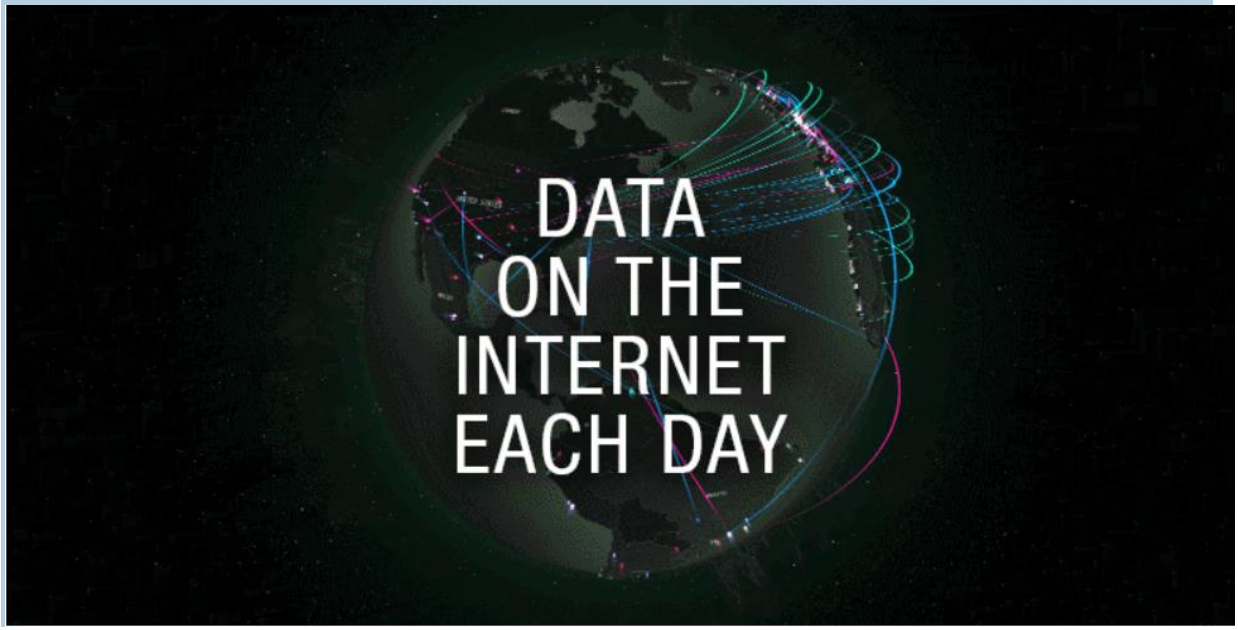
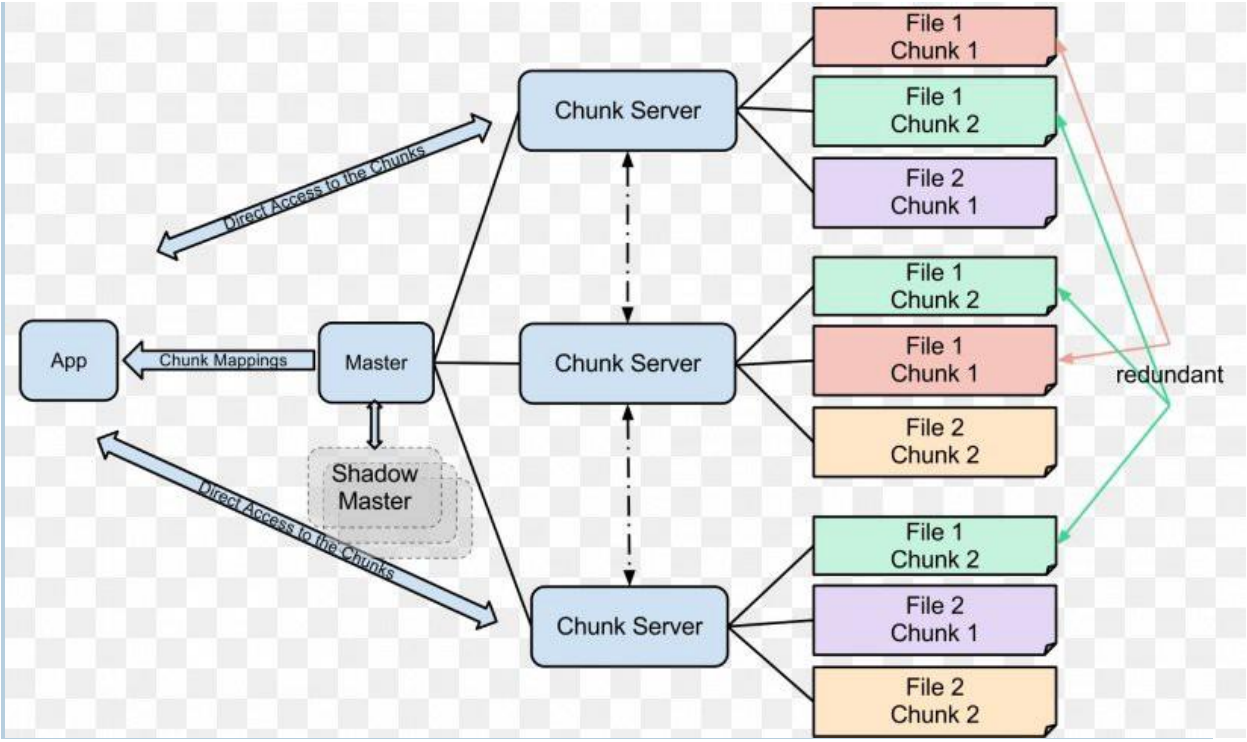
How big MNC's like Google, Facebook, Instagram etc stores, manages and manipulate Thousands of Terabytes of data with High Speed and High Efficiency.



