Automatic Extraction of Ontologies Wrapping Relational Data Sources

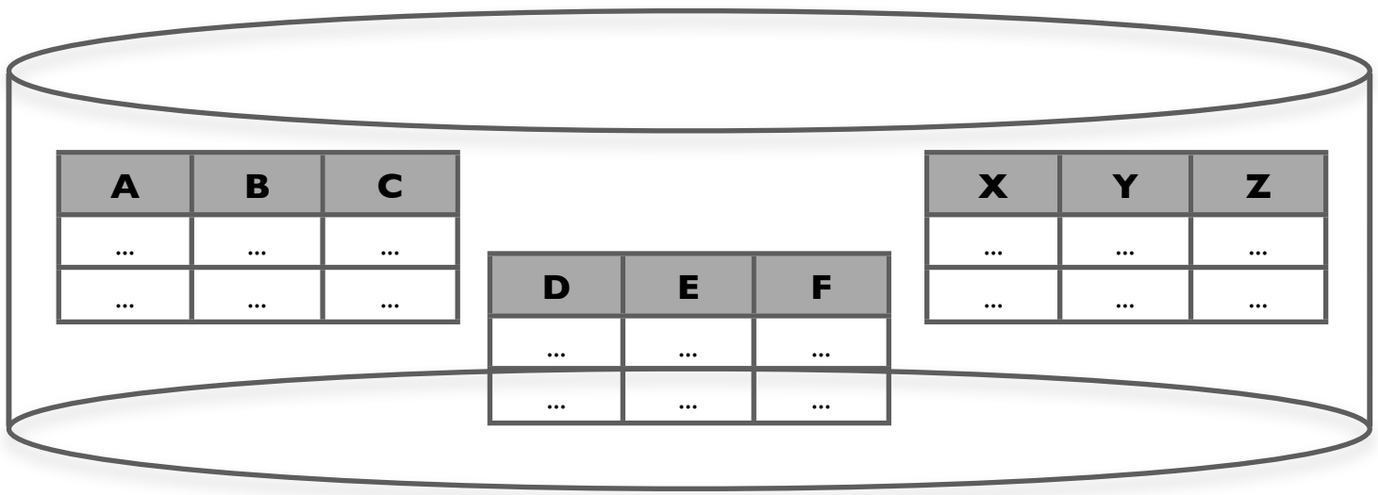
Lina Lubyte, Sergio Tessaris

KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano

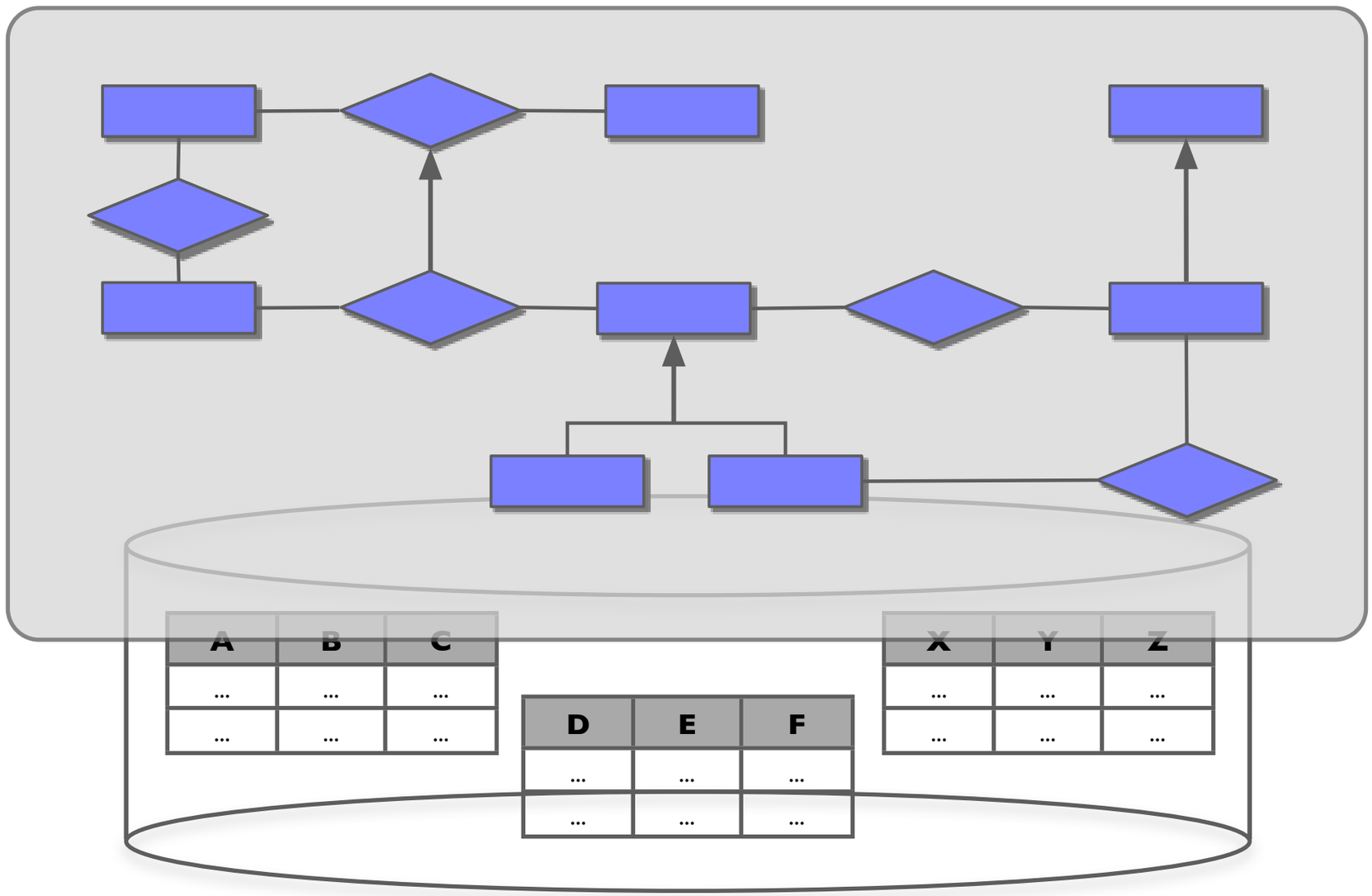
September 1, 2009

Semantic data access

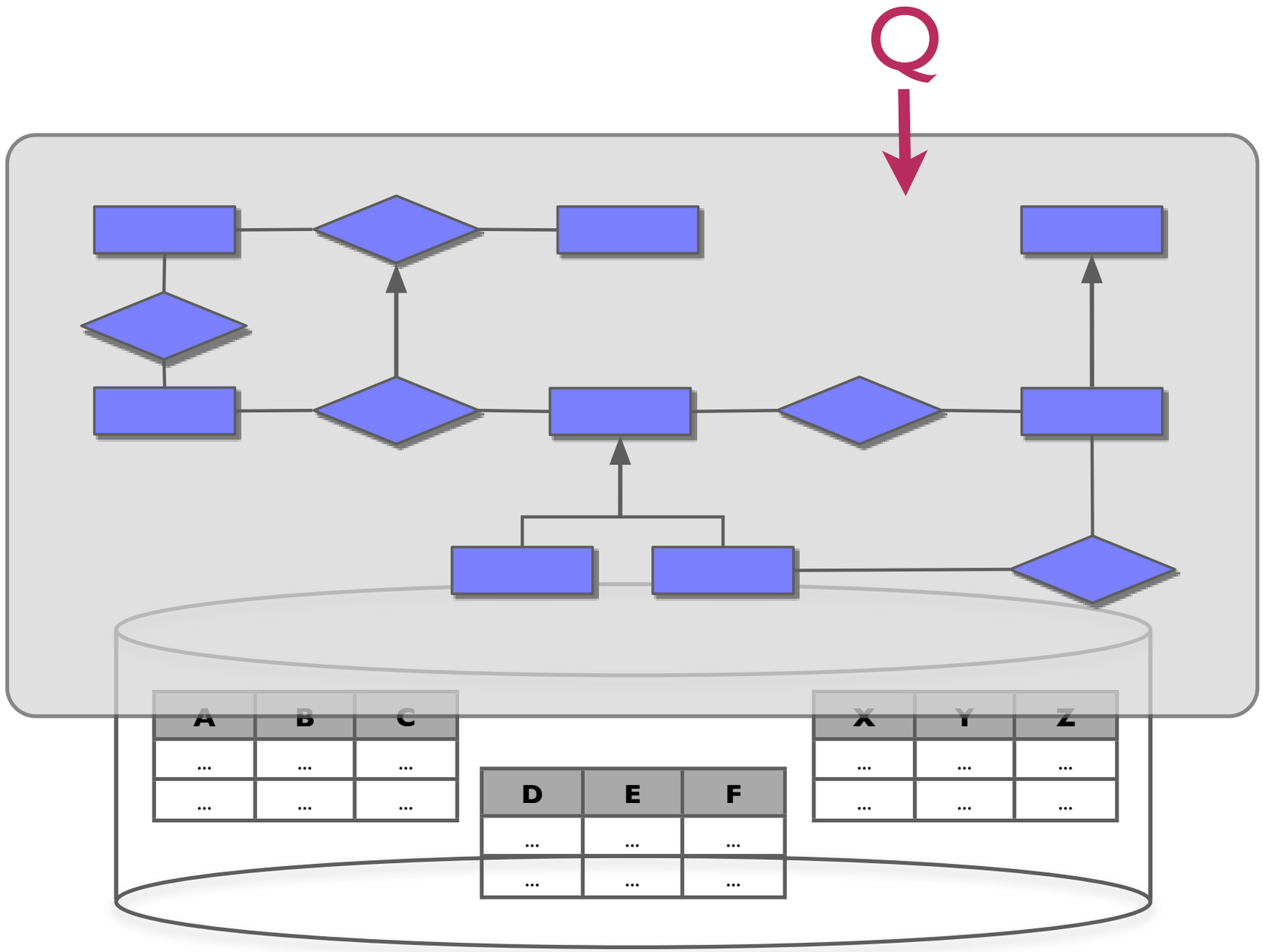
Semantic data access



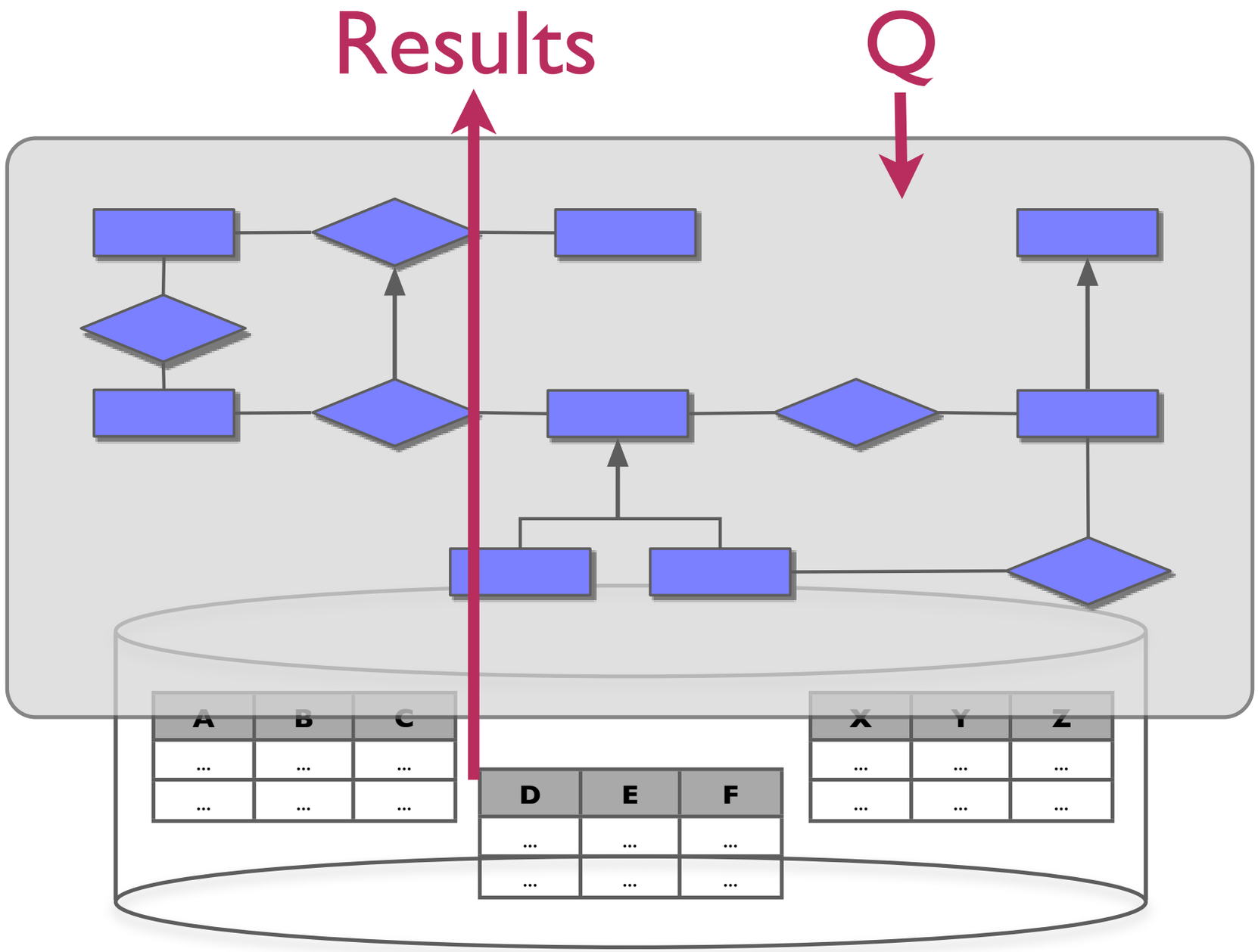
Semantic data access



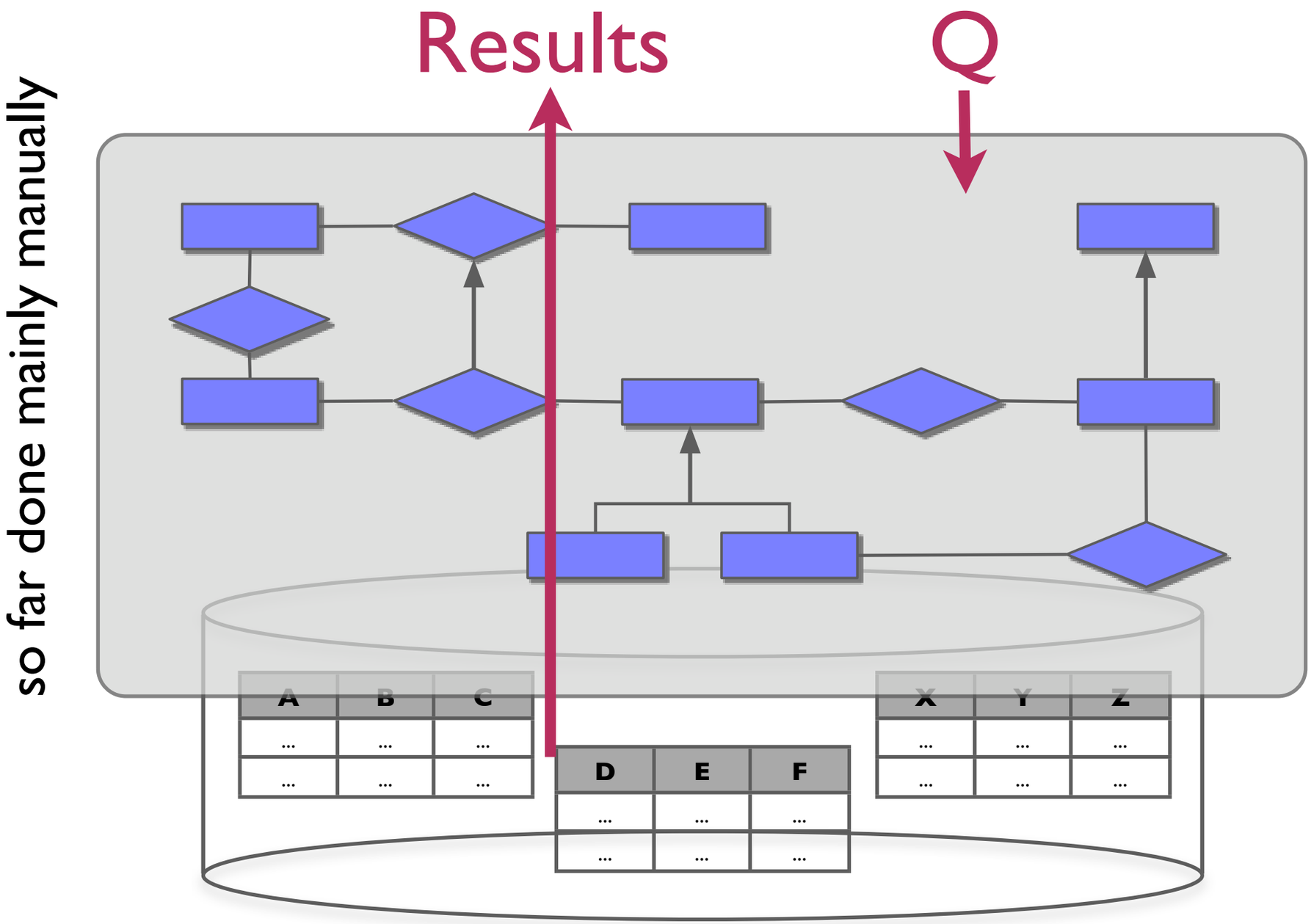
Semantic data access



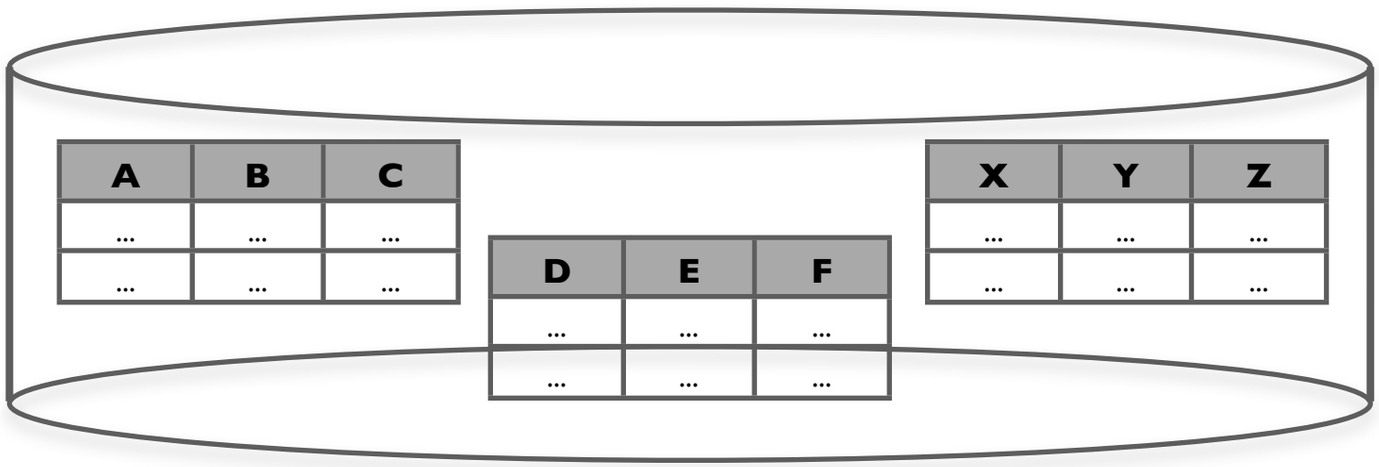
