ligDB-Online Query Processing Without (almost) any Storage

Evica Milchevski and Sebastian Michel
University of Kaiserslautern

contact: milchevski@cs.uni-kl.de
Motivation
Key Observation

- Data is never created in one big bang
- Instead, it is built up over time
- For instance through crowd-sourcing facts about entities (or even missing entries to tuples in traditional RDBMS.

<table>
<thead>
<tr>
<th>Name</th>
<th>Cuisine</th>
<th>IsOpen</th>
<th>Street</th>
<th>City</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe’s Diner</td>
<td>US</td>
<td>false</td>
<td>Barrow Street 4</td>
<td>NY</td>
<td>+13425 25352</td>
</tr>
</tbody>
</table>
Motivation and Idea

Instead **Let data go**

NoDB
Let it go DB --- Principle

Volume and Velocity Dimensions of BigDATA comes to rescue.

Register
Query
Result
Return when sufficiently answered

{“type”: restaurant
“name”: ?
“city”: “Chicago”
}

It triggers gathering of relevant data
Key Characteristics of ligDB

- No Storage
- No Schema
- Query-based
- Result Caching
- Live or Historic Results
- Query by Example
Architecture
Enrich, Annotate, Fuse

Data received as JSON objects

- Disambiguate
- Clean
- Standardize
Collects data needed to answer the queries

Not needed data is deleted
Cache

- Limited in-memory storage
- Handles results of historic queries
- Evicts too old to be useful data
Process

Handles the processing of queries
Query Processing

- Characteristics of ligDB

1. Query execution > milliseconds
   - Grouping queries
   - Sharing the load

2. The workload and the underlying data is dynamic
   - Adaptive, on-the-fly, query optimization
   - Adaptive indexing
Guide the query phrasing process
Query Types

- No schema → hard to phrase
  - Query by example

- Analytical and explorative queries
- Basic queries: Data gathering queries
- More advanced queries are possible
- **Possible issue:** the query selectivity

```json
{
  "business_id": ?
  "name": ?
  "city": "Chicago"
}
```
Query Advisor

- Key component of ligDB
- Base components
  - Statistics gathering component
  - Generic and domain specific ontologies
- Query history can be used as well
Challenges
Time Varying Results

- Monotone continuous queries as queries whose result is non-decreasing at any point of time
  - Append-only databases

- Query results can change at any time
  - Truth over time $\neq$ queries over time windows

- Combine historical data with streaming data?
Incomplete Results

- (Almost) never having the complete answer
  - Example: Query of top rated hotels in NY

- When are query results complete enough to be returned to the user?
  - Completeness/execution time tradeoff
  - Convergence
  - Stability
Scale Independence

- Queries trigger data collection
- Data subscribed to is performance critical
  - Ideally: bounded input data
  - Time restrictions of the queries
- How to predict if the query can be answered?
- Make the database interactive
  - Periodical answers
  - Periodical user “checks”
Cold-Start Problem

- Inspired by recommender systems
- In ligDB: How and which queries to phrase?
  - No schema exists
  - No previous knowledge
  - The user can resort to general data-gathering queries
  - The query advisor should be able to solve this problem
Key Characteristics of ligDB

- No Storage
- No Schema
- Query-based
- Result Caching
- Live or Historic Results
- Query by Example