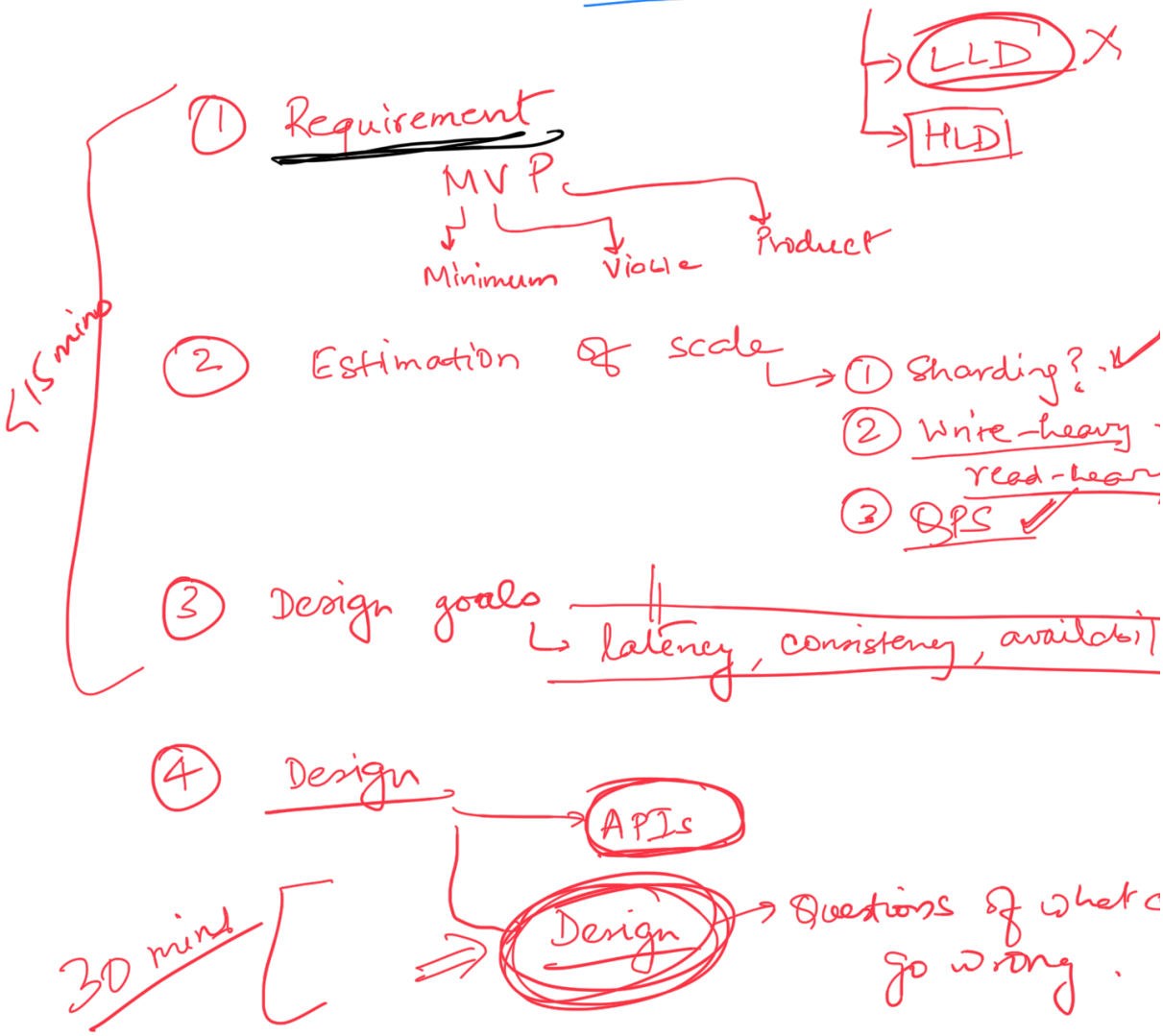
