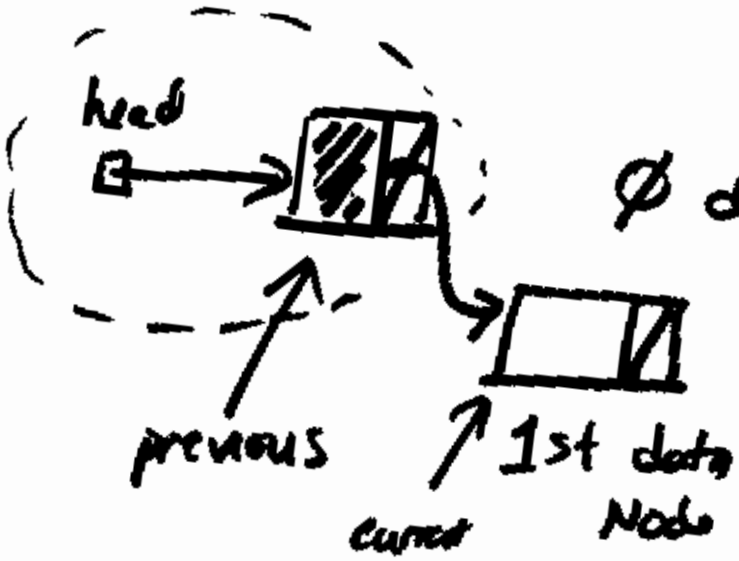
Topics

- Dummy Head Node
- NON-Linear Abstraction

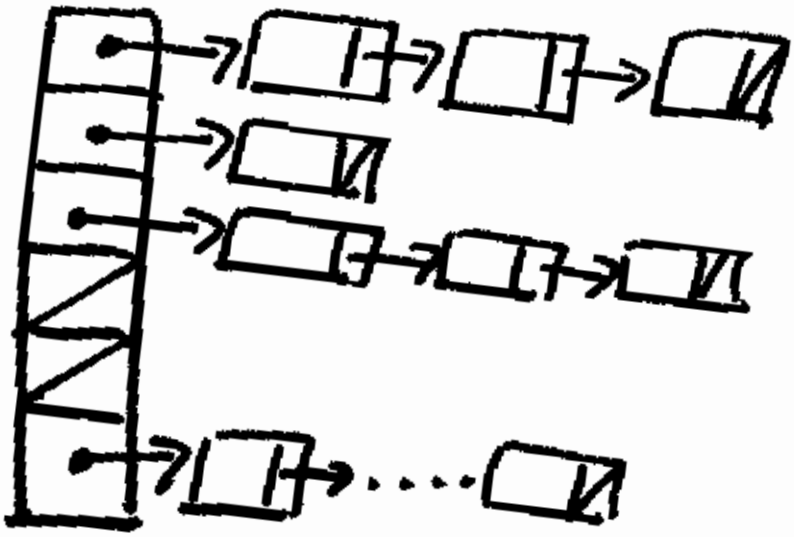


Remove 1st Node

```
previous = head;
current = head->next;

if(current)
{   previous->next = current->next;
    delete current;
}
```

