

OBDA: Ontology-Based Data Access

Tadeusz Pankowski

Politechnika Poznańska

Tadeusz.pankowski@put.poznan.pl

Ontology-Based Data Access

System OBDA jest trójwarstwowym systemem informatycznym, w którym

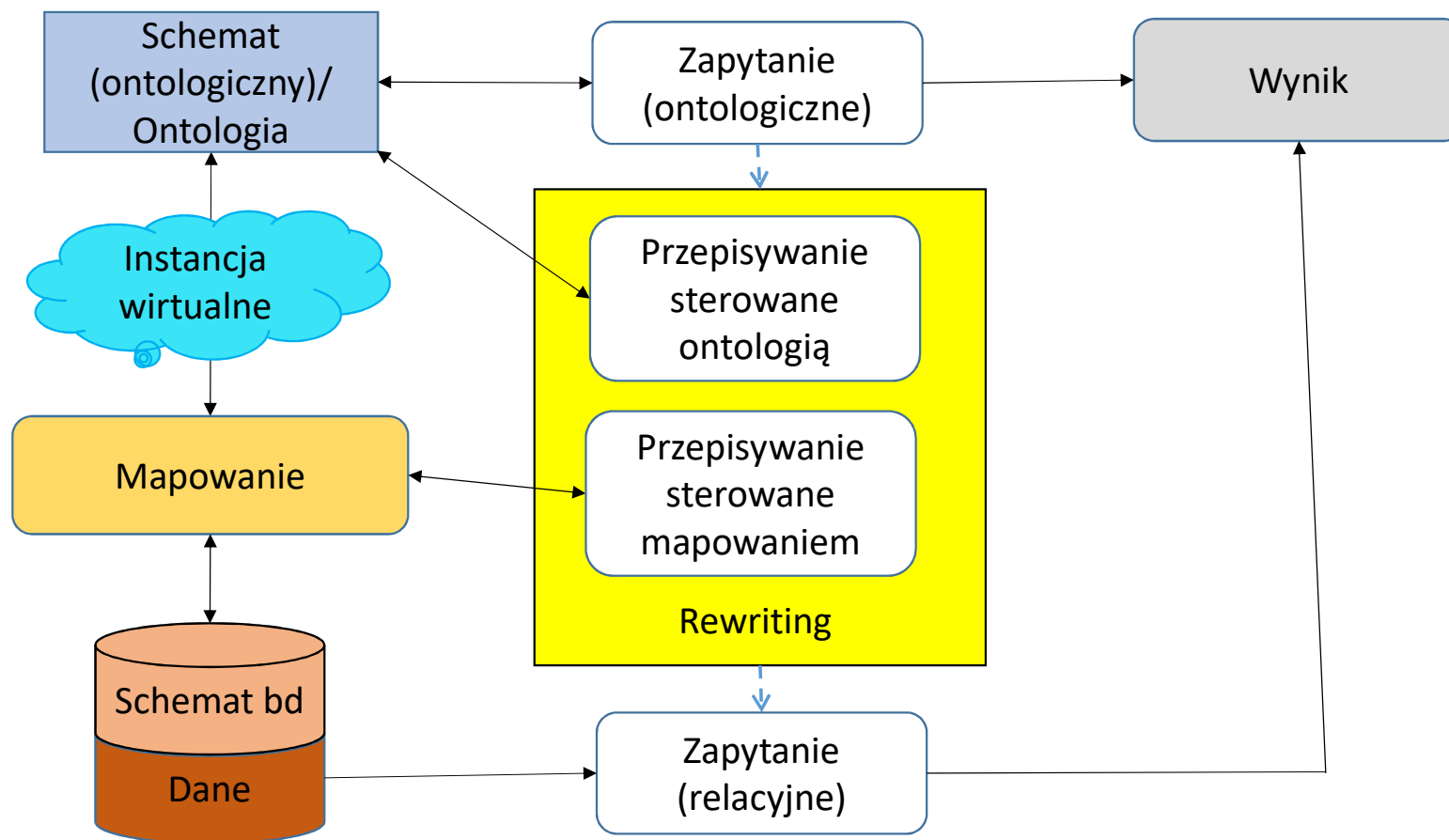
➤ wyróżniamy następujące warstwy (składowe)

- **ontologia** (*ontology*) (schemat konceptualny, intensjonalna część bazy wiedzy), ściślej – a część intensjonalna ontologii (tj. abstrakcyjny opis konceptualizacji dziedziny przedmiotowej),
- **bazy danych** (*dane źródłowe*) (jedna lub wiele) zawierające dane, ściślej – część ekstensjonalna ontologii,
- zbioru **mapowań** (*odwzorowań*) (*mappings*) określających powiązania między ontologią a bazami danych

➤ dostęp do danych

- **zapytania** formułowane są względem ontologii (schematu konceptualnego),
- mapowanie jest do **przepisywania** (*rewriting*) zapytań (najczęściej do SQL),
- przepisane zapytanie jest **wykonywane** w bazie danych (najczęściej relacyjnej)

Architektura systemu OBDA



Ontology-Based Data Access

OBDA może być traktowany jako

- system **integracji danych**, gdzie
- „**schemat globalny**” opisuje wiedzę o obiektach dziedziny, o powiązaniach między obiektami i właściwościach tych obiektów
- jest niezależny od **źródeł danych**.

Opis wykonany jest za pomocą pewnego języka opartego na logice.

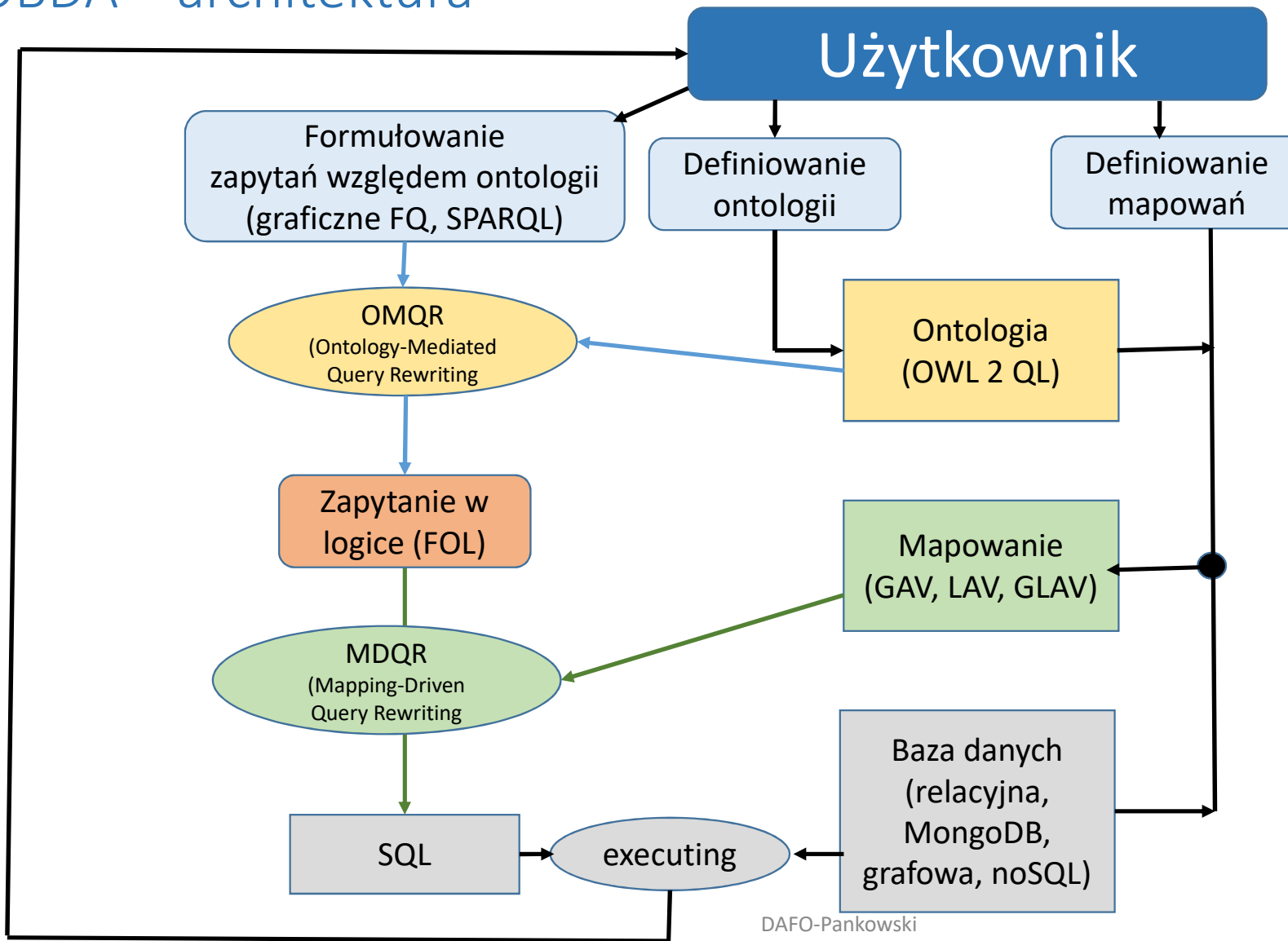
Semantyka systemów OBDA

- W przeciwieństwie do (kompletnych) baz danych, specyfikacja systemu OBDA reprezentuje tzw. zbiór możliwych światów ze względu na:
- obecność ontologii, która jest teorią logiczną, która może być spełniona przez wiele modeli,
- obecność mapowań, dla których istnieje wiele spełniających je modeli.

Odpowiadanie na zapytania w systemie OBDA jest logicznym **wnioskowaniem** w teorii składającej się z ontologii, mapowań i danych.

Równoważnie: odpowiada to wykonywaniu zapytań w systemach z **niepełną informacją** (*incomplete information*) [21] (*R. Reiter, On closed world databases*)

OBDA – architektura



DAFO jako przykład implementacji systemu OBDA

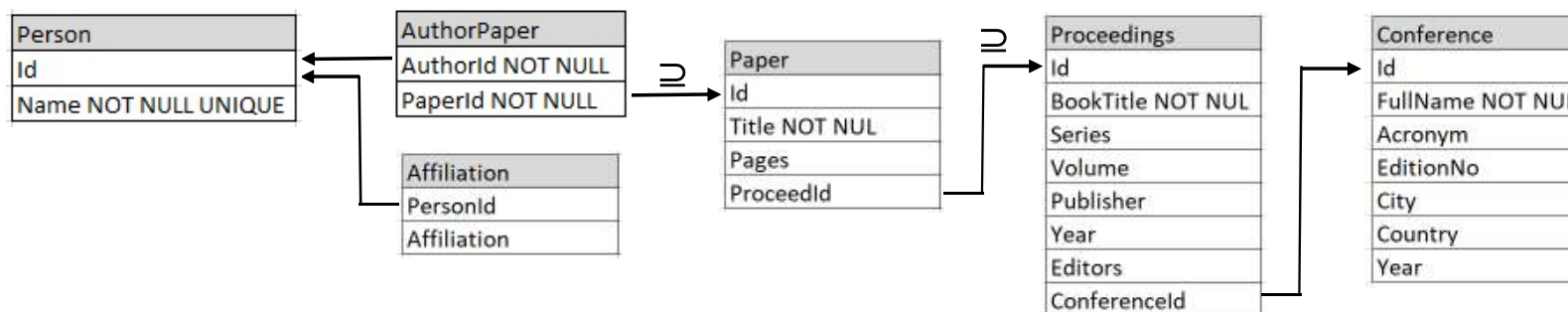
DAFO – Data Access based on Faceted Queries over Ontology

Dostęp do dAnyh oparty na zapytaniach Fasetowych względem Ontologii

DAFO – idea systemu

1. Punkt wyjścia – **schemat relacyjnej bazy**: tabele + warunki spójności, podejście niesformalizowane, diagram (notacja graficzna).
2. Przedstawienie schematu jako formalnie wyspecyfikowanej **ontologii**.
Język logiki pierwszego rzędu (**FOL**), ograniczony do logiki opisowej (description logic) (**DL**).
3. Problem: Jaki zestaw **reguł** (aksjomatów) jest potrzebny (i wystarczający) do zdefiniowania schematu relacyjnej bazy danych.
4. Język ontologiczny **OWL 2 QL**

Relacyjna baza danych – schemat bibliograficznej bazy danych BibRel (BIB_DB)



W bazie danych znajdują się informacje o osobach i ich artykułach przedstawionych na konferencjach naukowych.

Dane opracowano na podstawie bazy DBLP (<https://dblp.uni-trier.de/>)

Person – imię i nazwisko, musi być podane i jest jednoznaczne w tabeli.

Affiliation – każda osoba może być afiliowane w wielu instytucjach,

AuthorPaper – związek autorów z napisanymi artykułami

Paper – artykuł naukowy (konferencyjny)

Proceedings – materiały konferencyjne; jedna konferencja może mieć kilka tomów materiałów

Conference – konferencja.

Strzałki oznaczają klucze obce.

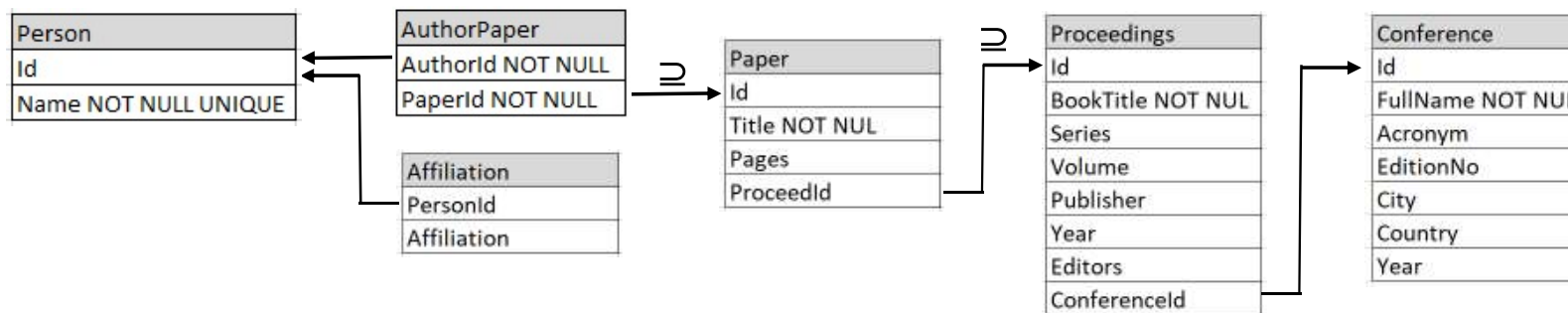
Symbol \subseteq oznacza zawieranie się kolumn (*inclusion dependency*) – szczególnym przypadkiem są klucze obce i niektóre ich odwrotności.

Każda tabela ma klucz główny. Jest nim **Id** lub grupa kolumn: (AuthorId,PaperId), (PersonId,Affiliation).