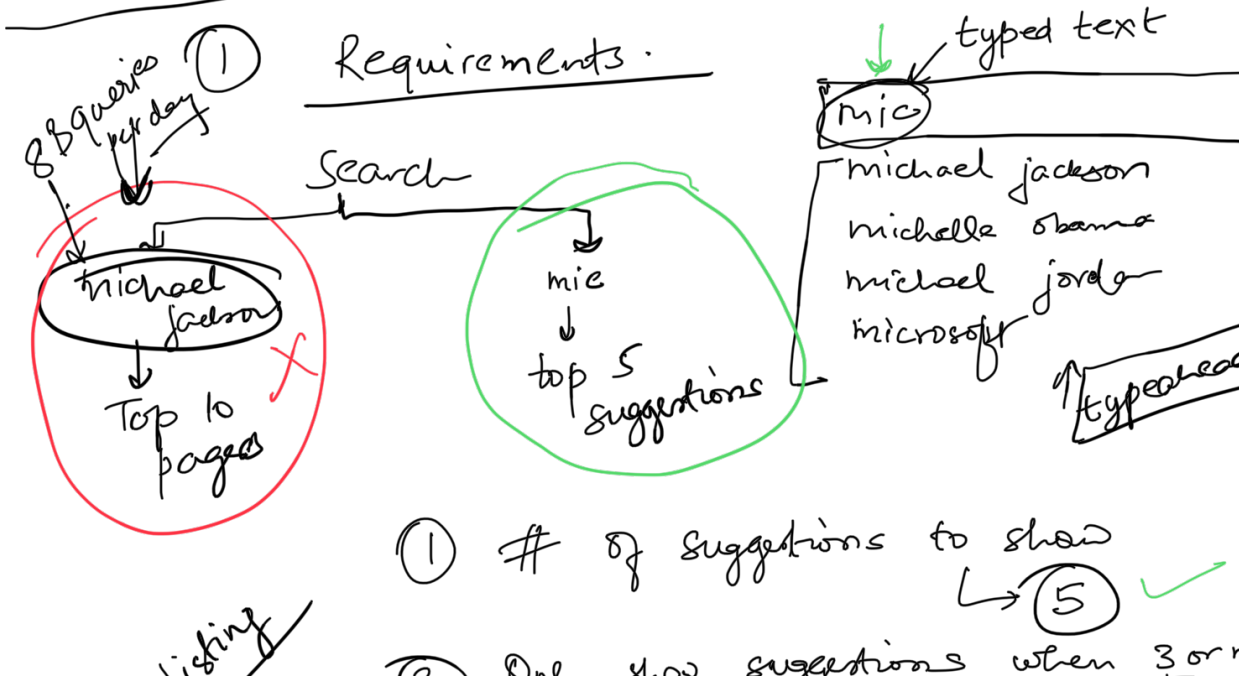
DESIGN SEARCH TYPEAHEAD