Semantic data access



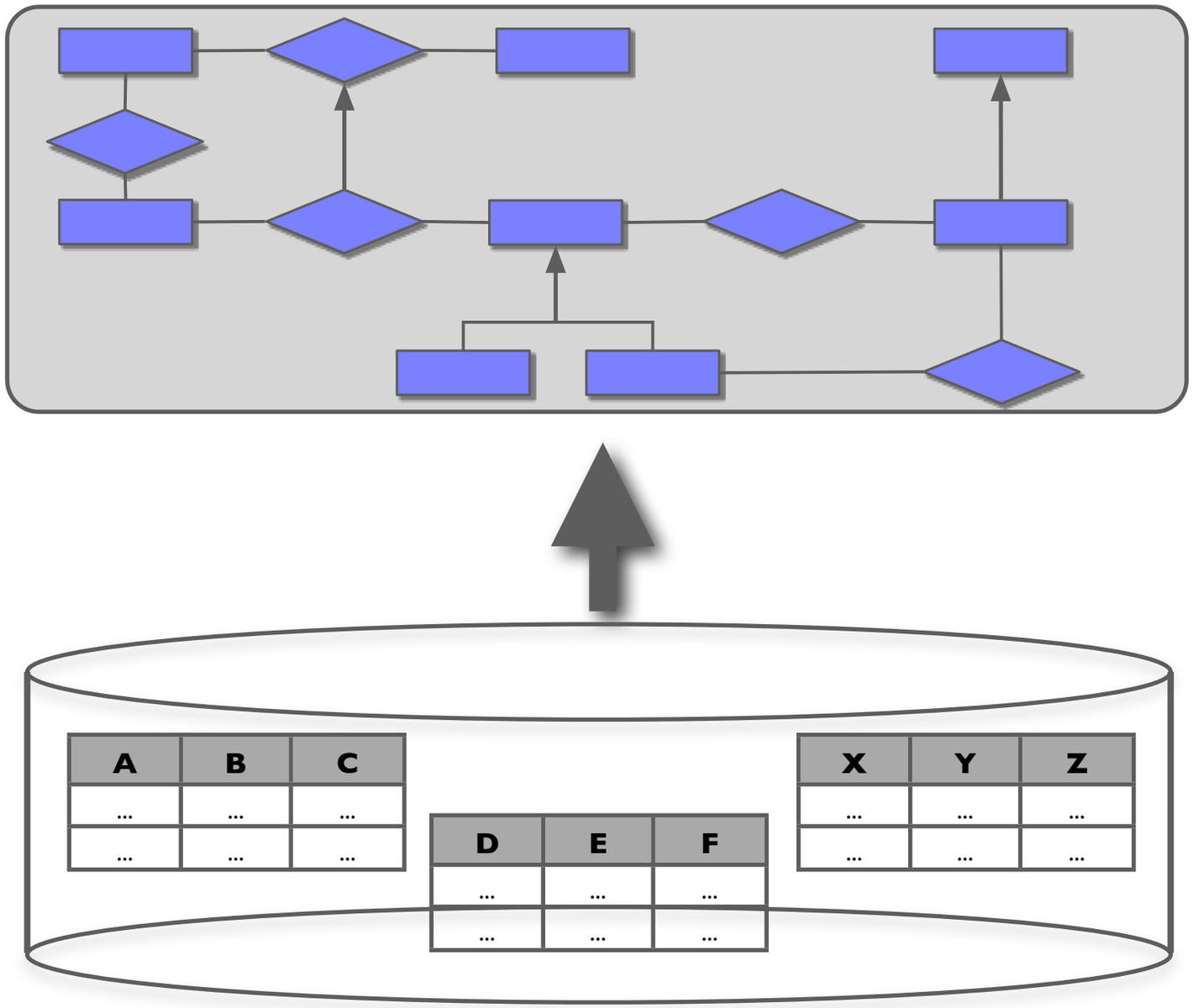
Semantic data access



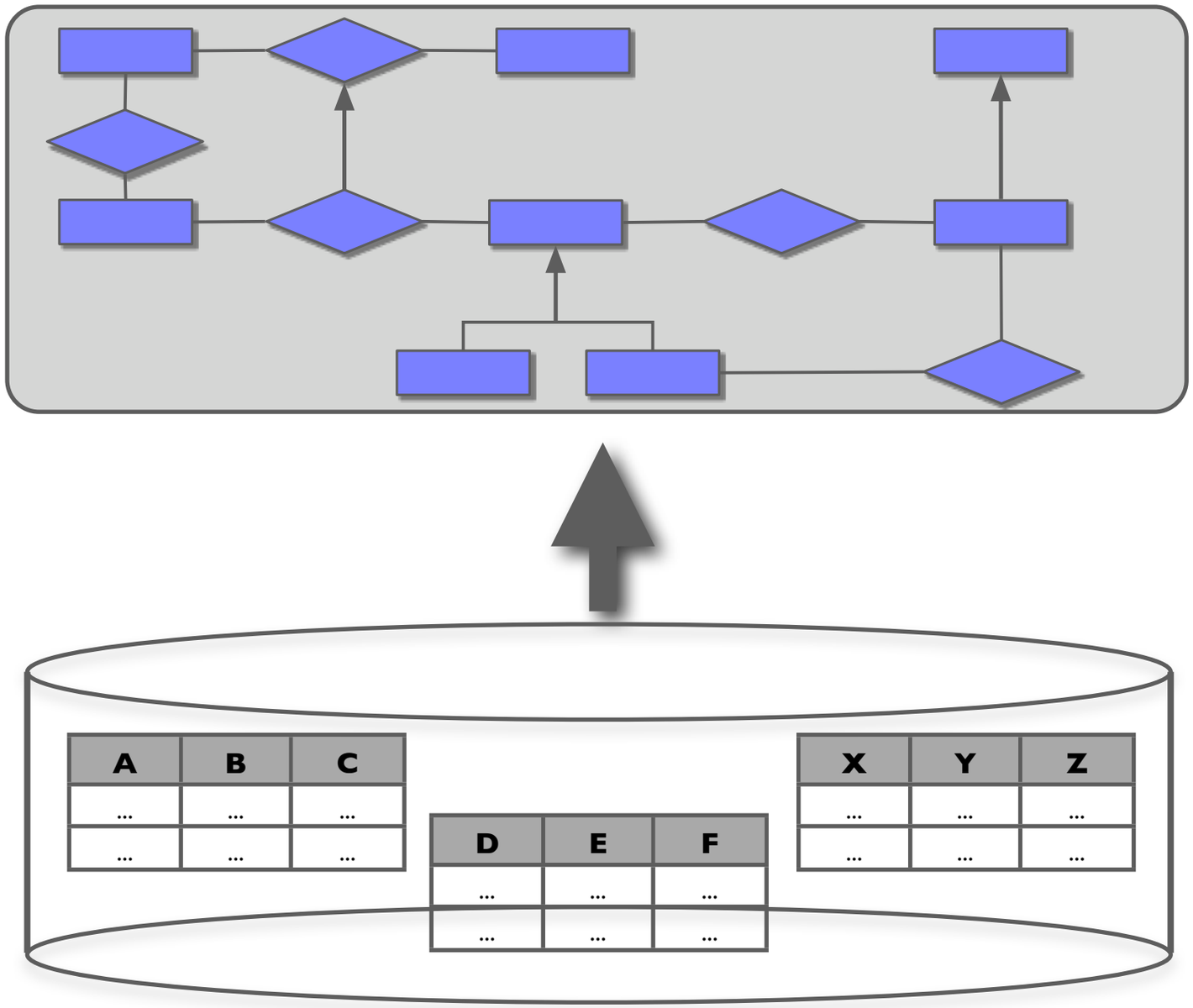
Extracting a wrapping ontology



Extracting a wrapping ontology

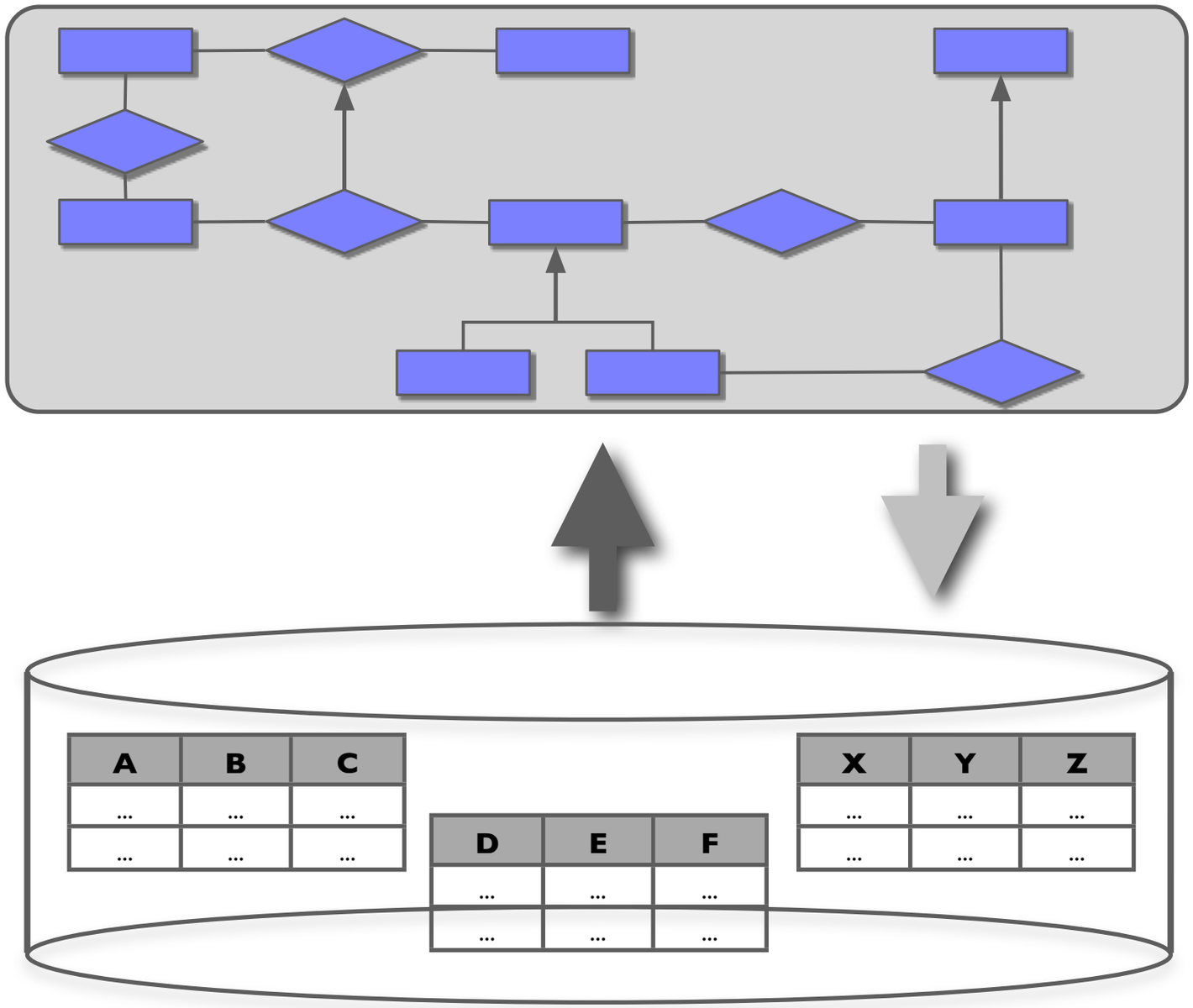


Extracting a wrapping ontology



preserve the semantics of the constraints in the DB

Extracting a wrapping ontology



preserve the semantics of the constraints in the DB

maintain the link with the DB

Outline

- Formal Framework
- Ontology Extraction
- Implementation and Case Study
- Related Work
- Current Work

Relational source

- A relational source DB = {Ψ, Σ}, with Ψ a relational schema and Σ set of integrity constraints:
  - nulls-not-allowed: $nonnull(r, A)$
  - unique: $unique(r, A)$
  - key: $key(r, A)$, if $nonnull(r, A)$ and $unique(r, A)$
  - inclusion: $r_1[s_1] \subseteq r_2[s_2]$
  - foreign key: if $r_1[s_1] \subseteq r_2[s_2]$ and $key(r_2, s_2)$
  - exclusion: $(r_1[s_1] \cap r_2[s_2]) = \emptyset$
  - covering: $(r_1[s_1] \cup \ldots \cup r_m[s_m]) \subseteq r_0[s_0]$

