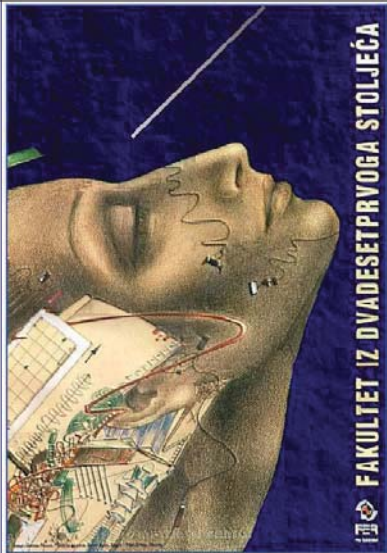


Baze podataka

Predavanja

1. Uvod

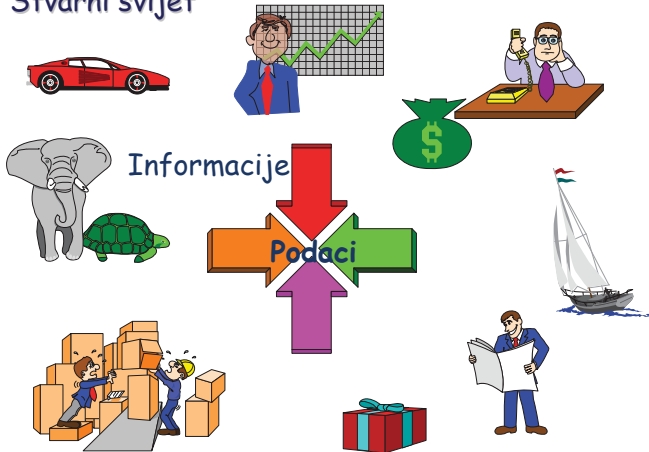
veljača 2008.



"Način na koji prikupljate informacije, upravljate njima i koristite ih, odredit će hoćete li pobijediti ili izgubiti."

Bill Gates

Stvarni svijet



Obrada podataka – sveučilište, knjižnica

- Sveučilište
 - podaci o studentima
 - podaci o nastavnicima
 - uspjeh na ispitu
- ISVU - Informacijski sustav visokih učilišta RH
- Knjižnica
 - podaci o korisniku knjižnice
 - podaci o knjigama
 - traženje knjiga i časopisa
 - posuđivanje knjiga



Obrada podataka - bankarstvo



- otvaranje računa
- novčane transakcije
- praćenje stanja na računu
- praćenje kupnji ostvarenih putem kreditnih kartica

Obrada podataka - telekomunikacije



- podaci o pozivima
- telefonski računi
- podaci o mreži
- podaci o kvarovima



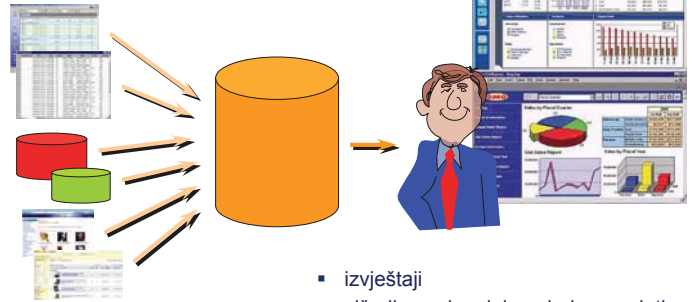
Obrada podataka – portali



- prikaz podataka iz različitih izvora

Obrada podataka – potpora odlučivanju

- integracija podataka iz različitih izvora



- izvještaji
- višedimenzionalni pogled na podatke
- dubinska analiza (data mining)
- vizualizacija podataka

Organizacija predmeta

Baze podataka

Ciljevi predmeta i očekivani ishod učenja

- Ovo je osnovni kolegij iz područja baza podataka kojemu je cilj upoznati studente sa sustavima za upravljanje bazama podataka, relacijskim modelom i relacijskim bazama podataka. Izučava se način oblikovanja relacijskih baza podataka i oblikovanje modela entiteti-veze, relacijska algebra, upitni jezik SQL i osnove zaštite baza podataka.
- Studenti će biti osposobljeni za modeliranje jednostavnijih baza podataka i postavljanje srednje složenih upita nad bazom podataka. Upoznat će se s osnovama zaštite baza podataka.

Predavači

- | | |
|------|-------------------------------------|
| 2.R1 | Doc.dr.sc. Boris Vrdoljak, ZPR |
| 2.R2 | Doc.dr.sc. Slaven Zakošek, ZPR |
| 2.R3 | Prof.dr.sc. Mirta Baranović, ZPR |
| 2.R4 | Doc.dr.sc. Jadranka Pečar-Ilić, IRB |
| 2.R6 | Prof.dr.sc. Zoran Skočir, ZOEM |
| | Dr.sc. Marko Banek, ZOEM |

IRB – Institut "Ruđer Bošković"

ZOEM – Zavod za osnove elektrotehnike i električna mjerenja, zgrada C, VI kat

ZPR – Zavod za primijenjeno računarstvo, zgrada D, III kat



Asistenti

2.R1	Mr.sc. Damir Pintar, ZOEM Mr.sc. Mihaela Vranić, ZOEM
2.R2	Mr.sc. Igor Mekterović, ZPR
2.R3	Mr.sc. Ljiljana Brkić, ZPR Lidia Rovin, dipl. ing., ZPR
2.R4	Mr.sc. Krešimir Križanović, ZPR
2.R6	Dr.sc. Marko Banek, ZOEM Mr.sc. Damir Jurić, ZOEM

ZOEM – Zavod za osnove elektrotehnike i električna mjerenja, zgrada C, VI kat
ZPR – Zavod za primijenjeno računarstvo, zgrada D, III kat

Administracija

- ZPR – zgrada D, III kat - gđa. Sonja Majstorović
- ZOEM – zgrada C, VI kat - gđa. Jasenka Haladin

Literatura

- J. D. Ullman, J. Widom: **A First Course in Database Systems**, Prentice-Hall, 2001.
- A. Silberschatz, H.F. Korth, S. Sudarshan: **Database Systems Concepts**, 5th Edition, McGraw-Hill, 2005.
- C.J. Date: **An Introduction to Database Systems**, 8th Edition, Addison Wesley, 2003.
- T. M. Connolly, C. E. Begg: **Database Systems: A Practical Approach to Design, Implementation, and Management**, Addison Wesley, 2004.
- S. Tkalac, **Relacijski model podataka**, DRIP, 1988.
- M. Varga: **Baze podataka**, DRIP, 1994.

Upute i obavijesti

- na URL stranici predmeta

<http://www.fer.hr/predmet/bazepod>

- forum na stranici predmeta služi isključivo za međusobnu komunikaciju studenata

Organizacija nastave

PREDAVANJA

- *Powerpoint* prezentacije
 - datoteke u PDF formatu nalazit će se u repozitoriju datoteka na URL stranici predmeta
- Odabrani primjeri prikazivat će se pomoću prikladnih programskih alata

Organizacija nastave

SAMOSTALAN RAD

- Učenje (slajdovi s predavanja i ostala literatura)
- Vježbanje, rješavanje domaćih zadaća
 - ili rad na vlastitom računalu (kod kuće, u domu, ...)
 - podrazumijeva mogućnost i sposobnost instaliranja potrebnih programskih sustava i alata na vlastitom računalu (na raspolaganju će biti detaljne upute)
 - ili rad u nadziranom laboratoriju u terminima koji će biti definirani naknadno
- provest će se anketa

Interakcija među sudionicima nastave

KONZULTACIJE

- Asistenti
 - unaprijed određeni termini konzultacija
 - e-mail (ime.prezime@fer.hr)
- Predavači
 - unaprijed određeni termini konzultacija
 - e-mail (ime.prezime@fer.hr)
- Termini će biti objavljeni na stranici predmeta.

Elementi ocjenjivanja

- aktivna prisutnost na predavanjima ⇒ 5 bodova
- rješavanje 6 domaćih zadaća ⇒ 6 bodova
- kontrolne zadaće 6 x 4 boda ⇒ 24 boda
- 1. međuispit ⇒ 15 bodova
- 2. međuispit ⇒ 20 bodova
- završni (ili ponovljeni završni) ispit ⇒ 30 bodova
- MOGUĆI BROJ BODOVA UKUPNO ⇒ 100 bodova
- Za prolaznu ocjenu potrebno je zadovoljiti **dva uvjeta**:
 - ostvariti ukupno ≥ 50 bodova
 - ostvariti ≥ 10 bodova na završnom (ili ponovljenom završnom) ispit

Međuispiti i završni ispiti

- Na 1. međuispitu provjerava se znanje gradiva 1. nastavne cjeline
- Na 2. međuispitu provjerava se znanje gradiva **1. i 2. nastavne cjeline**
- Na završnom ispitu i ponovljenom završnom ispitu provjerava se znanje **čitavog gradiva**

Ankete

- Studenti sudjeluju u vrednovanju kvalitete nastave i nastavnika putem ankete. Anketiranje se provodi za svaki predmet tri puta tijekom semestra.

Članak 20. stavak 5. Pravilnika o sveučilišnom preddiplomskom i diplomskom studiju na Sveučilištu u Zagrebu, Fakultetu elektrotehnike i računarstva

Uvod u baze podataka

Organizacijski sustav

- ORGANIZACIJSKI SUSTAV je složeni sustav koji sadrži tehničke i humane podsustave
 - poduzeće, ustanova, djelatnost, društvena organizacija, tehnički sustav kao npr. telekomunikacijska mreža i sl.
 - često se organizacijski sustav naziva i POSLOVNIM SUSTAVOM, iako pojam organizacijski sustav ima nešto šire značenje.
- Primjeri organizacijskih sustava:
 - Knjižnica
 - Sveučilište
 - Zračna luka

Informacija, podatak (*Information, Data*)

- INFORMACIJA je sadržaj koji primatelju opisuje nove činjenice.
- Taj sadržaj se materijalizira u obliku PODATAKA koji služe za prikaz informacija u svrhu spremanja, prijenosa i obrade.
- Podatak je skup simbola (znakova).
- Informacija je i obrađeni podatak koji za primatelja ima karakter novosti, otklanja neizvjesnost i služi kao podloga za odlučivanje.
 - Podatak izvan konteksta nema značenja
 - podatak: 4.62
 - Podatak koji interpretiramo i primjereno povežemo predstavlja informaciju
 - informacija: prosjek svih ocjena studenta Marka Horvata na studiju na FER-u u ovom trenutku je 4.62

Informacijski sustav (*Information System*)

Definicija:

- Ukupna infrastruktura, organizacija, osoblje i komponente koje služe za prikupljanje, obradu, pohranu, prijenos, prikaz, širenje i raspolaganje informacijama
- *The entire infrastructure, organization, personnel, and components for the collection, processing, storage, transmission, display, dissemination, and disposition of information* [INFOSEC-99].

Informacijski sustav (*Information System*)

- Informacijski sustav je dio svakog organizacijskog sustava
- Svrha mu je prikupljanje, obrada, pohranjivanje i distribucija informacija, koje su potrebne za praćenje rada i upravljanje organizacijskim sustavom ili nekim njegovim podsustavom.
 - Informacijski sustav je aktivni sustav koji može (ali ne mora) koristiti suvremenu informacijsku tehnologiju
 - Središnji dio informacijskog sustava je BAZA PODATAKA

Informacijski sustav, 1868.



Baza podataka (*Database*)

- BAZA PODATAKA je skup podataka koji su pohranjeni i organizirani tako da mogu zadovoljiti zahtjeve korisnika.
(M. Vetter, 1981.)
- BAZA PODATAKA je skup međusobno povezanih podataka, pohranjenih zajedno, uz isključenje bespotrebne zalihosti (redundancije), koji mogu zadovoljiti različite primjene. Podaci su pohranjeni na način neovisan o programima koji ih koriste. Prilikom dodavanja novih podataka, mijenjanja i pretraživanja postojećih podataka primjenjuje se zajednički i kontrolirani pristup. Podaci su strukturirani tako da služe kao osnova za razvoj budućih primjena.

(J. Martin, 1979.).

Entitet (*Entity*)

- bilo što, što ima suštinu ili bit i posjeduje značajke s pomoću kojih se može razlučiti od svoje okoline
- osobe: studenti, radnici, građani, ...
 - student Horvat Ivan (0036123456)
- ostala bića
 - Gupčeva Lipa, pas Lajka
- objekti: vozila, strojevi, uređaji, ulice, zgrade, mjesta, ...
 - Eiffelov toranj, cepelin Hindenburg
- apstraktni pojmovi: boje, predmeti nastavnog programa, ...
 - predmet Baze podataka u nastavnom programu FER-2
- događaji (nešto se desilo, dešava se ili se planira da će se desiti)
 - dana 24.11.2006. održava se proslava Dana Fakulteta na FER-u
- povezanost među objektima, osobama, događajima, ...
 - student Horvat Ivan (0036123456) **stanuje u** Zagrebu
- nešto o čemu želimo prikupljati i pohranjivati podatke

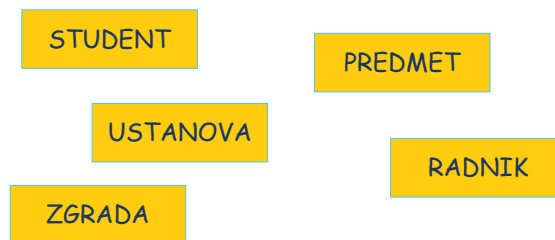
Atribut (*Attribute*)

- Entitet posjeduje neka SVOJSTVA ili ATTRIBUTE koji ga karakteriziraju.
 - Za Informacijski sustav visokih učilišta (ISVU) važna svojstva studenta Horvat Ivana su:
 - Matični broj studenta (JMBAG)
 - Ime
 - Prezime
 - Datum rođenja, ...
- Izbor svojstava (atributa) koje ćemo pratiti ovisi o namjeni informacijskog sustava
 - Horvat Ivan u informacijskom sustavu MUP-a bit će karakteriziran i atributima:
 - Boja kose
 - Boja očiju
 - Otisak prsta, ...

Skup entiteta (*Entity Set*)

- Slični entiteti se svrstavaju u skupove entiteta
- Slični su oni entiteti kojima se promatraju ista svojstva
- Svi entiteti koji su članovi istog skupa entiteta imaju iste atribute
- "atributi entiteta" \Leftrightarrow "atributi skupa entiteta"
- atributi skupa entiteta PREDMET
 - sifPred
 - nazPred
 - ectsBod
 - nastProg
- atributi skupa entiteta STUDENT
 - jmbag
 - ime
 - prezime
 - datRod

Skupovi entiteta - primjer



Domena i vrijednost atributa (*Domain, Attribute Value*)

- Za svaki entitet, atribut poprima vrijednosti iz određenog skupa vrijednosti koji predstavlja **domenu** tog atributa
- domene atributa za skup entiteta PREDMET
 - sifPred: skup šifara predmeta - cijelih brojeva iz intervala [1,999999]
 - nazPred: skup naziva predmeta - nizova znakova duljine do 80 znakova
 - ectsBod: skup vrijednosti ECTS bodova - realnih brojeva iz intervala [0.5, 20.0] (s jednom znamenkom iza decimalne točke)
 - nastProg: skup oznaka nastavnih programa - nizova znakova duljine do 5 znakova
- vrijednosti atributa za entitet Baze podataka (31503)
 - sifPred: 31503
 - nazPred: Baze podataka
 - ectsBod: 6.0
 - nastProg: FER-2

Skup entiteta - prikaz u obliku tablice

Skup entiteta PREDMET

atributi			
sifPred	nazPred	ectsBod	nastProg
31503	Baze podataka	6.0	FER-2
1228	Baze podataka	5.0	FER-1
19670	Matematika 1	7.0	FER-2
21006	Fizika 1	6.0	FER-2
90	Baze podataka	5.0	ETF-4

vrijednosti atributa

Identifikatori entiteta, ključevi

- Skupove atributa čije vrijednosti jednoznačno određuju entitet u promatranom skupu entiteta (dakle ne postoje dva entiteta s posve istim vrijednostima tih atributa) nazivamo IDENTIFIKATORIMA ili KLJUČEVIMA SKUPA ENTITETA.
- **Primjer:** u skupu entiteta PREDMET prikazanom na slici:

sifPred	nazPred	ectsBod	nastProg
31503	Baze podataka	6.0	FER-2
1228	Baze podataka	5.0	FER-1
19670	Matematika 1	7.0	FER-2
21006	Fizika 1	6.0	FER-2
90	Baze podataka	5.0	ETF-4

- skup atributa { sifPred } **jest** ključ skupa entiteta
- skup atributa { nazPred } **nije** ključ skupa entiteta
- skup atributa { nazPred, nastProg } **jest** ključ skupa entiteta

Modeliranje stvarnog svijeta

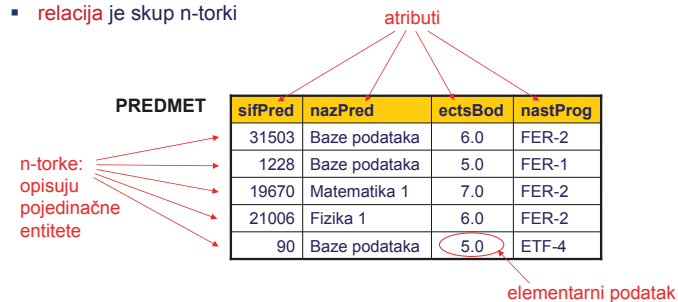
- Modeliranje stvarnog svijeta predstavlja preslikavanje stvarnog svijeta u oblik pogodan za računalnu obradu;
- Baza podataka nekog informacijskog sustava predstavlja sliku stvarnog organizacijskog sustava;
- Stvarni svijet, zbog njegove složenosti, ne možemo prikazati sa svim detaljima;
- Stvarni svijet predstavlja se pojednostavnjenim, nadomjesnim modelom;
- Model stvarnog svijeta predstavlja se uz pomoć nekog formalnog sustava;
- **Model podataka** je formalni sustav koji koristimo kod modeliranja baza podataka

Model podataka (Data Model)

- Model podataka je formalni sustav koji se sastoji od:
 - skupa objekata - osnovnih elemenata (koncepta) baze podataka
 - skupa operacija koje se provode nad objektima
 - skupa integritetskih ograničenja (*integrity constraints*)
 - implicitno ili eksplicitno definiraju skup konzistentnih stanja podataka, promjena stanja, ili oboje
- Povijesni razvoj modela podataka:
 - Hijerarhijski model
 - Mrežni model
 - Relacijski model
 - ER model
 - Objektni model
 - Objektno-relacijski model

Relacijski model podataka – objekti

- elementi skupa objekata u relacijskom modelu podataka su **relacije**
- relacija** je skup n-torki



- shema relacije** obuhvaća naziv relacijske sheme (PREDMET) i skup atributa: (sifPred, nazPred, ectsBod, nastProg)

Relacijski model podataka – operacije

- operacija "selekcija" u relacijskom modelu podataka:

predmet

sifPred	nazPred	ectsBod	nastProg
31503	Baze podataka	6.0	FER-2
1228	Baze podataka	5.0	FER-1
19670	Matematika 1	7.0	FER-2
21006	Fizika 1	6.0	FER-2
90	Baze podataka	5.0	ETF-4

$\sigma_{ectsBod=6.0}$ (predmet)

sifPred	nazPred	ectsBod	nastProg
31503	Baze podataka	6.0	FER-2
21006	Fizika 1	6.0	FER-2

- ostale operacije u relacijskom modelu podataka
 - unija, razlika, presjek, projekcija, ...

Relacijski model podataka – integritetska ograničenja

- pravilo domenskog integriteta u relacijskom modelu podataka:

predmet

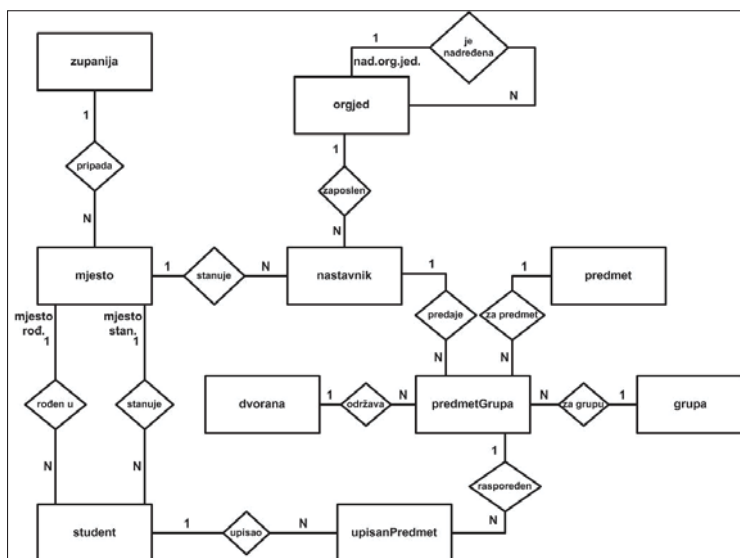
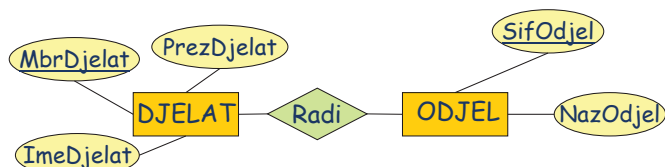
sifPred	nazPred	ectsBod	nastProg
31503	Baze podataka	6.0	FER-2
1228	Baze podataka	5.0	FER-1
19670	Matematika 1	7.0	FER-2
21006	Fizika 1	6.0	FER-2
90	Baze podataka	5.0	ETF-4

- domenski integritet:
 - vrijednost atributa sifPred mora biti iz intervala [1, 999999]
 - vrijednost atributa ectsBod mora biti iz intervala [0.5, 20.0]
 - ...
- ostala integritetska ograničenja u relacijskom modelu podataka:
 - entitetski, referencijski, ...

ER model podataka

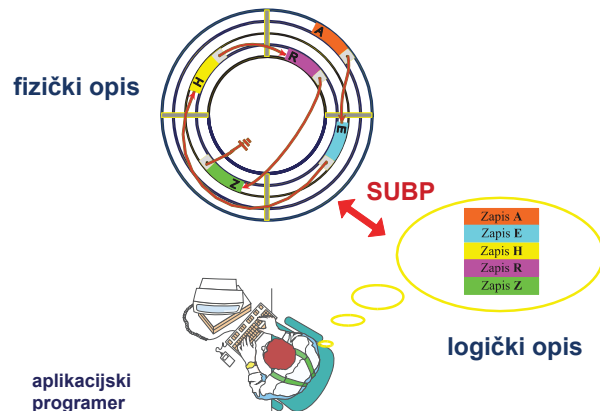
Model entiteti-veze
Entity-Relationship Model

- Postrelacijski model
- Zadržava dobra svojstva relacijskog modela
- Objekti ER modela su entiteti i njihove međusobne veze

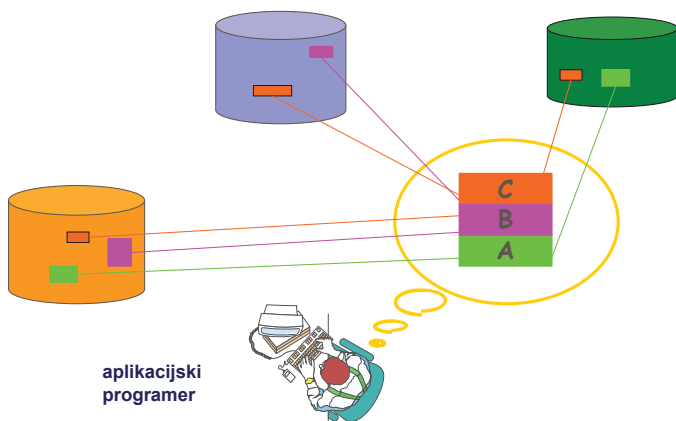


Arhitektura baze podataka

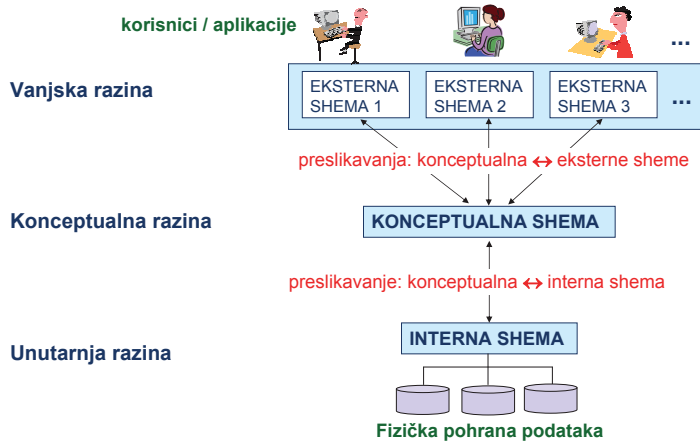
Fizička i logička organizacija podataka



Fizička i logička organizacija podataka



Arhitektura baze podataka

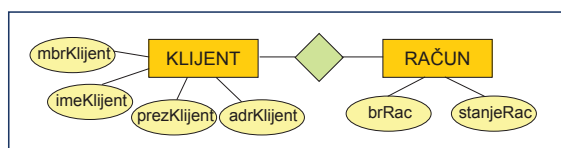


Arhitektura baze podataka

- Schema (struktura) baze podataka se opisuje na tri razine apstrakcije:
 - Na konceptualnoj razini opisuje se
 - KONCEPTUALNA SHEMA**
 - Na unutarnjoj razini opisuje se
 - INTERNA SHEMA**
 - Na vanjskoj razini opisuju se
 - EKSTERNE SHEME**
- Jedna baza podataka ima jednu konceptualnu, jednu internu i (najčešće) više eksternih shema
- Schema baze podataka se *relativno* rijetko mijenja
- Sadržaj ili instanca baze podataka (skup svih podataka baze podataka u određenom trenutku) se ČESTO mijenja

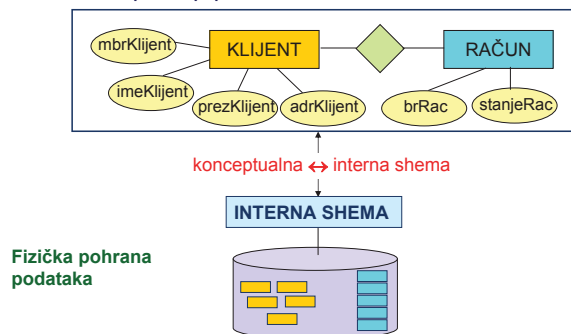
Konceptualna shema

- često se koristi i naziv LOGIČKA SHEMA
- sadrži opis svih entiteta i veza, atributa, domena i integritetska ograničenja
- konceptualna shema se može opisati korištenjem modela podataka, npr. relacijskog ili ER modela



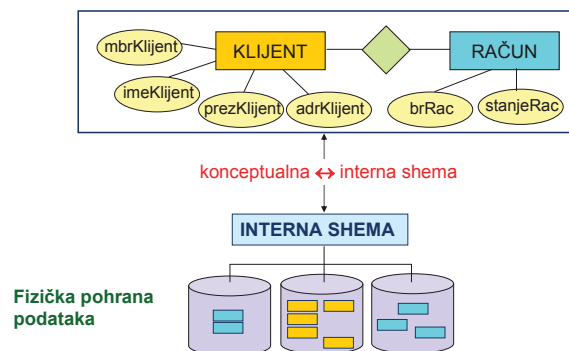
Interna shema

- opisuje detalje fizičke strukture pohrane i metode pristupa podacima: kako su podaci pohranjeni i koje se metode koriste za pristup podacima



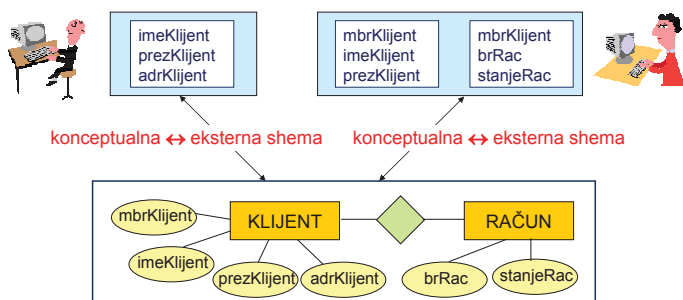
Fizička nezavisnost podataka

- izmjena interne sheme ne utječe na konceptualnu shemu



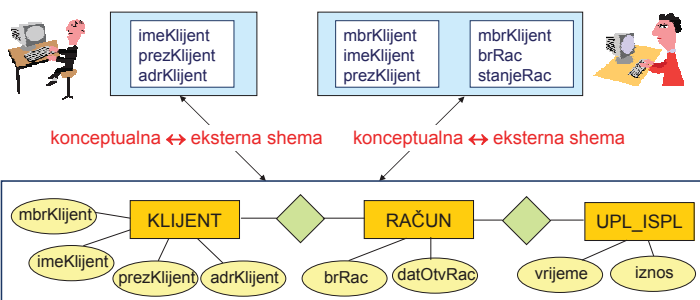
Eksterna shema

- eksterna shema opisuje "pogled" na dio baze podataka koji je namijenjen specifičnoj grupi korisnika
- osnova za opis eksternih shema je konceptualna shema



Logička nezavisnost podataka

- izmjena konceptualne sheme ne mora izazvati izmjenu eksternih shema → izmjena konceptualne sheme ne utječe na korisnike i aplikacijske programe koji ih koriste



Sustav za upravljanje bazom podataka - SUBP

Database Management System - DBMS

Sustav za upravljanje bazom podataka

Sustav za upravljanje bazom podataka - SUBP (*Database Management System - DBMS*) je **programski sustav** koji omogućava upravljanje podacima u bazi podataka. SUBP se temelji na odabranom modelu podataka.

Prema modelu podataka na kojem se temelje, **SUBP**-ove dijelimo na:

- hijerarhijske,
- mrežne,
- relacijske,
- objektno-relacijske,
- objektno-orijentirane.

Sustav za upravljanje bazom podataka

- sakriva od korisnika detalje fizičke pohrane podataka
- osigurava logičku i fizičku nezavisnost podataka
- omogućuje definiciju i rukovanje podacima
 - DDL - Data Definition Language
 - DML - Data Manipulation Language
- obavlja funkciju zaštite podataka
 - integritet podataka
 - pristup podacima - autorizacija, sigurnost
 - kontrola paralelnog pristupa
 - obnova u slučaju razrušenja
- obavlja optimiranje upita

Jezici baze podataka

- DDL (*Data Definition Language*)
 - omogućava imenovanje i opis entiteta, atributa, veza i pripadnih ograničenja integriteta i pravila sigurnosti
 - koristi se za definiranje nove sheme baze podataka ili modificiranje postojeće
 - obavljanje DDL operacija rezultira izmjenom sadržaja rječnika podataka (metapodataka)
- DML (*Data Manipulation Language*)
 - omogućava korištenje skupa operacija za rukovanje podacima u bazi podataka
 - upitni jezik (*query language*)
 - ne sasvim korektno, pojam se koristi ne samo za operacije dohвата podataka, već i za operacije izmjene, brisanja i unosa podataka
 - proceduralni jezici, neproceduralni jezici

Zašto koristimo sustave za upravljanje bazama podataka?

Zašto ne bismo koristili samo datotečne sustave?

Zašto SUBP, a ne samo datoteke?

1. Jednostavan pristup podacima

Datotečni sustav - podaci su raspršeni po datotekama koje mogu biti u različitim formatima. Za dobivanje odgovora na netipična pitanja potrebno je ili dotjerivati stare ili pisati nove programe. Primjer:

Potrebno je naći sve korisnike banke koji žive u području s određenim poštanskim brojem. Pretpostavimo da ne postoji gotovi program koji može izdvojiti tražene korisnike, ali postoji program koji vraća sve korisnike. Dva su moguća rješenja:

- a) službenik će iz popisa svih korisnika ručno izdvojiti one s određenim poštanskim brojem;
- b) tražit će se od programera da napiše odgovarajući program. Niti jedno od ova dva rješenja nije zadovoljavajuće. Svaki put kad se pojavi nova vrsta upita, opet će se morati ponoviti postupak

- SUBP omogućuje fleksibilniji dohvat i izmjenu podataka.

Zašto SUBP, a ne samo datoteke?

2. Upravljanje zalihostima i nekonzistentnošću

Isti podaci mogu biti pohranjeni u više datoteka, što može voditi ka nekonzistentnosti.

Pretpostavimo da se adrese korisnika bankovnog računa pohranjuju u dvije datoteke (jedna za tekući, a druga za devizni račun). Nakon što je korisnik A promijenio adresu stanovanja, bankovni službenik promijenio je podatak o adresi u jednoj datoteci, ali je zaboravio promijeniti adresu u drugoj.

Ako se mjesečni izvještaji o stanju računa šalju na kućnu adresu korisnika na temelju adrese pohranjene u drugoj datoteci, korisnik A neće dobiti izvještaj na svoju novu adresu.

- SUBP omogućuje bolju kontrolu zalihosti od datotečnog sustava.

Zašto SUBP, a ne samo datoteke?

3. Transakcijska obrada

Treba prebaciti 1000 kuna s računa A na račun B (to je jedna transakcija).

Ako se dogodi hardverska ili softverska pogreška za vrijeme izvođenja programa, moguće je da se 1000 kuna skine s računa A, ali se ne uspije prebaciti na račun B, što dovodi do nekonzistentnosti u bazi podataka. Prebacivanje se mora obaviti u potpunosti.

- SUBP obično ima ugrađenu podršku za transakcijsku obradu, što je vrlo teško ostvariti u datotečnom sustavu.

4. Složeni odnosi među podacima

- SUBP omogućuje predstavljanje različitih odnosa među podacima, definiranje novih odnosa kad se oni pojave te jednostavan dohvat i izmjenu međusobno povezanih podataka

Zašto SUBP, a ne samo datoteke?

5. Istovremeni pristup više korisnika

SUBP omogućuje većem broju korisnika pristup bazi podataka u isto vrijeme. Pri tome treba osigurati da u slučaju kad više korisnika koji nastoje promijeniti isti podatak, to se čini na kontrolirani način tako da rezultat izmjene bude točan i jednoznačan.

Pretpostavimo da službenici dvije turističke agencije nastoje istovremeno rezervirati isto sjedalo u zrakoplovu. SUBP mora osigurati da rezervaciju određenog sjedala u određenom trenutku može obavljati najviše jedan službenik.

U datotečnom sustavu je kontrolu istovremenog pristupa daleko teže provesti.

Zašto SUBP, a ne samo datoteke?

6. Autorizirani pristup

- Kad više korisnika koristi veću bazu podataka, obično neće svim korisnicima biti omogućen pristup svim podacima u bazi.

Na primjer, kako su podaci o financijama povjerljivi, samo će se autoriziranim osobama omogućiti pristup tim podacima.

- Neki korisnici će moći samo pregledavati podatke, dok će ih drugi moći i pregledavati i mijenjati.
- SUBP omogućava određivanje načina na koji će različiti korisnici pristupiti podacima.
- U datotečnom sustavu je jednostavno odrediti je li neka datoteka namijenjena za čitanje, pisanje ili oboje, no nije jednostavno definirati različiti način pristupa podacima za različite korisnike.

Uloge osoba u životnom ciklusu baze podataka



Baze podataka - uloge

1. Projektanti baze podataka

- razgovaraju s korisnicima da bi saznali njihove zahtjeve
- oblikuju bazu podataka prema zahtjevima korisnika definirajući strukturu za pohranu podataka (tj. model baze podataka)

2. Analitičari sustava i programeri aplikacija

- Analitičari sustava prikupljaju zahtjeve korisnika (npr. bankovnih službenika) i pišu specifikacije za razvoj aplikacija za pristup bazi podataka
- Programeri na temelju tih specifikacija izrađuju programe te ih testiraju, dokumentiraju i održavaju; moraju biti upoznati sa svojstvima i mogućnostima SUBP-a

Baze podataka - uloge

3. Administratori baze podataka

- instaliraju i nadograđuju SUBP
- odgovorni su za autorizaciju pristupa bazi podataka
- organiziraju, nadziru i optimiziraju korištenje baze

4. Korisnici

- Pristupaju bazi tako da postavljaju upite, mijenjaju podatke i izrađuju izvještaje

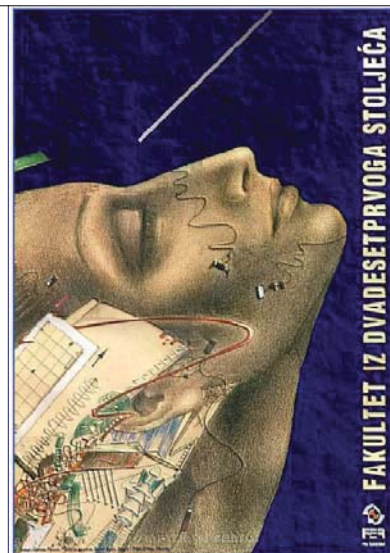
Razlikujemo nekoliko skupina korisnika:

- korisnici koji povremeno pristupaju bazi koristeći upitni jezik
- korisnici koji često koriste bazu postavljajući standardne upite i radeći standardne promjene koristeći programirana sučelja (npr. službenici u banci, turističkim agencijama...)
- sofisticirani korisnici koji su dobro upoznati s bazom podataka i koriste je na složeniji način (npr. inženjeri, znanstvenici, poslovni analitičari)

Baze podataka

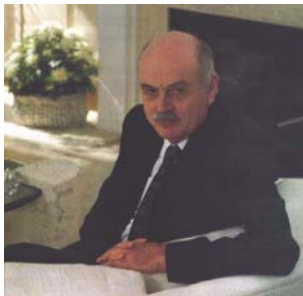
Predavanja
veljača 2008.

2. Relacijski model podataka



Relacijski model podataka

- E. F. Codd: "A Relational Model of Data for Large Shared Data Banks", Comm. ACM 13, No. 6, June 1970.



