Experimental Comparison of Relational and NoSQL Systems: the Case of Decision Support

Tomás F. Llano-Ríos
tfllan01@louisville.edu
University of Louisville,
Louisville, KY 40208,
USA

Mohamed Khalefa
khalefam@oldwestbury.edu
SUNY College Old
Westbury,
Old Westbury, NY
11568, USA

Antonio Badia
abadia@louisville.edu
University of Louisville,
Louisville, KY 40208,
USA
Contribution

- **Experimental comparison**
  - of relational and document-oriented NoSQL systems (PostgreSQL, MongoDB and Couchbase)
  - focused on Decision Support (we use TPC-H)
  - limited to a single-node setting
  - limited to hierarchical data (customer, orders and lineitem tables)

- **Analysis**
  - Query language influence on query optimization
  - Data model’s influence on query optimization
  - Possibilities for improvement
  - Is ongoing: We are still doing experiments
TPC-H Schema translated to document stores in 3 different schemas:

**S1: Embedding**

- **Customer**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>n</sub>: val<sub>n</sub>

- **orders:**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>m</sub>: val<sub>m</sub>

- **Lineitems:**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>l</sub>: val<sub>l</sub>

**S2: Linking**

- **Customer**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>n</sub>: val<sub>n</sub>

- **Orders**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>m</sub>: val<sub>m</sub>

- **Lineitems**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>n</sub>: val<sub>n</sub>

**S3: Hybrid**

- **Order**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>n</sub>: val<sub>n</sub>

- **lineitems:**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>m</sub>: val<sub>m</sub>

- **Customer**
  - key<sub>1</sub>: val<sub>1</sub>,
  - key<sub>2</sub>: val<sub>2</sub>, . . . ,
  - key<sub>l</sub>: val<sub>l</sub>
Experiment Setup

Databases

- **Relational**: PostgreSQL v10.6
- **NoSQL**: MongoDB v4.0, Couchbase CE v6.5

Queries

- Taken from the TPC-H benchmark
- Only involving Customer, Orders, and Lineitem
- Total query versions: 38
- Each version was run 5 times, average taken
- Time limit of 24 hours per query
- Couchbase queries were run at scale factor (SF) 1 only due to poor performance
Query 1: Scan over Lineitem only, S1/S3 at a disadvantage. S2 is superior because it does not need joins nor unnest.

Figure: Running times of query 1 on MongoDB and PostgreSQL
Query 3: Join order matters greatly, S2 is superior if orders are filtered first.

Figure: Running times of query 3 on MongoDB and PostgreSQL
Query 4: No lookups in S1 or S3, but S1 has extra unnest. Filter array and unnest is faster than unnest and then filter.

Figure: Running times of query 4 on MongoDB and PostgreSQL
Query 12: Orders $\bowtie$ Lineitem makes S2 the slowest. No lookups in S1 or S3, but S1 has extra unnest.

Figure: Running times of query 12 on MongoDB and PostgreSQL
Results

**Query 13:** Lookup slow with sub-queries (issue [SERVER-41171]); cannot effectively convert $\sigma(C \bowtie \delta \bowtie O)$ into $C \bowtie \delta \sigma(O)|C = \text{Customer} \land O = \text{Orders}$ in S2 nor S3. No lookups in S1.

**Figure:** Running times of query 13 on MongoDB and PostgreSQL
Query 22: Sub-query translates to self-lookup on S1, ordering of operators depends on schema, sub-queries in lookup are slow.

**Figure**: Running times of query 22 on MongoDB and PostgreSQL
Analysis

High Cost
- Scan single collection with large documents
- Unnesting large documents

Main document store limitations
- Join reordering
  - Must be done manually in MongoDB
  - Couchbase CE cannot express correlation as join
- Optimizer
  - No cost-based optimizer
  - Selectivity of conditions not considered
Conclusions

- Typical document store design (one or a few collections with complex documents that use embedding) is not always a good fit for DSS environments.

- Schema-less does not imply schema-free. Schema design matters in document stores for DSS environments.

- Navigational languages should be supported by an optimizer that is able to rewrite and reorder operations in a query.
Future work

- Extending comparison to column-oriented DBs.
- Exploring document storage as multi-dimensional arrays.
- Expanding further schemas and query sets (all TPC-H queries).
- Explore a distributed setup.
Data, Queries and Code

www.github.com/tllano11/dss-sql-vs-nosql-experiments

Questions?

Please contact us:

tfllan01@louisville.edu,
khaledfam@oldwestbury.edu,
abadia@louisville.edu