



Poglavje III

Podatkovni modeli

- Logični podatkovni modeli
 - Tradicionalni nerelacijski modeli
 - Hierarhični model
 - Mrežni model
 - Sodobni nerelacijski modeli
 - Relacijski model
- Podrobneje o relacijskem podatkovnem modelu



Opisovanje in shranjevanje podatkov v PB



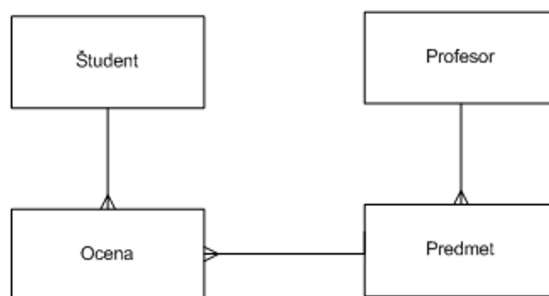
- Model, s katerim opišemo, kaj bi želeli hraniti ter kakšne povezave obstajajo med elementi, ki jih želimo hraniti, je podatkovni model.
- Podatkovni model je način, kako na visoki ravni abstrakcije opišemo podatke, ki jih želimo hraniti, ter skrijemo nepomembne podrobnosti.
- Podatkovni model lahko odraža uporabnikovo zaznavanje realnega sveta oziroma uporabnikovo predstavo, kako naj bodo podatki shranjeni.
- Torej gre za naše razumevanje oz. predstavo podatkov.

Logični podatkovni modeli



- Logični podatkovni model je formaliziran jezik za opis podatkov, ki ga razume ciljni SUPB.
- Poznamo več vrst logičnih modelov:
 - Hierarhični podatkovni model
 - Hierarhični SUPB: IBM **IMS**
 - Mrežni podatkovni model
 - Mrežni SUPB: **IDS** (Integrated data store, GE) in **IDMS** (Integrated Database Management System, BF Goodrich)
 - Relacijski podatkovni model
 - Relacijski SUPB: **MariaDB/MySQL, PostgreSQL, SQLite, IBM DB2, Oracle, MS SQL Server, MS Access,...**
 - Objektni podatkovni model
 - Objektni SUPB: **Objectstore, Versant,...**
 - Objektno-relacijski (hibridni SUPB): **IBM DB2, Informix, Oracle, ...**
 - NoSQL: razni podatkovni modeli, npr. ključ-vrednost, dokumentni, grafni, ...

Zakaj podatkovni modeli



- študentu pripada množica osebnih podatkov (ime, priimek, EMŠO, vpisna št. ...)
- en študent ima lahko več ocen (pri različnih predmetih) - razmerje tipa **eden proti več**
- en profesor lahko predava več predmetov
- pri enem predmetu imamo lahko več ocen (za različne študente)

Kako poljubno zapletene omejitve znotraj posameznega problema predstaviti na kar se da splošen in poenoten način?