Dr. Edgar Frank Codd (1923-2003)

Ciljevi relacijskog modela podataka:

- osigurati visoki stupanj nezavisnosti podataka
- postaviti temelje za rješavanje problema semantike, konzistentnosti i redundancije podataka (**normalizacija**)
- omogućiti razvoj DML jezika temeljenih na operacijama nad skupovima

Relacijski model podataka

- Važni projekti u ranim 70-tim: jezik ISBL temeljen na relacijskoj algebri, jezici SQUARE i SEQUEL (DBMS System R) temeljeni na relacijskoj algebri i predikatnom računu te Query-By-Example temeljen na predikatnom računu nad domenama
 - razvojem prototipova dokazuje se praktična upotrebljivost relacijskog modela
 - postavljaju se temelji za rješavanje problema implementacije u područjima upravljanja transakcijama, paralelnog pristupa, obnove, optimizacije upita, sigurnosti i konzistentnosti podataka
- Projekti su potaknuli:
 - razvoj strukturiranog upitnog jezika (SQL)
 - razvoj komercijalnih **relacijskih** sustava za upravljanje bazama podataka (RDBMS)
 - Ingres, Oracle, IBM DB2, Informix, ...
 - danas: u upotrebi je nekoliko stotina različitih RDBMS sustava

Relacijski model podataka

- objekti u relacijskom modelu podataka su RELACIJE

mjesto		
pbr	nazMjesto	sifZup
42000	Varaždin	7
51000	Rijeka	2
52100	Pula	4
51300	Delnice	2
42230	Ludbreg	7

zupanija	
sifZup	nazZup
2	Primorsko-goranska
7	Varaždinska
4	Istarska

- neformalna definicija:** relacija je imenovana dvodimenzionalna tablica
 - atribut je imenovani stupac relacije
 - domena je skup dopuštenih vrijednosti atributa
 - nad istom domenom može biti definiran jedan ili više atributa
 - n-torka (*tuple*) je redak relacije

Svojstva relacija

- relacija posjeduje ime koje je jedinstveno unutar sheme baze podataka
- atributi unutar relacije imaju jedinstvena imena
- jedan atribut može poprimiti vrijednost iz samo jedne domene
- u jednoj relaciji ne postoje dvije jednake n-torke
- redoslijed atributa unutar relacije je nebitan
- redoslijed n-torki unutar relacije je nebitan

zupanija	
sifZup	nazZup
2	Primorsko-goranska
7	Varaždinska
4	Istarska

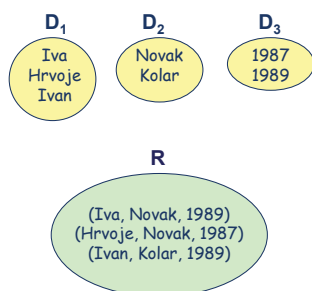
 \equiv

zupanija	
nazZup	sifZup
Varaždinska	7
Istarska	4
Primorsko-goranska	2

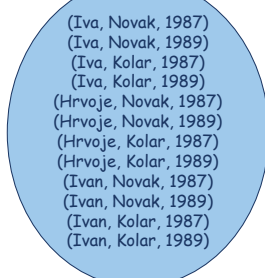
Matematička relacija

- Relacija R definirana nad skupovima D_1, D_2, \dots, D_n je podskup Kartezijevog produkta skupova D_1, D_2, \dots, D_n

$$R \subseteq D_1 \times D_2 \times \dots \times D_n$$



$$D_1 \times D_2 \times D_3$$



Relacijska shema (formalna definicija)

- Neka su zadani atributi A_1, A_2, \dots, A_n . Relacijska shema R (intenzija) je imenovani skup atributa

$$R = \{ A_1, A_2, \dots, A_n \}$$

- radi pojednostavljenja, koristit će se i sljedeća notacija:

$$R = A_1 A_2 \dots A_n$$

- uočite: poredak atributa u shemi relacije je nebitan

$$R = \{ A_1, A_2, A_3 \} \equiv \{ A_3, A_1, A_2 \}$$

- Primjer: relacijska shema MJESTO

$$MJESTO = \{ pbr, nazMjesto, sifZup \}$$

Relacijska shema (primjer)

- Zadani su atributi pbr, nazMjesto, sifZup
- Relacijska shema
MJESTO = { pbr, nazMjesto, sifZup }
identična je relacijskoj shemi
MJESTO = { sifZup, pbr, nazMjesto }

n-torka (formalna definicija)

- Neka je $R = \{ A_1, A_2, \dots, A_n \}$ relacijska shema; neka su D_1, D_2, \dots, D_n domene atributa A_1, A_2, \dots, A_n ; **n-torka t** definirana na relacijskoj shemi R je skup parova oblika *atribut: vrijednostAtributa*
 $t = \{ A_1:v_1, A_2:v_2, \dots, A_n:v_n \}$,
pri čemu je $v_1 \in D_1, v_2 \in D_2, \dots, v_n \in D_n$
- Uočite: poredak elemenata n-torke nije bitan
 $\{ A_1:v_1, A_2:v_2, A_3:v_3 \} \equiv \{ A_3:v_3, A_1:v_1, A_2:v_2 \}$
- Ponekad će se koristiti pojednostavljena notacija: pretpostavi li se da poredak vrijednosti atributa odgovara "poretku atributa" u relacijskoj shemi, n-torka se može prikazati na sljedeći način:
 $t = \langle v_1, v_2, \dots, v_n \rangle$








n-torka (primjer)

- Zadana je relacijska shema OSOBA = { matBr, ime, prez }, pri čemu su domene atributa:
dom (matBr) = { 1234, 1235, 1236, 1237 }
dom (ime) = { Iva, Hrvoje, Ivan }
dom (prez) = { Novak, Kolar }
 $t_1 = \{ \text{matBr:1234, ime:Iva, prez:Novak} \}$
 $t_2 = \{ \text{matBr:1236, ime:Hrvoje, prez:Novak} \}$
 $t_3 = \{ \text{matBr:1237, ime:Ivan, prez:Kolar} \}$
- n-torka t_1 se jednako ispravno može napisati na sljedeći način (poredak elemenata n-torke je nebitan)
 $t_1 = \{ \text{ime:Iva, prez:Novak, matBr:1234} \}$
- pojednostavljena notacija:
 $t_1 = \langle 1234, \text{Iva}, \text{Novak} \rangle$

Relacija (formalna definicija)

- Neka je $R = \{ A_1, A_2, \dots, A_n \}$ relacijska shema; neka su D_1, D_2, \dots, D_n domene atributa A_1, A_2, \dots, A_n ; **relacija r** (instanca relacije) definirana na shemi relacije R je skup n-torki koje su definirane na relacijskoj shemi R
- kad se želi naglasiti da je relacija r definirana na shemi relacije R, kao oznaka za relaciju koristi se $r(R)$ ili $r(\{ A_1, A_2, \dots, A_n \})$ ili $r(A_1 A_2 \dots A_n)$
- relacijska shema R: mijenja se relativno rijetko
- instanca relacije r: predstavlja trenutnu vrijednost relacije i često se mijenja (pri unosu/brisanju/izmjeni podataka)



Relacija (primjer)

- Zadana je relacijska shema STUDENT = { matBr, prez, slika }, pri čemu su domene atributa:
 - dom (matBr) = { 100, 102, 107, 111, 135 }
 - dom (prez) = { Novak, Kolar, Horvat, Ban }
 - dom (slika) = {  ,  ,  }
- student(STUDENT) = { { matBr:102, prez:Novak, slika:  }, { matBr:135, prez:Ban, slika:  } }
- IDENTIČNA RELACIJA (poredak n-torki i članova n-torki je nebitan):
student(STUDENT) = { { prez:Ban, matBr:135, slika:  }, { slika:  , matBr:102, prez:Novak } }

Relacija (primjer)

student(STUDENT) = { { prez:Ban, matBr:135, slika:  }, { slika:  , matBr:102, prez:Novak } }

- pojednostavljenje prikaza relacije (vizualizacija relacije tablicom)

student (STUDENT)		
matBr	prez	slika
102	Novak	
135	Ban	

A-vrijednost n-torke, X-vrijednost n-torke

- Oznaka $t(A)$ predstavlja vrijednost koju atribut A poprima u n-torki t . $t(A)$ se naziva A-vrijednost n-torke t .

- Primjer:

$t = \{ \text{matBr:102, prez:Novak, slika: } \langle \text{fotograf} \rangle \}$
 $t(\text{prez}) = \text{Novak}$

- Neka je $X \subseteq R$. n-torka t reducirana na skup atributa X naziva se X-vrijednost n-torke t i označava s $t(X)$

- Primjer:

$t = \{ \text{matBr:102, prez:Novak, slika: } \langle \text{fotograf} \rangle \}$
 $X = \{ \text{matBr, prez} \} \quad X \subseteq R$
 $t(X) = t(\{ \text{matBr, prez} \}) = \{ \text{matBr:102, prez:Novak} \}$

Stupanj i kardinalnost relacije

- stupanj relacije: broj atributa (stupaca) - *degree*
- kardinalnost relacije: broj n-torki (redaka) - *cardinality*

mjesto		
pbr	nazMjesto	sifZup
42000	Varaždin	7
51000	Rijeka	2
52100	Pula	4
51300	Delnice	2
42230	Ludbreg	7

stupanj = 3

kardinalnost = 5

Oznake:

$\text{deg}(\text{mjesto}) = 3$

$\text{card}(\text{mjesto}) = 5$

Shema i instanca baze podataka

- Shema baze podataka je skup relacijskih shema

$$\mathcal{R} = \{ R_1, R_2, \dots, R_n \}$$

- očito, relacijske sheme u jednoj shemi baze podataka moraju imati različita imena

- Instanca baze podataka definirana na shemi baze podataka $\mathcal{R} = \{ R_1, R_2, \dots, R_n \}$ je skup instanci relacija

$$r = \{ r_1(R_1), r_2(R_2), \dots, r_n(R_n) \}$$

- shema baze podataka se relativno rijetko mijenja
- instanca baze podataka se često mijenja

Operacije u relacijskom modelu podataka

Relacijska algebra

- Operacije relacijske algebre su:

- \cup unija (*union*)
- \cap presjek (*intersection*)
- \setminus razlika (*set difference*)
- \div dijeljenje (*division*)
- π projekcija (*projection*)
- σ selekcija (*selection*)
- \times Kartezijev produkt (*Cartesian product*)
- ρ preimenovanje (*renaming*)
- \bowtie spajanje (*join*)
- agregacija, grupiranje

Primjer: $r_4 = \sigma_{A=x \wedge B=y} (r_1 \cup (r_2 \cap r_3))$

- Karakteristika relacijske algebre - proceduralnost - navodi se redoslijed operacija koje se provode nad relacijama

Predikatni račun

- Operacije se specificiraju navođenjem predikata

$$r = \{ t \mid F(t) \}$$

- t je varijabla koja predstavlja:

- n-torke - n-torski račun
 - rezultat r je skup n-torki t za koje je vrijednost predikata F istina
- domene - domenski račun
 - rezultat je skup domena t za koje je vrijednost predikata F istina

- Primjer:

$$r_4 = \{ t \mid (r_1(t) \vee ((r_2(t) \wedge r_3(t)))) \wedge t(A)=x \wedge t(B)=y \}$$

- Predikatni račun je neproceduralan

- ne navodi se redoslijed operacija
- navode se predikati koje n-torke (domene) moraju zadovoljavati

SQL

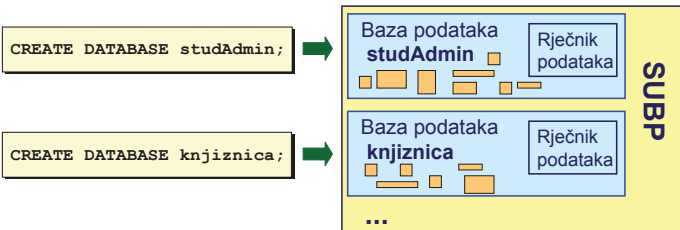
Kratki pogled

SQL - Kratki pogled

- SQL (*Structured Query Language*) je temeljen na relacijskom modelu podataka.
- nastao je na temelju jezika SEQUEL
- temelji se na predikatnom računu i relacijskoj algebri
- proglašen standardnim jezikom za relacijske sustave
- objekti u SQL-u su tablice, a ne (formalno definirane) relacije
 - poredak atributa (stupaca) u nekim je slučajevima značajan
 - u tablici ili rezultatu operacija nad tablicama moguća je pojava dvije ili više istih n-torki
 - ipak, postoje načini kako se to može spriječiti

SQL - Kratki pogled

- kreiranje nove instance baze podataka (kreiranje baze podataka)
 - jedan SUBP može istovremeno upravljati s više baza podataka



- Rječnik podataka sadrži opise relacijskih shema, integritetskih ograničenja, ...

`DROP DATABASE knjiznica;`

SQL - Kratki pogled

- opisivanje relacijske sheme (kreiranje relacije)
 - kreira praznu relaciju
 - ujedno je moguće definirati i integritetska ograničenja

```
CREATE TABLE mjesto (  
  pbr          INTEGER  
  , nazMjesto  CHAR(30)  
  , sifZup     SMALLINT  
);
```

mjesto		
pbr	nazMjesto	sifZup

`DROP TABLE mjesto;`

SQL - Kratki pogled

- upisivanje novih n-torki u relaciju

mjesto		
pbr	nazMjesto	sifZup

```
INSERT INTO mjesto  
VALUES (42000, 'Varaždin', 7);  
INSERT INTO mjesto  
VALUES (52100, 'Pula', 4);  
INSERT INTO mjesto  
VALUES (42230, 'Ludbreg', 7);
```

mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	Varaždin	7
52100	Pula	4

Treba li poredak n-torki u relaciji biti u skladu s redoslijedom upisa?



SQL - Kratki pogled

- dohvat podataka iz relacije

mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	Varaždin	7
52100	Pula	4

- dohvat podataka o mjestima čija šifra županije ima vrijednost 7

```
SELECT * FROM mjesto  
WHERE sifZup = 7;
```

pbr	nazMjesto	sifZup
42000	Varaždin	7
42230	Ludbreg	7

SQL - Kratki pogled

- izmjena vrijednosti atributa u relaciji

mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	Varaždin	7
52100	Pula	4

- naziv mjesta s poštanskim brojem 42000 promijeniti u VARAŽDIN

```
UPDATE mjesto
SET nazMjesto = 'VARAŽDIN'
WHERE pbr = 42000;
```

mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	VARAŽDIN	7
52100	Pula	4

SQL - Kratki pogled

- brisanje n-torki iz relacije

mjesto		
pbr	nazMjesto	sifZup
42230	Ludbreg	7
42000	VARAŽDIN	7
52100	Pula	4

- obrisati mjesta za koje šifra županije ima vrijednost 7

```
DELETE FROM mjesto
WHERE sifZup = 7;
```

mjesto		
pbr	nazMjesto	sifZup
52100	Pula	4

Relacijska algebra

Relacijska algebra

- Unarne operacije
 - projekcija, selekcija, preimenovanje
 - agregacija, grupiranje
- Binarne operacije
 - skupovske operacije (*set operations*)
 - temelje se na relacijama kao skupovima n-torki
 - unija, presjek, razlika
 - ostale binarne operacije
 - Kartezijev produkt, dijeljenje, spajanje

Relacijska algebra

- obavljanje operacije ne utječe na operande, npr.
$$r_3 = r_1 \cup r_2$$
 - obavljanjem prethodne operacije nastaje nova relacija r_3 , a relacije r_1 i r_2 se pri tome ne mijenjaju
- operandi su relacije, a rezultat obavljanja operacije je uvijek relacija. To znači:
 - skup relacija je **zatvoren** s obzirom na operacije relacijske algebre
 - ta činjenica omogućava da se rezultat jedne operacije upotrijebi kao operand u sljedećoj operaciji, što omogućava formiranje složenih izraza
$$r_5 = (r_1 \cup r_2) \times (r_3 \bowtie r_4)$$

Unijska kompatibilnost

- Dvije relacije su unijski kompatibilne ukoliko vrijedi:
 - relacije su istog stupnja
 - i
 - korespondentni atributi su definirani nad istim domenama

polozioMatem			polozioProgr		
matBr	ime	prez	mbr	prezSt	imeSt
12345	Ivo	Kolar	92632	Ban	Jura
13254	Ana	Horvat	67234	Novak	Iva

- relacije su istog stupnja
- $\text{dom}(\text{matBr}) = \text{dom}(\text{mbr})$
- $\text{dom}(\text{ime}) = \text{dom}(\text{imeSt})$
- $\text{dom}(\text{prez}) = \text{dom}(\text{prezSt})$

→ relacije su unijski kompatibilne

- kod ocjene jesu li relacije unijski kompatibilne
 - poredak atributa nije bitan
 - imena atributa nisu bitna

Unijska kompatibilnost

- dvije relacije koje imaju jednak broj atributa i jednaka imena atributa ne moraju ujedno biti unijski kompatibilne

zrakoplov		pecivo	
oznaka	naziv	oznaka	naziv
B-747	Boeing 747	ZE	Žemlja
A-360	Airbus 360	PR	Perec

- relacije su istog stupnja
- $\text{dom}(\text{zrakoplov.oznaka}) \neq \text{dom}(\text{pecivo.oznaka})$
- $\text{dom}(\text{zrakoplov.naziv}) \neq \text{dom}(\text{pecivo.naziv})$

→ relacije NISU unijski kompatibilne

- notacija *imeRelacije.imeAtributa* se često koristi kada je potrebno razlikovati istoimene attribute različitih relacija

Skupovske operacije: unija, presjek, razlika

- Skupovske operacije (unija, presjek, razlika) mogu se obavljati isključivo nad UNIJSKI KOMPATIBILNIM relacijama

Unija

- Rezultat operacije $r_1 \cup r_2$ je relacija čije su n-torke elementi relacije r_1 ili elementi relacije r_2 ili elementi obje relacije.
 - n-torke koje su elementi obje relacije u rezultatu se pojavljuju samo jednom (jer relacija je SKUP n-torki)

polozioMatem		
mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr		
mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

polozioBaremJedan =
polozioMatem \cup polozioProgr

polozioBaremJedan		
mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
105	Rudi	Kolar
107	Jura	Horvat

studenti koji su položili ili Matematiku ili Programiranje ili oba predmeta

$$r_1 \cup r_2 \equiv r_2 \cup r_1$$

Presjek

- Rezultat operacije $r_1 \cap r_2$ je relacija čije su n-torke elementi relacije r_1 i elementi relacije r_2

polozioMatem		
mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr		
mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

polozioOba =
polozioMatem \cap polozioProgr

polozioOba		
mbr	ime	prez
102	Ana	Novak
107	Jura	Horvat

studenti koji su položili i Matematiku i Programiranje

$$r_1 \cap r_2 \equiv r_2 \cap r_1$$

Razlika

- Rezultat operacije $r_1 \setminus r_2$ je relacija čije su n-torke elementi relacije r_1 i nisu elementi relacije r_2

polozioMatem		
mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr		
mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

polozioSamoMatem =
polozioMatem \setminus polozioProgr

polozioSamoMatem		
mbr	ime	prez
100	Ivan	Kolar
103	Tea	Ban

studenti koji su položili Matematiku, ali **nisu** položili Programiranje

$$r_1 \setminus r_2 \neq r_2 \setminus r_1$$

polozioSamoProgr = polozioProgr \setminus polozioMatem

polozioSamoProgr		
mbr	ime	prez
105	Rudi	Kolar

ŠTO AKO SE IMENA KORESPONDENTNIH ATRIBUTA RAZLIKUJU

- Unija, presjek, razlika: u slučajevima kada su relacije unijski kompatibilne, ali se u relacijama koriste različita imena korespondentnih atributa, primjenjuje se sljedeći dogovor (konvencija): **kao imena atributa u rezultatnoj relaciji koriste se imena atributa prvog operanda**

polozioMatem		
mbr	imeSt	prezSt
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr		
mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

polozioOba = polozioMatem \cap polozioProgr

polozioOba		
mbr	imeSt	prezSt
102	Ana	Novak
107	Jura	Horvat

Zadaci za vježbu

- zadane su unijski kompatibilne relacije
 - m (mbr ime prez) → studenti koji su položili Matematiku
 - d (mbr ime prez) → studenti koji su položili Dig. logiku
 - p (mbr ime prez) → studenti koji su položili Programiranje
- napisati izraze relacijske algebre koji određuju relacije koje sadrže studente (točnije rečeno n-torke):
 - koji su položili sva tri predmeta
 - koji su položili ili Matematiku ili Digitalnu logiku, ali ne oba predmeta (*ekskluzivni ili*)
 - koji su položili točno jedan (bilo koji) od ta tri predmeta
 - koji su položili bilo koja dva predmeta (ali nisu položili treći)

Dijeljenje (division)

- Zadane su relacije $r(R)$ i $s(S)$. Neka je $S \subseteq R$. Rezultat operacije $r \div s$ je relacija sa shemom $P = R \setminus S$. n-torka $t_r(P)$ se pojavljuje u rezultatu ako i samo ako za n-torku $t_s \in s$ vrijedi da se $t_r(P)$ u relaciji r pojavljuje u kombinaciji sa svakom n-torkom $t_s \in s$

polozen		predmet	
mbrSt	sifPred	sifPred	
100	1	1	
100	2	2	
101	1	3	
101	2		
101	3		
102	2		
102	3		
103	1		
103	2		
103	3		
104	3		

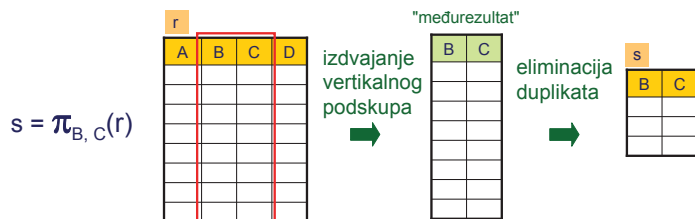
studenti koji su položili sve predmete sa šiframa u relaciji predmet

poloziliSve = polozen \div predmet

poloziliSve	
mbrSt	
101	
103	

Projekcija

- Zadana je relacija $r(R)$. Neka je skup atributa $\{A_1, A_2, \dots, A_k\} \subseteq R$
- Obavljanjem operacije $\pi_{A_1, A_2, \dots, A_k}(r)$ dobiva se relacija s sa shemom $\{A_1, A_2, \dots, A_k\}$ koja sadrži vertikalni podskup relacije r
 - $\deg(s) = k$
 - $\text{card}(s) \leq \text{card}(r)$ (jer se eliminiraju duplikati)



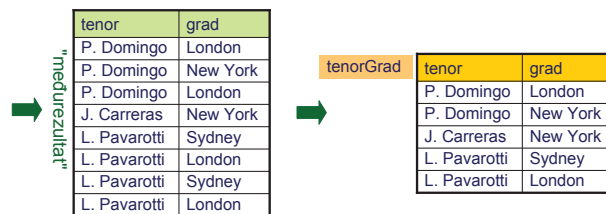
Projekcija (primjer)

Relacija nastup: u kojim gradovima su nastupali koji tenori kojeg datuma

nastup	tenor	grad	datum
	P. Domingo	London	15.2.1976
	P. Domingo	New York	27.3.1981
	P. Domingo	London	11.4.1987
	J. Carreras	New York	11.4.1987
	L. Pavarotti	Sydney	22.6.1992
	L. Pavarotti	London	15.2.1976
	L. Pavarotti	Sydney	19.1.1993
	L. Pavarotti	London	14.7.1993

Traži se: u kojim gradovima su nastupali koji tenori

$\text{tenorGrad} = \pi_{\text{tenor}, \text{grad}}(\text{nastup})$



SQL - Lista za selekciju

mjesto	pbr	nazMjesto	sifZup
42000	Varaždin	7	
52100	Pula	4	

- SELECT** **SELECT List** **FROM** **table**
- SELECT List** je lista za selekciju: dio SELECT naredbe koji određuje koji će se "stupci" pojaviti u rezultatu

`SELECT * FROM mjesto;` \equiv `SELECT mjesto.pbr, mjesto.nazMjesto, mjesto.sifZup FROM mjesto;`

- uz ime atributa može se navesti ime relacije (radi izbjegavanja dvosmislenosti u slučajevima kada se podaci dohvaćaju istovremeno iz više relacija čija se imena atributa podudaraju)
imeRelacije.imeAtributa
- u slučajevima kada takva dvosmislenost ne postoji, ime relacije se može (ali ne mora) ispustiti

SQL - Lista za selekciju

mjesto	pbr	nazMjesto	sifZup
42000	Varaždin	7	
52100	Pula	4	

zupanija	sifZupanija	nazZup
7		Varaždinska
4		Istarska

- u listi za selekciju se ne moraju navesti svi atributi relacije navedene u FROM dijelu naredbe:

`SELECT nazMjesto, pbr FROM mjesto;` \rightarrow

nazMjesto	pbr
Varaždin	42000
Ludbreg	52100

- u listi za selekciju se mogu navesti samo oni atributi koji se nalaze u doseg SELECT naredbe, tj. atributi relacije koja je navedena u FROM dijelu naredbe:

`SELECT nazMjesto, pbr, nazZup FROM mjesto;`

Neispravna naredba

SQL - Projekcija

- za ispravno obavljanje projekcije nije dovoljno u listi za selekciju samo navesti imena atributa prema kojima se obavlja projekcija:

- primjer koji ujedno pokazuje kako rezultat SQL naredbe ne mora uvijek biti relacija

```
SELECT tenor
, grad
FROM nastup;
```

Neispravna projekcija

tenor	grad
P. Domingo	London
P. Domingo	New York
P. Domingo	London
J. Carreras	New York
L. Pavarotti	Sydney
L. Pavarotti	London
L. Pavarotti	Sydney
L. Pavarotti	London

$\pi_{\text{tenor,grad}}(\text{nastup})$

```
SELECT DISTINCT tenor
, grad
FROM nastup;
```

Ispravna projekcija

tenor	grad
P. Domingo	London
P. Domingo	New York
J. Carreras	New York
L. Pavarotti	Sydney
L. Pavarotti	London

Selekcija

- Zadana je relacija $r(R)$. Neka je F predikat (formula, uvjet, *condition*) koji se sastoji od operandata i operatora
 - operandi su:
 - imena atributa iz R
 - konstante
 - operatori su:
 - operatori usporedbe: $< \leq = \neq > \geq$
 - logički operatori: $\wedge \vee \neg$
- Obavljanjem operacije $\sigma_F(r)$ dobiva se relacija sa shemom R koja sadrži one n -torke relacije r za koje je vrijednost predikata F istina (*true*)

Selekcija (primjer)

student	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	105	Jura	Novak	21000
	107	Ana	Ban	51000

rezultat = $\sigma_{\text{ime} = \text{'Ana'} \vee \text{postBr} > 31000}(\text{student})$

- Za svaku pojedinu n -torku relacije:
 - vrijednosti atributa uvrstavaju se u predikat - uvrštavanjem vrijednosti u predikat dobiva se **sud**
 - onda i samo onda kada je vrijednost dobivenog suda istina (*true*), n -torka se pojavljuje u rezultatu selekcije

$\rightarrow \text{'Ivan'} = \text{'Ana'} \vee 52000 > 31000 \rightarrow \text{true}$
 $\rightarrow \text{'Ana'} = \text{'Ana'} \vee 10000 > 31000 \rightarrow \text{true}$
 $\rightarrow \text{'Jura'} = \text{'Ana'} \vee 21000 > 31000 \rightarrow \text{false}$
 $\rightarrow \text{'Ana'} = \text{'Ana'} \vee 51000 > 31000 \rightarrow \text{true}$

rezultat	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	107	Ana	Ban	51000

SQL - Selekcija

- SELECT** *Select List* **FROM** *table* [**WHERE** *Condition*]
- Uvjet (*Condition*) se sastoji od operandata i operatora
 - operandi su:
 - imena atributa iz relacije *table*
 - konstante
 - operatori su:
 - operatori usporedbe: $< \leq = <> > \geq$
 - logički operatori: **AND OR NOT**
- Vrijednosti svake n -torke iz relacije *table* se uvrstavaju u *Condition* (a to je u stvari predikat). Ako je dobiveni sud istinit (*true*), n -torka se pojavljuje u rezultatu.

SQL - Selekcija

student	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	105	Jura	Novak	21000
	107	Ana	Ban	51000

$\sigma_{\text{ime} = \text{'Ana'} \vee \text{postBr} > 31000}(\text{student})$

```
SELECT * FROM student
WHERE ime = 'Ana'
OR postBr > 31000;
```

matBr	ime	prez	postBr
100	Ivan	Kolar	52000
102	Ana	Horvat	10000
107	Ana	Ban	51000

SQL - Projekcija i selekcija

student	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	105	Jura	Novak	21000
	107	Ana	Ban	51000

$\pi_{\text{ime}}(\sigma_{\text{ime} = \text{'Ana'} \vee \text{postBr} > 31000}(\text{student}))$

```
SELECT DISTINCT ime
FROM student
WHERE ime = 'Ana'
OR postBr > 31000;
```

"međurezultat"

matBr	ime	prez	postBr
100	Ivan	Kolar	52000
102	Ana	Horvat	10000
107	Ana	Ban	51000

ime
Ivan
Ana

Kartezijev produkt

- Zadana je relacija $r(R)$ i relacija $s(S)$, pri čemu je $R \cap S = \emptyset$.
- Obavljanjem operacije $r \times s$ dobiva se relacija $p(P)$, $P = R \cup S$.
n-torke relacije p se dobivaju spajanjem (ulančavanjem) svake n-torke iz relacije r sa svakom n-torkom iz relacije s

- $\deg(p) = \deg(r) + \deg(s)$
- $\text{card}(p) = \text{card}(r) \cdot \text{card}(s)$

Kartezijev produkt (primjer)

student			predmet	
mbr	ime	prez	sifra	naziv
100	Ivan	Kolar	1	Programiranje
102	Ana	Novak	2	Matematika
103	Tea	Ban		

upis = student \times predmet

upis				
mbr	ime	prez	sifra	naziv
100	Ivan	Kolar	1	Programiranje
100	Ivan	Kolar	2	Matematika
102	Ana	Novak	1	Programiranje
102	Ana	Novak	2	Matematika
103	Tea	Ban	1	Programiranje
103	Tea	Ban	2	Matematika

SQL - Kartezijev produkt

- SELECT SELECT List**
FROM table [, table]...
[WHERE Condition]
- navede li se u FROM dijelu naredbe više od jedne relacije, obavlja se operacija Kartezijevog produkta navedenih relacija

student		
mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban

predmet	
sifra	naziv
1	Programiranje
2	Matematika

student \times predmet

mbr	ime	prez	sifra	naziv
100	Ivan	Kolar	1	Programiranje
100	Ivan	Kolar	2	Matematika
102	Ana	Novak	1	Programiranje
102	Ana	Novak	2	Matematika
103	Tea	Ban	1	Programiranje
103	Tea	Ban	2	Matematika

```
SELECT *
FROM student, predmet;
```

```
SELECT student.*, predmet.*
FROM student, predmet;
```

SQL - Kartezijev produkt

- drugačija sintaksa:
- SELECT SELECT List**
FROM table [CROSS JOIN table]...
[WHERE Condition]

```
SELECT *
FROM student CROSS JOIN predmet;
```

- Kartezijev produkt triju relacija:

```
SELECT *
FROM r1 CROSS JOIN r2 CROSS JOIN r3;
```

Kartezijev produkt

- Što učiniti ukoliko je potrebno obaviti operaciju Kartezijevog produkta nad relacijama $r(R)$ i $s(S)$, u slučaju kada $R \cap S \neq \emptyset$

r	
A	B
1	a
2	b
3	c

s	
B	C
c	α
d	β

A	B	B	C
1	a	c	α
1	a	d	β
2	b	c	α
2	b	d	β
3	c	c	α
3	c	d	β

↑
NIJE RELACIJA!

→ Potrebno je koristiti operaciju preimenovanja

Preimenovanje (relacije, atributa)

- Zadana je relacija $r(\{A_1, A_2, \dots, A_n\})$
 - preimenovanje relacije:** operacijom preimenovanja $\rho_s(r)$ dobiva se relacija s koja ima jednaku shemu i sadržaj kao relacija r
 - preimenovanje relacije i atributa:** operacijom preimenovanja $\rho_{s(B_1, B_2, \dots, B_n)}(r)$ dobiva se relacija s čija shema umjesto atributa A_1, A_2, \dots, A_n sadrži attribute B_1, B_2, \dots, B_n , a sadržaj relacije s je jednak sadržaju relacije r

$p = r \times \rho_{s(B_2, C)}(s)$

r		
A	B	
1	a	
2	b	
3	c	

s	
B	C
c	α
d	β

p			
A	B	B2	C
1	a	c	α
1	a	d	β
2	b	c	α
2	b	d	β
3	c	c	α
3	c	d	β

SQL - Preimenovanje atributa

- ukoliko se drugačije ne navede, imena stupaca u rezultatu odgovaraju imenima atributa iz liste za selekciju
- implicitna imena stupaca rezultata se mogu promijeniti korištenjem operatora za preimenovanje **AS**

zupanja	sifZupanja	nazZup
	7	Varaždinska
	4	Istarska

```
SELECT sifZupanja AS sifraz
, nazZup AS nazZ
FROM zupanja;
```

- rezervirana riječ **AS** smije se ispustiti

sifraz	nazZ
7	Varaždinska
4	Istarska

```
SELECT sifZupanja sifraz
, nazZup nazZ
FROM zupanja;
```

SQL - Preimenovanje atributa

- Primjer u kojem je potrebno koristiti preimenovanje atributa
 - SQL naredba bi bila ispravna i bez preimenovanja, ali tada kao rezultat ne bismo dobili relaciju (jer bi u shemi rezultata postojala dva atributa istog imena)

r	A	B
	1	a
	2	b
	3	c

s	B	C
	c	α
	d	β

$$r \times \rho_{s(B2, C)}(s)$$

```
SELECT A, r.B, s.B AS B2, C
FROM r, s;
```

A	B	B2	C
1	a	c	α
1	a	d	β
2	b	c	α
2	b	d	β
3	c	c	α
3	c	d	β

Spajanje uz uvjet ili θ - spajanje (θ - join)

- Zadane su relacije $r(R)$ i $s(S)$ pri čemu je $R \cap S = \emptyset$. Neka je F predikat oblika $r.A_i \theta s.B_j$, pri čemu je $A_i \in R$, $B_j \in S$, a θ je operator usporedbe iz skupa operatora $\{<, \leq, =, \neq, >, \geq\}$
- Obavljanjem operacije $r \bowtie_F s$ dobiva se relacija koja sadrži n -torke iz $r \times s$ za koje je vrijednost predikata F istina (*true*), odnosno:

$$r \bowtie_F s = \sigma_F(r \times s)$$

što možemo reći o stupnju i kardinalnosti rezultata?

- Umjesto jednostavnog predikata $r.A_i \theta s.B_j$, može se koristiti složeni predikat dobiven primjenom logičkih operatora nad jednostavnim predikatima oblika $r.A_i \theta s.B_j$
- Problem spajanja uz uvjet relacija $r(R)$ i $s(S)$ kod kojih je $R \cap S \neq \emptyset$, rješava se na jednak način kao kod Kartezijevog produkta (korištenjem operatora preimenovanja)

Spajanje uz uvjet (primjer)

linija	zrakoplov		
let	udaljenost	tip	dolet
CA-825	700	B747	13000
LH-412	4800	A320	5400
BA-722	15000	DC-9	3100
CA-311	13000		

moгуćnost = linija $\bowtie_{\text{dolet} \geq \text{udaljenost}}$ zrakoplov

let	udaljenost	tip	dolet
CA-825	700	B747	13000
CA-825	700	A320	5400
CA-825	700	DC-9	3100
LH-412	4800	B747	13000
LH-412	4800	A320	5400
CA-311	13000	B747	13000

Linije i zrakoplovi koji na tim linijama mogu letjeti



SQL - Spajanje uz uvjet

- Koristi se ekvivalencija

$$r \bowtie_F s = \sigma_F(r \times s)$$

linija $\bowtie_{\text{dolet} \geq \text{udaljenost}}$ zrakoplov

linija	
let	udaljenost
CA-825	700
LH-412	4800
BA-722	15000
CA-311	13000

zrakoplov	
tip	dolet
B747	13000
A320	5400
DC-9	3100

```
SELECT *
FROM linija, zrakoplov
WHERE dolet >= udaljenost;
```

Kartezijev produkt
Selekcija

let	udaljenost	tip	dolet
CA-825	700	B747	13000
CA-825	700	A320	5400
CA-825	700	DC-9	3100
LH-412	4800	B747	13000
LH-412	4800	A320	5400
CA-311	13000	B747	13000

Linije i zrakoplovi koji na tim linijama mogu letjeti

SQL - Spajanje uz uvjet

- drugačija sintaksa:

```
SELECT SELECT List
FROM table [JOIN table ON joinCondition]...
[WHERE Condition]
```

```
SELECT *
FROM linija JOIN zrakoplov
ON dolet >= udaljenost;
```

- Spajanje uz uvjet triju relacija:

```
SELECT *
FROM r1
JOIN r2
ON joinCondition
JOIN r3
ON joinCondition;
```

SQL - Spajanje uz uvjet i selekcija

- Kako pronaći linije i zrakoplove koji na tim linijama mogu letjeti, ali samo za one linije na kojima je udaljenost veća od 4000 km

$\sigma_{\text{udaljenost} > 4000}(\text{linija} \bowtie \text{zrakoplov})$
dolet \geq udaljenost

```
SELECT *
FROM linija, zrakoplov
WHERE dolet >= udaljenost
AND udaljenost > 4000;
```

let	udaljenost	tip	dolet
LH-412	4800	B747	13000
LH-412	4800	A320	5400
CA-311	13000	B747	13000

ili

```
SELECT *
FROM linija
JOIN zrakoplov
ON dolet >= udaljenost
WHERE udaljenost > 4000;
```

SQL - Spajanje uz uvjet i projekcija

- Kako pronaći tipove zrakoplova koji se mogu iskoristiti za letove na postojećim linijama

$\pi_{\text{tip}}(\text{linija} \bowtie \text{zrakoplov})$
dolet \geq udaljenost

```
SELECT DISTINCT tip
FROM linija, zrakoplov
WHERE dolet >= udaljenost;
```

tip
B747
A320
DC-9

ili

```
SELECT DISTINCT tip
FROM linija
JOIN zrakoplov
ON dolet >= udaljenost;
```

Spajanje s izjednačavanjem (Equi-join)

- Spajanje relacija s izjednačavanjem je poseban oblik spajanja uz uvjet u kojem se kao θ operator koristi isključivo operator jednakosti (=)

mjesto	pbr	nazMjesto	sifZup
	42000	Varaždin	7
	52100	Pula	4
	42230	Ludbreg	7

zupanija	sifZupanija	nazZup
	7	Varaždinska
	4	Istarska

mjestouZupaniji = mesto \bowtie zupanija
sifZup = sifZupanija

mjestouZupaniji	pbr	nazMjesto	sifZup	sifZupanija	nazZup
	42000	Varaždin	7	7	Varaždinska
	52100	Pula	4	4	Istarska
	42230	Ludbreg	7	7	Varaždinska

- Problem spajanja s izjednačavanjem relacija $r(R)$ i $s(S)$ kod kojih je $R \cap S \neq \emptyset$, rješava se na jednak način kao kod Kartezijevog produkta (korištenjem operatora preimenovanja)

SQL - Spajanje s izjednačavanjem

- Koristi se ekvivalencija

$r \bowtie_F s = \sigma_F(r \times s)$

mjesto	pbr	nazMjesto	sifZup
	42000	Varaždin	7
	52100	Pula	4
	42230	Ludbreg	7

mjesto \bowtie zupanija
sifZup = sifZupanija

zupanija	sifZupanija	nazZup
	7	Varaždinska
	4	Istarska

```
SELECT *
FROM mjesto, zupanija
WHERE sifZup = sifZupanija;
```

ili

```
SELECT *
FROM mjesto
JOIN zupanija
ON sifZup = sifZupanija;
```

SQL - Spajanje s izjednačavanjem

- U slučaju kada u relacijama postoje istoimeni atributi

mjesto	pbr	nazMjesto	sifZup
	42000	Varaždin	7
	52100	Pula	4
	42230	Ludbreg	7

zupanija	sifZup	nazZup
	7	Varaždinska
	4	Istarska

mjesto \bowtie zupanija
mjesto.sifZup = zupanija.sifZup

```
SELECT mjesto.*
, zupanija.sifZup AS sifZup2
, zupanija.nazZup
FROM mjesto, zupanija
WHERE mjesto.sifZup = zupanija.sifZup;
```

slično i u slučaju korištenja drugačije sintakse

Prirodno spajanje (Natural Join)

- Prirodno spajanje obavlja se na temelju jednakih vrijednosti istoimenih atributa.
- Zadane su relacije $r(R)$ i $s(S)$. Neka je $R \cap S = \{A_1, A_2, \dots, A_n\}$. Obavljanjem operacije $r \bowtie s$ dobiva se relacija sa shemom $R \cup S$ koja sadrži n-torke nastale spajanjem n-torki $t_r \in r, t_s \in s$, za koje vrijedi $t_r(A_1) = t_s(A_1) \wedge t_r(A_2) = t_s(A_2) \wedge \dots \wedge t_r(A_n) = t_s(A_n)$.

mjesto	pbr	nazMjesto	sifZup
	42000	Varaždin	7
	52100	Pula	4
	42230	Ludbreg	7

zupanija	sifZup	nazZup
	7	Varaždinska
	4	Istarska

mjestouZupaniji = mjesto \bowtie zupanija

mjestouZupaniji	pbr	nazMjesto	sifZup	nazZup
	42000	Varaždin	7	Varaždinska
	52100	Pula	4	Istarska
	42230	Ludbreg	7	Varaždinska

što možemo reći o stupnju rezultata?