Ontology language

- DLR-DB system $S = \{R, K\}$, with R a relational schema and K set of axioms involving names from R:

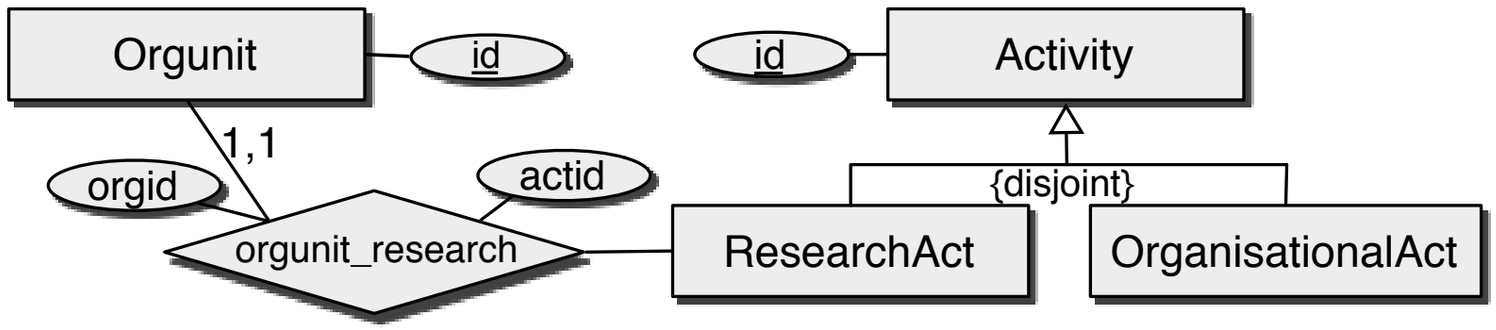
$R[s] \sqsubseteq R'[s']$	$\pi_s R^{\mathcal{D}} \subseteq \pi_{s'} R'^{\mathcal{D}}$	Inclusion
$R[s]$ disj $R'[s']$	$\pi_s R^{\mathcal{D}} \cap \pi_{s'} R'^{\mathcal{D}} = \emptyset$	Disjointness
$\text{key}(R[s_1, \ldots, s_k])$	for all $\phi_1, \phi_2 \in R^{\mathcal{D}}$ with $\phi_1 \neq \phi_2$, we have $\phi_1(s_i) \neq \phi_2(s_i)$ for some s_i, $1 \leq i \leq k$	Key
$R_1[s_1], \ldots, R_k[s_k]$ cover $R[s]$	$\pi_s R^{\mathcal{D}} \subseteq \bigcup_{i=1\ldots k} \pi_{s_i} R_i'^{\mathcal{D}}$	Covering

- able to express common constraints of ER diagrams
- can be reduced to DL DLR-Lite

  ➡ use of efficient (LogSpace) query answering technique

  ➡ compatible with OWL

Ontology language: example



orgunit_research[orgid] ⊑ Orgunit[id]

orgunit_research[actid] ⊑ ResearchAct[id]

key(Orgunit[id])
key(Activity[id])
key(ResearchAct[id])
key(OrganisationalAct[id])
key(orgunit_research[orgid])

Orgunit[id] ⊑ orgunit_research[orgid]

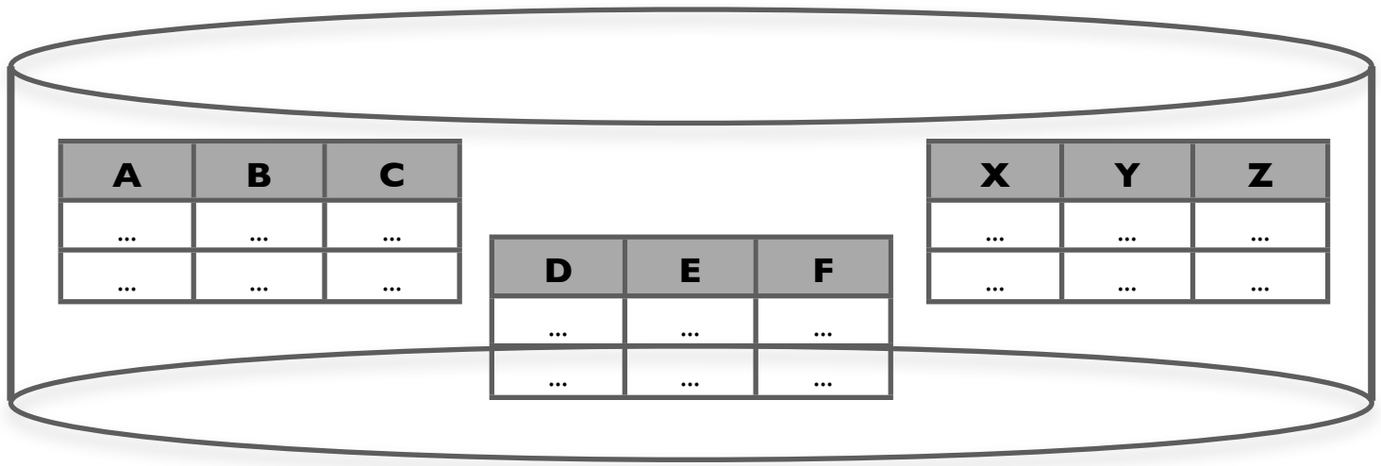
ResearchAct[id] ⊑ Activity[id]

OrganisationalAct[id] ⊑ Activity[id]

ResearchAct[id] disj OrganisationalAct[id]

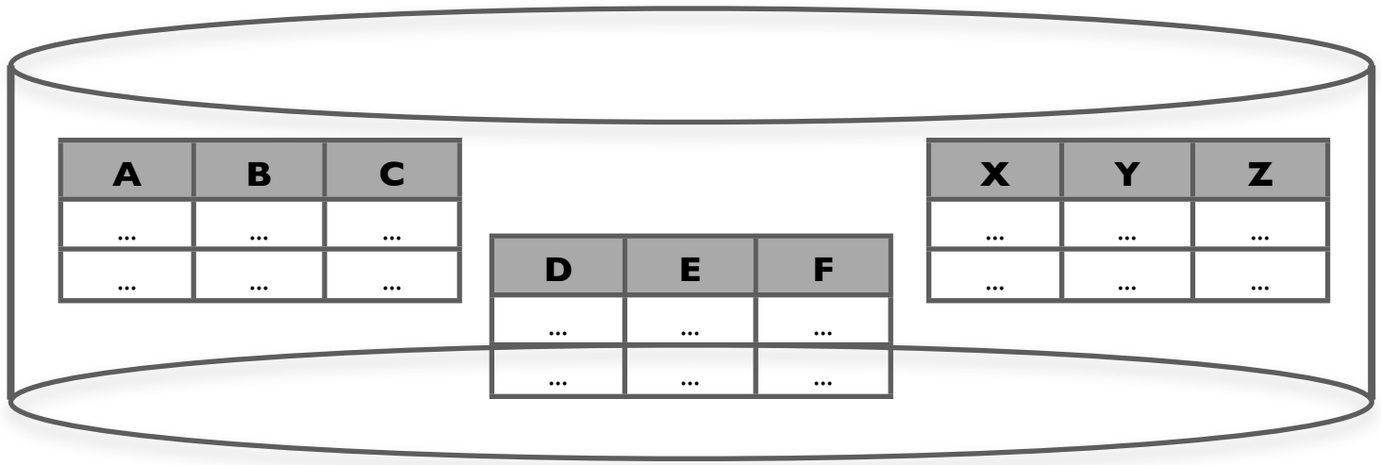
Ontology extraction: intuition

Basic idea: reverse the standard database modeling process



Ontology extraction: intuition

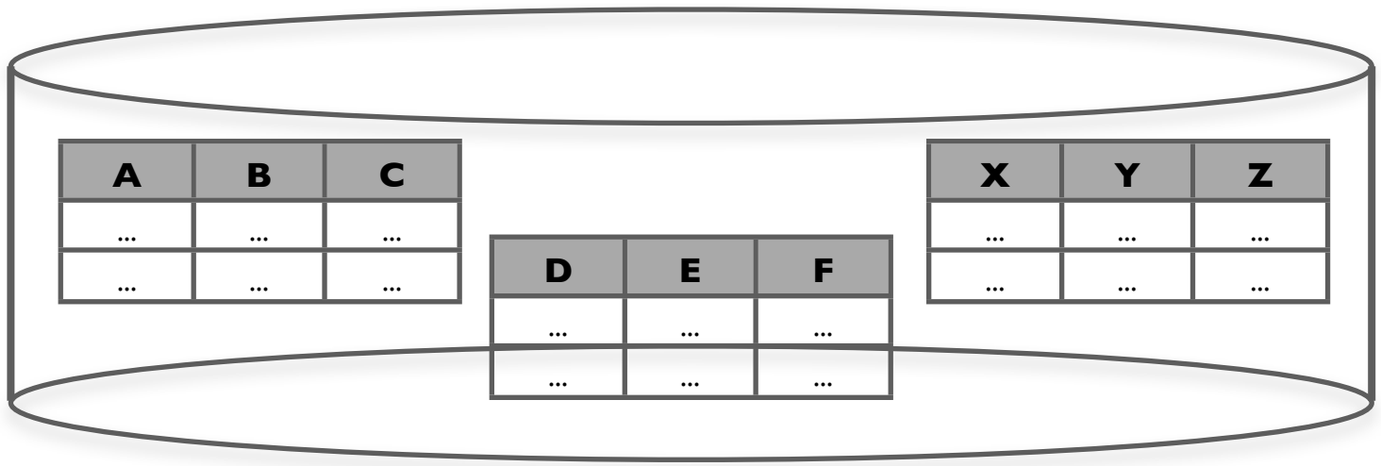
Basic idea: reverse the standard database modeling process



