**Problem Statement**

You are a part of the Software Development team working on the development of a search engine similar to Ask.com where the user can input the textual web search query and the software tool helps find the corresponding information. You have been assigned a task by the project manager to optimize the search service. You have an idea of using the typeahead suggestion that can help users to avoid typing long and complex typos and help people save time by allowing them to quickly complete the search they already intended to do. You have to design the typeahead suggestion service for the project. What would be the critical points to be focused on for the designing of the Typeahead Suggestion Service?

# Features (F.R)

We have to limit the scope of the req

i)how many minimum characters after which suggestions should show up?

2 characters

ii)how many suggestions to be shown?

Should return 10 search queries

iii)case sensivity

case insensitive

iv)scoring mechanism to tell which suggestion is better than other

based on frequency, count and user activity

# Estimations and Constrainsts

Read : Write 10 : 1

Tranffic estimates :

Lets say 10k queries / sec => 10k x 60 x 60 x 24 ~ 1 B queries / day (read throughput)

Memory req :

Let’s say 10% of the 1B request are unique search keys

Means 100 M are unique, lets take 40 bytes for each search term

40bytes x 100 M = 4 GB every day

# Design goals

Non fun req

i)High availability 99.9 % - almost responding everytime

ii)Fault tolerance - should not have any single point of failure

iii) minimal latency is maintained - suggestions should appear in real time (very quick)

# High Level Design (HLD)

i)what are the APIs

search API

/api/search?q=<search term>

We should return 10 closest suggestions to the client

ii)Algorithm

ba - > bat

ball

retrieval need to be quick

read-heavy application

The Trie datastructure should be the right datastructure to use here.

The word Trie is derived from the word retrieval.

n-ary tree, dfs search always reveals a word that has been stored

let us say we have 2 words ball , bat



Time complexity is O(k) , k is size of search term or the term which needs to be inserted.

Space complexity O(26 x N x k )

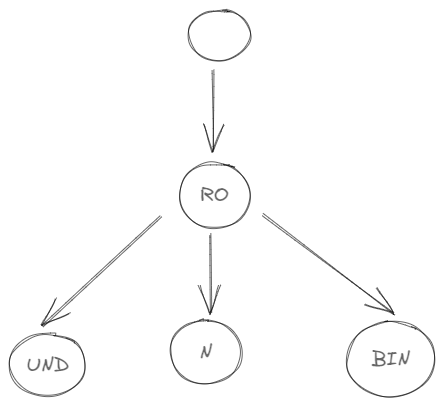
Alphabet set -> 26

N no of terms in trie determines how sparse or the wide the trie is

K length of the word determines the depth of the trie

To make the trie little more space efficient using compressed trie

Let’s take words Robin , Round , Ron



Just learn the algorithm of the trie.

Trie class looks like this

Class Trie{  
 children : [] ,

isEnd = false ,

frequency = 0 ,

topTenSug:[]

}

Funtions

1)Insertions

User will be inserting the n number of serach terms, we will log all of the search terms and insert into the trie asynchrounosly with cron jobs

Cron jobs will not interfere the actual search results that are retuned to user in real time but they will run as auxillary processes that will analyze all the search terms update the frequencies, top ten suggestions

Why cron jobs? In real time we should give the read/queries quickly. So we need to keep all the write queries run asynchrounously.