Non Linear



Value Oriented Abstractions

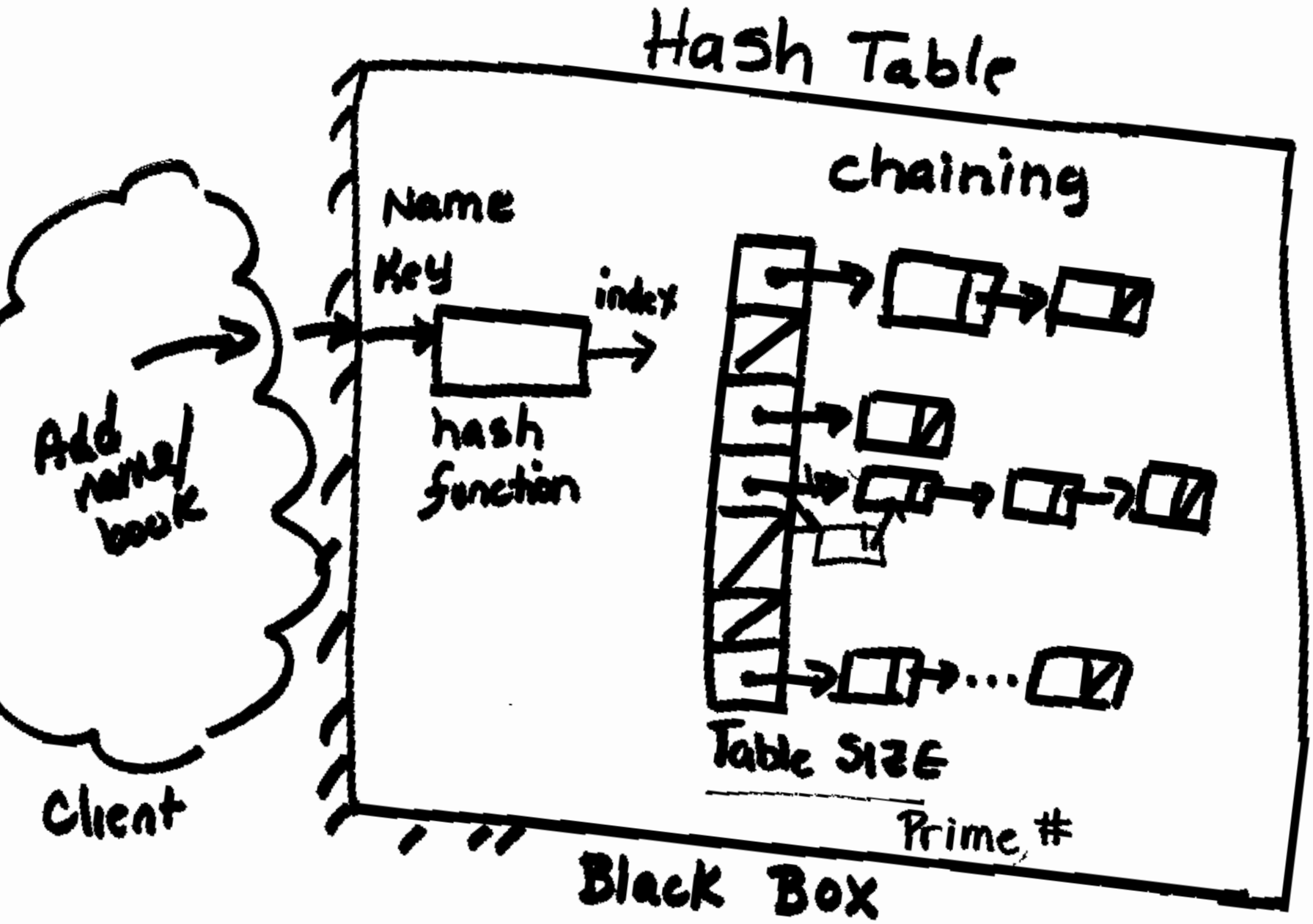
Insert/remove	Search	Display
Binary Search V+	Binary Search +	V+
Shift —		
Dictionary		
Binary Search V+	SeQ Search —	Sorting Alg —
DirectAccess V+		
Insert —		
SeQ Search —	SeQ Search —	V+
NO Shifting +		
SeQ Search —	SeQ Search —	Sorting Alg —
DirectAccess V+		
Insert —		
SeQ Search —	SeQ Search (?) —	V+
NO Shifting +		
SeQ Search —	SeQ Search —	Sorting Alg —
DirectAccess V+		
Insert —		
SeQ Search —	SeQ Search —	Sorting Alg —