Zakaj podatkovni modeli

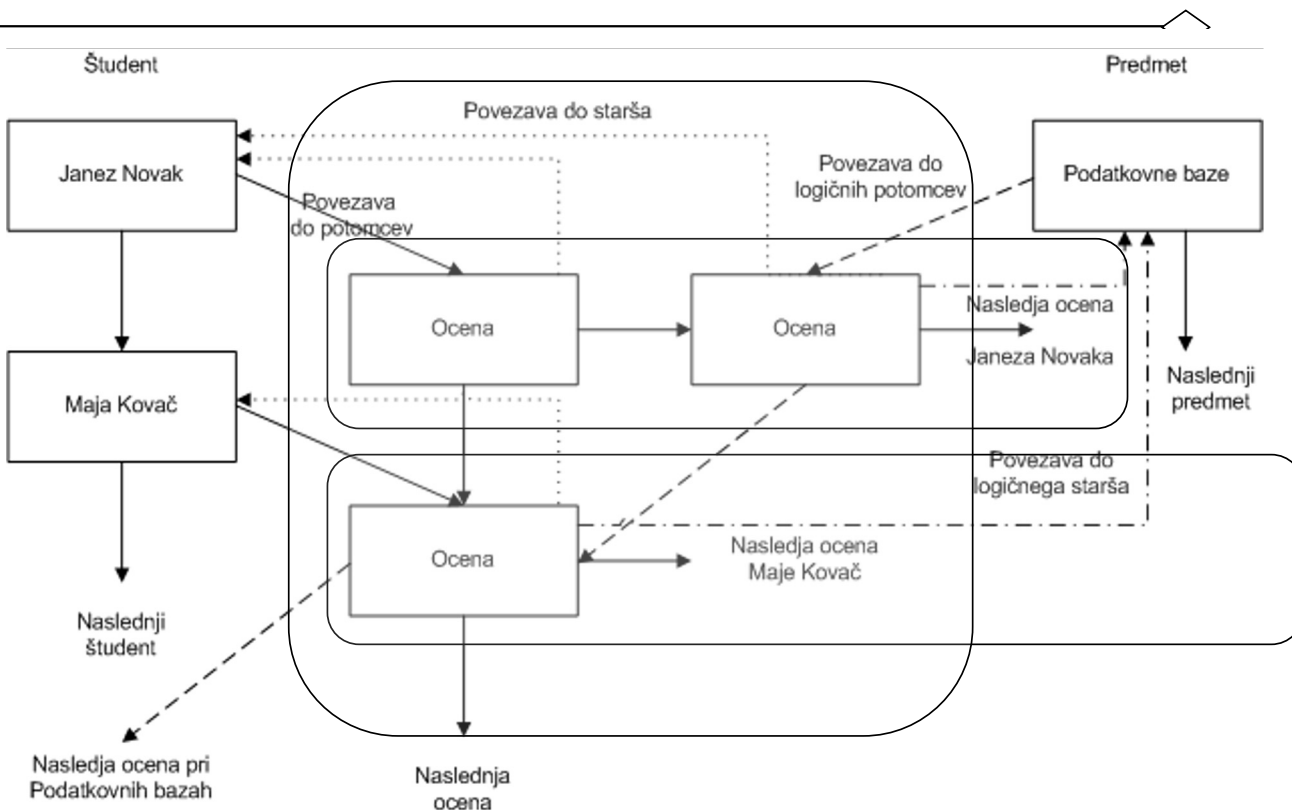


1. Za vsak problem uporabiti specializirano aplikacijo, ki ima **zakodirano** poznavanje problema in njegovih omejitev (klasične aplikacije, temelječe na datotečnem sistemu)
2. Zasnovati **splošen princip** (podatkovni model), s katerim bomo lahko predstavili (skoraj) poljubne omejitve znotraj katerega koli (skoraj) poljubnega problema

Hierarhični podatkovni model

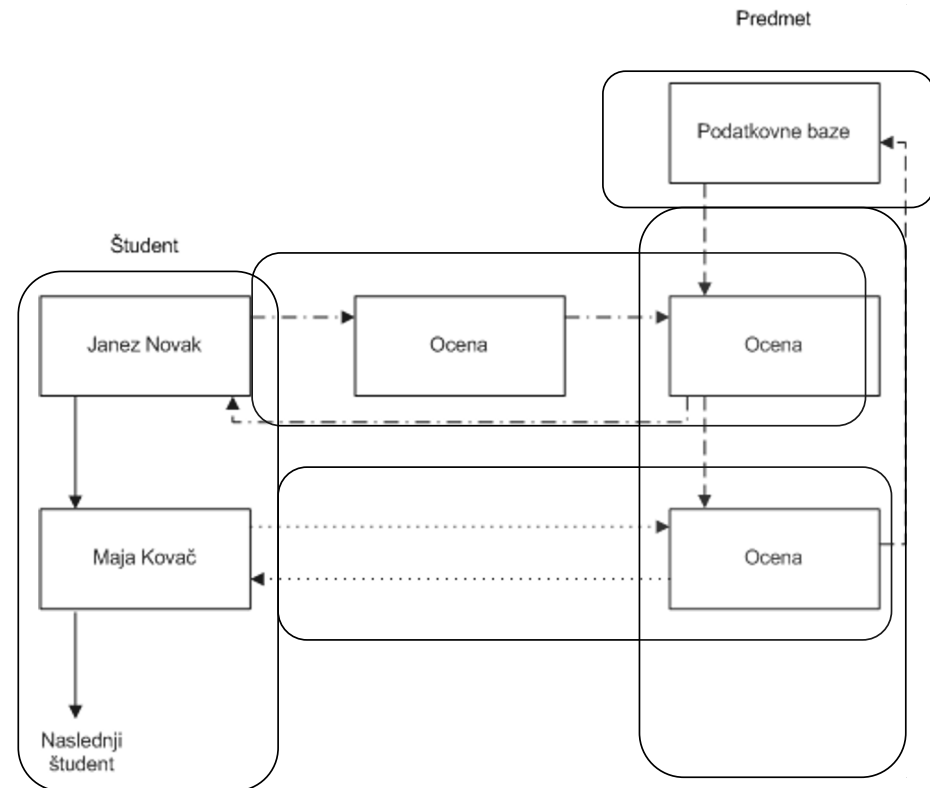
- Hierarhija kot drevo oz. acikličen graf
- Podatkovna baza kot množica (gozd) hierarij
- Povezava eden proti več med staršem in potomcem
- Vozlišča (zapisi v datotečnem sistemu) so povezana s fizičnimi kazalci
- Sistem IMS (IBM)
- Primer: študentu pripada več ocen

