NoSQL & Amazon DynamoDB

INF 551

Wensheng Wu

Roadmap

- NoSQL 
- Amazon DynamoDB
- Consistent hashing

Relational databases

- Mature & stable
  - Suitable for mission-critical applications, e.g., banking
- Feature-rich versatile query language: SQL
- ACID properties
  - In particular, strong consistency

Challenges

- Internet-scale systems & applications
  - E-commerce systems (e.g., Amazon)
  - Social media apps (e.g., Facebook, LinkedIn)
- Big data
  - Often unstructured or semi-structured
- New workloads
  - Write/update-heavy
  - Demand high availability
  - Can tolerate weak consistency

NoSQL databases

- NoSQL: Not only SQL

- Key features
  - Flexible (non-relational) data model
  - Can be easily scaled out (horizontal scalability)
  - Data replicated over multiple servers
  - Weaker consistency model
  - High availability

Scale out vs. scale up

- Scale up (vertical scaling)
  - Beefing up a computer system
  - E.g., adding more CPUs, RAMs, and storage
- Scale out (horizontal scaling)
  - Adding more (commodity) computers
  - Moving some data to new computers

Types

- Key-value, tuple/row stores
  - E.g., Redis (key-value), Amazon DynamoDB (row)
- Document stores (e.g., JSON/XML documents)
  - E.g., Apache CouchDB, MongoDB, XML databases
- Extensible record/column stores
  - E.g., Google BigTable, Apache Cassandra & HBase

Types

- Graph stores
  - E.g., Neo4J, Apache Spark (GraphX distributed graph data computation)

Extensible record store

- Similar to relational database
  - With rows & columns
  - Columns may be grouped into column family
- But different rows may have different columns
- Also called "wide column store"

Roadmap

- NoSQL
- Amazon DynamoDB 
- Consistent hashing

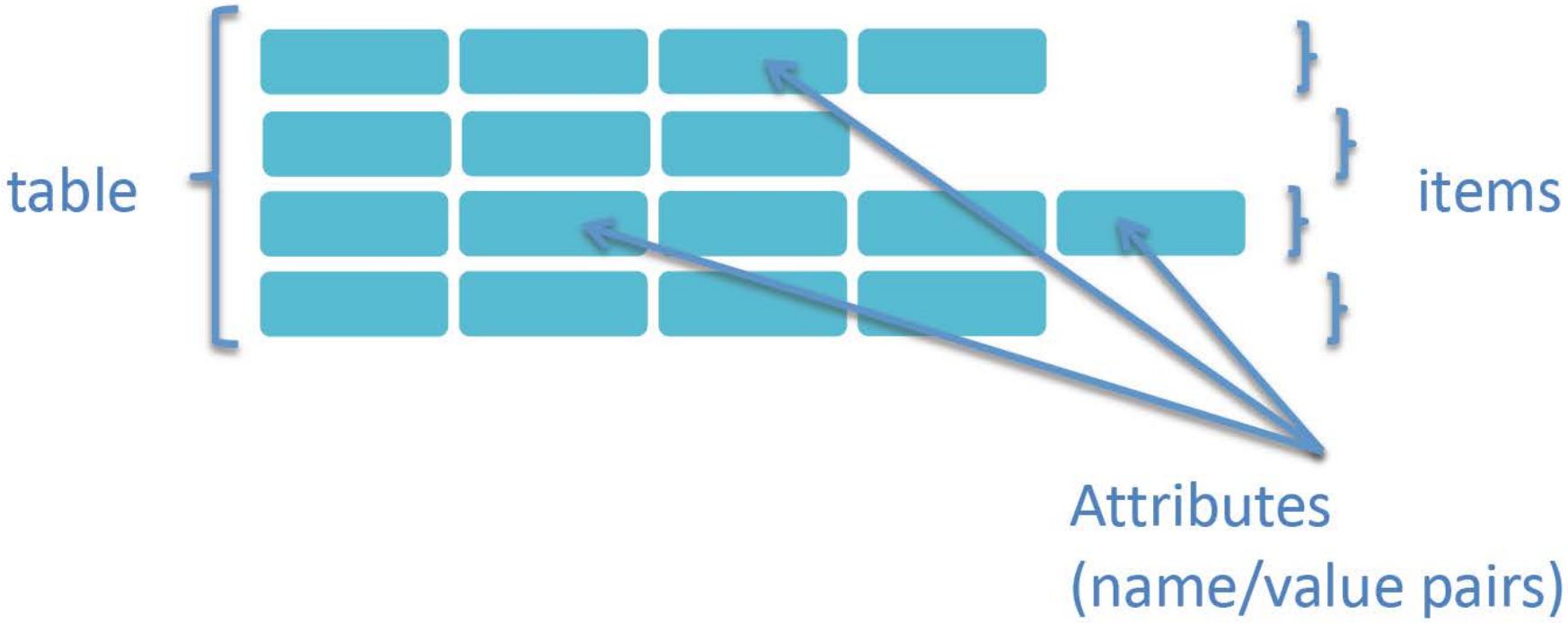
Amazon DynamoDB

- Schema-less: no predefined schema
- A database contains a single table (e.g., Music)
- The table consists of a set of items
  - E.g., a set of music CDs
- Each item contains a set of attributes
  - E.g., artist, title, year of CD

Items

- Similar to rows in relational databases
- But different rows may have different set of attributes
- Max size of an item: 400K
- No concept of columns in DynamoDB

