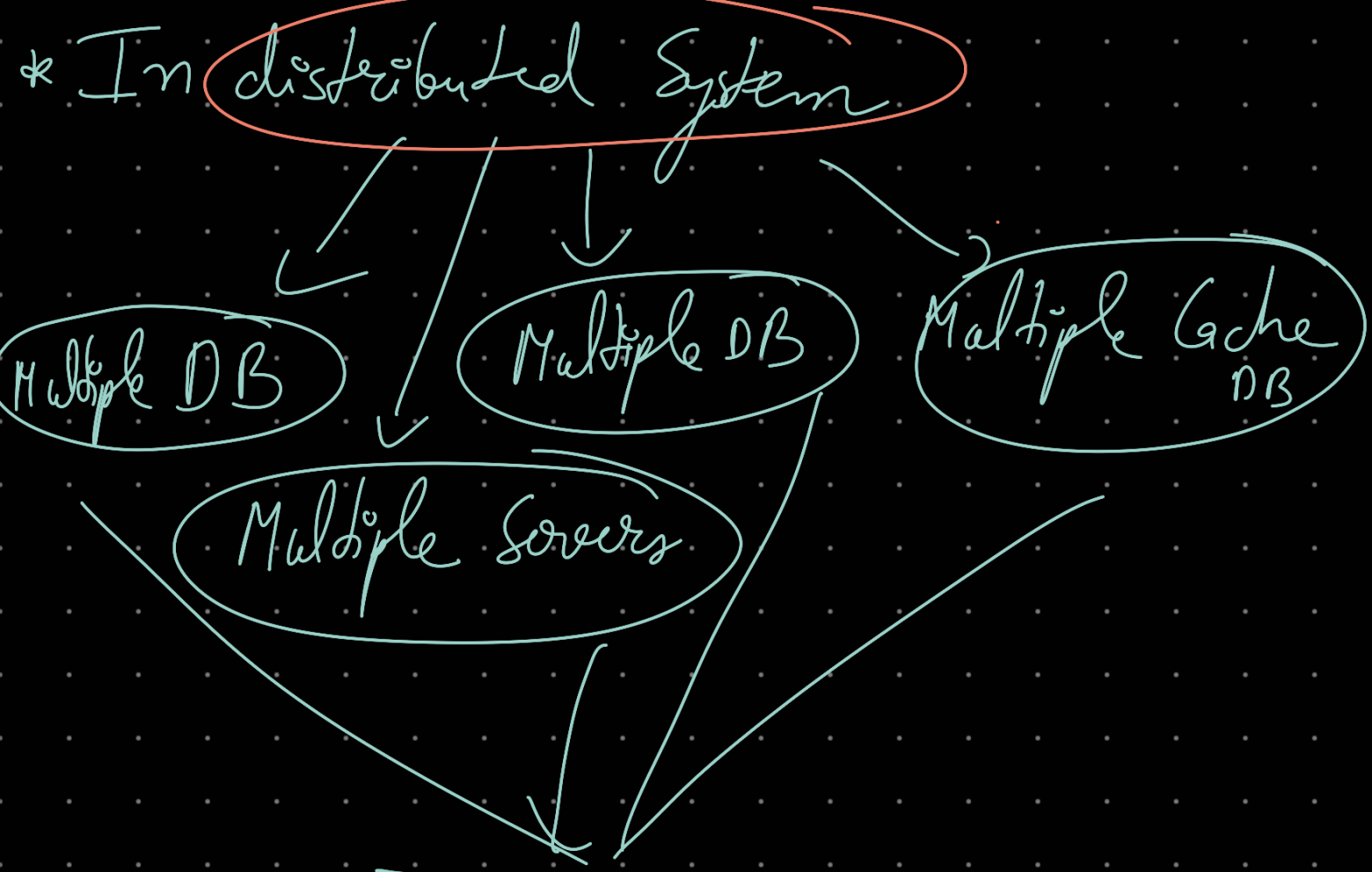
* In distributed System



Assume Multiple process on Multiple physical Machins

⇓

Need of Routing ?

