Systém Caché

Objektově orientovaný DB systém

Radek Burget
burgetr@fit.vutbr.cz

Datové modely

- **Jednoduché** (NoSQL)
  - Key-value (MUMPS, Redis, …)
  - Dokumentové (MongoDB, CouchDB, …)
  - Sloupcové (Apache HBase, ...)
- **Relační datový model**
  - Mnoho implementací
- **Objektový datový model**
  - Objektově-relační mapování (ORM)
- **Grafové**
  - Grafové databáze (Neo4J, OrientDB, …)
  - Sémantická úložiště (sémantický web, RDF)

Key-value databáze

- Asociativní pole jako datový model
- Typicky řetězcový klíč
  - Často seřazené klíče – efektivní procházení
  - Možná organizace klíčů, např. hierarchie
- Hodnoty různých datových typů
  - Jedna nebo více hodnot
- Např. Redis, MUMPS

Odbočka: MUMPS

- **M**assachusetts General Hospital **U**tility **M**ulti-**P**rogramming **S**ystem (60. léta v USA)
  - Postupně vzniklo několik komerčních i open-source implementací
- InterSystems Caché (1997 -)

Systém MUMPS

- Jazyk pro databázové aplikace
- Předpokládá existenci databáze společné všem aplikacím
- Proměnné začínající ^ se ukládají do databáze – **globály**

```
SET a = 5     //proměnná v RAM
SET ^a = 5    //proměnná v databázi
```

Ukázky MUMPS

```
; Loop for ever, read a line (quit on end of file), process that line
	For  Read input Quit:$ZEOF!'$Length(input)  Do	; input has entire input line
	.
	. Set i=$Piece(input," ",1)			; i - first number on line is starting integer for the problem
	. Set j=$Piece(input," ",2)			; j - second number on line is ending integer for the problem
	. Write $FNumber(i,",",0)," ",$FNumber(j,",",0)	; print starting and ending integers, formatting with commas
	.
	. Set k=$Piece(input," ",3)			; k - third number on input line is number of parallel streams
	. If streams>k Do				; print number of execution streams, optionally corrected
	..  Write " (",$FNumber(k,",",0)
	..  Set k=streams
	..  Write "->",$FNumber(k,",",0),")"
	. Else  Write " ",$FNumber(k,",",0)
	.
	. Set blk=+$Piece(input," ",4)			; blk - size of blocks of integers is optional fourth piece
	. Set tmp=(j-i+k)\k				; default / maximum block size
	. If blk&(blk'>tmp) Write " ",$FNumber(blk,",",0)	; print block size, optionally corrected
	. Else  Do
	..  Write " (",$FNumber(blk,",",0)
	..  Set blk=tmp
	..  Write "->",$FNumber(blk,",",0),")"
	.
	. ; Define blocks of integers for child processes to work on
	. Kill ^limits
	. Set tmp=i-1
```

Ukázky MUMPS

```
%DTC
%DTC ; SF/XAK - DATE/TIME OPERATIONS ;1/16/92  11:36 AM
    ;;19.0;VA FileMan;;Jul 14, 1992
    D    I 'X1!'X2 S X="" Q
    S X=X1 D H S X1=%H,X=X2,X2=%Y+1 D H S X=X1-%H,%Y=%Y+1&X2
    K %H,X1,X2 Q
    ;
C    S X=X1 Q:'X  D H S %H=%H+X2 D YMD S:$P(X1,".",2) X=X_"."_$P(X1,".",2) K X1,X2 Q
S    S %=%#60/100+(%#3600\60)/100+(%\3600)/100 Q
    ;
H    I X<1410000 S %H=0,%Y=-1 Q
    S %Y=$E(X,1,3),%M=$E(X,4,5),%D=$E(X,6,7)
    S %T=$E(X_0,9,10)*60+$E(X_"000",11,12)*60+$E(X_"00000",13,14)
TOH  S %H=%M>2&'(%Y#4)+$P("^31^59^90^120^151^181^212^243^273^304^334","^",%M)+%D
    S %='%M!'%D,%Y=%Y-141,%H=%H+(%Y*365)+(%Y\4)-(%Y>59)+%,%Y=$S(%:-1,1:%H+4#7)
    K %M,%D,% Q
    ;
DOW  D H S Y=%Y K %H,%Y Q
DW   D H S Y=%Y,X=$P("SUN^MON^TUES^WEDNES^THURS^FRI^SATUR","^",Y+1)_"DAY"
    S:Y<0 X="" Q
7    S %=%H>21608+%H-.1,%Y=%\365.25+141,%=%#365.25\1
    S %D=%+306#(%Y#4=0+365)#153#61#31+1,%M=%-%D\29+1
    S X=%Y_"00"+%M_"00"+%D Q
    ;
```