DynamoDB table structure



Primary key

- Each item is uniquely identified by a primary key

- Primary key consists of
  - partition key
  - (optional) sort key

Partition key

- Partition key
  - Partition (by hashing) the data across hosts for scalability & availability
- Pick an attribute with wide range of values & evenly distributed patterns for partition key
  - E.g., user ID
- E.g., artist name
  - Hash function may put "Rod Stewart" and "Maria Kelly" in the same partition

Sort key

- Allow searching within a partition
- E.g., year
  - So primary key = artist + year
- This allows search all CDs by a specific artist and produced in certain years

DynamoDB is not good for…

- Ad-hoc query
  - Since it does have query language like SQL & does not support joins
- OLAP
  - Require joining of fact and dimension tables
- BLOB (binary large objects) storage
  - E.g., images, videos
  - Better suited for Amazon S3

Example

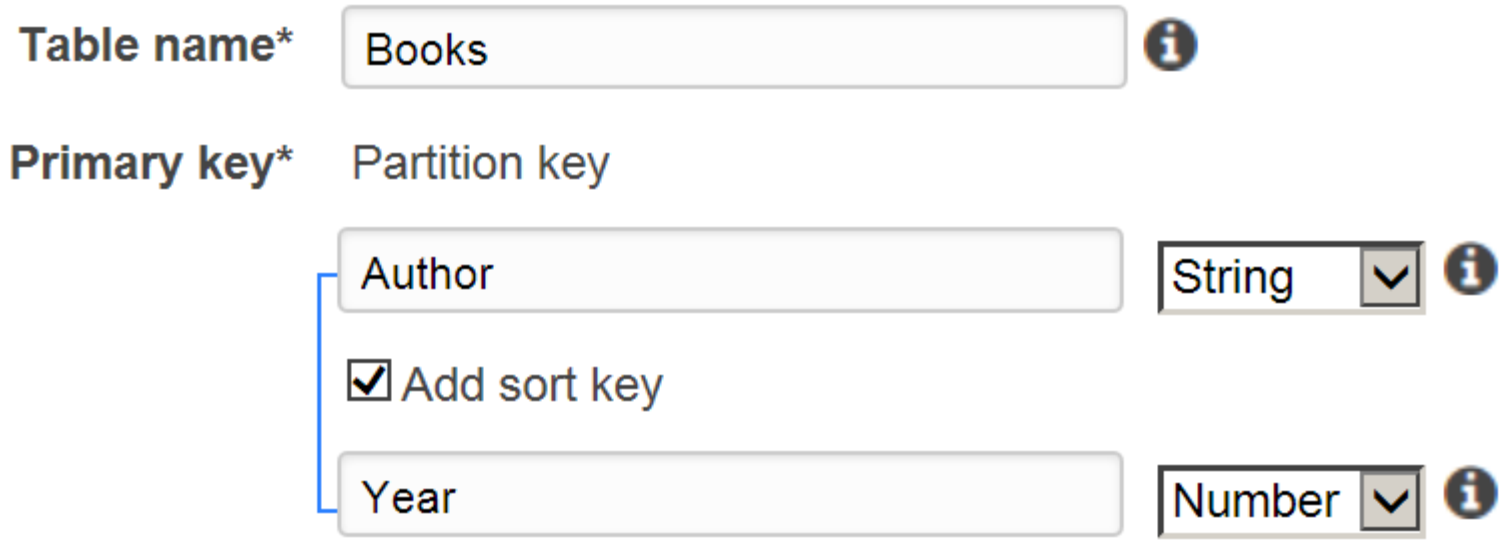
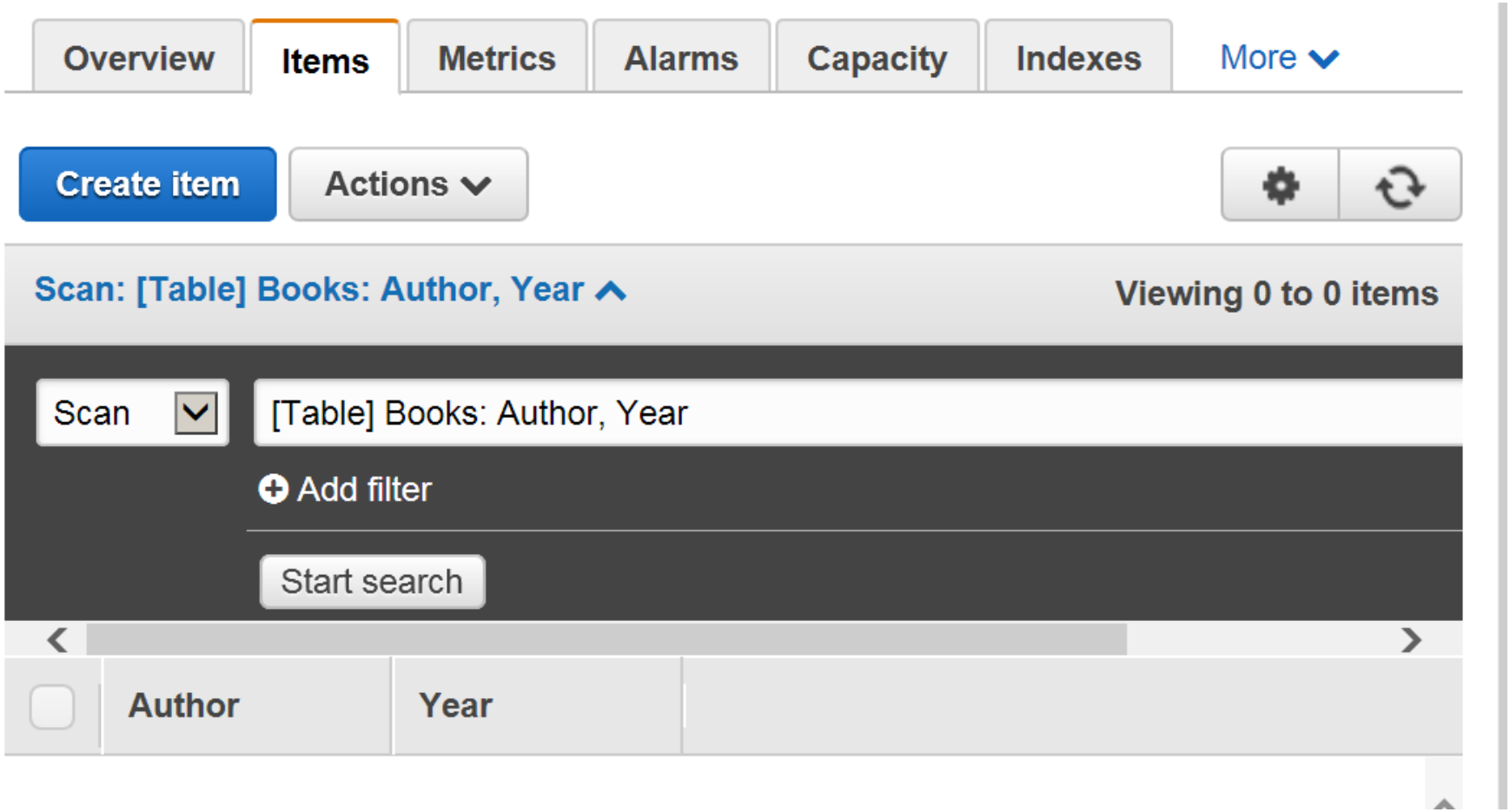
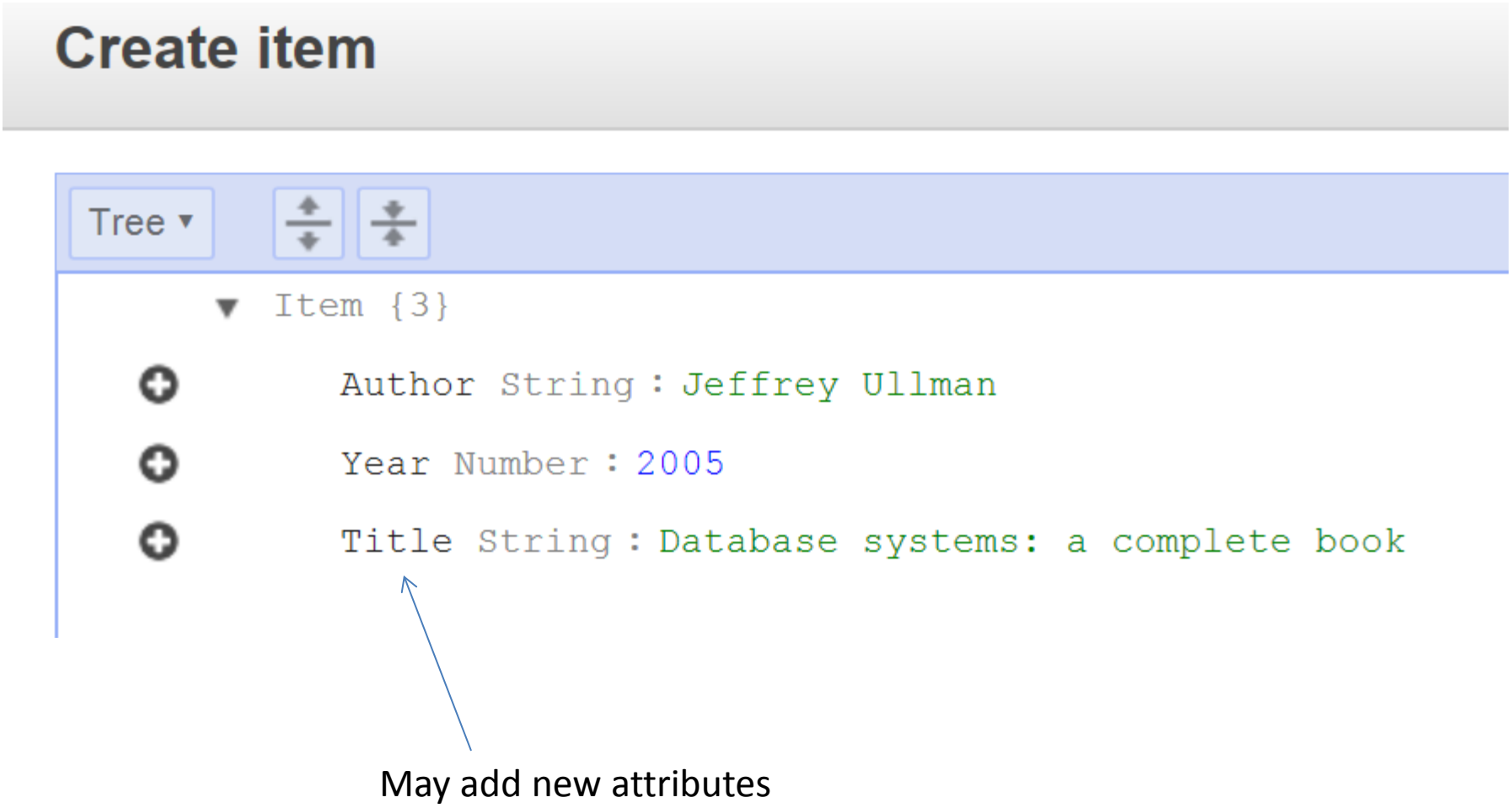


