NoSQL in a glimpse

https://github.com/federicodassereto/MAIA_NoSQL

Federico Dassereto*
federico.dassereto@edu.unige.it

Dibris

*Giovanna Guerrini
*Dan Suciu

UNIVERSITÀ
DEGLI STUDI
DI GENOVA

Not every data management/analysis problem is best solved using a traditional DBMS.

Database Management System (DBMS) provides efficient, reliable, convenient, and safe multi-user storage of and access to massive amounts of persistent data.

Alternative to traditional relational DBMS

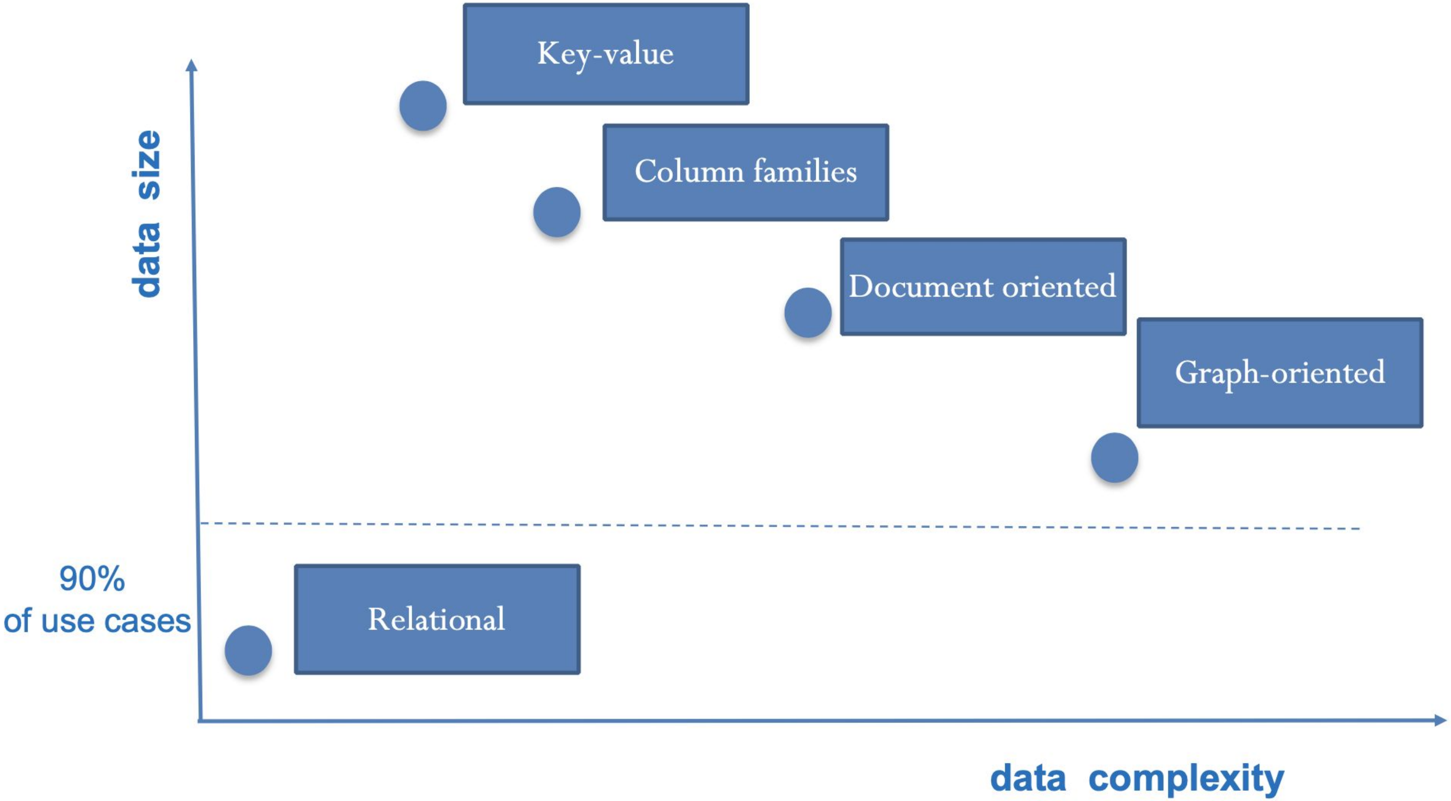
- Flexible schema
- Quicker/cheaper to set up
- Massive scalability
- Relaxed consistency higher performance & availability

Drawbacks

- No declarative query language => more programming

- Relaxed consistency => fewer guarantees

NoSQL data models



Focus on

Document Oriented

- CouchDB
- MongoDB
- SimpleDB

Graph Oriented

- Neo4J
- Gremlin
- ....

