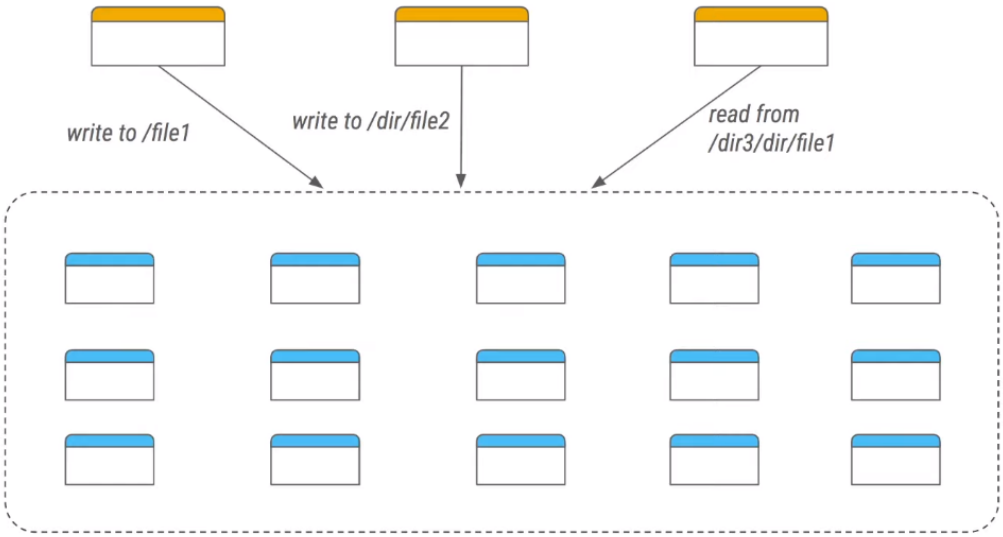
MapReduce paper: MapReduce: Simplified Data Processing on Large Clusters

