

| **Name: Khan Farhaan Firoz** | **Course Name: DC Lab** |
| --- | --- |
| **Assignment No: 2** | **Roll no: 19CO26** |

**Aim:** Designing distributed systems: Google case study.

**Theory:**

Outcomes:

Google is a company that is known for designing and operating highly scalable and fault-tolerant distributed systems. Some of the most well-known systems that Google has designed and deployed include Google Search, Google Maps, and Gmail. In this case study, we will look at some of the key design principles and techniques that Google uses to build and operate its distributed systems.

1. Scalability: One of the key design principles for Google's distributed systems is scalability. Google's systems are designed to handle a huge amount of data and traffic, and to be able to scale horizontally as the load increases. This means that Google's systems are designed to be able to add more computing resources as needed, without requiring any significant changes to the system's architecture or design.
2. Fault tolerance: Another important design principle for Google's distributed systems is fault tolerance. Google's systems are designed to be able to continue operating even if some of the nodes in the system fail. To achieve this, Google uses a technique called replication, where data is stored on multiple nodes, and the system is designed to be able to tolerate the failure of some of those nodes.
3. Data consistency: Google's distributed systems are designed to ensure data consistency across multiple nodes. This means that when data is updated in one node, it is immediately propagated to all the other nodes, so that all nodes have the same view of the data. This is achieved using a technique called distributed consensus, where all nodes in the system agree on the order of operations and ensure that they are executed in the same order on all nodes.
4. Partitioning: Google's systems are designed to partition data into smaller subsets, which are stored on different nodes. This allows Google to scale its systems horizontally, as additional nodes can be added to handle additional partitions. Partitioning also allows Google to isolate failures to individual partitions, rather than having the failure affect the entire system.
5. MapReduce: MapReduce is a programming model and a software framework that Google developed to process large data sets in parallel on a large number of nodes. MapReduce allows Google to process large data sets efficiently, by distributing the processing across multiple nodes.
6. Load balancing: Google's distributed systems are designed to balance the load across multiple nodes, to ensure that no single node is overloaded. Load balancing is achieved using a combination of techniques, including dynamic routing, traffic splitting, and automatic scaling.
7. Monitoring and debugging: Google's distributed systems are designed to be highly monitorable and debuggable. Google has developed a number of tools and systems to help monitor the performance and health of its systems, and to quickly identify and fix any issues that arise.

Conclusion:

Google's distributed systems are designed to be scalable, faulttolerant, and highly performant. Google uses a variety of techniques, including replication, partitioning, MapReduce, and load balancing, to achieve these goals. Additionally, Google places a strong emphasis on monitoring and debugging, to ensure that its systems remain performant and reliable.