- Vsak izmed staršev je povezan z množico potomcev
- Vsak potomec ima povezavo na svojega starša



Mrežni podatkovni model

- Povezave (tipa eden proti več) implementiramo kot množice
- Množice implementirane v obliki krožnega seznama
- Kateri kazalec pripada kateri množici
- Učinkovita implementacija, vendar manj kot hierarhični podatkovni model
- Implementacija povezav s fizičnimi kazalci med zapisi!
- Zapletena administracija
- Sistema IDS (Bachman; GE), IDMS (BF Goodrich, Cullinae, Computer Associates)





Relacijski podatkovni model



- Pojavi se leta 1970, predlaga ga Edgar Codd.
- Odgovor na prekomplicirane starejše podatkovne modele
- PB, ki temelji na relacijskem modelu, je implementirana kot množica relacij.
- Vsaka relacija je predstavljena kot tabela z vrsticami (elementi) in stolpci (atributi).
- Delo s podatki zasnovano na relacijskih in množičnih operacijah
- Je relativno enostaven za razumevanje:
 - Tudi neizkušeni lahko razumejo vsebino podatkovne baze;
 - Na voljo so enostavni vendar močni jeziki za poizvedovanje po vsebini PB.

O relacijskem podatkovnem modelu



- Osnovne prednosti:
 - je definiran formalno in osnovan na dobro definiranih matematičnih strukturah - relacijah;
 - ne vsebuje elementov fizičnega shranjevanja podatkov, s čimer je zagotovljena podatkovna neodvisnost;
 - relacije so predstavljive s tabelami, ki so človeku lahko razumljive.

Relacijski podatkovni model

Povezave med podatki niso realizirane s fizičnimi povezavami, ampak preko vrednosti atributov (stolpcev) v relacijah (tabelah)

Priimek	Ime	Vpisna številka	Letnik	Smer
Novak	Janez	63123456	2	3
Kovač	Marija	67894321	1	0
.
.
.
.
.

Tabela Študent

Vpisna številka	Šifra predmeta	Število ponavljanj	Datum	Ocena
63123456	63707	1	16. 6. 2010	5
67894321	63707	1	1. 2. 2010	9
63123456	63707	2	10. 2. 2011	7
.
.
.
.

Tabela Predmet

Ime predmeta	Šifra predmeta	.
Podatkovne baze	63707	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.

Tabela Ocena



Terminologija pri relacijskem modelu...

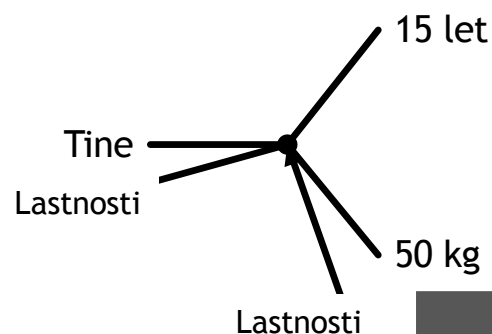


- Pri relacijskem modelu definiramo naslednjo terminologijo:
 - Relacija
 - Atribut
 - Domena
 - N-terica
 - Stopnja relacija
 - Števnost relacije
 - Relacijska PB

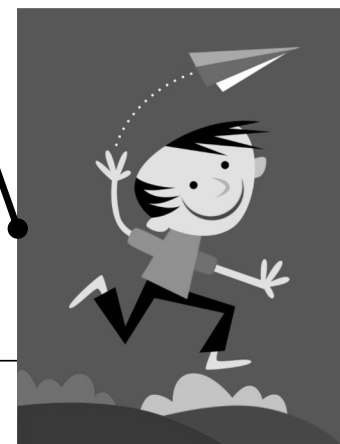
Terminologija pri relacijskem modelu...

