Architectures for Big Data

Key-value stores and Redis

Madhulika Mohanty
Inria

DataAI Master 2022
Institut Polytechnique de Paris

Slides courtesy of Silviu Maniu, Ioana Manolescu and Paweł Guzewicz

Key-value stores

- Popular systems for Big Data management, part of the **NoSQL** movement – especially on the cloud.
- Main idea:

  Trade simplicity for speed and scale

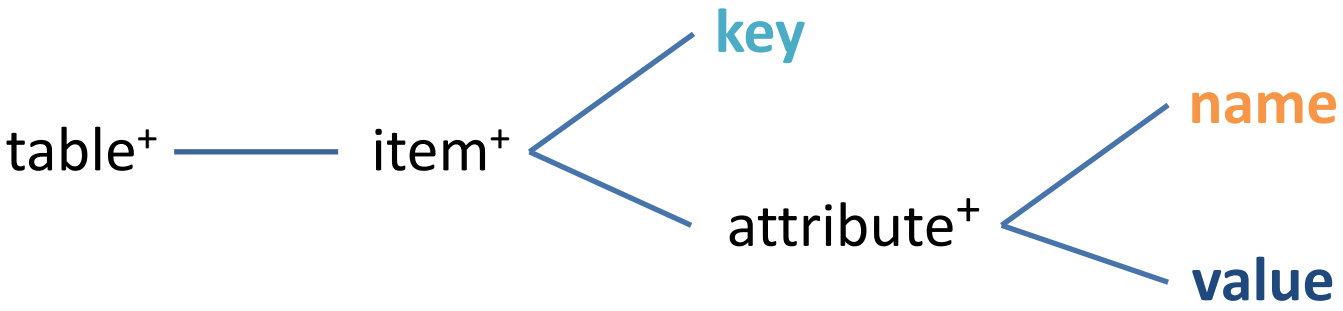
  Extremely simple data model
  - **key** = short byte sequence / integer
  - **value** = byte sequence (may recognize integers)
- No QL. Operations: **PUT(k, v)** and **v=GET(k)**
- **ACID** properties depending on the system; at least atomic PUT and GET
  - Some are in-memory -- durability up to implementation

Key-value data models

- Simplest model:
  - One key – one value
- Extensions:
  - **Organization**: key-value pairs belong to « collections » or « databases » or « tables »
  - **Multiplicity**: set or list of values
  - **Internal structure**:
    - One key – a list of *attributes*
    - Each attribute has a *name* and a *value / set of values*

Sample key-value data model: DynamoDB

- Provided by Amazon Web Services (AWS)



- Naming may vary (there is no standard). See doc for more details.
- Although it is called « table », *items in the same table may have nothing in common – no definite schema*!

Redis: one of the most popular key-value stores

- Data Types (https://redis.io/docs/data-types/tutorial/):
  - String
  - Hash (a set of key-value pairs on the same key)
  - List
  - (Sorted) Set
  - No nesting
- Operations:
  - Put, get
  - Set operations (union, intersection)
  - List operations: left/right push/pop (→queue / stack)
  - Arithmetic operations (attempts type conversion to integers) – Increment/decrement