25 Golden Rules to answer in a System Design Interview

By

Logicmojo

Rule #1

If we are dealing with a read-heavy system, it's good to consider using a Cache

Logicmojo

Rule #2

If we need low latency in system, it's good to consider using a Cache & CDN

Logicmojo

Rule #3

If we are dealing with a write-heavy system, it's good to consider using a Message Queue for Async processing

Logicmojo

Rule #4

If we need a system to be ACID complaint, we should go for RDBMS or SQL Database

Logicmojo

Rule #5

If data is unstructured & doesn't require ACID properties, we should go for NO-SQL Database

Logicmojo

Rule #6

If the system has complex data in the form of videos, images, files etc, we should go for Blob/Object storage

Logicmojo

Rule #7

If the system requires complex pre-computation like a news feed, we should consider using a Message Queue & Cache

Logicmojo

Rule #8

If the system requires searching data in high volume, we should consider using a search index, tries or search engine like Elasticsearch

Logicmojo

Rule #9

If the system requires to Scale SQL Database, we should consider using Database Sharding

Logicmojo

Rule #10

If the system requires High Availability, Performance, and Throughput, we should consider using a Load Balancer

Logicmojo

Rule #11

If the system requires faster data delivery globally, reliability, high availability, and performance, we should consider using a CDN

Logicmojo

Rule #12

If the system has data with nodes, edges, and relationships like friend lists, and road connections, we should consider using a Graph Database

Logicmojo

Rule #13

If the system needs scaling of various components like servers, databases, etc, we should consider using **Horizontal Scaling**

Logicmojo

Rule #14

If the system requires high performing database queries, we should consider using Database Indexes

Logicmojo

Rule #15

If the system requires bulk job processing, we should consider using

Batch Processing & Message Queues

Logicmojo

Rule #16

If the system requires reducing server load and preventing DOS attacks, we should consider using a Rate Limiter

Logicmojo

Rule #17

If the system has microservices, we should consider using an API Gateway (Authentication, SSL Termination, Routing etc)

Logicmojo

Rule #18

If the system has a single point of failure, we should implement Redundancy in that component

Logicmojo

Rule #19

If the system needs to be fault-tolerant, and durable, we should implement Data Replication (creating multiple copies of data on different servers)

Logicmojo

Rule #20

If the system needs user-to-user communication (bi-directional) in a fast way, we should consider using **Websockets**

Logicmojo

Rule #21

If the system needs the ability to detect failures in a distributed system, we should consider implementing **Heartbeat**

Logicmojo

Rule #22

If the system needs to ensure data integrity, we should consider implementing Checksum Algorithm

Logicmojo

Rule #23

If the system needs to transfer data between various servers in a decentralized way, we should go for Gossip Protocol

Logicmojo

Rule #24

If the system needs to scale servers with add/removal of nodes efficiently, no hotspots, we should implement

Consistent Hashing

Logicmojo

Rule #25

If the system needs anything to deal with a location like maps, nearby resources, we should consider using Quadtree, Geohash etc

Logicmojo

Cracking the MAANG Technical Interviews

Learn From Logicmojo Live Classes

Follow : https://www.logicmojo.com/

THANK YOU !!!
For more such content follow

Logicmojo Academy