Česká zemědělská univerzita v Praze
Provozně ekonomická fakulta
Katedra informačního inženýrství

Projekt z předmětu

Databázové a znalostní informační systémy

Kantýna

Vypracoval: Bc. Zdeněk Styblík
Obor: Informatika
Ročník: 1NK
Rok: 2014

Obsah

Úvod do problému....3
Konceptuální model....4
Data....6
RDBM....6
OODB....8
Závěr....14

Úvod do problému

Cílem této práce je zvolit vhodný problém a navrhnout jeho řešení v oblasti databází. Toto řešení implementovat dvěma způsoby a to relačně a objektově. Výsledek práce je porovnání obou přístupů.

Tato práce se zabývá firemní kantýnou, ve které si zákazníci(zaměstnanci firem v ramci technologického parku) mohou nakoupit jídlo a pití. K tomu, aby bylo možné nakupovat, se musí zaměstnanec zaregistrovat, resp. musí být do databáze vloženy jeho osobní údaje a číslo zaměstnanecké karty. Také si musí nabít kredit, který slouží k úhradě nákupu. Nákup nelze uskutečnit bez zaměstnance, který zajišťuje chod kantýny, a také musí být zakoupen alespoň jeden kus zboží, které je předmětem prodeje.

Popis tříd

Osoba obsahuje základní údaje - jméno, příjmení, telefon, idKarty, email - a je předkem pro *Zákazník* a *Zaměstnanec*.

Zákazník je *Osoba*, která nakupuje v kantýně. U zákazníka vedeme atribut kredit.

Zaměstnanec je *Osoba*, která prodává v kantýně a zajišťuje její chod. Evidujeme u ní výši mzdy, datum nástupu a ukončení pracovního poměru.

Nákup je třída, která dokumentuje jednotlivé nákupy zákazníků. Říká kdo a kdy nakoupil, a kterým zaměstnancem byl obsloužen. Nákup nelze provést bez obsluhy. Stejně tak nákup nemůže proběhnout bez jediné položky, resp. zakoupeného kusu zboží.

PoložkaNákupu je asociační třídou na *Nákup* a udává kolik kusů daného zboží jsme prodali a za kolik.

Zboží obsahuje obecné údaje o prodávaném zboží, a je předkem pro třídy *Nápoj* a *Potravina*. Obsahuje atributy jako název, cenu, stav skladových zásob, zemi původu, čárový kód.