Prirodno spajanje

- Rezultat prirodnog spajanje relacija $r(R)$ i $s(S)$ za koje vrijedi da je $R \cap S = \emptyset$ identičan je rezultatu obavljanja operacije Kartezijevog produkta $r \times s$

mjesto	pbr	nazMjesto	sifZup	zupanija	sifZupanija	nazZup
	42000	Varaždin	7		7	Varaždinska
	52100	Pula	4		4	Istarska
	42230	Ludbreg	7			

mjestouZupaniji = mesto ▷◁ zupanija

mjestouZupaniji	pbr	nazMjesto	sifZup	sifZupanija	nazZup
	42000	Varaždin	7	7	Varaždinska
	42000	Varaždin	7	4	Istarska
	52100	Pula	4	7	Varaždinska
	52100	Pula	4	4	Istarska
	42230	Ludbreg	7	7	Varaždinska
	42230	Ludbreg	7	4	Istarska

SQL - Prirodno spajanje

- prirodno spajanje se razlikuje od spajanja s izjednačavanjem po tome što se istoimeni atributi iz dviju relacija izbacuju (tako da od svakog ostane samo po jedan)

mjesto	pbr	nazMjesto	sifZup	zupanija	sifZup	nazZup
	42000	Varaždin	7		7	Varaždinska
	52100	Pula	4		4	Istarska
	42230	Ludbreg	7			

```
SELECT mesto.*, zupanija.nazZup
FROM mesto, zupanija
WHERE mesto.sifZup = zupanija.sifZup;
```

pbr	nazMjesto	sifZup	nazZup
42000	Varaždin	7	Varaždinska
52100	Pula	4	Istarska
42230	Ludbreg	7	Varaždinska

SQL - Prirodno spajanje

- drugačija sintaksa:

```
SELECT mesto.*, zupanija.nazZup
FROM mesto JOIN zupanija
ON mesto.sifZup = zupanija.sifZup;
```

Agregacija (aggregation)

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

- Kako izračunati prosjek ocjena na svim ispitima?

prosjeck	prosjeck
	3.625

Agregacija

- Zadana je relacija $r(R)$. Neka je atribut $A \in R$. Neka je \mathcal{AF} agregatna funkcija. Rezultat operacije agregacije $G_{\mathcal{AF}(A)}(r)$ je **relacija** stupnja 1 i kardinalnosti 1, pri čemu je vrijednost atributa određena primjenom funkcije \mathcal{AF} nad vrijednostima atributa A u svim n-torkama relacije r . Funkcija \mathcal{AF} može biti jedna od:
 - COUNT određuje broj pojava (broji sve, eventualni duplikati se također broje)
 - SUM izračunava sumu vrijednosti
 - AVG izračunava aritmetičku sredinu vrijednosti
 - MIN izračunava najmanju vrijednost
 - MAX izračunava najveću vrijednost
- naziv rezultatne relacije i atributa nije definiran operacijom, stoga se najčešće koristi u kombinaciji s operacijom preimenovanja
- također se koriste agregatne funkcije
 - COUNT-DISTINCT, SUM-DISTINCT, AVG-DISTINCT

Agregacija

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

- Prosjeck ocjena na svim ispitima (rješenje):

$P_{\text{prosjeck}}(\text{prosjeck})(G_{\text{AVG}(\text{ocjena})}(\text{ispit}))$

prosjeck	prosjeck
	3.625

```
SELECT AVG(ocjena) AS prosjeck
FROM ispit;
```

prosjeck	prosjeck
	3.625

Agregacija (primjeri ostalih agregatnih funkcija)

osoba	sifra	tezina	visina
	101	62	170
	103	94	186
	105	74	181
	107	62	165

$\rho_{rez1(broj1)}(G_{COUNT(sifra)}(osoba))$

rez1	broj1
	4

$\rho_{rez2(broj2)}(G_{SUM(tezina)}(osoba))$

rez2	broj2
	292

$\rho_{rez3(broj3)}(G_{AVG(visina)}(osoba))$

rez3	broj3
	175.5

$\rho_{rez4(broj4)}(G_{MAX(visina)}(osoba))$

rez4	broj4
	186

$\rho_{rez5(broj5)}(G_{MIN(tezina)}(osoba))$

rez5	broj5
	62

Moguće je odjednom izračunati više agregatnih vrijednosti:

$\rho_{rez6(broj6, broj7, broj8)}(G_{MIN(tezina), AVG(visina), MAX(visina)}(osoba))$

rez6	broj6	broj7	broj8
	62	175.5	186

SQL - Agregatne funkcije

- naziv rezultatnog atributa nije definiran operacijom, stoga se koristi AS operator za preimenovanje

```
SELECT COUNT(sifra) AS broj1 FROM osoba;
```

osoba	sifra	tezina	visina
	101	62	170
	103	94	186
	105	74	181
	107	62	165

broj1
4

```
SELECT SUM(tezina) AS broj2 FROM osoba;
```

broj2
292

```
SELECT AVG(visina) AS broj3 FROM osoba;
```

broj3
175.5

```
SELECT MAX(visina) AS broj4,
       MIN(tezina) AS broj5
FROM osoba;
```

broj4	broj5
186	62

SQL - Agregatne funkcije

- agregatne funkcije s DISTINCT

osoba	sifra	tezina	visina
	101	62	170
	103	94	190
	105	74	170
	107	62	170

```
SELECT COUNT(DISTINCT visina) AS broj1
FROM osoba;
```

broj1
2

```
SELECT SUM(DISTINCT tezina) AS broj2
FROM osoba;
```

broj2
230

```
SELECT AVG(DISTINCT visina) AS broj3
FROM osoba;
```

broj3
180

Agregacija i grupiranje

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

- Zadatak: izračunati prosječnu ocjenu za svaki pojedini predmet
 - prosjek za Matematiku
 - prosjek za Fiziku
 - ... i za sve ostale predmete čiji se naziv pojavljuje u relaciji

Agregacija i grupiranje

- Loše rješenje:

- Za svaki predmet napisati po jedan upit

$\rho_{\text{prosjek(prosjOcjMat)}}(G_{AVG(ocjena)}(\sigma_{nazPred='Matematika'}(ispit)))$

```
SELECT AVG(ocjena) AS prosjOcjMat
FROM ispit
WHERE nazPred = 'Matematika';
```

prosjOcjMat
3.25

$\rho_{\text{prosjek(prosjOcjFiz)}}(G_{AVG(ocjena)}(\sigma_{nazPred='Fizika'}(ispit)))$

```
SELECT AVG(ocjena) AS prosjOcjFiz
FROM ispit
WHERE nazPred = 'Fizika';
```

prosjOcjFiz
4

- itd. (za svaki naziv predmeta)
 - postoji li bolje rješenje?

Grupiranje (grouping)

- Zadana je relacija $r(R)$. Neka su atributi $A_1, A_2, \dots, A_m, B_1, B_2, \dots, B_n$ atributi sheme R . Opći oblik operacije grupiranja je sljedeći:

$$A_1, A_2, \dots, A_m \mathcal{G}_{\mathcal{AF}_1(B_1), \mathcal{AF}_2(B_2), \dots, \mathcal{AF}_n(B_n)}(r)$$

- određuju se grupe n -torki: u svakoj grupi se nalaze n -torke koje imaju jednake vrijednosti atributa A_1, A_2, \dots, A_m
- za svaku grupu n -torki izračunavaju se vrijednosti agregatnih funkcija $\mathcal{AF}_1(B_1), \mathcal{AF}_2(B_2), \dots, \mathcal{AF}_n(B_n)$
- za svaku grupu formira se n -torka s vrijednostima atributa A_1, A_2, \dots, A_m i izračunatim vrijednostima agregatnih funkcija

Agregacija i grupiranje

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

- Za svaki predmet ispisati prosječnu ocjenu (ispravno rješenje):

$\rho_{\text{prosjek}}(\text{nazPred}, \text{prosJocj})(\text{nazPred}, G_{\text{AVG}}(\text{ocjena})(\text{ispit}))$

- grupirati po nazPred
- za svaku grupu izračunati AVG(ocjena)
- za svaku grupu formirati po jednu n-torku s vrijednošću atributa nazPred i izračunatim prosjekom
- obaviti operaciju preimenovanja

prosjeak	prosJocj
Matematika	4
Fizika	3.25
Vjerojatnost	4

Agregacija i grupiranje

- Ispisati prosječnu i najveću ocjenu za svaki predmet i akademsku godinu:

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

- u istu grupu ulaze n-torke koje imaju jednake vrijednosti atributa nazPred i akGod

$\rho_{\text{prosjek1}}(\text{nazPred}, \text{akGod}, \text{prosJocj}, \text{maxOcj})(\text{nazPred}, \text{akGod}, G_{\text{AVG}}(\text{ocjena}), \text{MAX}(\text{ocjena})(\text{ispit}))$

prosje1	nazPred	akGod	prosJOcj	maxOcJ
	Matematika	2005	3.333	5
	Matematika	2006	3	3
	Fizika	2004	5	5
	Fizika	2006	3.5	5
	Vjerojatnost	2005	4	4

SQL - Grupiranje

- SELECT** *SELECT List*

FROM ...

[**WHERE** *Condition*]

[**GROUP BY** *column* [, *column*]...]

$\rho_{\text{prosjek1}}(\text{nazPred}, \text{akGod}, \text{prosJocj}, \text{maxOcj})(\text{nazPred}, \text{akGod}, G_{\text{AVG}}(\text{ocjena}), \text{MAX}(\text{ocjena})(\text{ispit}))$

```
SELECT nazPred
, akGod
, AVG(ocjena) AS prosjOcj
, MAX(ocjena) AS maxOcj
FROM ispit
GROUP BY nazPred, akGod;
```

nazPred	akGod	prosJocj	maxOcj
Matematika	2005	3.333	5
Matematika	2006	3	3
Fizika	2004	5	5
Fizika	2006	3.5	5
Vjerojatnost	2005	4	4

SQL - Grupiranje

- svi atributi koji se nalaze u listi za selekciju, a koji nisu argumenti agregatnih funkcija, **moraju** biti navedeni u GROUP BY dijelu naredbe

```
SELECT nazPred
, akGod
, mbrStud
, AVG(ocjena) AS prosjOcj
, MAX(ocjena) AS maxOcj
FROM ispit
GROUP BY nazPred, akGod;
```

NEISPRAVNO!

ispit	mbrStud	akGod	nazPred	ocjena
	100	2005	Matematika	3
	101	2005	Matematika	5
	102	2005	Matematika	2
	103	2006	Matematika	3
	100	2004	Fizika	5
	101	2006	Fizika	5
	102	2006	Fizika	2
	100	2005	Vjerojatnost	4

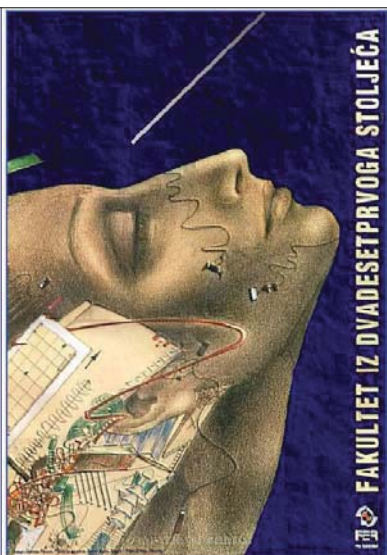
Zašto je to neispravno?
Za svaku grupu se formira samo po jedna n-torka: što s onim grupama u kojima postoji više vrijednosti atributa mbrStud?

nazPred	akGod	mbrStud	prosJocj	maxOcj
Matematika	2005	100, 101, 102 ?	3.333	5
Matematika	2006	?	3	3
Fizika	2004	?	5	5
Fizika	2006	?	3.5	5
Vjerojatnost	2005	?	4	4

Baze podataka

Predavanja
ožujak 2008.

3. Nepotpune informacije i NULL vrijednosti



NULL vrijednosti

- Ponekad se dešava da informacije koje treba unijeti u bazu podataka nisu potpune
 - neke informacije trenutno nisu poznate
 - neke informacije uopće ne postoje (nisu primjenjive)
 - neke informacije postoje, ali do njih nije moguće doći
- Informacije koje nedostaju prikazuju se kao poseban oblik podatka: NULL vrijednost

nije primjenjivo
(vidi datum rođenja)

clanoviKnjznice

mbr	ime	prez	pbr	datRodj	adresa	zanimanje
100	Maja	Novak	10000	01.5.2001	Ilica 1	NULL
105	Ivo	Kolar	21000	12.3.1973	NULL	odvjetnik
107	James	Bond	NULL	NULL	NULL	tajni agent

nedostupno

trenutno nepoznato

SQL - Interna pohrana NULL vrijednosti

- NULL vrijednost se interno pohranjuje drugačije od bilo koje druge dopuštene vrijednosti (nije 0, nije 0.0, nije prazan niz, ...)
- Način interne pohrane NULL vrijednosti je **nebitan** - NULL vrijednost je neovisna od tipa podatka kojeg predstavlja. U SQL naredbama, bez obzira na tip podatka, koristi se "konstanta" **NULL**

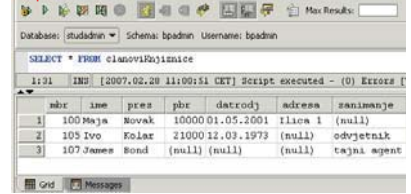
```
CREATE TABLE mjesto (  
    pbr          INTEGER  
    , nazMjesto  CHAR(30)  
    , sifZupanija SMALLINT  
);
```

```
INSERT INTO mjesto VALUES (10000, 'Zagreb', NULL);  
INSERT INTO mjesto VALUES (10001, NULL, 1);  
UPDATE mjesto SET sifZupanija = NULL  
WHERE pbr = 10001;
```

SQL - prikaz NULL vrijednosti

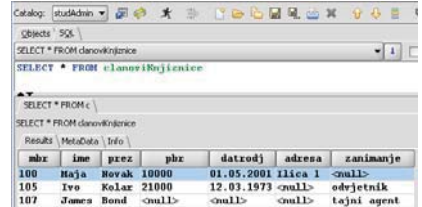
- Način na koji se NULL vrijednost prikazuje korisniku ovisi o programskom alatu koji se koristi:

Aqua Data Studio SQL Client



mbr	ime	prez	pbr	datrodj	adresa	zanimanje
1	100	Maja	Novak	10000	01.05.2001	Ilica 1 (null)
2	105	Ivo	Kolar	21000	12.03.1973	(null) odvjetnik
3	107	James	Bond	(null)	(null)	(null) tajni agent

Squirrel SQL Client



mbr	ime	prez	pbr	datrodj	adresa	zanimanje
100	Maja	Novak	10000	01.05.2001	Ilica 1	<null>
105	Ivo	Kolar	21000	12.03.1973	<null>	<null> odvjetnik
107	James	Bond	<null>	<null>	<null>	<null> tajni agent

SQL - Izrazi

- Izraz (*Expression*) se sastoji od
 - imena atributa
 - konstanti
 - operatora $+$ $-$ $*$ $/$ unarni $+$ $-$
 - zagrada $()$
- Izrazi se mogu koristiti
 - u listi za selekciju
 - u uvjetu u WHERE dijelu naredbe
 - i drugdje ...

SQL - Izrazi

- iznosi plaća za svaku osobu su navedeni u kunama
- ispisati matični broj, prezime i plaću izraženu u dolarima, za one osobe čija je plaća veća od 1000 eura
- 1 USD = 5.6 KN, 1 KN = 0.136 EUR

dohodak		
mbr	prez	placa
100	Novak	7500
102	Horvat	5600
105	Kolar	9000
107	Ban	4200

```
SELECT mbr  
    , prez  
    , placa/5.6  
FROM dohodak  
WHERE placa*0.136 > 1000;
```

mbr	prez	(expression)
100	Novak	1339.28571429
105	Kolar	1607.14285714

```
SELECT mbr  
    , prez  
    , placa/5.6 AS placaUSD  
FROM dohodak  
WHERE placa*0.136 > 1000;
```

mbr	prez	placaUSD
100	Novak	1339.28571429
105	Kolar	1607.14285714

NULL vrijednost u izrazima

- Neka je binarni operator $\alpha \in \{+, -, *, /\}$, a X i Y su izrazi
 - ako jedan ili oba operanda X, Y poprimaju NULL vrijednost, tada je rezultat izraza $X \alpha Y$ također NULL vrijednost

```
5 + NULL      → NULL  
NULL - NULL   → NULL  
NULL * 0      → NULL
```

- Neka je unarni operator $\beta \in \{+, -\}$, a X je izraz
 - ako operand X poprima NULL vrijednost, tada je rezultat izraza βX također NULL vrijednost

```
- NULL      → NULL
```

SQL - NULL vrijednost u izrazima

bodovi	mbr	prez	bodLab	bodMI
	101	Novak	12	NULL
	103	Ban	NULL	NULL
	107	Horvat	21	66.3
	109	Kolar	NULL	54.3

```
SELECT mbr  
    , prez  
    , bodLab + bodMI AS ukupBodova  
    , - bodLab AS negBodLab  
FROM bodovi;
```

mbr	prez	ukupBodova	negBodLab
101	Novak	NULL	-12
103	Ban	NULL	NULL
107	Horvat	87.3	-21
109	Kolar	NULL	NULL

NULL vrijednost u uvjetima usporedbe

- Neka su X i Y izrazi, a γ je operator usporedbe
 $\gamma \in \{<, \leq, =, \neq, >, \geq\}$
- ako niti jedan od operandi X, Y nije NULL vrijednost, tada je rezultat izraza $X \gamma Y$ logička vrijednost istina (*true*) ili logička vrijednost laž (*false*)
- ako jedan ili oba operandi X, Y jesu NULL vrijednosti, tada je rezultat izraza $X \gamma Y$ logička vrijednost nepoznato (*unknown*)

$7 \geq 5 \rightarrow \text{true}$
 $'atlas' > 'zvuk' \rightarrow \text{false}$
 $-17.8 \leq \text{NULL} \rightarrow \text{unknown}$
 $\text{NULL} = \text{NULL} \rightarrow \text{unknown}$
 $\text{NULL} \neq \text{NULL} \rightarrow \text{unknown}$

Operacija selekcije - NULL vrijednosti

- Obavljanjem operacije selekcije $\sigma_F(r)$ dobiva se relacija koja sadrži samo one n-torke relacije r za koje je vrijednost predikata F istina (*true*). To znači da se n-torke za koje je vrijednost predikata F laž (*false*) ili nepoznato (*unknown*) ne pojavljuju u rezultatu

r	A	B	$s = \sigma_{B \leq 50}(r)$	s	A	B
1	20	20	$20 \leq 50 \rightarrow \text{true}$	1	20	
2	NULL	50	$\text{NULL} \leq 50 \rightarrow \text{unknown}$			
3	60	60	$60 \leq 50 \rightarrow \text{false}$			

r	A	B	$s = \sigma_{B \neq 20}(r)$	s	A	B
1	20	20	$20 \neq 20 \rightarrow \text{false}$			
2	NULL	50	$\text{NULL} \neq 20 \rightarrow \text{unknown}$	3	60	
3	60	60	$60 \neq 20 \rightarrow \text{true}$			

SQL - Selekcija i NULL vrijednosti

student	matBr	prez	postBr
100	Kolar	52000	
102	Horvat	10000	
105	Novak	NULL	
107	Ban	10000	

$\sigma_{\text{postBr} = 10000}$

SELECT * FROM student
WHERE postBr = 10000;

matBr	prez	postBr
102	Horvat	10000
107	Ban	10000

$\sigma_{\text{postBr} \neq 10000}$

SELECT * FROM student
WHERE postBr <> 10000;

matBr	prez	postBr
100	Kolar	52000

- gdje je Novak ?

SQL - operatori usporedbe IS NULL, IS NOT NULL

student	matBr	prez	postBr
100	Kolar	52000	
102	Horvat	10000	
105	Novak	NULL	
107	Ban	10000	

- U SQL-u nije dopušteno operatore usporedbe $<, \leq, =, \neq, >, \geq$ koristiti u kombinaciji s "konstantom" NULL (npr. $=\text{NULL}$, $<>\text{NULL}$, ...)

SELECT * FROM student
WHERE postBr = NULL;

Neispravna naredba

SELECT * FROM student
WHERE postBr IS NULL;

matBr	prez	postBr
105	Novak	NULL

SELECT * FROM student
WHERE postBr IS NOT NULL;

matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
107	Ban	10000

- Rezultat logičkog izraza X IS NULL ili logičkog izraza X IS NOT NULL je uvijek ili *true* ili *false*

Trovalentna logika

- Osnovne logičke operacije - tablice istinitosti u prisustvu logičke vrijednosti *unknown*

AND	true	unknown	false
true	true	unknown	false
unknown	unknown	unknown	false
false	false	false	false

OR	true	unknown	false
true	true	true	true
unknown	true	unknown	unknown
false	true	unknown	false

NOT	true	false
true	false	true
unknown	unknown	unknown
false	true	false

Selekcija, logički operatori i NULL vrijednosti

$s = \sigma_{B \leq 50 \wedge C \neq 300}(r)$

r	A	B	C	$\text{NULL} \leq 50 \wedge 100 \neq 300 \rightarrow \text{unknown} (*)$
1	NULL	100	100	
2	20	200	200	$20 \leq 50 \wedge 200 \neq 300 \rightarrow \text{true}$
3	30	300	300	$30 \leq 50 \wedge 300 \neq 300 \rightarrow \text{false}$
4	40	NULL	300	$40 \leq 50 \wedge \text{NULL} \neq 300 \rightarrow \text{unknown}$
5	50	500	500	$50 \leq 50 \wedge 500 \neq 300 \rightarrow \text{true}$
6	60	NULL	300	$60 \leq 50 \wedge \text{NULL} \neq 300 \rightarrow \text{false} (**)$

- * $\text{NULL} \leq 50 \rightarrow \text{unknown}$; $100 \neq 300 \rightarrow \text{true}$; $\text{unknown} \wedge \text{true} \rightarrow \text{unknown}$
- ** $60 \leq 50 \rightarrow \text{false}$; $\text{NULL} \neq 300 \rightarrow \text{unknown}$; $\text{false} \wedge \text{unknown} \rightarrow \text{false}$

s	A	B	C
2	20	200	
5	50	500	

SQL - Logički operatori i NULL vrijednosti

bodovi	mbr	prez	bodLab	bodMI
	101	Novak	6	NULL
	103	Ban	NULL	NULL
	105	Horvat	12	44.0
	107	Kolar	NULL	85.0
	109	Pevec	20	15.0

Ovaj upit ne daje zadovoljavajući rezultat

```
SELECT *
FROM bodovi
WHERE bodLab < 10
OR bodMI + bodLab < 50;
```

```
SELECT *
FROM bodovi
WHERE bodLab IS NULL
OR bodMI IS NULL
OR bodLab < 10
OR bodLab + bodMI < 50;
```

mbr	prez	bodLab	bodMI
101	Novak	6	NULL
109	Pevec	20	15.0

mbr	prez	bodLab	bodMI
101	Novak	6	NULL
103	Ban	NULL	NULL
107	Kolar	NULL	85.0
109	Pevec	20	15.0

- Za prolaz je potrebno barem 10 bodova iz labosa i barem 50 bodova ukupno. Studentima koji nisu dolazili na labos ili izlazili na međuispite upisana je NULL vrijednost. Ispisati studente koji **nisu položili** ispit.

NULL vrijednosti i skupovi

- Neka skup S sadrži vrijednosti: $S = \{1, 2, 3, \text{NULL}\}$
- NULL vrijednost je nepoznata, ali može poprimiti i neku od vrijednosti 1, 2 ili 3
 - kardinalnost skupa S je neodređena (može biti 3 ili 4)
 - narušena je definicija skupa (u skupu nije dozvoljena pojava dviju ili više jednakih vrijednosti)
 - što je logička vrijednost suda $\text{NULL} \in S \rightarrow \text{unknown}$
 - što je logička vrijednost suda $4 \in S \rightarrow \text{unknown}$

NULL vrijednosti i skupovi

- Sustavi za upravljanje bazama podataka nisu u stanju međusobno razlikovati NULL vrijednosti, stoga se kao konvencija koristi sljedeći model rukovanja s NULL vrijednostima u skupovima:
 - dopuštena je pojava jedne i samo jedne NULL vrijednosti u skupu
 - element **e** je **kopija** jednog od elemenata u skupu:
 - ako vrijednost elementa **e** nije NULL, a u skupu postoji element s jednakom vrijednošću
- iii
 - ako vrijednost elementa **e** jest NULL, a u skupu S već postoji element s NULL vrijednošću

Kopija n-torke

- elementi relacije su n-torke
- Definicija kopije n-torke:
 - neka su t_1 i t_2 n-torke definirane na shemi $\{A_1, A_2, \dots, A_n\}$
 - $t_1 = \langle d_1, d_2, \dots, d_n \rangle$, $t_2 = \langle e_1, e_2, \dots, e_n \rangle$
 - n-torka t_1 je kopija n-torke t_2 ako i samo ako $\forall i, 1 \leq i \leq n$, vrijedi: $(d_i = e_i) \vee (d_i \text{ jest NULL} \wedge e_i \text{ jest NULL})$
- neformalno: ako su vrijednosti korespondentnih atributa n-torki ili jednake ili su obje NULL

Kopija n-torke

- Primjer:

osoba					student			
mbr	ime	prez	postBr		mbr	ime	prez	postBr
100	Ivan	Novak	10000	← nije kopija	100	Ivan	Novak	NULL
102	Ana	Horvat	21000	← jest kopija	102	Ana	Horvat	21000
103	Tea	Ban	52000	← nije kopija	103	Tea	Ban	21000
105	NULL	Kolar	NULL	← jest kopija	105	NULL	Kolar	NULL

Unija, razlika i presjek - NULL vrijednosti

- unija, razlika i presjek** su skupovske operacije: pri usporedbi elemenata (n-torki) treba voditi računa o definiciji kopije n-torke

r	A	B	C
1	a	α	
2	b	NULL	
3	NULL	γ	
4	NULL	NULL	

s	A	B	C
1	a	α	
2	b	NULL	
3	c	γ	
4	NULL	NULL	

$r \cup s$	A	B	C
1	a	α	
2	b	NULL	
3	NULL	γ	
3	c	γ	
4	NULL	NULL	

$r \cap s$	A	B	C
1	a	α	
2	b	NULL	
4	NULL	NULL	

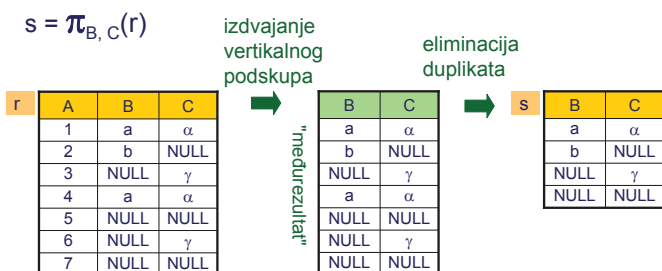
$r \setminus s$	A	B	C
3	NULL	γ	

$s \setminus r$	A	B	C
3	c	γ	

Projekcija - NULL vrijednosti

- pri obavljanju operacije **projekcije** potrebno je u fazi eliminacije duplikata voditi računa o definiciji kopije n-torke

$$s = \pi_{B,C}(r)$$



Kartezijev produkt - NULL vrijednosti

- pri obavljanju operacije **Kartezijevog produkta** NULL vrijednosti nemaju utjecaja

A	B	C
1	a	α
2	b	NULL
3	NULL	NULL

E	F
1	NULL
NULL	f

A	B	C	E	F
1	a	α	1	NULL
2	b	NULL	1	NULL
3	NULL	NULL	1	NULL
1	a	α	NULL	f
2	b	NULL	NULL	f
3	NULL	NULL	NULL	f

Spajanje uz uvjet i spajanje s izjednačavanjem - NULL vrijednosti

- pri obavljanju operacija **spajanja uz uvjet** i **spajanja s izjednačavanjem** potrebno je voditi računa o tome da se spajaju samo one n-torke za koje uvjet spajanja ima logičku vrijednost istina (*true*)

mogućnost = linija ▷◁ zrakoplov
dolet ≥ udaljenost

let	udaljenost
CA-825	700
LH-412	NULL
BA-722	4100
CA-311	13000

tip	dolet
B747	NULL
A320	5400
DC-9	3100

let	udaljenost	tip	dolet
CA-825	700	A320	5400
CA-825	700	DC-9	3100
BA-722	4100	A320	5400

Prirodno spajanje - NULL vrijednosti

- slično, pri obavljanju operacije **prirodnog spajanja** potrebno je voditi računa o tome da se spajaju samo one n-torke za koje uvjet spajanja ima logičku vrijednost istina (*true*)

pbr	nazMjesto	sifZup
42000	Varaždin	7
42230	Ludbreg	NULL
42220	Novi Marof	7

sifZup	nazZup
7	Varaždinska
NULL	Istarska

mjestouZupaniji = mjesto ▷◁ zupanija

pbr	nazMjesto	sifZup	nazZup
42000	Varaždin	7	Varaždinska
42220	Novi Marof	7	Varaždinska

- n-torka <42230, Ludbreg, NULL> neće se spojiti s n-torkom <NULL, Istarska> jer je rezultat usporedbe **NULL=NULL** → *unknown*

Agregacija - NULL vrijednosti

- ako su sve vrijednosti za koje se izračunava agregatna funkcija NULL vrijednosti, ili ako se agregatna funkcija izračunava za prazan skup vrijednosti
 - rezultat agregatne funkcije COUNT je nula
 - rezultat ostalih agregatnih funkcija je NULL
- ako među vrijednostima za koje se izračunava agregatna funkcija postoje vrijednosti koje nisu NULL vrijednosti
 - agregatna funkcija se izračunava tako da se NULL vrijednosti zanemaruju (ne uzimaju se u obzir pri izračunavanju)

Agregacija - NULL vrijednosti

mbrStud	nazPred	ocjena
100	Matematika	NULL
101	Matematika	4
102	Matematika	3
103	Matematika	3
100	Fizika	NULL
101	Fizika	3

SELECT COUNT(mbrStud) AS broj1
FROM ispit;

broj1
6

SELECT COUNT(ocjena) AS broj2
FROM ispit;

broj2
4

SELECT COUNT(ocjena) AS broj3 FROM ispit
WHERE mbrStud = 100;

broj3
0

SELECT COUNT(ocjena) AS broj4 FROM ispit
WHERE mbrStud = 200;

broj4
0

SELECT AVG(ocjena) AS broj5 FROM ispit;

broj5
3.25

SELECT AVG(ocjena) AS broj6 FROM ispit
WHERE mbrStud = 100;

broj6
NULL

SELECT AVG(ocjena) AS broj7 FROM ispit
WHERE mbrStud = 200;

broj7
NULL

Agregatna funkcija COUNT(*)

- Agregatna funkcija COUNT(*imeAtributa*)
 - broji n-torke u kojima vrijednost atributa *imeAtributa* nije NULL vrijednost
- Agregatna funkcija COUNT(*)
 - broji n-torke zanemarujući njihov sadržaj

ispit	mbrStud	nazPred	ocjena
	100	Matematika	NULL
	101	Matematika	4
	102	Matematika	3
	103	Matematika	3
	100	Fizika	NULL
	101	Fizika	3

```
SELECT COUNT(ocjena) AS brojOcj
, COUNT(*) AS brojRedaka
FROM ispit;
```

brojOcj	brojRedaka
4	6

- Ne postoji agregatna funkcija COUNT(DISTINCT *)

Grupiranje - NULL vrijednosti

- pri obavljanju operacije grupiranja, grupiranje n-torki se obavlja tako da se vodi računa o definiciji kopije n-torki
 - ako se grupiranje obavlja prema atributima iz skupa X, tada u istu grupu ulaze one n-torke čije su X-vrijednosti međusobne kopije

```
SELECT akGod, nazPred, AVG(ocjena) AS prosj
FROM ispit
GROUP BY akGod, nazPred;
```

ispit	mbrStud	akGod	nazPred	ocjena
t ₁	100	2005	NULL	3
t ₂	101	NULL	NULL	5
t ₃	102	2005	NULL	2
t ₄	103	2006	Fizika	3
t ₅	100	NULL	NULL	5
t ₆	101	2006	Fizika	5
t ₇	102	2005	NULL	2

X = { akGod, nazPred }

t₁(X), t₃(X) i t₇(X) su međusobne kopije

t₂(X) i t₅(X) su međusobne kopije

t₄(X) i t₆(X) su međusobne kopije

akGod	nazPred	pros
2005	NULL	2.3333
NULL	NULL	5
2006	Fizika	4

Vanjsko spajanje - uvod

student	matBr	prez
	101	Kolar
	102	Horvat
	103	Novak

upisanPred	matBr	nazPred
	101	Matematika
	101	Fizika
	101	Programiranje
	102	Fizika

upisani = student ▷◁ upisanPred

upisani	matBr	prez	nazPred
	101	Kolar	Matematika
	101	Kolar	Fizika
	101	Kolar	Programiranje
	102	Horvat	Fizika

- n-torka <103, Novak> neće se pojaviti u rezultatu jer u relaciji **upisanPred** ne postoji niti jedna n-torka koja zadovoljava uvjet spajanja s tom n-torkom

Vanjsko spajanje - uvod

mjesto	pbr	nazMjesto	sifZup
	42000	Varaždin	7
	42230	Ludbreg	NULL
	42220	Novi Marof	7

zupanija	sifZup	nazZup
	7	Varaždinska
	NULL	Istarska

mjestouZupaniji = mjesto ▷◁ zupanija

mjestouZupaniji	pbr	nazMjesto	sifZup	nazZup
	42000	Varaždin	7	Varaždinska
	42220	Novi Marof	7	Varaždinska

- n-torka <42230, Ludbreg, NULL> neće se pojaviti u rezultatu jer u relaciji **zupanija** ne postoji niti jedna n-torka koja zadovoljava uvjet spajanja (a ne može je niti biti, jer sifZup ima NULL vrijednost)

Lijevo vanjsko spajanje (Left outer join)

- sve n-torke relacije **student** će se pojaviti u rezultatu spajanja ako se primijeni operacija **lijevog vanjskog spajanja**

student	matBr	prez
	101	Kolar
	102	Horvat
	103	Novak

upisanPred	matBrSt	nazPred
	101	Matematika
	101	Fizika
	101	Programiranje
	102	Fizika

upisano = student *▷◁ upisanPred
matBr = matBrSt

upisano	matBr	prez	matBrSt	nazPred
	101	Kolar	101	Matematika
	101	Kolar	101	Fizika
	101	Kolar	101	Programiranje
	102	Horvat	102	Fizika
	103	Novak	NULL	NULL

- n-torkama "lijeve" relacije za koje ne postoje odgovarajuće n-torke u "desnoj" relaciji se kao vrijednosti atributa iz "desne" relacije postavljaju NULL vrijednosti

Lijevo vanjsko spajanje (Left outer join)

mjesto	pbr	nazMjesto	sifZupMj
	42000	Varaždin	7
	42230	Ludbreg	NULL
	42220	Novi Marof	7

zupanija	sifZup	nazZup
	7	Varaždinska
	NULL	Istarska

mjestouZupaniji = mjesto *▷◁ zupanija
sifZupMj = sifZup

mjestouZupaniji	pbr	nazMjesto	sifZupMj	sifZup	nazZup
	42000	Varaždin	7	7	Varaždinska
	42230	Ludbreg	NULL	NULL	NULL
	42220	Novi Marof	7	7	Varaždinska