https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf

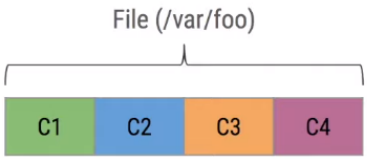
GFS



Distributed File Storage

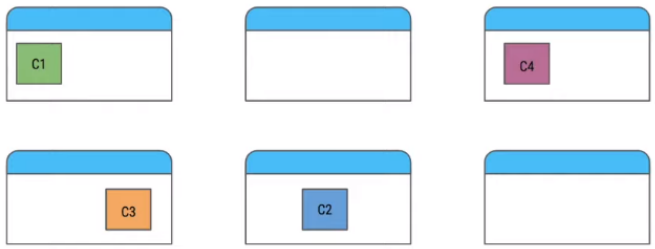
1 cluster = 100s of commodity servers

Chunks

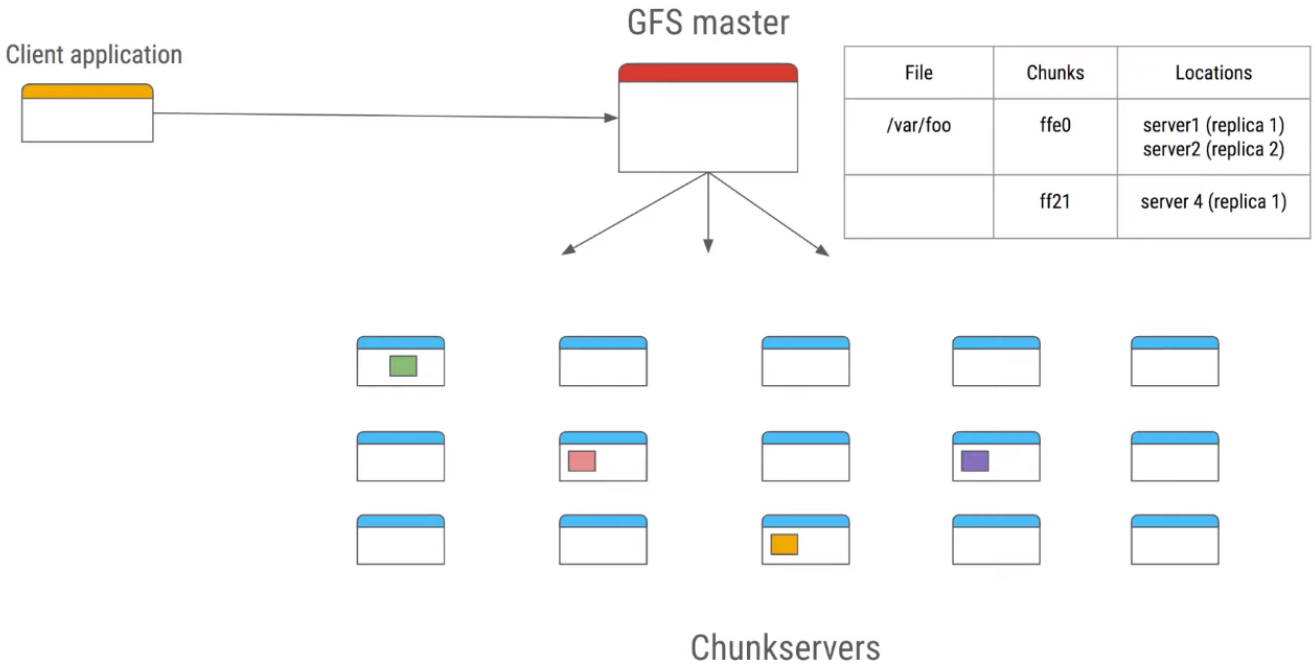


Files split into chunks

- Stored in Chunkservers



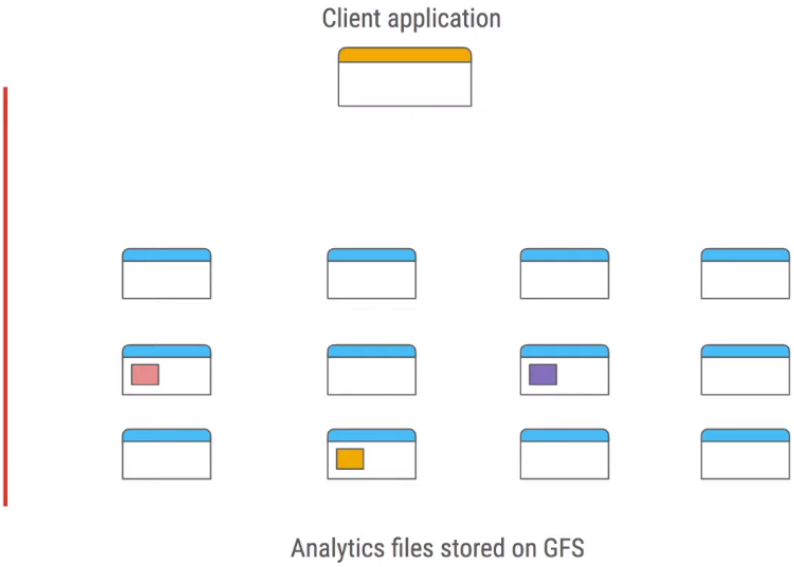
Chunks of single file are distributed on multiple machines



Use-Case

Google Music analytics stored in GFS.

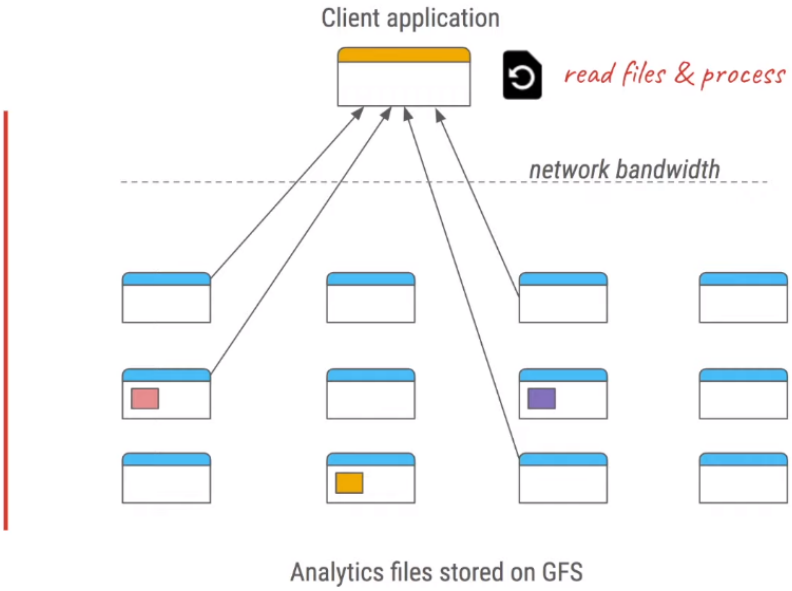
Write program to find while song was played how many times.