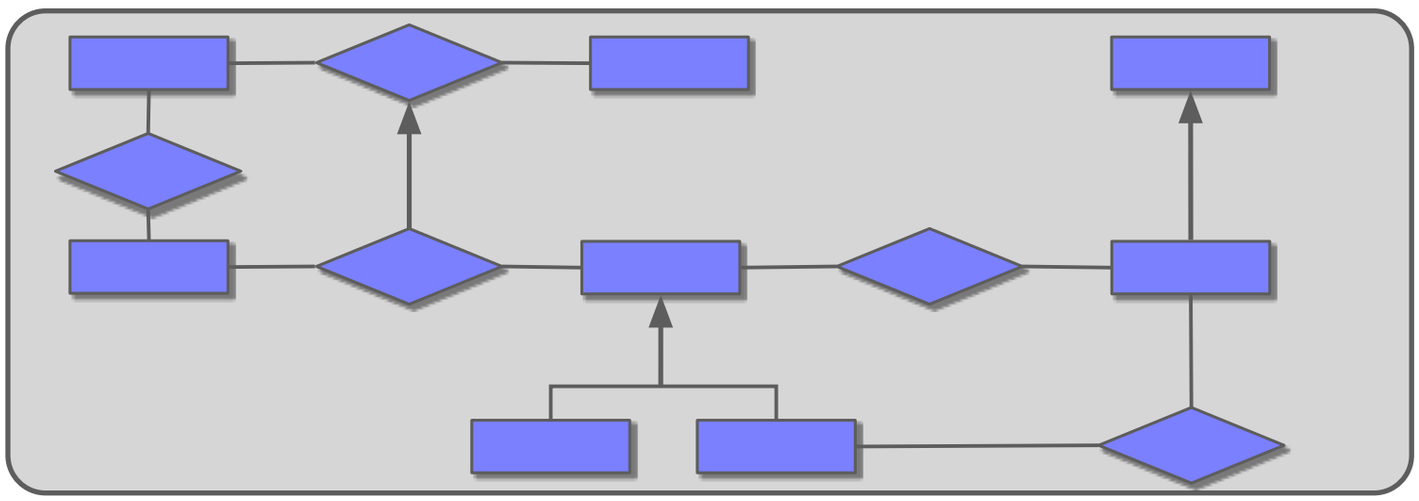
Step 1: classify relations by analyzing the integrity constraints in Σ

Ontology extraction: intuition

Basic idea: reverse the standard database modeling process



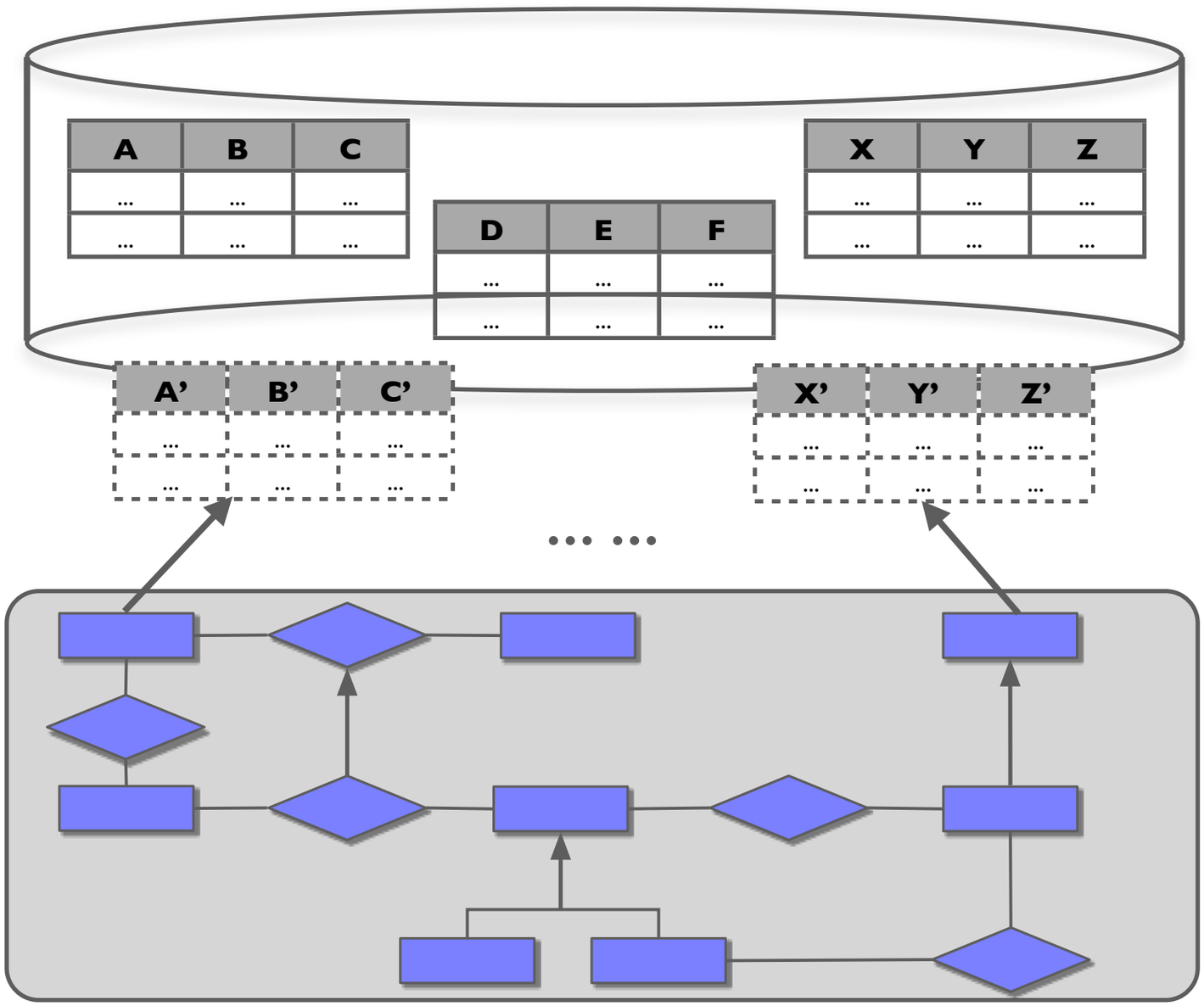
Step 1: classify relations by analyzing the integrity constraints in Σ



Step 2: build the ontology by means of DLR-DB axioms *K* and generate view in *R* for each ontology element

Ontology extraction: intuition

Basic idea: reverse the standard database modeling process



Step 1: classify relations by analyzing the integrity constraints in Σ

Step 2: build the ontology by means of DLR-DB axioms *K* and generate view in *R* for each ontology element

Classification of relations

Classification of relations

- Base, if *K* and ***FK*** don't share attributes

Classification of relations

- Base, if *K* and ***FK*** don't share attributes
- Relationship, if *K* is entirely composed of ***FK*** and |***FK***| > 1

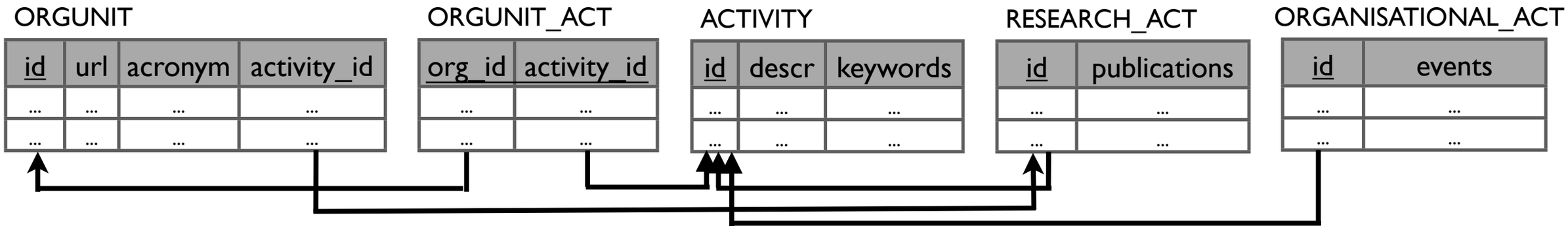
Classification of relations

- Base, if *K* and ***FK*** don't share attributes
- Relationship, if *K* is entirely composed of ***FK*** and |***FK***| > 1
- Specific, if *K* is among ***FK*** and either |***FK***| = 1, or *K* is referenced by some other relation

Classification of relations

- Base, if K and $\boldsymbol{FK}$ don't share attributes
- Relationship, if K is entirely composed of $\boldsymbol{FK}$ and $|\boldsymbol{FK}| > 1$
- Specific, if K is among $\boldsymbol{FK}$ and either $|\boldsymbol{FK}| = 1$, or K is referenced by some other relation
- Ambiguous, otherwise

Ontology extraction by example



nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] ∩ ORGANISATIONAL_ACT[id] = ∅

Ontology extraction by example

ORGUNIT

id	url	acronym	activity_id
...	...	...	...
...	...	...	...

ORGUNIT_ACT

org_id	activity_id
...	...
...	...

ACTIVITY

id	descr	keywords
...	...	...
...	...	...

RESEARCH_ACT

id	publications
...	...
...	...

ORGANISATIONAL_ACT

id	events
...	...
...	...

nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] $\cap$ ORGANISATIONAL_ACT[id] = $\varnothing$

Ontology extraction by example

ORGUNIT

id	url	acronym	activity_id
...	...	...	...
...	...	...	...

ORGUNIT_ACT

org_id	activity_id
...	...
...	...

ACTIVITY

id	descr	keywords
...	...	...
...	...	...

RESEARCH_ACT

id	publications
...	...
...	...

ORGANISATIONAL_ACT

id	events
...	...
...	...

nonnull(ORGUNIT, activity_id)
unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] ∩ ORGANISATIONAL_ACT[id] = ∅

orgunit

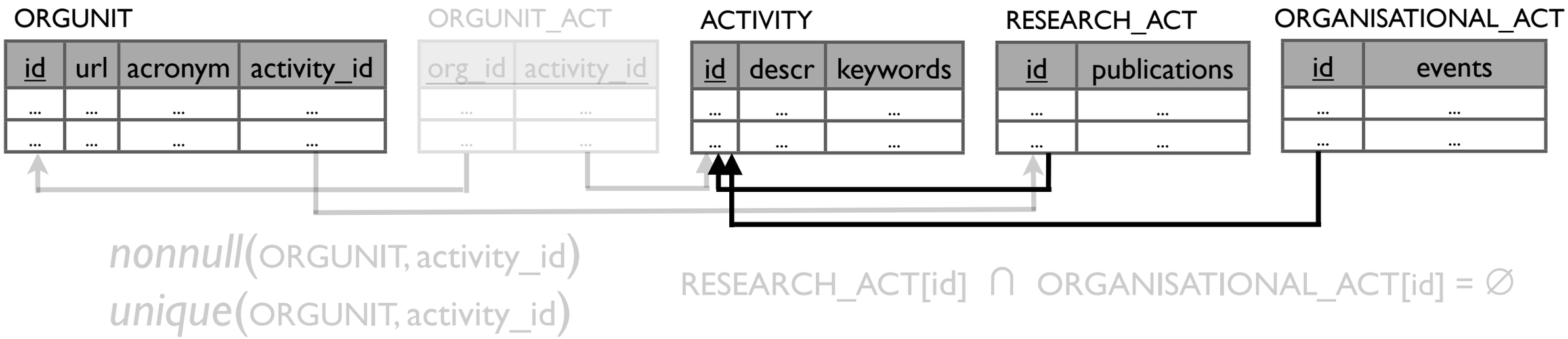
id	url	acronym
...	...	...
...	...	...

activity

id	descr	keywords
...	...	...
...	...	...

key(Orgunit[id])
key(Activity[id])

Ontology extraction by example



key(Orgunit[id])
key(Activity[id])

Ontology extraction by example

ORGUNIT

id	url	acronym	activity_id
...	...	...	...
...	...	...	...