- Relacijo si lahko predstavljamo kot dvodimenzionalno tabelo s stolpci in vrsticami.
 - Velja za logično strukturo podatkovne baze in ne za fizično.
 - Torej: tabela kot predstavitev relacije

Ime	Starost (v letih)	Teža (v kg)
Tine	15	50
Meta	20	45
Jure	40	80
Ana	5	10



Relacija




Terminologija pri relacijskem modelu...



- Atribut je poimenovan stolpec relacije.
- Vsak atribut pripada določenemu podatkovnemu tipu (domeni)

Atribut relacije



Ime	Starost (v letih)	Teža (v kg)
Tine	15	50
Meta	20	45
Jure	40	80
Ana	5	10

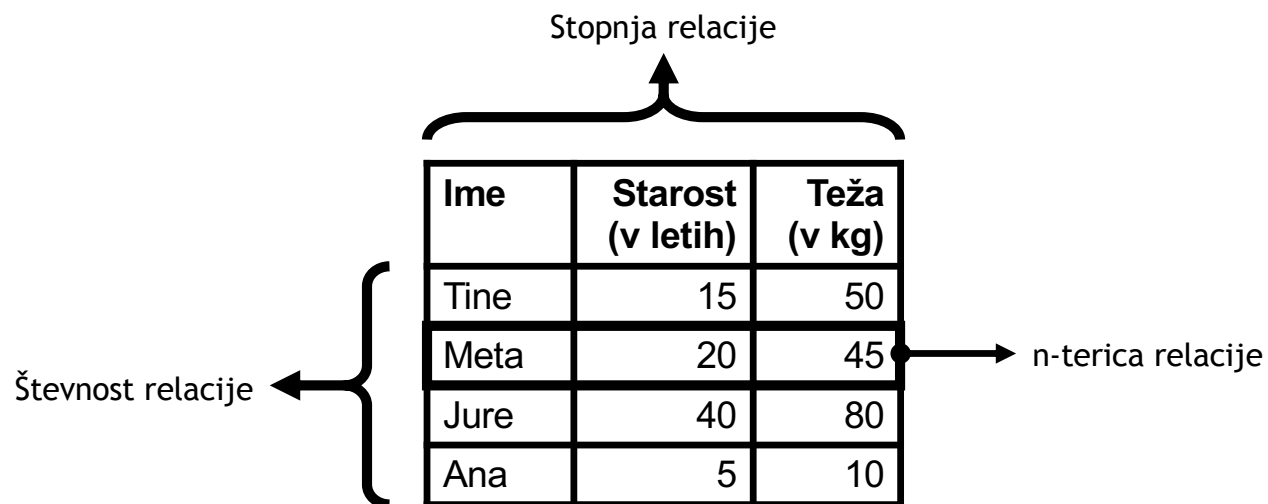
Terminologija pri relacijskem modelu...



- Domena je množica dovoljenih vrednosti enega ali več atributov, ki so vključeni v to domeno.
- Primeri domen:
 - Starost v letih: celo število 0-200
 - EMŠO: 13-mestno število s posebno strukturo
 - Spol: m ali ž
 - Ime: niz znakov dolžine do 50
 - Poštna številka: štirimestno število iz množice vseh poštnih števil v Sloveniji

Terminologija pri relacijskem modelu...

- N-terica (terka) je ena vrstica v relaciji.
- Števnost relacije je število n-teric relacije.
- Stopnja relacije je število atributov v relaciji.



Terminologija pri relacijskem modelu



- Relacijska podatkovna baza je množica normaliziranih relacij z enoličnimi imeni.
- Normalizirane relacije so relacije s posebej zaželenimi lastnostmi (spoznali bomo kasneje).

Pomen izraza relacija



- Relacija v dobesednem (opisovalnem) pomenu:
 - Elementi relacije (objekti, vrstice) izpolnjujejo določene pogoje
 - Oseba(Ime, Starost, Teža):
 - Oseba(Tine, 15, 50)
 - Oseba(Meta, 20, 45)
 - Oseba(Jure, 40, 80)
 - Tine, 15, 50 so vrednosti (opis lastnosti), ki pripadajo isti osebi
- Relacija v prenesenem (povezovalnem) pomenu:
 - Elementi v vrstici relacije (tabele) določajo n-terice drugih tabel, ki so med seboj v nekem razmerju
 - Uporaba za povezovanje elementov drugih relacij (tabel) med seboj
 - Zakonec(Oseba1, Oseba2): Zakonec(Meta, Jure)
 - Osebi Meta in Jure sta zakonca

