

What? Interview coaching from Googlers!

I want to know more (http://www.gainlo.co/?utm_source=blog&utm_medium=banner <http%3A//blog.gainlo.co/index.php/2016>



Design a Key-Value Store (Part I)

Since many people have emailed us saying they want to read more about system design interviews, we're going to cover more on this topic. I'm quite happy to hear a lot of feedbacks and if you have any suggestions or questions, do tell us by leaving a comment.

This week, I'm going to talk about key-value store. A key-value store is a very power technique that is used in almost every system in the world. It can be as simple as a hash table and at the same time, it can also be a distributed storage system. For instance, the underline system of Cassandra (https://en.wikipedia.org/wiki/Apache_Cassandra) is a key-value storage system and Cassandra is widely used in many companies like Apple, Facebook etc..

In this post, I'll cover topics like basic key-value storage system, distributed key-value storage and scaling issues including sharding, all of which are possible to be covered in system design interviews.

Basic key-value storage

How would you design a simple key-value storage system in a single machine?

The most straightforward thing is to use a hash table to store key-value pairs, which is how most this kind of systems work nowadays. A hash table allows you to read/write a key-value pair in constant time and it's extremely easy to use. Most languages have built-in support for this.

However, the downside is also obvious. Using a hash table usually means you need to store everything in memory, which may not be possible when the data set is big. There are two common solutions:

- Compress your data. This should be the first thing to think about and often there are a bunch of stuff you can compress. For example, you can store reference instead of the actual data. You can also use float32 rather than float64. In addition, using different data representations

like bit array (integer) or vectors can be effective as well.

- Storing in a disk. When it's impossible to fit everything in memory, you may store part of the data into disk. To further optimize this, you can think of the system as a cache system. Frequently visited data is kept in memory and the rest is on a disk.

Distributed key-value storage

The most interesting topic is definitely scaling the key-value storage into multiple machines. If you want to support big data, you will implement distributed system for sure. Let's see how we can design a distributed key-value storage system.

If you have read Design a Cache System (<http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/>), you will notice that a lot of concepts here are exactly the same.

Since a single machine doesn't have enough storage for all the data, the general idea here is to split the data into multiple machines by some rules and a coordinator machine can direct clients to the machine with requested resource. The question is how to split the data into multiple machines and more importantly, what is a good strategy to partition data?

Sharding

Suppose all the keys are URLs like <http://gainlo.co> (<http://gainlo.co>) and we have 26 machines. One approach is to divide all keys (URLs) based on the first character of the URL (after "www") to these 26 machines. For example, <http://gainlo.co> (<http://gainlo.co>) will be stored at machine G and <http://blog.gainlo.co> (<http://blog.gainlo.co>) will be stored at machine B. So what are disadvantages of this design?

Let's ignore cases where URL contains ASCII characters. A good sharding algorithm should be able to balance traffic equally to all machines. In other words, each machine should receive equal requests ideally. Apparently, the above design doesn't work well. First of all, the storage is not distributed equally. Probably there are much more URLs starting with "a" than "z". Secondly, some URLs are much more popular like Facebook and Google.

In order to balance the traffic, you'd better make sure that keys are distributed randomly. Another solution is to use the hash of URL, which usually have much better performance. To design a good sharding algorithm, you should fully understand the application and can estimate the bottleneck of the system.

Summary

There are too many interesting topics about key-value store to cover and I can hardly fit all of them into one article. As you can see, when scaling the system, there are much more issues to consider and that's why many people find distributed system difficult.

In the next post (<http://blog.gainlo.co/index.php/2016/06/21/design-key-value-store-part-ii/>), we'll continue the discussion and will cover more about scaling the system. Things like system availability, consistency will be discussed.

The post is written by [Gainlo \(http://www.gainlo.co/?utm_source=blog&utm_medium=footer-link\)](http://www.gainlo.co/?utm_source=blog&utm_medium=footer-link) - a platform that allows you to have mock interviews with employees from Google, Amazon etc..