Table name*
Books
Primary key*
Partition key
Author
String
Add sort key
Year
Number

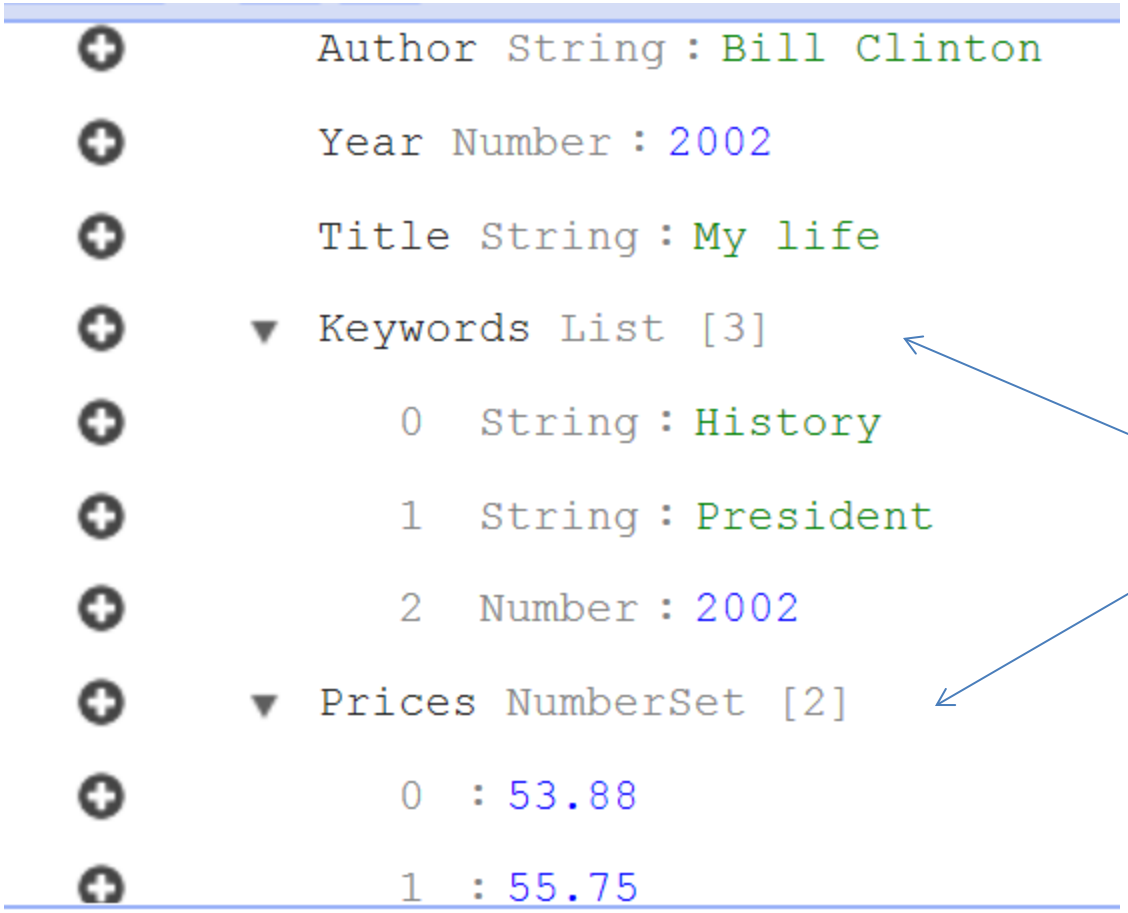
Example


Overview
Items
Metrics
Alarms
Capacity
Indexes
More
Create item
Actions
Scan: [Table] Books: Author, Year
Viewing 0 to 0 items
Scan
[Table] Books: Author, Year
Add filter
Start search
Author
Year

Example


Create item
Tree
Item {3}
Author String : Jeffrey Ullman
Year Number : 2005
Title String : Database systems: a complete book
May add new attributes