Pomen izraza relacija



Zakonec

Ime	Starost (v letih)	Teža (v kg)
Tine	15	50
Meta	20	45
Jure	40	80
Ana	5	10

Oseba

Oseba: opisovalna relacija
Zakonec: povezovalna relacija



Matematična definicija relacije



- Relacija, ki si jo predstavljamo kot tabelo, je
 - matematična relacija stopnje n nad domenami atributov
 - podmnožica kartezičnega produkta domen atributov.

$$r \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times \dots \times \text{dom}(A_n))$$



Relacijska shema: opis (oblike) relacije



- Relacijsko shemo sestavlja oznaka sheme R ter lista oznak atributov A_i s pripadajočimi oznakami domen D_i :

$$R(A_1:D_1, A_2:D_2, \dots, A_n:D_n)$$

- Vsaki relaciji pripada natanko ena relacijska shema, eni relacijski shemi pa lahko več relacij.
- Relacijska shema predstavlja semantiko ali pomen relacije.
- Predstavimo jo kot glavo tabele

Relacijska shema...



- Primer relacijske sheme ...

Študent(VpŠt, Ime, Priimek, Pošta, Kraj, Spol);

ali

Študent(VpŠt: number(8), Ime: char(20), Priimek: char(20),
Pošta: number(4), Kraj: char(30), Spol: char(1));

... in relacije

VpŠt	Ime	Priimek	Pošta	Kraj	Spol
24010632	Marko	Bric	5270	Ajdovščina	M
25089888	Iztok	Jerin	2000	Maribor	M
24135344	Maja	Klepec	1000	Ljubljana	Ž
24090909	Anita	Terčelj	4000	Kranj	Ž



Lastnosti relacij



- Ime relacije je enolično. V shemi podatkovne baze ni dveh relacij z enakim imenom.
- Vsaka celica tabele, ki predstavlja relacijo, vsebuje natančno eno atomarno vrednost.
- Vsak atribut relacije ima enolično ime. V isti relaciji ni dveh atributov, ki bi imela isto ime.
- Vrednosti nekega atributa so vse iz iste domene.

Lastnosti relacij



- Vsaka n-terica relacije je enolična → v relaciji ni dveh enakih n-teric.
- Vrstni red atributov v relaciji je nepomemben.
- Vrstni red n-teric v relaciji je nepomemben.
- Vsaki relaciji pripada natanko ena relacijska shema, eni relacijski shemi pa lahko več relacij.

Primeri

Ime	Starost (v letih) in teža (v kg)
Tine	Starost 15 Teža 50
Meta	Starost 20 Teža 45
Jure	Starost 40 Teža 80
Ana	Starost 5 Teža 10

→ Celice ne vsebujejo atomarnih vrednosti

Oseba	Telefon
Tine Mikuž	1 47 68 819; 041 467 766
Ana Pregelj	05 36 61 234; 05 36 61 235

→ Celice vsebujejo več vrednosti



Omejitve v relaciji (odvisnosti)

- Relacija je model nekega stanja v svetu: njena vsebina ne more biti poljubna.
- Realne omejitve ne omogočajo, da bi bili odnosi v svetu kakršnikoli; možna so le določena stanja.
- Odvisnosti so sredstvo, s katerim lahko v relacijskem modelu povemo, katere vrednosti relacij so veljavne in katere sploh ne morejo obstajati.

Funkcionalne odvisnosti...



- Poznamo več vrst odvisnosti:
 - Funkcionalne odvisnosti (functional dependency)
 - Večvrednostne odvisnosti (multivalued dependency)
 - Stične odvisnosti (join dependency)

- Obravnavali bomo samo funkcionalne odvisnosti

- Poslovna pravila (business rules) kot poseben primer odvisnosti (izvor bolj ali manj splošnih omejitev)

Funkcionalne odvisnosti...



- Predpostavimo, da obstaja relacijska shema R z množico atributov, katere podmnožici sta X in Y .
- V relacijski shemi R velja $X \rightarrow Y$ (X funkcionalno določa Y oziroma Y je funkcionalno odvisen od X), če v nobeni relaciji, ki pripada shemi R , ne obstajata dve n -terici, ki bi se ujemali v vrednostih atributov X in se ne bi ujemali v vrednostih atributov Y .
- Z besedami: obstaja neka funkcija f , s pomočjo katere lahko izračunamo vrednosti Y če poznamo vrednosti X , torej $f: X \rightarrow Y$ oziroma $Y=f(X)$

Primeri funkcionalnih odvisnosti

- Imamo relacijo s shemo

Stevila(X1, X2, Y)

$F = \{f1, f2, f3\} = \{X1 \ X2 \rightarrow Y, X1 \ Y \rightarrow X2, X2 \ Y \rightarrow X1\}$

X1	X2	Y
5	2	7
7	3	10
4	4	8
12	11	23

X1	X2	Y
5	2	3
7	3	4
4	4	0
12	11	1

X1	X2	Y
5	2	10
7	3	21
4	4	16
12	11	121

- Kakšne so konkretne funkcionalne odvisnosti f1, f2 in f3 v zgornjih primerih?

