Combining Provenance Management and Schema Evolution

Problem

- Research data management: trawling and archiving of data collected in scientific projects, experiments, or observations.
- Goal: Traceability, reconstructibility and replicability of the path from data collection to publication.

Calculation of a minimal part of the database (minimal sub-database)

- Different constraints for the sub-database to be determined:
  - Number of tuples of the original relation remains unchanged.
  - The sub-database can be mapped homomorphically to the original database.
- The sub-database is an intentional description of the original database.
- Question: Which additional information is required to be able to reconstruct the minimal part of the database \( J^* \) if the result and the evaluation query \( Q \) are both archived?
- Idea: Calculation of an inverse query \( Q_{prov} \) to determine the minimal sub-database
- Type of inverse depending on the additional information noted.

Unification of Provenance and Evolution

- Goal: Evaluation of provenance queries with changing data and schemas
- Idea: Combination of provenance with schema and data evolution
- Wanted: New minimal sub-database to be archived \( J^* \)
- Calculation of a new query \( Q'(J(S_I)) \) from the old query \( Q(J(S_I)) \)

Provenance Management and Schema Evolution:

- Schemas: \( S_I, S_J \) and \( S \)
- Minimal sub-databases: \( J^* \subseteq I \) and \( J^* \subseteq J \)
- Input for \( Q_{prov} \): \( N^* \subseteq K \)
- Query: \( Q \)
- Schema evolution: \( \varepsilon' \)
- Provenance Query \( Q_{prov} \)

Data Provenance \( Q_{prov} \)

- Information order: where \( \Rightarrow \) why \( \Rightarrow \) how
- Provenance types and answers:
  - \( \text{why} \) = type of rule name
  - \( \text{how} \) = tuple or base name
  - \( \text{why not} \) = provenance polymorphy or polymorphism

CHASE Inverse

- CHASE: Incorporating dependencies + in an object \( Q \), i.e. chase \((Q) = \sigma \)
- CHASE+ BACKCHASE:
  - Exact CHASE inverse: reconstructs the complete original database
  - Classical CHASE inverse: returns a result equivalent to the original database
  - Tuple preserving relaxed CHASE inverse: preserves the number of tuples
  - Result equivalent CHASE inverse: \( \text{chase}_{eq}(I) = \text{chase}_{eq}(I') \)
  - Reduction:

<table>
<thead>
<tr>
<th>Conditions for the existence of CHASE inverse</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHASE inverse</td>
</tr>
<tr>
<td>Exact</td>
</tr>
<tr>
<td>Classical</td>
</tr>
<tr>
<td>Tuple preserving</td>
</tr>
<tr>
<td>Result equivalent</td>
</tr>
</tbody>
</table>

Example:

\[
Q = \{ \text{Name} = \text{Tom}, \text{Subject} = \text{Math}, \text{Grade} = 2.0 \} 
\]

\[
Q_{prov} = \{ \text{Name} = \text{Tom}, \text{Subject} = \text{Math}, \text{Grade} = \text{2.0} \} 
\]

Query \( Q \)

- CHASE algorithm for evaluation of queries
- Approach: Description of the query \( Q \) as extended S-T TGDs and EGDs
  - Calculation of a CHASE inverse \( Q_{prov} \) to reconstruct a minimal sub-database \( J^* \)

Schema Evolution \( \varepsilon' \)

- CHASE algorithm for schema evolution
- Approach: Description of the schema evolution \( \varepsilon' \) as S-T TGDs and EGDs
  - Calculation of an inverse \( \varepsilon'^{-1} \) to reconstruct the old minimal sub-database \( I^* \)

Tanja Auge, Andreas Heuer

INSTITUTE OF COMPUTER SCIENCE | DATABASE RESEARCH GROUP | UNIVERSITY OF ROSTOCK, Albert-Einstein-Straße 22 | 18059 Rostock, Germany