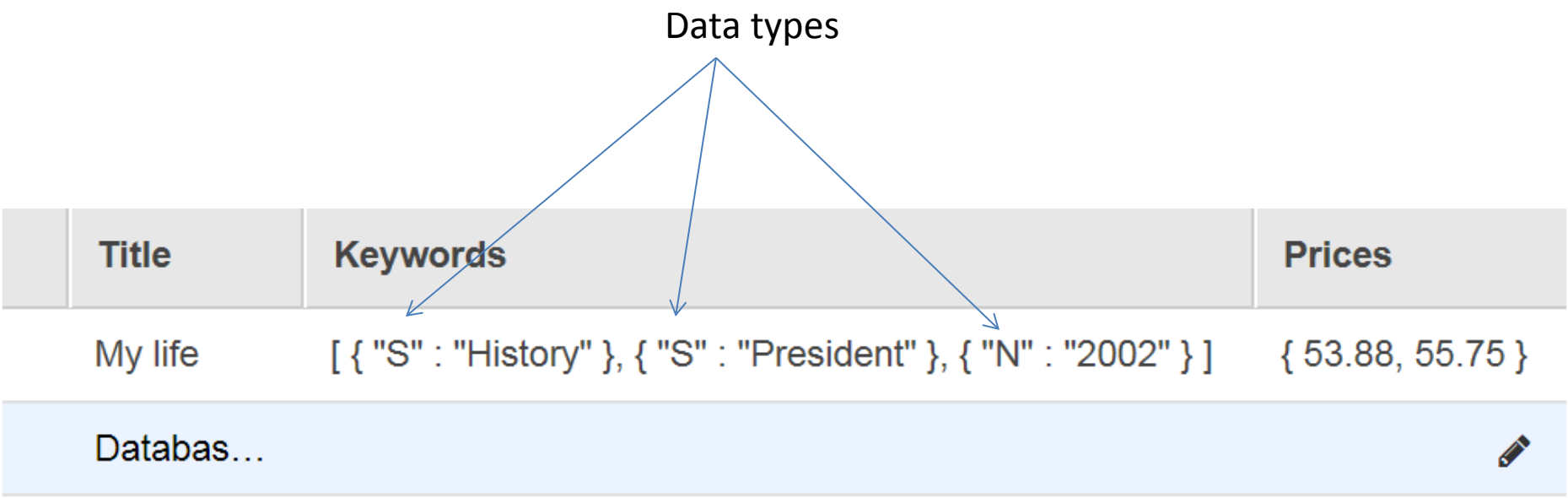
Example



Value can be a list or a set

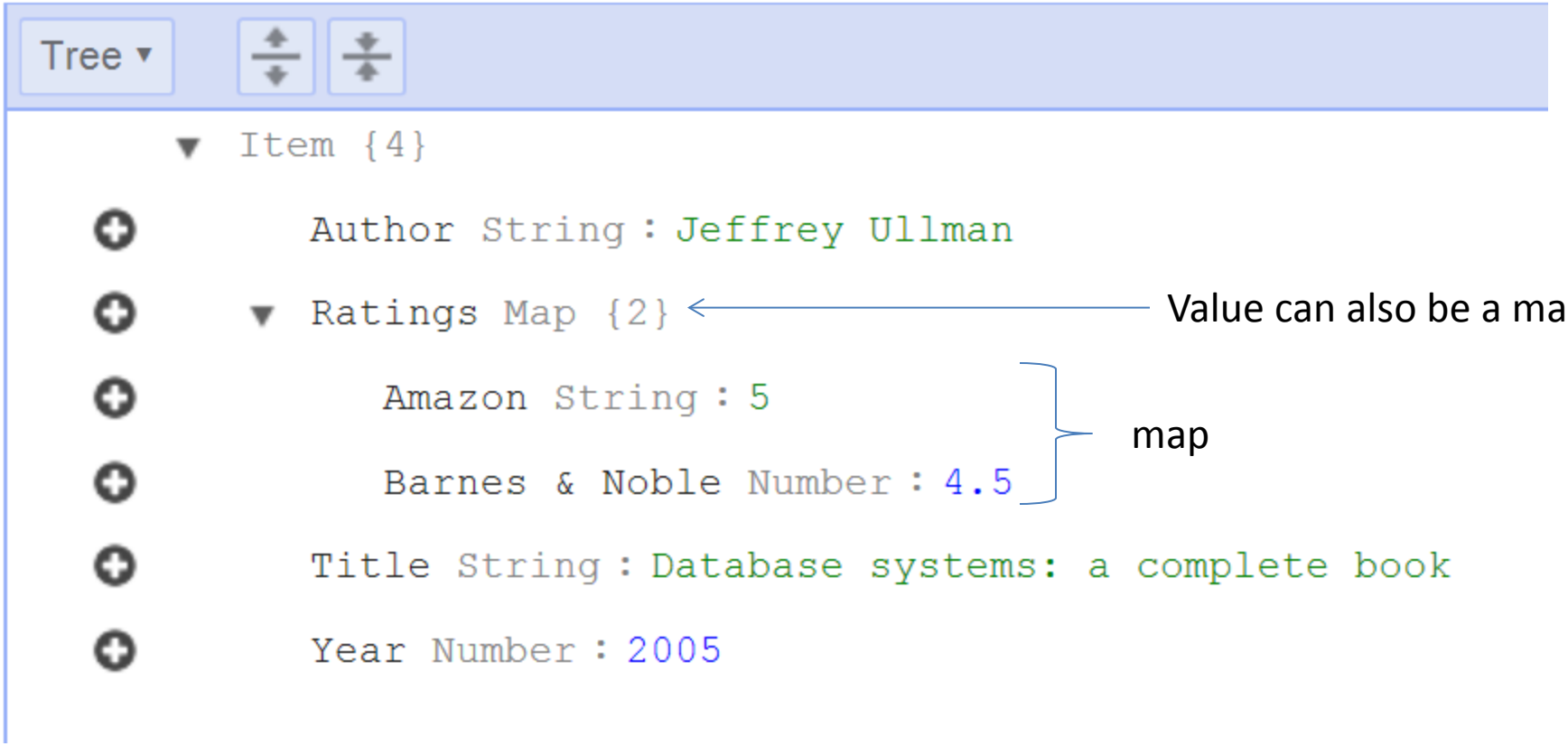
List: ordered, heterogeneous
Set: unordered, homogeneous

