SBQL

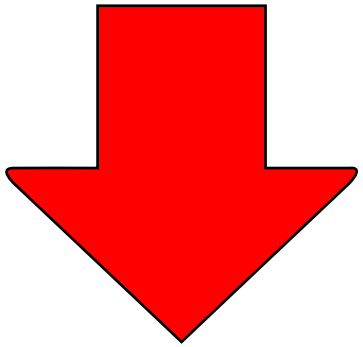
język zapytań dla obiektowych baz danych

Kamil Adamczyk

Uniwersytet Warszawski
20.IV.2009

Spis treści

Stack-Based Architecture (SBA)
Stack-Based Query Language (SBQL)
(prof. Kazimierz Subieta (PJWSTK) [1])



(?) “ODMG 4.0”

Abstract Store Model (AS)
Abstract Object Query Language (AOQL)
OMG - "Next-Generation Object Database Standardization" [3]

Obiektowe bazy danych

Obiektowa baza danych to zbiór obiektów, których zachowanie się, stan oraz związki są określone zgodnie z obiektowym modelem danych. [wikipedia]

- zapewniają tradycyjną funkcjonalność baz danych
- udostępniają dane w postaci obiektowej, czyli takiej samej w jakiej są przechowywane w programach
- znika konieczność odwzorowania pomiędzy modelem obiektowym a modelem relacyjnym

Model danych

Relacyjna baza danych (przypomnienie)

Pojęcia pierwotne:

A – zbiór nazw atrybutów

D – zbiór wartości atomowych (napisy, liczby, daty, wartości logiczne)

E – zbiór typów atomowych (integer, float, string, boolean, date)

K – zbiór nazw relacji (potem będzie to zbiór klas)

Schemat tabel (relacji)

kol: $K \rightarrow Pfin(A)$ (nazwie relacji przyporządkowujemy skończone zbiory nazw kolumn)

typ: $K \times A \rightarrow E$ (nazwie relacji i nazwie atrybutu przyporządkowujemy typ elementarny)

funkcja typ jest określona dla takich k i a należących od K i A, że a należy do kol(k)

Schemat bazy relacyjnej

SchRel = (K, kol, typ)

Obiektowy model danych

Pojęcia pierwotne

- **A** – zbiór nazw atrybutów
- **O** – zbiór identyfikatorów obiektów (OID)
- **D** – wartości atomowe (napisy, liczby, wartości logiczne, daty)
- **K** – zbiór nazw klas i **IsA** częściowy porządek na K
- **E** – zbiór typów atomowych (integer, float, string, boolean, date)

Wartości złożone W to najmniejszy zbiór o następujących właściwościach:

1. $D \subseteq W$ (skalar)
2. $O \subseteq W$ (referencja)
3. Jeśli $A1, A2, ..., An \in A$ $(Ai \neq Aj$ dla $i \neq j)$ oraz $w1, w2, ..., wn \in W$,
   to $[A1:w1, A2:w2, ..., An:wn] \in W$ (krotka)
4. Jeśli $w1, w2, ...,wm \in W$ $(wi \neq wj$ dla $i \neq j)$, to $\{w1, w2, ...,wm\} \in W$ (zbiór)

Obiekt

Para (o, w) przy czym $o \in O$, $w \in W$.

Obiektowy model danych

Typy T to najmniejszy zbiór o następujących właściwościach:

1. $E \subseteq T$ (typ elementarny)
2. $K \subseteq T$ (typ referencyjny)
3. Jeśli $t \in T$, to $\{t\} \in T$ (typ zbiorowy)
4. Jeśli $A1, A2, ..., An \in A$ ($Ai \neq Aj$ dla $i \neq j$) oraz $T1, T2, ..., Tn \in T$, to
   $[A1:T1, A2:T2, ..., An:Tn] \in T$ (typ krotkowy)

Hierarchia typów

≤ częściowy porządek na T (podtyp ≤ nadtyp)

≤ jest najmniejszym częściowym porządkiem o następujących właściwościach:

1. Jeśli K IsA L, to $K \leq L$ [Pracownik IsA Osoba, więc Pracownik ≤ Osoba]
2. Jeśli $t = [A1:T1, A2:T2, ..., An:Tn] \in T$ i $u = [B1:U1, B2:U2, ..., Bm:Um] \in T$ oraz $n \leq m$
   i dla każdego $i = 1, 2, ..., n$ istnieje j takie, że $Ai = Bj$ i $Uj \leq Ti$, to $u \leq t$.
3. Jeśli $t \leq u$, to $\{t\} \leq \{u\}$.

