Big Data Architectures

Key-value stores and Redis

Paweł Guzewicz
École Polytechnique, Inria

M2 Data and Knowledge 2019/2020
Université Paris Saclay

Slides courtesy of Silviu Maniu and Ioana Manolescu

Key-value stores

- Relatively recent class of systems, developed as part of the **NoSQL** movement
- Main idea:

  Trade simplicity for speed and scale

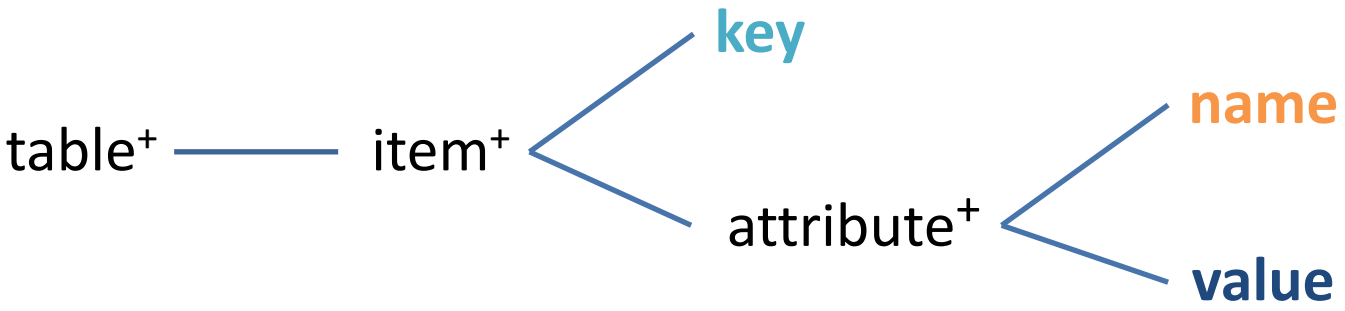
  Extremely simple data model
  - **key**=short byte sequence / integer
  - **value**=byte sequence (may recognize integers)
- No QL. Operations: **PUT(k, v)** and **GET(k,v)**
- **ACID** properties depending on the system; at least atomic PUT and GET
  - Some are in-memory thus no durability at all

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.
- Although it is called « table », *items in the same table may have nothing in common*!
- The interface is very similar to the so-called « Big Tables » (to be seen)

Redis: one of the most popular key-value stores

- Data model:
  - Hash (a set of key-value pairs on the same key)
  - List
  - Set
  - Values cannot be lists nor sets (no nesting!)
  - Databases
- Operations:
  - Put, get
  - Set operations (union, intersection)
  - List operations: left/right push/pop (→queue / stack)
  - Arithmetic operations (attempts type conversion to integers)