I'd like to learn more (http://www.gainlo.co/?utm_source=blog&utm_medium=footer-link <http://blog.gainlo.co/index.php/2016>




<http://www.facebook.com/sharer.php?u=http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/> 6



<http://twitter.com/share?url=http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/&text=Design+a+Key-value+Store+%28Part+I%29+>



<http://www.linkedin.com/shareArticle?>

mini=true&url=http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/: 19  (http://reddit.com/submit?url=http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/&title=Design a Key-value Store (Part I)): 0

Subscribe

We'll email you when there are new posts here.

Related Posts:

1. Design a Key-Value Store (Part II) (<http://blog.gainlo.co/index.php/2016/06/21/design-key-value-store-part-ii/>)
2. Design a Cache System (<http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/>)
3. Design eCommerce Website (Part I) (<http://blog.gainlo.co/index.php/2016/08/22/design-ecommerce-website-part/>)
4. Create a TinyURL System (<http://blog.gainlo.co/index.php/2016/03/08/system-design-interview-question-create-tinyurl-system/>)

2 thoughts on "Design a Key-Value Store (Part I)"



NIKKI

June 20, 2016 at 6:28 am (<http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/#comment-1211>)

Good Information. Thanks!

I have a question though, after discussing these options, how do we go about designing? I mean how to go about it on the whiteboard, Any insights would be helpful.

Reply (<http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/?replytocom=1211#respond>)



JAKE

June 21, 2016 at 9:13 am (<http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/#comment-1222>)

Hi Nikki,

Thanks for your comment. Usually for such large system, once you give an overview of each components, interviewers may dig deep at a particular area. By then, you're gonna draw some graphs and provide more details. Our second post will come this week ::)

Reply (<http://blog.gainlo.co/index.php/2016/06/14/design-a-key-value-store-part-i/?replytocom=1222#respond>)

LEAVE A REPLY

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

[◀ Flatten a Linked List \(http://blog.gainlo.co/index.php/2016/06/12/flatten-a-linked-list/\)](http://blog.gainlo.co/index.php/2016/06/12/flatten-a-linked-list/)[Design a Key-Value Store \(Part II\) ▶ \(http://blog.gainlo.co/index.php/2016/06/21/design-key-value-store-part-ii/\)](http://blog.gainlo.co/index.php/2016/06/21/design-key-value-store-part-ii/)

CATEGORIES

[Coding Interview Questions \(http://blog.gainlo.co/index.php/category/coding-interview-questions/\)](http://blog.gainlo.co/index.php/category/coding-interview-questions/)[Common Pitfalls \(http://blog.gainlo.co/index.php/category/common-pitfalls/\)](http://blog.gainlo.co/index.php/category/common-pitfalls/)[Common Questions \(http://blog.gainlo.co/index.php/category/common-questions/\)](http://blog.gainlo.co/index.php/category/common-questions/)[Facebook Interview Questions \(http://blog.gainlo.co/index.php/category/facebook-interview-questions/\)](http://blog.gainlo.co/index.php/category/facebook-interview-questions/)[Mock Interview Thoughts \(http://blog.gainlo.co/index.php/category/mock-interview-thoughts/\)](http://blog.gainlo.co/index.php/category/mock-interview-thoughts/)[Others \(http://blog.gainlo.co/index.php/category/others/\)](http://blog.gainlo.co/index.php/category/others/)[System Design Interview Questions \(http://blog.gainlo.co/index.php/category/system-design-interview-questions/\)](http://blog.gainlo.co/index.php/category/system-design-interview-questions/)[The Complete Guide to Google Interview Preparation \(http://blog.gainlo.co/index.php/category/google-interview-preparation/\)](http://blog.gainlo.co/index.php/category/google-interview-preparation/)[Uber Interview Questions \(http://blog.gainlo.co/index.php/category/uber-interview-questions/\)](http://blog.gainlo.co/index.php/category/uber-interview-questions/)

Gainlo - Mock Interview With Professionals

[Visit Gainlo \(http://www.gainlo.co/?utm_source=blog&utm_medium=site-footer http%3A//blog.gainlo.co/index.php](http://www.gainlo.co/?utm_source=blog&utm_medium=site-footer)