Facebook revealed some big, big stats on **big data** to a few reporters at its HQ today, including that its system processes 2.5 billion pieces of content and 500+ terabytes of data each day. It's pulling in 2.7 billion Like actions and 300 million photos per day, and it scans roughly **105 terabytes** of data each half hour. Plus it gave the first details on its new **"Project Prism"**.

Google currently processes over **20 petabytes** of data per day through an average of 100,000 MapReduce jobs spread across its massive computing clusters. The average MapReduce job ran across approximately 400 machines in September 2007, crunching approximately 11,000 machine years in a single month.



So how its data is stored?

•Key is formed using data from row,column and timestamp, where timestamp is the time when data was added or modified in the table.

Google and any other company which generates huge amount of data uses **cloud** to store it's data. Given that the number of users are always volatile, the data generated on a day's scale is also volatile. Therefore Google doesn't use an off the shelf type of storage to store their data. Purchasing **20 PB** of hardware

everyday is out of question. Google not only needs a **scalable** data, but at the same time needs durable one.

Google's solution to this: Distributed File System, Big Table and Object Based Storage!

- **Distributed multi-level map**
- **Fault-tolerant, persistent**
- **Scalability**

–Thousands of servers
–Terabytes of in-memory data
–Petabyte of disk-based data
–Millions of reads/writes per second, efficient scans