Hierarchická organizace dat

- Proměnná (globál) může mít libovolné množství indexů („subscript“)

```
SET ^osoba(1) = „Jan Novák“

SET ^osoba(1,“vek“) = 25

SET ^osoba(1,“adresa“,“ulice“) = „Za vodou“

SET ^osoba(1,“adresa“,“cislo“) = 50
```

Caché Object Script

- Vychází z MUMPS
- Novinky v syntaxi
  - Bloky, while cyklus, …
- Objektová rozšíření

Caché ObjectScript - příkazy

- Set - přiřazení, např. SET A = 1
- Do - volání metody nebo rutiny, např. DO X(43)
- Write – výpis hodnoty
- Quit - konec subrutiny, metody, cyklu
- Kill – zrušení proměnné (lokální i globálu)

Řídicí struktury

- If ... Elself ... Else – větvení
  - if a < 5 { … } else { … }
- For - např. FOR X = 1:1:100
- While & Do ... While - smyčky, např. WHILE B < 10 { SET B = B + 1 }.

Zvláštnosti FOR cyklu

```
FOR I = 1:1:5 W I,!

FOR I = 1:1:5 W I W:I=2 „*“

FOR I = 1:1:5 {
   w I
   w:I=2 „*“
}
```

Příklad rutiny

```
;
;soubor vek.mac
;
vekosoby(jmeno)
   SET roknar = ^osoba(jmeno, "rocnik")
   SET rok = $extract($zdate($horolog, 8), 0, 4)
   Write rok - roknar
```

```
Do vekosoby^vek(„jan“)
```

Objektový model v Caché

- Třídy definovány pomocí vnitřního Class Definition Language
  - Třída `Class`, vícenásobná dědičnost
  - Parametry třídy `Parameter`
  - Vlastnosti `Property(Parametr=hodnota)`
  - Metody `Method`, `ClassMethod`
  - Pojmenované dotazy

Práce s objekty

- Vytvoření instance třídy
  - `set p = ##class(demo.Person).%New()`
- Nastavení hodnot vlastností
  - `set p.surname = “Clinton”`
- Uložení do databáze (persistence)
  - `do p.%Save()`
- Zjištění identifikátoru
  - `w p.%Id()`

Přístup o objektům

- Přes ID
  - `set p = ##class(demo.Person).OpenId(id)`
- Pojmenovaný dotaz
- Ad-hoc dotaz (pseudo SQL)

Pojmenovaný dotaz

- Pro danou třídu
- Každá třída má předdefinovaný dotaz **Extent**

```
Set rset =
   ##class(%ResultSet).%New("demo.Person:Extent")
Do rset.Execute()
…
set p = rset.GetObject()
```

Vlastní dotaz

```
Query QueryName(Parameter As %String) As %SQLQuery

{

SELECT MyProperty, MyOtherProperty FROM MyClass WHERE
  (MyProperty = "Hello" AND MyOtherProperty =
  :Parameter) ORDER BY MyProperty

}
```

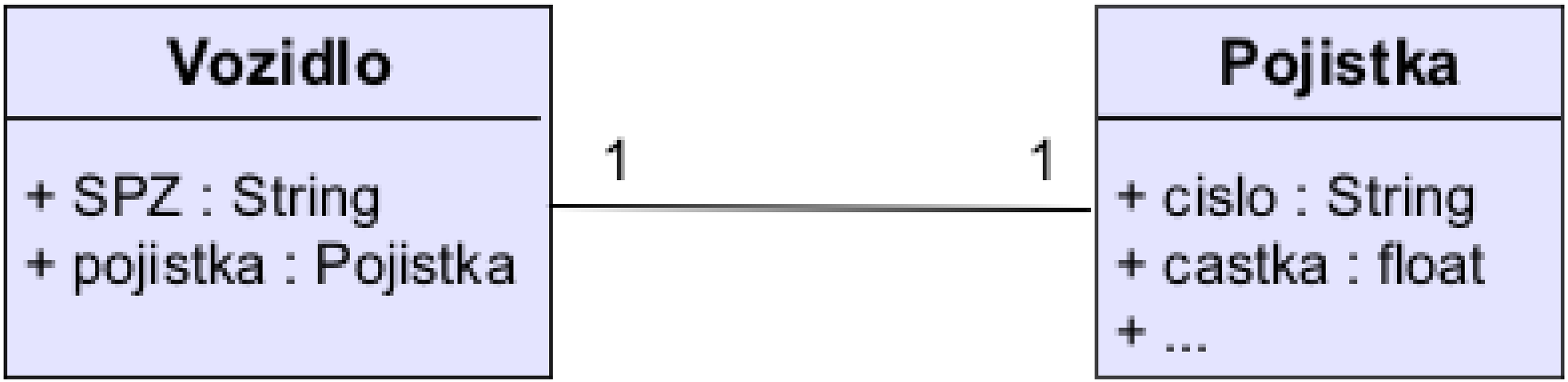
Ad-hoc dotazy

```
Set dotaz = "SELECT %ID FROM demo.Person WHERE
  name='Jan'"


Set rset = ##class(%ResultSet).%New()

Do rset.Prepare(dotaz)

Do rset.Execute()
```