Repozytorium BIB_DB:

<https://github.com/TPankowski/dafo>

Specyfikacja schematu jako ontologii



Tabele relacyjne bazy danych traktowane są jako jedna z możliwych metod implementacji **ontologii**.

Ontologia – formalna specyfikacja bazy danych.

$$O = (\Sigma, \mathcal{R}, \mathcal{A})$$

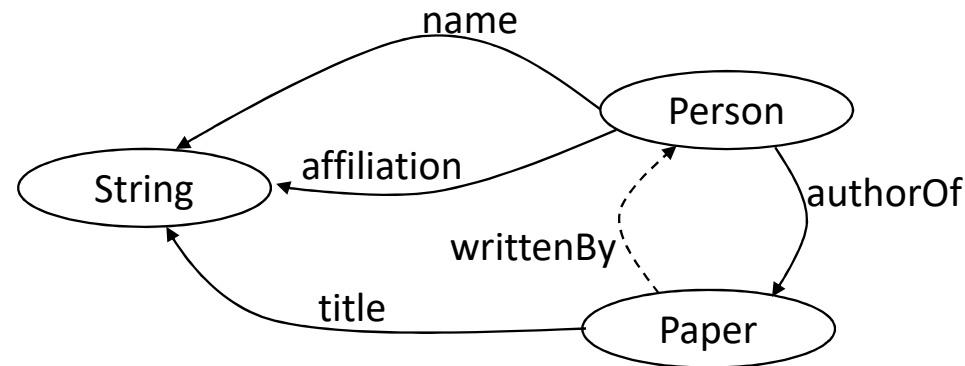
Σ – zbiór symboli występujących w ontologii:

- ❑ predykaty unarne (UP): **classes** (Person, Paper, ...), a wśród nich typy (String, Integer, ...);
- ❑ predykaty binarne (BP): **object properties**, powiązania między klasami (authorOf, ofConference, ...), **data properties (attributes)**, powiązania między klasami i typami (name, title, ...);

\mathcal{R} – zbiór reguł (aksjomatów), TBox. Tworzy część **intensjonalną** ontologii, wiedzę ogólną o świecie (dziedzinie)

\mathcal{A} – zbiór faktów (asercji), ABox, o postaci: C(a), P(a,b). Tworzy część **ekstensjonalną** ontologii, **stan** świata

Ontologia BibOn – predykaty



UP (unary predicates):

Classes:

Person, Paper

Types (type standardowe):

String

BP (binary predicates):

Data properties:

name, affiliation, title

Object properties:

authorOf, writtenBy

Predykaty ekstensjonalne (bazowe):

Ich ekstensje (zbiory wartości) są jawnie pamiętane w bazie danych:

Person, Paper, name, affiliation, title, authorOf

Predykaty intensjonalne (wirtualne):

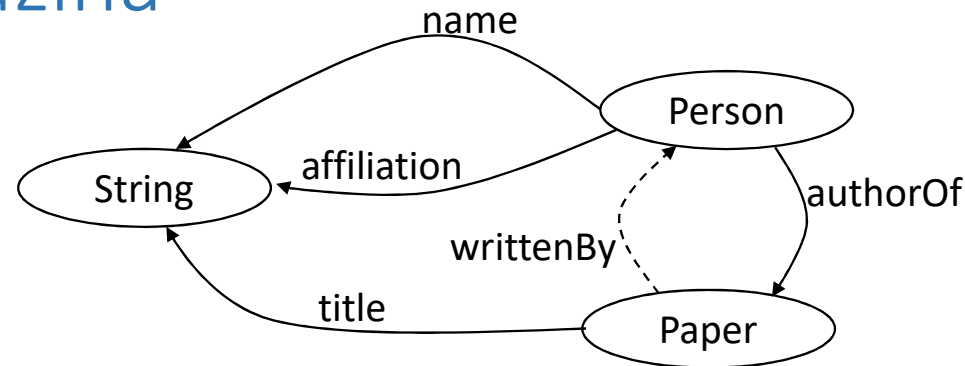
Ich ekstensje nie są jawnie pamiętane, są wyznaczane na podstawie predykatów ekstensjonalnych, jak **widoki** w bazach danych (istotne dla projektanta, a nie dla użytkownika):

writtenBy

Predykaty standardowe – typy standardowe.

String

Ontologia BibOn – reguły (aksjomaty). Dziedzina i przeciwdziedzina



Reguły dotyczące predykatów binarnych: **name**

1. Dziedzina (**domain**).

Dziedziną **name** jest **Person**,

FOL: $\forall x,y \text{ (name}(x,y) \rightarrow \text{Person}(x))$

DL: $\exists \text{name} \sqsubseteq \text{Person}$

$\text{dom}(\text{name}) = \text{Person}$

2. Przeciwdziedzina, obraz (**range**).

Przeciwdziedziną **name** jest **String**:

FOL: $\forall x,y \text{ (name}(x,y) \rightarrow \text{String}(y))$

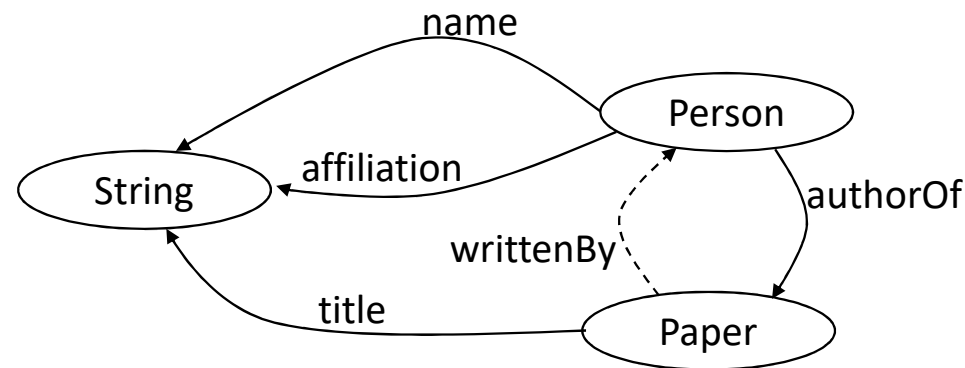
DL: $\exists \text{name}^- \sqsubseteq \text{String}$

$\text{rng}(\text{name}) = \text{String}$

FOL – first order language (język pierwszego rzędu)

DL – description logic (logika opisowa, deskrypcyjna)

Ontologia BibOn – reguły (aksjomaty). Totalność



Reguły dotyczące predykatów binarnych: **name**

3. Totalność właściwości na dziedzinie.
Inaczej: obowiązkowa przynależność elementów dziedziny do powiązania.

Każdy obiekt klasy **Person** musi mieć określone **name**, tzn. **name** jest totalną właściwością na **Person**.

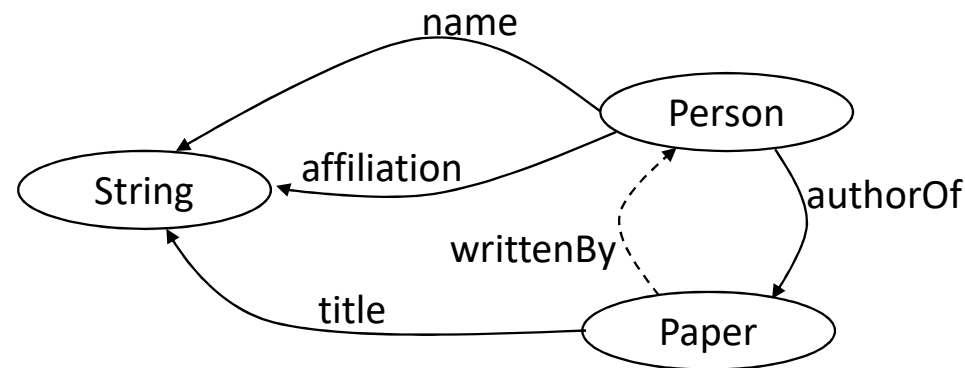
FOL: $\forall x (Person(x) \rightarrow \exists y name(x,y))$

DL: $Person \sqsubseteq \exists name$

4. Totalność właściwości na przeciwdziedzinie.
Inaczej: obowiązkowa przynależność elementów przeciwdziedziny do powiązania.

Nie dotyczy to **name**, bo nie każdy element typu String musi być wartością **name**

Ontologia BibOn – reguły (aksjomaty). Funkcyjność



Reguły dotyczące predykatów binarnych: **name**

5. Funkcyjność na dziedzinie.

name jest funkcją na **Person**, tzn. każda osoba ma co najwyżej jedno **name**.

FOL: $\forall x, y (\text{name}(x, y_1) \wedge \text{name}(x, y_2) \rightarrow y_1 = y_2)$

DL **(funct name)**,

6. Funkcyjność na przeciwdziedzinie – cecha klucza.

Odwrotność **name** jest funkcją na **String**. **name** ma więc właściwość klucza.

FOL: $\forall x, y (\text{name}(x_1, y) \wedge \text{name}(x_2, y) \rightarrow x_1 = x_2)$

DL **(funct name⁻)**,

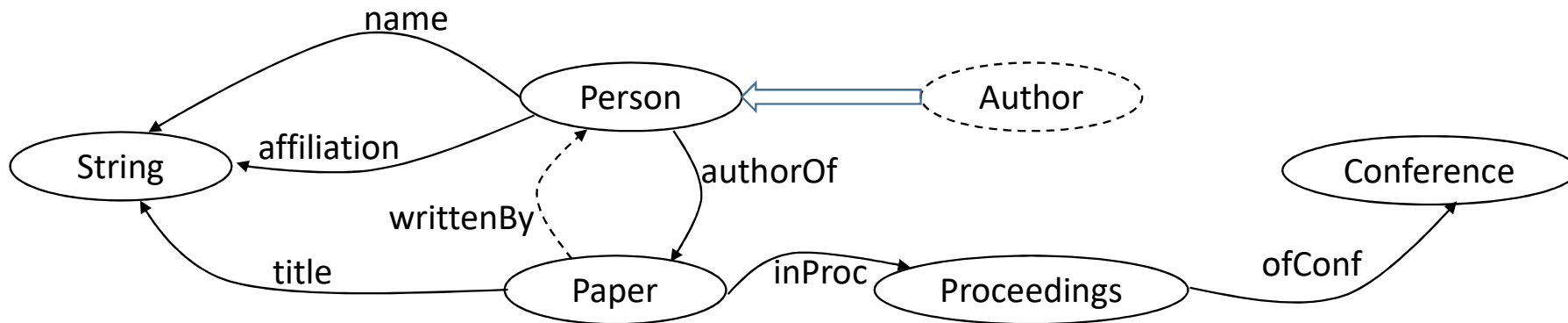
Ontologia BibOn – reguły dotyczące data properties (atrybutów) (BIB_ONTO)

Reguła:	name	affiliation	title
Domain	Person	Person	Paper
Range	String	String	String
Total on domain	Yes/Person	No	Yes/Paper
Total on range	No	No	No
Function on domain	Yes	No	Yes
Function on range (key)	No	No	Yes
	BP _{AF}	BP _{AW}	BP _{AF}

BP_{AF} – atrybuty funkcyjna

BP_{AW} – atrybuty wielowartościowe

Ontologia BibOn – reguły (aksjomaty).



Author jest podklasą **Person**:

$\text{Author} \sqsubseteq \text{Person}$

authorOf nie jest totalne na **Person**, ale jest totalne na podklasie **Author** swojej dziedziny **Person**:

FOL: $\forall x (\text{Author}(x) \rightarrow \exists y \text{authorOf}(x,y))$

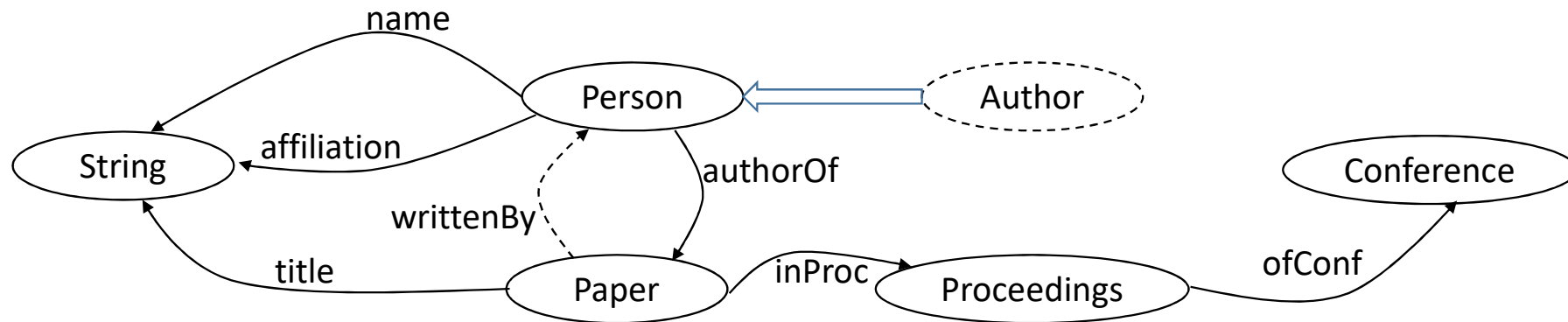
DL: $\text{Author} \sqsubseteq \exists \text{authorOf}$

authorOf jest także totalne na **Paper**:

FOL: $\forall y (\text{Paper}(y) \rightarrow \exists x \text{authorOf}(x,y))$

DL: $\text{Paper} \sqsubseteq \exists \text{authorOf}^-$

Ontologia BibOn – reguły (aksjomaty).



inProc – określa, w jakich materiałach konferencyjnych został opublikowany artykuł konferencyjny. Właściwości:

$\text{dom}(\text{inProc}) = \text{Paper}$,

$\text{rng}(\text{inProc}) = \text{Proceedings}$,

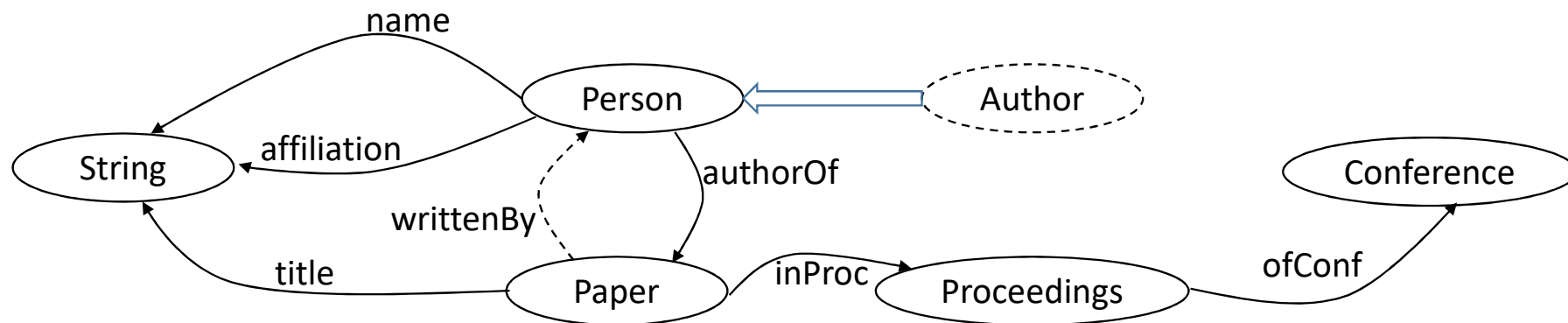
inProc jest totalny na **Paper** – każdy artykuł jest gdzieś opublikowany,

inProc jest totalny na **Proceedings** – każdy tom materiałów konferencyjnych zawiera co najmniej jeden artykuł konferencyjny,

inProc jest funkcją na **Paper** – artykuł jest co najwyżej w jednych materiałach.

inProc nie jest kluczem, tzn. jego odwrotność nie jest funkcją na **Proceedings**. Jeden tom materiałów zawiera wiele artykułów.