Get the top 10 trending songs

Problems

- High network bandwidth
- Multi-TB file(s)
- Slow to process

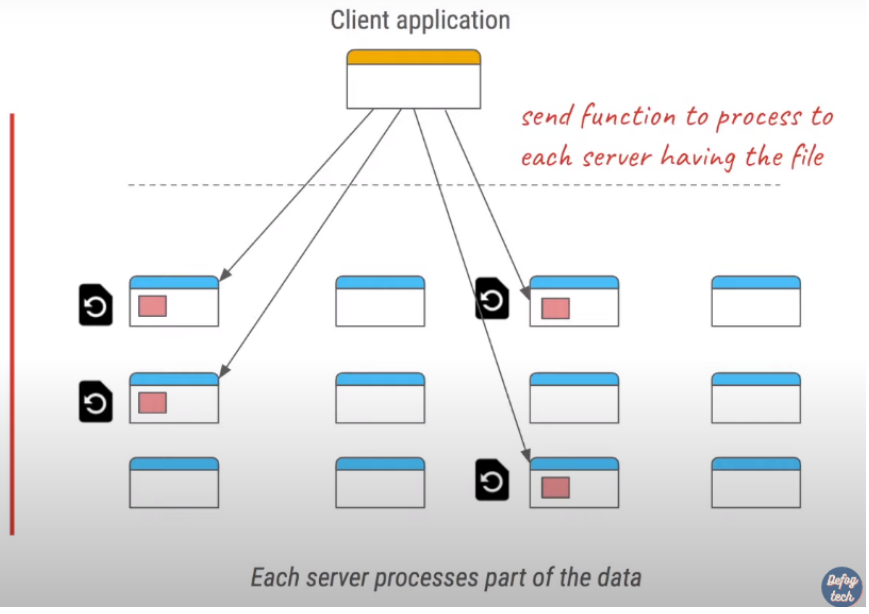


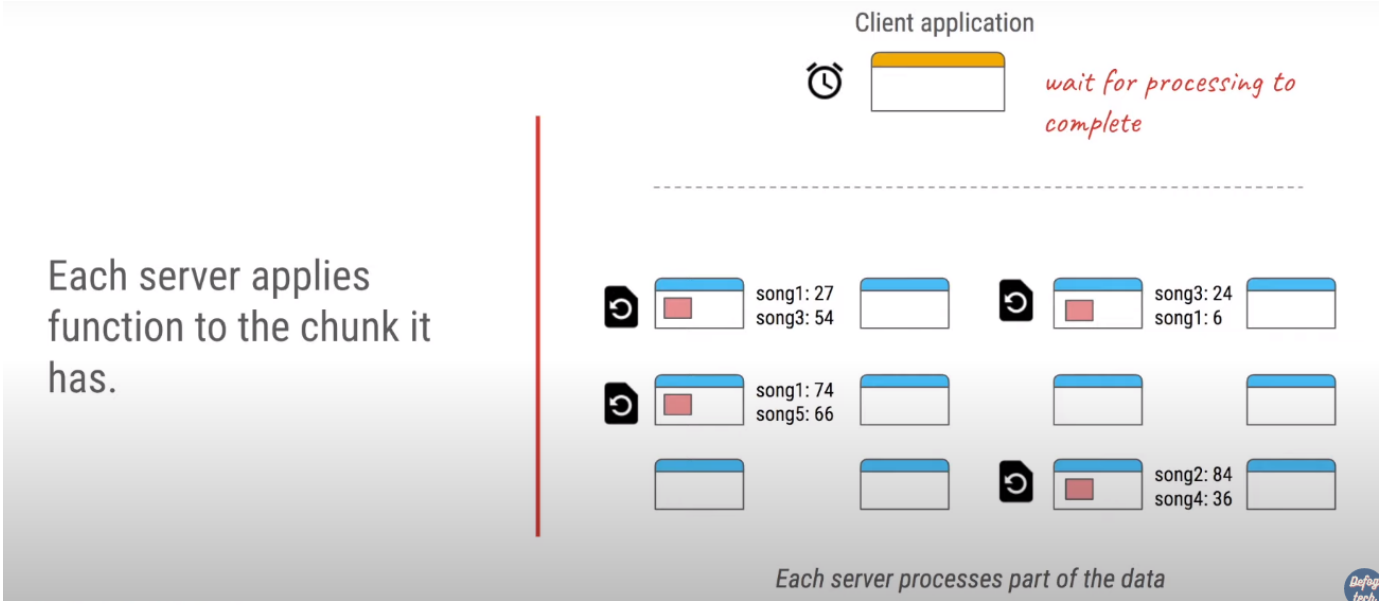
The most straightforward approach involves utilizing GFS directly within the client application: extracting all the necessary data and processing it. However, this approach introduces several challenges:

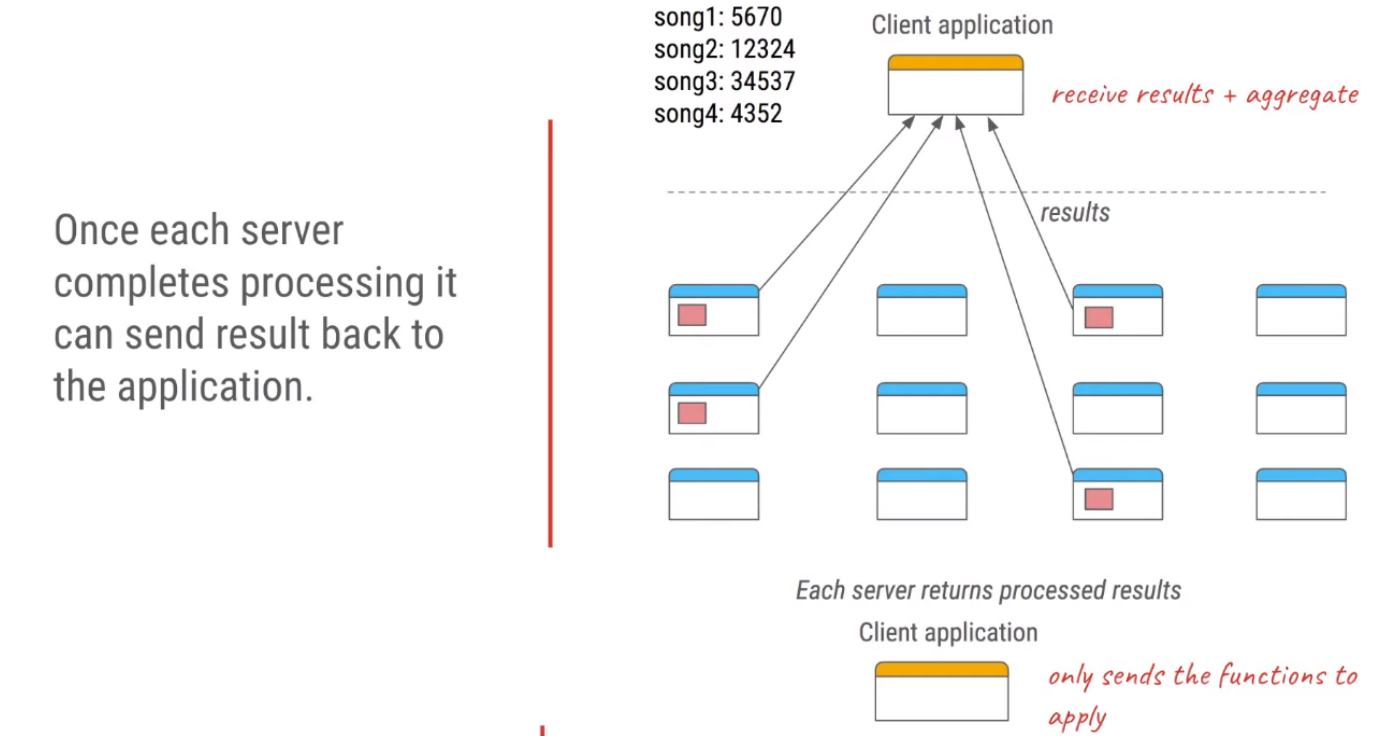
- The files are notably large, and attempting to retrieve all of them to the client application can significantly strain the network's bandwidth.
- When a single client application endeavors to process multiple terabytes of files, a substantial amount of time is inevitably consumed.