ORGUNIT_ACT

org_id	activity_id
...	...
...	...

ACTIVITY

id	descr	keywords
...	...	...
...	...	...

RESEARCH_ACT

id	publications
...	...
...	...

ORGANISATIONAL_ACT

id	events
...	...
...	...

nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] ∩ ORGANISATIONAL_ACT[id] = ∅

orgunit

id	url	acronym
...	...	...
...	...	...

activity

id	descr	keywords
...	...	...
...	...	...

research_act

id	publications
...	...
...	...

organisational_act

id	events
...	...
...	...

key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] ⊑ Activity[id]

OrganisationalAct[id] ⊑ Activity[id]

Ontology extraction by example

ORGUNIT

id	url	acronym	activity_id
...	...	...	...
...	...	...	...

ORGUNIT_ACT

org_id	activity_id
...	...
...	...

ACTIVITY

id	descr	keywords
...	...	...
...	...	...

RESEARCH_ACT

id	publications
...	...
...	...

ORGANISATIONAL_ACT

id	events
...	...
...	...

nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] $\cap$ ORGANISATIONAL_ACT[id] = $\varnothing$

orgunit

id	url	acronym
...	...	...
...	...	...

activity

id	descr	keywords
...	...	...
...	...	...

research_act

id	publications
...	...
...	...

organisational_act

id	events
...	...
...	...

key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] $\sqsubseteq$ Activity[id]

OrganisationalAct[id] $\sqsubseteq$ Activity[id]

Ontology extraction by example

ORGUNIT

id	url	acronym	activity_id
...	...	...	...
...	...	...	...

ORGUNIT_ACT

org_id	activity_id
...	...
...	...

ACTIVITY

id	descr	keywords
...	...	...
...	...	...

RESEARCH_ACT

id	publications
...	...
...	...

ORGANISATIONAL_ACT

id	events
...	...
...	...

nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] $\cap$ ORGANISATIONAL_ACT[id] = $\varnothing$

orgunit

id	url	acronym
...	...	...
...	...	...

activity

id	descr	keywords
...	...	...
...	...	...

research_act

id	publications
...	...
...	...

organisational_act

id	events
...	...
...	...

key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] $\sqsubseteq$ Activity[id]

OrganisationalAct[id] $\sqsubseteq$ Activity[id]

ResearchAct[id] disj OrganisationalAct[id]

Ontology extraction by example

ORGUNIT

id	url	acronym	activity_id
...	...	...	...
...	...	...	...

ORGUNIT_ACT

org_id	activity_id
...	...
...	...

ACTIVITY

id	descr	keywords
...	...	...
...	...	...

RESEARCH_ACT

id	publications
...	...
...	...

ORGANISATIONAL_ACT

id	events
...	...
...	...

nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] $\cap$ ORGANISATIONAL_ACT[id] = $\varnothing$

orgunit

id	url	acronym
...	...	...
...	...	...

activity

id	descr	keywords
...	...	...
...	...	...

research_act

id	publications
...	...
...	...

organisational_act

id	events
...	...
...	...

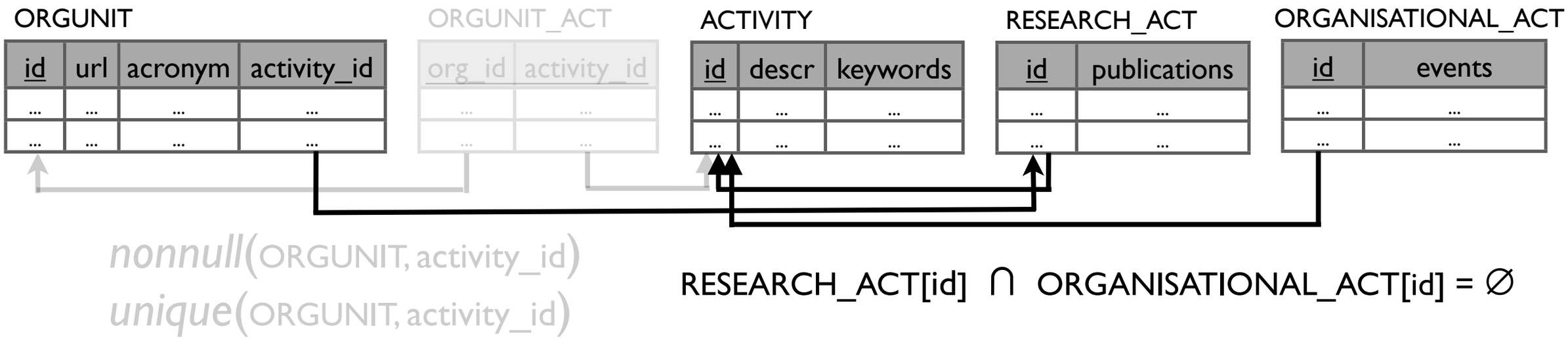
key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] $\sqsubseteq$ Activity[id]

OrganisationalAct[id] $\sqsubseteq$ Activity[id]

ResearchAct[id] disj OrganisationalAct[id]

Ontology extraction by example



key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] ⊑ Activity[id]

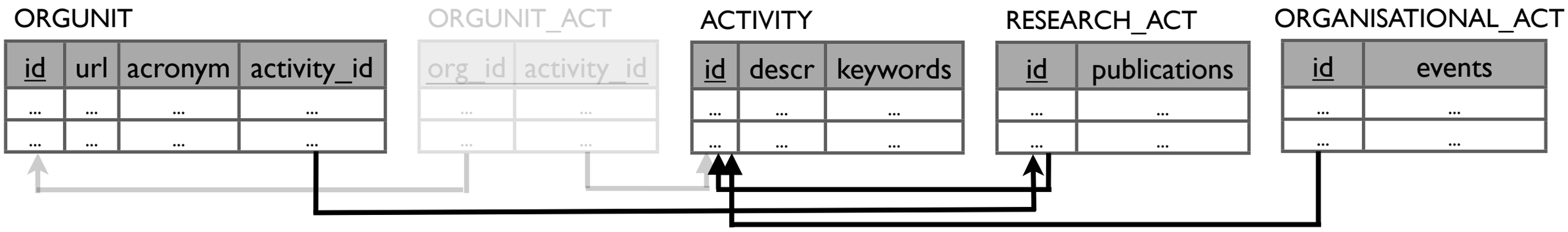
OrganisationalAct[id] ⊑ Activity[id]

ResearchAct[id] disj OrganisationalAct[id]

orgunit_research[org_id] ⊑ Orgunit[id]

orgunit_research[res_act_id] ⊑ ResearchAct[id]

Ontology extraction by example



nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] ∩ ORGANISATIONAL_ACT[id] = ∅

key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] ⊑ Activity[id]

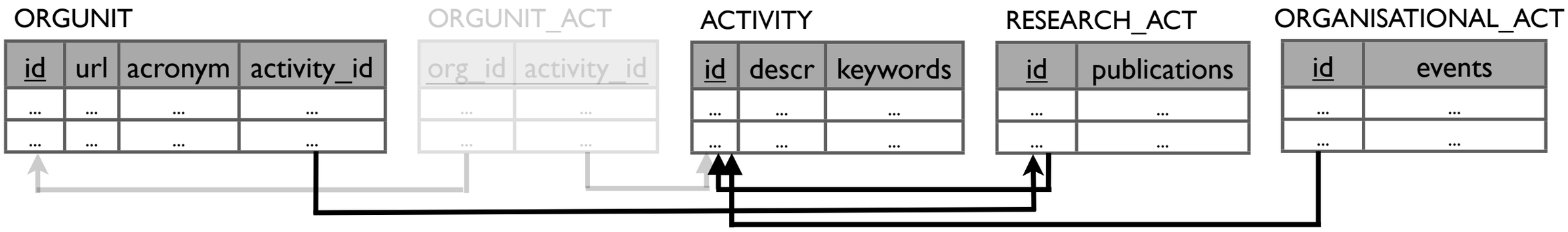
OrganisationalAct[id] ⊑ Activity[id]

ResearchAct[id] disj OrganisationalAct[id]

orgunit_research[org_id] ⊑ Orgunit[id]

orgunit_research[res_act_id] ⊑ ResearchAct[id]

Ontology extraction by example



nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] ∩ ORGANISATIONAL_ACT[id] = ∅

key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] ⊑ Activity[id]

OrganisationalAct[id] ⊑ Activity[id]

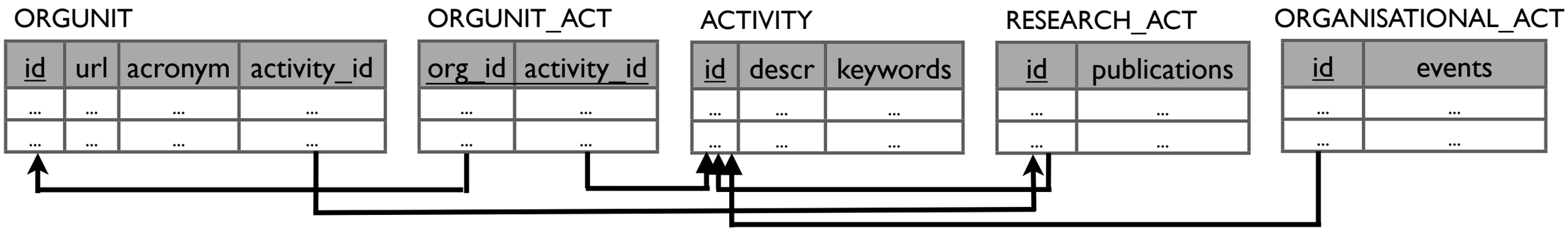
ResearchAct[id] disj OrganisationalAct[id]

orgunit_research[org_id] ⊑ Orgunit[id]

orgunit_research[res_act_id] ⊑ ResearchAct[id]

Orgunit[id] ⊑ orgunit_research[org_id]
key(orgunit_research[org_id])

Ontology extraction by example



nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] ∩ ORGANISATIONAL_ACT[id] = ∅

key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] ⊑ Activity[id]

OrganisationalAct[id] ⊑ Activity[id]

