

# Logično načrtovanje

Logično načrtovanje v okviru načrtovanja podatkovne baze.

# Koraki logičnega načrtovanja <sup>(1)</sup>

- K2.1: **Za entitetne tipe kreiraj relacije**
- K2.2: **Preveri relacije z normalizacijo**
- K2.3: **Preveri relacije s pregledom uporabniških transakcij**
- K2.4: **Preveri omejitve integritete**

# Koraki logičnega načrtovanja <sup>(2)</sup>

- **K2.5: Preveri model z uporabnikom**
- **K2.6: Združi lokalne modele v globalni model (opcijsko)**
- **K2.7: Preveri zmožnosti modela za razširitve**

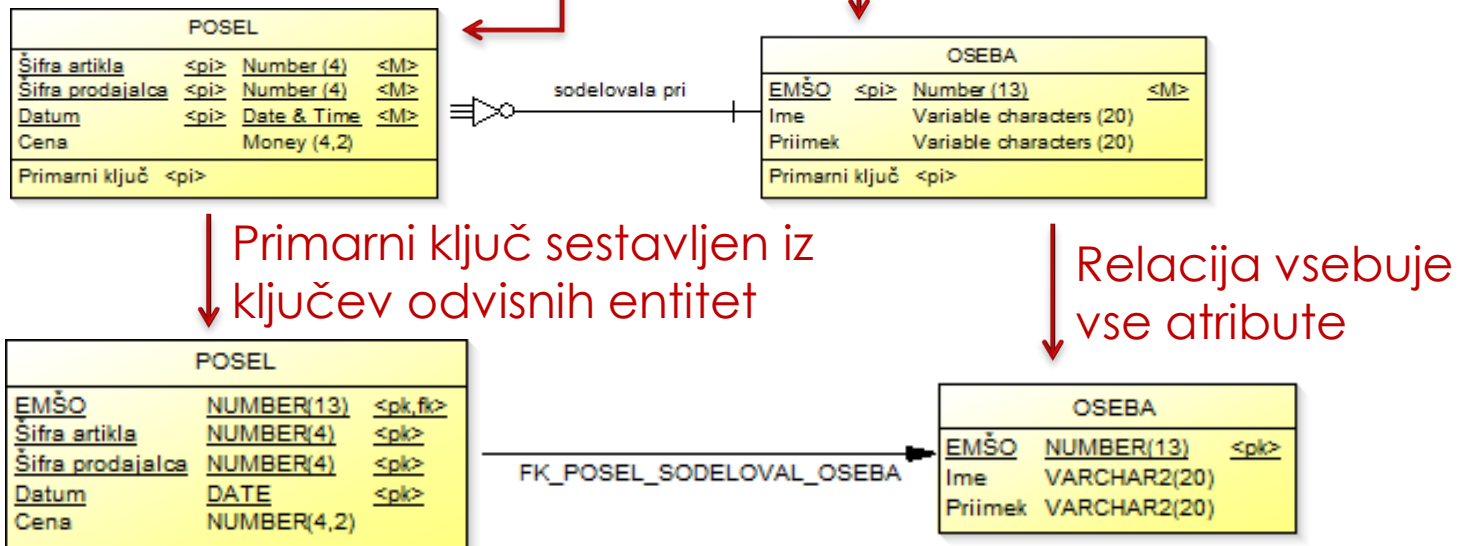
# Preslikave konceptualnega v logični model

Konceptualni model		Logični model
<b>entiteta</b>	→	zapis v tabeli
<b>entitetni tip</b>	→	tabela
<b>atribut</b>	→	stolpec v tabeli
<b>entitetni identifikator</b>	→	primarni ključ
<b>povezava 1:n</b>	→	referenca + tuj ključ
<b>povezava m:n</b>	→	vmesna tabela + pripadajoči referenci
šibki entitetni tip	→	tuj ključ je tudi del primarnega ključa

# Preslikava v logični model (1)

(šibki in močni entitetni tip)

- Preslikava šibkega in močnega entitetnega tipa ter povezave 1:m



**POSEL** (#EMŠO, Sifra\_art, Sifra\_prod, Datum, Cena)

**OSEBA** (EMŠO, Ime, Priimek)

# Preslikava v logični model (2)

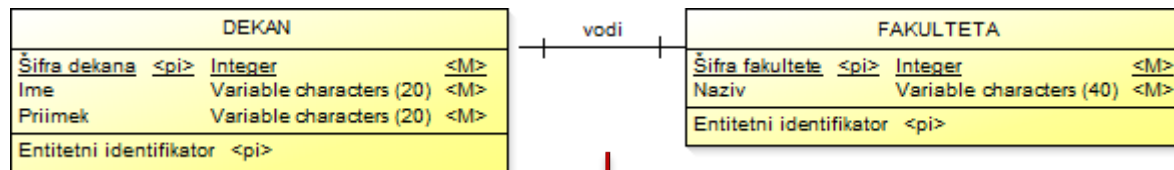
(povezava 1:1)

- Preslikava **povezave 1:1**
  - Ne moremo enostavno določiti očeta in otrok
  - Možne so naslednje omejitve:
    - **Obveznost** na obeh straneh
    - **Obveznost** na eni in **neobveznost** na drugi strani
    - **Neobveznost** na obeh straneh

# Preslikava v logični model (3)

(povezava 1:1)

- Preslikava **povezave 1:1**
- **Obveznost** na obeh straneh



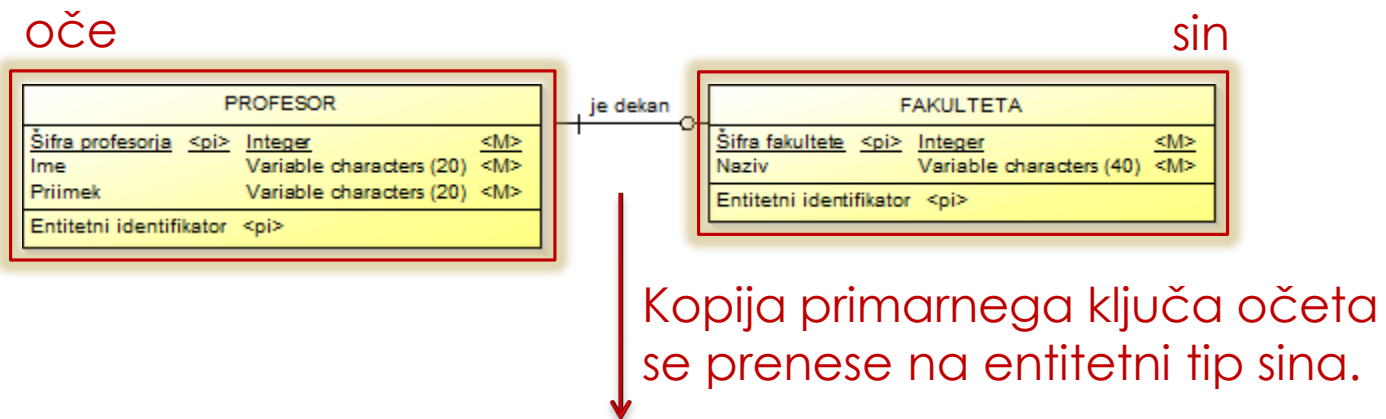
Združimo v eno relacijo, primarni ključ izberemo iz izbranega entitetnega tipa.

