Google Typeahead

Agenda

1. How to approach any HLD Problem in an interview
2. Design Google Typeahead.
3. Deep Dives (Problem Statements / Followups) related to Typeahead.
   - How to handle large reads & writes at same time
   - How to handle recency for suggestions

How to approach any HLD Problem in an interview

(45 Mins)

0. Overview (Optional: only if you haven't heard about the interview wants you to design)

Min^m Viable Product

1. Functional Req → MVP features that should be supported

by the design you will come with

(Core Features)

(4-5 Core features)

2.) NON FUNCTIONAL REQUIREMENTS

(how those features should work)

for every feature

⇒ Consistency

⇒ Consistency

3.) BoE Calculation

(Back of Envelope)

[Estimation of Scale]

Approx

Metrics

1.) QPS / RPS : Queries / sec

Req / sec

2.) Storage

50 GB

5 PB

RDB

Any other

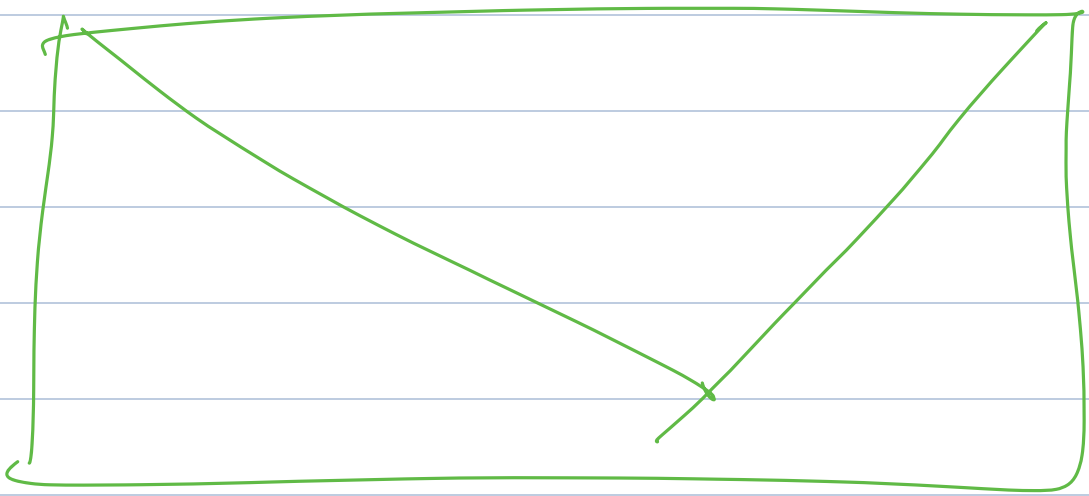
Do we need Sharding?

files / images / video

3. N/W bandwidth (optional)

↳ uploading fees

DB ⇒ 1TB can be in 1 Machine



10000 users / day

$$\frac{10000}{24 \times 60 \times 60} \rightarrow \frac{10000}{86400} \approx 100000$$

# of sec in a day = 10^5

# of days in an year = 400

$10^9 \Rightarrow 1\ GB$

$10^6 \Rightarrow 1\ MB$

$10^3 \Rightarrow 1\ KB$

$10^{12} \Rightarrow 1\ TB$

why ?

→ Sharding

→ is my sys read heavy

or write heavy

or both

4.) API Design

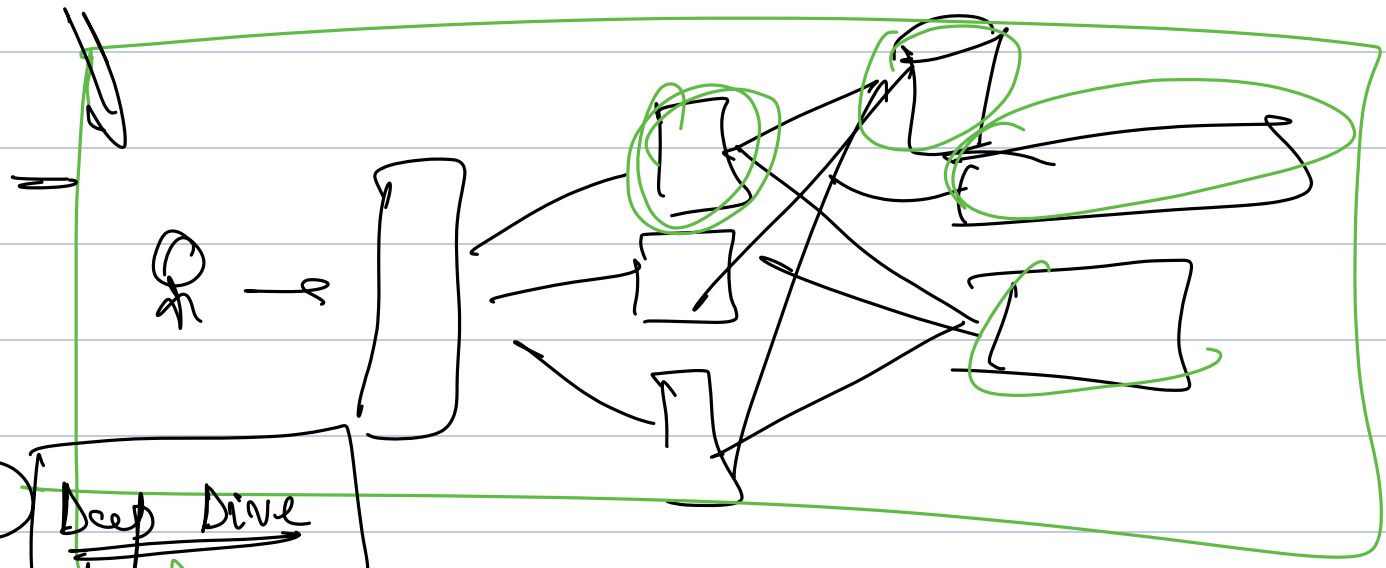
→ list down the APIs that your sys has to support

get Suggestions( )

5.) **Design**

↳ block diagrams

↳ etc



6) **Deep Dive**

Follow ups related to design decisions

↳ Deep dive on certain parts of sys.

0.) Overview

1.) F^n Req

2.) Non F^n Req

3.) BoE Calculations

4.) API Design

5.) HLD

6.) Deep Dive

⇓

40 Min

Design Google Typeahead

Overview

F^N Requirements

meat

reality

1.) given a string get sugg^n based on that

2.) Suggestions based on prefix

~~3.) spellcheck~~

~~4.) personalization~~

5.) 10 suggestions every time

weather

6) sugg^ns should be the top frequent queries with the prefix above

7.) recency factor

Sac

Sac

lin tendulkar

K

ufic

lin Bowne

red Games

10 most occurring ones

Sac

hin Tendulka

Sac

Donald Trump

Till 2016

donald duck

1995

2017

donald

F^n Req

(4-5 Mins)

1. get Suggestions for Prefix
2. 10 Sugg^n
3. Recency Factor

Non F^n Req

Con 4/5 Avail

Cons 4/5 low latency

What corp means?