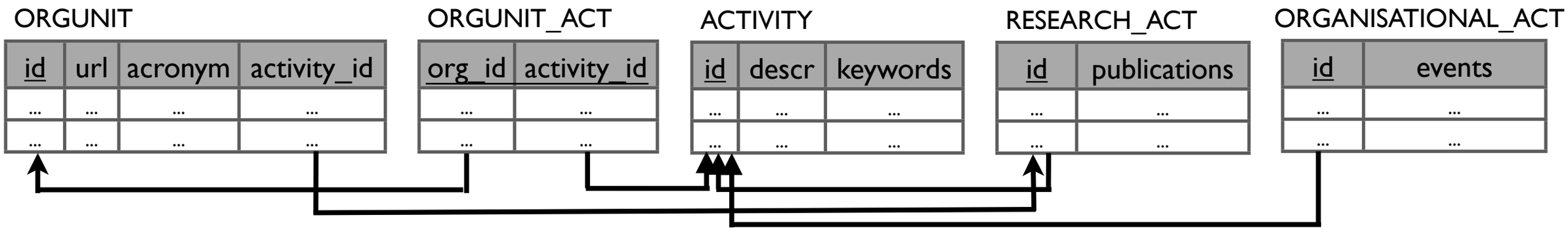
ResearchAct[id] disj OrganisationalAct[id]

orgunit_research[org_id] ⊑ Orgunit[id]

orgunit_research[res_act_id] ⊑ ResearchAct[id]

Orgunit[id] ⊑ orgunit_research[org_id]
key(orgunit_research[org_id])

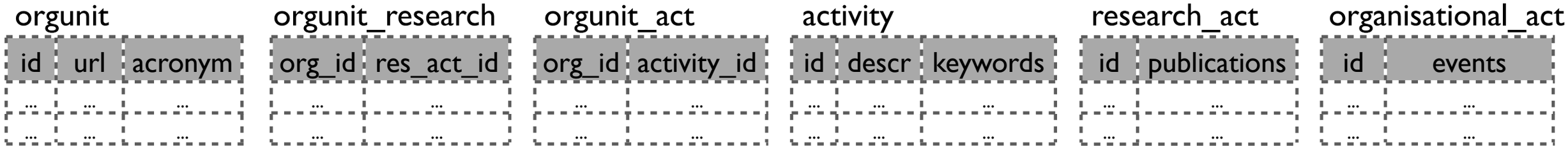
Ontology extraction by example



nonnull(ORGUNIT, activity_id)

unique(ORGUNIT, activity_id)

RESEARCH_ACT[id] ∩ ORGANISATIONAL_ACT[id] = ∅



key(Orgunit[id]) key(ResearchAct[id])
key(Activity[id]) key(OrganisationalAct[id])

ResearchAct[id] ⊑ Activity[id]

OrganisationalAct[id] ⊑ Activity[id]

ResearchAct[id] disj OrganisationalAct[id]

orgunit_research[org_id] ⊑ Orgunit[id]

orgunit_research[res_act_id] ⊑ ResearchAct[id]

Orgunit[id] ⊑ orgunit_research[org_id]
key(orgunit_research[org_id])

orgunit_act[org_id] ⊑ Orgunit[id]

orgunit_act[activity_id] ⊑ Activity[id]

Correctness and completeness

- Correctness and completeness using the general concept of information capacities [Miller et al.; '93]
- ***Theorem:*** the ontology extraction procedure is an equivalence preserving schema transformation

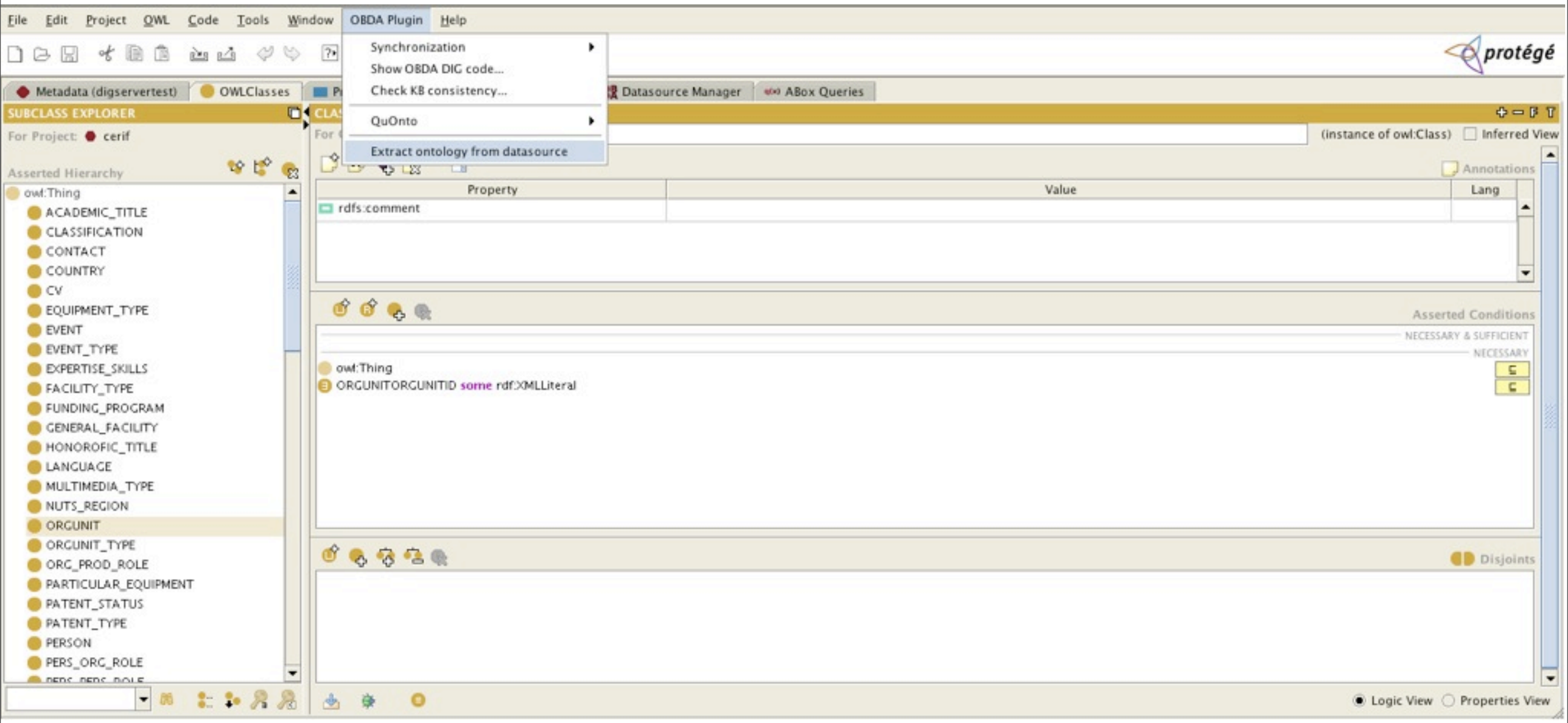
➡ no information loss

➡ queries expressed over the ontology can be answered by simply expanding the generated views!

Ontology extraction in Protégé

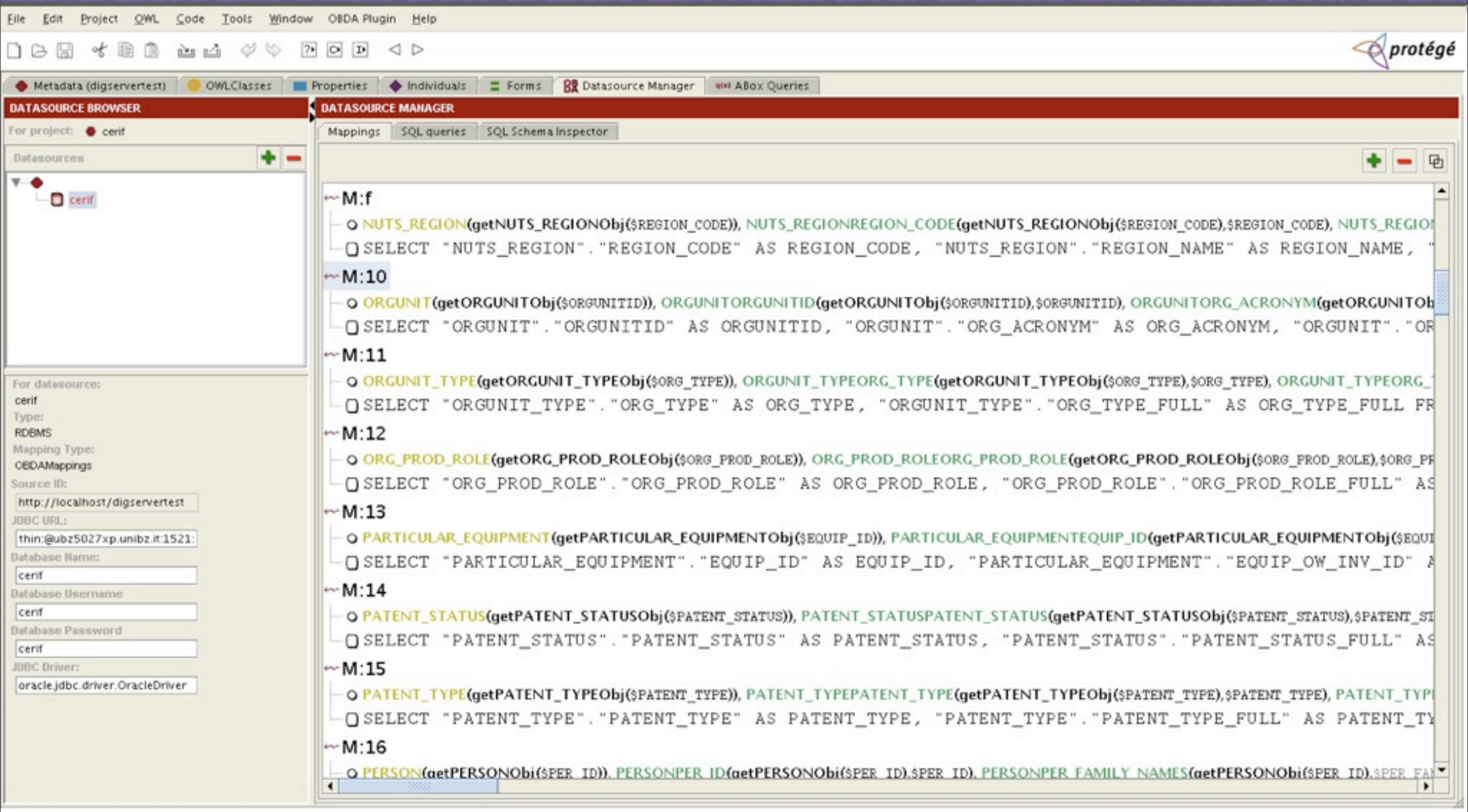
[Rodriguez-Muro et al., IIMAS'08; Poggi et al., OWLED'08; Calvanese et al., OWLED'08],