SQL - Lijevo vanjsko spajanje (*Left outer join*)

mjesto	pbr	nazMjesto	sifZupMj	zupanija	sifZup	nazZup
	42000	Varaždin	7		7	Varaždinska
	42230	Ludbreg	NULL		NULL	Istarska
	42220	Novi Marof	7			

mjesto $\triangleright \triangleleft$ zupanija
sifZupMj = sifZup

```
SELECT mjesto.*, zupanija.* ← ili SELECT *
FROM mjesto LEFT OUTER JOIN zupanija
ON sifZupMj = sifZup;
```

pbr	nazMjesto	sifZupMj	sifZup	nazZup
42000	Varaždin	7	7	Varaždinska
42230	Ludbreg	NULL	NULL	NULL
42220	Novi Marof	7	7	Varaždinska

Desno vanjsko spajanje (*Right outer join*)

- sve n-torke relacije **nastavnik** će se pojaviti u rezultatu spajanja ako se primijeni operacija **desnog vanjskog spajanja**

student	mbrSt	prezSt	temaSt	nastavnik	sifNast	prezNast	temaNast
	101	Horvat	Tranzistori		202	Ban	Teslini izumi
	103	Novak	Teslini izumi		204	Toplek	Elektrane
	105	Kolar	Teorija kaosa		206	Oreb	Teslini izumi
					209	Pernar	Teorija kaosa

moguciMent = student $\triangleright \triangleleft$ * nastavnik
temaSt = temaNast

- n-torkama "desne" relacije za koje ne postoje odgovarajuće n-torke u "lijevoj" relaciji se kao vrijednosti atributa iz "lijeve" relacije postavljaju NULL vrijednosti

moguciMent	mbrSt	prezSt	temaSt	sifNast	prezNast	temaNast
	103	Novak	Teslini izumi	202	Ban	Teslini izumi
	NULL	NULL	NULL	204	Toplek	Elektrane
	103	Novak	Teslini izumi	206	Oreb	Teslini izumi
	105	Kolar	Teorija kaosa	209	Pernar	Teorija kaosa

SQL - Desno vanjsko spajanje (*Right outer join*)

student	mbrSt	prezSt	temaSt	nastavnik	sifNast	prezNast	temaNast
	101	Horvat	Tranzistori		202	Ban	Teslini izumi
	103	Novak	Teslini izumi		204	Toplek	Elektrane
	105	Kolar	Teorija kaosa		206	Oreb	Teslini izumi
					209	Pernar	Teorija kaosa

student $\triangleright \triangleleft$ * nastavnik
temaSt = temaNast

```
SELECT student.*, nastavnik.* ← ili SELECT *
FROM student RIGHT OUTER JOIN nastavnik
ON temaSt = temaNast;
```

mbrSt	prezSt	temaSt	sifNast	prezNast	temaNast
103	Novak	Teslini izumi	202	Ban	Teslini izumi
NULL	NULL	NULL	204	Toplek	Elektrane
103	Novak	Teslini izumi	206	Oreb	Teslini izumi
105	Kolar	Teorija kaosa	209	Pernar	Teorija kaosa

Puno vanjsko spajanje (*Full outer join*)

- sve n-torke iz obje relacije će se pojaviti u rezultatu spajanja ako se primijeni operacija **punog vanjskog spajanja**

student	mbrSt	prezSt	temaSt	nastavnik	sifNast	prezNast	temaNast
	101	Horvat	Tranzistori		202	Ban	Teslini izumi
	103	Novak	Teslini izumi		204	Toplek	Elektrane
	105	Kolar	Teorija kaosa		206	Oreb	Teslini izumi
					209	Pernar	Teorija kaosa

student $\ast \triangleright \triangleleft$ * nastavnik
temaSt = temaNast

moguciMent	mbrSt	prezSt	temaSt	sifNast	prezNast	temaNast
	101	Horvat	Tranzistori	NULL	NULL	NULL
	103	Novak	Teslini izumi	202	Ban	Teslini izumi
	NULL	NULL	NULL	204	Toplek	Elektrane
	103	Novak	Teslini izumi	206	Oreb	Teslini izumi
	105	Kolar	Teorija kaosa	209	Pernar	Teorija kaosa

SQL - Puno vanjsko spajanje (*Full outer join*)

student	mbrSt	prezSt	temaSt	nastavnik	sifNast	prezNast	temaNast
	101	Horvat	Tranzistori		202	Ban	Teslini izumi
	103	Novak	Teslini izumi		204	Toplek	Elektrane
	105	Kolar	Teorija kaosa		206	Oreb	Teslini izumi
					209	Pernar	Teorija kaosa

student $\ast \triangleright \triangleleft$ * nastavnik
temaSt = temaNast

```
SELECT student.*, nastavnik.* ← ili SELECT *
FROM student FULL OUTER JOIN nastavnik
ON temaSt = temaNast;
```

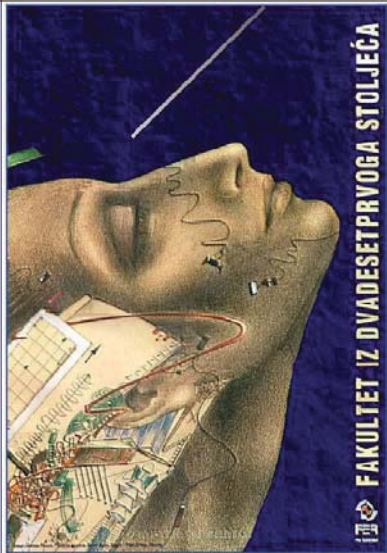
Prirodno vanjsko spajanje

- kod vanjskog spajanja uz uvjet i vanjskog spajanja s izjednačavanjem u shemi rezultata se pojavljuju svi atributi obje relacije
- kod prirodnog lijevog vanjskog spajanja iz sheme rezultata se izbacuju istoimeni atributi desnog operanda (jer ionako mogu poprimiti ili vrijednosti jednake vrijednostima korespondentnih atributa lijevog operanda ili NULL vrijednosti)
- kod prirodnog desnog vanjskog spajanja iz rezultata se izbacuju istoimeni atributi lijevog operanda (jer ionako mogu poprimiti ili vrijednosti jednake vrijednostima korespondentnih atributa desnog operanda ili NULL vrijednosti)
- kod prirodnog punog vanjskog spajanja potrebno je u shemi rezultata zadržati sve attribute obje relacije, te primijeniti operator preimenovanja atributa

Baze podataka

Predavanja
ožujak 2008.

4. SQL (1. dio)



SQL - Uvod

- objedinjuje funkcije jezika za definiciju podataka (DDL) i jezika za rukovanje podacima (DML)
- razvoj započeo 70-tih godina
 - IBM San José Research Laboratory (California, USA)
- Structured Query Language* je standardni jezik relacijskih baza podataka (*database language*)
 - 1986. godine - SQL-86 ili SQL1 (prva verzija standarda)
 - 1992. godine - SQL-92 ili SQL2
 - 1999. godine - SQL:1999
 - 2003. godine - SQL:2003
- proizvođači komercijalnih sustava često ugrađuju i svoje nestandardne DDL i DML naredbe
 - programski kod postaje neprenosiv između različitih SQL sustava
 - otežava se usaglašavanje oko budućih standarda.

SQL - Uvod

- neproceduralnost - naredbom je dovoljno opisati što se želi dobiti kao rezultat - nije potrebno definirati kako do tog rezultata doći
- u SUBP ugrađeni optimizator upita pronalazi najefikasniji način obavljanja upita

županija		mjesto		
sifZup	nazZup	pbr	nazMjesto	sifZup
2	Primorsko-goranska	42000	Varaždin	7
7	Varaždinska	51000	Rijeka	2
4	Istarska	52100	Pula	4
		42230	Ludbreg	7

- ispisati podatke o mjestima u Varaždinskoj županiji. Rezultate poredati prema nazivu mjesta

```
SELECT mjesto.* FROM mjesto, zupanija
WHERE mjesto.sifZup = zupanija.sifZup
AND nazZup = 'Varaždinska'
ORDER BY nazMjesto;
```

SQL - Vrste objekata

• Baza podataka	<i>Database</i>
• Relacija (tablica)	<i>Table</i>
• Atribut (stupac, kolona)	<i>Column</i>
• Virtualna tablica (pogled)	<i>View</i>
• Sinonim	<i>Synonym</i>
• Integritetsko ograničenje	<i>Constraint</i>
• Indeksi	<i>Index</i>
• Pohranjena procedura	<i>Stored Procedure</i>
• Varijabla u pohranjenoj proceduri	<i>SPL variable</i>
• Okidač	<i>Trigger</i>

SQL - Identifikatori

- Identifikatori (imena objekata) se formiraju iz slova, znaka '_' i znamenki. Prvi znak od ukupno 128 značajnih (signifikantnih) znakova mora biti slovo ili znak '_'

- ispravno formirani identifikatori

```
stud
ispiti2000godine
stud_ispit
_1mjesece
```

- neispravno formirani identifikatori

```
_11.mjesece
11mjesece
stud-ispit
```

SQL - Rezervirane riječi

- SQL je "neosjetljiv" (*case insensitive*) na razliku između velikih i malih slova kada su u pitanju rezervirane riječi (SELECT, UPDATE, DELETE, FROM, WHERE, ...) i identifikatori

```
SELECT * FROM mjesto
WHERE sifZupanija = 7
```

≡

```
select * FrOm MJesto
wHERE SIFZupanIJA = 7
```

- Međutim, razlika između velikih i malih slova postoji kad su u pitanju nizovi znakova

'Ivan' ≠ 'IVAN'

SQL - Format naredbi

- SQL je jezik slobodnog formata naredbi (jednako kao C)

```
SELECT * FROM mjesto
WHERE sifZupanija = 7
```

≡

```
SELECT
*
FROM
mjesto WHERE
sifZupanija = 7
```

SQL - Korištenje komentara

- "blok komentari" (jednako kao u programskom jeziku C)
 - dio teksta omeđen oznakama `/* i */`

```
/* ovo je komentar koji se
   proteže kroz više redaka teksta */
```

- "linijski komentari"
 - mjesto u retku na kojem se nalaze znakovi `--` predstavlja početak komentara koji se proteže do kraja retka

```
-- ovo je komentar
SELECT * FROM mjesto -- ovo je komentar
WHERE pbr = 10000 -- ovo je komentar
```

SQL - Tipovi podataka

INTEGER

- cijeli broj pohranjen u 4 bajta u aritmetici dvojnog komplementa. Dopusćeni raspon brojeva određen je intervalom

$$[-2^{n-1}, 2^{n-1} - 1] \quad n=32$$

- dakle, raspon brojeva bi trebao biti:

`[-2147483648, 2147483647]`

- u stvarnosti je manji, jer se vrijednost -2147483648 koristi za pohranu `NULL` vrijednosti. Raspon brojeva koji se mogu prikazati je:

`[-2147483647, 2147483647]`

Konstante:				
5	-30000	0	1765723712	NULL

SQL - Tipovi podataka

SMALLINT

- cijeli broj pohranjen u 2 bajta. Raspon brojeva koji se mogu prikazati je `[-32767, 32767]`

Konstante:			
5	-30000	0	NULL

CHAR(m)

- znakovni niz (*string*) s unaprijed definiranom maksimalnom duljinom $m \leq 32767$. Npr: `CHAR(24)`.

Konstante:		
'Ana'	'12345'	NULL
'Dvostruki navodnik " unutar niza'		
'Jednostruki navodnik ' ' unutar niza'		

- uočite:** koriste se **jednostruki** navodnici (drugačije nego u jeziku C)

SQL - Tipovi podataka

NCHAR(m)

- jednako kao i `CHAR` tip podatka, ali omogućava ispravno leksikografsko uređenje nizova znakova koji sadrže nacionalne kodne stranice (*character set*). Koristi se onda kada se predviđa potreba za leksikografskim poretom nizova znakova u kojima se pojavljuju specifični nacionalni znakovi (Č, Ć, Š, Đ, Ž, ...), npr. za atribut prezime

SQL - Tipovi podataka

REAL

- odgovara tipu podatka `float` u jeziku C (IEEE-754 format prikaza - jednostruka preciznost)

Konstante:				
23	-343.23	232.233E3	23.0e-24	NULL

DOUBLE PRECISION

- odgovara tipu podatka `double` u jeziku C (IEEE-754 format prikaza - dvostruka preciznost)

Konstante:				
23	-343.23	232.233E3	23.0e-302	NULL

SQL - Tipovi podataka

DECIMAL(m,n)

- ukupni broj znamenki (precision, $m \leq 32$)
- broj znamenki iza decimalne točke (scale, $n \leq m$)
- npr., DECIMAL(15, 3) predstavlja decimalni broj sa ukupno najviše 15 znamenki, od toga se najviše 3 znamenke nalaze iza decimalne točke
- razlikuje se od float ili double tipa podatka u jeziku C
 - ukoliko se za pohranu broja 1.3 koristi tip podatka DECIMAL(2,1), broj će biti pohranjen kao numeričke počke
 - ukoliko se za pohranu broja 1.3 koristi tip podatka float u jeziku C, u memoriji će se zapravo pohraniti broj 1.2523162842 (num. pogreška zbog karakteristika IEEE-754 formata pohrane)

Konstante - primjer za za DECIMAL(7, 2):

5 8.1 -12345.67 0 NULL

SQL - Tipovi podataka

DATE

- podaci ovog tipa se uvijek prikazuju u obliku datuma (npr. 18.11.2006). Interno je podatak predstavljen brojem dana proteklih od 31.12.1800 ovaj tip podatka omogućava korištenje sljedećih operacija zbrajanja i oduzimanja:
 - `dat1 - dat2` rezultat je podatak tipa INTEGER - broj dana proteklih između `dat1` i `dat2`
 - `date_add(date, interval)` rezultat je podatak tipa DATE - izračunava koji datum je `interval` dana nakon dana `date`
 - `date_sub(date, interval)` rezultat je podatak tipa DATE - izračunava koji datum je `interval` dana prije dana `date`

Konstante:

'17.2.2007' '16.07.1606' NULL

NULL vrijednost

clanoviKnjiznice

mbr	ime	prez	pbr	datRodj	adresa	zanimanje
100	Ana	Novak	10000	01.5.2001	ulica 1	NULL
105	Ivo	Kolar	21000	12.3.1973	NULL	odvjetnik
107	James	Bond	NULL	NULL	NULL	tajni agent

nije primjenjivo
(vidi datum rođenja)

nedostupno trenutno nepoznato

- Način na koji se NULL vrijednost prikazuje korisniku ovisi o programskom alatu koji se koristi. Npr:

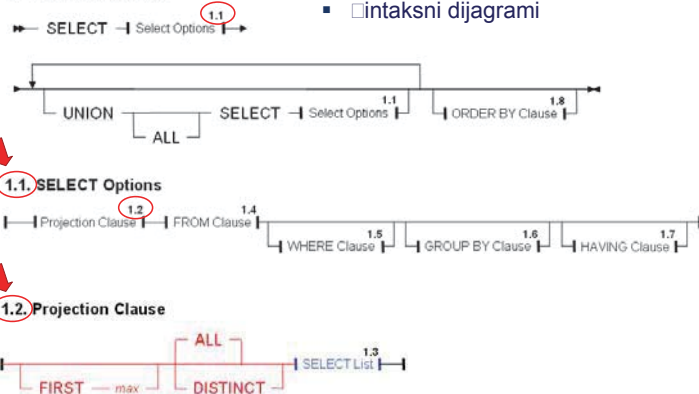
nička porana NULL vrijednosti

- NULL vrijednost se interno pohranjuje drugačije od bilo koje druge dopuštene vrijednosti (nije 0, nije 0.0, nije prazan niz, ...)
- primjer:
 - -32768 se interno koristi za prikaz NULL vrijednost kada se radi o vrijednosti atributa ili varijable tipa SMALLINT. Zato je dopušteni raspon vrijednosti za taj tip samo [-32767, 32767]
 - -2147483648 se interno koristi za prikaz NULL vrijednosti kada se radi o vrijednosti atributa ili varijable tipa INTEGER
- Interni prikaz NULL vrijednosti je za korisnika **nevažan** - NULL vrijednost je neovisna od tipa podatka kojeg predstavlja. Bez obzira na tip podatka, uvijek se koristi konstanta NULL

NE NULL esto ALUE 10000, 'adreb', -32768
NE NULL esto ALUE 10000, 'adreb', NULL

SELECT Statement

1. SELECT Statement



Projection Clause

primjeri:

student	matBr	prez	postBr
	100	Kolar	52000
	102	Horvat	10000
	105	Kolar	52000
	107	Ban	10000

SELECT ALL prez, postbr FROM student

SELECT DISTINCT prez, postbr FROM student

SELECT 2 FROM student

prez	postBr
Kolar	52000
Horvat	10000
Kolar	52000
Ban	10000

Ne zna se koje dvije n-torke će se dobiti kao prve dvije - poredak n-torki u relaciji (niti u tablici) nije definiran

Projection Clause

Primjeri:

student	matBr	prez	postBr
	100	Kolar	52000
	102	Horvat	10000
	105	Kolar	52000
	107	Ban	10000

```
SELECT * FROM student WHERE postBr = 52000
```

prez
Horvat
Ban

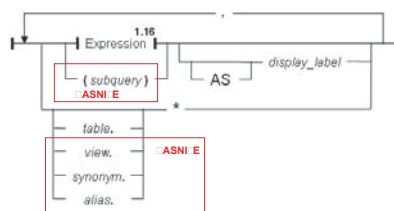
U ovom primjeru: ne zna se koje su to prve dvije i n-torke

```
SELECT matBr FROM student WHERE postBr = 10000
```

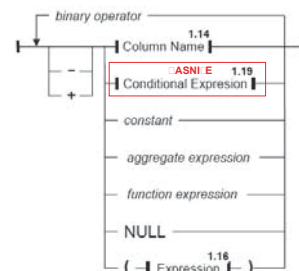
matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
105	Kolar	52000
107	Ban	10000

SELECT List

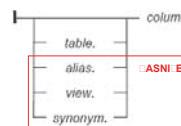
1.3 SELECT List



1.16. Expression



1.14. Column Name



SELECT List

Primjer:

mjesto	pbr	naz	jesto	sifZup
	42000	paraždin	7	
	52100	ula	4	

```
SELECT mjesto, pbr, naz, jesto, sifZup FROM mjesto
```

pbr	pbr	naz	jesto	sifZup	pbr	pbr	naz	jesto	sifZup
42000	42000	paraždin	7		42000	42000	paraždin	7	
52100	52100	ula	4		52100	52100	ula	4	

U ovom primjeru rezultat nije relacija

Operatori (Expression)

Unarni operatori

+

Binarni operatori

+

-

*

/

||

AND

OR

NOT

IS

LIKE

IN

AS

ON

TO

FROM

WHERE

GROUP BY

HAVING

ORDER BY

ASC

DESC

NULL

IS NULL

IS NOT NULL

IS DISTINCT FROM

IS NOT DISTINCT FROM

IS TRUE

IS FALSE

IS UNKNOWN

IS NOT UNKNOWN

IS NULL

IS NOT NULL

IS TRUE

IS FALSE

IS UNKNOWN

IS NOT UNKNOWN

IS NULL

IS NOT NULL

IS TRUE

IS FALSE

IS UNKNOWN

IS NOT UNKNOWN

IS NULL

IS NOT NULL

IS TRUE

IS FALSE

IS UNKNOWN

IS NOT UNKNOWN

Operatori (primjeri)

Unarni, binarni operatori i konstante

```
SELECT * FROM bodovi WHERE bodLab = 'A' AND bodLab = 'B'
```

bodovi	mbr	ime	prez	bodLab	bodLab
	100	Ana	Novak	12	67.2
	107	Ivo	Ban	17	54.3

```
SELECT * FROM bodovi WHERE bodLab = 'A' AND bodLab = 'B' AND bodLab = 'C'
```

mbr	(e.pression)	(e.pression)
100	71.2	0.712
107	71.3	0.713

```
SELECT * FROM bodovi WHERE bodLab = 'A' AND bodLab = 'B' AND bodLab = 'C' AND bodLab = 'D'
```

mbr	(e.pression)
100	-12
107	-17

```
SELECT * FROM bodovi WHERE bodLab = 'A' AND bodLab = 'B' AND bodLab = 'C' AND bodLab = 'D' AND bodLab = 'E'
```

mbr	(e.pression)
100	Ana Novak
107	Ivo Ban

```
SELECT * FROM bodovi WHERE bodLab = 'A' AND bodLab = 'B' AND bodLab = 'C' AND bodLab = 'D' AND bodLab = 'E' AND bodLab = 'F'
```

(e.pression)
100 - Ana
107 - Ivo

Funkcije (function expression)

- AB
- AC
- AD
- AE
- AF
- AG
- AH
- AI
- AL
- AM
- AN
- AO
- AP
- AQ
- AR
- AS
- AT
- AV
- AW
- AX
- AY
- AZ

Funkcije (function expression)

ABS (num_expression)

- računa apsolutnu vrijednost izraza

num_expression mora biti numerički tip podatka (INTEGER, DECIMAL, FLOAT, ...)

rezultat funkcije tip podatka ovisi o tipu podatka ulaznog argumenta

MOD (dividend, divisor)

- računa ostatak cjelobrojnog dijeljenja djeljenika i djelitelja (djelitelj ne smije biti 0)
- pri računanju uzima se samo cjelobrojni dio argumenata

dividend (djeljenik) numerički tip podatka (INTEGER, DECIMAL, FLOAT, ...)

divisor (djelitelj) numerički tip podatka (INTEGER, DECIMAL, FLOAT, ...)

rezultat funkcije cijeli broj

Funkcije (function expression)

ROUND (expression[, rounding_factor])

- zaokružuje vrijednost izraza **expression**
- ukoliko se ne navede **rounding_factor** uzima se da je njegova vrijednost 0

expression izraz koji se obrađuje

numerički tip podatka (INTEGER, DECIMAL, FLOAT, ...)

rounding_factor decimalni broj koji određuje broj decimala

cjelobrojni tip podatka

rezultat funkcije tip podatka ovisi o tipu podatka ulaznog argumenta (**expression**)

Funkcije (function expression)

SUBSTRIN (source_string FROM start_position [FOR length])

- vraća podniz zadanog niza
- ukoliko se **length** ne navede vraća se podniz koji počinje na **start_position**, a završava gdje i niz **source_string**

source_string zadani niz čiji se podniz traži funkcijom
mora biti izraz tipa niza znakova

start_position broj koji predstavlja poziciju prvog znaka podniza u zadanom nizu **source_string**
mora biti izraz cjelobrojnog tipa

length(duljina) broj znakova koje funkcija treba vratiti počevši od **start_position**
mora biti izraz cjelobrojnog tipa

Funkcije (function expression)

UPPER (expression)

- sva mala slova (a-z) koja se pojavljuju u zadanom nizu **expression** zamjenjuje odgovarajućim velikim slovima (A-Z)

LOWER (expression)

- sva velika slova (A-Z) koja se pojavljuju u zadanom nizu **expression** zamjenjuje odgovarajućim malim slovima (a-z)

expression zadani niz nad kojim se vrši pretvorba slova
mora biti izraz tipa niza znakova

Funkcije (function expression)

TRIM(source_expression)

- funkcija vraća niz znakova koji nastaje tako da se s početka i kraja niza **source_expression** izbace sve praznine

expression zadani niz iz kojeg funkcija izbacuje praznine
mora biti izraz tipa niza znakova

Funkcije (function expression)

CHAR_LENGTH(expression)

- funkcija vraća broj znakova u zadanom nizu **expression** uključujući i prateće praznine

OCTET_LENGTH(expression)

- funkcija vraća broj bite-ova zadanog niza **expression** uključujući i prateće praznine

expression mora biti izraz tipa niza znakova

Funkcije (primjeri) Funkcije s nizovima

```

SELECT ALE student
FROM ba
, i_e N A 25
, rezije N A 25

```

student

jmbag	ime	prezime
0036368145	omislav	Božanić
003636216	inda	Kekez

Ispisuje korisničko ime i broj znakova koji ga čine

```

SELECT U E
FROM ALEN
FROM student

```

(e.pression)	(e.pression)
bpadmin	7
bpadmin	7

Ispisuje broj znakova u imenu i broj znakova u imenu iz kojeg su izbačene praznine

```

SELECT ALEN i_e
FROM ALEN i_e
FROM student

```

(e.pression)	(e.pression)
25	8
25	5

Ispisuje broj znakova i broj bajtova koji čine prezime iz kojeg su izbačene praznine

```

SELECT ALEN rezije
FROM ELEN rezije
FROM student

```

(e.pression)	(e.pression)
7	
5	5

Zbog utf8:
ž-2 okteta
č-2 okteta

Funkcije (primjeri) Funkcije s datumom

```

SELECT ALE nastavnik
FROM sifNastavnik N E
, datu a oslen d
, datu a oslenDo D A E
FROM nastavnik

```

nastavnik

sifNastavnik	datumZaposlen_d	datumZaposlen_o
1	22.01.15	20.02.2007
2	01.06.2004	01.03.2007

Napomena: pretpostavka je da se sljedeći upit izveo dana 27.02.2007.

Broj dana koji je protekao nakon prestanka zaposlenja nastavnika

```

SELECT sifNastavnik
FROM DA - datu a oslenDo
FROM nastavnik

```

sifNastavnik	(e.pression)
1	7
2	-2

Ispisuje dan, mjesec i godinu datuma zaposlenja nastavnika

```

SELECT DA datu a oslen d
FROM N datu a oslen d
, EA datu a oslen d
FROM nastavnik

```

(e.pression)	(e.pression)	(e.pression)
22	1	15
1	6	2004

Funkcije (primjeri) Funkcije s datumom

```

SELECT ALE nastavnik
FROM sifNastavnik N E
, datu a oslen d
, datu a oslenDo D A E
FROM nastavnik

```

nastavnik

sifNastavnik	datumZaposlen_d	datumZaposlen_o
1	22.01.15	20.02.2007
2	01.06.2004	01.03.2007

Ispisuje redni broj dana u tjednu datuma prestanka zaposlenja nastavnika

```

SELECT sifNastavnik
FROM EE DA datu a oslenDo
FROM nastavnik

```

sifNastavnik	(e.pression)
1	2
2	4

Ispisuje datum koji odgovara sljedećem danu nakon prestanka zaposlenja nastavnika

```

SELECT D N datu a oslenDo
FROM DA datu a oslenDo 1
, EA datu a oslenDo
FROM nastavnik

```

(e.pression)
21.02.2007
02.03.2007

Funkcije i NULL vrijednosti

- Neka je binarni operator $\alpha \in \{=, <, >, <=, >=, <>, <=>, >=<, <=<, >=>\}$ a ϕ i ψ su izrazi
 - ako jedan ili oba operanda ϕ , ψ poprimaju NULL vrijednost, tada je rezultat izraza α također NULL vrijednost
- Neka je unarni operator $\beta \in \{+, -, *, /, \text{etc.}\}$ a ϕ je izraz
 - ako operand ϕ poprima NULL vrijednost, tada je rezultat izraza β također NULL vrijednost
- lično vrijedi i za funkcije
 - ukoliko se kao jedan ili više argumenata funkcije zada NULL vrijednost, rezultat funkcije će također biti NULL vrijednost

Funkcije i NULL vrijednosti (primjer)

bodovi	mbr	prez	bodLab
	101	Novak	12
	103	Ban	NULL
	107	N	21
	10	Kolar	NULL

```

SELECT mbr
FROM bodLab, 10 A ostatak
, U N rez 1 2 A odniz
FROM bodovi

```

mbr	ostatak	podniz
101	2	No
103	NULL	a
107	1	NULL
10	NULL	o

WHERE Clause

1.1. SELECT Options



1.5. WHERE Clause

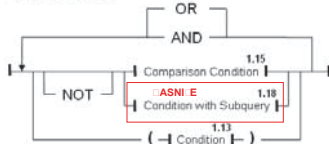


- rijednosti svake n-torke iz relacije ta se uvrstavaju u $oniti$ (a to je u stvari predikat). Ako je dobiveni sud istinit ($true$), n-torka se pojavljuje u rezultatu
- oguči rezultati izračunavanja uvjeta: $true$, $false$, $onon$

Condition (ponavljanje)

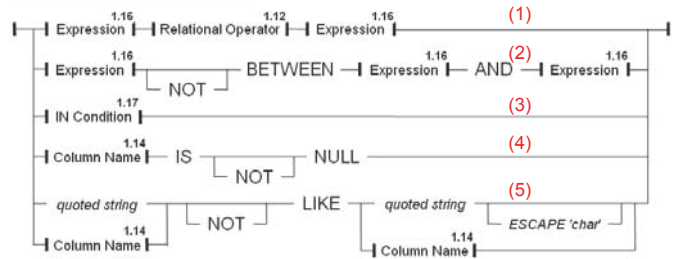
- SELECT** *List* **FROM** *table* [**WHERE** *Condition*]
- uvjet (*Condition*) se sastoji od operandi i operatora
 - operandi su:
 - imena atributa iz relacije *table*
 - konstante
 - operatori su:
 - operatori usporedbe: `<` `<=` `>` `>=` `=` `<>`
 - logički operatori: **AND** **OR** **NOT**

1.13. Condition



Uvjet usporedbe (Comparison Condition)

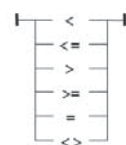
1.15. Comparison Condition



Uvjet usporedbe (Comparison Condition) (□)

Expression (1.16) Relational Operator (1.12) Expression (1.16)

1.12. Relational Operator



student	matBr	ime	prez	postBr
	100	Ivan	Kolar	52000
	102	Ana	Horvat	10000
	105	Jura	Novak	21000

```
SELECT * FROM student
WHERE prez = 'Kolar'
```

matBr	ime	prez	postBr
102	Ana	Horvat	10000

Uvjet usporedbe (Comparison Condition) (□)

Expression (1.16) NOT BETWEEN Expression (1.16) AND Expression (1.16)

stanje_klad	sifArt	min	ma	stanje
	1	10	50	50
	2	20	60	30
	3	10	80	5
	4	10	10	15.0
	5	10	20	10

```
SELECT * FROM stanje_klad
WHERE stanje BETWEEN 10 AND 50
```

sifArt	min	ma	stanje
1	10	50	50
2	20	60	30

```
SELECT * FROM stanje_klad
WHERE stanje NOT BETWEEN 10 AND 50
```

sifArt	min	ma	stanje
3	10	80	5

Uvjet usporedbe (Comparison Condition) (□)

1.17. IN Condition

Expression (1.16) NOT IN (constant)

student	matBr	prez
	100	Kolar
	102	Horvat
	103	Novak
	105	Horvat
	107	Novak
	109	Ban

```
SELECT * FROM student
WHERE prez IN ('Kolar', 'Horvat')
```

matBr	prez
100	Kolar
102	Horvat
105	Horvat

```
SELECT * FROM student
WHERE prez NOT IN ('Kolar', 'Horvat')
```

matBr	prez
103	Novak
109	Ban

- ako *expression* ima vrijednost `NULL`, tada je rezultat logička vrijednost `unknown`, bez obzira na vrijednosti navedene u skupu

Uvjet usporedbe (Comparison Condition) (□)

Column Name (1.14) IS NOT NULL

student	matBr	prez	postBr
	100	Kolar	52000
	102	Horvat	10000
	105	Novak	21000
	107	Ban	10000

```
SELECT * FROM student
WHERE postBr IS NULL
```

matBr	prez	postBr
105	Novak	21000

```
SELECT * FROM student
WHERE postBr IS NOT NULL
```

matBr	prez	postBr
100	Kolar	52000
102	Horvat	10000
107	Ban	10000

Uvjetni izraz (Conditional Expression) (CASE)

```

CASE
  WHEN 5 THEN 'izvrstan'
  WHEN 4 THEN 'vrlo dobar'
  WHEN 3 THEN 'dobar'
  WHEN 2 THEN 'dovoljan'
  WHEN 1 THEN 'nedovoljan'
  ELSE 'neispravno'
END AS opis
  
```

ispit	matBr	ocjena
	100	5
	102	3
	103	1
	107	N
	10	6

matBr	ocjena	opis
100	5	izvrstan
102	3	dobar
103	1	nedovoljan
107	N	neispravno
10	6	neispravno

- ukoliko više izraza uz WHEN zadovoljava uvjet jednakosti, rezultat izraza je *expression* naveden uz prvi WHEN koji zadovoljava uvjet
- ukoliko se ELSE dio izraza ne navede, a niti jedan izraz ne zadovoljava uvjet jednakosti, tada je rezultat izraza NULova vrijednost

Unija (UNION)

1. SELECT Statement

SELECT → Select Options →



- UNION ALL - može se graditi od jednog ili više SELECT
- UNION - uz izbacivanje duplikata (kopija n-torki)
- UNION ALL - bez izbacivanja duplikata (kopija n-torki)
- imena stupaca (atributa rezultantne relacije) određuju se na temelju imena stupaca iz prvog navedenog SELECT

Unija (UNION)

- polozioat UNION poloziorog UNION polozioiglog

polozioat	mbr	ime	prez
	100	Ivan	N
	102	Ana	Novak
	105	Rudi	Kolar
	111	ura	Horvat

poloziorog	mbr	ime	prez
	100	Ivan	N
	103	N	Ban
	105	Rudi	Kolar

polozioiglog	mbr	ime	prez
	102	Ana	Novak
	103	N	Ban
	105	Rudi	Kolar
	111	ura	Horvat

```

SELECT * FROM polozioat
UNION
SELECT * FROM poloziorog
UNION
SELECT * FROM polozioiglog
  
```

mbr	ime	prez
100	Ivan	N
102	Ana	Novak
103	N	Ban
105	Rudi	Kolar
111	ura	Horvat

Unija (UNION)

- rezultat sljedeće naredbe nije relacija

polozioat	mbr	ime	prez
	100	Ivan	N
	102	Ana	Novak
	105	Rudi	Kolar
	111	ura	Horvat

poloziorog	mbr	ime	prez
	100	Ivan	N
	103	N	Ban
	105	Rudi	Kolar

polozioiglog	mbr	ime	prez
	102	Ana	Novak
	103	N	Ban
	105	Rudi	Kolar
	111	ura	Horvat

```

SELECT * FROM polozioat
UNION ALL
SELECT * FROM poloziorog
UNION ALL
SELECT * FROM polozioiglog
  
```

mbr	ime	prez
100	Ivan	N
102	Ana	Novak
105	Rudi	Kolar
111	ura	Horvat
100	Ivan	N
103	N	Ban
105	Rudi	Kolar
102	Ana	Novak
103	N	Ban
105	Rudi	Kolar
111	ura	Horvat

Unija (UNION)

- naredba je ispravna ukoliko su korespondentni atributi istih tipova podataka (INTEGER-INTEGER, CHAR-CHAR, ...), ali odgovornost je korisnika (programera) voditi računa o unijskoj kompatibilnosti
- npr. sljedeća naredba će se obaviti, ali rezultat je besmislen

pecivo	oznaka	naziv
	ZE-33	Žemlja s makom
	R-3	erec sa sezamom

zrakoplov	oznaka	naziv
	R-3	iper R-3 Cub
	B-747	Boeing 747
	A-360	Airbus 360

```

SELECT * FROM pecivo
UNION
SELECT * FROM zrakoplov
  
```

oznaka	naziv
ZE-33	Žemlja s makom
R-3	erec sa sezamom
R-3	iper R-3 Cub
B-747	Boeing 747
A-360	Airbus 360



erec sa sezamom



FROM Clause

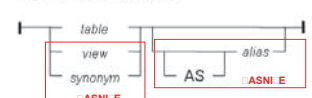
1.1. SELECT Options



1.4. FROM Clause



1.9. Table Reference



- klasična sintaksa (classical, comma-separated)
- ANSI sintaksa za spajanje relacija

FROM Clause (1)

1.4. FROM Clause



1.9. Table Reference



- klasična sintaksa (*classical*, *classic*, *alias*, *ite*) može se koristiti za obavljanje operacija:
 - Kartezijev produkt
 - spajanje uz uvjet i spajanje s izjednačavanjem
 - prirodno spajanje
- uvjeti spajanja se navode u `WHERE` dijelu `SELECT` naredbe, zajedno s eventualnim uvjetima selekcije (uvjeti spajanja i uvjeti selekcije se u tom slučaju povezuju logičkim operatorom `AND`)

FROM Clause (2)

- Zadane su relacije: $r(A, B)$ $s(C, D)$ $t(E, F)$

$r \times s$

```
SELECT r.A, s.C
FROM r, s
```

$r \bowtie s$
 $A=C \wedge B \geq D$

```
SELECT r.A, s.C
FROM r, s
WHERE r.A = s.C
AND r.B >= s.D
```

$r \bowtie s$
 $B=C$

```
SELECT r.A, s.C
FROM r, s
WHERE r.B = s.C
```

$\sigma_{B \geq 5}(r \bowtie s)$

```
SELECT r.A, s.C
FROM r, s
WHERE r.B = s.C
AND s.D >= 5
```

FROM Clause (3)

- $r(A, B)$ $s(C, D)$ $t(E, F)$

$(r \times s) \bowtie t$

```
SELECT r.A, s.C, t.E
FROM r, s, t
WHERE r.D = s.D AND t.D
```

$(r \bowtie s) \bowtie t$

```
SELECT r.A, s.C, t.E
FROM r, s, t
WHERE r.D = s.D AND t.D
```

$\sigma_{C=100}(s \bowtie t)$

```
SELECT s.C, t.E
FROM s, t
WHERE s.D = t.D
AND s.C = 100
```

FROM Clause (4)

1.4. FROM Clause



1.9. Table Reference



- ANSI sintaksa za spajanje relacija. može se koristiti za obavljanje operacija:
 - Kartezijev produkt
 - spajanje uz uvjet i spajanje s izjednačavanjem
 - prirodno spajanje
 - vanjsko spajanje
 - vanjsko spajanje uz uvjet
 - vanjsko spajanje s izjednačavanjem
 - prirodno vanjsko spajanje

FROM Clause (5)

1.10. ANSI Joined Tables



1.11. Join Options



```
r JOIN s
r INNER JOIN s ON uvjet(a,a)
r LEFT JOIN s ON uvjet(a,a)
r RIGHT JOIN s ON uvjet(a,a)
r FULL JOIN s ON uvjet(a,a)
```

FROM Clause (6)

- Rezervirane riječi `LEFT` i `INNER` se smiju izostaviti:

$r \text{ INNER JOIN } s \equiv r \text{ JOIN } s$

$r \text{ LEFT JOIN } s \equiv r \text{ LEFT JOIN } s$

$r \text{ RIGHT JOIN } s \equiv r \text{ RIGHT JOIN } s$

$r \text{ FULL JOIN } s \equiv r \text{ FULL JOIN } s$

FROM Clause ()

- Zadane su relacije: $r(A, B)$ $s(C, D)$ $t(E, F)$

$r \times s$

```

ELE r, s
r
s

```

$r \bowtie s$
 $A=C \wedge B \geq D$

```

ELE r, s
r
s
NNE J s
N A AND B D

```

$r \bowtie s$
 $B=C$

```

ELE r, s
r
s
NNE J s
N B C

```

$\sigma_{B=5}(r \bowtie s)$

```

ELE r, s
r
s
NNE J s
N B C
EE D 5

```

FROM Clause ()

- $r(A, B)$ $s(C, D)$ $t(E, F)$ $p(G, H)$

$(r \times s) \bowtie t$

```

ELE r, s, t
r
s
NNE J t
N s.D t.D

```

$(s \bowtie t) \bowtie p$

```

ELE s, t, p
s
t
NNE J t
N s.D t.D
NNE J p
N t.E p.E

```

$\sigma_{C=100}(s \bowtie t)$

```

ELE s, t
s
NNE J t
N s.D t.D
EE C 100

```

FROM Clause ()

- $r(A, B)$ $s(C, D)$ $t(E, F)$ $p(G, H)$

$(r \times s) \bowtie t$

```

ELE r, s, t
r
s
NNE J s
LE U J t
N s.D t.D
N s.D t.D

```

$(s \bowtie t) \bowtie p$

```

ELE s, t, p
s
t
NNE J t
N s.D t.D
NNE J p
N t.E p.E

```

$\sigma_{C=100}(s \bowtie t)$

```

ELE s, t
s
LE U J t
N s.D t.D
EE C 100

```

FROM Clause ()

- koliko se obavlja operacija spajanja i selekcija, uvjete spajanja treba navesti u \bowtie dijelu, a uvjete selekcije treba navesti u σ HERE dijelu σ naredbe
- iako, u slučaju kada se ne koristi vanjsko spajanje, rezultat upita ne ovisi o tome je li uvjet selekcije naveden u \bowtie ili σ HERE dijelu naredbe

student	matBr	prez	pbr
	101	Kolar	10000
	102	Horvat	21000

mjesto	pbr	naz
	10000	Zagreb
	21000	plit

```

ELE student
NNE J mesto
N br t br
EE rez 'olar'

```

```

ELE student
NNE J mesto
N br t br
AND rez 'olar'

```

- u ovom slučaju, oba upita daju isti rezultat

matBr	prez	pbr	pbr	naz
101	Kolar	10000	10000	Zagreb

FROM Clause ()

- Ako se koristi vanjsko spajanje, navođenje uvjeta selekcije u \bowtie dijelu umjesto σ HERE dijelu može **biti** utjecati na rezultat

student	matBr	prez	pbr
	101	Kolar	10000
	102	Horvat	21000

mjesto	pbr	naz
	10000	Zagreb
	21000	plit

```

ELE student
LE U J mesto
N br t br
EE rez 'olar'

```

- nek nakon obavljenog spajanja prema uvjetu navedenom u \bowtie dijelu naredbe, obavlja se selekcija n-torki prema uvjetu navedenom u σ HERE dijelu naredbe

matBr	prez	pbr	pbr	naz
101	Kolar	10000	10000	Zagreb

FROM Clause ()

- vdje je prikazan upit sličan prethodnom, ali u kojem je uvjet selekcije napisan na pogrešnom mjestu

student	matBr	prez	pbr
	101	Kolar	10000
	102	Horvat	21000

mjesto	pbr	naz
	10000	Zagreb
	21000	plit

```

ELE student
LE U J mesto
N br t br
AND rez 'olar'

```

- vdje će se pojaviti sve n-torke iz relacije student - uz one n-torke relacije student koje ne zadovoljavaju uvjet spajanja (uočite koji je uvjet spajanja ovdje naveden) dodat će se N-torki vrijednosti

matBr	prez	pbr	pbr	naz
101	Kolar	10000	10000	Zagreb
102	Horvat	21000	N	N

FROM Clause ()

- Ložički promatrano kada se u upitu spajaju više od dvije relacije, redoslijed spajanja je s lijeva na desno: spajaju se prve dvije relacije, zatim se dobiveni rezultat spaja s trećom navedenom relacijom, zatim se dobiveni rezultat spaja s četvrtom navedenom relacijom, itd.

