Katedra za računarstvo
Elektronski fakultet u Nišu

Računarstvo i informatika

Baze podataka

Letnji semestar 2020/2021

O predmetu

Prof.dr Leonid Stoimenov

Baze podataka

- **Šifra: 2OER4O04**
- **Semestar: IV**
- **Fond: 2+2+1**
- **ESPB: 6**

- **Informacije:**

  http://cs.elfak.ni.ac.rs/nastava

  Kurs: Baze podataka

  Šifra za pristup: entitet2021

Nastava

Prof. dr Leonid Stoimenov

Kancelarija: M2-2 (CG&GIS Laboratorija)
E-mail: leonid.stoimenov@elfak.ni.ac.rs
Konsultacije: svakog dana, uz najavu mailom

Doc. dr Aleksandar Stanimirović

Kancelarija: 331 (CG&GIS Laboratorija)
E-mail: aleksandar.stanimirovic@elfak.ni.ac.rs
Konsultacije: svakog dana, uz najavu mailom

Doc. dr Miloš Bogdanović
Doc. dr Nataša Veljković
Milena Frtunić Gligorijević
Darko Puflović

Kancelarija: M2-2 (CG&GIS Lab)
E-mail: {milos.bogdanovic, natasa.veljkovic, milena.frtunic, darko.Puflovic}@elfak.ni.ac.rs

Plan predavanja i vežbi

- **Predavanja**
- Uvod u Baze podataka
- Uvod u DBMS
- Projektovanje baza podataka
- Konceptualni modeli podataka
- Relacioni model podataka

D. ZAD 1 ILI KOLOKVIJUM **T**

- Relaciona algebra
- Funkcionalne zavisnosti
- Normalne forme
- Uvod u transakcionu obradu
- Arhitekture sistema BP

- **Vežbe**
- Projektovanje baza podataka
- Relacioni model podataka
- SQL
  - definisanje podataka
  - ažuriranje podataka
  - manipulacija podacima

D. ZAD 2,3 ILI KOL. **P1**

- Razvoj aplikacija nad bazama podataka
- SQL pogledi i indeksi
- Ugrađeni SQL i ADO.NET
- Windows aplikacije i BP

D. ZAD 4,5 ILI KOL. **P2**

Literatura

- Beleške sa časa
- Materijal dostupan na sajtu predmeta

Knjige

- **L.Stoimenov, Uvod u baze podataka, Elektronski fakultet, 2014, Edicija: Udžbenici**
- **R.Elmasri, S.Navathe, *Fundamentals of Database System*, Addison Wesley, 2010, 6/e**
- **S. Đorđević-Kajan, L. Stoimenov, Praktikum za vežbe na računaru iz predmeta Strukture i baze podataka, II deo: Baze podataka,** Elektronski fakultet u Nišu, 2004, Edicija: Pomoćni udžbenici.

Obaveze studenata u toku semestra

- **Predavanja i vežbe**
  - Obavezno prisustvo - provere, testovi i sl
  - Aktivnost u toku semestra!!
  - Obavezna priprema za lab.vežbe
  - Dozvoljen izostanak sa jedne lab.v.

Predispitne aktivnosti
(u zavisnosti od epidemiološke situacije):

1. Domaći zadaci

- **DZ 1 (za T deo)**
- **DZ 2,3 (kol P1)**
- **DZ 4,5 (kol P2)**

ILI

2. Kolokvijumi:

- **Kol. T (April) - 50% od T dela**
- **Kol. P1 (April), Kol. P2 (Jun), po 50% od P dela**

Završni ispit

- Konačna ocena: T deo: 60%, P deo: 40%
- Ispitni rok:
  - Radi se u terminu za ispit, na papiru!
  - 4 sata za rad, Zadaci i pitanja

    (I) Parcijalni ispit, **uz prethodno položeni kolokvijum**

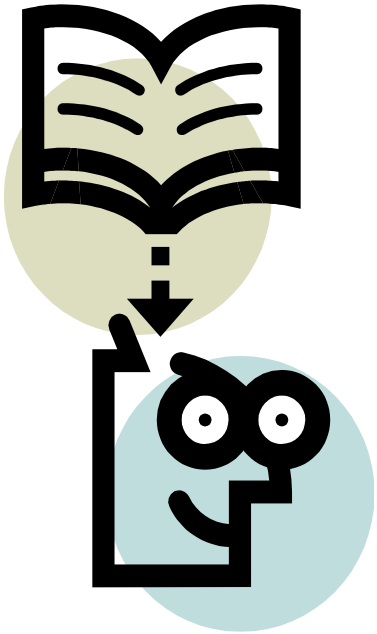
    ili

    (II) Ispit u celini
- Jun, Septembar, Oktobar
- Uslov za polaganje ispita:
  - student koji je sakupio minimum **50% od ukupnog broja poena**

Konačna ocena

- Od **50** do **60.9** poena : **6**
- Od **61** do **70.9** poena : **7**
- Od **71** do **80.9** poena : **8**
- Od **81** do **90.9** poena : **9**
- Od **91** do **100** poena : **10**

Pitanja, ideje, komentari



Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Uvod u Baze podataka

Prof.dr Leonid Stoimenov

Uvod u baze podataka - Pregled

- Osnovni pojmovi
  - Šta je **Podatak, informacija**?
  - Šta je **Baza podataka**?
  - Šta je **Sistem za upravljanje bazama podataka**?
  - Šta je **Aplikacija nad bazom podataka**?
  - Šta je **Sistem baza podataka**?
- Konvencionalna obrada i obrada zasnovana na bazama podataka
- Sistem baza podataka
- Kategorizacija korisnika baza podataka
- Istorijat baza podataka
- Prednosti i nedostaci

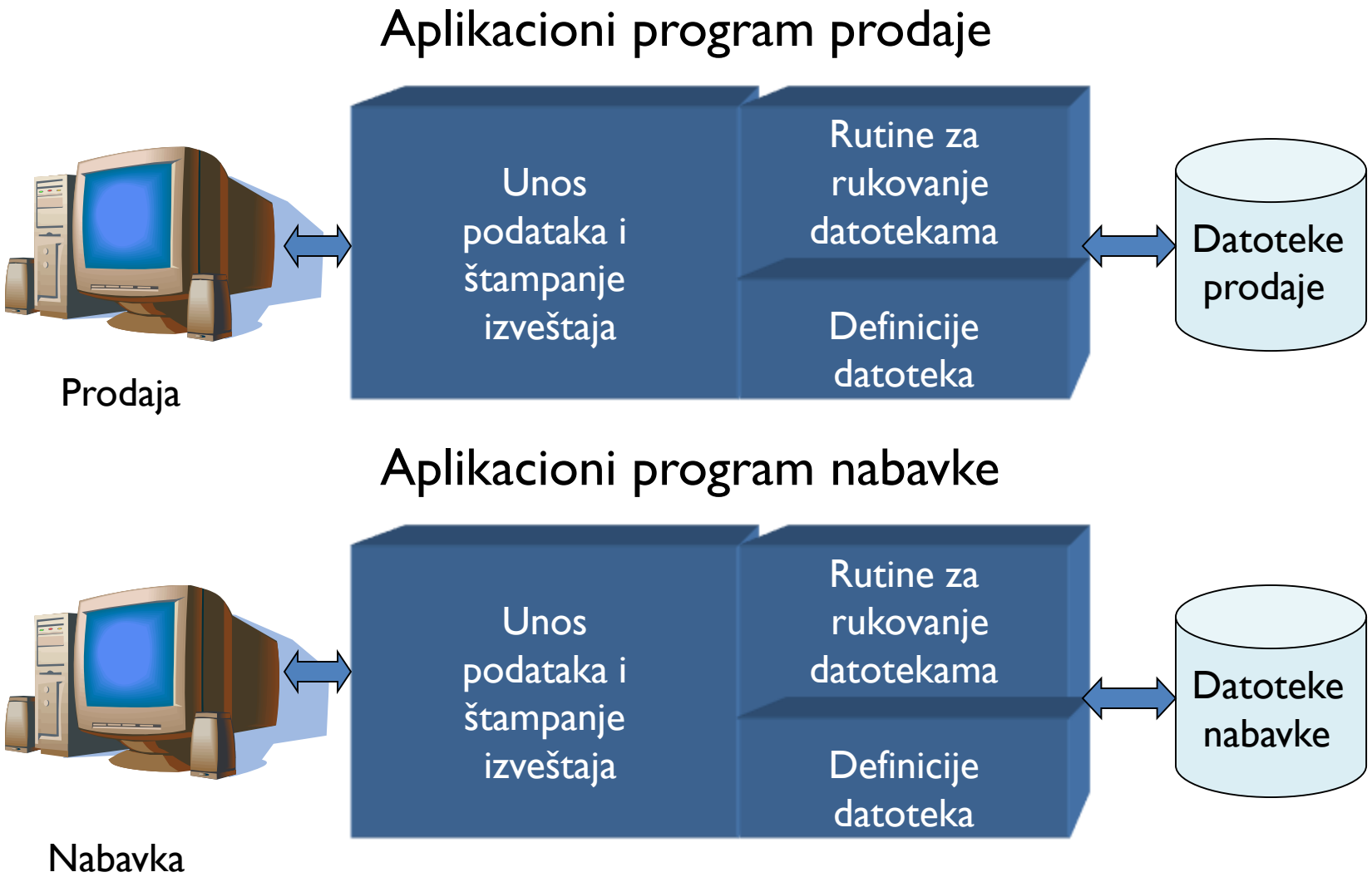
Podatak i informacija

- **Podatak** (lat. *datum* – deo informacije, eng. *Data*)
  - Jednostavna neobrađena izolovana činjenica koja ima neko značenje i koja se obrađuje i čuva na računaru
  - Podaci su znakovni prikaz činjenica i pojmova koji opisuju svojstva objekata
- **Obrada podataka** – proces prevođenja podataka u informacije
- **Informacija** (lat. *Informare*, eng. *Information*) - rezultat analize i organizacije podataka na način da daje/generiše novo znanje
- **Znanje** ...
  - (kurs Veštačka inteligencija, VII semestar)
- **Mudrost** ...

Konvencionalni sistemi

- Tradicionalni pristup (sistem zasnovan na datotekama engl. *File-based system*)
  - *Ad hoc* pristup: datoteka ili skup datoteka podataka potrebnih za definisanu aplikaciju
  - Razvijaju se programi koji obrađuju podatke iz datoteka
  - Svaki program definiše sopstvene podatke i upravlja tim podacima
- Obrada podataka (engl. *Data Processing*)
- Često se koristi pojam Automatska obrada podataka (AOP)
  - pojedine aplikacije jednog informacionog sistema (IS) međusobno nezavisne i kod koje se za svaku aplikaciju kreiraju i održavaju posebne datoteke sa svim potrebnim podacima

Obrada zasnovana na datotekama



Osnovne karakteristike sistema zasnovanog na datotekama

I

- Definicija podataka je ugrađena u aplikacioni program
  - poželjno je da bude odvojena i nezavisna
- Ne postoji kontrola pristupa podacima, osim one koju vrše aplikacioni programi
  - poželjna potpuna kontrola pristupa podacima i to nezavisno od konkretnog aplikacionog programa

Prednosti i nedostaci konvencionalne obrade

- Prednosti
  - Jednostavnost projektovanja i realizacije
- Nedostaci
  - Nepovezanost aplikacija
  - Ponavljanje podataka
  - Neusaglašenost (nekonzistentnost) podataka
  - Čvrsta povezanost programa i podataka
    - Programi za obradu podataka zavise od načina struktuiranja podataka
    - Promena strukture podataka zahteva promenu programa
  - Ograničena mogućnost zajedničkog korišćenja podataka
  - Ograničena raspoloživost podataka
  - Neadekvatna realizacija oporavka od pada sistema

Posledice:

- Obrada podataka je skupa

Rešenje problema: baza podataka

- Baza podataka treba da obezbedi:
  - Nezavisnost struktura podataka od programa koji ih obrađuju,
  - Nezavisnost programa od struktura podataka;
  - Minimalnost ponavljanja podataka;
  - Da obrada podataka nije vezana za programski jezik opšte namene, već za viši "upitni" jezik;
  - Korišćenje baze podataka od strane većeg broja korisnika.

Šta je baza podataka?

- **Baza podataka** *(engl. Data Base, **Database**):*
  - Integrisana kolekcija podataka o nekoj organizaciji
    - **Primer organizacija**: kompanija, banka, fakultet, grad, biblioteka, supermarket
- **Primer:** Fakultetska baza podataka
  - Obuhvata podatke o:
    - **Entitetima – objekti realnog sveta**
      - studijski programi,
      - studenti,
      - predmeti,
      - učionice
    - **Vezama** među ovim entitetima,
      - student sluša predmet,
      - predmet se izodi u učionici,
      - predmet pripada studijskom programu

Baza podataka kao kolekcija podataka

- **Baza podataka** predstavlja **kolekciju povezanih** podataka organizovanih u logičke celine predstavljene **tabelama**.
- Skup podataka pripremljen tako da se mogu jednostavno **koristiti**, tj. pregledati, pretraživati, sortirati, upoređivati, itd., ali i menjati.
- Podaci u bazama podataka su organizovani u **dvodimenzionalne tabele** ili **relacije**
- **Relacione** baze podataka
  - Skup relacija/tabela
  - Tabela može da ima više **kolona**, gde svaka kolona predstavlja neku **osobinu** ili **atribut**.
  - **Vrste** tabele čine konkretni podaci, odnosno konkrente vrednosti osobina/atributa nekog objekta.

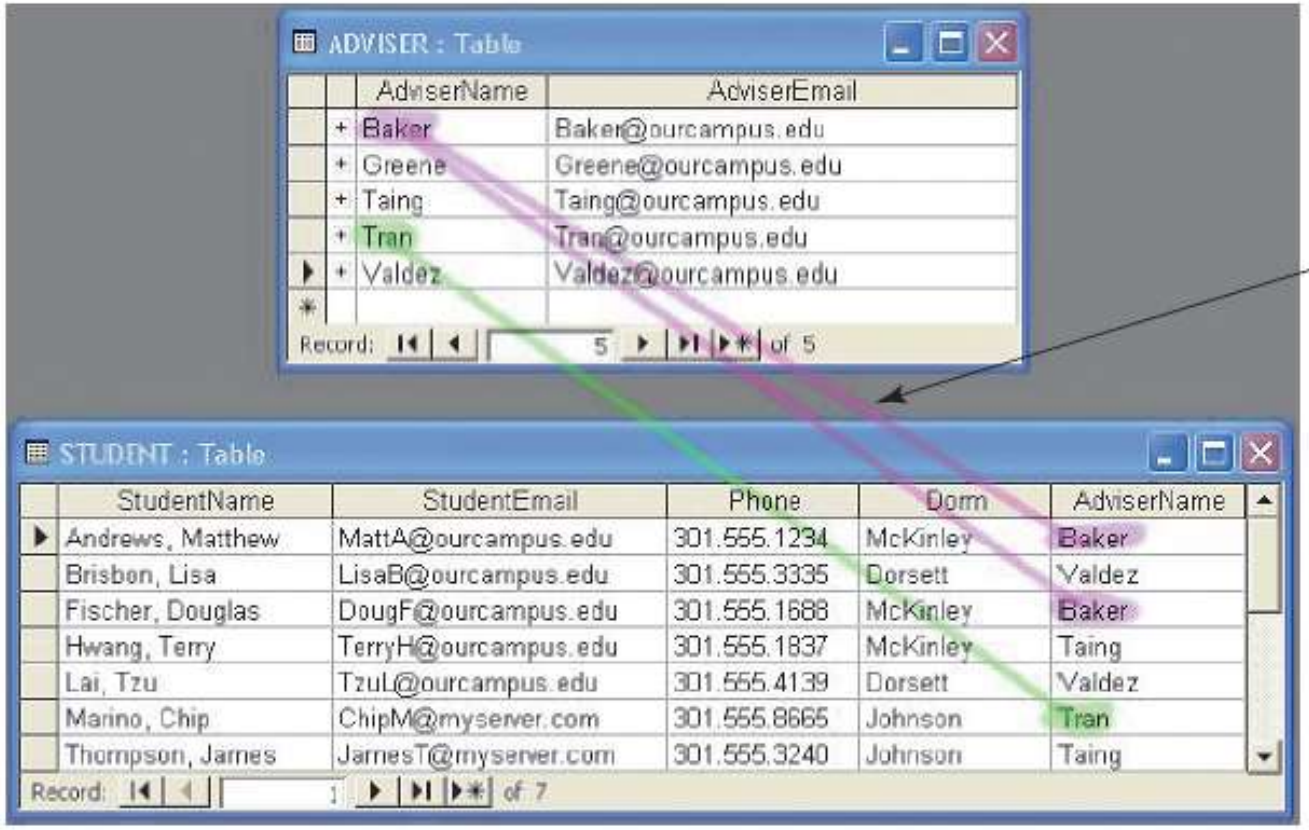
Podaci u bazi podataka

- Dvodimenzionalna tabela (slika: tabela Student)
  - **Kolone** - osobine tj atributi objekta realnog sveta
  - **Vrste** – predstavlja jedan slog podataka o nekom objektu

Ime	Prezime	Indeks	MBR
Petar	Petrović	1111	123456
Milan	Milanović	2222	654321
Jovan	Jovanović	3333	345612

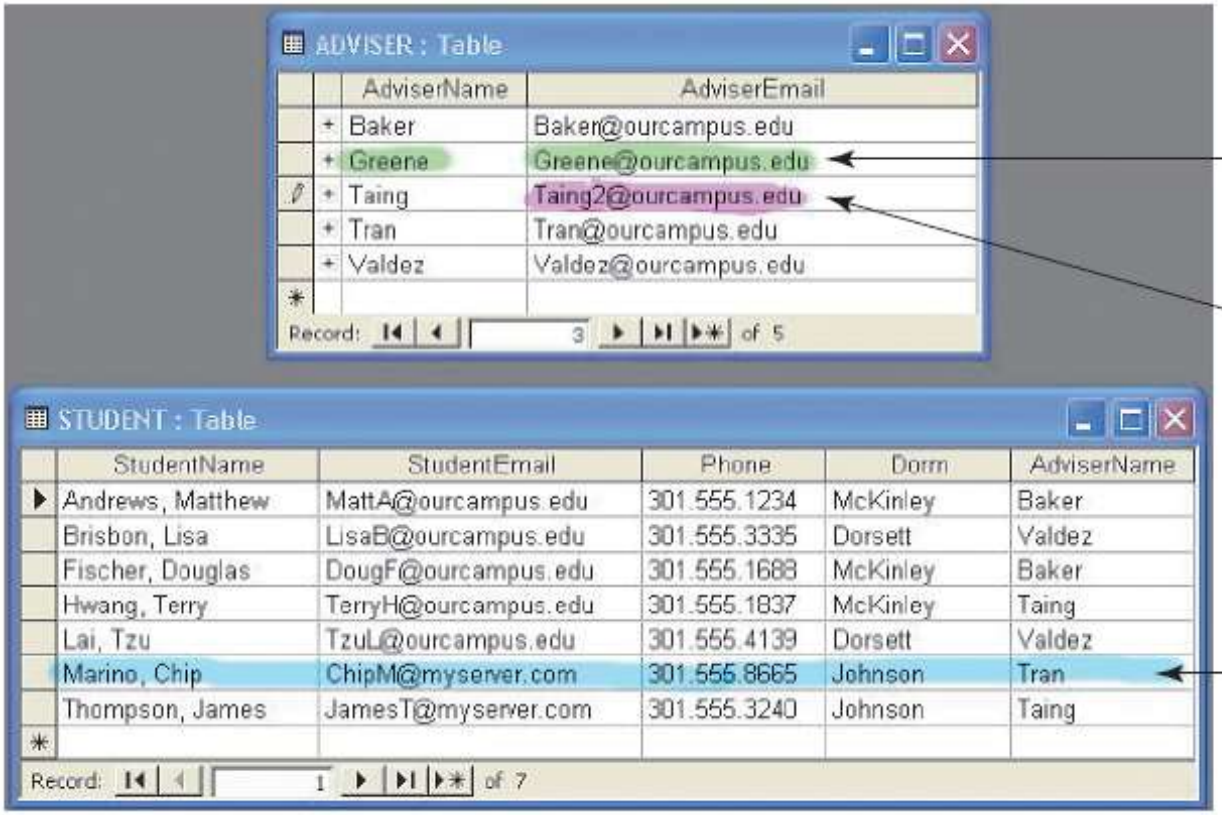
- Redosled podataka u tabeli nije relevantan
- Nema duplikata ili je njihov broj minimalan

Relaciona baza podataka – primer veza između podataka



Podaci o **ADVISER**u su povezani sa **STUDENT**om preko **AdviserData**

Relacione baze podataka – primer rešavanja problema ažuriranja podataka



Dodata nova vrsta – ne zahtevaju se podaci o STUDENTu

Promenjen podatak – podaci ostaju konzistentni

Obrisana vrsta – ne gube se podaci o ADVISERu

Baza podataka kao aspekt realnog sveta

- Slučajni skup podataka nije baza podataka!!
- Baza podataka se projektuje i gradi za specifičnu namenu
- Namenjena je konkretnim korisnicima za konkretne namene
- Baza podataka
- predstavlja neki aspekt realnog sveta organizacije
- tzv **minisvet** – deo realnog sveta za koji je neophodno čuvati i obrađivati podatke
  - Baza podataka za Fakultet – podaci o studentima, ispitima, nastavnicima
- **Promene u minisvetu** utiču na bazu podataka
  - Promena u nastavnim planovima utiče na bazu podataka
    - Promene u podacima
    - Promene u tabelama

Baze podataka: više posla?

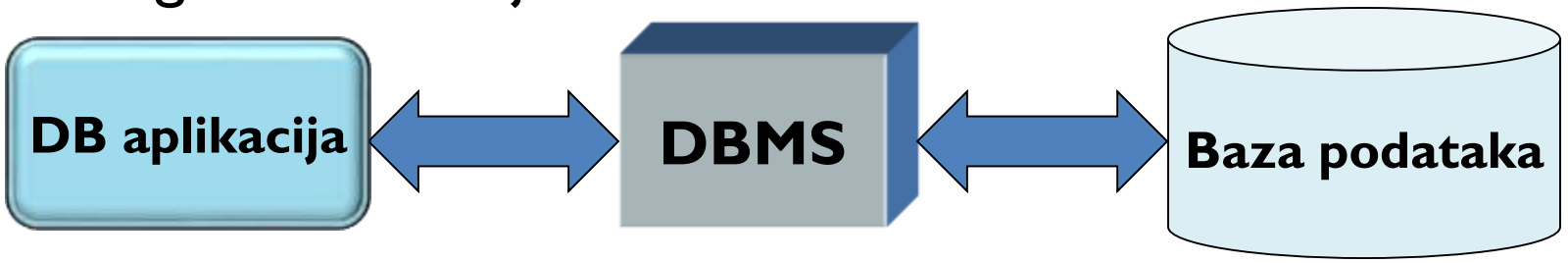
- Relacione baze podataka su na prvi pogled komplikovanije od konvencionalnog pristupa
- Ipak, prednosti su značajne
  - pre svega minimizacija redundanse podataka, kao i
  - očuvanje kompleksnih veza između podataka
- Relaciona baza podataka predstavlja osnovu za korisničke forme i izveštaje
- Više drugačijih poslova – administracija, održavanje podataka...

Šta je sistem za upravljanje bazama podataka?

- ***Sistem za upravljanje bazama podataka*** *(engl. Data Base Management System, DBMS)*
- Softverski sistem koji omogućava definisanje, kreiranje i manipulisanje bazom podataka
  - **Definisanje:** Specificiranje tipova podataka, struktura i ograničenja za podatke koje treba memorisati u bazi podataka
  - **Kreiranje:** Proces memorisanja podataka na nekom medijumu koji kontroliše DBMS
  - **Manipulisanje:** Postavnjanje upita bazi podataka da bi se našli specifični podaci, ažuriranje baze podataka da bi se unele promene nastale u mini svetu i generisanje izveštaja na osnovu podataka memorisanih u bazi podataka

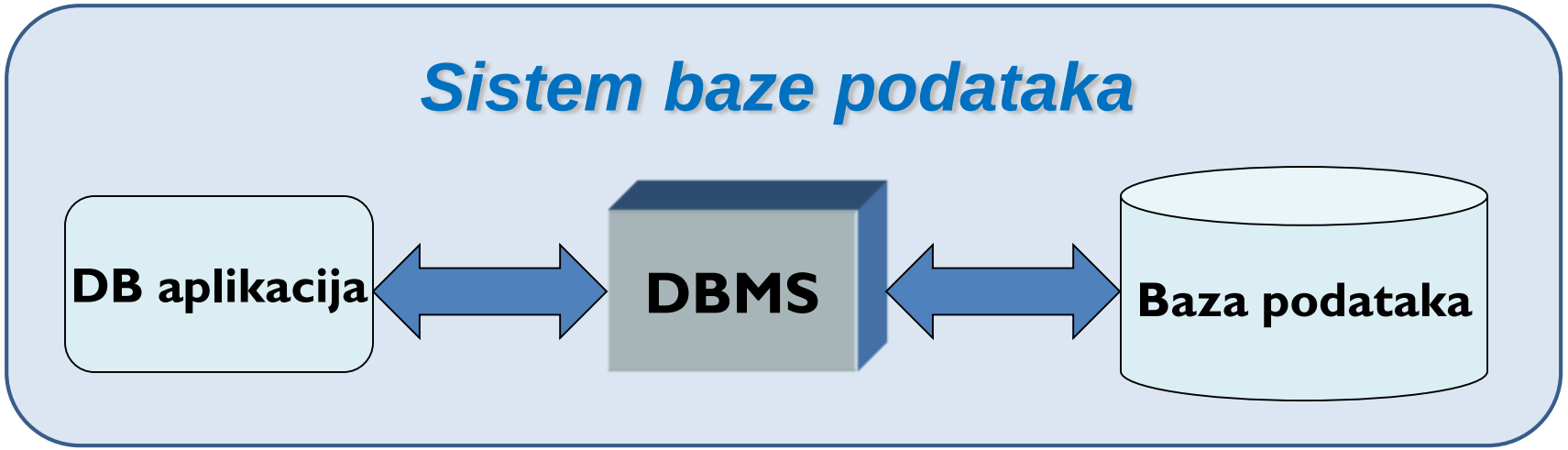
Šta je aplikacija baze podataka?

- ***Aplikacija baze podataka*** *(engl. Database application, DB application)*
  - Program koji interaguje sa bazom podataka u toku svog izvršenja
- Interakcija programa i baze podataka se obavlja preko DBMS-a
- Program šalje zahteve DBMS-u i DBMS vraća programu odgovor na ovaj zahtev

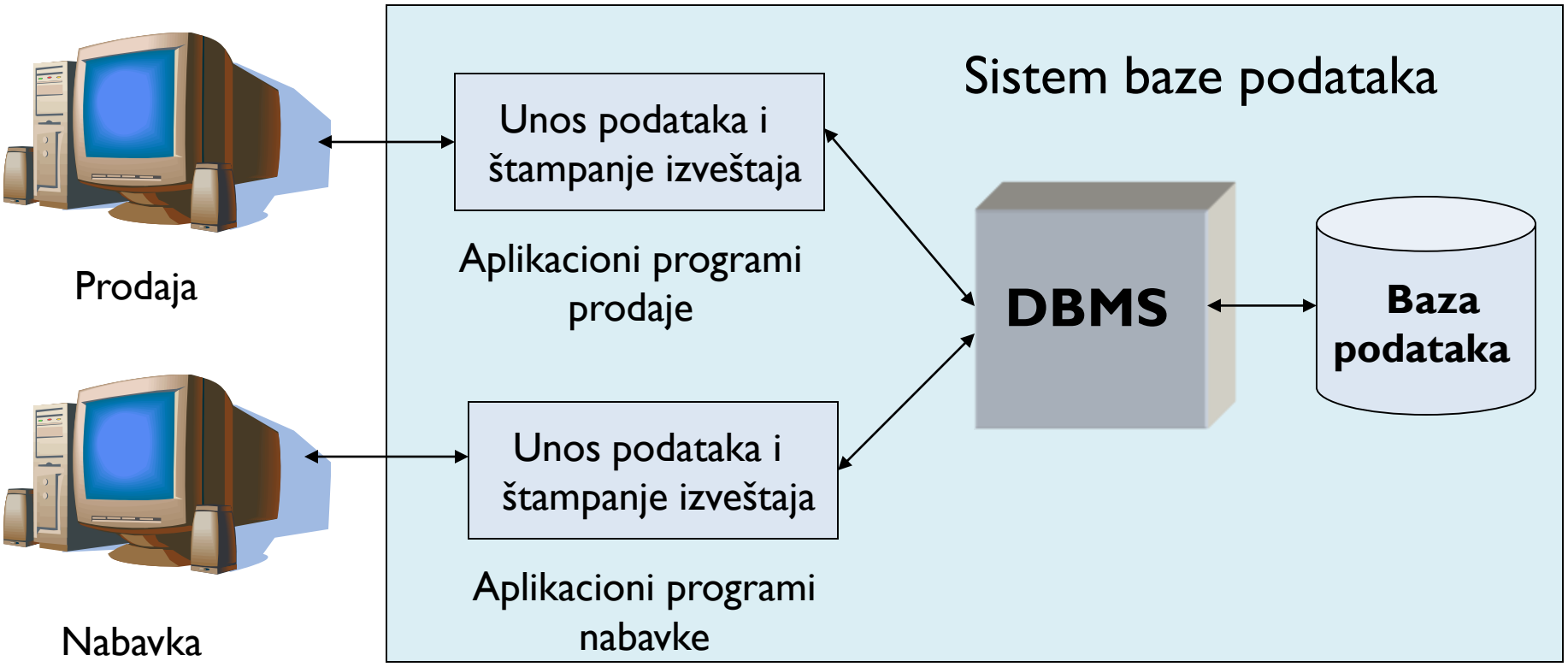


Šta je sistem baze podataka?

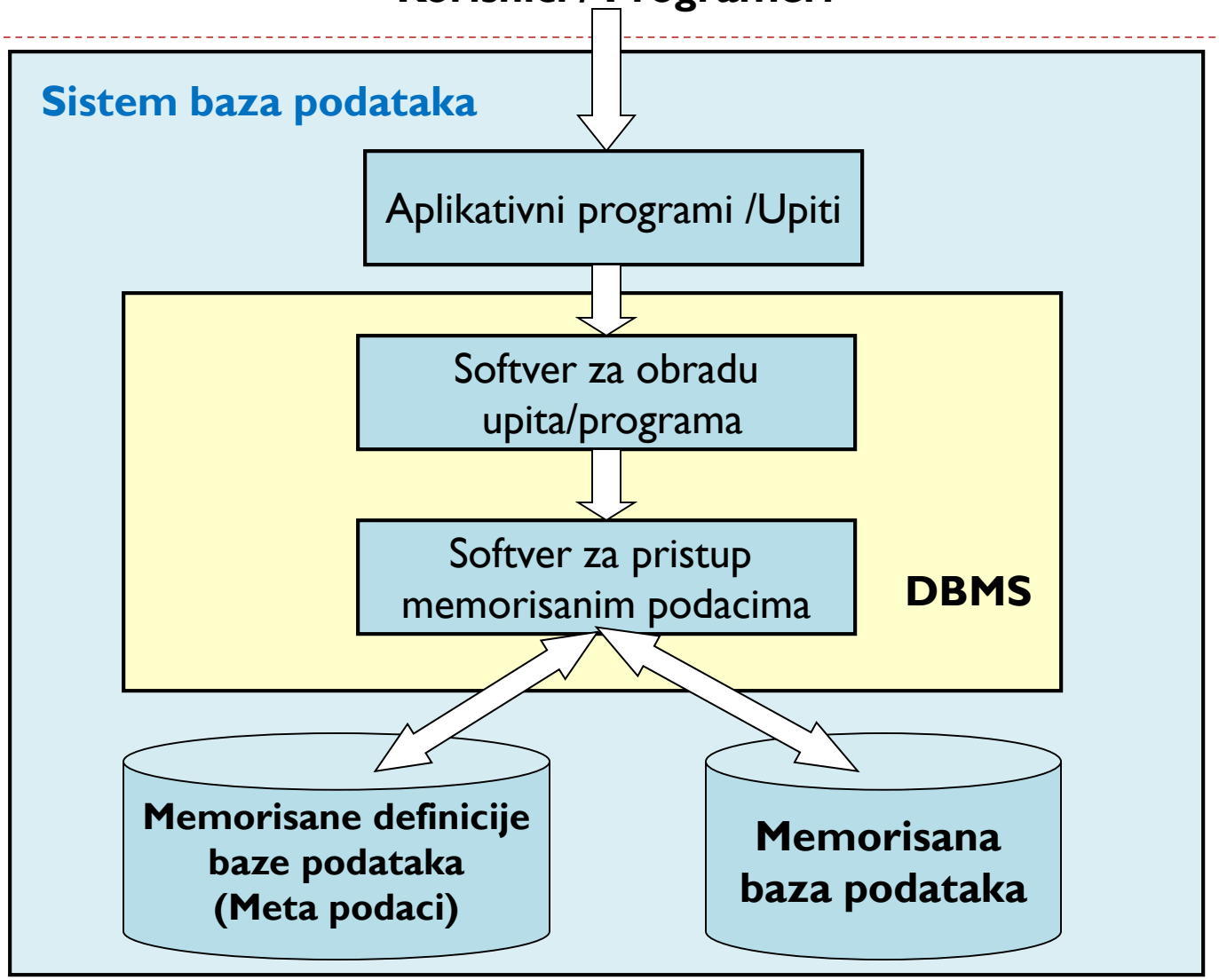
- ***Sistem baze podataka*** *(engl. Database System)*
  - Kolekcija aplikacionih programa koji interaguju sa bazom podataka, DBMS i baza podataka



Obrada zasnovana na bazama podataka



Uprošćena struktura sistema baze podataka



Prilaz zasnovan na bazama podataka

- Baza podataka
  - Deljiva kolekcija logički povezanih podataka i definicija tih podataka, projektovana sa ciljem da zadovolji informacione potrebe neke organizacije
- Važne odrednice
  - Podaci su deljivi
  - Baza podataka sadrži i podatke i njihove definicije (opisi, šeme)
  - Definicije podataka se nalaze u **katalogu sistema (rečnik sistema)** i nazivaju se **meta podaci**
  - Definicije podataka su odvojene od aplikacionih programa
  - Podaci su logički povezani (entiteti, atributi, veze)

Svojstva prilaza zasnovanog na BP

- **Opisi podataka i ograničenja nad podacima** (tzv. **meta podaci**) su sastavni deo sistema baze podataka
  - Opisi se nalaze u sistemskom **katalogu** koji je dostupan i DBMS-u i korisnicima kojima je potrebno poznavanje strukture baze podataka
- Nezavisnost programa i podataka
  - Opisi podataka su u DBMS katalogu i nezavisni su od programa za pristup podacima

Uloge u okruženju baza podataka (osoblje)

- **Administrator podataka i administrator baze podataka**
  - Data Administrator (DA)
  - Database Administrator (DBA)
- **Projektant baze podataka (eng. Database Designer)**
  - Projektant logičke baze podataka
  - Projektant fizičke baze podataka
- **Inženjer razvoja aplikacija (eng. Application Developer)**
  - aplikacioni programer
- **Krajnji korisnici (eng. End-Users)**
  - Slučajni korisnici
  - Naivni ili parametarski korisnici
  - Napredni ili Sofistični korisnici

Administrator podataka

- Odgovoran je za menadžment podataka kao resursa, uključujući:
  - Planiranje, razvoj i održavanje standarda, politike i procedura
  - Konceptualno i logičko projektovanje baze podataka
- Njegov zadatak je da obezbedi DB podršku za ostvarenje postavljenih ciljeva korporacije za koju radi

Administrator baze podataka

- Odgovoran je za fizičku realizaciju sistema baze podataka, uključujući:
  - Fizičko projektovanje i implementaciju baze podataka
  - Sigurnost i integritet
  - Održavanje sistema u radnom stanju
  - Obezbeđenje odgovarajućih performansi aplikacija za potrebe korisnika (monitoring i reorganizacija, po potrebi)

Glavni zadaci DB i DBA

DA

- Učestvuje u strateškom planiranju IS-a
- Utvrđuje dugoročne ciljeve
- Nameće standarde, politiku i procedure
- Utvrđuje zahteve za podacima
- Razvija konceptualni i ligički projekat baze podataka
- Razvija i održava zajednički model podataka
- Koordinira razvoj sistema
- Orijentisan je ka menadžmentu
- **DBMS nezavistan**

DBA

- Evaluira nove DBMS-e
- Izvršava planove radi ostvarenja ciljeva
- Nameće standarde, politiku i procedure
- Implementira zahteve za podacima
- Razvija logički i fizički projekat baze podataka
- Implementira fizički projekat baze podataka
- Nadzire i kontroliše bazu podataka
- Orijentisan je ka tehnici
- DBMS zavistan

Korisnici baze podataka

- Akteri na sceni
  - Administrator baze podataka
  - Projektanti baze podataka
  - Aplikacioni programeri
  - Sistem analitičari
  - Krajnji korisnici
    - Slučajni
    - Naivni
    - Napredni
- Akteri iza scene
  - Projektanti DBMS-a
  - Inženjeri razvoja DBMS-a
  - Operateri
  - Osoblje za održavanje sistema baze podataka

Krajnji korisnici

- **Krajnji korisnici** su lica čiji posao zahteva pristup bazi podataka radi pretraživanja, ažuriranja i generisanja izveštaja
- Vrste krajnjih korisnika
  - **Slučajni korisnici** povremeno pristupaju bazi podataka, njihovi zahtevi su vrlo promenjivi i nisu unapred definisani
    - Za njih se projektuju posebni upitni jezici i dodatna softverska pomagala
  - **Naivni (parametarski) korisnici** ažuriraju bazu podataka i koristeći standardne upite dobijaju potrebne podatke iz baze podataka
    - Primer ovih korisnika su službenici na šalterima
    - Ovi korisnici uvek obavljaju istu obradu nad bazom podataka i za njih se razvijaju posebne DB aplikacije
  - **Napredni (sofistički) korisnici** imaju znanje o DBMS-u tako da mogu formulisati vrlo složene zahteve
    - Primer takvih korisnika su inženjeri, naučnici, istraživači, analitičari

Tipovi sistema baza podataka

- Single-user:
  - Podrška samo za jednog korisnika u isto vreme
- Desktop:
  - Single-user baza podataka na personalnom računaru
- Multi-user:
  - Podrška za više korisnika u isto vreme
- Workgroup:
  - Multi-user baza podataka za mali broj korisnika ili jedno odeljenje preduzeća
- Enterprise:
  - Multi-user baza podataka koja podržava veliki broj korisnika ili celo preduzeće/organizaciju

Tipovi sistema baza podataka

Klasifikacija po lokaciji:

- Centralizovane:
  - Podaci su na jednoj lokaciji
- Distribuirane:
  - Podaci mogu biti na više lokacija

Po načinu korišćenja:

- Transakcione (ili produkcione):
  - Podrška za *day-to-day* operacije preduzeća
- Data warehouse:
  - Čuva podatke koji se koriste za generisanje informacija potrebnih za strateške odluke.
  - Često sadrže istorijske podatke i imaju drugačiju strukturu

Prednosti DBMS-a

- **Nezavisnost podataka**
  - Aplikacioni programi su nezavisni od reprezentacije i načina memorisanja podataka
  - DBMS nudi aplikaciji apstraktni pogled na podatke
- **Efikasan pristup podacima**
  - DBMS koristi posebne tehnike za efikasno memorisanje i pretraživanje podataka
- **Integritet i bezbednost podataka**
  - DBMS kontroliše integritet podataka koristeći ograničenja koja je definisao projektant baze podataka
  - DBMS nadgleda pristupne privilegije korisnika, tako da svaki korisnik može videti samo podatke za koje poseduje pristupne privilegije
- **Administracija podataka**
  - Kada su podaci deljivi centralizovana administracija podataka donosi značajno poboljšanje
  - Za to je zadužen administrator baze podataka
- **Konkurentni pristup i oporavak**
  - DBMS upravlja konkurentnim pristupom podacima od strane više korisnika
  - DBMS ima ugražene mehanizme oporavka baze podataka od različitih neispravnosti
- **Kraće vreme razvoja aplikacija**
  - DBMS ima ugrađene različite funkcije za rad sa podacima što aplikacionim programerima značajno olakšava zadatak razvoja aplikacije
  - DBMS nudi interfejs iz jezika visokog nivoa ka bazi podataka

Nedostaci DBMS-a

- Složenost
- Veličina
- Troškovi za DBMS
- Dodatni troškovi za HW
- Troškovi konverzije
- Performanse
- Veći uticaj neispravnosti

Istorijat DBMS-a - 1.deo

- 60-tih godina 20-tog veka
  - projekat Apollo, na inicijativu predsednika Kenedija
  - Rezultat ovog projekta je GUAM (Generalized Update Access Method)
- Sredinom 60-tih
  - IBM je prihvatio GUAM ideju i razvio IMS (Information Management System) – to je **hijerarhijski DBMS**
- Sredinom 60-tih godina
  - General Electric je razvio IDS (Integrated Data Store) – to je **mrežni DBMS**
- Prvi DBMS opšte namene koga je projektovao Charles Bachman, prvi dobitnik Turing-ove nagrade koju dodeljuje ACM (ekvivalent Nobelove nagrade za Računarske nauke)
- Bachman je nagradu dobio za rad u oblasti baza podataka
- 1967. je formirana, u okviru CODASYL-a, Database Task Group (DBTG) koja je uradila standard za DBMS:
- 1969 draft report
- 1971 final report

Istorijat DBMS-a - 2.deo

- 1970. E.F.Codd iz IBM Research Lab je predložio **relacioni model**
- Prvi komercijalni relacioni DBMS su proizvedeni krajem 70-tih i početkom 80-tih godina 20-tog veka:
  - Projekat System R (IBM) krajem 70-tih (SQL)
  - DB2 i SQL/DS (IBM) 80-tih
  - Oracle (Oracle Corporation) 80-tih
- Danas postoji na hiljade relacionih DBMS-ova za mainframe i PC okruženja:
  - INGRES (Computer Associates)
  - INFORMIX, DB2 (IBM)
  - Office Access, Visual FoxPro (Microsoft)
  - InterBase, JDataStore (Borland)
  - R:Base (R:Base Technologies)

Istorijat DBMS-a - 3.deo

- 1976. Chen je predložio ER model
- 1979. Codd je predložio proširenu verziju relacionog modela RM/T i 1990. RM/V2 koji se ubraja u semantičke modele podataka
- 90-tih godina 20-tog veka Objektno-orijentisani DBMS (OODBMS) i Objektno-relacioni DBMS (ORDBMS)
- Početak ovog veka
  - Postrelacione BP
  - XML native BP
- ... Relacione baze podataka + nove tehnologije
  - NoSQL baze podataka
  - Trendovi vraćanja na stare, dobre vrednosti RDBMS

Baze podataka

Prof. dr Leonid Stoimenov

Katedra za računarstvo
Elektronski fakultet u Nišu

Uvod u baze podataka

Kraj predavanja

Pitanja ???

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Modeli podataka i projektovanje baza podataka

Prof.dr Leonid Stoimenov

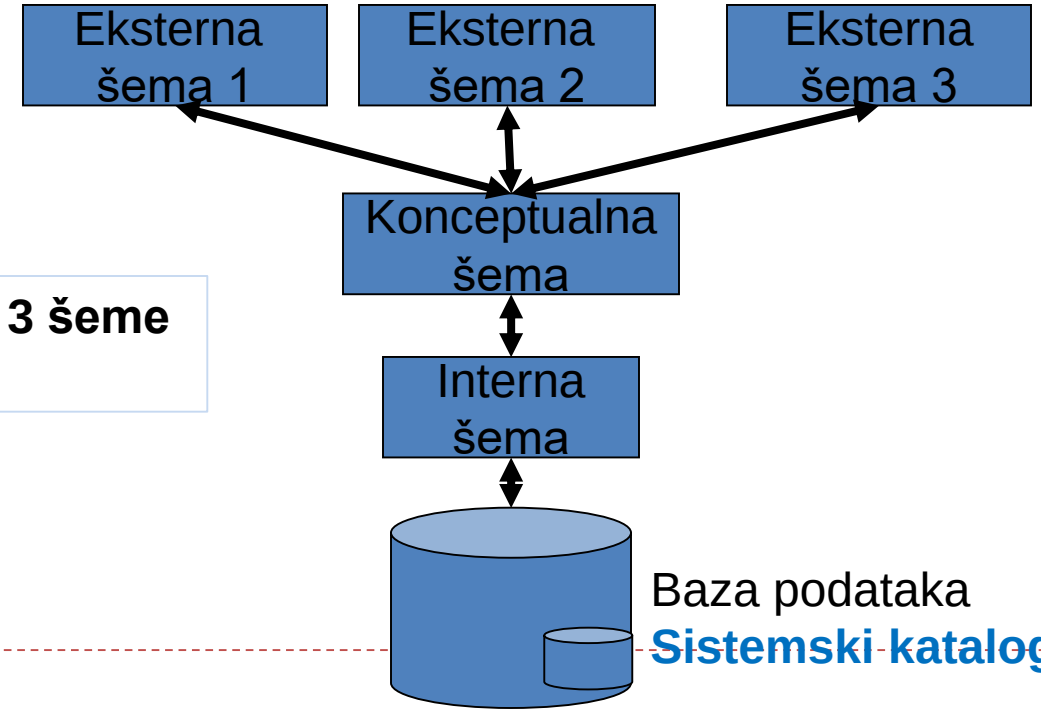
Pregled predavanja

3 termina za predavanja

- Arhitektura 3-šeme DBMSa
- Modeli podataka
- Projektovanje baza podataka
- Konceptualno modeliranje

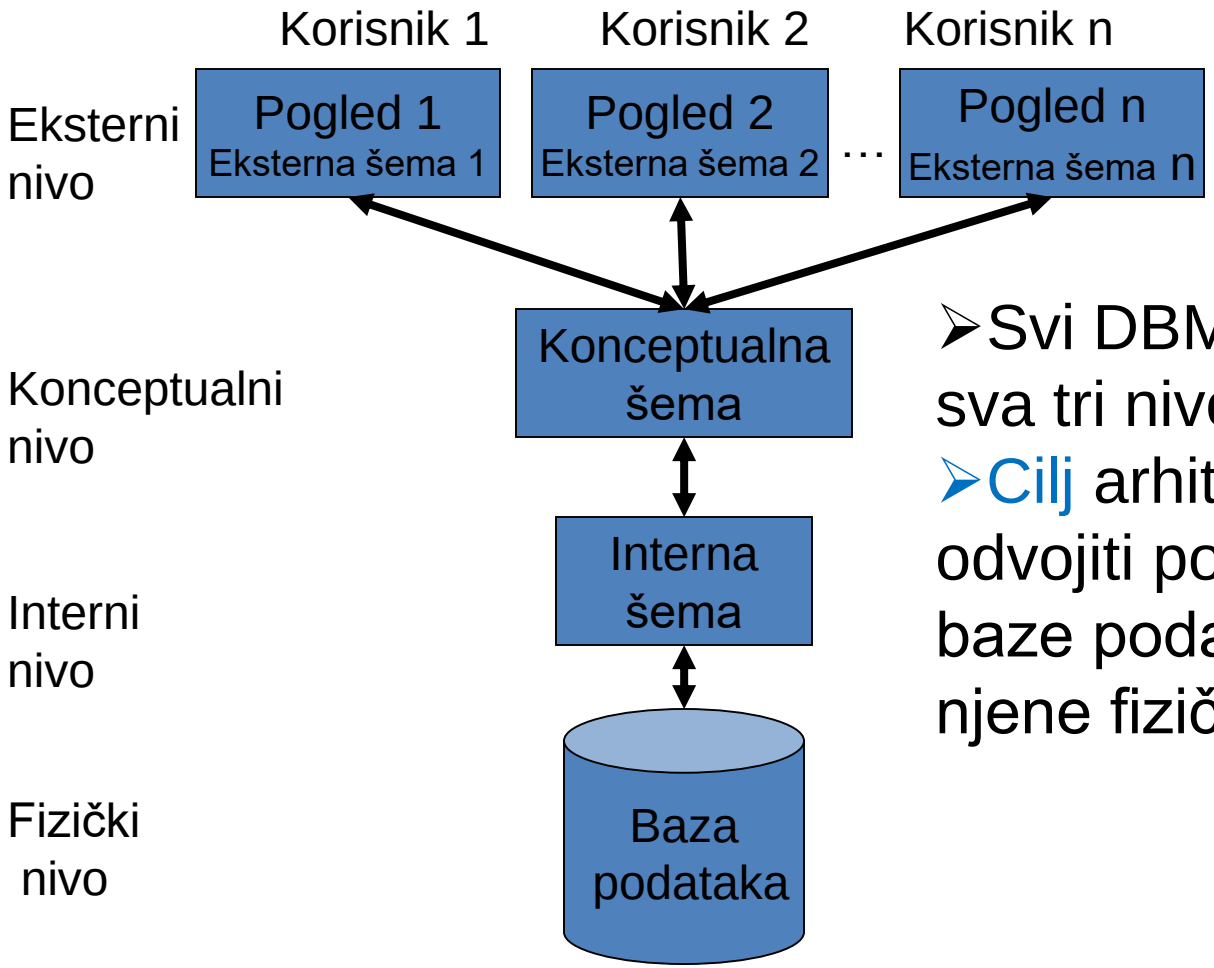
Nivoi apstrakcije u DBMS-u

- U DBMS-u podaci su opisani u **tri nivoa apstrakcije**:
  - Eksterna šema
  - Konceptualna (Logička) šema
  - Intena (Fizička) šema
- Šeme se memorišu u **sistemskom katalogu** DBMS-a



Arhitektura 3 šeme DBMSa

Arhitektura tri šeme DBMS-a
(ANSI-SPARC arhitektura)



- Svi DBMS-i ne razlikuju sva tri nivoa
- Cilj arhitekture 3-šeme je odvojiti pogled korisnika baze podataka od načina njene fizičke reprezentacije

Šeme, preslikavanja, instance

- **Šema baze podataka** je **opis baze podataka**
- U arhitekturi 3 šeme postoje
  - **Eksterna šema** (često se naziva i **podšema**): ima ih više
  - **Konceptualna šema**: postoji jedna za bazu podataka
  - **Interna šema**: postoji jedna za bazu podataka
- DBMS je odgovoran za **preslikavanja** između ovih šema
  - Preslikavanje konceptualne u internu šemu i obrnuto
  - Preslikavanje eksterne u konceptualnu šemu i obrnuto

- **Instancu baze podataka** čine **podaci** koji se trenutno nalaze u bazi podataka
- Jednoj šemi baze podataka može odgovarati više instanci

- Šema baze podataka se naziva i **intenzija** baze podataka, a instanca **ekstenzija** ili **stanje** baze podataka

Eksterni nivo (nivo pogleda)

- Korisnički pogled na bazu podataka
- Svaki pogled na bazu podataka
  - opisuje samo onaj deo baze podataka koji je od interesa za konkretnu grupu korisniku i
  - prikriva ostali deo baze podataka od ove grupe korisnika
- Sadrži veliki broj **eksternih šema** ili **pogleda korisnika**
- Različiti pogledi mogu imati različite reprezentacije istih podataka

Konceptualni (logički) nivo

- Pogled na bazu podataka **svih korisnika** baze podataka
- Opisuje koji se podaci pamte u bazi podataka, kako su ti podaci struktuirani i kako su povezani
- Sadrži **konceptualnu šemu** koja predstavlja **globalni opis baze podataka**
- Opisuje:
  - entitete, atribute i njihove veze,
  - ograničenja nad podacima,
  - semantičke informacije o podacima,
  - sigurnost i integritet informacija
- **Ne sadrži** nikakve detalje o **načinu memorisanja** podataka!

Interni nivo

- Fizička reprezentacija baze podataka na računaru
- Sadrži **internu (fizičku) šemu** koja opisuje kako su podaci memorisani u bazi podataka
- Opisuje sve detalje o memorisanju podataka, definiše strukturu i organizaciju fajlova koji se koriste za smeštanje baze podataka
- Koristi usluge fajl sistema OS-a za smeštanje podataka na memorijski medijum, za formiranje indeksa, za pretraživanje podataka itd.
- Na internom nivou se definiše:
  - Dodela prostora na disku za podatke i indekse
  - Opis slogova
  - Smeštanje slogova
  - Kompresija podataka
  - Tehnike enkripcije podataka

Fizički nivo

- Ispod internog nivoa je **fizički nivo** kojim može da upravlja OS po direktivama DBMS-a
- Funkcije OS-a i DBMS-a na fizičkom nivou nisu strogo razgraničene, pa variraju od jednog do drugog DBMS-a
- Neki DBMS-i koriste metode pristupa OS-a, dok drugi imaju sopstvene fajl sisteme

Opis i memorisanje podataka u DBMS-u

- Korisnici baze podataka
  - su u nekom realnom svetu, i
  - podaci memorisani u bazi podataka predstavljaju neke aspekte tog realnog sveta (mini svet)
  - Primer: baza podataka Fakultet
- DBMS omogućava korisnicima
  - da **opišu** (definišu) podatke koje žele memorisati u bazi podataka
  - Za opis podataka koristi se **model podataka**

Modeli podataka

- **Model podataka** je integrisana kolekcija koncepata za opis i manipulaciju podacima, vezama između podataka i ograničenjima nad podacima i vezama
  - To je matematička apstrakcija koja se koristi za projektovanje modela realnog sistema

- Model podataka ima tri **komponente**:
  - **Strukturnu** koja se sastoji od skupa pravila prema kojima se baza podataka može konstruisati
  - **Integritetnu** koja definiše ograničenja nad vrednostima atributa, veze između podataka i međusobnu uslovljenost podataka
  - **Operacijsku** koja definiše operacije nad strukturama podataka i koja modelira dinamičke osobine realnog sistema

Tipovi modela podataka

- **Model podataka je kolekcija koncepata visokog nivoa** kojima se opisuje tip, struktura i relacije među podacima

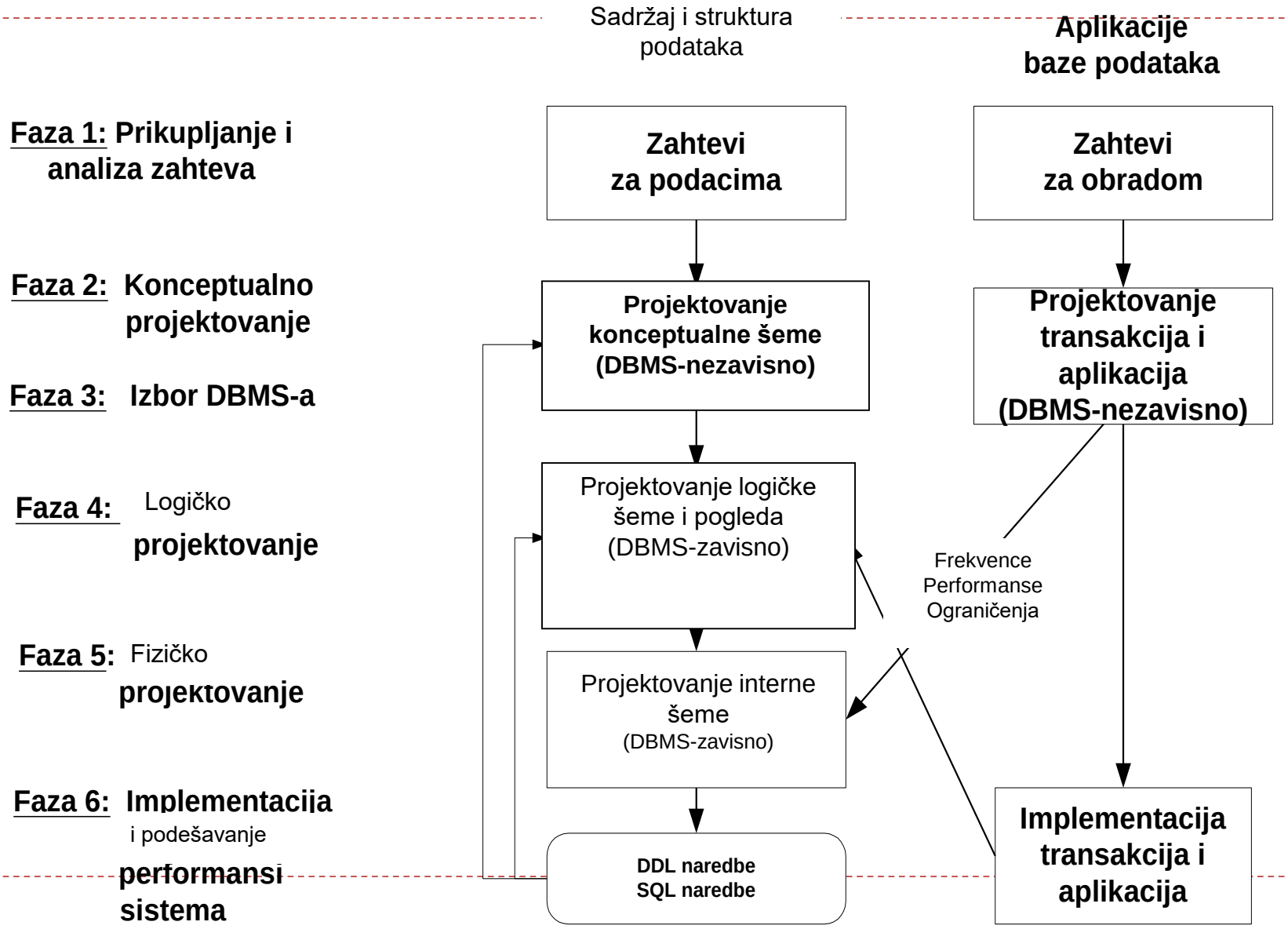
- U odnosu na ANSI-SPARC arhitekturu DBMS-a modeli podataka mogu biti:
  - Eksterni modeli podataka – služe za predstavljanje pogleda korisnika na realni svet
  - **Konceptualni modeli podataka** – služe za predstavljanje logičkog pogleda ili pogleda zajednice na realni sistem nezavisno od DBMS-a
  - Interni modeli podataka – služe za predstavljanje konceptualne šeme na takav način da je razumljiva za DBMS

Projektovanje baze podataka

- **Problem:**
  - Projektovati **logičku** i **fizičku** strukturu jedne ili više baza podataka da bi se zadovoljile informacione potrebe korisnika u organizaciji za definisani skup operacija

- **Ciljevi:**
  - Zadovoljiti informacione zahteve specificiranih korisnika i aplikacija
  - Obezbediti prirodno i lako za razumevanje struktuiranje informacija
  - Podržati zahteve obrade i zahteve performansi (vreme odgovora, vreme obrade i memorijski prostor)

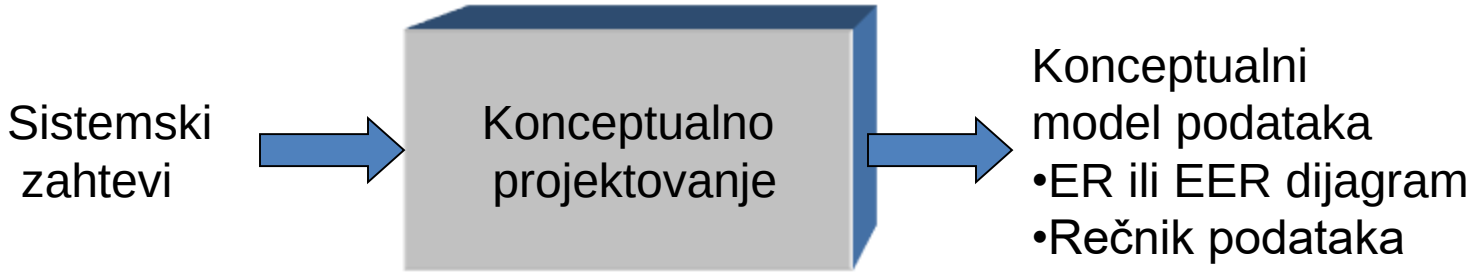
Proces projektovanja baze podataka



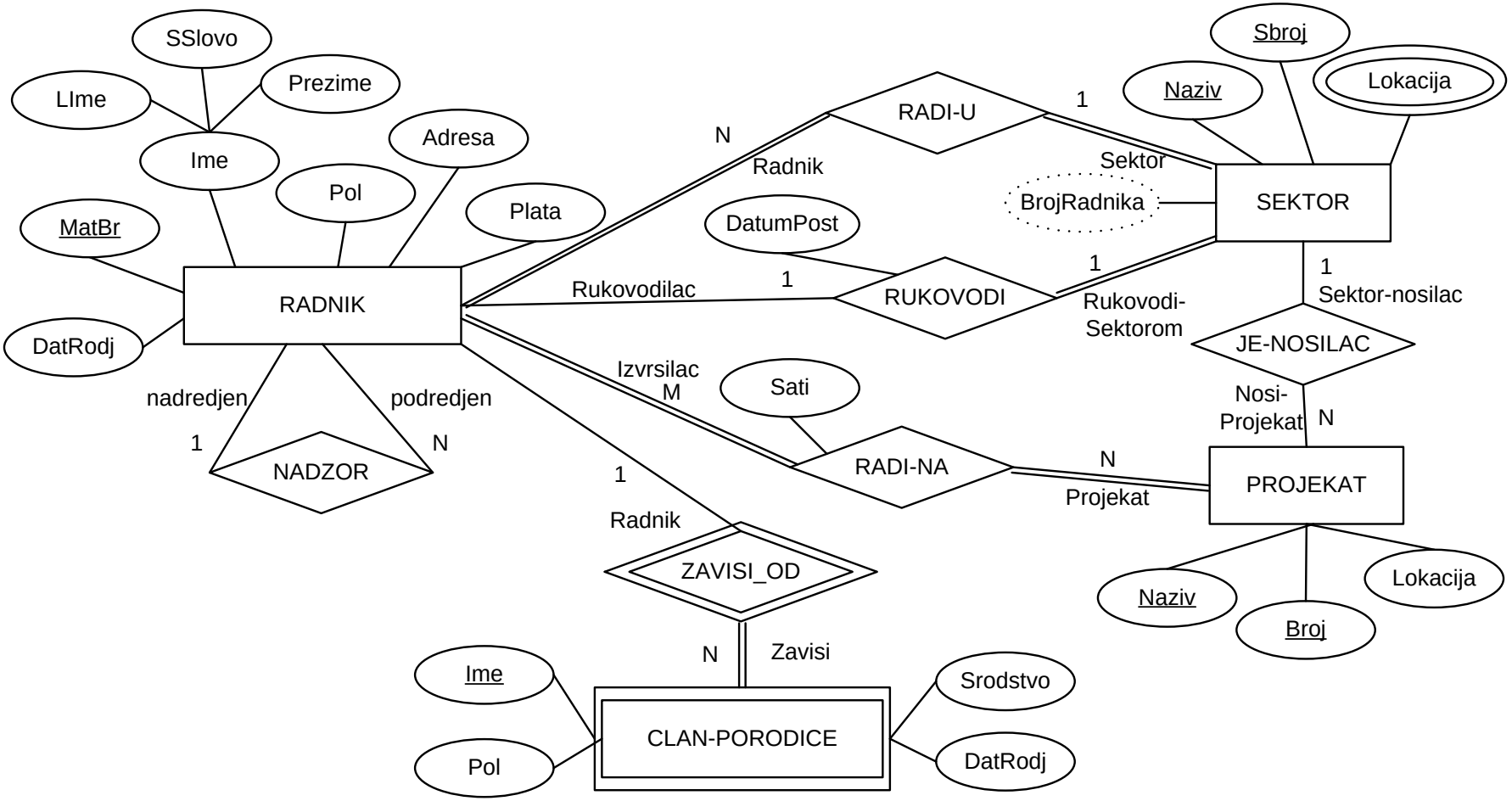
Faza	Korak
1. Konceptualno projektovanje **Cilj:** **Projektovanje ER ili EER modela baze podataka** **Izlaz:** **ER ili EER dijagram i opis dijagrama u Rečniku podataka**	1. Identifikovanje tipova entiteta 2. Identifikovanje tipova veza 3. Identifikovanje atributa za tipove entiteta i veza 4. Određivanje domena atributa 5. Određivanje ključeva kandidata i izbor primarnog ključa 6. Razmatranje upotrebe koncepata proširenog modeliranja 7. Provera da li model zadovoljava transakcije 8. Provera da li model zadovoljava postavljene zahteve
2. Logičko projektovanje **Cilj:** **Projektovanje relacionog modela baze podataka** **Izlaz:** **Šema relacione baze podataka**	1. Preslikavanje konceptualnog u relacioni model 2. Validacija relacija korišćenjem normalizacije (cilj je da svaka relacija bude bar u BCNF) 3. Definisanje ograničenja integriteta 4. Provera da li model zadovoljava transakcije 5. Provera da li model zadovoljava postavljene zahteve
3. Implementaciono projektovanje **Cilj:** **Projektovanje relacione šeme baze podataka za Oracle DBMS** **Izlaz:** **SQL opis implementacione šeme relacione baze podataka**	1. Prevođenje logičkog modela podataka u implementacioni model za ciljni DBMS (Oracle) 2. Provera da li model zadovoljava transakcije 3. Provera da li model zadovoljava postavljene zahteve 4. Opis implementacione šeme putem jezika za opis podataka (podskup SQLa) i mehanizama izabranog DBMSa

