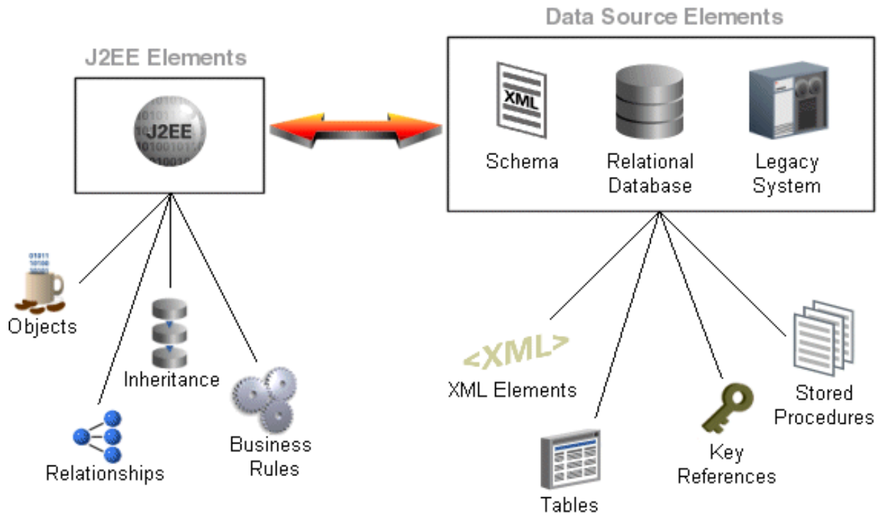
On the performance impact of using JSON, beyond impedance mismatch

Moditha Hewasinghage
Sergi Nadal
Alberto Abelló

Problem of impedance mismatch

- Overhead generated by transformation from internal structures, to relational, and finally to programming structures
  - OO concepts are mathematical graphs
  - Relational schemas are tabular

- Flexible data formats to overcome the issue (JSON)
  - Can be directly mapped from disk to memory
  - Breaking the normal forms is encouraged
    - Nested structures
    - Arrays
  - Skip schema declaration



Oracle dev guide (https://www.oracle.com)

Denormalization and schemaless

- Is it a conscious database design choice ?
- Is it a limitation of NOSQL systems ?
- Need to consider the benefits and drawbacks of different alternatives
- The flexibility of NOSQL comes at a price
  - Each associated design choice change the physical representation
  - Impact on performance
- Today, binary design decisions based on rules
- Important to consider the pros and cons of different alternatives during the design process

Our contribution

- Empirically quantify the impact of design choices in semi-structured data
- Focus on JSON (most popular data format used on the Web)
- Identify the main design characteristics of semi-structured data and compare them to their structured counterpart
- Compare the different alternatives in a relational (PostgreSQL) and non-relational (MongoDB) DBMS