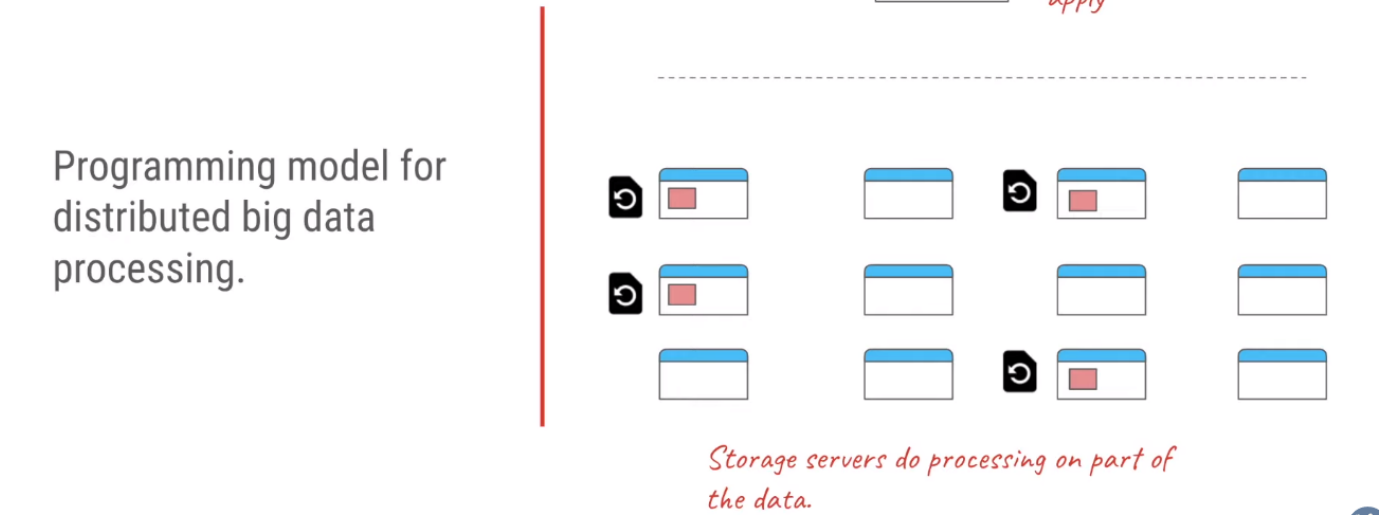
To address these issues, an alternative solution emerges: rather than transporting data to the client, relocate the processing function to the data itself.

Instead of bringing data to client, bring the processing function to the data.