Schemat struktury

SchStruct = (K, IsA, typ)

typ: $K \rightarrow T$ musi spełniać warunek: $K \text{ IsA } L \Rightarrow typ(K) \leq typ(L)$

Języki zapytań

OQL - Object Query Language

```
SELECT struct( w: x.wiek, p: x.płeć )
FROM (SELECT y FROM Pracownicy AS y
            WHERE y.gr_zawodowa = 10) AS x
WHERE x.nazwisko = "Nowak"
```

```
SELECT d.adres
FROM Osoby AS x, x.dzieci AS d
WHERE x.adres.ulica = "Chrobrego"
      AND count(x.dzieci) >= 2
      AND d.adres.miasto != x.adres.miasto
```

Jezyki zapytań

- **XML-QL**
- **XSLT**

- **OCL** - Object Constraint Language
- **ODL** - Object Definifion Language

- **AOQL** (**SBQL**)

Dostępne produkty

1. POET
2. ***Versant***
3. Objectivity/DB
4. Gemstone
5. **ObjectStore**
6. Jasmine

Inne:

- opcja obiektowa w Oracle
- loxim, ODRA, jloxim

SBQL - główne cechy

- podejście stosowe w którym zakładamy, że języki zapytań są szczególnym przypadkiem języków programowania
- zapytania pełnią rolę wyrażeń znanych z języków programowania
- posiada mechanizm mocnej kontroli typów
- został on wymieniony w publikacji OMG (Object Management Group) jako podstawa do dalszych prac nad standaryzacją obiektowych baz danych [3]

Model danych

Stack-Based Architecture (SBA)

- **as0**: obejmuje dowolnie powiązane hierarchiczne struktury danych; nie obejmuje klas, dziedziczenia, interfejsu i hermetyzacji.

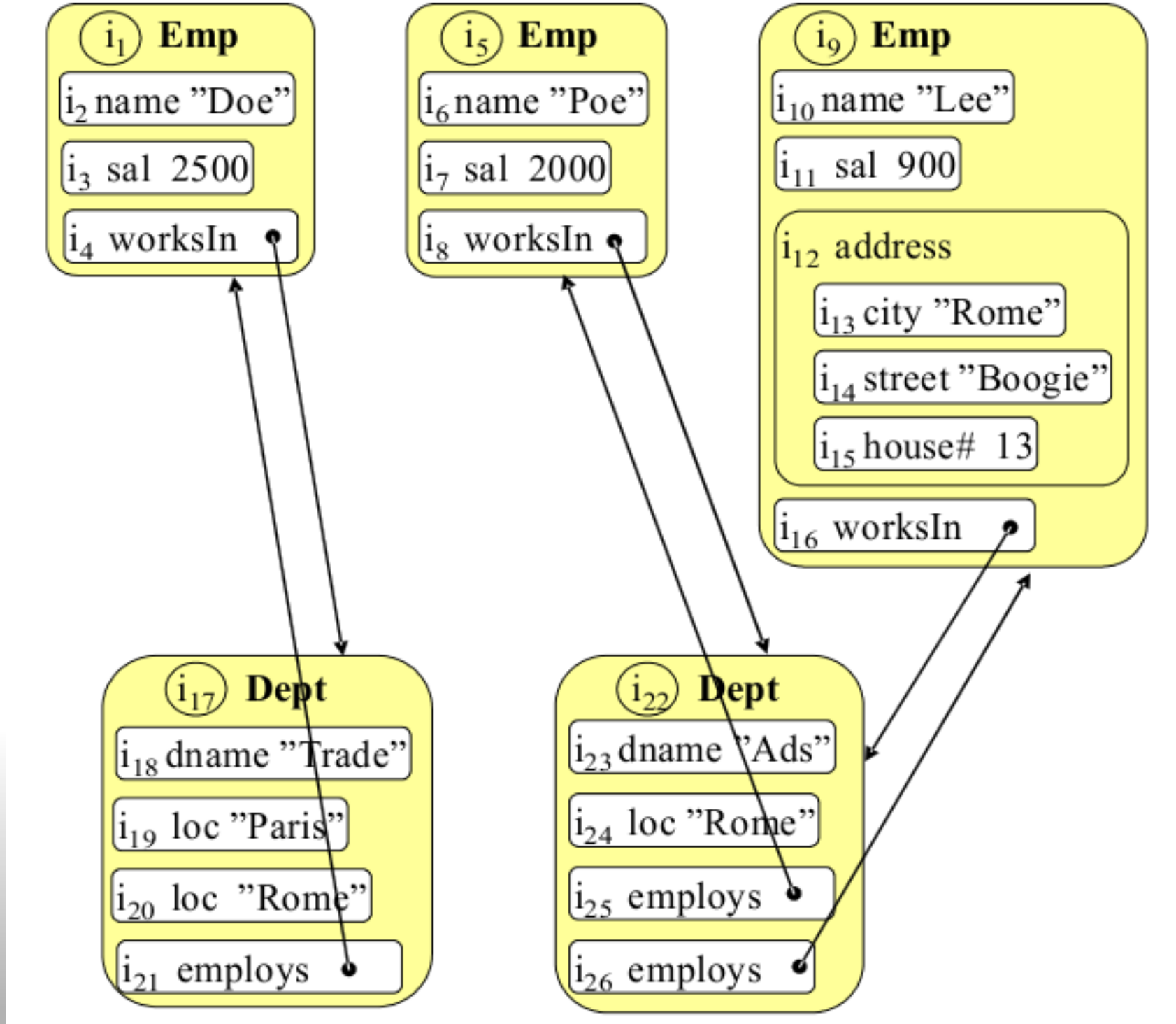
  Model as0 pozwala wyjaśnić semantykę relacyjnych języków zapytań (szczególnie SQL), przykrywa koncepcję zagnieżdżonych relacji, struktury implikowane przez XML i dane określane jako pół-strukturalne.

