Google File System is a scalable distributed file system for large distributed data-intensive applications. It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients. Its goal has been driven by observations of our application workloads and technological environment, both current and anticipated, that reflect a marked departure from some earlier file system assumptions.

There are traditional choices and radically different points in design space, they are: 1)First, component failures are the norms rather than the exception. 2)Files are huge by traditional standards. 3)Third, most files are mutated by appending new data rather than overwriting existing data. 4)Fourth, co-designing the applications and the file system API benefits the overall system by increasing flexibility.

Observed Challenges: 1)System stores a modest number of large files. 2)Workloads consist of two kinds of reads: large streaming reads and small random reads. 3)Workloads also have many large, sequential writes that append data to files. 4)High sustained bandwidth is more important than low latency.

Advantages: A large chunk size offers several advantages. 1)First, it reduces the client's need to interact with the master. 2)Second, on a large chunk client is more likely to perform many operations on a given chunk, it can reduce network overhead. 3)Third, it reduces the size of the metadata stored on the master.
Disadvantages: 1)A large chunk size with lazy space allocation. 2)Delay hinders user efforts to fine-tune usage when storage is tight. 3)Applications that create and delete temporary files may not be able to reuse the storage right away.
There are two simple yet effective strategies, they are: 1)Fast recovery 2)Replication. Replication consists of two modes: 1)Chunk Replication 2)Master Replication **Diagnostic Tools:** Extensive and detailed diagnostic logging has helped immeasurably in problem isolation, debugging, and performance analysis, while incurring only a minimal cost.
Measurements: Measurement performance on a GFS cluster consists of one master, two master replicas, 16 chunk servers, and 16 clients.

Conclusion: Google file system demonstrates the qualities essential for supporting large-scale data processing workloads on commodity hardware. It also delivers high aggregate throughput to many concurrent readers and writers performing a variety of tasks. We achieve this by separating file system control, which passes through the master, from data transfer, which passes directly between chunk servers and clients.