System Design Fundamental Concepts

1. Introduction to Architecting System At Scale.

<https://lethain.com/introduction-to-architecting-systems-for-scale/>

1. Load Balancers

<https://avinetworks.com/what-is-load-balancing/>

1. Caching

[https://aws.amazon.com/caching/#:~:text=In%20computing%2C%20a%20cache%20is,the%20data's%20pr](https://aws.amazon.com/caching/#:~:text=In%20computing%2C%20a%20cache%20is,the%20data's%20primary%20storage%20location.)

[imary%20storage%20location.](https://aws.amazon.com/caching/#:~:text=In%20computing%2C%20a%20cache%20is,the%20data's%20primary%20storage%20location.)

1. Data Partitioning

<https://docs.oracle.com/cd/E11882_01/server.112/e25523/partition.htm#:~:text=2%20Partitioning%20Concepts&text=Partitioning%20allows%20tables%2C%20indexes%2C%20and,a%20finer%20level%20of%20granularity.>

1. Indexing

<https://www.geeksforgeeks.org/indexing-in-databases-set-1/>

<https://dataschool.com/sql-optimization/how-indexing-works/>

1. Proxies(open proxies/reverse proxies)

<https://medium.com/must-know-computer-science/system-design-proxies-ef5f2c2676f2>

1. Redundancy And Replication

<https://medium.com/must-know-computer-science/system-design-redundancy-and-replication-e9946aa335ba>

1. RDBMS vs No SQL

<https://www.innoarchitech.com/blog/how-choose-right-database-system-relational-rdbms-vs-nosql-vs-newsql>(Must read, its super useful)

1. CAP Theorem.

<https://www.ibm.com/cloud/learn/cap-theorem>

1. Consistent Hashing

<https://medium.com/system-design-blog/consistent-hashing-b9134c8a9062>

1. Event Driven Architecture

<https://aws.amazon.com/event-driven-architecture/>

1. Publisher-Subscriber Pattern

<https://docs.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber>

1. Message Queues

<https://aws.amazon.com/message-queue/>

1. API Design

<https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design>