Document-oriented Data Model

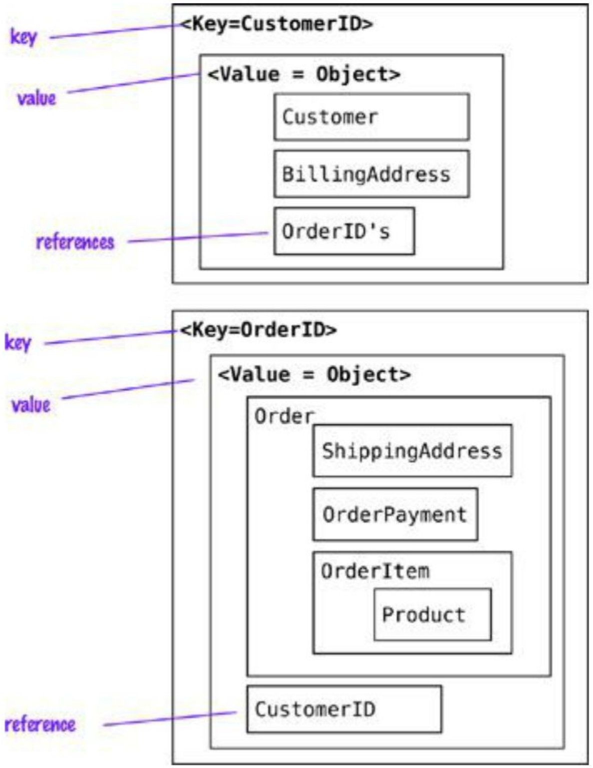
Each data instance is represented in the form (key, document)

- key is an identifier
- document is the aggregate, corresponding to a set of <name, nested-document> pairs

Usually nested-documents are represented according to a semi-structured data model like XML, RDF, JSON

Structure of the aggregate visible

Document-oriented Data Model



```
# Customer object
{
“1": {
"name": "Martin",
"billingAddress": [{"city": “Chicago"}]
      }
}


# Order object
{
"99": {
              "order":{
              "orderDate":"Nov-20-2011",
              "orderItems":[{"productId":27, "price": 32.45}],
              "orderPayment":[{"ccinfo":"1000-1000-1000-1000",
                                        "txnId":"abelif879rft"}],
              "shippingAddress":{"city":"Chicago”}
                                  }
    customerId": 1
      }
}
```

Sub-document

Document-oriented - Interaction

Basic lookup based on the key and field names in the aggregates

- Void put(key, document)
- Document get(key)
- Void set(key, name, value)
- Document get(key, name)
- Void remove(key)

Example

We can query inside (nested) documents, at any level:

- find all orders issued in November 2011 (condition on OrderDate)
- find orderPayment information associated with order 99

```
# Customer object
{
“1": {
"name": "Martin",
"billingAddress": [{"city": “Chicago"}]
      }
}


# Order object
{
"99": { "order":{
                    "orderDate":"Nov-20-2011",
                    "orderItems":[{"productId":27, "price": 32.45}],
                    "orderPayment":[{"ccinfo":"1000-1000-1000-1000",
                                                "txnId":"abelif879rft"}],
                    "shippingAddress":{"city":"Chicago”}
                                          }
     customerId": 1
       }
}
```

