Finish design of FB Messenger

Design

1.) How sender/receiver get/send messages
⇒ websocket

2.) One Req ⇒ Server Multiple Times
⇒ we need a way for server to identify a duplicate req
Idempotency key

3.) Sharding

4.) Consistency

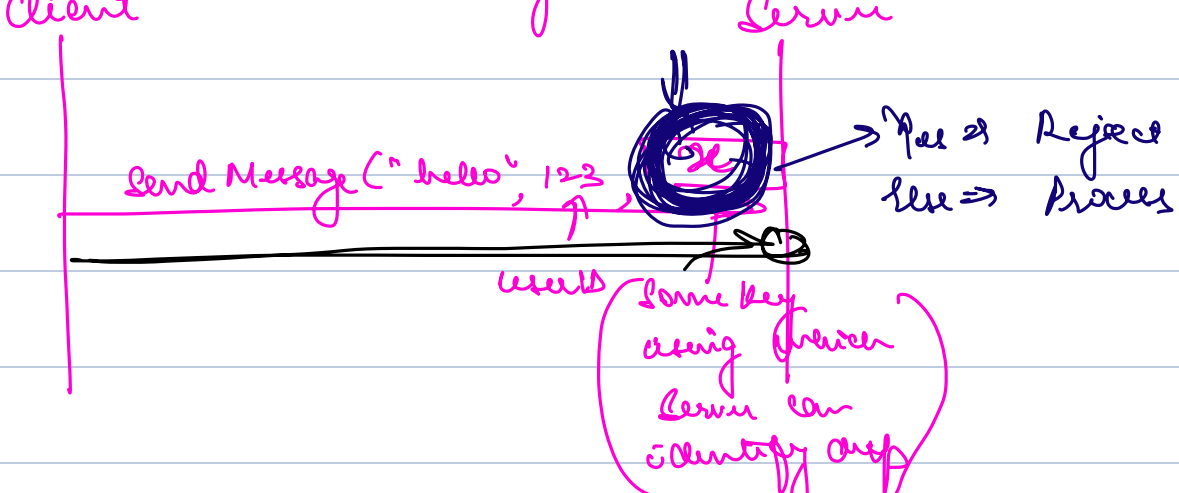
5.) DB to choose

IDEMPOTENCY

even if we send req multiple times, should only act once