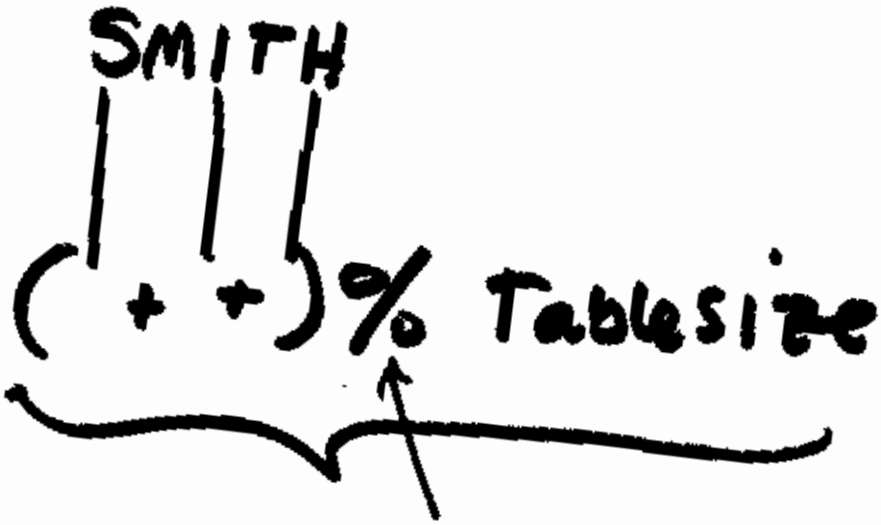
Array-Sorted

Array-Unsorted

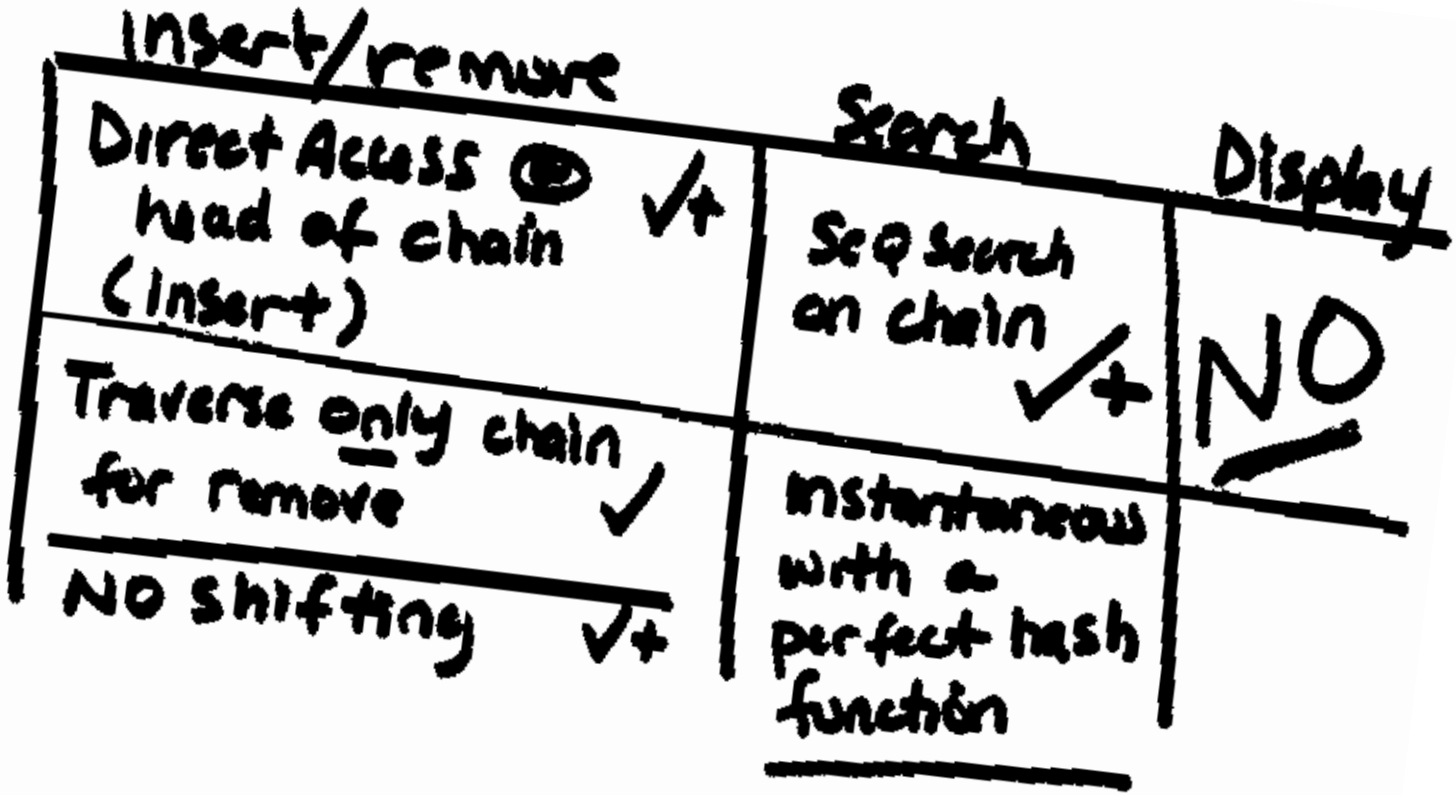
LL-Sorted

LL-Unsorted


Hash Table
chaining
Name
Key
index
hash function
Add name/book
Client
Table Size
Prime #
Black Box


SMITH
