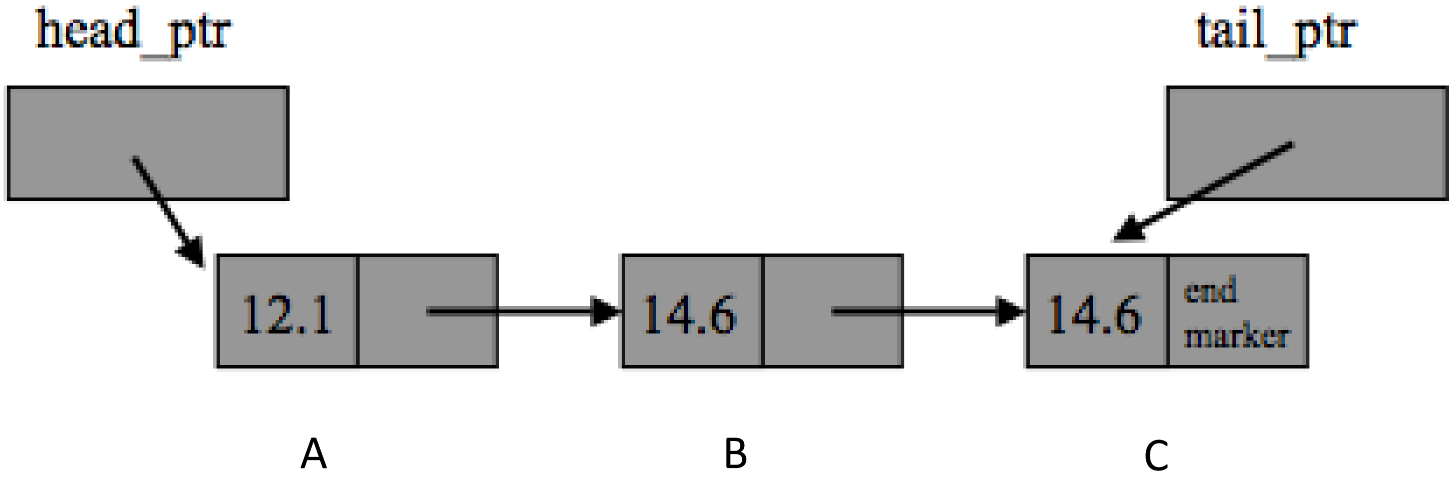
Single Linked List



Simple Q containing 3 items/nodes A, B and C (numbers in this case)

The head_ptr points to first item in the Q

The tail_ptr points to last item in Q

End marker is typically NULL (zero?)

The basics

- Head does not link to tail.

  You should think of them as separate items.

  Head points to the start of the list and tail to the end.

- Head and tail start out empty (pointing to NULL).

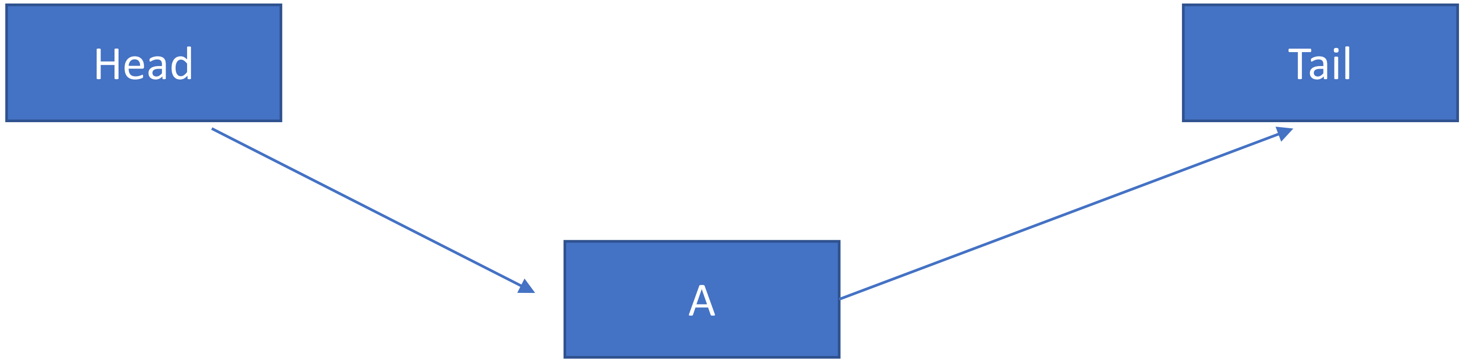
- With the first new Node (let's call it A) added

  Both head and tail are set to point to A

  Since it is the only entry it is by definition both the start and end.