ind

india 1031

india 1032

india

ind

india

indic

---

(3) BOE Calculations

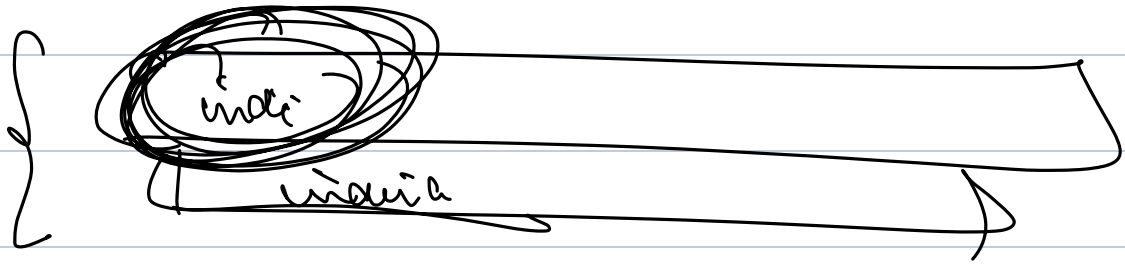
"Guesstimate"

india

DAU = 2B

Or # searches / person / day → 10

= # searches → 20B



"i"

"in"

"ind"

"indi"

avg # of chars types b/f search = 6

→ # of search sugg^n / day

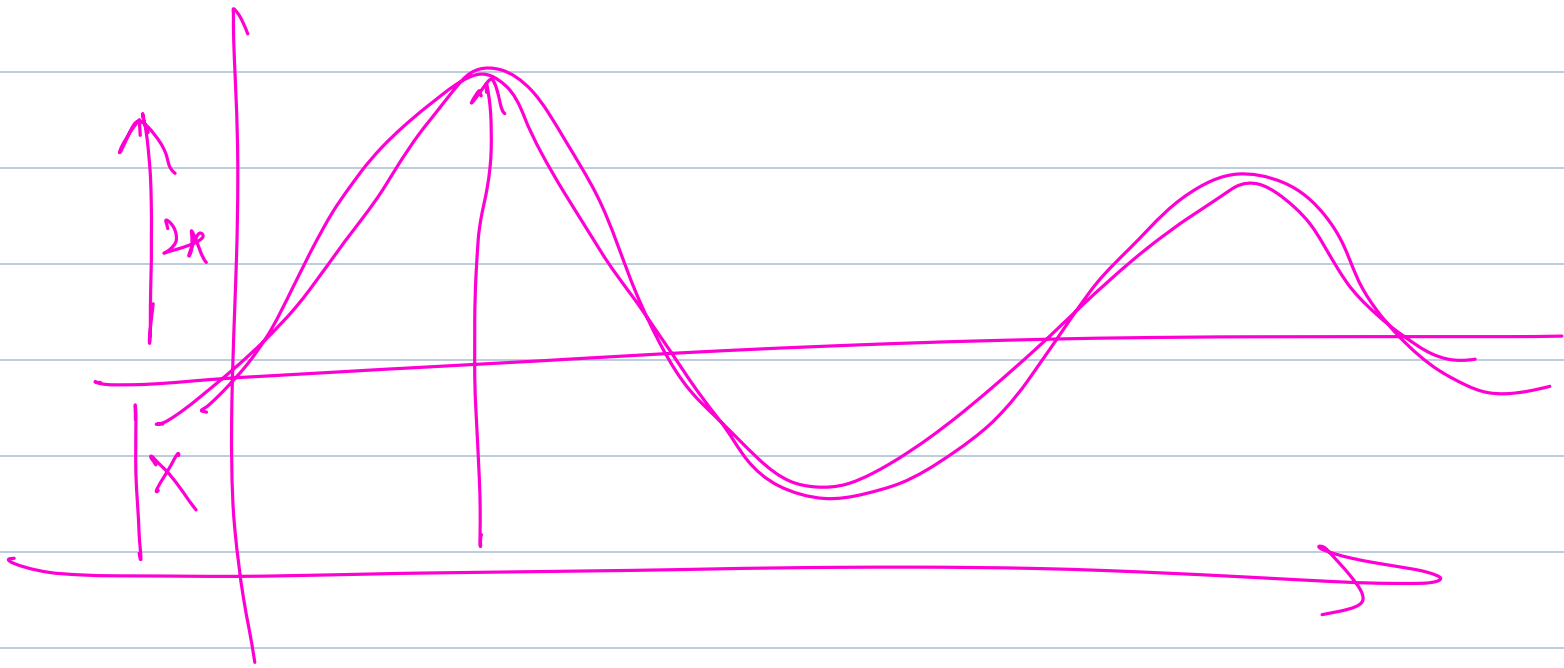
20B × 6 = 120B

# of search sugg^n / sec → $\frac{120 \times 10^9}{10^5}$

→ 1200000

→ 1.2M / sec

assume peak = 2x average → 2.4 M / sec


2π
X

w

| Weather

Ahmedabad

10 %

QPS 2.5 M | Reads

Write : when search happens

→ 20B / day

→ $\frac{20 \times 10^9}{10^5}$ → 200,000

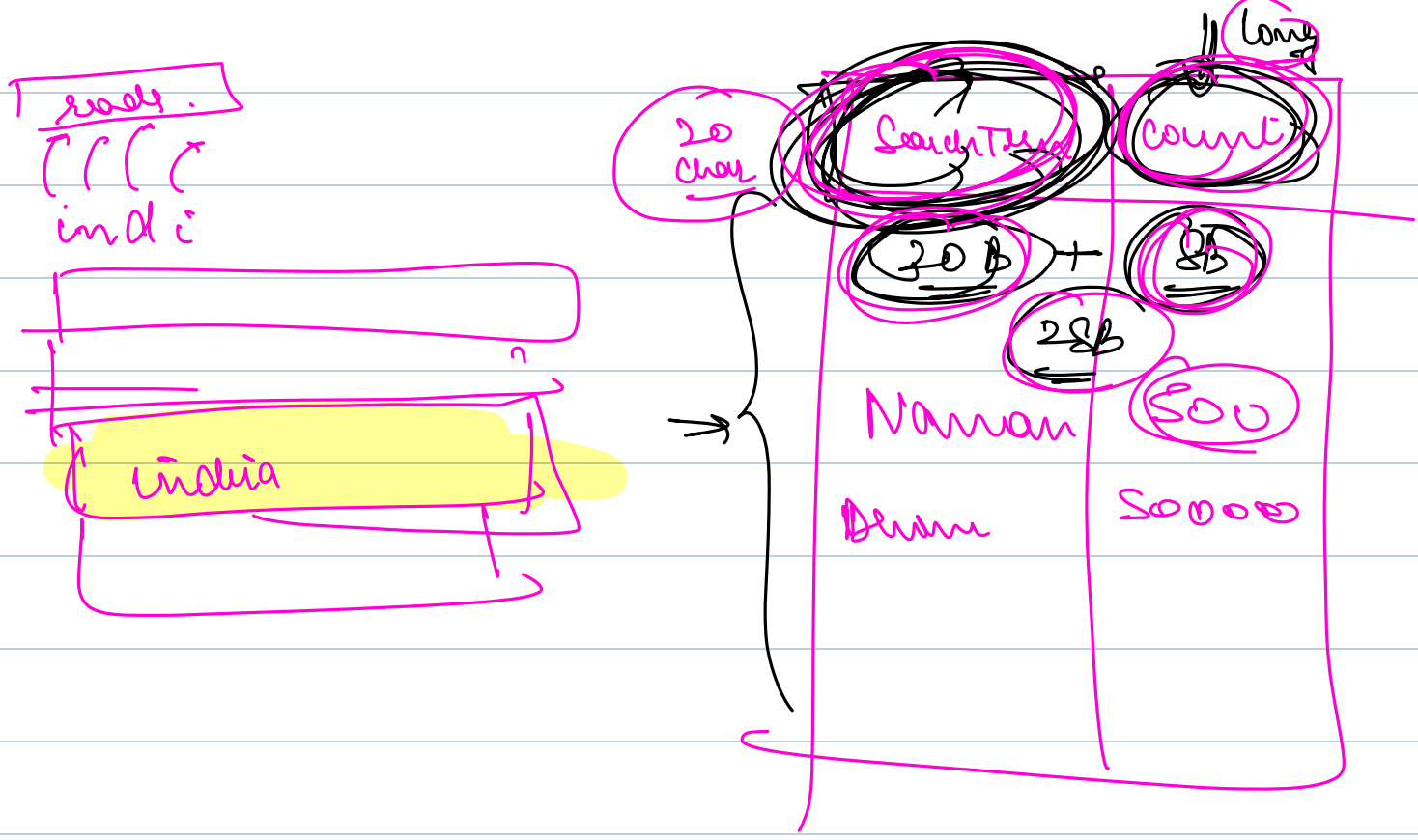
→ 200 k / sec

peak = (2x) = 400 k / sec Write