Example

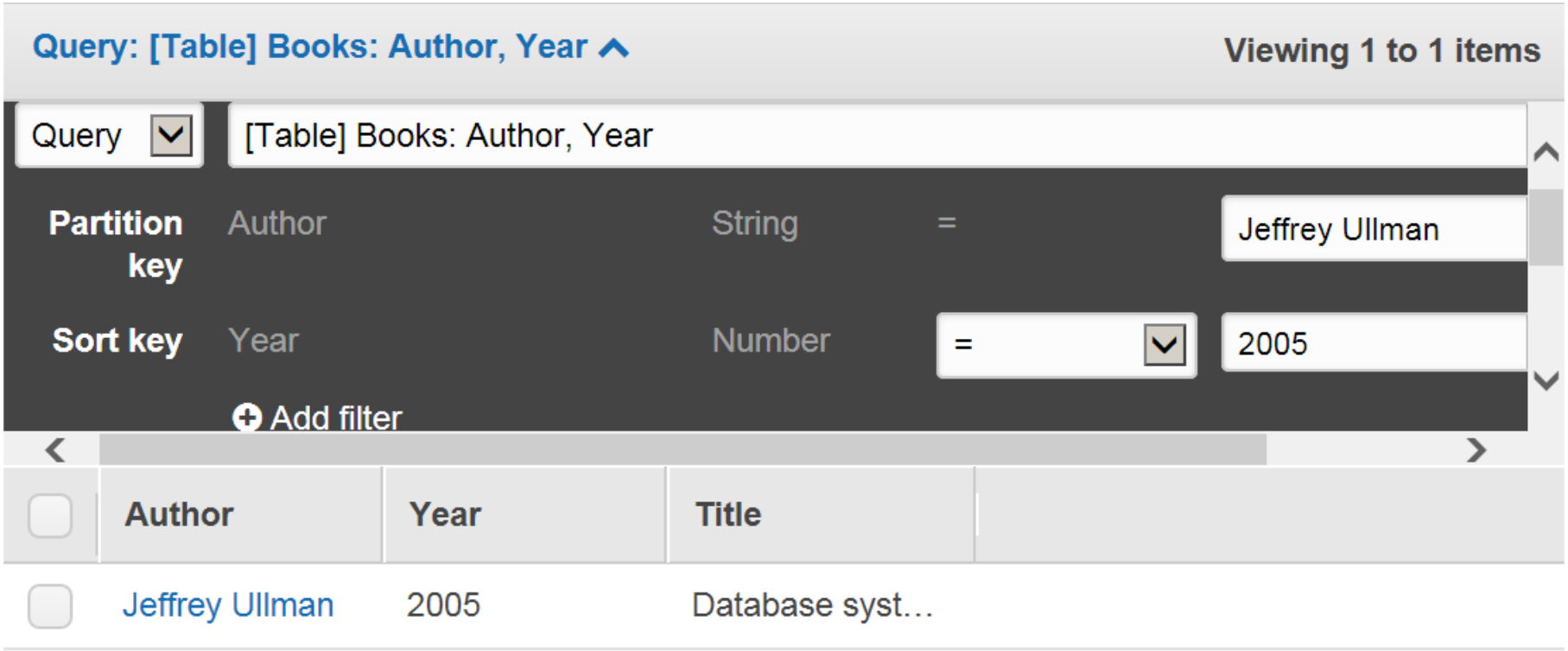

Data types
Title
Keywords
Prices
My life
[{ "S" : "History" }, { "S" : "President" }, { "N" : "2002" }]
{ 53.88, 55.75 }
Databas…