() konačni rezultat će sigurno odgovarati rezultatu koji bi se dobio kada bi se relacije spajale s lijeva na desno. Fizički promatrano, upit će se možda izvesti drugačijim redoslijedom, ali o tome brine dio B-a koji se naziva optimizator upita

FROM Clause ()

- ako se ne koristi vanjsko spajanje, redoslijed spajanja je ionako irelevantan, jer vrijedi:

$$(r_1 \bowtie r_2) \bowtie r_3 \equiv r_1 \bowtie (r_2 \bowtie r_3)$$

- ako se koristi vanjsko spajanje, redoslijed spajanja jest važan jer:

$$(r_1 \Join r_2) \bowtie r_3 \neq r_1 \Join (r_2 \bowtie r_3)$$

FROM Clause ()

stud			mjesto			zupanija	
mbr	prez	pbr_t	pbr	naz_jesto	sifZup_j	sifZup	nazZup
100	Horvat	42000	42000	araždin	7	7	araždinska
102	Novak	21000	21000	plit	N	4	Istarska

(stud \bowtie mjesto) \bowtie zupanija
pbr_t=pbr sifZup_j=sifZup

```

ELE stud, mjesto, zupanija
    stud
    LE UESTO JON esto
    N br_t br
    NNE JON zupanija
    N sif_u sif_u
    
```

- prvo se spajaju relacije stud i mjesto, a zatim se dobiveni rezultat spaja s relacijom zupanija

mbr	prez	pbr_t	pbr	naz_jesto	sifZup_j	sifZup	nazZup
101	Horvat	42000	42000	araždin	7	7	araždinska

FROM Clause ()

stud \bowtie (mjesto \bowtie zupanija)
pbr_t=pbr sifZup_j=sifZup

da bismo izraz relacijske algebre mogli napisati u obliku naredbe, napisat ćemo ga u drugačijem obliku

\equiv (mjesto \bowtie zupanija) \bowtie stud
sifZup_j=sifZup pbr_t=pbr

```

ELE stud, mjesto, zupanija
    mjesto
    NNE JON zupanija
    N sif_u sif_u
    UESTO JON stud
    N br_t br
    
```

- prvo se spajaju relacije mjesto i zupanija, a zatim se dobiveni rezultat spaja s relacijom stud

mbr	prez	pbr_t	pbr	naz_jesto	sifZup_j	sifZup	nazZup
101	Horvat	42000	42000	araždin	7	7	araždinska
102	Novak	21000	N	N	N	N	N

Preimenovanje relacija unutar upita

- relacija se unutar upita može preimenovati u alias ime
 - alias ime je vidljivo samo unutar upita (ne utječe na stvarno ime relacije u bazi podataka)

1.9. Table Reference



- rezervirana riječ AS se smije ispustiti
- na relaciju koja je u upitu dobila alias ime, moguće je referencirati se isključivo preko tog istog alias imena

```

ELE naz_esto, naz_zupani_a
    esto A to_n
    , zu_zupani_a A z2
    E to_n.sif_u_zupani_a count.sif_u_zupani_a
    
```

```

ELE naz_esto, naz_zupani_a
    esto A to_n
    JON zu_zupani_a A z2
    N to_n.sif_u_zupani_a count.sif_u_zupani_a
    
```

Preimenovanje relacija unutar upita

- iako se preimenovanjem relacija može skratiti duljina teksta upita, u praksi se to ne preporuča jer upiti postaju manje razumljivi

```

ELE o_b, rezije, br, naz_esto
    osoba A o
    , esto A
    , zaoslen A z1
    , zu_zupani_a A z2
    E o_b z1_b
    AND o_br br
    AND sif_u z2.sif_u
    AND z2.naz_u 'araždinska'
    AND z1.radno_esto 'Dinačar'
    
```

- preimenovanje relacija unutar upita treba se koristiti onda kada se ista relacija pojavljuje u više uloga unutar istog upita

Paralelno spajanje

student	mbr	prez	pbrRod	pbrTan
100	Kolar	10000	21000	
102	Novak	21000	10000	
103	Ban	10000	10000	

mjesto	pbr	naz_jesto
	10000	Zagreb
	21000	plit

- Kako dobiti sljedeći rezultat:

mbr	prez	pbrRod	pbrTan	naz_jestoR
100	Kolar	10000	21000	Zagreb
102	Novak	21000	10000	plit
103	Ban	10000	10000	Zagreb

- ko je lako:

```

ELE mbr, prez
, pbrRod, pbrTan, naz_jesto A naz_jesto
student
, pbrRod, pbrTan, naz_jesto
E student.pbrRod mjesto.pbr
AND student.pbrTan mjesto.pbr
    
```

Paralelno spajanje

student	mbr	prez	pbrRod	pbrTan
100	Kolar	10000	21000	
102	Novak	21000	10000	
103	Ban	10000	10000	

mjesto	pbr	naz_jesto
	10000	Zagreb
	21000	plit

- Kako dobiti sljedeći rezultat:

mbr	prez	pbrRod	naz_jestoR	pbrTan	naz_jesto
100	Kolar	10000	Zagreb	21000	plit
102	Novak	21000	plit	10000	Zagreb
103	Ban	10000	Zagreb	10000	Zagreb

Paralelno spajanje

student	mbr	prez	pbrRod	pbrTan
100	Kolar	10000	21000	
102	Novak	21000	10000	
103	Ban	10000	10000	

mjesto	pbr	naz_jesto
	10000	Zagreb
	21000	plit

```

ELE mbr, prez
, pbrRod, pbrTan, naz_jesto A naz_jesto
, pbrTan, pbrRod, naz_jesto A naz_jesto
student
, pbrRod, pbrTan, naz_jesto
E student.pbrRod mjesto.pbr
AND student.pbrTan mjesto.pbr
    
```

NEIZRAVN RJEŠENJE

mbr	prez	pbrRod	naz_jestoR	pbrTan	naz_jesto
103	Ban	10000	Zagreb	10000	Zagreb

- plit nije dobar jer jednu n-torku iz relacije student pokušavamo spojiti s jednom n-torkom iz relacije mjesto uz sljedeći uvjet spajanja: vrijednost atributa pbrRod, te istovremeno i vrijednost atributa pbrTan iz relacije student su jednake vrijednosti atributa pbr iz relacije mjesto

Paralelno spajanje

- Kad bismo načinili dvije kopije relacije mjesto: mjestoR i mjesto, sa shemama i sadržajem jednakim relaciji mjesto:

student	mbr	prez	pbrRod	pbrTan
100	Kolar	10000	21000	
102	Novak	21000	10000	
103	Ban	10000	10000	

mjestoR	pbr	naz_jesto
	10000	Zagreb
	21000	plit

mjesto	pbr	naz_jesto
	10000	Zagreb
	21000	plit

```

ELE mbr, prez
, pbrRod, pbrTan, naz_jestoR A naz_jestoR
, pbrTan, pbrRod, naz_jesto A naz_jesto
student, mjestoR, mjesto
E student.pbrRod mjestoR.pbr
AND student.pbrTan mjesto.pbr
    
```

Paralelno spajanje

- Hoćemo li uvijek kada treba postaviti upit takvog tipa prvo napraviti kopije relacija

NEĆEMO

Paralelno spajanje

- Ispravno rješenje:

mjesto	pbr	naz_jesto
	10000	Zagreb
	21000	plit

student	mbr	prez	pbrRod	pbrTan
100	Kolar	10000	21000	
102	Novak	21000	10000	
103	Ban	10000	10000	

mjestoR	pbr	naz_jesto
	10000	Zagreb
	21000	plit

mjesto	pbr	naz_jesto
	10000	Zagreb
	21000	plit

```

ELE mbr, prez
, pbrRod, pbrTan, naz_jestoR A naz_jestoR
, pbrTan, pbrRod, naz_jesto A naz_jesto
student, mjestoR, mjesto
E student.pbrRod mjestoR.pbr
AND student.pbrTan mjesto.pbr
    
```

- u upitu se ista relacija pojavljuje u dvije različite uloge

Refleksivno spajanje

- Jedinice n-torke iz relacije povezane su s drugim n-torkama iz iste relacije

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	NULL
2	Odjel A	1
3	Odjel B	1
4	Pododjel X	2
5	Pododjel Y	2
6	Pododjel Z	3

- Uprava nema nadređenu org. jedinicu
- Odjelu A neposredno nadređena jedinica je Uprava
- Odjelu B neposredno nadređena jedinica je Uprava
- Pododjelu X neposredno nadređena jedinica je Odjel A
- itd.



Refleksivno spajanje

- Kako dobiti sljedeći rezultat

sifOrgjed	nazOrgjed	sifNadorgjed	nazNadorgjed
1	Uprava	NULL	NULL
2	Odjel A	1	Uprava
3	Odjel B	1	Uprava
4	Pododjel X	2	Odjel A
5	Pododjel Y	2	Odjel A
6	Pododjel Z	3	Odjel B

- radi se o spajanju relacije same sa sobom
- problem je sličan i slično se rješava kao u slučaju paralelnog spajanja
- relacija orgjed treba se u upitu pojaviti dva puta, jednom u ulozi organizacijske jedinice, a jednom u ulozi njezine nadređene organizacijske jedinice

Refleksivno spajanje

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	NULL
2	Odjel A	1
3	Odjel B	1
4	Pododjel X	2
5	Pododjel Y	2
6	Pododjel Z	3

sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	NULL
2	Odjel A	1
3	Odjel B	1
4	Pododjel X	2
5	Pododjel Y	2
6	Pododjel Z	3

```

SELECT orgjed.sifOrgjed
, orgjed.nazOrgjed
, orgjed.sifNadorgjed
, nadorgjed.nazOrgjed AS nazNadorgjed
FROM orgjed, orgjed AS nadOrgjed
WHERE orgjed.sifNadorgjed = nadOrgjed.sifOrgjed;
  
```

Refleksivno spajanje

```

SELECT orgjed.sifOrgjed
, orgjed.nazOrgjed
, orgjed.sifNadorgjed
, nadorgjed.nazOrgjed AS nazNadorgjed
FROM orgjed, orgjed AS nadOrgjed
WHERE orgjed.sifNadorgjed = nadOrgjed.sifOrgjed;
  
```

sifOrgjed	nazOrgjed	sifNadorgjed	nazNadorgjed
2	Odjel A	1	Uprava
3	Odjel B	1	Uprava
4	Pododjel X	2	Odjel A
5	Pododjel Y	2	Odjel A
6	Pododjel Z	3	Odjel B

- Nema organizacijske jedinice Uprava? Kako to popraviti?

Refleksivno spajanje

```

SELECT orgjed.sifOrgjed
, orgjed.nazOrgjed
, orgjed.sifNadorgjed
, nadorgjed.nazOrgjed AS nazNadorgjed
FROM orgjed
LEFT OUTER JOIN orgjed AS nadOrgjed
ON orgjed.sifNadorgjed = nadOrgjed.sifOrgjed;
  
```

sifOrgjed	nazOrgjed	sifNadorgjed	nazNadorgjed
1	Uprava	NULL	NULL
2	Odjel A	1	Uprava
3	Odjel B	1	Uprava
4	Pododjel X	2	Odjel A
5	Pododjel Y	2	Odjel A
6	Pododjel Z	3	Odjel B

Refleksivno spajanje

- Kako dobiti sljedeći rezultat
 - uz svaku organizacijsku jedinicu ispisati nazive neposredno podređenih organizacijskih jedinica
 - ako org. jedinica ima više od jedne podređene org. jedinice, u popisu se pojavljuje više puta
 - u popisu se moraju naći i one organizacijske jedinice koje nemaju niti jednu podređenu organizacijsku jedinicu

sifOrgjed	nazOrgjed	nazPodorgjed
1	Uprava	Odjel A
1	Uprava	Odjel B
2	Odjel A	Pododjel X
2	Odjel A	Pododjel Y
3	Odjel B	Pododjel Z
4	Pododjel X	NULL
5	Pododjel Y	NULL
6	Pododjel Z	NULL

Refleksivno spajanje

orgjed			orgjed (u ulozu podređene org.jedinice)		
sifOrgjed	nazOrgjed	sifNadorgjed	sifOrgjed	nazOrgjed	sifNadorgjed
1	Uprava	NULL	1	Uprava	NULL
2	Odjel A	1	2	Odjel A	1
3	Odjel B	1	3	Odjel B	1
4	Pododjel X	2	4	Pododjel X	2
5	Pododjel Y	2	5	Pododjel Y	2
6	Pododjel Z	3	6	Pododjel Z	3

```
SELECT orgjed.sifOrgjed
, orgjed.nazOrgjed
, podOrgjed.nazOrgjed AS nazPodorgjed
FROM orgjed
LEFT OUTER JOIN orgjed AS podOrgjed
ON podOrgjed.sifNadorgjed = orgjed.sifOrgjed;
```

reimenovanje relacija i atributa

- još jedan primjer u kojem se koristi preimenovanje relacije
- ispisati podatke o svim osobama čija je plaća manja od plaće osobe sa šifrom 103

osoba	sifra	ime	prez	placa
	100	Ana	Novak	6000
	101	Ana	Kolar	5000
	102	Ivan	Kolar	3000
	103	Ana	Novak	5000
	104	Ivan	Ban	4000

```
SELECT osoba.a
FROM osoba
INNER JOIN osoba AS osoba2
ON osoba.a < osoba2.a
AND osoba2.sifra = 103;
```

sifra	ime	prez	placa
102	Ivan	Kolar	3000
104	Ivan	Ban	4000

GROUP BY Clause

1.1. SELECT Options



1.6. GROUP BY Clause



- U GROUP BY dijelu naredbe se navodi jedan ili više atributa relacija koje su navedene u FROM dijelu naredbe

GROUP BY Clause

ispit

matBr	nazPredmet	ocjena
100	Matematika	3
100	Programiranje	2
100	Fizika	5
101	Matematika	2
101	Programiranje	2
101	Fizika	3
102	Matematika	4

```
SELECT nazPredmet AS naziv
, Avg(ocjena) AS prosjek
FROM ispit
GROUP BY nazPredmet;
```

naziv	prosjek
Matematika	3
Programiranje	2
Fizika	4

- U GROUP BY nije dopušteno koristiti izraze ili zamjenska imena atributa (*display_label*)

```
SELECT nazPredmet AS naziv
, Avg(ocjena) AS prosjek
FROM ispit
GROUP BY naziv;
```

HAVING Clause

ispit	matBr	nazPredmet	ocjena
	100	Matematika	3
	100	Programiranje	2
	100	Fizika	5
	101	Matematika	2
	101	Programiranje	2
	101	Fizika	3
	102	Matematika	4

```
SELECT nazPredmet AS naziv
, Avg(ocjena) AS prosjek
FROM ispit
GROUP BY nazPredmet;
```

naziv	prosjek
Matematika	3
Programiranje	2
Fizika	4

- Kako u rezultatu prikazati samo one grupe koje zadovoljavaju neki uvjet, npr. kako u rezultatu prikazati samo one predmete za koje je prosjek ocjena veći od 2?

```
SELECT nazPredmet AS naziv
, Avg(ocjena) AS prosjek
FROM ispit
GROUP BY nazPredmet
HAVING Avg(ocjena) > 2;
```

naziv	prosjek
Matematika	3
Fizika	4

HAVING Clause

- U Condition koji se navodi u HAVING dijelu naredbe dopušteno je u izrazima izvan agregatnih funkcija koristiti samo one attribute koji su navedeni u GROUP BY dijelu naredbe

1.1. SELECT Options



1.7. HAVING Clause



```
SELECT nazPredmet AS naziv
, Avg(ocjena) AS prosjek
FROM ispit
GROUP BY nazPredmet
HAVING nazPredmet < 'Matematika';
```


HAVING Clause

- Primjer ispisati nazive predmeta i njihove prosječne ocjene, ali samo za one predmete u kojima je najveća ikad dobivena ocjena bila **manja ili jednaka 4**

ispit

matBr	nazPredmet	ocjena
100	Matematika	3
100	Programiranje	2
100	Fizika	5
101	Matematika	2
101	Programiranje	2
101	Fizika	3
102	Matematika	4

```
SELECT nazPredmet AS naziv,
       AVG(ocjena) AS prosjek
FROM ispit
GROUP BY nazPredmet
HAVING MAX(ocjena) <= 4;
```

naziv	prosjek
Matematika	3
Programiranje	2

HAVING Clause

- U rezultatu se pojavljuju one grupe za koje se navedeni uvjet (*Condition*) izračuna kao logička vrijednost *true*. U rezultatu se ne pojavljuju one grupe za koje se navedeni uvjet izračuna kao logička vrijednost *false* ili *unknown*

ispit

matBr	nazPredmet	ocjena
100	Matematika	3
100	Programiranje	2
100	Fizika	NULL
101	Matematika	2
101	Programiranje	2
101	Fizika	NULL
102	Matematika	4

```
SELECT nazPredmet AS naziv,
       AVG(ocjena) AS prosjek
FROM ispit
GROUP BY nazPredmet
HAVING AVG(ocjena) < 3;
```

naziv	prosjek
Matematika	3

ORDER BY Clause

- Koristi se za sortiranje rezultata upita
- Ispisati podatke o položajima ispita poredati ih prema ocjenama, tako da se bliže početku liste nalaze studenti s većim ocjenama. Studente koji imaju međusobno jednake ocjene poredati prema prezimenima, tako da se manja prezimena ispisuju prije većih prezimena (tj. po abecedi)

polozajiProg

matBr	prez	ocjena
100	Orvat	3
107	Novak	3
102	Orvat	5
101	Kolar	5
103	Kolar	2
104	Orvat	3

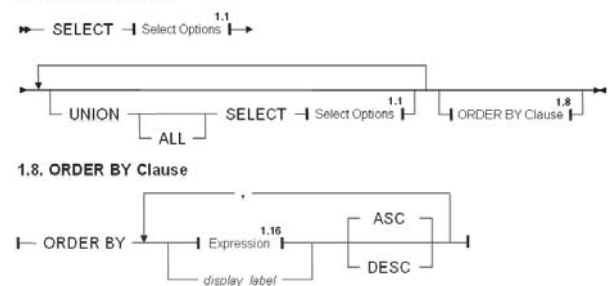
```
SELECT *
FROM polozajiProg
ORDER BY ocjena DESC,
       prez ASC;
```

matBr	prez	ocjena
102	Orvat	5
101	Kolar	5
104	Orvat	3
100	Orvat	3
107	Novak	3
103	Kolar	2

silazno (descending)
A uzlazno (ascending)

ORDER BY Clause

1. SELECT Statement



- Ako se smjer sortiranja ne navede, podrazumijeva se uzlazni (A) smjer sortiranja

ORDER BY Clause

- U ORDER BY dijelu naredbe mogu se koristiti i izrazi koji nisu navedeni u listi za selekciju
- ORDER BY dio naredbe je jedino mjesto u SQL naredbi u kojem je dopušteno referencirati se na zamjensko ime atributa (*display_label*)
- U jednoj SQL naredbi može se pojaviti samo jedan ORDER BY dio naredbe
 - ukoliko se u SQL naredbi koristi UNION, ORDER BY se nalazi iza posljednjeg SQL dijela naredbe
- SQL standard zahtijeva da se NULL vrijednosti pri sortiranju smatraju ili uvijek manjim ili uvijek većim od svih drugih vrijednosti
 - Informirani NULL vrijednosti pri sortiranju uvijek tretira kao da su manje od svih ostalih vrijednosti

ORDER BY Clause

bodoviLab	mbr	prez	bodLab	bodProg	bodoviProg	mbr	prez	bodLab	bodProg
	101	Novak	20	30		102	Kolar	12	NULL
	103	Orvat	NULL	20		104	Novak	30	0
	107	Ban	10	80					

```
SELECT mbr, bodLab, bodProg AS bodMbr
FROM bodoviLab
UNION
SELECT mbr, bodLab, bodProg AS bodMbr
FROM bodoviProg
ORDER BY bodMbr;
```

mbr	prez	bodLab	bodProg	ukupno
102	Kolar	12	NULL	NULL
103	Orvat	NULL	20	NULL
104	Novak	30	0	30
101	Novak	20	30	50
107	Ban	10	80	90

Redoslijed obavljanja dijelova SQL naredbe

1. FROM
2. WHERE
3. GROUP BY
4. HAVING
5. ORDER BY

6. UNION
7. ORDER BY



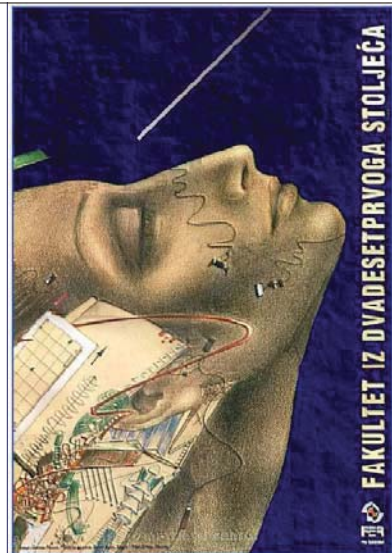
rezultat

- Logički pravilno, tj. konačni rezultat će sigurno odgovarati rezultatu koji bi se dobio kada bi se operacije obavljale navedenim redoslijedom. Fizički promatrano, upit će se moći izvršiti drugačijim redoslijedom.

Baze podataka

Predavanja
ožujak/travanj 2008.

SQL
2 dio



podupit

podupit Subqueries

vozilo	sifVoz	nosivost	teret	sifTeret	tezina
	101	2500		1001	1800
	102	2000		1002	1200
	103	800		1003	1000
	104	1000			

- Ispisati podatke o vozilima čija je nosivost veća od težine najtežeg tereta
- Pogrešan način:** prvo obaviti upit kojim se određuje težina najtežeg tereta

```
SELECT MAX(tezina) FROM teret;
```

- Zapamtiti dobiveni rezultat (1800), te napisati novi upit:

```
SELECT *
FROM vozilo
WHERE nosivost > 1800;
```

sifVoz	nosivost
101	2500
102	2000

podupit Subqueries

- Ispravan način:

```
SELECT *
FROM vozilo
WHERE nosivost > (SELECT MAX(tezina)
FROM teret);
```

podupit

1800

sifVoz	nosivost
101	2500
102	2000
103	800
104	1000

2500 > (SELECT MAX ...) → true
 2000 > (SELECT MAX ...) → true
 800 > (SELECT MAX ...) → false
 1000 > (SELECT MAX ...) → false

sifTeret	tezina
1001	1800
1002	1200
1003	1000

sifVoz	nosivost
101	2500
102	2000

- U navedenom primjeru je rezultat podupita jednak za svaku n-torku iz relacije vozilo, stoga je (fizički promatrano) rezultat podupita dovoljno izračunati samo jednom tijekom obavljanja upita

podupit Subqueries

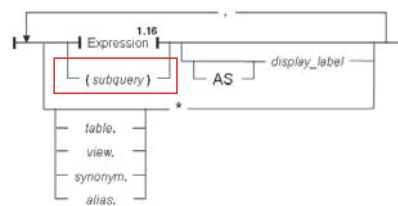
- podupit je upit koji je ugrađen u neki drugi upit
 - upit u kojeg je podupit ugrađen naziva se vanjski upit (*outer query*)
 - osim izraza **podupit** (*subquery*), u literaturi se također koristi i izraz **unijeđeni podupit** (*nested query*)
- podupit se u vanjski upit može ugraditi
 - u uvjet (*Condition*) u WHERE dijelu vanjskog upita
 - u uvjet (*Condition*) u HAVING dijelu vanjskog upita
 - u listu za selekciju (*SELECT List*) vanjskog upita
- podupit može sadržavati sve do sada spomenute dijelove SELECT naredbe osim ORDER BY dijela naredbe
- u vanjski upit se može ugraditi više podupita, u svaki od podupita se može ugraditi više podupita, itd.

Skalarni podupit *Scalar subquery*

- za početak, razmatrat će se najjednostavniji oblik podupita: podupit čiji je rezultat **jedna jedinstvena** vrijednost (skalarni podupit)
- npr. podatak tipa: cijeli broj, niz znakova, datum, itd.
- može se reći: rezultat skalarnog podupita je "relacija" stupnja jedan i kardinalnosti jedan
- vrijednost atributa n-torke dotične "relacije" se u vanjskom upitu koristi kao skalarna vrijednost

Podupit u listi selekcija

1.3 SELECT List



vozilo	sifVoz	nosivost
	101	2500
	102	2000
	103	800
	104	1000

teret	sifTeret	tezina
	1001	1800
	1002	1200
	1003	1000

- Ispisati podatke o svim vozilima. Uz svako vozilo ispisati podatak o najvećoj težini tereta

```
SELECT
    , (SELECT MAX(tezina)
      FROM teret AS t
      FROM vozilo;
```

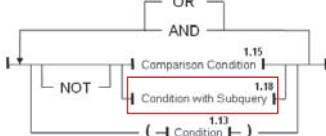
sifVoz	nosivost	maxTezina
101	2500	1800
102	2000	1800
103	800	1800
104	1000	1800

Podupit u dijelu naredbe

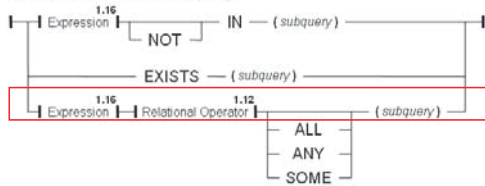
1.5. WHERE Clause



1.13. Condition



1.18. Condition with Subquery



- Ako se u WHERE ili HAVING dijelu naredbe koristi ovdje označeni oblik uvjeta, dopušteno je koristiti **isključivo skalarnu podupitu**

Podupit u dijelu naredbe

- Ispisati podatke o studentima koji stanuju u mjestu Ludbreg

stud	mbr	prez	pbrSt
	100	Horvat	42230
	101	Kolar	21000
	102	Novak	42230

mjesto	pbr	nazMjesto
	42000	Varaždin
	42230	Ludbreg
	21000	Split

```
SELECT * FROM stud
WHERE pbrSt = (SELECT pbr FROM mjesto
               WHERE nazMjesto = 'Ludbreg');
```

- Često se problem može riješiti bez podupita. U konkretnom slučaju, **olije** rješenje glasi:

```
SELECT stud.*
FROM stud
JOIN mjesto
ON stud.pbrSt = mjesto.pbr
WHERE nazMjesto = 'Ludbreg';
```

Podupit u dijelu naredbe

- Ispisati podatke o studentima koji su rođeni u mjestu Ludbreg, a stanuju u mjestu Varaždin

stud	mbr	prez	pbrRod	pbrSt
	100	Horvat	42230	42230
	101	Kolar	21000	42000
	102	Novak	42230	42000

mjesto	pbr	nazMjesto
	42000	Varaždin
	42230	Ludbreg
	21000	Split

```
SELECT * FROM stud
WHERE pbrRod = (SELECT pbr FROM mjesto
                WHERE nazMjesto = 'Ludbreg')
AND pbrSt = (SELECT pbr FROM mjesto
              WHERE nazMjesto = 'Varaždin');
```

mbr	prez	pbrRod	pbrSt
102	Novak	42230	42000

- Postoji **olije** rješenje u kojem se koriste operacije spajanja i selekcije. Riješiti za vježbu!

Podupit u dijelu naredbe

stud	mbr	prez	pbrSt
	100	Horvat	42230
	101	Kolar	21000
	102	Novak	42230

mjesto	pbr	nazMjesto
	42000	Varaždin
	42230	Ludbreg
	21000	Split

- Ukoliko podupit čiji bi rezultat trebao biti skalar vrati više od jedne n-torke ili više nego jedan atribut, sustav će dojaviti pogrešku

```
SELECT * FROM stud
WHERE pbrSt = (SELECT pbr FROM mjesto
               WHERE nazMjesto LIKE 'r');
```

- Ukoliko podupit čiji bi rezultat trebao biti skalar ne vrati niti jednu n-torku, dobivena skalarna vrijednost će biti NULL vrijednost

```
SELECT * FROM stud
WHERE pbrSt = (SELECT pbr FROM mjesto
               WHERE nazMjesto = 'Grad Srebrenica');
```

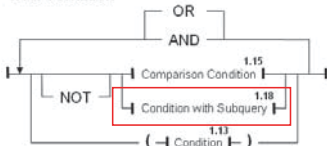
Podupit i A i N dijel naredbe

1.7. HAVING Clause

HAVING — 1.13 Condition —

- Podupiti u HAVING dijelu naredbe koriste se na jednak način kao u WHERE dijelu naredbe

1.13. Condition



Podupit i A i N dijel naredbe

Primjer:

dvorana	oznDv	kapacitet
D1		150
D2		200
A201		80

raspored	predmet	oznGr	brojSt
	Matematika	M1	200
	Matematika	M2	50
	Matematika	M3	50
	Fizika	F1	250
	Fizika	F2	150
	Elektronika	E1	50
	Elektronika	E2	50
	Elektronika	E3	100
	Elektronika	E4	150

- Ispisati nazive predmeta za koje se u "D dvoranama" predavanja mogu održati istovremeno za sve grupe. Uz svaki takav predmet ispisati ukupni broj studenata na predmetu

```
SELECT pred_e
, SUM(brojSt) AS ukupnoStud
FROM raspored
WHERE pred_e =
(SELECT SUM(brojSt)
FROM dvorana
WHERE oznDv LIKE 'D');
```

predmet	ukupnoStud
Matematika	300
Elektronika	350

350

Korelirani podupit i Correlated subquery

- Ako se u podupitu koriste atributi iz vanjskog upita, za podupit i vanjski upit se kaže da su korelirani (*correlated*)
- Za podupit koji je koreliran s vanjskim upitom koristi se naziv korelirani podupit (*correlated subquery*)
- Najčešće se korelirani podupit mora (fizički) izvršiti po jedanput za svaku n-torku iz vanjskog upita
- Sljedeći podupit nije korelirani podupit: u podupitu se ne koriste atributi vanjskog upita. Rezultat podupita ne ovisi o vrijednostima n-torki iz vanjskog podupita, stoga se taj podupit (fizički) treba izvršiti samo jednom tijekom jednog obavljanja vanjskog upita

```
SELECT *
FROM ozi_o
WHERE nosi_o = (SELECT MA_ezina
FROM ere);
```

1800

Korelirani podupit i Correlated subquery

- ispisati podatke o strojevima koji su ukupno korišteni više od dopuštenog broja radnih sati

stroj	oznStr	dopBrSati
S1		1000
S2		1500
S3		500

radStroja	oznStr	godina	brSatiRada
	S1	2002	700
	S1	2003	100
	S1	2004	300
	S2	2002	700
	S2	2003	500
	S3	2005	600

```
SELECT oznStr, dopBrSati
FROM stroj
WHERE dopBrSati >
(SELECT SUM(brSatiRada)
FROM radStroja
WHERE oznStr = stroj.oznStr);
```

oznStr	dopBrSati
S1	1000
S3	500

- korelirani podupit: rezultat podupita ovisi o vrijednostima atributa vanjskog upita - za svaku n-torku vanjskog upita dobiva se drugačiji rezultat podupita

Korelirani podupit i Correlated subquery

```
SELECT oznStr, dopBrSati
FROM stroj
WHERE dopBrSati >
(SELECT SUM(brSatiRada)
FROM radStroja
WHERE radStroja.oznStr = stroj.oznStr);
```

oznStr	dopBrSati
S1	1000
S3	500

- upit se (logički promatrano) obavlja na sljedeći način:
 - vanjski upit uzima jednu n-torku iz relacije stroj. Na temelju sadržaja te n-torke i sadržaja relacije radStroja, u podupitu se izračunava suma sati rada dotičnog stroja. Ukoliko je uvjet usporedbe zadovoljen, testirana n-torka se pojavljuje u rezultatu
 - postupak se ponavlja za svaku n-torku relacije stroj

oznStr	dopBrSati
S1	1000
S2	1500
S3	500

1000 < (SELECT SUM ... WHERE oznStr = 'S1') → true
 1500 < (SELECT SUM ... WHERE oznStr = 'S2') → false
 500 < (SELECT SUM ... WHERE oznStr = 'S3') → true

Korelirani podupit i Correlated subquery

- Primjer: korelirani podupit u listi za selekciju
- uz svaki stroj koji je korišten više od dopuštenog broja sati, ispisati broj sati korištenja stroja

```
SELECT oznStr
, dopBrSati
, (SELECT SUM(brSatiRada)
FROM radStroja
WHERE radStroja.oznStr = stroj.oznStr) AS koristenSati
FROM stroj
WHERE dopBrSati >
(SELECT SUM(brSatiRada)
FROM radStroja
WHERE radStroja.oznStr = stroj.oznStr);
```

oznStr	dopBrSati	koristenSati
S1	1000	1100
S3	500	600

Korelirani podupiti *Correlated subquery*

- Rješenje istog problema bez korištenja podupita:

```
SELECT s.oznStr, doBrSati, SUM(rSatiRada) AS korisTenSati
FROM s
JOIN radSroja
ON s.oznStr = radSroja.oznStr
GROUP BY s.oznStr, doBrSati
HAVING doBrSati < SUM(rSatiRada);
```

oznStr	dopBrSati	koristenSati
S1	1000	1100
S3	500	600

Korelirani podupiti *Correlated subquery*

- Primjer: korelirani podupit u HAVING dijelu naredbe

ispit

mbr	predmet	akGod	ocj
100	Matematika	2005	2
101	Matematika	2005	3
102	Matematika	2005	4
100	Fizika	2005	2
101	Fizika	2005	5
100	Elektronika	2005	3
101	Elektronika	2005	3
110	Matematika	2006	3
111	Matematika	2006	5
110	Fizika	2006	3
111	Fizika	2006	3
112	Fizika	2006	3
113	Fizika	2006	2
111	Elektronika	2006	5
112	Elektronika	2006	4

