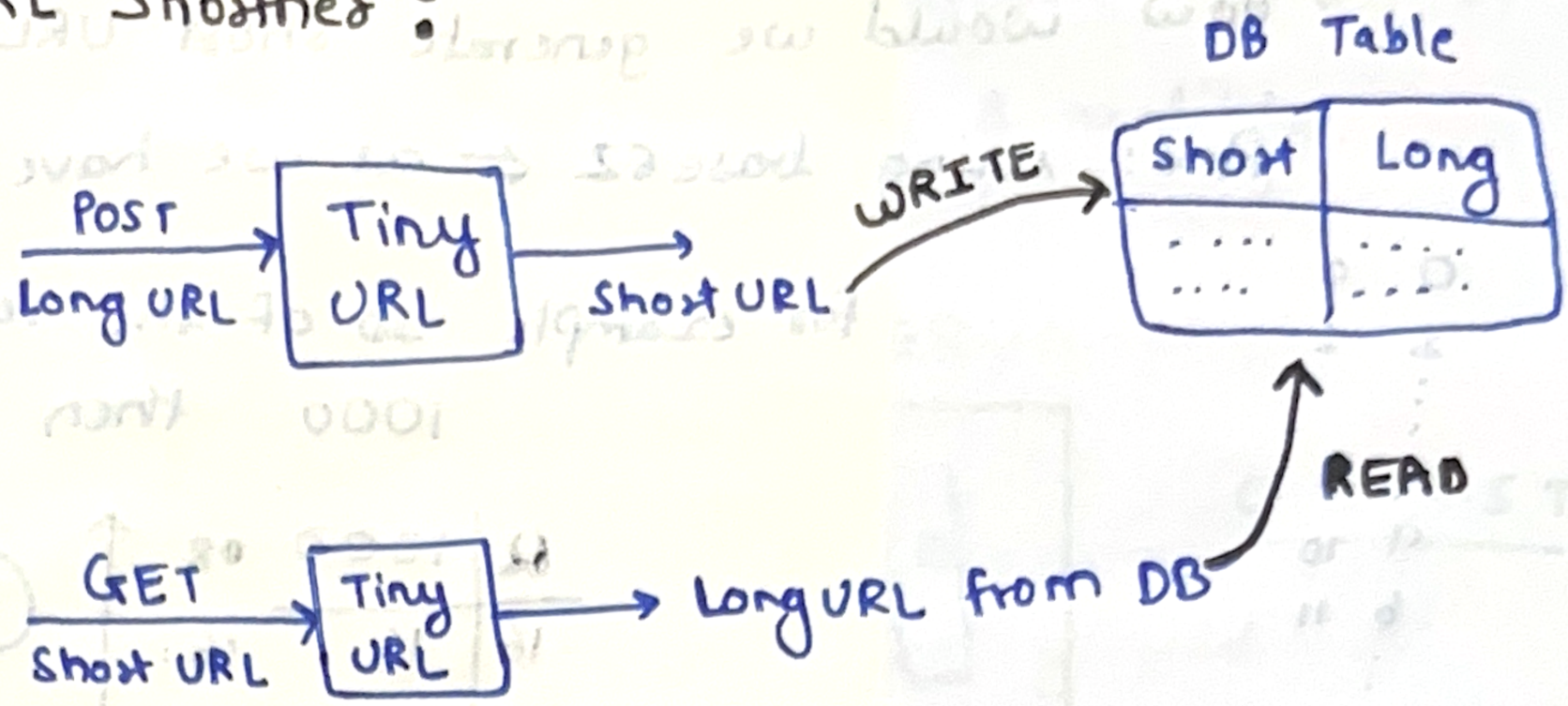
Design URL Shortner :



✳ Requirement analysis :

↳ How many chasacters ?

Ans: to support 10 M traffic per day for min 100 yrs

calculating numbers ⇒ $10M \times 365 \times 100 = 365B$ URLs
(days, years)

allowed chars ⇒ [0-9] | [a-z] | [A-Z] = 62 chars

so $\underline{62^7 = 3.5T}$ we should use 7 chars

↳ should we use hash function ?

- No, as MD5 and SHA-1 generate length more than 7 (more than req.)
- we can't trim hash f^n output as it can lead to duplicates. (collission)

↳ How would we generate short URL then?

Ans: using base 62 ⇐ as we have 62 chars

0	0
1	1
⋮	
9	9
a	10
b	11
⋮	
z	
A	
B	
⋮	
Z	

for example ID of the request is 1000 then

62	1000	98
16	16	16

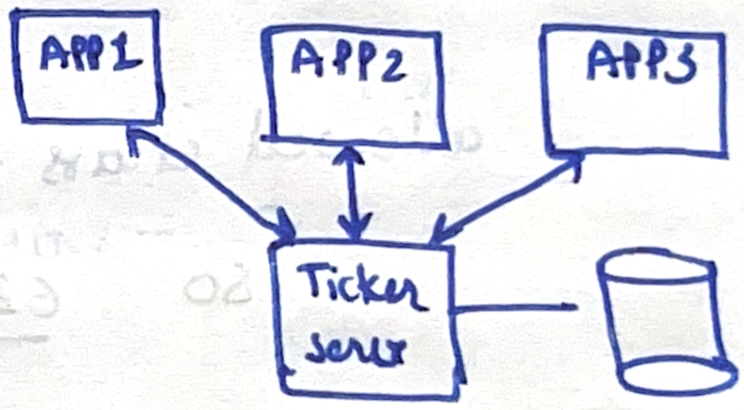
(98) ↑ How

problems: ① where would we generate ID?

