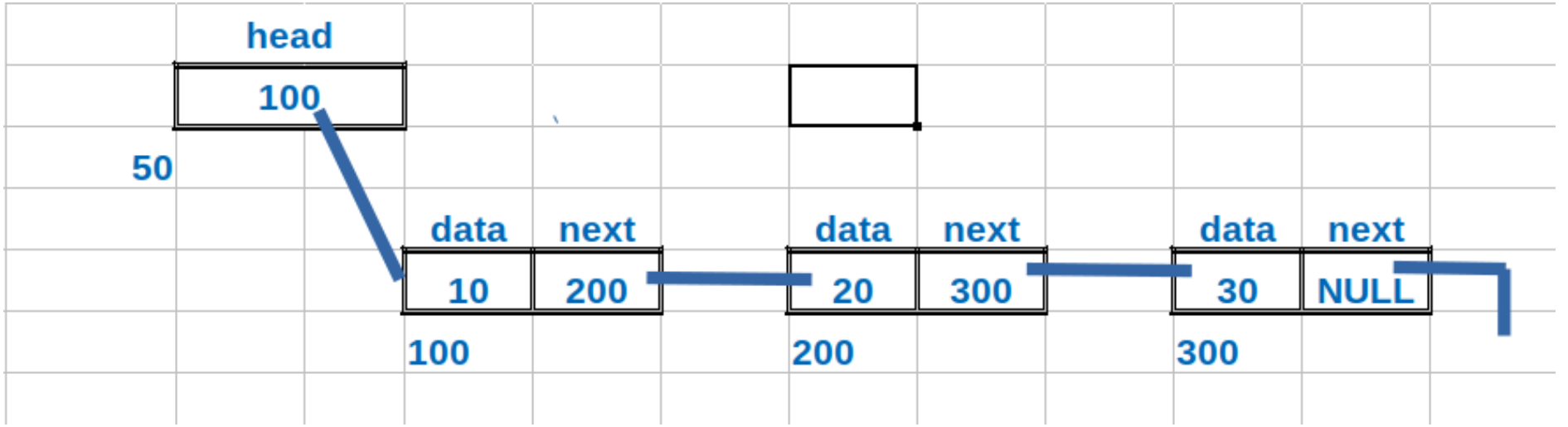
Linked List

- **Terminologies:**
  - **Linked list is a linear data structure that contains multiple records linked to each other.**
  - **Each item in the list is called as “Node”.**
  - **Each node contains data and pointer (address) to the next node.**
  - **Typically linked lists are implemented as self-referential structure/class. The structure/class contains pointer of the same type to hold address of next node.**

```
struct node
{
    int data;
    struct node * next;
};
```

Singly Linear Linked List diagram



Difference in array and linked list

	array	linked list
1	Array cannot grow or shrink dynamically (realloc() is not efficient).	Linked list can grow/shrink dynamically.
2	Array has contiguous memory. Hence random access is possible.	Nodes are not in contiguous memory. Hence only sequential access is allowed.
3	Arrays do not have memory overheads.	Linked lists have memory overheads e.g. each node contains address of 'next' node.
4	Insert/Delete at middle position need to shift remaining elements. Hence will be slower.	Insert/Delete at middle position need to modify links (next/prev pointers). Hence will be faster.
5	To deal to fixed number of elements and frequent random access, arrays are better.	To deal to dynamic number of elements and frequent insertion/deletion operators, linked lists are better.

Types of Linked List:

- **Singly Linear LinkedList**
- **Singly Circular LinkedList**
- **Doubly Linear LinkedList**
- **Doubly Circular LinkedList**

Operations can be performed onLinkedList:

1. add_at_last, add_at_first, add_at_position
2. del_from_last, del_from_first, del_from_position
3. traverse (display)
4. reverse
5. search
6. sort
7. merge