Ontologia BibOn – reguły (aksjomaty).



ofConf – określa, jakiej konferencji dotyczyły materiały konferencyjne. Właściwości:

$\text{dom}(\text{ofConf}) = \text{Proceedings}$,

$\text{rng}(\text{ofConf}) = \text{Conference}$,

ofConf jest totalny na **Proceedings** – każdy tom materiałów konferencyjnych dotyczy jakiejś konferencji,

ofConf jest totalny na **Conference** – każda konferencja publikuje materiały konferencyjne,

ofConf jest funkcją na **Proceedings** – materiały dotyczą jednej konferencji.

ofConf nie jest kluczem, tzn. jedna konferencja może wydać kilka tomów materiałów.

Ontologia BibOn – reguły dotyczące object properties (związków)

Reguła:	authorOf	writtenBy	inProc	ofConf
Domain	Person	Paper	Paper	Proceedings
Range	Paper	Person	Proceedings	Conference
Total on (sub)domain	Yes/Author	Paper	Paper	Proceedings
Total on (sub)range	Yes/Paper	Author	Proceedings	Conference
Function on domain	No	No	Yes	Yes
Function on range (key)	No	No	No	No
	BP _{ZW}	BP _{ZW}	BP _{ZF}	BP _{ZF}

BP_{ZF} – związki funkcyjna

BP_{ZW} – związki wielowartościowe

Ontologia – aksjomaty (reguły)

1. Subsumpcja klas i właściwości (*subsumption*)
 - $C' \sqsubseteq C,$ $\forall x (C'(x) \rightarrow C(x))$
 - $P' \sqsubseteq P,$ $\forall x,y (P'(x,y) \rightarrow P(x,y))$
2. Rozłączność (*disjointness*)
 - $C' \sqsubseteq \neg C$ $\forall x (C'(x) \rightarrow \neg C(x))$
 - $P' \sqsubseteq \neg P$ $\forall x,y (P'(x,y) \rightarrow \neg P(x,y))$
3. Dziedzina i przeciwdziedzina (*domain and range*)
 - $\exists P \sqsubseteq C,$ $\forall x,y (P(x,y) \rightarrow C(x))$
 - $\exists P^- \sqsubseteq C,$ $\forall x,y (P(x,y) \rightarrow C(y))$
4. Totalność – przynależność obowiązkowa (*mandatory membership, totality on domain*)
 - $C \sqsubseteq \exists P$ $\forall x (C(x) \rightarrow \exists y P(x,y))$
 - $C \sqsubseteq \exists P^-$ $\forall y (C(y) \rightarrow \exists x P(x,y))$
 - $C \sqsubseteq \exists P.C'$ $\forall x (C(x) \rightarrow \exists y P(x,y) \wedge C'(y))$
 - $C \sqsubseteq \exists P^-.C'$ $\forall y (C(y) \rightarrow \exists x P(x,y) \wedge C'(x))$
5. Funkcyjność (*functionality*)
 - (**funct** P) $\forall x,y (P(x,y_1) \wedge P(x,y_2) \rightarrow y_1 = y_2)$
 - (**funct** P⁻) $\forall x,y (P(x_1,y) \wedge P(x_2,y) \rightarrow x_1 = x_2)$

Repozytorium: BIB_ONTO

<https://github.com/TPankowski/dafo>

Mapowanie ontologii do relacyjnej bazy danych: LAV (Local As View)

Ontologia – relacyjna baza danych.

Mapowania LAV, GAV i GLAV

1. **LAV** – Local As View, oznacza mapowanie schematu globalnego (ontologii) na schematy lokalne (relacyjne).
2. Schemat lokalny jest wtedy **widokiem** (*view*) nad schematem globalnym.
3. **GAV** – Global As View, oznacza mapowanie schematów lokalnych na schemat globalny.
4. Schemat globalny jest wówczas widokiem nad zbiorem schematów lokalnych.
5. **GLAV** – zbiór reguł mapujących, z których niektóre są LAV, a inne GAV.

Ekstensjonalna część ontologii (ABox)

- **Ontologia** – formalna specyfikacja bazy danych.

$$O = (\Sigma, \mathcal{R}, \mathcal{A})$$

- Σ – zbiór symboli występujących w ontologii:
 - ❑ predykaty unarne (UP): **classes** (Person, Paper, String, Integer;
 - ❑ predykaty binarne (BP): **object properties (links)**, powiązania między klasami (authorOf, ofConference, ...), **data properties (attributes)**, powiązania między klasami i typami (name, title, ...);
- \mathcal{R} – zbiór **reguł** (aksjomatów), TBox (część **intensjonalna** ontologii), wiedza ogólna o świecie.
- \mathcal{A} – zbiór **faktów** (asercji), ABox (część **ekstensjonalną** ontologii), **stan** świata.

$\mathcal{A} = \{C(a), \dots, P(a,b), \dots\} =$

Person(1), Person(2), ...,	- unary predicates (klasy)
Paper(101), Paper(102), ...,	
name(1, "John Smith"),	- data properties (atrybuty)
name(2, "Eva Lang"), ...,	
authorOf(1,101), authorOf(1,102), authorOf(2,101), ...}	- object properties (związki)

Schemat relacyjny i schemat bazy danych

1. $Rel = \{R_1, \dots, R_n\}$ – zbiór nazw relacji (tabel),
2. $Att = \{Id, A_1, A_2, \dots\}$ – zbiór atrybutów,
3. $att(R) \subseteq Att$ – typ (zbiór atrybutów) tabeli R
4. Ograniczenia (warunki spójności), $Constr(R)$:
 - a) klucz główny (na jednej tabeli)
 - b) NOT NULL (na jednej tabeli)
 - c) UNIQUE (na jednej tabeli)
 - d) klucz obcy (na dwóch tabelach)
 - e) zależność zawierania (\subseteq) (na dwóch tabelach)
5. **Schemat relacyjny**: nazwa + typ + ograniczenia (a – c), $(R, att(R), Constr(R))$.
6. **Schemat bazy danych**: zbiór schematów relacyjnych + ograniczenia (d – e).


Mapowanie predykatów unarnych – klas.

1. Założenie:
 - każda klasa C mapowana jest na pewien schemat relacyjny $R_C(\text{Id}, \dots)$, gdzie Id jest kluczem głównym schematu,
 - różne klasy mapowane są na różne schematy relacyjne.
2. Formalnie:
 - $\text{MapTab}(C) = R_C$ – klasa C mapowana jest na schemat relacyjny R_C ,
 - $\text{Id} \in \text{attr}(R_C)$ – w zbiorze atrybutów schematu R_C jest klucz główny Id
 - jeśli $C \neq C'$, to $R_C \neq R_{C'}$ – różnym klasom przypisywane są różne schematy relacyjne
 - $R_C[\text{Id}] \cap R_{C'}[\text{Id}] = \emptyset$ – wartości kluczy głównych różnych schematów są rozłączne
3. Mapowanie instancji klas: lewa strona: FOL, prawa strona RRK (*schema mapping*):
 - $\forall x C(x) \rightarrow \exists r R_C(r) \wedge r.\text{Id} = x$
4. Uwaga: reguły powyższe nazywane są regułami (zależnościami) **tgds** (*tuple generating dependencies*). Jeśli nie ma w R_C krotki r , to jest ona generowana i dołączana do R_C . Na początku wszystkie atrybuty r mają wartość NULL.

Mapowanie klas na schematy relacyjne

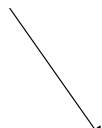
Ontologia: $\mathcal{A} = \{\text{Person}(1), \text{Person}(2), \dots, \text{Paper}(101), \text{Paper}(102), \dots\}$

Person



Id	Name
1	
2	

Paper



Id	Title	Procid
101		
102		

```
CREATE TABLE Person(  
  Id int PRIMARY KEY,  
  ...  
)
```

```
CREATE TABLE Paper(  
  Id int PRIMARY KEY,  
  ...  
)
```

Mapowanie binarnych predykatów na schematy relacyjne

Funkcja wektorowe 3-elementowa funkcja mapująca:

$$(\text{TabMap}, \text{DomMap}, \text{RngMap}) : \text{BP} \rightarrow \text{Rel} \times \text{Att} \times \text{Att}$$

przypisuje każdemu predykatowi binarnemu $P \in \text{BP}$ trzy elementy schematu:

- $\text{TabMap}(P) = R_p$ – nazwa relacji (tabeli),
- $\text{DomMap}(P) = A_p^d$ – atrybut ze zbioru $\text{attr}(R_p)$ zwany atrybutem dziedziny dla P ,
- $\text{RngMap}(P) = A_p^r$ – atrybut ze zbioru $\text{attr}(R_p)$ zwany atrybutem przeciwdziedziny dla P

Sposób zdefiniowania funkcji mapującej zależy od grupy, do której mapowany predykat należy.

Predykaty binarne dzielone są na cztery grupy zależnie od ich przeciwdziedziny i funkcyjności:

1. BP_{AF} – atrybuty funkcyjne (*data properties – functional*), np.: name, title
2. BP_{AW} – atrybuty wielowartościowe (*data properties – multivalued*), np.: affiliation
3. BP_{ZF} – związki funkcyjne (*object properties – functional*), np.: inProc, ofConf
4. BP_{ZW} – związki wielowartościowe (*object properties – multivalued*), np.: authorOf

Mapowanie atrybutów funkcyjnych (*functional data properties*)

1. $P \in BP_{AF}$ – P jest atrybutem funkcyjnym. Określony jest na klasie, a wartości przyjmuje w typie standardowym. Dla każdej wartości ze swojej dziedziny przyjmuje co najwyżej jedną wartość. Na przykład: name, title, ...
2. Niech C będzie dziedziną P:

$$\text{dom}(P) = C$$

Wówczas:

$\text{TabMap}(P) = R_p = R_C$	– atrybut P mapowany jest do tabeli przypisanej jego dziedzinie C,
$\text{DomMap}(P) = A_p^d = \text{Id}$	– atrybutem dziedziny dla P jest klucz główny tabeli R_p ,
$\text{RngMap}(P) = A_p^r$	– atrybutem przeciwdziedziny dla P jest pewien atrybut ze zbioru $\text{attr}(R_p)$

3. Mapowanie instancji klas: lewa strona: FOL, prawa strona RRK:

$$\blacksquare \quad \forall x, y \, P(x, y) \rightarrow \exists r \, R_p(r) \wedge r.\text{Id} = x \wedge r. A_p^r = y$$

Przykład (zakładamy jednoznaczność nazw w ontologii):

$\text{Map}(\text{Name}) = (\text{Person}, \text{Id}, \text{Name}),$

$\text{Map}(\text{Title}) = (\text{Paper}, \text{Id}, \text{Title}),$

Mapowanie atrybutów funkcyjnych (*functional data properties*)

Map(Name) = (Person, Id, Name),


Map(Title) = (Paper, Id, Title),

Mapowanie instancji atrybutów funkcyjnych:

$$\forall x, y \text{ name}(x, y) \rightarrow \exists r \text{ Person}(r) \wedge r.\text{Id} = x \wedge r.\text{Name} = y$$

Ontologia: $\mathcal{A} = \{\text{name}(1, \text{"John Smith"}), \text{name}(2, \text{"Eva Lang"}), \dots, \text{title}(101, \text{"SQL"}), \text{title}(102, \text{"AI"}), \dots\}$

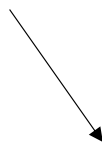
Person



Id	Name
1	John Smith
2	Eva Lang

```
CREATE TABLE Person(  
  Id int PRIMARY KEY,  
  Name varchar(40),  
  ...  
)
```

Paper



Id	Title	Procid
101	SQL	
102	AI	

```
CREATE TABLE Paper(  
  Id int PRIMARY KEY,  
  Title varchar(80),  
  ...  
)
```

Mapowanie atrybutów wielowartościowych (*multivalued data properties*)

1. $P \in BP_{AW}$ – P jest atrybutem wielowartościowym. Określony jest na klasie, a wartości przyjmuje w typie standardowym. Dla każdej wartości ze swojej dziedziny może mieć więcej niż jedną wartość. Na przykład: affiliation, ...
2. Niech $\text{dom}(P) = C$. Wówczas:
 - $\text{TabMap}(P) = R_p$ – atrybut P mapowany jest do 2-kolumnowej tabeli,
 - $\text{DomMap}(P) = A_p^d$ – atrybutem dziedziny należy do $\text{attr}(R_p)$ i jest kluczem obcym odwołującym się do klucza głównego tabeli R_C
 - $\text{RngMap}(P) = A_p^r$ – atrybutem przeciwdziedziny należy do $\text{attr}(R_p)$, para (A_p^d, A_p^r) tworzy **klucz główny** tabeli R_p .
3. Mapowanie instancji klas: lewa strona: FOL, prawa strona RRK:
 - $\forall x, y P(x, y) \rightarrow \exists r R_p(r) \wedge r. A_p^d = x \wedge r. A_p^r = y$

Przykład (zakładamy jednoznaczność nazw w ontologii):

$\text{Map}(\text{affiliation}) = (\text{Affiliation PersonId}, \text{Affiliation}),$

Mapowanie atrybutów wielowartościowych (*multivalued data properties*)

Map(affiliation) = (Affiliation PersonId, Affiliation),

Mapowanie instancji atrybutów funkcyjnych:

$$\forall x,y \text{ (affiliation (x,y) } \rightarrow \exists r \text{ Affiliation (r) } \wedge r.\text{PersonId} = x \wedge r.\text{Affiliation} = y)$$

Ontologia: $\mathcal{A} = \{\text{affiliation}(1, \text{"Google"}), \text{affiliation}(1, \text{"MIT"}), \text{affiliation}(2, \text{"MS"}), \dots\}$

↓

Affiliation	
PersonId	Affiliation
1	Google
1	MIT
2	MS

```
CREATE TABLE Affiliation(  
  PersonId int references Person(Id),  
  Affiliation varchar(40),  
  PRIMARY KEY (PersonId, Affiliation)  
)
```

DAFO-Pankowski

Mapowanie związków funkcyjnych (*functional object properties*)

1. $P \in BP_{ZF}$ – P jest związkiem funkcyjnym. Jego dziedziną i przeciwdziedziną są klasy. Dla każdego obiektu ze swojej dziedziny przyjmuje co najwyżej jedną wartość – obiekt z przeciwdziedziny. Na przykład: inProc, ofConf, ...
2. Niech C będzie dziedziną, a D przeciwdziedziną P :

$$\text{dom}(P) = C, \quad \text{rng}(P) = D$$

Wówczas:

$\text{TabMap}(P) = R_p = R_C$	– atrybut P mapowany jest do tabeli przypisanej jego dziedzinie C ,
$\text{DomMap}(P) = A_p^d = \text{Id}$	– atrybutem dziedzinowym jest klucz główny tabeli R_p ,
$\text{RngMap}(P) = A_p^r$	– atrybut przeciwdziedzinowy należy do $\text{attr}(R_p)$ i jest kluczem obcym odwołującym się do klucza głównego tabeli R_D .