  The A.next pointer is set to point to NULL.

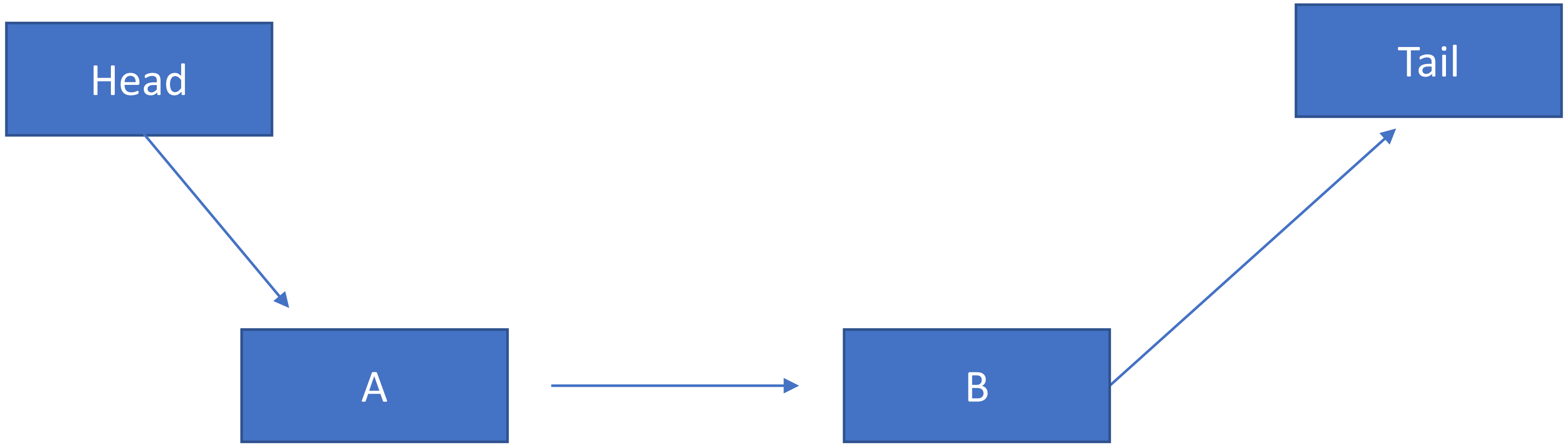
What does it look like now?



The basics – Cont.

- When you add a second node (B)

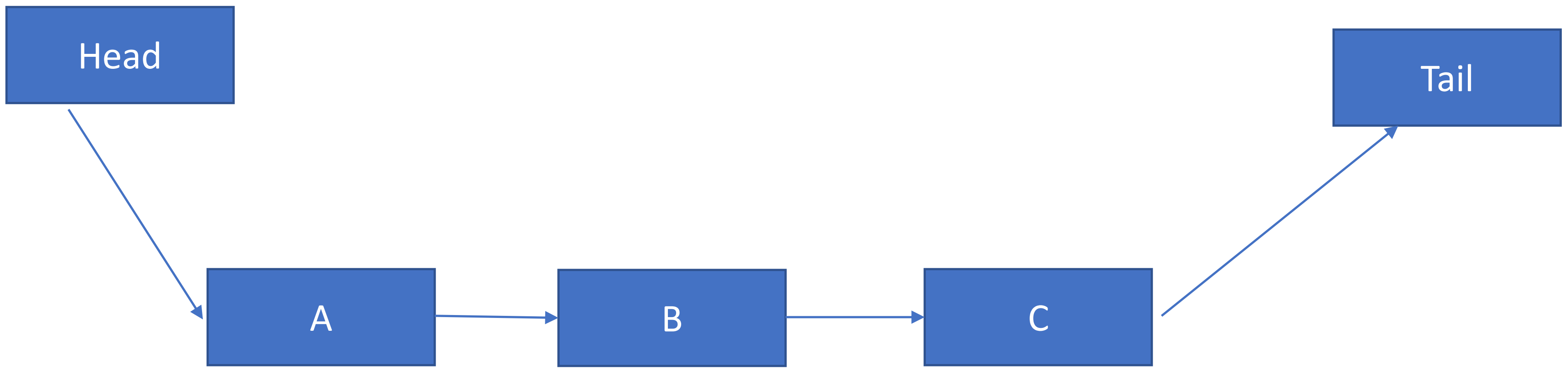
  Then the head stays pointing at A,
  The Tail is set to to point to B,
  The A.next pointer to set to point to B.
  The B.next pointer is set to point to NULL.