Konceptualno projektovanje baze podataka

- Cilj: Projektovati konceptualni model baze podataka koji što vernije opisuje realni sistem za koji se projektuje baza podataka
- Koristi se neki model podataka visokog nivoa
  - ER ili EER model podataka



Primer ER modela baze podataka PREDUZEĆE



(E)ER model podataka

- (Enhanced) Entity Relationship (ER) model
- Za konceptualno projektovanje baze podataka
- ER dijagram za grafičko predstavljanje
- Nazivi:
  - Model entiteta i veza
  - Model entiteta i poveznika
  - Model objekat-veze
- Autor Chen, 1976
- Postoji više verzija ER modela i više grafičkih notacija

Strukturna komponenta ER modela podataka

- Dva osnovna pojma su u osnovi ovog modela:
  - Entitet
  - Veza
- Osnovna ideja
  - Realni sistem (realni svet ili neki njegov deo - *minisvet*) se može opisati pomoću ova dva osnovna „koncepta“

- **Entitet** u realnom svetu je “nešto” što se može jednoznačno identifikovati
  - Može se odnositi na realni subjekat, objekat, događaj, pojavu ili apstraktni pojam
  - Entitet može biti objekat koji **fizički** egzistira (npr. osoba, kola, kuća ili radnik) ili objekat koji **konceptualno** egzistira (npr. kompanija, posao ili predmet na univerzitetu)
- **Veza** određuje odnos (vezu) između dva ili više entiteta

Entitet u ER modelu

Entitet je

subjekat, objekat, događaj, pojava ili apstraktni pojam o kome se prikupljaju, memorišu, obrađuju i prezentiraju podaci u automatizovanim informacionim sistemima i koji se može **jednoznačno identifikovati** i na taj način izdvojiti u skupu sličnih entiteta

Primeri Entiteta za mini svet Preduzeće i mini svet Fakultet:

- **Primer PREDUZEĆE:**
  - radnik
  - sektor (organizaciona jedinica)
  - radno mesto
  - projekat
  - plan
- **Primer FAKULTET:**
  - student
  - pofesor
  - predmet
  - laboratorija
  - udžbenik

Kandidati za entitete u realnom sistemu

- Organizacione jedinice
  - fakultet, katedra, biblioteka,…
- Lokacije
  - učionica, raskrsnica, radarska pozicija, laboratorija,…
- Uloge
  - radnik, službenik, student, profesor, stanovnik, referent,…
- Događaji koji se pamte
  - ispit, utakmica, predavanje, ispit,…
- Uređaji
  - proizvodna traka, računar, skener, radar, antena, …
- Drugi sistemi sa kojima naš sistem interaguje
  - IS ministarstva, IS univerziteta, baza podataka stanovništva,…

Skup entiteta

- Entiteti koji egzistiraju u ljudskom intelektu kao predstave realnih subjekata, objekata ili događaja mogu se klasifikovati u **skupove sličnih entiteta**

- **Primeri skupa entiteta**
  - studenti jednog fakulteta
  - radnici jednog preduzeća
  - proizvodi jednog preduzeća
  - zgrade jednog preduzeća
  - uplate na žiro račun

Tip entiteta u ER modelu

- **Tip entiteta** *(egl. Entity Type)* je **model** skupa entiteta

- Obeležavanje tipa entiteta
  - **E(A1, A2, ... , An)**
  - gde je E **ime** tipa entiteta, a Ai | i = 1,n **atributi** odabrani za modeliranje entiteta E

- **Primer :**
  - **RADNIK1** (IME, DATUM-ROĐENJA, ADRESA, TELEFON, SEKTOR, LD, ČLANOVI-PORODICE)
  - **RADNIK2** (IME, SEKTOR, LD)
  - **RADNIK3** (MATIČNI-BROJ-RADNIKA, MATIČNI-BROJ-STANOVNIKA, IME-RADNIKA)

Tipovi entiteta RADNIK1, RADNIK2, RADNIK3 modeliraju na razne načine entitet RADNIK iz realnog sistema

Skup entiteta u ER modelu

- **Skup entiteta** *(engl. entity set)* je kolekcija svih entiteta određenog tipa entiteta u bazi podataka u nekom trenutku
- Za skup entiteta se koristi **ime tipa entiteta**

- Primer:
  - Ime RADNIK se koristi i za tip entiteta i za trenutni skup svih entiteta radnik u bazi podataka kojoj taj tip entiteta pripada

- Tip entiteta opisuje **šemu** ili **intenziju** skupa entiteta koji dele istu strukturu
- Kolekcija entiteta određenog tipa entiteta grupisana u skup entiteta se takođe naziva **ekstenzija** tipa entiteta

Pojava (instanca) tipa entiteta

- Pojava ili instanca tipa entiteta se odnosi na *skup podataka o određenom entitetu* (iz skupa entiteta koji je modeliran odgovarajućim tipom entiteta)

- **Primer**
  - Dve pojave (instance) tipa entiteta RADNIK2(IME, SEKTOR, LD):

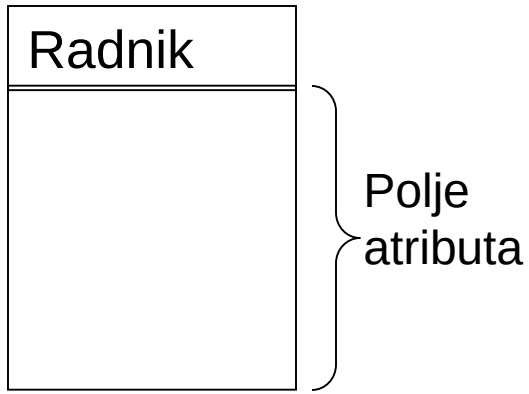
    SAVA KRSTIĆ, INSTITUT, 85000

    JOVAN RISTIĆ, TEST, 57000

Predstavljanje tipa entiteta u ER dijagramu

ERD notacija:
Pravougaonik sa upisanim imenom tipa entiteta

UML notacija



Ime entiteta: RADNIK, odnosno Radnik
Ograničenje: Imena entiteta u jednom ER dijagramu su jedinsvena

Atribut (obeležje) u ER modelu

- Svaki objekat realnog sveta ima osobine **(obeležja)** koji se kod tipa entiteta predstavljaju **atributima** – to svojstva/obeležja koja ga opisuju, i preko kojih se čuvaju podaci
  - Svi entiteti jednog skupa imaju bar jedno zajedničko svojstvo na osnovu koga su svrstani u isti skup
    - Obično postoji više takvih zajedničkih osobina koji opisuju skup entiteta
  - Ova svojstva se kod Tipa entiteta nazivaju atributima
- Primer
  - Entitet RADNIK može imati atribute
    - Matični broj radnika, ime, prezime, datum rodjenja,…
  - Entitet STUDENT može imati atribute
    - Broj indeksa, ime, prezime, datum rođenja, godina studija, …

Obeležavanje atributa

- **Stil A**
  - Obeležavaju se velikim slovom sa crticom ili znakom za podvlačenje kao poveznikom između reči
  - **Primer :**

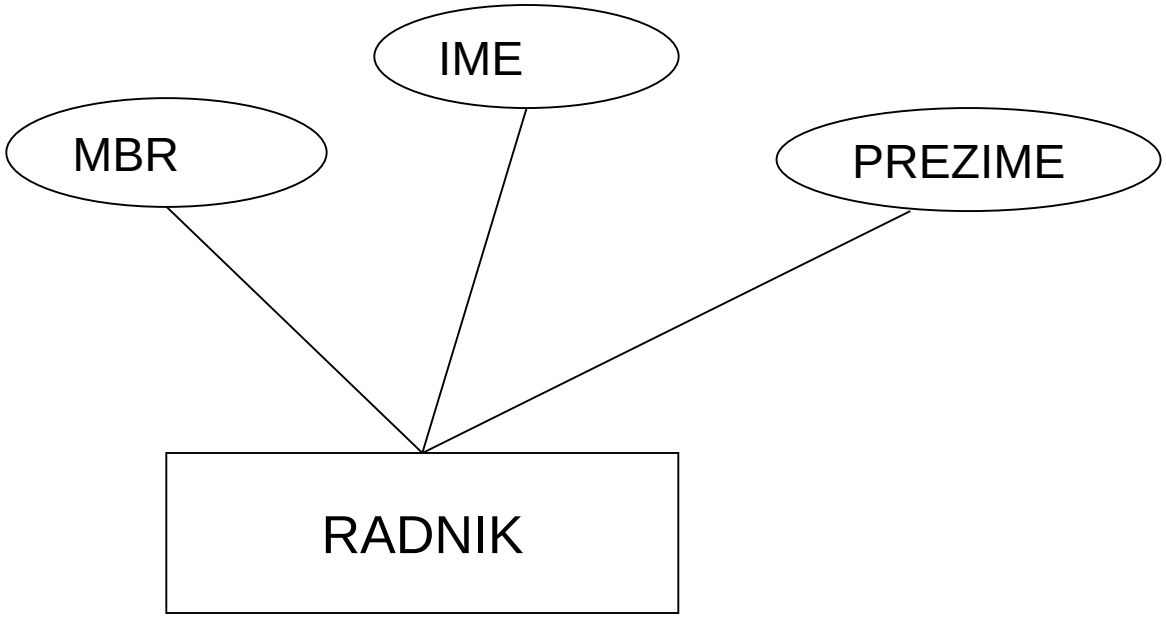
IME	IME
DATUM_UPISA	DATUM-UPISA
BOJA_KOLA	BOJA-KOLA

- Stil B (UML notacija)
  - Obeležavaju se nizom reči koje se pišu malim slovima, osim prvog slova svake reči. Nema delimitera između reči.
  - **Primer :**

Ime
DatumUpisa
BojaKola

Predstavljanje atributa u ER dijagramu

Označavanje: elipsa sa upisanim imenom atributa, povezana linijom sa pravougaonikom koji označava odgovarajući tip entiteta kome atribut pripada



Slika: Tip entiteta RADNIK sa atributima MBR, Ime i Prezime

Prosti i složeni atributi

- Atribut je **prost (elementaran)** ako se dalje ne može dekomponovati ili ako se u konretnoj situaciji ne dekomponuje na komponente koje čine atribut
- **Primer elementarnih atributa**

  OCENA-STUDENATA

  BOJA-AUTOMOBILA

  NAZIV-PROIZVODA
- Vrednost elementarnog atributa je **elementarni (prost) podatak**
- Atribut je **složen (kompozitan)** ako je sastavljen od više elementarnih atributa
- **Primer složenih atributa**

  ADRESA-STUDENTA

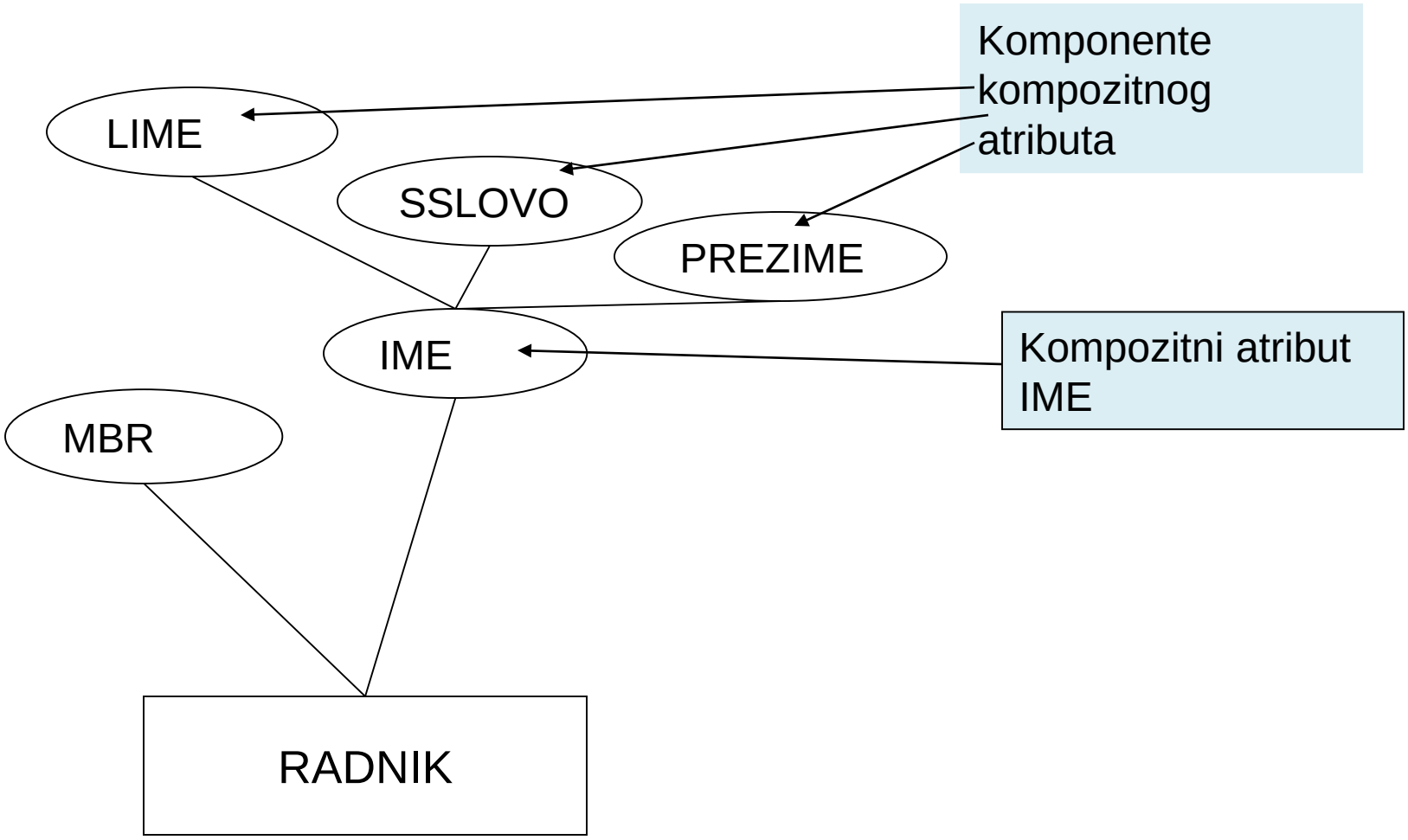
  IME-STUDENTA

  DATUM-UPISA
- Vrednost složenog atributa je **složeni (strukturni) podatak**

Dekomponovanje složenih atributa

- Složeni atribut se može podeliti (dekomponovati) na manje delove koji predstavljaju više osnovnih atributa sa nezavisnim značenjem
- Primeri:
  - **ADRESA_STUDENTA** se može dekomponovati na
    - ULICA, BROJ_ZGRADE, BROJ_STANA, GRAD
  - **IME_STUDENTA** se može dekomponovati na
    - LIME, SREDNJE_SLOVO, PREZIME
  - **DATUM_UPISA** se može dekomponovati na
    - DAN, MESEC, GODINA
- Mi ćemo dekomponovan složeni atribut označavati na sledeći način:
  - **ADRESA_STUDENTA(** ULICA, BROJ_ZGRADE, BROJ_STANA, GRAD)
  - **IME_STUDENTA** (LIME, SREDNJE_SLOVO, PREZIME)
  - **DATUM_UPISA** (DAN, MESEC, GODINA)

Predstavljanje složenih atributa u ER dijagramu



Složeni atributi – kada i zašto?

- Složeni atributi su korisni za modeliranje situacije gde se korisnici kod pretraga ponekad koriste atribut kao celinu, a ponekad koriste pojedine komponente tog atributa
- Ako se složeni atribut uvek referencira kao celina treba ga modelirati kao prost atribut

- Primer:

  ADRESA_STUDENTA(ULICA, BROJ_ZGRADE, BROJ_STANA, GRAD)

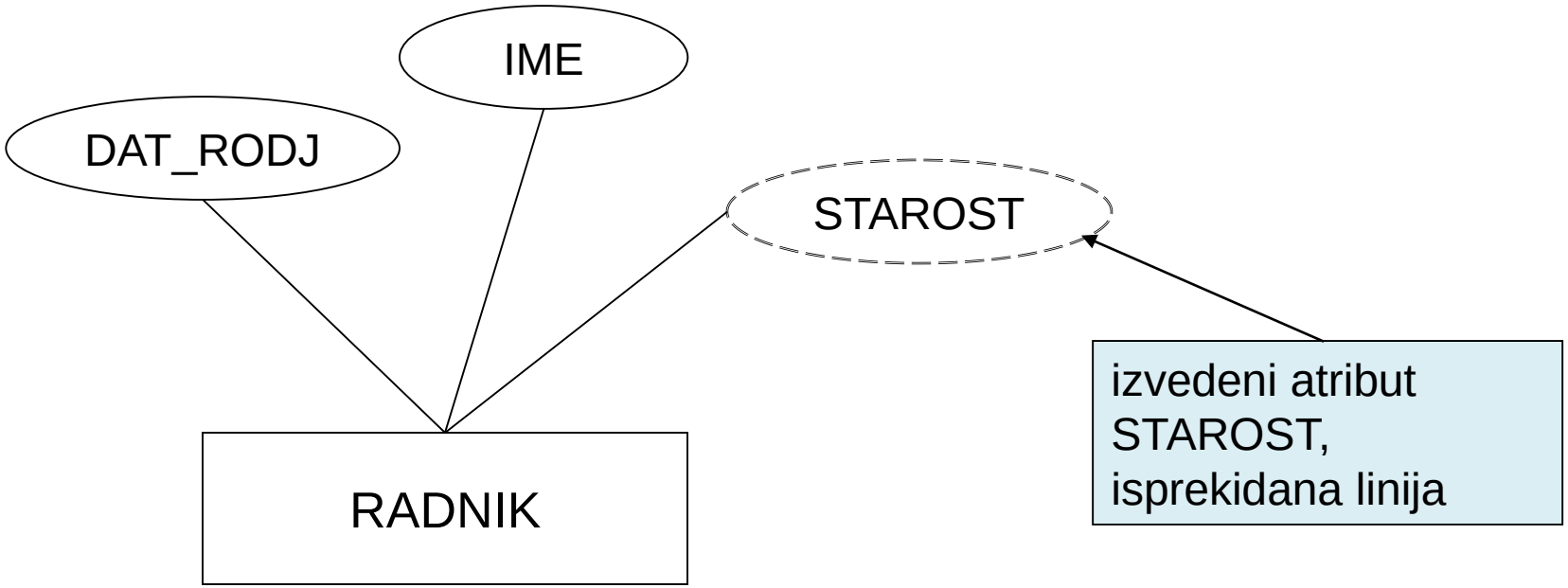
  Ako nema potrebe za nezavisnim referenciranjem pojedinih komponenti adrese, adresu studenta treba modelirati kao prost atribut

Izvedeni atribut

- To je atribut čije se vrednosti dobijaju primenom nekog algoritma na vrednosti drugih atributa (elementarnih, složenih, izvedenih)
- Vrednost izvedenog atributa je **izvedeni podatak**

- **Prmer 1**
  - Atribut SREDNJA_OCENA studenta se može dobiti primenom algoritma za izračunavanje srednje vrednosti na atribut OCENA u entitetu STUDENT
- **Primer 2**
  - Atribut BROJ_ZAPOSLENIH u preduzeću se izvodi primenom algoritma prebrojavanja pojava entiteta RADNIK

Predstavljanje izvedenih atributa u ER dijagramu



Domen atributa

- **Domen atributa** deifiniše **skup vrednosti atributa** (*engl.value set*) iz kojeg atribut uzima vrednosti
- **Obeležavanje domena**

  dom(ime_atributa)

- **Primer**

  dom(IME) = Character(15)

  dom(BOJA_KOLA) = (bela, crvena, zelena)

  dom(OCENA_STUDENTA) = (5, 6, 7, 8, 9, 10)

  dom(PRELAZNA_OCENA) = (6, 7, 8, 9, 10)

Pojava (instanca) atributa

- To je konkretna vrednost koju atribut uzima iz svog domena da predstavi podatak o određenom entitetu
  - Samo instanca atributa može predstavljati podatak
- **Primer**
  - Student PERA ILIĆ ima broj indeksa RE 5034/88.
  - Tada je podatak
    - PERA ILIĆ instanca atributa IME_STUDENTA, a
    - RE5034/88 instanca atributa BROJ_INDEKSA

Vrednost atributa

- Svaki entitet za svaki atribut definisan tipom entiteta ima određenu vrednost

- **Primer**

  RADNIK(IME,ADRESA,DATUM_ROĐENJA, KUCNI_TELEFON, ZANIMANJE,SEKTOR,LD)

  - Vrednosti atributa dve instance tipa entiteta RADNIK (dva konkretna entiteta odnosno dva radnika):

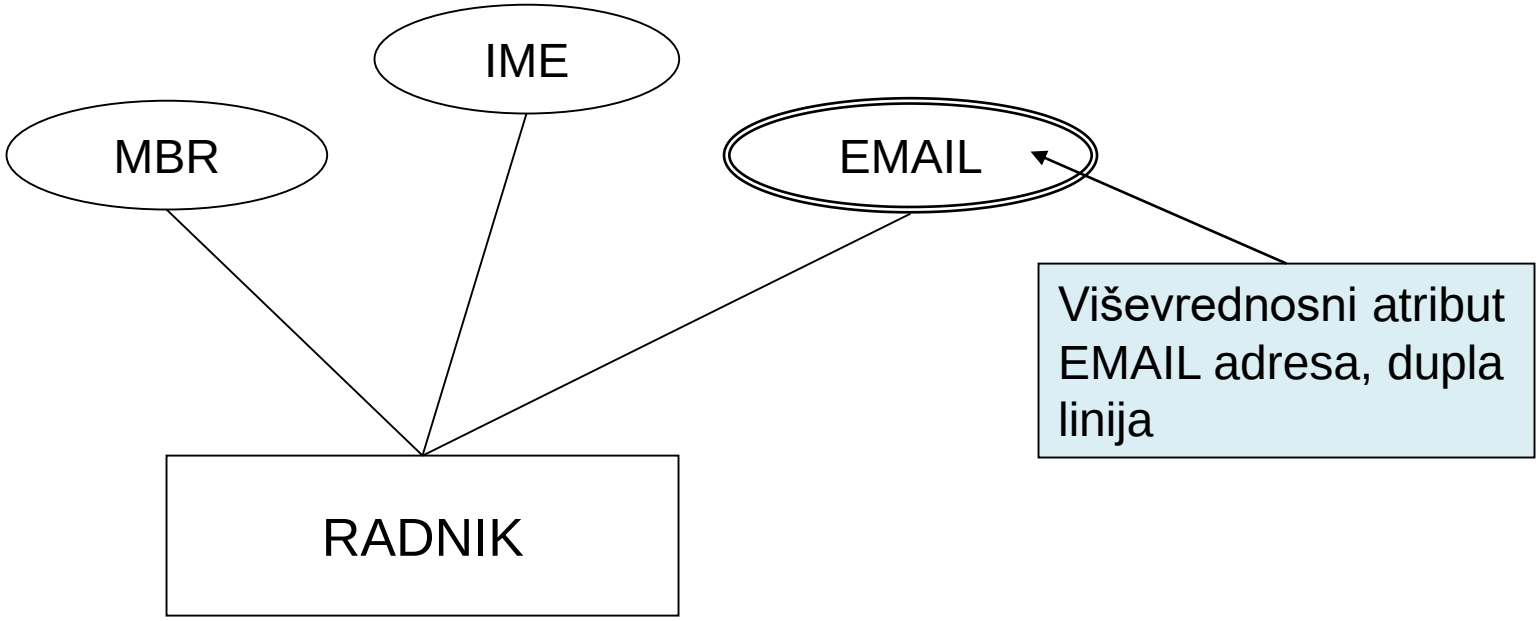
Pera Ilić, Niška16 18000 Niš YU, 05 - FEB – 68, 018 - 315 – 072, inženjer elektronike, Razvoj, 35000

Mina Stojanović, Majakovskog 29 18000 NIŠ YU, 25 - JUL – 85, 018 - 515 – 799, diplomirani pravnik, Zajedničke službe, 25000

Jednovrednosni i viševrednosni atributi

- Za određeni entitet jedan atribut može imati jednu ili više vrednosti
  - U prvom slučaju atribut je jednovrednosni, a u drugom viševrednosni
- **Primer**
  - Atribut DATUM_ROĐENJA entiteta RADNIK je jednovrednosni atribut jer radnik može imati samo jedan datum rođenja, npr.05-FEB-68
  - Atribut KUĆNI_TELEFON entiteta RADNIK je viševrenosni atribut jer radnik može imati više telefona, npr.520-505 i 224-061
  - Atribut EMAIL_ADRESA entiteta RADNIK je viševrenosni atribut jer radnik može imati više email adresa

Predstavljanje viševrednosnog atributa u ER dijagramu



NULL vrednost atributa

- Neki atributi mogu biti bez vrednosti za neki entitet
  - Tada se koristi NULL vrednost
- NULL vrednost predstavlja specijalno ograničenje domena
  - Putem ovog ograničenja se specificira da li atribut može imati nedefinisanu vrednost
- **Primer**
  1. Ako radnik IVANA GOCIĆ nema telefon u stanu, atribut TELEFON za ovog radnika će imati NULL vrednost
  2. Ako radnik IVANA GOCIĆ nije trenutno raspoređena ni u jednom sektoru, atribut SEKTOR će imati NULL vrednost
  3. Ako je adresa radnika IVANE GOCIĆ nepoznata, atribut ADRESA će imati NULL vrednost
- **Semantika NULL vrednosti**
  - Neprimerena vrednost (primeri 1 i 2)
  - Postojeća ali nepoznata vrednost (primer 3)

Ključ

- **Ključ** je atribut ili minimalni skup atributa koji ima jedinstvenu vrednost za sve pojave određenog tipa entiteta
- Svaki tip entiteta poseduje bar jedan ključ
- Tip entiteta može imati više ključeva - to su **ključevi kandidati**
  - jedan od ključeva kandidata je **primarni ključ**

- **Primer**

  U tipu entiteta

  RADNIK3 (MATIČNI_BROJ_RADNIKA, MATIČNI_BROJ_STANOVNIKA, IME_RADNIKA)

  ključ može biti MATIČNI_BROJ_RADNIKA i MATIČNI_BROJ_STANOVNIKA ukoliko važi pravilo da svi radnici imaju različite matične brojeve radnika i različite matične brojeve stanovnika

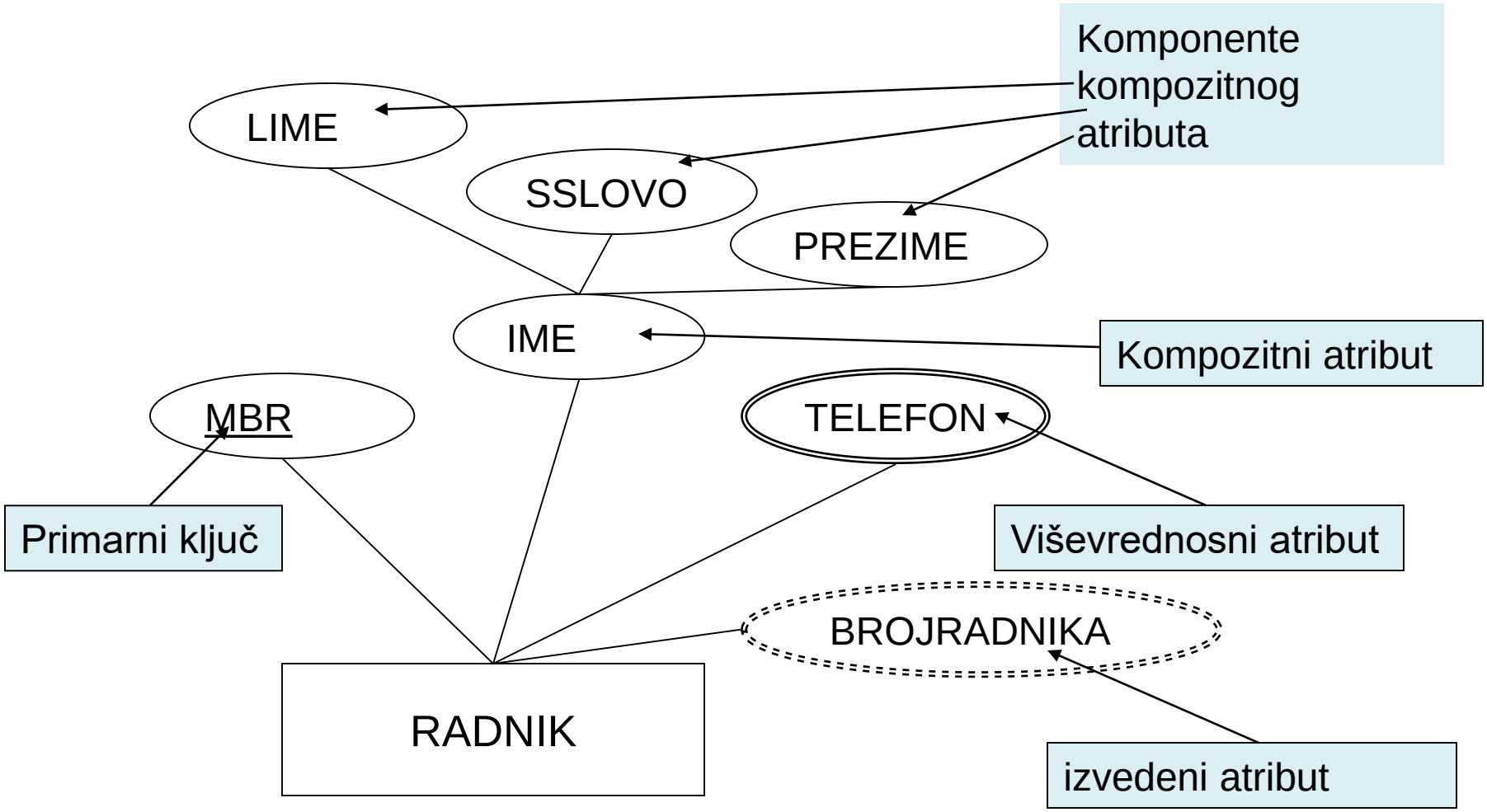
Obeležavanje primarnog ključa

- **Koriste se dva načina**
  - primarni ključ u tipu entiteta se **podvlači kontinualnom linijom**
  - iza imena primarnog ključa se stavlja povisilica (#)
- **Primer**

RADNIK3(MATIČNI_BROJ_RADNIKA, MATIČNI_BROJ_STANOVNIKA, IME)

RADNIK3(MATIČNI_BROJ_RADNIKA# , MATIČNI_BROJ_STANOVNIKA, IME)

Predstavljanje atributa u ER dijagramu



Skup poveznika

- U realnom sistemu između entiteta postoje određeni odnosi (relacije) koje treba modelirati

- **Skup poveznika** $R=\{e_1,e_2,\ldots,e_n | e_i \in E_i, i=1,\ldots,n\}$ predstavlja relaciju (u matematičkom smislu) između n ($n \geq 2$) skupova entiteta
  - Skupovi E_i ne moraju biti različiti
  - Svaka n-torka $(e_1,e_2,\ldots,e_n)$ u R predstavlja jedan poveznik
  - Svaki entitet u n-torci ima svoju ulogu
  - Ako se uloge entiteta u n-torki eksplicitno navedu tada redosled navođenja entiteta nije bitan
  - Između dva ista skupa entiteta može postojati više različitih skupova poveznika

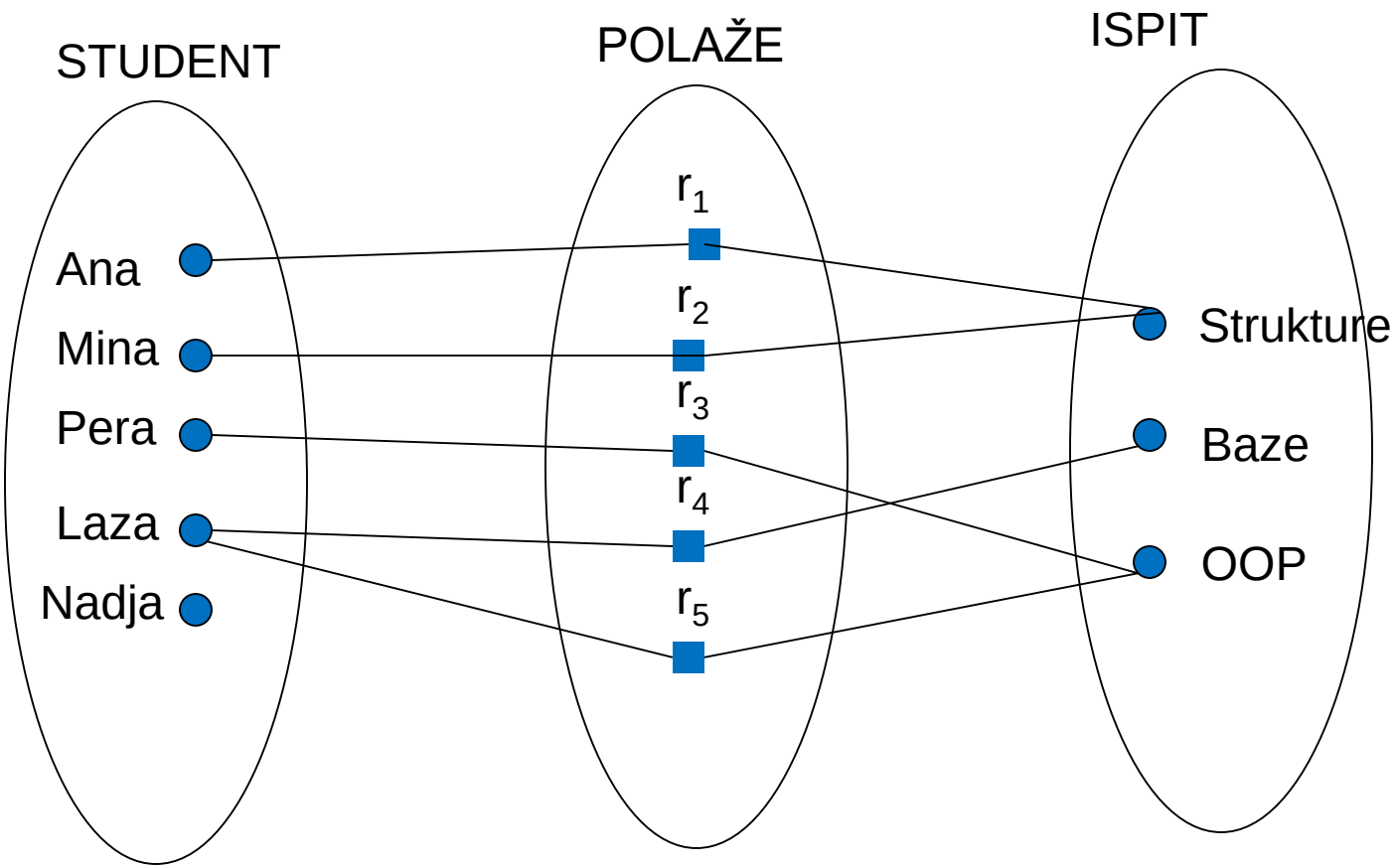
- Ako poveznik povezuje entitete istog skupa, naziva se **rekurzivnim**

Tip poveznika

- Tip poveznika je **model** skupa poveznika.
  - Obično se za naziv tipa poveznika koristi naziv skupa poveznika, tj. $R(E_1,E_2,\ldots,E_n;B_1,B_2,\ldots,B_m)$
  - Za svaki tip entiteta $E_i \mid i=1,n$ se kaže da participira u tipu poveznika R

- Primeri tipa poveznika
  - Modeliranje veze između radnika i projekata, radnik radi na projektima
    RADI-NA(RADNIK,PROJEKAT)
  - Modeliranje veze između radnika i projekata, radnik rukovodi projektom
    RUKOVODI(RADNIK,PROJEKAT)
  - Modeliranje veze između radnika i radnika, odnos nadređeni-podređeni
    JE-RUKOVODILAC(RADNIK,RADNIK)

Neke pojave tipa poveznika POLAŽE



Stepen tipa poveznika

- Stepen tipa poveznika je broj tipova entiteta koji participiraju u tipu poveznika
  - Tip poveznika stepena 1 naziva se rekurzivni
  - Tip poveznika stepena 2 se naziva binarni
  - Tip poveznika stepena 3 se naziva ternarni
  - ....
- Primer
  - Rekurzivni tipovi poveznika

    JE-RUKOVODILAC(RADNIK,RADNIK)

    U-BRAKU(OSOBA,OSOBA)

  - Binarni tipovi poveznika

    RADI-NA(RADNIK,PROJEKAT)

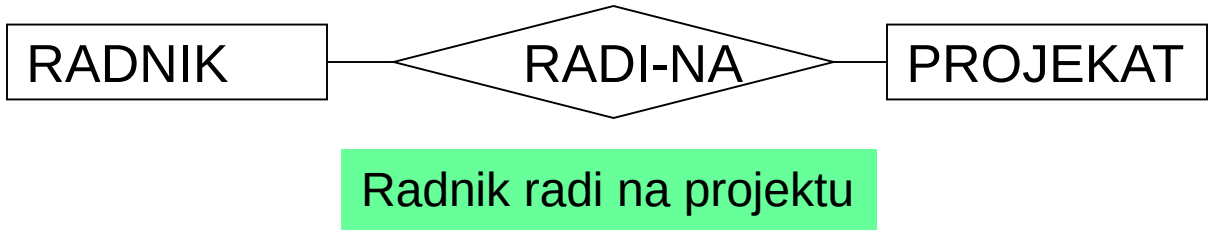
    RUKOVODI(RADNIK,PROJEKAT)

    JE-RUKOVODILAC(RADNIK,RADNIK)
  - Ternarni tip poveznika

    SNABDEVA(DOBAVLJAČ,DEO,PROJEKAT)

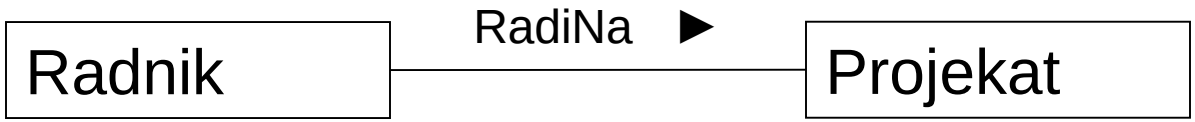
Predstavljanje tipa poveznika u ER dijagramu

ERD: romb, sa navedenim imenom tipa veze

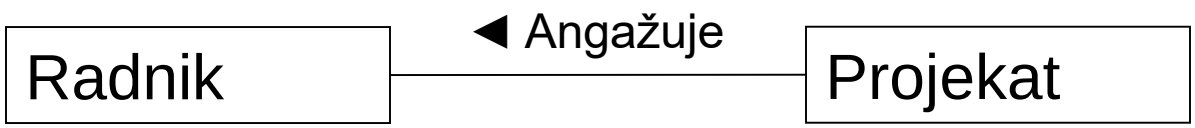


Radnik radi na projektu

UML notacija

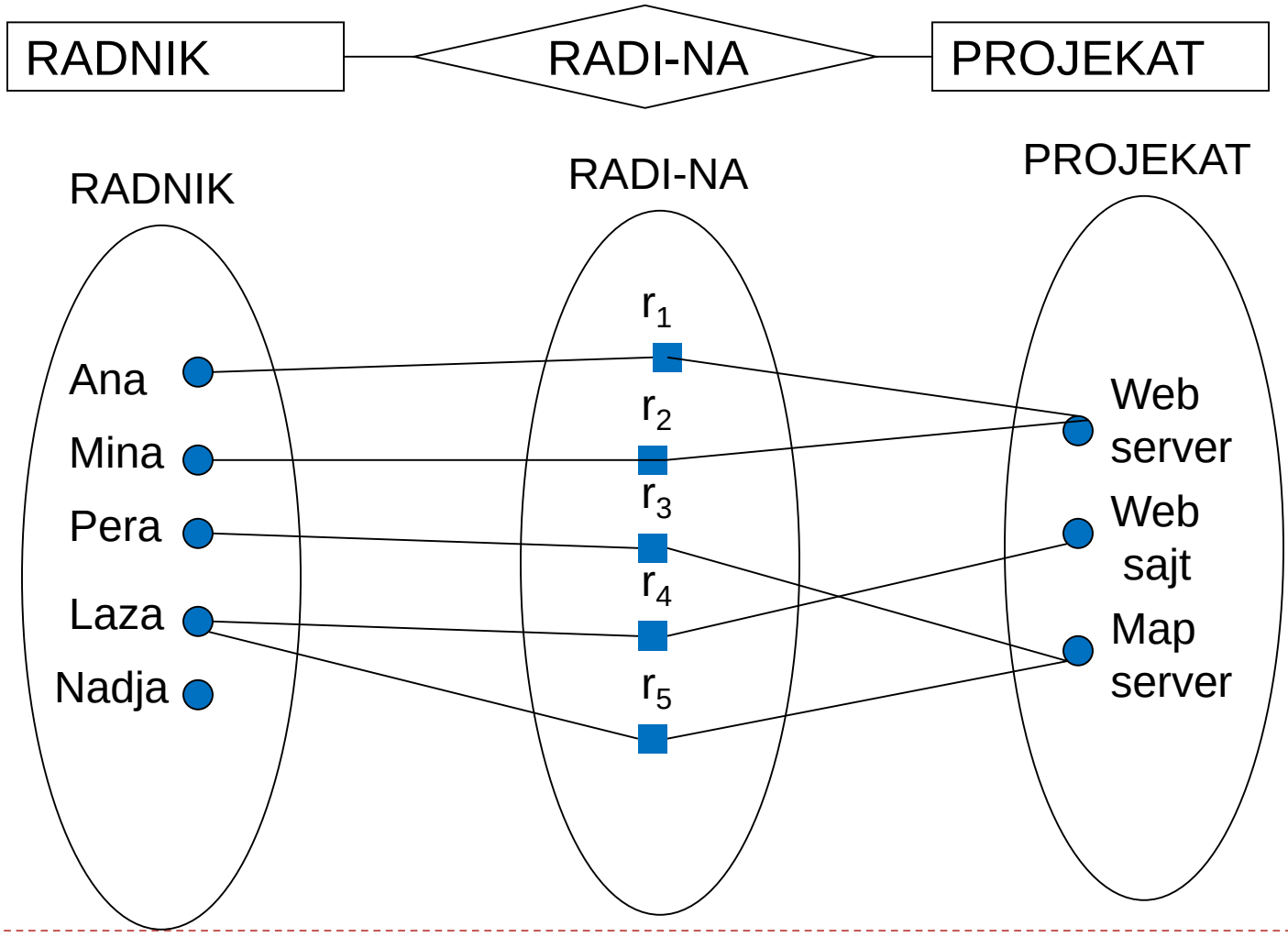


Radnik radi na projektu

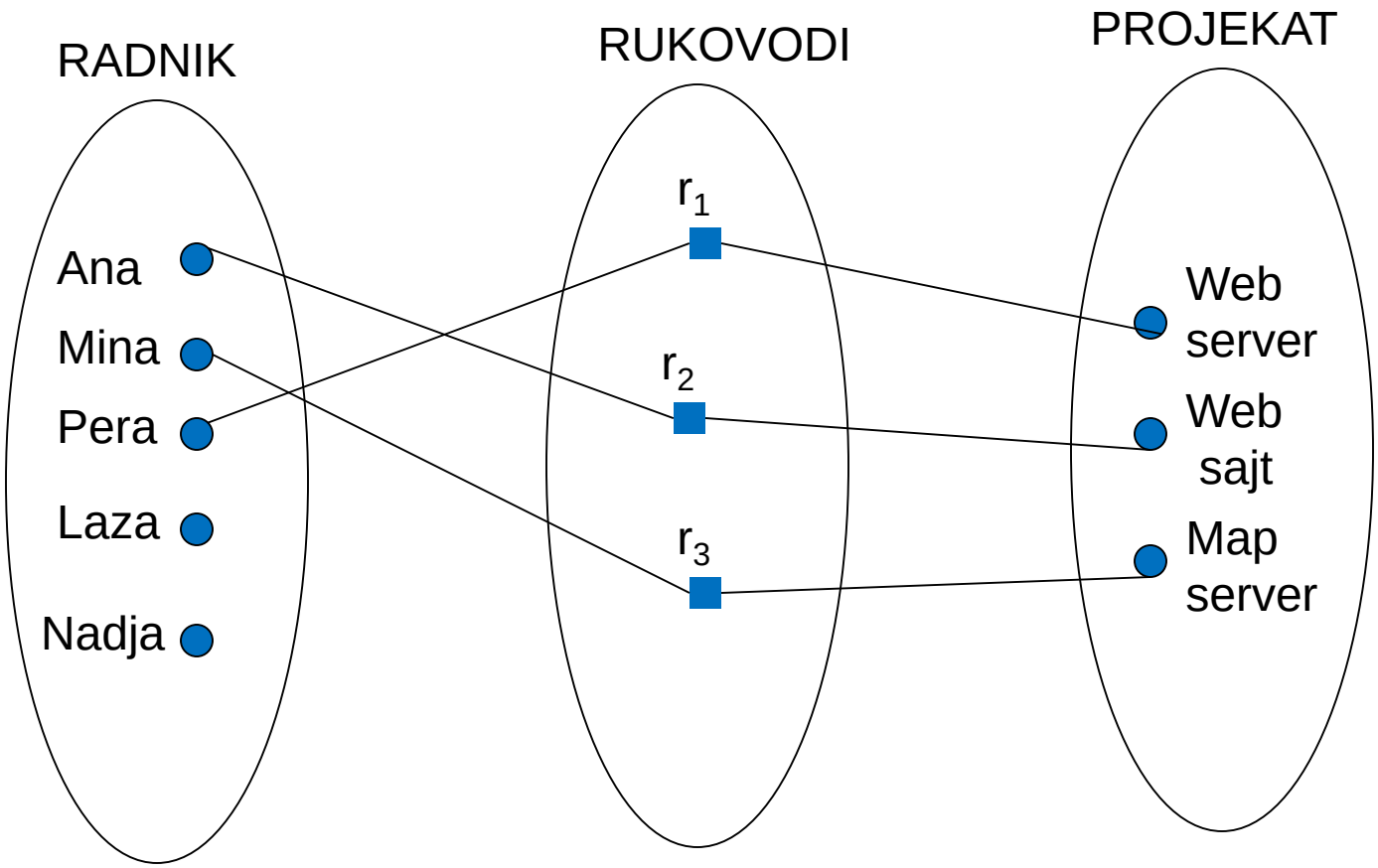


Projekat angažuje radnike

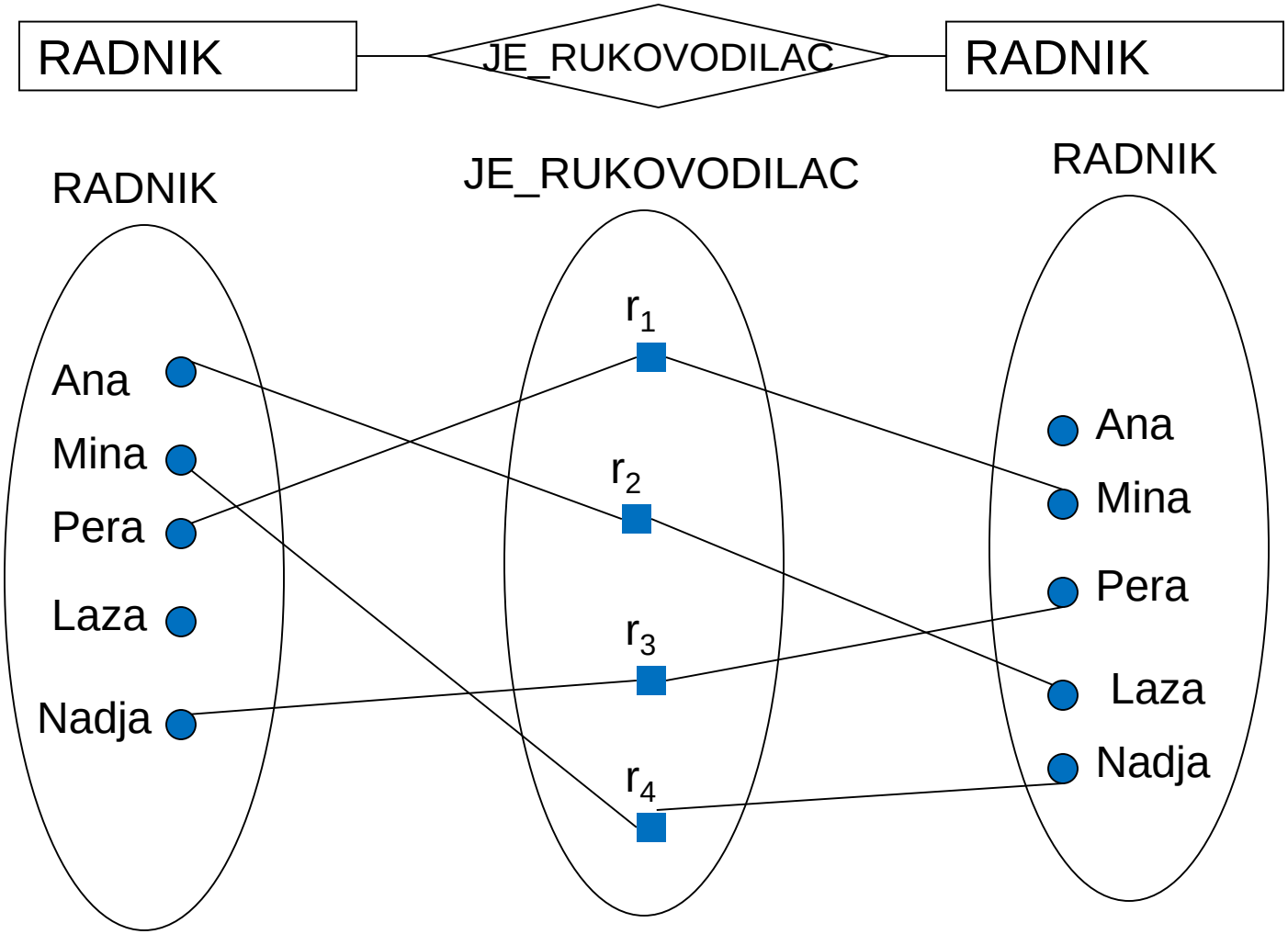
Neke pojave tipa poveznika RADI-NA



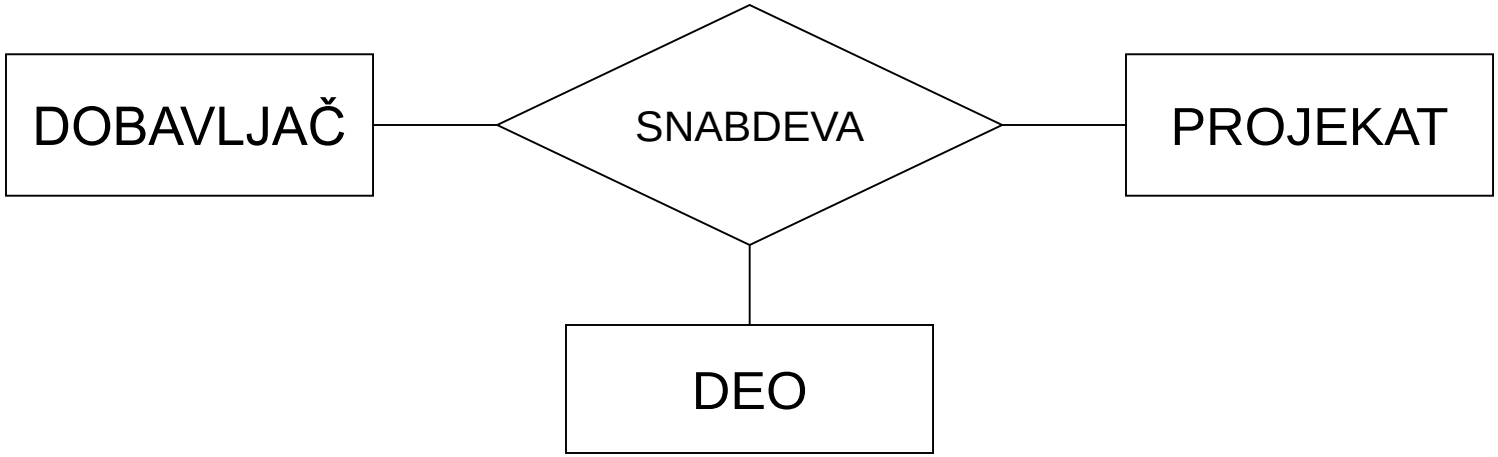
Neke pojave tipa poveznika RUKOVODI



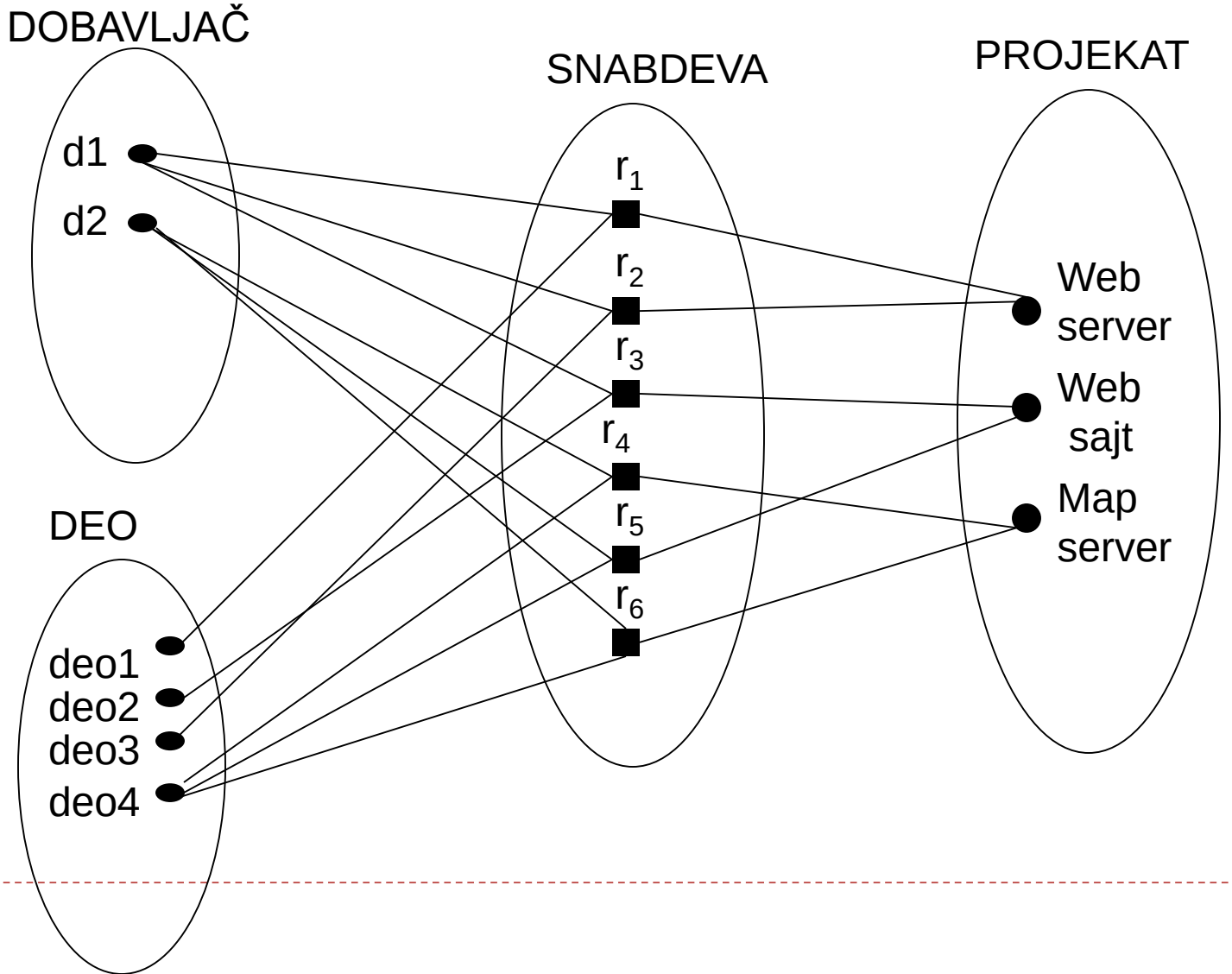
Neke pojave tipa poveznika
JE_RUKOVODILAC



Ternarni tip poveznika SNABDEVA



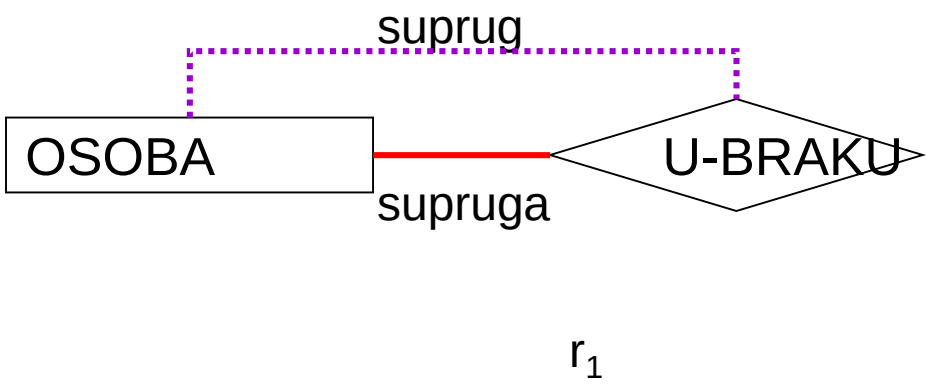
Neke pojave ternarnog tipa poveznika SNABDEVA

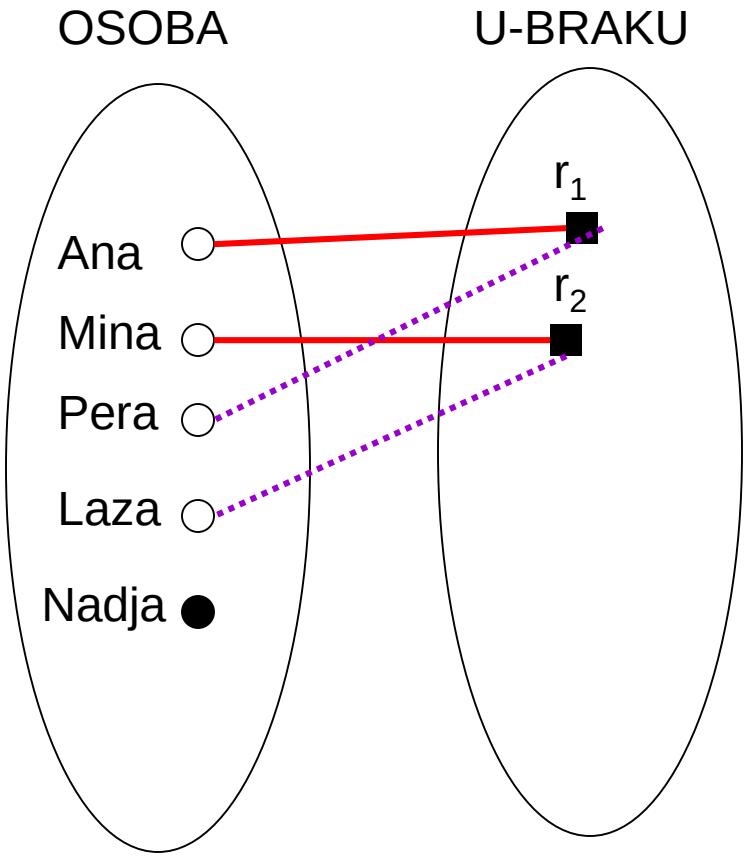


Ime uloge

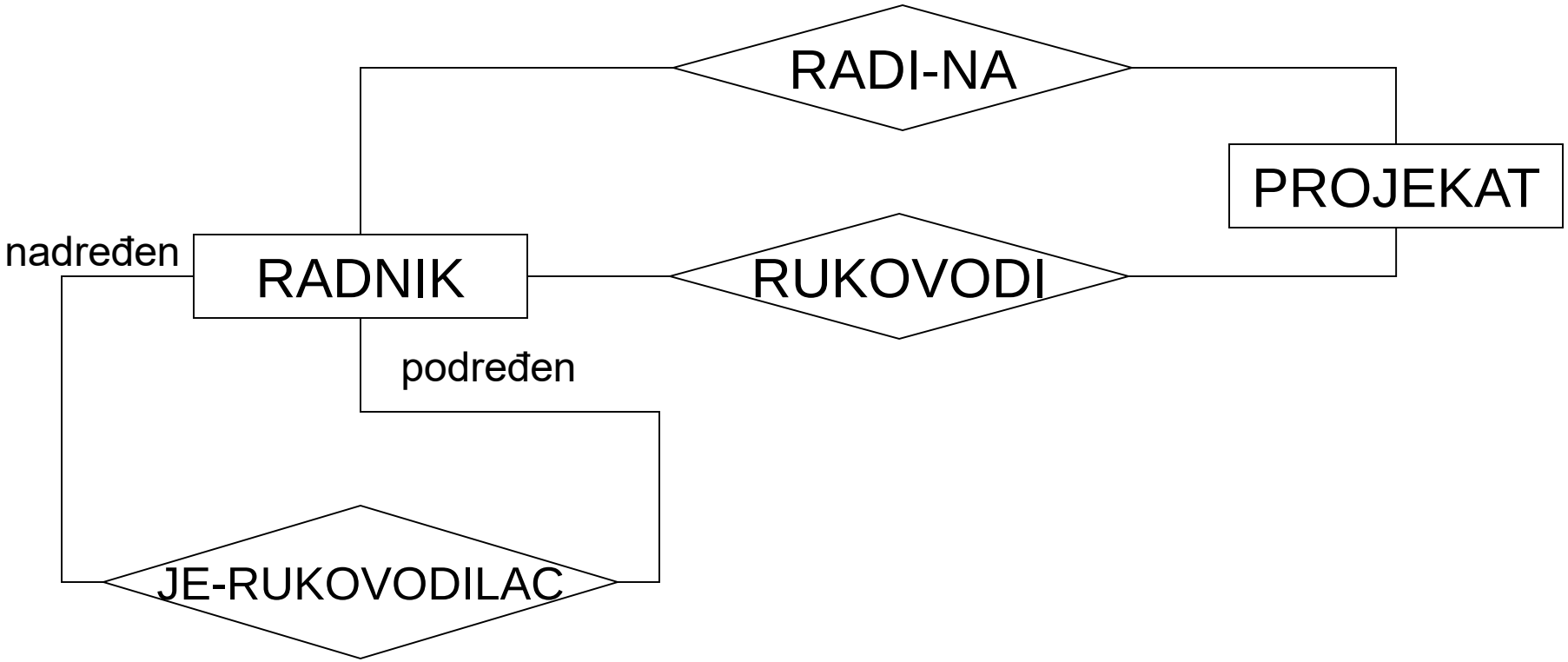
- Svaki tip entiteta koji participira u tipu poveznika ima posebnu **ulogu** u tom odnosu
- **Ime uloge** označava ulogu koju igra entitet koji participira u povezniku u svakoj instanci poveznika
- Pomaže u razumevanju odnosa koji poveznik modelira
- Ime uloge je posebno važno kod rekurzivnih poveznika
- Primer
  - U tipu poveznika RADI-NA(RADNIK,PROJEKAT) tip entiteta RADNIK je u ulozi **zaposlen**, a PROJEKAT u ulozi **poslodavac**
  - U tipu poveznika JE-RUKOVODILAC(RADNIK,RADNIK) tip entiteta RADNIK igra dvojaku ulogu, kao **nadređeni** i **podređeni** radnik u hijerarhiji rukovođenja

