How to design certain things like

1. Facebook post search

2. How does elastic search work

3. Facebook live comments some pressing questions

Work on how to scale each section of the code

1.

And then we have

1.

Print all the notes that needs to be covered in each

Please take a look at the following again today:

Facebook live comments

1. Why zookeeper is so powerful

2. Why consistent hasing used for load balancing clinet connections here

3. Why facebook live comment needs CDN for sure

4. What are some downsides to using SSE in live feed?

5. How to scale redis?

Today worked on base65 encoidng url shortener

1. Also worked on gaming leaderboard as said

2. How does MongoDB scale so well?

3. Can look at distributed caching next after this

-

4. Work on this one next

- Maintain a distributed cache,

1. Today worked on top 10 songs in 5 minute window span system desing problem

Worked on traffic light in frontend and then some other stuff here

- and this

What are map reduce jobs?

And then here

New trick when using binary search here

What’s the use case of mapreduce?

1. Using top k system design where you are mapping and then combing the data at the end here

2.The above is very important here

3.

**What is MapReduce?**

MapReduce is a Java-based, distributed execution framework within the [Apache Hadoop Ecosystem](https://www.databricks.com/glossary/hadoop-ecosystem). It takes away the complexity of distributed programming by exposing two processing steps that developers implement: 1) Map and 2) Reduce. In the Mapping step, data is split between parallel processing tasks. Transformation logic can be applied to each chunk of data. Once completed, the Reduce phase takes over to handle aggregating data from the Map set.. In general, MapReduce uses [Hadoop Distributed File System (HDFS)](https://www.databricks.com/glossary/hadoop-distributed-file-system-hdfs) for both input and output. However, some technologies built on top of it, such as Sqoop, allow access to relational systems.

Inspired by Google,

A MapReduce system is usually composed of three steps (even though it's generalized as the combination of Map and Reduce operations/functions). The MapReduce operations are:

* **Map:** The input data is first split into smaller blocks. The Hadoop framework then decides how many mappers to use, based on the size of the data to be processed and the memory block available on each mapper server. Each block is then assigned to a mapper for processing. Each ‘worker’ node applies the map function to the local data, and writes the output to temporary storage. The primary (master) node ensures that only a single copy of the redundant input data is processed.
* **Shuffle, combine and partition:** worker nodes redistribute data based on the output keys (produced by the map function), such that all data belonging to one key is located on the same worker node. As an optional process the combiner (a reducer) can run individually on each mapper server to reduce the data on each mapper even further making reducing the data footprint and shuffling and sorting easier. Partition (not optional) is the process that decides how the data has to be presented to the reducer and also assigns it to a particular reducer.
* **Reduce:** A reducer cannot start while a mapper is still in progress. Worker nodes process each group of <key,value> pairs output data, in parallel to produce <key,value> pairs as output. All the map output values that have the same key are assigned to a single reducer, which then aggregates the values for that key. Unlike the map function which is mandatory to filter and sort the initial data, the reduce function is optional