Linked List Design: Node vs LinkedList

1. Can Methods Be Written Inside the Node Class?

Yes, Python allows methods such as traversal or insertion to be written inside the Node class. This approach works technically and is sometimes seen in beginner tutorials.

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

    def traverse(self):
        current = self
        while current:
            print(current.data, end=' -> ')
            current = current.next
        print('None')
```

2. Why This Design Is Problematic

A Node represents a single element, not the entire linked list. Placing list operations inside the Node class assumes the calling node is always the head. This can lead to partial traversals and logical bugs.

3. Recommended and Industry-Standard Design

The clean design separates responsibilities. The Node class stores data and the reference to the next node. The LinkedList class manages traversal, insertion, deletion, and maintains the head.

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None

    def traverse(self):
        current = self.head
        while current:
            print(current.data, end=' -> ')
            current = current.next
        print('None')
```

4. Key Takeaway

For strong DSA fundamentals and interview readiness, always separate Node and LinkedList responsibilities. Treat Node as a simple data container and LinkedList as the manager of all operations.