Neke pojave rekurzivnog tipa poveznika U-BRAKU





ER dijagram, označavanje uloga



Ograničenja nad tipom poveznika

- Tipovi poveznika imaju izvesna ograničenja koja limitiraju moguće kombinacije entiteta koji mogu participirati u skupu poveznika
- Ova ograničenja omogućavaju modeliranje prirode odnosa koji postoje među entitetima u realnom svetu
- Primer
  - Radnik **može** raditi na više projekata
  - Radnik **može** rukovoditi samo jednim projektom
  - Svaki radnik **mora** raditi bar na jednom projektu
  - Svaki radnik, osim glavnog menadžera preduzeća, **mora** imati rukovodioca

- Dva glavna tipa ograničenja nad tipom poveznika su
  - **Odnos kardinaliteta**
  - **Participacija**

Kardinalnost binarnog tipa poveznika

- Odnos kardinaliteta binarnog tipa poveznika R(E1,E2) specificira **maksimalni broj** instanci poveznika u kojima entiteti E1 i E2 mogu participirati
- Mogući odnosi kardinaliteta tipa poveznika R(E1,E2)
  - 1:1, 1:N, N:1, M:N, gde je N ≥0 i M≥0

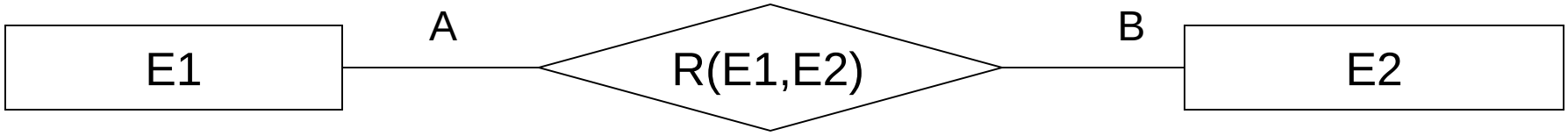
- Primer
  - Tip poveznika

    RADI-U(RADNIK,SEKTOR) ima odnos kardinaliteta N:1

    što znači da:
    - jedan radnik može biti u vezi (u ulozi zaposlenog) samo sa jednim sektorom,
    - ali svaki sektor (u ulozi poslodavca) može biti u vezi sa proizvoljnim brojem radnika (u ulozi zaposlenih)

Predstavljanje kardinaliteta tipa poveznika u ER dijagramima

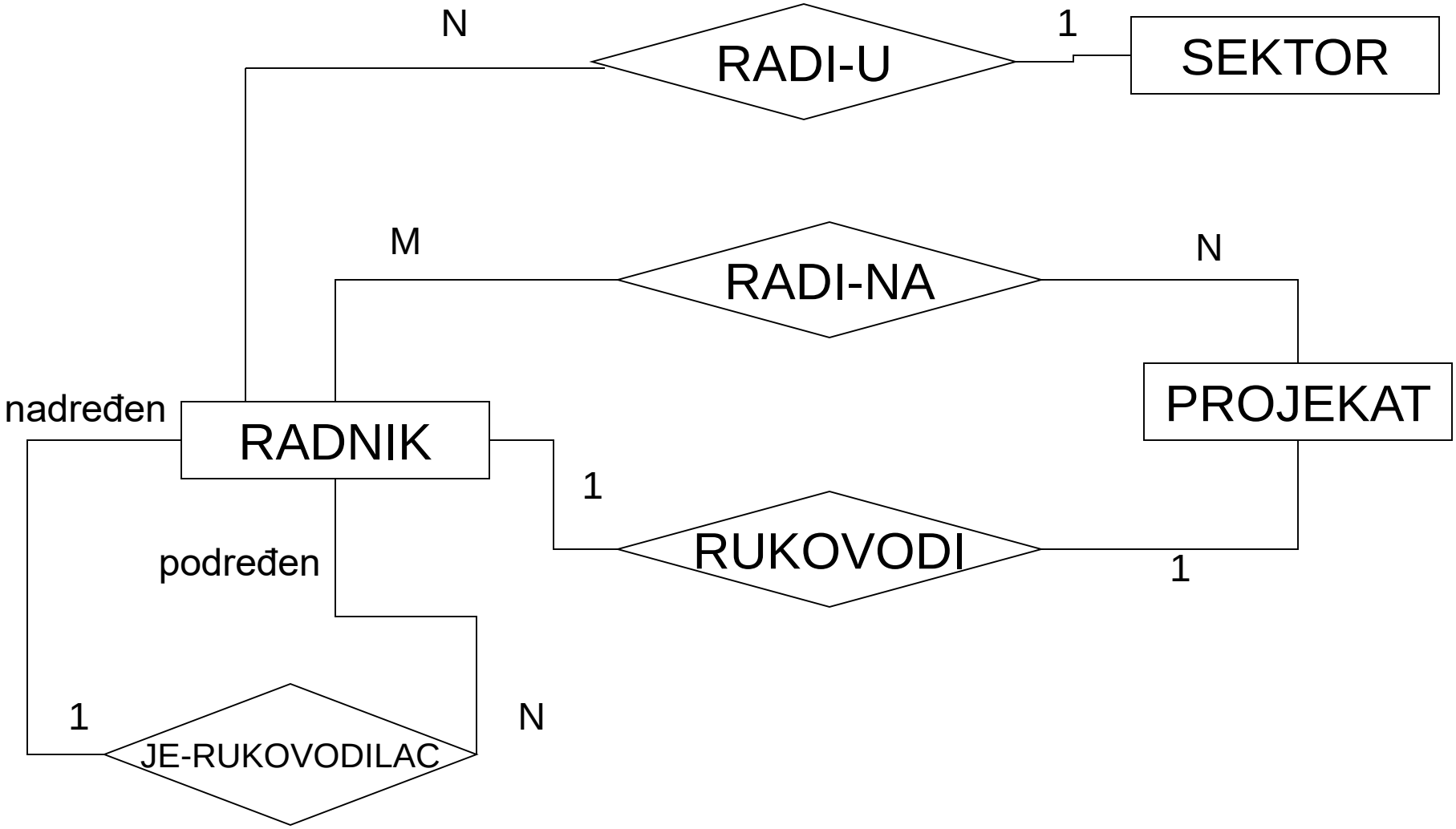
- Označavanje kardinaliteta **A:B** tipa poveznika **R(E1,E2),**
- A se navodi uz tip poveznika E1, a B uz tip poveznika E2 (ili obrnuto u nekim notacijama)



Primer predstavljanja kardinaliteta tipa poveznika u ER dijagramima

- Niže su prikazana ova dva načina označavanja kardinaliteta za tip poveznika RASPOREĐEN(RADNIK,RADNO_MESTO) za realni sistem u kojem:
  - **Radnik** može biti raspoređen samo na jedno radno mesto, a
  - Na jedno **radno mesto** može biti raspoređeno više radnika

ER dijagram sa označenim kardinalitetima tipa poveznika



Ograničenje participacije

- Ograničenje participacije specificira **minimalni broj** instanci poveznika u kojima entitet može participirati
  - Ponekad se naziva **minimalno ograničenje kardinaliteta**

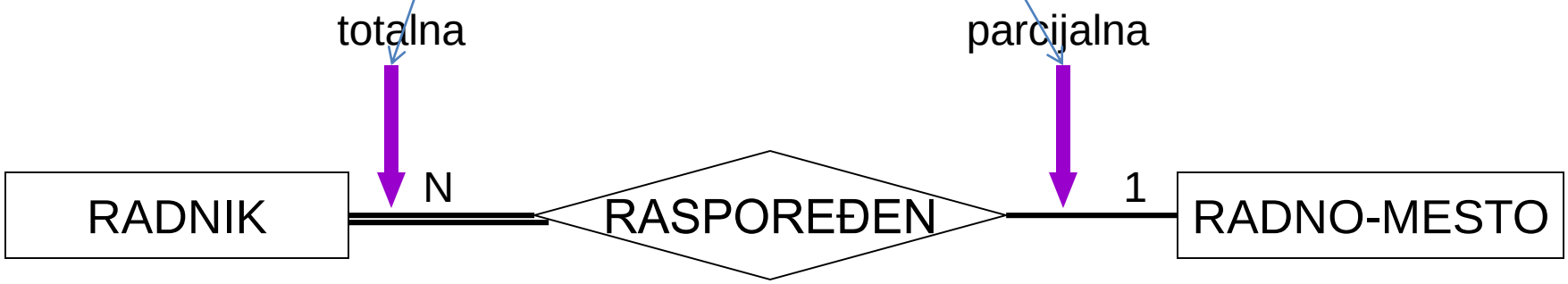
- Postoje dva tipa ograničenja participacije
  - **Totalna (egzistencijalna)** – svaki entitet mora participirati u nekoj instanci tipa poveznika
  - **Parcijalna** – neki entiteti mogu participirati u instancama ipa poveznika

- Ograničenje participacije specificira da li postojanje (egzistencija) entiteta zavisi od toga da li je u vezi sa drugim entitetom preko tipa poveznika

Predstavljanje participacije tipa poveznika u ER dijagramima

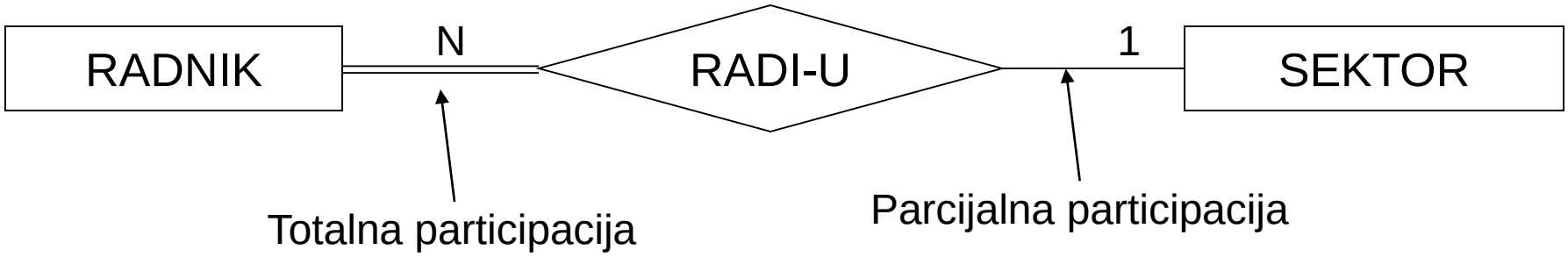
- Za označavanje participacije tipa poveznika R(E1,E2) koristićemo dva različita tipa linija koje spajaju tip entiteta sa tipom poveznika
  - **Jednostruku liniju** za parcijalnu participaciju i
  - **Dvostruku liniju** za totalnu participaciju

- Na primeru su prikazana dva načina za realni sistem u kojem:
  - **Radnik mora biti raspoređen** na tačno jedno radno mesto, a
  - Na jedno radno mesto **može biti raspoređeno** više radnika, ali mogu postojati radna mesta na koja niko nije raspoređen



Primer ograničenja participacije

- U tipu poveznika RADI-U(RADNIK,SEKTOR)
  - tip entiteta **RADNIK ima totalnu participaciju** ako u preduzeću vlada pravilo da svaki radnik mora biti zaposlen u nekom sektoru
  - Tip entiteta **SEKTOR ima parcijalnu participaciju** ako u preduzeću vlada pravilo da mogu postojati sektori bez zaposlenih



Strukturalna ograničenja tipa poveznika ili (min,max) ograničenja

- Posmatra se binarna relacija R između skupova pojava dva tipa entiteta E1 i E2
- Ova relacija se može predstaviti putem dva preslikavanja

  R1: E1 $\rightarrow \mathcal{P}$(E2)

  R2: E2 $\rightarrow \mathcal{P}$(E1)

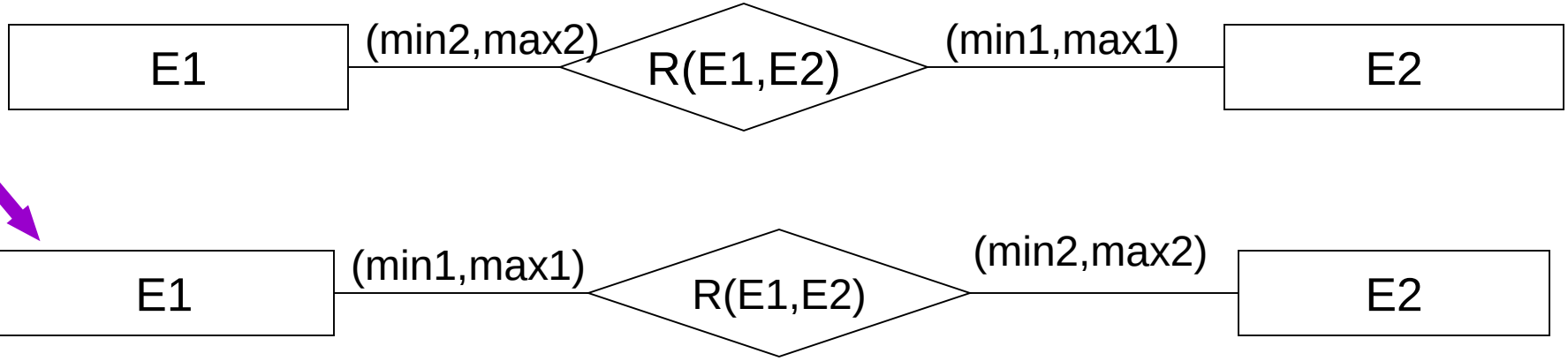
  gde je $\mathcal{P}$(E) partitivni skup skupa E
- Za svako od ovih preslikavanja se definiše minimalni i maksimalni kardinalitet preslikavanja
- Preslikavanjima R1 i R2 se može dati sledeća semantička interpretacija:
  - R1 je uloga entiteta iz skupa E1, a R2 uloga entiteta iz skupa E2 u njhovoj vezi opisanoj relacijom R(E1,E2)

(min,max) ograničenja

- Minimalna vrednost definiše participaciju entiteta u tipu poveznika R(E1,E2)
  - Ako je min=1 preslikavanje je totalno
  - Ako je min=0 preslikavanje je parcijalno
- Maksimalne vrednosti definišu odnos kardinaliteta tipa poveznika R(E1,E2)
  - max1:max2
  - 1:1
  - 1:N
  - N:M

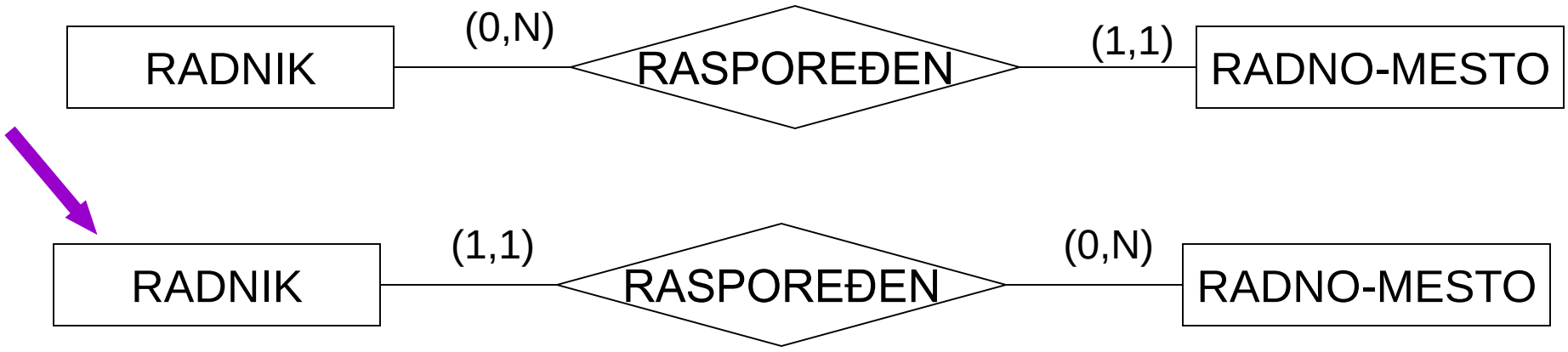
Predstavljanje (min,max) ograničenja tipa poveznika u ER dijagramima

- Za označavanje (min,max) ograničenja tipa poveznika R(E1,E2) postoje dva načina:
  - Brojnost preslikavanja se upisuje uz tip entiteta u koji se on preslikava. Par (min1,max1) se navodi na potegu uz tip poveznika E2, a par (min2,max2) na potegu uz tip poveznika E1 ili
  - Brojnost preslikavanja se upisuje uz sam tip entiteta. Par (min1,max1) se navodi na potegu uz tip poveznika E1, a par (min2,max2) na potegu uz tip poveznika E2



Primer predstavljanje (min,max) ograničenja tipa poveznika u ER dijagramima

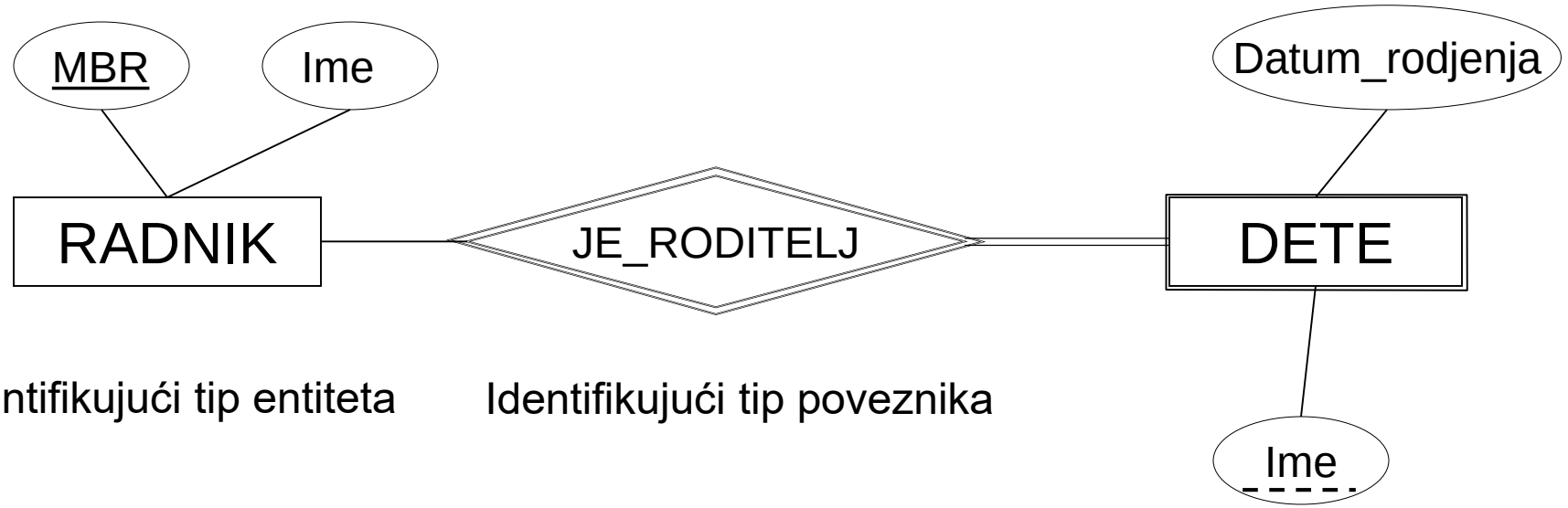
- Na primeru niže prikazana su ova dva načina za isti realni sistem u kojem:
  - Radnik može biti raspoređen samo na jedno radno mesto, a
  - Na jedno radno mesto može biti raspoređeno više radnika, a može radno mesto biti slobodno



Slabi tip entiteta

- Tip etiteta može biti
  - Regularni
  - Slabi
- Tip entiteta koji nema sopstvene ključne atribute je slabi tip entiteta
- Entiteti slabog tipa entiteta se identifikuju preko veze sa nekim drugim tipom entiteta (regularnim ili slabim) u kombinaciji sa nekim svojim atributom
  - Taj drugi tip entiteta je identifikujući (vlasnički, roditeljski ili dominantni) tip entiteta
  - Poveznik koji povezuje slabi tip entiteta sa vlasnikom se naziva identifikujući tip poveznika
  - Slabi tip entiteta uvek totalno participira u identifikujućoj vezi
- Slabi tip entiteta ima **parcijalni ključ** (diskriminator) koji predstavlja podskup skupa atributa slabog tipa entiteta koji na jedinstven način identifikuje slabe entitete koji su u vezi sa istim vlasničkim tipom entiteta

ER dijagram: slabi tip entiteta



Identifikujući tip entiteta

Identifikujući tip poveznika

Parcijalni ključ

Integritetna komponenta ER modela podataka

- ER model ima dobro definisana ograničenja koja se uglavnom eksplicitno definišu
- To su:
  - Integritet domena
  - Kardinalnost tipa poveznika
  - Participacija tipa poveznika
  - Slabi tip entiteta
  - Ključ tipa entiteta

Integritet (ograničenje) domena

- Ovaj tip ograničenja imaju gotovo svi modeli podataka
- Opšti oblik ograničenja domena:
  (tip podatka, dužina podatka, uslov)
- Tip podatka
  - Definiše vrstu znakova putem kojih se izražava vrednost stributa
- Dužina podatka
  - Definiše maksimalni broj znakova koji mogu biti upotrebljeni za izražavanje vrednosti atributa
- Tip podatka i dužina podatka su standardna ograničenja domena
- Često nisu dovoljno precizna, pa se dodaje treća komponenta uslov koja preciznije definiše ograničenje domena

Standardna ograničenja domena

- To su tipovi podataka sa dužinama tih podataka:
  - INTEGER(broj cifara)
  - REAL(broj cifara ispred zapete, broj cifara iza zapete
  - CHARACTER(broj znakova)
  - DATE
  - LOGICAL
  - ...
- Primer:
  - Ako je dom(OCENA) pridruženo standardno ograničenje INTEGER(2), tada važi dom(OCENA)={0,1,2,...,99}
  - Ako je dom(NAZIV_PREDMETA) pridruženo standardno ograničenje CHARACTER(15), tada važi da naziv predmeta može biti bilo koji niz dužine 15 znakova

Null vrednost kao specijalni tip ograničenja domena

- Ovim ograničenjem se specificira da li atribut može imati nedefinisanu vrednost
- Ima dva značenja:
  - **Postojeća, ali nepoznata vrednost**
  - **Neprimereno svojstvo**

Pregled metodologije konceptualnog projektovanja korišćenjem ER/EER modela podataka

1. Identifikovati tipove entiteta
2. Identifikovati tipove poveznika
3. Identifikovati i pridružiti atribute tipovima entiteta i poveznika
4. Utvrditi domene atributa
5. Utvrditi atribute koji čine kandidate za ključeve, primarni ključ i alternativne ključeve
6. Razmotriti korišćenje naprednih koncepata modeliranja - EER (opciono)
7. Proveriti model podataka na redundancu
   - Radi provere da li ima redundance u modelu
8. Validirati konceptualni model u odnosu na transakcije korisnika
   - Radi provere da projektovani model podržava zahteve transakcija korisnika
9. Pregledati konceptualni model podataka sa korisnikom
   - Radi provere da su zahtevi korisnika podržani

Projektovanje baze podataka

Korak 1: Izgraditi konceptualni model podataka

- **Korak 1.1** Identifikovati tipove entiteta
- **Korak 1.2** Identifikovati tipove poveznika
- **Korak 1.3** Identifikovati i pridružiti atribute tipovima entiteta i poveznika
- **Korak 1.4** Utvrditi domene atributa
- **Korak 1.5** Utvrditi atribute koji čine kandidate za ključeve, primarni ključ i alternativne ključeve
- **Korak 1.6** Razmotriti korišćenje naprednih koncepata modeliranja (opciono)
- **Korak 1.7** Proveriti model podataka na redundancu
- **Korak 1.8** Validirati konceptualni model u odnosu na transakcije korisnika - radi provere da projektovani model podržava zahteve transakcija korisnika
- **Korak 1.9** Pregledati konceptualni model podataka sa korisnikom - radi provere da su zahtevi korisnika podržani

Projektovanje baze podataka

Korak 2: Izgraditi i proveriti logički model podataka

Korak 2.1 Pronaći relacije za logički model podataka
- Preslikati konceptualni u relacioni model podataka koristeći poznati algoritam

Korak 2.2 Validirati relacije korišćenjem normalizacije
- Prevesti šeme relacija u što je moguće višu normalnu formu

Korak 2.3 Validirati relacije u odnosu na transakcije korisnika

Korak 2.4 Proveriti ograničenja integriteta

Korak 2.5 Pregledati logički model podataka sa korisnikom

Korak 2.6 Integrisati logičke modele podataka u globalni logički model podataka *(opciono)*

Korak 2.7 Proveriti model podataka na budući rast

Projektovanje baze podataka

Korak 3: Prevesti logički model podataka na ciljni DBMS

- **Korak 3.1** Projektovati bazne relacije
- **Korak 3.2** Projektovati reprezentaciju izvedenih podataka
- **Korak 3.3** Projektovati opšta ograničenja

Korak 4: Projektovati organizaciju fajlova i indekse

- **Korak 4.1** Analizirati transakcije
- **Korak 4.2** Izabrati organizacije fajlova
- **Korak 4.3** Izabrati indekse
- **Korak 4.4** Proceniti potreban prostor na diskovima

Korak 5: Projektovati poglede korisnika

Korak 6: Projektovati bezbedonosne mehanizme

Korak 7: Razmotriti uvođenje kontrolisane redundance

Korak 8: Pratititi (*monitoring*) sistem koji je pušten u eksploataciju i vršiti njegovo podešavanje (tuning)

Baze podataka 2020/2021

Katedra za računarstvo
Elektronski fakultet u Nišu

ER model - Primeri

Prof. dr Leonid Stoimenov
Doc. dr Aleksandar Stanimirović

Model entiteta i veza

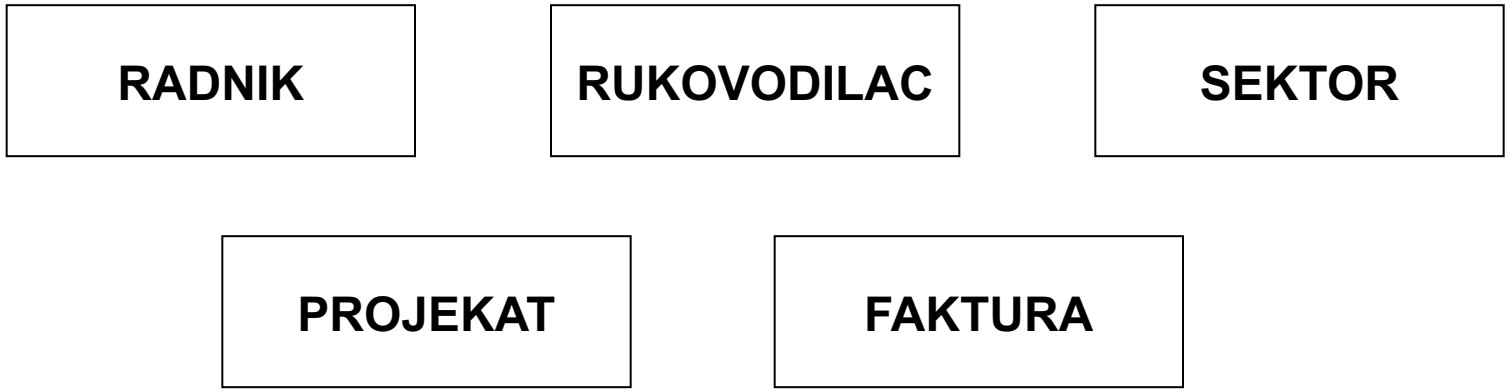
- **Tvorac modela je Peter Chen (1976)**
- **Osnovna ideja**
  - Realan savet (ili njegov deo) može se opisati pomoću dva primitivna koncepta: entitet i veza.
  - **Entitet** je bilo koji objekat koji se može jednoznačno identifikovati.
  - **Veza** je relacija između dva ili više entiteta.
- **Namena modela**
  - Za formiranje konceptualnog (logičkog) modela podataka.
  - Pogodno sredstvo za komunikaciju između korisnika i analitičara i projektanta softvera.

Model entiteta i veza

- Osnovni koncepti ER model su:
  - entiteti i tip entiteta; jaki i slabi tip entiteta
  - atributi i vrednost atributa; ključni atribut; domen atributa
  - veze i tip veze; specijalni tipovi veza
  - ER dijagram (grafička reprezentacija ER modela)
- Entitet je subjekat, objekat, pojam, događaj ili stanje o kojima se prikupljaju, obrađuju i prezentiraju podaci u automatizovanim informacionim sistemima a koji se može jednoznačno identifikovati.

ER dijagram - notacija

- Entitet se može identifikovati imenom i listom svojstava.
- U ER dijagramu tip eniteta se crta kao pravougaonik sa upisanim imenom. Ime se upisuje velikim slovima.

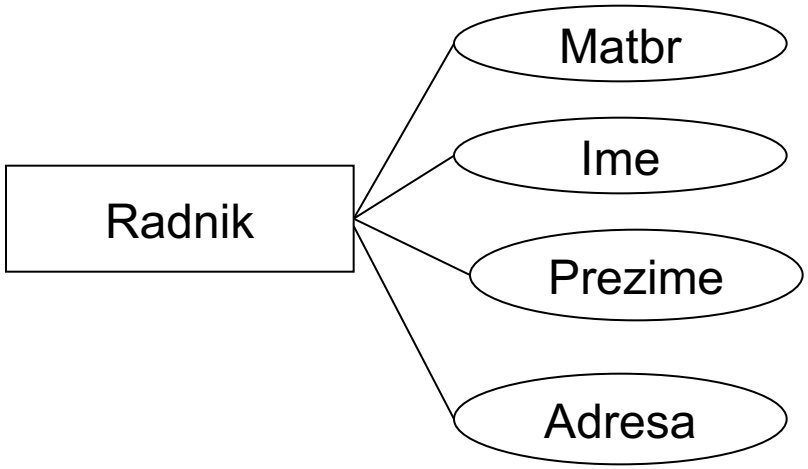


ER dijagram – notacija

- Atributi predstavljaju zajedničke osobine koje poseduju svi entiteti jednog skupa entiteta.
- Domen atributa predstavlja skup vrednosti koje neki atribut može da ima.
- Domen atributa se definiše tipom podataka, dužinom podataka i opsegom vrednosti.

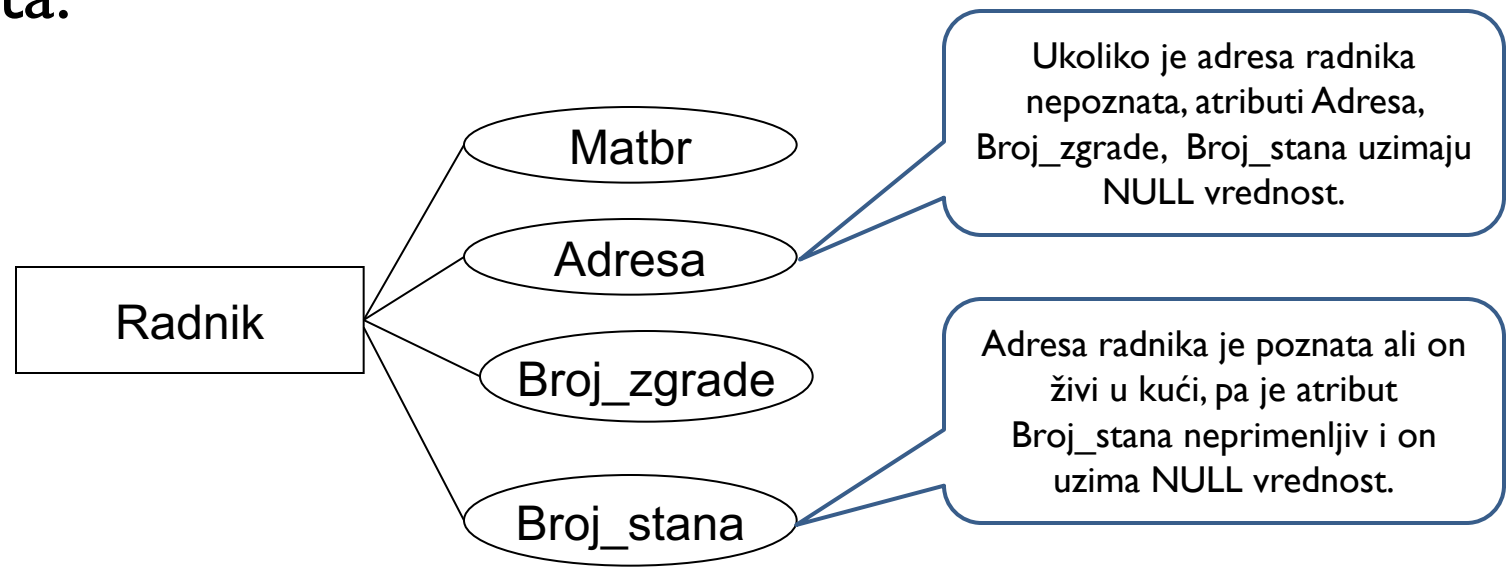
ER dijagram – notacija

- U ER dijagramu atributi se prikazuju kao elipse sa upisanim nazivom i povezuju se neusmernim potegom sa tipom entiteta ili tipom veze na koji se odnose.



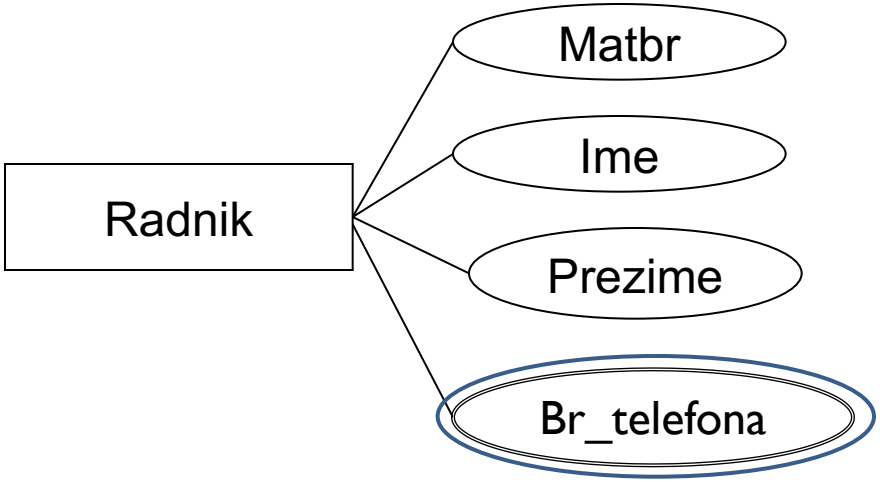
ER dijagram - notacija

- Neki atributi mogu biti bez vrednosti za neki entitet ili mogu biti neprimenjivi za konkretne entitete. (**Primer:** Adresa).
- Tada se za taj entitet kreira NULL vrednost tog atributa.



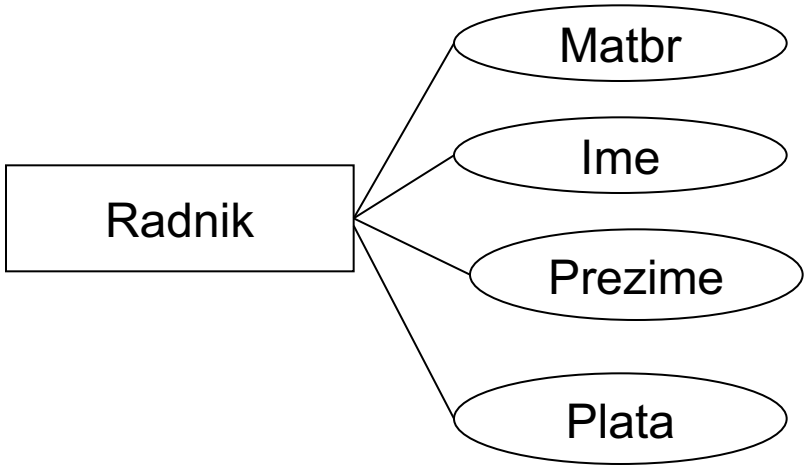
ER dijagram - notacija

- Jednovrednosni atribut je atribut koji za pojavu određenog entiteta može uzeti samo jednu vrednost (**Primer:** Matbr – samo jedan matični broj, ime – osoba ima samo jedno ime)
- Viševrednosni atribut je atribut za pojavu određenog eniteta može uzeti više vrednosti (**Primer:** Br_telefona – radnik može imati više brojeva).



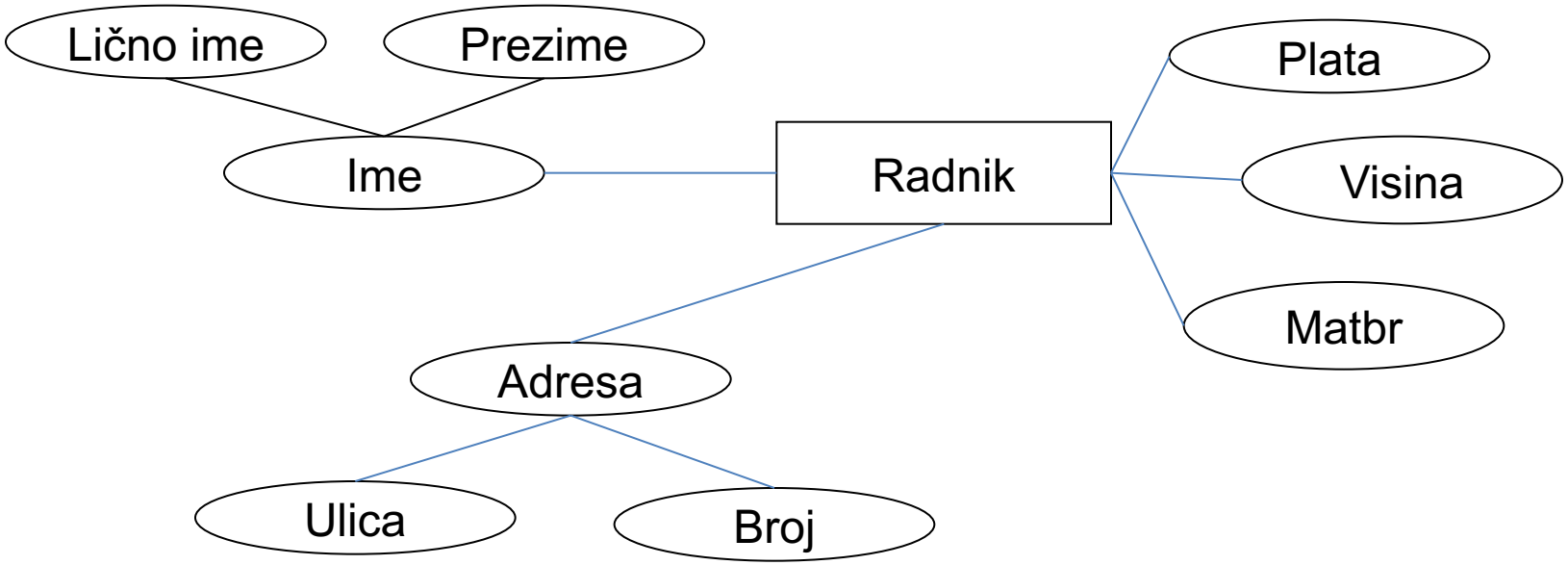
ER dijagram - notacija

- Prost atribut je atribut koji se dalje ne može dekomponovati odnosno ne može doći do razdvojene primene komponenti atributa. (**Primer:** Plata, Visina, MBR.)
- Vrednost prostog atributa je prost podatak.



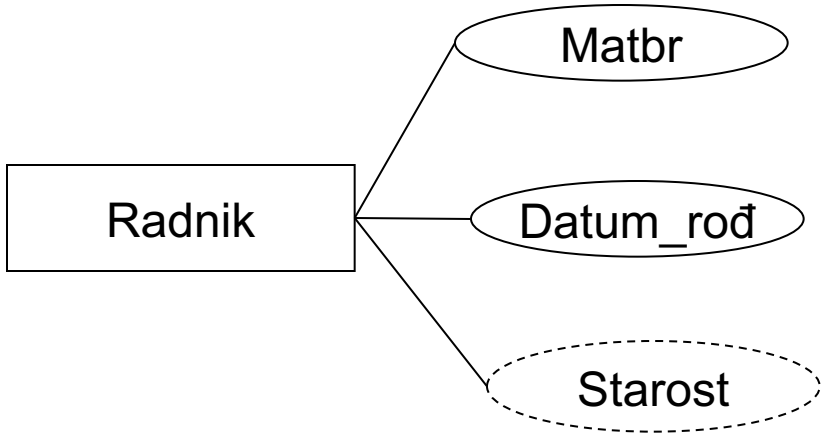
ER dijagram - notacija

- Složeni atribut je atribut koji se sastoji od niza prostih atributa (**Primer:** Ime, Adresa).
- Vrednost složenog atributa je strukturni podatak.



ER dijagram - notacija

- Izvedeni atributi su atributi čija se vrednost može dobiti iz vrednosti drugih atributa. (**Primer:** Starost = Tekući_datum – Datum_rođ)
- Vrednost izvedenog atributa je izvedeni podatak.
- Obično se ne čuvaju u bazi podataka.

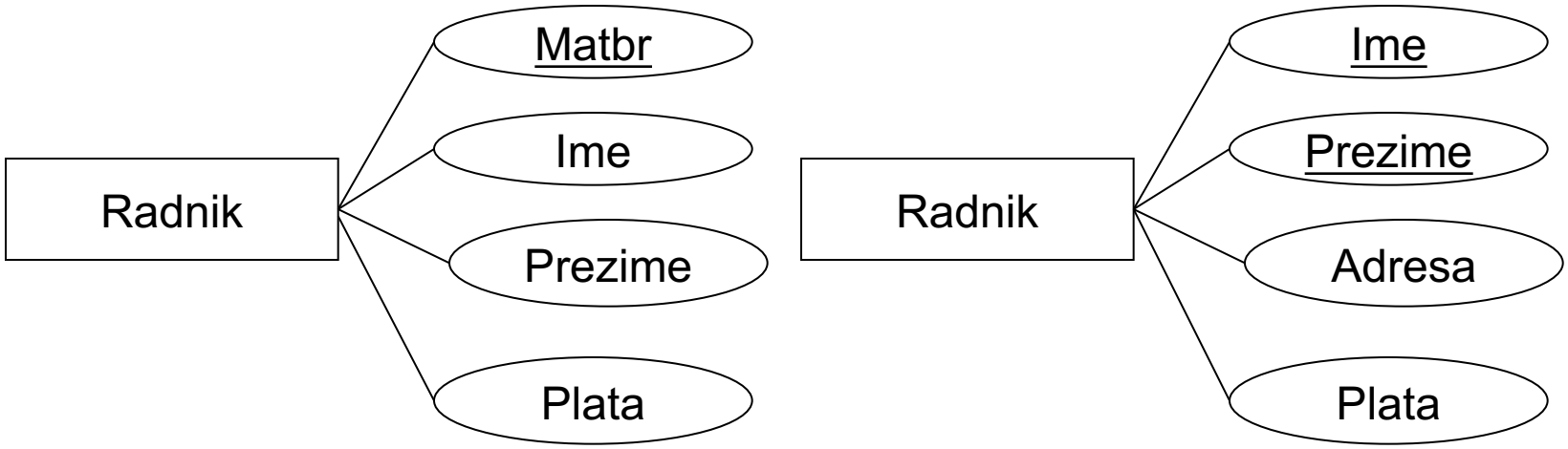


ER dijagram - notacija

- Ključ entiteta predstavlja atribut ili skup atributa čije vrednosti jednoznačno identifikuju svaku pojavu entiteta (**Primer:** Matbr).
- Uklanjanjem bilo koje komponente (atributa) ključa, on gubi svoje svojstvo identifikacije.
- Surogat ključ - Uvodi se tamo gde ne možemo odrediti prirodan podskup atributa koji bi činili ključ (Id, Redni broj, Broj indeksa i sl.).

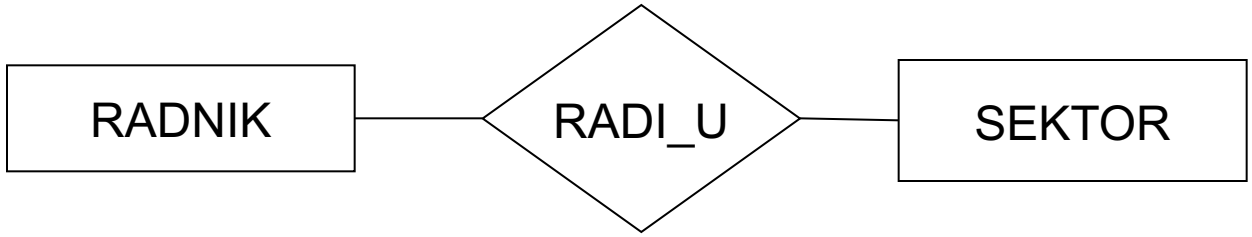
ER dijagram - notacija

- Kod ključnih atributa naziv atributa je podvučen (**Primer**: MBR, Ime i Prezime).



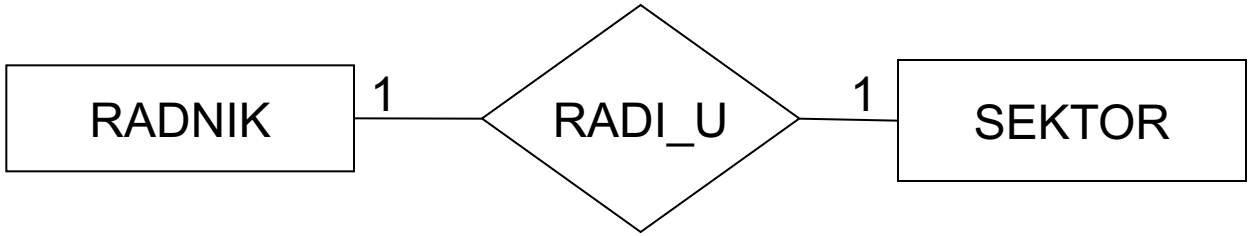
ER dijagram notacija

- Tip veze modelira relacije između entiteta u istom ili različitim skupovima.
- **Veza uvek funkcioniše u oba smera** (Ako je RADNIK u vezi sa SEKTOROM važi i obratno).
- U ER dijagrami veza se predstavlja kao romb u koji se upisuje ime te veze.



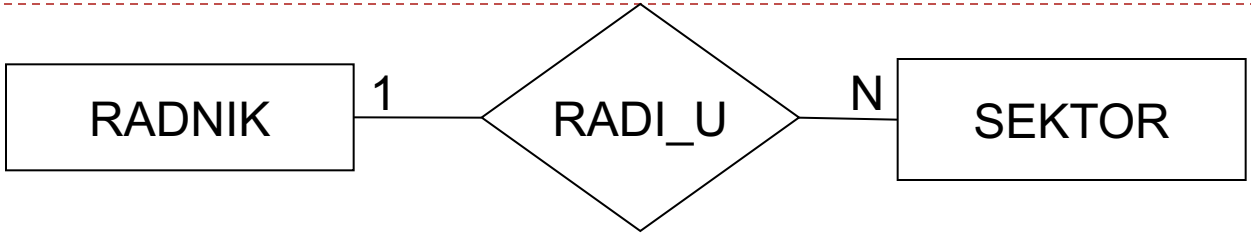
ER dijagram - notacija

- Kardinalnost veze definiše broj entiteta jedne vrste koji su u vezi sa određenim brojem entiteta druge vrste.
  - Jedan-prema-jedan (1:1)
  - Jedan-prema-više (1:N) i Više-prema-jedan (N:1)
  - Više-prema-više (M:N)

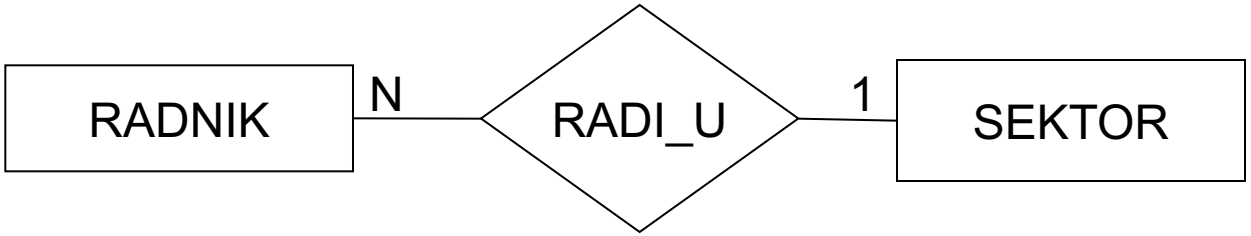


Radnik radi u **jednom** sektoru. Sektor ima **jednog** radnika.

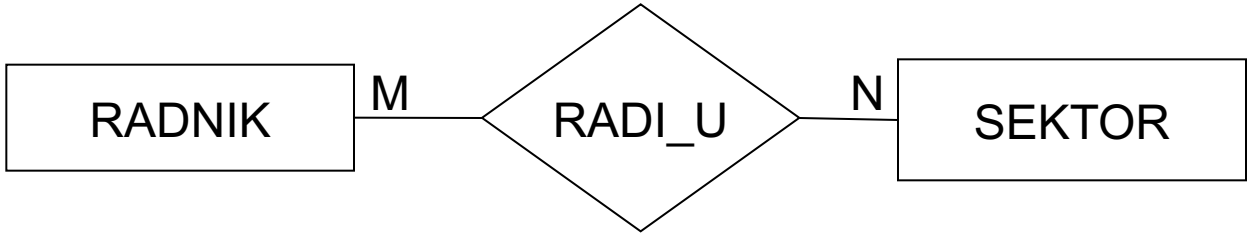
ER dijagram - notacija



Radnik radi u **više** sektora. Sektor ima **jednog** radnika.



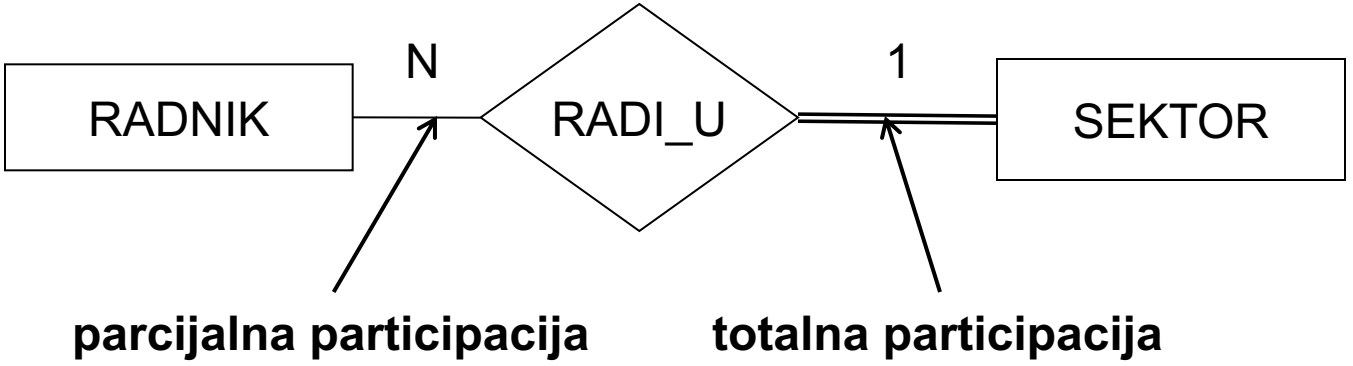
Radnik radi u **jednom** sektoru. Sektor ima **više** radnika.



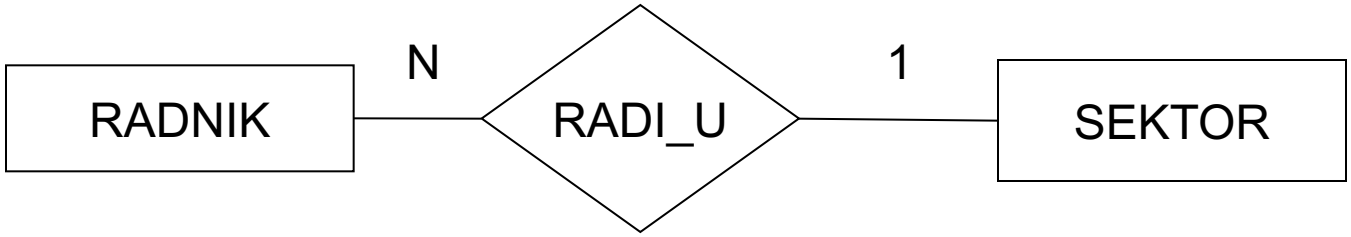
Radnik radi u **više** sektora. Sektor ima **više** radnika.

ER dijagram - notacija

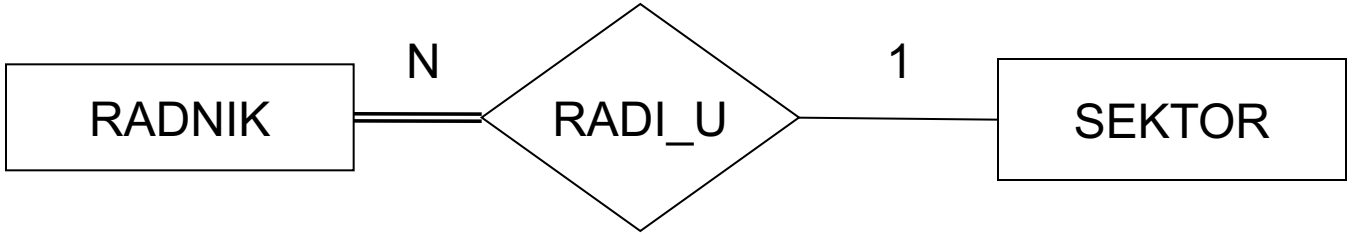
- Participacija entiteta u vezi definiše da li svi entiteti određenog tipa učestvuju u vezi ili ne.
  - Totalna (egzistencijalna) participacija
  - Parcijalna participacija



ER dijagram - notacija

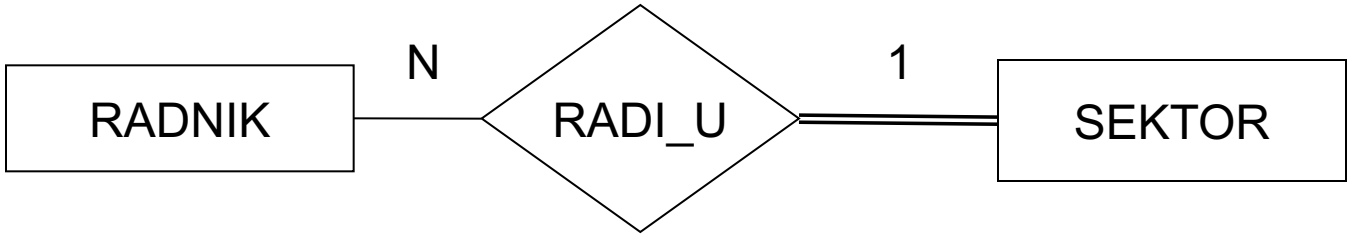


Radnik **ne mora** da radi ni u jednom sektoru a **može** da radi najviše u jednom sektoru.
Sektor **ne mora** da ima ni jednog radnik a **može** da ima više radnika.

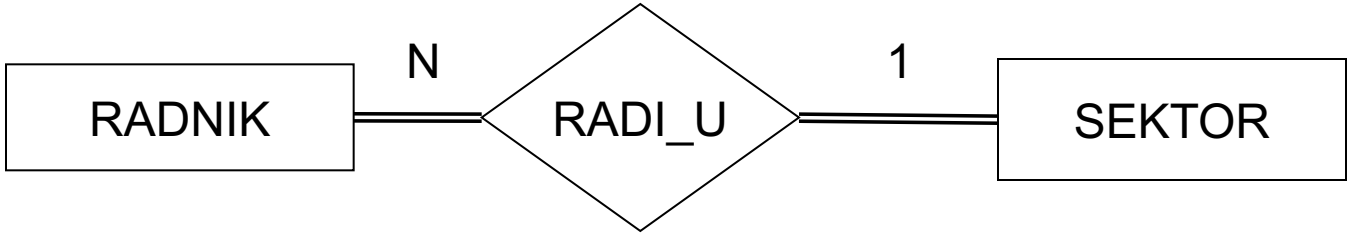


Radnik **mora** da radi u jednom sektoru i **može** da radi najviše u jednom sektoru.
Sektor **ne mora** da ima ni jednog radnik a **može** da ima više radnika.

ER dijagram - notacija



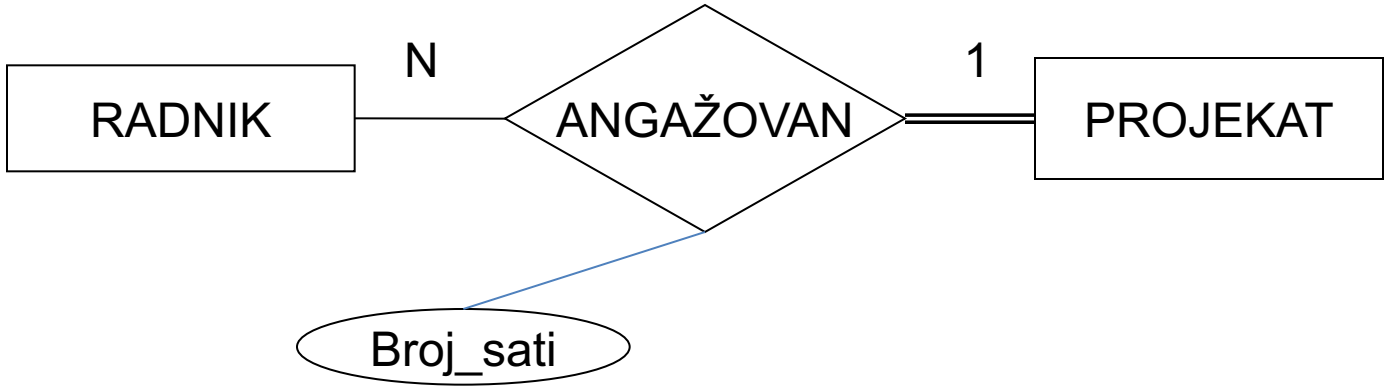
Radnik **ne mora** da radi ni u jednom sektoru a **može** da radi najviše u jednom sektoru.
Sektor **mora** da ima bar jednog radnik a **može** da ima više radnika.



Radnik **mora** da radi u jednom sektoru i **može** da radi najviše u jednom sektoru.
Sektor **mora** da ima bar jednog radnik a **može** da ima više radnika.

ER dijagram - notacija

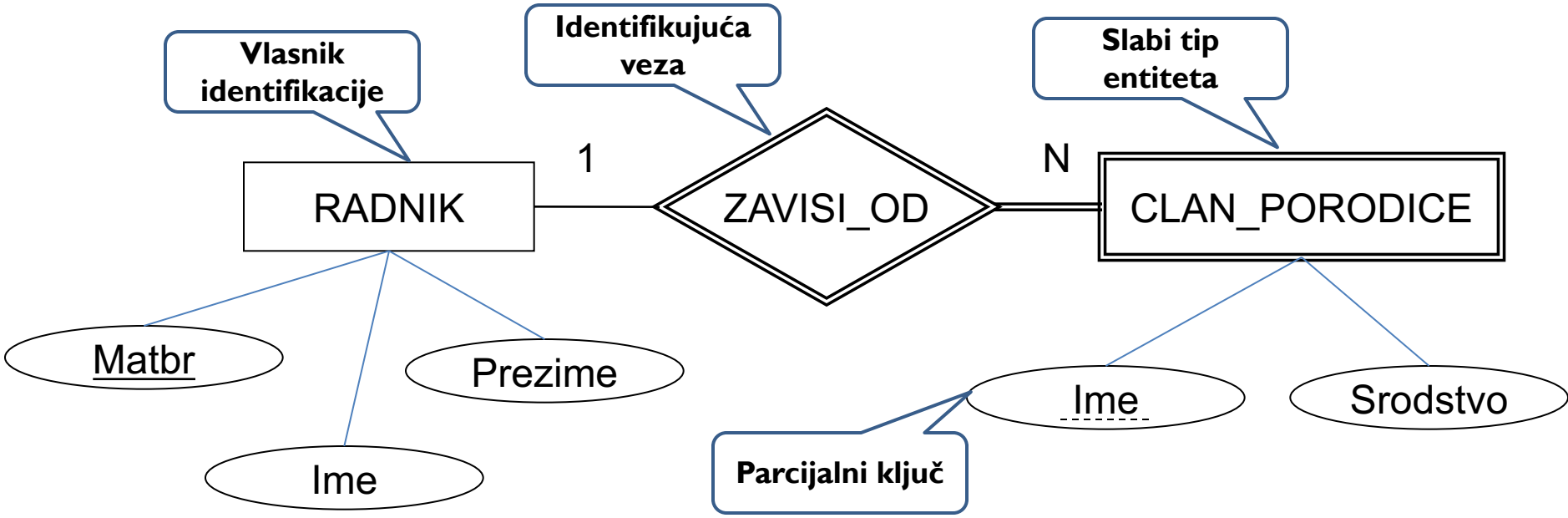
- Atributi veze predstavljaju zajedničku osobinu koju imaju sve veze određenog tipa.