- ispisati predmete čija je prosječna ocjena za 2006. godinu veća od prosječne ocjene tog istog predmeta za 2005. godinu

```
SELECT predmet
FROM ispit AS is
WHERE avg(ocj) > (
    SELECT avg(ocj)
    FROM ispit
    WHERE predmet = is.predmet
    AND is.akGod = 2005);
```

predmet
Matematika
Elektronika

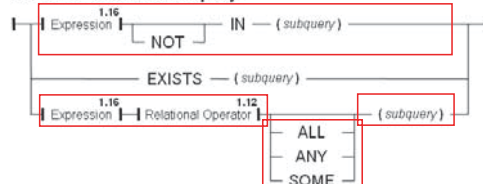
Korištenje atributa vanjskog upita u podupitu

- u podupitu se mogu koristiti atributi iz vanjskog upita (obratno ne vrijedi)
- ukoliko se imena atributa (relacija) vanjskog upita podudaraju s imenima atributa (relacija) podupita:
 - ime atributa (relacije) navedeno u podupitu se odnosi na ime atributa (relacije) iz podupita
 - ime atributa (relacije) navedeno u vanjskom upitu se odnosi na ime atributa (relacije) vanjskog upita
- ukoliko je potrebno razriješiti dvosmislenost (npr. ista relacija se koristi u FROM dijelu vanjskog upita i FROM dijelu podupita, a u podupitu se koriste atributi relacije iz vanjskog upita), dovoljno je preimenovati relaciju u vanjskom upitu ili u podupitu
 - prethodni primjer ilustrira takav slučaj

Jednostupčani podupiti *Single-column subquery*

- Rezultat jednostupčanog podupita je relacija stupnja jedan, s (moguće) više n-torki
 - također je dopušteno da jednostupčani podupit vrati jednu ili niti jednu n-torku
- jednostupčani podupiti se koriste u WHERE dijelu ili HAVING dijelu vanjskog upita
- jednostupčani podupiti se ne koriste u listi za selekciju

1.18. Condition with Subquery



Jednostupčani podupiti *Single-column subquery*

- Ispisati podatke o studentima čije je prezime različito od svih prezimena nastavnika

stud	mbr	ime	prez
100	Ivan	Horvat	
101	Ana	Kolar	
102	Marko	Novak	

nastavnik	jmbg	prez
12345	Kolar	
23456	Ban	
34567	Pernar	

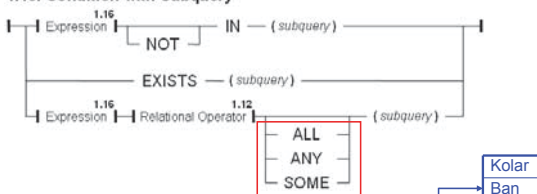
```
SELECT *
FROM stud
WHERE prez <> (SELECT prez
               FROM nastavnici);
```

Kolar
Ban
Pernar

- Ovakvo napisan podupit **nije ispravan** - sustav će dojaviti pogrešku jer pomoću relacijskog operatora <> pokušavamo usporediti skalarnu vrijednost i rezultat podupita koji sadrži tri vrijednosti
- Potrebno je koristiti oblike usporedbe s IN, ALL, ANY, SOME

Jednostupčani podupiti *Single-column subquery*

1.18. Condition with Subquery



```
SELECT *
FROM stud
WHERE prez <> ALL (SELECT prez
                   FROM nastavnici);
```

Kolar
Ban
Pernar

mbr	ime	prez
100	Ivan	Horvat
102	Marko	Novak

- uvjet selekcije će biti zadovoljen za one n-torke iz relacije stud čija je vrijednost atributa prez različita od vrijednosti svih članova (multi)skupa dobivenog obavljanjem podupita

jednosupčani podpi Single-column subquery

- izraz { \square | $\square =$ | $=$ | $\square \square$ | \square | $\square =$ } **ALL** (podupit)
 - true ako je izraz { \square | $\square =$ | $=$ | $\square \square$ | \square | $\square =$ } od svih vrijednosti dobivenih podupitom
- izraz { \square | $\square =$ | $=$ | $\square \square$ | \square | $\square =$ } **OD** (podupit)
 - true ako je izraz { \square | $\square =$ | $=$ | $\square \square$ | \square | $\square =$ } od barem jedne vrijednosti dobivene podupitom
- AN** je sinonim za **OD**

jednosupčani podpi Single-column subquery

- Ispisati podatke o dvoranama čiji je kapacitet veći od broja studenata u barem jednoj od grupa

dvorana	oznDv	kapacitet
	D1	150
	D2	120
	A201	80

grupa	oznGr	brojSt
	M1	100
	F1	170
	E4	150

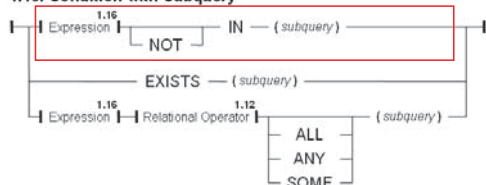
```
SELECT  $\square$ 
FROM d $\square$ orana
WHERE  $\square$ all $\square$ i $\square$ e  $\square$  SOME  $\square$ SELECT  $\square$ rojS $\square$ 
FROM gr $\square$ a $\square$ ;
--> 100
--> 170
--> 150
```

oznDv	kapacitet
D1	150
D2	120

- za vježbu riješiti bez podupita (spajanje, selekcija, projekcija)

jednosupčani podpi Single-column subquery

1.18. Condition with Subquery



- izraz **IN** (podupit)
 - true ako se u (multi)skupu vrijednosti dobivenih podupitom nalazi barem jedan element jednak vrijednosti izraza
 - ekvivalentno sa: izraz = **SOME** (podupit)
- izraz **NO** **IN** (podupit)
 - true ako se u (multi)skupu vrijednosti dobivenih podupitom ne nalazi niti jedan element jednak vrijednosti izraza
 - ekvivalentno sa: izraz \neq **ALL** (podupit)

jednosupčani podpi Single-column subquery

- Ispisati podatke o studentima koji su bilo koji predmet položili tijekom akademske godine 2005.

stud	mbr	prez
	100	Horvat
	101	Kolar
	102	Novak
	103	Ban

ispit	mbr	predmet	akGod	ocjena
	100	Matematika	2006	1
	100	Elektronika	2005	2
	100	Fizika	2005	3
	102	Elektronika	2005	2
	102	Fizika	2006	5
	103	Elektronika	2005	NULL
	103	Matematika	2007	4

```
SELECT  $\square$ 
FROM s $\square$ ud
WHERE  $\square$ r IN
     $\square$ SELECT  $\square$ r
    FROM is $\square$ i
    WHERE a $\square$ od =  $\square$ 
    AN $\square$  o $\square$ jena  $\square$ ;
```

mbr	prez
100	Horvat
102	Novak

- za vježbu riješiti bez podupita (spajanje, selekcija, projekcija)

NULL vrijednos i jednosupčani podpi

- izraz **relOp ALL** (podupit)
 - ako je podupitom dobiven skup vrijednosti { x_1, x_2, \dots, x_n }, efektivno se uvjet izračunava na sljedeći način:
 - izraz **relOp** x_1 AND izraz **relOp** x_2 AND ... AND izraz **relOp** x_n
- izraz **relOp OD** (podupit)
 - ako je podupitom dobiven skup vrijednosti { x_1, x_2, \dots, x_n }, efektivno se uvjet izračunava na sljedeći način:
 - izraz **relOp** x_1 OR izraz **relOp** x_2 OR ... OR izraz **relOp** x_n
- izraz **IN** (podupit)
 - ekvivalentno sa: izraz = **OD** (podupit)
- izraz **NO** **IN** (podupit)
 - ekvivalentno sa: izraz \neq **ALL** (podupit)

rijeri podpi i NULL vrijednos

- naročitu pažnju pri korištenju podupita čiji rezultat može sadržavati NULL vrijednosti treba obratiti na uvjete selekcije oblika:

```
WHERE expression relationalOperator ALL  $\square$ subquery
```

```
WHERE expression NOT IN  $\square$ subquery
```

ukoliko se u rezultatu ovakvih podupita nalazi makar jedna NULL vrijednost, rezultat izračunavanja uvjeta selekcije nikad neće biti true

uplata	rbrUpI	iznosUpI	isplata	rbrIspl	iznosIspl
	1	50		1	80
	2	NULL		2	100
	3	200		3	NULL
	4	120		4	150

80
100
NULL
150

rbrUpl	iznosUpl
1	50
2	NULL
3	200
4	120

50 > 80 OR 50 > 100 OR 50 > NULL OR 50 > 150 → *unknown*
 NULL > 80 OR NULL > 100 OR NULL > NULL OR NULL > 150 → *unknown*
 200 > 80 OR 200 > 100 OR 200 > NULL OR 200 > 150 → *true*
 120 > 80 OR 120 > 100 OR 120 > NULL OR 120 > 150 → *true*

rbrUpl	iznosUpl
3	200
4	120

uplata	rbrUpI	iznosUpI	isplata	rbrIspl	iznosIspl
	1	50		1	80
	2	NULL		2	100
	3	200		3	NULL
	4	120		4	150

80
100
NULL
150

rbrUpl	iznosUpl
1	50
2	NULL
3	200
4	120

50 > 80 AND 50 > 100 AND 50 > NULL AND 50 > 150 → false
 NULL > 80 AND NULL > 100 AND NULL > NULL AND NULL > 150 → unknown
 200 > 80 AND 200 > 100 AND 200 > NULL AND 200 > 150 → unknown
 120 > 80 AND 120 > 100 AND 120 > NULL AND 120 > 150 → false

rbrUpl	iznosUpl
--------	----------

uplata	rbrUpI	iznosUpI	isplata	rbrIspl	iznosIspl
	1	50		1	80
	2	NULL		2	100
	3	100		3	NULL
	4	80		4	150

80
100
NULL
150

rbrUpl	iznosUpl
1	50
2	NULL
3	100
4	80

50 = 80 OR 50 = 100 OR 50 = NULL OR 50 = 150 → *unknown*
 NULL = 80 OR NULL = 100 OR NULL = NULL OR NULL = 150 → *unknown*
 100 = 80 OR 100 = 100 OR 100 = NULL OR 100 = 150 → *true*
 80 = 80 OR 80 = 100 OR 80 = NULL OR 80 = 150 → *true*

rbrUpl	iznosUpl
3	100
4	80

uplata	rbrUpI	iznosUpI	isplata	rbrIspl	iznosIspl
	1	50		1	80
	2	NULL		2	100
	3	100		3	NULL
	4	80		4	150

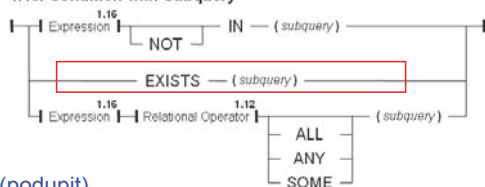
80
100
NULL
150

rbrUpl	iznosUpl
1	50
2	NULL
3	100
4	80

50 \neq 80 AND 50 \neq 100 AND 50 \neq NULL AND 50 \neq 150 \rightarrow unknown
 NULL \neq 80 AND NULL \neq 100 AND NULL \neq NULL AND NULL \neq 150 \rightarrow unknown
 100 \neq 80 AND 100 \neq 100 AND 100 \neq NULL AND 100 \neq 150 \rightarrow false
 80 \neq 80 AND 80 \neq 100 AND 80 \neq NULL AND 80 \neq 150 \rightarrow false

rbrUpl	iznosUpl
--------	----------

1.18. Condition with Subquery



- **□□□□□ (podupit)**
 - *true* ako rezultat podupita sadrži barem jednu n-torku (bilo kakvu). Pri tome nije važno koliko u dobivenoj n-torci ili n-torkama ima atributa (podupit ne mora biti jednostupčan) niti koje su vrijednosti njihovih atributa
- **NO□□□□□ (podupit)**
 - *true* ako rezultat podupita ne sadrži niti jednu n-torku
- na rezultat vanjskog upita ne utječe eventualna pojava NULL vrijednosti u rezultatu podupita

- Ispisati podatke o studentima koji u akademskoj godini u kojoj su upisali studij nisu položili niti jedan ispit

stud	mbr	prez	akGodUpis
	100	Horvat	2005
	101	Kolar	2005
	102	Novak	2006
	103	Ban	2005

ispit	mbr	predmet	akGod	ocjena
	100	Matematika	2005	1
	100	Fizika	2006	2
	100	Elektronika	2007	4
	100	Matematika	2006	3
	101	Matematika	2005	2
	101	Fizika	2005	5
	102	Matematika	2006	4

ovdje se npr. moglo napisati samo mbr: dobio bi se jednak rezultat

mbr	prez	akGodUpis
100	Horvat	2005
103	Ban	2005

Modpi i AN dijel naredbe

- Svi prikazani oblici podupita mogu se također koristiti i u HAVING dijelu naredbe
- Primjer: ispisati naziv(e) predmeta s najvećim prosjekom

```
SELECT red_e
FROM ispi
GROUP BY red_e
HAVING AVG(ocjena) = ALL
(SELECT AVG(ocjena)
FROM ispi
GROUP BY red_e);
```

ispit		
mbr	predmet	ocjena
100	Matematika	4
101	Matematika	5
102	Matematika	3
100	Fizika	3
101	Fizika	4
101	Elektronika	5
102	Elektronika	3

4.0
3.5
4.0

predmet
Matematika
Elektronika

Neporeno korićenje podpi

- Ispisati podatke o upisima predmeta svih studenata koji stanuju u Zaboku

- Loše rješenje:

```
SELECT * FROM isanred_e
WHERE jtag IN
(SELECT jtag FROM sden
WHERE rSan IN
(SELECT r FROM jes_o
WHERE nazMjes_o = a_o);
```

- Bolje rješenje:

```
SELECT isanred_e.
FROM isanred_e, sden, jes_o
WHERE isanred_e.jtag = sden.jtag
AND sden.rSan = jes_o.r
AND nazMjes_o = a_o;
```

Neporeno korićenje podpi

- Ispisati podatke o studentima i nazivima mjesta u kojima stanuju. U rezultatu trebaju biti i studenti čije je mjesto stanovanja nepoznato
- Priložno rješenje:

```
SELECT sden.
, (SELECT nazMjes_o FROM jes_o
WHERE r = rSan) AS nazMjes_o
FROM sden;
```

- Spravno rješenje:

```
SELECT sden., nazMjes_o
FROM sden
LEFT OUTER JOIN jes_o
ON rSan = r;
```

Priložno LOKALNO

resjek

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

studenata koji su položili i Matematiku i Programiranje

polozioMatem ∩ polozioProgr

```
SELECT *
FROM polozioMatem
WHERE EXISTS
(SELECT * FROM polozioProgr
WHERE polozioProgr.r = polozioMatem.r
AND polozioProgr.ime = polozioMatem.ime
AND polozioProgr.prez = polozioMatem.prez);
```

- Ispisuju se one n-torke relacije vanjskog upita za koje podupit spije pronaći barem jednu n-torku koja ima jednake vrijednosti atributa mbr, ime i prez kao n-torka iz vanjskog upita

mbr	ime	prez
102	Ana	Novak
107	Jura	Horvat

resjek po oc spajanja

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	Ana	Novak
103	Tea	Ban
107	Jura	Horvat

polozioProgr

mbr	ime	prez
102	Ana	Novak
105	Rudi	Kolar
107	Jura	Horvat

polozioMatem ∩ polozioProgr

- Isti rezultat se može dobiti na sljedeći način:

```
SELECT polozioMatem.
FROM polozioMatem, polozioProgr
WHERE polozioMatem.r = polozioProgr.r
AND polozioMatem.ime = polozioProgr.ime
AND polozioMatem.prez = polozioProgr.prez;
```

mbr	ime	prez
102	Ana	Novak
107	Jura	Horvat

resjek prisv NULL vrijednos

polozioMatem

mbr	ime	prez
100	Ivan	Kolar
102	NULL	Novak
103	Tea	Ban
107	NULL	NULL

polozioProgr

mbr	ime	prez
102	NULL	Novak
105	Rudi	Kolar
107	NULL	NULL

polozioMatem ∩ polozioProgr

- Paziti: što je kopija n-torke?

mbr	ime	prez
102	NULL	Novak
107	NULL	NULL

```
SELECT *
FROM polozioMatem
WHERE EXISTS
(SELECT * FROM polozioProgr
WHERE
(polozioProgr.r = polozioMatem.r OR
polozioProgr.r IS NULL AND polozioMatem.r IS NULL)
AND
(polozioProgr.ime = polozioMatem.ime OR
polozioProgr.ime IS NULL AND polozioMatem.ime IS NULL)
AND
(polozioProgr.prez = polozioMatem.prez OR
polozioProgr.prez IS NULL AND polozioMatem.prez IS NULL);
```

resjek prisv NULL vrijednos

polozioMatem			polozioProgr		
mbr	ime	prez	mbr	ime	prez
100	Ivan	Kolar	102	NULL	Novak
102	NULL	Novak	105	Rudi	Kolar
103	Tea	Ban	107	NULL	NULL
107	NULL	NULL			

- Isti rezultat može se dobiti pomoću spajanja

```
SELECT oozioMatem.
FROM oozioMatem, oozioProgr
WHERE
  oozioMatem.mbr = oozioProgr.mbr OR
  oozioMatem.mbr IS NULL AND oozioProgr.mbr IS NULL
  AND
  oozioMatem.ime = oozioProgr.ime OR
  oozioMatem.ime IS NULL AND oozioProgr.ime IS NULL
  AND
  oozioMatem.prez = oozioProgr.prez OR
  oozioMatem.prez IS NULL AND oozioProgr.prez IS NULL;
```

Razlika

polozioMatem			polozioProgr		
mbr	ime	prez	mbr	ime	prez
100	Ivan	Kolar	102	Ana	Novak
102	Ana	Novak	105	Rudi	Kolar
103	Tea	Ban	107	Jura	Horvat
107	Jura	Horvat			

studenti koji su položili Matematiku, ali nisu položili Programiranje

polozioMatem polozioProgr

```
SELECT
  FROM oozioMatem
  WHERE NOT EXISTS
    (SELECT FROM oozioProgr
     WHERE oozioProgr.mbr = oozioMatem.mbr
       AND oozioProgr.ime = oozioMatem.ime
       AND oozioProgr.prez = oozioMatem.prez);
```

- Ispisuju se one n-torke relacije vanjskog upita za koje podupit ne spije pronaći niti jednu n-torku koja ima jednake vrijednosti atributa mbr, ime i prez kao n-torka iz vanjskog upita

mbr	ime	prez
100	Ivan	Kolar
103	Tea	Ban

Razlika prisv NULL vrijednos

polozioMatem			polozioProgr		
mbr	ime	prez	mbr	ime	prez
100	Ivan	Kolar	102	NULL	Novak
102	NULL	Novak	105	Rudi	Kolar
103	Tea	Ban	107	NULL	NULL
107	NULL	NULL			

polozioMatem polozioProgr

- Paziti: što je kopija n-torke?

mbr	ime	prez
100	Ivan	Kolar
103	Tea	Ban

```
SELECT
  FROM oozioMatem
  WHERE NOT EXISTS
    (SELECT FROM oozioProgr
     WHERE
       oozioProgr.mbr = oozioMatem.mbr OR
       oozioProgr.mbr IS NULL AND oozioMatem.mbr IS NULL
       AND
       oozioProgr.ime = oozioMatem.ime OR
       oozioProgr.ime IS NULL AND oozioMatem.ime IS NULL
       AND
       oozioProgr.prez = oozioMatem.prez OR
       oozioProgr.prez IS NULL AND oozioMatem.prez IS NULL);
```

SQL naredbe za izmjenu sadržaja relacije

INSERT
UPDATE
DELETE

INSERT

- INSERT naredba se koristi za unos jedne n-torke (1) ili skupa n-torki (2) u relaciju table

2. INSERT Statement

INSERT INTO table (column) VALUES Clause

2.1. VALUES Clause

VALUES (constant)

bez navedene liste atributa

uz navedenu listu atributa

INSERT navedene liste atributa

- U relaciju se upisuje jedna n-torka pri čemu vrijednosti svi atributa n-torke oraj navedene redosljedo kojim su atributi navedeni u CREATE TABLE naredbi kojom je relacija definirana

```
CREATE TABLE stud
  mbr INTEGER
  ime NCHAR
  prez NCHAR
  stipend CHAR
  pbrStan INTEGER
  ;
```

stud				
mbr	ime	prez	stipend	pbrStan

Znači: ako se prilikom unosa nove n-torke ne navede vrijednost za atribut stipend, sustav će kao vrijednost tog atributa postaviti vrijednost N

```
INSERT INTO stud VALUES
  , Ivan
  , Horvatic
  , N
  , NULL
  ;
```

stud				
mbr	ime	prez	stipend	pbrStan
100	Ivan	Horvat	N	NULL

INSERT 1. navedene liste atributa

- Opisani oblik INSERT naredbe ima neke nedostatke:
 - u slučaju kada se u relaciju upisuje n-torka čiji relativno veliki broj atributa treba postaviti na NULL vrijednost ili pretpostavljenu vrijednost (*default value*)
 - ipak se moraju navesti vrijednosti **svi** atributa
 - u slučaju kada se relacijska shema promijeni (npr. promijeni se "redosljed" atributa)
 - budući da vrijednosti atributa moraju biti navedene redosljedom atributa u CREATE TABLE naredbi, INSERT naredbe koje su napisane prije promjene relacijske sheme više neće biti ispravne
- zbog navedenih nedostataka, preporuča se korištenje oblika INSERT naredbe s navedenom listom atributa

INSERT 1. naveden listom atributa

- U **prvom dijelu naredbe** navode se imena atributa (i njihov redosljed) čije će vrijednosti (u odgovarajućem redosljedu) biti navedene u drugom dijelu naredbe
 - atributi čije vrijednosti nisu navedene u INSERT naredbi, postavljaju se na pretpostavljenu (*default*) vrijednost (ukoliko je takva definirana u CREATE TABLE naredbi) ili na NULL vrijednost

```
INSERT INTO stud (mbr, ime, prez, stipend, pbrStan)
VALUES (100, 'Ivan', 'Horvat', N, NULL);
```

mbr	ime	prez	stipend	pbrStan
100	Ivan	Horvat	N	NULL
101	NULL	Kolar	N	10000

ime? stipend?

INSERT 2.

2. INSERT Statement



- u relaciju *table* upisuju se n-torke dobivene dijelom INSERT naredbe koji je sličan SELECT naredbi - u sintaksnom dijagramu taj je dio naredbe označen sa **SELECT Statement (subset)**
 - SELECT Statement (subset)** može sadržavati sve prethodno opisane dijelove SELECT naredbe **osim**
 - ORDER BY**
 - FIRST n**
 - UNION**

INSERT 2. navedene liste atributa

- u relaciju *polozioFiz* upisati podatke o studentima koji su položili predmet Fizika

mbr	ime	prez	pbrSt
102	Ana	Novak	10000
105	Rudi	Kolar	21000
107	Jura	Horvat	41000
109	Tea	Ban	51000

mbr	predmet	ocjena
102	Elektronika	1
102	Matematika	3
105	Fizika	3
105	Matematika	4
107	Fizika	1
109	Fizika	5

mbr	imeSt	prezSt
105	Rudi	Kolar
109	Tea	Ban

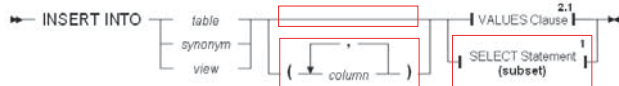
```
INSERT INTO polozioFiz
SELECT stud.mbr, ime, prez
FROM stud, ispit
WHERE stud.mbr = ispit.mbr
AND ispit.predmet = 'Fizika'
AND ispit.ocjena > 0;
```

imena atributa u SELECT listi ne moraju odgovarati imenima atributa u relaciji polozioFiz

mbr	imeSt	prezSt
105	Rudi	Kolar
109	Tea	Ban

INSERT 2.

2. INSERT Statement



- jednako kao kod oblika INSERT naredbe za unos jedne n-torke u relaciju
 - bez navedene liste imena atributa, broj i redosljed atributa u n-torkama dobivenim obavljanjem SELECT dijela naredbe mora odgovarati broju i redosljedu atributa u CREATE TABLE naredbi (prikazano u prethodnom primjeru)
 - uz navedenu listu imena atributa, atributi čije vrijednosti nisu navedene u INSERT naredbi, postavljaju se na pretpostavljenu (*default*) vrijednost (ukoliko je takva definirana u CREATE TABLE naredbi) ili na NULL vrijednost

INSERT 2. naveden listom atributa

- u relaciju *prosjek* upisati podatke o prosječnim ocjenama pojedinih predmeta. Za vrijednost atributa *akGodina* postaviti NULL vrijednost

```
INSERT INTO prosjek (predmet, prosOcJ, akGod)
SELECT ispit.predmet, ispit.ocjena, NULL
FROM ispit
WHERE ispit.predmet = 'Fizika';
```

mbr	predmet	ocjena
102	Elektronika	1
102	Matematika	3
105	Fizika	3
105	Matematika	4
107	Fizika	1
109	Fizika	5

predmet	prosOcJ	akGod
Fizika	3.0	NULL

4. DELETE Statement



- DELETE naredba briše one n-torke relacije *table* za koje se uvjet naveden u WHERE dijelu naredbe izračuna kao *true*
 - ako se WHERE dio naredbe ne navede, iz relacije *table* se brišu *sve* n-torke
- u WHERE dijelu naredbe mogu se koristiti svi oblici uvjeta (*Condition*) koji se koriste u WHERE dijelu SELECT naredbe, ali u FROM dijelu podupita nije dopušteno koristiti relaciju *table*

```
DELETE FROM jesio
WHERE r IN (SELECT r FROM jesio WHERE nazMjesio LIKE ' ');
```

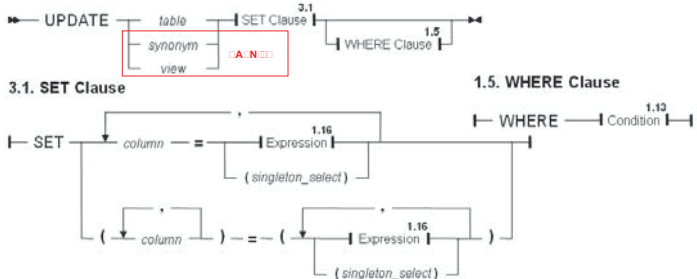
- iz relacije *polozioFiz* obrisati n-torke onih studenata koji (sudeći prema podacima iz relacije *ispit*) nisu položili predmet Fizika

stud				ispit			polozioFiz		
mbr	ime	prez	pbrSt	mbr	predmet	ocjena	mbr	imeSt	prezSt
102	Ana	Novak	10000	102	Elektronika	1	102	Ana	Novak
105	Rudi	Kolar	21000	102	Matematika	3	105	Rudi	Kolar
107	Jura	Horvat	41000	105	Fizika	3	107	Jura	Horvat
109	Tea	Ban	51000	105	Matematika	4	109	Tea	Ban
				107	Fizika	1	111	Ivan	Polak
				109	Fizika	5			

```
DELETE FROM polozioFiz
WHERE r NOT IN
(SELECT r
FROM ispit
WHERE predmet = 'Fizika'
AND ocjena > 0);
```

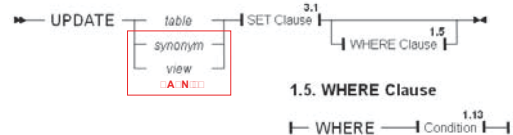
mbr	imeSt	prezSt
105	Rudi	Kolar
109	Tea	Ban

3. UPDATE Statement



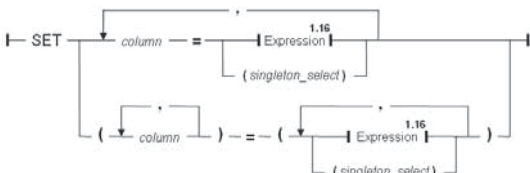
- UPDATE naredba mijenja vrijednosti atributa postojećih n-torki relacije *table*. U WHERE dijelu naredbe opisuje se koje n-torke će biti promijenjene; u SET dijelu naredbe opisuje se koji će atributi biti postavljeni na koje vrijednosti

3. UPDATE Statement



- UPDATE naredba mijenja vrijednosti atributa onih n-torki relacije *table* za koje se uvjet naveden u WHERE dijelu naredbe izračuna kao *true*
 - ako se WHERE dio naredbe ne navede, mijenjaju se vrijednosti atributa u svim n-torkama relacije *table*
- u WHERE dijelu naredbe mogu se koristiti svi oblici uvjeta (*Condition*) koji se koriste u WHERE dijelu SELECT naredbe, ali u FROM dijelu podupita nije dopušteno koristiti relaciju *table*

3.1. SET Clause



- SET dio naredbe određuje nove vrijednosti atributa. Nova vrijednost atributa može biti definirana kao:
 - Expression* - konstante, NULL, atributi iz relacije *table*, binarni i unarni operatori, funkcije, uvjetni izraz, zagrade
 - singleton_select* - jednako kao skalarni podupit (korelirani ili nekorelirani)
 - relaciju *table* nije dopušteno koristiti u FROM dijelu

- Bodove na međuispitu uvećati za 10 bodova onim studentima koji time ne bi stekli ukupno (bodLab+bodMI) više od 100 bodova

bodovi	mbr	prez	bodLab	bodMI
	101	Novak	30	55
	102	Polak	NULL	20
	103	Kolar	20	10
	104	Ban	10	80
	105	Horvat	50	49
	106	Seljan	10	NULL

```
UPDATE bodovi
SET bodMI = bodMI + 10
WHERE bodLab < 10 AND bodMI < 10;
```

bodovi	mbr	prez	bodLab	bodMI
	101	Novak	30	65
	102	Polak	NULL	20
	103	Kolar	20	20
	104	Ban	10	90
	105	Horvat	50	49
	106	Seljan	10	NULL

UPDATE

- Bodove na međuispitu uvećati za 2 boda studentima koji time ne bi stekli više od 50 bodova za međuispit, bodove za laboratorij povećati za 3 boda onim studentima koji time ne bi stekli ukupno više od 40 bodova za laboratorij

bodovi			
mbr	prez	bodLab	bodMI
101	Novak	30	55
102	Polak	NULL	20
103	Kolar	20	10
104	Ban	10	80
105	Horvat	50	49
106	Seljan	10	NULL

bodovi			
mbr	prez	bodLab	bodMI
101	Novak	33	55
102	Polak	NULL	22
103	Kolar	23	12
104	Ban	13	80
105	Horvat	50	49
106	Seljan	13	NULL

```

UPDATE odozi SET
  odMI = CASE
    WHEN odMI <= 50 THEN odMI + 2
    ELSE odMI
  END,
  odLa = CASE
    WHEN odLa <= 40 THEN odLa + 3
    ELSE odLa
  END;

```

UPDATE

- U sintaksnom dijagramu za *SET Clause* može se vidjeti da postoje dva slična, jednako vrijedna oblika:
 - SET atribut1=vrijednost1, atribut2=vrijednost2, ...
 - SET (atribut1, atribut2, ...)=(vrijednost1, vrijednost2, ...)
- Prethodna UPDATE naredba se mogla napisati na sljedeći način:

```

UPDATE odozi SET
  odMI, odLa =
  CASE
    WHEN odMI <= 50 THEN odMI + 2
    ELSE odMI
  END,
  CASE
    WHEN odLa <= 40 THEN odLa + 3
    ELSE odLa
  END;

```

UPDATE

- U relaciji mjesto zamijeniti stare poštanske brojeve i nazive (samo onim mjestima za koje postoje opisani novi poštanski brojevi i nazivi)

mjesto		konverzija		
pbr	nazMjesto	stariPbr	noviPbr	noviNaziv
41000	ZAGREB	51400	52000	Pazin
51400	PAZIN	52000	52100	Pula
52000	PULA	54000	31000	Osijek
54000	OSIJEK			

mjesto	
pbr	nazMjesto
41000	ZAGREB
52000	Pazin
52100	Pula
31000	Osijek

```

UPDATE mjeso
SET pbr, nazMjeso =
  (SELECT noviPbr,
   nazMjesto
   FROM konverzija
   WHERE konverzija.stariPbr = mjeso.pbr),
  (SELECT noviNaziv,
   nazMjesto
   FROM konverzija
   WHERE konverzija.stariPbr = mjeso.pbr)
WHERE pbr IN (SELECT stariPbr
              FROM konverzija);

```

UPDATE

- Nastavnicima koji su na ispitima iz BP podijelili (prosječno) najveće ocjene, povećati plaću za 20000

nast			ispitBP		
sifNast	prez	placa	mbrSt	ocjena	sifNast
101	Novak	52000	1001	5	101
102	Kolar	55000	1002	4	101
103	Horvat	48000	1003	3	101
104	Ban	57000	1004	2	102
			1005	4	102
			1006	5	103
			1007	5	103
			1008	2	103
			1009	3	104

nast		
sifNast	prez	placa
101	Novak	72000
102	Kolar	55000
103	Horvat	68000
104	Ban	57000

```

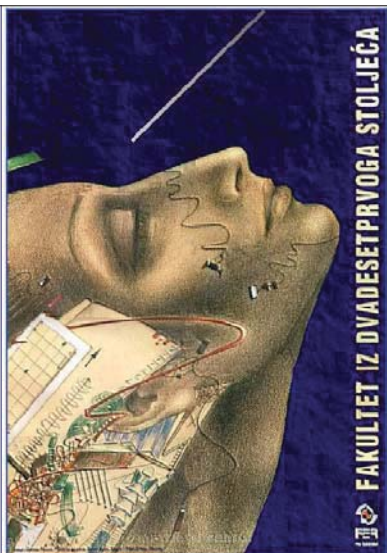
UPDATE nast SET placa = placa + 20000
WHERE sifNast IN
  (SELECT sifNast
   FROM ispitBP
   GROUP BY sifNast
   HAVING AVG(ocjena) >= ALL
    (SELECT AVG(ocjena)
     FROM ispitBP
     GROUP BY sifNast));

```

Baze podataka

Predavanja
travanj 2008.

Oblikovanje
sheme relacijske
baze podataka
1 dio



Oblikovanje sheme baze podataka

- cilj: oblikovati shemu baze podataka s dobrim svojstvima
- karakteristike loše koncipirane sheme baze podataka:
 - redundancija (čije su posljedice):
 - neracionalno korištenje prostora za pohranu
 - anomalija unosa
 - anomalija izmjene
 - anomalija brisanja
 - pojava lažnih n-torki

riječ lože konspirane s e e a e podataka

- Prodavaonice šalju svoje narudžbe proizvođaču:

on ilica 2 1 are Kraš Ravnice bb 10 000 Zagreb Narudžba br. 13/25 datum: 1.5.2006 Molimo isporučite nam 1200 komada proizvoda Napolitanke (šifra 129) i 2000 komada proizvoda Albert keks (šifra 139)	Diona-28 Bolska 7 21 000 Split Kraš Ravnice bb 10 000 Zagreb Narudžba br. 43-21 datum: 7.2.2006 Molimo isporučite nam 1200 komada proizvoda Napolitanke (šifra 129) i 1800 komada proizvoda Domaćica (šifra 221)	Konzum-7 Ilica 20 10 000 Zagreb Kraš Ravnice bb 10 000 Zagreb Narudžba br. 41/56 datum: 4.2.2007 Molimo isporučite nam 1100 komada proizvoda Napolitanke (šifra 129)
---	--	---

- proizvođač želi pohraniti podatke o narudžbama u svoju bazu podataka. Svi podaci se pohranjuju u relaciju **narudžbaArtikla**

narudžbaArtikla								
nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina

Neracionalno korištenje prostora za pohranu

- Sadržaj relacije nakon unosa podataka iz prispjelih narudžbi:

narudžbaArtikla								
nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- na više mjesta se ponavlja isti (redundantan) podatak:
 - Konzum-7 je prodavaonica u Zagrebu
 - adresa prodavaonice Konzum-7 je Ilica 20
 - naziv artikla sa šifrom 129 je Napolitanke
 - naziv mjesta s poštanskim brojem 10000 je Zagreb
 - datum narudžbe s brojem 13/25 je 1.5.2006
 - itd.

Anomalija unosa

narudžbaArtikla								
nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- ne mogu se unijeti podaci o artiklima koje nitko nije naručio
- ne mogu se unijeti podaci o prodavaonicama koje ništa nisu naručile
- ...
- svaki put kad se unosi novi podatak o narudžbi nekog artikla, mora se ponovno upisivati i naziv i mjesto i adresa prodavaonice koja taj artikl naručuje
 - pri tome treba paziti da se podaci za istu prodavaonicu uvijek jednako unesu da bi se zadržala konzistentnost podataka

Anomalija izmjena

- ako neka prodavaonica promijeni adresu, promjenu adrese potrebno je obaviti na više mjesta da bi se zadržala konzistentnost podataka

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- npr. prodavaonica Konzum-7 se preseli jedan kućni broj dalje od centra

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 22	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 22	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 22	41/56	4.2.2007	129	Napolitanke	1100

Anomalija brisanja

- brisanjem svih narudžbi za neki artikl gube se podaci o artiklu

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- npr. ako se obriše posljednja n-torka o narudžbama artikla Domaćica, podatke o tom artiklu više nećemo imati u bazi podataka

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

Pokušaj (neuspješni) popravka sheme baze podataka

- Podaci o narudžbama će se pohranjivati u dvije relacije

$\text{narudžba} = \pi_{\text{nazProd, pbr, nazMjesto, adresa, brNar, datNar, sifArtikl}}(\text{narudžbaArtikla})$

$\text{artikl} = \pi_{\text{sifArtikl, nazArtikl, kolicina}}(\text{narudžbaArtikla})$

narudžba							
nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	

artikl		
sifArtikl	nazArtikl	kolicina
129	Napolitanke	1200
139	Albert keks	2000
221	Domaćica	1800
129	Napolitanke	1100

- Ovakva shema baze podataka uzrokovat će pojavu lažnih (*spurious*) n-torki
- dolazi do gubitka informacije!

