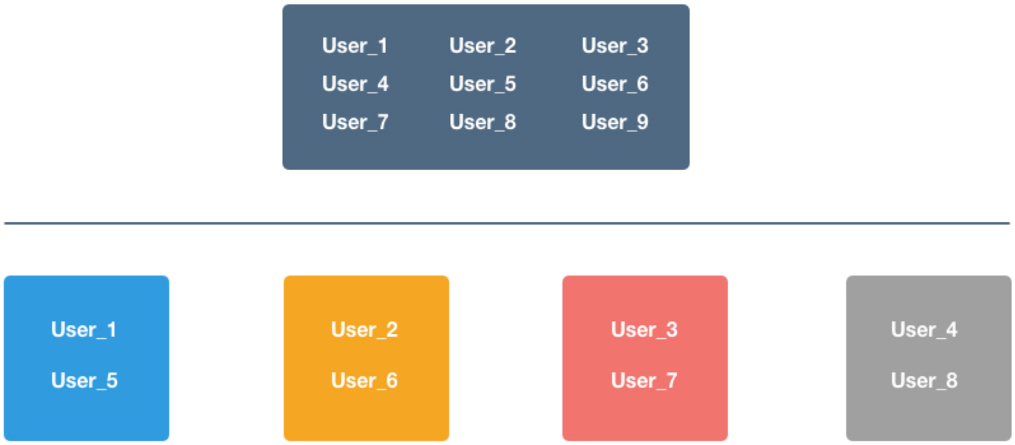
System Design (/courses/system-design) / Storage Scalability (/courses/system-design/topics/storage-scalability/) / Sharding A Database

Sharding a Database

Bookmark

Let's design a sharding scheme for key-value storage.



Features:

> *This is the first part of any system design interview, coming up with the features which the system should support. As an interviewee, you should try to list down all the features you can think of which our system should support. Try to spend around 2 minutes for this section in the interview. You can use the notes section alongside to remember what you wrote.*

Q: What is the amount of data that we need to store?
A: Let's assume a few 100 TB.

Q: Will the data keep growing over time? If yes, then at what rate?
A: Yes. At the rate of 1TB per day.

Q: Can we make assumptions about the storage of machines available with me?
A: Let's assume that machines have a RAM of 72G and a hard disk capacity of 10TB.

Q: How many machines do I have to begin with?
A: Let's assume we have 20 machines to begin with. More machines will be available on request if need be.

Q: Are all key value entries independent?
A: Yes. A typical query would ask for value corresponding to a key.

Estimation:

> *This is usually the second part of a design interview, coming up with the estimated numbers of how scalable our system should be. Important parameters to remember for this section is the number of queries per second and the data which the system will be required to handle.*
> *Try to spend around 5 minutes for this section in the interview.*

Total storage size : 100 TB as estimated earlier
Storage with every machine : 10TB
Q: What is the minimum number of machines required to store the data?

A: Assuming a machine has 10TB of hard disk, we would need minimum of 100TB / 10 TB = 10 machines to store the said data. Do note that this is bare minimum. The actual number might be higher.
In this case, we have 20 machines at our disposal.

2 ()

Q: How frequently would we need to add machines to our pool ?

A: The data grows at 1TB per day. That means that we generate data that would fill the storage of 1 machine (10TB) in 10 days. Assuming, we want to keep a storage utilization of less than 80%, we would need to add a new machine every 8 days.

0 ()

Deep Dive:

> *Lets dig deeper into every component one by one. Discussion for this section will take majority of the interview time(20-30 minutes).*

> Note : In questions like these, the interviewer is looking at how you approach designing a solution. So, saying that I'll use a distributed file system like HDFS is not a valid response. It's okay to discuss the architecture of HDFS with details around how HDFS handles various scenarios internally.

Q: Can we have a fixed number of shards?

A: One qualification for a shard is that the data within a shard should fit on a single machine completely.
As in our case, the data is growing at a fast pace, if we have a fixed number of shards, data within a shard will keep growing and exceed the 10TB mark we have set per machine. Hence, we cannot have a fixed number of shards. The shards will have to increase with time.

1 ()

Q: How many shards do we have and how do we distribute the data within the shard?