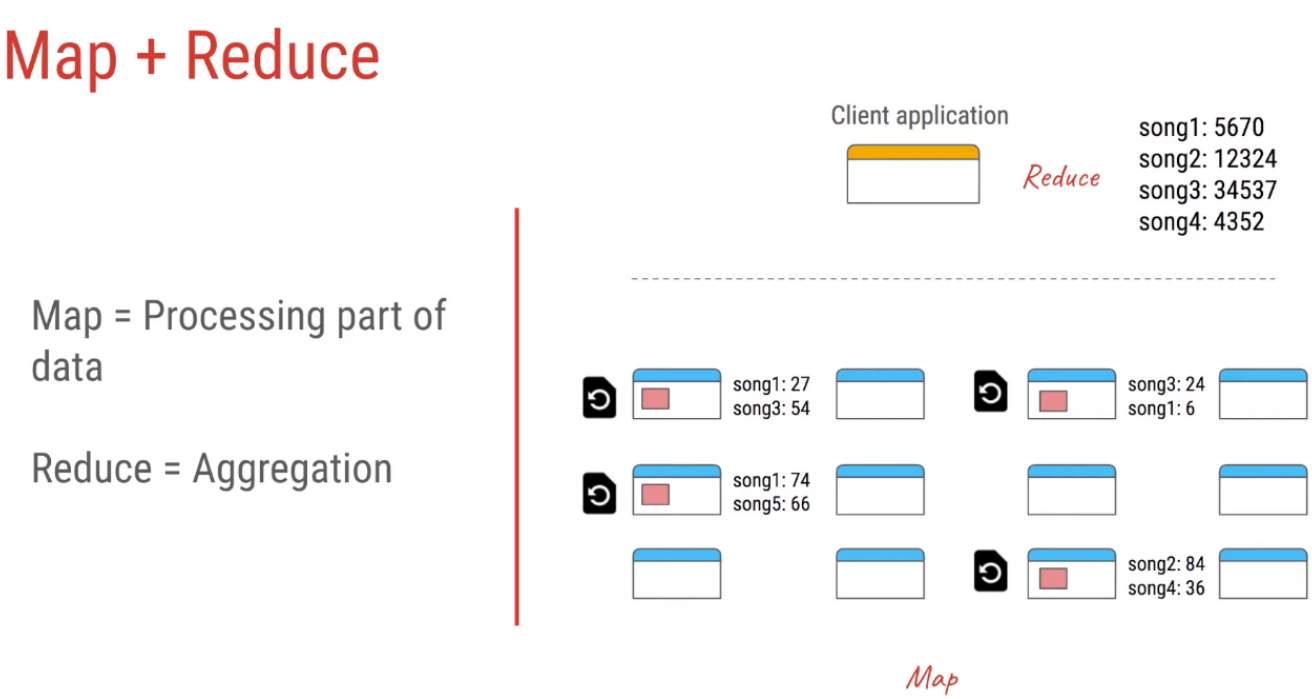






This programming paradigm, in which instead of extracting data from multiple servers, you deploy your code onto numerous servers and instruct them to execute or process that code, is known as MapReduce.

Map + Reduce



The mapping function involves iterating through the file line by line, and for each song, it is stored in a hashmap. A counter is maintained for each unique key in the hashmap. When the same song is encountered again, the corresponding count is incremented.

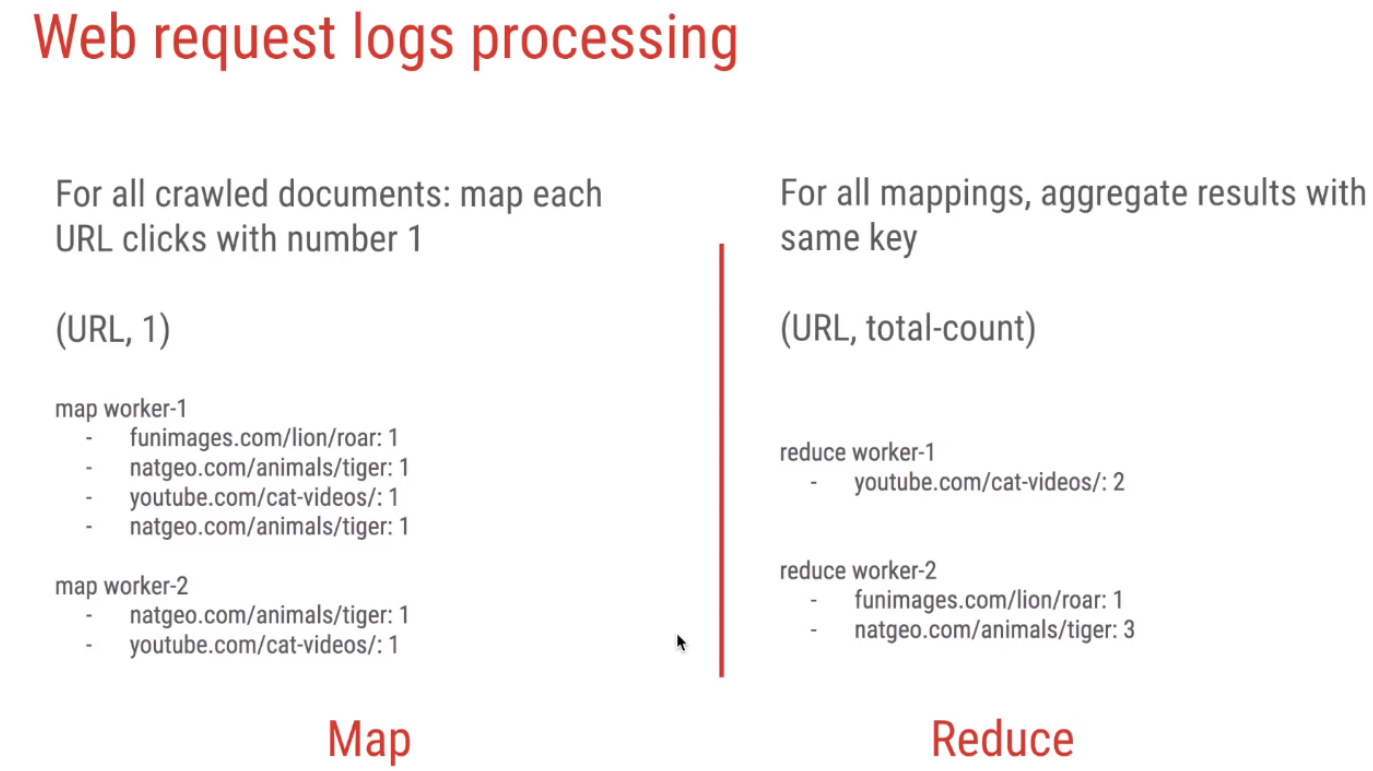
The second component is the reduce function, responsible for aggregating the results from each of these individual server functions.