⇒ Messaging

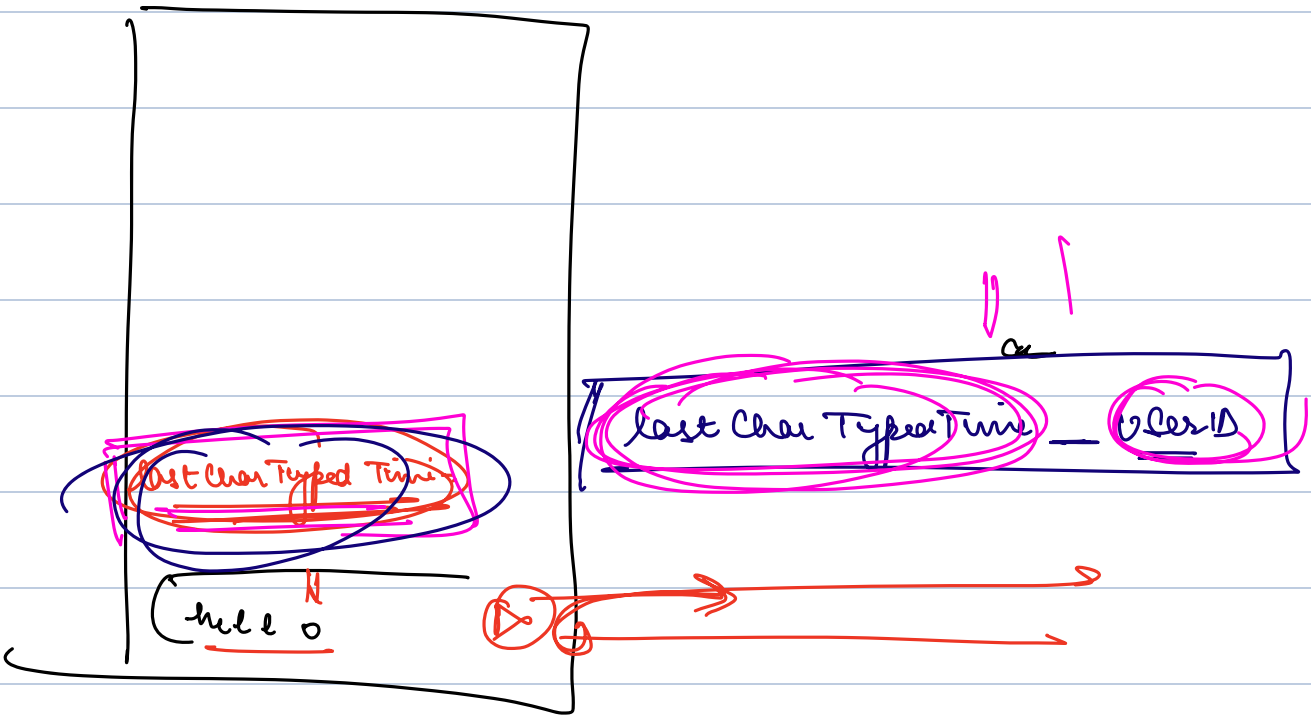
⇒ Payments

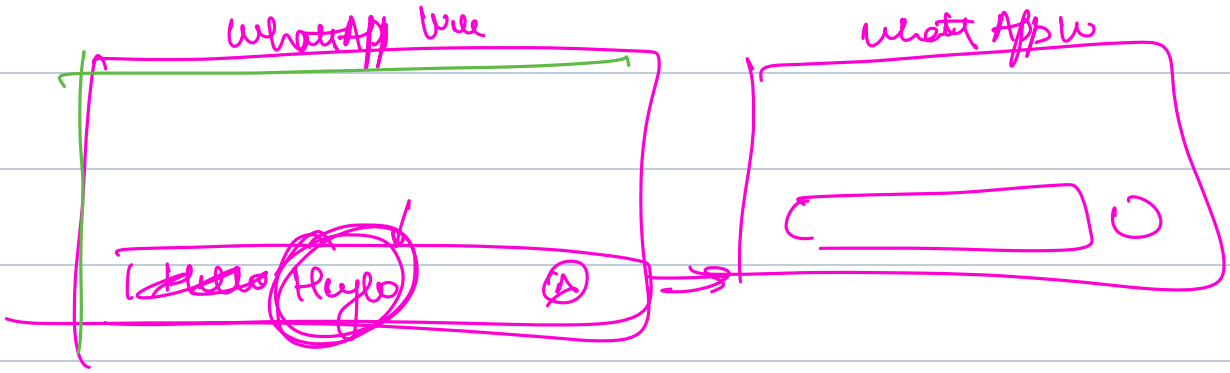


timestamp

↳ which timestamp

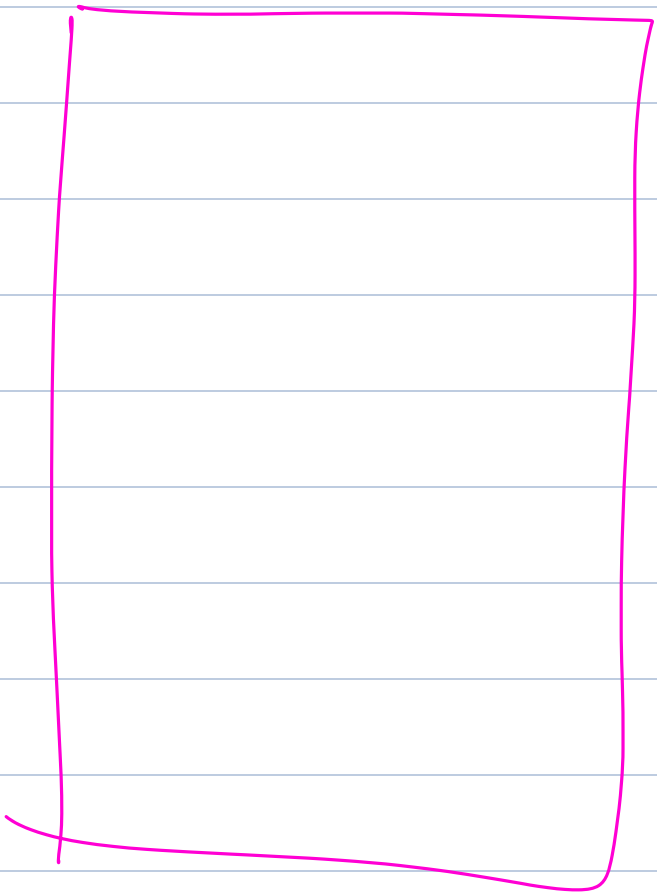
↳ server_timestamp

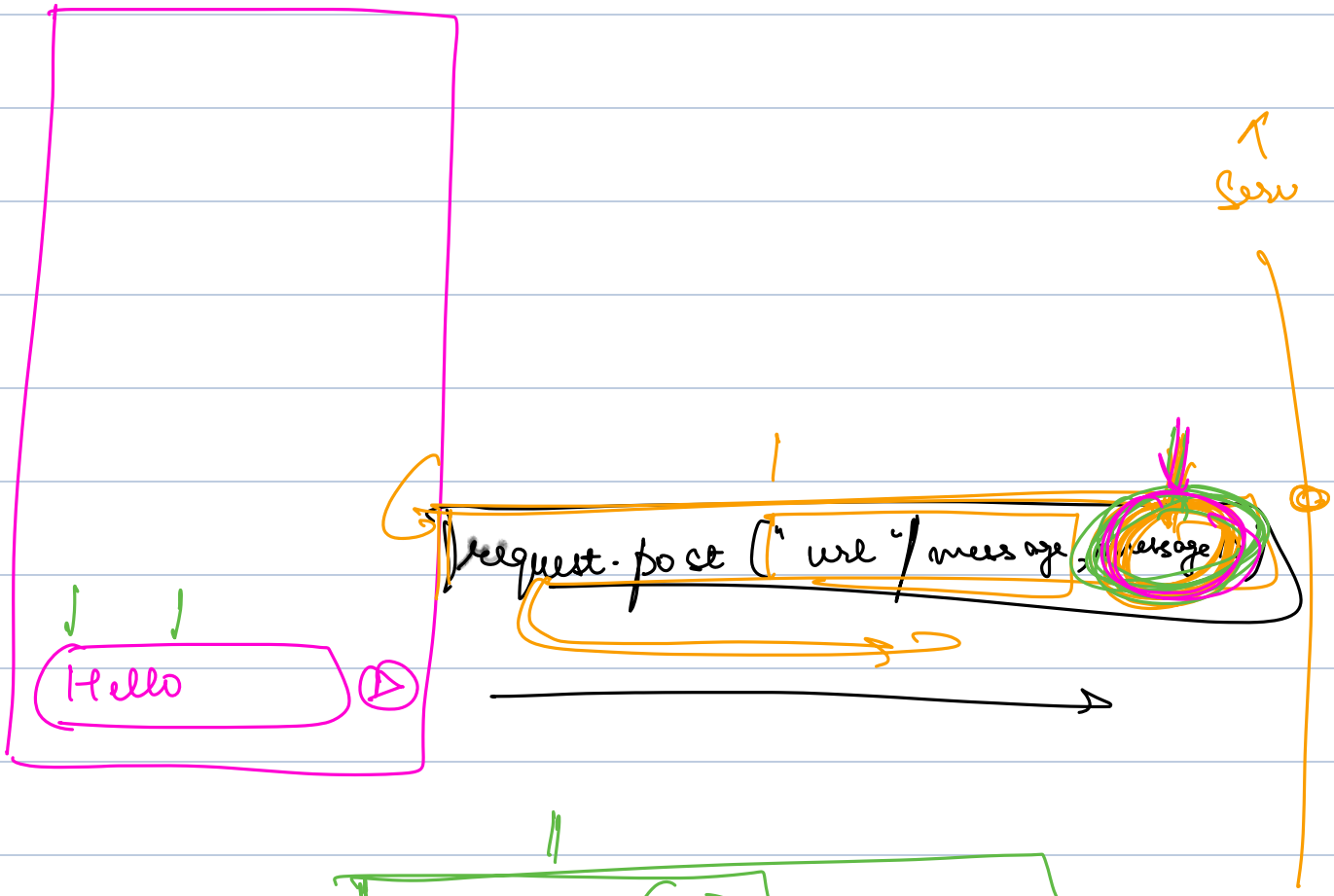




$\frac{1}{10^3}$ $\quad 2 \times 10^{+6}$

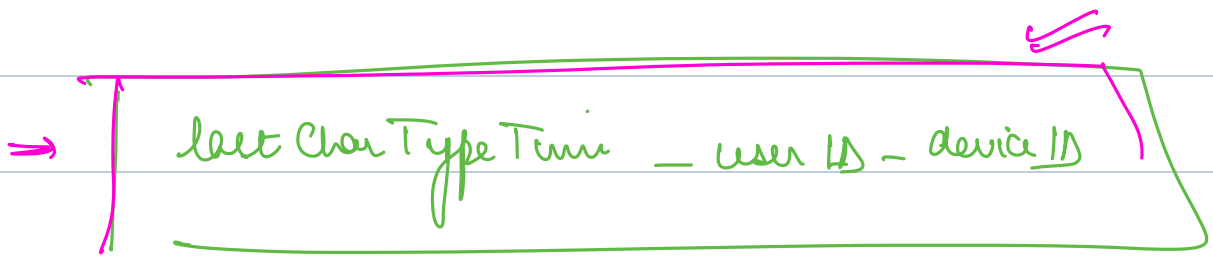