SBQL - model danych

Podstawowe pojęcia

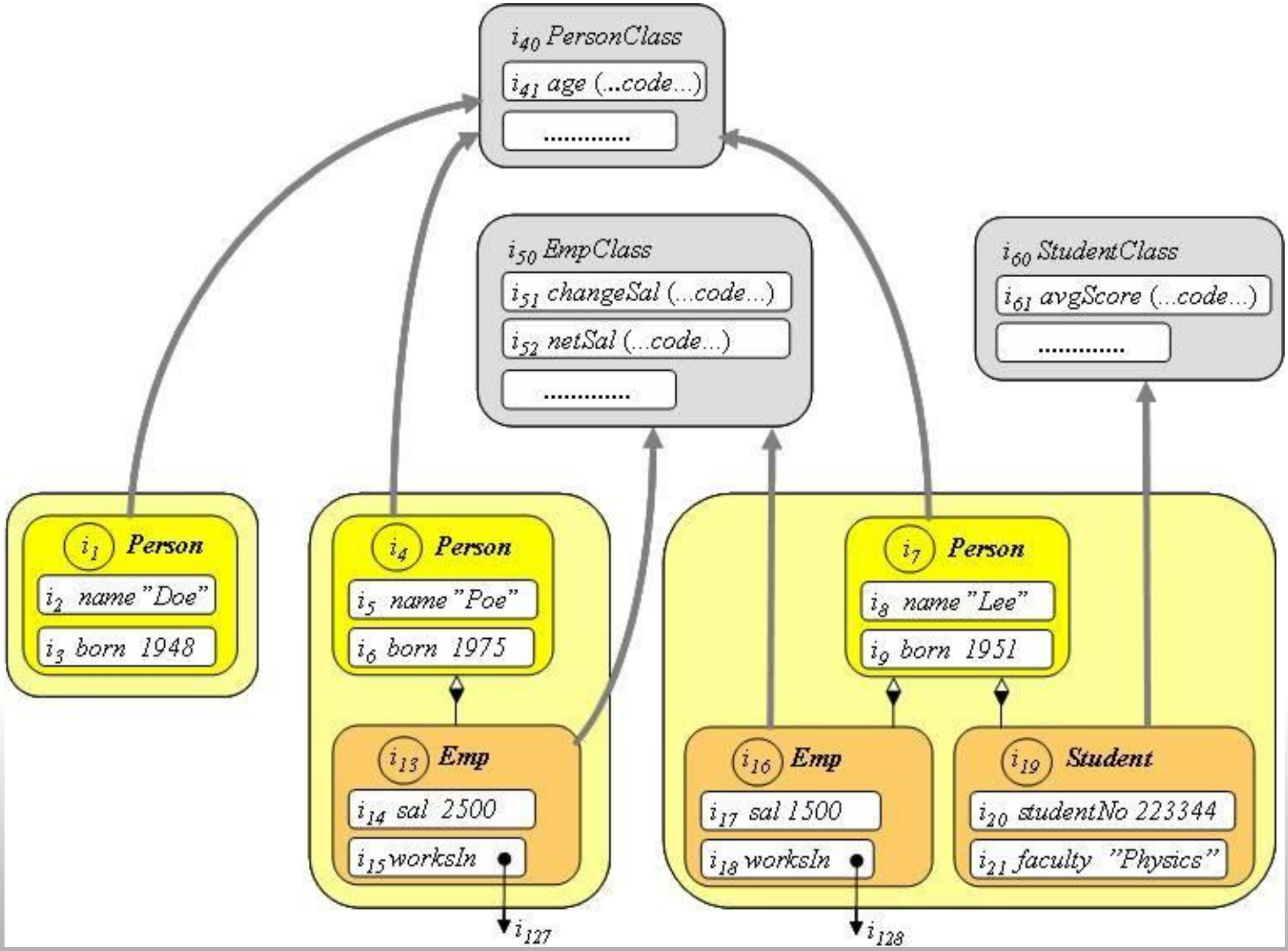
- **I** - zbiór identyfikatorów (i, i1, i2, ... - oznaczenia identyfikatorów)
- **N** - zbiór nazw (n, n1, n2, ... - oznaczenia nazw)
- **V** - zbiór wartości atomowych (v, v1, v2, ... - oznaczenia wartości)