Primeri funkcionalnih odvisnosti



- Imamo relacijo s shemo ...

Oseba(Ime, Priimek, PostnaSt, ImeKraja, ImeUlice, HisnaSt, Spol, EMSO, RojstniDatum)

- Kaj pomeni ta shema? Katere odvisnosti veljajo?

EMSO \rightarrow RojstniDatum

EMSO \rightarrow Spol

EMSO \rightarrow RojstniDatum, Spol (krajše)

PostnaSt \rightarrow ImeKraja



ImeUlice \rightarrow PostnaSt
Ime \rightarrow Spol
RojstniDatum \rightarrow EMSO
Spol \rightarrow EMSO



Uporaba funkcionalnih odvisnosti



- Opredelitev omejitev znotraj relacij, ki pripadajo isti relacijski shemi
- Vse relacije, ki pripadajo isti relacijski shemi, imajo isto množico funkcionalnih odvisnosti
- Opredelitev ključev s funkcionalnimi odvisnostmi



Ključni relacije...



- Ker je relacija množica n-teric, se v njej vse n-terice med seboj razlikujejo.
- Če v shemi nastopajo funkcionalne odvisnosti za sklicivanje na posamezno n-terico ni potrebno poznati vseh vrednosti atributov n-terice.
- Množici atributov, ki določajo vsako n-terico, pravimo ključ relacije oziroma pravilneje ključ relacijske sheme.

Ključni relacije...



- Predpostavimo, da obstaja
 - relacijska shema R z atributi $\{A_1, A_2, \dots, A_n\}$
 - neka podmnožica atributov $X \subseteq \{A_1, A_2, \dots, A_n\}$
- Množica $X \subseteq \{A_1, A_2, \dots, A_n\}$ je ključ relacijske sheme oziroma vseh njej pripadajočih relacij, če sta izpolnjena naslednja dva pogoja:
 - (1) $X \rightarrow A_1 A_2 \dots A_n$
 - (2) ne obstaja X' , ki bi bila prava podmnožica X in ki bi tudi funkcionalno določala $A_1 A_2 \dots A_n$

Ključí relacije...

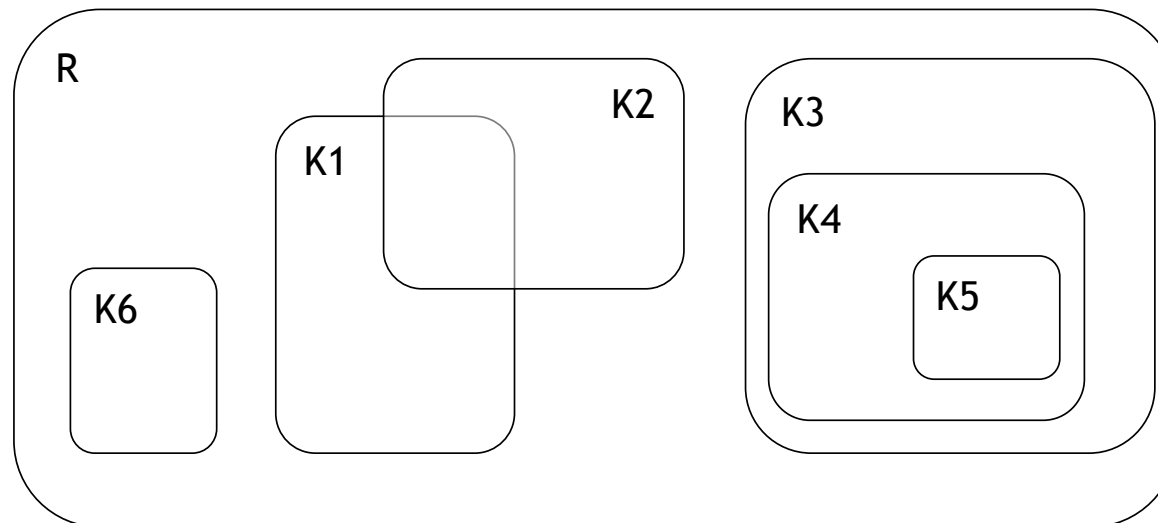


- Poznamo več kategorij ključev:
 - Kandidat za ključ (key candidate)
 - Primarni ključ (primary key)
 - Nadključ (superključ, superkey)
 - Tuji ključ (foreign key)