- **Self-managing**

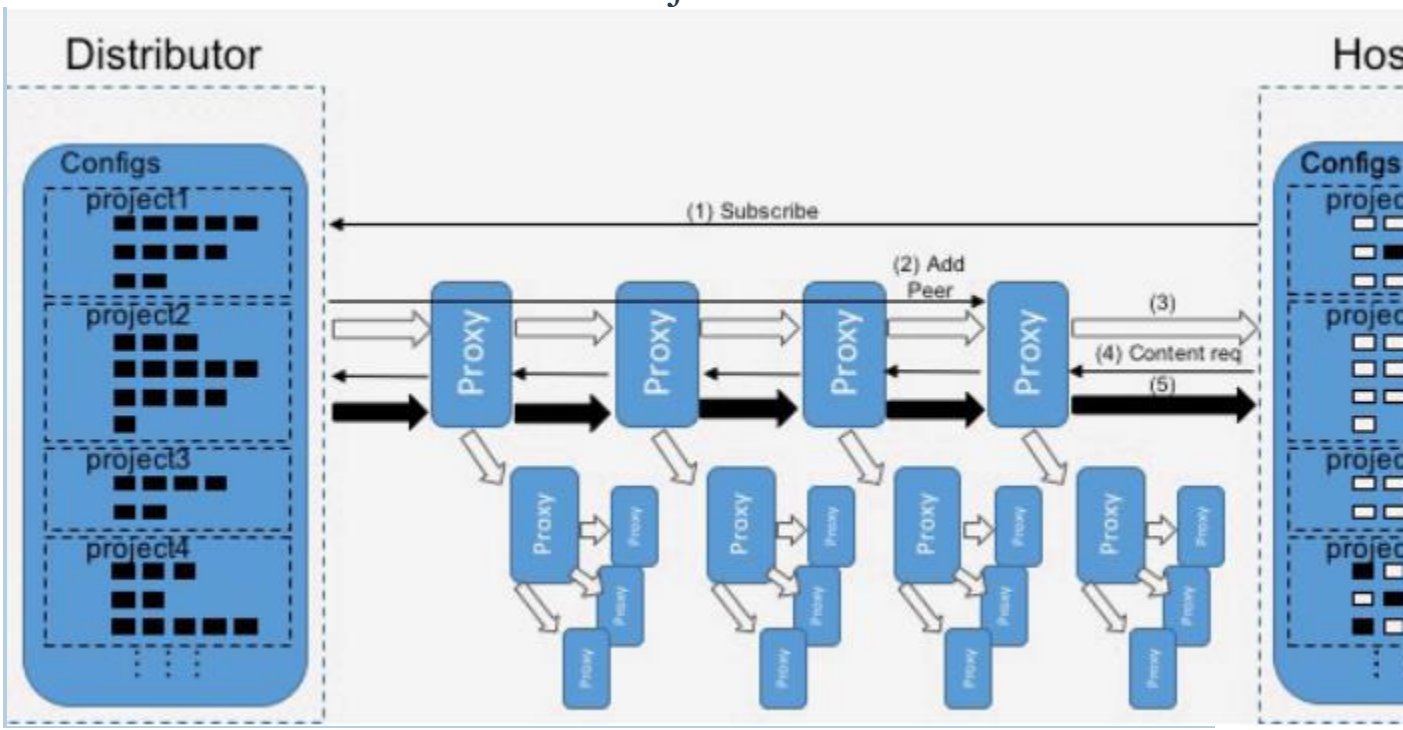
–Servers can be added/removed dynamically
–Servers adjust to load imbalance

A brief idea about its infrastrucure :

•Building blocks:
–Google File System (GFS): Raw storage
–Scheduler: schedules jobs onto machines
–Lock service: Chubby-distributed lock manager
–MapReduce: simplified large-scale data processing