Blacklis

characters are typed

Pre-fetch

③ Trends → Yes. we need to acco for trends.

→ No for personalisation

④ Ranking of suggestions / top 5

① frequency of search term

② Trends

⑤ Fuzzy match (spelling mistakes

↳ every suggestion has typed text has prefix

---

② Estimation of scale

8B

michael jackson

searc

8B searc

5 characte

mich

top 5

"top 5

1 search

1 query

2 quer

120B qu

QPS

of read

$$\frac{120 * 10^8}{2^{24} * 60 * 60} \Rightarrow \frac{10^8}{42} \approx 10^6$$

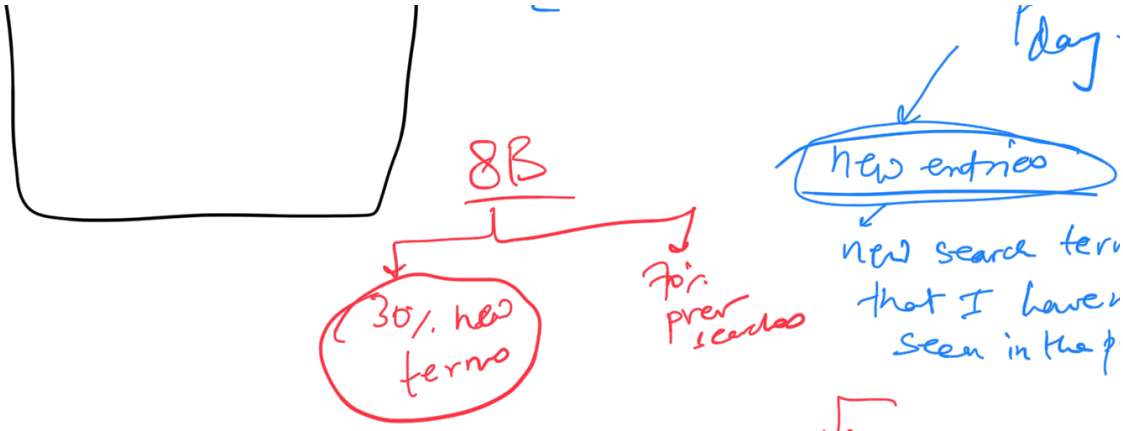
> 1 million QPS

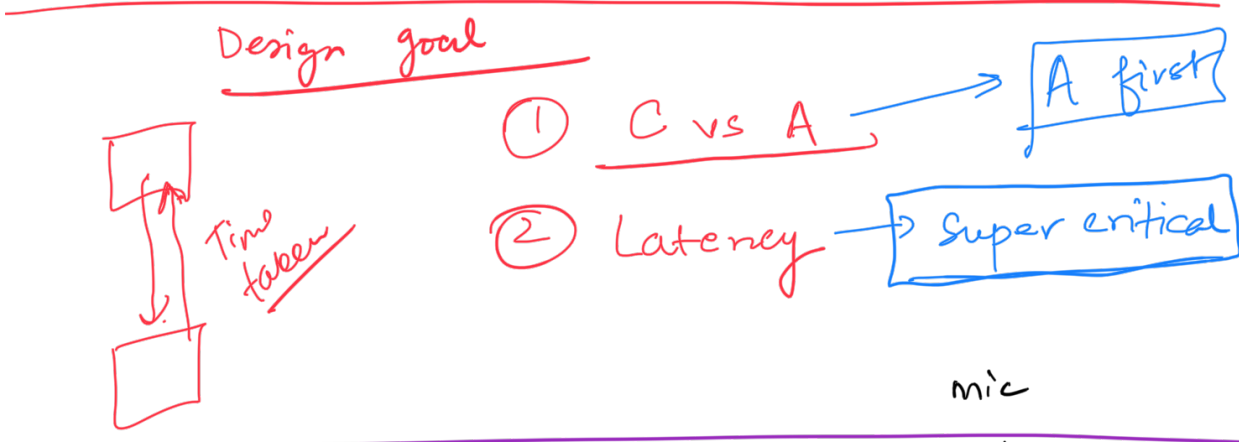
DB

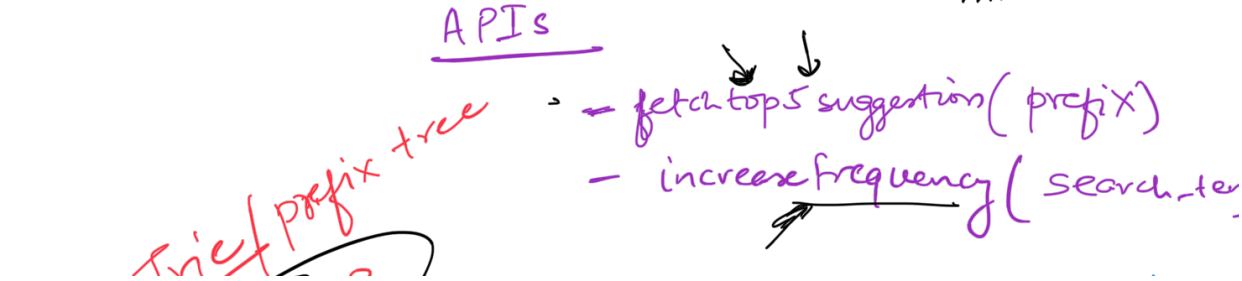
prev_search_queries → frequency

100 → 101

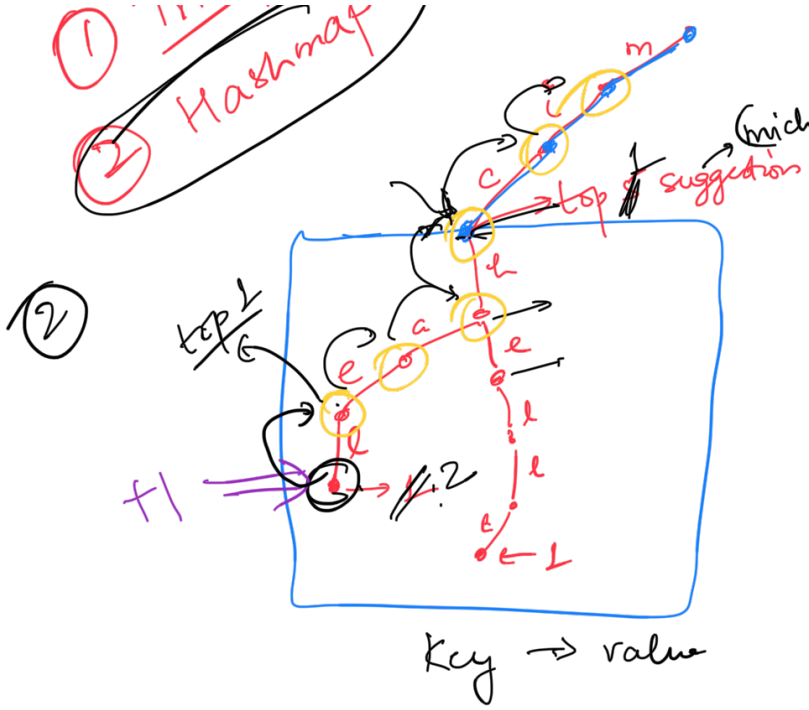