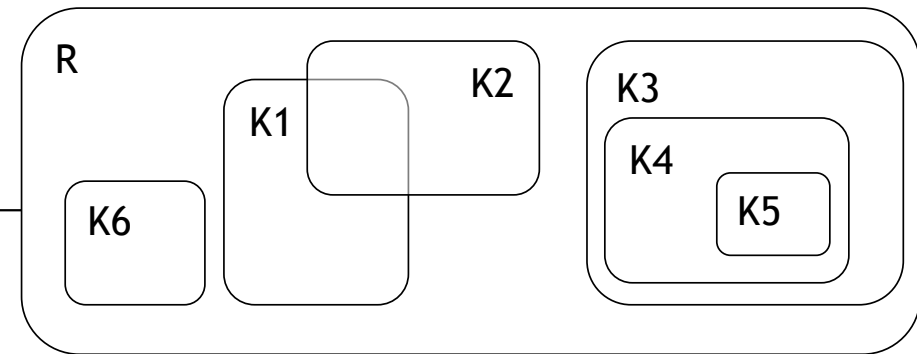
Ključí relacije...



- Kandidat za ključ je vsaka podmnožica atributov relacije, ki ima lastnosti ključa.
- Naj bodo K1, K2 in K4 vsi kandidati za ključe sheme R
- Kaj lahko rečemo o R (vsi atributi), K3, K5 in K6?



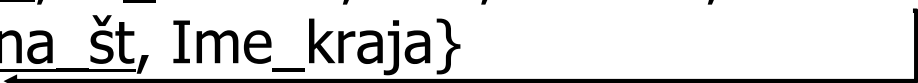
Ključni relacije



- Primarni ključ (ali samo ključ) je tisti kandidat za ključ, ki ga izberemo za enolično identifikacijo n-teric relacij v podatkovni bazi. Primer: K1
- Nadključ je vsaka množica atributov, v kateri je vsebovan vsaj en kandidat za ključ
⇒ kandidat za ključ je podmnožica nadključa.
Primer: K3, R
- Tuji ključ je množica atributov, v okviru ene relacije, ki je enaka kandidatu za ključ neke druge (lahko pa tudi iste) relacije. Nastopa v relacijah povezovalnega pomena.

Ključí relácie (primeri)

R1 = {EMŠO, Št_osebne, Ime, Priimek, #Poštna_št}
R2 = {Poštna_št, Ime_kraja}



▪ Kategorije ključev:

- Kandidat za ključ (key candidate): EMŠO, Št_osebne, Poštna_št (v R2)
- Primarni ključ (primary key): EMŠO, Poštna_št (v R2)
- Nadključ (superključ, superkey): npr. {EMŠO, Ime, Priimek}
- Tuji ključ (foreign key): Poštna_št (v R1)

Primeri ključev



- Relacija: Študent(Ime, Priimek, Naslov, EMŠO, VpŠt)
- Kandidata za (primarni) ključ: {EMŠO}, {VpŠt}
- Primarni ključ: {VpŠt} (izberemo, ker je primernejši za našo rabo)
- Kaj pa {Ime, Priimek, Naslov}? Ali pa {Ime, Priimek, EMŠO}?
- Primarni ključ implementira tudi določeno **pomensko omejitev!**
- Nadključi (izmed vseh možnih):
 - {Ime, Priimek, EMŠO}, {Priimek, VpŠt}, ...
- Zapis ključa kot množica: zavite oklepaje {} običajno izpustimo

Omejitve nad podatki

- Funkcionalne odvisnosti so osnova za zagotavljanje celovitosti in skladnosti podatkov v podatkovni bazi (določajo integritetne omejitve).
 - Poznamo več vrst omejitev:
 - Omejitve domene (Domain constraints)
 - Števnost (Multiplicity)
 - Pravila za celovitost podatkov (Integrity constraints)
 - Celovitost entitet (Entity Integrity)
 - Celovitost povezav (Referential Integrity)
 - Splošne omejitve (General constraints): poslovna pravila
- } Posamezni atributi

} Ključi (vrstice)

} Relacija (tabela)

Števnost in oznaka NULL



- Oznaka NULL:
 - Označuje vrednost atributa, ki je trenutno neznana ali nepomembna za dano n-terico.
 - Gre za nepopolne podatke ali podatke pri izjemnih primerih.
 - Predstavlja odsotnost podatka. Ni enako kot 0 ali prazen niz znakov, ki sta dejanski vrednosti.
 - Oznako NULL lahko dovolimo ali prepovemo

- Oznaka NULL je lahko problematična pri dejanski implementaciji, saj so operacije nad relacijskim podatkovnim modelom osnovane na predikatnem računu prvega reda, kjer sta edini možni vrednosti true in false.