The basics – Cont.

- When a third node (C) is added:

  Neither the head nor A change,
  B.next is set to point to C, and.
  C.next is set to point to NULL (i.e.: the tail)



Removal of node ‘A’

- We take the node immediately before node A, and point its next pointer to the node immediately after node A

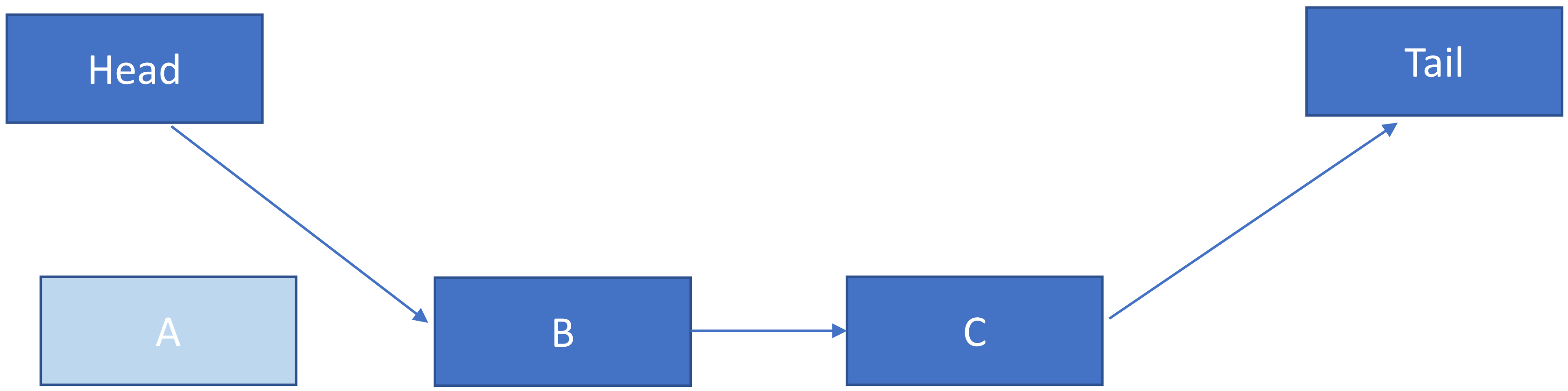
  Head is updated to point to B

  Tail is left pointing to C

  The memory used for A is given back to

- What does it look like now ?

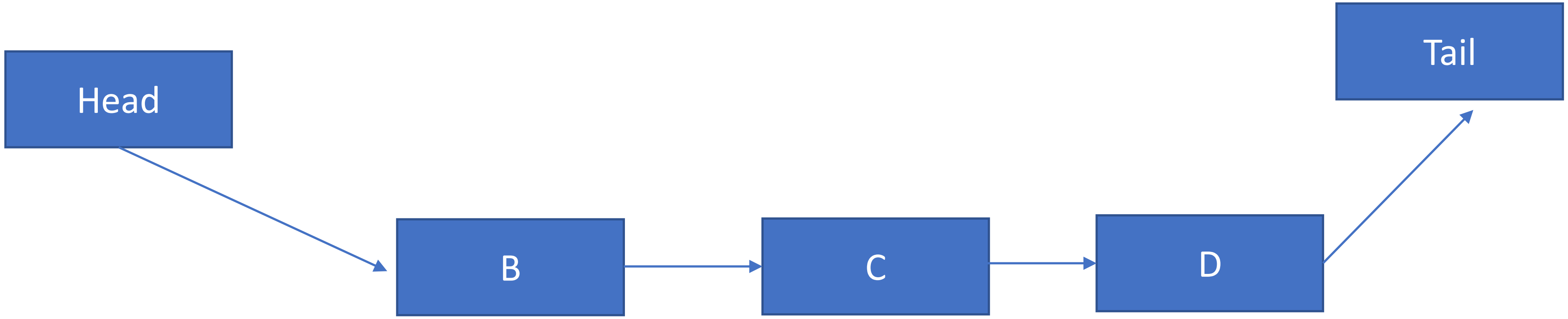
What does it look like now?



New node Insertion

- We insert node D after node C

We update the next pointer of node D to point to whatever node C is pointing to – in this case to null (the tail).
Then we update the next pointer of node C to now point to node D.


Head
B
C
D
Tail