8B searc queries per


8B
30% new terms
for prev searches
new entries
new search ter
that I have
seen in the
8B → 30% of 8B → 2.4B
2.4B * 100 bytes
GB
⇒ 240 GB ← 1 day
240GB * 365 * 10
≅ 1PB
1 Sharding → YES
2 8B writes
120B reads
read heav
3 1 million QPS


Design goal
1 C vs A
A first
2 Latency
Super critical
Time taken
mic
mid


APIs
- fetch top 5 suggestion(prefix)
- increase frequency(search_te
trie/prefix tree

① Trie
② Hashmap



mic

More storag

TRIE ↓ Reduce lat

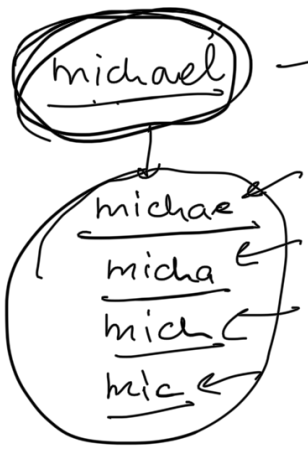
→ Top 5 suggesti on every

→ increase freq → update all ancestors

⇒ search_term ⇒ frequency Key → val

mic → (2, top 5 suggestions with freq)

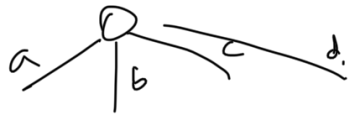
michael → (6, top 5 suggestions)