**DEKAN** (Šifra dekana, Ime, Priimek, Naziv\_fakultete)

# Preslikava v logični model (4)

(povezava 1:1)

- Preslikava **povezave 1:1**
- Obveznost** na eni in **neobveznost** na drugi strani



**PROFESOR** (Šifra profesorja, Ime, Priimek)

**FAKULTETA** (Šifra fakultete, Naziv, #Šifra\_profesorja)



# Preslikava v logični model (5)

(povezava 1:1)

- Preslikava **povezave 1:1**
  - **Neobveznost** na obeh straneh
    - Težko določiti očeta in otroka povezave.
    - Ko pridobimo dovolj podatkov, določimo ključ.

# Preslikava v logični model (6)

(rekurzivna povezava 1:1)

- Preslikava **rekurzivne povezave 1:1**
  - Možne so naslednje omejitve:
    - **Obveznost** na obeh straneh
      - 1 relacija in 2 kopiji primarnega ključa.
    - **Neobveznost** na obeh straneh
      - 1 dodatna relacija, ki ima 2 atributa – kopiji primarnega ključa.
    - **Obveznost** na eni in **neobveznost** na drugi strani

# Preslikava v logični model (7)

(rekurzivna povezava 1:1)

- Preslikava **rekurzivne povezave 1:1**
- Obveznost** na eni in **neobveznost** na drugi strani



**PDELAVEC** (Sifra\_delavca, Ime, Priimek, Naziv, #Sifra\_mentorstva)

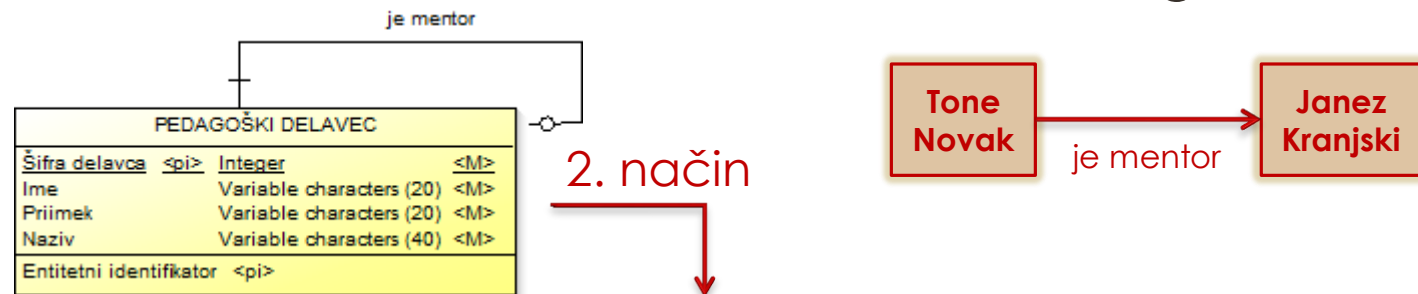
PDELAVEC (1, "Janez", "Kranjski", "mag.", **NULL**)

PDELAVEC (2, "Tone", "Novak", "dr.", 1)

# Preslikava v logični model (8)

(rekurzivna povezava 1:1)

- Preslikava **rekurzivne povezave 1:1**
- **Obveznost** na eni in **neobveznost** na drugi strani



**PDELAVEC** (Sifra delavca, Ime, Priimek, Naziv)

**MENTOR** (#Sifra mentorja, #Sifra delavca)



PDELAVEC (1, "Janez", "Kranjski", "mag.")

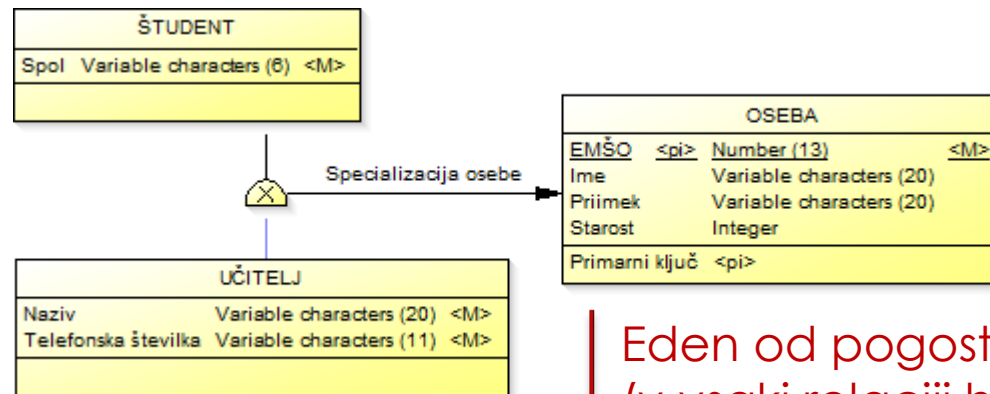
PDELAVEC (2, "Tone", "Novak", "dr.")

MENTOR (**2**, **1**)

# Preslikava v logični model (9)

(generalizacija/specializacija)

- Preslikava **nad tipov** in **pod tipov**
- Več različnih možnosti



Eden od pogostih pristopov  
(v vsaki relaciji hranimo  
samo dodatne attribute)

**OSEBA** (EMŠO, Ime, Priimek, Starost)

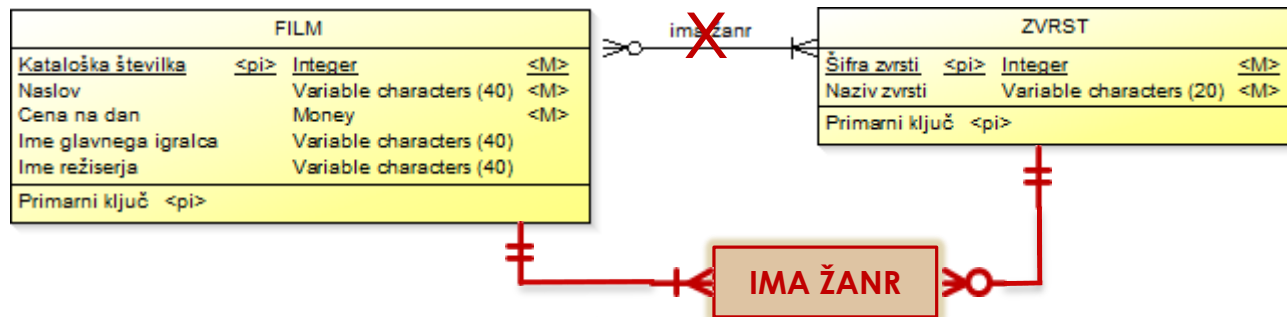
**STUDENT** (#EMŠO, Spol)

**UCITELJ** (#EMŠO, Naziv, Telefonska\_stevilka)

# Preslikava v logični model (10)

(povezava m:n)

## ○ Preslikava **povezave m:n**



Vpeljava vmesne entitete, ključ je sestavljen iz ključev odvisnih entitet.

