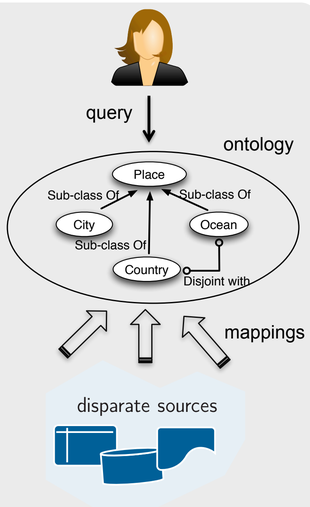
KeywDB: Keyword Driven Mapping Construction

Optique
D. Zheleznyakov, E. Kharlamov, I. Horrocks
V. Klungre, M. G. Skjæveland, D. Hovland, M. Giese, A. Waaler
University of Oxford
University of Oslo

Semantic Access to Databases

Databases
- Optimised for query answering
- Historically evolve in user-unfriendly form
- Statoil
  - Exploration & Production Data Store (EPDS)
  - Has been developed for 15 years
  - 3K tables, 37K columns, 700 GB data

Ontology Based Data Access
- Ontology: conceptual domain model
- Mappings: relate ontological terms to DBs



Connecting Data to Ontologies

Problems
- Connect new DBs to the ontology
- Add new vocabulary to the Ontology

Existing approaches
- Direct mappings: mirror the structure
- May not work in many applications

Project Goals:
- Facilitate discovery of mappings that reflect users' expectations
- Enable discovering of quality mappings in industry: Statoil

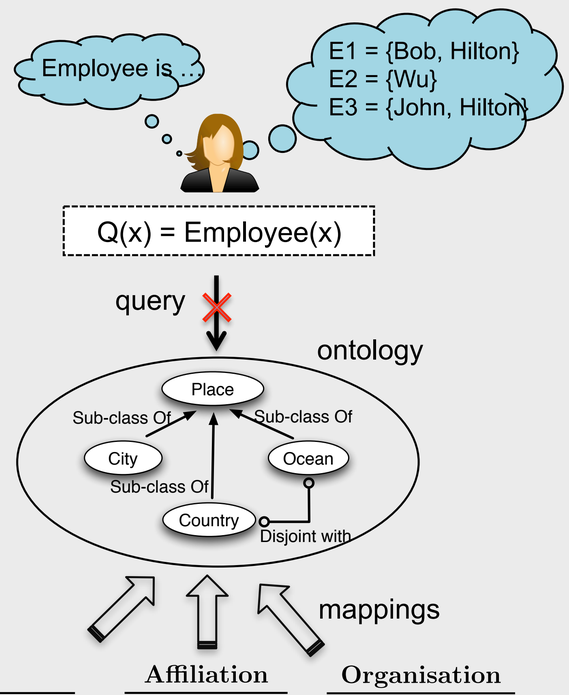
Keyword Driven Approach: General Idea

User
- Describes: what she expects from the ontology
- Provides: examples of entities of the missing class C
- Example = set of keywords
- Keyword = a characteristics, or attribute value for entities in C

System
- Returns a ranked list of queries
  - r_1:SQL_1, r_2:SQL_2, …, r_n:SQL_n
- Each query represents C
- In materialisation of each query SQL_i
  - each tuple corresponds to an entity of C
  - some user's ex. are "among" the tuples
- The higher the rank, the better the query captures user's expectations

```
mappingid   Class - Employee
target      ex:pid a ex:Employee
source      SELECT pid
            FROM   Person, Organisation, Affiliation
            WHERE  Person.pid = Affiliation.person,
                   Organisation.pid = Affiliation.org
```

```
mappingid   Property - hasName
target      ex:pid ex:hasName 'fname'
source      SELECT Person.pid, Person.fname
            FROM   Person, Organisation, Affiliation
            WHERE  Person.pid = Affiliation.person,
                   Affiliation.pid = Affiliation.org
```

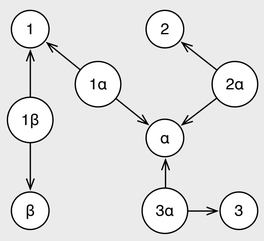


Person		
pid	*fname*	*lname*
1	Bob	Hilton
2	Bob	Lee
3	John	Wu

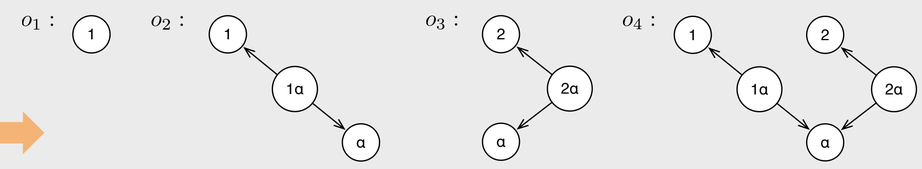
Affiliation	
person	*org*
1	β
2	α
3	α
1	α