A: Lets say our number of shards is S. One way to shard is that for every key, we calculate a numeric hash H, and assign the key to the shard corresponding to H % S.
There is one problem here though. As we discussed earlier, the number of shards will have to increase. And when it does, our new number of

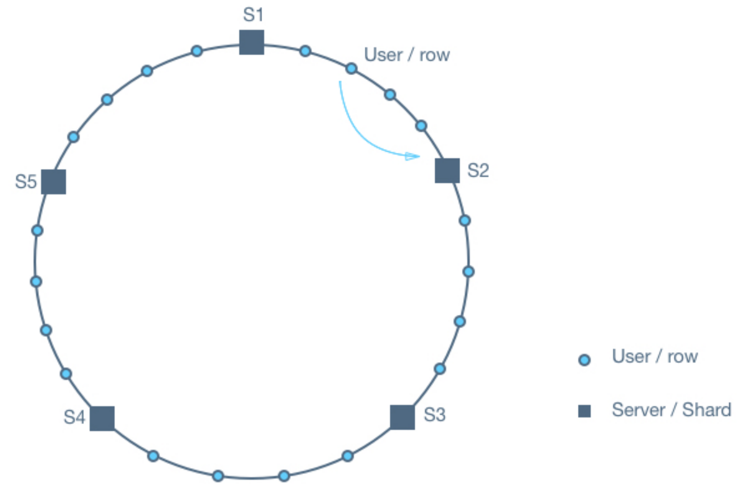
shard becomes S+1.
As, such H%(S+1) changes for every single key causing us to relocate each and every key in our data store. This is extremely expensive and highly undesirable.

1 ()

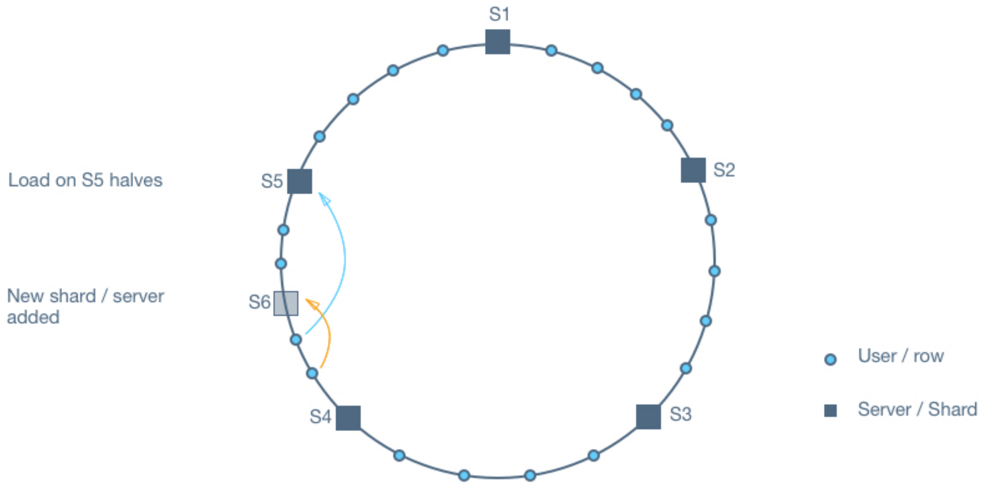
Q: Can we think of a better sharding strategy?

Hint: Consistent Hashing.

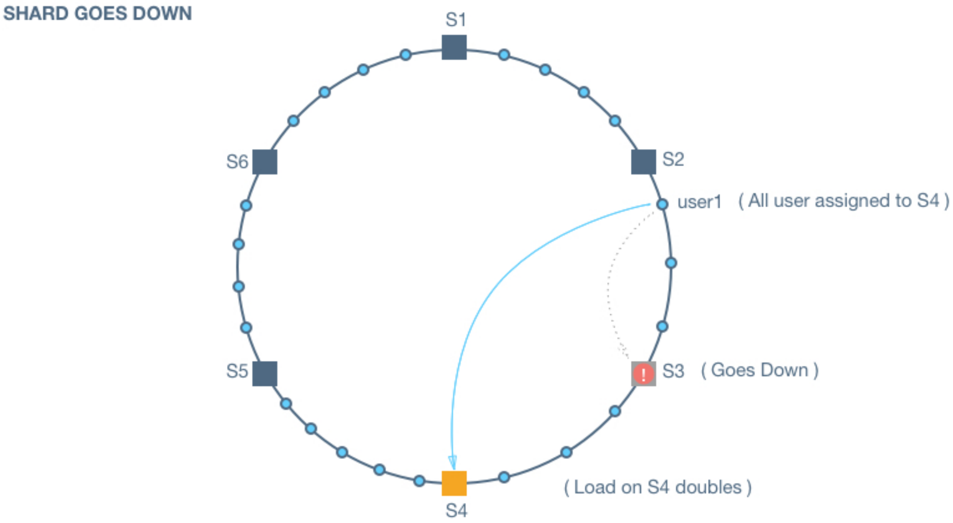
A: Consistent hashing is ideal for the situation described here. Lets explore consistent hashing here.
Let's say we calculate a 64 bit integer hash for every key and map it to a ring. Lets say we start with X shards. Each shard is assigned a position on the ring as well. Each key maps to the first shard on the ring in clockwise direction.