ER dijagram - notacija

- Slabi tip entiteta je tip entiteta za koji je nemoguće odrediti ključ. (**Primer**: CLAN_PORODICE)
- Mora biti povezan sa tipom entiteta koji ima sposobnost identifikacije – vlasnik identifikacije. (**Primer**: RADNIK)
- Identifikujuća veza je veza između slabog tipa entiteta i vlasnika identifikacije. (**Primer**: ZAVISI_OD)
- Parcijalni ključ je skup atributa slabog tipa entiteta koji zajedno sa primarnim ključem vlasnika identifikacije omogućava jednoznačnu identifikaciju entiteta slabog tipa. (**Primer:** Ime)

ER dijagram - notacija



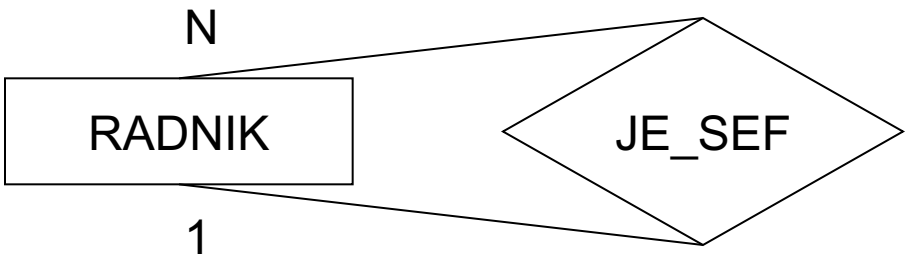
Identifikujuća veza sa strane slabog tipa entiteta mora biti totalna (svaki instanca slabog tipa entiteta mora biti u vezi sa instancom vlasnika identifikacije)
Kardinalost na vlasnika identifikacije mora biti 1 (slabi tip entiteta može imati samo jednog vlasnika identifikacijne)

ER dijagram - notacija

- Stepen tipa veze je broj entiteta koji učestvuju u vezi:
  - unarna (rekurzivna veza)
  - binarna (povezuje dva tipa entiteta)
  - ternarna (povezuje tri tipa entiteta)
  - n-arna (povezuje više od tri tipa entiteta)

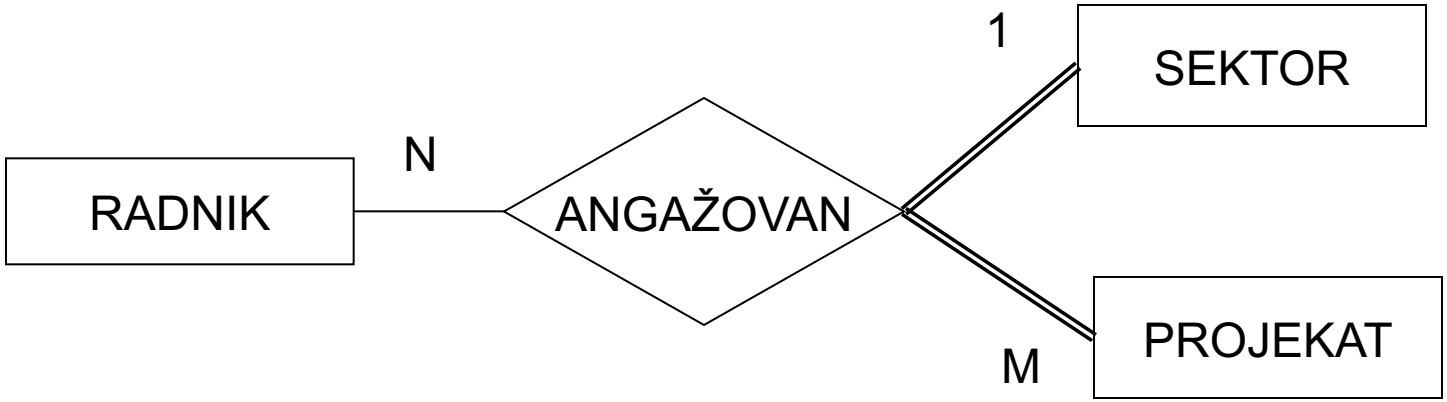
ER dijagram - notacija

Unarna veza



Binarna veza

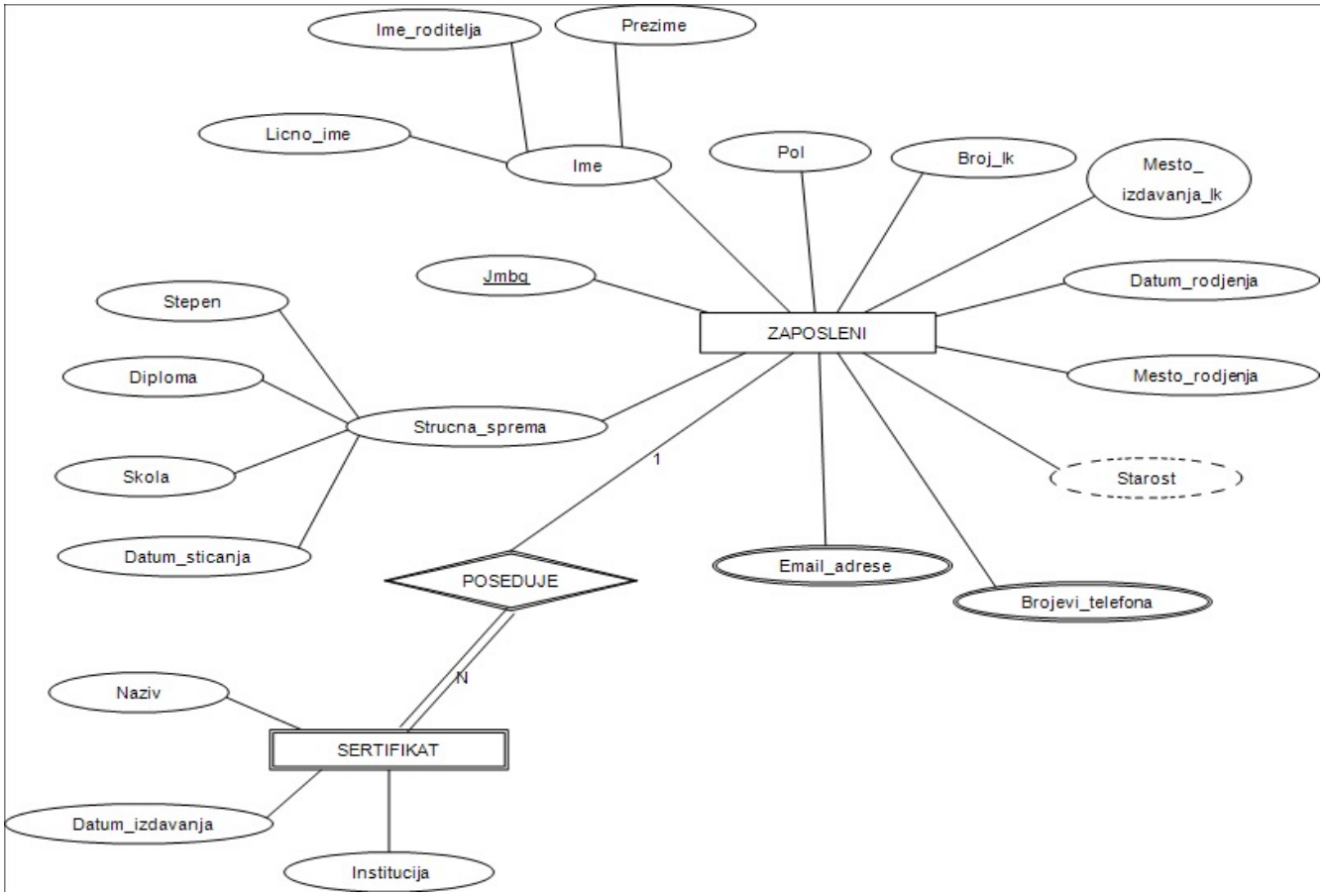
Ternarna veza



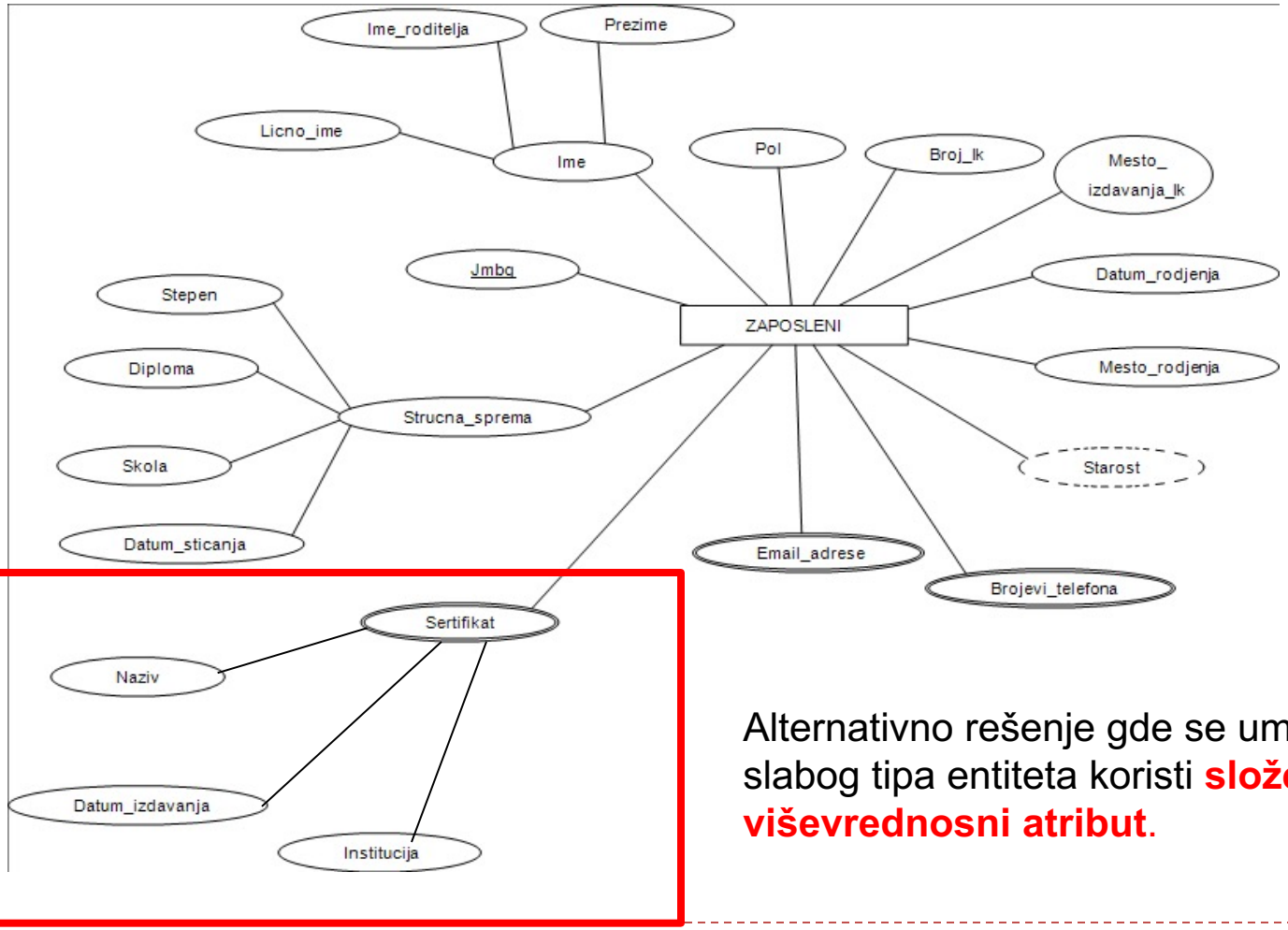
ER dijagram - Atributi

- Za sve stalno zaposlene radnike u banci pamti se: JMBG, ime (lično ime, ime roditelja, prezime), pol, broj lične karte i mesto izdavanja, broj radne knjižice i mesto izdavanja, datum i mesto rođenja, starost, adresa stanovanja, brojevi telefona, e-mail adrese.
- Osim toga za sve zaposlene se pamti stručna sprema koju poseduju, diploma, naziv škole koja je izdala diplomu i datum sticanja diplome.
- Pored toga toga radnici mogu da imaju i dodatne sertifikate o osposobljenosti za pojedine poslove. Za svaki sertifikat koji zaposleni poseduje pamti se naziv sertifikata, datum izdavanja i naziv institucije koja je sertifikat izdala.

ER dijagram - Atriibuti



ER dijagram - Atributi



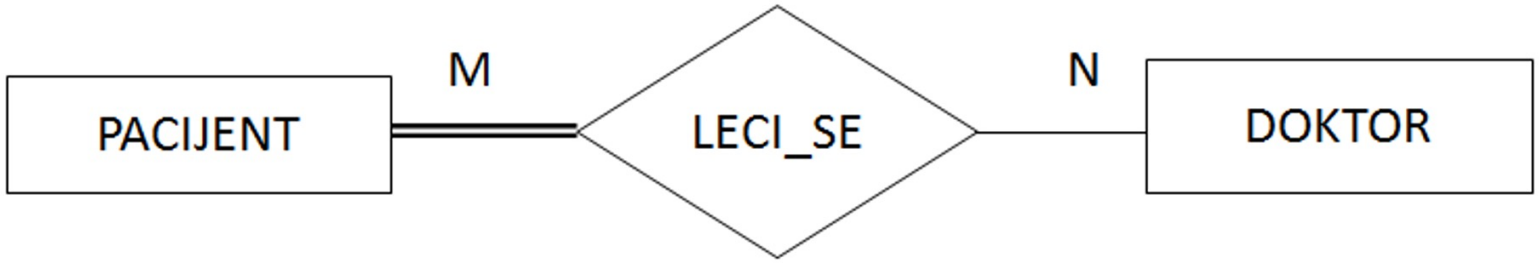
Alternativno rešenje gde se umesto slabog tipa entiteta koristi **složeni viševrednosni atribut**.

ER dijagram - Veze

- Pacijent može da ima više doktora, i jedan doktor ima više pacijenata.
- Fakultet ima više studenata, ali jedan student može da studira samo na jednom fakultetu.
- Avion može da ima više putnika, ali jedan putnik može da bude na samo jednom letu u određeno vreme.
- Država ima više regiona, svaki region ima više gradova.
- Ćevapdžinica prodaje nekoliko vrsti pljeskavica, i isti dodaci (salate) se mogu koristiti uz različite tipove pljeskavica.

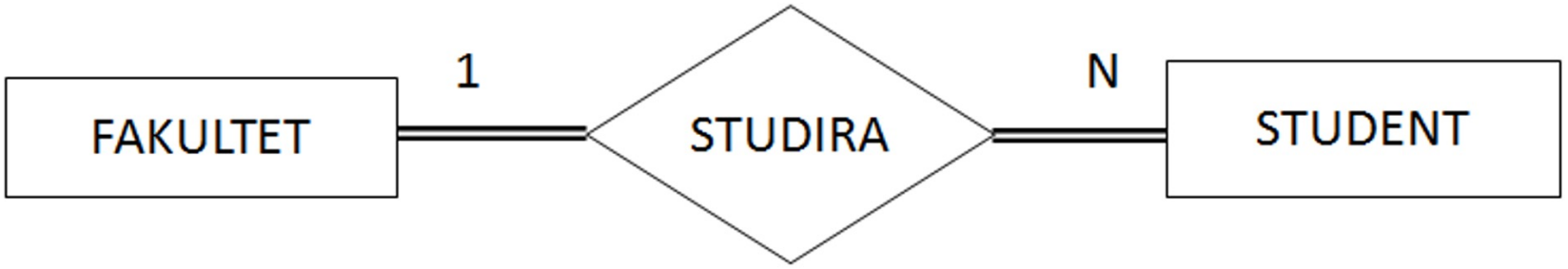
ER dijagram - Veze

- Pacijent može da ima više doktora, i jedan doktor ima više pacijenata.



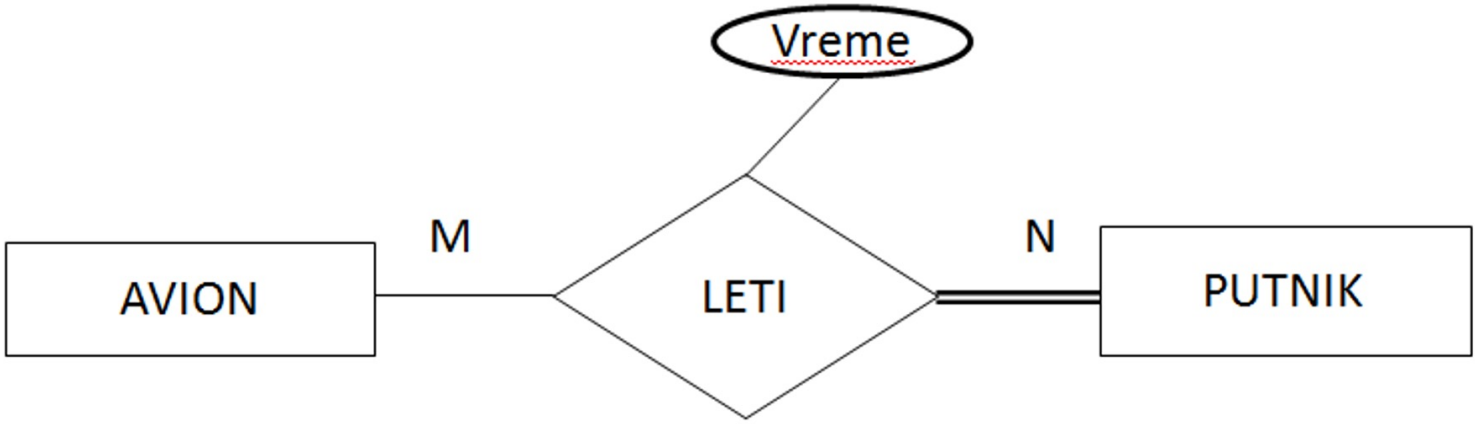
ER dijagram - Veze

- Fakultet ima više studenata, ali jedan student može da studira samo na jednom fakultetu.



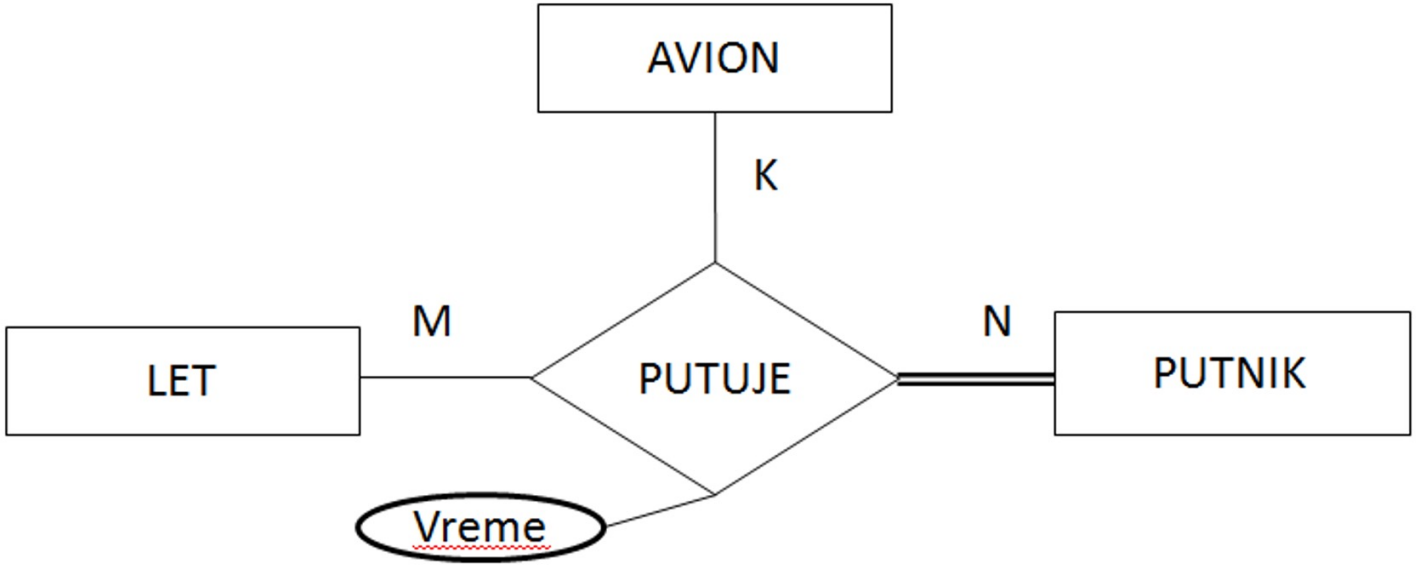
ER dijagram - Veze

- Avion može da ima više putnika, ali jedan putnik može da bude na samo jednom letu u određeno vreme.



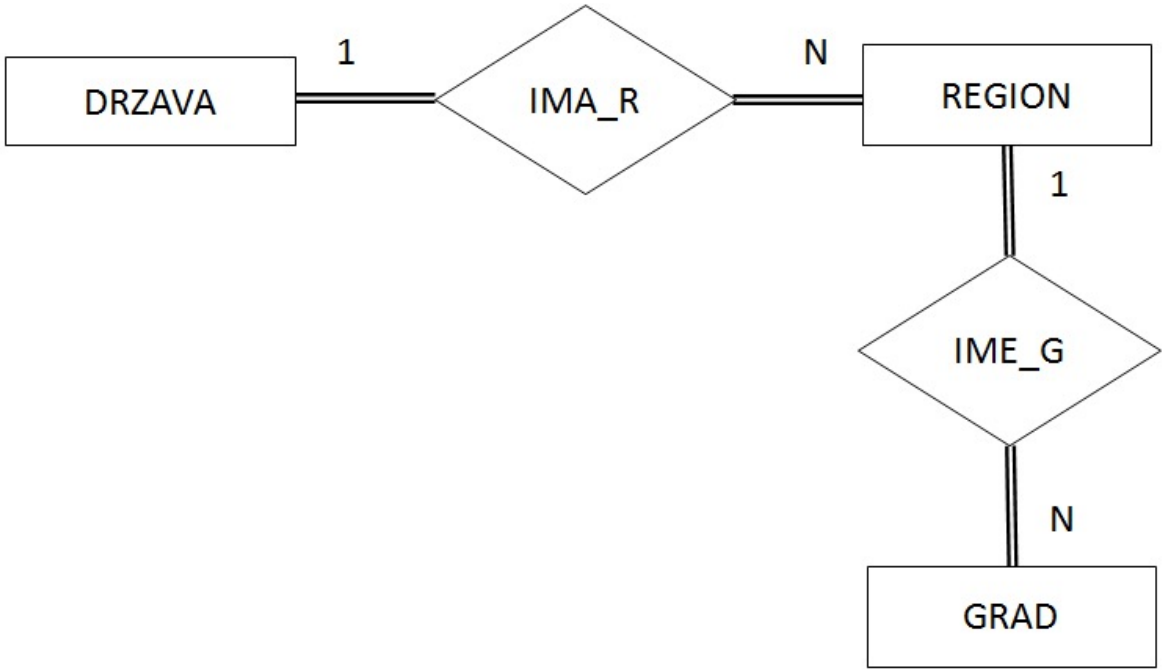
ER dijagrami - Veze

- Alternativno rešenje: Let se tretira kao poseban entitet pa uvodimo ternarnu vezu.



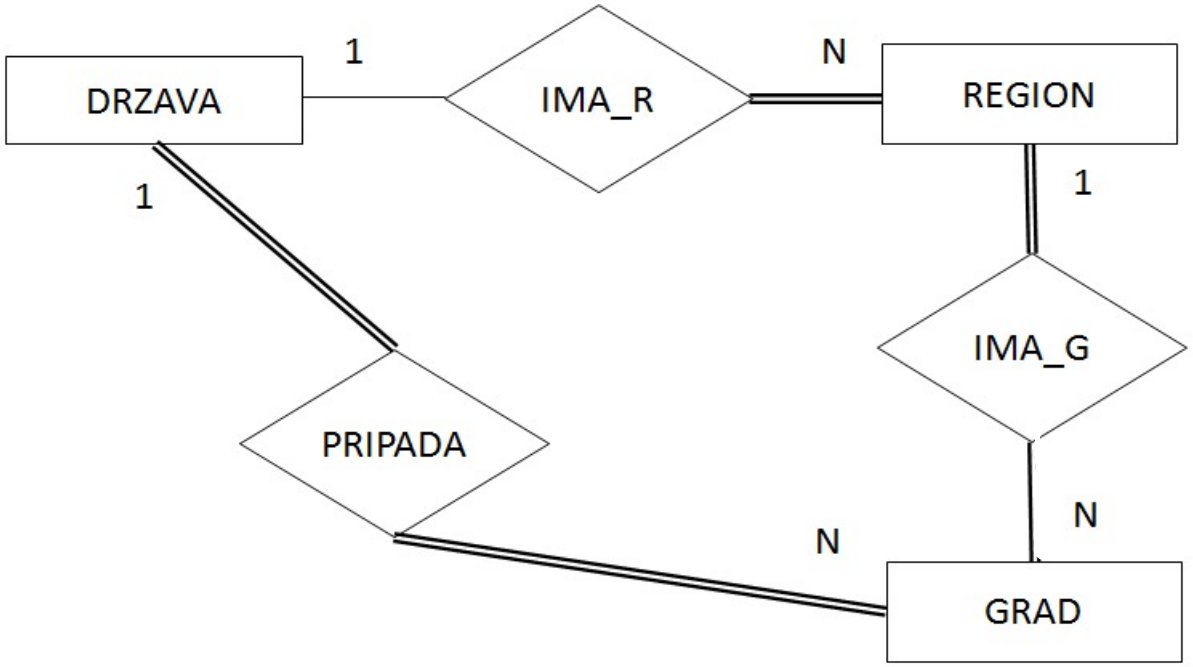
ER dijagrami - Veze

- Država ima više regiona, svaki region ima više gradova.



ER dijagrami - Veze

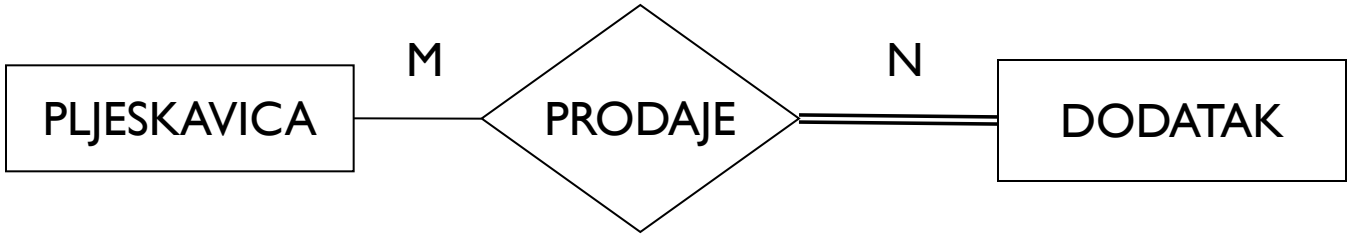
- Alternativa: Država ne mora da ima regione. U tom slučaju država koja nema regione ne može da ima gradove. Zbog toga se uvodi još jedna veza koja definiše da grad pripada državi i grad više ne učestvuje totalno u vezi IMA_G.



ER dijagram - Veze

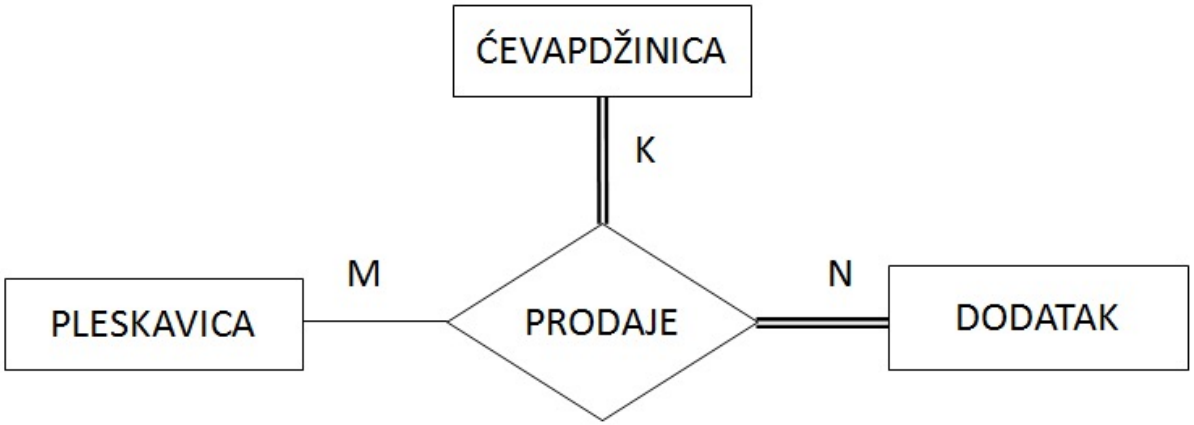
- Ćevapdžinica prodaje nekoliko vrsti pljeskavica, i isti dodaci (salate) se mogu koristiti uz različite tipove pljeskavica.

- Za slučaj da poostoji samo jedna ćevapdžinica.



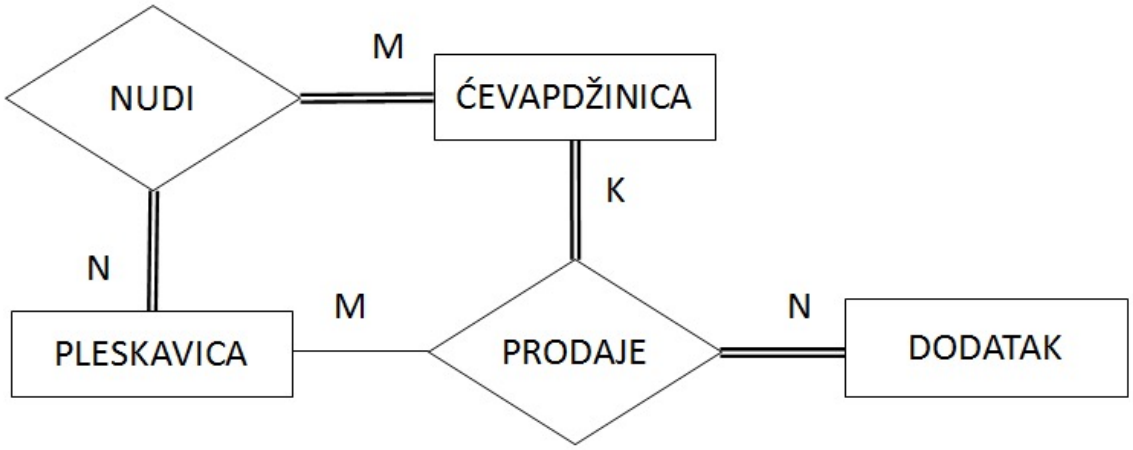
ER dijagram - Veze

- Za slučaj da imamo lanac ćevapdžinica:



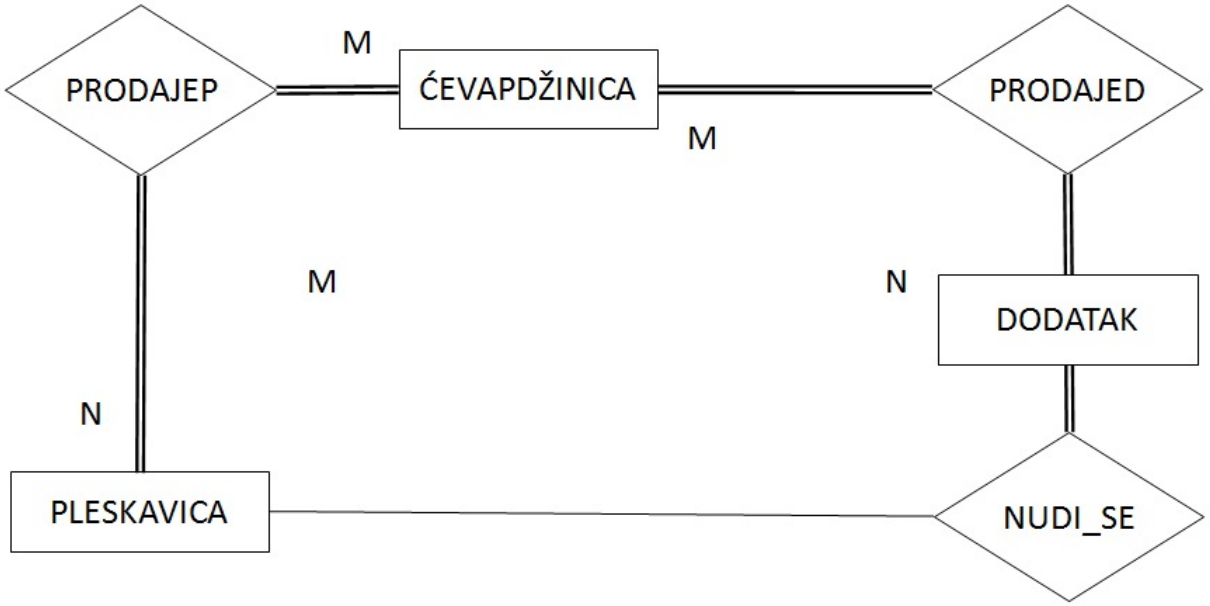
ER diijagram -Veze

- Kod rešenja sa ternarnom vezom ne postoji mogućnost da se neka pljeskavice prodaje bez priloga. Ta informacija bi zahtevala dodatnu binarnu vezu.



ER dijagram - Veze

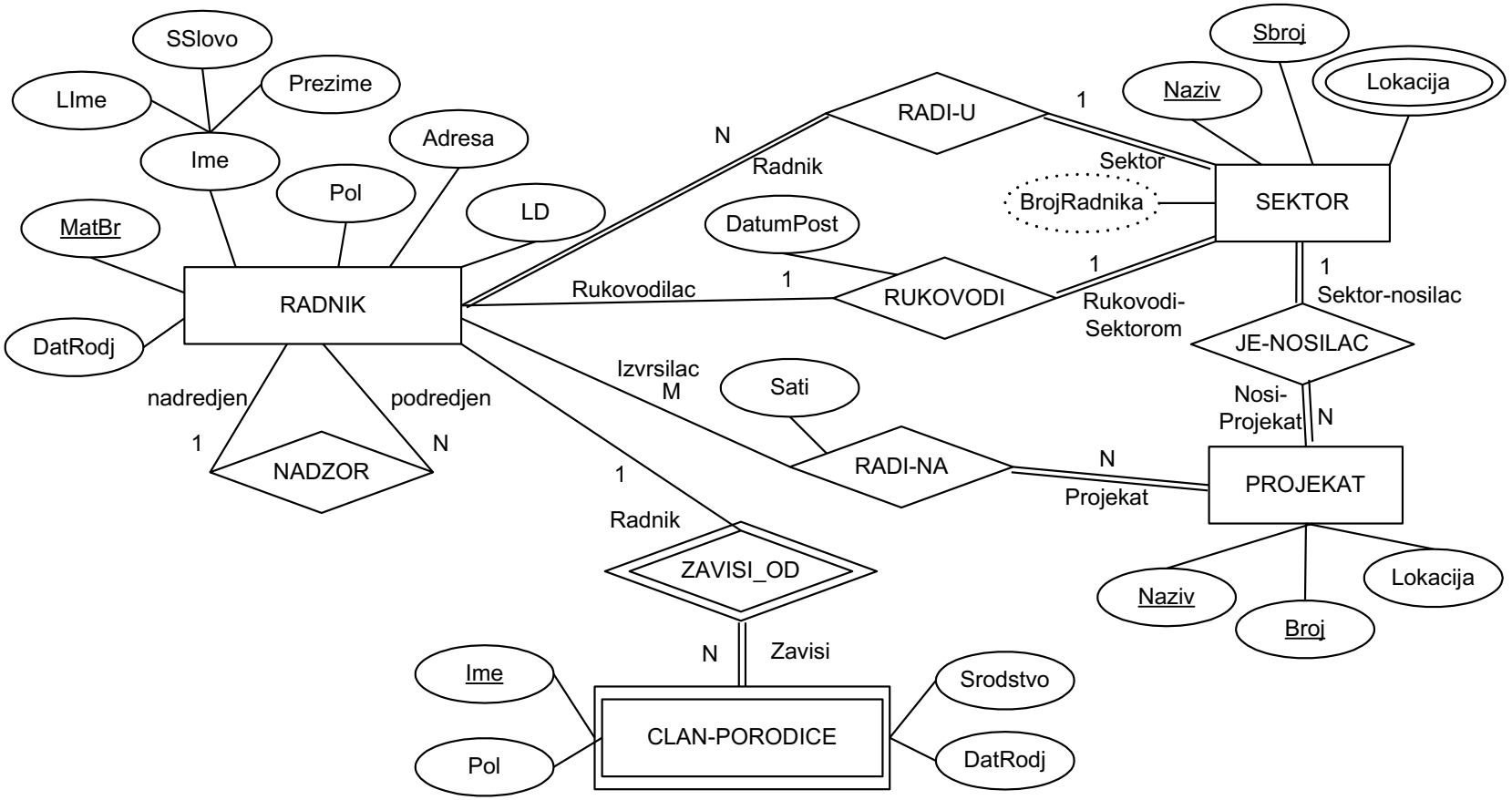
- Alternativno rešenje je da se ternarna veza zameni sa tri binarne. U tom slučaju se gubi semantika. Znamo koji dodaci se prodaju sa kojim pljeskavicama, znamo koje pljeskavice se prodaju u kojim ćevapdžinicama i znamo koji dodaci se prodaju u kojim ćevapdžinicama. Gubimo informaciju da se u nekoj ćevapdžinici prodaje baš određena pljeskavica sa određenim prilogom.



Baza podataka - PREDUZEĆE

- Preduzeće ima više sektora. Svaki sektor ima ime, broj i rukovodioca. Sektor ima bat četiri radnika. Vodi se evidencija o datumu kada je rukovodilac postavljen na tu funklciju. Sektor može imati više lokacija.
- U sektoru se radi na više projekata. Svaki projekat ima ime, broj i jedinstvenu lokaciju.
- Za svakog radnika se pamti ime, matični broj, adresa, plata, pol i datum rođenja. Svaki radnik radi u jednom sektoru a može biti angažovan na više projekata koje ne vodi isti sektor. Pri tome se vodi evidencija o broju radnih časova koje radnik povede na nekom projektu. Za svakog radnika se vodi evidencija o njegovom neposrednom rukovodiocu.
- Vodi se evidencija i o članovima porodica radnika. Za svakog člana se evidentira ime, pol, datum rođenja i srodstvo.

Baza podataka PREDUZEĆE

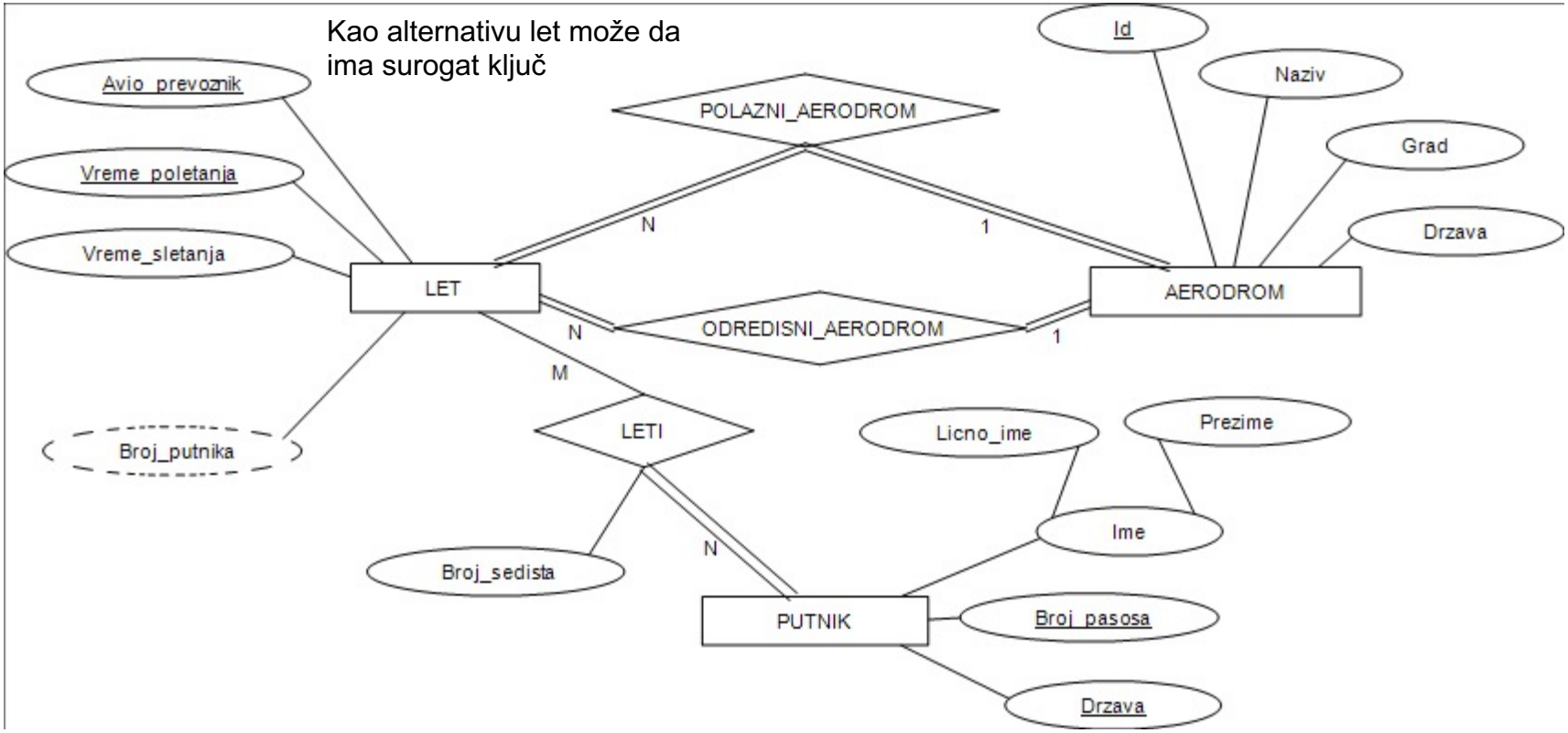


Primeri

Potrebno je čuvati podatke o avio letovima iz Niša. Za svaki let se čuvaju informacije o avio prevozniku, broju putnika, vremenu poletanja i sletanja, kao i polaznom i odredišnom aerodromu. Za aerodrome se čuvaju naziv, jedinstveni ID, grad, država. Za putnika se čuvaju imena, brojevi pasoša i broj sedišta na letu.

Na osnovu navedenih zahteva projektovati bazu podataka AERODROM_NIŠ i nacrtati odgovarajući ER dijagram. Za svaki tip entiteta odrediti kandidate za ključeve.

Primeri



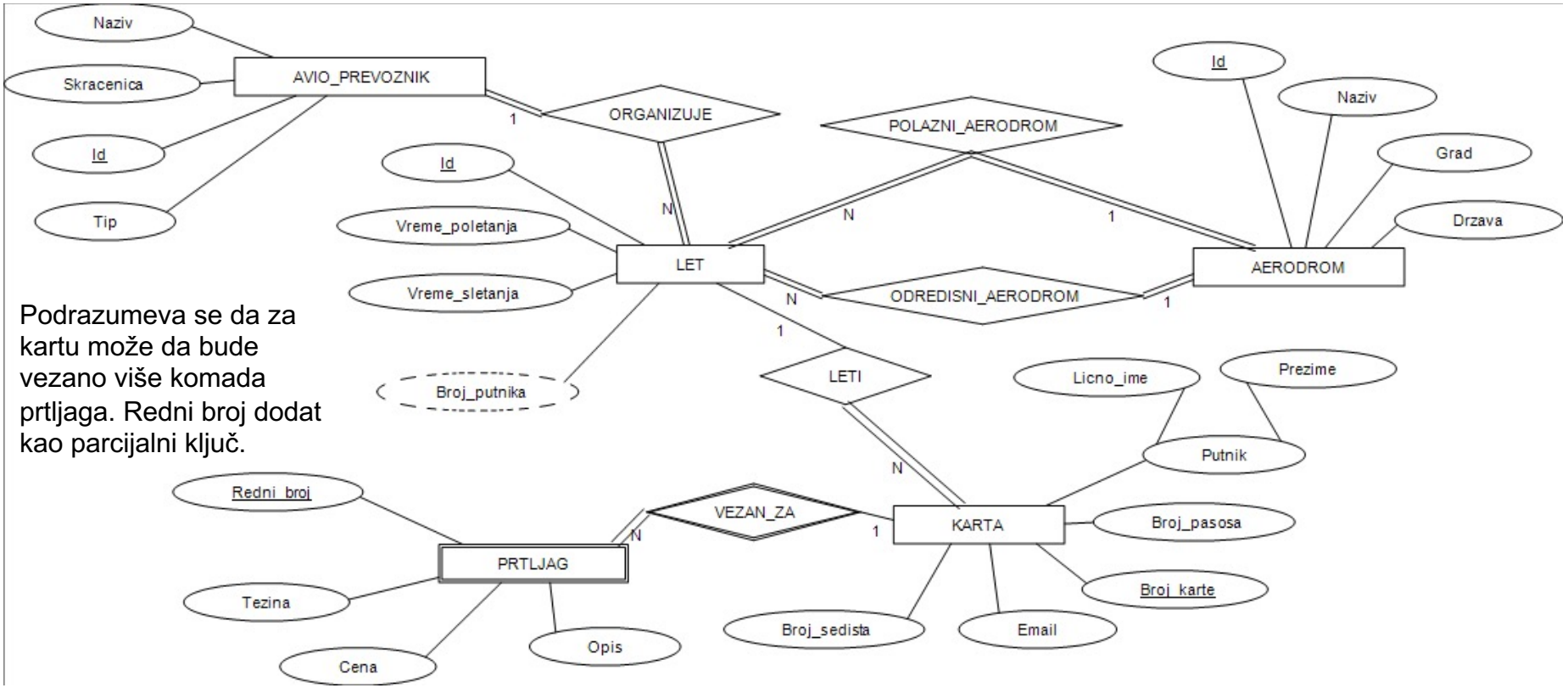
Dodata je država da bi imali garanciju da je broj pasoša jedinstven

Primeri

Potrebno je čuvati podatke o avio letovima sa Niškog aerodroma za low cost kompanije. Za svaki let se čuvaju informacije o avio prevozniku, broju putnika, vremenu poletanja i sletanja, kao i polaznom i odredišnom aerodromu. Za aerodrome se čuvaju naziv, jedinstveni ID, grad, država. Za avio prevoznike se čuva naziv, skraćenica, jedinstveni identifikator, tip. Avio prevoznici organizuju veći broj letova sa Niškog aerodroma, kao i ka Niškom aerodromu. Čuvaju se i informacije o prodatim kartama – za svaku kartu se čuvaju ime i prezime putnika, broj pasoša, kontakt email adrese, jedinstveni broj karte, let na koji se karta odnosi, broj sedišta. Karta se odnosi samo na jedan let i ne mogu da se izdaju povratne karte. Ako putnik ima prtljag, za njega se čuva informacija čiji je (vezuje se za kartu), težina, cena, opis.

Na osnovu navedenih zahteva projektovati bazu podataka AERODROM_NIŠ i nacrtati odgovarajući ER dijagram. Za svaki tip entiteta odrediti kandidate za ključeve.

Primeri



Podrazumeva se da za kartu može da bude vezano više komada prtljaga. Redni broj dodat kao parcijalni ključ.

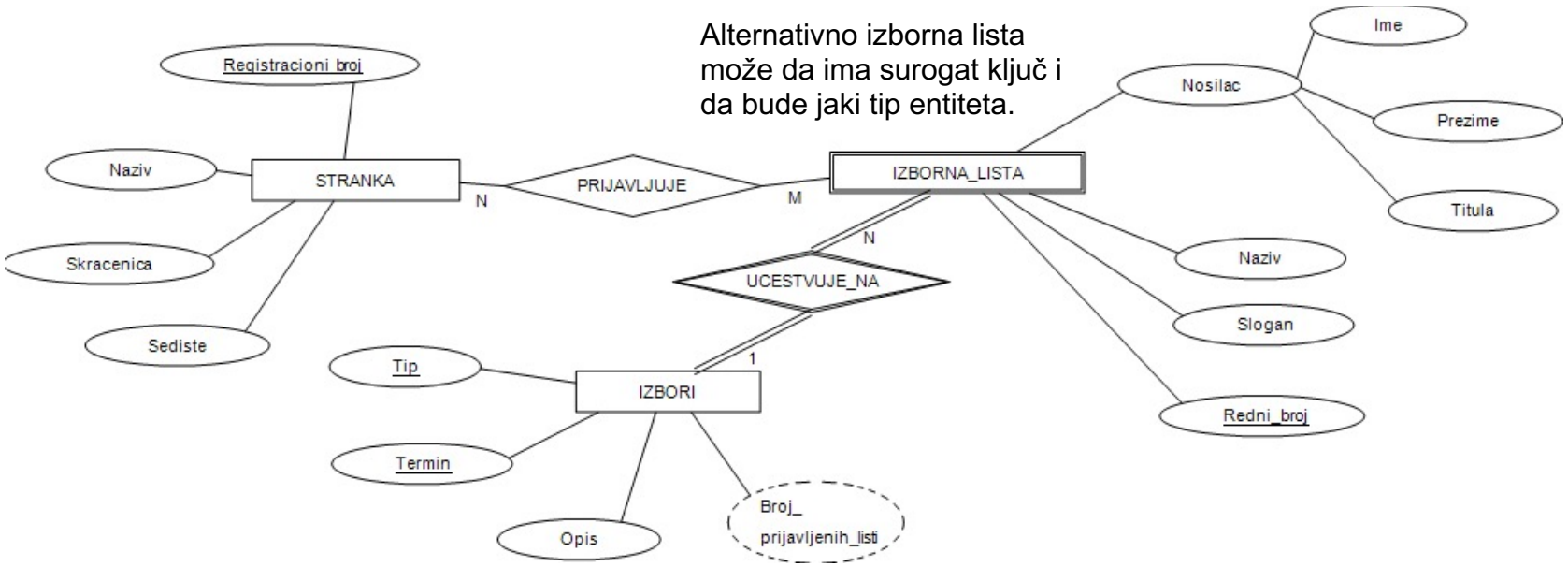
Alternativno prtljag može da bude viševrednosni složeni atribut.

Primeri

Potrebno je čuvati podatke o strankama koje učestvuju na izborima. Vodi se evidencija samo o registrovanim strankama koje su prijavljene za izbore, i za koje se čuvaju podaci o registracionom broju iz evidencije, punom nazivu, skraćenici, sedištu stranke. Ne mogu da postoje stranke sa istim nazivima i skraćenicama. Potrebno je čuvati i podatke o izbornim listama. Za svaku izbornu listu se čuvaju podaci o nazivu, sloganu, imenu nosioca liste (ime, prezime, titula), rednom broju pod kojim je evidentirana i strankama koje su je prijavile. Izbornu listu prijavljuje jedna ili više stranaka. Stranka može da učestvuje samo na jednoj izbornoj listi na određenim izborima. Izbornu listu može da prijavi i grupa građana, pri čemu su svi podaci koji treba da se čuvaju isti, osim podataka o strankama koje su je prijavile. Izborna lista se prijavljuje za određene izbore, za koje se pamti tip (čije vrednosti mogu da budu napr „lokalni“, „pokrajinski“ ili „republički“), termin izbora, kratak opis sa dodatnim informacijama, broj prijavljenih izbornih listi. Izborna lista mora da se prijavi samo na jedan tip izbora. Za svaki tip izbora mora da postoji najmanje jedna prijavljena lista da bi se čuvali podaci o njima.

Na osnovu navedenih zahteva projektovati bazu podataka IZBORI i nacrtati odgovarajući ER dijagram. Za svaki tip entiteta odrediti kandidate za ključeve.

Primeri

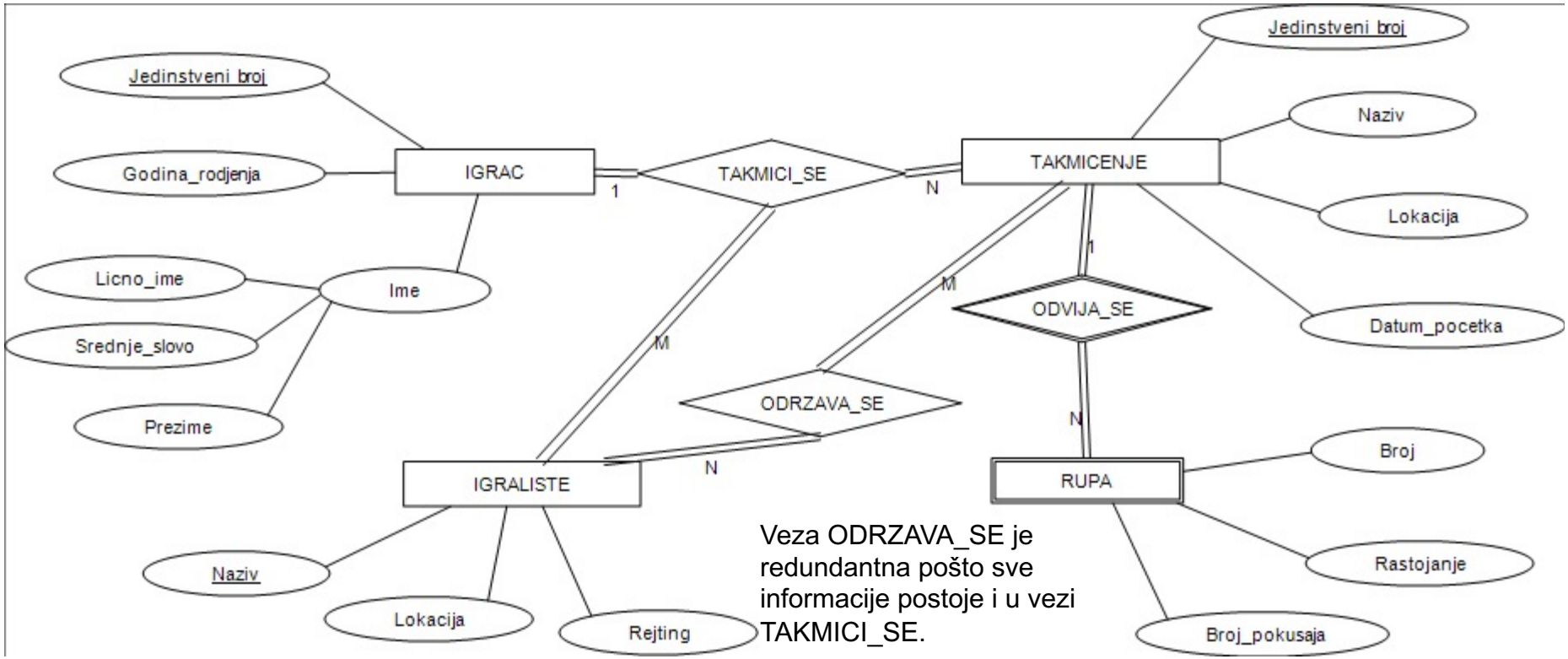

Alternativno izborna lista može da ima surogat ključ i da bude jaki tip entiteta.
Registracioni broj
Naziv
STRANKA
Skracenica
Sediste
N
PRIJAVLJUJE
M
IZBORNA_LISTA
Ime
Nosilac
Prezime
Titula
N
UCESTVUJE_NA
Naziv
Slogan
1
Tip
IZBORI
Redni_broj
Termin
Opis
Broj_
prijavljenih_listi

Primeri

Na osnovu navedenih zahteva projektovati bazu podataka za vođenje statistike za takmičenja u golfu i nacrtati odgovarajući ER dijagram. Za svaki tip entiteta odrediti kandidate za ključeve.

Treba pamtiti informacije o igračima (ime (ime, srednje slovo, prezime), godina rođenja, jedinstveni broj igrača) i takmičenjima (jedinstveni broj, lokacija, naziv, datum početka). Svako takmičenje se održava na više igrališta, a za svako igralište treba pamtiti naziv, lokaciju i rejting. Neki igrač u okviru nekog takmičenja može da igra na više igrališta, a na jednom igralištu igra samo po jedan igrač. Takmičenje u golfu se odvija na više rupa, a za svaku rupu pamti se broj, rastojanje od početne pozicije, broj pokušaja za ubacivanje loptice posle kog se broje negativni poeni. Za igrače treba pamtiti poene za svaki rupu.

Primeri



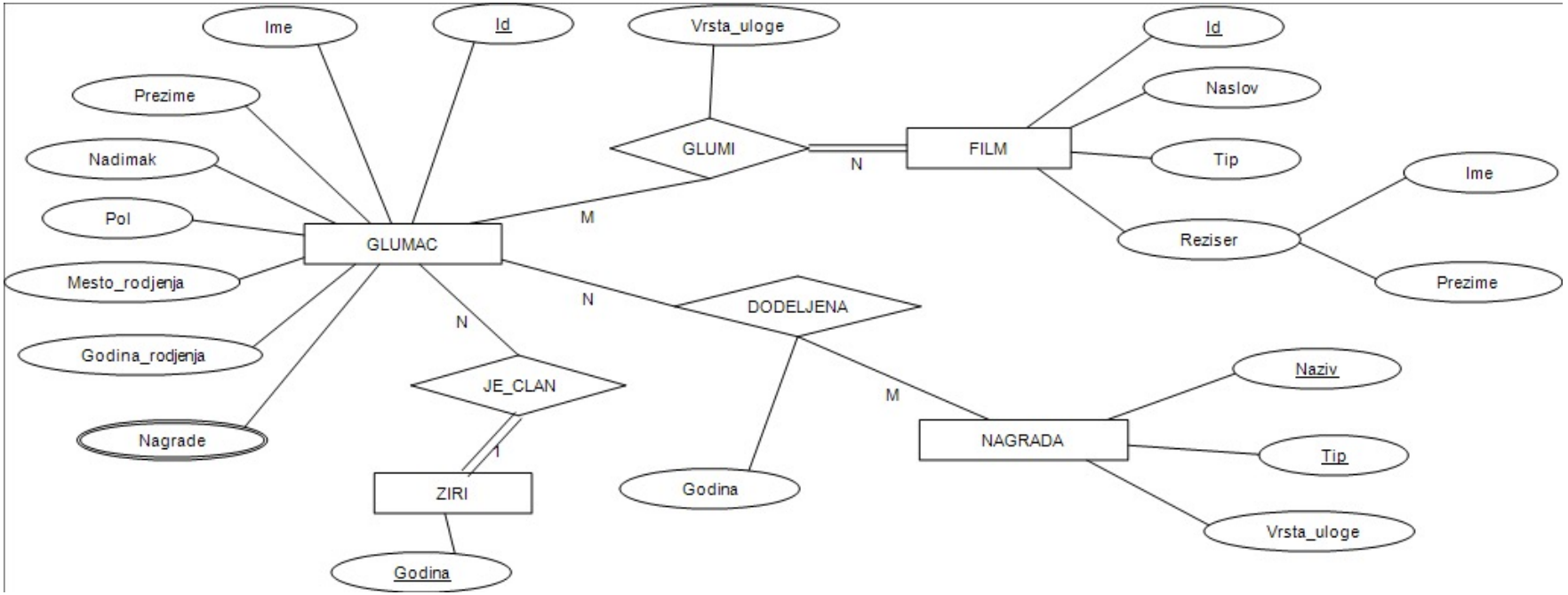
Veza ODRZAVA_SE je redundantna pošto sve informacije postoje i u vezi TAKMICI_SE.

Primeri

Na osnovu navedenih zahteva projektovati bazu podataka za Festival glumačkih ostvarenja i nacrtati odgovarajući ER dijagram. Za svaki tip entiteta odrediti kandidate za ključeve.

Potrebno je čuvati podatke o filmovima koji se prikazuju na festivalu. Za svaki film čuvaju se podaci o jedinstvenom identifikatoru koji mu je dodeljen na Festivalu, naslovu, tipu filma, režiseru (ime i prezime) i glumcima. Za glumce se čuvaju podaci: ime i prezime, nadimak, pol, mesto i godina rođenja, nagrade koje je osvajao. Glumac može da igra u više filmova koji se prikazuju na festivalu, pri tome se za svaku od tih uloga pamti vrsta uloge (glavna, sporedna, statista, kaskader). Na festivalu se dele nagrade glumcima. Za nagradu se čuvaju podaci o nazivu i tipu nagrade (muška, ženska), vrsti uloge za koju se dodeljuje, kao i podaci o prethodnim dobitnicima. Nagrade su pojedinačne i svaka od njih se dodeljuju svake godine samo jednom glumcu ili može i da se ne dodeli ako žiri tako proceni. Baza podataka treba da čuva i informacije o članovima žirija za svaku godinu festivala. Članovi žirija su iz reda glumaca i samo renomirani glumci mogu da budu izabrani.

Primeri

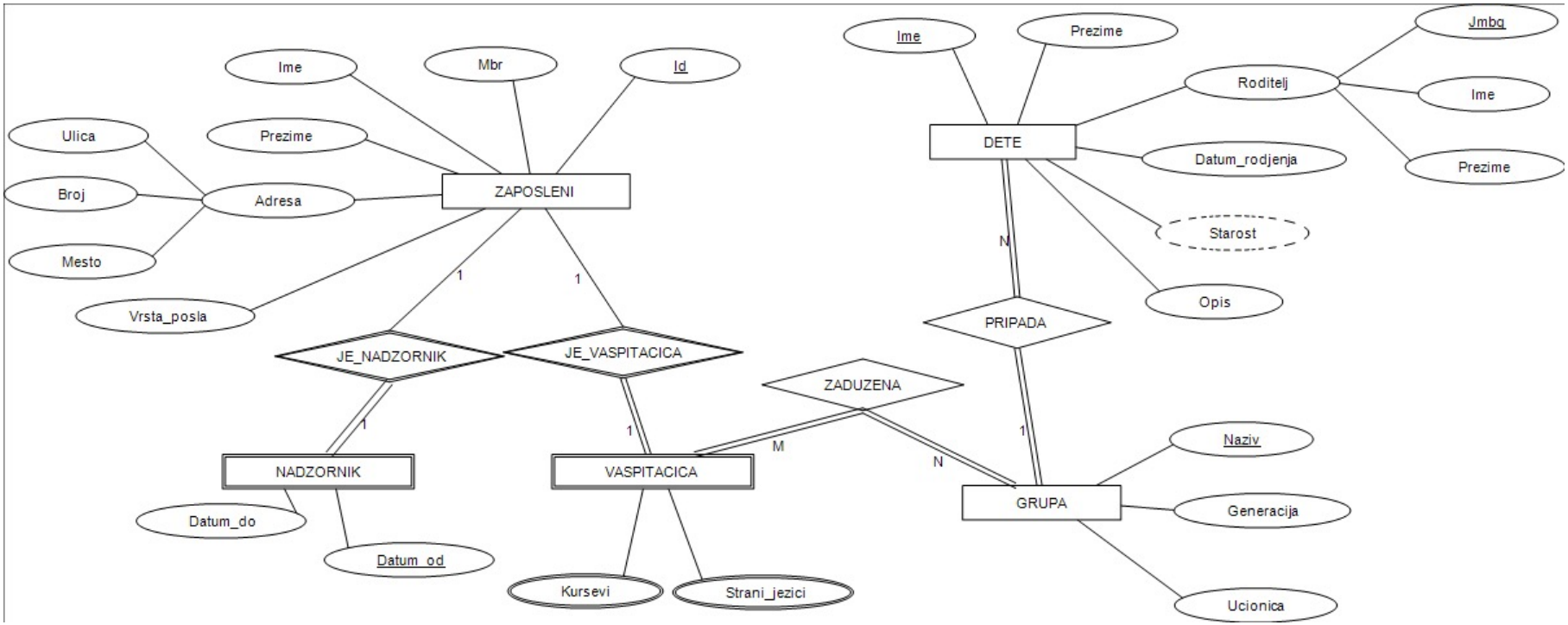

Ime
Id
Vrsta_uloge
Id
Prezime
Naslov
Nadimak
GLUMI
FILM
Tip
Pol
Ime
GLUMAC
Reziser
Mesto_rodjenja
Prezime
N
M
N
DODELJENA
Godina_rodjenja
N
JE_CLAN
M
Naziv
Nagrade
NAGRADA
1
Tip
ZIRI
Godina
Vrsta_uloge
Godina

Primeri

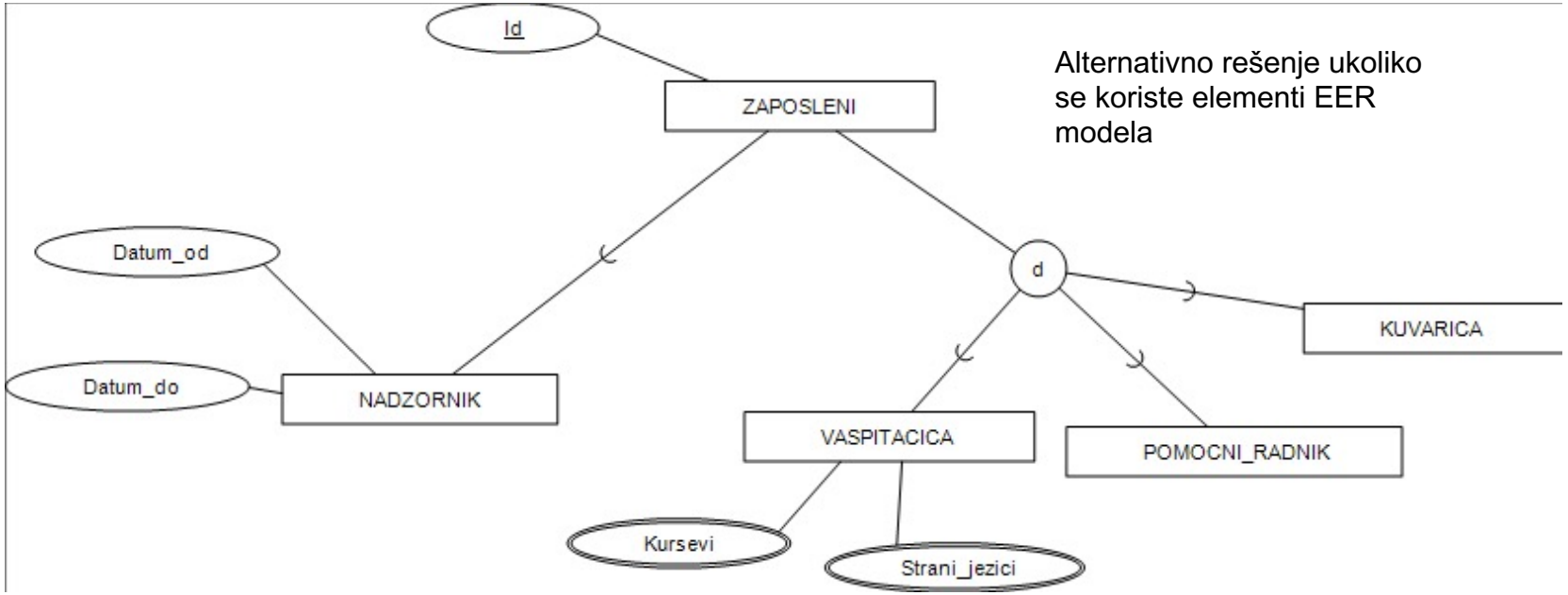
Na osnovu navedenih zahteva projektovati bazu podataka Vrtić i nacrtati odgovarajući ER dijagram. Za svaki tip entiteta odrediti kandidate za ključeve.

Potrebno je čuvati podatke o zaposlenima u vrtiću. Za svakog zaposlenog čuvaju se podaci o jedinstvenom identifikatoru, matičnom broju, imenu i prezimenu, i adresi stanovanja (ulica, broj, mesto). Zaposleni mogu da budu vaspitači, pomoćni radnici i kuvarice. Jedan od radnika u vrtiću je nadzornik, koji se postavlja za određeni period. Za pomoćne radnike i kuvarice se ne čuva ništa dodatno od podataka. Za vaspitače se čuvaju dodatni podaci o završenim kursevima za usavršavanje, poznavanju stranog jezika. Vaspitači su zaduženi za određene grupe u vrtiću. Svaka grupa se odnosi na jednu generaciju dece, ima svoju učionicu, jedinstveni naziv i dve zadužene vaspitačice. U grupi ima do 30-oro dece, a za svako dete se čuvaju podaci o roditelju (ime, prezime, JMBG), ime i prezime deteta, datum rođenja, godine starosti, kratak opis deteta.