Typeahead

Read → 2.5 M / sec

Writes → 500k / sec



Storage

⇒ 28B Search Term

1 day ⇒ 20B

⇒ 10% unique ⇒ 2B terms / day

$\boxed{\text{we should plan for 5 years}}$

$2B \times 5 \times 400$

$\Rightarrow 2B \times 2000$

$\Rightarrow 2 \times 10^9 \times 2 \times 10^3 \Rightarrow \boxed{4 \times 10^{12}}$

4T each term in 5 years

4×10^{12} ~~x 8B~~ 50

$\Rightarrow 200 \times 10^{12}$ B

200 TB

A C U ME

→ Sharding

Cs more than 1 TB

BoE Calc

Sharding ✓

Read v/s Write

500k Write

2.5M Reads

→ W : R
1 : 5

Read Heavy

Bolt

1 : 20

5 : 100

Will Start at 9:10AM

Contd. Design Google Type Ahead

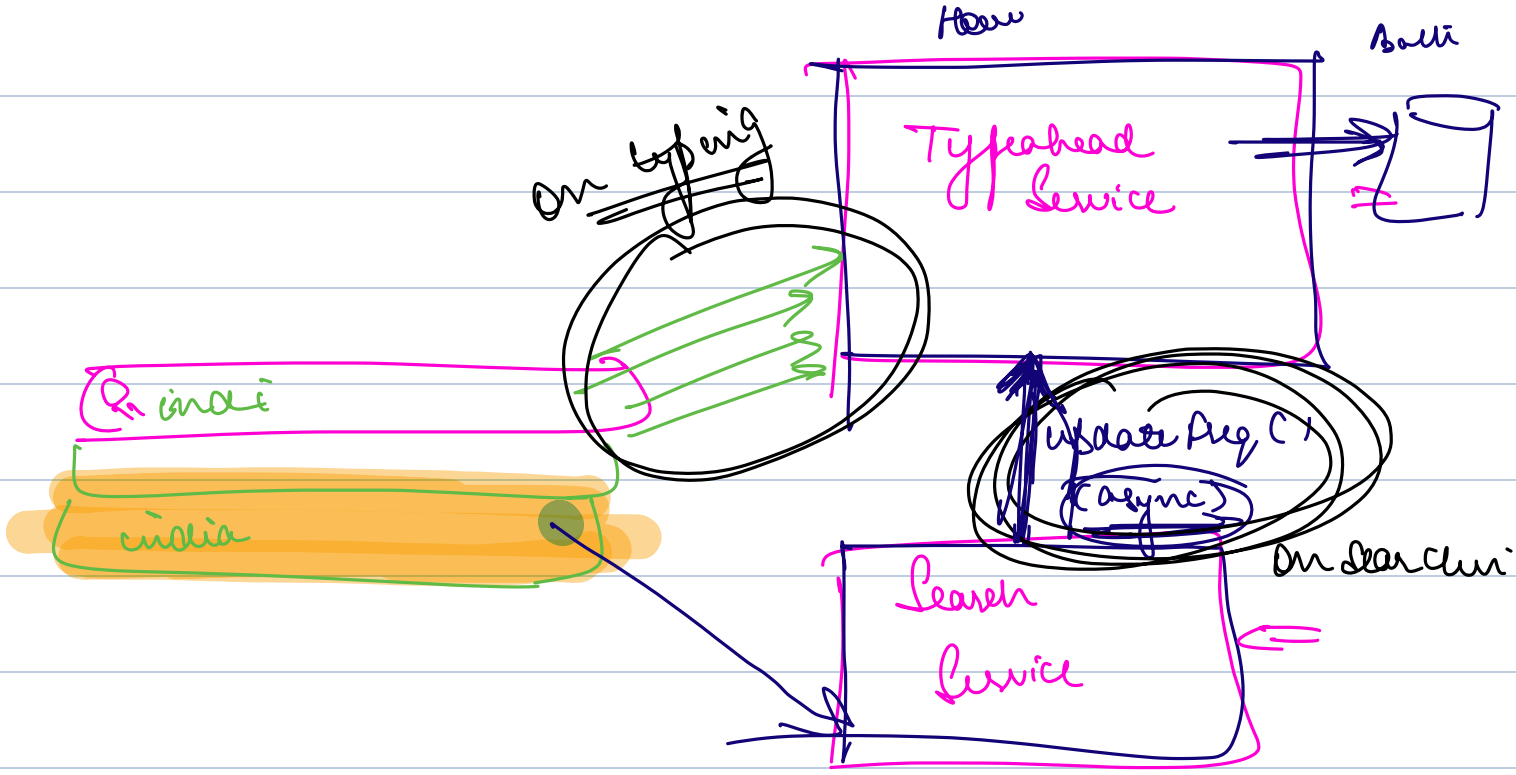
API DESIGN

① List <Suggestion> getSuggestions(String prefix)