(+ +) % Tablesize

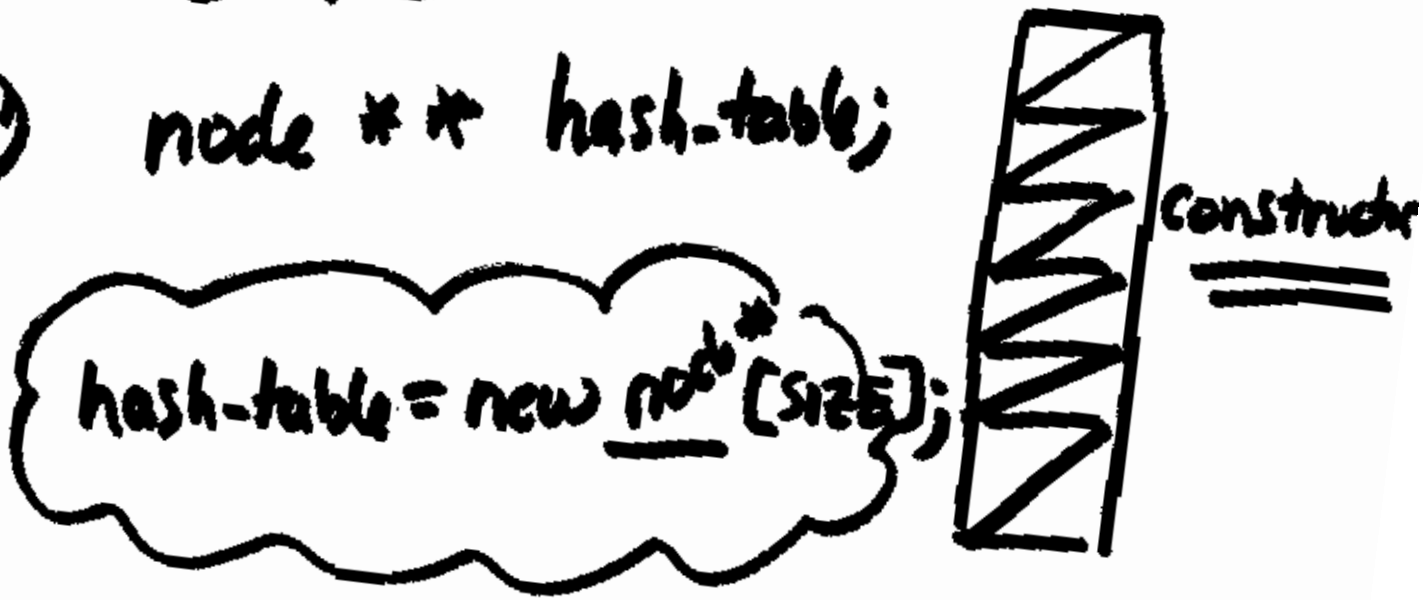
Hash Table


insert/remove
Search
Display
Direct Access ✓+
head of chain
(insert)
Seq search
on chain ✓+
NO
Traverse only chain
for remove ✓
NO shifting ✓+
instantaneous
with a
perfect hash
function