What happens if we need to add another shard ? Or what if one of the shard goes down and we need to re-distribute the data among remaining shards?

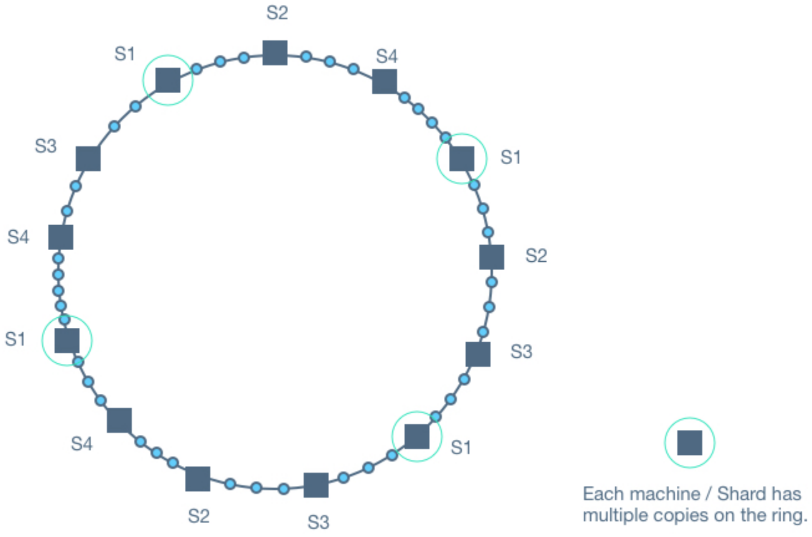


Similarly, there is a problem of cascading failure when a shard goes down.

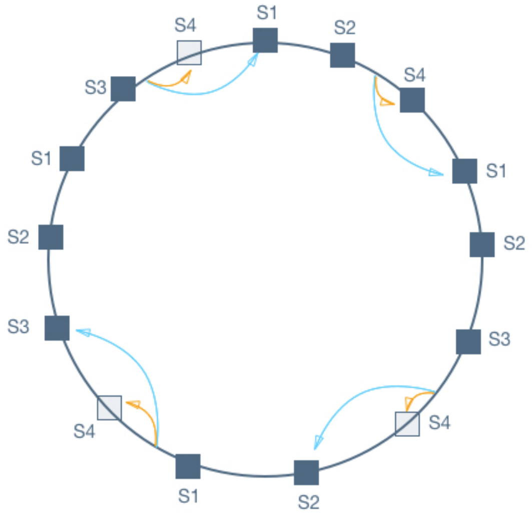


Modified consistent hashing

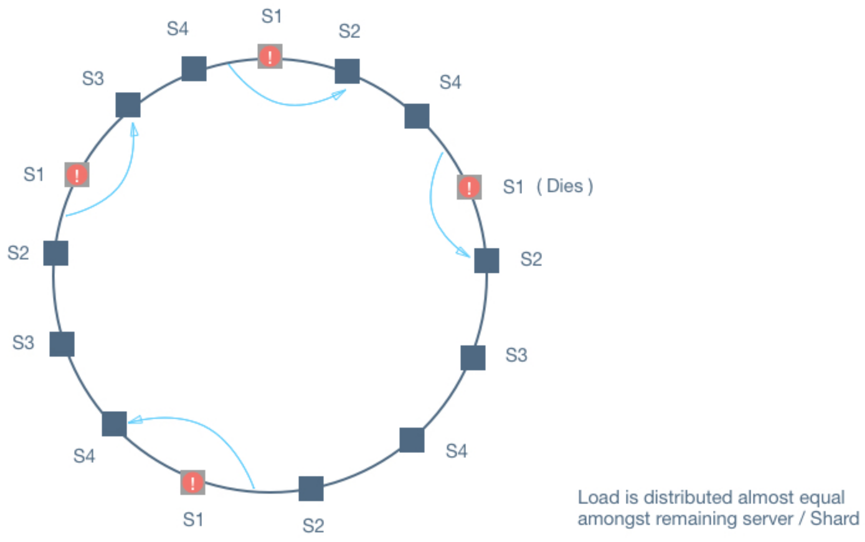
What if we slightly changed the ring so that instead of one copy per shard, now we have multiple copies of the same shard spread over the ring.



Case when new shard is added :



Case when a shard goes down : No cascading failure. Yay!