Practical Implementation of MapReduce

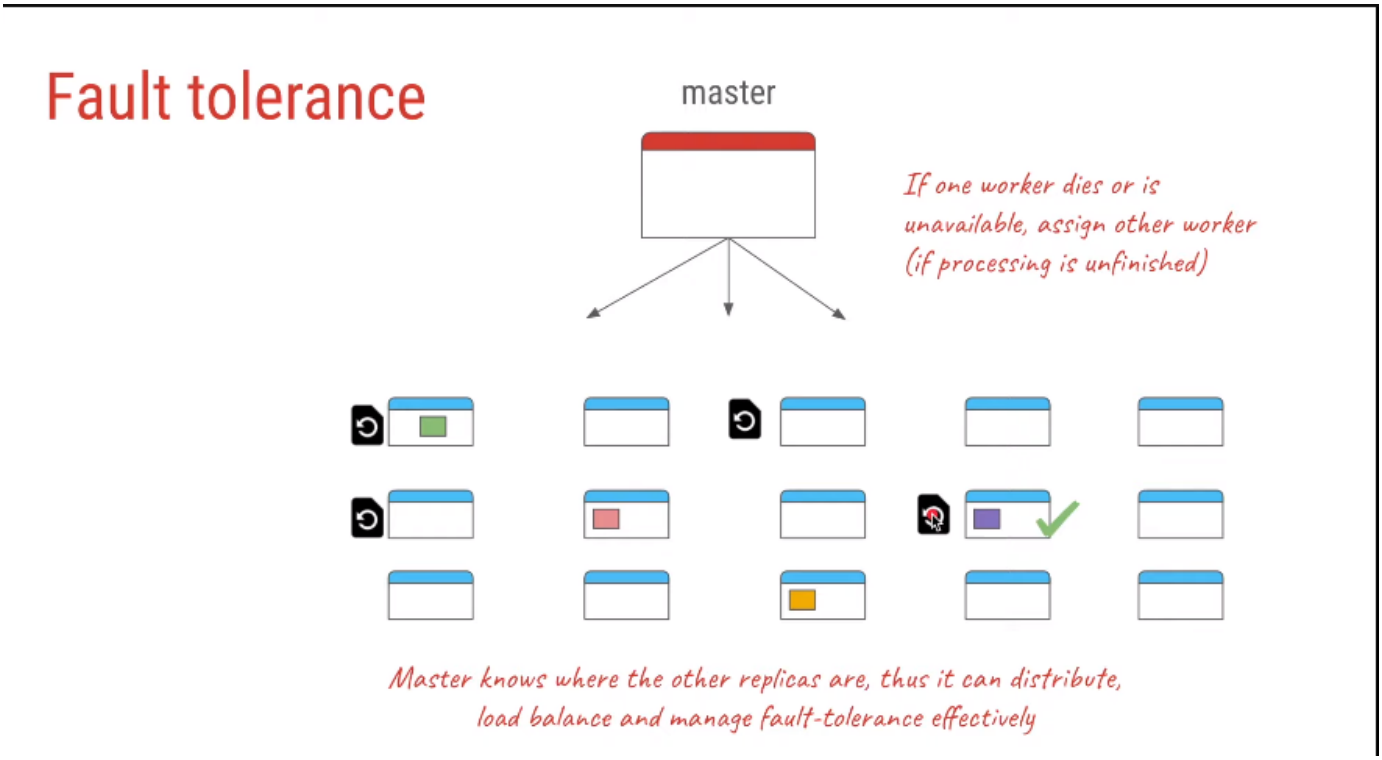
- The client application doesn't handle the entire implementation itself.
- Instead, there's a distinct component known as the Master that manages these tasks.
- Here, the client application simply informs the master about its mapreduce job, provides the Map and Reduce functions, and specifies the files to be processed.
- Utilizing the GFS, the master identifies the storage location of the files or their components across servers.
- On each server, a separate Worker process is initiated by the master.
- These worker processes execute the designated function, engaging in mapping (processing the file) within their respective servers.
- This approach effectively avoids the need to move files externally. The process is executed right where the file is located.
- Throughout this process, the master monitors progress by means of heartbeats. The worker processes periodically transmit heartbeat data, indicating the percentage of completion.
- Furthermore, the workers refrain from transmitting results to the master. Instead, they save results locally, sending only the file name and path to the master. The master compiles the file paths for results across all servers.