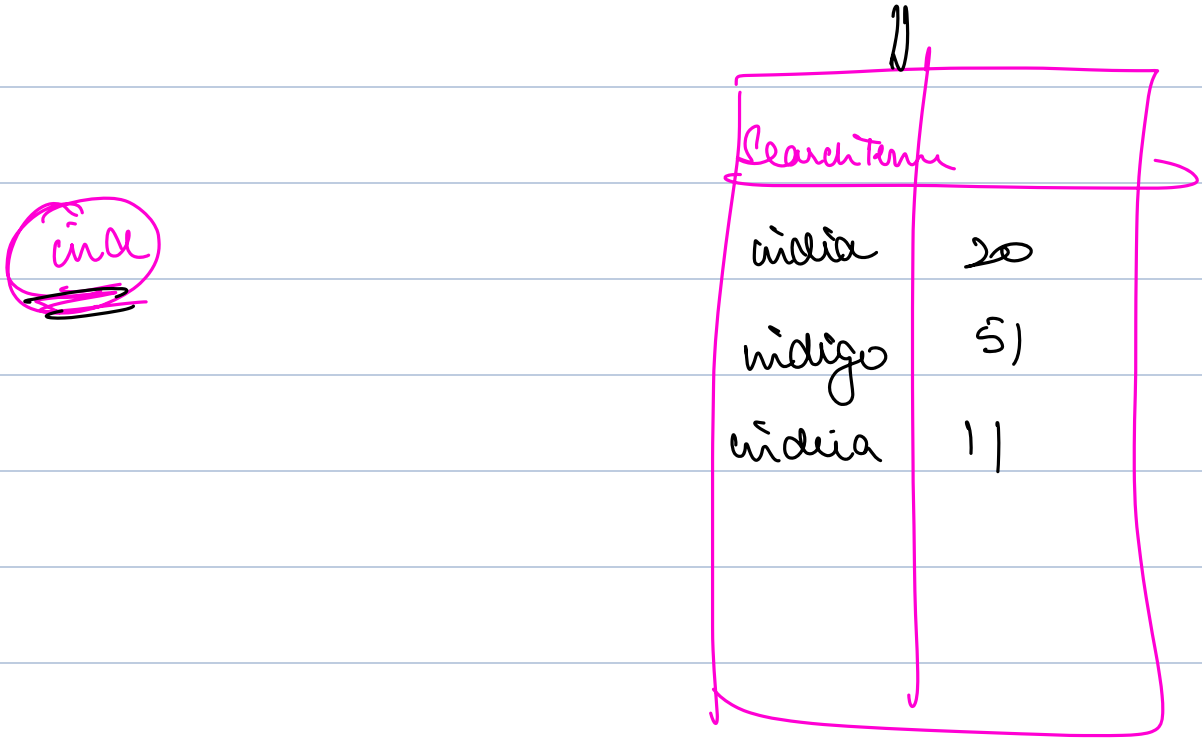
② void updateFreq(SearchTerm)

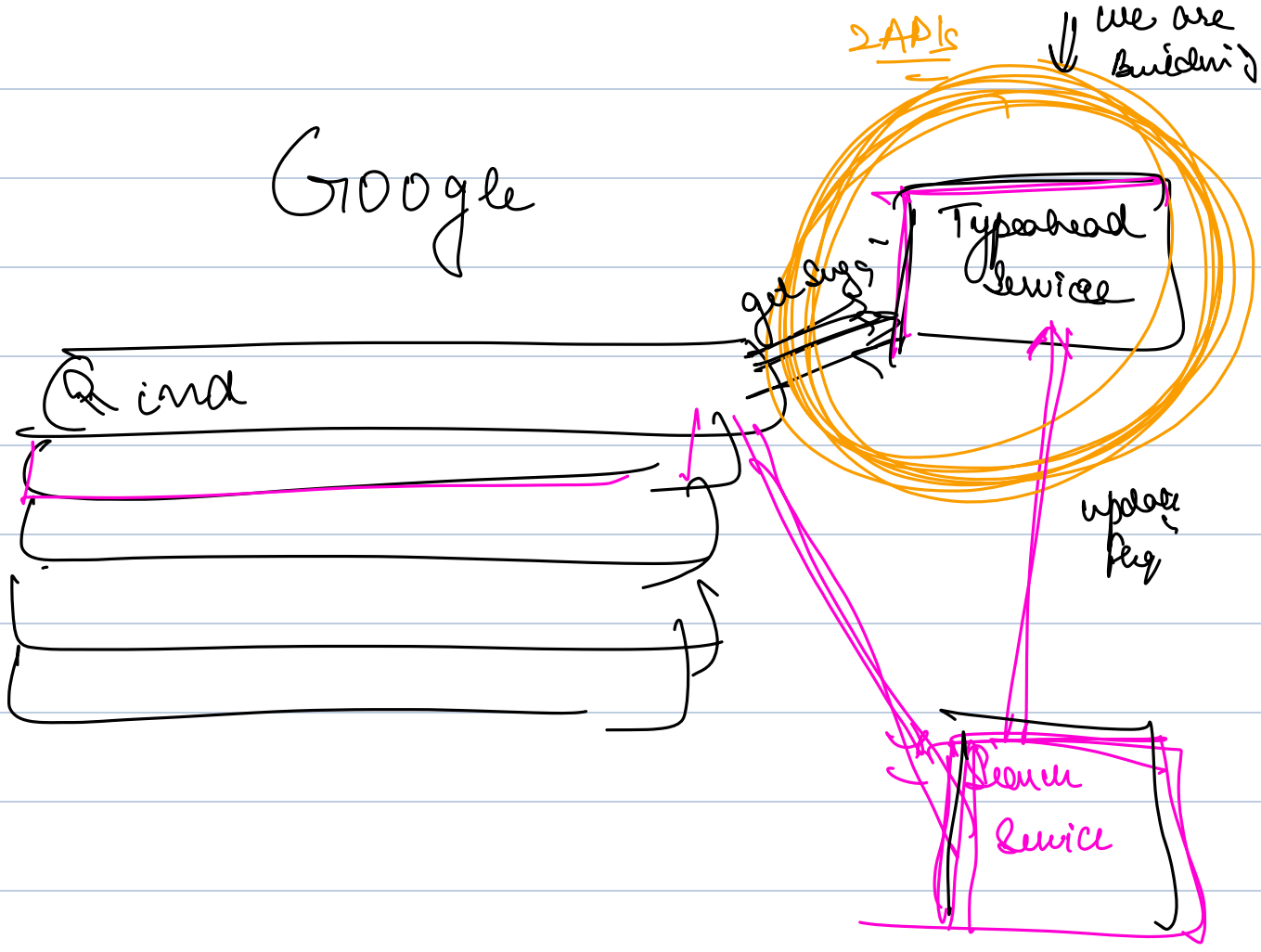
Q in

~~Searches~~


Typeahead Service
on typing
Q. indi
india
Search Service
(async)