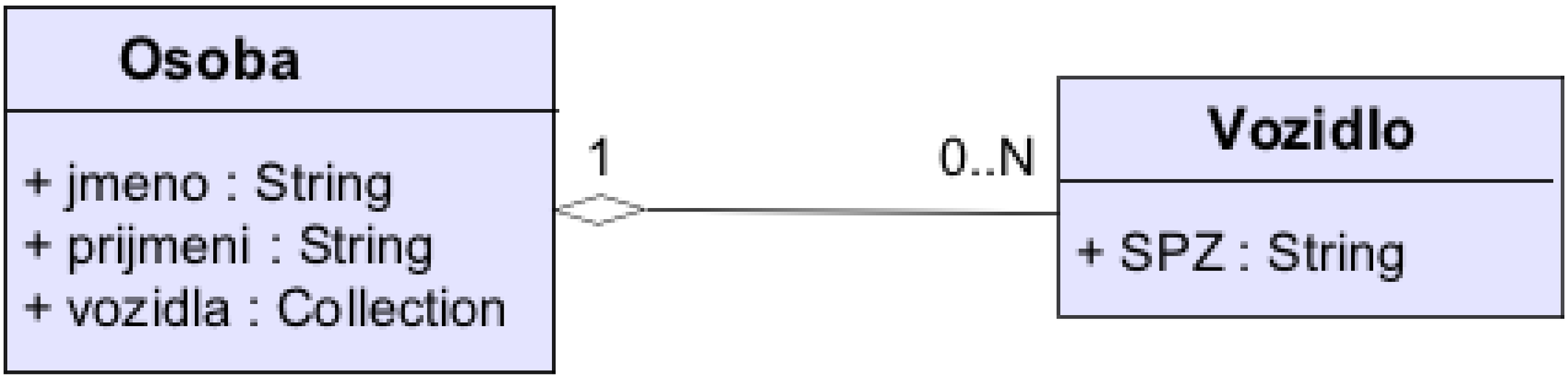
Vztahy v objektové databázi

- Z hlediska objektové databáze
  - Vztahy 1:1
  - Vztahy 1:N
  - Vztahy M:N

Vztahy 1:1


Vozidlo
+ SPZ : String
+ pojistka : Pojistka
1
1
Pojistka
+ cislo : String
+ castka : float
+ ...

Vztahy 1:N


Osoba
+ jmeno : String
+ prijmeni : String
+ vozidla : Collection
1
0..N
Vozidlo
+ SPZ : String

Kolekce v Caché

- Array
  - Asociativní pole
  - K hodnotám se přistupuje přes klíč
- List
  - Uspořádaný seznam
  - K hodnotám se přistupuje přes index 1..n

List

- Deklarace
  - `Property jmena as %String [ Collection = list ]`
- Metody
  - `jmena.Insert(“Jan”)`
  - `jmena.GetAt(index)`
  - `jmena.SetAt(“John”, index)`
  - `jmena.InsertAt(“Jana”, index)`
  - `jmena.Count()`

Relation

- Vlastnosti vyjadřující relace

`Relation <jmeno> as Typ [parametry]`

- Parametry:

  Cardinality = one | many

  parent | children

  Inverse = *vlastnost protitřídy*

- Vždy oboustranné
- Caché zabezpečí konzistenci vztahu

Příklad

```
Class demo.Person Extends  …
{
  Relationship cars
  As demo.Car
  [ Cardinality = many, Inverse = owner ];

}
```

Příklad

```
Class demo.Car Extends …
{
  Relationship owner
    As demo.Person
    [ Cardinality = one, Inverse = cars ];

}
```

Příklad

```
set c = ##class(demo.Cars).%New()
set p = ##class(demo.Person).%New()

do p.cars.Insert(c)
//…vlastnost c.owner bude mít hodnotu p

do p.%Save()
//…uloží se automaticky i c
```

Objektová reprezentace

- Relace se projeví
  - Na straně ONE jako běžná vlastnost daného typu
  - Na straně MANY jako **kolekce**

    (konkrétně reprezentovaná datovým typem **%RelationshipObject,** se stejným rozhraním, jako ostatní kolekce)

Vztahy Parent – Children

- Objekty Children jsou závislé na Parent
  - Smazání otcovského objektu maže odkazované objekty
  - Závislé objekty nemohou být asociovány s jiným otcovským

Otázky?