



Explore
(/explore)




Tracks
(/tracks)



My Courses
(/mycourses)



Edpresso
(/edpresso)



Refer a
Friend
(/refer-a-friend)



Create
courses/grokking-
system-design-
/view/mEN8IJXV

Grokking the System Design Interview

(/collection/5668639101419520/56490502)

87% completed



Q Search Course

Glossary of System Design Basics



- System Design Basics
(/courses/grokking-the-system-design-interview/B892KY261z2)
- Key Characteristics of Distributed Systems
(/courses/grokking-the-system-design-interview/YQWGjZZVz9)
- Load Balancing
(/courses/grokking-the-system-design-interview/3jEwI04BL7Q)
- Caching
(/courses/grokking-the-system-design-interview/3j6NnJrpp5p)
- Data Partitioning
(/courses/grokking-the-system-design-interview/mEN8IJXV1LA)
- Indexes
(/courses/grokking-the-system-design-interview/gxkVE8NEvXj)
- Proxies
(/courses/grokking-the-system-design-interview/N8G9MvM4OR2)
- Redundancy and Replication
(/courses/grokking-the-system-design-interview/xV1qvj6PKk)
- SQL vs. NoSQL
(/courses/grokking-the-system-design-interview/YQIK1mDPgpK)
- CAP Theorem
(/courses/grokking-the-system-design-interview/RMkqx1Egxqz)
- Consistent Hashing
(/courses/grokking-the-system-design-interview/B81vnyp0GpY)
- Long-Polling vs WebSockets vs Server-Sent Events
(/courses/grokking-the-system-design-interview/gx7wZzWn5Vj)

Appendix



- Contact Us
(/courses/grokking-the-system-design-interview/7nnxwPxAl7i)

Data Partitioning

We'll cover the following



- 1. Partitioning Methods
- 2. Partitioning Criteria
- 3. Common Problems of Data Partitioning

Data partitioning is a technique to break up a big database (DB) into many smaller parts. It is the process of splitting up a DB/table across multiple machines to improve the manageability, performance, availability, and load balancing of an application. The justification for data partitioning is that, after a certain scale point, it is cheaper and more feasible to scale horizontally by adding more machines than to grow it vertically by adding beefier servers.

1. Partitioning Methods

#

There are many different schemes one could use to decide how to break up an application database into multiple smaller DBs. Below are three of the most popular schemes used by various large scale applications.

a. Horizontal partitioning: In this scheme, we put different rows into different tables. For example, if we are storing different places in a table, we can decide that locations with ZIP codes less than 10000 are stored in one table and places with ZIP codes greater than 10000 are stored in a separate table. This is also called a range based partitioning as we are storing different ranges of data in separate tables. Horizontal partitioning is also called as Data Sharding.

The key problem with this approach is that if the value whose range is used for partitioning isn't chosen carefully, then the partitioning scheme will lead to unbalanced servers. In the previous example, splitting location based on their zip codes assumes that places will be evenly distributed across the different zip codes. This assumption is not valid as there will be a lot of places in a thickly populated area like Manhattan as compared to its suburb cities.

b. Vertical Partitioning: In this scheme, we divide our data to store tables related to a specific feature in their own server. For example, if we are building Instagram like application - where we need to store data related to users' photos they upload and people

(<https://discuss.educative.io/c/grokking-the-system-design-interview-design-gurus/glossary-of-system-design-basics-sharding-or-data-partitioning>)