- **Obiekt atomowy**: trójka <i, n, v>.
- **Obiekt pointerowy**: trójka <i1, n, i2>. Obiekt jest identyfikowany przez i1, natomiast i2 jest pointerem (referencją) do innego obiektu.
- **Obiekt złożony**: trójka <i, n, T>, gdzie T jest zbiorem dowolnych obiektów. Powyższa reguła umożliwia rekurencyjne tworzenie obiektów o nieograniczonej złożoności i o nieograniczonej liczbie poziomów hierarchii.


i1 Emp
i2 name "Doe"
i3 sal 2500
i4 worksIn
i5 Emp
i6 name "Poe"
i7 sal 2000
i8 worksIn
i9 Emp
i10 name "Lee"
i11 sal 900
i12 address
i13 city "Rome"
i14 street "Boogie"
i15 house# 13
i16 worksIn
i17 Dept
i18 dname "Trade"
i19 loc "Paris"
i20 loc "Rome"
i21 employs
i22 Dept
i23 dname "Ads"
i24 loc "Rome"
i25 employs
i26 employs

Model danych

- **as1**: uzupełnia as0 o pojęcia klasy, dziedziczenia i wielodziedziczenia w najczęściej spotykanym rozumieniu; nie obejmuje hermetyzacji i interfejsu.
- **as2**: uzupełnia model as1 oraz nieco go modyfikuje wprowadzając dziedziczenie pomiędzy obiektami oraz dynamiczne role. Można go również uważać jako model odwzorowujący koncepcję wielu interfejsów do obiektu.
- **as3**: uzupełnia model as1 lub as2 o pojęcie hermetyzacji - podział własności klas i obiektów na publiczne i prywatne.


i_{40} PersonClass
i_{41} age (...code...)
.....
i_{50} EmpClass
i_{51} changeSal (...code...)
i_{52} netSal (...code...)
.....
i_{60} StudentClass
i_{61} avgScore (...code...)
.....
i_1 Person
i_2 name "Doe"
i_3 born 1948
i_4 Person
i_5 name "Poe"
i_6 born 1975
i_{13} Emp
i_{14} sal 2500
i_{15} worksIn
i_{127}
i_7 Person
i_8 name "Lee"
i_9 born 1951
i_{16} Emp
i_{17} sal 1500
i_{18} worksIn
i_{128}
i_{19} Student
i_{20} studentNo 223344
i_{21} faculty "Physics"

Podejście stosowe

- Zakłada, że języki zapytań są szczególnym przypadkiem języków programowania.

- Kluczową rolę odgrywa stos środowisk (**environment stack**)

Przykłady

- 2000 Literał
- Emp Nazwa
- Sal Nazwa
- 2+2 Operator algebraiczny
- sal > 2000

Przykłady

- Emp **where** (sal **>** 2000)
- Emp **where** (sal **>** 2000) **.** (name, (worksIn**.**Dept))
- ((Emp **as** e) **where** ((e**.**sal) **>** (2000 + x + y)))**.**e**.**name
- ((Emp **as** e) **join** ((e**.**worksIn**.**Dept) **as** d)) .
  (e**.**name, Dependent **join with auxiliary** d**.**dname)

Przykłady

- ∀ (Emp where sal < 1000) ((worksIn.Dept.dname) = “Ads”)
- ∃ (Emp as e) count(e.address) = 0
- bag(1, 1, 2, 3, 5, 8, 3, 13)
  Bag constructor

Składnia

query ::= literal

query ::= name

query ::= unaryAlgOperator query

unaryAlgOperator ::= count | sum | max | - | sqrt | not | avg| ...

query ::= query binaryAlgOperator query

binaryAlgOperator ::= =|<| >| +| -| *| /| and| or| intersect|...

Składnia

query ::= query NonAlgOperator query

NonAlgOperator ::= where | . | join | ∀ | ∃

query ::= ∀ query query | ∃ query query

query ::= query as name
query ::= query group as name

query ::= if query then query
query ::= if query then query else query

Składnia

querySeq ::= query | query, querySeq
Sequence of queries

query ::= struct(querySeq) | (querySeq)
Structure constructor

query ::= bag( ) | bag(querySeq)
Bag constructor

query ::= sequence( ) | sequence(querySeq)
Sequence constructor

Materiały

1. www.sbql.pl
2. http://www.ipipan.waw.pl/~subieta/wyklady
3. http://www.omg.org/docs/mars/07-09-13.pdf
4. http://stencel.mimuw.edu.pl/obd