Revise These 10 Concepts

before your next

System Design Interview

and

Ace It Like a Pro!

DesignGurus.org

1. **Consistency models**

There are various consistency models that can be used in a distributed system, including **strong consistency**, **eventual consistency**, and **causal consistency**.

Understanding the trade-offs between these models and when to use them is crucial to designing distributed systems.

2. Replication

Replication is a key technique for increasing the availability and reliability of a distributed system. There are various approaches to replication, including **primary-secondary**, **leader-follower**, and **quorum-based**, and it's important to understand the trade-offs between them.

3. Partitioning

Partitioning, or sharding, is a technique for scaling a distributed system by dividing the data across multiple servers or data centers. There are various approaches to partitioning, including **range-based**, **hash-based**, and **consistent hashing**, and it's important to understand the trade-offs between them.

4. Fault tolerance

Fault tolerance is the ability of a distributed system to continue operating in the face of failures or errors. There are various techniques for achieving fault tolerance, including **replication**, **failover**, and **self-healing**, and it's important to understand the trade-offs between them.

5. Load balancing

Load balancing is a technique for distributing incoming requests across multiple servers or resources in a distributed system. There are various algorithms and approaches to load balancing, and it's important to understand the trade-offs between them.

6. Latency

Latency refers to the time it takes for a request to be sent from one point in the system to another, and for the response to be received.

Latency is an important performance metric for distributed systems, as it can affect the speed at which the system can process requests and the user experience.

7. Throughput

Throughput refers to the rate at which the system can process requests or handle a workload.

Throughput is an important performance metric for distributed systems, as it can affect the capacity of the system and the time it takes to complete a given task.

8. Caching

Caching is a technique for storing frequently accessed data in a fast, local store in order to improve the performance of a system.

Understanding how to design systems that use caching can be useful for improving the performance and scalability of a distributed system.

9. CAP theorem

The CAP theorem states that it is impossible for a distributed system to simultaneously provide **Consistency**, **Availability**, and **Partition tolerance**. Understanding this trade-off is crucial to designing distributed systems.

10. Proxies

Reverse proxy: This type of proxy server is used to retrieve resources on behalf of a client from one or more servers. It's called a reverse proxy because it acts on behalf of the client, rather than the server.

Forward proxy: This type of proxy server is used to retrieve resources on behalf of a client from the Internet. It acts as an intermediary between the client and the server, forwarding requests from the client to the server and returning responses from the server to the client.

➡ All these concepts are discussed in "Grokking the System Design Interview" and "Grokking the Advanced System Design Interview" from DesignGurus.org

➡ **New year sale**: **20% off on all courses.**

DesignGurus.org