Let's considered Distributed Database

1 Million records → split into → DB 1, DB 2, ⋮, DB n

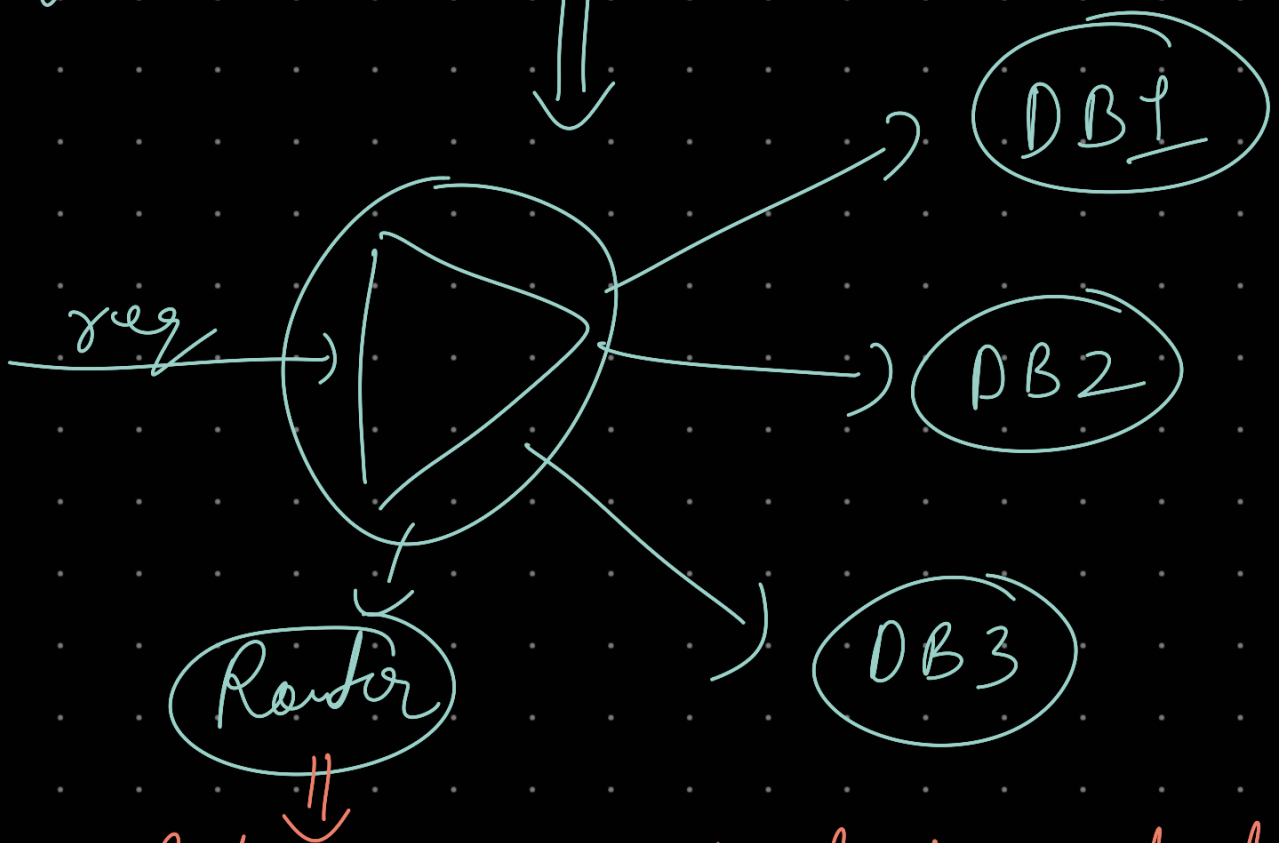
* We go distributed when one Machine can't handle load due to RAM, CPU

⇓

challenge

How to route request client to Right DBMS Machine?