- Once processing concludes, the partial results from each server remain on their respective local hard drives.
- Rather than funneling all data to the master for aggregation, a set of reduce workers is established by the master—similar to the earlier map workers—on any available server. The master assigns these reduce workers the task of processing the file paths for all partial results.
- These reduce workers collaborate to distribute and handle the task efficiently.
- Upon final aggregation of all counts, the ultimate output is stored at a designated location within the GFS or External database.

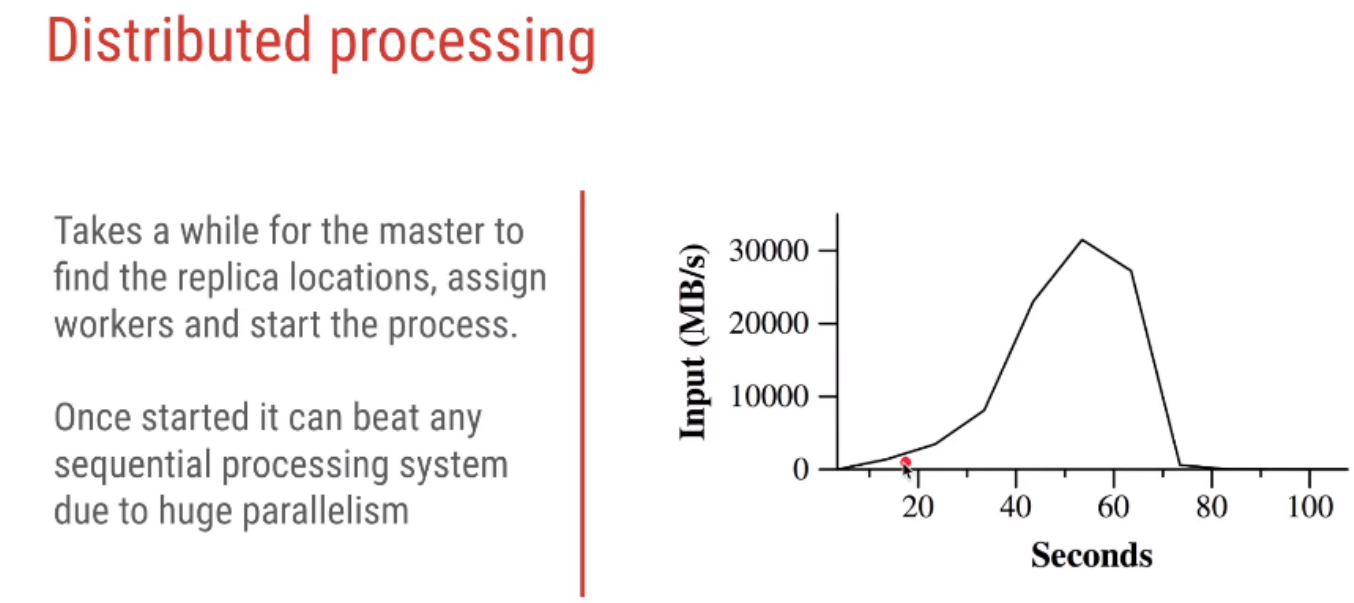
Use Case:



Fault Tolerance



Processing Speeds



References:

1. https://www.youtube.com/watch?v=MAJ0aW5g17c
2. https://medium.com/analytics-vidhya/introduction-to-mapreduce-a98f3c80febc
3. https://iamshobhitagarwal.medium.com/writing-your-first-map-reduce-program-in-hadoop-using-python-e5885331b67b