Ontology extraction in Protégé



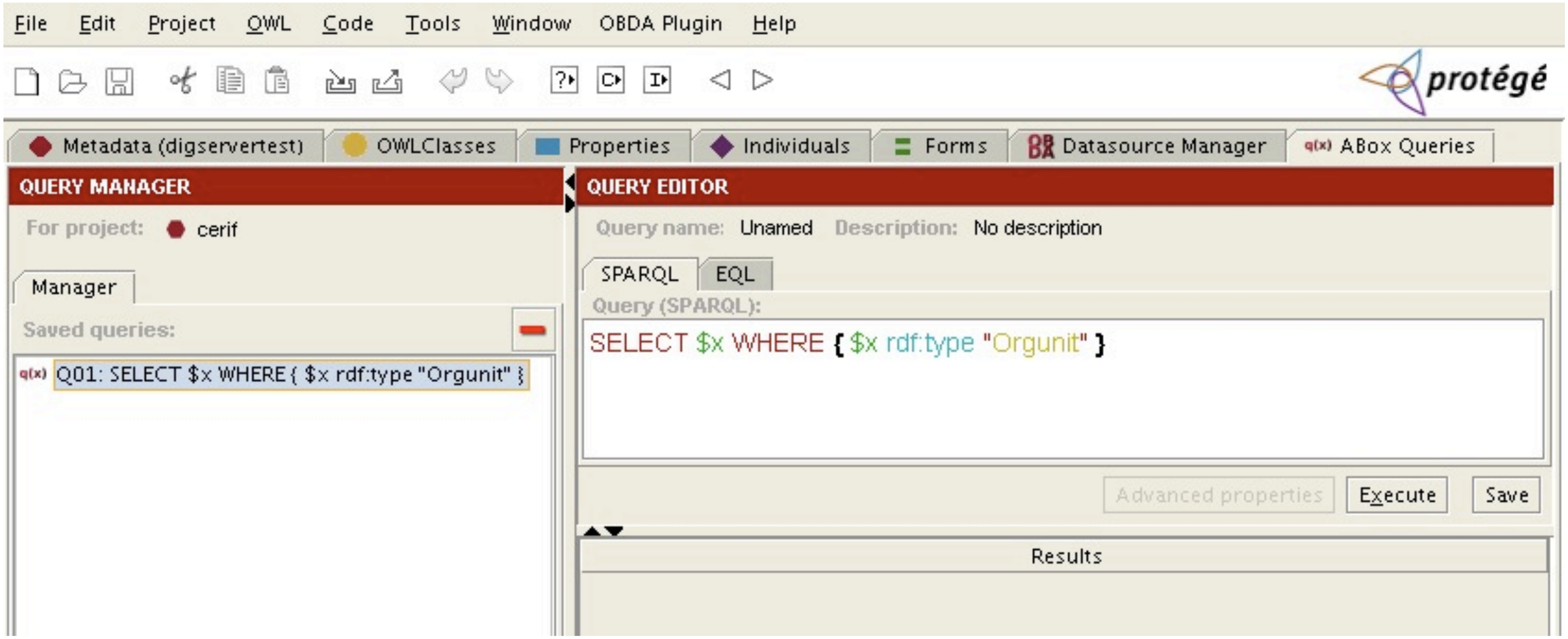
[Rodriguez-Muro et al., IIMAS'08; Poggi et al., OWLED'08; Calvanese et al., OWLED'08],

Ontology extraction in Protégé



[Rodriguez-Muro et al., IIMAS'08; Poggi et al., OWLED'08; Calvanese et al., OWLED'08],

Ontology extraction in Protégé



[Rodriguez-Muro et al., IIMAS'08; Poggi et al., OWLED'08; Calvanese et al., OWLED'08],

CERIF case study

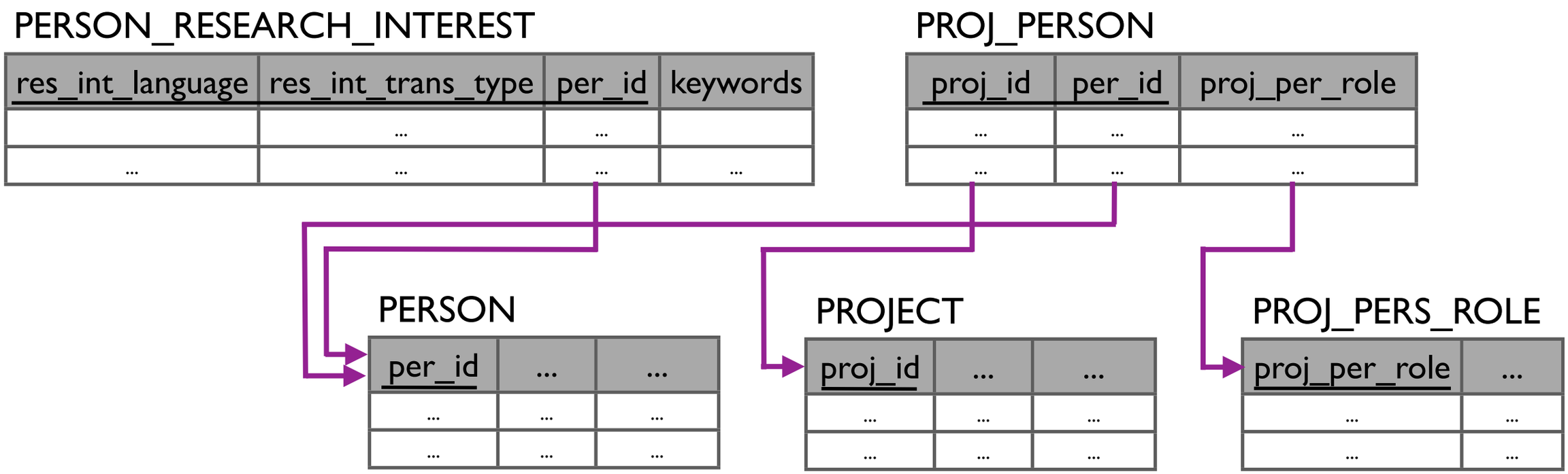
- CERIF (Common European Research Information Framework) database schema
  - http://cordis.europa.eu/cerif/src/toolkit.htm
  - 123 tables; rich with constraints
  - correct ontology constructs were generated for 77 tables

Relation type	#Classified
base	52
relationship	25
specific	0
ambiguous	46

Axiom	#Extracted
inclusion	68
key	63
disjointness	0
covering	0

CERIF: ambiguous relations

- All 46 ambiguous relations fall into one of the two types:



CERIF: ambiguous relations

- All 46 ambiguous relations fall into one of the two types:

PERSON_RESEARCH_INTEREST

res_int_language	res_int_trans_type	per_id	keywords
	...	...	
...	...	...	...

PROJ_PERSON

proj_id	per_id	proj_per_role
...	...	...
...	...	...

RESEARCH_INTEREST

language	trans_type
...	...
...	...

PERSON

per_id	...	...
...	...	...
...	...	...

PROJECT

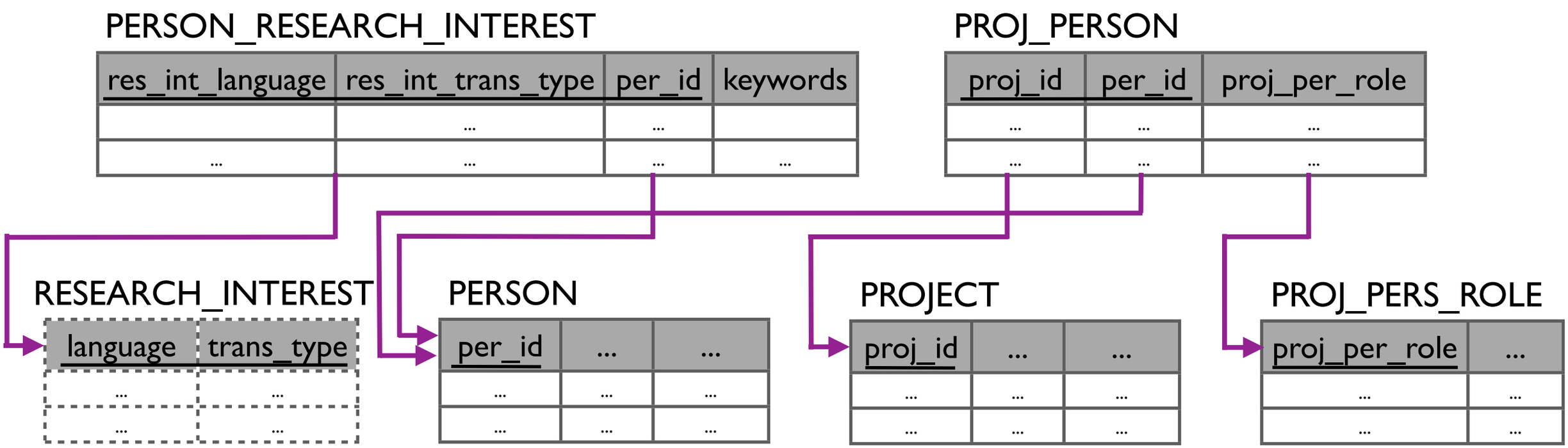
proj_id	...	...
...	...	...
...	...	...

PROJ_PERS_ROLE

proj_per_role	...
...	...
...	...

CERIF: ambiguous relations

- All 46 ambiguous relations fall into one of the two types:



➡ a slight extension of the algorithm correctly generates CERIF ontology

Related work

Related work

- Database reverse engineering (DBRE) in the 90s
  - [Chiang, Barron et al.; '94], [Johannesson; '94], etc.

Related work

- Database reverse engineering (DBRE) in the 90s
  - [Chiang, Barron et al.; '94], [Johannesson; '94], etc.
- RDBs and ontologies for the SW and Information Integration
  - [Astrova; '04], [Volz et al.; '04] use DBRE to migrate DB content into an ontology and to deeply annotate a DB
  - SWARD and Virtuoso automatically generate RDF views over RDBs

Related work

- Database reverse engineering (DBRE) in the 90s
  - [Chiang, Barron et al.; '94], [Johannesson; '94], etc.
- RDBs and ontologies for the SW and Information Integration
  - [Astrova; '04], [Volz et al.; '04] use DBRE to migrate DB content into an ontology and to deeply annotate a DB
  - SWARD and Virtuoso automatically generate RDF views over RDBs
- Linking RDBs and ontologies
  - D2R MAP, R2O, [Poggi et al.;'08] allow to declaratively state mappings
  - [An et al; '05] presents semi-automatic mapping discovery

Advantages of our approach

Advantages of our approach

- Extraction of a formal ontology wrapping relational sources
  - automatic generation of mappings
  - enables automated reasoning to support the designer

Advantages of our approach

- Extraction of a formal ontology wrapping relational sources
  - automatic generation of mappings
  - enables automated reasoning to support the designer
- Ensuring faithfulness of the extracted model
  - no data loss

Current work

- Tool for exploring the logical schema of a DB and for facilitating the annotation of the schema by means of constraints
- Experiments with other real-world DB schemas to identify other peculiarities that schemas exhibit