⇓



1 solution store meta data which key present on which DB (static routing)

⇓

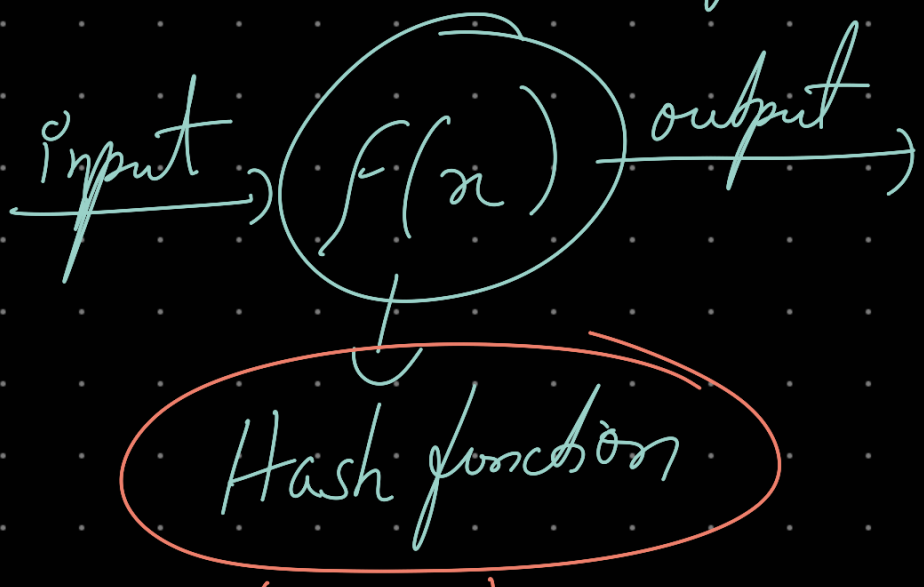
What if we have 1 crore Record

⇓

This metadata size will increase

⇓

To solve → Need function



Hash function

SHA256 MD5 ..... etc

Simple hash function (mod n)

key	hash		cache 0
K1	0	cache 0	cache 1
K2	0	cache 0 *	h=f(n) % 2
K3	1	cache 1	
K4	1	cache 1	
K5	1	cache 1 *	
K6	0	cache 0 *	

⇒ mod 2

⇓

New server added, fraction changed mod 3

New cache server added : cache 2

key	hash		
K1	0	cache 0	
K2	2	cache 2	*
K3	1	cache 1	
K4	1	cache 1	
K5	0	cache 0	*
K6	2	cache 2	*

50% data movement

This required time, I/O + Networking

⇓

How to solve

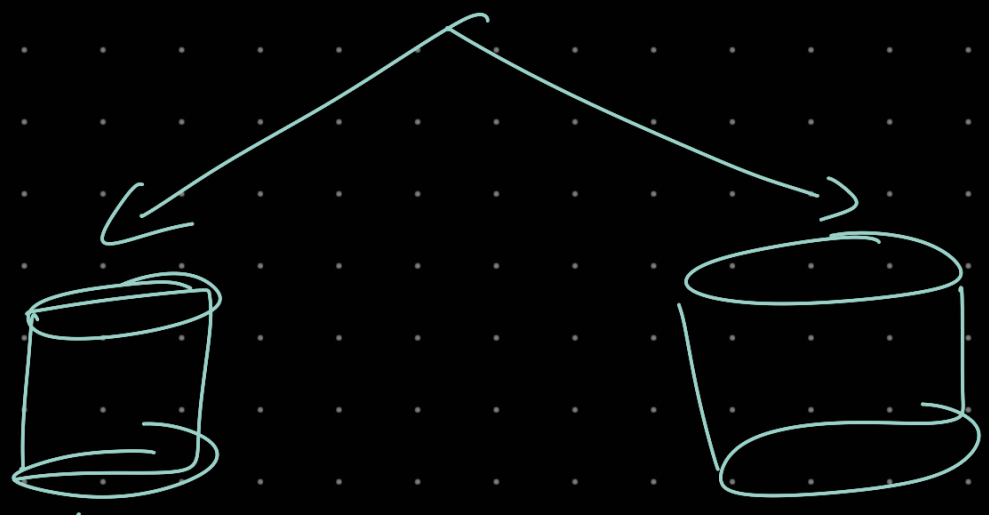
⇓

[Consistent Hashing]