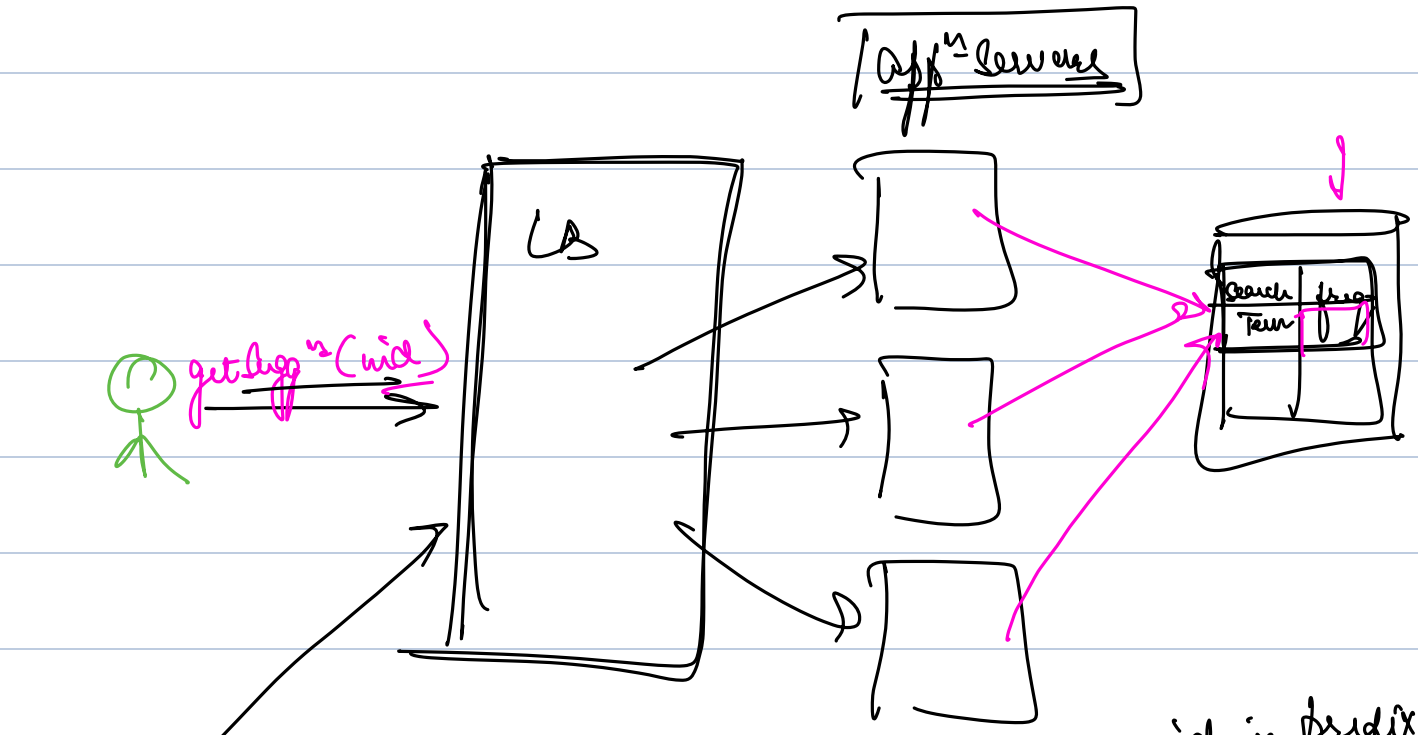
in
ENTER


indi
Search Term
india 20
indigo 51
india 11



1. Get Suggestions (prefix)
2. updateFreq (Search Term)

DESIGN



uid is prefix

getSuggⁿ(prefix)

Search Service

N log N

select *
from terms
where term like "uid %"
order by freq desc
limit 10;

O(N)

updateFreq(searchTerm)

update terms
set freq = freq + 1
where term = "..."

O(N)
unless we index "term"
O(log N)

This is going to be v.v. slow

→ get Sugg^{n} we need sub ms latency

→ O(N) algo won't have that

OPTIMAL SOLN

Search Terms → String

queries

prefixes

Searchterm

updateFreq(String)

getSuggestions(String)

prefixes

HashMap

freq Map (HashMap)

k → searchTerm, v → freq

searchTerm	freq
india	1

O(1)

```
updateFreq (term) {
    if(!map.contains(term))
        map.put(term, 0)

    map[term] += 1
}
```

get Sugg^n

⇒ we can't afford to compute sugg^n on fly

→ PRE COMPUTE

in memory

2 Maps

$10^x 20 \Rightarrow$ 200B

sugg^n ac

India =

Freq Map

term	freq
India	1

Suggestions

prefix	list<sugg^n>
8 x 10^B	
10TB	

Pro: getSugg^n $\Rightarrow$ O(1)

Con: EXTRA STORAGE

5 year

1 day $\Rightarrow$ 20B searches

10% unique $\Rightarrow$ 2B unique term

5 year $= (2 \times 10^9) \times (5) \times (400)$

$\Rightarrow (2 \times 10^9) \times (2 \times 10^3)$

$\Rightarrow \boxed{4 \times 10^{12}}$

20 chars

80×10^{12}

4TB Disk → 50 USD

$80 \times 10^{12} \times 200$

16×10^{15} → 16 PB

$\frac{16000}{4}$ → 4000

4000 × 50 $

20 9000 $

Salary of 1 SWE for 6 months

COCTS ⇒ SWEs

O(1)

```
get Sugg^n (prefix) {
    return
        Sugg^n Map. get (prefix);
}
```

```
update freq (term) {
    freqMap[term] += 1                    ⇒ O(1)

    for all prefixes of term.
        cur lastTerm = seq[prefix][-1]
        if freqMap[term] > freqMap[curlast]
            seq[pref][-1] = term
}
```

law of ln

O(K)

O(1)

top 3

indigo

⇒ india

lal

i = [indigo, injury, italy]

10 log 10 = 10

in in = [indigo, injury, ~~interpol~~]

india

when we search for something, suggⁿ of all prefixes of term may change.

⇒ for all prefixes of term:

if freq of searchTerm > prev suggⁿ of prefix :

updateSuggestions (prefix)

update freq ⇒ O(1)

get suggⁿ ⇒ O(1)