① Trie
② Hashmap

Trie

- writes are cheaper



aaa
aab
aac

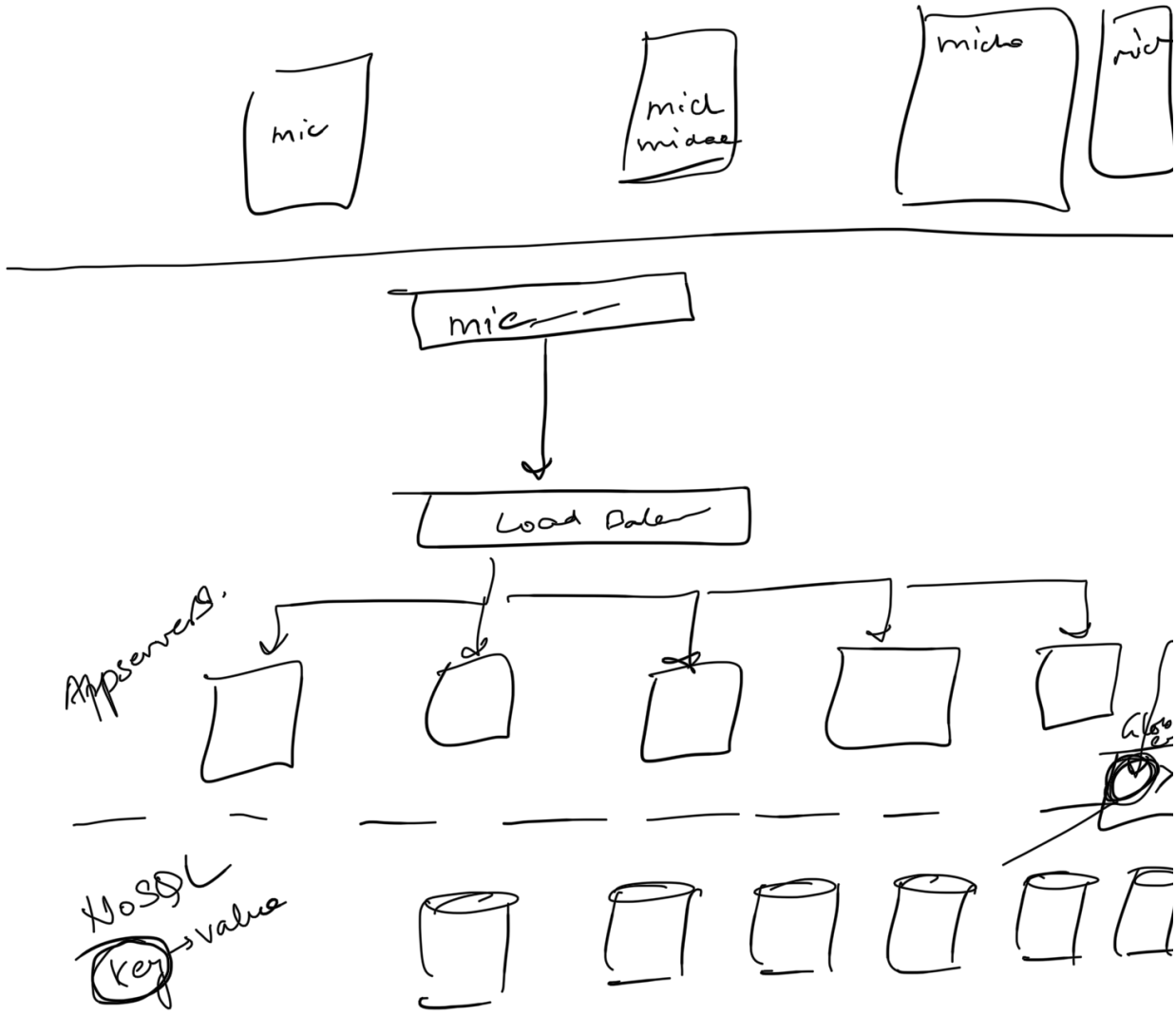
↓ Key → val

→ Hashmap

- Easier to shard
- Solution avai
- Both of bo

writes are expensi

1 write → 10 machi



① Writes are expens

① increasefreq → 10 + mod

Michael → 1

mic
mic

① Batch writes (1 hour)