S1
S2
S4
S1 (Dies)
S2
S3
S4
S2
S1
S4
S3
S2
S1
S3
S4
Load is distributed almost equal amongst remaining server / Shard

1 ()

🏆 **You have now mastered this problem!**

Discussion

↑ vkb087 (/profile/vkb087) 4 months ago
7 Great Doc. You can find more information about consistent hashing implementation and detail here.
↓

http://www.tom-e-white.com/2007/11/consistent-hashing.html
reply

↑ Ajit (/profile/Ajit) 3 months ago
0 Java implementation of Sharading
↓ reply

↑ Ajit (/profile/Ajit) 3 months ago
0 http://sleeplessinslc.blogspot.in/2008/09/hibernate-shards-maven-simple-example.html
↓ reply

↑ skcoder (/profile/skcoder) 3 months ago
4 It looks like this is not addressing the question of how data is actually relocated. I would like to understand how data is
↓ physically moved onto the new node when it joins the cluster. Which part of the system does this? Similarly data would need to be relocated to all of the surviving nodes when a node leaves. Is this all done before the node comes online? Also when the node leaves, I suppose this is assuming there must be a redundant copy of data still online that can be moved to the surviving nodes. Would we be expected to come up with this as well in an interview?
reply

↑ swapnil_marghade (/profile/swapnil_marghade) 3 months ago
0 Storing data in multiple shards is same as replication factor. We can address this concern by using right term as
↓ "Replication factor" !
reply

↑ karan3296 (/profile/karan3296) 25 days ago
1 Not sure they are replicating here. This is just distributing data. Replication is mentioned in the dynamo paper
↓ and is a feature of HDFS.
reply

↑ AB_kyusak (/profile/AB_kyusak) 3 months ago

0 I cant understand,what advantage we gain by maintaining the multiple copies of the shrad? wht advantage does
↓ modified consistent hashing has over consistent hashing.It might sound a dumb question but i am new to the topic :)
reply

> ↑ Nivas (/profile/Nivas) 25 days ago
> 0 1.All I can think of about maintaining multiple copies per shard is it increases the availability.
> ↓ reply

> ↑ Nivas (/profile/Nivas) 25 days ago
> 0 With respect to modified consistent hashing, it makes sure that order in which shards follow each other is not
> ↓ deterministic. So if a shard fails the load will not be on a single shard, it will be shared by all subsequent shards that follow the failed instance at different places.
> reply

> ↑ karan3296 (/profile/karan3296) 25 days ago
> 0 It allows for heterogeneity too. Beefier nodes can be used to spread more virtually along the ring whereas
> ↓ weaker servers can be less prominent on the ring.
> reply

↑ kumar955 (/profile/kumar955) 3 months ago
0 I think he needs to expand the answer, what happens if a user shared size increases - do we split the data or not? Do we
↓ need to support replicas? How data center to data center replication happens etc?
reply

↑ kulkav (/profile/kulkav) 3 months ago
0 http://www.project-voldemort.com/voldemort/design.html
↓ reply

↑ Srivathsan_Venkatavaradhan (/profile/Srivathsan_Venkatavaradhan) 2 months ago
0 I think modified consistent hashing explained above is not correct. Please refer
↓ https://ihong5.wordpress.com/2014/08/19/consistent-hashing-algorithm/ . It has nothing to do with shards.
reply

↑ ramanatnsit (/profile/ramanatnsit) about 1 month ago
0 How does the system remain available during the time when a node failure occurs and redistribution of keys are in
↓ process. Is it sacrificing availability for consistency
reply

> ↑ Nivas (/profile/Nivas) 25 days ago
> 0 Each shard will have a leader/master and some replicas. During failure or redistribution the leader will be
> ↓ changed. So availability is not sacrificed. But this may result in increased load in the machine where the elected replica is located if not managed at right time.
> reply

↑ eipie (/profile/eipie) 22 days ago
0 What is this obsession with 72GB RAM? Isn't is better / easier to pick powers of two? Or maybe perhaps 100GB treating
↓ 1GB = 1000MB for simplicity of calculations?
reply

Write your comment here. Press Enter to submit.

We are in beta! We would love to hear your feedback. **Loved InterviewBit? Write us a testimonial. (http://www.quora.com/What-is-your-review-of-InterviewBit)**

About Us (/pages/about_us/) | FAQ (/pages/faq/) | Contact Us (/pages/contact_us/) | Terms (/pages/terms/) | Privacy Policy (/pages/privacy/)

System Design Questions (/courses/system-design/) | Google Interview Questions (/problems/google-interview-questions) |

Facebook Interview Questions (/problems/facebook-interview-questions) | Amazon Interview Questions (/problems/amazon-interview-questions) |

Microsoft Interview Questions (/problems/microsoft-interview-questions)

Like Us (https://www.facebook.com/interviewbit) Follow Us (https://twitter.com/interview_bit) ✉ Email (mailto:hello@interviewbit.com)