Representation differences

Schema variability

- Metadata representation
- Attribute optionality

Schema declaration

- Data types
- Integrity constraints

Structural complexity

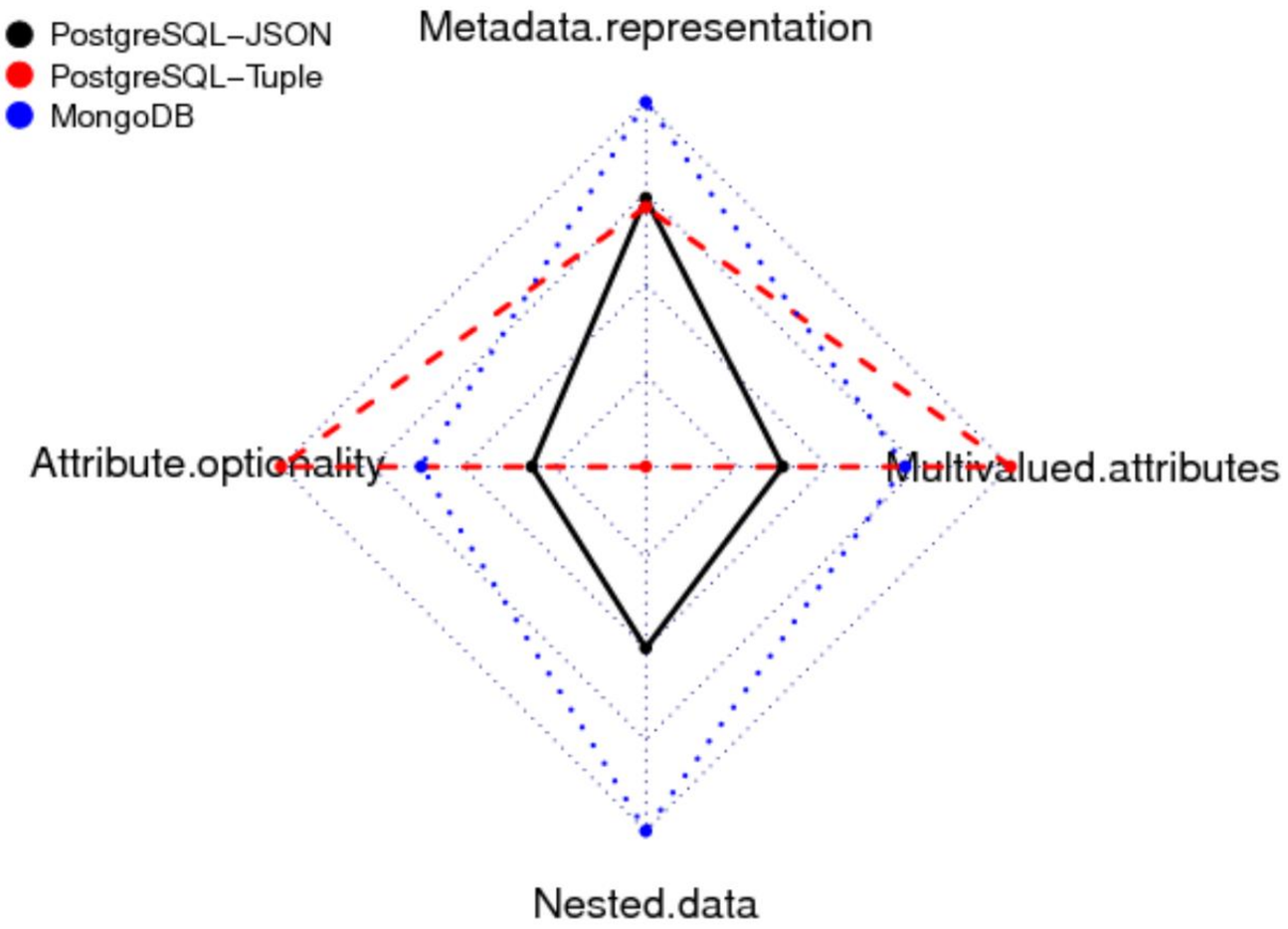
- Nested structures
- Arrays

Experimental setting

- MongoDB v4.2 (JSON) and PostgreSQL v12 (Relational & JSON) default parameters on databases except no compression in MongoDB
- char(24) primary key in Postgres (equivalent to MongoDB _id)
- JSONB to store JSON in Postgres
- 1 million random documents inserted in 100 batches of 10 000 documents
- Program in Java using latest drivers
- Measure storage size, insertion, query times
- Cache cleared and DBs restarted before each query
- *db.collection.status()* and *pg_total_relation_size()* used to measure the storage size
- https://github.com/dtim-upc/MongoDBTests