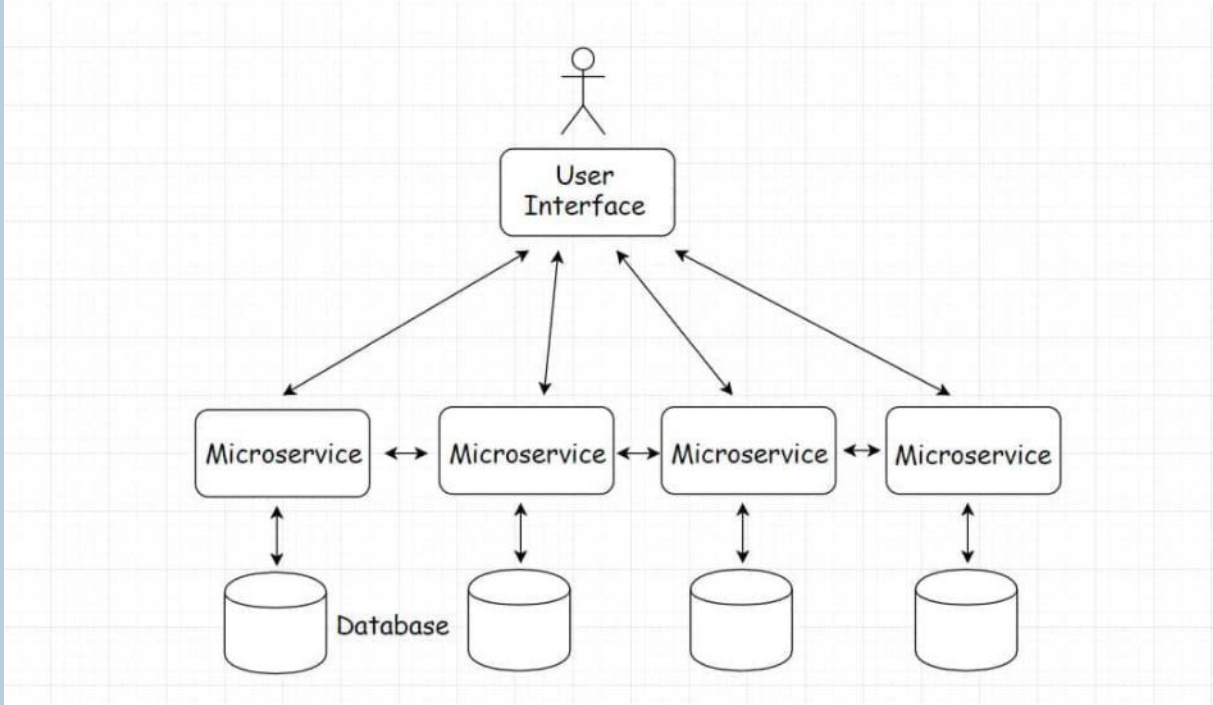
- **BigTable uses of building blocks:**
  –GFS: stores persistent data (SSTable file format for storage of data)
  –Scheduler: schedules jobs involved in BigTable serving
  –Lock service: master election, location bootstrapping
  –Map Reduce: often used to read/write BigTable data

- **Compression**, to reduce the size of the data.
  •Many opportunities for compression
  –Similar values in the same row/column at different timestamps
  –Similar values in different columns
  –Similar values across adjacent row



Facebook, today, is not a **monolithic architecture**.The social network as a whole consists of several different loosely coupled components plugged in together like Lego blocks. For

instance, photo sharing, messenger, social graph, user post etc. are all different loosely coupled ***microservices*** running in conjunction with each other. And every microservice has a separate persistence layer to keep things easy to manage.



Zero to Software/Application Architect learning track is a series of four courses that I am writing (*2 courses published*) with an aim to educate you, step by step, on the domain of software architecture & distributed system design. The learning track takes you right from having no knowledge in it to making you a pro in designing large scale distributed systems like *YouTube*, Netflix, Google Stadia & so on.

Software (backend) design is a crucial skill that I believe every developer should be deft in. With this architecture series not only you'll be interview-ready for your system design interview rounds but you'll also develop a solid foundation on the domain, becoming a better software engineer as a whole.

The courses included in this series help you understand various architecture styles, technology stacks, their trade-offs & why they are used in the industry. You'll get an insight into the intricacies of distributed systems/services that have to deal with heavy traffic influx on a continual basis. You'll understand concepts such as distributed locking, high-frequency transactions, data consistency, thundering herd problem, ways of efficiently managing petabyte-scale data & much more.