$\rightarrow$ 2 Million



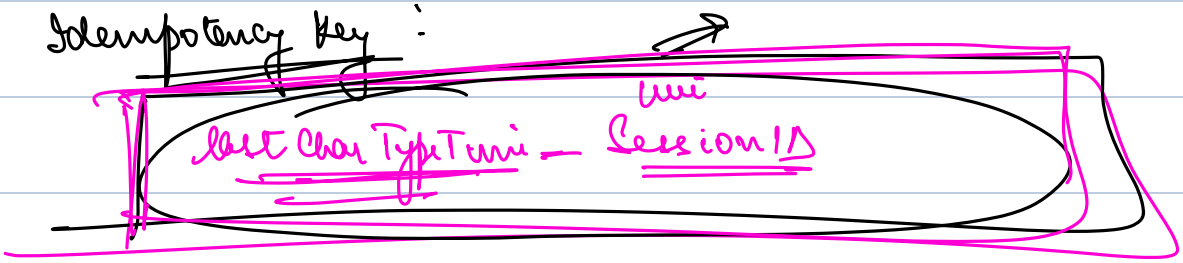


Message ID → last char Type Time — user ID

10137210 —124



Idempotency key :



Server:

if (! idempKey.isPresent()){

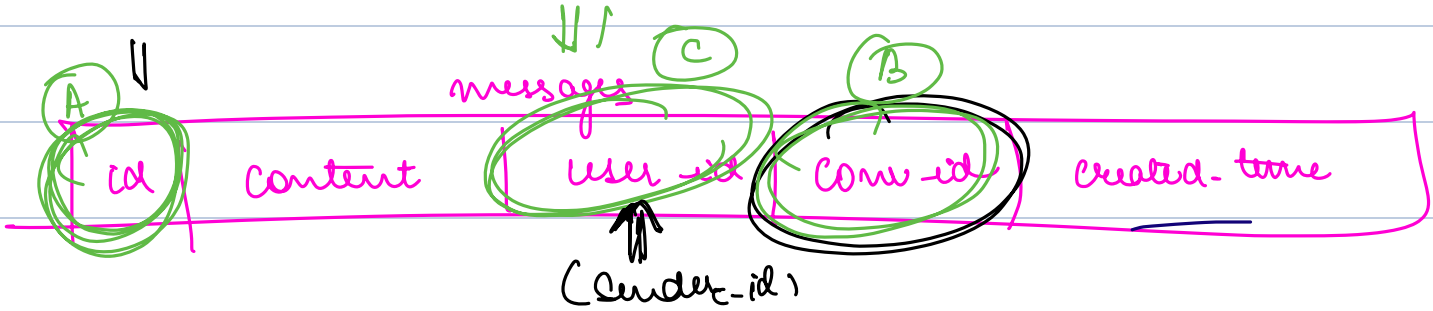
} else {

return;

}

We need Sharding

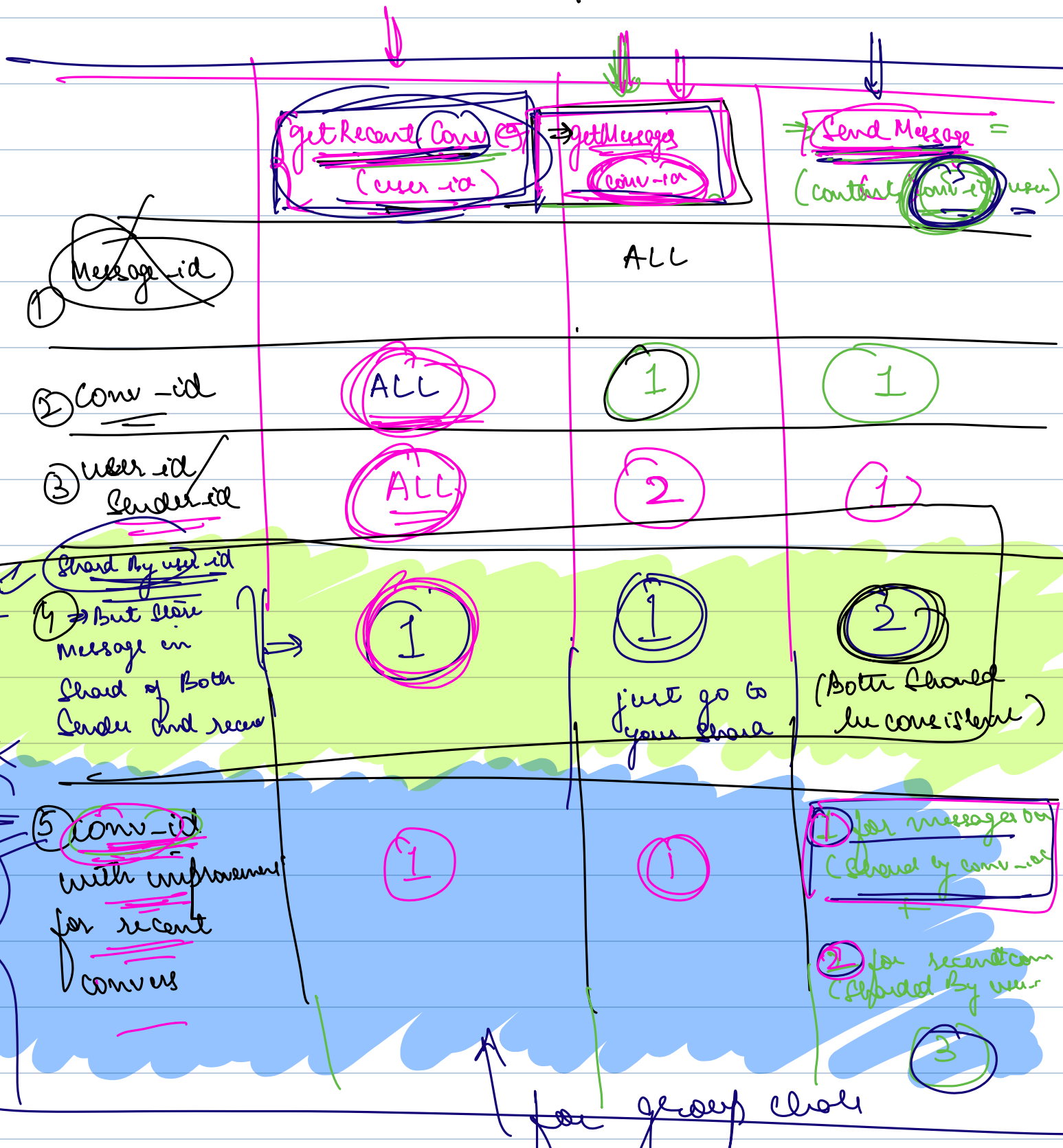
↳ what will be sharding key ?

