information present in the Header node is global for all nodes such as **Count of Nodes in a List**, **Maximum** among all Items, **Minimum** value among all Items etc. This gives useful information about the Linked list.

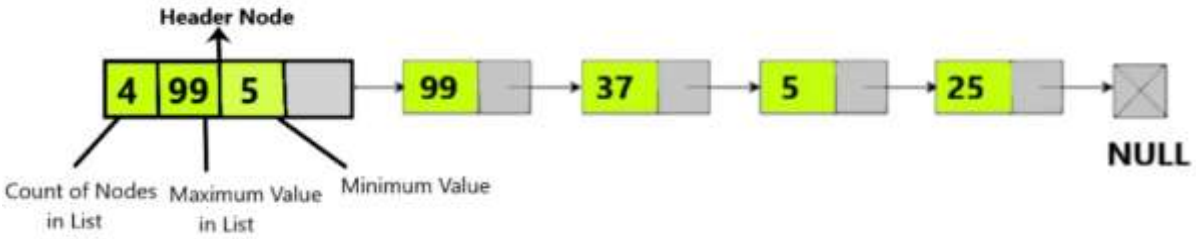
Now, Header Linked List is of two types:

Grounded Header Linked List

Circular Header Linked List.

Grounded Header Linked List

In this type of Header Linked List, the last node of the list points to NULL or holds the reference to NULL Pointer. The head pointer points to the Header node of the list. If the there is no node to the next of head pointer or head.next equals NULL then we know that the Linked List is empty. The operations performed on the Header Linked List are same as Singly Linked List such as Insertion, Deletion, and Traversal of nodes. Let us understand this with an example:

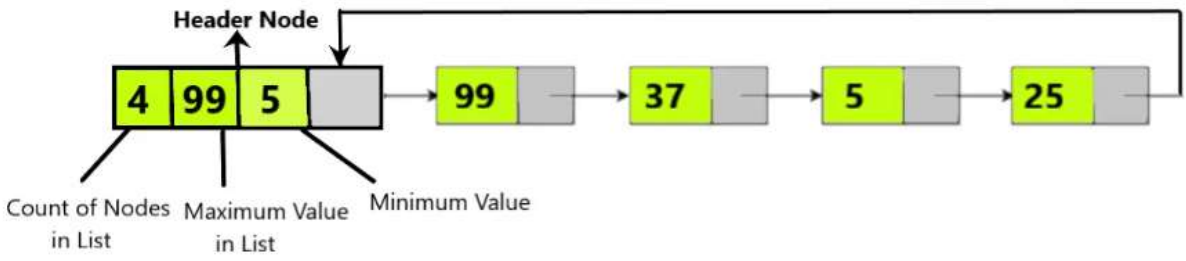


The above image shows a Grounded Header Linked List. We can see the Header Node at the beginning followed by the actual nodes of the list where the last node points to NULL. The header node maintains global information about the list. Firstly, we have to create Header Node first whose data field will be NULL or 0 initially. After that we

can perform any operations on that linked list and store the global information about the linked list on the header node. Here, we store the count or Total no. of nodes present in the Linked List, the Maximum and Minimum value among all nodes in the list. So, in the above Linked List Total Nodes are 4 , Maximum Value is 99 and Minimum value is 5 which are present as fields in Header Node.

Circular Header Linked List

A Linked List whose last node points back to the First node or the Head Node of the list is called a Circular Linked List. Similarly, if the last node of the Header Linked List points back to Header Node then it is a Circular Header Linked List. The **last** node of the list does not hold **NULL** reference. In this case, We have to use external pointers to maintain the last node. The end of the list is not known while traversing. It can perform same operations as its counterpart such as Insert, Delete and Search. Let us see an example.



We can see the last node of the list points to the header node and not the node 99. There might be problem in traversing through the nodes. It is because the structure of the last node and header node are different which might cause an Type Mismatch Exception. So while traversing we fix the Header node and keep on traversing until node.next points to header node. The rest features of the Header remains same as its Grounded counterpart.