But unfortunately they aren't in RAM

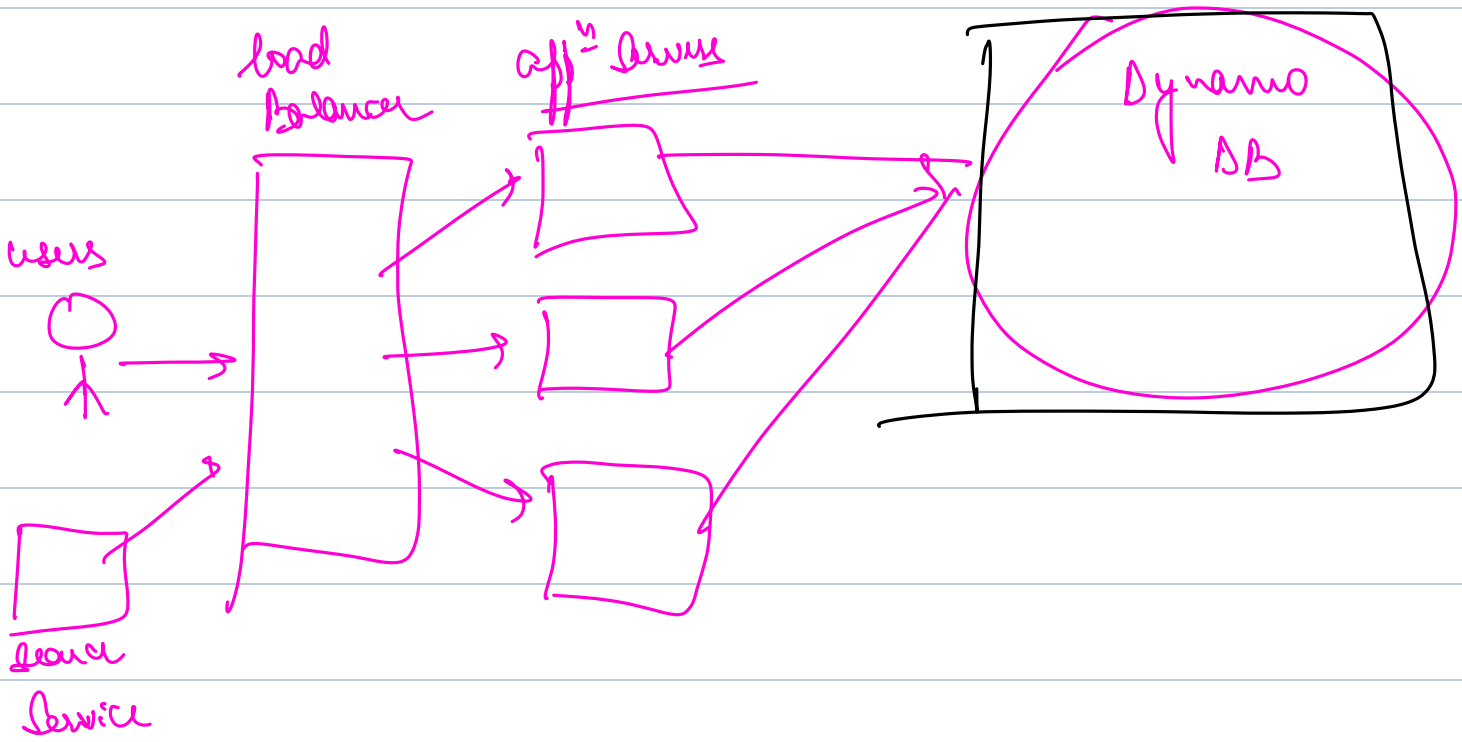
⇒ DISK

⇒ Map like Working

K-V DB

⇒ ~~can use Redis~~ ⇒ costly

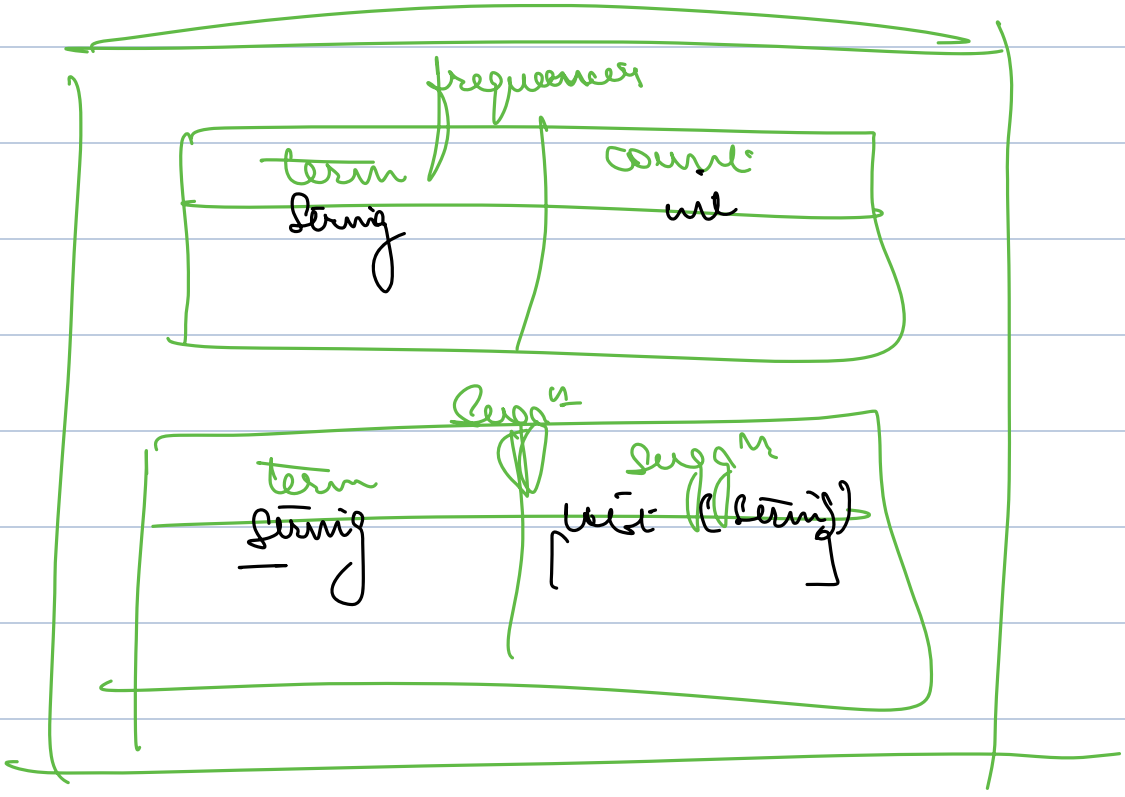
or Dynamo DB

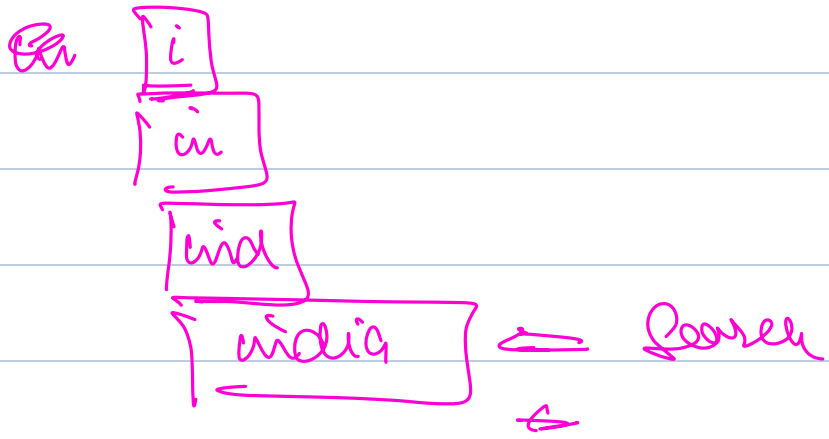


Data can't fit in 1 machine

⇒ Sharding

⇒ Sharding Key ?





i [india]

in [india]

ind [india]

Break till 10:20

Sharding key?

india

p

hash(hash(term))

→ hash(term)

Sharding Key

1. prefix
2. length (prefix)
3. First Char

Opt^n

LEN (TERM) ⇒ unbalanced

Shard 1

freq	Count
a	
b	
c	
d	
e	

Shard 2

freq
ag
ah
ac
ad
ae

1 ⇒ 26

2 ⇒ 26^26

3 " 26^26^2

if sk = len (term)
each in diff shard

Search (india)

indi
ind
in
i

① unbalanced shards

② too many shards to be access

Opn-2 First Char

even more Creative idea

→ at max 26 shards

→ never have upper limit

→ unbalance

→ a, y/s y

Opn-3 Term as Sharding key

india → updates will keep accessing many shards

indi

india

indi

ind

in

i

all of them may belong to diff machine

Opt 2

have just 3 chars as a sharding and 1 top shard for all terms < 3 chars

a
aa
b
ab
ac
cd

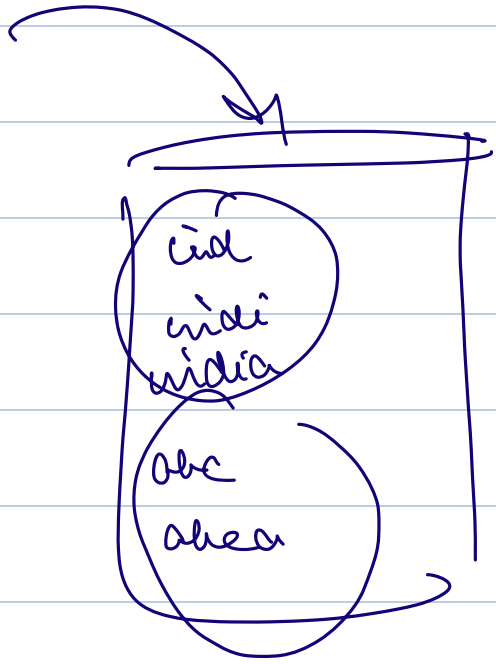
abc
ind
indi
aacc

india

sharding key

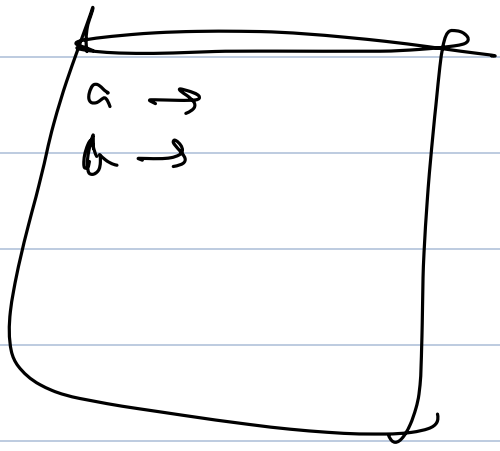
india →	ind	same shard (1)
indi →	ind	
ind →	ind	
in, i →	shard if < 3 char	same shard. (2)