3. Mapowanie instancji klas: lewa strona: FOL, prawa strona RRK:

$$\blacksquare \quad \forall x, y \, P(x, y) \rightarrow \exists r \, R_p(r) \wedge r.\text{Id} = x \wedge r. A_p^r = y$$

Przykład :

$\text{Map}(\text{inProc}) = (\text{Paper}, \text{Id}, \text{ProcId})$

$\text{Map}(\text{ofConf}) = (\text{Proceeding}, \text{Id}, \text{ConfId})$

Mapowanie atrybutów funkcyjnych (*functional data properties*)

Map(inProc) = (Paper, Id, ProcId)

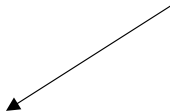
Map(ofConf) = (Proceeding, Id, ConfId)

Mapowanie instancji atrybutów funkcyjnych:

$$\forall x, y \text{ inProc}(x, y) \rightarrow \exists r \text{ Paper}(r) \wedge r.\text{Id} = x \wedge r.\text{ProcId} = y$$

Ontologia: $\mathcal{A} = \{\text{inProc}(101, 1001), \text{inProc}(102, 1001), \text{inProc}(103, 1002), \dots, \text{ofConf}(1001, 2001), \text{ofConf}(1002, 2002), \dots\}$

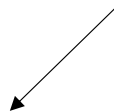
Paper



Id	Title	ProcId
101	SQL	1001
102	AI	1001
103	XML	1002

```
CREATE TABLE Paper(  
  Id int PRIMARY KEY,  
  Title varchar(80),  
  ProcId int references Proceedings(Id),  
  ...)
```

Proceedings



Id	BookTitle	ConfId
1001		2001
1002		2002

```
CREATE TABLE Proceedings(  
  Id int PRIMARY KEY,  
  ConfId int references Conference(Id),  
  ...)
```

Mapowanie związków wielowartościowych (*multivalued object properties*)

1. $P \in BP_{zw}$ – P jest związkiem wielowartościowym. Jego dziedziną i przeciwdziedziną są klasy. Każdy obiekt z dziedziny może wiązać z wieloma obiektami przeciwdziedziny. Na przykład: authorOf, ...
2. Niech $dom(P) = C$, $rng(P) = D$. Wówczas:

$TabMap(P) = R_p$	– atrybut P mapowany jest do 2-kolumnowej tabeli,
$DomMap(P) = A_p^d$	– atrybut dziedziny należy do $attr(R_p)$ i jest kluczem obcym odwołującym się do klucza głównego tabel R_C
$RngMap(P) = A_p^r$	– atrybut przeciwdziedziny należy do $attr(R_p)$ i jest kluczem obcym odwołującym się do klucza głównego tabel R_D

Para (A_p^d, A_p^r) tworzy **klucz główny** tabeli R_p .
3. Mapowanie instancji klas: lewa strona: FOL, prawa strona RRK:
 - $\forall x, y P(x, y) \rightarrow \exists r R_p(r) \wedge r. A_p^d = x \wedge r. A_p^r = y$

Przykład :

Map(authorOf) = (AuthorPaper, PersonId, PaperId),

Mapowanie atrybutów wielowartościowych (*multivalued data properties*)

Map(authorOf) = (AuthorPaper, PersonId, PaperId),

Mapowanie instancji atrybutów funkcyjnych:

$$\forall x,y \text{ (authorOf}(x,y) \rightarrow \exists r \text{ AuthorPaper}(r) \wedge r.\text{PersonId} = x \wedge r.\text{PaperId} = y)$$

Ontologia: A = {authorOf(1, 101), authorOf(1, 102), authorOf(2, 102), ...}

↓

PersonId	PaperId
1	101
1	102
2	102

```
CREATE TABLE AuthorPaper(  
  PersonId int references Person(Id),  
  PaperId int references Person(Id),  
  PRIMARY KEY (PersonId, PaperId)  
)
```

DAFO-Pankowski

Mapowanie. Podsumowanie (1)

Person

<u>Id</u>	Name
1	John Smith
2	Eva Lang

Paper

<u>Id</u>	Title	ProcId
101	SQL	1001
102	AI	1001
103	XML	1002

Proceedings

<u>Id</u>	BookTitle	ConfId
1001	ACM Proc	2001
1002	DEXA Proc	2002

Affiliation

<u>PersonId</u>	<u>Affiliation</u>
1	Google
1	MIT
2	MS

AuthorPaper

<u>PersonId</u>	<u>PaperId</u>
1	101
1	102
2	102

Conference

<u>Id</u>	Acronym	Country
2001	ACM	USA
2002	DEXA	France

Nazwa predykatu	Rodzaj	TabMap	DomMap	RngMap	Warunki spójności
Person	UP	Person			Pkey: Id
name	BP _{AF}	Person	Id	Name	
affiliation	BP _{AW}	Affiliation	PersonId	Affiliation	Pkey: (PersonId, Affiliation) Fkey: PersonId → Person.Id
inProc	BP _{ZF}	Paper	Id	ProcId	Fkey: ProcId → Proceedings.Id
authorOf	BP _{ZW}	AuthorPaper	PersonId	PaperId	Pkey: (PersonId, PaperId) Fkey: PersonId → Person.Id

Mapowanie. Podsumowanie (2)

Omawiana część mapowania uwzględnia:

1. Sygnaturę wyliczającą predykaty z podziałem na unarne i binarne oraz na ekstensjonalne i intensjonalne.
2. Reguły definiujące dziedziny i przeciwdziedziny predykatów binarnych.
3. Reguły określające funkcyjność predykatów binarnych na ich dziedzinach.
4. Z powyższych reguł wynika mapowanie predykatów na schematy relacyjne z uwzględnieniem kluczy głównych i kluczy obcych

Nazwa predykatu	Rodzaj	TabMap	DomMap	RngMap	Warunki spójności
Person	UP	Person			Pkey: Id
name	BP _{AF}	Person	Id	Name	
affiliation	BP _{AW}	Affiliation	PersonId	Affiliation	Pkey: (PersonId, Affiliation) Fkey: PersonId → Person.Id
inProc	BP _{ZF}	Paper	Id	ProcId	Fkey: ProcId → Proceedings.Id
authorOf	BP _{ZW}	AuthorPaper	PersonId	PaperId	Pkey: (PersonId, PaperId) Fkey: PersonId → Person.Id

Ontologia – aksjomaty (reguły)

1. Subsumpcja klas i właściwości (*subsumption*)
 - $C' \sqsubseteq C,$ $\forall x (C'(x) \rightarrow C(x))$
 - $P' \sqsubseteq P,$ $\forall x,y (P'(x,y) \rightarrow P(x,y))$
2. Rozłączność (*disjointness*)
 - $C' \sqsubseteq \neg C$ $\forall x (C'(x) \rightarrow \neg C(x))$
 - $P' \sqsubseteq \neg P$ $\forall x,y (P'(x,y) \rightarrow \neg P(x,y))$
3. Dziedzina i przeciwdziedzina (*domain and range*)
 - $\exists P \sqsubseteq C,$ $\forall x,y (P(x,y) \rightarrow C(x))$
 - $\exists P^- \sqsubseteq C,$ $\forall x,y (P(x,y) \rightarrow C(y))$
4. Totalność – przynależność obowiązkowa (*mandatory membership*)
 - $C \sqsubseteq \exists P$ $\forall x (C(x) \rightarrow \exists y P(x,y))$
 - $C \sqsubseteq \exists P^-$ $\forall y (C(y) \rightarrow \exists x P(x,y))$
 - $C \sqsubseteq \exists P.C'$ $\forall x (C(x) \rightarrow \exists y P(x,y) \wedge C'(y))$
 - $C \sqsubseteq \exists P^-.C'$ $\forall y (C(y) \rightarrow \exists x P(x,y) \wedge C'(x))$
5. Funkcyjność (*functionality*)
 - (**funct** P) $\forall x,y (P(x,y_1) \wedge P(x,y_2) \rightarrow y_1 = y_2)$
 - (**funct** P⁻) $\forall x,y (P(x_1,y) \wedge P(x_2,y) \rightarrow x_1 = x_2)$

Mapowanie reguł: totalność predykatu na klasie dziedzinowe (na domenie)

Totalność – przynależność obowiązkowa) (*mandatory membership*).

P ma określoną wartość na **każdym obiekcie swojej dziedzinowej klasy**.

$$\rho = C \sqsubseteq \exists P, \quad \text{lub:} \quad \rho = \forall x (C(x) \rightarrow \exists y P(x,y))$$

Kolumna atrybutu przeciwdziedziny nie może być NULL:

$$\begin{aligned} \text{Map}(\rho) &= \{A_p^r \text{ NOT NULL} \} && \text{– dla predykatów funkcyjnych,} \\ \text{Map}(\rho) &= \{A_p^r \text{ NOT NULL, } R_c[\text{Id}] \subseteq R_p[A_p^d]\} && \text{– dla predykatów wielowartościowych.} \end{aligned}$$

Przykłady dla predykatów funkcyjnych:

1. $\forall x (\text{Person}(x) \rightarrow \exists y \text{ name}(x,y))$ – każda osoba ma jakieś nazwisko:
Person: Name **varchar(40)** NOT NULL
2. $\forall x (\text{Paper}(x) \rightarrow \exists y \text{ inProc}(x,y))$ – każdy artykuł występuje w jakichś materiałach konferencyjnych:
Paper: ProcId **int references** Proceedings(Id) NOT NULL

Mapowanie reguł: totalność predykatu na klasie dziedzinowe (na domenie)

Totalność – przynależność obowiązkowa) (*mandatory membership*).

P ma określoną wartość na **każdym obiekcie swojej dziedzinowej klasy**.

$$\rho = C \sqsubseteq \exists P, \quad \text{lub:} \quad \rho = \forall x (C(x) \rightarrow \exists y P(x,y))$$

Kolumna atrybutu przeciwdziedziny nie może być NULL:

$$\begin{aligned} \text{Map}(\rho) &= \{A_p^r \text{ NOT NULL} \} && \text{– dla predykatów funkcyjnych,} \\ \text{Map}(\rho) &= \{A_p^r \text{ NOT NULL, } R_c[\text{Id}] \subseteq R_p[A_p^d]\} && \text{– dla predykatów wielowartościowych.} \end{aligned}$$

Przykłady dla predykatów wielowartościowych:

1. $\forall x (\text{Person}(x) \rightarrow \exists y \text{affiliation}(x,y))$ – każda osoba ma jedną lub więcej afiliacji:
Affiliation: Affiliation **varchar(80)** NOT NULL
Person.Id \subseteq Affiliation.PersonId – implementowane za pomocą trigera.

Person		Affiliation	
Id	Name	PersonId	Affiliation
1	John Smith	1	Google
2	Eva Lang	1	MIT
		2	MS

Mapowanie reguł: totalność predykatu na klasie przeciwdziedziny (na range)

Totalność – przynależność obowiązkowa) (*mandatory membership*).

P ma jako wartość przyjmuje **każdy obiekt swojej klasy przeciwdziedziny**.

$$\rho = C \sqsubseteq \exists P^-, \quad \text{lub:} \quad \rho = \forall y (C(y) \rightarrow \exists x P(x,y))$$

$\text{Map}(\rho) = \{R_C[\text{Id}] \subseteq R_P[A_p']\}$ – dla predykatów (funkcyjnych i wielowartościowych).

Przykłady:

1. $\forall y (\text{Paper}(y) \rightarrow \exists x \text{authorOf}(x,y))$ – każda publikacja ma co najmniej jednego autora:
 $\text{Paper.Id} \subseteq \text{AuthorPaper.PaperId}$ – implementowane za pomocą triggera.
2. $\forall y (\text{Conference}(y) \rightarrow \exists x \text{ofConf}(x,y))$ – każda konferencja ma co najmniej jeden tom materiałów:
 $\text{Conference.Id} \subseteq \text{Proceedings.ConfId}$ – implementowane za pomocą triggera.

Paper

Id	Title	ProclId
101	SQL	1001
102	AI	1001
103	XML	1002

AuthorPaper

PersonId	PaperId
1	101
1	102
2	102

Proceedings

Id	BookTitle	ConfId
1001	ACM Proc	2001
1002	DEXA Proc	2002

Conference

Id	Acronym	Country
2001	ACM	USA
2002	DEXA	France

Mapowanie reguł: funkcyjność na przeciwdziedzinie

Odwrotność P jest funkcją:

$$\rho = (\text{funct } P^-) \quad \text{inaczej} \quad \rho = \forall x,y (P(x_1,y) \wedge \text{name}(x_2,y) \rightarrow x_1 = x_2)$$

$\text{Map}(\rho) = A_p^r$ UNIQUE – jednoznaczna wartość w kolumnie.

Przykłady:

1. $\forall x,y (\text{name}(x_1,y) \wedge \text{name}(x_2,y) \rightarrow x_1 = x_2)$ – nazwisko jednoznacznie identyfikuje osobę

Person: Name **varchar**(40) NOT NULL UNIQUE

Definiowanie predykatów intensjonalnych (pochodnych)

Reguły przepisывania (rewriting rules)

1. Inwersja (inversion)

$$\boxed{\square} P' \equiv P^-, \quad \forall x, y P'(x, y) \Leftrightarrow P(y, x))$$

2. Kompozycja (chain)

$$\boxed{\square} P_1 \bullet P_2 \equiv P, \quad \forall x, y \exists z (P_1(x, z) \wedge P_2(z, y) \Leftrightarrow P(x, y))$$

3. Specjalizacja sterowana wartością (value-driven specialization):

$$\boxed{\square} \exists P.\{a\} \equiv C, \quad \forall x \exists y (P(x, y) \wedge y = a \Leftrightarrow C(x))$$

4. Specjalizacja sterowana klasą (class-driven specialization):

$$\boxed{\square} \exists P.C' \equiv C, \quad \forall x \exists y (P(x, y) \wedge C'(y) \Leftrightarrow C(x))$$

Predykat występujący po prawej stronie reguły jest predykatem **intensjonalnym**.

Podczas translacji zapytania predykaty intencjonalne są rekursywnie zastępowane ich definicjami (lewe strony reguł), po odpowiedniej **unifikacji zmiennych**.