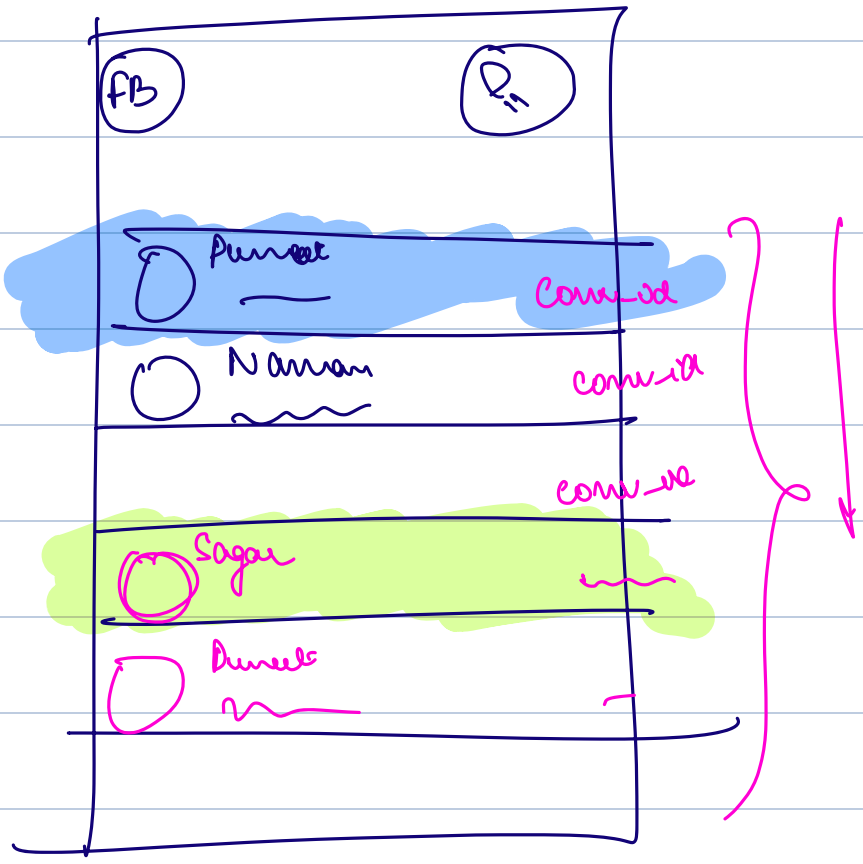


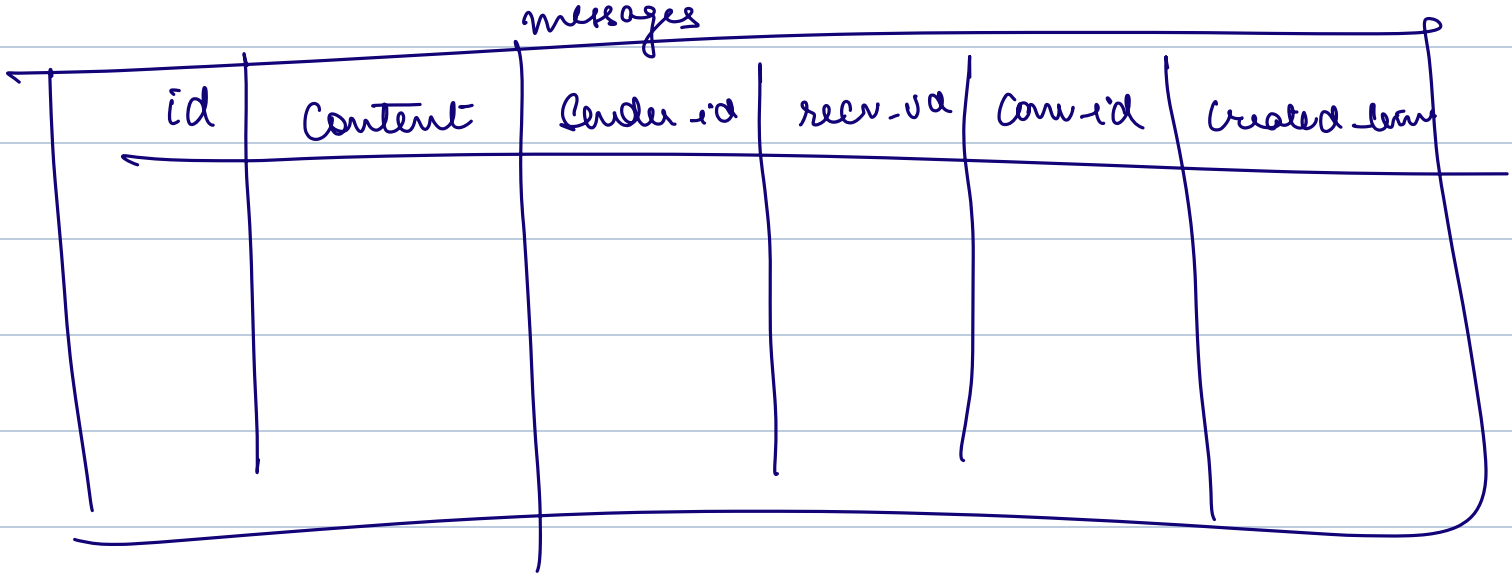
content length

Conversations - users

Conv_id	user_id
101	1
101	2