Graph data stores

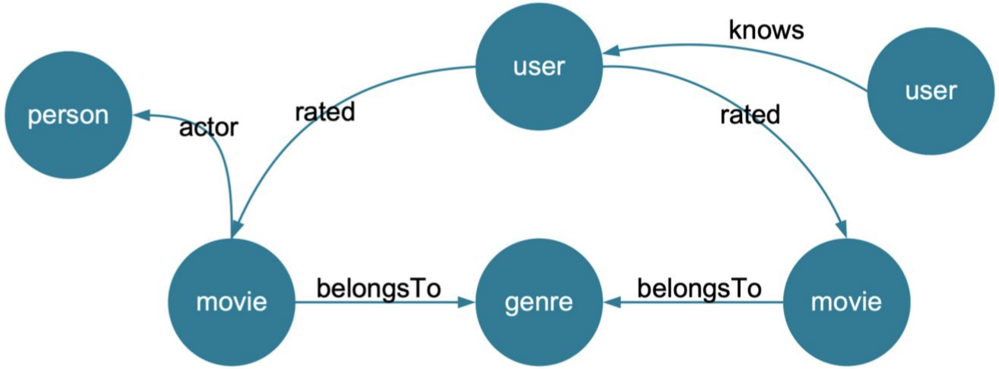
- Graph databases are motivated by a different frustration with relational databases
  - Complex relationships require complex join
  - Modeling of many-to-many relationships

- Goal
  - Capture data consisting of complex relationships
  - Data naturally modelled as graphs
    - Social graphs
    - Web & Semantic Webs
    - Networks (roads, rails, …)

  - A graph is a generic, intuitive, and well-known data structure

Graph data stores: Vertices and Edges

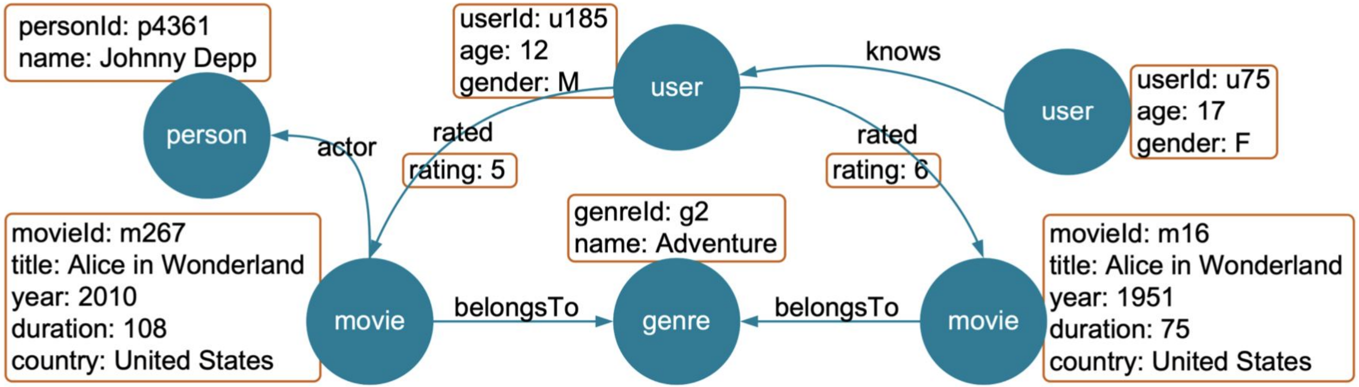
- A graph G=(V,E) is a pair of vertices and edges connecting them



Vertices (or nodes)
– Label: E.g. person, movie

Edges
– Type: E.g. actor, rated, belongsTo

Properties



- Vertices (or nodes)
  - Label: E.g. person, movie
  - Properties: E.g. name, age
- Edges
  - Type: E.g. actor, rated, belongsTo
  - Properties: E.g. rating

Schema less data model

- Each entity (object) can have different properties, just like a document database
- Any entity can have a relationship with any other entity
- Relationships have a type, and any pair of entities can have a relationship of any type
- Relationships have properties, so can be thought of as entities that join other entities
- Entity pairs can have more than one relationship