Pojava lažnih n-torki

- Obavljanjem operacije \bowtie artikl dobije se **više n-torki** nego ih je bilo u relaciji narudzbaArtikla (neke n-torke u rezultatu su **lažne** - označene su zvjezdicom)

$\text{narudzbaArtikla}_2 = \text{narudzba} \bowtie \text{artikl} \neq \text{narudzbaArtikla}$

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1100
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1200

- to bi se dogodilo ako se na temelju relacija narudzba i artikl pokuša izračunati ukupni broj naručenih proizvoda Napolitanke

```
SELECT SUM(kolicina)
FROM narudzba, artikl
WHERE narudzba.sifArtikl = artikl.sifArtikl
AND nazArtikl = 'Napolitanke';
```

Ispravna shema baze podataka

mjesto		prodavaonica		artikl	
pbr	nazMjesto	nazProd	pbr	sifArtikl	nazArtikl
10000	Zagreb	Konzum-7	10000	129	Napolitanke
21000	Split	Diona-28	21000	139	Albert keks
				221	Domaćica

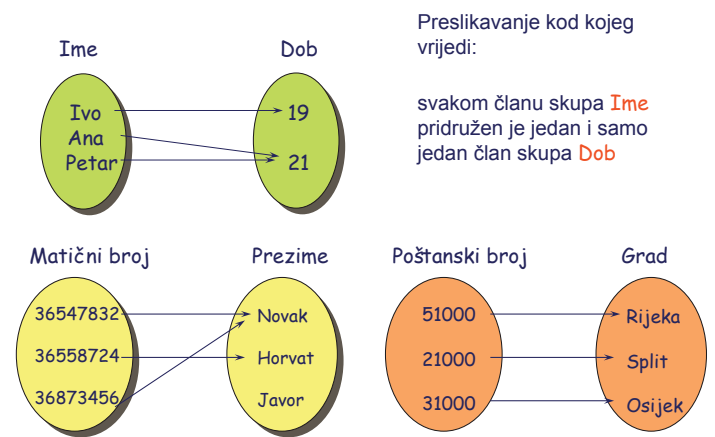
narudzba			stavkaNarudzbe		
brNar	nazProd	datNar	brNar	sifArtikl	kolicina
13/25	Konzum-7	1.5.2006	13/25	129	1200
43-21	Diona-28	7.2.2006	13/25	139	2000
41/56	Konzum-7	4.2.2007	43-21	129	1200
			43-21	221	1800
			41/56	129	1100

- Za vježbu provjerite
 - postoji li redundancija u ovoj bazi podataka
 - je li moguća pojava lažnih n-torki

Kako odrediti zamjenu za loše koncipiranu relacijsku shemu

- proučavanjem značenja podataka (semantike)
- proučavanjem zavisnosti među podacima
- uvođenjem ograničenja koja su ovisna o semantici podataka
- najvažnije su FUNKCIJSKE ZAVISNOSTI

Funkcija



Funkcijske zavisnosti - definicija

- Neka je r relacija sa shemom R i neka su A i B skupovi atributa, $A \subseteq B$

Funkcijska zavisnost $A \rightarrow B$ vrijedi na shemi R ukoliko

u svim dopuštenim stanjima relacije $r(R)$ svaki par n-torki t_1 i t_2 koje imaju jednake A -vrijednosti, također imaju jednake B -vrijednosti, odnosno:

$$t_1(A) = t_2(A) \Rightarrow t_1(B) = t_2(B)$$

Kratika za funkcijsku zavisnost je FZ.

Funkcijske zavisnosti - primjer

- relacija osoba(OSOBA)

osoba	matBr	prezime	ime	postBr	grad
	11234	Novak	Iosip	21000	Split
	12345	Horvat	Ivan	10000	Zagreb
	22211	Kolar	Ante	21000	Split
	33345	Ban	Ivan	31000	Osijek
	23456	Kolar	Ana	31000	Osijek

- Funkcijska zavisnost **postBr** \rightarrow **grad** vrijedi na shemi OSOBA jer svaki par n-torki koje imaju jednake vrijednosti atributa **postBr** također imaju jednake vrijednosti atributa **grad** (i to vrijedi ne samo za trenutno stanje relacije, nego za sva dopuštena stanja relacije)
- Vrijedi li funkcijska zavisnost prezime** \rightarrow **postBr**

funkcijske zavisnosti

- funkcijske zavisnosti proizlaze iz značenja podataka (semantike) a ne iz trenutnog stanja relacije

- Primjer: relacija osoba(OSOBA)

osoba	matBr	prezime	ime	pbr
	11234	Kolar	Ante	21000
	22211	Kolar	Ante	31000
	33345	Ban	omo	10000

- promatranjem samo trenutnog stanja relacije mogli bismo (pogrešno) zaključiti da vrijedi FZ prezime → ime
- međutim, poznavanjem značenja podataka u relaciji možemo zaključiti da je u gore prikazanu relaciju dopušteno unijeti n-torku <76555, Kolar, Zrinka, 51000>
- ⇒ FZ prezime → ime ne vrijedi na shemi OSOBA

Priroda funkcijskih zavisnosti

- Postojanje funkcijske zavisnosti ne može se dokazati na temelju postojećih podataka u relaciji.
- Analizom postojećih podataka u relaciji moguće je tek pretpostaviti da bi funkcijska zavisnost mogla vrijediti.
- Dokaz za postojanje FZ treba tražiti u značenju pojedinih atributa.

Priroda funkcijskih zavisnosti

$R = \{A, B, \dots\}$

$r(R)$

A	B	
a	α	1
b	γ	1
b	α	1
c	α	3
a	α	1

Vrijedi li FZ $AB \rightarrow \dots$ na shemi R?

- moguće je da vrijedi, ali to ne možemo sa sigurnošću tvrditi
- bez poznavanja značenja atributa A, B i α , ne možemo zaključiti koje funkcijske zavisnosti zaista vrijede na shemi R

$r(R)$

A	B	
a	α	1
b	γ	1
b	α	1
c	α	3
a	α	1

Vrijedi li FZ $B \rightarrow A$ na shemi R?

Sa sigurnošću možemo tvrditi: **NE**

Priroda funkcijskih zavisnosti

- Ako u relacijskoj shemi R vrijedi FZ $\dots \rightarrow \dots$, relacija $r(R)$ ne može sadržavati dvije n-torke koje imaju jednake \dots -vrijednosti i različite \dots -vrijednosti
- Primjer: ako u relacijskoj shemi $R = \{\text{matBr, prezime, grad, telefon}\}$ vrijedi FZ $\text{matBr} \rightarrow \text{prezime}$ tada relacija $r(R)$ ne smije sadržavati dvije n-torke s istim matičnim brojem i različitim prezimenom

Priroda funkcijskih zavisnosti - primjer

ispit	mbr	sifPred	datIsplit	sifNast	ocjena
	101	10	30.1.2006	1003	1
	101	10	15.1.2007	1002	4
	102	10	30.1.2006	1001	3
	102	11	15.1.2006	1002	5

- studenta α je na ispitu iz predmeta β na datum γ nastavnik δ ocijenio ocjenom ϵ
- vrijedi li FZ $\text{mbr sifNast} \rightarrow \text{ocjena}$
 - ne, jer bi to značilo da nastavnik δ studentu α uvijek mora dati istu ocjenu
- vrijedi li FZ $\text{mbr sifPred} \rightarrow \text{ocjena}$
 - ne, jer bi to značilo da student α iz predmeta β mora dobiti uvijek istu ocjenu
- vrijedi li FZ $\text{mbr datIsplit} \rightarrow \text{ocjena}$
 - ne, jer bi to značilo da student α na datum γ mora uvijek mora dobiti sve jednake ocjene
- vrijedi li FZ $\text{mbr sifPred sifNast} \rightarrow \text{ocjena}$ **NE (Zašto?)**
- vrijedi li FZ $\text{mbr sifPred datIsplit} \rightarrow \text{ocjena}$ **DA (Zašto?)**

funkcijske zavisnosti - S primjer

ispit	mbr	sifPred	datIsplit	sifNast	ocjena
	101	10	30.1.2006	1003	1
	101	10	15.1.2007	1002	4
	102	10	30.1.2006	1001	3
	102	11	15.1.2006	1002	5

- pomoću SELECT naredbe ispitati bi li u relaciji ispit eventualno mogla vrijediti FZ $\text{mbr sifNast} \rightarrow \text{ocjena datIsplit}$
- ispituju se svi parovi n-torki t_1, t_2 koje imaju jednake \dots -vrijednosti (u primjeru $\dots = \text{mbr, sifNast}$)
- ako postoji par n-torki t_1, t_2 koje imaju iste \dots -vrijednosti, a različite \dots -vrijednosti (u primjeru $\dots = \text{ocjena, datIsplit}$), tada FZ sigurno ne vrijedi

```
SELECT *
FROM ispit AS t1, ispit AS t2
WHERE t1.mbr = t2.mbr
  AND t1.sifNast = t2.sifNast
  AND (t1.ocjena <> t2.ocjena
       OR t1.datIsplit <> t2.datIsplit);
```

n-torke t_1, t_2 koje imaju jednake \dots -vrijednosti ... a različite \dots -vrijednosti

- ako takve n-torke ne postoje, onda FZ **možda** vrijedi

Armstrongovi aksiomi

- Projektant sheme baze podataka specificira FZ koje su mu semantički očite, no obično vrijede i brojne druge FZ koje mogu biti izvedene iz početnih FZ. Korištenjem Armstrongovih aksioma izvode se nove FZ.

A1 S1 N1 I AKSIOMI

Neka je R relacijska shema, neka su α, β, γ, Z skupovi atributa i neka vrijedi:
 $\alpha \subseteq R, \beta \subseteq R, \gamma \subseteq R$

A-1 KSIOM S

- Ako je $\alpha \subseteq \beta$, tada vrijedi $\alpha \rightarrow \beta$

A-2 ČANOM

- Ako u shemi R vrijedi $\alpha \rightarrow \beta$, tada vrijedi i $\alpha \rightarrow Z$

A-3 ANZIOM N S

- Ako u shemi R vrijedi $\alpha \rightarrow \beta$ i $\beta \rightarrow \gamma$, tada vrijedi i $\alpha \rightarrow \gamma$

Armstrongovi aksiomi

A-1 REFLEKSIIVNOST

- Ako je $\alpha \subseteq \beta$ tada vrijedi $\alpha \rightarrow \beta$
 - uvijek vrijedi $\alpha \rightarrow \alpha$

PRIMER: osoba(OSOBA)

matBr	prezime	ime	postBr	grad
11234	Novak	osip	21000	Split
12345	orvat	Ivan	10000	Zagreb
23456	Kolar	Ana	31000	Osijek
34567	Novak	osip	31000	Osijek

$\alpha \subseteq \beta$ prezime ime $\alpha \subseteq \beta$ prezime

$\alpha \subseteq \beta \Rightarrow$ u relaciji α vrijedi i FZ prezime ime \rightarrow prezime

$\alpha \subseteq \beta \Rightarrow$ u relaciji α vrijedi i FZ prezime ime \rightarrow prezime ime

Armstrongovi aksiomi

A-2 UVEĆANJE

- Ako u shemi α vrijedi $\alpha \rightarrow \beta$ tada vrijedi i $\alpha \rightarrow Z$
 - možemo uvećati lijevu stranu funkcijske zavisnosti

PRIMER: osoba(OSOBA)

matBr	prezime	ime	postBr	grad
11234	Novak	osip	21000	Split
12345	orvat	Ivan	10000	Zagreb
23456	Kolar	Ana	31000	Osijek
34567	Novak	osip	31000	Osijek

α relaciji α vrijedi FZ matBr \rightarrow ime

\Rightarrow u relaciji α vrijedi i FZ matBr prezime \rightarrow ime

\Rightarrow u relaciji α vrijedi i FZ matBr prezime grad \rightarrow ime

Armstrongovi aksiomi

A-3 TRANZITIVNOST

- Ako u shemi α vrijedi $\alpha \rightarrow \beta$ i $\beta \rightarrow \gamma$ tada vrijedi i $\alpha \rightarrow \gamma$
 - $\alpha \rightarrow \gamma$ je tranzitivna zavisnost

PRIMER: osoba(OSOBA)

matBr	prezime	ime	postBr	grad
11234	Novak	osip	21000	Split
12345	orvat	Ivan	10000	Zagreb
23456	Kolar	Ana	31000	Osijek
34567	Novak	osip	31000	Osijek

α relaciji α vrijede FZ matBr \rightarrow postBr i postBr \rightarrow grad

\Rightarrow u relaciji α vrijedi i FZ matBr \rightarrow grad

Pravila koja proizlaze iz Armstrongovih aksioma

Neka je R relacijska shema, neka su α, β, γ, Z skupovi atributa i neka vrijedi:
 $\alpha \subseteq R, \beta \subseteq R, \gamma \subseteq R, \alpha \subseteq \beta$

P-1 PAKIOM NIOM (pravilo o aditivnosti)

- Ako u shemi R vrijedi $\alpha \rightarrow \beta$ i $\alpha \rightarrow \gamma$, tada vrijedi i $\alpha \rightarrow \beta \gamma$

P-2 PAKIOM DKOM PZIOM (pravilo o projektivnosti)

- Ako u shemi R vrijedi $\alpha \rightarrow \beta \gamma$, tada vrijedi i $\alpha \rightarrow \beta$

P-3 PAKIOM PSKOM ANZIOM N S

- Ako u shemi R vrijedi $\alpha \rightarrow \beta$ i $\beta \rightarrow \gamma$, tada vrijedi i $\alpha \rightarrow \gamma$

Pravila koja proizlaze iz Armstrongovih aksioma

P-1 PRAVILO UNIJE (pravilo o aditivnosti)

- Ako u shemi α vrijedi $\alpha \rightarrow \beta$ i $\alpha \rightarrow \gamma$ tada vrijedi i $\alpha \rightarrow \beta \gamma$

PRIMER: osoba(OSOBA)

matBr	prezime	ime	postBr	grad
11234	Novak	osip	21000	Split
12345	orvat	Ivan	10000	Zagreb
23456	Kolar	Ana	31000	Osijek
34567	Novak	osip	31000	Osijek

α relaciji α vrijede FZ matBr \rightarrow ime i matBr \rightarrow prezime

\Rightarrow u relaciji α vrijedi i FZ matBr \rightarrow ime prezime

Pravila koja proizlaze iz Armstrongovih aksioma

P-2 PRAVILO DEKOMPOZICIJE (pravilo o projektivnosti)

- Ako u shemi \square vrijedi $\square \rightarrow \square Z$ tada vrijedi i $\square \rightarrow \square$

PRIMER: osoba(OSOBA)

matBr	prezime	ime	postBr	grad
11234	Novak	osip	21000	Split
12345	orvat	Ivan	10000	Zagreb
23456	Kolar	Ana	31000	Osijek
34567	Novak	osip	31000	Osijek

\square relaciji oso \square vrijedi FZ $\text{matBr} \rightarrow \text{ime prezime}$

\Rightarrow u relaciji oso \square vrijedi i FZ $\text{matBr} \rightarrow \text{ime}$

\Rightarrow u relaciji oso \square vrijedi i FZ $\text{matBr} \rightarrow \text{prezime}$

Pravila koja proizlaze iz Armstrongovih aksioma

P-3 PRAVILO PSEUDOTRANZITIVNOSTI

- Ako u shemi \square vrijedi $\square \rightarrow \square$ i $\square \rightarrow \square Z$ tada vrijedi i $\square \rightarrow \square Z$

PRIMER: zaposlenje(ZAPOSL: EN: E)

jmbg	strSprema	funkcija	zaposlOd	zaposlDo	placa
101	SS	direktor	1.1.2001	31.12.2002	10000
101	SS	tajnik	1.1.2003	31.12.2003	8000
102	SS	direktor	1.1.2004	31.12.2004	9000
102	SS	tajnik	1.1.2001	31.12.2002	7000
103	SS	direktor	1.1.2005	31.12.2005	10000
101	SS	direktor	1.1.2006	31.12.2006	10000

\square relaciji $\square \text{pos} \square \square$ vrijede FZ

$\text{jmbg} \rightarrow \text{strSprema}$ i $\text{funkcija strSprema} \rightarrow \text{placa}$

\Rightarrow u relaciji $\square \text{pos} \square \square$ vrijedi i FZ $\text{jmbg} \rightarrow \text{placa}$

Primjer korištenja aksioma i pravila

\square z pretpostavku da na relacijskoj shemi $R = \square A, B, \square, D, E, F$ \square vrijedi skup funkcijskih zavisnosti $F = \square A \rightarrow BD, B \rightarrow \square, D \rightarrow E$ \square dokazati da vrijedi FZ $AE \rightarrow A$.

Dokaz:

- $A \rightarrow BD$ (P2: dekompozicija) $\Rightarrow A \rightarrow B$
- $A \rightarrow B \wedge B \rightarrow \square$ (A3: tranzitivnost) $\Rightarrow A \rightarrow \square$
- (A1: refleksivnost) $\Rightarrow A \rightarrow A$
- $A \rightarrow A \wedge A \rightarrow \square$ (P1: unija) $\Rightarrow A \rightarrow A \square$
- $A \rightarrow A \square$ (A2: uvećanje) $\Rightarrow AE \rightarrow A \square$

Pravilo o akumulaciji

Sljedeće dodatno pravilo omogućuje \square algoritamski pristup rješavanju sličnih zadataka

PRAVILO O AKUMULACIJI

- Ako u shemi \square vrijedi
 - $\square \rightarrow \square Z$ i $Z \rightarrow \square$ tada vrijedi i $\square \rightarrow \square Z$

Primjer korištenja pravila o akumulaciji

\square z pretpostavku da na relacijskoj shemi $R = \square A, B, \square, D, E$ \square vrijedi skup funkcijskih zavisnosti $F = \square A \rightarrow BD, B \rightarrow \square, D \rightarrow E$ \square dokazati da vrijedi FZ $AE \rightarrow A$.

Označimo lijevu stranu FZ s \square ($\square = AE$), a desnu stranu FZ s \square ($\square = A$).

Dokaz (primjenom A-1, pravila o akumulaciji i P-2):

- korak: $\square \rightarrow \square$
 - (A1: refleksivnost) $\Rightarrow AE \rightarrow AE$
 - u sljedećim koracima pomoću pravila akumulacije \square uvećavati desnu stranu FZ \square sve dok desna strana ne sadrži \square
- $\left\{ \begin{array}{l} AE \rightarrow AE \wedge A \rightarrow BD \text{ (akumulacija)} \Rightarrow AE \rightarrow AE BD \end{array} \right.$
- $\left\{ \begin{array}{l} AE \rightarrow AE BD \wedge B \rightarrow \square \text{ (akumulacija)} \Rightarrow AE \rightarrow AE BD \square \end{array} \right.$
- u zadnjem koraku, kad (i ako) desna strana FZ sadrži \square
 - $\left\{ \begin{array}{l} AE \rightarrow AE BD \square \text{ (P2: dekompozicija)} \Rightarrow AE \rightarrow A \square \end{array} \right.$

Primjer korištenja pravila o akumulaciji (za vježbu 1)

$R = \square \square, M, N, P, \square, R$ $F = \square \square \rightarrow R, M \rightarrow P \square, P \square \square \rightarrow N$ \square dokazati da vrijedi FZ $M \square R \rightarrow \square N$.

- (A1: refleksivnost) $\Rightarrow M \square R \rightarrow M \square R$
- $M \square R \rightarrow M \square R \wedge M \rightarrow P \square$ (akumulacija) $\Rightarrow M \square R \rightarrow M \square R P \square$
- $M \square R \rightarrow M \square R P \square \wedge P \square \square \rightarrow N$ (akumulacija) $\Rightarrow M \square R \rightarrow M \square R P \square N$
- $M \square R \rightarrow M \square R P \square N$ (P2: dekompozicija) $\Rightarrow M \square R \rightarrow \square N$

Primjer korištenja pravila o akumulaciji (za vježbu 2)

$R = \{M, N, P, Q, R\}$, $F = \{M \rightarrow R, M \rightarrow P, P \rightarrow N\}$
dokazati da vrijedi FZ $M \rightarrow N$.

- (A1: refleksivnost) $\Rightarrow M \rightarrow M$
- $M \rightarrow M \wedge Q \rightarrow R$ (akumulacija) $\Rightarrow M \rightarrow M \wedge R$
- $M \rightarrow M \wedge M \rightarrow P$ (akumulacija) $\Rightarrow M \rightarrow M \wedge P$
- ne postoji FZ kojom bi se moglo nastaviti \Rightarrow uvećavati desnu stranu
- $\Rightarrow M \rightarrow N$ ne vrijedi

Ključ entiteta i ključ relacije

- entitet je bilo što, što ima suštinu ili bit i posjeduje značajke s pomoću kojih se može razlučiti od svoje okoline
- ključ entiteta sadrži one atribute koji omogućuju da se pojedini entiteti mogu razlučiti od okoline
- relacijom se opisuje skup entiteta

Ključ relacije je skup atributa koji nedvosmisleno određuje n-torke relacije.

- Ključ relacije ima svojstvo da funkcijski određuje atribute u preostalom dijelu relacije

Ključ relacije

- ključ relacijske sheme R je skup atributa K , $K \subseteq R$, koji ima sljedeća svojstva:
 - $K \rightarrow (R - K)$ (također vrijedi i $K \rightarrow R$)
 - ključ funkcijski određuje atribute u preostalom dijelu relacijske sheme
 - ne postoji $K' \subset K$ za kojeg vrijedi $K' \rightarrow R$
 - ključ je minimalan skup atributa koji funkcijski određuje atribute u preostalom dijelu relacijske sheme

Ključ relacije - primjer

osoba	matBr	prezime	ime	postBr	grad
	11234	Novak	osip	21000	Split
	12345	orvat	Ivan	10000	Zagreb
	23456	Kolar	Ana	31000	Osijek
	34567	Novak	osip	10000	Zagreb

Ključ: $K_{\text{osoba}} = \text{matBr}$

$\text{matBr} \rightarrow \text{prezime}$
 $\text{matBr} \rightarrow \text{ime}$
 $\text{matBr} \rightarrow \text{postBr}$
 $\text{matBr} \rightarrow \text{grad}$

Za $K = \text{matBr}$, prezime također vrijedi

$K \rightarrow \text{ime, postBr, grad}$

ali K nije ključ jer postoji $K' \subset \text{matBr}$ i $K' \rightarrow R$, za kojeg vrijedi

$K' \rightarrow \text{prezime, ime, postBr, grad}$

Ključevi relacije

- mogući ključevi ($\{ \text{matBr, prezime, ime, postBr, grad} \}$)
- primarni ključ (matBr) odabire se jedan od mogućih ključeva
- alternativni ključevi ($\{ \text{prezime, ime, postBr, grad} \}$) ostali mogući ključevi

PRIMER:

djelatnik	matBr	prezime	ime	MB
	11234	Novak	osip	1403970330103
	12345	orvat	Ivan	2812964310267
	23456	Kolar	Ana	0111959335208
	34567	Novak	osip	0301949320319

- mogući ključevi:
 - matBr
 - MB
- primarni ključ: matBr
- alternativni ključ: MB

Struktura relacije

- Relacijska shema sastoji se od:
 - atributa koji su dio ključa (ključni atributi, ključni dio relacije)
 - atributa iz zavisnog dijela relacije (neključni atributi, neključni dio relacije)

PRIMER:

djelatnik	matBr	prezime	ime	MB
	11234	Novak	osip	1403970330103
	12345	orvat	Ivan	2812964310267
	23456	Kolar	Ana	0111959335208
	34567	Novak	osip	0301949320319

- primarni ključ: matBr
- alternativni ključ: MB
 - ključni atributi, ključni dio relacije:
 - matBr
 - MB
 - neključni atributi, neključni dio relacije:
 - prezime
 - ime

Zadatak:

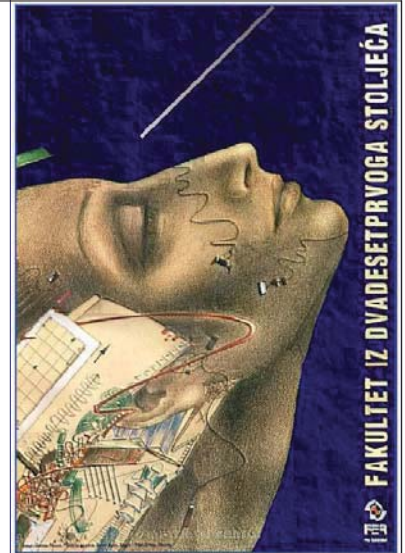
- Odrediti moguće ključeve, primarni ključ, alternativne ključeve, ključni dio relacije, neključni dio relacije
 - uzeti u obzir da klub tijekom istog dana može igrati najviše jednu utakmicu

utakmicaPrvenstva	domaci	gosti	datum	rezultat
Arsenal	Liverpool		12.06.2007	2:1
Arsenal	Liverpool		08.03.2008	2:1
Newcastle	Everton		08.03.2008	3:3
Everton	Liverpool		22.03.2008	4:0
Liverpool	Everton		05.04.2008	5:2

Baze podataka

Predavanja
travanj 2008.

7. Pregledavanje sheme relacijske baze podataka (2. dio)



Normalizacija

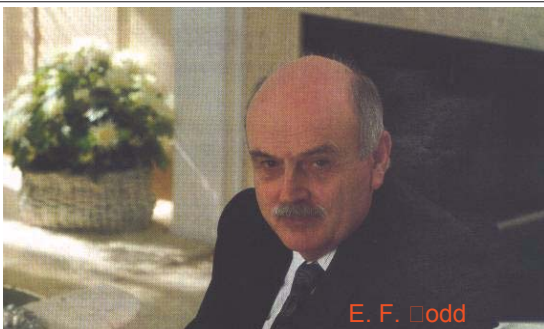
Postupci normalizacije

- Preko znanja o međusobnim funkcijskim zavisnostima atributa relacije koriste se u postupcima normalizacije.
- ili:
 - ukloniti redundanciju
 - anomalije unosa, izmjene i brisanja
 - neracionalno korištenje prostora za pohranu
 - spriječiti pojavu lažnih n-torki
- Postupci normalizacije omogućavaju da se postupno, **točno definiranom metodom**, odredi dobra zamjena za loše koncipiranu relacijsku shemu

E. F. Codd:

Normalized data base structure: A brief tutorial

Original paper: <http://www.dbms.org/ftp/codd/normalization.pdf>



E. F. Codd

"I called it normalization because then-President Nixon was talking a lot about normalizing relations with China. I figured that if he could normalize relations, so could I."
(A "FIRESIDE" CHAT', DBMS, Dec. 1993)

Normalne forme

- Prva normalna forma - 1 NF
- Druga normalna forma - 2 NF
- Treća normalna forma - 3 NF
- Boice-Coddova normalna forma - BCNF
 - emelji se na FUNKCIJSKIM ZAVISNOSTIMA
- Četvrta normalna forma - 4NF
 - emelji se na KVAZINEZNAČNIM ZAVISNOSTIMA
- Projekcijsko-spojna normalna forma - P4NF
 - emelji se na SPOJNIM ZAVISNOSTIMA

Postupci normalizacije

- Dekompozicija
 - početne relacije (relacijske sheme) se dekomponiraju na temelju uočenih funkcijskih zavisnosti
- Sinteza
 - zadan je skup atributa i nad njima skup funkcijskih zavisnosti iz kojih se sintetiziraju relacijske sheme koje zadovoljavaju 3NF

Dekompozicija relacijske sheme (relacije)

- Dekompozicijom (razlaganjem) relacijska shema R zamjenjuje se shemama R_1, R_2, \dots, R_n , $R_i \subseteq R$, pri čemu vrijedi $R = R_1 \bowtie R_2 \bowtie \dots \bowtie R_n$
- Dekompozicijom se relacija $r(R)$ zamjenjuje relacijama $r_1(R_1), r_2(R_2), \dots, r_n(R_n)$, pri čemu je $r_i(R_i) = \pi_{R_i}(r)$, za $i = 1, \dots, n$
- Relacija $r(R)$ se dekomponira na relacije $r_1(R_1), r_2(R_2), \dots, r_n(R_n)$ **bez gubitaka informacija** (lossy decomposition) ako vrijedi:

$$r_1(R_1) \bowtie r_2(R_2) \bowtie \dots \bowtie r_n(R_n) = r(R)$$

odnosno

$$\pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \dots \bowtie \pi_{R_n}(r) = r(R)$$

Dekompozicija relacije - primjer

- Zadana je relacija:

A	B	C	D
a1	b1	c1	d1
a2	b2	c1	d1

- Relaciju $r(R)$ dekomponirati na relacije
 - $r_1(R_1)$, $R_1 = \{A, C\}$
 - $r_2(R_2)$, $R_2 = \{B, C\}$
 - $r_3(R_3)$, $R_3 = \{C, D\}$

$r_1(R_1)$

A	C
a1	c1
a2	c1

$r_2(R_2)$

B	C
b1	c1
b2	c1

$r_3(R_3)$

C	D
c1	d1

- Je li dekompozicija obavljena bez gubitaka informacija?

$$r_1(R_1) \bowtie r_2(R_2) \bowtie r_3(R_3)$$

A	B	C	D
a1	b1	c1	d1
a1	b2	c1	d1
a2	b1	c1	d1
a2	b2	c1	d1

⇒ **N**

Razlaganje relacije bez gubitaka na dvije projekcije

- Relacija se bez gubitaka razlaže na svoje dvije projekcije ako:
 - projekcije imaju zajedničketribute
 - zajednički atributi su ključ u barem jednoj od projekcija

PRIMER:

osoba	matBr	prez	ime	postBr	nazMj
	11234	Novak	Josip	21000	Split
	12345	Horvat	Ivan	10000	Zagreb
	23456	Kolar	Ana	31000	Osijek
	34567	Novak	Josip	10000	Zagreb

$$osoba_1 = \pi_{matBr, prez, ime, postBr}(osoba)$$

$$mjesto = \pi_{postBr, nazMj}(osoba)$$

- Može li se relacija osoba dekomponirati bez gubitaka informacija na relacije osoba₁ i mjesto? Odnosno, vrijedi li:

$$osoba \equiv osoba_1 \bowtie mjesto$$

Primjer razlaganja relacije na dvije projekcije

osoba₁

matBr	prez	ime	postBr
11234	Novak	Josip	21000
12345	Horvat	Ivan	10000
23456	Kolar	Ana	31000
34567	Novak	Josip	10000

$$OSOBA_1 = \{matBr, prez, ime, postBr\}$$

$$K_{OSOBA_1} = \{matBr\}$$

mjesto

postBr	nazMj
21000	Split
10000	Zagreb
31000	Osijek

$$MESTO = \{postBr, nazMj\}$$

$$K_{MESTO} = \{postBr\}$$

$$OSOBA_1 \cap MESTO = \{postBr\}$$

$$\Rightarrow osoba \equiv osoba_1 \bowtie mjesto$$

osoba

matBr	prez	ime	postBr	nazMj
11234	Novak	Josip	21000	Split
12345	Horvat	Ivan	10000	Zagreb
23456	Kolar	Ana	31000	Osijek
34567	Novak	Josip	10000	Zagreb

Prva normalna forma (1NF)

- Definicija:

Relacijska shema je u 1NF ako:

- domene atributa sadrže samo jednostavne (nedjeljive) vrijednosti
- vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa
- neključni atributi relacije funkcijski ovise o ključu relacije

- Shema baze podataka $\{R_1, R_2, \dots, R_n\}$ je u 1NF ako je svaka relacijska shema R_1, R_2, \dots, R_n u 1NF

Prva normalna forma - primjer

- Poduzeće evidentira podatke o radnicima
 - $RADNIK = \{matBr, prezime, ime, datRod, sifOdjel\}$

radnik (RADNIK)	matBr	prezime	ime	datRod	sifOdjel
	1111	Novak	Ivan	28.12.1970	50
	1121	Kolar	Iva	16.10.1965	30
	1133	orvat	Krešo	19.03.1978	50

$K_{RADNIK} = \{matBr\}$

- domene svih atributa sadrže jednostavne (nedjeljive) vrijednosti
- vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa
- neključni atributi relacije funkcijski ovise o ključu relacije

⇒ Relacijska shema RADNIK je u 1NF

Prva normalna forma - primjer

- Poduzeće evidentira podatke o radnicima i njihovoj djeci - korisnicima zdravstvenog osiguranja.
 - $RADNIK_1 = \{matBr, prezime, ime, imenaDjece\}$

radnik ₁ (RADNIK ₁)	matBr	prezime	ime	imenaDjece
	1111	Novak	Ivan	asna, edran
	1121	Kolar	Iva	Ivan
	1133	orvat	Krešo	Petar, Ana, Ivan

$K_{RADNIK_1} = \{matBr\}$

- domena atributa i $\{asna, edran\}$ ne sadrži jednostavne (nedjeljive) vrijednosti

⇒ Relacijska shema RADNIK₁ nije u 1NF

Prva normalna forma - primjer

- Poduzeće evidentira podatke o radnicima i njihovoj djeci - korisnicima zdravstvenog osiguranja.
 - $RADNIK_2 = \{matBr, prezime, ime, imeDj, datRodDj\}$

radnik ₂ (RADNIK ₂)	matBr	prezime	ime	imeDj	datRodDj
	1111	Novak	Ivan	asna edran	21.01.1995 13.12.1997
	1121	Kolar	Iva	Ivan	23.03.2000
			Petar		22.02.1998
	1133	orvat	Krešo	Ana	19.09.2000
			Ivan		05.11.2002

$K_{RADNIK_2} = \{matBr\}$

- domene sadrže jednostavne vrijednosti, ali vrijednost atributa i $\{asna, edran\}$ nije uvijek samo jedna vrijednost iz domene tog atributa (isto vrijedi i za atribut $\{asna, edran\}$)

⇒ Relacijska shema RADNIK₂ nije u 1NF

Normalizacija na 1NF - izdvajanjem atributa u posebnu relaciju

- u posebnu relaciju izdvaja se skup atributa koji se ponavlja s jednakom kratnošću, zajedno s ključem originalne relacije

$RADNIK_2 = \{matBr, prezime, ime, imeDj, datRodDj\}$ $K_{RADNIK_2} = \{matBr\}$

radnik ₃ (RADNIK ₃)	matBr	prezime	ime
	1111	Novak	Ivan
	1121	Kolar	Iva
	1133	orvat	Krešo

$RADNIK_3 = \{matBr, prezime, ime\}$

$K_{RADNIK_3} = \{matBr\}$

dijete (DIJE)	matBr	imeDj	datRodDj
	1111	asna	21.01.1995
	1111	edran	13.12.1997
	1121	Ivan	23.03.2000
	1133	Petar	22.02.1998
	1133	Ana	19.09.2000
	1133	Ivan	05.11.2002

$DIJE = \{matBr, imeDj, datRodDj\}$

$K_{DIJE} = \{matBr, imeDj\}$

- operacija je izvedena bez gubitaka informacija - relacijske sheme imaju zajedničke attribute (matBr), zajednički atributi su ključ u RADNIK₃

Normalizacija na 1NF - promjenom ključa

$RADNIK_2 = \{matBr, prezime, ime, imeDj, datRodDj\}$ $K_{RADNIK_2} = \{matBr\}$

$RADNIK_4 = \{matBr, prezime, ime, imeDj, datRodDj\}$

$K_{RADNIK_4} = \{matBr, imeDj\}$

radnik ₄ (RADNIK ₄)	matBr	prezime	ime	imeDj	datRodDj
	1111	Novak	Ivan	asna	21.01.1995
	1111	Novak	Ivan	edran	13.12.1997
	1121	Kolar	Iva	Ivan	23.03.2000
	1133	orvat	Krešo	Petar	22.02.1998
	1133	orvat	Krešo	Ana	19.09.2000
	1133	orvat	Krešo	Ivan	05.11.2002

Druga normalna forma (2NF)

- Definicija:

Relacijska shema R je u 2NF ako je u 1NF i ako je

- svaki atribut iz zavisnog dijela potpuno funkcijski ovisan o svakom ključu relacije

- Shema baze podataka $\{R_1, R_2, \dots, R_n\}$ je u 2NF ako je svaka relacijska shema R_1, R_2, \dots, R_n u 2NF

Potpuna funkcijska zavisnost

- Skup atributa α **potpuno je funkcijski ovisan** o skupu atributa β relacijske sheme R ako:
 - α funkcijski ovisi o β
 - ne postoji pravi podskup od β koji funkcijski određuje α

PRIMER:

Zadan je skup FZ $F = \alpha AB \rightarrow DE, E \rightarrow F$

Je li $\alpha D, E$ potpuno funkcijski ovisan o $\alpha A, B$, α α

Da, jer ne postoji skup $Z \subset \alpha A, B, \alpha$ takav da $Z \rightarrow \alpha D, E$

Nepotpuna funkcijska zavisnost

Zadana je relacijska shema R i skupovi atributa α i β iz R, tj. $\alpha \subseteq R, \beta \subseteq R$. Neka u R vrijedi FZ $\alpha \rightarrow \beta$.

FZ $\alpha \rightarrow \beta$ je **nepotpuna** ako postoji skup atributa Z koji je podskup od α , za koji vrijedi $Z \rightarrow \beta$ odnosno

FZ $\alpha \rightarrow \beta$ je nepotpuna ako $(\exists Z) (Z \subset \alpha) : Z \rightarrow \beta$

PRIMER:

Zadan je skup FZ $F = \alpha AB \rightarrow D, B \rightarrow E, E \rightarrow D$

Je li αD potpuno funkcijski ovisan o $\alpha A, B, \alpha$ α

Ne, jer postoji skup $\alpha B, \alpha \subset \alpha A, B, \alpha$ takav da $\alpha B, \alpha \rightarrow \alpha D$

Druga normalna forma - primjer

$RADNIK_4 = \alpha matBr, prezime, ime, imeDj, datRodDj$ $K_{RADNIK_4} = \alpha matBr, imeDj$

radnik ₄ (RADNIK ₄)	matBr	prezime	ime	imeDj	datRodDj
	1111	Novak	Ivan	asna	21.01.1995
	1111	Novak	Ivan	edran	13.12.1997
	1121	Kolar	Iva	Ivan.	23.03.2000
	1133	orvat	Krešo	Petar	22.02.1998
	1133	orvat	Krešo	Ana	19.09.2000
	1133	orvat	Krešo	Ivan	05.11.2002

- relacijska shema $RADNIK_4$ zadovoljava 1NF
- postoji $\alpha Z: matBr \rightarrow prezime, ime$
- $matBr, imeDj \rightarrow prezime, ime$ je nepotpuna FZ!