Example

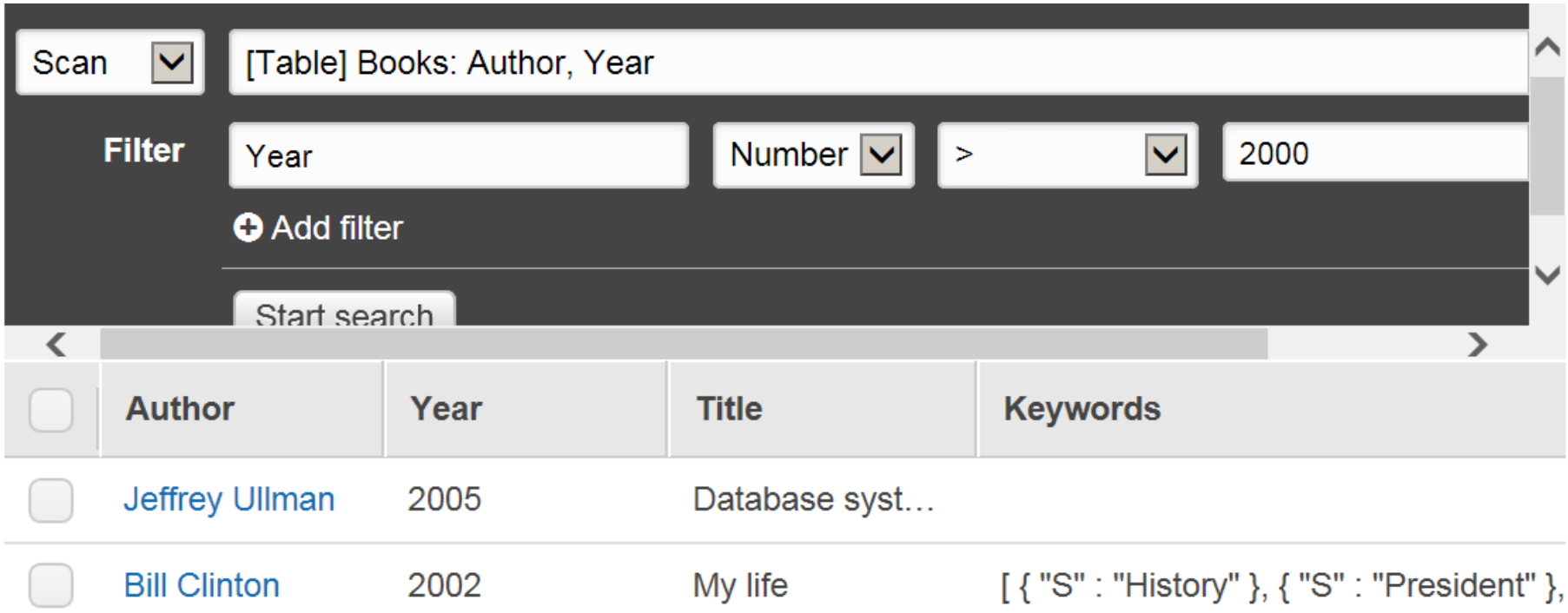
Map: contains a list of key-value pairs



Query


Query: [Table] Books: Author, Year
Viewing 1 to 1 items
Query
[Table] Books: Author, Year
Partition key
Author
String
=
Jeffrey Ullman
Sort key
Year
Number
=
2005
Add filter
Author
Year
Title
Jeffrey Ullman
2005
Database syst…

Scan


Scan
[Table] Books: Author, Year
Filter
Year
Number
>
2000
Add filter
Start search
Author
Year
Title
Keywords
Jeffrey Ullman
2005
Database syst…
Bill Clinton
2002
My life
[{ "S" : "History" }, { "S" : "President" },

Roadmap

- NoSQL
- Amazon DynamoDB
- Consistent hashing 

Hash function

- A hash function h(x) = y
  - x: a value of arbitrary size or length, e.g., a string of characters
  - y: a fixed-size or fixed-range value, e.g., 128 bits, [0, n-1], [0, 1]

- For example
  - h(s) = (sum of values of characters in string s) % 11

Partitioning by hashing

- Items are stored in different servers based on hash values of their partition keys: h(k)

- Suppose there are n nodes in a cluster
- h(k) is typically a very big number, e.g., 128 bits

- Assign item with key k to node: h(k) % n

Problem in scaling out

- The number of servers (n) grows
- Key k is now assigned to h(k) % (n + 1)
  - Which may be very different from h(k) % n
- Consequence:
  - Almost all items (or keys) need to be moved (mapped) to different servers