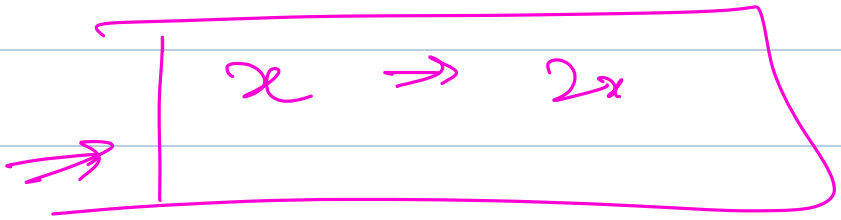


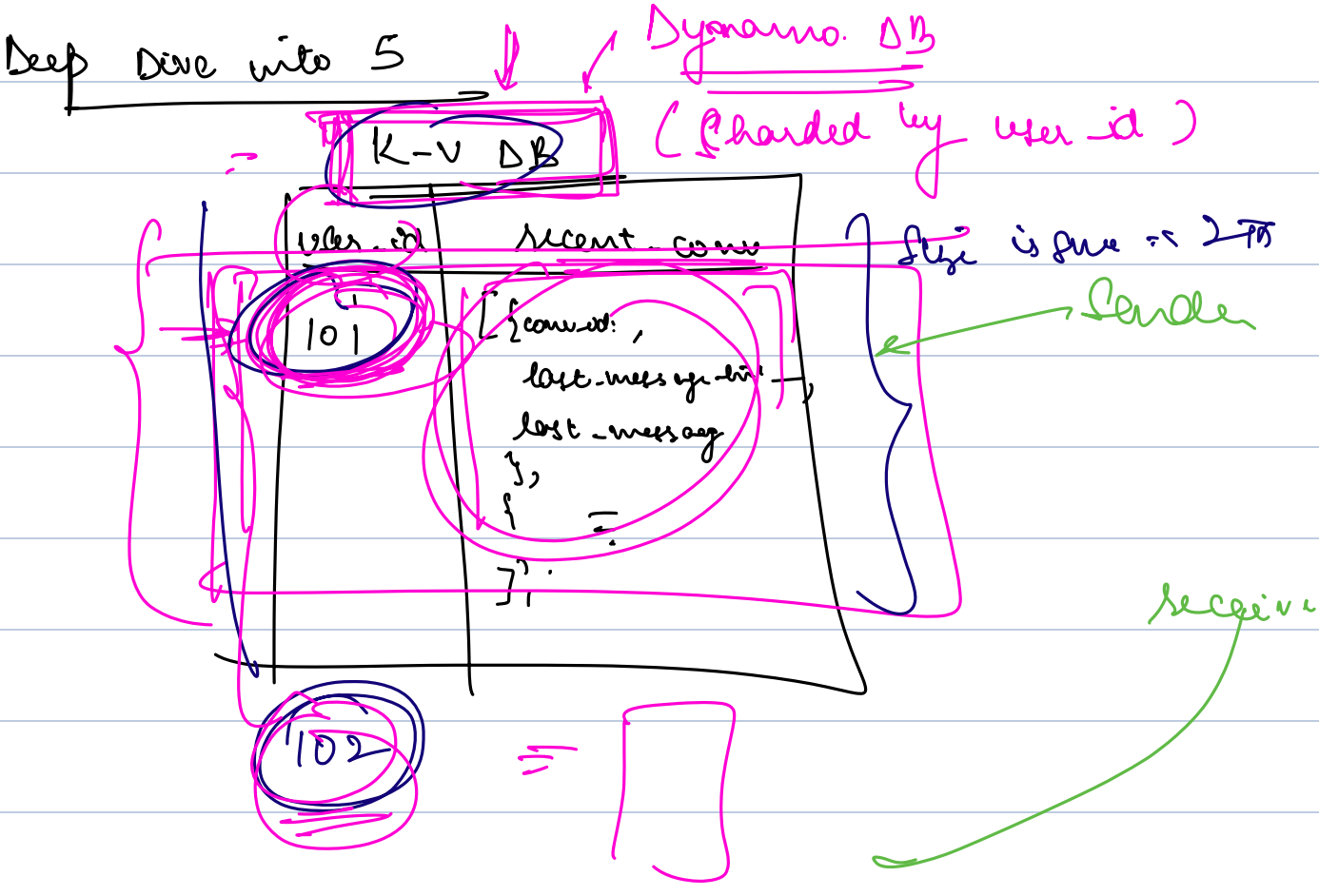


- get conv_id, max(creation_time)
  from messages
  where sender_id = {} or
  recv_id = {}