1 Million Record

will cover later

↳ we decided to shard



⇓

We need to create some DS and Algorithm to distribute 1 Million Keys to DB1 and DB2

⇓

This became a distributed system now

⇓

Hash ring → Virtual circle

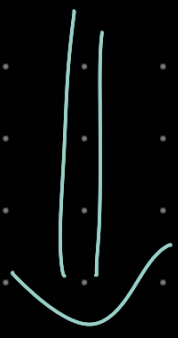
MD-5 / SHA-256

∴ Range of f(x) is continuous circle

E.g → Assume a function

$$0 < y < 2^{32} - 1$$

Last value are adjacent could be consider as circle



① Assign Server/DB/Cache Nodes to position on Ring

↳ Use any unique identifier of node

↳ IP/Name/MAC

Eg → hash(192.168.1.1) → 12345

Any hash function

We have 3 nodes (C1, C2, C3)

Node 1 → h(x) → 12345

Node 2 → h(x) → 67890

Node 3 → h(x) → 23456

② Use same hash function to hash Data key

E.g:- User123 → h(n) = 54321

(3) Assign this Data key to closest clockwise Node

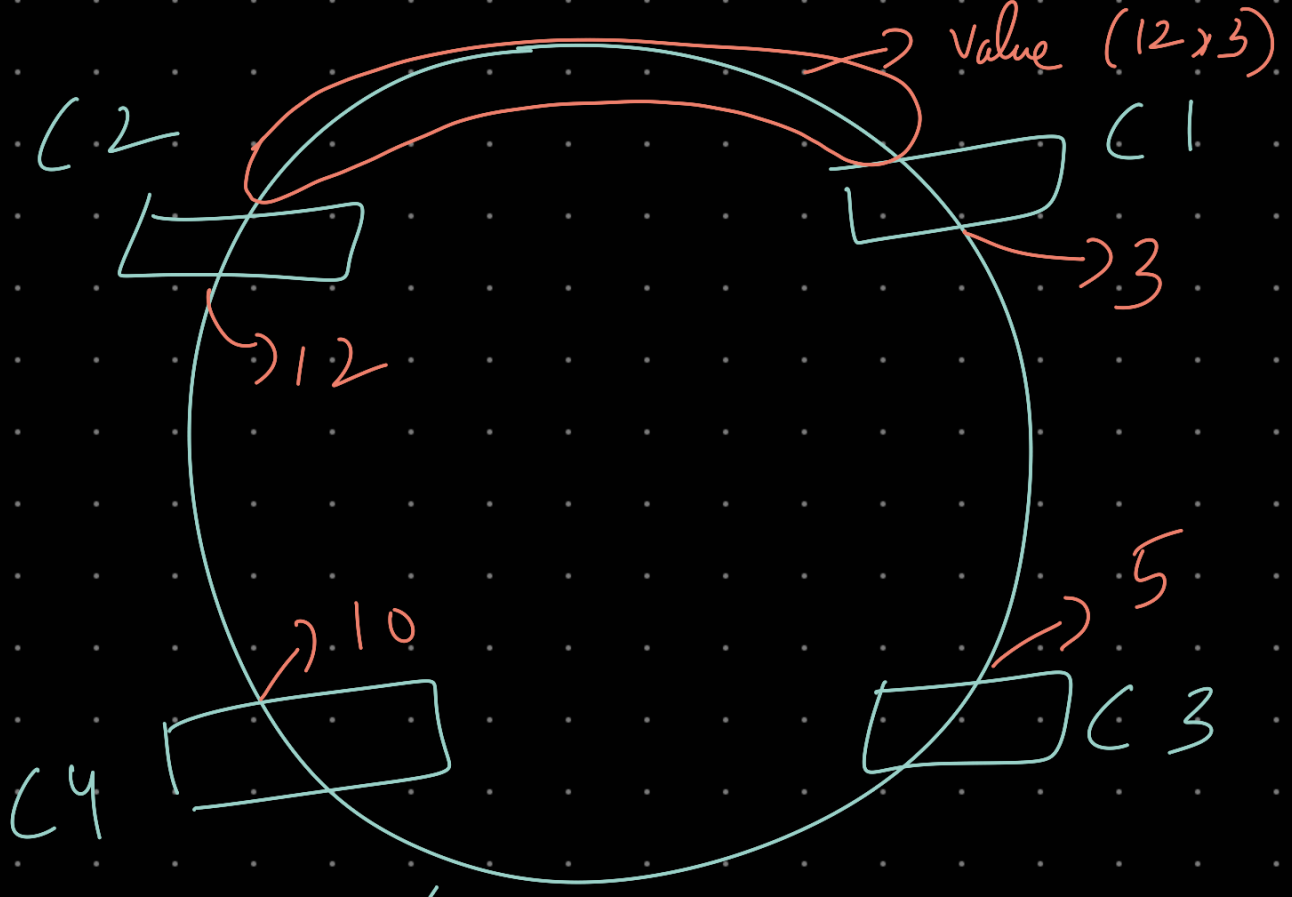
23456 < 54321 < 67890
(Node 3) (User123) (Node 2)