$26 * 26 * 26 \Rightarrow 17576$



Key ⇒ first 3 chars

$26 * 26$ 900



getKey (String) {

for values >= 3 char

if str. len ≥ 3

return str[:3]

else

return -1

}

unique machine to store values of 1 and 2 chars

Skey

abc

Skey

and

$26 \times 26 \times 26 \rightarrow (26)^3$

Shards == # Machines

Data

No DB that stores data on disk that can handle large # reads
. large # writes

Read Heavy System

or

Write Heavy System

either reduce reads

or reduce writes

R↓ or W↓

→ caching

120B

↓

1B

20B →

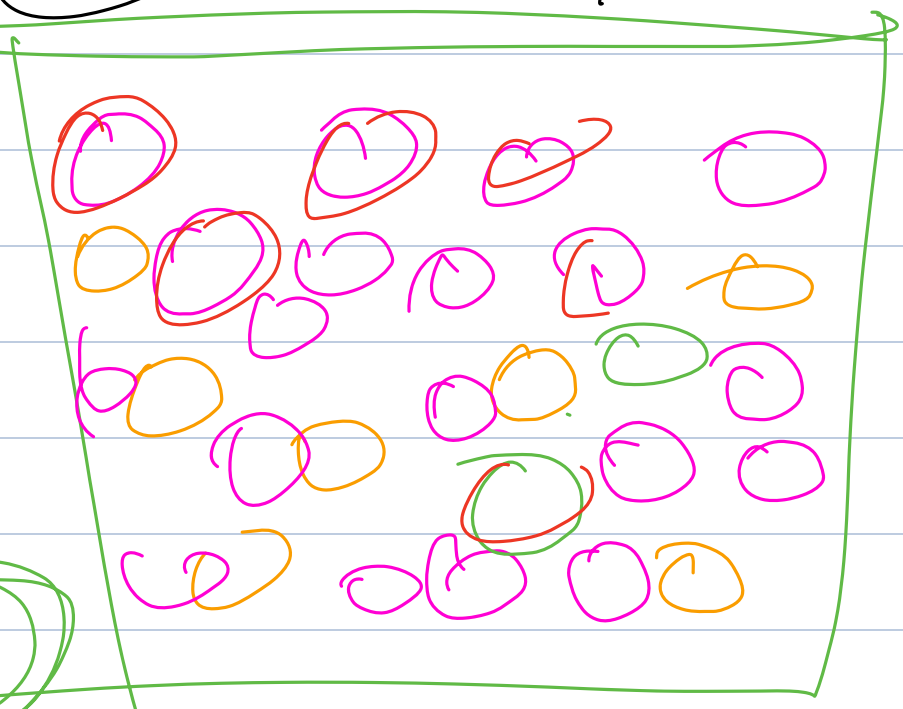
Reduce Usable Technique!

10 hrs

find the max^m occurring colour

↓

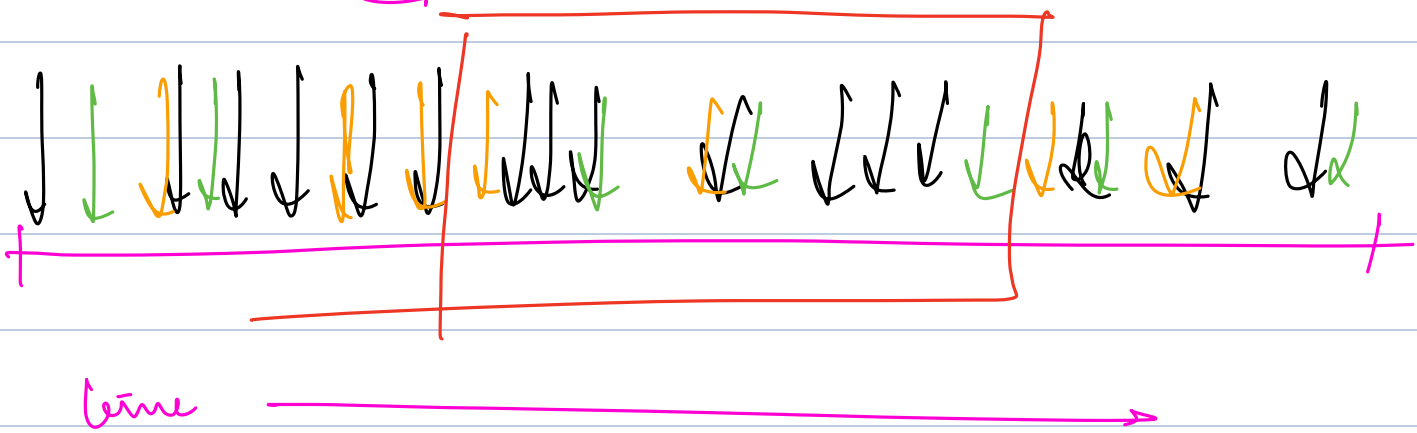
2 min



[Blue, Green]

→ [Green, Blue]

top 2




```
updateFreq (term) {
    int r = Random(1, 100);
    if (r != 1) return
    freqMap[term]++
    for all prefixes of Term
}
```

99%

R : W

120B 20B → 1% of 20B

120B : 0.2B

Read Heavy System

Sampling

india = 50000 ⇒ 500

indigo = 26000 ⇒ 260

Can we handle Recency?

india T ≠ ≠ 4 5 6 7

tall

freq Map

Searching Trend = ~~50000~~ 51000

Searching Barrel - ~~14000~~ 24000

today

ST. 1000

Sb, 10000

Sac

Saeini Tendulke

Saeini Bank

	Day 0	Day 1	Day 2	Day 3	Day 4	
ST	10000	11000	12000	13000	14000	15000	16000
SB	3000	6000	8500	10500	12000	13000	13500

	Day 0		Day 1		Day 2		Day 3		Day 4		
ST	10000	5000	6000	3000	4000	2000	3000	1500	2500	1250	2250	1125	2225
SB	3000	1500	4500	2250	4750	2375	4875	2150	3650	1500	2800	1900	2400

Time Decay

↳ at the start of day, half the count of everyone

$X \mid Y \mid Z \mid A \mid B \mid C$

$\frac{X}{2} \quad \left(\frac{X}{2}+Y\right) \quad \left(\frac{X}{4}+\frac{Y}{2}+Z\right) \quad \left(\frac{X}{8}+\frac{Y}{4}+\frac{Z}{2}+A\right)$

Ser

200M writes

$\frac{200 \times 10^6}{10^5}$ $\times 2000$ w/s

10TB of data

process

24 Machines

512 GB RAM

DOM

NITHIN

NITISH

get Search (NIT)