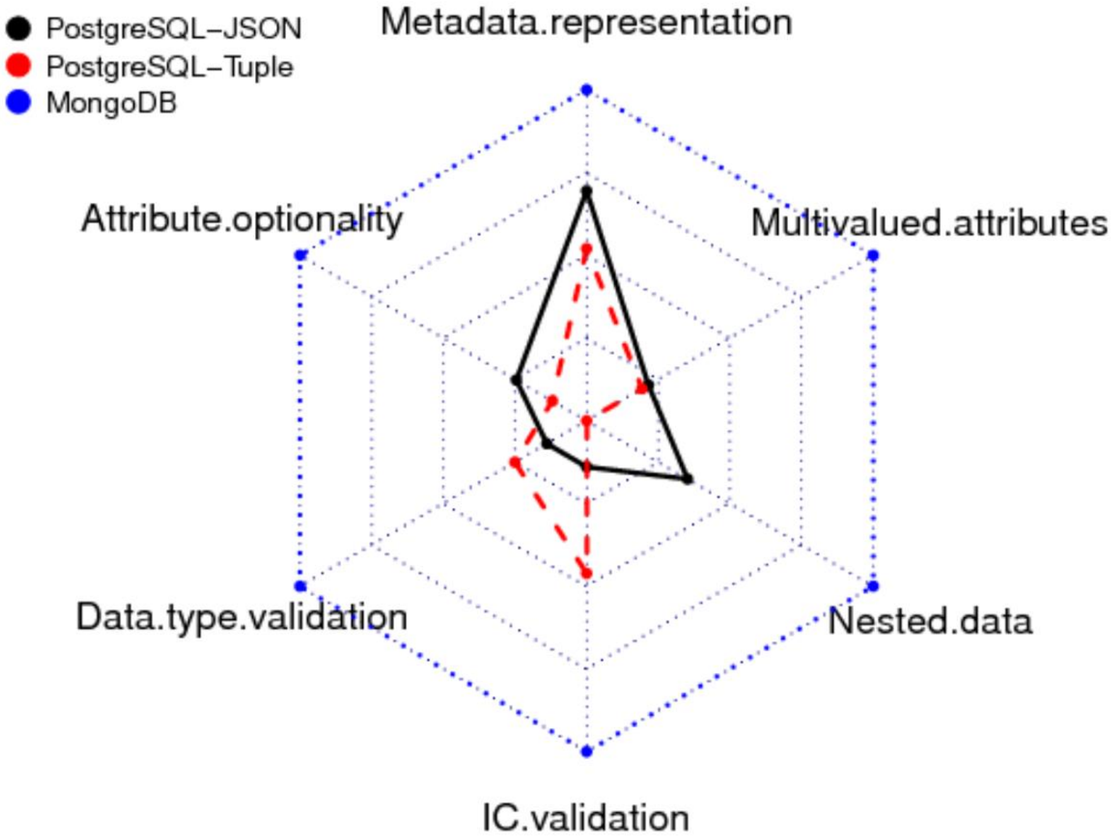
Summary – Storage size

- Storing tuples takes less storage space for integers
- MongoDB BSON has better encoding that reduces the storage space
- If data is text, JSON prevails (metadata experiment)
- Validation does not have any impact on storage space



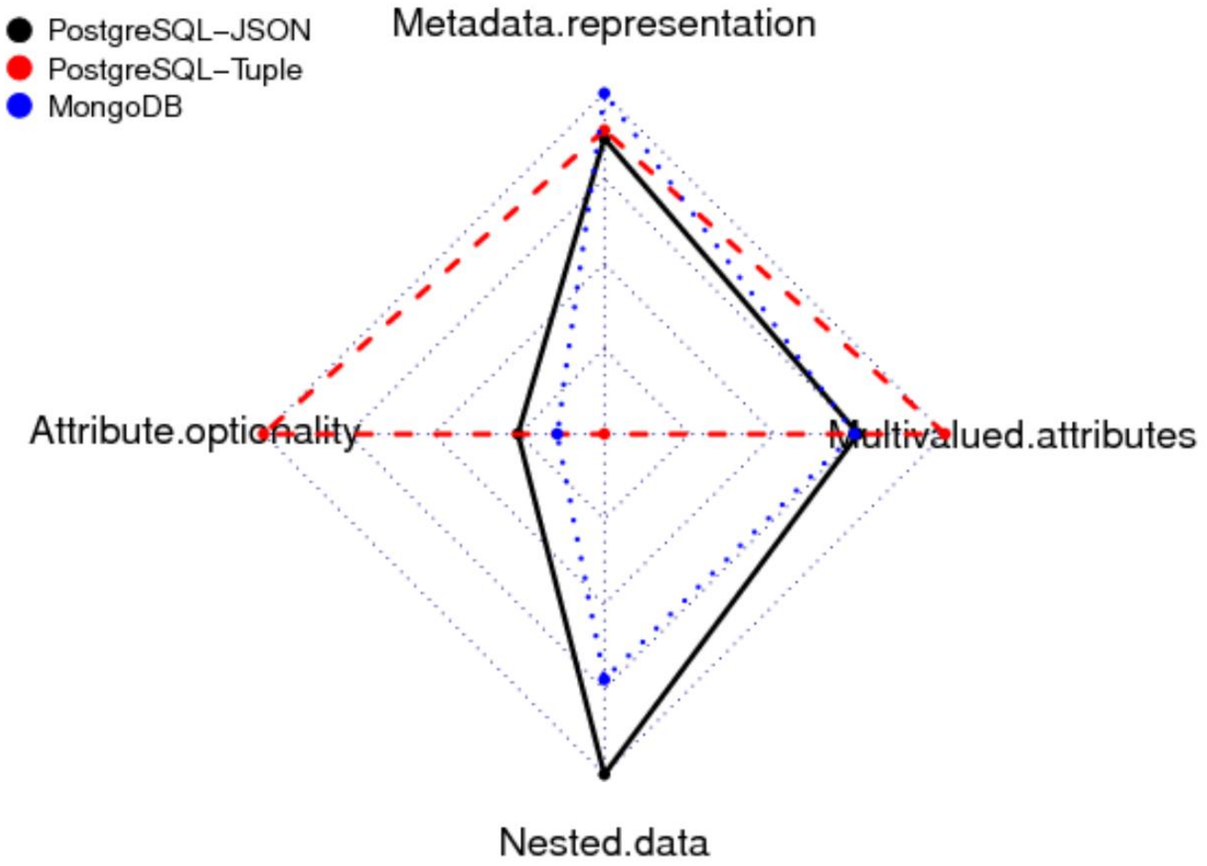
Summary – Insertion time

- Having ACID properties hinders PostgreSQL performance
- MongoDB insertion is always faster (delayed flushing to disk)
- JSON is better for attribute optionality, nesting, multivalued attributes and large text storage
- JSON validation is expensive



Summary – Aggregation

- Relational aggregation performs better
- Within relational, tuples are better for aggregations
- JSON at a disadvantage due to parsing



Conclusion and future work

- The decision of relational vs correlational is not trivial
- Storage size mostly depends on the engine and the encoding
- Relational is slower in inserts but faster in aggregations
- Extend the experiments
  - Caching mechanisms
  - Indexing structures
- Other DBMS features also affect performance
  - Concurrency control
  - Distribution
  - Connection pools
  - DB setup parameters