Omejitve (celovitost, integriteta) entitet in povezav



- Omejitve entitet
 - Omejitve na nivoju posamezne n-terice v relaciji (vrstice v tabeli)
 - V osnovni relaciji ne sme imeti noben atribut, ki je del ključa, oznake NULL.

- Omejitve povezav
 - Če v relaciji obstajajo tuji ključi, potem morajo:
 - (a) njihove vrednosti ustrezati tistim, ki so v obliki ključa zapisane v eni izmed n-teric neke druge ali iste relacije
 - (b) ali pa mora biti tuji ključ v celoti označen z NULL.

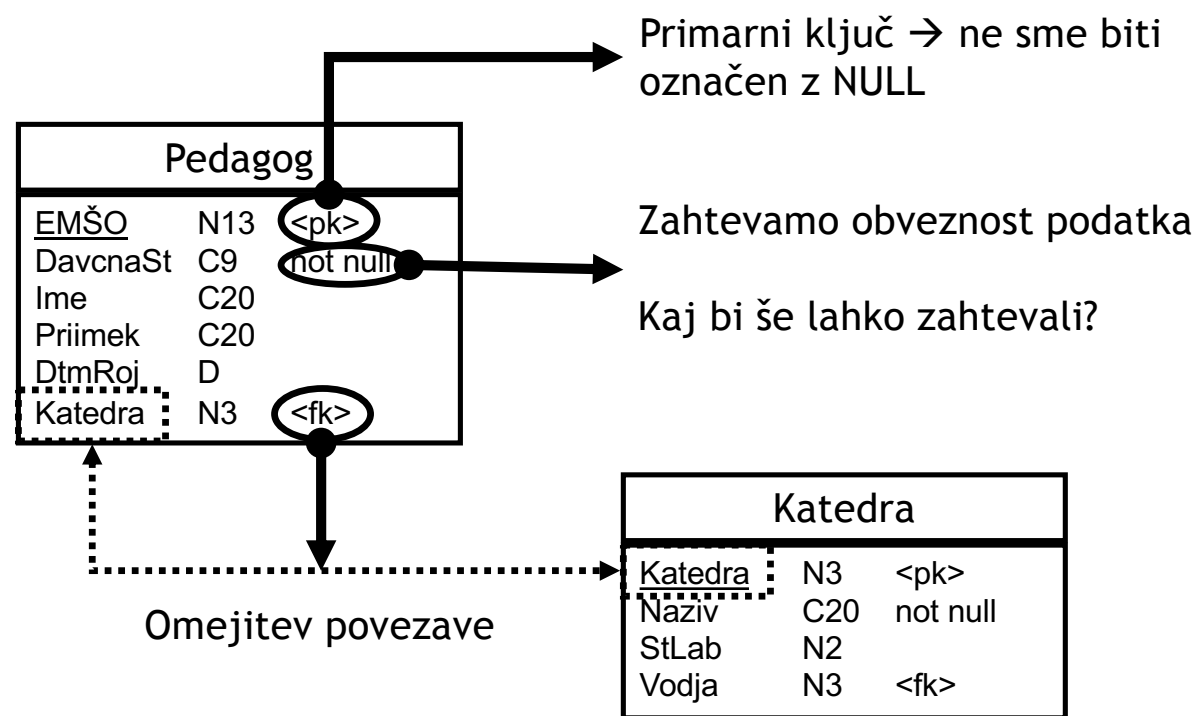
Splošne omejitve



- Splošne omejitve

- Dodatna pravila, ki jih določi uporabnik ali skrbnik podatkovne baze, ki definirajo ali omejujejo nek vidik področja, za katerega je narejena podatkovna baza (poslovna pravila)
- Na nivoju relacije (tabele) ali množice relacij (podatkovne baze)

Primeri omejitev



Splošna omejitev: pedagog (učitelj) lahko predava največ tri predmete.