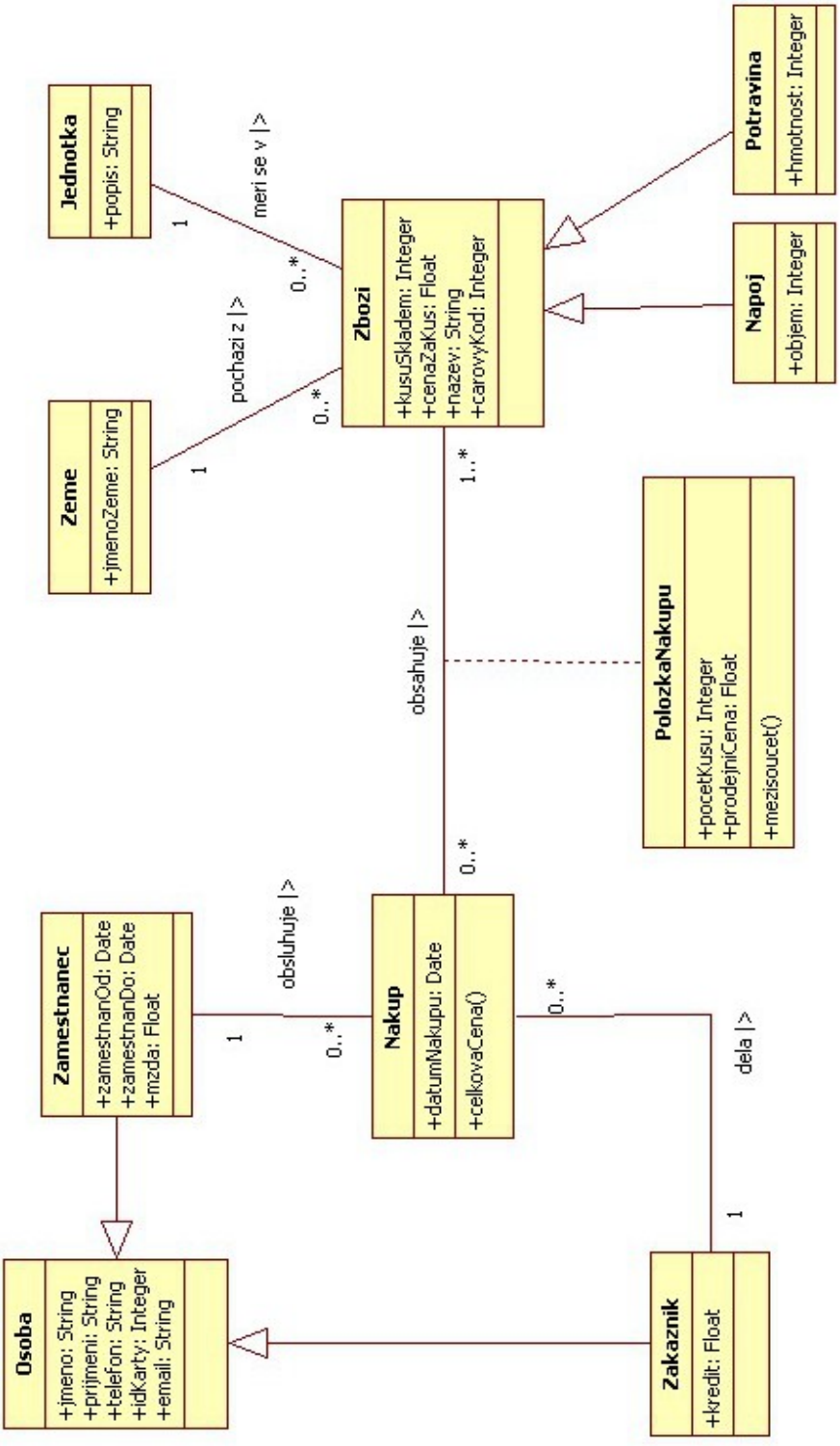
Nápoj obsahuje atributy specifické pro nápoje.

Potravina obsahuje atributy specifické pro potraviny.

Země udává zemi, ve které bylo zboží vyrobeno.

Jednotka udává slovní popis měrných jednotek pro dané zboží.

Konceptuální model



Dotazy

- Vyber zákazníka, který si zakoupil "Pivo Plzeň"
- (Zakaznik // (λ z | z ◄ nakup ◄ polozky ◄ zbozi ◄ nazev = 'Pivo Plzen')) >> [jmeno, prijmeni]
- Vyber jméno a příjmení Zaměstnance, který ještě nikoho neobsloužil
- (Zamestnanec // (λ z | z ◄ nakup | = 0))
- Vyber jméno, příjmení Zákazníka, který si ještě nic nekoupil
- (Zakaznik // (λ z | z ◄ nakup | = 0))
- Vyber zákazníka, který utratil více nez 50
- (Zakaznik // (λ z | z ◄ nakup ◄ celkovaCena > 50)) >> [jmeno, prijmeni]
- Vyber zákazníka, který si koupil 'Deli'
- (Zakaznik // (λ z | z ◄ nakup ◄ polozky ◄ zbozi ◄ nazev = 'Deli')) >> [jmeno, prijmeni]