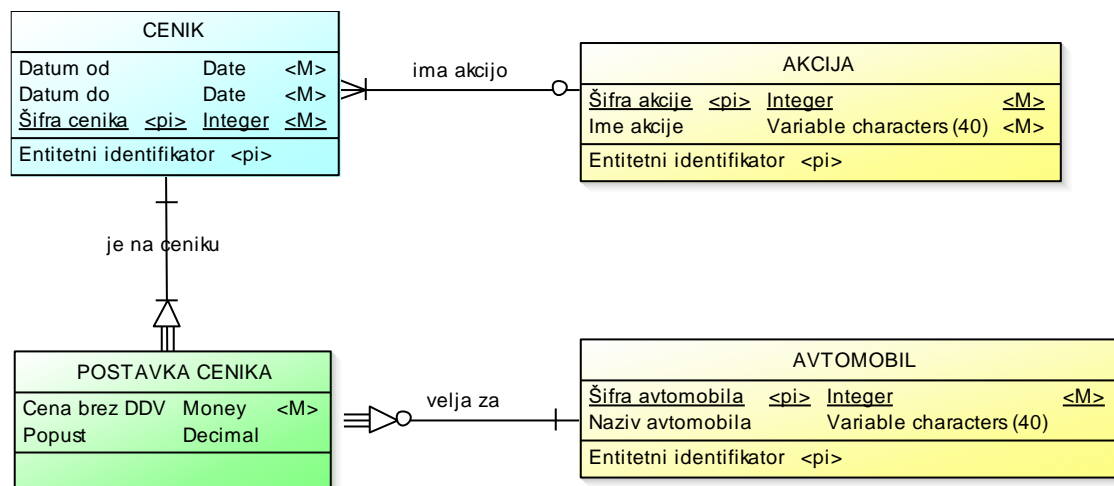
**FILM** (Kat\_st, Naslov, Cena\_na\_dan, Glavni\_igralec, Reziser)

**ZVRST** (Sifra\_zvrsti, Naziv)

**IMA\_ZANR** (#Kat\_st, #Sifra\_zvrsti)

# Prodaja avtomobilov

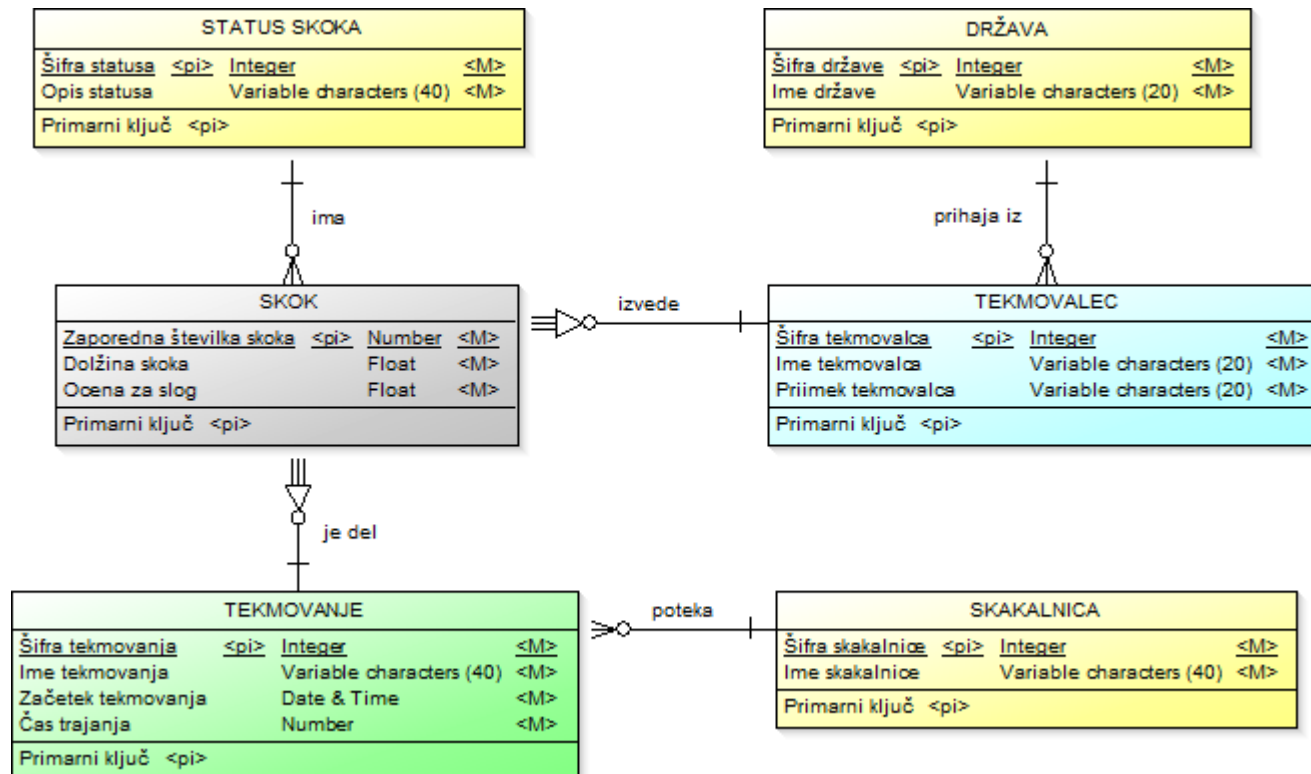
Naloga preslikave konceptualnega modela v logični model.