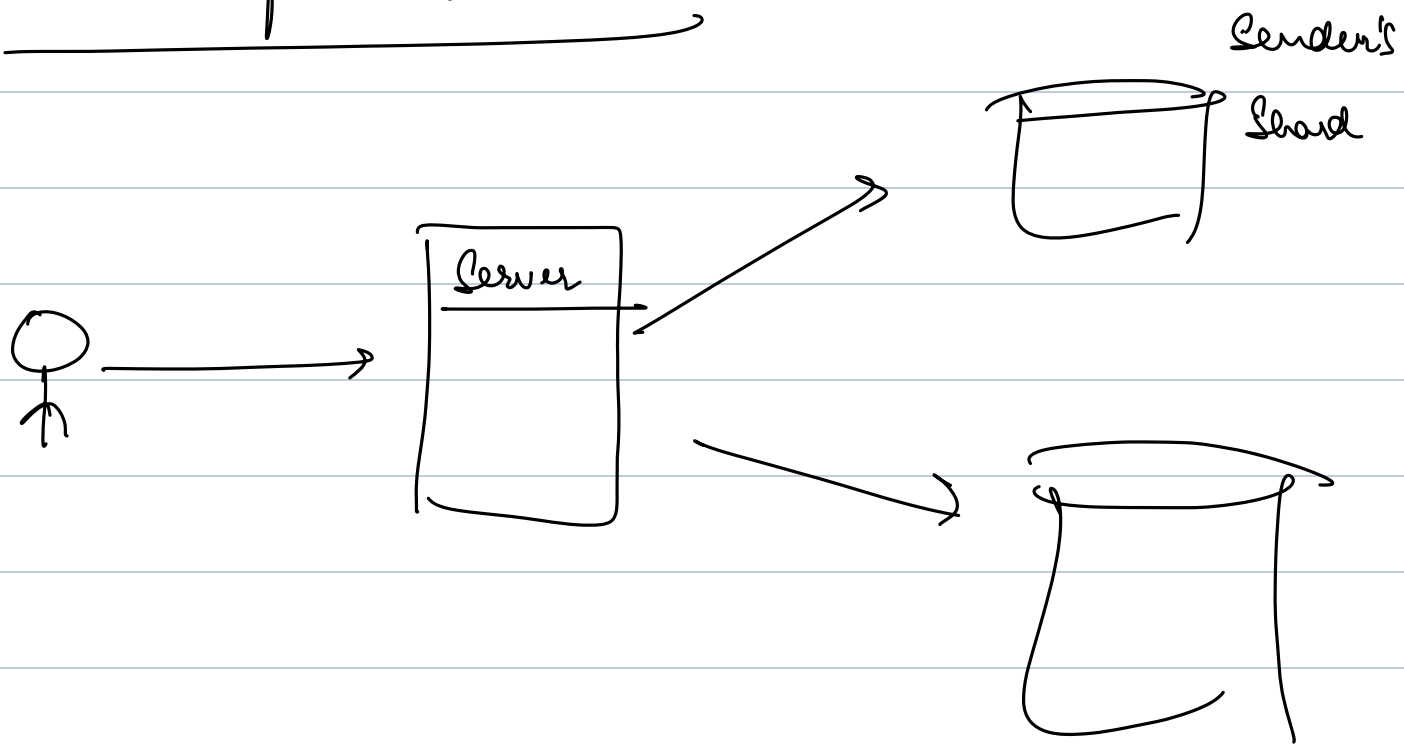
group by conv_id

limit 10

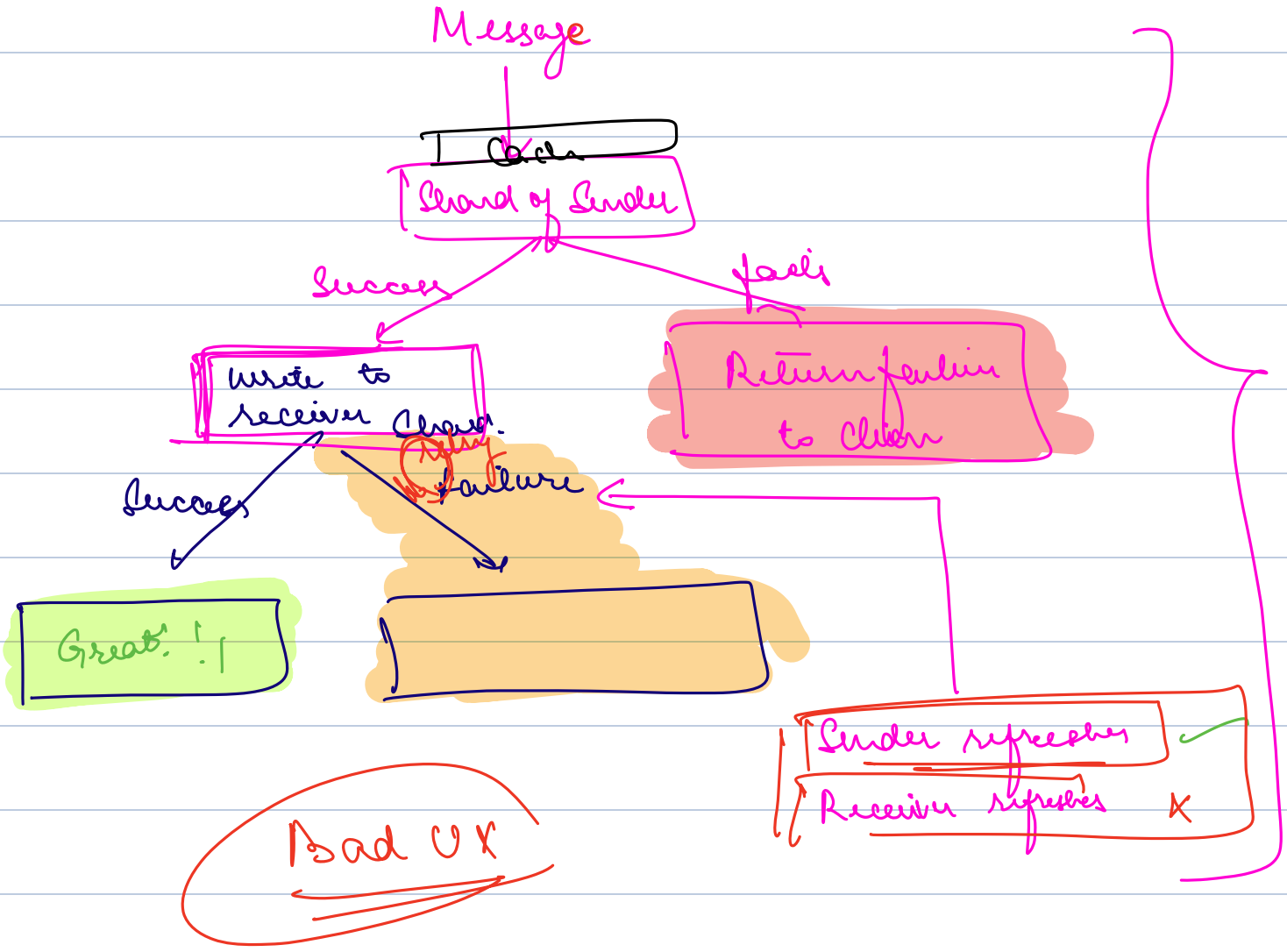


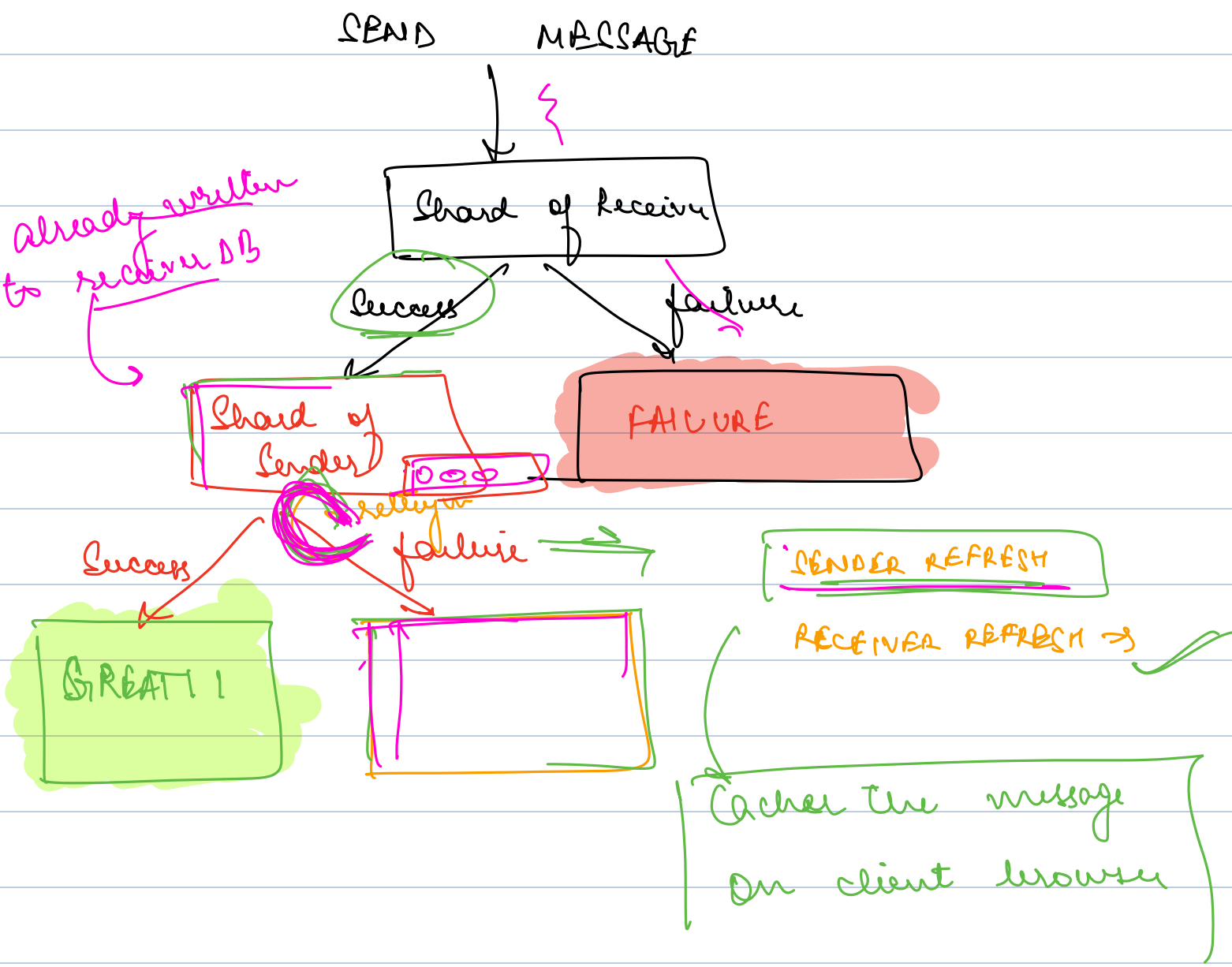


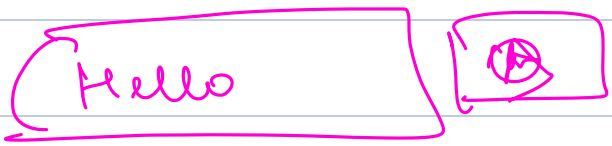
How to keep consistent




```
sendMessage() {
    s1 = connecting(sender_id).
    try {
        s1.save(___ _)
    } catch (Exception e) {
        return fail
    }
}
```


Message
Broad of Sender
Success
fails
write to receiver Channel.
Return failure to Client
Success
failure
Great !
Sender refreshes
Receiver refreshes
Bad UX


SEND MESSAGE
Shard of receiver
already written to receiver DB
Success
failure
Shard of Sender
FAILURE
retry
failure
Success
GREAT!!
SENDER REFRESH
RECEIVER REFRESH
Cache the message on client browser


Hello

Which DB

Both read and write Heavy

Can we reduce writes ?