Exercises

mongoDB

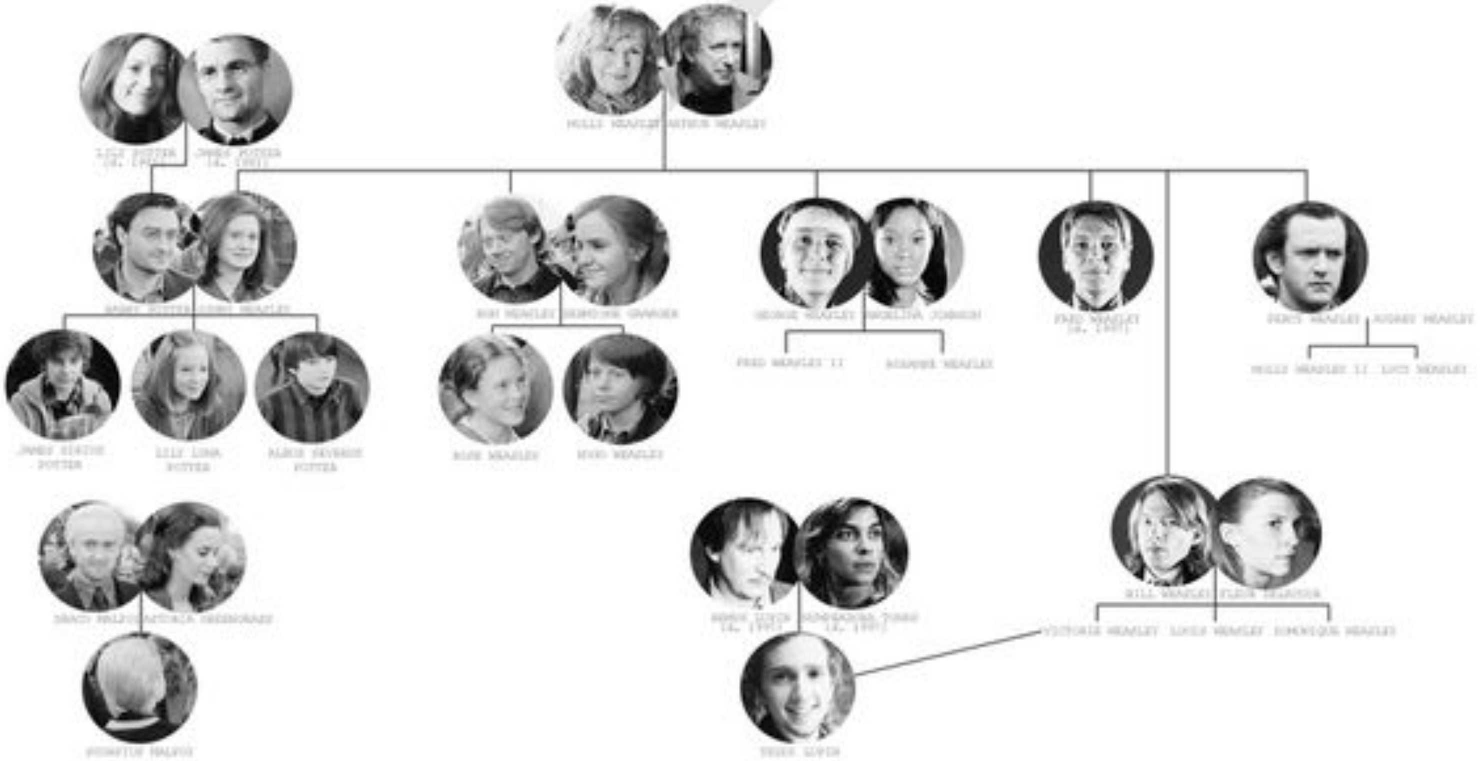
Harry Potter

Gremlin

$G = (V, E)$

how i met your mother

Harry Potter
FAMILY TREES



Why mapping it via documents instead of graph?

Workflow

Create a document for each family (at least 3)

Query the DB to obtain simple informations

Which MongoDB function is the best one?

Try to add new unrelated documents

Try to retrieve documents by common features

Is it easier than joins on SQL?

How I met your mother relationships among characters

Workflow

Create a graph G=(V,E) where V are characters and E are relations

Query the DB to obtain simple informations

How do we traverse the graph?

Try to filter nodes by ‘Location’

Try to add new Nodes and Relations

Try to simulate an SQL join

Is it easier than joins on SQL?