Mapped to

User123 → stored in → Node 2

Let's visualise

Consider C1, C2, C3, C4



(, value (0, 15)

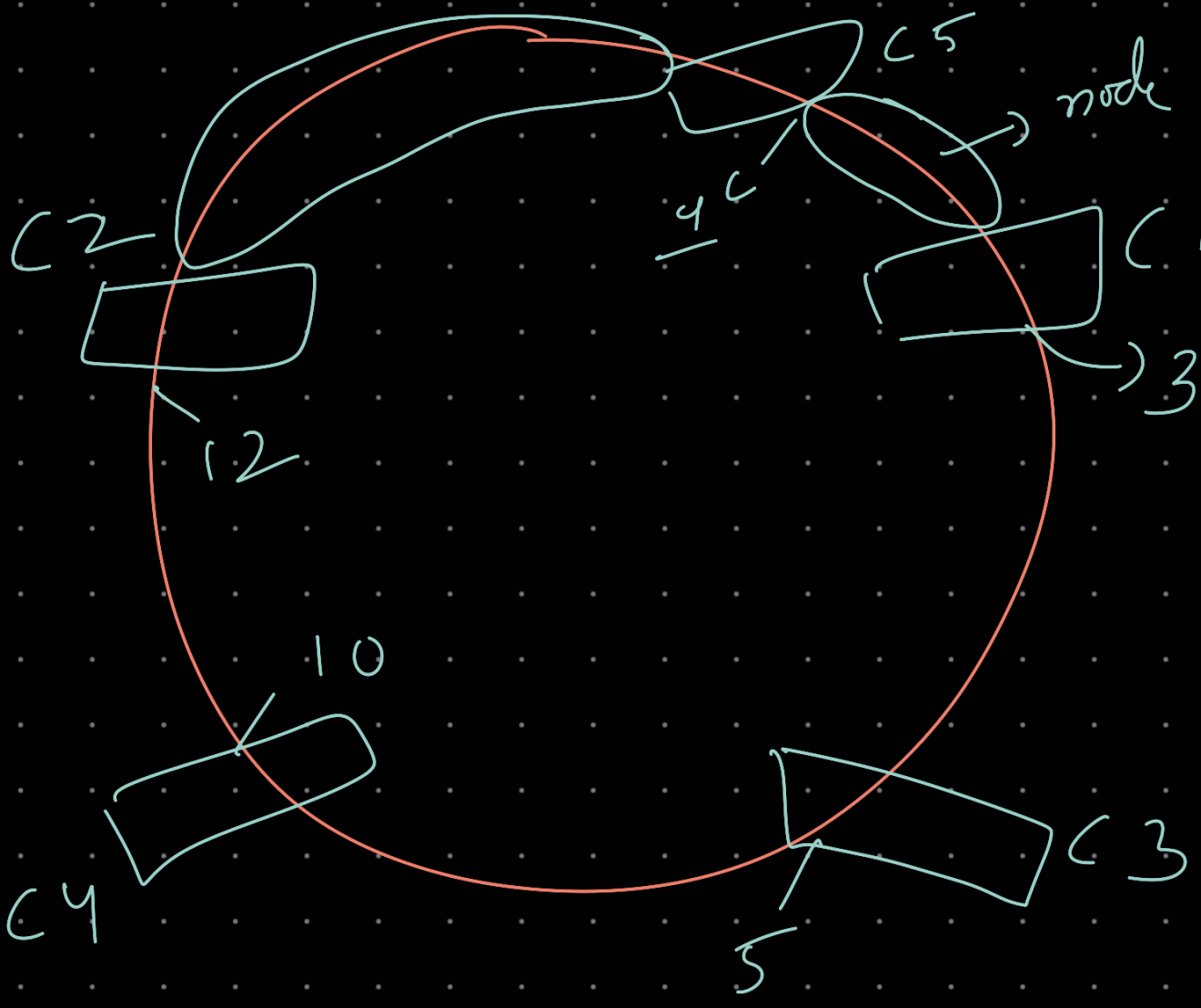
||

add C5 → h(x) → Assume location (1)

Any Key Mapped to value between 12 to 3, goes to 3 currently

||

Adding C5 to position 1



12 to 4 → goes to C5 now

1 to 3 → remains to C1

other keys → No data movement

↓

Data Movement is minimized

↓↓

Problem with Consistent Hashing