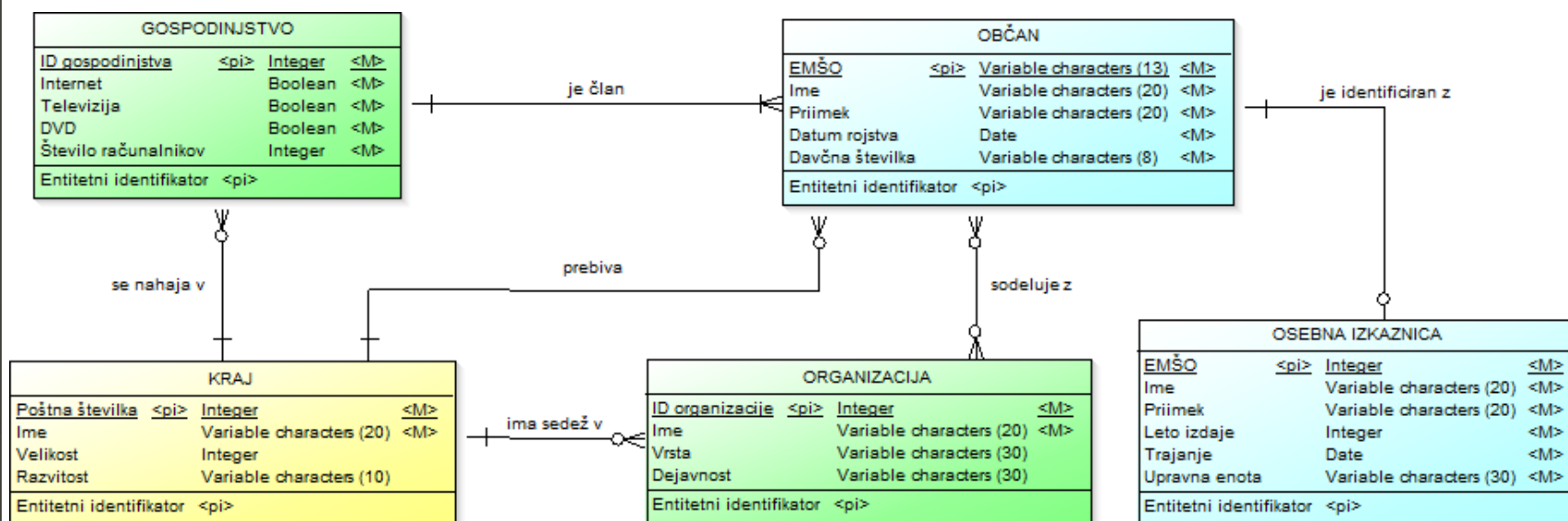
# Smučarski skoki

Naloga preslikave konceptualnega modela v logični model.



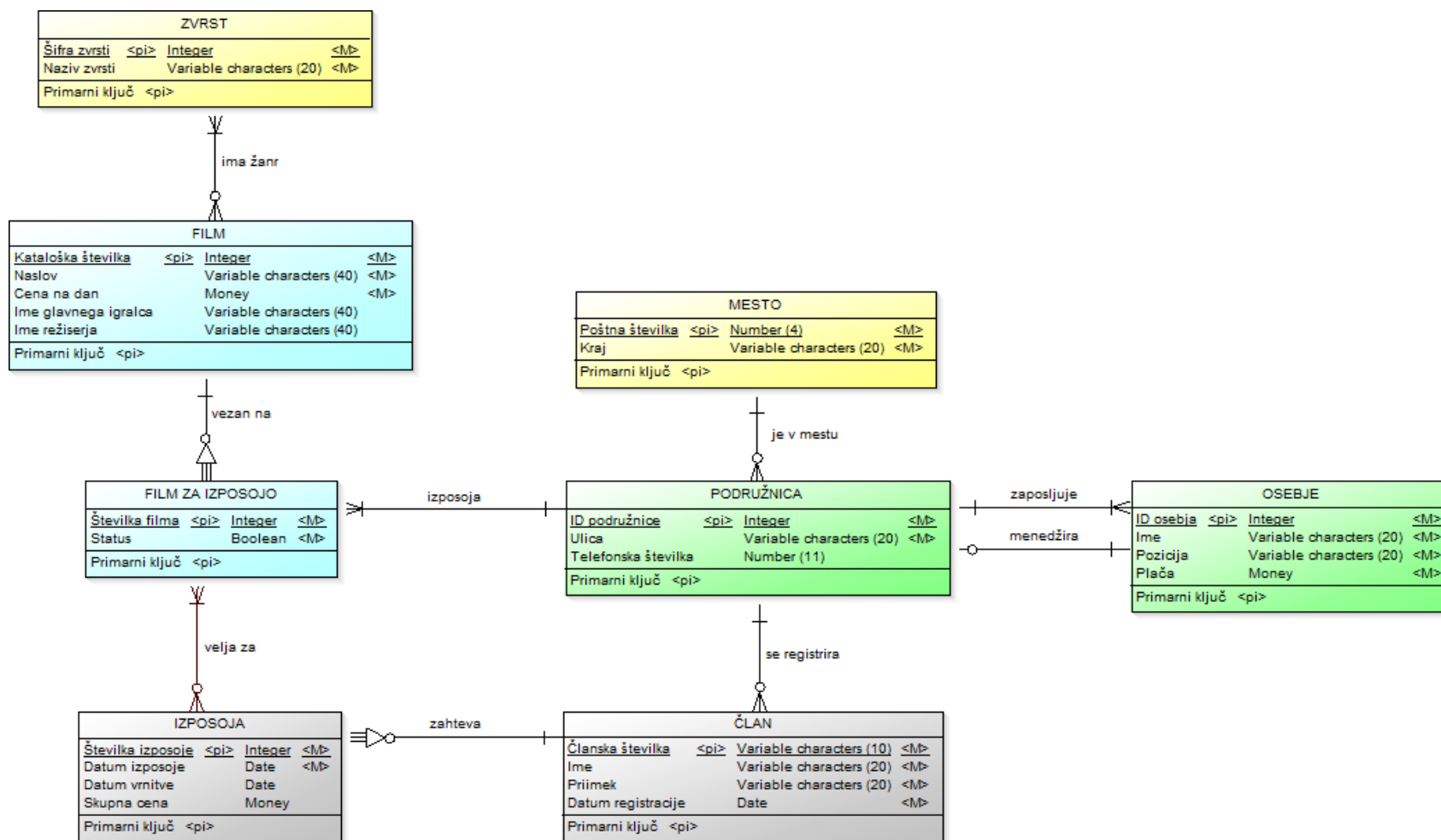
# Gospodinjstvo

Naloga preslikave konceptualnega modela v logični model.