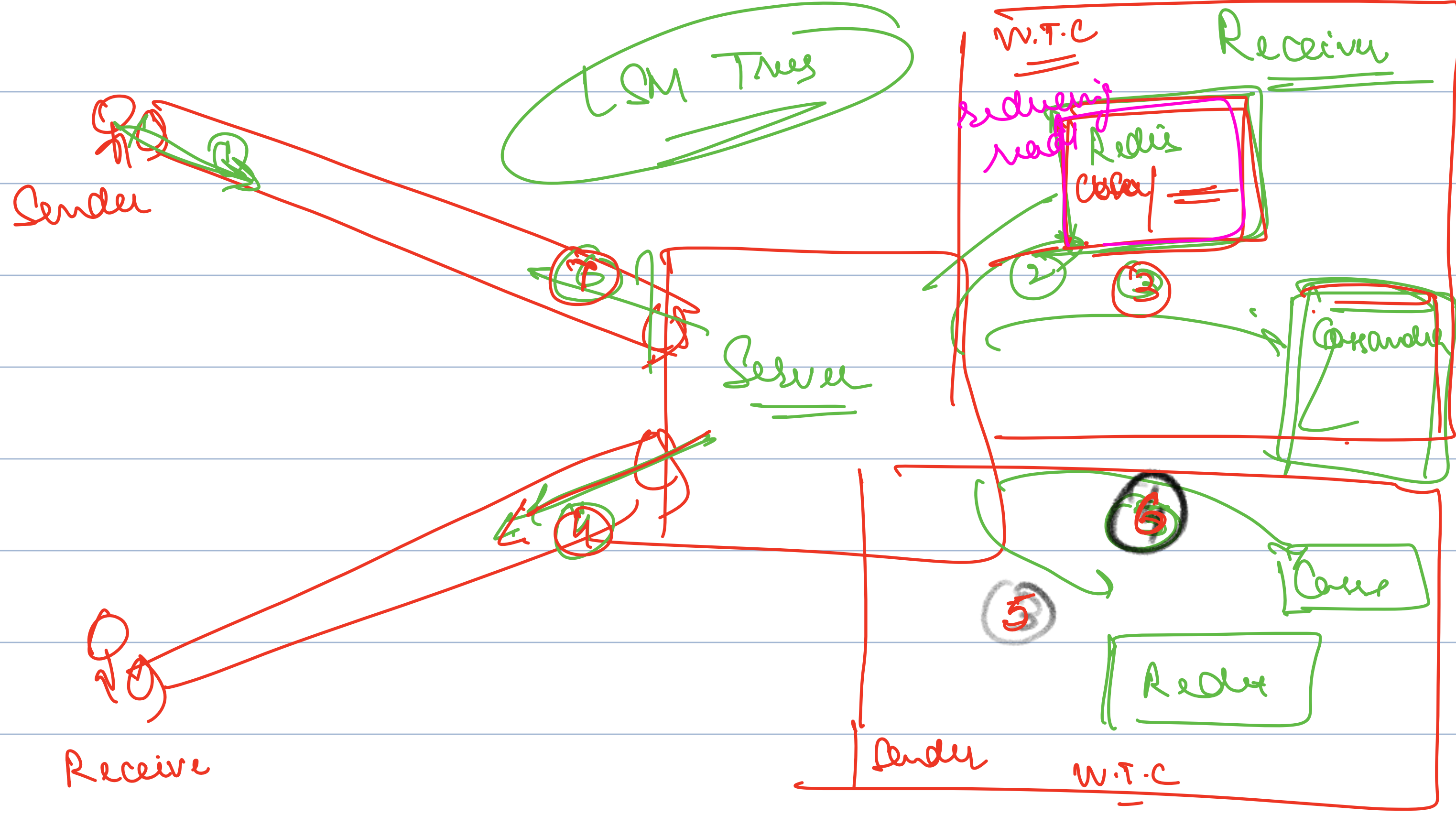
⇒ WRITE HEAVY DB

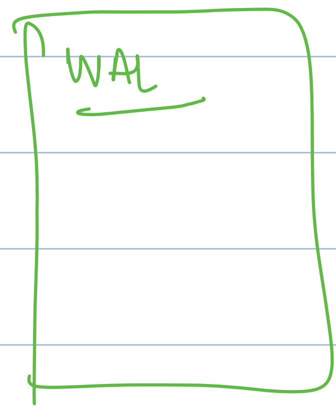
⇒ Cassandra

I have to figure out how to reduce reads

⇒ CACHING

⇒ WRITE THROUGH CACHE



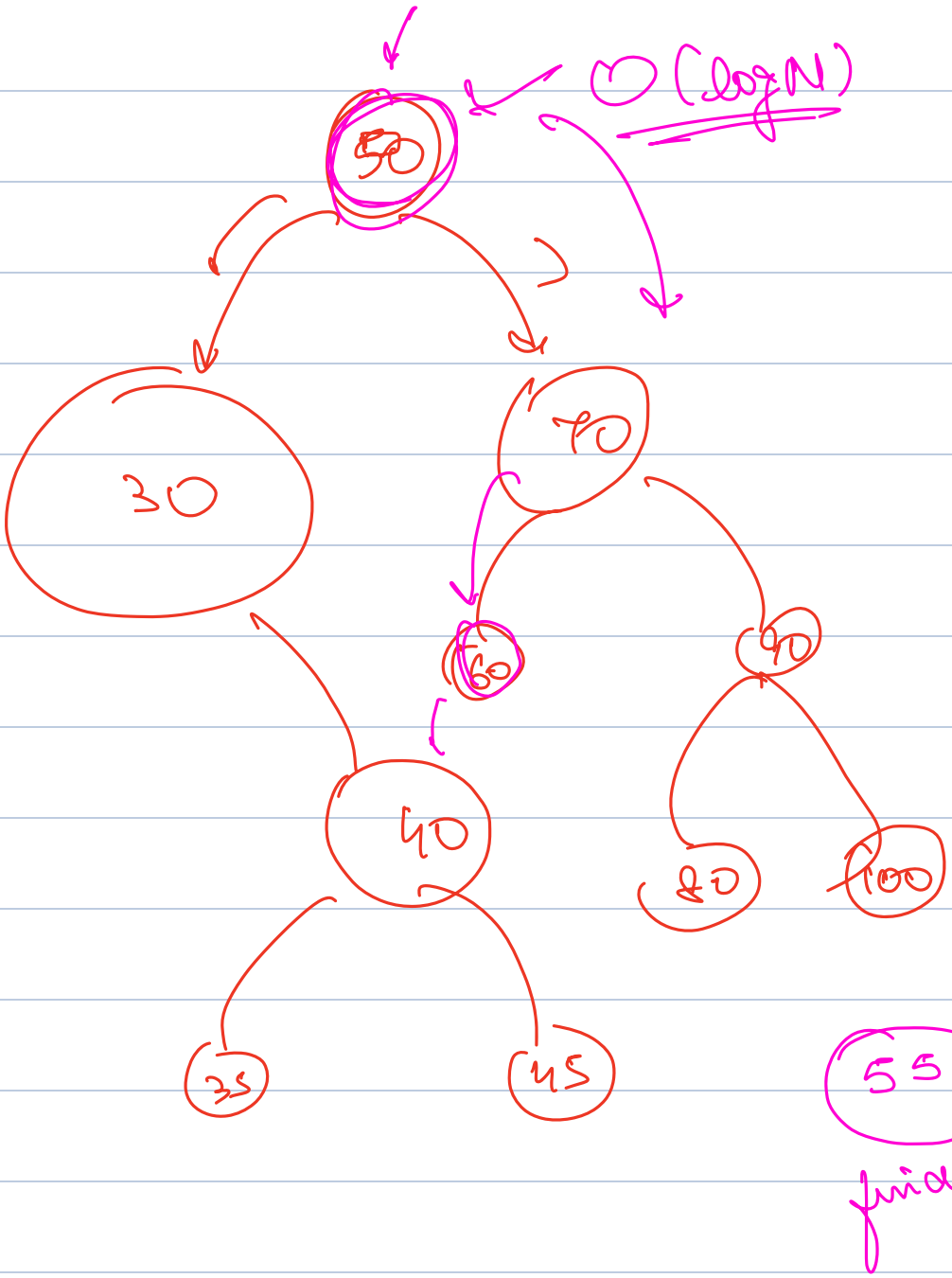


Load

1M conc u

~~Machine~~ Single machine SQL DB

15K concurrent user grow


O(logN)
50
30
70
60
90
40
80
100
35
45
55
find first node
k <= 5