↳ Hash function gives random value

↳ So Uneven distribution of data across nodes

⇓ How to solve

Each physical node ← Multiple Virtual node

Eg:- Node 1 → 12345, 52351, 56789

3 virtual nodes

⇓

Data distributed across multiple virtual node

⇓

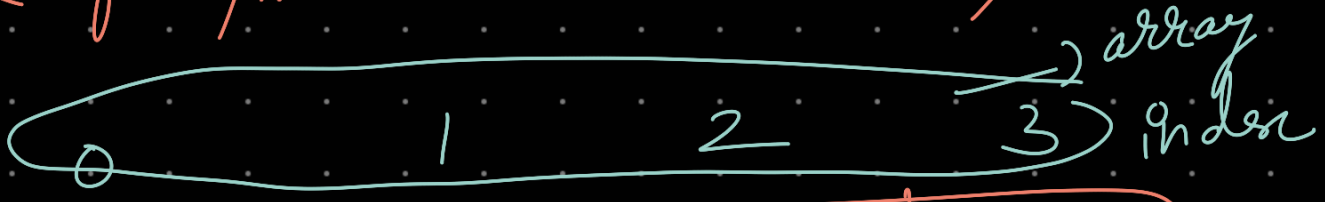
How to Implement this using Code?

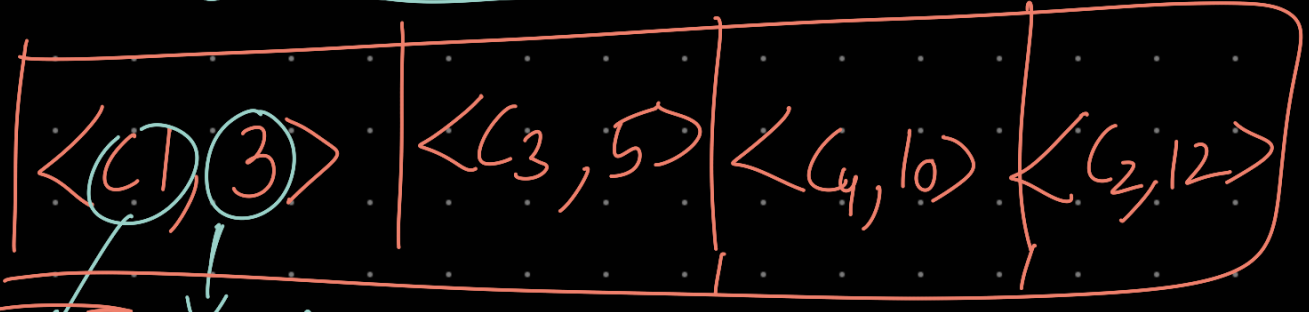
① Create fake Linked list and array of size = no. of nodes