Jest to proces **przepisywania zapytania** – *query rewriting*.

W efekcie, zapytanie składa się tylko z predykatów **ekstensjonalnych**.

Reguły przepisывania (rewriting rules) – Przykłady

1. Inwersja (inversion)

$$\boxed{\square} \quad P' \equiv P^-, \quad \forall x, y \quad P'(x, y) \Leftrightarrow P(y, x)$$

writtenBy \equiv authorOf⁻

2. Złożenie (chain)

$$\boxed{\square} \quad P_1 \bullet P_2 \equiv P, \quad \forall x, y \exists z (P_1(x, z) \wedge P_2(z, y) \Leftrightarrow P(x, y))$$

inProc \bullet ofConf \equiv presentedAt,

3. Specjalizacja sterowana wartością (value-driven specialization):

$$\boxed{\square} \quad \exists P.\{a\} \equiv C, \quad \forall x \exists y (P(x, y) \wedge y = a \Leftrightarrow C(x))$$

\exists acronym. $\{$ 'ACM' $\} \equiv$ ACMConf

\exists country. $\{$ 'USA' $\} \equiv$ USAConf

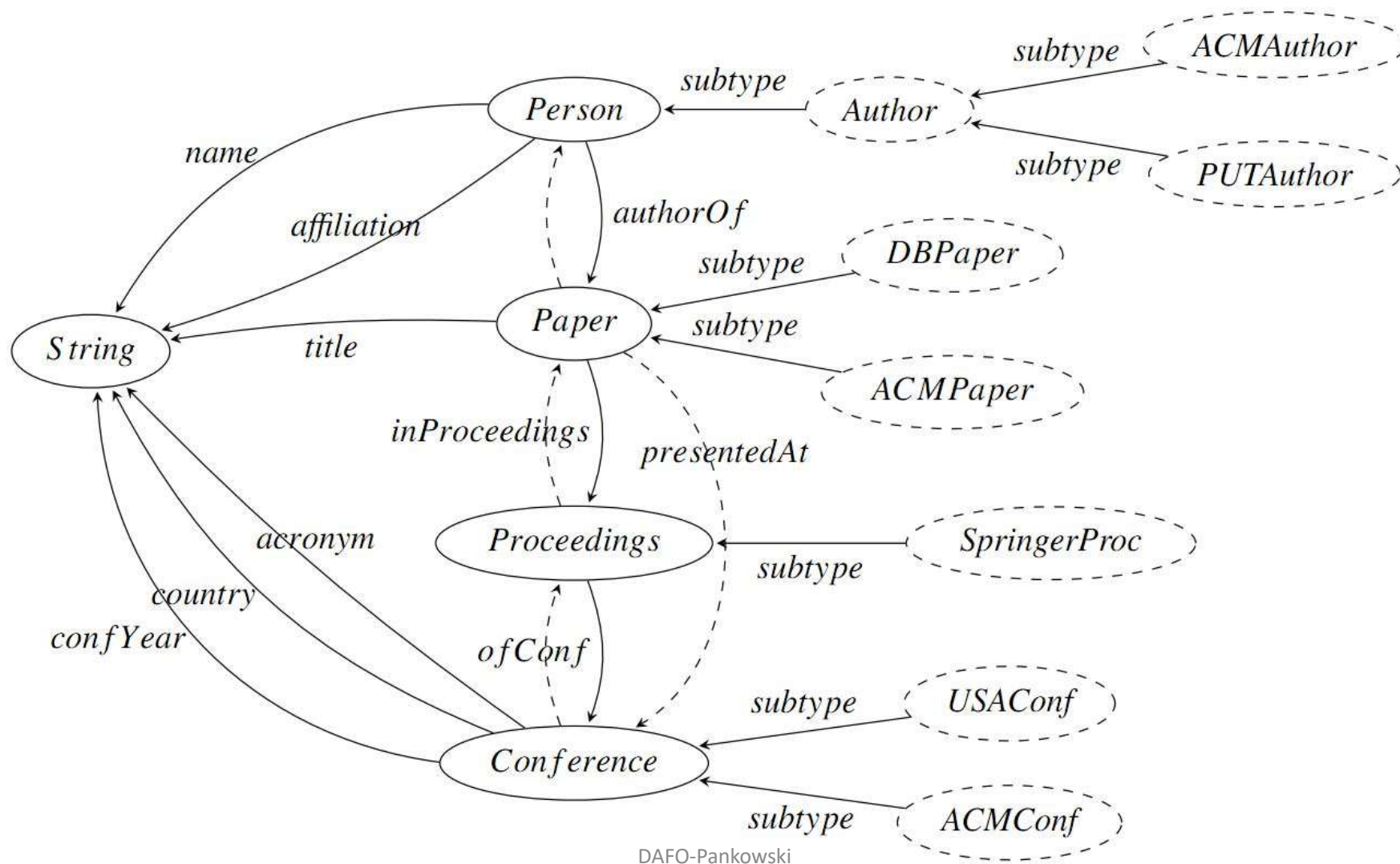
4. Specjalizacja sterowana klasą (class-driven specialization):

$$\boxed{\square} \quad \exists P.C' \equiv C, \quad \forall x \exists y (P(x, y) \wedge C'(y) \Leftrightarrow C(x))$$

\exists presentedAt.ACMLConf \equiv ACMPaper

\exists authorOf.ACMPaper \equiv ACMAuthor

BibOn – Ontologia



Query formulation

All objects of a given class

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees:

Edges of (sub)trees:

Edge1: →

ACCEPT

Faceted interface:



EXECUTE

FO & SQL FORMS

NESTED ANSWER

Answers

1. Serge Abiteboul; INRIA, France
2. Foto N. Afrati; University of Athens, Greece
3. Marcelo Arenas; Pontificia Universidad Catolica de Chile, Chile
4. Magdalena Balazinska; University of Washington, Seattle, Washington, USA
5. Pablo Barceló; University of Chile, Chile
6. Michael Benedikt; University of Oxford, UK
7. Iovka Boneva; University of Lille, France

Queries in FOL and SQL

DAFO - creating and executing faceted queries

FO before rewriting:

$\exists x$ Person(x)

FO after rewriting:

$\exists x$ Person(x)

SQL:

```
SELECT x.Id AS Id FROM Person AS x
```

T. Pankowski, DAFO

DAFO - creating and executing faceted queries

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees:

Edges of (sub)trees:

Edge1: →

Edge2: →

Edge3: →

ACCEPT

Faceted interface:



- Set to AND
- Set to OR
- EXCLude
- INCLude
- Count OFF
- Count ON
- Attribute values
- Remove unchecked SIBLINGS
- Remove ALL unchecked
- Focus On
- Explore
- Clone subtree

Answers

T. Pankowski, DAFO

Persons with given affiliations

DAFO - creating and executing faceted queries

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees:

Edges of (sub)trees:

Edge1: →

Faceted interface:

Answers

1. Ireneusz Czarnowski; Gdynia Maritime University, Poland
2. Czesław Jedrzejek; PUT, Poland
3. Piotr Jedrzejowicz; Gdynia Maritime University, Poland
4. Tadeusz Morzy; PUT, Poland
5. Tadeusz Pankowski; PUT, Poland
6. Andrzej Szwaab; PUT, Poland
7. Robert Wrembel; PUT, Poland

T. Pankowski, DAFO

Persons with given affiliations

DAFO - creating and executing faceted queries

FO before rewriting:



FO after rewriting:



SQL:

```
SELECT DISTINCT x.PersonId AS Id FROM Affiliation AS x WHERE (x.Affiliation = 'Gdynia Maritime University, Poland'  
OR x.Affiliation = 'PUT, Poland')
```

T. Pankowski, DAFO

Persons with a given affiliation

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees:

Edges of (sub)trees:

Edge1: →

Faceted interface:

Answers

1. Ireneusz Czarnowski; Gdynia Maritime University, Poland
2. Czeslaw Jedrzejek; PUT, Poland
3. Piotr Jedrzejowicz; Gdynia Maritime University, Poland
4. Filip Mazowiecki; University of Warsaw, Poland
5. Tadeusz Morzy; PUT, Poland
6. Filip Murlak; University of Warsaw, Poland
7. Tadeusz Pankowski; PUT, Poland
8. Pawel Parys; University of Warsaw, Poland
9. Andrzej Szwabe; PUT, Poland
10. Robert Wrembel; PUT, Poland

DAFO-Pankowski

Persons with a given affiliation

DAFO - creating and executing faceted queries

FO before rewriting:

└─ Person(x)
 └─ affiliation(x,x1)
 └─ x1 = %Poland%

FO after rewriting:

└─ Person(x)
 └─ affiliation(x,x1)
 └─ x1 = %Poland%

SQL:

```
SELECT DISTINCT x.PersonId AS Id FROM Affiliation AS x WHERE (x.Affiliation LIKE '%Poland%' )
```

T. Pankowski, DAFO

ACM or/and DEXA conferences

Query template (preselection of elements)

(Sub)trees constituting the query:

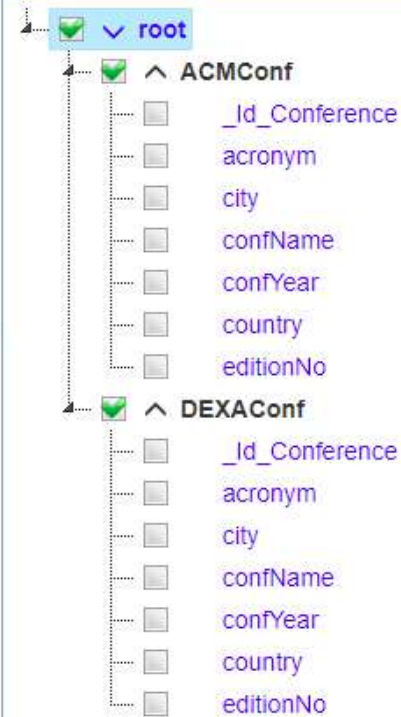
Roots of subtrees: ACMConf ▾ DEXAConf ▾ ----- ▾

Edges of (sub)trees:

Edge1: ----- ▾ → ----- ▾

ACCEPT

Faceted interface:



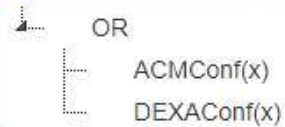
EXECUTE

FO & SQL FORMS

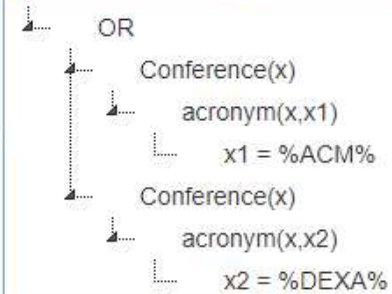
NESTED ANSWER

ACM or DEXA conferences

FO before rewriting:



FO after rewriting:



SQL:

```
SELECT DISTINCT x.Id AS Id FROM Conference AS x WHERE (x.Acronym LIKE '%ACM%' )
UNION
(SELECT DISTINCT x.Id AS Id FROM Conference AS x WHERE (x.Acronym LIKE '%DEXA%' ))
```

T. Pankowski, DAFO

ACM or DEXA conferences

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees: Person ▾ ----- ▾ ----- ▾

Edges of (sub)trees:

Edge1: Person ▾ → ACMConf ▾

Edge2: Person ▾ → DEXAConf ▾

Edge3: ----- ▾ → ----- ▾

ACCEPT

Faceted interface:



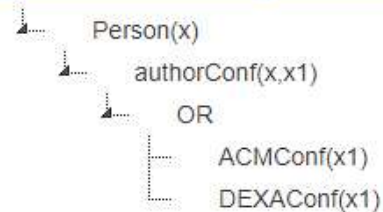
EXECUTE

FO & SQL FORMS

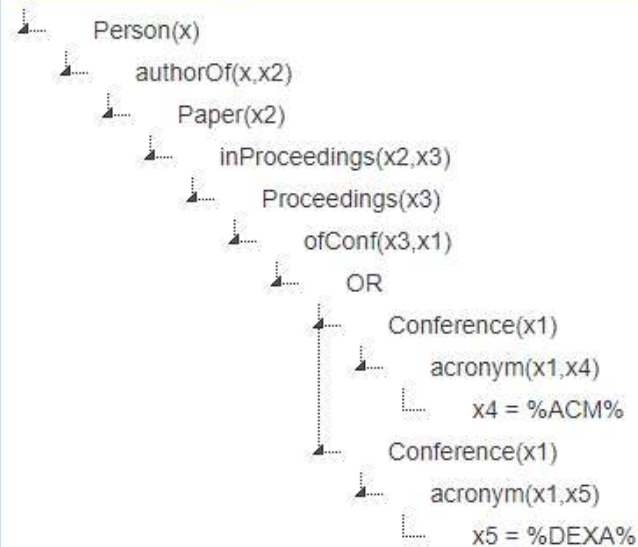
NESTED ANSWER

ACM or DEXA conferences

FO before rewriting:



FO after rewriting:



SQL:

```

SELECT DISTINCT x.PersonId AS Id FROM AuthorPaper AS x
JOIN(SELECT DISTINCT x2.Id AS Id FROM Paper AS x2
JOIN(SELECT DISTINCT x3.Id AS Id FROM Proceedings AS x3
JOIN(SELECT DISTINCT x1.Id AS Id FROM Conference AS x1 WHERE (x1.Acronym LIKE '%ACM%' )
UNION
(SELECT DISTINCT x1.Id AS Id FROM Conference AS x1 WHERE (x1.Acronym LIKE '%DEXA%' ))
)AS x1 ON x1.Id = x3.ConfId
)AS x3 ON x3.Id = x2.ProcId
)AS x2 ON x2.Id = x.PaperId
  
```

ACM and DEXA conferences

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees: Person ▾ ----- ▾ ----- ▾

Edges of (sub)trees:


Edge1: Person ▾ → ACMConf ▾

Edge2: Person ▾ → DEXAConf ▾

Edge3: ----- ▾ → ----- ▾

[ACCEPT](#)

Faceted interface:



```
graph TD; root[✓ root] --> Person[✓ ^ Person]; Person --> authorConf[✓ ^ authorConf]; authorConf --> ACMConf[✓ ^ ACMConf]; authorConf --> DEXAConf[✓ ^ DEXAConf];
```

[EXECUTE](#) [FO & SQL FORMS](#)

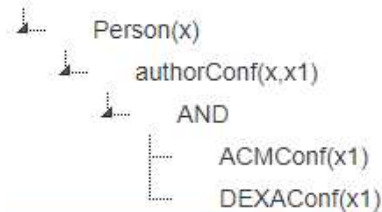
[NESTED ANSWER](#)