Private

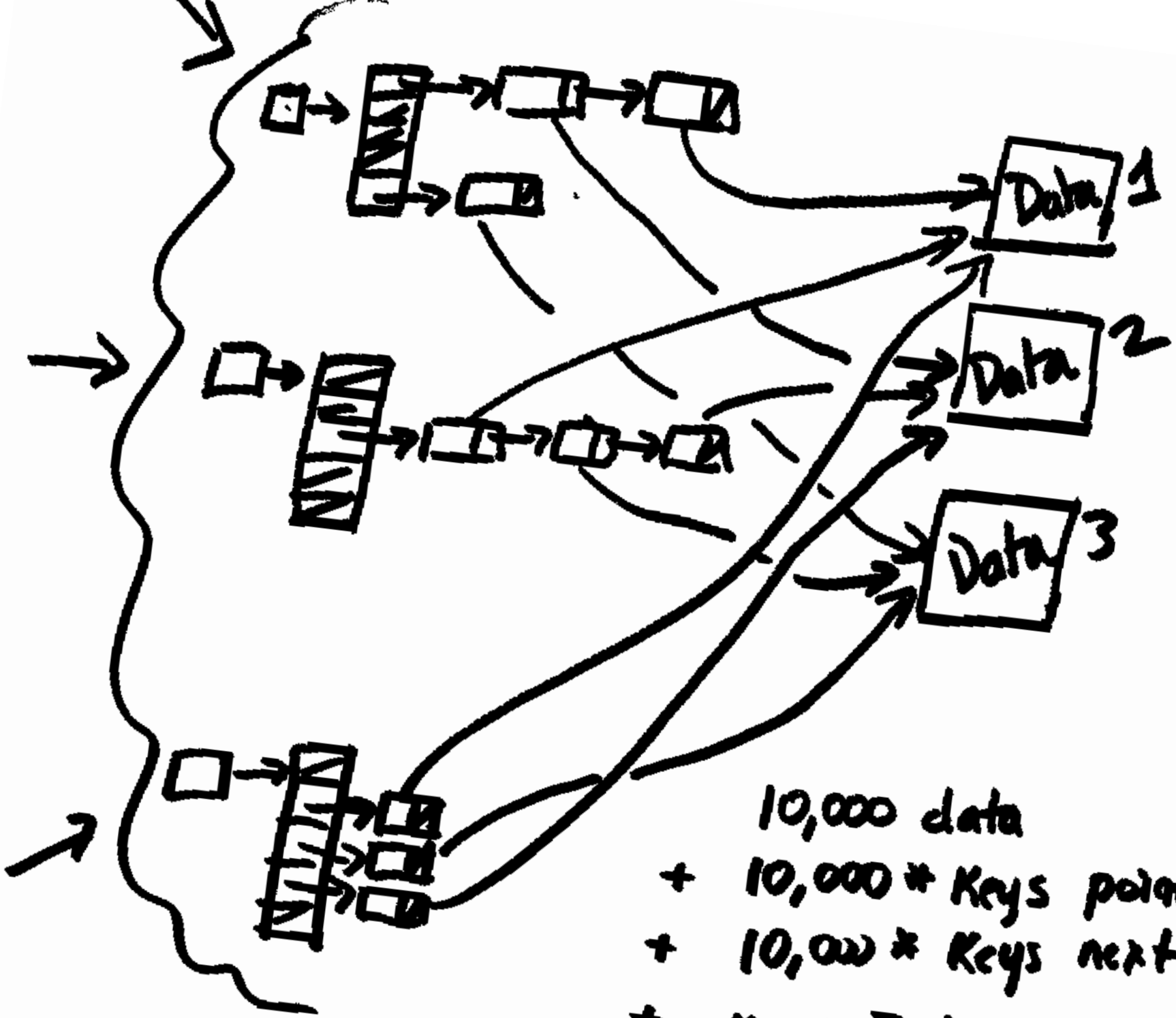
① node * hash_table[101];

② node ** hash_table;



Insert

```
index = (Key[0] + Key[2] + Key[4]) %
                                TableSize;
current = hash_table[index];
hash_table[index] = newnode;
newnode->next = current;
```


Data 1
Data 2
Data 3
10,000 data
+ 10,000 * Keys pointer to data
+ 10,000 * Keys next
+ Key * Tablesize pointer