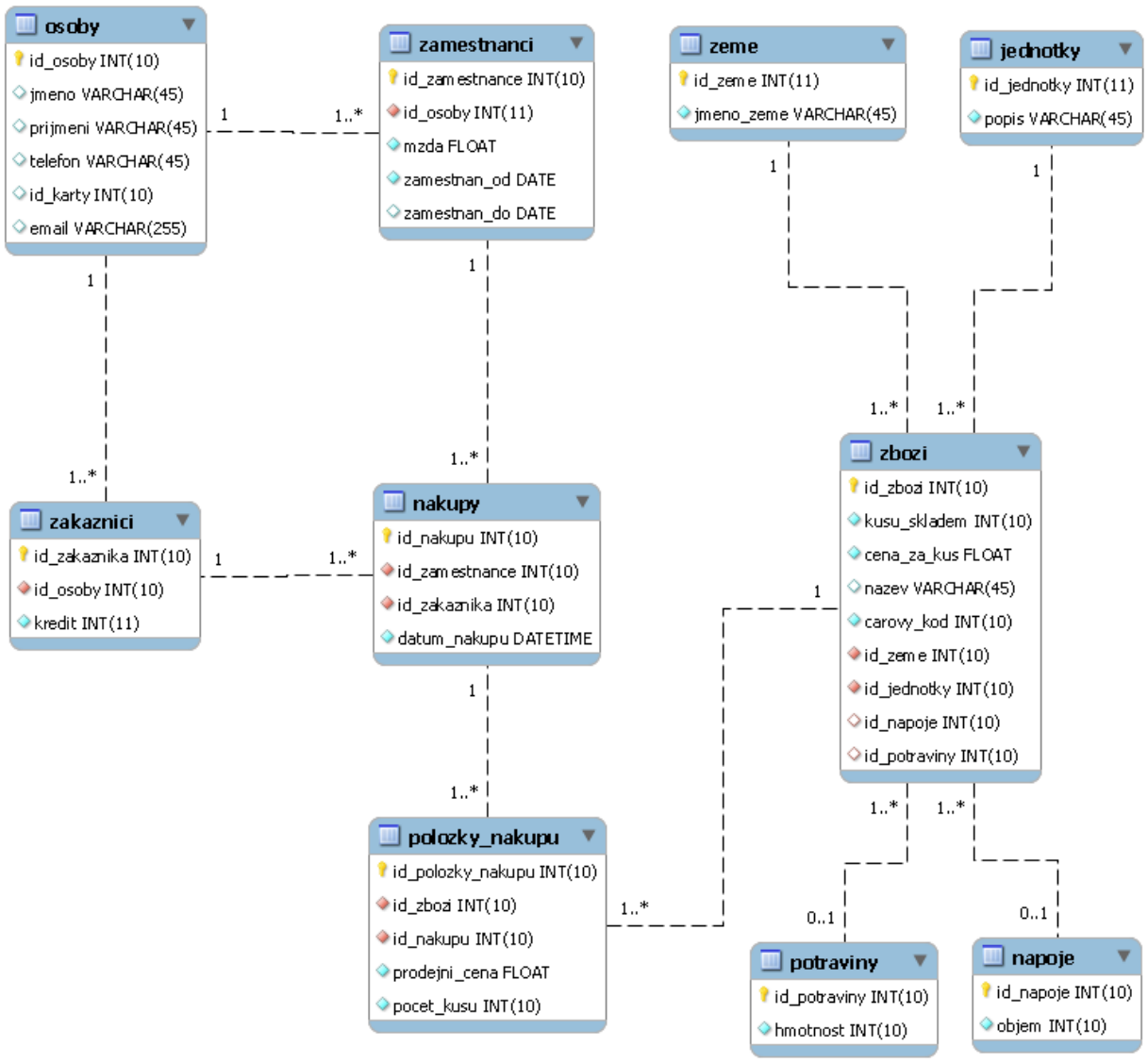
Omezení

- Mzda zaměstnanců nesmí být vyšší než 25000
- Zamestnanec // (λ a | | a ◄ mzda| =< 25000))
- Žádné zboží nemá cenu za kus vyšší než 100.
- Zbozi // (λ b | | b ◄ cenaZaKus| =< 100))
- Kredit zákazníka nesmí být menší než 0
- Zakaznik // (λ c | | c ◄ kredit| < 0))
- Zbozi skladem nesmí být menší než 0
- Zbozi // (λ d | | d ◄ kusuSkladem| < 0))
- Zbozi skladem nesmí být větší než 100
- Zbozi // (λ d | | d ◄ kusuSkladem| > 100))

Data

Data viz příloha.