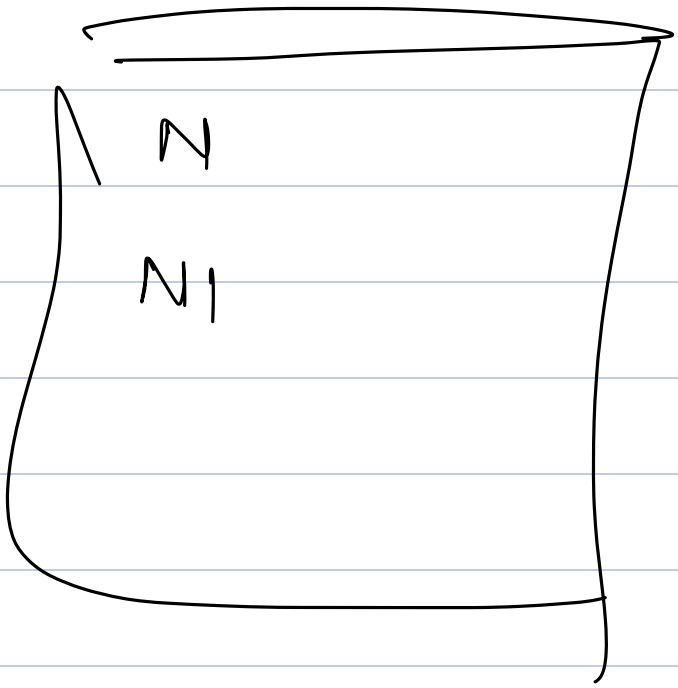
get Search (NIT)

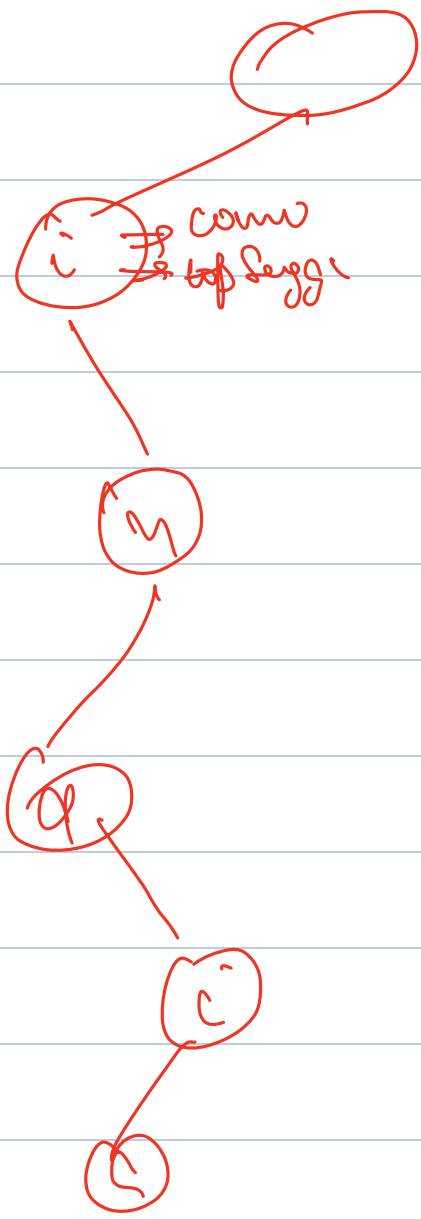
3 chars
→ 4 chars

NITISH BHARADWAJ

	freq
NIT	
NITI	
NITIS	
NITISH	
NITISH_	
NITISH_B	
NITISH_BH	1

top Sugg^n



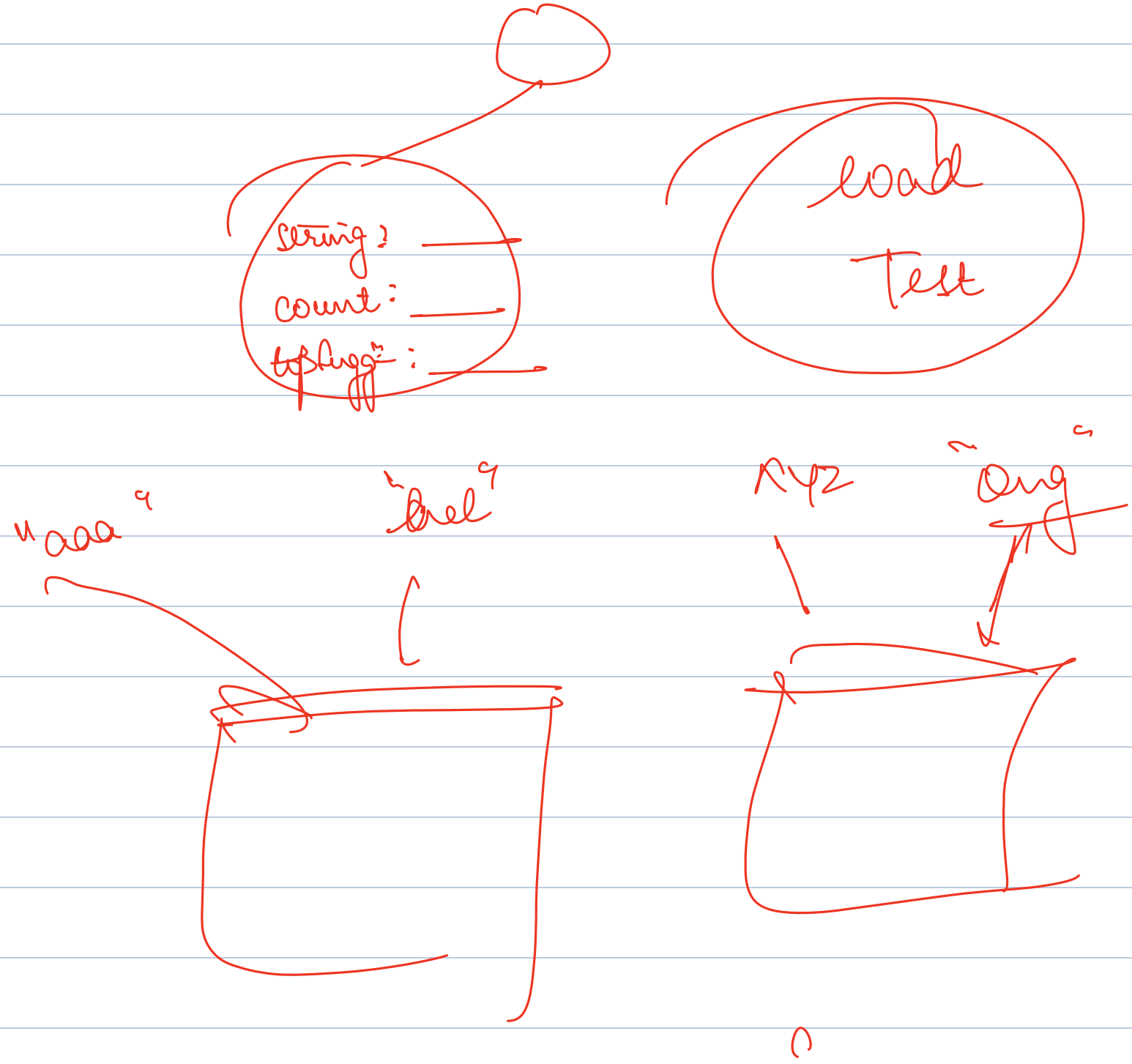


TrieNode {

count :

topSeg

}


String :
count :
toString :
load
Test
"aaa"
"bbb"
xyz
"ccc"