Primeri


Ime
Mbr
Id
Ulica
Prezime
Broj
Adresa
ZAPOSLENI
Mesto
Vrsta_posla
1
1
JE_NADZORNIK
JE_VASPITACICA
1
1
NADZORNIK
VASPITACICA
Datum_do
Datum_od
Kursevi
Strani_jezici
Ime
Prezime
Jmbg
Roditelj
Ime
Prezime
DETE
Datum_rodjenja
Starost
Opis
N
PRIPADA
1
ZADUZENA
M
N
Naziv
GRUPA
Generacija
Ucionica

Primeri


Id
ZAPOSLENI
Alternativno rešenje ukoliko se koriste elementi EER modela
Datum_od
d
KUVARICA
Datum_do
NADZORNIK
VASPITACICA
POMOCNI_RADNIK
Kursevi
Strani_jezici

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Relacioni model podataka

Prof.dr Leonid Stoimenov

Pregled predavanja

- Istorijat relacionog modela podataka
- Strukturna komponenta
- Integritetna komponenta
- SQL - podjezik za definisanje podataka

Istorijat relacionog modela podataka

- 1970 E. Codd je predložio relacioni model

[E. Codd, "A Relational Model for Large Shared Data Banks", CACM, vol.13, No.6, June 1970]

- 1980 prve komercijalne implementacije
  - ORACLE RDBMS
  - SQL/DS system na IBM-ovom operativnom sistemu MVS
- Primeri komercijalnih RDBMS-ova
  - IBM-ova DB2 familija
  - Informix
  - Oracle
  - Sybase
  - Microsoft-ov Access i SQL Server
  - Foxbase
  - Paradox

Neke karakteristike relacionog modela podataka

- Jednostavan i elegantan
  - Baza podataka je **kolekcija više relacija**
  - Svaka relacija je **tabela** sa vrstama i kolonama
- Tabelarna reprezentacija
  - omogućava da se lako razume sadržaj baze podataka, i
  - da se za manipulaciju podacima koristi jednostavan jezik visokog nivoa SQL
- SQL obuhvata DDL, DML, SDL

Koncepti relacionog modela podataka

- Koncepti na nivou intenzije
  - Atribut
  - Šema relacije
  - Šema baze podataka
- Koncepti na nivou ekstenzije:
  - Domen atributa
  - Torka
  - Pojava relacije
  - Pojava baze podataka

- Svi ostali koncepti se izvode iz osnovnih primenom određenih formalnih – matematičkih pravila

Relacioni model – neformalni pogled

- U relacionom modelu podataka baza podataka se predstavlja kao kolekcija **relacija**
- **Relacija** je tabela vrednosti, koja sadrži **vrste** i **kolone**
- U terminologiji relacionog modela podataka:
  - **vrste** se nazivaju **torke**,
  - **kolone** su **atributi**,
  - **tabela** se naziva **relacija**

Ime	Prezime	Indeks	MBR
Petar	Petrović	1111	123456
Milan	Milanović	2222	654321
Jovan	Jovanović	3333	345612

Relacioni model – neformalni pogled

- Svaka **vrsta** tabele predstavlja kolekciju vrednosti podataka koje su u nekom odnosu
  - ER model: vrsta tabele odgovara nekoj pojavi entiteta iz realnog sveta ili nekoj pojavi poveznika
- Svaka **kolona** tabele predstavlja kolekciju vrednosti podataka koja opisuje neku odabranu osobinu realnog sistema
  - ER model: kolona tabele odgovara nekom atributu tipa entiteta ili atributu tipa poveznika ili tipu poveznika
- Sve vrednosti u jednoj koloni pripadaju istom tipu podataka

Definicija relacije

- Neka je

$$A=\{A_1, A_2, ..., A_n\}$$

skup atributa realnog sistema,

i neka je svakom atributu A_i pridružen domen D_i

  - Domeni ne moraju biti različiti

- **Definicija relacije:**

  Relacija R na skupovima $D_1, D_2, ..., D_n$ je skup **n-torki (torki)** od kojih svaka sadrži prvi element iz D_1, drugi iz D_2, ... , n-ti iz D_n

- **Domen** D se odnosi na skup **atomičnih** vrednosti
  - Domeni se specificiraju preko **tipa podataka**
  - Korisno je da svaki domen ima **naziv**

Domen (podsetnik)

- Logička definicija domena:
  - Lokalni_telefonski_broj: skup 6-cifrenih telefonskih brojeva koji su validni na području lokalne mrežne grupe
  - Ocena_studenta: celi broj između 6 i 10
  - Kod_obrazovnog_profila: skup oznaka profila Elektronskog fakulteta: E,EEN,EMK,MIM,RI,T,US

- Tip podataka ili format domena:
  - Lokalni_telefonski_broj: CHARACTER string oblika ccc-ccc
  - Prelazna_ocena_studenta: INTEGER broj iz opsega 6-10
  - Kod_obrazovnog_profila: CHARACTER string iz skupa {E,EEN,EMK,MIM,RI,T,US}

Šema relacije

- **Šema relacije** je imenovana dvojka, u oznaci **N(R,C)**, gde je:
  - N naziv šeme relacije,
  - R⊆U skup atributa relacije, a
  - C skup ograničenja integriteta relacije

- **N** je neformalna komponenta definicije koja opisuje njenu semantiku u prirodnom jeziku (*naziv šeme relacije*)
- Skup ograničenja integriteta **C** opisuje odnose između elemenata domena atributa iz **R**
  - Ograničava koje će se torke pojaviti u instanci ove relacije

Primer šeme relacije

- Posmatraćemo tip entiteta STUDENT sa atributima {IND, IME, PREZIME, BPI}
- Pri čemu je uočeno da važe ograničenja

  (γ1) svaki student ima broj indeksa i ne postoje dva studenta sa istim brojem indeksa

  (γ2) broj položenih ispita (BPI) je veći od nule i manji od 50

- **Šema relacije** koja predstavlja model ove klase entiteta bi imala oblik:

STUDENT({IND,IME,PREZIME,BPI}, {γ1 , γ2})

Obeležavanje šeme relacije

- Uobičajeno je da se šema relacije obeležava sa

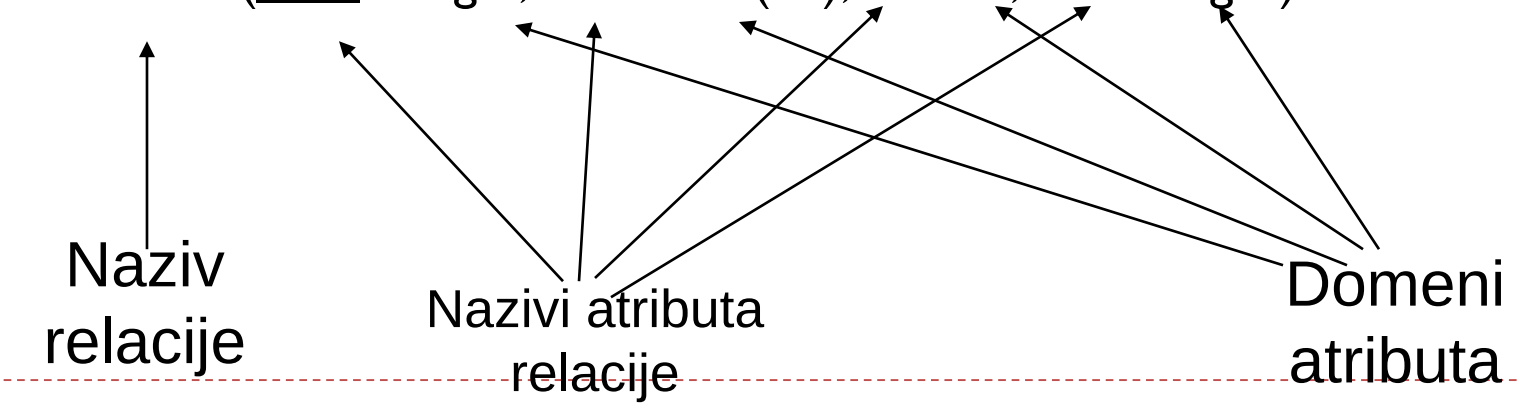
  $\mathbf{R(\underline{A_1}, A_2, \ldots, A_n)}$

  $\mathbf{R(\underline{A_1}:D_1, A_2:D_2, \ldots, A_n:D_n)}$

  - R naziv (ime) relacije
  - $A_1, A_2, \ldots, A_n$ lista atributa relacije
  - $D_1, D_2, \ldots, D_n$ lista domena atributa relacije
  - $\underline{A_1}$ ograničenje tipa "primarni ključ relacije"
- **Primer :**

RADNIK2 (MBR, LIME, LD, SBR)

RADNIK1 (MBR:integer, LIME:char(15),LD:real, SBR:integer)



Pojava (instanca) relacije

- Pojava relacije **r** nad šemom relacije **(R,C)**, u oznaci **r(R)**, je skup n-torki **t = {t_1,t_2,...,t_k}** koje zadovoljavaju ograničenja C
  - Svaka n-torka predstavlja uređenu listu od n-vrednosti, t=<v_1,v_2,...v_n>, gde $v_i \in D_i$ ili v_i=null
  - i-ta vrednost torke t, koja odgovara atributu A_i, se obeležava sa t[A_i]

- **Primer:** Jedna pojava relacije radnik nad šemom relacije RADNIK (MBR, LIME, LD, SBR) je:
  <2203,MIRA,50000,10>
  <2817,PERA,40000,20>
  <2932,MIRA,35000,10>
  <2995,GOCA,18000,30>

Tabelarno predstavljanje relacije

- Relacija **r** nad šemom relacije **$R(A_1, A_2, ..., A_n)$** se predstavlja **tabelom** čije kolone odgovaraju atributima, a vrste pojedinim n-torkama vrednosti ovih atributa
  - Sve vrste moraju biti različite

- **Primer**:Relacija radnik definisana nad šemom relacije RADNIK(MBR, LIME, LD,SBR) se predstavlja tabelom

radnik

MBR	LIME	LD	SBR
2203	MIRA	50000	10
2817	PERA	40000	20
2932	MIRA	35000	10
2995	GOCA	18000	30

Svojstva relacije

- Šema relacije ne sadrži dva jednaka naziva atributa
- Redosled naziva atributa u relaciji nije bitan
- Relacija ne sadrži dve jednake torke
- Redosled torki u relaciji nije bitan

- **Razlog:**
  - Prva dva svojstva proizilaze iz definicije šeme relacije (skup naziva atributa, po definiciji skup ne može sadržati dva međusobno jednaka elementa)
  - Druga dva svojstva proizilaze iz definicije relacije (skup torki). Posledica promene redosleda atributa i torki u relaciji neće biti gubitak informacije sadržane u relaciji.

Šema i pojava relacione baze podatka

- Šema relacione baze podataka predstavlja konačan skup šema relacija baze podataka R_i (i=1, 2,..., m) i skup međurelacionih ograničenja IC
  - Označava se sa **$S(R_1,R_2,...,R_m;IC)$**

- Pojava (instanca) baze podataka nad šemom $S(R_1,R_2,...,R_m;IC)$, je skup pojava r_i šema relacija R_i (i=1, 2,..., m), pri čemu svako r_i zadovoljava ograničenja integriteta IC
  - Označava se sa **s**

Primer: Šema relacione baze podataka PREDUZEĆE

PREDUZEĆE ({RADNIK, SEKTOR,PROJEKAT};IC)
RADNIK(MBR, LIME, LD, ***SBR***)
SEKTOR(SBR, SNAZIV, ***SLOK***)
PROJEKAT(PBR, PNAZIV, ***SBR, PRUK***)

- Svaka šema relacije sadrži specifikaciju primarnog ključa (podvučeni atribut u šemi relacije) i specifikaciju stranih ključeva (***italic*** u šemi relacije)

Pojava baze podataka PREDUZEĆE

radnik

MBR	LIME	LD	SBR
2203	MIRA	50 000	10
2817	PERA	40 000	20
2932	MIRA	35 000	10
2995	GOCA	18 000	30
3305	LAZA	60 000	40
3515	JOVAN	20 000	20
3819	VLADA	65 000	30

sektor

SBR	SNAZIV	SLOK
10	PROIZVODNJA	NIŠ
20	PROIZVODNJA	BEOGRAD
30	INŽINJERING	NIŠ
40	RAZVOJ	NIŠ

projekat

PBR	PNAZIV	SBR	PRUK
100	PC	10	2203
200	HOST	20	3305
300	LAN	30	3819
400	VIPX	40	2817

Ključ relacije (1)

Definicija ključa:

Skup atributa $X \subseteq R$ predstavlja ključ šeme relacije (R,C), ako za svako r važe sledeća dva uslova:

U1: $(\forall u,v \in r)(u[X]=v[X] \Rightarrow u=v)$

U2: $(\forall Y \subset X)(\neg$ **U1**$)$

U1: definiše jedinstvenost vrednosti kjuča

U2: definiše minimalnost skupa atributa ključa

- Jedna šema relacije može imati više ključeva
  - To su ekvivalentni ključevi ili ključevi kandidati
  - Jedan od ekvivalentnih ključeva se bira za primarni ključ

Ključ relacije – dodatni pojmovi

- Super ključ (važi uslov U1)
  - Atribut ili skup atributa koji na jedinstven način identifikuje torku u relaciji

- Kandidat ključ (važe uslovi U1 i U2)
  - Super ključ čiji nijedan pravi podskup nema svojstvo super ključa u toj relaciji

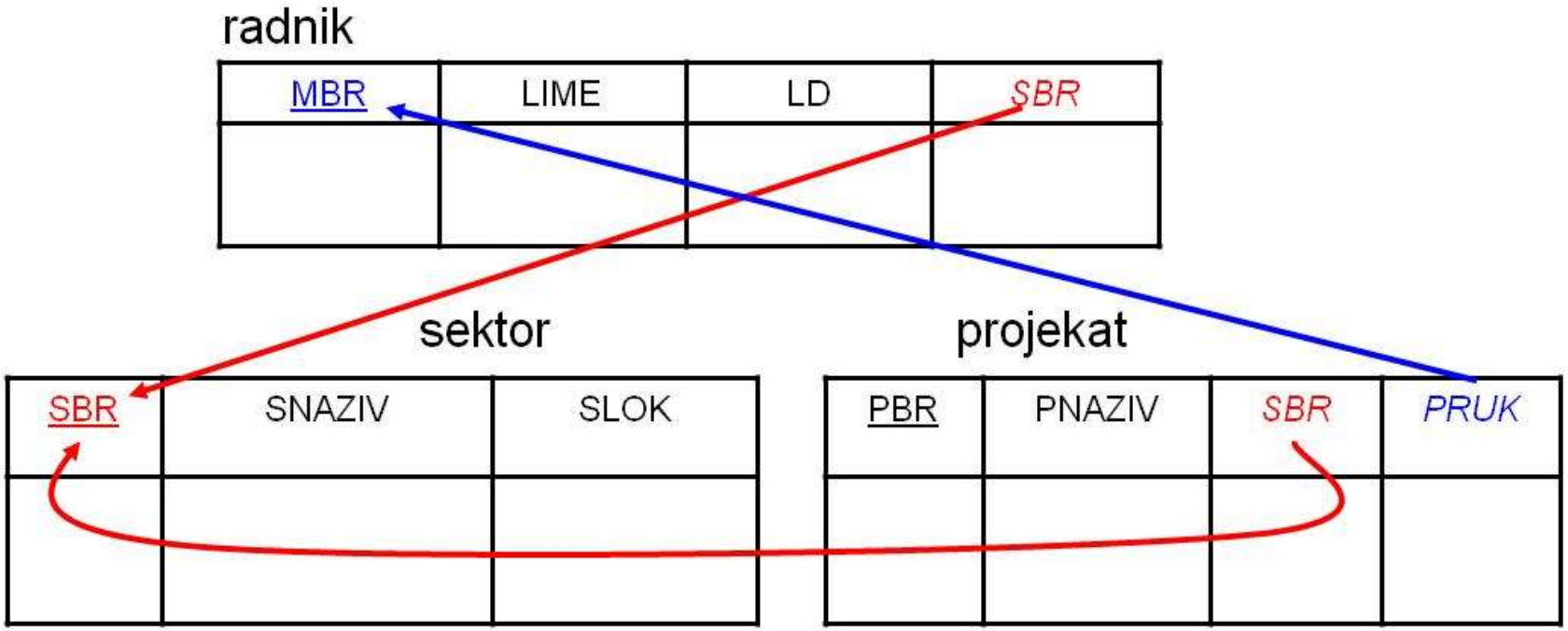
Ključ relacije – dodatni pojmovi

- Primarni ključ (važe uslovi U1 i U2)
  - Kandidat ključ izabran od strane projektanta baze podataka da na jedinstven način identifikuje torke relacije
  - Preko ovog ključa se najčešće vrši traženje u relaciji
  - On se koristi kao strani ključ u drugim šemama relacija
- Strani ključ
  - Atribut ili skup atributa jedne relacije koji se uparuje sa ključem kandidatom druge ili iste relacije

Strani (spoljnji) ključ relacije

- Skup atributa SK relacije r je **strani ključ** relacije r ako zadovoljava sledeća dva uslova:
  1. Ako se relacija r referencira na relaciju q, tada atributi iz SK relacije r **moraju imati isti domen** kao atributi primarnog ključa PK relacije q
  2. Ako se torka t1 relacije r referencira na torku t2 relacije q, tada **vrednost stranog ključa** SK **u torki t1** relacije r mora biti **jednaka vrednosti primarnog ključa** PK **u torki t2** relacije q ili može imati nedefinisanu (NULL) vrednost

Strani (spoljnji) ključ relacije



Integritetna komponenta

- Koriste se pri formiranju i ažuriranju baze podataka u cilju održavanja sadržaja baze podataka u saglasnosti sa uočenim odnosima između atributa realnog sistema

  - **Ograničenja torki** – proveravaju se za svaku torku relacije
  - **Relaciona ograničenja** – proveravaju se međusobni odnosi više torki jedne relacije
    - Ovim se reguliše lokalna konzistentnost – usaglašenost pojave relacije sa definicijom šeme relacije
  - **Međurelaciona ograničenja**
    - Reguliše se globalna konzistentnost baze podataka - usaglašenost pojave baze podataka sa definicijom šeme baze podataka

Integritet ključeva

- Vrednosti ključeva kandidata moraju biti jedinstvene u pojavi šeme relacije
  - Ovim se definiše ograničenje jedinstvenosti
- **Razlog :**
  - Jedinstvenost vrednosti ključeva kandidata je posledica činjenice da relacija predstavlja skup torki, te stoga u relaciji ne mogu postojati dve iste torke

Integritet entiteta

- Niti vrednost primarnog ključa, niti bilo koje njegove komponente, ne sme imati nepoznatu (NULL) vrednost u pojavi relacije
- **Razlog:**
  - Pošto primarni ključ služi za identifikaciju pojedinih torki relacije, to nepoznata vrednost primarnog ključa bi onemogućila da se neke torke relacije identifikuju

Referencijalni integritet

- To je zahtev da referencirana torka mora postojati (kada semantika podataka to zahteva)
- Važan tip referencijalnog integriteta je **ograničenje stranog ključa**
  - Ako u relaciji postoji strani ključ, tada vrednost stranog ključa mora biti jednaka vrednosti ključa kandidata neke torke referencirane relacije ili imati null vrednost
- To je ograničenje koje se definiše između dve relacije u cilju održavanja konzistentnosti među torkama ovih relacija
- **Razlog:**
  - Referencijalni integritet obezbeđuje da se svaka torka iz jedne relacije referencira samo na postojeću torku u drugoj ili istoj relaciji

„Semantički“ integritet - Opšta ograničenja

- Dodatna pravila koja specificira korisnik ili DBA, a koja definišu ili ograničavaju neke aspekte realnog sistema

- Primer
  - Broj zaposlenih u svakom sektoru preduzeća ne sme biti veći od 20
  - Plata rukovodioca mora biti veća od plata njegovih radnika
  - Ako korisnik postavi takvo ograničenje, onda on očekuje da ga DBMS proverava i da neće dozvoliti da se u relaciju sektor unese 21-vi radnik, odnosno u relaciju radnik LD radnika koji je veći od LD-a njegovog rukovodioca

Specifikacija ograničenja

- U relacijama baze podataka i između relacija baze podataka postoji veliki broj ograničenja integriteta
- Ova ograničenja treba eksplicitno uneti u šemu baze podataka
- U šemi relacije se eksplicitno specificiraju
  - Domeni atributa
  - Primarni ključ
  - Strani ključevi i referenca na primarni ključ
  - Da li atribut može imati NULL vrednost
  - Da li atribut mora imati jedinstvenu vrednost
  - Da li atribut ima podrazumevanu vrednost
  - Ostala semantička ograničenja (koristi se mehanizam trigera)

Ograničenja integriteta i DBMS

- Ograničenja integriteta se specificiraju u šemi baze podataka i eksplicitno specificiraju korišćenjem DDL-a, tako da ih DBMS može automatski nadzirati
- Semantička ograničenja se mogu specificirati i kontrolisati unutar aplikacionih programa za ažuriranje baze podataka ili se mogu specificirati u šemi baze podataka korišćenjem jezika za specificiranje ograničenja
- Većina komercijalnih DBMS-a podržava integritet ključa i entiteta, dok manji broj podržava referencijalni i semantički integritet

SQL - definisanje podataka

Specificiranje tipa relacije

STUDENT (Ind:INTEGER, Ime:STRING,
Adresa:STRING,Status:STRING), **Ključ: {Ind}**

```
CREATE TABLE STUDENT (
   Ind          INTEGER,
   Ime          CHAR(20),
   Adresa       CHAR(50),
   Status       CHAR(10) )
```

SQL- definisanje podataka
Specificiranje ograničenja ključa (1)

STUDENT (Ind:INTEGER, Ime:STRING,
Adresa:STRING,Status:STRING), **Ključ: {Ind}**

```
CREATE TABLE STUDENT (
   Ind          INTEGER,
   Ime          CHAR(20),
   Adresa       CHAR(50),
   Status       CHAR(10),
   PRIMARY KEY (Ind ) )
```

SQL - definisanje podataka
Specificiranje ograničenja ključa (2)

PREDMET(KatId:STRING,
PrKod:STRING,PrNaziv:STRING, Opis:STRING),
Ključevi: {PrKod}, {KatId,PrNaziv}

```
CREATE TABLE PREDMET (
   KatId       CHAR(6),
   PrKod       CHAR(4),
   PrNaziv     CHAR(20),
   Opis        CHAR(100),
   PRIMARY KEY (PrKod),
   UNIQUE  (KatId,PrNaziv) )
```

SQL - definisanje podataka Specificiranje NULL i podrazumevanih vrednosti

STUDENT(Ind:INTEGER, Ime:STRING,Adresa:STRING,Status:STRING),
Ključ: {Ind}

```
CREATE TABLE STUDENT (
    Ind             INTEGER,
    Ime             CHAR(20) NOT NULL,
    Adresa          CHAR(50),
    Status CHAR(10) DEFAULT 'Brucoš',
    PRIMARY KEY (Ind ) )
```

ILI

```
CREATE TABLE STUDENT (
    Ind             INTEGER PRIMARY KEY,
    Ime             CHAR(20) NOT NULL,
    Adresa          CHAR(50),
    Status CHAR(10) DEFAULT 'Brucoš' )
```

SQL - definisanje podataka
Specificiranje opštihograničenja

ZAPISNIK(StudId:INTEGER, PrKod:STRING, Rok:STRING, Ocena:INTEGER), **Ključ: {StudId,PrKod,Rok}**

```
CREATE TABLE ZAPISNIK (
    StudId      INTEGER,
    PrKod       CHAR(6),
    Rok         CHAR(6),
    Ocena       INTEGER NOT NULL,
    PRIMARY KEY (StudId,PrKod,Rok),
    CHECK (Ocena IN ( 5,6,7,8,9,10) AND VALUE IS NOT NULL),
    CHECK (StudId >0 AND StudId <7000) )
```

SQL - definisanje podataka
Kreiranje korisnički definisanog domena

```
CREATE DOMAIN OCENA INTEGER
   CHECK (VALUE IN (5,6,7,8,9,10) AND VALUE IS NOT NULL )
CREATE DOMAIN OCENA INTEGER
   CHECK (Ocena >4 AND Ocena <11 AND VALUE IS NOT NULL)

CREATE TABLE ZAPISNIK (
   StudId          INTEGER,
   PrKod           CHAR(6),
   Rok             CHAR(6),
   Ocena           OCENA,
   PRIMARY KEY (StudId,PrKod,Rok),
   CHECK           (StudId >0 AND StudId <7000) )
```

SQL - definisanje podataka

Specificiranje ograničenja stranog ključa

STUDENT(Ind, Ime, Adresa, Status)
PROFESOR(Id, Ime, KatId)
PREDMET(KatId, PrKod, PrNaziv, Opis)
ZAPISNIK(*StudId, PrKod*, Rok, Ocena)
IZBOR(*StudId, PrKod*, Semestar)
NASTAVA(ProfId, PrKod, Semestar)

```
CREATE TABLE ZAPISNIK (
    StudId          INTEGER,
    PrKod           CHAR(6),
    Rok             CHAR(6),
    Ocena           OCENA,
    PRIMARY KEY (StudId,PrKod,Rok),
    FOREIGN KEY (StudId) REFERENCES STUDENT(Ind),
    FOREIGN KEY (PrKod) REFERENCES PREDMET )
```

SQL - manipulacija podacima

Unos, brisanje i ažuriranje torki

```
INSERT
INTO STUDENT(Ind, Ime, Adresa,Status)
VALUES (11708, 'Ana', 'Nišavska 11 Pirot', 'apsolvent')
```

- **Mogu se izostaviti imena atributa (kolona) u klauzuli INTO,**
- **Vrednosti atributa moraju navesti u redosledu koji je definisan**
  - **Šemom relacije**
  - **Redosledom atributa**
- **Dobar stil je da se imena atributa eksplicitno navedu**

SQL - manipulacija podacima

Unos, **brisanje** i **ažuriranje** torki

```
DELETE
FROM STUDENT S
WHERE S.ime= 'Ana'
```

- **Iz tabele STUDENT briše torku koja u koloni Ime ima vrednost** Ana

```
UPDATE STUDENT S
SET S.Status= 'posldiplomac'
WHERE S.ind= 11708
```

- **U tabeli STUDENT ažurira torku koja u koloni Ind ima vrednost** 11708 **tako što postavlja novu vrednost** posldiplomac **u koloni Status**

Operacijska komponenta

- Nad relacijama je definisan skup operacija
  - **Operacije** omogućavaju **specificiranje upita** nad bazom podataka
  - **Rezultat** pretraživanja je **nova relacija** koja može biti formirana od jedne ili više relacija
- Postoje dva tipa operacija nad relacionim modelom podataka:
  - Operacije relacione algebre
  - Operacije relacionog računa
    - Nad torkama
    - Nad domenima

Relacioni model podataka

Pitanja ???

Baze podataka 2020/2021

Katedra za računarstvo
Elektronski fakultet u Nišu

Prevođenje ER modela u relacioni model - Primeri

Prof. dr Leonid Stoimenov
Doc. dr Aleksandar Stanimirović

Predstavljanje šeme relacione baze podataka

RADNIK(LIme, SSlovo, Prezime, MatBr, DatRodj, Pol, Plata, Adresa, *MatBrojS*, *BrSek*)

PROJEKAT(Naziv, LokPr, BrojPr, *BrS*)

Predstavljanje šeme relacione baze podataka

RADNIK(LIme, SSlovo, Prezime, MatBr, DatRodj, Pol, Plata, Adresa, MatBrojS, BrSek)

Primary key MatBr

Foreign key MatBrojS ***references*** RADNIK(MatBr)

Foreign key BrojSek ***references*** SEKTOR(SBroj)

Dijagram relacija

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa	*MatBrojS*	*BrSek*

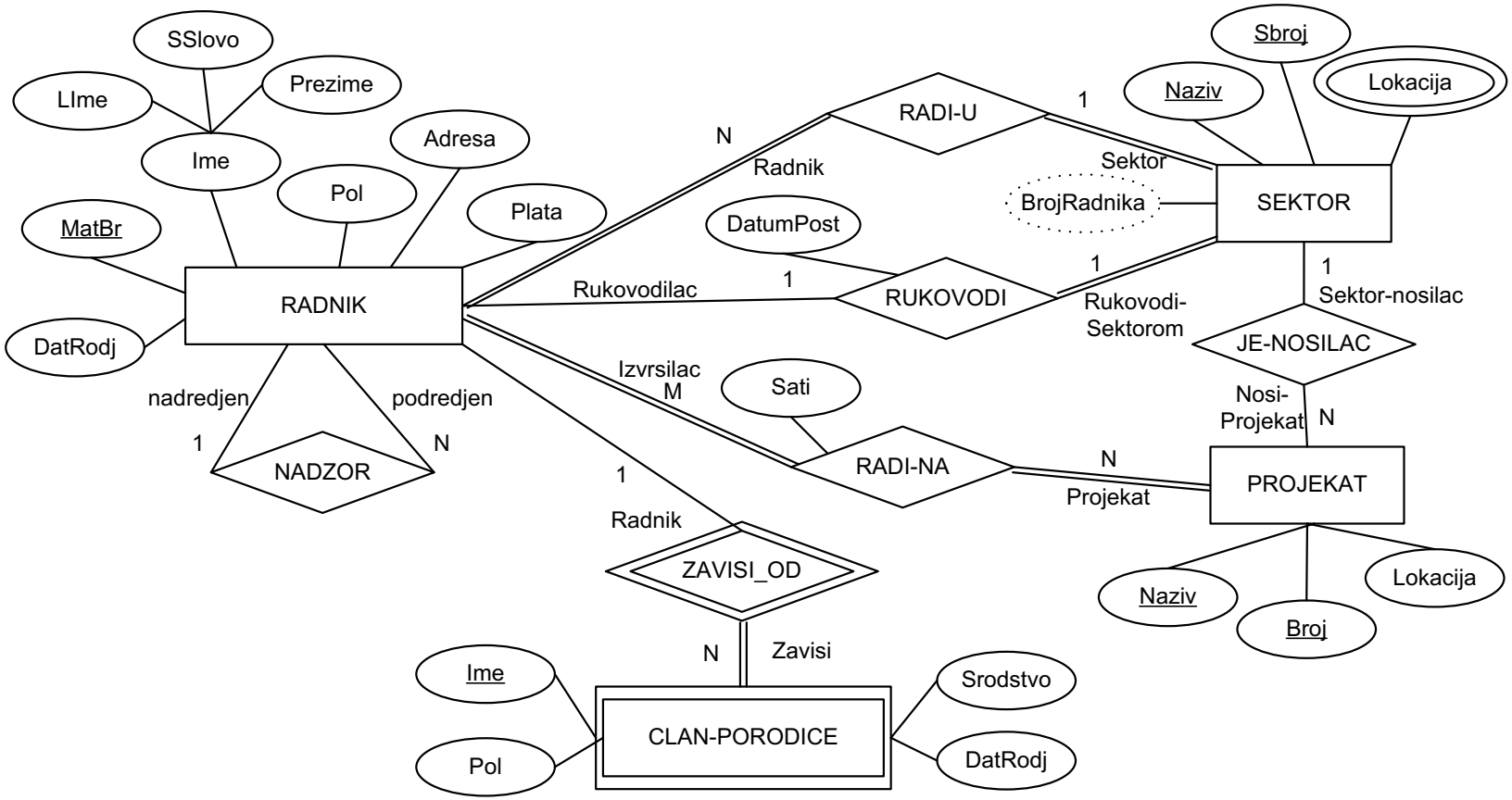
PROJEKAT

Naziv	LokPr	BrojPr	*BrS*

Prevođenje ER modela u relacioni model

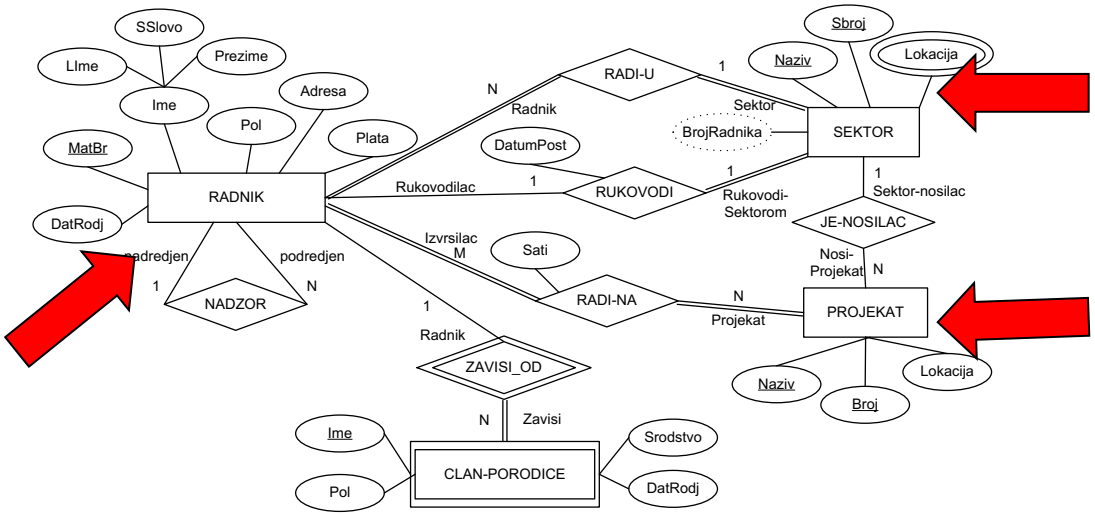
- Jedna od značajnih prednosti ER modela nad ostalim modelima za konceptualno projektovanje je mogućnost direktnog prevođenja u relacioni model (implementacioni model).
- Preslikavanje se odvija kroz sedam sukcesivnih koraka.
- Primer: baza podataka **PREDUZEĆE**.

Baza podataka PREDUZEĆE



1. korak – Prevođenje regularnih tipova entiteta

Za svaki regularni tip entiteta E u ER šemi kreira se relacija R koja sadrži sve proste atribute i sve proste komponente svih složenih atributa iz E. Jedan od ključnih atributa iz E se uzima za primarni ključ relacije R. Ako je ključni atribut u E složen, tada se njegov skup prostih atributa uzima zajedno kao primarni ključ u R.



1. korak – Prevođenje regularnih tipova entiteta

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa

PROJEKAT

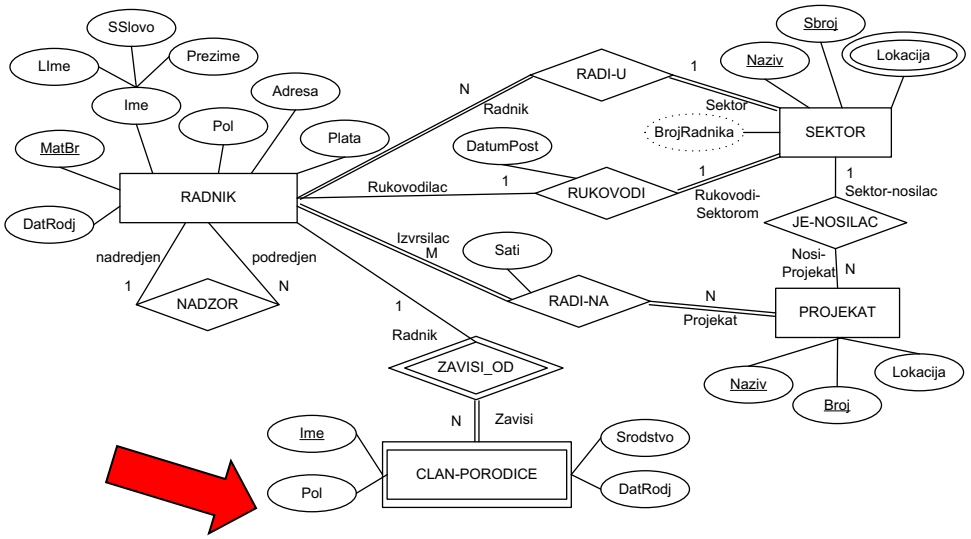
Naziv	LokPr	BrojPr

SEKTOR

Naziv	SektorBr

2. korak – Prevođenje slabih tipova entiteta

Za svaki slabi tip entiteta S u ER šemi čiji je vlasnik tip entiteta E kreira se relacija R koja sadrži sve proste atribute iz S i sve proste komponente svih složenih atributa iz S. Relacija R kao spoljašnji ključ sadrži sve atribute primarnog ključa relacije tipa entiteta E. Za primarni ključ relacije R uzima se kombinacija primarnog ključa vlasnika E i parcijalnog ključa slabog tipa entiteta S ukoliko on postoji.



2. korak – Prevođenje slabih tipova entiteta

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa

PROJEKAT

Naziv	LokPr	BrojPr

SEKTOR

Naziv	SektorBr

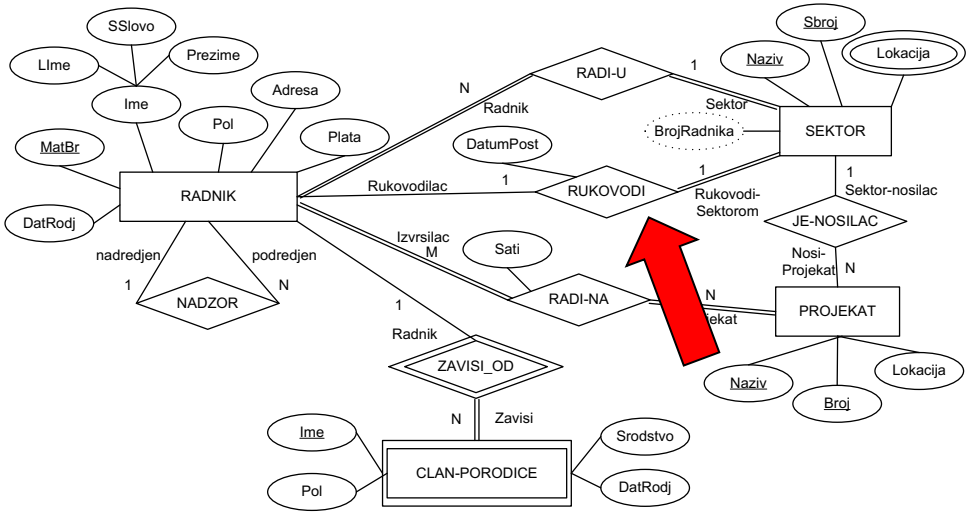
CLAN_PORODICE

MatBrRad	Ime	Pol	Srodstvo	DatRodj

3. korak – Prevođenje veza 1:1

Za svaki binarni tip veza R(1:1) u ER šemi identifikuju se relacije S i T koje odgovaraju tipovima entiteta koji participiraju u R. Bira se jedna od ove dve relacije, recimo S, i u nju se kao spoljašnji ključ uključuje primarni ključ relacije T. Za relaciju S treba birati tip entiteta koji totalno participira u R. Kao atribute relacije S treba uključiti sve proste atribute i sve proste komponente složenih atributa tipa veze R.

Može se izabrati i alternativno rešenje po kome se formira zajednička relacija za tipove entiteta S i T u koju se uključuju svi atributi iz obe relacije. Ovo je rešenje dobro kada oba tipa entiteta totalno participiraju u R i kada ne participiraju ni u jednom drugom tipu veze.



3. korak – Prevođenje veza 1:1

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa

PROJEKAT

Naziv	LokPr	BrojPr

SEKTOR

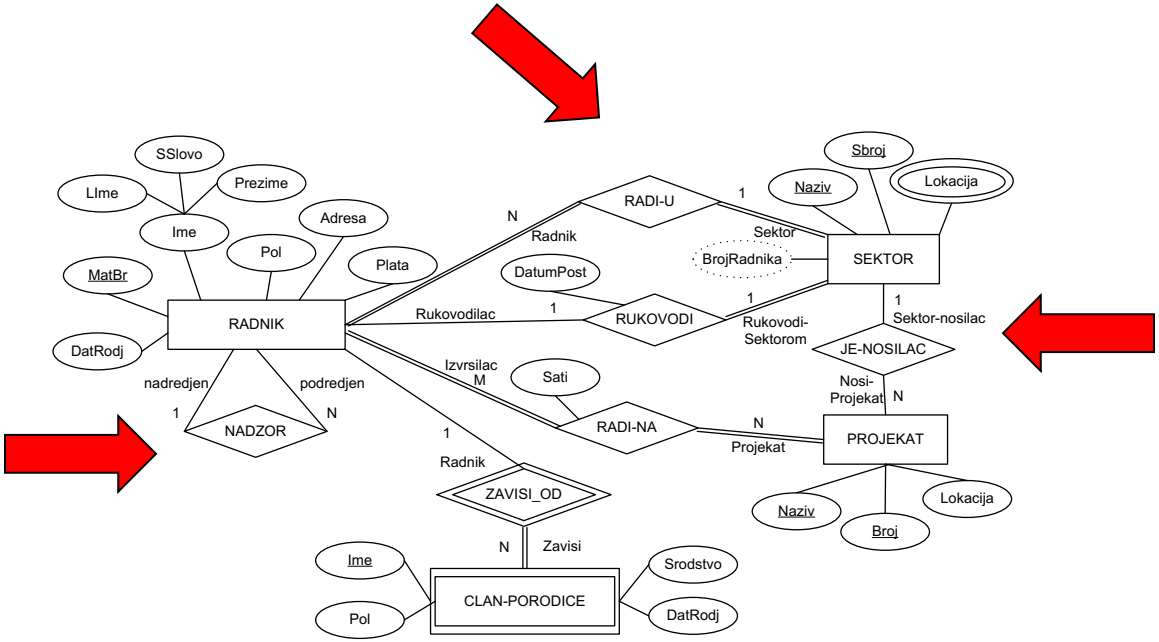
Naziv	SektorBr	*MatBrR*	DatPost

CLAN_PORODICE

MatBrRad	Ime	Pol	Srodstvo	DatRodj

4. korak – Prevođenje veza 1:N

Za svaki binarni tip veze R(1:N) u ER šemi identifikuje se relacija S koja participira na N strani. U S se kao spoljašnji ključ uključuje primarni ključ relacije T koja predstavlja tip entiteta koji participira na 1 strani. Kao atribute relacije S treba uključiti sve proste atribute i sve proste komponente složenih atributa tipa veze R.



4. korak – Prevođenje veza 1:N

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa	*MatBrojS*	*BrSek*

PROJEKAT

Naziv	LokPr	BrojPr	*BrS*

SEKTOR

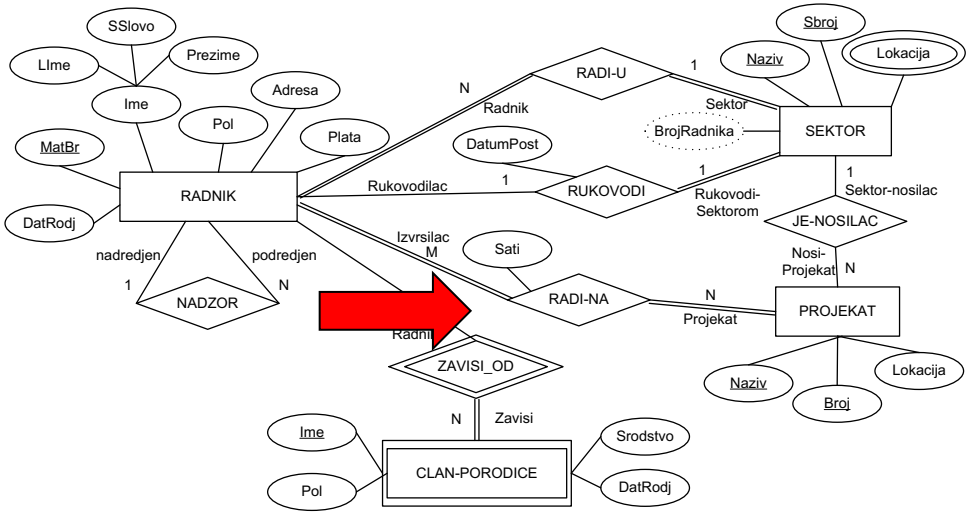
Naziv	SektorBr	*MatBrR*	DatPost

CLAN_PORODICE

MatBrRad	Ime	Pol	Srodstvo	DatRodj

5. korak – Preslikavanje veza M:N

Za svaki binarni tip veze R(M:N) u ER šemi kreira se nova relacija S. U S se kao spoljašnji ključevi uključuju primarni ključevi relacija koje predstavljaju tipove entiteta koji participiraju u R. Kombinacija ovih spoljašnjih ključeva formira primarni ključ relacije S. U relaciji S se takođe uključuju svi prosti atributi i sve proste komponente složenih atributa tipa veze R.



5. korak – Preslikavanje veza M:N

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa	MatBrojS	*BrSek*

PROJEKAT

Naziv	LokPr	BrojPr	BrS

SEKTOR

Naziv	SektorBr	*MatBrR*	DatPost

CLAN_PORODICE

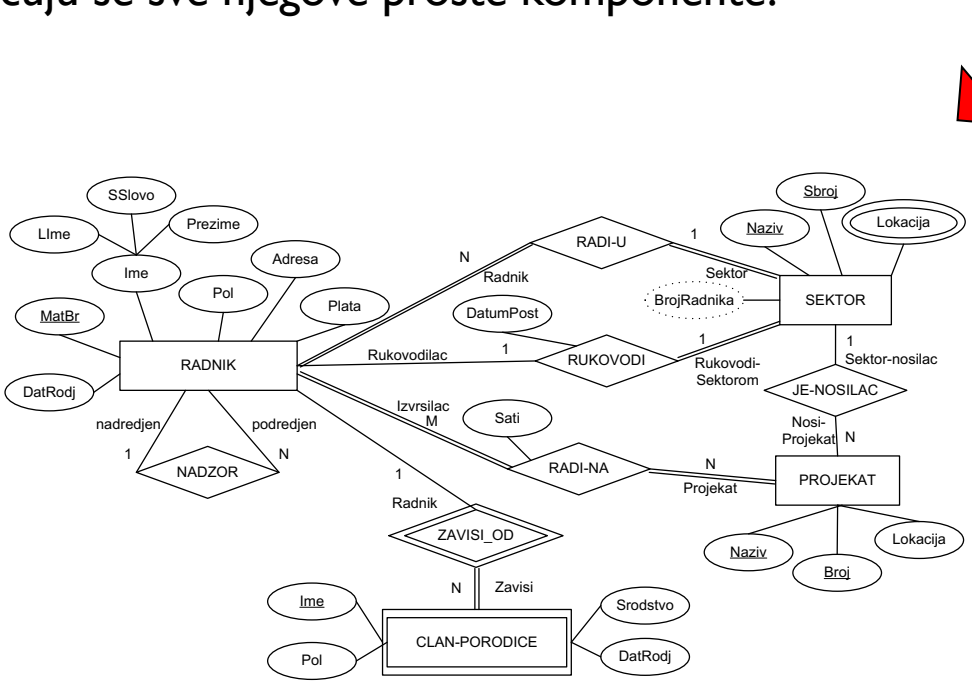
MatBrRad	Ime	Pol	Srodstvo	DatRodj

RADI_NA

Mbr	*BrPr*	Sati

6. korak – Preslikavanje viševrednosnih atributa

Za viševrednosni atribut A tipa entiteta E ili tipa poveznika P kreira se nova relacija R koja sadrži atribut A i primarni ključ K relacije kojom je predstavljen tip entiteta E ili tip veze P. Za primarni ključ relacije R bira se konbinacija A i K. Ako je viševrednosni atribut složen uključuju se sve njegove proste komponente.



6. korak – Preslikavanje viševrednosnih atributa

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa	MatBrojS	*BrSek*

PROJEKAT

Naziv	LokPr	BrojPr	BrS

SEKTOR

Naziv	SektorBr	*MatBrR*	DatPost

CLAN_PORODICE

MatBrRad	Ime	Pol	Srodstvo	DatRodj

RADI_NA

Mbr	BrPr	Sati

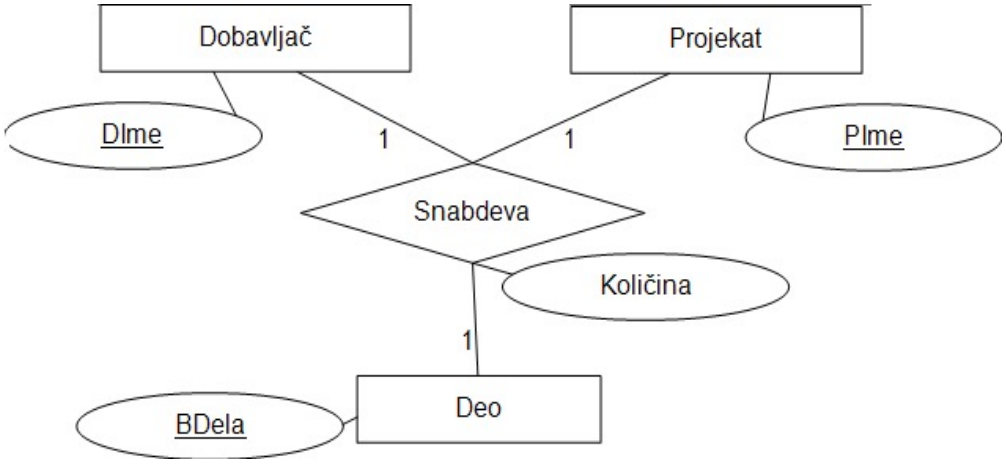
LOK_SEK

BrS	Lokacija

7. korak – Preslikavanje n-arnih veza

Za svaki n-arni tip veze R, n > 2, kreira se nova relacija S. U S se kao spoljašnji ključevi uključuju primarni ključevi relacija koje predstavljaju tipove entiteta koji participiraju u R. Kombinacija ovih spoljašnjih ključeva formira primarni ključ relacije S. U relaciji S se takođe uključuju svi prosti atributi i sve proste komponente svih složenih atributa n-arnog tipa veze.

Ako neki tip entiteta E participira u R sa ograničenjem (min,max) i ako je min = max=1, tada se kao primarni ključ relacije S može uzeti onaj spoljašnji ključni atribut kojim se relacija S referencira na relaciju kojom je predstavljen tip entiteta E.



DOBAVLJAČ

DIme

PROJEKAT

PIme

DEO

BDela

SNABDEVA

DIme	*Pime*	*BDela*	Količina

Relacioni model baze podataka PREDUZEĆE

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Pol	Plata	Adresa	*MatBrojS*	*BrSek*

PROJEKAT

Naziv	LokPr	BrojPr	BrS

SEKTOR

Naziv	SektorBr	*MatBrR*	DatPost

CLAN_PORODICE

MatBrRad	Ime	Pol	Srodstvo	DatRodj

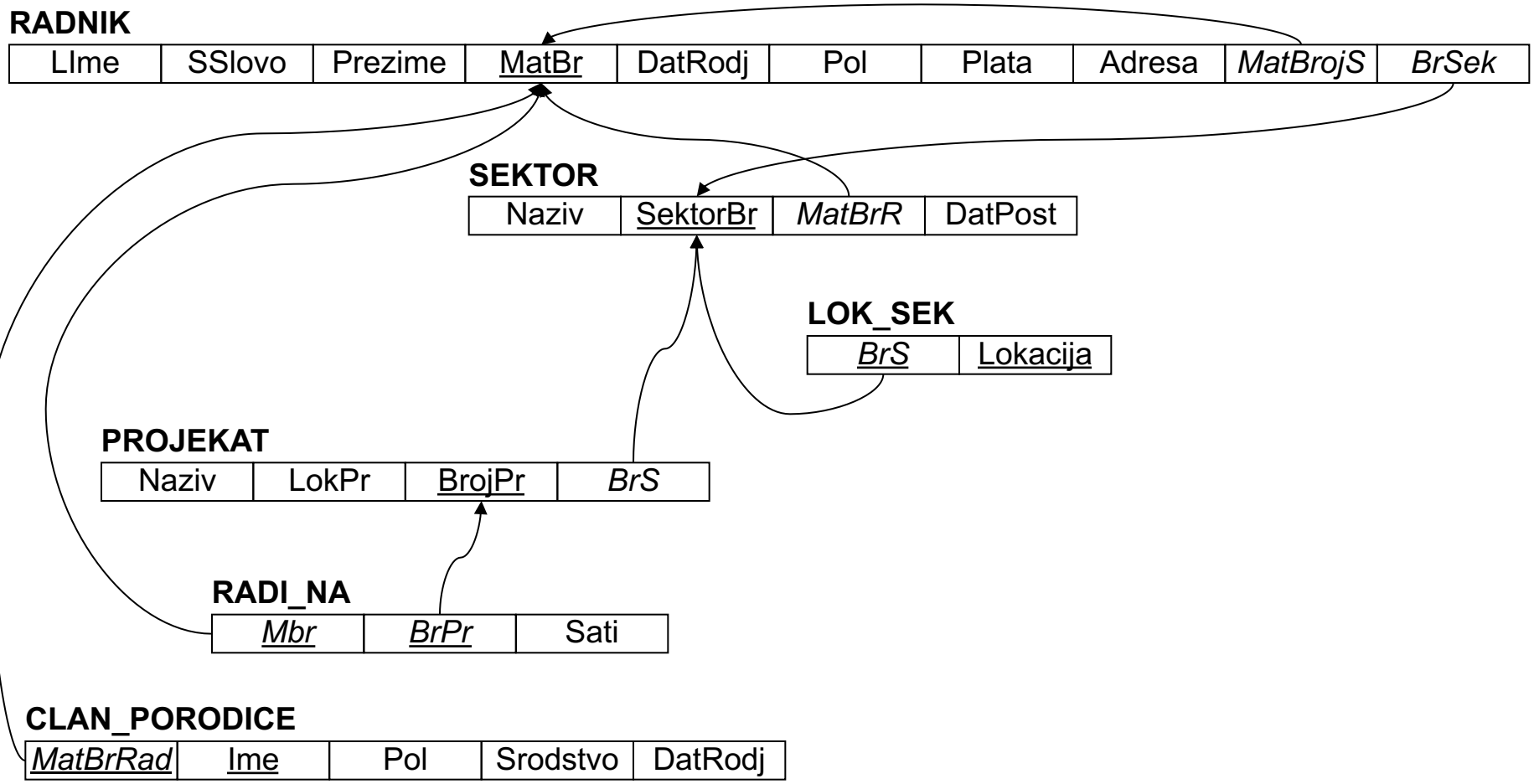
RADI_NA

Mbr	BrPr	Sati

LOK_SEK

BrS	Lokacija

Baza podataka - PREDUZEĆE



Baza podataka - PREDUZEĆE

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Adresa	Pol	Plata	MatBrS	BrSek
Marko	J	Petrović	123456789	9.1.1965	Obilićev Venac	M	30000	333445555	5
Sima	F	Todorović	333445555	8.12.1955	Dušanova 32	M	40000	888665555	5
Jelena	P	Janković	453453453	31.7.1972	Vizantijski bule	Ž	25000	333445555	5
Velibor	T	Jovanović	666884444	15.9.1962	Knjaževačka 13	M	36000	333445555	5
Jovan	S	Obradović	888665555	10.11.1947	Nikole Kopernik	M	55000		1
Aleksandra	S	Petrović	987654321	20.6.1941	Knjaževačka 11	Ž	43000	888665555	4
Stanko	Lj	Manojlović	987987987	29.3.1969	Nemanjina 23	M	25000	987654321	4
Valentina	D	Kovačević	999887777	19.1.1968	Knjeginje Ljubic	Ž	25000	987654321	4

SEKTOR

Naziv	SBroj	MatBrR	DatPost
Razvoj	5	333445555	22.5.1988
Administracija	4	987654321	1.1.1985
Uprava	1	888665555	19.6.1981

LOK_SEK

BrS	Lokacija
5	Niš
5	Pirot
4	Leskovac
4	Niš
1	Niš

PROJEKAT

Naziv	LokPr	BrojPr	BrS
ProizvodX	Niš	1	5
ProizvodY	Pirot	2	5
ProizvodZ	Niš	3	5
Reorganizacija	Niš	10	1
Informacioni sis	Leskovac	20	4
Godišnji izvešta	Niš	30	4

Baza podataka - PREDUZEĆE

RADI_NA

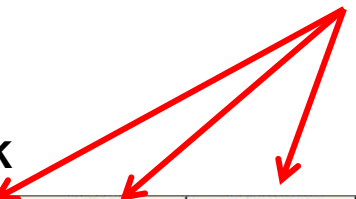
MatBr	BrPr	Sati
123456789	1	32
123456789	2	7
666884444	3	40
453453453	1	20
453453453	2	20
333445555	2	10
333445555	3	10
333445555	10	10
333445555	20	10
999887777	30	30
999887777	10	10
987987987	10	25
987987987	20	5
987654321	30	20
987654321	20	15
888665555	20	

CLAN_PORODICE

MatBrRad	Ime	Pol	Srodstvo	DatRodj
333445555	Nevena	Ž	ćerka	5.4.1986
333445555	Milan	M	sin	25.10.1983
333445555	Teodora	Ž	supruga	3.5.1958
987654321	Veselin	M	suprug	28.2.1942
123456789	Mihajlo	M	sin	4.1.1988
123456789	Ana	Ž	ćerka	30.12.1988
123456789	Jelena	Ž	supruga	5.5.1966

Baza podataka - PREDUZEĆE

- Kompozitni atribut



RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Adresa	Pol	Plata	MatBrS	BrSek
Marko	J	Petrović	123456789	9.1.1965	Obilićev Venac	M	30000	333445555	5
Sima	F	Todorović	333445555	8.12.1955	Dušanova 32	M	40000	888665555	5
Jelena	P	Janković	453453453	31.7.1972	Vizantijski bule	Ž	25000	333445555	5
Velibor	T	Jovanović	666884444	15.9.1962	Knjaževačka 13	M	36000	333445555	5
Jovan	S	Obradović	888665555	10.11.1947	Nikole Kopernik	M	55000	0	1
Aleksandra	S	Petrović	987654321	20.6.1941	Knjaževačka 11	Ž	43000	888665555	4
Stanko	Lj	Manojlović	987987987	29.3.1969	Nemanjina 23	M	25000	987654321	4
Valentina	D	Kovačević	999887777	19.1.1968	Knjeginje Ljubic	Ž	25000	987654321	4

Baza podataka - PREDUZEĆE

- Unarna (rekurzivna) veza

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Adresa	Pol	Plata	MatBrS	BrSek
Marko	J	Petrović	123456789	9.1.1965	Obilićev Venac	M	30000	333445555	5
Sima	F	Todorović	333445555	8.12.1955	Dušanova 32	M	40000	888665555	5
Jelena	P	Janković	453453453	31.7.1972	Vizantijski bule	Ž	25000	333445555	5
Velibor	T	Jovanović	666884444	15.9.1962	Knjaževačka 1	M	36000	333445555	5
Jovan	S	Obradović	888665555	10.11.1947	Nikole Koperni	M	55000	0	1
Aleksandra	S	Petrović	987654321	20.6.1941	Knjaževačka 11	Ž	43000	888665555	4
Stanko	Lj	Manojlović	987987987	29.3.1969	Nemanjina 23	M	25000	987654321	4
Valentina	D	Kovačević	999887777	19.1.1968	Knjeginje Ljubic	Ž	25000	987654321	4

Baza podataka - PREDUZEĆE

- Veza 1:1

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Adresa	Pol	Plata	MatBrS	BrSek
Marko	J	Petrović	123456789	9.1.1965	Obilićev Venac	M	30000	333445555	5
Sima	F	Todorović	333445555	8.12.1955	Dušanova 32	M	40000	888665555	5
Jelena	P	Janković	453453453	31.7.1972	Vizantijski bule	Ž	25000	333445555	5
Velibor	T	Jovanović	666884444	15.9.1962	Knjaževačka 13	M	36000	333445555	5
Jovan	S	Obradović	888665555	10.11.1947	Nikole Kopernik	M	55000	0	1
Aleksandra	S	Petrović	987654321	20.6.1941	Knjaževačka 11	Ž	43000	888665555	4
Stanko	Lj	Manojlović	987987987	29.3.1969	Nemanjina 23	M	25000	987654321	4
Valentina	D	Kovačević	999887777	19.1.1968	Knjeginje Ljubic	Ž	25000	987654321	4

SEKTOR

Naziv	SBroj	MatBrR	DatPost
Razvoj	5	333445555	22.5.1988
Administracija	4	987654321	1.1.1985
Uprava	1	888665555	19.6.1981

Baza podataka - PREDUZEĆE

- Veza 1:N

SEKTOR

Naziv	SBroj	MatBrR	DatPost
Razvoj	5	333445555	22.5.1988
Administracija	4	987654321	1.1.1985
Uprava	1	888665555	19.6.1981

PROJEKAT

Naziv	LokPr	BrojPr	BrS
ProizvodX	Niš	1	5
ProizvodY	Pirot	2	5
ProizvodZ	Niš	3	5
Reorganizacija	Niš	10	1
Informacioni sis	Leskovac	20	4
Godišnji izvešta	Niš	30	4

Baza podataka - PREDUZEĆE

- Slabi tip entiteta

RADNIK

LIme	SSlovo	Prezime	MatBr	DatRodj	Adresa	Pol	Plata	MatBrS	BrSek
Marko	J	Petrović	123456789	9.1.1965	Obilićev Venac	M	30000	333445555	5
Sima	F	Todorović	333445555	8.12.1955	Dušanova 32	M	40000	888665555	5
Jelena	P	Janković	453453453	31.7.1972	Vizantijski bule	Ž	25000	333445555	5
Velibor	T	Jovanović	666884444	15.9.1962	Knjaževačka 13	M	36000	333445555	5
Jovan	S	Obradović	888665555	10.11.1947	Nikole Kopernik	M	55000	0	1
Aleksandra	S	Petrović	987654321	20.6.1941	Knjaževačka 11	Ž	43000	888665555	4
Stanko	Lj	Manojlović	987987987	29.3.1969	Nemanjina 23	M	25000	987654321	4
Valentina	D	Kovačević	999887777	19.1.1968	Knjeginje Ljubic	Ž	25000	987654321	4

CLAN_PORODICE

MatBrRad	Ime	Pol	Srodstvo	DatRodj
333445555	Nevena	Ž	ćerka	5.4.1986
333445555	Milan	M	sin	25.10.1983
333445555	Teodora	Ž	supruga	3.5.1958
987654321	Veselin	M	suprug	28.2.1942
123456789	Mihajlo	M	sin	4.1.1988
123456789	Ana	Ž	ćerka	30.12.1988
123456789	Jelena	Ž	supruga	5.5.1966