Organisation	
oid	*cname*
α	Hilton
β	UOXF

Data Graph:



Sub-graphs for keywords:



E1: Bob → 1, Hilton → 1

E1: Bob → 1, Hilton → a

E1: Bob → 2, Hilton → a

E1: Hilton → 1, Bob → 2

Turn RDB data into a Graph
- Each tuple → node
- 2 semantically related tuples → edge

Map each example entity E into the graph
- Map each keyword of E map to a node
- Take minimum sub-graphs "covering" E

Compute queries from sub-graphs
- Convert each sub-graph into a query
- Unify obtained queries

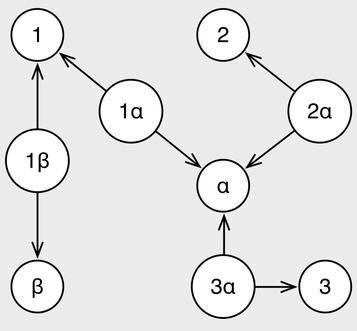
Rank queries based on
- Quality of keyword match and distribution
- Size and compactness of sub-graphs

0.8: Q1 = Person(x,y,z), Affiliation(x,u), Organisation(u,w)
0.2: Q2 = Person(x,y,z)

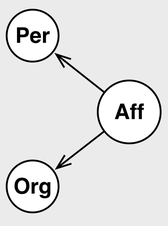
Research Challenges

Graphs
- Data graph: too large – good to define semantics
- Schema graph: does not help much (no keyword info)
- Keyword driven schema graph: good balance, practical

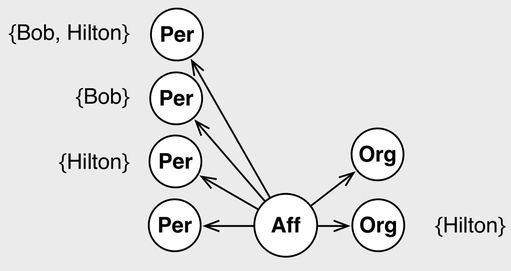
Data Graph



Schema Graph

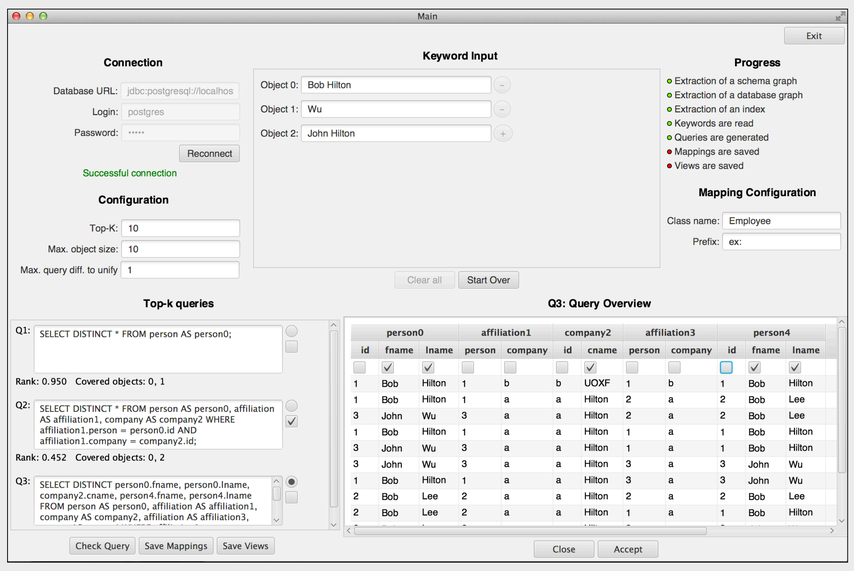


Keyword Driven Schema Graph



Challenges
- Efficiency
  - candidate sub-graph selection
  - indexes for keyword match, node reachability
- Effectiveness
  - target queries are in top-k
  - small number of "simple" keywords is enough
- Top-K queries
  - top-k without exact ranking
  - approximation of ranking

KeywDB System



Main features
- Allows for multiple examples, each with several keywords
- Computation of
  - Schema graph, keywords driven schema graph
- Inverted index for keywords
- Reachability index for keywords driven schema graph
- Support for mapping configuration via attribute selection

Flexible configuration
- Top-k, maximal query size, query similarity