Računalništvo in informatika

Program: Mehatronika

dr. Hubert Fröhlich, univ. dipl. el.

Podatkovne baze

Podatkovne baze

Podatki

- osnova za odločanje in izvajanje akcij
- tiskana oblika
- elektronska oblika (v rač., na pom. medijih)
- so temelj vseh informacijskih sistemov

- Obseg podatkov!
- “Informacijska onesnaženost” (zanesljivi/nezanesljivi pod.)

- Iskanje podatkov in popularni iskalniki

Podatkovna baza

Štiri zahteve (ANSI – 1975)

1. Povezanost in urejenost podatkov
2. Sočasnost uporabe (en ali več uporabnikov)
3. Enoličnost podatkov (podaki se ne ponavljajo
4. Shranjena v računalniku (elektronska oblika)

Modeli v podatkovnih bazah

Model pove na kakšen način so podatki v bazi organizirani. Poznamo:

- Hierarhične
- Mrežne
- Objektne
- Relacijske
- Večdimenzionalni (OLAP)

Danes se najpogosteje uporablja relacijski model in OLAP

Podatkovne baze – relacijski model

- Matematično je to model, ki podatkovno bazo obravnava kot zbirko predikatov na končni množici spremenljivk, {x | P(x)} in opisov omejitev možnih vrednosti in kombinacij vrednosti spremenljivk.
- Relacijski model omogoča snovalcem PB da ustvarijo konsistentno logično predstavitev informacij
- Organizacija podatkov v PB na način, ki minimizira redundanco podatkov v PB se imenuje normalizacija PB. Njen osnovni namen je, da razstavi relacije, ki vsebujejo anomalije v manjše dobro strukturirane relacije. Za to predpisuje t.i. normalne oblike (normal form), ki so hierarhično urejene

Podatkovne baze - osnovni pojmi

- **RELACIJA**: je podatkovna struktura, ki ima naslovno vrstico in neurejeno množico n-teric istega tipa. Običajno jo predstavimo kot tabelo. V relaciji nimamo podvojenih vrednosti.
- **ATRIBUT**: je par (ime, tip)
- **N-terica**: je podatkovna struktura, ki je sestavljena iz enega ali več atributov.
- **Primarni ključ**: je en ali več atributov, ki enoumno določajo posamezne n-terice. S primarnim ključem dosežemo, da ni podvojevanja zapisov. Primer – EMŠO, ISBN … Poznamo tudi sekundarni ključ in tuji ključ.
- **Kardinalnost**: (številnost) pove moč množice. V RPB jih uporabljamo tudi za izražanje odnosov med tabelami (relacijami). Možni so: 1 – 1, 1- mnogo, mnogo – mnogo
- **Funkcionalna odvisnost**: Dva atributa sta funkcionalno odvisna, če iz poznavanja enega lahko določimo vrednost drugega (vrednost atributa A je odvisna od vrednosti atributa B)
- **Tranzitivna odvisnost**: je posredna funkcionalna odvisnost(vrednost atributa A je odvisna od vrednosti B, vrednost B pa od vrednosti C -> A je tranzitivno odvisen od C, ne moremo pa izvedeti vrednosti A direktno iz C!!!)

Relacija - primer

Ime tabele (relacije) = Dijaki

Atributi tabele Dijaki:

IDDijak, Priimek, Ime, Razred, Telefon

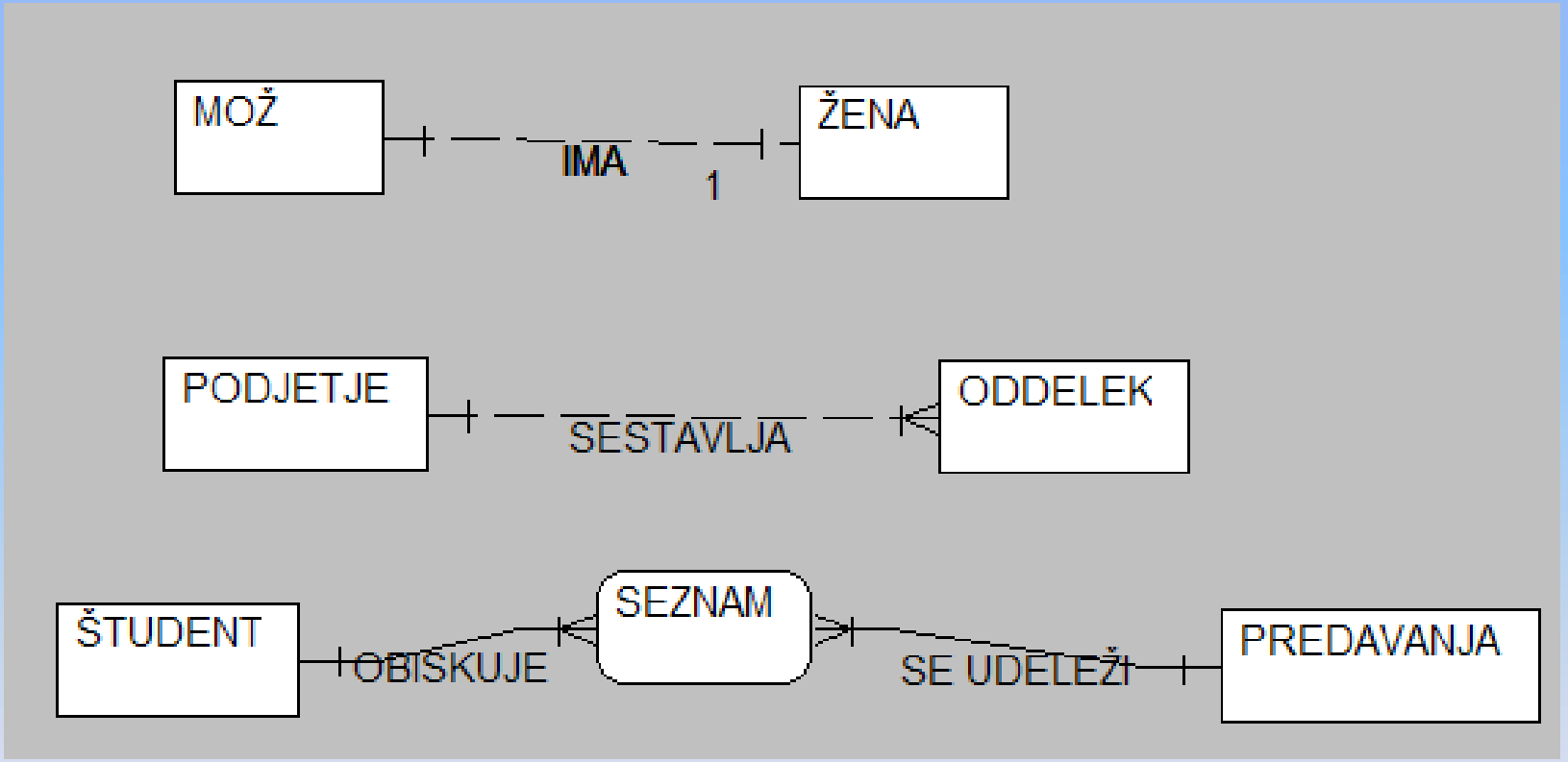
Dijaki				
IDDijak	Priimek	Ime	Razred	Telefon
10205	Mlinar	Mateja	G2A	01-123-333
10301	Dolenc	Mitja	G2A	03-313-313
10305	Verk	Marija	G2C	NULL
10309	Žavbi	Jana	G2B	NULL
10310	Juh	Polona	G2C	01-111-111

Podatkovni tipi in integritetne omejitve atributov:

- IDDijak = TEXT(5); zahtevan vnos 5 števk, vrednost se v tabeli ne ponovi
- Priimek = TEXT(20); Besedilo dolžine največ 20 znakov
- Ime = TEXT(10); besedilo dolžine največ 10 znakov
- Razred = TEXT(3); zahteva vnos ČrkaŠtevkaČrka; "G2A"|"G2B"|"G2C"|null
- Telefon = TEXT(11) ; besedilo dolžine največ 11 znakov

Za primarni ključ bi tu lahko izbrali IDDijak

Kardinalnost relacij - primeri



Primer relacije s podvajanjem vrednosti

Delavec (DelavecID, Priimek, OddelekID, ImeOddelke, KrajOddelka)

Delavec				
Delavec ID	Priimek	OddelekID	ImeOddelka	KrajOddelka
104	Perko	3	Prodaja	Ljubljana
180	Medved	2	Finance	Maribor
192	Volk	2	Finance	Maribor
197	Pihlar	3	Prodaja	Ljubljana
199	Kalin	2	Finance	Maribor

Anomalija pri dodajanju podatkov

Delavec				
Delavec ID	Priimek	OddelekID	ImeOddelka	KrajOddelka
104	Perko	3	Prodaja	Ljubljana
180	Medved	2	Finance	Maribor
192	Volk	2	**Fonance**	Maribor
197	Pihlar	3	Prodaja	Ljubljana
199	Kalin	2	Finance	Maribor

- Ne moremo dodati oddelka v katerem ni še nihče zaposlen
- Z napačnim vnosom smo porušili integriteto podakov, imamo dva oddelka z istim OddelekID
- Ko bomo iskali vse zaposlene v oddleku Finance, Volk ne bo izpisan
- Za avtomatizacijo preverjanja vnosov je treba napisati dodatno programsko kodo