Baza podataka - PREDUZEĆE

- Viševrednosni atribut

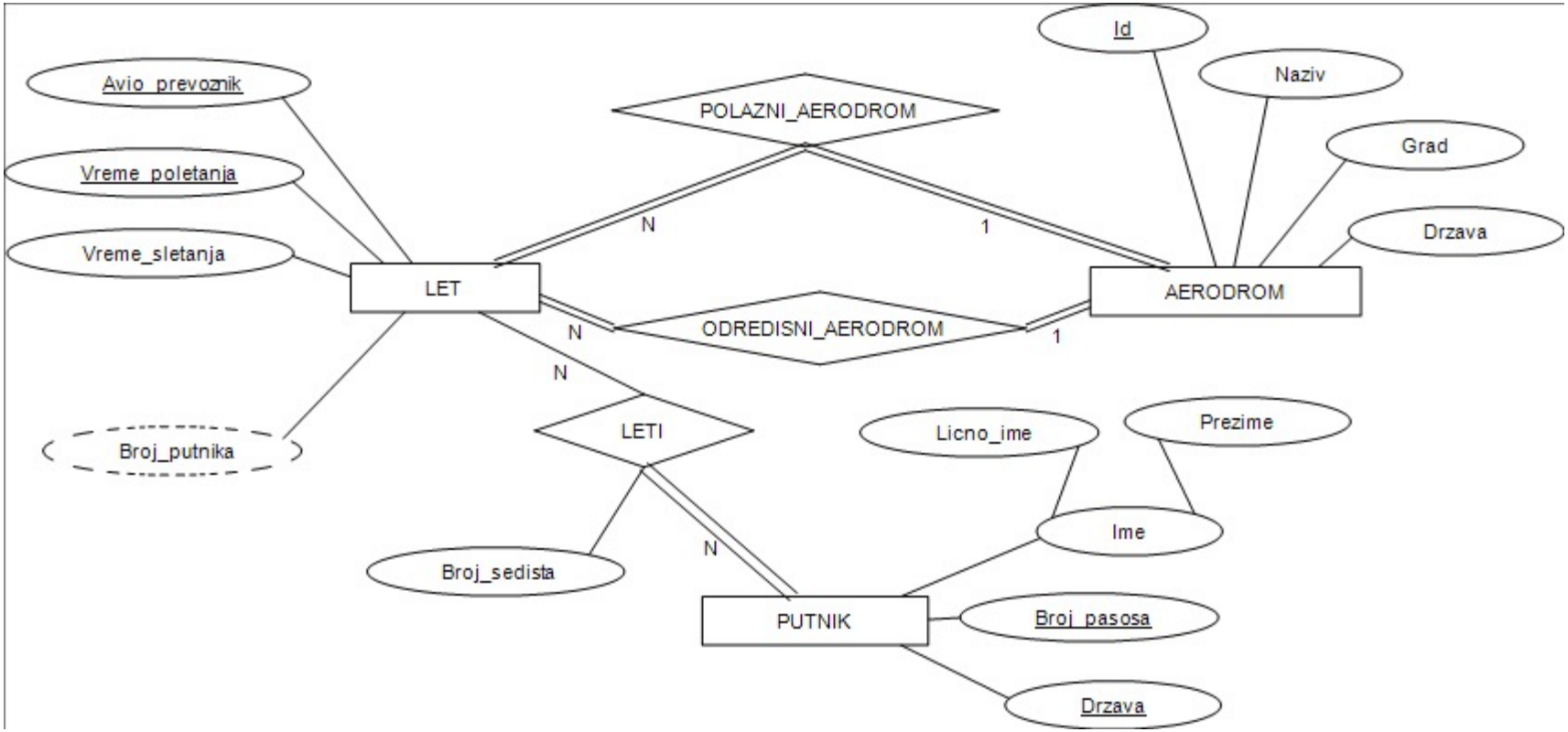
SEKTOR

Naziv	SBroj	MatBrR	DatPost
Razvoj	5	333445555	22.5.1988
Administracija	4	987654321	1.1.1985
Uprava	1	888665555	19.6.1981

LOK_SEK

BrS	Lokacija
5	Niš
5	Pirot
4	Leskovac
4	Niš
1	Niš

Primeri


Id
Naziv
Grad
Drzava
Avio_prevoznik
POLAZNI_AERODROM
Vreme_poletanja
Vreme_sletanja
N
1
LET
AERODROM
N
ODREDISNI_AERODROM
1
N
Broj_putnika
LETI
Licno_ime
Prezime
Ime
N
Broj_sedista
PUTNIK
Broj_pasosa
Drzava

Primeri

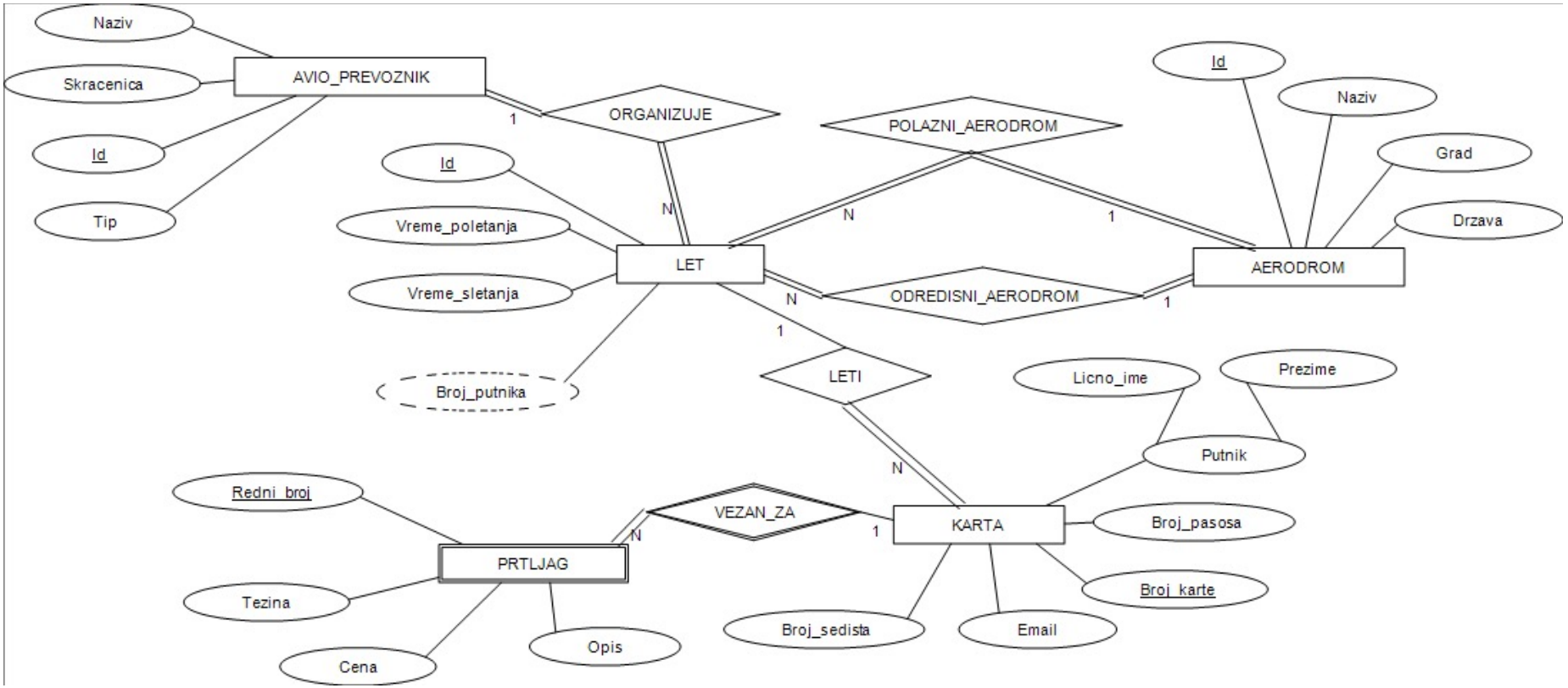
LET(Avio_prevoznik,Vreme_poletanja,Vreme_sletanja, *Polazni_aerodrome_id*, *Odredisni_aerodrome_id*)

PUTNIK(Licno_ime, Prezime, Broj_pasosa, Drzava)

AERODROM(Id, Naziv, Grad, Drzava)

LETI(*Avio_prevoznik*, *Vreme_poletanja*, *Broj_pasosa*, *Drzava*, Broj_sedista)

Primeri


Naziv
Skracenica
AVIO_PREVOZNIK
Id
Tip
1
ORGANIZUJE
N
Id
Vreme_poletanja
Vreme_sletanja
LET
Broj_putnika
POLAZNI_AERODROM
N
1
Id
Naziv
Grad
Drzava
AERODROM
N
ODREDISNI_AERODROM
1
1
LETI
N
Licno_ime
Prezime
Putnik
Redni_broj
PRTLJAG
Tezina
Cena
Opis
N
VEZAN_ZA
1
KARTA
Broj_pasosa
Broj_karte
Broj_sedista
Email

Primeri

AVIO_PREVOZNIK(Naziv, Skracenica, Id, Tip)

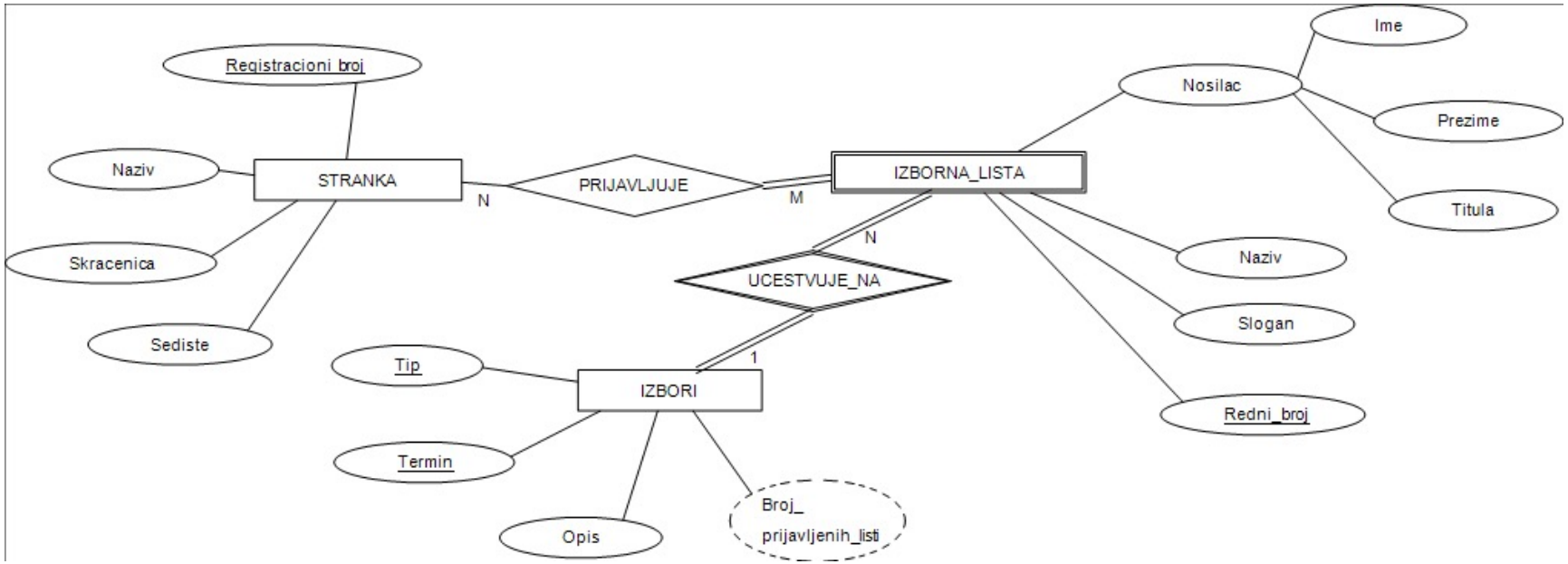
LET(Vreme_poletanja, Vreme_sletanja, Id, *Prevoznik_id*, *Polazni_aerodrome_id*, *Odredisni_aerodrome_id)*

AERODROM(Id, Naziv, Grad, Drzava)

KARTA(Licno_ime, Prezime, Broj_pasosa, Broj_karte, Email, Broj_sedista, *Let_id*)

PRTLJAG(*Broj_karte*, Redni_broj, Tezina, Cena, Opis)

Primeri


Registracioni broj
Naziv
STRANKA
Skracenica
Sediste
N
PRIJAVLJUJE
M
IZBORNA_LISTA
Ime
Nosilac
Prezime
Titula
N
UCESTVUJE_NA
Naziv
Slogan
1
Tip
IZBORI
Termin
Opis
Broj_
prijavljenih_listi
Redni_broj

Primeri

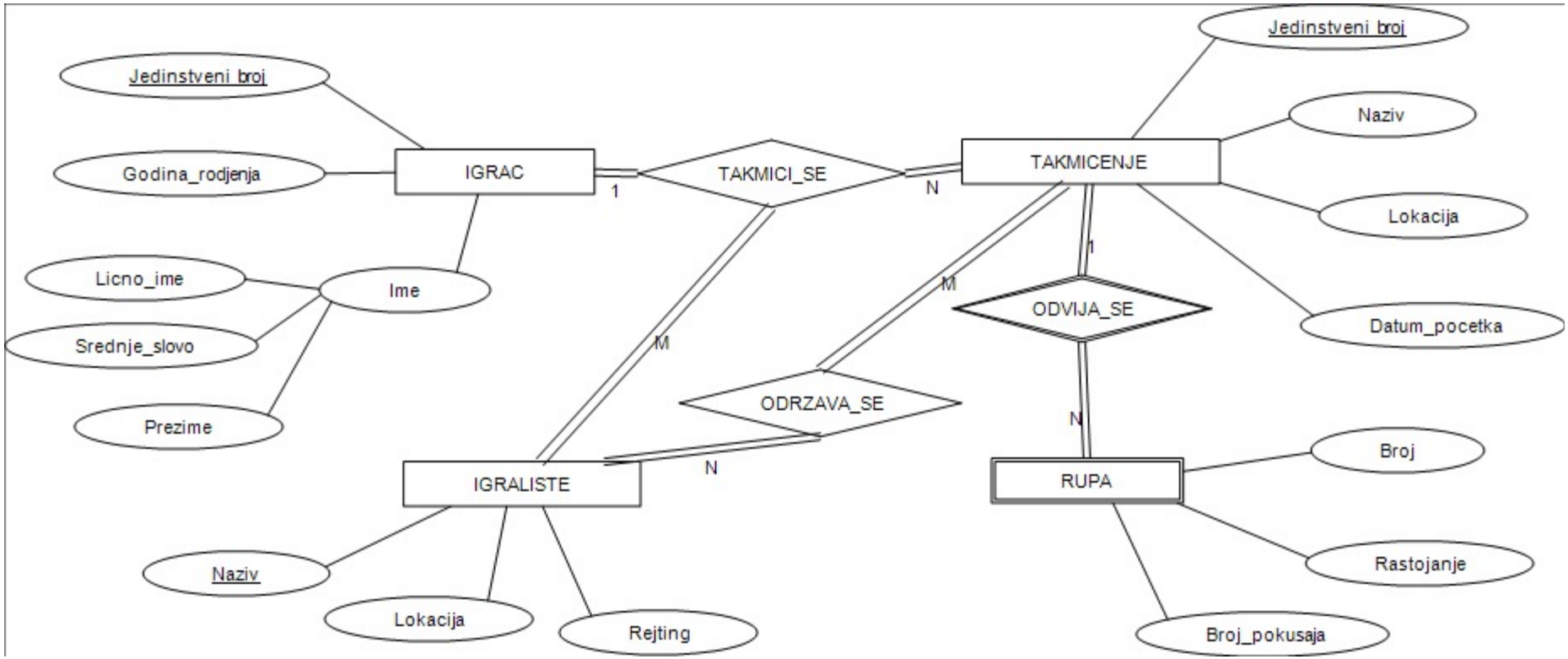
STRANKA(Registracion_broj, Naziv, Skracenica, Sediste)

IZBORI(Tip, Termin, Opis)

IZBORNA_LISTA(Ime, Prezime, Titula, Naziv, Slogan, *Tip*, *Termin*, Redni_broj)

PRIJAVLJUJE(*Stranka*, *Tip*, *Termin*, *Redni_broj*)

Primeri


Jedinstveni broj
Godina_rodjenja
IGRAC
1
TAKMICI_SE
N
TAKMICENJE
Jedinstveni broj
Naziv
Lokacija
Datum_pocetka
Licno_ime
Ime
Srednje_slovo
Prezime
M
ODVIJA_SE
1
N
ODRZAVA_SE
M
N
IGRALISTE
Naziv
Lokacija
Rejting
RUPA
Broj
Rastojanje
Broj_pokusaja

Primeri

IGRAC(Jedinstveni_broj, Godina_rodjenja, Licno_ime, Srednje_slovo, Prezime)

IGRALISTE(Naziv, Lokacija, Rejting)

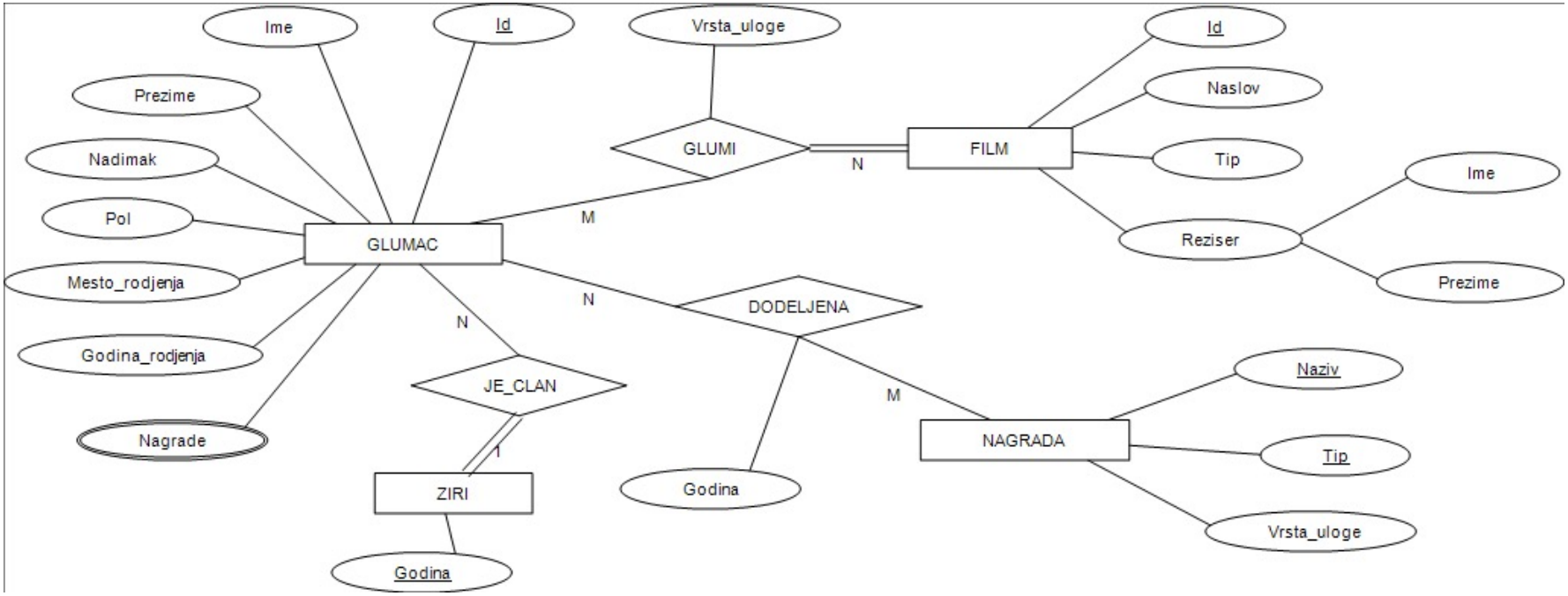
TAKMICENJE(Jedinstveni_broj, Naziv, Lokacija, Datum_pocetka)

RUPA(*Takmicenje*, Broj, Rastojanje, Broj_pokusaja)

ODRZAVA_SE(*Takmicenje*, *Igraliste*)

TAKMICI_SE(*Igrac*, *Igraliste*, *Takmicenje*)

Primeri


Ime
Id
Vrsta_uloge
Id
Prezime
Naslov
Nadimak
GLUMI
N
FILM
Tip
Pol
M
Ime
GLUMAC
Reziser
Mesto_rodjenja
N
Prezime
DODELJENA
N
Godina_rodjenja
JE_CLAN
Naziv
M
Nagrade
NAGRADA
1
Tip
ZIRI
Godina
Vrsta_uloge
Godina

Primeri

GLUMAC(Id, Ime, Prezime, Nadimak, Mesto_rodjenja, Godina_rodjenja, *Godina_ziri*)

FILM(Id, Naslov, Tip, Ime, Prezime)

NAGRADA(Naziv, Tip, Vrsta_uloge)

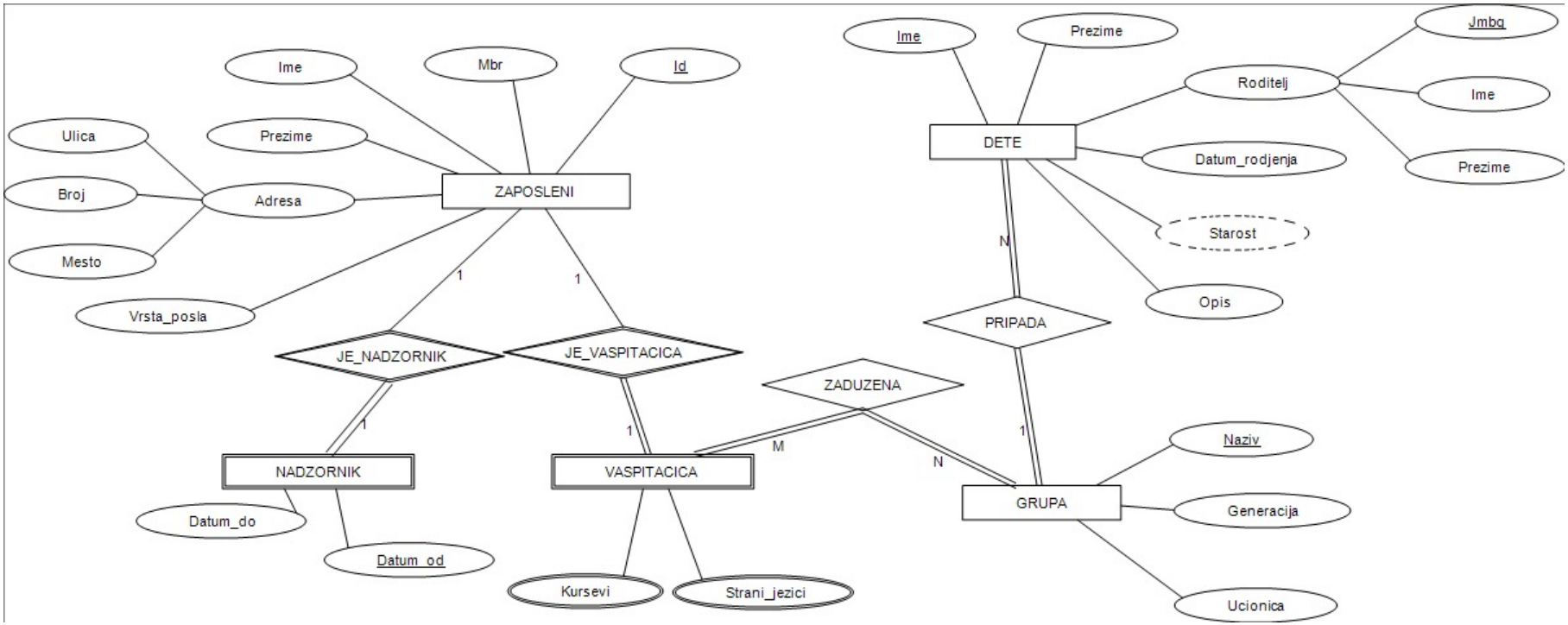
ZIRI(Godina)

GLUMI(*Glumac*, *Film*, Vrsta_uloge)

DODELJENA(*Glumac*, *NagradaNaziv*, *NagradaTip*, Godina)

NAGRADE(*Glumac*, Nagrada)

Primeri


Ime
Mbr
Id
Ulica
Prezime
Broj
Adresa
ZAPOSLENI
Mesto
Vrsta_posla
1
1
JE_NADZORNIK
JE_VASPITACICA
1
1
NADZORNIK
VASPITACICA
Datum_do
Datum_od
Kursevi
Strani_jezici
Ime
Prezime
Jmbg
Roditelj
Ime
Prezime
DETE
Datum_rodjenja
Starost
Opis
N
PRIPADA
1
ZADUZENA
M
N
Naziv
GRUPA
Generacija
Ucionica

Primeri

ZAPOSLENI(Id, Mbr, Ime, Prezime, Ulica, Broj, Mesto, Vrsta_posla)

DETE(Ime, Prezime, JMBG_roditelja, Ime_roditelja, Prezime_roditelja, Datum_rodjenja, Opis, *Grupa*)

GRUPA(Naziv, Generacija, Ucionica)

NADZORNIK(*Zaposleni*, Datum_od, Datum_do)

VASPITAC(*Zaposleni*)

ZADUZENA(*Zaposleni*, *Grupa*)

KURSEVI(*Zaposleni*, Kurs) STRANI_JEZICI(*Zaposleni*, Jezik)

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Relaciona algebra

Prof.dr Leonid Stoimenov

Zašto je važna relaciona algebra?

- Obezbeđuje **formalnu osnovu** za operacije relacionog modela
- Koristi se kao osnova za
  - **implementaciju** i
  - **optimizaciju** upita u RDBMS-u

- Neki od njenih koncepata su ugrađeni u SQL standardni upitni jezik za relacioni model podataka

Relaciona algebra – Uvod (1)

- Relaciona algebra je formalni jezik za relacioni model
- Zasniva se na matematičkoj teoriji skupova

- **Izraz relacione algebre -** sekvenca operacija relacione algebre

- Izrazi relacione algebre se sastoje od **operatora** i **operanada**
  - **Operandi** su **relacije** (skupovi torki ili reprezentanti skupova torki)
  - **Rezultat** je takođe **relacija**

Relaciona algebra – Uvod (2)

- Za iskazivanje **upita** putem relacione algebre koriste se dve vrste operatora
  - **Specijalni operatori** relacione algebre
    - selekcija,
    - projekcija,
    - spoj,
    - deljenje
  - **Standardni operatori** matematičke teorije skupova
    - Unija,
    - presek,
    - razlika,
    - Dekartov proizvod

Operacije relacione algebre

- **Osnovni skup operacija** relacione algebre:
  - Selekcija (SELECT)
  - Projekcija (PROJECT)
  - Unija (UNION)
  - Razlika (DIFFERENCE, MINUS, EXCEPT)
  - Dekartov proizvod (CARTESIAN PRODUCT, CROSS PRODUCT, CROSS JOIN)

Operacije relacione algebre

- **Prošireni skup operacija** relacione algebre:
  - Presek (INTERSECTION)
  - Deljenje (DIVISION)
  - Spoj (JOIN)
    - Unutrašnji (INNER)
    - Spoljašnji (OUTER)
      - Levi (LEFT)
      - Desni (RIGHT)
      - Potpuni (FULL)
    - Polu spoj (SEMI JOIN)
- Operacije iz proširenog skupa se mogu izvesti iz operacija osnovnog skupa

Selekcija

- To je operacija kojom se iz relacije **izdvajaju** one **torke** koje imaju zadatu vrednost specificiranih atributa
- Atributi i njihove vrednosti po kojima se vrši selekcija se zadaju **uslovom selekcije**

radnik

MBR	LIME	LD	*SBR*
2203	MIRA	50 000	**10**
2817	PERA	40 000	20
2932	MIRA	35 000	**10**
2995	GOCA	18 000	30
3305	LAZA	60 000	40
3515	JOVAN	20 000	20
3819	VLADA	65 000	30

Selekcija

- Operacija selekcije se označava sa

$$\sigma_{\text{<uslov selekcije>}}(\text{<ime relacije>})$$

- **Rezultat**: RELACIJA
- Selekcija se vrši iz zadate relacije <ime relacije>
- Atributi i njihove vrednosti po kojima se vrši selekcija se zadaju **uslovom selekcije**
- Redosled atributa rezultantne relacije ekvivalentan je redosledu atributa relacije nad kojom je primenjena selekcija

Uslov selekcije

- **Prost uslov selekcije** je logički izraz oblika:

  $A_i \, \theta \, A_j$

  $A_i \, \theta \, C$

  $C \, \theta \, A_i$

- gde je
  - $\theta \in \{<, =, >, <=, >=, !=\}$
  - A_i i A_j imena atributa relacije nad kojom se selekcija izvodi
  - C konstanta koja uzima vrednosti iz domena atributa Ai

- **Složeni uslov selekcije** se sastoji od prostih uslova selekcije koji su povezani operatorima $\wedge$, $\vee$ i $\neg$, odnosno AND, OR i NOT

Primer selekcije

- Iz relacije radnik nad šemom relacije

  RADNIK (MBR,LIME,LD,SBR)

  izabrati sve radnike koji ispunjavaju uslov:

  (a) LD > 20000

  (b) LD > 20000 i SBR = 10

Primer selekcije

- Iz relacije radnik nad šemom relacije

  RADNIK (MBR,LIME,LD,SBR)

  izabrati sve radnike koji ispunjavaju uslov:

  (a) LD > 20000

  (b) LD > 20000 i SBR = 10

- Odgovarajuće operacije relacione algebre su:

  (a) $\sigma_{LD>20000}$ (radnik)

  (b) $\sigma_{LD>20000\ AND\ SBR=10}$ (radnik)

Primer selekcije

radnik

MBR	LIME	LD	*SBR*
2203	MIRA	50 000	10
2817	PERA	40 000	20
2932	MIRA	35 000	10
2995	GOCA	18 000	30
3305	LAZA	60 000	40
3515	JOVAN	20 000	20
3819	VLADA	65 000	30

(a) $\sigma_{LD>20000}$ (radnik)

MBR	LIME	LD	SBR
2203	MIRA	50 000	10
2817	PERA	40 000	20
2932	MIRA	35 000	10
3305	LAZA	60 000	40
3819	VLADA	65 000	30

(b) $\sigma_{LD>20000 \text{ AND } SBR=10}$ (radnik)

MBR	LIME	LD	SBR
2203	MIRA	50 000	10
2932	MIRA	35 000	10

Svojstva selekcije

- Selekcija je **komutativna** operacija

$$\sigma_{<uslov1>}(\sigma_{<uslov2>}(r)) = \sigma_{<uslov2>}(\sigma_{<uslov1>}(r))$$

- Posledica
  - Sekvenca operacija selekcije može biti primenjena u bilo kom redosledu ili zamenjena jednom operacijom selekcije sa složenim uslovom

$$\sigma_{<uslov1>}(\sigma_{<uslov2>}(\ldots \sigma_{<uslovn>}(r)) =$$

$$\sigma_{<uslov1> \text{ AND } <uslov2> \text{ AND } \ldots \text{ AND } <uslovn>}(r)$$

Projekcija

- To je operacija kojom se iz relacije **izdvajaju kolone** koje odgovaraju atributima po kojima se vrši projekcija i kojom se iz tako dobijene relacije eliminišu jednake torke

- Operator projekcije se označava sa

$$\pi_{\text{<lista atributa>}}(\text{<ime relacije>})$$

radnik

MBR	LIME	LD	*SBR*
2203	MIRA	50 000	10
2817	PERA	40 000	20
2932	MIRA	35 000	10
2995	GOCA	18 000	30
3305	LAZA	60 000	40
3515	JOVAN	20 000	20
3819	VLADA	65 000	30

Projekcija

π <lista atributa>(<ime relacije>)

- Rezultat: **relacija**
- <lista atributa> definiše za koje atribute se vrši projekcija
- Redosled atributa rezultantne relacije definisan je listom atributa projekcije
- Projekcija nije komutativna

Primer projekcije

- Projektovati po atributima LIME, SBR relaciju radnik definisanu nad šemom relacije RADNIK(MBR, LIME, LD, SBR)

radnik

MBR	LIME	LD	*SBR*
2203	MIRA	50 000	10
2817	PERA	40 000	20
2932	MIRA	35 000	10
2995	GOCA	18 000	30
3305	LAZA	60 000	40
3515	JOVAN	20 000	20
3819	VLADA	65 000	30

$\pi_{LIME,SBR}(radnik)$

LIME	SBR
MIRA	10
PERA	20
GOCA	30
LAZA	40
JOVAN	20
VLADA	30

$\pi_{SBR,LIME}(radnik)$

SBR	LIME
10	MIRA
20	PERA
30	GOCA
40	LAZA
20	JOVAN
30	VLAD A

Preimenovanje

- $\rho_{s(B1,B2,\ldots,Bn)}(r)$
  - Preimenuje ime relacije i imena atributa
- $\rho_{s}(r)$
  - Preimenuje ime relacije
- $\rho_{(B1,B2,\ldots,Bn)}(r)$
  - Preimenuje atribute relacije

s je novo ime relacije r(A1,A2,…,An)

B1,B2,…,Bn su nova imena atributa A1,A2,…,An

Operacije relacione algebre iz teorije skupova

- Standardne matematičke operacije nad skupovima
  - Unija
  - Presek
  - Razlika
  - Dekartov proizvod

Unija

- Unija relacija r i s je skup torki koje pripadaju relacijama r ili s ili obema.

- Uslov za obavljanje ove operacije je da su r i s **unijski kompatibilne relacije**
  - To znači da obe relacije imaju **isti stepen** i da su **domeni** korespodentnih atributa u obe relacije **isti**

- Rezultujuća relacija ima imena atributa prve relacije
- Unija je komutativna i asocijativna operacija

$$r \cup s = s \cup r$$

$$(r \cup s) \cup t = r \cup (s \cup t)$$

Razlika

- Razlika relacija r i s je skup torki koje pripadaju relaciji r ali ne pripadaju relaciji s
- Relacije r i s treba da su unijski kompatibilne
- Rezultujuća relacija ima imena atributa prve relacije
- Razlika nije komutativna operacija

$$r - s \neq s - r$$

Presek

- Presek relacija r i s je skup torki koje pripadaju obema relacijama

- Uslov za obavljanje ove operacije je da su r i s unijski **kompatibilne** relacije
- Rezultujuća relacija ima imena atributa prve relacije

- Presek se može izraziti preko razlike na sledeći način:

$$r \cap s = r - (r - s)$$

- Presek je komutativna i asocijativna operacija

$$r \cap s = s \cap r$$

$$(r \cap s) \cap t = r \cap (s \cap t)$$

Primer skupovnih operacija (1)

student

IND	IME	DATR	GRAD
1211	PERA	010971	NI
1213	VERA	130872	NI
1215	MILA	150171	LE
1217	VERA	220371	SV

kandidat

ID	IME	DATR	GRAD
1213	VERA	130872	NI
1217	VERA	220371	SV
1223	MIKA	101072	LE

student - kandidat

IND	IME	DATR	GRAD
1211	PERA	010971	NI
1215	MILA	150171	LE

student $\cup$ kandidat

IND	IME	DATR	GRAD
1211	PERA	010971	NI
1213	VERA	130872	NI
1215	MILA	150171	LE
1217	VERA	220371	SV
1223	MIKA	101072	LE

Primer skupovnih operacija (2)

student

IND	IME	DATR	GRAD
1211	PERA	010971	NI
1213	VERA	130872	NI
1215	MILA	150171	LE
1217	VERA	220371	SV

kandidat

ID	IME	DATR	GRAD
1213	VERA	130872	NI
1217	VERA	220371	SV
1223	MIKA	101072	LE

student ∩ kandidat

IND	IME	DATR	GRAD
1213	VERA	130872	NI
1217	VERA	220371	SV

kandidat - student

ID	IME	DATR	GRAD
1223	MIKA	101072	LE

Dekartov proizvod

- Dekartov proizvod relacija r i s nad šemama relacije R(A1, A2, ..., An) i S(B1, B2, ..., Bm)
  je relacija q nad šemom relacije
  Q(A1, A2 ,..., An, B1, B2, ..., Bm)
  koju čine torke dužine n+m, gde prvih n komponenti čini torku u r, a drugih m torku u s
  - Relacija q ima po jednu torku za sve kombinacije torki, jedna iz r, jedna iz s
  - Ako r ima K_r torki i ako s ima K_s torki, tada q ima K_r x K_s torki
- Notacija

$$q = r \times s$$

Primer Dekartovog proizvoda relacija

r

A	B	C
a1	b1	c1
a2	b2	c2

s

D	E
d1	e1
d2	e2
d3	e3

r x s

A	B	C	D	E
a1	b1	c1	d1	e1
a1	b1	c1	d2	e2
a1	b1	c1	d3	e3
a2	b2	c2	d1	e1
a2	b2	c2	d2	e2
a2	b3	c2	d3	e3

Deljenje

- Količnik relacija r i s, r/s je skup torki t koje se javljaju u r u kombinaciji sa svim torkama iz s

- Deljenje se može izraziti pomoću operacija (π, x, -) na sledeći način :

π_y (r) -> q1

π_y ((s x q1) - r) -> q2

q1 - q2 -> q

- Deljenje nije ni komutativna, ni asocijativna operacija

Primer delenja

r

A	B
a1	b1
a1	b2
a3	b1
a4	b2
a4	b3
a5	b1
a5	b2

s

B
b1
b2

r/s

A
a1
a5

Θ-join (Θ-spoj)

- Spoj relacija r i s nad šemama relacija
  R(A1, A2, ..., An) i **S(B1, B2, ..., Bm)**
  je relacija q nad šemom **Q(A1, A2, ..., An, B1, B2, ..., Bm)**
  koja ima po jednu torku za svaku kombinaciju torki,
  jedna iz r i jedna iz s,

  kad god ova kombinacija zadovoljava **uslov spoja**
  (što je glavna razlika u odnosu na Dekartov proizvod)
- Za spoj se koristi oznaka:

  $$\mathbf{r} \bowtie_{\text{<uslov spoja>}} \mathbf{s}$$

  <uslov spoja> je izraz formata: <uslov> ∧ ... ∧ <uslov>

  gde je <uslov> oblika **Ai θ Bj** $\theta \in \{=, <, <=, >, >=, !=\}$

- Torke čiji atributi spoja imaju NULL vrednost **ne pojavljuju** se u rezultatu

Varijacije θ-spoja

- **Equijoin (ekvi spoj)** je spoj gde je uslov spoja oblika

  Ai = Bj

- **Natural join (prirodni spoj)** je ekvi spoj gde je iz rezultata isključen jedan od dva jednaka atributa (Ai ili Bj)
  - Označavaćemo ga sa $*$
  - Takođe se označava sa $\bowtie$, bez uslova
  - Uslov je da oba atributa u uslovu spoja **imaju isto ime**
  - Ukoliko to nije slučaj koristimo operaciju preimenovanja

Primer spoja

radnik

MBR	LIME	LD	*SBR*
2203	MIRA	50 000	10
2817	PERA	40 000	20
2932	MIRA	35 000	10
2995	GOCA	18 000	30
3305	LAZA	60 000	40
3515	JOVAN	20 000	20
3819	VLADA	65 000	30

projekat

PBR	PNAZIV	*SBR*	*PRUK*
100	PC	10	2203
200	HOST	20	3305
300	LAN	30	3819
400	VIPX	40	2817

radnik⋈ radnik.SBR=projekat.SBR projekat

MBR	IME	LD	SBR	PBR	PNAZIV	SBR	PRUK
2203	MIRA	50 000	10	100	PC	10	2203
2932	MIRA	35 000	10	100	PC	10	2203
2817	PERA	40 000	20	200	HOST	20	3305
3515	JOVAN	20 000	20	200	HOST	20	3305
2995	GOCA	18 000	30	300	LAN	30	3819
3819	VLADA	65 000	30	300	LAN	30	3819
3305	LAZA	60 000	40	400	VIPX	40	2817

Primer - prirodni spoj

radnik * projekat
radnik ⋈ projekat

MBR	IME	LD	SBR	PBR	PNAZIV	PRUK
2203	MIRA	50 000	10	100	PC	2203
2932	MIRA	35 000	10	100	PC	2203
2817	PERA	40 000	20	200	HOST	3305
3515	JOVAN	20 000	20	200	HOST	3305
2995	GOCA	18 000	30	300	LAN	3819
3819	VLADA	65 000	30	300	LAN	3819
3305	LAZA	60 000	40	400	VIPX	2817

Outer join (Spoljašnji spoj)

- **Levi (Left) spoj** – rezultat zadržava sve torke iz leve relacije bez obzira na to da li ispunjavaju uslov spoja, ali se u desni deo upisuje NULL vrednost

- **Desni (Right) spoj** - rezultat zadržava sve torke iz desne relacije bez obzira na to da li ispunjavaju uslov spoja, ali se u levi deo upisuje NULL vrednost

- **Puni (Full) spoj** - rezultat zadržava sve torke bez obzira na to da li ispunjavaju uslov spoja, stim što neuparene torke dobijaju NULL vrednost na drugoj strani

Zašto spoljašnji spoj?

- Da bi se dobila unija torki u slučaju kada relacije nisu unijski kompatibilne
- Ova operacija obezbeđuje uniju svih torki relacija koje su parcijalno kompatibilne
  - Samo neki atributi su unijski kompatibilni

Funkcije agregacije

$$\text{<atributi grupisanja>}\,\mathcal{F}\,\text{<lista funkcija>}(r)$$

- gde su:
  - <atributi grupisanja> lista atributa relacije r
  - <lista funkcija> je lista parova (<funkcija><atribut>)
  - <funkcija> može biti
    - SUM
    - AVERAGE
    - MAXIMUM
    - MINIMUM
    - COUNT
- Primer: Za svaki sektor naći broj radnika u tom sektoru i prosečan lični dohodak

$$\text{SBR}\,\mathcal{F}\,\text{COUNT MBR, AVERAGE LD}(\text{radnik})$$

Relacioni model podataka

Pitanja ???

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Primeri upita u relacionoj algebri

Prof.dr Leonid Stoimenov

Primeri korišćenja operacija RA (1)

Date su pojave šema relacija R(MESTO,STABLO), L(NAZIV) i Z(IME).

- Definisati operacije: spoj, unija i selekcija i na primeru datih pojava R i L pokazati kako rade.
- Prikazati međurezultate i konačni rezultat primene sledećih operacija relacione algebre nad datim relacijama:

$$\sigma_{IME=\text{“Bor”}} (\sigma_{STABLO=\text{“Jasen”}} (R) \times Z)$$

$$\pi_{STABLO} (R) - ((L \cup Z) \bowtie_{NAZIV=IME} Z)$$

L
NAZIV
Palma
Javor
Jasen

Z
IME
Bor
Smreka

R	
MESTO	STABLO
Parkić	Palma
Parkić	Javor
Parkić	Jasen
Čair	Palma
Čair	Javor
Čegar	Palma
Čegar	Javor
Čegar	Jasen
Čegar	Bor
G.Polje	Javor

Primeri korišćenja operacija RA (rešenje)

$\sigma_{IME=\text{“Bor”}}\ (\sigma_{STABLO=\text{“Jasen”}}\ (R) \times Z)$

(1) $\sigma_{STABLO=\text{“Jasen”}}\ (R)$

MESTO	STABLO
Parkić	Jasen
Čegar	Jasen

(2) $\sigma_{STABLO=\text{“Jasen”}}\ (R) \times Z$

MESTO	STABLO	IME
Parkić	Jasen	Bor
Parkić	Jasen	Smreka
Čegar	Jasen	Bor
Čegar	Jasen	Smreka

(3) $\sigma_{IME=\text{“Bor”}}\ (\sigma_{STABLO=\text{“Jasen”}}\ (R) \times Z)$

MESTO	STABLO	IME
Parkić	Jasen	Bor
Čegar	Jasen	Bor

L
NAZIV
Palma
Javor
Jasen

Z
IME
Bor
Smreka

R

MESTO	STABLO
Parkić	Palma
Parkić	Javor
Parkić	Jasen
Čair	Palma
Čair	Javor
Čegar	Palma
Čegar	Javor
Čegar	Jasen
Čegar	Bor
G.Polje	Javor

Primeri korišćenja operacija RA (rešenje)

$$\pi_{STABLO}(R) - ((L \cup Z) \bowtie_{NAZIV=IME} Z)$$

(1) $(L \cup Z)$

NAZIV
Palma
Javor
Jasen
Bor
Smreka

(2) $(L \cup Z) \bowtie_{NAZIV=IME} Z$

NAZIV	IME
Bor	Bor
Smreka	Smreka

(3) $\pi_{STABLO}(R)$

STABLO
Palma
Javor
Jasen
Bor

(4) $\pi_{STABLO}(R) - ((L \cup Z) \bowtie_{NAZIV=IME} Z)$

??

L

NAZIV
Palma
Javor
Jasen

Z

IME
Bor
Smreka

R

MESTO	STABLO
Parkić	Palma
Parkić	Javor
Parkić	Jasen
Čair	Palma
Čair	Javor
Čegar	Palma
Čegar	Javor
Čegar	Jasen
Čegar	Bor
G.Polje	Javor

Primeri korišćenja operacija RA (2)

Date su pojave šema relacija R(MESTO,STABLO), L(NAZIV) i Z(IME).

- Definisati operacije: presek, projekcija i spoj i na primeru datih pojava R i L pokazati kako rade.
- Prikazati međurezultate i konačni rezultat primene sledećih operacija relacione algebre nad datim relacijama:

$$Z - (\pi_{STABLO}(R) \cap Z)$$

$$R \,/\, L$$

$$Z \bowtie_{IME=NAZIV} (\sigma_{STABLO=\text{“Bor”}}(R) \times L)$$

L
NAZIV
Palma
Javor
Jasen

Z
IME
Bor
Smreka

R	
MESTO	STABLO
Parkić	Palma
Parkić	Javor
Parkić	Jasen
Čair	Palma
Čair	Javor
Čegar	Palma
Čegar	Javor
Čegar	Jasen
Čegar	Bor
G.Polje	Javor

Primeri korišćenja operacija RA (3)

Date su pojave šema relacija R(MESTO,STABLO), L(NAZIV) i Z(IME).

- Prikazati međurezultate i konačni rezultat primene sledećih operacija relacione algebre nad datim relacijama:

$\pi_{MESTO}(\sigma_{MESTO=\text{“Parkić”}}(R))$

$R - L$

$\pi_{STABLO}(R) \cup Z$

$\pi_{IME}((\sigma_{NAZIV=\text{”Palma”}}((Z \bowtie_{IME=STABLO} R) \times L)$

L
NAZIV
Palma
Javor
Jasen

Z
IME
Bor
Smreka

R	
MESTO	STABLO
Parkić	Palma
Parkić	Javor
Parkić	Jasen
Čair	Palma
Čair	Javor
Čegar	Palma
Čegar	Javor
Čegar	Jasen
Čegar	Bor
G.Polje	Javor

Primeri upita u relacionoj algebri

Šema relacione baze podataka

Student(Indeks, Ime, Adresa, Mentor)

Predmet(ID, Naziv)

Izabrao(Indeks,IDP)

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

Primeri

- Primer 1. Prikazati Indekse i Imena studenata
- Primer 2. Prikazati ID-ove onih predmeta koje je izabrao neki student (izabrani predmeti)
- Primer 3. Podaci o studentima čiji je mentor Petar Petrović

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

Primeri

- Primer 1. Prikazati Indekse i Imena studenata

  $\pi_{Indeks,\ Ime}(Student)$

- Primer 2. Prikazati ID-ove onih predmeta koje je izabrao neki student (izabrani predmeti)

  $\pi_{IDP}(Izabrao)$

- Primer 3. Podaci o studentima čiji je mentor Petar Petrović

  $\sigma_{\ Mentor = \text{“Petar Petrović”}}\ (Student)$

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

Primeri

- Primer 4. Prikazati nazive izabranih predmeta.
- Primer 5. Prikazati ID-ove predmeta koje nije izabrao nijedan student
- *Za vežbu: Studenti koji nisu izabrali nijedan kurs.*

Primeri

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

- Primer 4. Prikazati nazive izabranih predmeta.

$$\pi_{Naziv}(Predmet \bowtie_{ID = IDP} Izabrao)$$

- Primer 5. Prikazati ID-ove predmeta koje nije izabrao nijedan student

$$\pi_{ID}(Predmet) - \pi_{IDP}(Izabrao)$$

- *Za vežbu: Studenti koji nisu izabrali nijedan kurs.*

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

Primeri

- Primer 6. Nazivi predmeta koje nije izabrao nijedan student.

- Primer 7. Imena studenata i nazivi predmeta koje su izabrali.

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

Primeri

- Primer 6. Nazivi predmeta koje nije izabrao nijedan student.

$r \leftarrow \pi_{ID}(Predmet) - \pi_{IDP}(Izabrao)$ - *prethodni primer*

$\pi_{Naziv}(r \bowtie_{r.ID=Predmet.ID} Predmet)$

- Primer 7. Imena studenata i nazivi predmeta koje su izabrali.

$r_1 \leftarrow Predmet \bowtie_{ID=IDP} Izabrao$

$r_2 \leftarrow r_1 \bowtie_{r1.Indeks=Student.Indeks} Student$

$r \leftarrow \pi_{Ime,Naziv}(r_2)$

Primeri

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

- Primer 8. Indeksi studenata koji su izabrali predmet "Baze podataka".

- *Za vežbu:* Indeksi studenata koji su izabrali predmet "Baze podataka" ILI "Strukture podataka".

  Ideja 1: promena uslova kod selekcije

  Ideja 2: rešenje iz prethodnog primera (2) + unija

- *Za vežbu:* Indeksi studenata koji su izabrali oba predmeta:"Baze podataka" i "Strukture podataka".

Student(Indeks, Ime, Adresa, Mentor)
Predmet(ID, Naziv)
Izabrao(Indeks,IDP)

Primeri

- Primer 8. Indeksi studenata koji su izabrali predmet "Baze podataka".

  $r_1 \leftarrow \sigma_{\text{Naziv = "Baze podataka"}}(\text{Predmet})$

  $r_2 \leftarrow r_1 \bowtie_{\text{ID=IDP}} \text{Izabrao}$

  $r \leftarrow \pi_{\text{Indeks}}(r_2)$

- *Za vežbu:* Indeksi studenata koji su izabrali predmet "Baze podataka" ILI "Strukture podataka".

  Ideja 1: promena uslova kod selekcije

  Ideja 2: rešenje iz prethodnog primera (2) + unija

- *Za vežbu:* Indeksi studenata koji su izabrali oba predmeta:"Baze podataka" i "Strukture podataka".

Primeri upita u relacionoj algebri

- Sledeći upiti se odnose na šemu baze podataka PREDUZEĆE koja sadrži sledeće šeme relacija

RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

SEKTOR(SBR,SNAZIV,SLOK,MBR)

PROJEKAT(PBR,PNAZIV,SBR,PRUK)

RADINA(MBR,PBR,SATI)

ČLAN_PORODICE(MBR,IME,DRODJ,SRODSTVO)

Primeri upita u relacionoj algebri (2)

- **Primer 1:** Naći ime, prezime i adresu svih radnika koji rade u sektoru 5

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  SEKTOR(SBR,SNAZIV,SLOK,MBR)

- Rešenje:

- **Primer 2:** Naći imena, prezimena i adrese svih radnika koji rade u sektoru 'PROIZVODNJA' u Nišu

Primeri upita u relacionoj algebri (2)

- **Primer 1:** Naći ime, prezime i adresu svih radnika koji rade u sektoru 5

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  SEKTOR(SBR,SNAZIV,SLOK,MBR)

- Rešenje:

$$r \leftarrow \pi_{LIME,\ PREZIME,ADRESA}\ \sigma_{SBR=5}\ (radnik)$$

- **Primer 2:** Naći imena, prezimena i adrese svih radnika koji rade u sektoru 'PROIZVODNJA' u Nišu

$$r1 \leftarrow \sigma_{SNAZIV='PROIZVODNJA'\ AND\ SLOK='NIŠ'}(sektor)$$

$$r2 \leftarrow r1 \bowtie_{r1.SBR=radnik.SBR} radnik$$

$$r \leftarrow \pi_{MBR,LIME,LD}(r2)$$

Primeri upita u relacionoj algebri (3)

- **Primer 3:** Naći ime, prezime i adresu svih radnika koji rade u sektoru 'RAZVOJ'

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  SEKTOR(SBR,SNAZIV,SLOK,MBR)

- Rešenje

Primeri upita u relacionoj algebri (3)

- **Primer 3:** Naći ime, prezime i adresu svih radnika koji rade u sektoru 'RAZVOJ'

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  SEKTOR(SBR,SNAZIV,SLOK,MBR)

- Rešenje

  $r1 \leftarrow \sigma_{SNAZIV='RAZVOJ'}(sektor)$

  $r2 \leftarrow radnik \bowtie_{radnik.SBR=r1.SBR} r1$

  $r \leftarrow \pi_{LIME,\ PREZIME,ADRESA}(r2)$

- Drugi oblik upita

$r \leftarrow \pi_{LIME,\ PREZIME,ADRESA}(radnik \bowtie_{radnik.SBR=sektor.SBR} (\sigma_{SNAZIV='RAZVOJ'}(sektor)))$

Primeri upita u relacionoj algebri (4)

- **Primer 4:** Naći imena radnika koji rade na svim projektima koje nadgleda sektor broj 5

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  PROJEKAT(PBR,PNAZIV,SBR,PRUK)

  RADINA(MBR,PBR,SATI)

- Rešenje:

Primeri upita u relacionoj algebri (4)

- **Primer 4:** Naći imena radnika koji rade na svim projektima koje nadgleda sektor broj 5

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  PROJEKAT(PBR,PNAZIV,SBR,PRUK)

  RADINA(MBR,PBR,SATI)

- Rešenje:

  $r1 \leftarrow \sigma_{SBR=5}(projekat)$ projekti koje nadgleda sektor 5

  $r2 \leftarrow r1 \bowtie_{r1.PBR = radina.PBR} radina$

  $r3 \leftarrow r2 \bowtie_{r2.MBR = radnik.MBR} radnik$ radnici koji rade na projektima sek 5

  $r4 \leftarrow \pi_{LIME,PREZIME}\ (r3)$

Primeri upita u relacionoj algebri (5)

- **Primer 5:** Naći imena radnika koji imaju 2 i više izdržavanih lica

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  ČLAN_PORODICE(MBR,IME,DRODJ,SRODSTVO)

- Rešenje:

Primeri upita u relacionoj algebri (5)

- **Primer 5:** Naći imena radnika koji imaju 2 i više izdržavanih lica

  RADNIK(MBR,LIME,PREZIME,DRODJ,ADRESA,LD,SBR,SEF)

  ČLAN_PORODICE(MBR,IME,DRODJ,SRODSTVO)

- Rešenje:

  $r1(MBR,BROJ\_IZL) \leftarrow {}_{MBR}\mathcal{F}_{\mathbf{COUNT}\ IME}(ČLAN\_PORODICE)$

  $r2 \leftarrow \sigma_{BROJ\_IZL \geq 2}(r1)$

  $r \leftarrow \pi_{LIME,PREZIME}(r2 \bowtie_{r2.MBR = radnik.MBR} radnik)$

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Funkcionalne zavisnosti

Prof.dr Leonid Stoimenov

Pregled

- Uvod
- Izvođenje funkcionalnih zavisnosti
- Pravila za izvođenje FZ
- Zatvarač skupa FZ
- Zatvarač skupa atributa

Uvod

- Podsetnik: Šema relacije R(A;$\mathbb{F}$)
  - A skup atributa,
  - $\mathbb{F}$ skup **funkcionalnih zavisnosti** kojima se specificiraju ograničenja
- Funkcionalna zavisnost (FZ) je prvi tip ograničenja definisan u okviru relacionog modela podataka

- FZ je izraz oblika **f: X→Y**
  - **f** predstavlja **naziv** funkcionalne zavisnosti
  - **X i Y** su skupovi atributa

- Koristi se skraćena notacija **X→Y**:
  - da **Y funkcionalno zavisi od X** ili
  - da **X funkcionalno određuje Y**
  -

- Primer: Grad, Adresa → PoštanskiKod

Definicija FZ

- Neka je **r** relacija nad šemom relacije **R(A;C)** i neka su **X** i **Y** podskupovi skupa atributa **A={A1, A2, ..., An}**

- U šemi relacije R postoji funkcionalna zavisnost **f: X → Y** ako i samo ako je svakom elementu iz **dom(X)** pridružen najviše jedan element iz **dom(Y)**
  - na osnovu vrednosti atributa X može odrediti odgovarajuća vrednost atributa Y
- Posmatrano na nivou torki **X→Y** znači da za bilo koje dve torke **t1** i **t2** iz **r** važi implikacija

  t1[X] = t2[X] => t1[Y] = t2[Y]
  - ako je **t1[X] = t2[X],** tada mora biti **t1[Y] = t2[Y]**
  - vrednost **Y** komponente torke u **r** zavisi od vrednosti **X** komponente ili da vrednost **X** komponente funkcionalno određuje vrednost **Y** komponente

Primer 1

- Zadata je šema relacije R(R;F)
  POSETILAC({IME,ADRESA};{IME→ADRESA})
- Funkcionalna zavisnost **IME→ADRESA** može se tumačiti na sledeći način:
  - Vrednost atributa **IME** na jedinstven način određuje vrednost atributa **ADRESA**
  - Ako u instanci relacije **posetilac(POSETILAC)** postoji torka **<GAGA, Nišavska 14>**,

  tada kad god se u nekoj instanci relacije **posetilac(POSETILAC)** u koloni **IME** pojavi vrednost **GAGA**, tada u koloni **ADRESA** ove torke mora biti vrednost **Nišavska 14**

Primer 2

- U šemi relacije
  RADPROJ(MBR,RIME,SATI,PBR,PNAZIV,PLOK,SBR,PRUK)
  iz semantike atributa znamo da postoje sledeće funkcionalne zavisnosti:
  1. **MBR → RIME**
  2. **MBR → SBR**
  3. **PBR → {PNAZIV, PLOK}**
  4. **{MBR,PBR} → SATI**
  5. **PBR → PRUK**
- Šta znače ove FZ?
  1. Matični broj radnika jednoznačno određuje ime radnika
  2. Matični broj radnika jednoznačno određuje broj sektora u kome radnik radi
  3. Broj projekta jednoznačno određuje naziv i lokaciju projekta
  4. Matični broj radnika i broj projekta jednoznačno određuju vreme (SATI) angažovanja radnika na projektu
  5. Broj projekta jednoznačno određuje rukovodioca projekta

Primer 2 (nast.)

RADPROJ(MBR,RIME,SATI,PBR,PNAZIV,PLOK,SBR,PRUK)

1. **MBR → RIME**
2. **MBR → SBR**
3. **PBR → {PNAZIV, PLOK}**
4. **{MBR,PBR} → SATI**
5. **PBR → PRUK**

- Semantika ovih ograničenja je:
  1. Svaki radnik ima jedinstven matični broj (**MBR**)
  2. Radnik može raditi samo u jednom sektoru (**SBR**)
  3. Svaki projekat ima jedinstven broj projekta (**PBR**)
  4. Radnik može biti angažovan na više projekata određeni broj sati
  5. Svaki projekat ima samo jednog rukovodioca

Izvođenje funkcionalnih zavisnosti

- Projektanti baze podataka specificiraju FZ koje su **semantički očigledne**
- U realnom sistemu nije moguće specificirati sve moguće FZ za datu situaciju
- U svim relacijama nad šemom relacije R u kojoj važi $\mathbb{F}$ postoje i mnoge druge FZ
  - Ove FZ se mogu **izvesti** iz FZ koje postoje u $\mathbb{F}$
  - Potreban je **alat za izvođenje** novih FZ iz postojećih
- Da bismo uveli takav alat potrebno je upoznati sledeće koncepte:
  - zatvarač skupa FZ
  - zatvarač skupa atributa
  - pravila izvođenja FZ

Zatvarač skupa funkcionalnih zavisnosti

Definicija:

Zatvarač ili **zatvaranje** skupa funkcionalnih zavisnosti F**, u oznaci** F^+, je skup funkcionalnih zavisnosti sadržan u F i skup svih funkcionalnih zavisnosti koje se mogu izvesti iz F

- Za potrebe izvođenja novih FZ iz postojećih definišu se **pravila** ili **aksiome izvođenja** funkcionalnih zavisnosti
-
- Kardinalni broj zatvarača je vrlo veliki čak i za mali broj FZ u $\mathbb{F}$ i za mali broj atributa

Pravila (aksiome) izvođenja funkcionalnih zavisnosti

P1 (**Refleksivnost**) : Ako je $Y \subseteq X$, tada važi $X \rightarrow Y$

P2 (**Proširenje**) : Ako je $X \rightarrow Y$ i $Z \subseteq W$, tada važi $XW \rightarrow YZ$

P3 (**Tranzitivnost**) : Ako je $X \rightarrow Y$ i $Y \rightarrow Z$, tada važi $X \rightarrow Z$

P4 (**Dekompozicija**) : Ako je $X \rightarrow YZ$, tada važi $X \rightarrow Y$ i $X \rightarrow Z$

P5 (**Unija**) : Ako $X \rightarrow Y$ i $X \rightarrow Z$, tada važi $X \rightarrow YZ$

P6 (**Pseudotranzitivnost**): Ako $X \rightarrow Y$ i $YW \rightarrow Z$, tada važi $XW \rightarrow Z$

- Napomena: X,Y,W i Z podskupovi skupa atributa A šeme relacije $\mathbb{R}(A;\mathbb{F})$ i gde XY označava uniju skupova X i Y

- Pravila P1-P3 se nazivaju **Armstrongove aksiome**

Pravila (aksiome) izvođenja funkcionalnih zavisnosti

- Pravilo P1 dovodi do definisanja tzv. **trivijalnih FZ**
  - To su zavisnosti za koje važi da je desna strana FZ podskup leve
- Ako je Z=W u pravilu P2, tada ovo pravilo postaje
  - **Ako je X→Y, tada važi XZ→YZ**
- Ako je W prazan skup u Pravilu 6, tada ono prelazi u tranzitivnost
- Na osnovu pravila unije (P5) sledi da se svaki skup funkcionalnih zavisnosti F može zameniti ekvivalentnim skupom G u kojem se na desnoj strani svake FZ nalazi samo 1 atribut

- Iscrpnom primenom pravila izvođenja P1-P6 na skup funkcionalnih zavisnosti F dobija se skup funkcionalnih zavisnosti F^+

Algoritam za nalaženje F^+

Ulaz : Skup atributa A={A1, A2,..., An} i skup funkcionalnih zavisnosti F={f_1, f_2,..., f_m} šeme relacije $\mathbb{R}(A;\mathbb{F})$

Izlaz : $\mathbb{F}^+$ (funkcionalno zatvaranje skupa $\mathbb{F}$)

1. $\mathbb{F}^+ := \mathbb{F}$;
2. repeat
3. za svaku funkcionalnu zavisnost *f* u $\mathbb{F}^+$
4. primeniti pravilo refleksivnosti i pravilo proširenja na *f ;*
5. dodati rezultujuće funkcionalne zavisnosti u $\mathbb{F}^+$;
6. za svaki par funkcionalnih zavisnost f_i *i* f_j u $\mathbb{F}^+$
7. ako se f_i *i* f_j mogu kombinovati korišćenjem tranzitivnosti
8. dodati rezultujuću funkcionalnu zavisnost u $\mathbb{F}^+$;
9. until (ne menja se $\mathbb{F}^+$)

Zatvarač skupa atributa

- **Definicija:**

  Zatvarač skupa atributa X (gde je $X \subseteq A$) u odnosu na skup funkcionalnih zavisnosti $\mathbb{F}$ je skup atributa Y koje funkcionalno određuje X:

  $$X^+_{\mathbb{F}} = \{Y \mid X \rightarrow Y \in \mathbb{F}^+\}$$

Algoritam za nalaženje X^+

Ulaz : Konačan skup atributa A šeme relacija $\mathbb{R}(A;\mathbb{F})$,skup funkcionalnih zavisnosti $\mathbb{F}$ nad A i podskup X skupa A

Izlaz : X^+ (zatvaranje X u odnosu na $\mathbb{F}$).