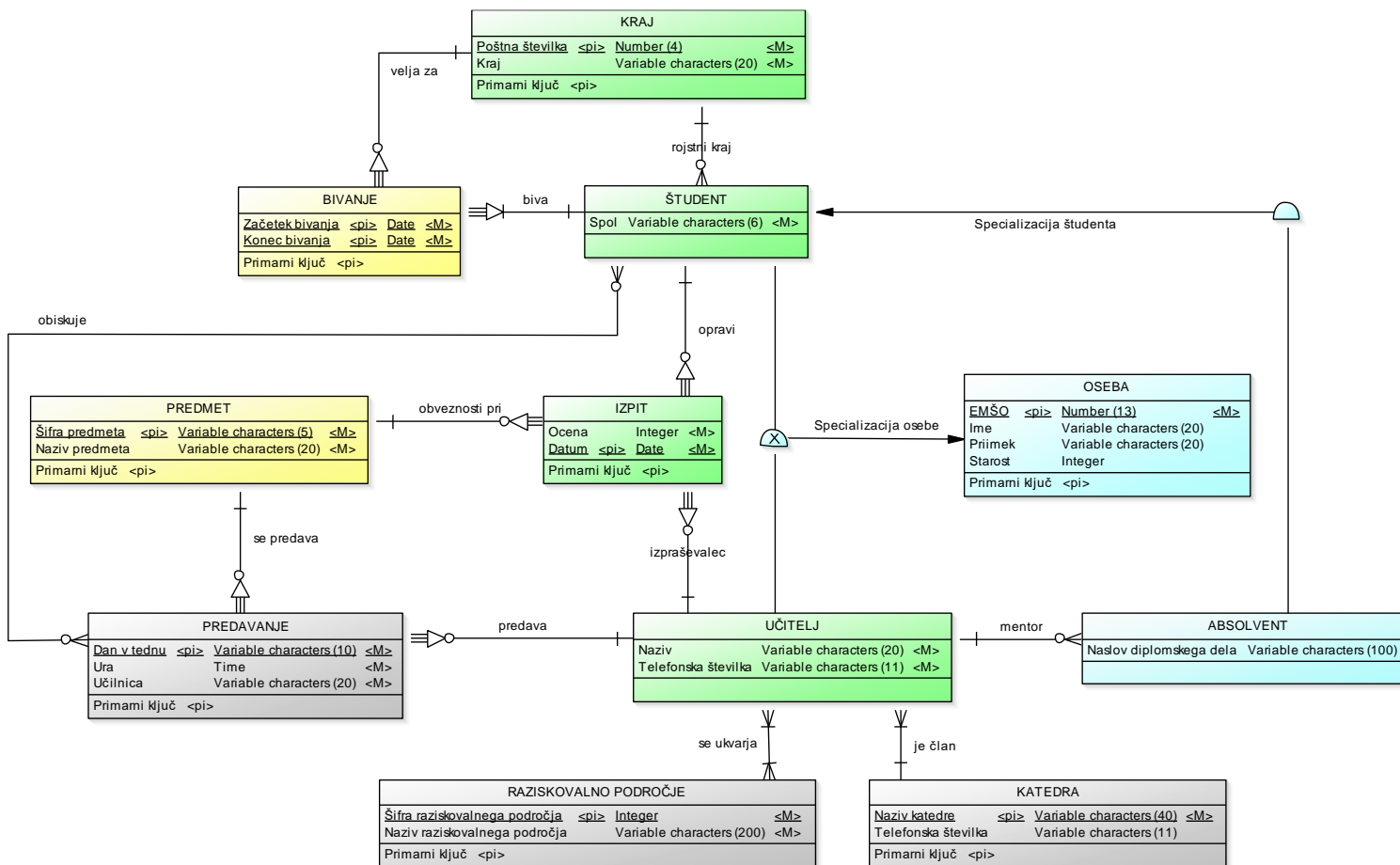
# Videoteka

Naloga preslikave konceptualnega modela v logični model.



# Fakulteta

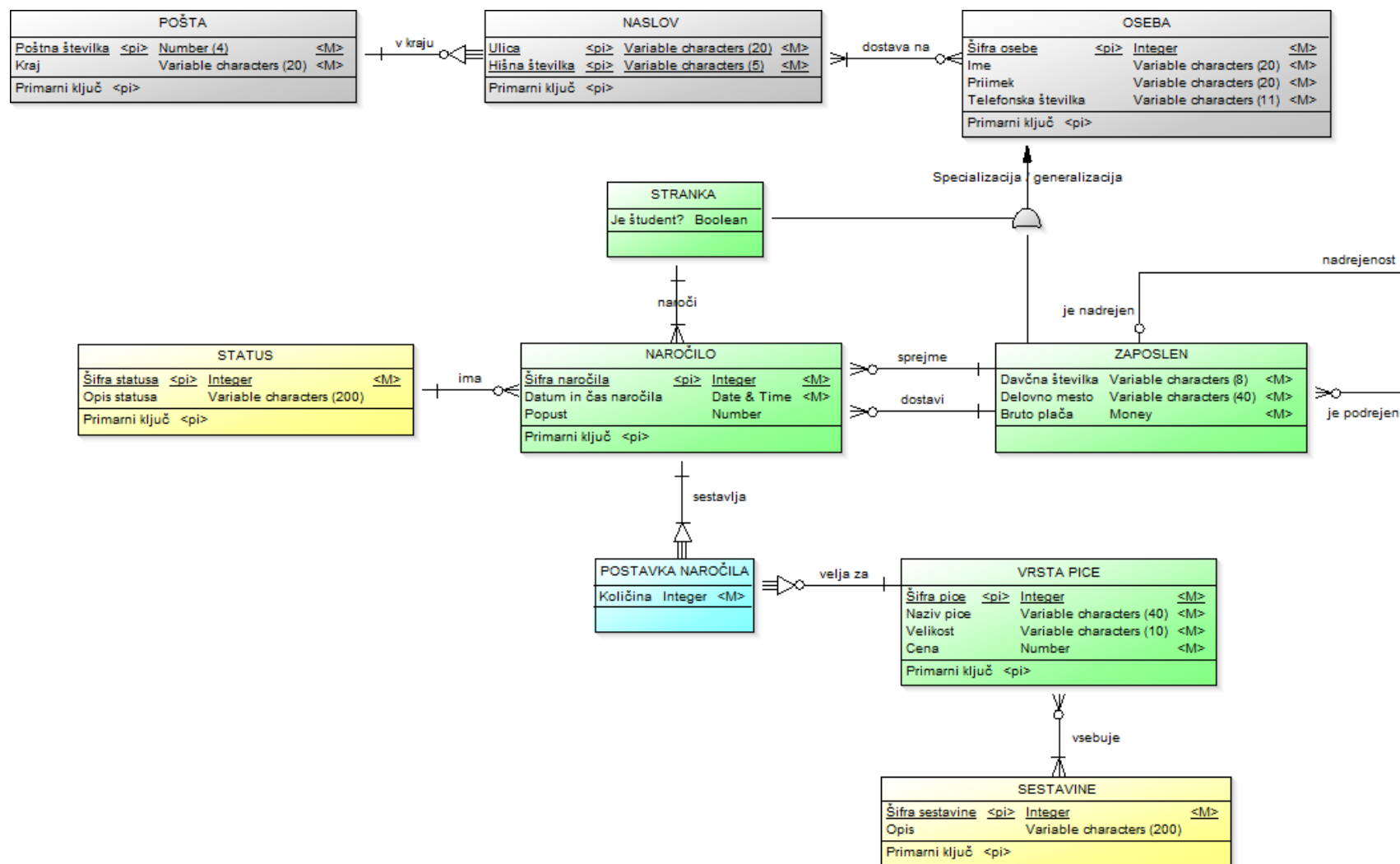
Naloga preslikave konceptualnega modela v  
logični model.