Anomalije pri brisanju podatkov

Delavec				
Delavec ID	Priimek	OddelekID	ImeOddelka	KrajOddelka
104	**Perko**	**3**	**Prodaja**	**Ljubljana**
180	Medved	2	Finance	Maribor
192	Volk	2	Finance	Maribor
197	**Pihlar**	**3**	**Prodaja**	**Ljubljana**
199	Kalin	2	Finance	Maribor

- Lahko izgubimo podatke – če izbrišemo vse zaposlene v oddelku 3 smo hkrati izgubili tudi podatke o oddelku 3

Anomalije pri spreminjanju podatkov

Delavec				
Delavec ID	Priimek	OddelekID	ImeOddelka	KrajOddelka
104	Perko	3	Prodaja	Ljubljana
180	Medved	2	Finance	**Kranj**
192	Volk	2	Finance	**Kranj**
197	Pihlar	3	Prodaja	Ljubljana
199	Kalin	2	Finance	**Maribor**

- Lahko izgubimo integriteto podatkov
- Da jo zagotovimo je treba napisati programsko kodo – zamudno in kompliciran

Normalizacija podatkov

Cilji:

- Integriteta podatkov
- Odpravljanje redundance podatkov
- Enostavne poizvedbe na “univerzalen način” (E.F.Codd)
- Zmanjšati potrebo po restrukturiranju relacij pri uvajanju novih tipov podatkov

Normalizacija podatkov I

- 1NO (ponavljajoče skupine) – Relacija je v 1NO, če ne vsebuje sestavljenih atributov. Vsak atribut mora biti elementaren, ne pa množica vrednosti (primer: ime in priimek)
- 2NO (delne odvisnosti) – Relacija je v 2NO, če je v 1NO in so hkrati vsi atributi odvisni od celega primarnega ključa. Noben atribut ne sme biti odvisen samo od dela ključa (v primeru, da je ta sestavljen iz večih atributov). Primer:

  Postavke_računa(**Št_računa, Šifra_artikla,** Naziv_Artikla, Količina**)** ->

  Postavke_računa(**Št_računa, Šifra_artikla,** Količina**),**

  Artikli(Š**ifra_artikla,** Naziv_Artikla**)**

Normalizacija podatkov II

- 3NO (tranzitivne odvisnosti) – Relacija je v 3NO če je v 2NO in hkrati noben atribut ni tranzitivno odvisen od ključa (vsi so funkcionalno odvisni) in hkrati noben atribut v relaciji ni odvisen od nobenega drugega atributa kot samo od ključa.

  Račun(**Številka računa,** Datum, Šifra kupca, ime_kupca, naslov_kupca**)** ->

  Račun(**Številka računa,** Datum, Šifra_kupca**)**

  Kupec(**Šifra kupca,** ime_kupca, naslov_kupca**)**

Sistem za upravljanje baze podatkov - SUPB

Je vmesnik med uporabnikom in PB in ga sestavlja zbirka programov, ki skrbijo za delovanje in vzdrževanje PB. Njegove naloge:

- Kreiranje podatkovnih struktur (DDL – kreiranje, spreminjanje in brisanje tabel, baz …) – primer: SQL
- Vzdrževanje podatkovne baze (DML – branje, spreminjanje, vstavljanje, brisanje podatkov.) Primer je SQL
- Zaščito podatkov pred neavtoriziranim dostopom (user level security)
- Izvajanje transakcij

Pravila ACID za izvajanje transakcij

ACID = Atomicity, Consistency, Isolation, Durability

- Atomicity – vsaka transakcija je atomska, torej nedeljiva. Če se ne izvede do konca, se mora PB vrniti v stanje pred začetkom transakcije (roll-back). Velja pravilo vse ali nič.
- Consistency – vsaka transakcija mora ohranjati konsistentnost PB. To pomeni da mora ohranjati tipe podatkov, relacije in kardinalnosti. Če transakcija krši pravila, jih SUPB lahko vsili ali naredi roll-back
- Isolation - Nobena operacija ne sme vplivati na podatke, ki so trenutno v obdelavi znotraj transakcije. Podatki, ki so predmet transakcije so zaklenjeni dokler se transakcija ne konča.
- Durability – Zagotavlja, da bodo podatki, ki jih je neka transakcija ustvarila, po končanju transakcije obdržali svoje vrednosti tudi če je kasneje prišlodo napake ali okvare. Najpogosteje je to realizirano z dnevnikom transakcij, ki omogoča kasnejše restavriranje konsistentnega stanja DB pred nastopom okvare. Transakcija šteje za končano šele ko je vpisana v dnevnik (log).