1. $X^{+}:=X$;
2. **repeat**
3. staro$X^+:= X^+$;
4. **za svaku** FZ $Y \rightarrow Z$ u $\mathbb{F}$
5. **ako** $Y \subseteq X^+$ **tada** $X^+ := X^+ \cup Z$;
6. **until** (X^+ = staroX^+)

Primer: Zatvarač skupa atributa

- U šemi relacije

  RADPROJ = (**MBR**, RIME, SATI, **PBR**, PNAZIV, PLOK)

  postoje sledeće funkcionalne zavisnosti :

  f1: {MBR, PBR} -> SATI

  f2: MBR -> RIME

  f3: PBR -> {PNAZIV, PLOK}

- Prema gornjem algoritmu dobija se :

  X1 = {MBR}, $X1^+$ = {MBR, RIME}

  X2 = {PBR}, $X2^+$ = {PBR, PNAZIV, PLOK}

  X3 = {MBR, PBR}, $X3^+$ = {MBR, PBR, SATI, RIME, PNAZIV, PLOK}

Primena zatvarača skupa

- Ima više primena:
  1. Za proveru da li je X super ključ relacije
     - Proveravamo da li X^+ sadrži sve atribute relacije
  2. Za proveru da li funkcionalna zavisnost $X \rightarrow Y$ važi u $\mathbb{F}$
     - Proveravamo da li je $Y \subseteq X^+$
  3. Kao alternativni način za izračunavanje $\mathbb{F}^+$
     - Za svako $X \subseteq A$ šeme relacije $R(A;\mathbb{F})$ nalazimo X^+ i za svako $Y \subseteq X^+$ dodajemo u $\mathbb{F}^+$ funkcionalnu zavisnost $X \rightarrow Y$

Ključ relacije

- **Definicija:**
  - Neka je $\mathbb{R}(\{A1, A2, ..., An\}; \mathbb{F})$ šema relacije i neka je $X \subseteq \{A1, A2, ..., An\}$
  - X je ključ šeme relacije $\mathbb{R}$ ako i samo ako ima sledeća svojstva:

    S1. Svojstvo identifikacije: X funkcionalno određuje sve atribute šeme relacije, tj. $\{X \rightarrow Ai \mid i=1, 2, ..., n\}$

    S2. Svojstvo minimalnosti: Nijedan pravi podskup od X nema tu osobinu,

    tj. za $X'' \subset X$ važi $\{X''\text{-/->} Ai \mid i=1,2,...,n\}$
- Iz definicije ključa relacije sledi da svaka relacija ima barem jedan ključ
  - to je A = {A1, A2, ..., An}

Super ključ relacije

- Ako za neko $X \subseteq \{A1, A2, ..., An\}$ važi samo svojstvo S1 iz definicije ključa, tada X predstavlja super ključ relacije R
- Treba uočiti da super ključ sadrži ključ relacije

- Može se reći da je ključ relacije njen minimalni i stoga neredundantni super ključ

Ključevi kandidati

- Ako skup ključeva K šeme relacije R(R;$\mathbb{F}$) ima više od jednog elementa, tada su svi oni **ključevi kandidati**
- Jedan od ključeva kandidata se bira za **primarni ključ** šeme relacije, dok se ostali nazivaju **sekundarnim ključevima**
- Primer:
  - U šemi relacije RADNIK(**MBR**, LIME, LD, SBR) , {MBR} je jedini ključ kandidat, tako da je on istovremeno i primarni ključ
  - U šemi relacije RADNIK(**MBR**,JMB, LIME, LD, SBR) , ključevi kandidati su {MBR,JMB} ali je MBR izabran za primarni ključ

Primarni ključ, primarni i sporedni atributi

- **Primarni ključ** je ključ koji je izabran da jedinstveno identifikuje entitete predstavljene relacijom. Stoga nijedan od njegovih atributa nesme nikada imati nedefinisanu **(null)** vrednost
- Svaka šema relacije mora imati primarni ključ
- Atribut A_i šeme relacije R je **primarni (ključni) atribut** ako je član primarnog (bilo kog) ključa relacije zadate šemom R
- Ostali atributi $A_j \neq A_i$ šeme relacije R su **neprimarni (sporedni, neključni)**
- Primer:
  - U šemi relacije RADPROJ atributi MBR i PBR su primarni (ključni), dok su svi ostali neprimarni (sporedni, neključni)

Potpuna i parcijalna funkcionalna zavisnost

- Neka su X i Y skupovi atributa
- Funkcionalna zavisnost $X \rightarrow Y$ je **potpuna** ako izbacivanjem bilo kog atributa A iz X ova zavisnost više ne postoji

- Ako postoji atribut A koji se može izbaciti iz X, a da se funkcionalna zavisnost $X \rightarrow Y$ zadrži, onda je ova funkcionalna zavisnost **parcijalna**

- **Primer:**
  - U šemi relacije STUDENT(**BRIND**, JMB, IME GODINA) funkcionalna zavisnost {BRIND, JMB} $\rightarrow$ IME nije potpuna budući da važi funkcionalna zavisnost BRIND $\rightarrow$ IME

Tranzitivna funkcionalna zavisnost

- Neka su X i Y skupovi atributa šeme relacije R
- Funkcionalna zavisnost X→Y u šemi relacije R je **tranzitivna** zavisnost ako postoji skup Z, koji nije podskup nijednog ključa relacije R, za koji važe FZ X→Z, Z→Y
  - U odnosu na primarni ključ to znači da X treba da bude primarni ključ, a Z može biti ili kandidat za ključ ili njegov deo
- **Primer:**
  - U šemi relacije RADSEK(RIME, **RMBR,** DATRODJ, ADRESA, SBR, SNAZIV, SRUK) postoji tranzitivna funkcionalna zavisnost RMBR→SBR i SBR→SRUK, a SBR nije podskup nijednog ključa relacije

Zavisnost ključa

- Ako je K jedan ključ šeme relacije (A;$\mathbb{F}$), tada važi funkcionalna zavisnost K→A
- Ova se funkcionalna zavisnost naziva **zavisnošću ključa**
- Primer:
  - Ako je zadata šema relacije R({A,B,C,D,E};{A→BC,CD→E,B→D,E →A}) i ako je A ključ ove šeme relacije, tada u ovoj relaciji postoji sledeća zavisnost ključa: A→A,B,C,D,E

Algoritam za nalaženje jednog ključa

Ulaz: Šema relacije (R;$\mathbb{F}$); skup atributa R i skup funkcionalnih zavisnosti $\mathbb{F}$ nad skupom atributa R

Izlaz: Ključ K šeme relacije (R;$\mathbb{F}$)

Metoda: Algoritam nalazi samo jedan ključ K za zadati skup atributa R i zadati skup FZ $\mathbb{F}$ nad skupom R

1. K := R;
2. Za **svaki** atribut X u K
   izračunati $(K\text{-}X)^+$ u odnosu na $\mathbb{F}$;
   ako $(K\text{-}X)^+$ sadrži sve atribute u R
   tada K := K-{X}

Primer

Zadata je šema relacije $(R;\mathbb{F})$, gde je $R=(A,B,C,D,E)$ i $\mathbb{F}=(A\rightarrow BC, CD\rightarrow E, B\rightarrow D, E\rightarrow A)$

Naći ključ ove relacije

1. $K=\{A,B,C,D,E\}$
2. Iz K izbacujemo atribute redom

   Iz K izbacujemo atribut A

   Izračunavamo $\{K-A\}^+_F$, tj. $\{K-A\}^+_F=\{B,C,D,E,A\}$;
   $\{K-A\}^+$ sadrži sve atribute iz R,
   pa je K:= K-A :=$\{B,C,D,E\}$;

   Iz K=$\{B,C,D,E\}$ izbacujemo B

   Izračunavamo $\{K-B\}^+_F$, tj. $\{K-B\}^+_F=\{C,D,E,A,B\}$;
   $\{K-B\}^+$ sadrži sve atribute iz R,
   K :=K-B, tj. K=$\{C,D,E\}$;

Primer

Iz K izbacujemo atribut C

Izračunavamo $\{K-C\}^+_F$, tj. $\{K-C\}^+_F = \{D,E,A,B,C\}$;

$\{K-C\}^+$ sadrži sve atribute iz R,

K :=K-C, tj. K={D,E};

Iz K izbacujemo atribut D

Izračunavamo $\{K-D\}^+_F$, tj. $\{K-D\}^+_F = \{E,A,B,C,D\}$;

$\{K-C\}^+$ sadrži sve atribute iz R, pa

K:=K-D, tj. K ={E};

Iz K izbacujemo atribut E

Izračunavamo $\{K-E\}^+_F$, tj. $\{K-E\}^+_F = \emptyset$;

$\{K-E\}^+$ ne sadrži sve atribute iz R, pa K ostaje nepromenjeno, tj. K ={E};

3. KRAJ algoritma.

Ključ zadate šeme relacione je {E}

Algoritam za nalaženje svih ključeva šeme relacije

Ulaz: Šema relacije $(R;\mathbb{F})$

Izlaz: Skup ključeva K šeme relacije $(R;\mathbb{F})$

1. $K := \emptyset$;
2. **do** traži_ključ $(\forall X \subseteq R)$ *// za **svaki podskup** skupa atributa R*
   if $R=X^+_F$ **then** *// svojstvo identifikacije*
   do redukcija $(\forall A \in X)$
   if $A \in \{X-A\}^+$ **then** $X=\{X-A\}$ **endif** *// svojstvo minimalnosti*
   enddo redukcija
   $K=K \cup X$
   endIf
3. **enddo** traži_ključ

Primer

- Zadata je šema relacije $(R;\mathbb{F})$,
  gde je $R=(A,B,C,D,E)$ i
  $\mathbb{F}=(A\rightarrow BC, CD\rightarrow E, B\rightarrow D, E\rightarrow A)$
  Naći sve ključeve ove relacije

Ulaz: $R=(A,B,C,D,E)$ i $\mathbb{F}=(A\rightarrow BC, CD\rightarrow E, B\rightarrow D, E\rightarrow A)$

1. $K=\varnothing$
2. Ispitujemo da li svojstvo ključa ima svaki podskup skupa atributa R, tj.{A}, {B}, {C},{D},{E}, {A,B},{A,C}, {A,D}, {A,E}, {B,C}, {B,D}, {B,E}, {C,D}, {C,E}, {D,E},{A,B,C},{A,B,D}, {A,B,E},{A,C,D} {A,C,E},{A,D,E} {B,C,D},{B,C,E},{B,D,E}, {C,D,E}, {A,B,C,D}, {A,B,C,E},{A,B,D,E},{A,C,D,E},{B,C,D,E}
3. KRAJ algoritma.

Izlaz: K={A, E,BC,CD}

Funkcionalne zavisnosti

Pitanja ???

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Analiza šeme relacione BP

Prof.dr Leonid Stoimenov

Pregled

- Uvod
- Anomalije kod ažuriranja
- Dekompozicija

Šta je analiza šeme relacione baze podataka?

- Proces određivanja kvaliteta projektovanih šema relacija
- U teoriji baza podataka definisani su formalizmi za analizu kvaliteta šeme relacije i prevođenje šeme relacije u ekvivalentnu bolju šemu
- Postoje dva takva formalizma:
  - Logičke zavisnosti i
  - Normalne forme

Kvalitet projektovane šeme relacije

- Kvalitet se može posmatrati sa
  - logičkog (sa strane korisnika)
  - fizičkog nivoa (sa strane sistema)

- Na **logičkom nivou** od šeme se zahteva da bude jasna i laka za razumevanje

- Na **fizičkom nivou** od šeme se zahteva da obezbedi ažurni skup podataka uz minimalan utrošak računarskih resursa

Kvalitet projektovane šeme relacije

- Za merenje kvaliteta šeme relacije koriste se sledeći faktori:
  - semantika atributa relacije
  - redukovanje redundantnih vrednosti u torkama
  - redukovanje NULL vrednosti u torkama
  - isključivanje lažnih torki

Logičke zavisnosti

- Zavisnosti izražavaju činjenicu da jedan objekat zavisi od drugih objekata
- U relacionom modelu podataka objekti su atributi i torke
- U šemi relacije **R**(R;$\mathbb{F}$) nad skupom atributa R=(A1, A2, ..., An) mogu postojati različita ograničenja zavisnosti kao posledica osobina i pravila funkcionisanja realnog sistema koji se ovom relacijom modelira
  - najčešće se susreću **funkcionalne** i **višeznačne** zavisnosti

Anomalije pri obradi relacija (1)

- Nepoželjno ponašanje relacija pri izvođenju operacija ažuriranja naziva se anomalijom ažuriranja
- Na osnovu operacije koja indukuje anomalije ažuriranja razlikujemo:
  - anomalije unosa
  - anomalije brisanja
  - anomalije modifikacije

Anomalije pri obradi relacija (2)

- Anomalije ažuriranja ćemo razmotriti na primeru relacije radsek nad šemom relacije:

  RADSEK(RMBR, RIME, RLD, SBR, SNAZIV, *SRUK*)

- Ova relacija sadrži atribute dva entiteta: RADNIK i SEKTOR
  - Entitet RADNIK je opisan atributima {RMBR, RIME, RLD}, a entitet SEKTOR atributima {SBR, SNAZIV, SRUK}
  - U šemi relacije **RADSEK** postoje sledeće funkcionalne zavisnosti nad atributima:

    f1: RMBR $\rightarrow$ {RIME, RLD, SBR}

    f2: SBR $\rightarrow$ {SNAZIV, SRUK}
- Pri izvođenju operacija nad ovom relacijom javljaju se sve tri vrste anomalija

Primer loše projektovane relacije

RADSEK(RMBR, RIME, RLD, SBR, SNAZIV, *SRUK*)

RADSEK

RMBR	**RIME**	**RLD**	**SBR**	**SNAZIV**	***SRUK***
111	Laza	100000	10	Projektni biro	333
222	Mara	60000	10	Projektni biro	333
333	Nada	150000	10	Projektni biro	333
444	Djura	50000	20	Kontrola	null

Loše projektovana relacija

RADSEK

RMBR	RIME	RLD	SBR	SNAZIV	*SRUK*
111	Laza	100000	10	Projektni biro	333
222	Mara	60000	10	Projektni biro	333
333	Nada	150000	10	Projektni biro	333
444	Djura	50000	20	Kontrola	null

Dobro projektovane relacije

RADNIK

RMBR	RIME	RLD	*SBR*
111	Laza	10000	10
222	Mara	60000	10
333	Nada	150000	10
444	Djura	50000	20

SEKTOR

SBR	SNAZIV	*SRUK*
10	Projektni biro	333
20	Kontrola	null

Anomalije unosa (1)

Relacija **RADSEK(RMBR**, RIME, RLD, SBR, SNAZIV, *SRUK*)
poseduje anomalije unosa koje se ogledaju u sledećem :

- U relaciju **RADSEK** se ne mogu uneti podaci o **sektoru** ukoliko u taj sektor nije raspoređen nijedan radnik.
- Ukoliko bi se takav unos dozvolio tada bi atributi RMBR,RIME i RLD imali nedefinisanu (NULL) vrednost, što nije dozvoljeno budući da je RMBR primarni ključ relacije

Anomalije unosa (2)

- Relacija **RADSEK(RMBR**, RIME, RLD, SBR, SNAZIV, *SRUK*) **poseduje anomalije unosa** koje se ogledaju u sledećem :
  - Pri unosu podataka o novom **radniku** potrebno je, uz konkretne vrednosti za sve atribute entiteta RADNIK, uneti konkretne vrednosti za sve atribute entiteta SEKTOR.
  - Pri svakom unosu moramo voditi računa da podaci o sektoru budu konzistentni sa već unetim podacima o tom sektoru.
  - Ponavljanje podataka o sektoru u torkama svih radnika tog sektora govori o prisustvu redundance u relaciji.

Anomalije unosa (3)

RADSEK

RMBR	RIME	RLD	SBR	SNAZIV	*SRUK*
111	Laza	100000	10	Projektni biro	333
222	Mara	60000	10	Projektni biro	333
333	Nada	150000	10	Projektni biro	333
444	Djura	50000	20	Kontrola	Null
?	?	?	30	Razvoj	Null
555	Pera	20000	30	Razvoj	Null

Anomalije unosa (4)

- Opisane anomalije su posledica **tranzitivne funkcionalne zavisnosti** među atributima relacije:

  RMBR → {RIME, RLD, SBR},

  SBR → {SNAZIV, SRUK},

  SBR nije podskup ključa relacije

- Anomalije se mogu otkloniti **dekompozicijom** šeme relacije **RADSEK** tako da se svaka funkcionalna zavisnost predstavi posebnom relacijom
- Rezultat ove dekompozicije su sledeće šeme relacija:

  RADNIK(**RMBR**, RIME, RLD, *SBR*)

  SEKTOR(**SBR**, SNAZIV, *SRUK*)

- Može se uočiti da svaka nova šema relacije sadrži samo atribute jednog entiteta

Anomalije brisanja

- Relacija **RADSEK**(**RMBR**, RIME, RLD, SBR, SNAZIV, *SRUK*) poseduje anomalije brisanja koje se ogledaju u sledećem:
  - Ako iz relacije **RADSEK** obrišemo sve radnike koji rade u nekom sektoru, tada ćemo obrisati i podatke o tom sektoru. Na taj način se iz baze podataka gubi informacija o sektorima koji trenutno nemaju raspoređene radnike.

RADSEK

RMBR	RIME	RLD	SBR	SNAZIV	*SRUK*
111	Laza	100000	10	Projektni biro	333
222	Mara	60000	10	Projektni biro	333
333	Nada	150000	10	Projektni biro	333
444	Djura	50000	20	Kontrola	null

Anomalija modifikacije

- Relacija **RADSEK(RMBR**, RIME, RLD, SBR, SNAZIV, *SRUK*) poseduje anomalije modifikacije koje se ogledaju u sledećem:
  - Pri promeni vrednosti nekog od atributa entiteta **SEKTOR** (na primer, SRUK) potrebno je da se izmeni vrednost atributa SRUK u torkama svih radnika koji rade u tom sektoru.
  - Dok proces izmene traje postoji nekozistentnost u podacima. U našem primeru svi radnici koji rade u sektoru čiji atribut SRUK menjamo neće imati istu vrednost za SRUK. Ova pojava se naziva anomalijom modifikacije, a posledica je ranije pomenute tranzitivne zavisnosti među atributima relacije.

Rezime o anomalijama ažuriranja

- Razmatrane anomalije ažuriranja su posledica izvesnih **neželjenih struktura funkcionalnih zavisnosti** u šemi relacije
- Eliminisanjem svih nepoželjnih struktura funkcionalnih zavisnosti eliminišu se i anomalije ažuriranja
- Za šeme relacija koje su formirane pod ovim restrikcijama kaže se da su u **normalnoj formi**, a postupak kojim se šema relacije prevodi u neku normalnu formu naziva se **normalizacija**
- Pri normalizaciji se koristi tehnika **dekompozicije**
  - Dekompozicija šeme relacije nije proizvoljna. Po pravilu se zasniva na određenoj strukturi funkcionalnih zavisnosti među atributima relacije
- Dekompozicijom se svaka funkcionalna zavisnost među atributima šeme relacije izoluje u posebnu šemu relacije

Dekompozicija relacija

- **Dekompozicija** je proces zamene šeme relacije **R**(R;$\mathbb{F}$) skupom šema relacija D={**R**1(R1;$\mathbb{F}$1), ..., **R**m(Rm;$\mathbb{F}$m)} pri čemu moraju biti ispunjeni sledeći uslovi:
  1. **Očuvanje atributa:** Nijedan atribut nesme biti izgubljen, što znači da se svaki atribut iz **R** mora naći bar u jednoj šemi **R**i dekompozicije D
  2. **Spoj-bez-gubitaka (neaditivni spoj) -** osigurava da se pri prirodnom spoju relacija dobijenih dekomponovanjem ne jave lažne torke. Svojstvo spoj-bez-gubitaka odnosi se na gubitak informacije, a ne na gubitak torki
  3. **Očuvanje zavisnosti.** Dekompozicija obezbeđuje očuvanje zavisnosti ako su sve funkcionalne zavisnosti šeme (R;$\mathbb{F}$) zadržane u D
- **Dekompozicija je dobra** ako je skup šema relacija D ekvivalentan šemi **R** i ako su isključene anomalije

Svojstvo spoj-bez-gubitaka za binarnu dekompoziciju

- Dekompozicija D={**R1,R2**} šeme relacije **R(**R;$\mathbb{F}$**)** ima svojstvo spoj-bez-gubitaka u odnosu na skup funkcionalnih zavisnosti $\mathbb{F}$ u **R** ako i samo ako važi:

  (Uslov 1) $R1 \cap R2 \rightarrow R1 \in \mathbb{F}^+$ ili

  (Uslov 2) $R1 \cap R2 \rightarrow R2 \in \mathbb{F}^+$, gde je:

  - $R1 \cap R2 = Z$ skup zajedničkih atributa obe komponente
  - (Uslov 1) Z je super ključ komponente R1
  - (Uslov 2) Z je super ključ komponente R2
- Ako je $R1 \cap R2$ super ključ šeme relacije **R1** ili **R2**, tada binarna dekompozicija ima svojstvo spoj-bez-gubitaka
  - Za proveru da li je $R1 \cap R2$ super ključ treba koristiti algoritam za nalaženje zatvarača skupa atributa

Analiza šeme

Pitanja ???

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Normalne forme i normalizacija

Prof.dr Leonid Stoimenov

Pregled

- Uvod
- Normalne forme
- Normalizacija
  - 1NF
  - 2NF
  - 3NF
  - BCNF

Uvod

- Nakon projektovanja BP sledi analiza projektovane šeme i donošenje odluke da li je projektovana šema dobra ili je treba dalje doterivati
- Da bismo doneli takvu odluku potrebno je da shvatimo koje probleme, ako ih ima, može da proizvede ta šema
- Kao vodič za to služe **normalne forme** koje nam obezbeđuju informaciju o tome koji se problemi neće javiti ako je šema u nekoj normalnoj formi
- U teoriji relacionog modela podataka je predloženo više normalnih formi

Svrha normalizacije

Proces normalizacije je formalna metoda koja se oslanja na funkcionalnim zavisnostima, odnosno na zavisnostima između ključeva (primarnog i kandidata) i ostalih atributa relacije.

Normalizacija je postupak kojim se dobija skup relacija sa željenim osobinama tj ograničenjima definisanim normalnom formom, kako bi se izbegle anomalije ažuriranja.

Proces Normalizacije

Normalizacija se izvršava u više koraka.

- Svaki korak odgovara jednoj normalnoj formi sa željenim osobinama definisanim normalnom formom
- Svaka sledeća normalna forma je stroža i uvodi više restrikcija
- Svaka sledeća normalna forma smanjuje mogućnost pojave anomalija

Kod relacionog modela, jedino je prva normalna forma (1NF) je kritična kod kreiranja relacija, sve ostale NF su opcione.

Funkcionalne zavisnosti i normalizacija

Osnovne karakteristike FZ kod normalizacije

- Definišu jedan-na-jedan relacije između atributa sa leve- i atributa sa desne- strane FZ;
- Postoje sve vreme;
- Netrivijalne su.

Normalne forme

1. Prva normalna forma (1NF)
2. Druga normalna forma (2NF)
3. Treća normalna forma (3NF)
4. Boyce-Coddova normalna forma (BCNF)

definišu se na osnovu funkcionalnih zavisnosti (FZ) među atributima

5. Četvrta normalna forma (4NF)
6. Peta normalna forma (5NF) ili normalna forma projekcija-spoj (PJNF)
7. Normalna forma domen-ključ (DKNF)

4NF se zasniva na višeznačnim zavisnostima (VVZ)
5NF se zasniva na zavisnosti spoja (ZS)
DKNF uzima u obzir sve moguće zavisnosti i ograničenja

PRVA NORMALNA FORMA (1NF)

- **Definicija:**

Šema relacije **R**(R;$\mathbb{F}$) je u prvoj normalnoj formi (1NF) ako i samo ako domeni relacije R sadrže samo proste (atomične) vrednosti

  - Proste vrednosti su vrednosti koje se ne mogu dalje deliti ili ako u konkretnoj situaciji nisu rastavljene na komponente

PRVA NORMALNA FORMA (1NF)

- U 1NF **nisu dozvoljeni** viševrednosni i kompozitni atributi i njihove kombinacije,
- Praktično, nisu dopuštene relacije u relacijama (tzv. ugneždene relacije) ili relacije kao atributi torki
  - Ovaj uslov je ukinut kod ugneždenog relacionog modela i kod objektno-relacionih sistema
- Istorijski je 1NF uvedena da bi se onemogućila pojava viševrednosnih atributa, kompozitnih atributa i njihovih kombinacija

1NF

Ponavljanje = (propertyNo, pAddress, rentStart, rentFinish, rent, ownerNo, oName)

Nenormalizovana relacija (NNF)

Tabela sadrži više vrednosti atributa koje se ponavljaju

ClientNo	cName	propertyNo	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	John kay	PG4	6 lawrence St,Glasgow	1-Jul-00	31-Aug-01	350	CO40	Tina Murphy
		PG16	5 Novar Dr, Glasgow	1-Sep-02	1-Sep-02	450	CO93	Tony Shaw
CR56	Aline Stewart	PG4	6 lawrence St,Glasgow	1-Sep-99	10-Jun-00	350	CO40	Tina Murphy
		PG36	2 Manor Rd, Glasgow	10-Oct-00	1-Dec-01	370	CO93	Tony Shaw
		PG16	5 Novar Dr, Glasgow	1-Nov-02	1-Aug-03	450	CO93	Tony Shaw

Test za 1NF

Provera za prvu normalnu formu:

- Proveriti da li su svi atributi šeme relacije prosti (atomični)
- Ako jesu, tada je šema relacije u 1NF
- Ako nisu tada šema relacije nije u 1NF
  - Takvu šemu treba prevesti u 1NF, tj. treba izvršiti **1NF normalizaciju**

1NF u praksi

- Relacija u 1NF: presek jedne kolone i jedne vrste sadrži samo jednu vrednost.

Normalizacija

- Ukloniti horizontalne redundance u podacima
  - Ne postoje dve kolone koje čuvaju istu informaciju
  - Ne postoji kolona koja čuva više od jedne vrednosti
- Svaka vrsta treba da bude jedinstvena
  - Koristiti primarni ključ
- Postupak normalizacije

  izdvojiti grupu koja se ponavlja u novu relaciju, koja sadrži taj atribut kao i primarni ključ originalne relacije.

1NF u praksi

- Šta se dobija:
  - Lakše zadavanje upita/uređenja
  - Skalabilno
  - Svaka vrsta se može identifikovati kod ažuriranja
  - Ako se konceptualno projektovanje uradi kako treba, relacija je sigurno u 1NF!
  - Uslov 1NF je ukinut kod kod objektno-relacionih sistema!

Druga normalna forma (2NF)

- Druga normalna forma (2NF) se bazira na konceptu **potpune funkcionalne zavisnosti**

Definicija:

Funkcionalna zavisnost $X \rightarrow Y$ je **potpuna funkcionalna zavisnost** ako izbacivanjem bilo kog atributa A iz X ova zavisnost više ne važi

Definicija:

Funkcionalna zavisnost $X \rightarrow Y$ je **parcijalna funkcionalna zavisnost** ako se neki atribut $A \in X$ može izbaciti iz X i da zavisnost i dalje važi

Druga normalna forma (2NF)

Definicija:

Šema relacije **R**(R;$\mathbb{F}$) je u drugoj normalnoj formi (2NF) ako svaki atribut A u R ispunjava jedan od sledeća dva uslova:

1) javlja se u nekom ključu kandidatu

2) nije parcijalno zavisan od nekog ključa kandidata

Druga normalna forma (2NF)

Interpretacija definicije:

1. Šema relacije **R**(R;$\mathbb{F}$) je u drugoj normalnoj formi (2NF) ako svaki njen neključni atribut **nije parcijalno zavistan** od nekog ključa šeme relacije

2. Šema relacije **R**(R;$\mathbb{F}$) je u drugoj normalnoj formi (2NF) ako svaki njen neključni atribut potpuno funkcionalno zavisi od svakog ključa šeme relacije **R**

Test za 2NF

- Treba proveriti da li je svaki neključni atribut potpuno funkcionalno zavistan od svakog ključa kandidata

Postupak:

1. Naći sve ključeve šeme relacije
   a. Naći sve ključne atribute
   b. Naći sve neključne atribute
2. Izvršiti proveru uslova potpune funkcionalne zavisnosti

Specijalni slučajevi 2NF: Ako se u šemi relacije **R** svi ključevi kandidati sastoje samo od **jednog atributa** ili ako su svi atributi ključni, onda je šema relacije R u 2NF

2NF - Primer 1

Da li je šema relacije **R**(R;$\mathbb{F}$) u 2NF?

R=(A,B,C,D)

$\mathbb{F}$={AB→C,B→DC, BC→A}

2NF test

1. Naći sve ključeve šeme relacije
   K={B}
2. Kako jedini ključ K={B} sadrži jedan atribut, **R** je u 2NF.
3. KRAJ TESTA

2NF - Primer 2

- Da li je šema relacije **R**(R,$\mathbb{F}$) u 2NF?

  R=(A,B,C,D,E)

  $\mathbb{F}$={A→BC,CD→E,B→D,E →A}

2NF test

1. Naći sve ključeve šeme relacije

   K1=A, K2=E, K3=BC,K4=CD
2. Naći sve ključne atribute

   KA = {A, B,C,D,E}
3. Naći sve neključne atribute

   NKA = { }
4. Kako su svi atributi ključni, **R** je u 2NF.
5. KRAJ TESTA

2NF - Primer 3

- Da li je šema relacije **R**(R,$\mathbb{F}$) u 2NF?

  R=(S,A,I,P)

  $\mathbb{F}$={SI$\rightarrow$P, S$\rightarrow$ A}

2NF test

1. Naći sve ključeve šeme relacije

   K=SI
2. Naći sve ključne atribute

   KA = {S,I}
3. Naći sve neključne atribute

   NKA = {A,P }
4. Proveriti da li A parcijalno zavisi od ključa SI

   Kako je za **FZ**: S$\rightarrow$A , S $\subset$ K, to neključni atribut A parcijalno zavisi od ključa SI i šema relacije **nije u 2NF**.

   KRAJ TESTA

2NF u praksi

- Relacija mora da bude u 1NF, uz dodatni uslov da svaki neprimarni atribut potpuno zavisi od ključa

Normalizacija

- Eliminisati vertikalnu redundancu u podacima
  - Ista vrednost ne sme se ponavljati u nekoj vrsti
- Kompozitni ključevi
  - Sve kolone u vrsti moraju da se **referenciraju na sve delove ključa**

Šta se dobija

- Povećava se efikasnost čuvanja podataka
- Smanjuje ponavljanja podataka

2NF Normalizacija

- Preduslov da relacija bude u 1NF
- Eliminacija parcijalnih zavisnosti
  - Ako postoji parcijalna zavisnost, vršimo dekompoziciju šeme relacije na osnovu te FZ
  - U novu šemu relacije izdvajamo parcijalno zavisne atribute i atribut koji ih određuje (**ključ** nove relacije)
  - „Stara“ šema relacije sadrži sve atribute osim parcijalno zavisnih.

2NF Normalizacija

Primer: Šema relacije R=(S,A,I,P) nije u 2NF,

$\mathbb{F}$={SI→P, S→ A}

Ključ: SI

FZ koja narušava 2NF: S→A

(za A postoji parcijalna FZ u odnosu na ključ SI)

Normalizacija dekompozicijom: na osnovu S→A

R1=(S,I,P) (iz R izbašen parc. zav. atr. A)

R2=(S,A) (nova šema rel. na osnovu FZ)

Primer 2NF Normalizacije

Relacija

ClientRental(clientNo, propertyNo, rentStart, rentFinish, cName, pAddress, rent, ownerNo, oName)

ima sledeće FZ:

fd1	clientNo, propertyNo → rentStart, rentFinish	(Primarni Ključ)
fd2	clientNo → cName	(Parcijalna zavisnost)
fd3	propertyNo → pAddress, rent, ownerNo, oName	(Parcijalna zavisnost)
fd4	ownerNo → oName	(Tranzitivna zavisnost)
fd5	clientNo, rentStart → propertyNo, pAddress, rentFinish, rent, ownerNo, oName	(Ključ kandidat)
fd6	propertyNo, rentStart → clientNo, cName, rentFinish	(Ključ kandidat)

Primer (nast)

fd2 clientNo → cName
fd3 propertyNo → pAddress, rent, ownerNo, oName

Eliminacija parcijalnih zavisnosti, kreiranje 3 nove relacije

Client (clientNo, cName)
PropertyOwner (propertyNo, pAddress, rent, ownerNo, oName)
Rental (clientNo, propertyNo, rentStart, rentFinish)

Client

ClientNo	cName
CR76	John Kay
CR56	Aline Stewart

Rental

ClientNo	propertyNo	rentStart	rentFinish
CR76	PG4	1-Jul-00	31-Aug-01
CR76	PG16	1-Sep-02	1-Sep-02
CR56	PG4	1-Sep-99	10-Jun-00
CR56	PG36	10-Oct-00	1-Dec-01
CR56	PG16	1-Nov-02	1-Aug-03

PropertyOwner

propertyNo	pAddress	rent	ownerNo	oName
PG4	6 lawrence St,Glasgow	350	CO40	Tina Murphy
PG16	5 Novar Dr, Glasgow	450	CO93	Tony Shaw
PG36	2 Manor Rd, Glasgow	370	CO93	Tony Shaw

Treća normalna forma (3NF)

Definicija:

Šema relacije **R**(R;$\mathbb{F}$) je u trećoj normalnoj formi (3NF) u odnosu na skup funkcionalnih zavisnosti $\mathbb{F}$ ako za svaku funkcionalnu zavisnost u $\mathbb{F}^+$ oblika **X→A** (X⊆R, A je jedan atribut iz R) važi :

1) A⊆X (X→A je trivijalna funkcionalna zavisnost) **ili**
2) X je super ključ šeme relacije **R**(R;$\mathbb{F}$) **ili**
3) A je ključni atribut šeme relacije **R**(R;$\mathbb{F}$)

Odnos 2NF i 3NF

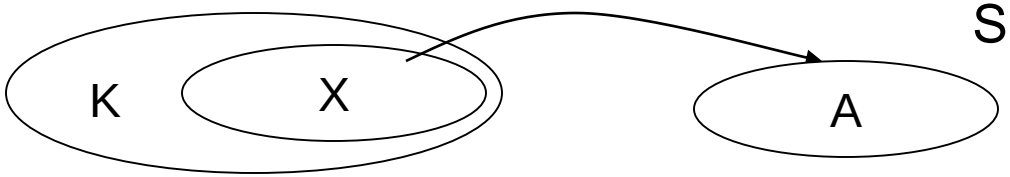
- Svaka 3NF relacija je takođe u 2NF
- Postoje relacije koje su u 2NF, a nisu u 3NF

Treći uslov 3NF definicije

- Ako FZ X→A ne zadovoljava uslov 3) iz 3NF definicije, razlozi mogu biti:

1. X je pravi podskup nekog ključa K
   - X→A je parcijalna zavisnost i par (X,A) je redundantan
2. X nije pravi podskup nijednog ključa
   - X→A je tranzitivna zavisnost zbog lanca zavisnosti K→X→A
     (ne možemo pridružiti X vrednost ključu K, a da ne pridružimo A vrednost X-u).
   - Ovaj uslov vodi ka anomalijama unosa, brisanja i modifikacije.

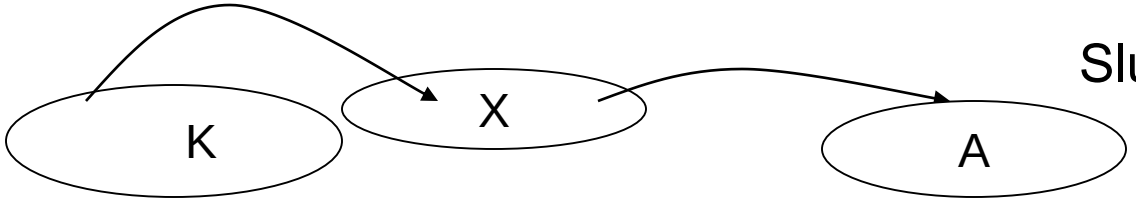
Parcijalne i tranzitivne zavisnosti



Slučaj 1: A nije u ključu, $X \subseteq K$

parcijalna zavisnost

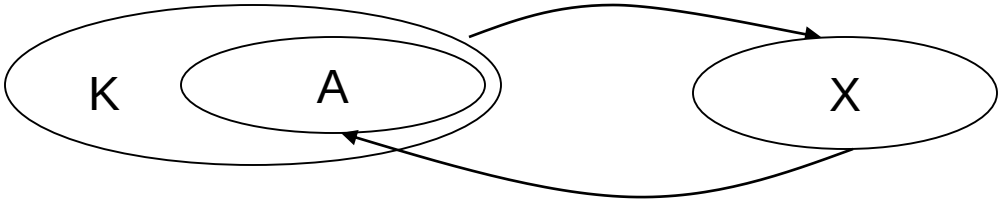
$X \rightarrow A$ je parcijalna zavisnost zavisnosti $K \rightarrow A$



Slučaj 2: A nije u ključu, $X \not\subset K$

tranzitivna zavisnost

$K \rightarrow X \rightarrow A$



Slučaj 3: A je u ključu, $X \not\subset K$

tranzitivna zavisnost

$K \rightarrow X \rightarrow A$

3NF test

- Treba proveriti da li svaka zavisnost iz F^+ zadovoljava uslov 1) ili 2) ili 3) iz 3NF definicije
- Može se dokazati da je dovoljno proveriti samo funkcionalne zavisnosti iz F
  - ako nijedna od FZ iz F ne narušava 3NF ograničenja, tada to neće činiti nijedna zavisnost iz F^+

- **Specijalni slučaj:** Ako F sadrži FZ čije desne strane sadrže više od jednog atributa, tada primenom pravila dekompozicije skup F treba transformisati u skup F' gde sve FZ sadrže po jedan atribut sa desne strane

3NF u praksi

- Instanca šeme koja nije u 3NF sadrži redundantne informacije koje su posledica FZ
- Relacija mora da bude u 2NF
  - Ako je relacija u 2NF, postoje dobre šanse da je i u 3NF
- Sve kolone moraju da direktno zavise od primarnog ključa

- Ako ste dobro odradili konceptualno projektovanje u preslikavanje na relacioni model, relacije su uglavnom u 3NF
- Šta se dobija
  - Nema redundanse u podacima

3NF normalizacija

- Preduslov: relacija je u 2NF
- Eliminacija tranzitivnih zavisnosti
  - Dekompozicija relacije na osnovu takve zavisnosti:

  Izdvoje se atributi iz takvih zavisnosti u novu relaciju zajedno sa kopijom atributa koji ih određuje/ju.

Primer 3NF normalizacije

Funkcionalne zavisnosti za Client, Rental i PropertyOwner:

Client

fd2	clientNo → cName	(Prim.ključ)

Rental

fd1	clientNo, propertyNo → rentStart, rentFinish	(Prim.ključ)
fd5	clientNo, rentStart → propertyNo, rentFinish	(Ključ kand.)
fd6	propertyNo, rentStart → clientNo, rentFinish	(Ključ kand.)

PropertyOwner

fd3	propertyNo → pAddress, rent, ownerNo, oName	(Prim. ključ)
fd4	ownerNo → oName	(Tranzitivna zavisnost)

Primer 3NF normalizacije

Rezultat normalizacije relacije PropertyOwner dekompozicijom na osnovu

fd4: ownerNo → oName

Client (clientNo, cName)

Rental (clientNo, propertyNo, rentStart, rentFinish)

PropertyOwner (propertyNo, pAddress, rent, ownerNo)

Owner (ownerNo, oName)

Primer 3NF normalizacije

Client

ClientNo	cName
CR76	John Kay
CR56	Aline Stewart

Rental

ClientNo	propertyNo	rentStart	rentFinish
CR76	PG4	1-Jul-00	31-Aug-01
CR76	PG16	1-Sep-02	1-Sep-02
CR56	PG4	1-Sep-99	10-Jun-00
CR56	PG36	10-Oct-00	1-Dec-01
CR56	PG16	1-Nov-02	1-Aug-03

PropertyOwner

propertyNo	pAddress	rent	ownerNo
PG4	6 lawrence St,Glasgow	350	CO40
PG16	5 Novar Dr, Glasgow	450	CO93
PG36	2 Manor Rd, Glasgow	370	CO93

Owner

ownerNo	oName
CO40	Tina Murphy
CO93	Tony Shaw

Boyce-Codd-ova normalna forma (BCNF)

Definicija:

Šema relacije **R**(R;$\mathbb{F}$) je u Boyce-Codd-ovoj normalnoj formi (BCNF) u odnosu na skup funkcionalnih zavisnosti $\mathbb{F}$ ako za svaku funkcionalnu zavisnost u $\mathbb{F}^+$ oblika $X \rightarrow Y$ ($X \subseteq R, Y \subseteq R$) važi:

1) $Y \subseteq X$ ($X \rightarrow Y$ je trivijalna funkcionalna zavisnost) ili
2) X je super ključ šeme relacije

- U BCNF sve netrivijalne FZ su oblika

super ključ → skup atributa

Odnos 3NF i BCNF

- 3NF sadrži treći uslov koji ne postoji u BCNF, što predstavlja relaksaciju BCNF uslova
- Svaka BCNF relacija je takođe u 3NF
- Postoje relacije koje su u 3NF, a nisu u BCNF

BCNF test

- Treba proveriti da li svaka zavisnost iz F^+ zadovoljava ograničenje 1) ili 2) iz definicije BCNF
- Dovoljno je proveriti samo funkcionalne zavisnosti iz F
  - ako nijedna od FZ iz $\mathbb{F}$ ne narušava BCNF ograničenja, tada to neće činiti nijedna zavisnost iz $\mathbb{F}^+$

BCNF test

Ulaz: Šema relacije **R**(R;$\mathbb{F}$)

Izlaz: TRUE ako je šema relacije **R**(R;$\mathbb{F}$) u BCNF ili FALSE ako nije

Metoda: Proveriti da li sve netrivijalne FZ X→Y iz $\mathbb{F}$ zadovoljavaju BCNF uslov 2), odnosno da li je X super ključ

Algoritam:

1. **za svaku** netrivijalnu FZ X→Y iz $\mathbb{F}$ **do**
2. Izračunati $X^+_{\mathbb{F}}$ // zatvarač skupa atributa X u odnosu na $\mathbb{F}$
3. **if** ($X^+_{\mathbb{F}} \neq$ R) **then** return(FALSE) // **X nije super ključ, šema // relacije nije u BCNF**
4. **enddo**
5. return(TRUE) // **šema relacije je u BCNF**

BCNF – Primer 1

- Da li je šema relacije **R**(R;$\mathbb{F}$) u BCNF?

  R={A,B,C,D} i

  $\mathbb{F}$={AB→C, BC→A, B→D}

BCNF test: Sve FZ iz $\mathbb{F}$ su netrivijalne i treba ih proveriti

1. Proveriti da li FZ AB→C zadovoljava BCNF ograničenje
   - Naći $(AB)^+_{\mathbb{F}}$=(A,B,C,D)
   - Kako je $(AB)^+_{\mathbb{F}}$=R zaključujemo da ova FZ **zadovoljava** BCNF ograničenje
2. Proveriti da li FZ BC→A zadovoljava BCNF ograničenje
   - Naći $(BC)^+_{\mathbb{F}}$=(B,C,A,D)
   - Kako je $(BC)^+_{\mathbb{F}}$=R zaključujemo da ova FZ **zadovoljava** BCNF ograničenje

BCNF – Primer 1

- Da li je šema relacije **R**(R;$\mathbb{F}$) u BCNF?

  R={A,B,C,D} i

  $\mathbb{F}$={AB→C, BC→A, B→D}

BCNF test: Sve FZ iz $\mathbb{F}$ su netrivijalne i treba ih proveriti

(nastavak)

3. Proveriti da li FZ B→D zadovoljava BCNF ograničenje
   - Naći $(B)^{+}_{\mathbb{F}}$=(B,D)
   - Kako je $(B)_{\mathbb{F}}^{+} \neq$R zaključujemo da ova FZ **NE zadovoljava** BCNF ograničenje, ova **šema nije u BCNF.**
     KRAJ TESTIRANJA

BCNF - Primer 2

Da li je šema relacije **R**(R;$\mathbb{F}$) u BCNF?

R=(A,B,C,D)

$\mathbb{F}$={AB→C,B→DC, BC→A}

BCNF test

1. Proveriti da li FZ AB→C zadovoljava BCNF uslov 2)

   $(AB)^+$=(A,B,C,D)

   Kako je $(AB)^+$=R, to je AB super ključ, **zadovoljava** uslov 2)

2. Proveriti da li FZ B→DC zadovoljava BCNF uslove

   $(B)^+$=(A,B,C,D)

   Kako je $(B)^+$=R, to je B super ključ, **zadovoljava** uslov 2)

BCNF - Primer 2

Da li je šema relacije **R**(R;$\mathbb{F}$) u BCNF?

R=(A,B,C,D)

$\mathbb{F}$={AB→C,B→DC, BC→A}

(**nastavak**)

3. Proveriti da li FZ BC→A zadovoljava BCNF uslove

$(BC)^+$=(A,B,C,D)

Kako je $(BC)^+$=R, to je BC super ključ, zadovoljava 2)

R je u BCNF, KRAJ TESTA

Svojstva BCNF šeme

- Instanca BCNF šeme ne sadrži redundantne informacije koje su posledica FZ
- Anomalije modifikacije i brisanja se ne javljaju u BCNF šemama
- Relacije sa više od 1 ključa još uvek imaju anomalije unosa
  - Da bi se izbegao ovaj problem jedan od ključeva se označava za primarni i zabranjuje se da njegovi atributi imaju NULL vrednosti

BCNF u praksi

- Relacija je u BCNF ako, i samo ako, svaki atribut koji određuje neki drugi atribut je ključ kandidat.
- Razlika između 3NF i BCNF je u tome što za zavisnost A → B,
  - 3NF dozvoljava da B može da bude primarni atribut, dok A nije kandidat za ključ
  - BCNF zahteva da takva zavisnost može da postoji samo ako je A kandidat za ključ

BCNF normalizacija (Primer 1)

- Normalizacija šeme relacije **R**(R;$\mathbb{F}$) do BCNF?

  R={A,B,C,D} i

  $\mathbb{F}$={AB→C, BC→A, B→D}

BCNF test: Sve FZ iz $\mathbb{F}$ su netrivijalne i treba ih proveriti

FZ koja narušava BCNF: B→D

Dekompozicija:

R1={A,B,C} (početna relacija bez desne strane FZ B→D)

R2={B,D} (na osnovu B→D)

Primer BCNF normalizacije

ClientInterview

ClientNo	interviewDate	interviewTime	staffNo	roomNo
CR76	13-May-02	10.30	SG5	G101
CR76	13-May-02	12.00	SG5	G101
CR74	13-May-02	12.00	SG37	G102
CR56	1-Jul-02	10.30	SG5	G102

fd1 clientNo, interviewDate → interviewTime, staffNo, roomNo (Prim.ključ)
fd2 staffNo, interviewDate, interviewTime→ clientNo (Kand.ključ)
fd3 roomNo, interviewDate, interviewTime → clientNo, staffNo (Kand.ključ)
fd4 staffNo, interviewDate → roomNo (nije ključ kandidat)

Posledica: potencijalne anomalije ažuriranja.
Na primer, treba ažurirati dve torke ako se menja vrednost roomNo za staffNo SG5 i 13-May-02.

Primer BCNF normalizacije

Moramo da eliminišemo FZ koja narušava BCNF tako što ćemo da uradimo dekompoziciju po toj FZ na dve relacije Interview i SatffRoom:

fd4 staffNo, interviewDate → roomNo

Interview (clientNo, interviewDate, interviewTime, staffNo)

StaffRoom(staffNo, interviewDate, roomNo)

Interview

ClientNo	interviewDate	interviewTime	staffNo
CR76	13-May-02	10.30	SG5
CR76	13-May-02	12.00	SG5
CR74	13-May-02	12.00	SG37
CR56	1-Jul-02	10.30	SG5

StaffRoom

staffNo	interviewDate	roomNo
SG5	13-May-02	G101
SG37	13-May-02	G102
SG5	1-Jul-02	G102

Denormalizacija

- Spajanje relacija u jednu kako bi se povećala efikasnost nekih upita/pretraga
- Može se koristiti ako se proceni da je dobitak kod pretrage veći od problema sa anomalijama
- Preporuka: izvršiti najpre normalizaciju **kako** treba, pa tek onda denormalizaciju **ako** treba
- Koristite jedino ako ne postoji drugi način optimizacije izvršenja upita
  - Pokušajte sa temp tabelama, UNION-ma, VIEW-ma, ugnježdenim upitima i sl

Normalne forme i normalizacija

Pitanja ???

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

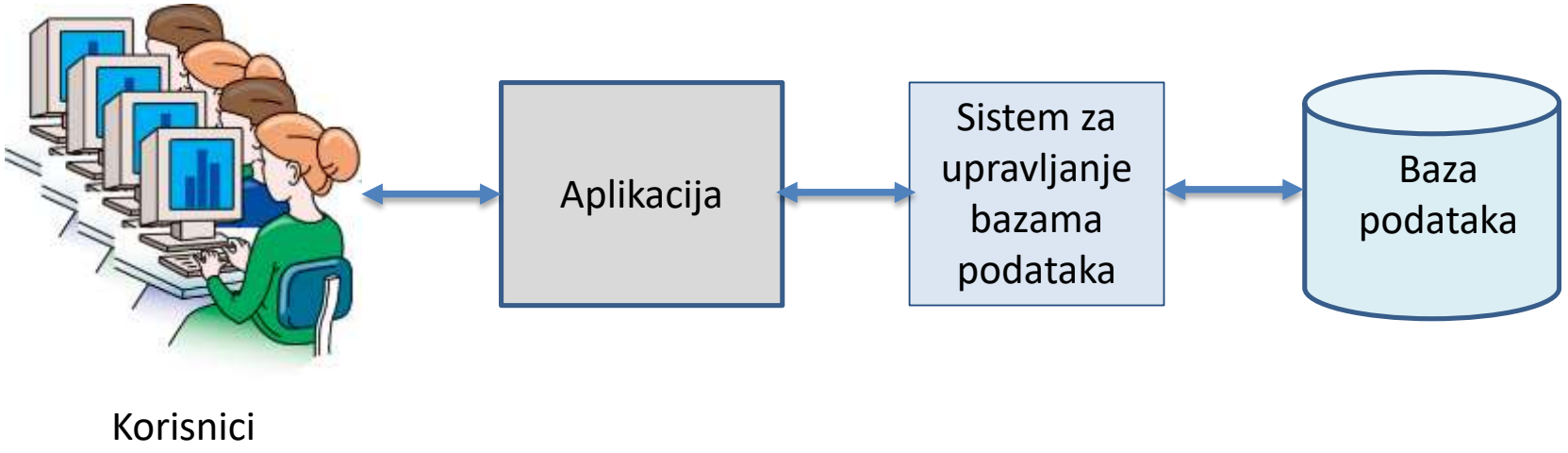
Transakciona obrada

Prof.dr Leonid Stoimenov

Pregled

- Višekorisnički DBMS
- Transakcije
- Upravljanje transakcijama
  - Kontrola konkurencije
  - Oporavak baze podataka

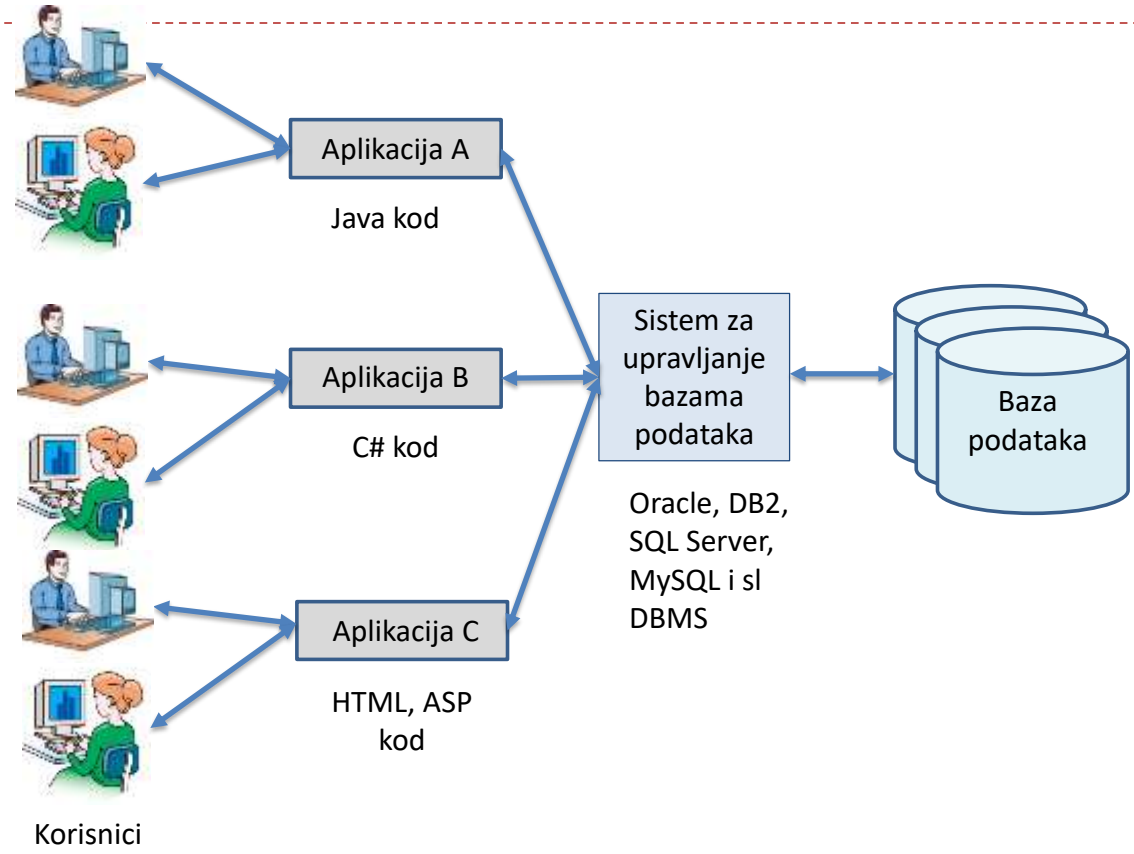
Višekorisnički DBMS



Sistem baza podataka

- Bazu podataka obično koristi više različitih aplikacija
  - Aplikativni programi pisani na nekom programskom *host* jeziku
  - Interaktivni upit ("ad hoc" upit) realizovan na nekom upitnom jeziku

Višekorisnički DBMS



- Višekorisnički DBMS omogućava većem broju korisnika da konkurentno koriste bazu podataka
  - višekorisnički DBMS zahteva i višekorisnički OS

Transakcije

- Različiti korisnici mogu pokrenuti više različitih izvršenja istog "programa" DBMSa koji pristupa i eventualno ažurira neka polja u bazi podataka

- **Transakcija** je program u izvršenju DBMSa koji predstavlja logičku jedinicu obrade baze podataka
  - sadrži jednu ili više operacija koje pristupaju bazi podataka (umetanje, brisanje, modifikacija, pretraživanje,…)

- Transakcija mora da vidi bazu podataka kao konzistentnu
- Konkurentno korišćenje baze podataka preko DBMSa krije u sebi određene opasnosti
  - **Primer:** Jedna transakcija učita u svoj radni prostor neki podatak iz baze podataka, a druga ga odmah nakon toga u bazi podataka izmeni. Prva transakcija će koristiti netačan podatak.

ACID svojstva transakcije

- ***Atomičnost (Atomicity).*** **Transakcija je atomska jedinica obrade što znači da se mora izvršiti u celini**
  - Za očuvanje atomičnosti transakcije zadužen je deo DBMSa **Menadžer oporavka**
    - U slučaju kraha sistema menadžer oporavka mora obezbediti **poništavanje** uticaja transakcije na bazu podataka
- ***Konzistentnost (Consistency).*** Korektno izvođenje svake transakcije treba da prevede bazu podataka iz jednog u drugo konzistentno stanje
  - Za očuvanje ovog svojstva odgovoran je **programer aplikacije** nad bazom podataka ili DBMS komponenta za **proveru integriteta**

ACID svojstva transakcije (nast.)

- ***Izolovanost (Isolation).*** **Sve promene koje nad bazom podataka čini jedna transakcija su nevidljive za ostale transakcije sve dok ih transakcija ne komituje (potvrdi).** Ako se ovo ograničenje strogo poštuje, rešava se problem privremenog ažuriranja, tj. nije neophodno kaskadno vraćanje transakcije unazad. Različite metode konkurencije i oporavka uvode različite varijante ovog ograničenja
  - Za očuvanje ovog svojstva odgovorna je DBMSova **metoda kontrole konkurencije**
- ***Postojanost (Durability).*** **Sve promene koje transakcija učini nad bazom podataka nakon komitovanja nesmeju biti izgubljene**
  - Za očuvanje ovog svojstva odgovoran je DBMSov **menadžer oporavka**

„Granice“ transakcije

- Korisnik specificira granica transakcije u aplikacionom programu – uz eksplicitno navođenje naredbi
  - **BEGIN TRANSACTION** i
  - **END TRANSACTION**
  - Sve operacije za pristup bazi podataka koje se nalaze između ove dve naredbe smatraju se jednom transakcijom
- Jedan aplikacioni program može sadržati više transakcija
- U DBMS-u se implementira protokol **kontrole konkurencije** i **zaključavanja podataka** koji upravlja konkurentnim pristupima bazi podataka

COMMIT i ROLLBACK

- **COMMIT** specificira uspešan kraj transakcije
  - Sve promene BP koje su posledica operacija transakcije su zapamćene
  - Te promene su nadalje vidljive drugim transakcijama
- **ROLLBACK** signalizira da je neuspešan kraj transakcije
  - Sve promene BP koje su posledica tranakcije se poništavaju
  - Stanje je kao da transakcija nikad nije ni izvršena

Primer transakcije u SQL-u

```
UPDATE authors
SET au_fname = 'John'
WHERE au_id = '172-32-1176‘
```

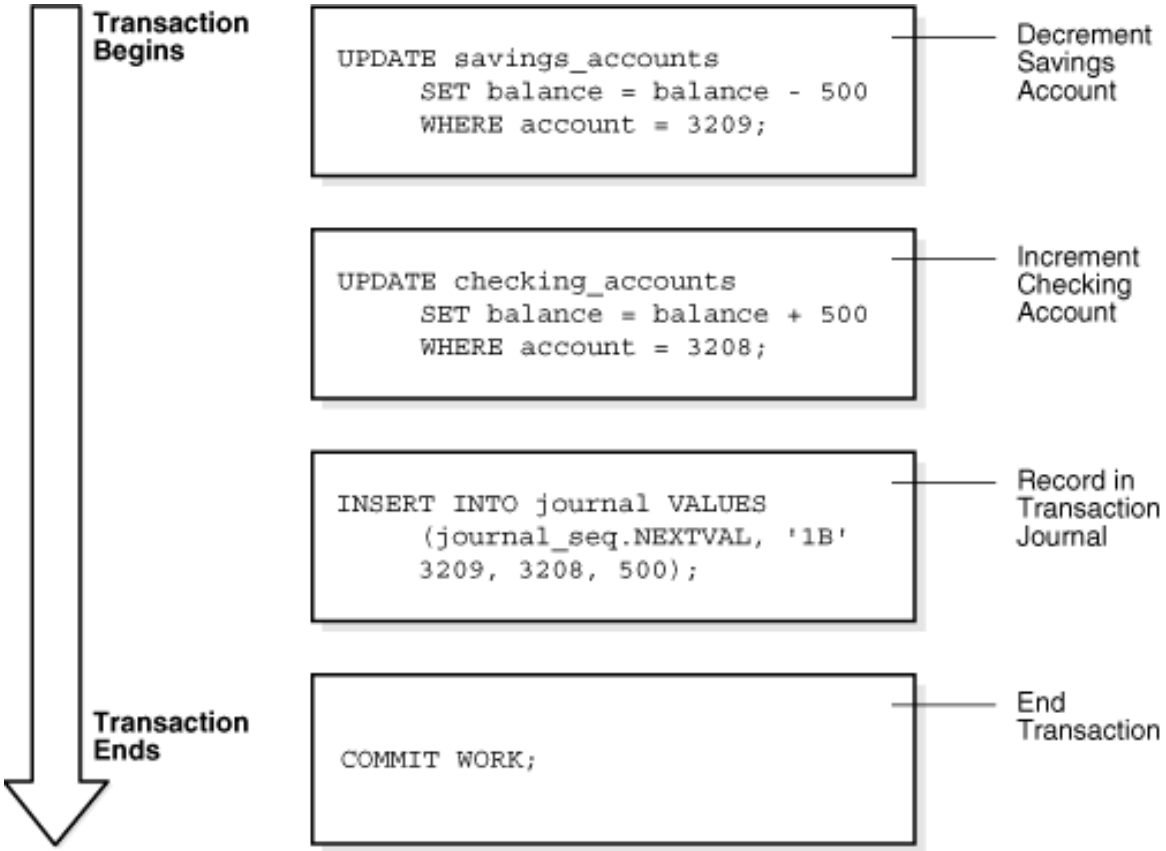
Autocommit transaction

```
BEGIN TRANSACTION
   UPDATE authors  SET au_fname = 'John'
    WHERE au_id = '172-32-1176'

   UPDATE authors SET au_fname = 'Marg'
    WHERE au_id = '213-46-8915'
COMMIT
END TRANSACTION
```

Primer transakcije u SQL-u

- BEGIN TRANSACTION



- END TRANSACTION

Transakcije na nivou DBMSa

- Transakcije se mogu posmatrati preko operacija za pristup i ažuriranje elemenata (objekata) baze podataka
  - **read_item** (kraće **read** ili **r**)
  - **write_item** (kraće **write** ili **w**)

Primer transakcije

- Transakcija T1

```
read_item(X);
X=X-N;
write_item(X);
read_item(Y);
Y=Y+N;
write_item(Y);
```

- Transakcija T2

```
read_item(X);
X=X+M;
write_item(X);
```

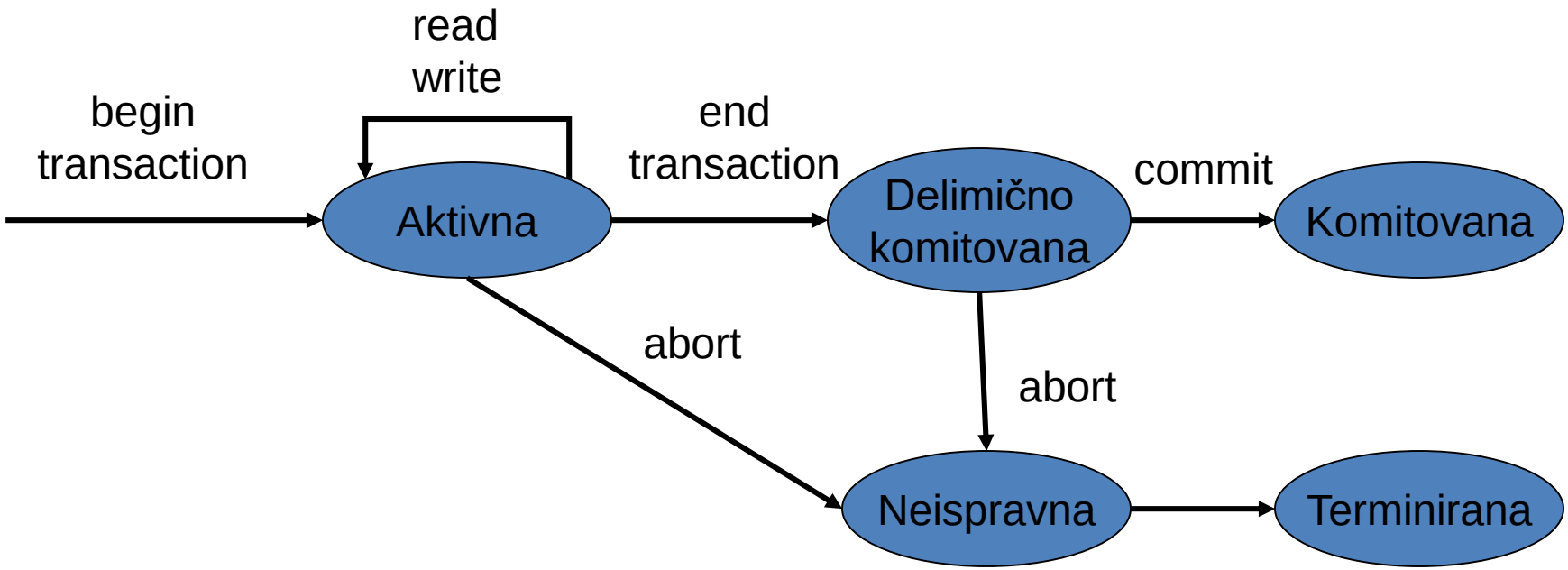
Operacije nad transakcijama

- **begin_transaction:** markira početak izvodjenja transakcije
- **read** ili **write:** specificira operacije čitanja i upisa elemenata baze podataka koje se izvršavaju kao deo transakcije
- **end_transaction:** označava da su završene *read* i *write* operacije iz transakcije i markira krajnju granicu izvršenja transakcije. U ovoj tački je neophodna provera da li su promene u bazi podataka trajno izvedene (tj. da li je transakcija **komitovana**) ili je izvedeno **abortovanje** transakcije iz nekog razloga

Operacije nad transakcijama

- **commit_transaction:** daje informaciju da je transakcija uspešno izvedena tako da su promene izvršene u okviru transakcije sigurno komitovane u bazu podataka i da neće biti poništene
- **rollback (abort):** signalizira da je transakcija završena neuspešno tako da treba ponoviti sve operacije ažuriranja koje su ovom transakcijom obuhvaćene
- **undo:** slična je operaciji rollback osim što se primenjuje na jednu operaciju, a ne na čitavu transakciju
- **redo:** ukazuje da određene operacije transakcije treba ponovo izvesti da bi osigurali uspešnost izvodjenja svih operacija komitovane transakcije nad bazom podataka

Dijagram stanja transakcije



Menadžer transakcija

- Menadžer transakcija forisira ACID svojstva
  - Određuje redosled izvršenja
  - Koristi COMMIT i ROLLBACK da obezbedi atomičnost
  - Katanci i timestamps se koriste da se obezbedi konzistentnost i izolacija za konkurentne transakcije
  - Održava Log za obezbeđivanje postojanosti kod pada sistema

Kontrola konkurencije

- **Kontrola konkurencije** je proces upravljanja simultanim (konkurentnim) operacijama nad bazom podataka da bi se onemogućila njihova međusobna interferencija
- DBMS prepliće akcije različitih transakcija da bi poboljšao performanse – povećao protočnost ili poboljšao vreme odgovora za kratke transakcije
  - Nisu sva preplitanja dozvoljena

Zašto je neophodna kontrola konkurencije?