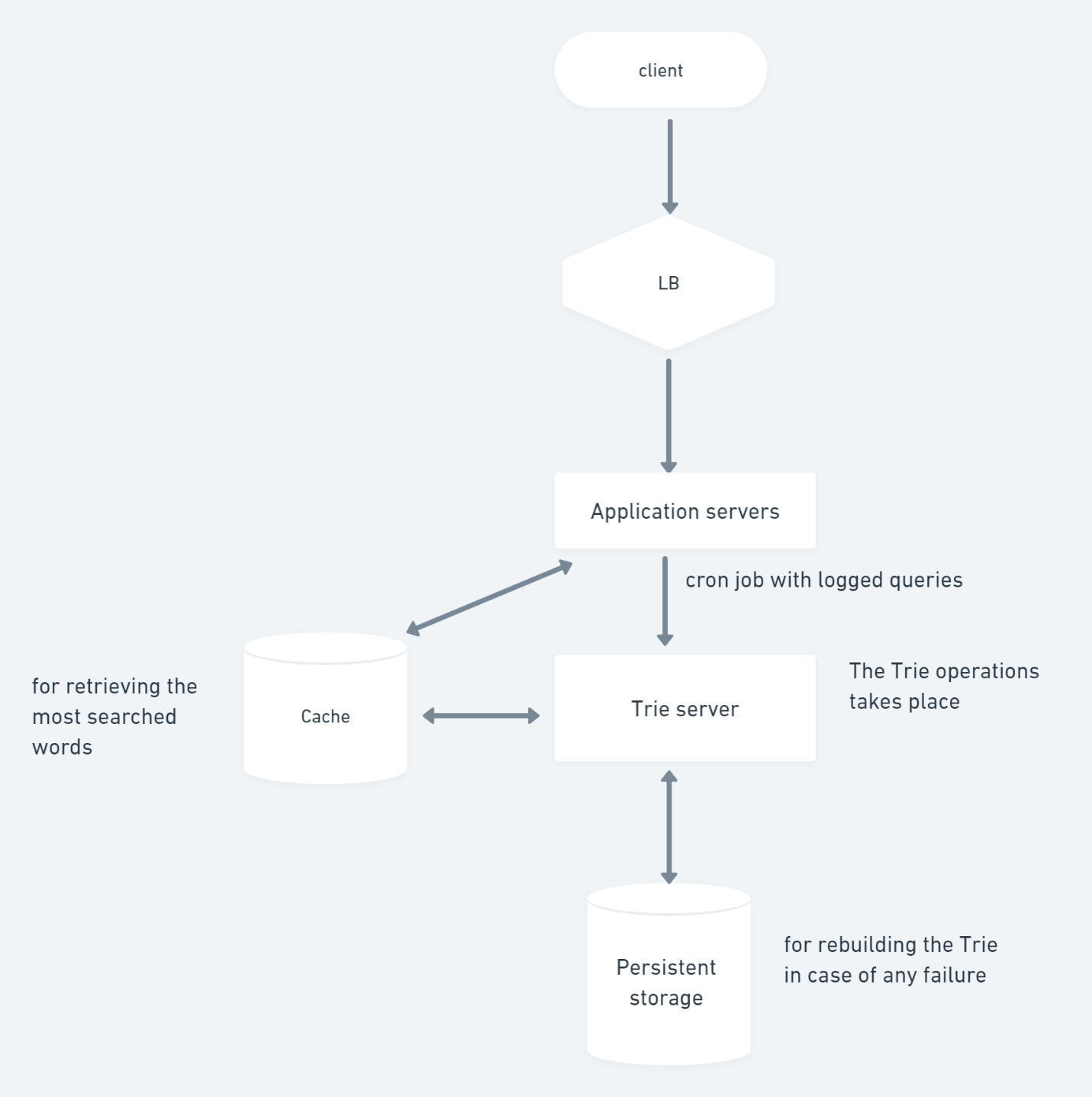
2)Updations

Updation happens through cron jobs as mentioned above, we will take the snapshot T of the current trie and update it to T’, and should be seamlessly replicated to the all servers

3)Deletions

Special case of the updation, let’s say there are some restricted search terms that need to be deleted from the trie.

Maintain a list of restricted words.



Client Side Technologies

We need to provide the real-time experience to the user.

i)The moment user enters website it should create a connection only then request will go, for this we can use the websockets

ii)While the user start typing and let say there is 50 ms gap, then client should send the request to server, so there we are not overburneing the user with in appropriate results

iii)Client can store the search reults in the local storage, so that request do not need to be sent to the server.