E.g. → 4 nodes ⟶ 4 size

what to store in array element

< Connection val of DB/Node , Hash value >





↳ Used for query DB

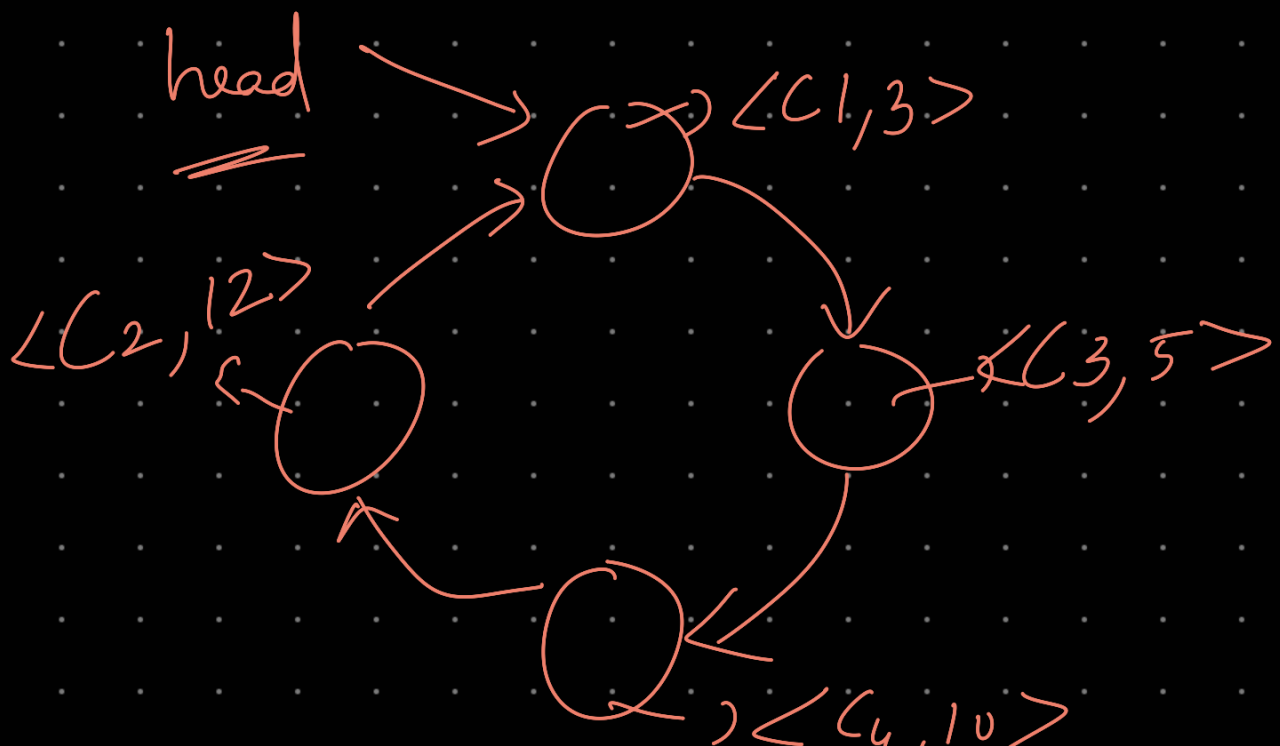
Key 1 → h(x) → Assume gave 0

↓ hash value

⇓

Linear search to array, compare hash value of each index; 3 > 0 → store here

& can use linked list as well



⇓

If, Array will very large?

⇓

Use binary search

⇓

Can use BST

⇓

For Range Query → Use Segment Tree

* Does consistent hashing Expensive?

Low RAM Low Time comp

* Consistent Hashing can be implemented in Backend Server.