- Mogu se javiti različiti problemi kada se transakcije izvršavaju konkurentno bez kontrole
- Karakteristični su sledeći problemi:
  - problem gubitka pri ažuriranju
  - problem privremenog ažuriranja
  - problem nekorektnog sabiranja
  - problem neponovljivog čitanja
- Primeri transakcija

T1	T2
read(X);	**read(X);**
X:=X-N;	**X:=X+M;**
write(X);	**write(X);**
read(Y);	
Y:=Y+N;	
write(Y);	

Problem gubitka pri ažuriranju

- Vrednost X je nekorektna jer T2 očitava vrednost X pre nego što T1 izmeni njenu vrednost u bazi podataka i stoga je izgubljena ažurirana vrednost X iz transakcije T1
- Primer:
  - Ulaz: X=5,N=2,M=1
  - Izlaz: X=6, tačno je X=4

T1	T2
read(X);	
X:=X-N;	
	read(X);
	X:=X+M;
write(X);	
read(Y);	
	write(X);
Y:=Y+N;	
write(Y);	

Problem privremenog ažuriranja ili prljavog čitanja

- Ako iz nekog razloga dođe do pada transakcije nakon ažuriranja nekog elementa (u našem slučaju elementa X), potrebno je taj element vratiti na staru vrednost
- Međutim, može se desiti da neka druga transakcija pristupi tom elementu pre nego što se element vrati na staru vrednost
- Ova druga transakcija čita privremenu vrednost tog elementa zbog čega se javlja neispravnost

T1	T2
read(X);	
X:=X-N;	
write(X);	
	read(X);
	X:=X+M;
	write(X);
read(Y);	
Pad transakcije	

Problem netačnog sumiranja

- T3 čita X posle oduzimanja N i čita Y pre dodavanja N, tako da je zbir neispravan
- U ovoj transakciji su nepravilno isprepletane operacije transakcija T1 koja ažurira elemente baze podataka X i Y i transakcije T3 koja čita te iste elemente

T1	T3
	suma:=0;
	read(A);
	suma:=suma+A;
read(X);	
X:=X-N;	
write(X);	
	read(X);
	suma:=suma+X;
	read(Y);
	suma:=suma+Y;
read(Y);	
Y:=Y+N;	
write(Y);	

Problem neponovljivog čitanja

- Javlja se kada transakcija T čita element podatka X dva puta
- Ako neka druga transakcija T' ažurira element podatka X između ova dva čitanja, T će dobiti dve različite vrednosti za isti element podatka X

Metode zaključavanja

- Za kontrolu konkurentnog pristupa podacima koristi se **zaključavanje**
- Kada transakcija pristupi bazi podataka koristi **katanac (lock)** da zabrani ostalim transakcijama pristup bazi podataka da bi se preventivno delovalo na pojavu nekorektnih rezultata
- Postoji nekoliko varijacija metode zaključavanja, ali osnove metode su zajedničke:
  - Transakcija mora postaviti katanac na element podatka pre operacije čitanja i upisa u bazu podataka
  - Katanac sprečava ostale transakcije da modifikuju ili čak i da čitaju taj element (ekskluzivni katanac)
  - Može se zaključati cela baza, jedan slog ili jedno polje u slogu
    - To je granularnost zaključavanja

Oporavak baze podataka

- **Oporavak baze podataka** je proces vraćanja (restauracije) baze podataka u korektno stanje nakon pojave neke neispravnosti

Zašto je neophodan oporavak baze podataka?

- Pri lansiranju neke transakcije DBMS mora obezbediti sledeće:
  - da se sve operacije transakcije uspešno izvedu i da se njihov efekat trajno zapiše u bazu podataka, ili
  - da transakcija ne utiče na bazu podataka, niti na bilo koju drugu transakciju
- DBMS mora osigurati da se ne desi da samo neke operacije transakcije imaju uticaj na bazu podataka
  - To se može desiti ako transakcija padne u toku njenog izvodjenja

Razlozi zbog kojih može doći do pada transakcije (1)

1. ***Pad računara:*** Za vreme izvođenja transakcije došlo je do greške u hardveru ili softveru. Pri tom može doši do gubitka sadržaja memorije
2. ***Greška u sistemu ili u transakciji.*** Neke operacije u transakciji mogu dovesti do pada sistema. Na primer, prekoračenje ili deljenje nulom, nedozvoljena vrednost nekog parametra, logička greška u programu ili operater mogu prekinuti izvršenje transakcije
3. ***Transakcija je detektovala lokalnu grešku ili uslov izuzeća***. U toku izvođenja transakcije se može desiti neki događaj koji zahteva otkazivanje transakcije. Na primer, nisu nađeni podaci neophodni za rad

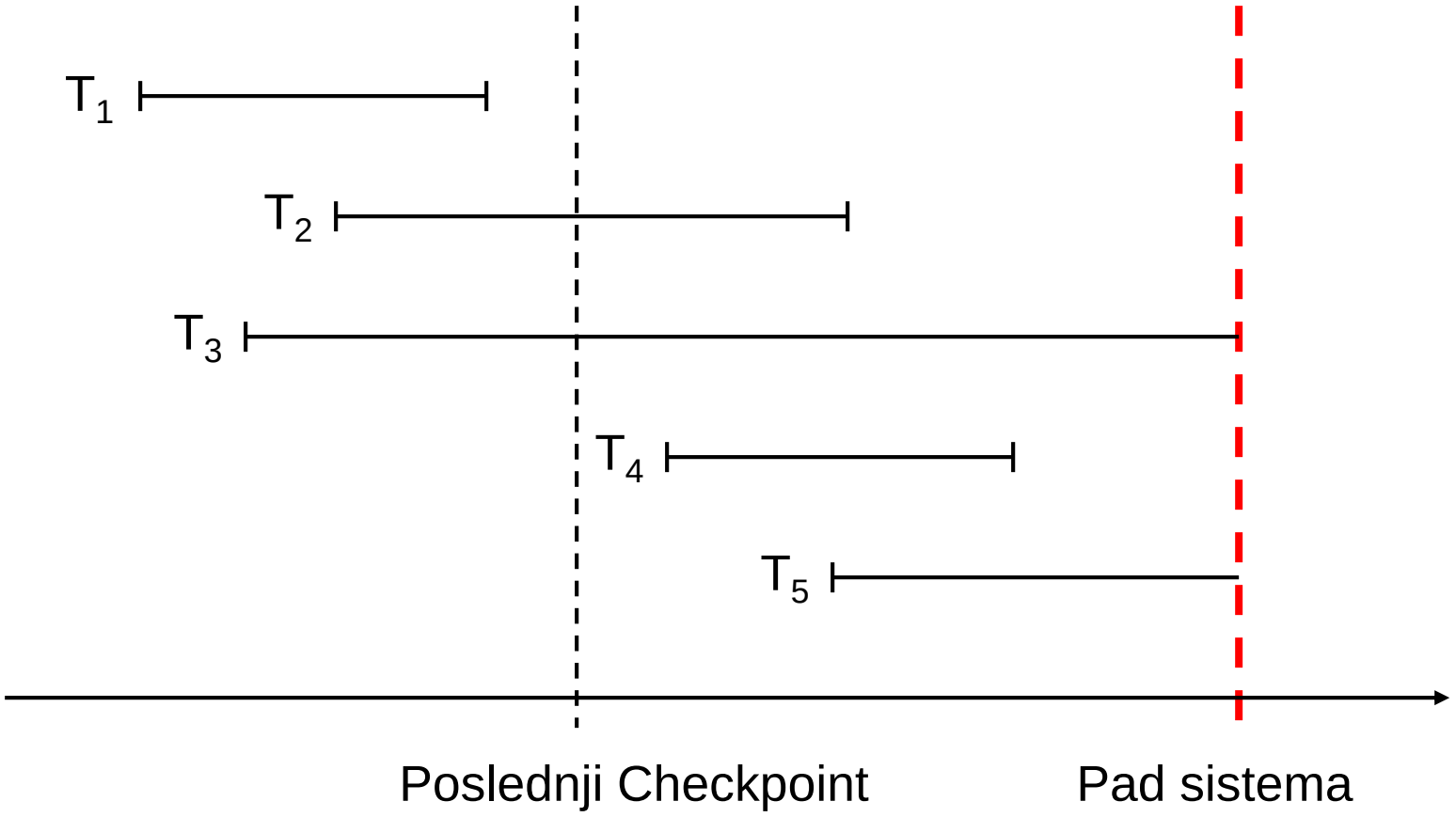
Razlozi zbog kojih može doći do pada transakcije (2)

4. ***Prisilna primena kontrole konkurencije.*** Metoda kontrole konkurencije može doneti odluku o abortiranju transakcije, naravno uz mogućnost njenog restartovanja.
5. ***Neispravnost diska***. Može doći do gubitaka podataka sa diska zbog njegove neispravnosti ili zbog problema u izvodjenju operacije čitanja sa diska ili zapisa na disk.
6. ***Fizički problemi i katastrofe.*** Može doći do fizičkog uništavanja medijuma ili do montiranja pogrešnog medijuma.

Transakcije i oporavak

- Transakcije su **osnovne jedinice oporavka** u sistemu baze podataka
- **Zadatak menadžera oporavka** je da garantuje 2 od 4 ACID svojstva transakcije – **atomičnost i postojanost**
- Menadžer oporavka treba da osigura, pri oporavku od nesipravnosti, da se **svi efekti** transakcije permanentno zabeleže u bazi podataka **ili nijedan** od njih
- Situacija je komplikovana jer upis u bazu podataka nije atomična akcija, pa je moguće da je transakcija izvela operaciju komitovanja, a da njen efekat još nije permanentno upisan u bazu podataka

Primer oporavka transakcija



Oporavak transakcije

UNDO i REDO: liste transakcija

UNDO = sve transakcije startovne na poslednjem checkpoint-u

REDO = prazna lista

Za svaku stavku u Log-u, počev od poslednjeg checkpoint-a

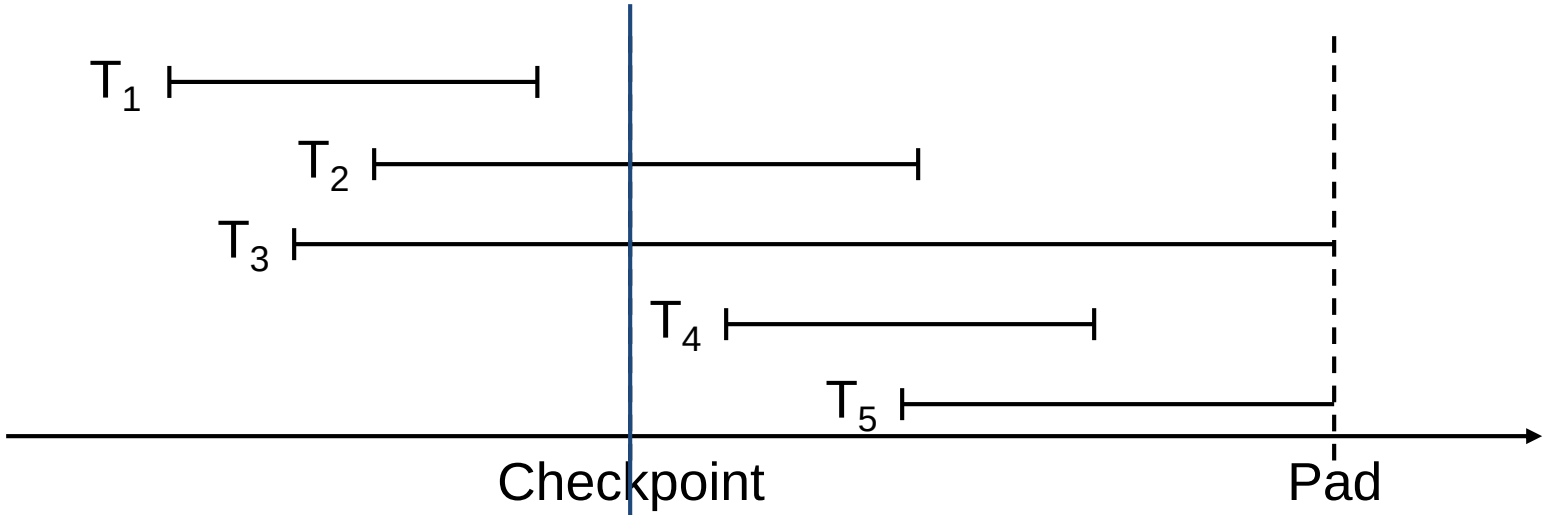
Ako nađen BEGIN TRANSACTION za T

Dodaj T u UNDO

Ako nađen COMMIT za T

Prebaci T iz UNDO u REDO

Oporavak transakcije



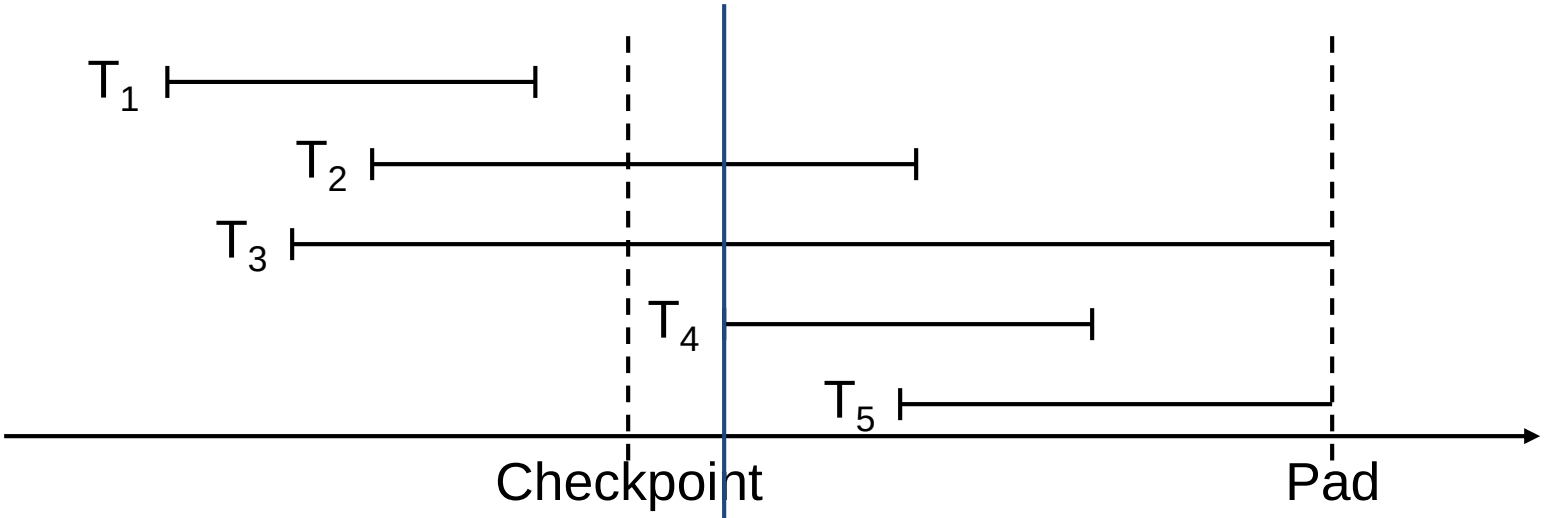
UNDO: T_2, T_3

REDO:

Poslednji Checkpoint

Aktivne transakcije: T_2, T_3

Oporavak transakcije



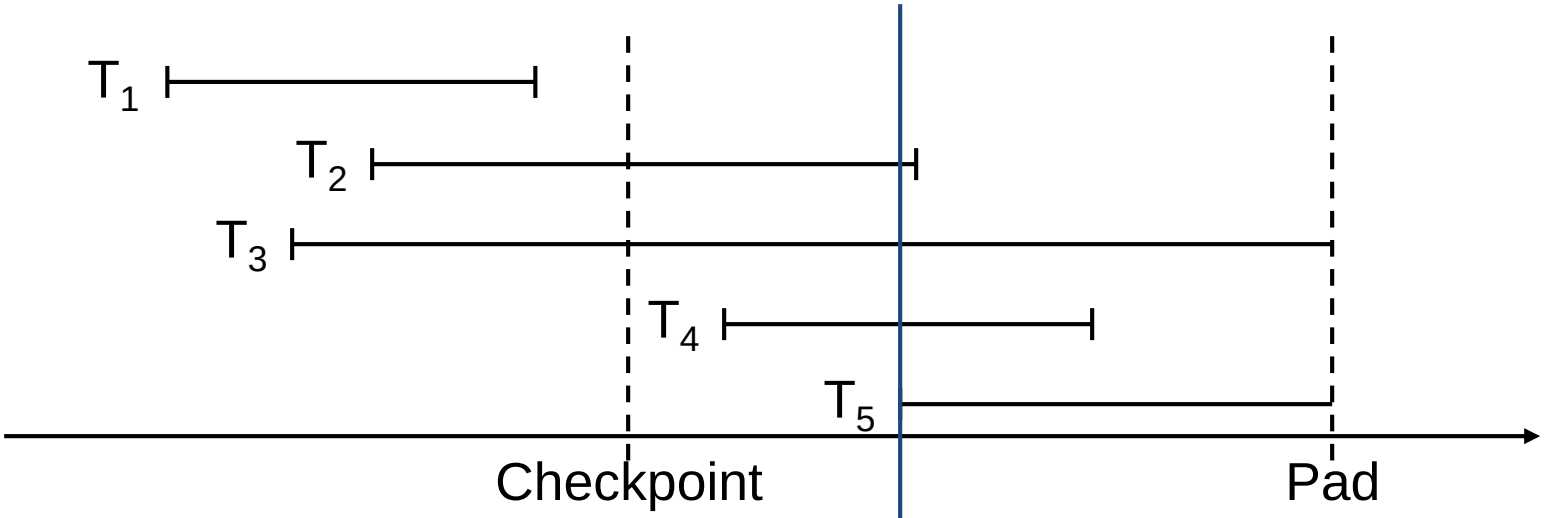
UNDO: T_2, T_3, T_4

REDO:

T4 počinje

Dodaj T_4 u UNDO

Oporavak transakcije



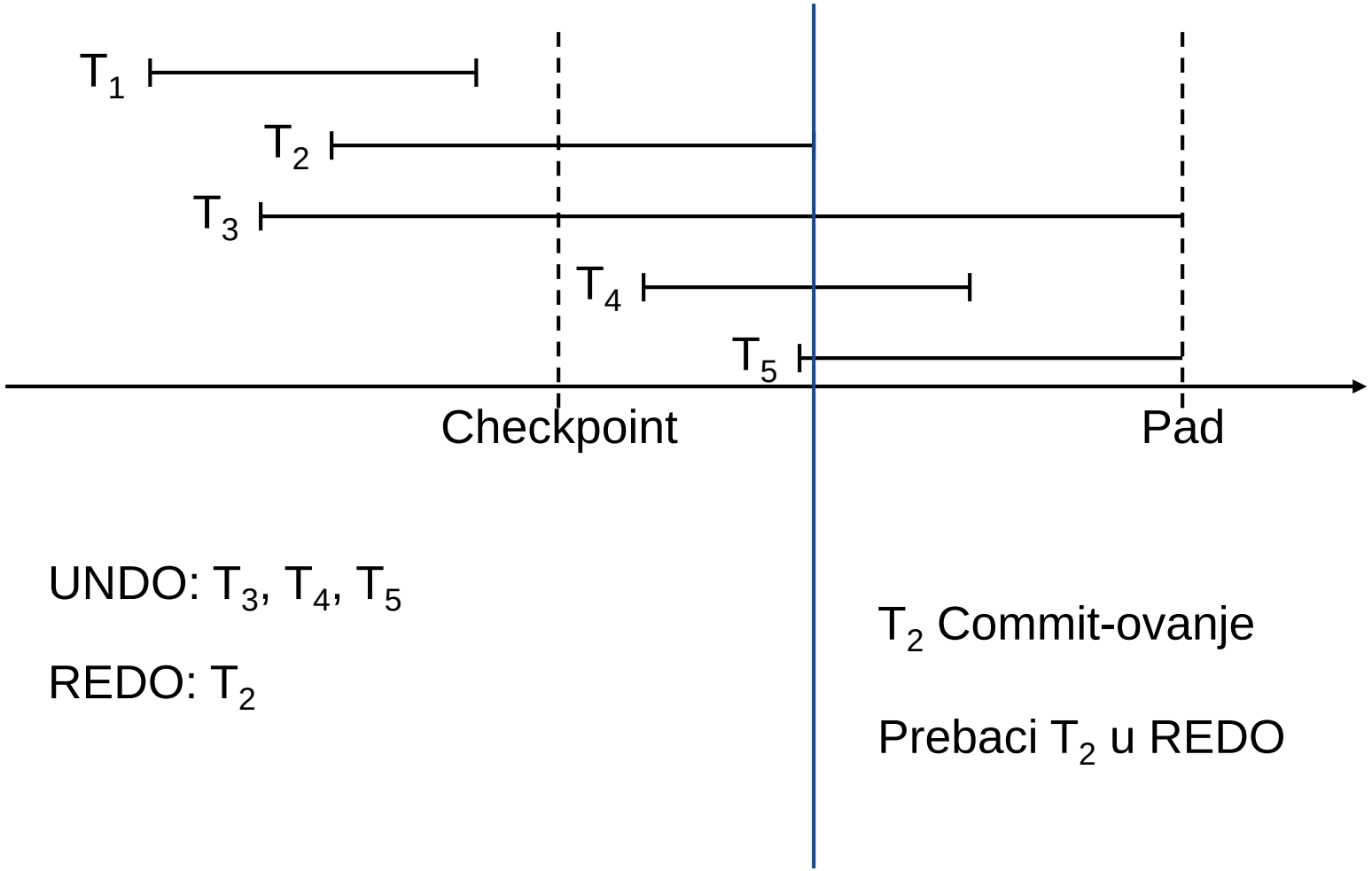
UNDO: T_2, T_3, T_4, T_5

REDO:

T_5 počinje

Dodaj T_5 u UNDO

Oporavak transakcije



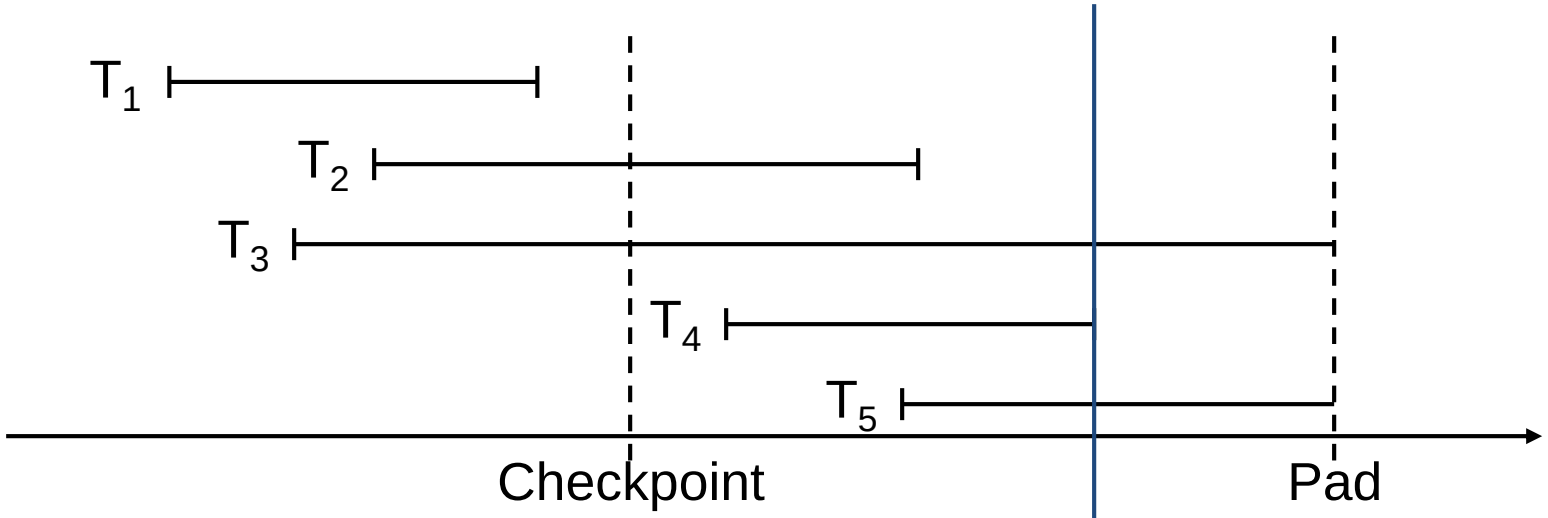
UNDO: T_3, T_4, T_5

REDO: T_2

T_2 Commit-ovanje

Prebaci T_2 u REDO

Oporavak transakcije



UNDO: T_3, T_5

REDO: T_2, T_4

T_4 Commit-ovanje

Prebaci T_4 u REDO

Transakciona obrada

Pitanja ???

Baze podataka

Katedra za računarstvo
Elektronski fakultet u Nišu

Arhitekture sistema baza podataka

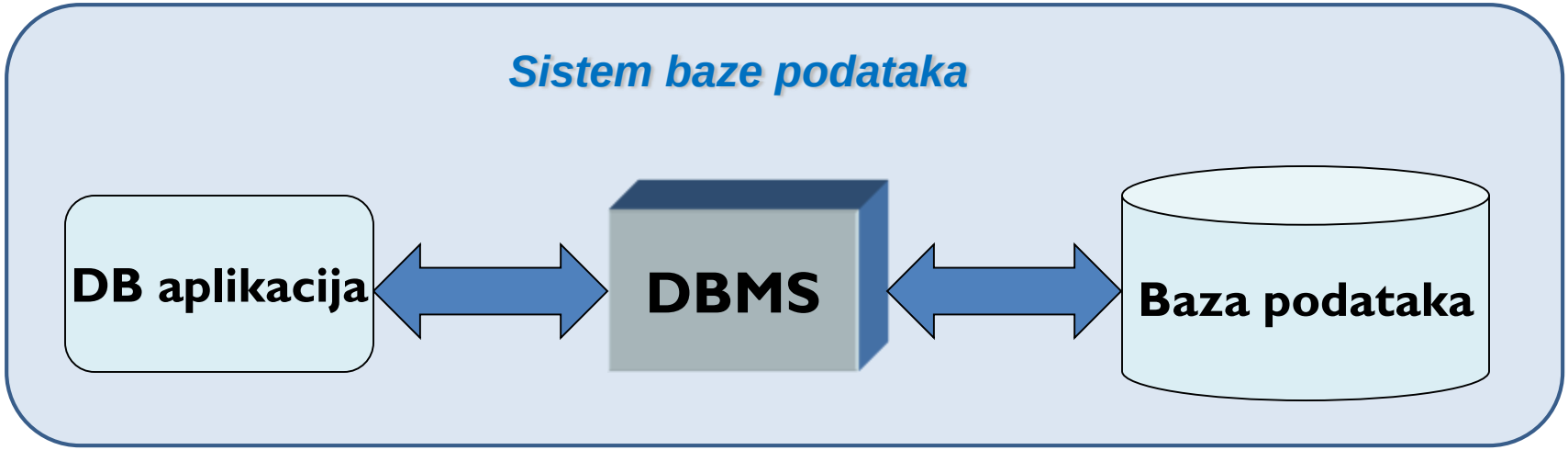
Prof.dr Leonid Stoimenov

Pregled

- Podsetnik
- Arhitekture višekorisničkih DB sistema

Podsetnik

- **Baza podataka** predstavlja kolekciju **povezanih** podataka organizovanih u logičke celine predstavljene **tabelama**.
- **Sistem za upravljanje bazama podataka** (DBMS) - Softverski sistem koji omogućava definisanje, kreiranje i manipulisanje bazom podataka
- **Aplikacija baze podataka -** Program koji interaguje sa bazom podataka u toku svog izvršenja
- **Sistem baze podataka -** Kolekcija aplikacionih programa koji interaguju sa bazom podataka, DBMS i baza podataka

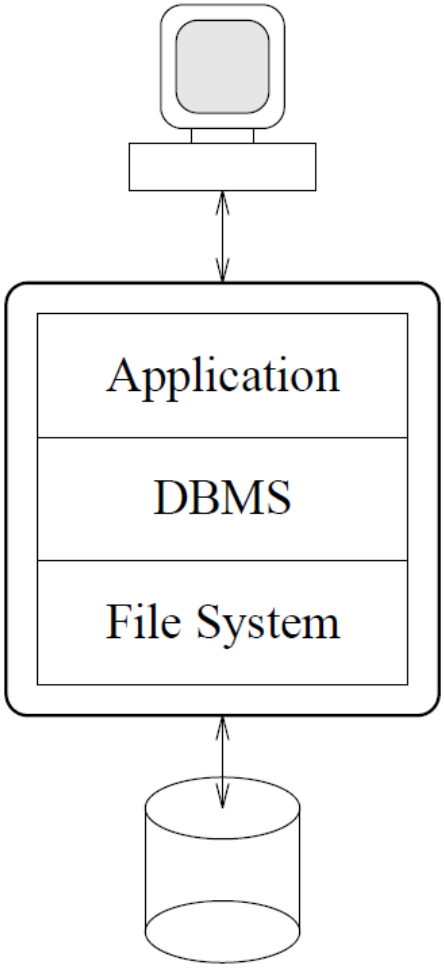


Struktura aplikacije nad bazom podataka

- Komponente aplikacije nad BP su
  - Baza podataka – čuva podatke
  - Transakciona logika – obrada podataka (kod BP), kontrolisana od strane DBMSa, na nivou read-write operacija
  - Poslovna logika i logika obrade podataka – obrada podataka (kod aplikacija) koja je definisana poslovnim procesima
  - Korisnički interfejs – prrezentacija podataka korisniku (forme, izveštaji, validacija i kontrola unosa podataka, poruke)

Monolitni sistemi

- Aplikacije
  - Interakcija sa korisnikom
  - Aplikaciono-specifični zadaci
- DBMS
  - Optimizacija upita
  - Obrada upita
  - Upravljanje Transakcijijama
  - Sigurnost i upravljanje integritetom
- Fajl sistem
  - Čuvanje podataka



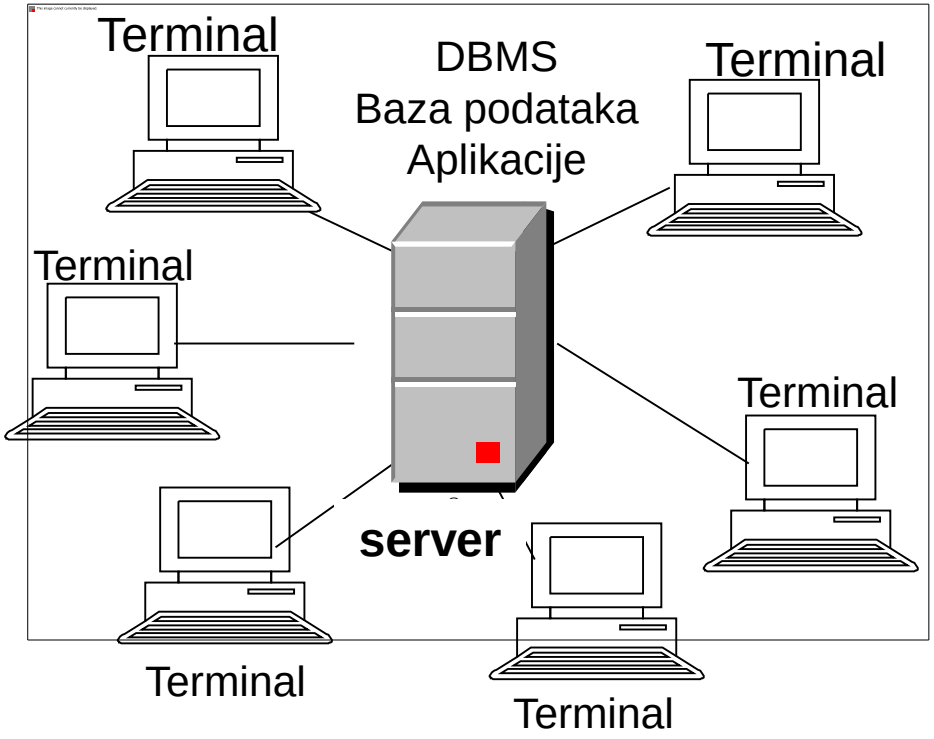
Arhitekture višekorisničkih sistema baze podataka

- Teleprocesing
- Fajl-server
- Klijent-server
  - u 2 nivoa (two-tier) - tradicionalni
  - u 3 nivoa (three-tier)

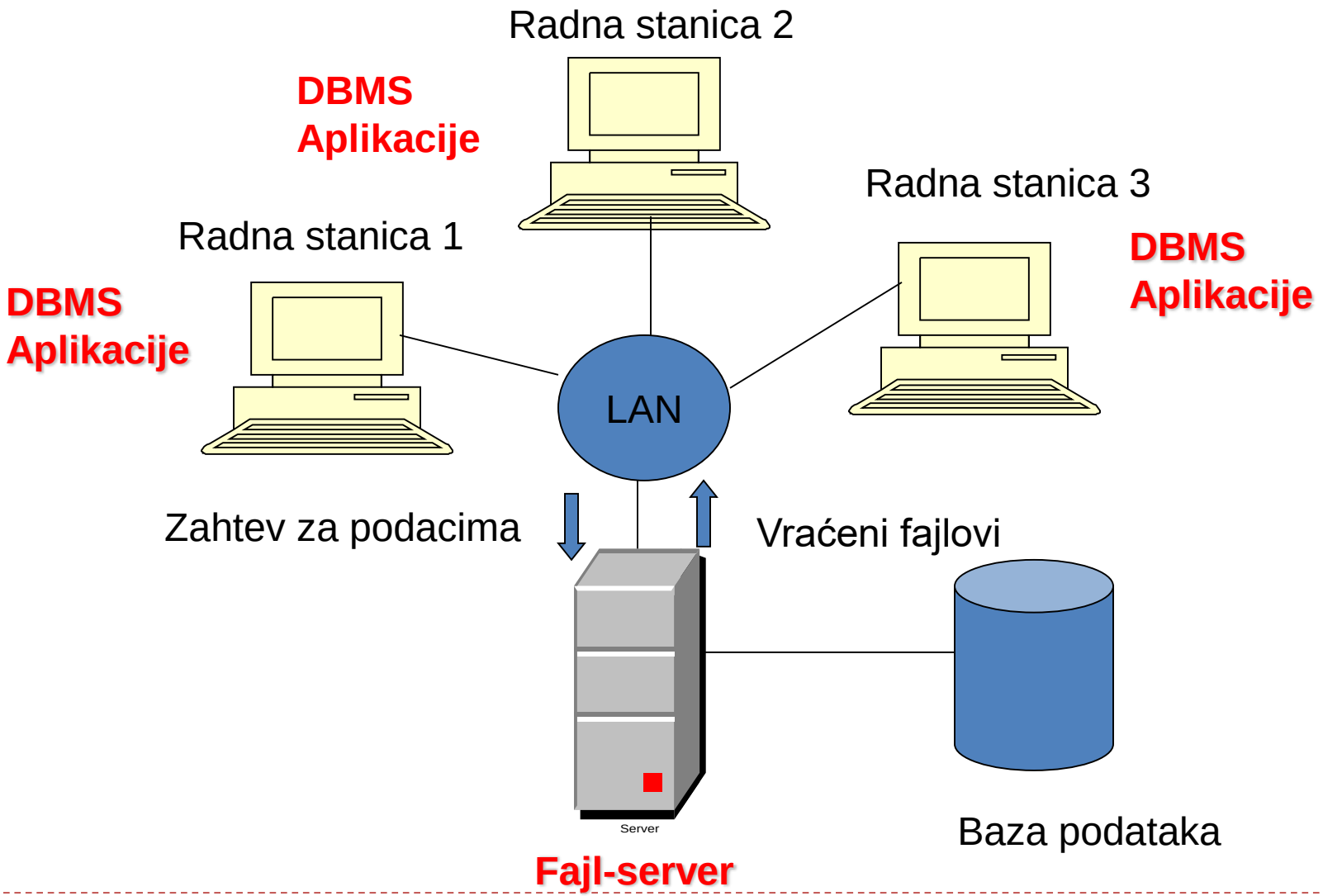
Teleprocesing

- Postoji jedan centralni procesor i veći broj terminala
- Sva obrada se vrši na centralnom procesoru

- Terminali preko komunikacionog podsistema OS-a šalju poruke aplikacionom programu koji koristeći usluge DBMS-a obrađuje zahtev i vraća poruku korisnikovom terminalu

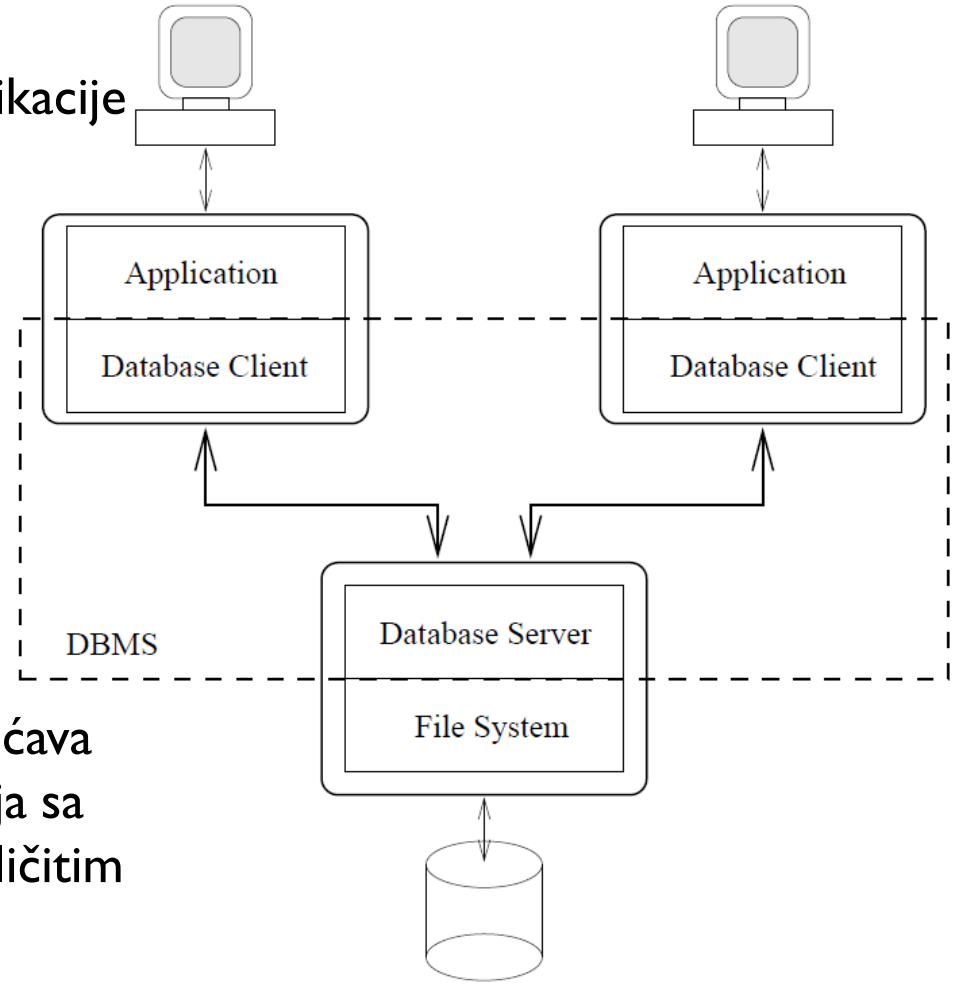


Fajl-server arhitektura (1)



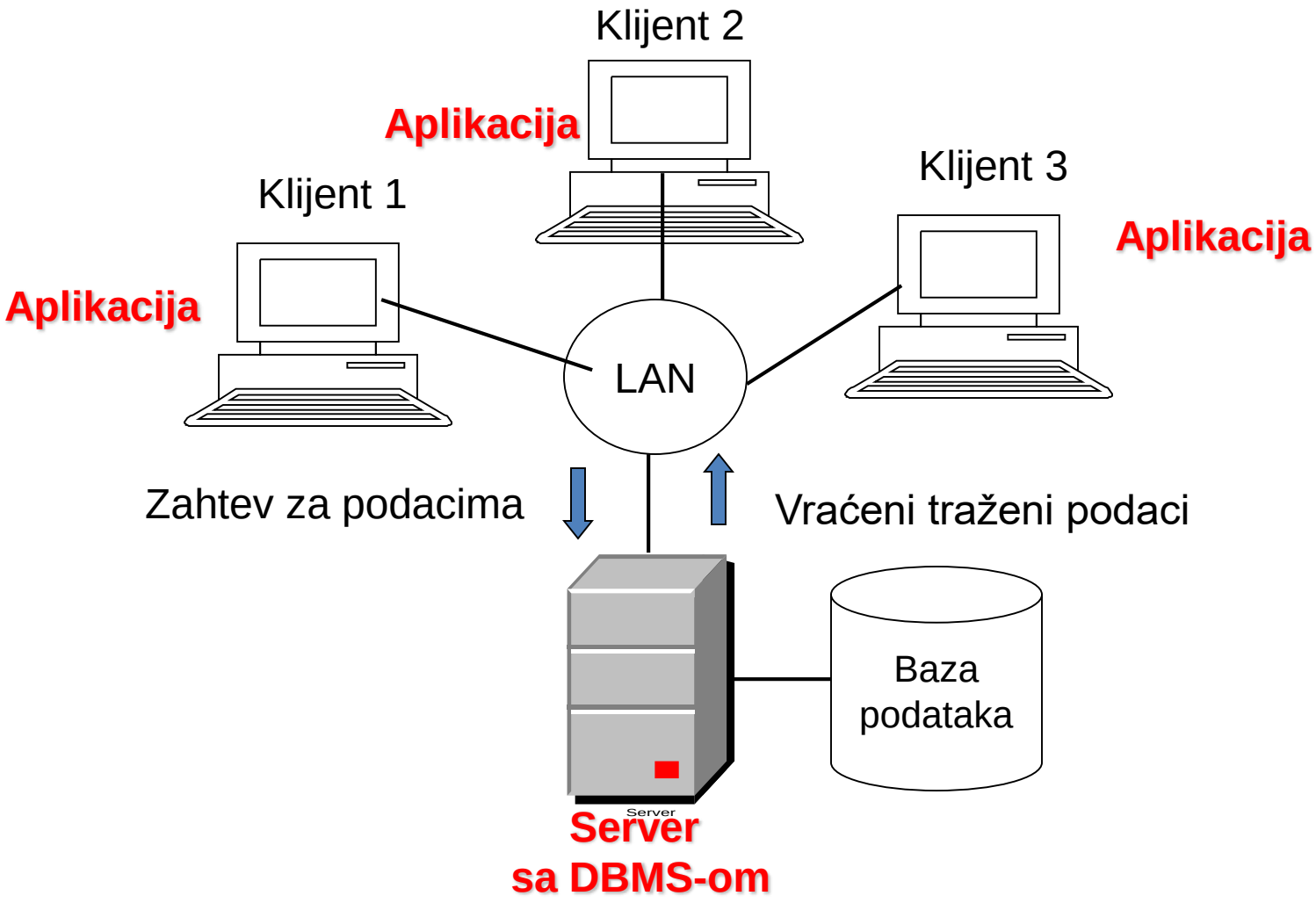
Klijent-server sistemi

- **DBMS Klijent**
  - Prihvata zahteve od strane aplikacije
  - Pakuje ih u poruke i
  - Šalje serveru BP
  - Prihvata odgovore
  - Prosleđuje ih aplikaciji
- **DBMS Server**
  - DBMS funkcionalnost



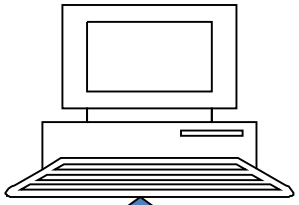
- Odvajanje klijenta i servera omogućava da upravljanje podacima i interakcija sa korisnikom bude izvršavano na različitim procesorima

Implementacija tradicionalne klijent-server arhitekture u dva nivoa (two-tier)



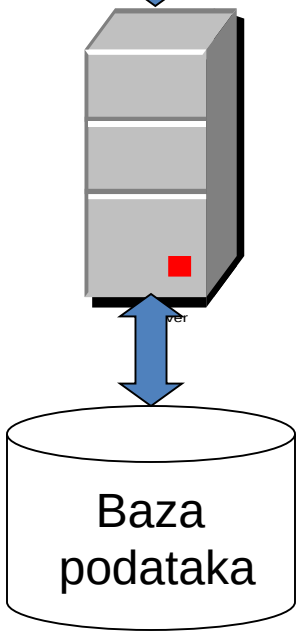
Dva nivoa (two-tier)

Prvi nivo
Klijent



Zadaci
- Korisnički interfejs
- Poslovna logika
- Logika obrade podataka

Drugi nivo
Server baze podataka
(sa DBMS-om)



Zadaci
- Validacija podataka
- Pristup podacima

Dvoslojna klijent-server arhitektura

- Klijent je odgovoran za prezentaciju podataka korisnicima
- Server obezbeđuje servise podataka klijentima
  - Podaci mogu dolaziti od relacionog DBMS-a, objektno-relacionog DBMS-a, objektno-orijentisanog DBMS-a, "legacy" DBMS-a ili od nekog sistema za pristup podacima
- Klijent se obično izvršava na PC-u i umrežen je sa centralizovanim serverom baze podataka
- Tipična interakcija između klijenta i servera:
  - Klijent prihvata zahtev korisnika, proverava sintaksu, generiše zahteve bazi podataka u formi SQL-a ili nekog drugog jezika koji odgovara aplikacionoj logici, šalje poruku serveru i čeka na odgovor
  - Server prihvata zahtev od klijenta i obrađuje ga, a zatim odgovor šalje natrag klijentu
    - Obrada obuhvata proveru autorizacije, obezbeđenje integriteta, kontrolu konkurencije i oporavka, održavanje sistemskog kataloga, pretraživanje i ažuriranje baze podataka

Prednosti dvoslojne klijent server arhitekture

- Omogućava širi pristup postojećim bazama podataka
- Poboljšava performanse
  - Klijentski i serverski proces se mogu paralelno izvršavati na različitim mašinama
  - Mogu se bolje podesiti performanse servera pošto su mu zadaci preciznije formulisani
- Redukuju se troškovi hardvera
  - Samo server treba da bude dovoljno moćan da upravlja bazama podataka
- Redukuju se komunikacioni troškovi
  - Deo operacija se izvodi na klijentu, pa se kroz mrežu šalju samo zahtevi bazi podataka
- Veća je konzistentnost podataka
  - Server rukuje proverom integriteta, dok su aplikacije oslobođene
- Sasvim prirodno se preslikava na arhitekturu otvorenih sistema

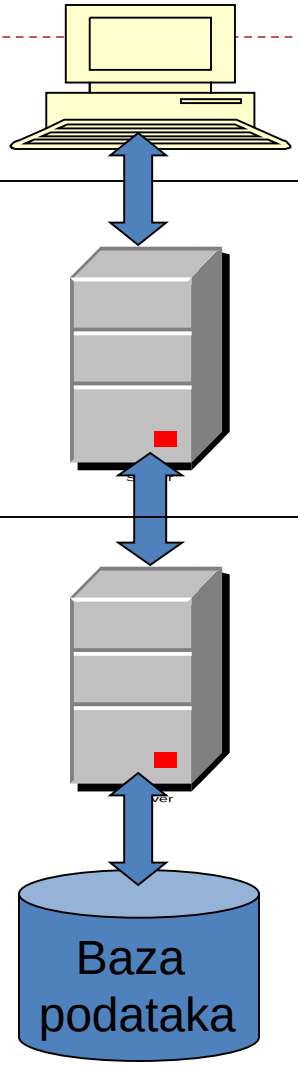
Nedostaci dvoslojne klijent-server arhitekture

- "Debeo" klijent iziskuje više resursa na klijentskim mašinama (moćniji CPU, veći kapacitet RAM-a i diskova)
- Značajni troškovi administriranja na klijentskoj strani
- Ovi nedostaci su otklonjeni kod troslojne klijent-server arhitekture

Tri nivoa (three-tier)

<u>Prvi nivo</u>
Klijent

<u>Zadaci</u>
• Korisnički interfejs



<u>Drugi nivo</u>
Aplikacioni server

<u>Zadaci</u>
• Poslovna logika
• Logika obrade podataka

<u>Treći nivo</u>
Server baze podataka
(sa DBMS-om)

<u>Zadaci</u>
• Validacija podataka
• Pristup podacima

Troslojna klijent server arhitektura

- Klijent je odgovoran jedino za korisnički interfejs aplikacije i može izvoditi samo prostu logiku obrade (npr. Validaciju ulaznih podataka)
  - Radi se o "tankom" klijentu
- Poslovna logika je smeštena u srednjem sloju koji je fizički povezan sa klijentom i serverom baze podataka preko LAN-a i WAN-a
  - Jedan aplikacioni server se projektuje za više klijenata

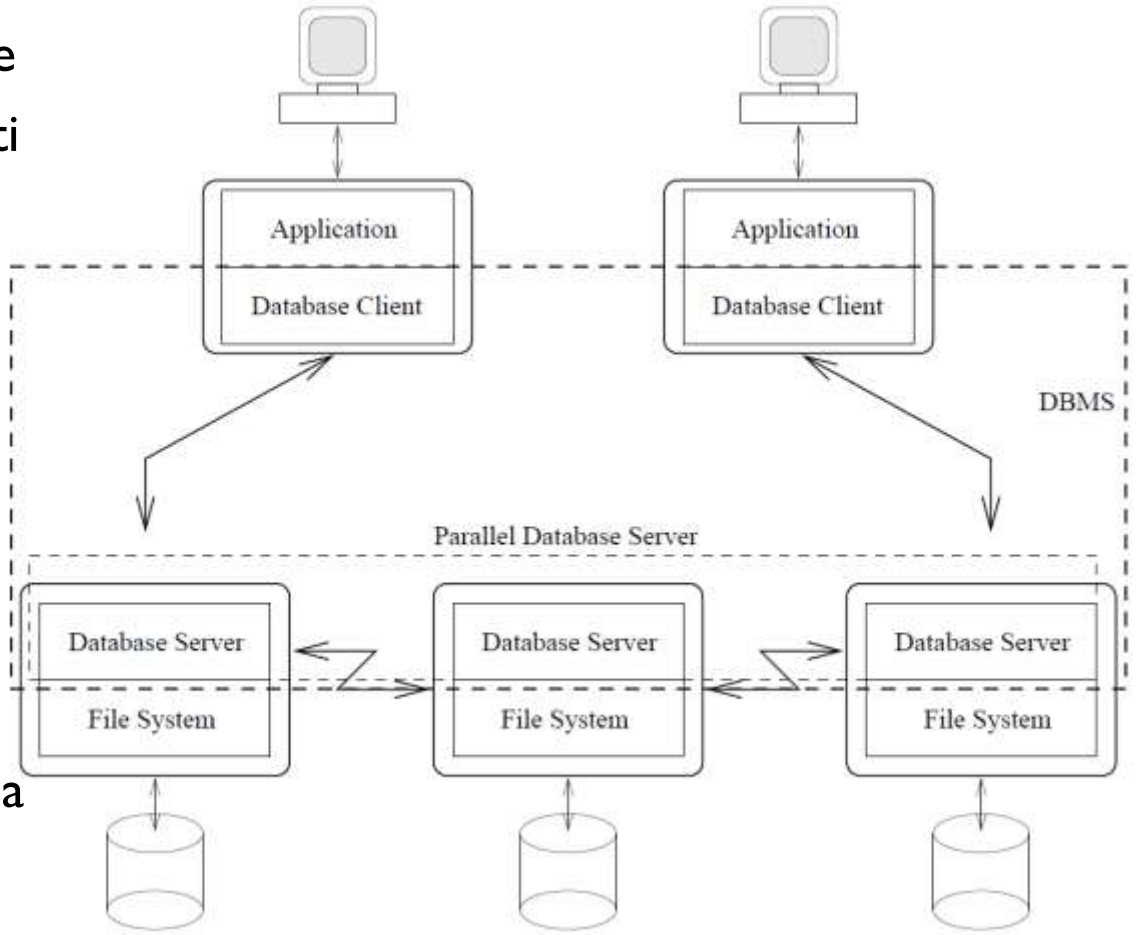
Prednosti troslojne klijent server arhitekture

- Manji troškovi za hardver (klijenti su tanki)
- Održavanje aplikacije je centralizovano
  - Gro poslovne logike je na jednom mestu (aplikacionom serveru) umesto na više klijenata
  - Nema distribucije softvera, što je bio izvor mnogih problema i troškova
- Dodatna modularnost omogućava lakšu zamenu jednog sloja bez uticaja na ostale
- Lakše je balansiranje opterećenja pošto je poslovna logika odvojena od funkcija baze podataka
- Sasvim prirodno se preslikava na **Web okruženje**
  - Klijent je **Web browser**
  - Aplikacioni server je **Web server**
- Pogodna je za Internet i intranet okruženje

Paralelni/distribuirani server BP

- Podaci su distribuirani
- Relaciije mogu biti fragmetisane
- Relacije (ili fragmenti) mogu biti replicirane na nekoliko mesta

- **Serveri-komponente treba da rade zajedno**
  - Postoji jedna zajednička šema
  - Distribucija podataka je transparentna
  - Replikacija je transparentna
  - Fragmentacija je transparentna



Paralelni u odnosu na Distribuirani server

- **Paralelni server BP**
  - Serveri se nalaze blizu jedan drugog
  - Mora da postoji brza, stalna komunikacija između servera, obično preko LANa ili deljive memorije
  - Upiti se obično obrađuju kooperativno od svih servera

- **Distribuirani server BP**
  - Serveri mogu da se nalaze bilo gde
  - Server-to-server komunikacija može da bude spora, obično preko WANa
  - Upiti se obrađuju obično na jednom od servera

Horizontalna fragmentacija

Kompletna relacija

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
2	Kmart	Ottawa	671.05
3	Eatons	Toronto	301.00
4	The Bay	Ottawa	162.99

Horizontalna fragmetacija relacije (na 2 lokacije):

Lokacija 1: Ottawa

Vno	Vname	City	Vbal
2	Kmart	Ottawa	671.05
4	The Bay	Ottawa	162.99

Lokacija 2: Toronto

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
3	Eatons	Toronto	301.00

Vertikalna fragmentacija

Kompletna relacija

Vno	Vname	City	Vbal
1	Sears	Toronto	200.00
2	Kmart	Ottawa	671.05
3	Eatons	Toronto	301.00
4	The Bay	Ottawa	162.99

Vertikalna fragmetacija relacije (na 2 lokacije):

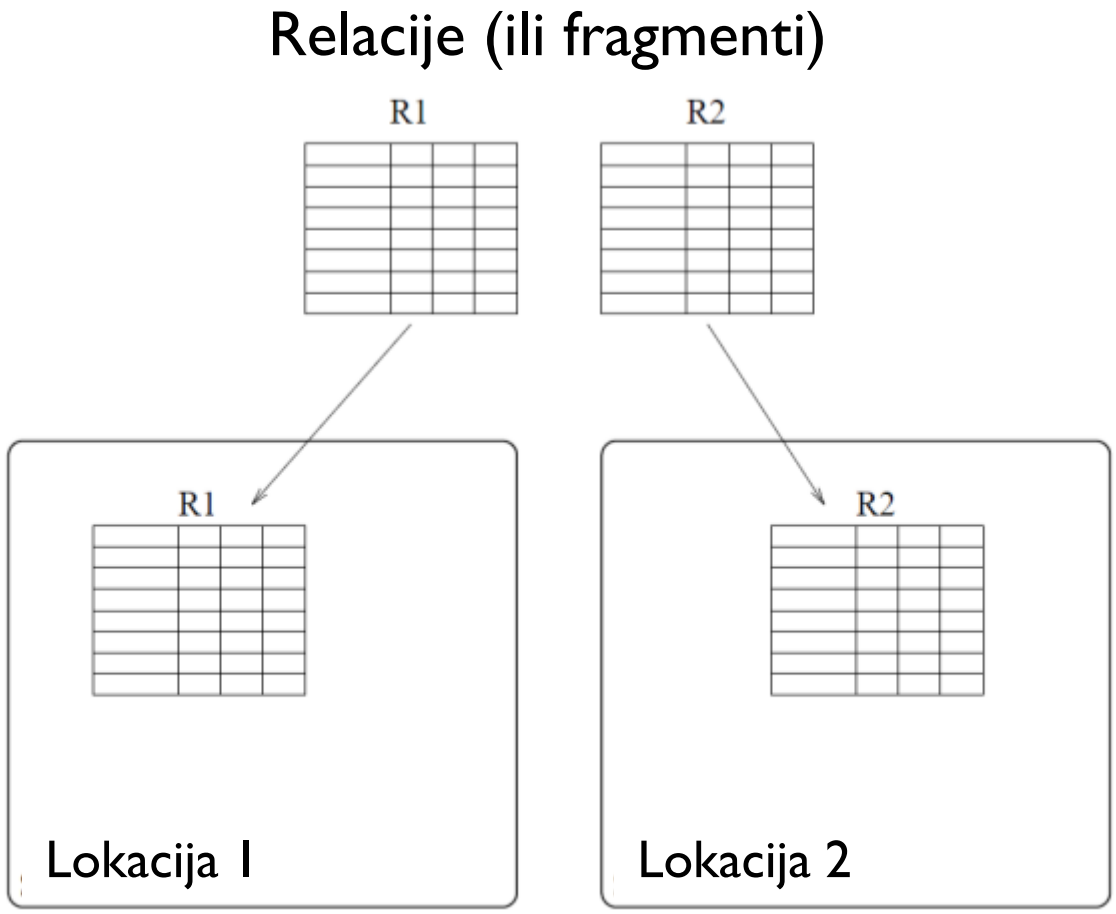
Lokacija 1

Vno	Vname	City
1	Sears	Toronto
2	Kmart	Ottawa
3	Eatons	Toronto
4	The Bay	Ottawa

Lokacija 2

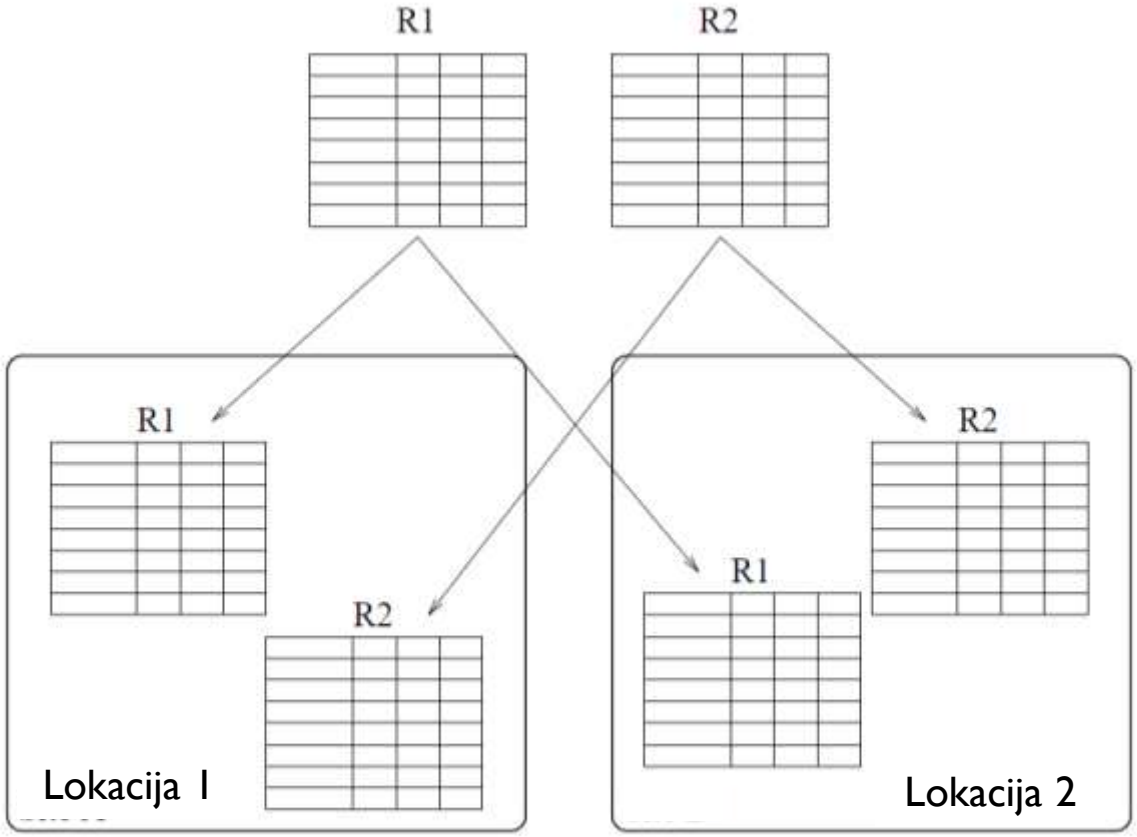
Vno	Vbal
1	200.00
2	671.05
3	301.00
4	162.99

Distribucija podataka



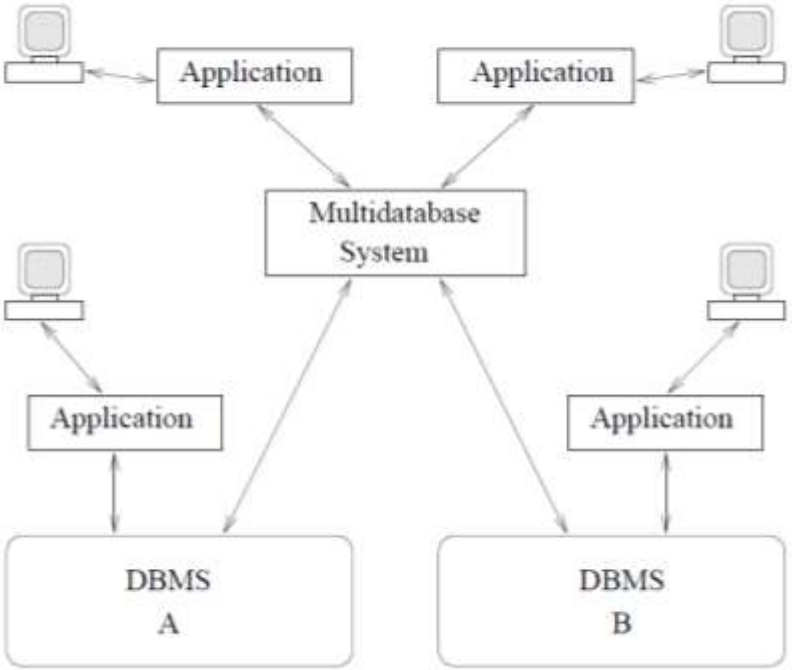
Replikacija podataka

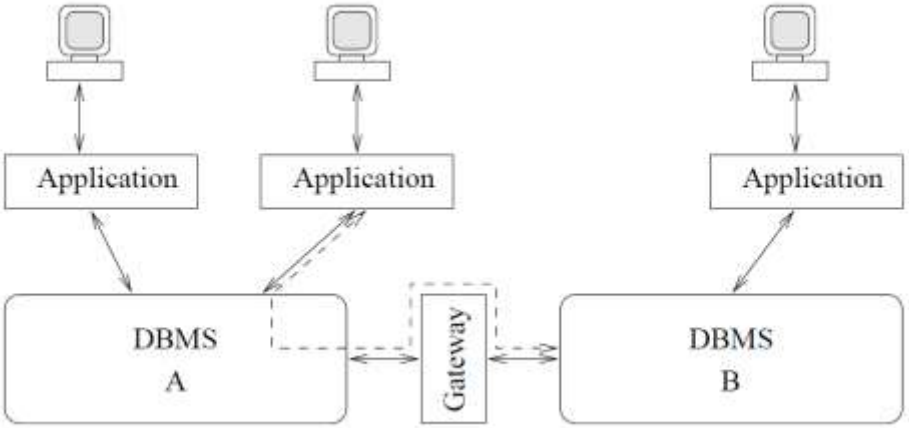
Relacije (ili fragmenti)



Sistemi multi-baza podataka

- Aplikacije "vide" jedan sistem BP
- Sistemi BP su autonomni
- Pristup može da ide i preko komponente tzv "gateway"





Arhitekture sistema baza podataka

Pitanja ???