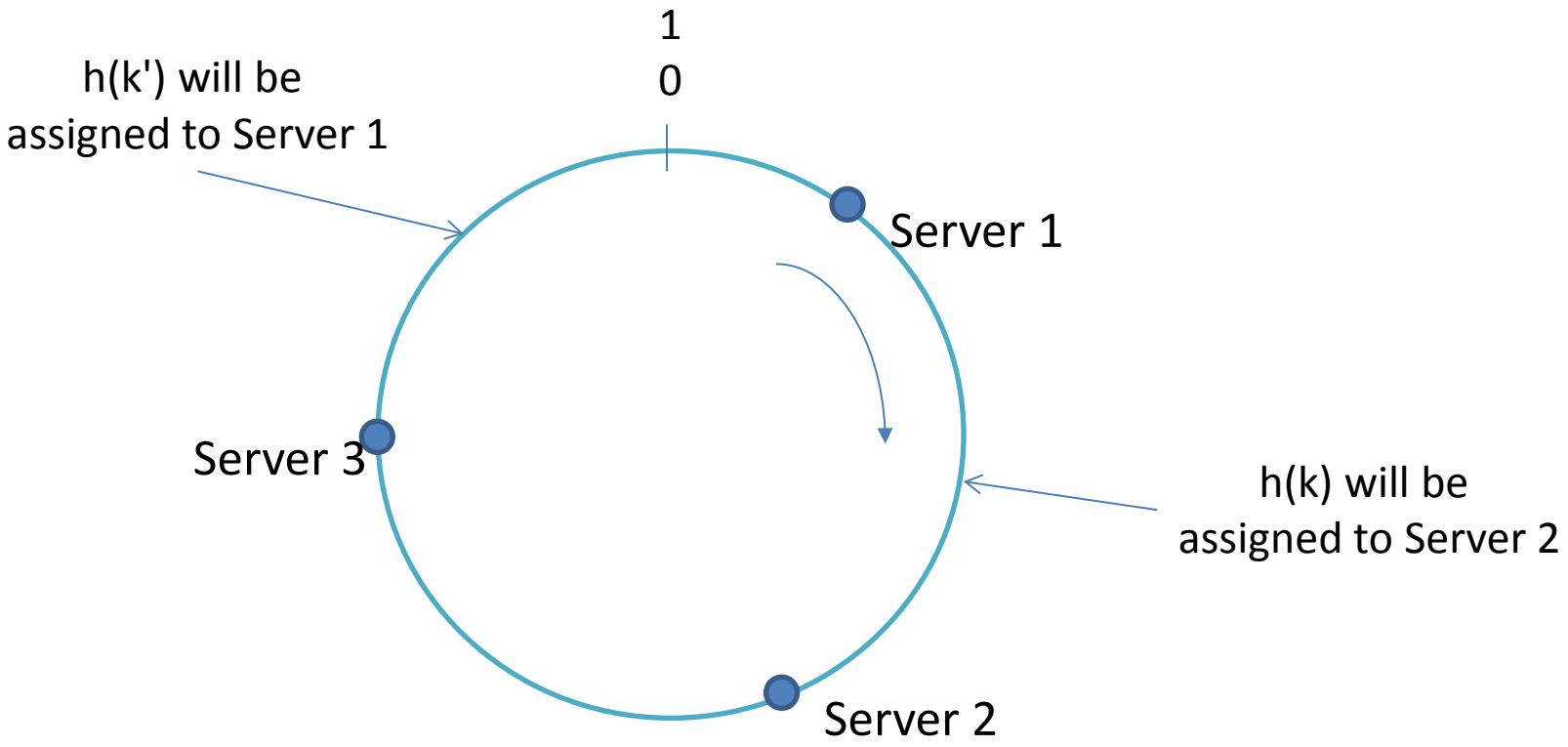
Example

- Suppose h(k) = 12345
- Currently, 4 nodes in a cluster
  - h(k) % 4 = 1
- Adding one more node
  - h(k) % 5 = 0

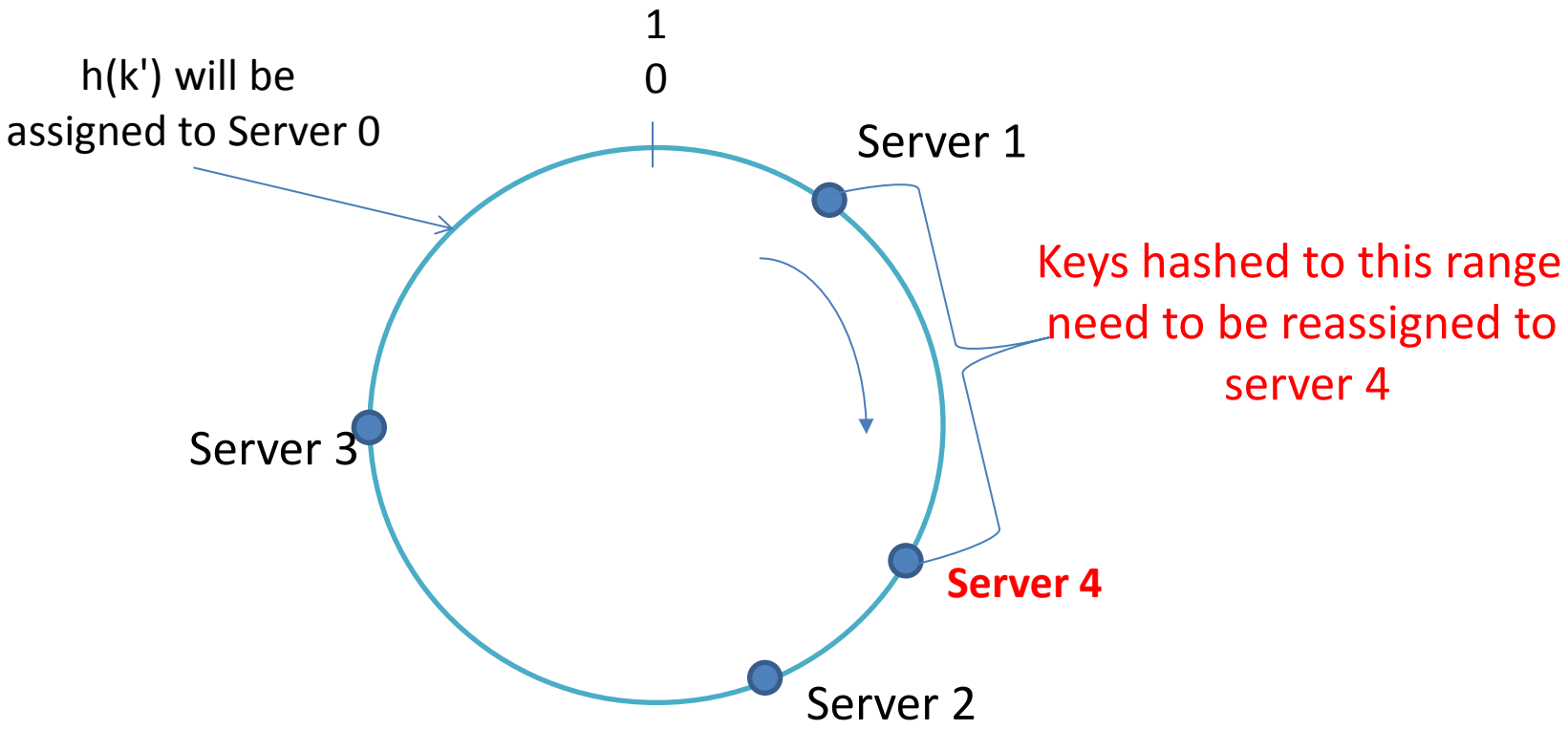
Consistent hashing

- Hash key to a value in a fixed range, say [0, 1]
  - E.g., h'(k) = h(k) / max(h(k))
- Assign each server to a point in the same range
  - E.g., by hashing machine serial to range [0, 1]
- Assign each key to the first machine with a larger hash value
  - If over the range, find next one from the beginning

Assign keys to machines



Adding a new server



How much improvement?

- m = # of keys
- n = # of servers

- m/n = # of keys to be moved on average

- Typically, m/n << m with a large n
  - And increasing n => reducing movement

References

- Dynamo: Amazon's Highly Available Key-value Store
  - http://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf
- Best Practices for Migrating from RDBMS to Amazon DynamoDB
  - https://d0.awsstatic.com/whitepapers/migration-best-practices-rdbms-to-dynamodb.pdf