RDBM

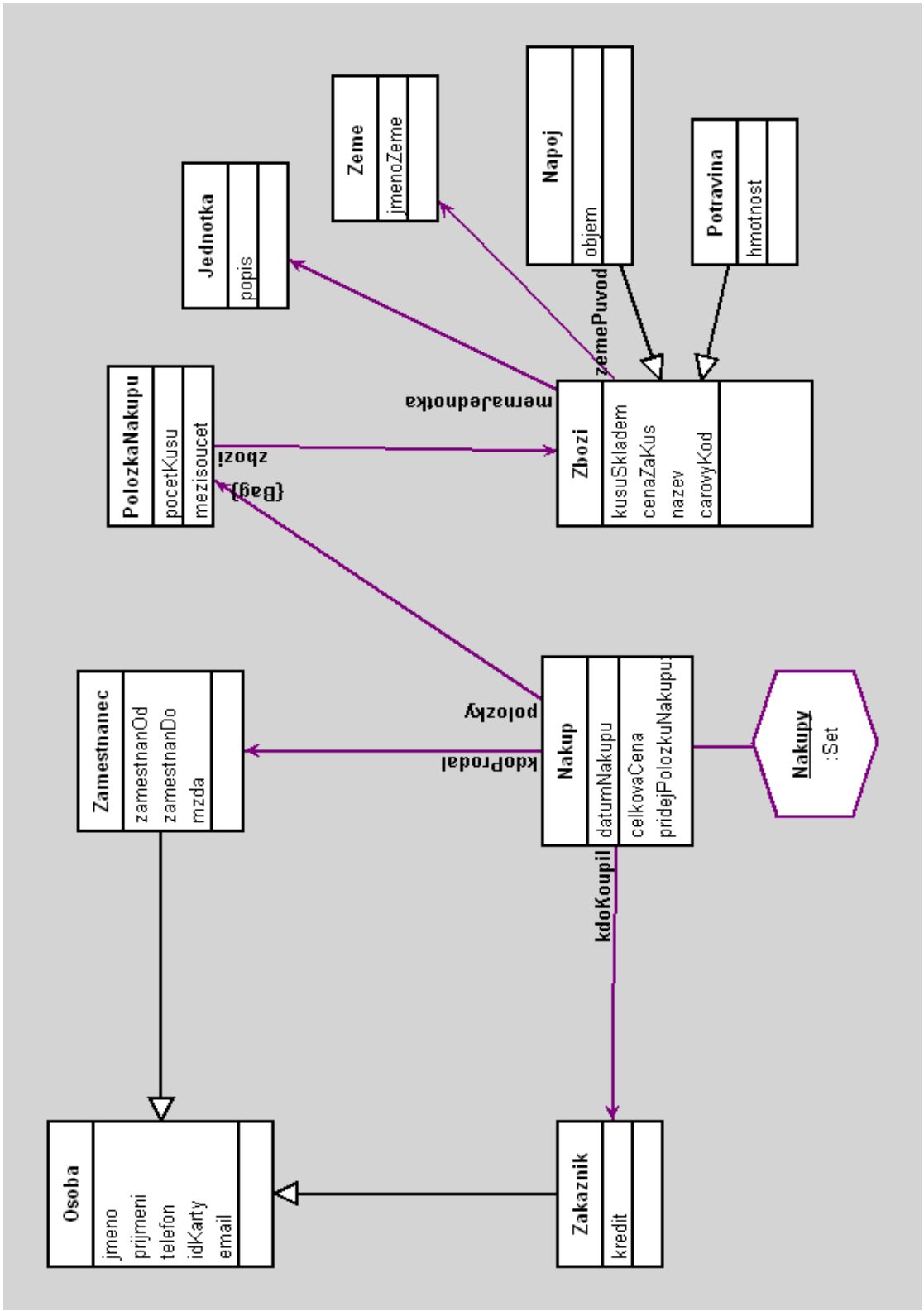


Dotazy

- Vyber zákazníka, který si zakoupil "Pivo Plzeň"
- SELECT osoby.jmeno, osoby.prijmeni FROM osoby JOIN zakaznici ON osoby.id_osoby = zakaznici.id_osoby JOIN nakupy ON zakaznici.id_zakaznika = nakupy.id_zakaznika WHERE id_nakupu = (SELECT id_nakupu FROM polozky_nakupu WHERE id_zbozi = (SELECT id_zbozi FROM zbozi WHERE nazev = 'Pivo Plzen'));