Answers

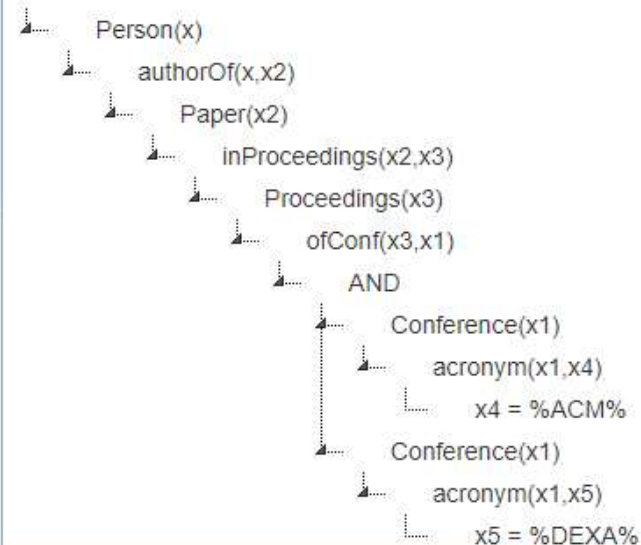
T. Pankowski, DAFO

ACM and DEXA conferences

FO before rewriting:



FO after rewriting:



SQL:

```

SELECT DISTINCT x.PersonId AS Id FROM AuthorPaper AS x
JOIN(SELECT DISTINCT x2.Id AS Id FROM Paper AS x2
JOIN(SELECT DISTINCT x3.Id AS Id FROM Proceedings AS x3
JOIN(SELECT DISTINCT x1_L.Id AS Id FROM(SELECT DISTINCT x1.Id AS Id FROM Conference AS x1 WHERE (x1.Acronym LIKE '%ACM%')) AS x1_L
JOIN(SELECT DISTINCT x1.Id AS Id FROM Conference AS x1 WHERE (x1.Acronym LIKE '%DEXA%')) AS x1 ON x1.Id = x1_L.Id
AS x1 ON x1.Id = x3.ConfId
AS x3 ON x3.Id = x2.ProcId
AS x2 ON x2.Id = x.PaperId

```

ACM and DEXA: poprawnie. Konieczne klonowanie

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees:

Edges of (sub)trees:

Edge1: →

Edge2: →

Edge3: →

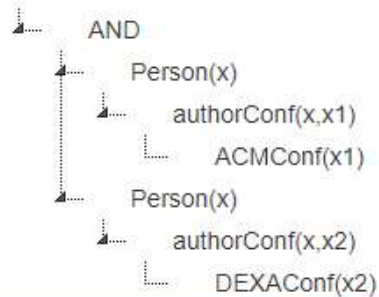
Faceted interface:

Answers

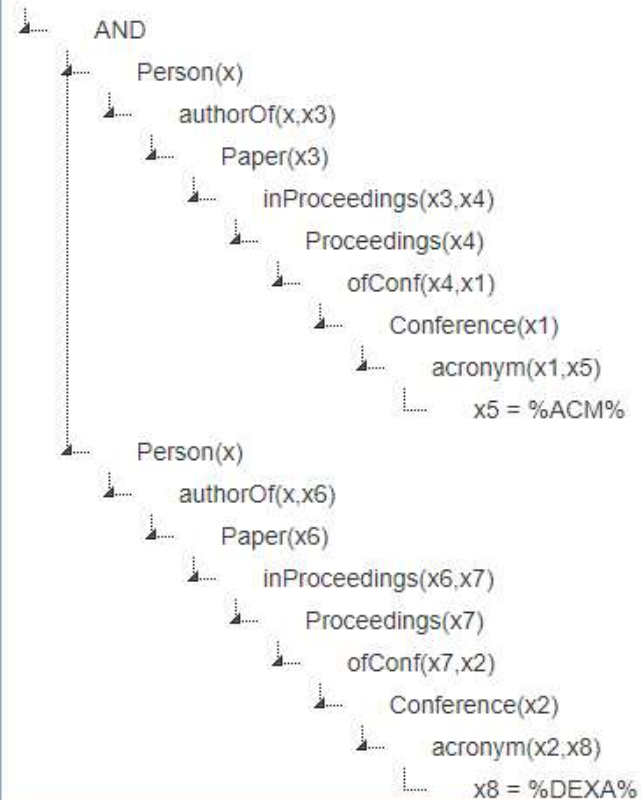
1. Serge Abiteboul; INRIA, France
2. Foto N. Afrati; University of Athens, Greece
3. Marcelo Arenas; Pontificia Universidad Catolica de Chile, Chile
4. Angela Bonifati; University of Lyon, France
5. Jan Chomicki; University at Buffalo, USA
6. Alin Deutsch; University of California, San Diego, USA
7. Georg Gottlob; University of Oxford, UK; TU Vienna, Austria
8. Maurizio Lenzerini; Sapienza University of Rome, Italy
9. Leonid Libkin; University of Edinburgh, UK
10. Tova Milo; Tel Aviv University, Israel
11. Martin Theobald; University of Ulm, Germany; University of Luxembourg

ACM and DEXA: poprawnie. Konieczne klonowanie

FO before rewriting:



FO after rewriting:



ACM and DEXA: poprawnie. Konieczne klonowanie

SQL:

```
SELECT DISTINCT x_L.Id AS Id FROM (SELECT DISTINCT x.PersonId AS Id FROM AuthorPaper AS x
JOIN (SELECT DISTINCT x3.Id AS Id FROM Paper AS x3
JOIN (SELECT DISTINCT x4.Id AS Id FROM Proceedings AS x4
JOIN (SELECT DISTINCT x1.Id AS Id FROM Conference AS x1 WHERE (x1.Acronym LIKE '%ACM%')
) AS x1 ON x1.Id = x4.ConfId
) AS x4 ON x4.Id = x3.ProcId
) AS x3 ON x3.Id = x.PaperId) AS x_L
JOIN (SELECT DISTINCT x.PersonId AS Id FROM AuthorPaper AS x
JOIN (SELECT DISTINCT x6.Id AS Id FROM Paper AS x6
JOIN (SELECT DISTINCT x7.Id AS Id FROM Proceedings AS x7
JOIN (SELECT DISTINCT x2.Id AS Id FROM Conference AS x2 WHERE (x2.Acronym LIKE '%DEXA%')
) AS x2 ON x2.Id = x7.ConfId
) AS x7 ON x7.Id = x6.ProcId
) AS x6 ON x6.Id = x.PaperId
) AS x ON x.Id = x_L.Id
```

T. Pankowski, DAFO

Telewizory (395 modeli)

Filtrowanie

Produkt w promocji

☐ Oferta OUTLET!

Cena

od do zł

Producenci

- ☐ Philips
☐ LG
☐ Samsung
☐ Sony

[Pokaż więcej \(10\)](#) ▼

rozdzielczość nominalna

- ☐ 3840 x 2160 (4K Ultra HD)
☐ 1920 x 1080 (Full HD)
☐ 1366 x 768 (HD Ready)
☐ 1024 x 600

[Pokaż więcej \(1\)](#) ▼

technologia wykonania

- ☐ LCD - LED
☐ OLED
☐ DLED
☐ ELED

przekątna ekranu

- ☐ 55 cali (138 cm)
☐ 65 cali (163 cm)
☐ 43 cale (108 cm)
☐ 32 cale (80 cm)

[Pokaż więcej \(20\)](#) ▼

zakrzywiony ekran

- ☐ Nie
☐ Tak

HDR

- ☐ Tak

Smart TV

- ☐ Tak

Ekran

technologia 3D

- ☐ Brak

współczynnik płynności obrazu

od do

Dźwięk

ilość kanałów

- ☐ 2.0
☐ 2.1
☐ 4.2
☐ 2.2

Zapytania fasetowe: faceted search, faceted queries

<https://www.komputronik.pl/category/5431/telewizory.html>

Dane fizyczne

kolor obudowy

- ☐ Czarny
☐ Srebrny
☐ Biały
☐ Szary

[Pokaż więcej \(3\)](#) ▼

szerokość

od do mm

wysokość

od do mm

głębokość

od do mm

Fasetowy widok ontologii (nad ontologią)

- **Faseta** – to para predykat-wartość
- **Fasetowy widok** – zbiór hierarchicznie zorganizowanych faset.
- **Interfejs fasetowy** – drzewo skierowane stanowiące hierarchiczne rozpięcie fragmentu ontologii wraz z zestawem operacji wykonywanych na tym drzewie.
- Wierzchołki interfejsu fasetowego reprezentują predykaty (unarne lub binarne) lub wartości.
- **Zapytanie fasetowe** – fragment interfejsu fasetowego zdefiniowany przez użytkownika w wyniku wykonywania operacji na tym interfejsie.

Definiowanie hierarchicznego widoku

Query template (preselection of elements)

(Sub)trees constituting the query:

Roots of subtrees: PUTAuthor ▾ ----- ▾

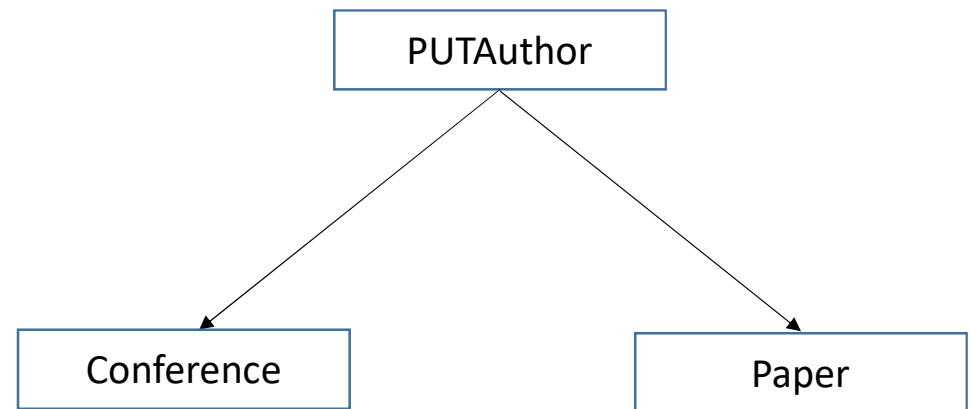
Edges of (sub)trees:

Edge1: PUTAuthor ▾ → Conference ▾

Edge2: PUTAuthor ▾ → Paper ▾

Edge3: ----- ▾ → ----- ▾

ACCEPT



Użytkownik określa szkielet planowanego zapytania podając drzewo (lub las drzew):

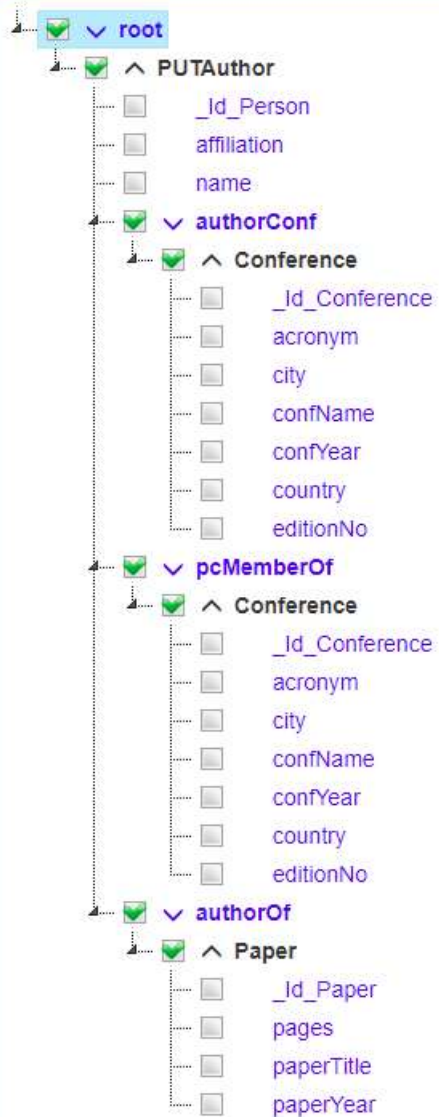
- korzeń drzewa określa typ odpowiedzi, a
- pozostałe klasy (hierarchicznie uporządkowane) będą wykorzystywane zdefiniowania do ograniczeń.

Przykład:

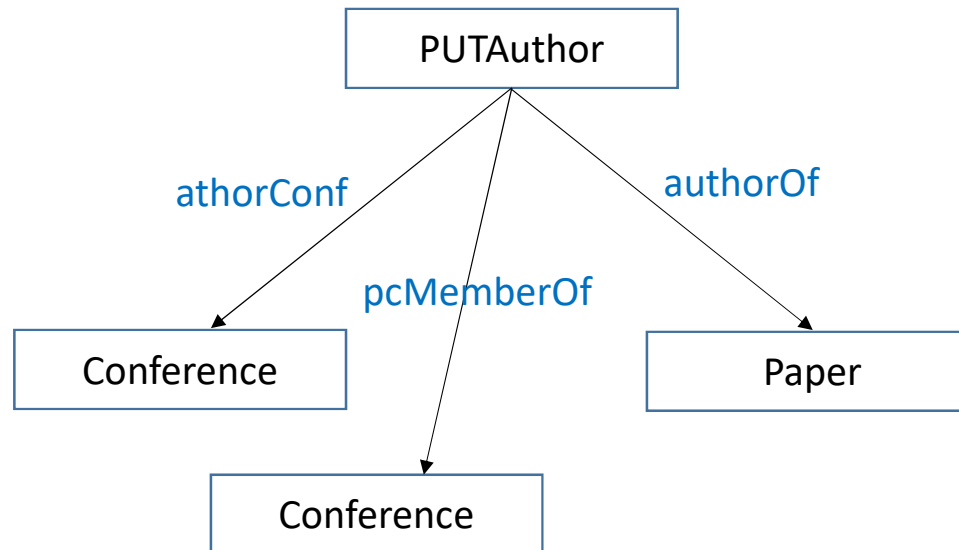
Pytanie będzie dotyczyło autorów z PUT, wybranych ze względu na konferencje i napisane artykuły.

Wskazane drzewo stanowi punkt wyjścia do wygenerowania interfejsu fasetowego.