michael jackson	12
roger fe	1200

Trends

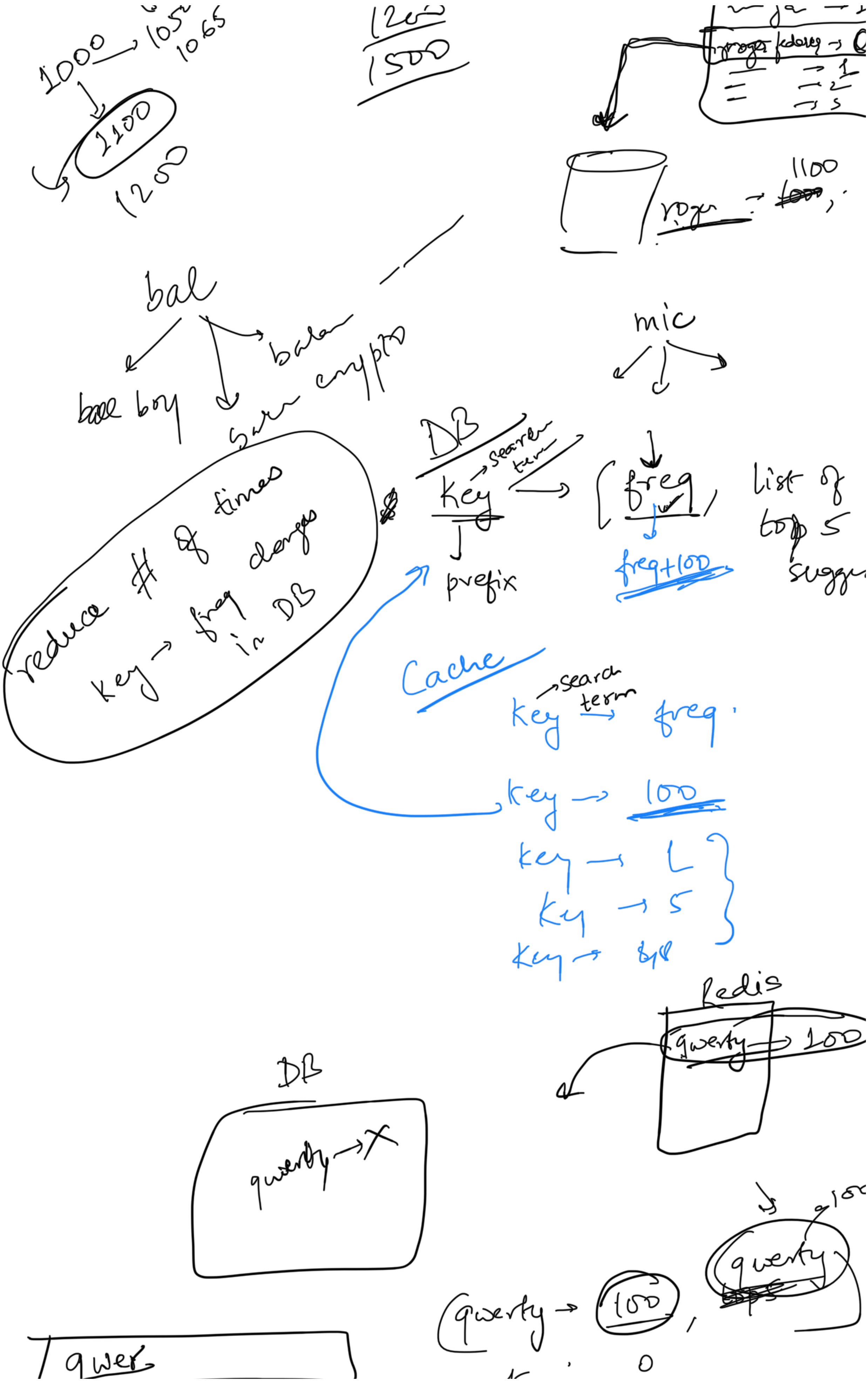
Trendy

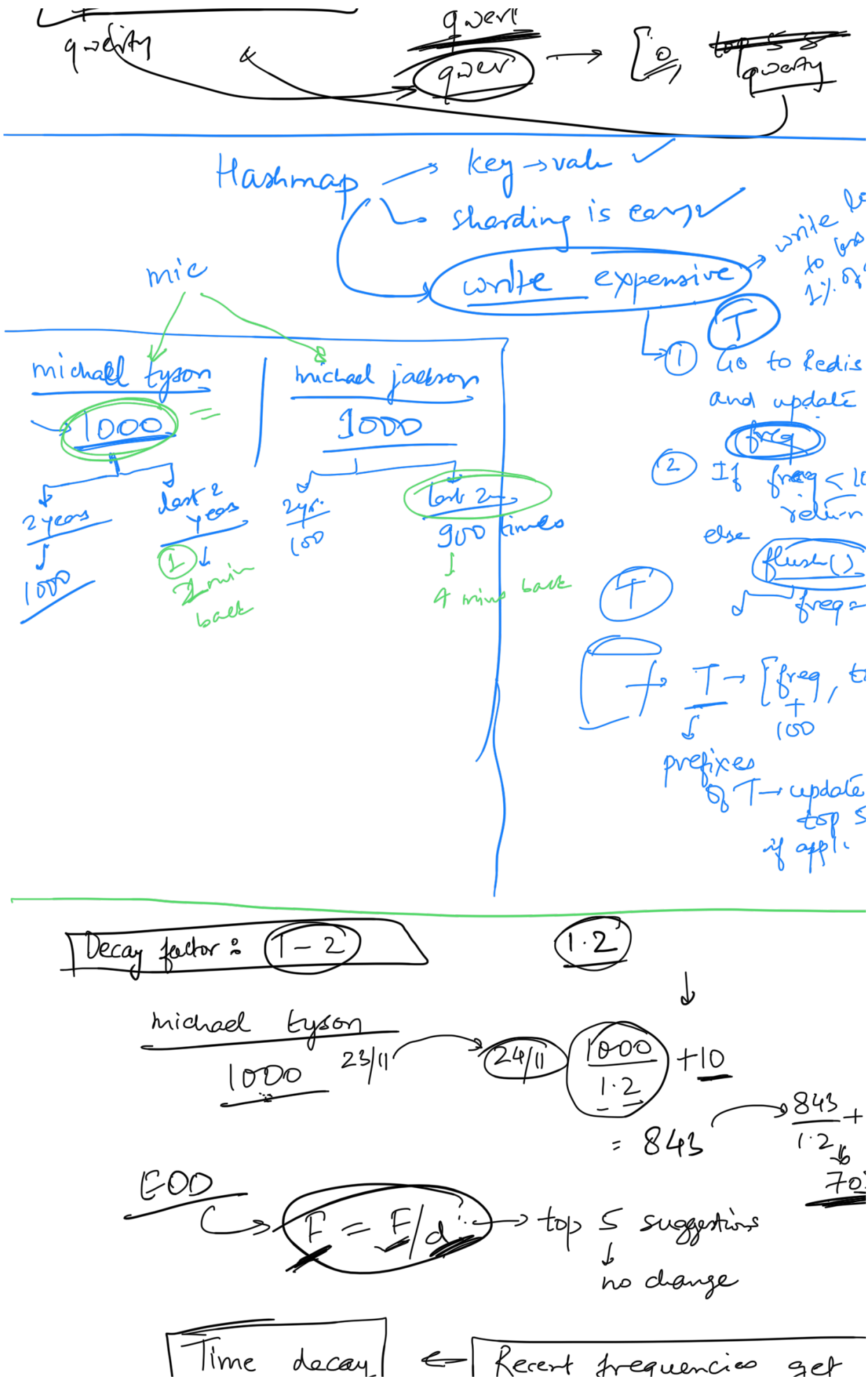
roger federer → ~~100~~ 1000

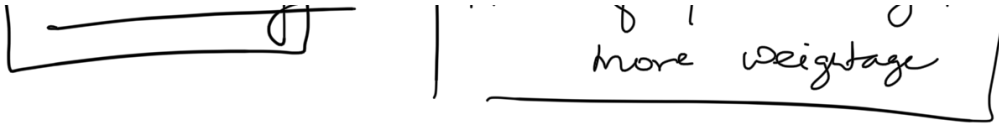
1 2

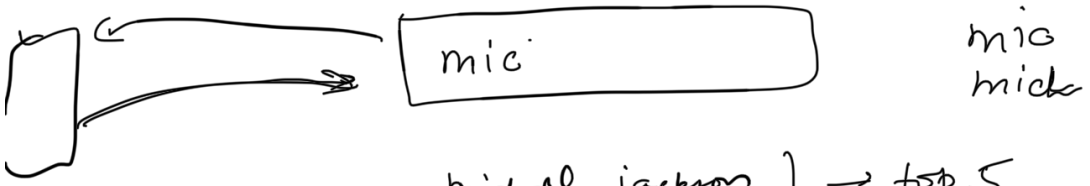
Redis

asca → 1


1000
1100
1200
1500
bal
bal boy
balan
swan crypto
reduce # of times key → freq changes in DB
DB
Key → search term
freq
list of top 5 sugg
prefix
freq+100
Cache
Key → search term → freq
key → 100
key → 1
key → 5
Redis
qwerty → 100
DB
qwerty → X
qwerty → 100
qwerty
qwer


Hashmap → key → val
sharding is easy
write expensive
mic
michael tyson
1000
2 years
1000
last 2 years
michael jackson
1000
24hr.
100
last 2
900 times
4 mins back
Go to Redis
and update
freq
If freq
else
flush()
T → [freq,
+
100
prefixes
Decay factor : 1-2
1.2
michael tyson
1000
23/11
24/11
1000
1.2
+10
= 843
843
1.2
EOD
F = F/d
top 5 suggestions
no change
Time decay
Recent frequencies get





mic
mick

michael jackson
michelle obama
} → top 5

5 entries