- Vyber jméno a příjmení Zaměstnance, který ještě nikoho neobsloužil
- SELECT osoby.jmeno, osoby.prijmeni, FROM osoby JOIN zamestnanci ON osoby.id_osoby = zamestnanci.id_osoby LEFT JOIN nakupy ON nakupy.id_zamestnance = zamestnanci.id_zamestnance WHERE nakupy.id_nakupu IS NULL;
- Vyber jméno, příjmení Zákazníka, který si ještě nic nekoupil
- SELECT osoby.jmeno, osoby.prijmeni FROM osoby JOIN zakaznici ON osoby.id_osoby = zakaznici.id_osoby LEFT JOIN nakupy ON nakupy.id_zakaznika = zakaznici.id_zakaznika WHERE nakupy.id_nakupu IS NULL;
- Vyber zákazníka, který utratil více nez 50
- SELECT osoby.jmeno, osoby.prijmeni FROM osoby JOIN zakaznici ON osoby.id_osoby = zakaznici.id_osoby JOIN nakupy ON zakaznici.id_zakaznika = nakupy.id_zakaznika WHERE nakupy.id_nakupu = (SELECT id_nakupu FROM polozky_nakupu WHERE (polozky_nakupu.prodejni_cena * polozky_nakupu.pocet_kusu) > 50);
- Vyber zákazníka, který si koupil 'Deli'
- SELECT osoby.jmeno, osoby.prijmeni FROM osoby JOIN zakaznici ON osoby.id_osoby = zakaznici.id_osoby JOIN nakupy ON zakaznici.id_zakaznika = nakupy.id_zakaznika WHERE nakupy.id_nakupu = (SELECT polozky_nakupu.id_nakupu FROM polozky_nakupu WHERE polozky_nakupu.id_zbozi = (SELECT zbozi.id_zbozi FROM zbozi WHERE zbozi.nazev = 'Deli'));

OODB


Jednotka
popis
Zeme
jmenoZeme
Napoj
objem
Potravina
hmotnost
zemePuvod
PolozkaNakupu
pocetKusu
mezisoucet
zbozi
{bag}
mernaJednotka
Zbozi
kusuSkladem
cenaZaKus
nazev
carovyKod
Zamestnanec
zamestnanOd
zamestnanDo
mzda
polozky
kdoProdal
Nakup
datumNakupu
celkovaCena
pridejPolozkuNakupu
Nakupy
:Set
kdoKoupil
Osoba
jmeno
prijmeni
telefon
idKarty
email
Zakaznik
kredit

Třídy

Osoba
instance variables email :String idKarty :Number jmeno :String prijmeni :String telefon :String
methods email email: idKarty idKarty: initialize jmeno jmeno: prijmeni prijmeni: telefon telefon:

code of non-accessing methods:

- **initialize**

```
"generated by Daskalos"

     super initialize.
     jmeno := nil.
     prijmeni := nil.
     telefon := nil.
     idKarty := nil.
     email := nil.
```

PolozkaNakupu
instance variables pocetKusu :Number prodejniCena :Number zbozi :Object
methods initialize mezisoucet pocetKusu pocetKusu: prodejniCena prodejniCena: zbozi

zbozi:

code of non-accessing methods:

- **initialize**

```
"generated by Daskalos"

     super initialize.
     pocetKusu := nil.
     prodejniCena := nil.
     zbozi := nil.
```

- **mezisoucet**

```
        "vrati pocetKusu * prodejniCena"

        | vysledek |
        vysledek := pocetKusu * prodejniCena.
        ^vysledek
```

Nakup
instance variables datumNakupu :String kdoKoupil :Object kdoProdal :Object polozky :Bag
methods celkovaCena datumNakupu datumNakupu: initialize kdoKoupil kdoKoupil: kdoProdal kdoProdal: polozky pridejPolozkuNakupu:

code of non-accessing methods:

- **celkovaCena**

```
        "vrati celkovou cenu vsech polozek"

        | celkovaCena |
        celkovaCena := polozky sum: [:s | s mezisoucet].
        ^celkovaCena
```

- **initialize**

```
"generated by Daskalos"

     super initialize.
     datumNakupu := nil.
     kdoProdal := nil.
```

```
        polozky := Bag new.
        kdoKoupil := nil.
```

- **pridejPolozkuNakupu: aPolozkaNakupu**

```
        "prida polozku nakupu do polozky"

        self polozky add: aPolozkaNakupu
```

Zeme
instance variables jmenoZeme :Object
methods initialize jmenoZeme jmenoZeme:

code of non-accessing methods:

- **initialize**

```
    "generated by Daskalos"

        super initialize.
        jmenoZeme := nil.
```

Jednotka
instance variables popis :String
methods initialize popis popis:

code of non-accessing methods:

- **initialize**

```
    "generated by Daskalos"

        super initialize.
        popis := nil.
```

Zbozi
instance variables carovyKod :Number cenaZaKus :Number kusuSkladem :Number mernaJednotka

:Object nazev :String zemePuvodu :Object
methods carovyKod carovyKod: cenaZaKus cenaZaKus: initialize kusuSkladem kusuSkladem: mernaJednotka mernaJednotka: nazev nazev: zemePuvodu zemePuvodu:

code of non-accessing methods:

- **initialize**

```
"generated by Daskalos"

     super initialize.
     kusuSkladem := nil.
     cenaZaKus := nil.
     nazev := nil.
     carovyKod := nil.
     mernaJednotka := nil.
     zemePuvodu := nil.
```

Zamestnanec
instance variables mzda :Number zamestnanDo :Date zamestnanOd :Date
methods initialize mzda mzda: zamestnanDo zamestnanDo: zamestnanOd zamestnanOd:

code of non-accessing methods:

- **initialize**

```
"generated by Daskalos"

     super initialize.
```

```
        zamestnanOd := nil.
        zamestnanDo := nil.
        mzda := nil.
```

Zakaznik
instance variables kredit :Number
methods initialize kredit kredit:

code of non-accessing methods:

- **initialize**

```
"generated by Daskalos"

    super initialize.
    kredit := nil.
```

Potravina
instance variables hmotnost :Number
methods hmotnost hmotnost: initialize

code of non-accessing methods:

- **initialize**

```
"generated by Daskalos"

    super initialize.
    hmotnost := nil.
```

Napoj
instance variables objem :Number
methods initialize objem objem:

code of non-accessing methods:

- **initialize**

```
"generated by Daskalos"

    super initialize.
    objem := nil.
```

Závěr

Cílem tohoto projektu bylo vyřešit jedno stejné zadání a to, jak relačně, tak objektově. Díky tomu je možné nyní posoudit výhody a nevýhody jednotlivých řešení. V konceptuálním modelu kantýny nám figurovalo 9 tříd. Při implementaci v relační databázi bylo třeba přidat vazební tabulky v relacích m:n a opatřit tabulky primárními klíči. Implementace v objektové databázi nevyžaduje definici nových položek.

Obecně lze říci, že implementace podle objektového paradigmatu je jednodušší, protože není třeba záznamy propojovat přes cizí klíče. Co se týče manipulace s daty, je objektový přístup také pohodlnější, protože umožňuje vytváření kolekcí a tím usnadňuje přístup k logicky provázaným objektům bez složitého spojování tabulek v dotazech.

Výše zmíněné výhody objektového přístupu jsou nesporné. Konkrétní metodika při vytváření tohoto projektu je už jiná záležitost. Ačkoliv má použití programu Daskalos mnoho výhod, pro praktické využití není příliš vhodný. A pokud bychom měli použít opravdouvou objektovou databázi, pak by byla její konfigurace ještě náročnější. Tedy pro někoho, kdo běžně nepoužívá objektové databázem, by se s největší pravděpodobností jednalo o nadlidský úkol. Nezbývá než doufat, že se brzy objeví nějaký balík, který by si uživatel doma lehce nainstaloval a spustil, a který by mu sloužil pro více než výukové účely.