Faceted interface:



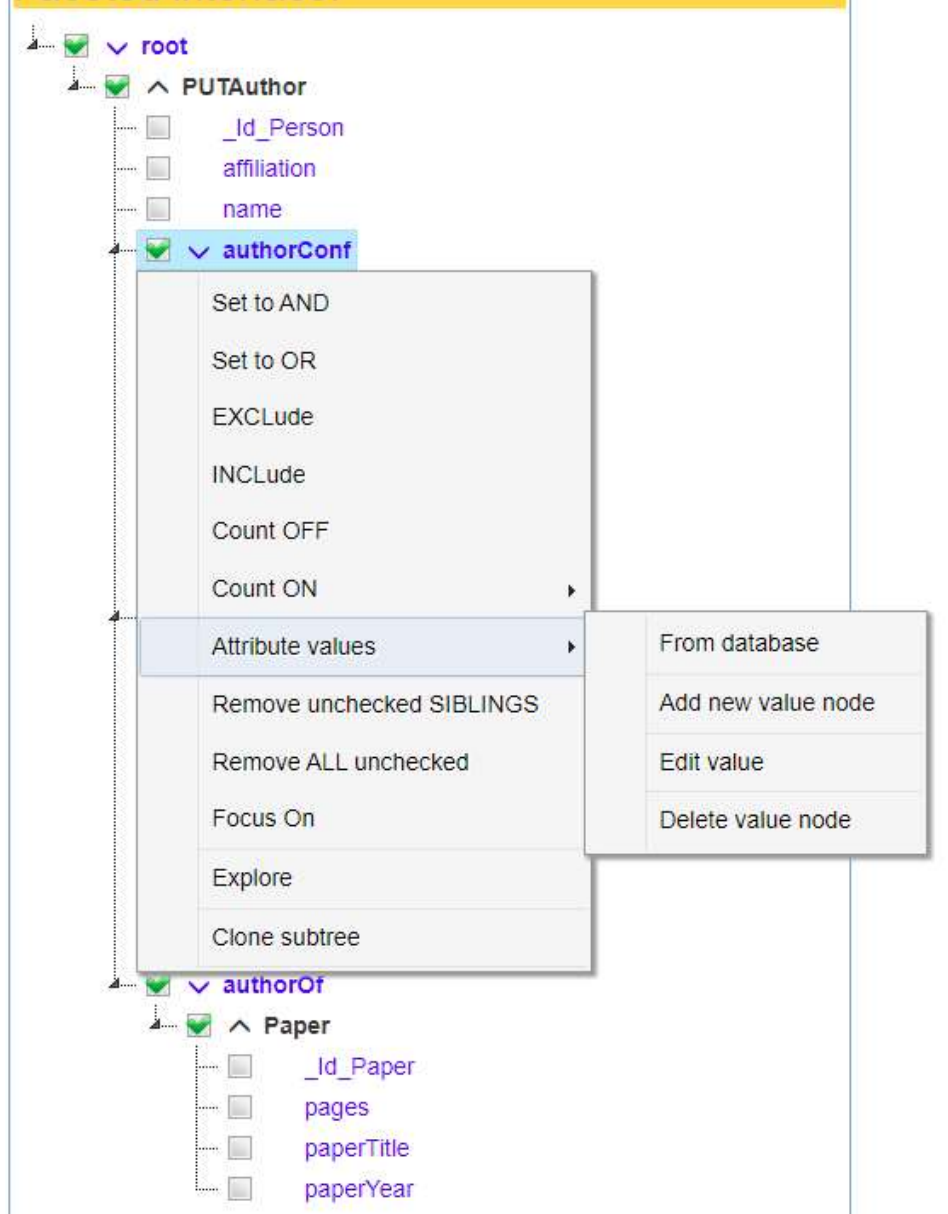
Interfejs fasetowy



Wygenerowany interfejs fasetowy zawiera:

- wszystkie wskazane klasy z uwzględnieniem ich hierarchicznego uporządkowania,
- wszystkie predykaty binarne łączące te klasy (mogą to być złożenia lub odwrotności predykatów ekstensjonalnych),
- wszystkie atrybuty (*data properties*) wskazanych klas.

Faceted interface:

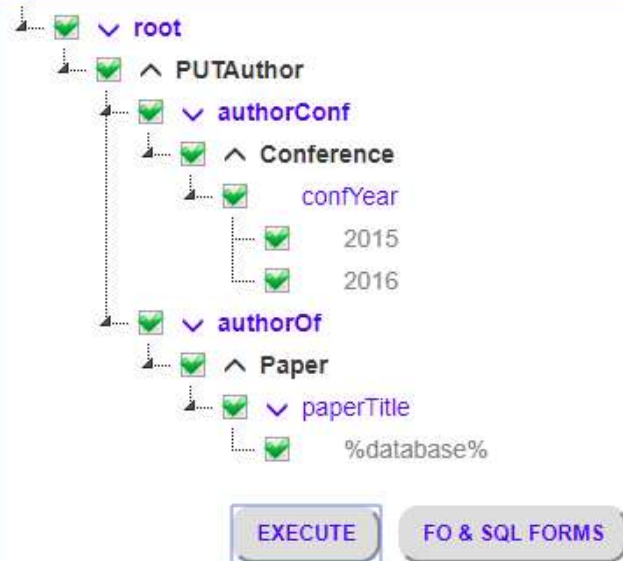


Operacje na interfejsie fasetowym

Meny kontekstowe (prawy przycisk myszy) zawiera wykaz operacji, które można wykonywać na interfejsie.

Zapytanie fasetowe

Faceted interface:

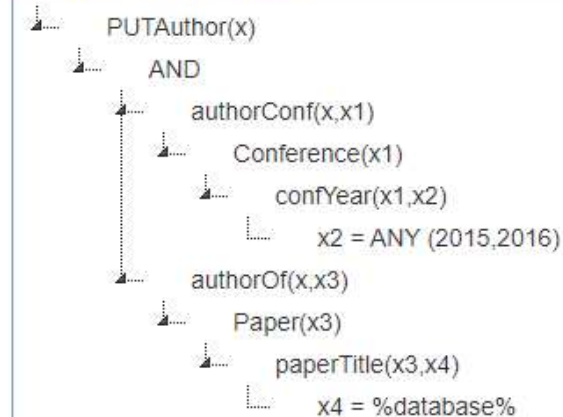


Answers

1. Czesław Jędrzejek; PUT, Poland
2. Tadeusz Morzy; PUT, Poland
3. Tadeusz Pankowski; PUT, Poland
4. Robert Wrembel; PUT, Poland

Semantyka zapytania:

FO before rewriting:



PUTAuthor, który:

1. miał jakiś artykuł na jakiejś konferencji w roku 2015 lub 2016, i
2. jest autorem jakiegoś artykułu z baz danych

NIE

Chcemy, aby był to artykuł z baz danych na którejś z tych konferencji !!

Zapytanie fasetowe

Query template (preselection of ele

(Sub)trees constituting the query:

Roots of subtrees: PUTAuthor ▾ ----- ▾

Edges of (sub)trees:

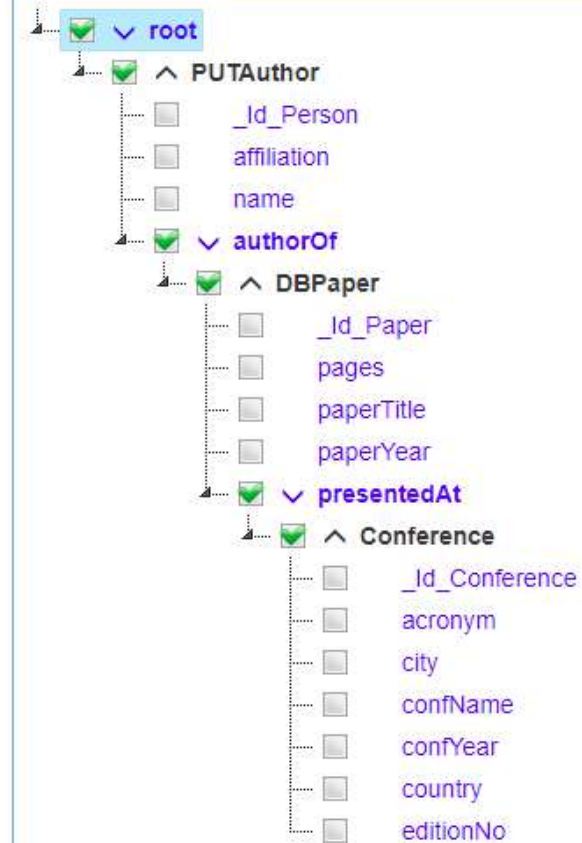
Edge1: PUTAuthor ▾ → DBPaper ▾

Edge2: DBPaper ▾ → Conference ▾

Edge3: ----- ▾ → ----- ▾

ACCEPT

Faceted interface:

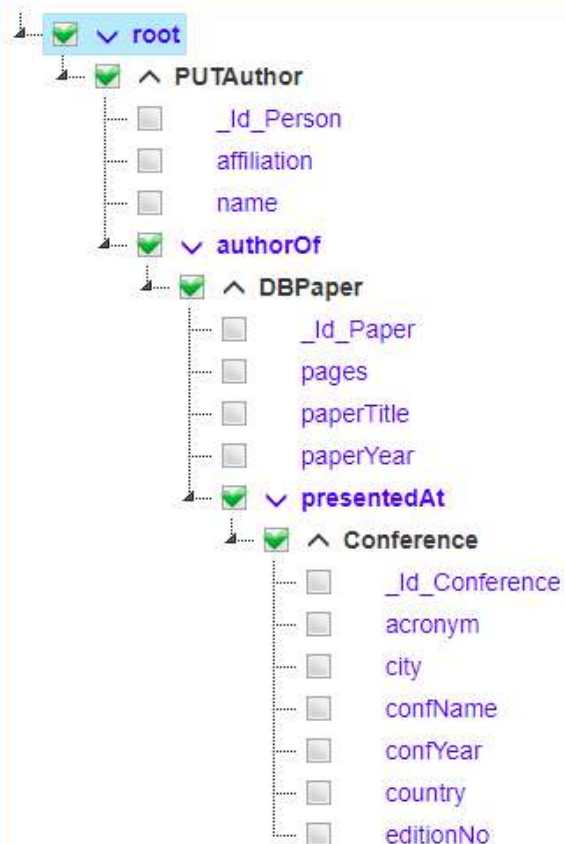


EXECUTE

FO & SQL FORMS

Zapytanie fasetowe

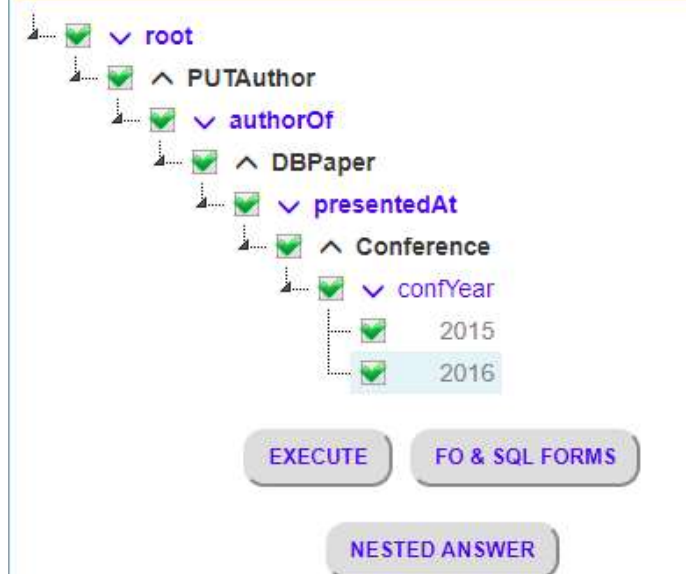
Faceted interface:



EXECUTE

FO & SQL FORMS

Faceted interface:



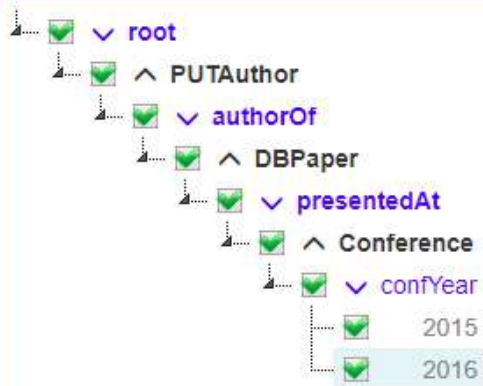
EXECUTE

FO & SQL FORMS

NESTED ANSWER

Zapytanie fasetowe

Faceted interface:

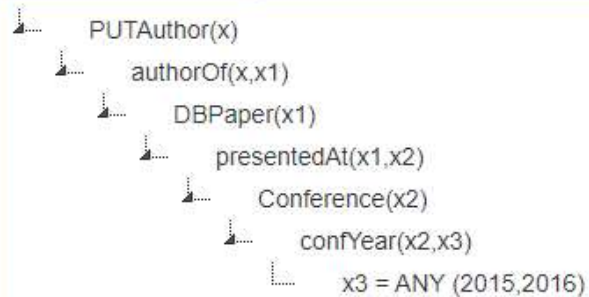


EXECUTE

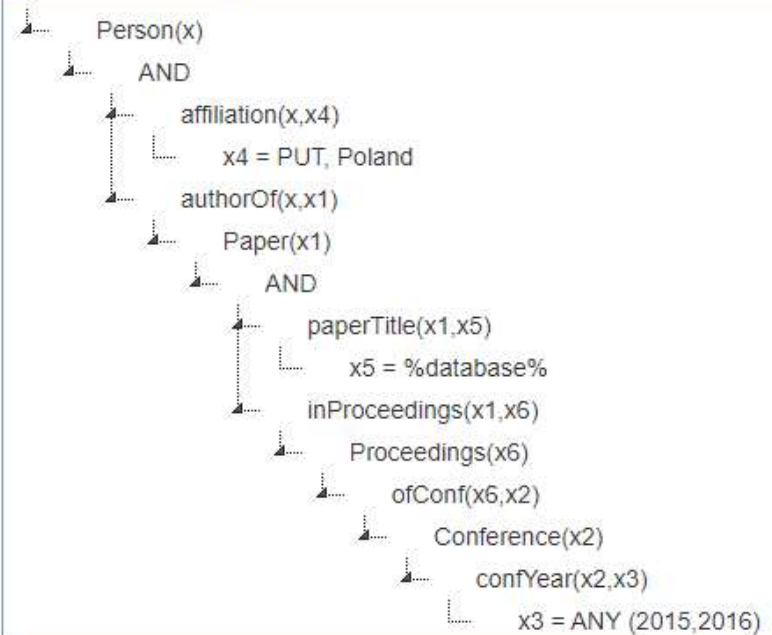
FO & SQL FORMS

NESTED ANSWER

FO before rewriting:

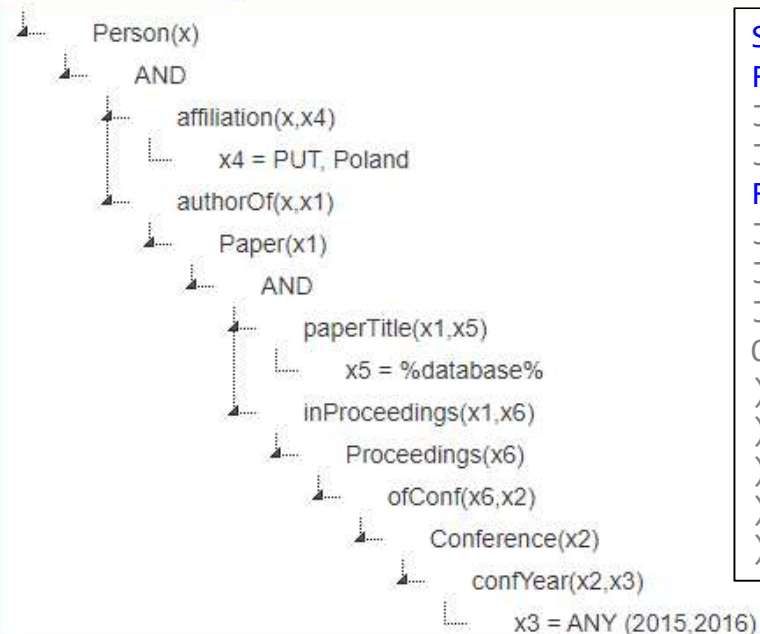


FO after rewriting:



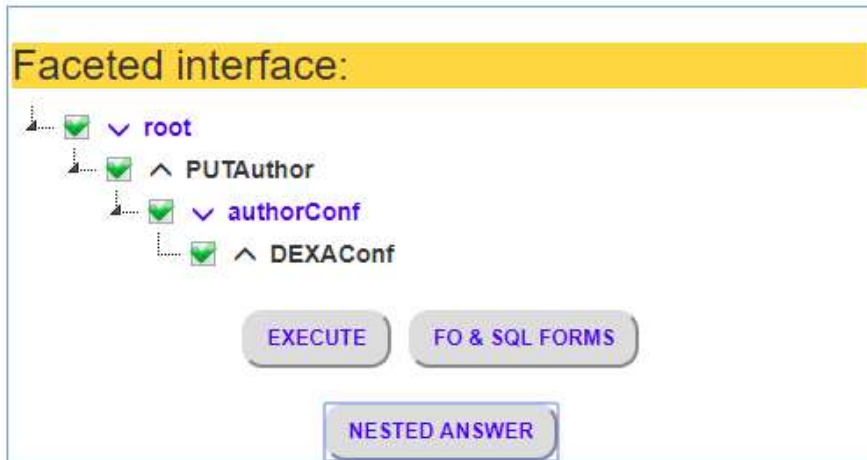
Zapytanie fasetowe: postać FOL i SQL

FO after rewriting:



```
SELECT DISTINCT x_L.Id AS Id FROM(SELECT DISTINCT x.PersonId AS Id
FROM Affiliation AS x WHERE (x.Affiliation = 'PUT, Poland' ))AS x_L
JOIN(SELECT DISTINCT x.PersonId AS Id FROM AuthorPaper AS x
JOIN(SELECT DISTINCT x1_L.Id AS Id FROM(SELECT DISTINCT x1.Id AS Id
FROM Paper AS x1 WHERE (x1.Title LIKE '%database%' ))AS x1_L
JOIN(SELECT DISTINCT x1.Id AS Id FROM Paper AS x1
JOIN(SELECT DISTINCT x6.Id AS Id FROM Proceedings AS x6
JOIN(SELECT DISTINCT x2.Id AS Id FROM Conference AS x2 WHERE (x2.Year = '2015'
OR x2.Year = '2016'))
)AS x2 ON x2.Id = x6.ConfId
)AS x6 ON x6.Id = x1.ProcId
)AS x1 ON x1.Id = x1_L.Id
)AS x1 ON x1.Id = x.PaperId
)AS x ON x.Id = x_L.Id
```

Zapytania zagnieżdżone



Nested answers (expand/collapse items):

- 1. Tadeusz Morzy; PUT, Poland
 - 1. DEXA, 2015, 26. DEXA 2015: Valencia, Spain, Database and Expert Systems Applications (DEXA), 26 , Spain, Valencia
 - 2. DEXA Workshop, 2001, 12. DEXA Workshop 2001: Munich, Germany, DEXA Workshops, 12 , Germany, Munich
 - 3. DEXA Workshop, 1999, 10. DEXA Workshop 1999: Florence, Italy, DEXA Workshops, 10 , Italy, Florence
 - 4. DEXA, 1991, 2. DEXA 1991: Berlin, Germany, Database and Expert Systems Applications (DEXA), 2 , Germany, Berlin
 - 5. DEXA, 1990, 1. DEXA 1990: Vienna, Austria, Database and Expert Systems Applications (DEXA), 1 , Austria, Vienna
- 2. Tadeusz Pankowski; PUT, Poland
- 3. Robert Wrembel; PUT, Poland

T. Pankowski, DAFO

Implementacja

Ontologia

Złożenie (Chain)

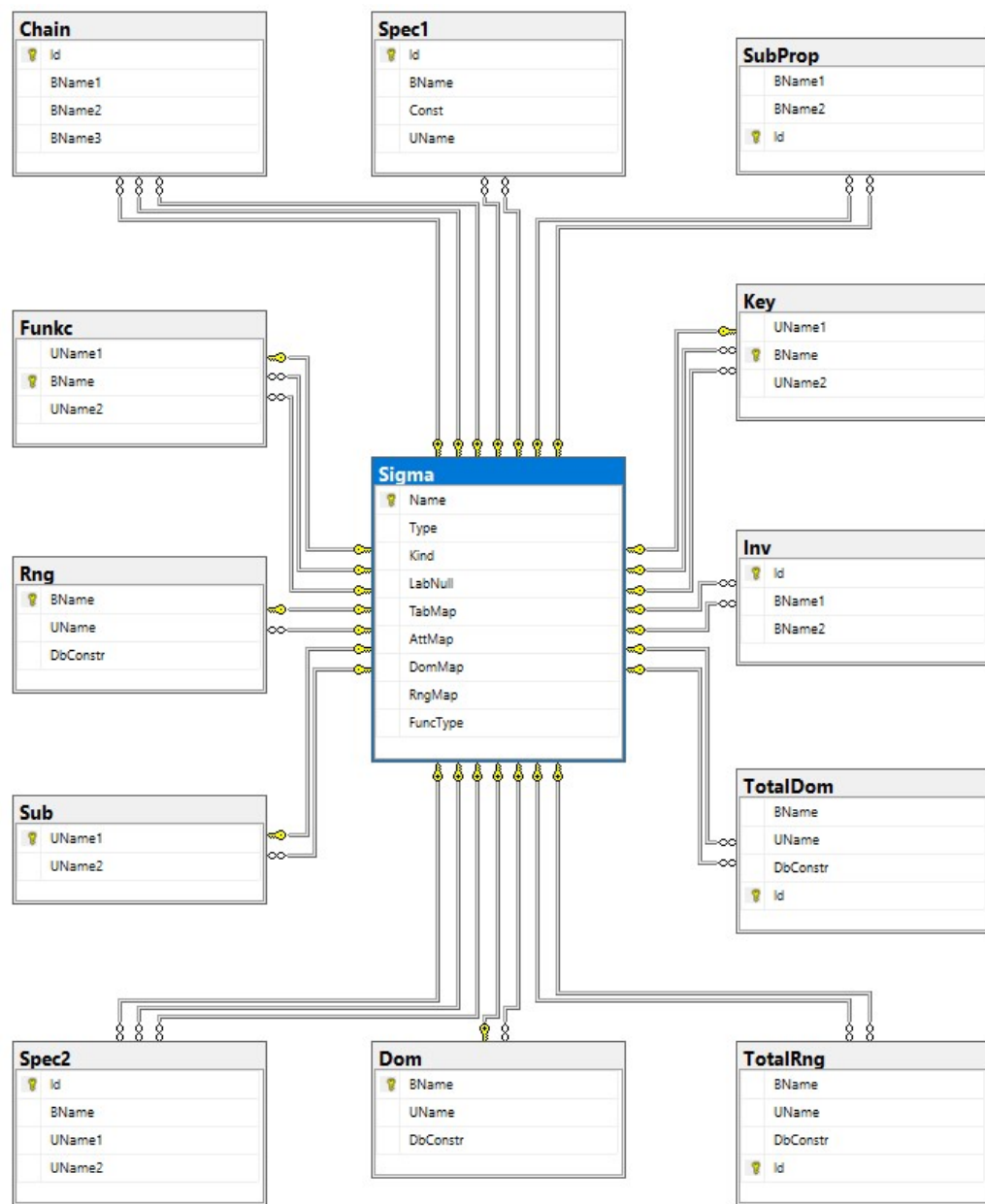
Id	BName1	BName2	BName3
1	inProceedings	ofConf	presentedAt
2	confProceedings	includesPaper	presentedPaper
3	inProceedings	proceedingsYear	paperYear
7	authorOf	presentedAt	authorConf
8	presentedPaper	writtenBy	confAuthor

Przeciwdziedzina (Rng)

BName	UName	DbConstr
<u>Id_Person</u>	Integer	T
affiliation	String	N
authorConf	Conference	N
authorOf	Paper	N
name	String	N
pcMemberOf	Conference	T
personKey	String	N

Specjalizacja sterowana klasą (Spec2)

Id	BName	UName1	UName2
1	authorOf	Paper	Author
2	presentedAt	ACMConf	ACMPaper
3	presentedAt	DEXAConf	DEXAPaper
4	authorOf	ACMPaper	ACMAuthor



Specjalizacja sterowana wartością (Spec1)

Id	BName	Const	UName
1	acronym	%ACM%	ACMConf
2	acronym	%DEXA%	DEXAConf
3	publisher	%Springer%	SpringerProceedings
4	acronym	%KES%	KESConf
5	acronym	%TPDL%	TPDLConf
6	country	USA	USACConf
7	affiliation	PUT, Poland	PUTAuthor
9	paperTitle	%database%	DBPaper
10	paperTitle	%ontolog%	ONTOPaper
11	paperTitle	%query%	QUERYPaper

Inwersja (Inv)

Id	BName1	BName2
1	authorOf	writtenBy
2	inProceedings	includesPaper
3	ofConf	confProceedings
4	presentedAt	presentedPaper
7	confAuthor	authorConf
11	pcMemberOf	confPCMember

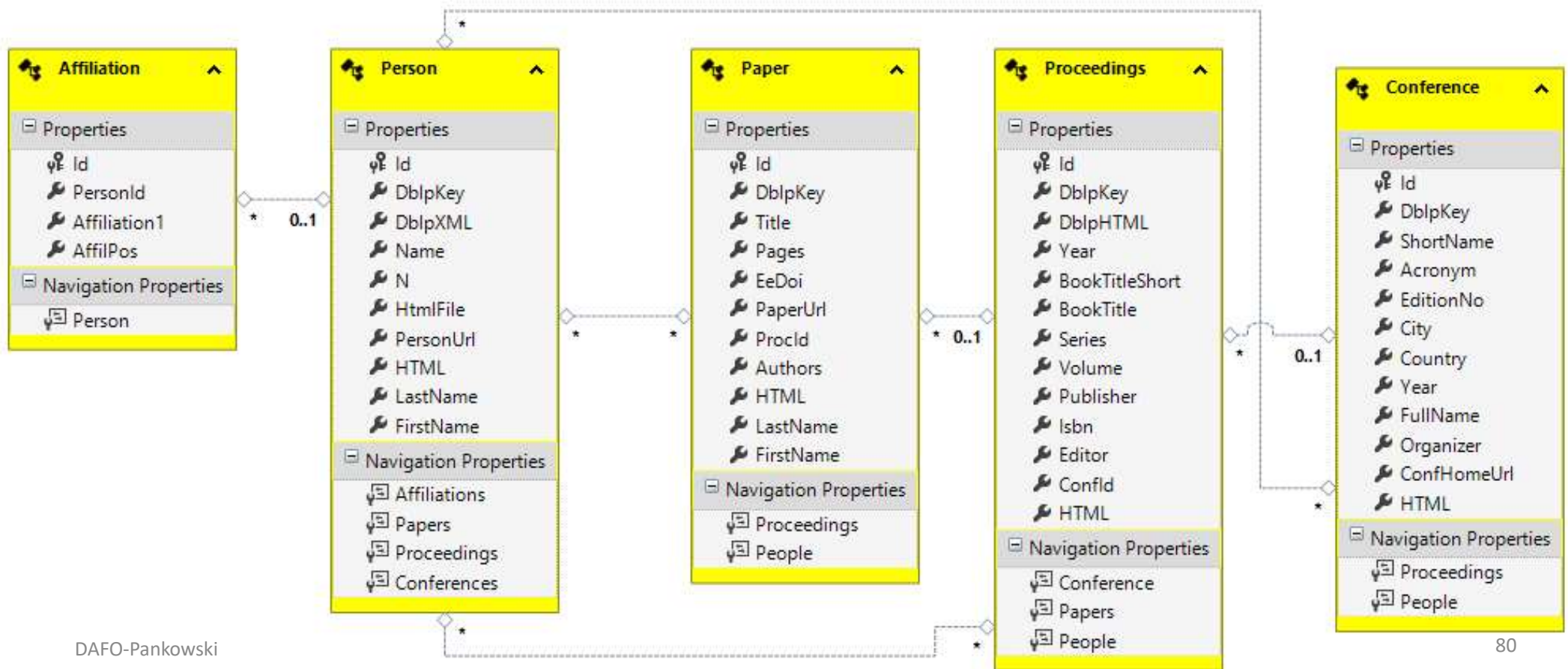
Dziedzina (Dom)

BName	UName	DbConstr
<u>Id_Person</u>	Person	T
affiliation	Person	N
authorConf	Person	N
authorOf	Person	N
name	Person	N
pcMemberOf	Person	T
personKey	Person	N

Database in SQL Server

Baza danych: DAFO_DBLP

Diagram bazy danych (w VS 2017)



Mapowanie Table Sigma in BIB_ONTO

Name	Type	Kind	TabMap	DomMap	RngMap	FuncType
_Id_Conference	B	E	Conference	Id	Id	DF
_Id_Paper	B	E	Paper	Id	Id	DF
_Id_Person	B	E	Person	Id	Id	DF
_Id_Proceedings	B	E	Proceedings	Id	Id	DF
acronym	B	E	Conference	Id	Acronym	DF
affiliation	B	E	Affiliation	PersonId	Affiliation	DM
authorOf	B	E	AuthorPaper	PersonId	PaperId	OM
city	B	E	Conference	Id	City	DF
Conference	U	E	Conference	Id	NULL	NULL
confKey	B	E	Conference	Id	DblpKey	DF
confName	B	E	Conference	Id	ShortName	DF
confYear	B	E	Conference	Id	Year	DF
country	B	E	Conference	Id	Country	DF
editionNo	B	E	Conference	Id	EditionNo	DF
editor	B	E	Proceedings	Id	Editor	DF
inProceedings	B	E	Paper	Id	ProcId	OF
name	B	E	Person	Id	Name	DF
ofConf	B	E	Proceedings	Id	ConfId	OF
pages	B	E	Paper	Id	Pages	DF
Paper	U	E	Paper	Id	NULL	NULL
paperKey	B	E	Paper	Id	DblpKey	DF
paperTitle	B	E	Paper	Id	Title	DF
pcMemberOf	B	E	PCMember	PersonId	ConfId	OM
Person	U	E	Person	Id	NULL	NULL
personKey	B	E	Person	Id	DblpKey	DF
Proceedings	U	E	Proceedings	Id	NULL	NULL
proceedingsKey	B	E	Proceedings	Id	DblpKey	DF
proceedingsTitle	B	E	Proceedings	Id	BookTitle...	DF
proceedingsYear	B	E	Proceedings	Id	Year	DF
publisher	B	E	Proceedings	Id	Publisher	DF
series	B	E	Proceedings	Id	Series	DF
volume	B	E	Proceedings	Id	Volume	DF