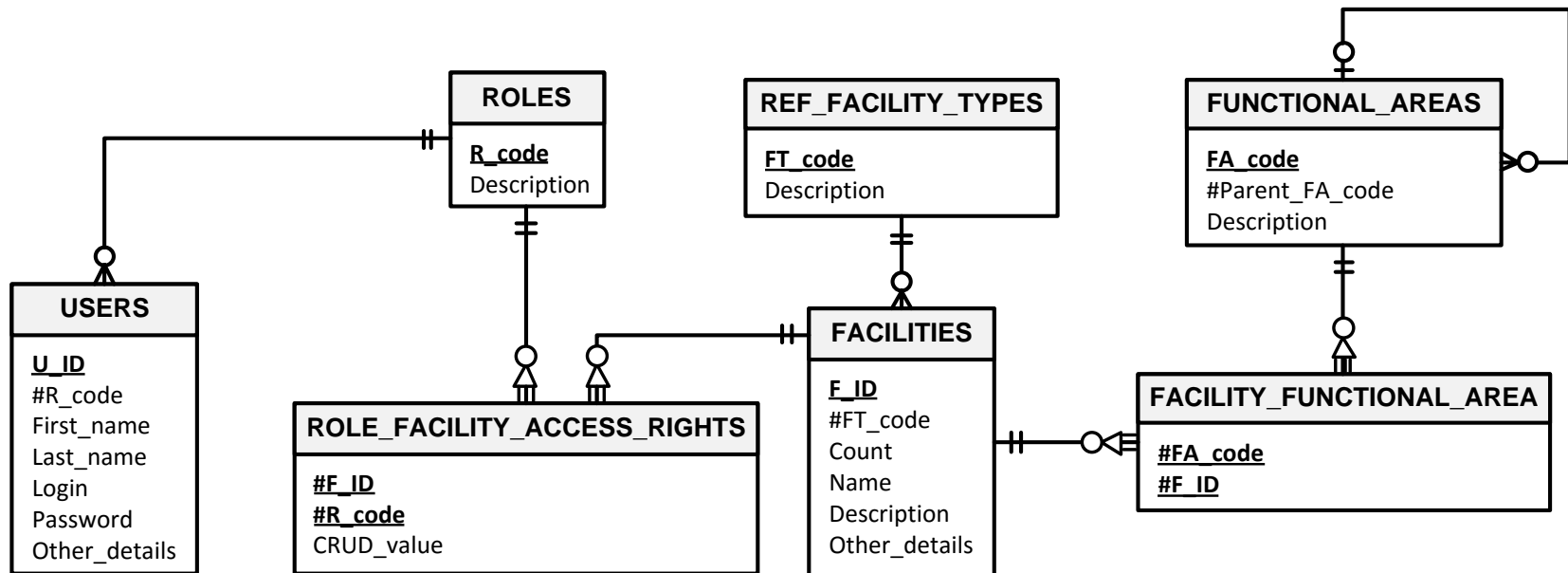
# Picerija

Naloga preslikave konceptualnega modela v logični model.



# Vzvratno inženirstvo

**Reverse engineering** - naloga preslikave logičnega modela v konceptualni model.



**ROLES** (R\_code, Description)

**REF\_FACILITY\_TYPES** (FT\_code, Description)

**FUNCTIONAL\_AREAS** (FA\_code, #Parent\_FA\_code, Description)

**USERS** (U\_ID, #R\_code, First\_name, Last\_name, Login, Password, Other\_details)

**FACILITIES** (F\_ID, #FT\_code, Count, Name, Description, Other\_details)

**ROLE\_FACILITY\_ACCESS\_RIGHTS** (#F\_ID, #R\_code, CRUD\_value)

**FACILITY\_FUNCTIONAL\_AREA** (#FA\_code, #F\_ID)

# Koraki logičnega načrtovanja

(K2.2) (1)

- **K2.2: Preveri relacije z normalizacijo**
  - **Normalizacija** je postopek s katerim pridemo do množice primerno strukturiranih relacij, ki ustrezajo kriteriju normalne oblike (NO).
    - Funkcionalne odvisnosti.
    - Ključni relacije.
    - Problemi zaradi redundance v osnovnih relacijah.

# Koraki logičnega načrtovanja

(K2.2) (2) (funkcionalne odvisnosti)

- **R** ... relacijska shema
- **X** → **Y** ... X funkcionalno določa Y oz. Y je funkcionalno odvisen od X
- Pogoji za funkcionalno odvisnost (FO)
  - V nobeni relaciji iz R **ne obstajata** dve n-terici, ki bi se **ujemali v** vrednostih atributov **X** in se **ne bi ujemali v** vrednostih atributov **Y**.
- Primer:
  - **A** → **B** ne drži, velja pa npr. **B** → **A**

A	B
1	4
1	5
3	7

# Koraki logičnega načrtovanja

(K2.2) (3) (funkcionalne odvisnosti)

- Primer relacijske sheme
  - **Zaposlenec** (Sifra\_zaposlenca, Ime, Naslov)
- Primarni ključ enolično določa ostale attribute
  - **Sifra\_zaposlenca** → Ime
  - **Sifra\_zaposlenca** → Naslov
- Lahko zapišemo tudi v obliki:
  - **Sifra\_zaposlenca** → (Ime, Naslov)
- Trivialne FO ponavadi ne zapisujemo
  - **Sifra\_zaposlenca** → Sifra\_zaposlenca

# Koraki logičnega načrtovanja

(K2.2) (4) (funkcionalne odvisnosti)

- Primer relacijske sheme
  - **Pica** (Sifra\_pice, Naziv\_pice, Velikost\_pice, Cena\_pice, Sifra\_sestavine, Opis\_sestavine)
- Primer nekaj FO
  - $\mathbf{F} \equiv \{ \text{Sifra\_pice} \rightarrow (\text{Naziv\_pice}, \text{Velikost\_pice}), (\text{Sifra\_pice}, \text{Velikost\_pice}) \rightarrow \text{Cena\_pice}, \text{Sifra\_sestavine} \rightarrow \text{Opis\_sestavine} \}$



# Koraki logičnega načrtovanja

(K2.2) (5) (ključi)

- Relacijska shema z atributi  $A_1, A_2, \dots, A_n$  in  $X$  podmnožica te sheme.
- **$X$  je ključ**, če:
  - $X \rightarrow A_1, A_2, \dots, A_n$
  - ne obstaja  $X'$ , kjer  $X' \subset X$  in  $X' \rightarrow A_1, A_2, \dots, A_n$
- Kandidat za ključ je vsak  $X$ , ki relacijo enolično določa.
- Shema lahko ima več ključev, izberemo najprimernejšega, ki postane **primarni ključ**.

# Koraki logičnega načrtovanja

(K2.2) (6) (ključi)

- Primer določanja kandidatov za primarni ključ, ki enolično določajo relacijo.

Zaposlenec
Sifra_zaposlenca
EMSO
Ime
Naslov

← **Oba** atributa  
Sifra\_zaposlenca in  
EMSO **enolično**  
**določata relacijo.**

- Načrtovalec se nato odloči**, kateri izmed obeh kandidatov bo postal primarni ključ.

# Koraki logičnega načrtovanja

(K2.2) (7) (redundanca)

- Cilj načrtovanja PB je grupiranje atributov v relacije na takšen način, da je **čim manj redundance med podatki**.

Sifra_pice	Naziv_pice	Velikost_pice	Cena_pice	Sifra_sestavine	Opis_sestavine
15	Vražja	velika	8 €	12	tabasko
15	Vražja	velika	8 €	37	feferoni
17	Vražja	majhna	6 €	15	šunka
18	Morska	majhna	7 €	3	tuna
19	Morska	velika	9 €	12	tabasko

ponavljajoči atributi

# Koraki logičnega načrtovanja

(K2.2) (8) (anomalije)

- Odvečni podatki v relacijah povzročajo ažurne anomalije (npr. dodajanje).

Sifra_pice	Naziv_pice	Velikost_pice	Cena_pice	Sifra_sestavine	Opis_sestavine
15	Vražja	velika	8 €	12	tabasko
15	Vražja	velika	8 €	37	feferoni
17	Vražja	majhna	6 €	15	šunka
18	Morska	majhna	7 €	3	tuna
19	Morska	velika	9 €	12	tabasko

Če želimo **dodati novo pico** in ji dodati več sestavin, moramo vedno dodati še vse podrobnosti pice.