② How to fill rest of the bit to make it 7 digit?

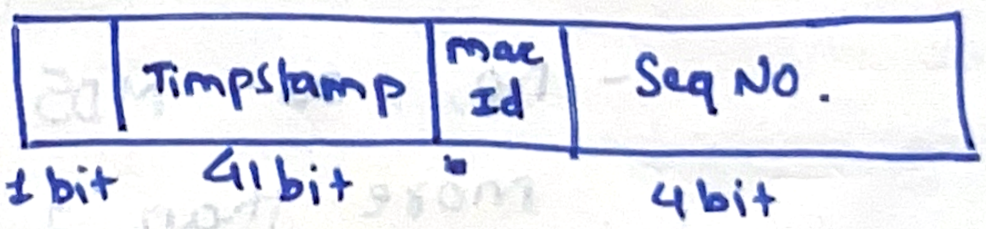
Q1: ID generator: ⇒ DB Id column in one DB is not feasible

Approach - ① : Ticker Server



* issue is it is single point of failure & still can't scale with DB.

Approach - ② : Snowflake



Twitter uses it ⇒ this will keep the Id unique and seq No are limited and would cycle itself upon exaustion

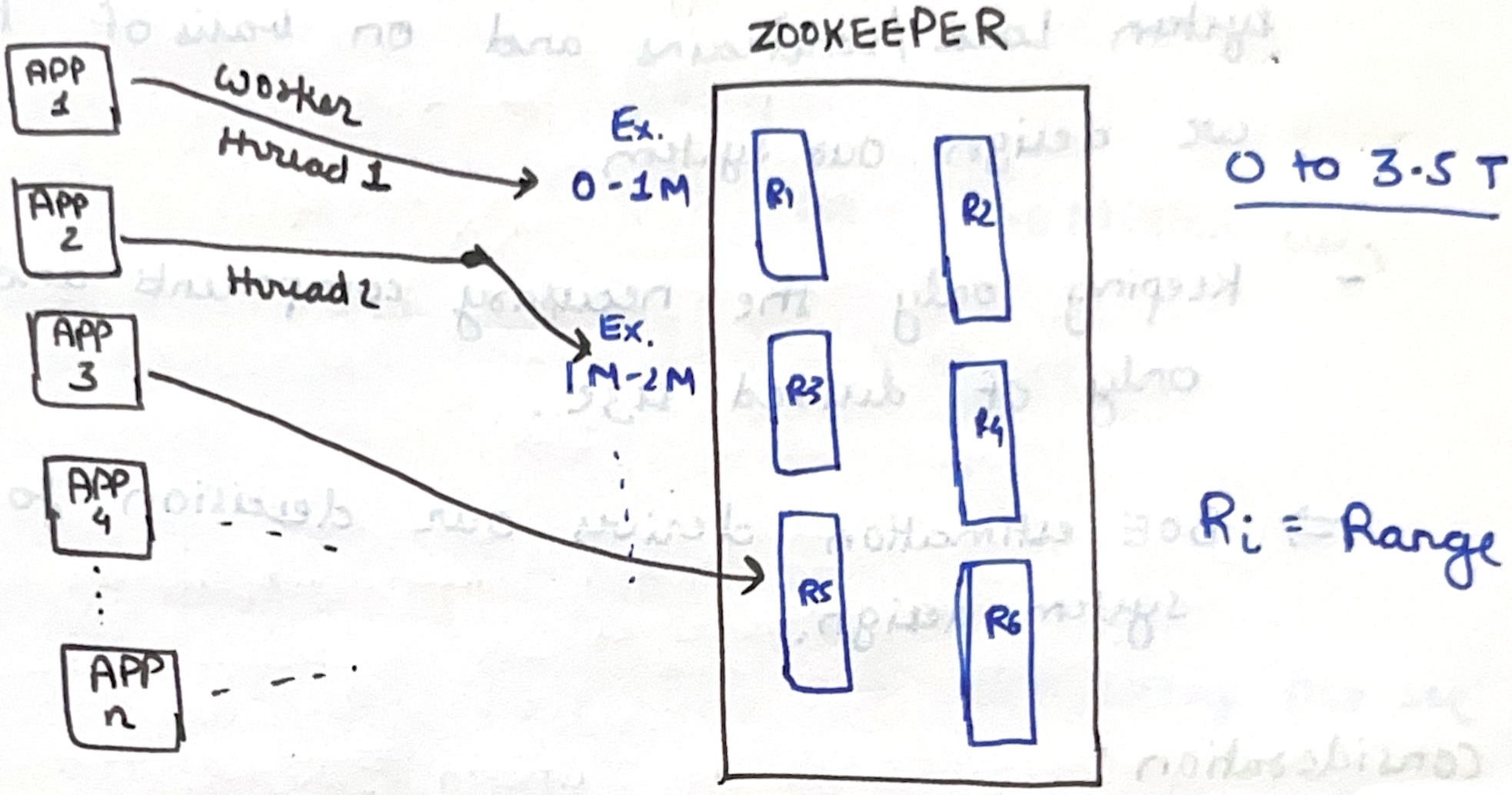
{ same TS, same Machine would use diff seqNo. and vice versa

↳ Keeping it unique.

ID generation

it can't be the ID colum of DB as we have to fulfill 1M/day request we can't keep one DB & sync bet^ the DB to keep ID unique would be the problem.

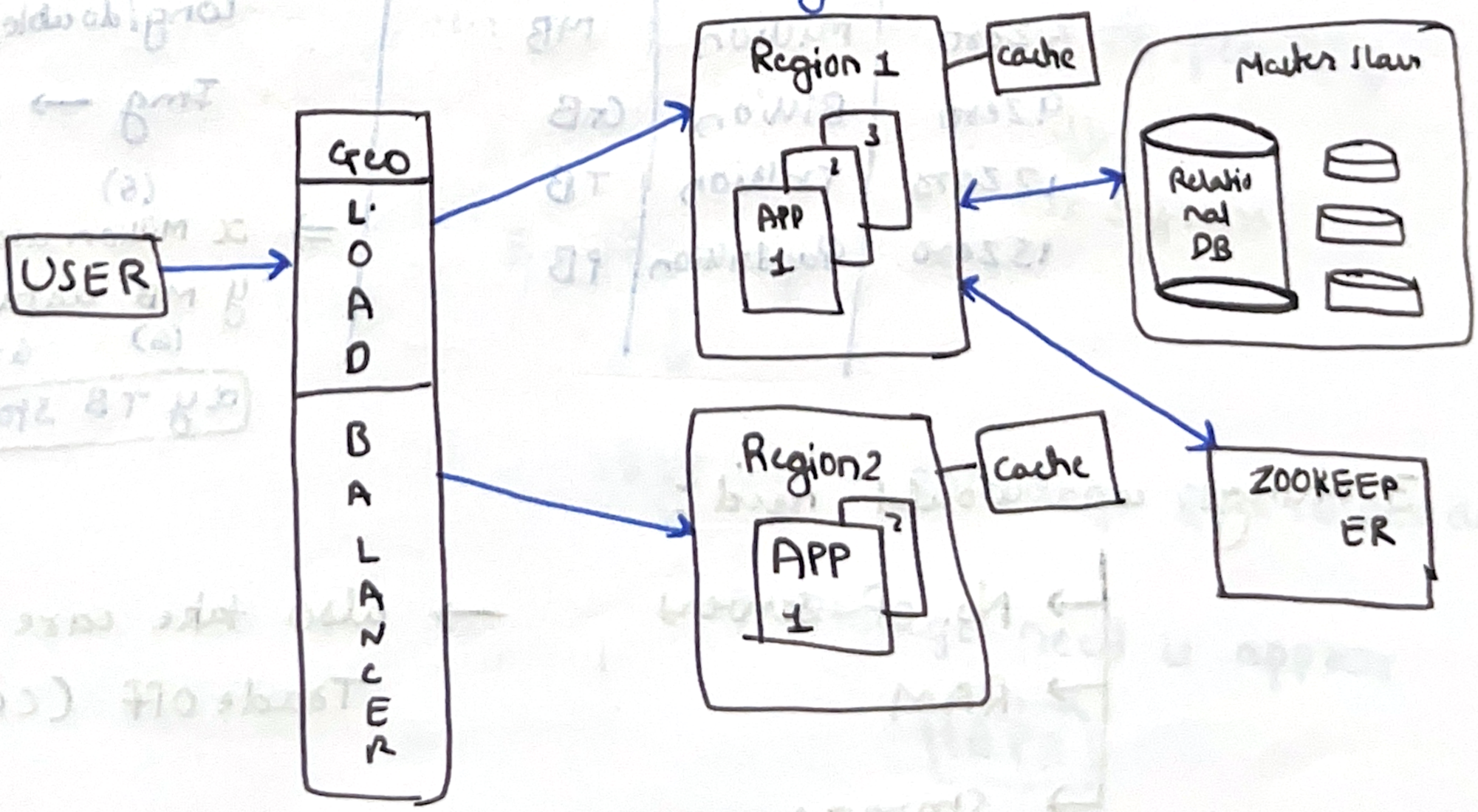
Approach - ③: Zookeeper : distributed system can coordinate with each other efficiently & reliably.



⇒ each worker thread would be assigned a range and once the range exhaust' it moves to next range.

Q2 filling up the range:

we can add some padding like ==== ....



* Here each region would have Its own database but to handle consistency in case of failure, we can have an active Passive - setup where main Regions dB can serve app & write & reads are handled from there itself but async data replication is happening to secondary region