# Koraki logičnega načrtovanja

(K2.2) (9) (anomalije)

- Odvečni podatki v relacijah povzročajo ažurne anomalije (npr. brisanje).

Sifra_pice	Naziv_pice	Velikost_pice	Cena_pice	Sifra_sestavine	Opis_sestavine
15	Vražja	velika	8 €	12	tabasko
15	Vražja	velika	8 €	37	feferoni
17	Vražja	majhna	6 €	15	šunka
18	Morska	majhna	7 €	3	tuna
19	Morska	velika	9 €	12	tabasko

Če **zbrišemo** majhno Vražjo **pico**, izgubimo podatek o sestavini šunka, ki nastopa samo pri tej pici.

# Koraki logičnega načrtovanja

(K2.2) (10) (anomalije)

- Odvečni podatki v relacijah povzročajo ažurne anomalije (npr. spreminjanje).

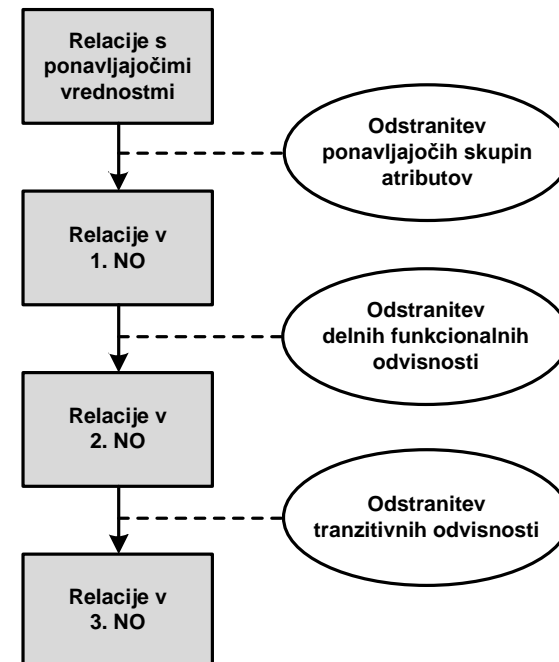
Sifra_pice	Naziv_pice	Velikost_pice	Cena_pice	Sifra_sestavine	Opis_sestavine
15	Vražja	velika	8 €	12	tabasko
15	Vražja	velika	8 €	37	feferoni
17	Vražja	majhna	6 €	15	šunka
18	Morska	majhna	7 €	3	tuna
19	Morska	velika	9 €	12	tabasko

Če **spremenimo** naziv sestavine, moramo to storiti pri vseh picah, sicer pride do nekonsistentnosti.

# Koraki logičnega načrtovanja

(K2.2) (11) (normalizacija)

- **Normalizacija** je postopek preoblikovanja relacij v obliko, pri kateri do ažurnih anomalij ne more priti.
- Poznamo več stopenj normalnih oblik (NO), mi bomo spoznali:
  - **1. NO**, **2. NO** in **3. NO** ter
  - **4. PNO** (četrta poslovna normalna oblika)



# Koraki logičnega načrtovanja


(K2.2) (12) (normalizacija v 1. NO)

- **1. normalna oblika (1. NO)**
  - Potrebni pogoji
    - **Nima ponavljajočih skupin** (atributi niso več vrednostni)
    - Opređeljene **funkcionalne odvisnosti** in **primarni ključ**.



# Koraki logičnega načrtovanja

(K2.2) (13) (normalizacija v 1. NO)

- Primer relacije
  - **Voznik** (Ime, Priimek, St\_dovoljenja, Posta, Kraj, Datum\_in\_ura, Znesek, St\_tock )  
  
Prekrški voznika
- Koraki normalizacije v 1. NO
  1. **Odpravimo ponavljajoče skupine**
    - Voznik (Ime, Priimek, St\_dovoljenja, Posta, Kraj, Datum\_in\_ura, Znesek, St\_tock)

# Koraki logičnega načrtovanja

(K2.2) (14) (normalizacija v 1. NO)

- **Voznik** (Ime, Priimek, St\_dovoljenja, Posta, Kraj Datum\_in\_ura, Znesek, St\_tock)
- Koraki normalizacije v 1. NO
  2. **Določimo funkcionalne odvisnosti**
    - $\text{St\_dovoljenja} \rightarrow (\text{Ime}, \text{Priimek}, \text{Posta}, \text{Kraj})$
    - $(\text{St\_dovoljenja}, \text{Datum\_in\_ura}) \rightarrow (\text{Znesek}, \text{St\_tock})$
    - $\text{Posta} \rightarrow \text{Kraj}$
  3. **Določimo ključ**
    - **Voznik (Ime, Priimek, Posta, Kraj, St\_dovoljenja, Datum in ura, Znesek, St\_tock)**

# Koraki logičnega načrtovanja

(K2.2) (15) (normalizacija v 2. NO)

- **2. normalna oblika (2. NO)**
  - Potrebni pogoji
    - Relacija je v **1. NO**.
    - **Ni parcialnih odvisnosti** (odvisnosti od dela primarnega ključa)
  - Zadostni pogoj
    - Če ima relacija  $n$  atributov in je **primarni ključ** sestavljen iz **1** ali  **$n$  atributov**.

# Koraki logičnega načrtovanja

(K2.2) (16) (normalizacija v 2. NO)

- **Voznik** (Ime, Priimek, Posta, Kraj, St\_dovoljenja, Datum\_in\_ura, Znesek, St\_tock)
- Koraki normalizacije v 2. NO
  1. **Atribute, delno odvisne od ključa prenesemo v novo relacijo in dodamo odvisne dele ključa.**
    - **Voznik (Ime, Priimek, Posta, Kraj, St\_dovoljenja)**
      - St\_dovoljenja → (Ime, Priimek, Posta, Kraj)
    - **Prekrsek (#St\_dovoljenja, Datum\_in\_ura, Znesek, St\_tock)**
      - (St\_dovoljenja, Datum\_in\_ura) → (Znesek, St\_tock)

# Koraki logičnega načrtovanja

(K2.2) (17) (normalizacija v 3. NO)

## ◉ 3. normalna oblika (3. NO)

- ◉ Potrebni pogoji
  - ◉ Relacija je v **2. NO**.
  - ◉ **Ni tranzitivnih funkcionalnih odvisnosti** (med atributi, ki niso del primarnega ključa ni odvisnosti).
- ◉ Zadostni pogoj
  - ◉ Če ima relacija  $n$  atributov in je **primarni ključ** sestavljen iz  **$n-1$**  ali  **$n$  atributov**.

# Koraki logičnega načrtovanja

(K2.2) (18) (normalizacija v 3. NO)

- **Voznik** (Ime, Priimek, Posta, Kraj, St\_dovoljenja)
- **Prekrsek** (#St\_dovoljenja, Datum\_in\_ura, Znesek, St\_tock)
- Koraki normalizacije v 3. NO
  1. **Odstranimo tranzitivne odvisnosti med atributi, ki niso del primarnega ključa.**
    - **Lokacija** (Posta, Kraj)
    - **Voznik** (Ime, Priimek, #Posta, St\_dovoljenja)
      - Posta → Kraj
    - **Prekrsek** (#St\_dovoljenja, Datum\_in\_ura, Znesek, St\_tock)

# Koraki logičnega načrtovanja

(K2.2) (19) (normalizacija v 4. PNO)

- **4. poslovna normalna oblika (4. PNO)**
  - Potrebni pogoji
    - Relacija je v **3. NO**.
    - **Atributi** so **odvisni** od primarnega ključa in **od vrednosti ključa**.
    - Neobvezen prenesen atribut iz druge relacije, ki je v celoti odvisen od ključa je obvezen.

# Koraki logičnega načrtovanja

(K2.2) (20) (normalizacija v 4. PNO)

- Vozniku dodamo nove attribute
  - **Voznik** (Ime, Priimek, #Posta, St\_dovoljenja, Podjetje, Obstojece\_tocke)  

Zasebni

Poklicni
- Koraki normalizacije v 4. PNO
  1. Na podlagi odvisnosti od vrednosti ključa določimo nove relacije.
    - **Voznik** (Ime, Priimek, #Posta, St\_dovoljenja)
    - **Poklicni\_voznik** (#St\_dovoljenja, Podjetje)
    - **Zasebni\_voznik** (#St\_dovoljenja, Obstojece\_tocke)



# Račun

Naloga iz normalizacije na podlagi podatkov iz „flat file“ oblike.

# Naloga

Št. računa	Datum računa	Št. kupca	Naziv kupca	Znesek računa	Št. izdelka	Naziv izdelka	Količina	Cena	Vrednost
023/05	12. 01. 2005	02-100	INTAL	50	22001	Vijak 8	500	0,04	21
					22005	Vijak 10	50	0,06	29
035/05	18. 01. 2005	01-230	BPS	125	22001	Vijak 8	1.000	0,04	42
					22008	Vijak 12	500	0,08	41
					22010	Vijak 20	250	0,17	42
042/05	22. 01. 2005	02-100	INTAL	58	22005	Vijak 10	1.000	0,06	58

**Tabela predstavlja podatke o računih,** vendar ne ustreza pogojem za relacijo.

Predstavite jo v obliki relacije in normalizirajte do 3. NO!

# Reverz

Naloga iz normalizacije na podlagi dokumenta in podanih funkcionalnih odvisnosti.

## Dokument reverza

---

Številka reverza : **ID\_reverza**  
 Datum : **Datum**  
 Iz skladišča : **ID\_skladisca**  
                                   **Ime\_skladisca**  
                                   **ID\_poste**   **Ime\_poste**  
 Izdal referent : **ID\_referenta**  
                                   **Ime\_referenta**  
 Datum vrnitve blaga : **Datum\_vrnitve**

Artikli

<b>ID_artikla</b>	<b>Ime_artikla</b>	<b>Kratika_ME</b>	<b>Dav_sk</b>	<b>Tarifa</b>	<b>Kolicina</b>
-------------------	--------------------	-------------------	---------------	---------------	-----------------

---

## Funkcionalne odvisnosti

---

ID_reverza → Datum	ID_artikla → Ime_artikla
ID_reverza → ID_skladisca	ID_artikla → ID_ME
ID_reverza → Datum_vrnitve	ID_artikla → ID_Dav_sk
ID_reverza → ID_referenta	ID_ME → Kratica_ME
ID_reverza → ID_poste	ID_Dav_sk → Dav_sk
ID_referenta → Ime_referenta	ID_Dav_sk → ID_tarife
ID_poste → Ime_poste	ID_tarife → Tarifa
ID_skladisca → Ime_skladisca	
ID_skladisca → ID_poste	

---

# Koraki logičnega načrtovanja (K2.3)

- **K2.3: Preveri relacije s pregledom uporabniških transakcij**
  - Podobno kot korak **K1.8** pri konceptualnem načrtovanju preverimo še logični model.
  - Preverjamo predvsem napake, do katerih je prišlo pri pretvorbi iz konceptualnega modela.

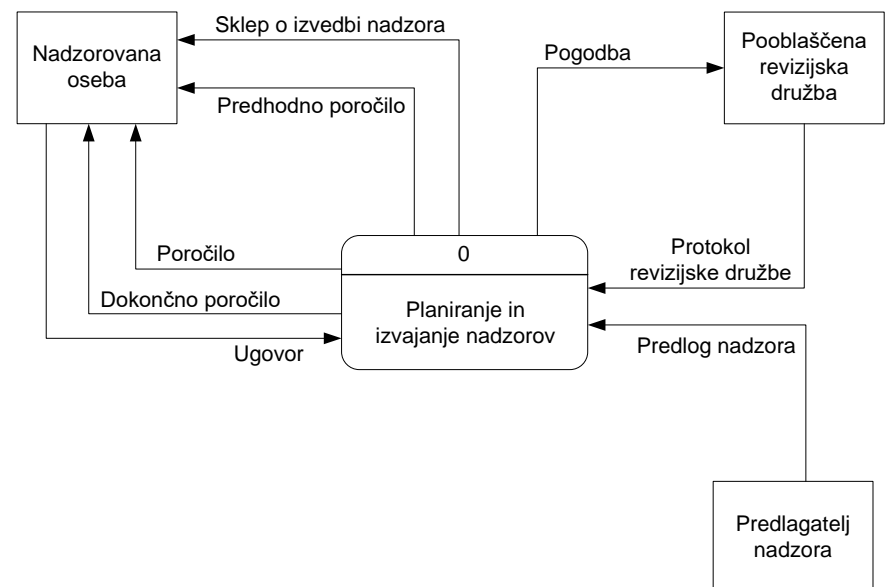
# Koraki logičnega načrtovanja (K2.4)

- K2.4: **Preveri omejitve integritete**
  - Obveznost atributov (**Mandatory**)
  - Omejitve domen atributov
  - Števnost
  - Omejitev entitet in povezav
    - **Foreign Key** Post\_st **references** Posta(Post\_st)  
ON UPDATE **CASCADE** ON DELETE **NO ACTION**  
(prikaz s PD)
  - Splošne omejitve (npr. triggerji)

# Koraki logičnega načrtovanja (K2.5)

## ○ K2.5: Preveri model z uporabnikom

- Ponovno preveriti ali model ustreza uporabniškim zahtevam.
- Pomagamo si lahko z različnimi tehnikami (npr. DFD – Diagram podatkovnih tokov)



# Koraki logičnega načrtovanja (K2.6)

- **K2.6: Združi lokalne modele v globalni model (opcijsko)**
  - Pri združevanju pogosto pride do neskladnosti, zato je potrebno globalni model ponovno preveriti tako kot lokalne.



# Koraki logičnega načrtovanja (K2.7)

- **K2.7: Preveri zmožnosti modela za razširitve**
  - Model predstavimo do takšne mere generičnosti, da imamo pri morebitnih razširitvah čim bolj enostavno delo.