\Rightarrow Relacijska shema $RADNIK_4$ nije u 2NF

Normalizacija na 2NF

- Normalizacijom na 2NF nastaju:

- relacijska shema koja sadrži skup atributa koji su bili nepotpuno funkcijski ovisni o ključu i dio ključa o kojem su potpuno funkcijski ovisni
- relacijska shema koja sadrži ključ originalne relacije i skup atributa koji su potpuno funkcijski ovisni o ključu

$RADNIK_5 = \alpha matBr, prezime, ime$

$K_{RADNIK_5} = \alpha matBr$

radnik ₅ (RADNIK ₅)	matBr	prezime	ime
	1111	Novak	Ivan
	1121	Kolar	Iva
	1133	orvat	Krešo

$DIJETE = \alpha matBr, imeDj, datRodDj$

$K_{DIJETE} = \alpha matBr, imeDj$

dijete (DIJETE)	matBr	imeDj	datRodDj
	1111	asna	21.01.1995
	1111	edran	13.12.1997
	1121	Ivan.	23.03.2000
	1133	Petar	22.02.1998
	1133	Ana	19.09.2000
	1133	Ivan	05.11.2002

Normalizacija na 2NF

Neka su α, β, Z atributi ili skupovi atributa. Zadana je relacijska shema $R = \alpha \alpha Z$ i na njoj skup funkcijskih zavisnosti $F = \alpha \alpha \rightarrow Z, \alpha \rightarrow Z$. Ključ relacije $K_R = \alpha \alpha$. R je u 1NF. **Zadovoljava li R 2NF?**

- funkcijska zavisnost $\alpha \rightarrow Z$ je nepotpuna
- R ne zadovoljava 2NF

Normalizacijom na 2NF shema R se zamjenjuje shemama:

$R_1 = \alpha Z$ $R_2 = \alpha \alpha$

$K_{R_1} = \alpha$ $K_{R_2} = \alpha \alpha$

Relacija $r(R)$ se normalizacijom na 2NF zamjenjuje projekcijama:

$r_1 = \pi_{\alpha Z}(r)$ $r_2 = \pi_{\alpha \alpha}(r)$

- operacija je izvedena bez gubitaka informacija - relacijske sheme imaju zajedničke attribute (α), zajednički atributi su ključ u R_1 .

reća normalna forma (3NF)

- Definicija:

Relacijska shema je u 3NF ako je u 1NF i ako:

- niti jedan atribut iz zavisnog dijela nije tranzitivno funkcijski ovisan o bilo kojem ključu relacije
- Shema baze podataka $\alpha = \alpha R_1, R_2, \alpha, R_n$ je u 3NF ako je svaka relacijska shema R_1, R_2, α, R_n u 3NF

Tranzitivna funkcionalna zavisnost

Zadano je:

- relacijska shema R,
- skupovi atributa $X \subseteq R$, $Y \subseteq R$, $Z \subseteq R$

Skup atributa Z je tranzitivno ovisan o X ako vrijedi:

- $X \rightarrow Y$ i $Y \rightarrow Z$

Tranzitivna funkcionalna zavisnost - primjer

Zadana je relacijska shema $R = \{A, B, D, E, F\}$ i skup FZ $F = \{AB \rightarrow D, D \rightarrow EF, D \rightarrow AB\}$

EF je tranzitivno funkcionalno ovisan o AB, jer

- $AB \rightarrow D, D \rightarrow EF$

AB nije tranzitivno funkcionalno ovisan o EF, jer iako

- $AB \rightarrow D, D \rightarrow EF$
- nije zadovoljen uvjet $AB \rightarrow EF$

reći normalna forma - primjer

$OSOBA = \{matBr, prez, ime, postBr, nazMjesto\}$ $K_{OSOBA} = \{matBr\}$

osoba (OSOBA)	matBr	prez	ime	postBr	nazMjesto
	1111	Novak	Ivan	10000	Zagreb
	1121	Kolar	Iva	31000	Osijek
	1133	Orvat	Krešo	10000	Zagreb

- Relacijska shema OSOBA zadovoljava 1NF.

- vrijedi FZ: $matBr \rightarrow postBr$
- vrijedi FZ: $postBr \rightarrow nazMjesto$
- ne vrijedi FZ: $postBr \rightarrow matBr$

→ $matBr \rightarrow nazMjesto$ je tranzitivna zavisnost !

- Relacijska shema OSOBA ne zadovoljava 3NF.

Normalizacija na 3NF

$OSOBA = \{matBr, prez, ime, postBr, nazMjesto\}$ $K_{OSOBA} = \{matBr\}$

Normalizacijom na 3NF nastaju:

- relacijska shema koja sadrži skup atributa relacijske sheme OSOBA koji su tranzitivno ovisni o ključu ($\{matBr\}$) te srednji skup atributa uočene tranzitivne zavisnosti ($\{postBr, nazMjesto\}$)
- relacijska shema koja sadrži ključ relacijske sheme OSOBA ($\{matBr\}$) i neključne attribute relacijske sheme OSOBA koji nisu tranzitivno ovisni o ključu

$MES = \{postBr, nazMjesto\}$ $K_{MES} = \{postBr\}$

$K_{MES} = \{postBr\}$

mjesto (MES)	postBr	nazMjesto
	10000	Zagreb
	31000	Osijek

$OSOBA_1 = \{matBr, prezime, ime, postBr\}$

$K_{OSOBA_1} = \{matBr\}$

osoba ₁ (OSOBA ₁)	matBr	prezime	ime	postBr
	1111	Novak	Ivan	10000
	1121	Kolar	Iva	31000
	1133	Orvat	Krešo	10000

Normalizacija na 3NF

Neka su X, Y, Z atributi ili skupovi atributa. Zadana je relacijska shema $R = \{X, Y, Z\}$ i na njoj skup funkcionalnih zavisnosti $F = \{X \rightarrow Y, Y \rightarrow Z\}$. Ključ relacije $K_R = \{X\}$. R je u 1NF. **Zadovoljava li R 3NF?**

- funkcijska zavisnost $X \rightarrow Y$ je tranzitivna
- R ne zadovoljava 3NF

Normalizacijom na 3NF shema R se zamjenjuje shemama:

$$R_1 = \{X, Y\} \quad R_2 = \{Y, Z\}$$
$$K_{R_1} = \{X\} \quad K_{R_2} = \{Y\}$$

Relacija r(R) se normalizacijom na 3NF zamjenjuje projekcijama:

$$r_1 = \pi_{X,Y}(r) \quad r_2 = \pi_{Y,Z}(r)$$

- operacija je izvedena bez gubitaka informacija - relacijske sheme imaju zajedničke attribute (Z), zajednički atributi su ključ u R_2 .

reći normalna forma - komentar

Normalizacija na 2NF nije nužni preduvjet za provođenje normalizacije na 3NF jer se nepotpune FZ mogu promatrati kao tranzitivne FZ.

Primjer: zadana je shema $R = \{X, Y, Z\}$ i na njoj skup funkcionalnih zavisnosti $F = \{X \rightarrow Y, Y \rightarrow Z\}$. Ključ relacije $K_R = \{X\}$. R je u 1NF, ali nije u 2NF jer postoji nepotpuna FZ $X \rightarrow Z$. Međutim, postoji i tranzitivna funkcionalna zavisnost $X \rightarrow Z$ ($X \rightarrow Y \wedge Y \rightarrow Z$).

Normalizacijom na 3NF shema R se zamjenjuje shemama:

$$R_1 = \{X, Y\} \quad R_2 = \{Y, Z\}$$
$$K_{R_1} = \{X\} \quad K_{R_2} = \{Y\}$$

R_1 i R_2 su u 2NF i 3NF

Preporuka: normalizaciju ipak obavljati postupno
1NF → 2NF → 3NF

Normalizacija na 3N - primjer

OSOBA₂ = {matBr, prezime, ime, MB}

osoba2 (OSOBA2)	matBr	prez	ime	MB
	1111	Novak	Ivan	1403970330103
	1121	Kolar	Iva	2812968310267
	1133	orvat	Krešo	0301979320319

- postoji FZ: matBr → prez ime MB
- postoje FZ: MB → prez ime i MB → matBr

- matBr i MB su mogući ključevi
- Relacijska shema OSOBA₂ zadovoljava 3NF.

K₁_{OSOBA2} = matBr

K₂_{OSOBA2} = MB

Normalizacija na 3N - dodatna razmatranja

Neka su A, B, Z atributi ili skupovi atributa. Zadata je relacijska shema R = {A, B, Z} i na njoj skup funkcijskih zavisnosti F = {A → B, B → Z, Z → A}. Neka je ključ K_R = A.

- vrijedi A → Z i B → Z, ali A → B nije tranzitivna FZ jer vrijedi i Z → A

- Zbog A → Z i B → Z funkcijsku zavisnost A → B nije potrebno ukloniti jer u tom slučaju nema redundancije.
- Z je također mogući ključ u R

K_{1R} = A

K_{2R} = Z

A i Z su mogući ključevi.

- Relacijska shema R zadovoljava 3NF.

1. primjer normalizacije

- Zadana je relacijska shema: ISPI = {matBr, prez, ime, sifPred, nazPred, datlsp, ocj, sifNas, prezNas} i trenutna vrijednost relacije ispit(ISPI):

matBr	prez	ime	sifPred	nazPred	datlsp	ocj	sifNas	prezNas
1111	Novak	Ivan	1001	Mat-1	29.01.06	1	1111	Pašić
1111	Novak	Ivan	1001	Mat-1	05.02.06	3	1111	Pašić
1111	Novak	Ivan	1003	Fiz-1	28.06.06	2	3333	orvat
1111	Novak	Ivan	1002	Mat-2	27.06.06	4	2222	Brnetić
1234	Kolar	Petar	1001	Mat-1	29.01.06	3	2222	Brnetić

- funkcijske zavisnosti odrediti na temelju značenja podataka
- odrediti primarni ključ relacije (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu)
- postupno normalizirati relacijsku shemu ISPI na 2NF i 3NF

1. primjer normalizacije - 1N

matBr	prez	ime	sifPred	nazPred	datlsp	ocj	sifNas	prezNas
1111	Novak	Ivan	1001	Mat-1	29.01.06	1	1111	Pašić
1111	Novak	Ivan	1001	Mat-1	05.02.06	3	1111	Pašić
1111	Novak	Ivan	1003	Fiz-1	28.06.06	2	3333	orvat
1111	Novak	Ivan	1002	Mat-2	27.06.06	4	2222	Brnetić
1234	Kolar	Petar	1001	Mat-1	29.01.06	3	2222	Brnetić

- Određivanje ključa: ako se (pogrešno) pretpostavi da je K = {matBr} Bi li tada postojali neključni atributi koje ključ funkcijski ne određuje?
- matBr → prez ime
- međutim:
- matBr ↛ sifPred matBr ↛ nazPred
matBr ↛ datlsp matBr ↛ ocj
matBr ↛ sifNas matBr ↛ prezNas

1. primjer normalizacije - 1N

matBr	prez	ime	sifPred	nazPred	datlsp	ocj	sifNas	prezNas
1111	Novak	Ivan	1001	Mat-1	29.01.06	1	1111	Pašić
1111	Novak	Ivan	1001	Mat-1	05.02.06	3	1111	Pašić
1111	Novak	Ivan	1003	Fiz-1	28.06.06	2	3333	orvat
1111	Novak	Ivan	1002	Mat-2	27.06.06	4	2222	Brnetić
1234	Kolar	Petar	1001	Mat-1	29.01.06	3	2222	Brnetić

- Ako se pretpostavi K = {matBr, sifPred, datlsp} Bi li tada postojali neključni atributi koje ključ funkcijski ne određuje?

- matBr sifPred datlsp → prez ime nazPred ocj sifNas prezNas
postoji li skup X ⊂ {matBr, sifPred, datlsp} za kojeg vrijedi X → R?
⇒ N ⊂ {matBr, sifPred, datlsp} je mogući ključ

K_{ISPI} = {matBr, sifPred, datlsp}

- zadovoljen je uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu

1. primjer normalizacije - 2N

matBr	prez	ime	sifPred	nazPred	datlsp	ocj	sifNas	prezNas
1111	Novak	Ivan	1001	Mat-1	29.01.06	1	1111	Pašić
1111	Novak	Ivan	1001	Mat-1	05.02.06	3	1111	Pašić
1111	Novak	Ivan	1003	Fiz-1	28.06.06	2	3333	orvat
1111	Novak	Ivan	1002	Mat-2	27.06.06	4	2222	Brnetić
1234	Kolar	Petar	1001	Mat-1	29.01.06	3	2222	Brnetić

- Postoje li neključni atributi koji ne ovise o čitavom ključu nego samo o dijelu ključa? matBr → prez ime
student = π_{matBr, prez, ime}(ispit)
ispit₁ = π_{matBr, sifPred, nazPred, datlsp, ocj, sifNas, prezNas}(ispit)

matBr	sifPred	nazPred	datlsp	ocj	sifNas	prezNas
1111	1001	Mat-1	29.01.06	1	1111	Pašić
1111	1001	Mat-1	05.02.06	3	1111	Pašić
1111	1003	Fiz-1	28.06.06	2	3333	orvat
1111	1002	Mat-2	27.06.06	4	2222	Brnetić
1234	1001	Mat-1	29.01.06	3	2222	Brnetić

1. primjer normalizacije - 2N (nastavak)

- Postoje li neključni atributi koji ne ovise o čitavom ključu nego samo o dijelu ključa

sifPred → nazPred

predmet = $\pi_{\text{sifPred}, \text{nazPred}}(\text{ispit}_1)$

ispit₂ = $\pi_{\text{matBr}, \text{sifPred}, \text{datlsp}, \text{ocj}, \text{sifNas}, \text{prezNas}}(\text{ispit}_1)$

$K_{\text{PREDME}} = \text{sifPred}$

$K_{\text{ISPI}_2} = \text{matBr}, \text{sifPred}, \text{datlsp}$

predmet (PREDME)	
sifPred	nazPred
1001	Mat-1
1003	Fiz-1
1002	Mat-2

2N → 3N → 4K.

ispit ₂ (ISPI ₂)					
matBr	sifPred	datlsp	ocj	sifNas	prezNas
1111	1001	29.01.06	1	1111	Pašić
1111	1001	05.02.06	3	1111	Pašić
1111	1003	28.06.06	2	3333	Orvat
1111	1002	27.06.06	4	2222	Brnetić
1234	1001	29.01.06	3	2222	Brnetić

2N → 4K.

1. primjer normalizacije - 3N

- Postoje li neključni atributi koji tranzitivno ovise o ključu

matBr sifPred datlsp → sifNas sifNas → prezNas

nastavnik = $\pi_{\text{sifNas}, \text{prezNas}}(\text{ispit}_2)$

ispit₃ = $\pi_{\text{matBr}, \text{sifPred}, \text{datlsp}, \text{ocj}, \text{sifNas}}(\text{ispit}_2)$

$K_{\text{NAS} \rightarrow \text{A} \rightarrow \text{NIK}} = \text{sifNas}$

nastavnik (NAS → A → NIK)	
sifNas	prezNas
1111	Pašić
3333	Orvat
2222	Brnetić

3N → 4K.

$K_{\text{ISPI}_3} = \text{matBr}, \text{sifPred}, \text{datlsp}$

ispit ₃ (ISPI ₃)				
matBr	sifPred	datlsp	ocj	sifNas
1111	1001	29.01.06	1	1111
1111	1001	05.02.06	3	1111
1111	1003	28.06.06	2	3333
1111	1002	27.06.06	4	2222
1234	1001	29.01.06	3	2222

3N → 4K.

1. primjer normalizacije - 3N

student (S → DEN)		
matBr	prez	ime
1111	Novak	Ivan
1234	Kolar	Petar

$K_{\text{S} \rightarrow \text{DEN}} = \text{matBr}$

predmet (PREDME)	
sifPred	nazPred
1001	Mat-1
1003	Fiz-1
1002	Mat-2

$K_{\text{PREDME}} = \text{sifPred}$

nastavnik (NAS → A → NIK)	
sifNas	prezNas
1111	Pašić
3333	Orvat
2222	Brnetić

$K_{\text{NAS} \rightarrow \text{A} \rightarrow \text{NIK}} = \text{sifNas}$

ispit₃ (ISPI₃)

matBr	sifPred	datlsp	ocj	sifNas
1111	1001	29.01.06	1	1111
1111	1001	05.02.06	3	1111
1111	1003	28.06.06	2	3333
1111	1002	27.06.06	4	2222
1234	1001	29.01.06	3	2222

$K_{\text{ISPI}_3} = \text{matBr}, \text{sifPred}, \text{datlsp}$

- Relacijska shema baze podataka S → S:

$S \rightarrow S = \text{S} \rightarrow \text{DEN}, \text{PREDME}, \text{NAS} \rightarrow \text{A} \rightarrow \text{NIK}, \text{ISPI}_3$

- Schema baze podataka S → S zadovoljava 3N

2. primjer normalizacije

Zadana je relacijska shema $R = AB \rightarrow DEF$ i na njoj skup funkcijskih zavisnosti

$F = AB \rightarrow DEF, A \rightarrow D, B \rightarrow F, F \rightarrow$

Domene atributa sadrže samo jednostavne vrijednosti, vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa.

Odrediti primarni ključ relacije (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu), te shemu postupno normalizirati na 2NF i 3NF.

2. primjer normalizacije

$R = AB \rightarrow DEF$

$F = AB \rightarrow DEF, A \rightarrow D, B \rightarrow F, F \rightarrow$

- drediti primarni ključ relacije

rijedi li $AB \rightarrow DEF$

DA

postoji li skup $\alpha \subset AB$ za kojeg vrijedi $\alpha \rightarrow R$

NI

⇒ AB je mogući ključ i može se odabrati kao primarni ključ sheme R .

$R = AB \rightarrow DEF$

$K_R = AB$

R je u 1NF

2. primjer normalizacije - 2N

$R = AB \rightarrow DEF$

$K_R = AB$

$F = AB \rightarrow DEF, A \rightarrow D, B \rightarrow F, F \rightarrow$

- Normalizacija na 2N

Svi atributi iz zavisnog dijela moraju biti potpuno funkcijski ovisni o ključu.

R $AB \rightarrow DEF$

- $AB \rightarrow D$ je nepotpuna FZ, jer vrijedi $A \rightarrow D$

R nije u 2NF

Normalizacijom na 2NF se R zamjenjuje shemama:

$R_1 = AD$

$K_{R_1} = A$

R_1 je u 2NF

$R_2 = AB \rightarrow EF$

$K_{R_2} = AB$

R_2 nije u 2NF

2. primjer normalizacije - 2NF (nastavak)

$R_1 = AD$ $K_{R_1} = A$
 $R_2 = AB \square EF \square \square$ $K_{R_2} = AB \square$
 $F = \square AB \square \rightarrow DEF \square \square, A \rightarrow D, B \square \rightarrow F \square \square, F \square \rightarrow \square \square$

Svi atributi iz zavisnog dijela moraju biti **potpuno** funkcijski ovisni o ključu.



- $AB \square \rightarrow F \square \square$ je nepotpuna FZ, jer vrijedi $B \square \rightarrow F \square \square$ R_2 nije u 2NF

Normalizacijom na 2NF se R_2 zamjenjuje shemama:

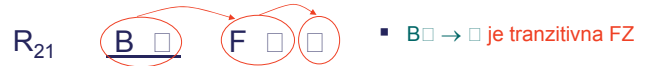
$R_{21} = B \square F \square \square$ $K_{R_{21}} = B \square$ R_{21} je u 2NF
 $R_{22} = AB \square E$ $K_{R_{22}} = AB \square$ R_{22} je u 2NF

2. primjer normalizacije - 3NF

$R_1 = AD$ $K_{R_1} = A$ R_1 je u 3NF
 $R_{21} = B \square F \square \square$ $K_{R_{21}} = B \square$ R_{21} nije u 3NF
 $R_{22} = AB \square E$ $K_{R_{22}} = AB \square$ R_{22} je u 3NF
 $F = \square AB \square \rightarrow DEF \square \square, A \rightarrow D, B \square \rightarrow F \square \square, F \square \rightarrow \square \square$

• Normalizacija na 3NF

Niti jedan atribut iz zavisnog dijela ne smije biti tranzitivno ovisan o ključu.



Normalizacijom na 3NF se R_{21} zamjenjuje shemama:

$R_{211} = B \square F \square$ $K_{R_{211}} = B \square$ R_{211} je u 3NF
 $R_{212} = F \square \square$ $K_{R_{212}} = F \square$ R_{212} je u 3NF

Shema baze podataka u 3NF sastoji se od relacijskih shema:

R_1, R_{22}, R_{211} i R_{212}

Boce-oddova normalna forma - BNF

• Definicija:

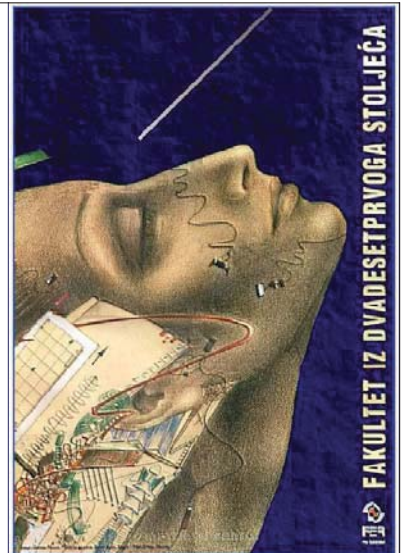
Relacijska shema R je u **BNF** ako je u 1NF i ako niti jedan atribut nije tranzitivno funkcijski ovisan o bilo kojem ključu relacije

- Iako je BNF stroža od 3NF, rijetki su slučajevi da je relacijska shema u 3NF, a da istovremeno nije i u BNF.
- Normalizaciju na BNF nije nužno provoditi.
- Smatra se da shema baze podataka ima dobra svojstva ako zadovoljava 3NF.

Baze podataka

Predavanja
travanj 2008.

8. prikazivanje sheme relacijske baze podataka (3. dio - primjeri)



Zadatak 1

- Prodavaonice šalju svoje narudžbe proizvođaču:

Konzum-7 Ilica 20 10 000 Zagreb Kraš Ravnice bb 10 000 Zagreb Narudžba br. 13/25 datum: 1.5.2006 Molimo isporučite nam 1200 komada proizvoda Napolitanke (šifra 129) i 2000 komada proizvoda Albert keks (šifra 139)	Diona-28 Bolška 7 21 000 Split Kraš Ravnice bb 10 000 Zagreb Narudžba br. 43-21 datum: 7.2.2006 Molimo isporučite nam 1200 komada proizvoda Napolitanke (šifra 129) i 1800 komada proizvoda Domaćica (šifra 221)	Konzum-7 Ilica 20 10 000 Zagreb Kraš Ravnice bb 10 000 Zagreb Narudžba br. 41/56 datum: 4.2.2007 Molimo isporučite nam 1100 komada proizvoda Napolitanke (šifra 129)
---	---	--

- proizvođač želi pohraniti podatke o narudžbama u svoju bazu podataka. Svi podaci se pohranjuju u relaciju **narudžbaArtikla**

narudžbaArtikla								
nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina

Zadatak 1

- Sadržaj relacije nakon unosa podataka iz prispjelih narudžbi:

narudžbaArtikla								
nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolška 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolška 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- Normalizirajte relaciju **narudžbaArtikla** na 1NF, 2NF, 3NF ako vrijedi da je svaki broj narudžbe jedinstven (ne može se desiti da brojevi narudžbi prispjelih iz različitih prodavaonica budu jednaki)

$brNar \rightarrow nazProd$

Zadatak 1

NA DZBAA IK A nazProd pbr nazMjesto adresa brNar datNar
sifArtikl nazArtikl kolicina

- trenutna vrijednost relacije narudzbaArtikla (NA DZBAA IK A) :

narudzbaArtikla

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- odrediti funkcijske zavisnosti na temelju značenja podataka
- odrediti primarni ključ relacije (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu)
- postupno normalizirati relacijsku shemu NAR DZBAAR IK A na 2NF i 3NF

Zadatak 1 - 1N

narudzbaArtikla

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- Određivanje ključa: bi li brNar bio dobar odabir za ključ
- Postoje li neključni atributi koji ne ovise o broju narudžbe (brNar)
- brNar → nazProd pbr nazMjesto adresa datNar
međutim:
 - brNar ↗ sifArtikl brNar ↗ nazArtikl brNar ↗ kolicina
- O kojim atributima funkcijski ovisi atribut nazArtikl sifArtikl → nazArtikl
- O kojim atributima funkcijski ovisi atribut kolicina brNar sifArtikl → kolicina
sifArtikl ↗ kolicina

Zadatak 1 - 1N

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- Pretpostavimo K = {brNar, sifArtikl}
 - Provjerite postoje li neključni atributi koje ključ funkcijski ne određuje.
- brNar sifArtikl → nazProd pbr nazMjesto adresa datNar nazArtikl kolicina
postoji li skup $\emptyset \subset \{brNar, sifArtikl\}$ za kojeg vrijedi $\emptyset \rightarrow R$
 - ⇒ N ⇒ {brNar, sifArtikl} je mogući ključ
- $K_{NAR:DZBAAR:IK:A} = \{brNar, sifArtikl\}$
- zadovoljen je uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu

Zadatak 1 - 2N

narudzbaArtikla

nazProd	pbr	nazMjesto	adresa	brNar	datNar	sifArtikl	nazArtikl	kolicina
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	129	Napolitanke	1200
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006	139	Albert keks	2000
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	129	Napolitanke	1200
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006	221	Domaćica	1800
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007	129	Napolitanke	1100

- Postoje li neključni atributi koji ovise o dijelu ključa
 - vrijedi: brNar → nazProd pbr nazMjesto adresa datNar
⇒ Na koje relacije treba razložiti relaciju narudzbaArtikla
Koji su ključevi novonastalih relacija
- narudzba = $\pi_{nazProd, pbr, nazMjesto, adresa, brNar, datNar}$ (narudzbaArtikla)
 $K_{NAR:DZBA} = \{brNar\}$
- stavkaNarudzbe = $\pi_{brNar, sifArtikl, nazArtikl, kolicina}$ (narudzbaArtikla)
 $K_{S:A:KANAR:DZBE} = \{brNar, sifArtikl\}$

Zadatak 1 - 2N

brNar → nazProd pbr nazMjesto adresa datNar

narudzba

nazProd	pbr	nazMjesto	adresa	brNar	datNar
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007

- narudzba ima jednostavan ključ
⇒ 2N □ K

stavkaNarudzbe

brNar	sifArtikl	nazArtikl	kolicina
13/25	129	Napolitanke	1200
13/25	139	Albert keks	2000
43-21	129	Napolitanke	1200
43-21	221	Domaćica	1800
41/56	129	Napolitanke	1100

esu li relacije narudzba i stavkaNarudzbe u 2N □

Zadatak 1 - 2N

- Je li stavkaNarudzbe u 2NF (postoje li neključni atributi koji ovise o dijelu ključa)

stavkaNarudzbe

brNar	sifArtikl	nazArtikl	kolicina
13/25	129	Napolitanke	1200
13/25	139	Albert keks	2000
43-21	129	Napolitanke	1200
43-21	221	Domaćica	1800
41/56	129	Napolitanke	1100

- vrijedi: sifArtikl → nazArtikl
⇒ Na koje relacije treba razložiti relaciju stavkaNarudzbe
Koji su ključevi novonastalih relacija

artikl = $\pi_{sifArtikl, nazArtikl}$ (stavkaNarudzbe) $K_{PREDE} = \{sifArtikl\}$

stavkaNarudzbe₁ = $\pi_{brNar, sifArtikl, kolicina}$ (stavkaNarudzbe)

$K_{S:A:KANAR:DZBE1} = \{brNar, sifArtikl\}$

Zadatak 1 - 2N

stavkaNarudzbe

brNar	sifArtikl	nazArtikl	kolicina
13/25	129	Napolitanke	1200
13/25	139	Albert keks	2000
43-21	129	Napolitanke	1200
43-21	221	Domaćica	1800
41/56	129	Napolitanke	1100

sifArtikl → nazArtikl

esu li relacije artikl i stavkaNarudzbe₁ u 2N

artikl

sifArtikl	nazArtikl
129	Napolitanke
139	Albert keks
221	Domaćica

2N

stavkaNarudzbe₁

brNar	sifArtikl	kolicina
13/25	129	1200
13/25	139	2000
43-21	129	1200
43-21	221	1800
41/56	129	1100

2N

Zadatak 1 - 3N

- Postoje li neključni atributi koji tranzitivno ovise o ključu

narudzba

nazProd	pbr	nazMjesto	adresa	brNar	datNar
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007

esu li relacija narudzba u 3N

stavkaNarudzbe₁

brNar	sifArtikl	kolicina
13/25	129	1200
13/25	139	2000
43-21	129	1200
43-21	221	1800
41/56	129	1100

3N

artikl

sifArtikl	nazArtikl
129	Napolitanke
139	Albert keks
221	Domaćica

3N

Zadatak 1 - 3N

- Postoje li u relaciji narudzba neključni atributi koji tranzitivno ovise o ključu

nazProd	pbr	nazMjesto	adresa	brNar	datNar
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007

- rijeti: brNar → nazProd nazProd → pbr nazMjesto adresa nazProd ↗ brNar

⇒ Na koje relacije treba razložiti relaciju narudzba
Koji su ključevi novonastalih relacija

prodavaonica = $\pi_{\text{nazProd, pbr, nazMjesto, adresa}}(\text{narudzba})$

$K_{\text{PRODAONICA}} = \text{nazProd}$

narudzba₁ = $\pi_{\text{brNar, nazProd, datNar}}(\text{narudzba})$

$K_{\text{NARUDZBA}_1} = \text{brNar}$

Zadatak 1 - 3N

narudzba

nazProd	pbr	nazMjesto	adresa	brNar	datNar
Konzum-7	10000	Zagreb	Ilica 20	13/25	1.5.2006
Diona-28	21000	Split	Bolska 7	43-21	7.2.2006
Konzum-7	10000	Zagreb	Ilica 20	41/56	4.2.2007

brNar → nazProd nazProd → pbr nazMjesto adresa nazProd ↗ brNar

esu li relacije prodavaonica i narudzba₁ u 3N

prodavaonica

nazProd	pbr	nazMjesto	adresa
Konzum-7	10000	Zagreb	Ilica 20
Diona-28	21000	Split	Bolska 7

3N

narudzba₁

brNar	nazProd	datNar
13/25	Konzum-7	1.5.2006
43-21	Diona-28	7.2.2006
41/56	Konzum-7	4.2.2007

3N

Zadatak 1 - 3N

- Postoje li neključni atributi koji tranzitivno ovise o ključu u relaciji prodavaonica

nazProd	pbr	nazMjesto	adresa
Konzum-7	10000	Zagreb	Ilica 20
Diona-28	21000	Split	Bolska 7

- nazProd → pbr pbr → nazMjesto pbr ↗ nazProd

⇒ Na koje relacije treba razložiti relaciju prodavaonica
Koji su ključevi novonastalih relacija

mjesto = $\pi_{\text{pbr, nazMjesto}}(\text{prodavaonica})$

$K_{\text{MESTO}} = \text{pbr}$

prodavaonica₁ = $\pi_{\text{nazProd, pbr, adresa}}(\text{prodavaonica})$

$K_{\text{PRODAONICA}_1} = \text{nazProd}$

Zadatak 1 - 3N

prodavaonica

nazProd	pbr	nazMjesto	adresa
Konzum-7	10000	Zagreb	Ilica 20
Diona-28	21000	Split	Bolska 7

- nazProd → pbr pbr → nazMjesto pbr ↗ nazProd

esu li relacije mjesto i prodavaonica₁ u 3N

mjesto

pbr	nazMjesto
10000	Zagreb
21000	Split

3N

prodavaonica₁

nazProd	pbr	adresa
Konzum-7	10000	Ilica 20
Diona-28	21000	Bolska 7

3N

Zadatak 1 - 3N

mjesto		prodavaonica ₁			artikl	
pbr	nazMjesto	nazProd	pbr	adresa	sifArtikl	nazArtikl
10000	Zagreb	Konzum-7	10000	Ilica 20	129	Napolitanke
21000	Split	Diona-28	21000	Bolska 7	139	Albert keks
					221	Domaćica

narudzba ₁			stavkaNarudzbe ₁		
brNar	nazProd	datNar	brNar	sifArtikl	kolicina
13/25	Konzum-7	1.5.2006	13/25	129	1200
43-21	Diona-28	7.2.2006	13/25	139	2000
41/56	Konzum-7	4.2.2007	43-21	129	1200
			43-21	221	1800
			41/56	129	1100

- Schema baze podataka u 3NF sastoji se od relacijskih shema:
mjesto, prodavaonica₁, artikl, narudzba₁, stavkaNarudzbe₁

Zadatak 2

Zadana je relacijska shema $R = AB \square DEF$ i na njoj skup funkcijskih zavisnosti:

$$F = \square AB \rightarrow \square D, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \square$$

Domene atributa sadrže samo jednostavne vrijednosti, vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa.

Odrediti primarni ključ relacijske sheme (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu), te shemu postupno normalizirati na 2NF i 3NF.

Zadatak 2 - 1N

$$R = AB \square DEF$$

$$F = \square AB \rightarrow \square D, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \square$$

- Odrediti primarni ključ relacije.

$$AB \rightarrow \square D$$

$$AB \rightarrow EF$$

$$\Rightarrow AB \rightarrow \square DEF \text{ (P-1: unija)}$$

postoji li skup $\square \subset AB$ za kojeg vrijedi $\square \rightarrow R \square$

NE

$$\Rightarrow \square \square AB \square D \square \square$$

$$K_{\square} = AB$$

R je u 1NF

Zadatak 2 - 2N

$$R = AB \square DEF$$

$$K_R = AB$$

$$F = \square AB \rightarrow \square D, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \square$$

Postoje li atributi iz zavisnog dijela koji nisu potpuno funkcijski ovisni o ključu

R



- $AB \rightarrow F$ je nepotpuna FZ, jer vrijedi $A \rightarrow F$

R nije u 2NF

Odredite relacijske sheme kojima treba zamijeniti relacijsku shemu R. Odredite ključeve.

$$R_1 = AF$$

$$K_{R_1} = A$$

R_1 je u 2NF

$$R_2 = AB \square DE$$

$$K_{R_2} = AB$$

R_2 je u 2NF

Zadatak 2 - 3N

$$R_1 = AF$$

$$K_{R_1} = A$$

$$R_2 = AB \square DE$$

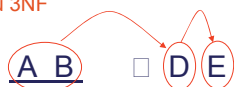
$$K_{R_2} = AB$$

$$F = \square AB \rightarrow \square D, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \square$$

- esu li R_1 i R_2 u 3N
- Postoje li u R_1 i R_2 neključni atributi koji tranzitivno ovise o ključu

R_1 je u 3NF

R_2



- $AB \rightarrow E$ je tranzitivna FZ

Odredite relacijske sheme kojima treba zamijeniti relacijsku shemu R_2 . Odredite ključeve.

$$R_{21} = DE$$

$$K_{R_{21}} = D$$

$$R_{22} = AB \square D$$

$$K_{R_{22}} = AB$$

Zadatak 2 - 3N

- esu li R_{21} i R_{22} u 3N

$$F = \square AB \rightarrow \square D, AB \rightarrow EF, A \rightarrow F, D \rightarrow E \square$$

$$R_1 = AF$$

$$K_{R_1} = A$$

R_1 je u 3NF

$$R_{21} = DE$$

$$K_{R_{21}} = D$$

R_{21} je u 3NF

$$R_{22} = AB \square D$$

$$K_{R_{22}} = AB$$

R_{22} je u 3NF

Schema baze podataka u 3NF sastoji se od relacijskih shema:

$$R_1, R_{21}, R_{22}$$

Zadatak 3

biblioteci se evidentiraju **posudbe (primjeraka) knjiga**.

Relacijska shema **POSDBAPRIM** sastoji se od sljedećih atributa:

sif_{cln} - šifra člana
 prez_{cln} - prezime člana
 ime_{cln} - ime člana
 pbr - poštanski broj mjesta stanovanja člana
 nazMj - naziv mjesta stanovanja člana
 adr_{cln} - adresa člana
 invBrPrim - inventarski broj primjerka
 datPos - datum posudbe
 dat_{vr} - datum vraćanja (datum kad je primjerak vraćen)
 sifKnj - šifra knjige
 nazKnj - naziv knjige
 siflzd - šifra izdavača
 nazlzd - naziv izdavača

rijede sljedeća pravila:

- jedan član istoga dana može posuditi više primjeraka
- jedan član isti primjerak može posuditi više puta, ali ne istog dana
- jedna knjiga ima jednog izdavača

Zadatak 3

posudbaPrimj						posud			
sif _{cln}	prez _{cln}	ime _{cln}	pbr	nazMj	adr _{cln}	invBrPrim	datPos	dat _{vr}	sifKnj, nazKnj, siflzd, nazlzd
123	Novak	lasna	10000	Zagreb	nska 6	11234	3.1.2007.	567	Kiklop, 12, A, M
124	orvat	Krešo	10020	Zagreb	Siget 8	21345	3.1.2007.	2.2.2007.	351, eto, 12, A, M
234	rgić	Ana	10000	Zagreb	Krčka 1	19435	29.1.2007.	459	Bajke, 15, BZ

- $K_{\text{PosudbaPrimj}} = \{sif_{cln}\}$
 $sif_{cln} \rightarrow prez_{cln} ime_{cln} pbr nazMj adr_{cln} posud$
- rijednosti atributa *posu* su n-torke koje sadrže vrijednosti atributa:
 $i_{10000}r_{10000}r_{10000}os_{10000}r_{10000}r_{10000}si_{10000}r_{10000}r_{10000}si_{10000}r_{10000}r_{10000}$
- Normalizirajte relacijsku shemu **POSDBAPRIM** na 1NF (izdvajanjem atributa u novu relaciju), 2NF i 3NF

Zadatak 3 1NF

clan	sif	prez	ime	pbr	nazMj	adr
	123	Novak	lasna	10000	Zagreb	nska 6
	124	orvat	Krešo	10020	Zagreb	Siget 8
	234	rgić	Ana	10000	Zagreb	Krčka 1

$K_{\text{AN}} = \{sif_{cln}, prez_{cln}, ime_{cln}, pbr, nazMj, adr_{cln}\}$ $K_{\text{AN}} = \{sif_{cln}\}$

$POSDBA = \{sif_{cln}, invBrPrim, datPos, dat_{vr}, sifKnj, nazKnj, siflzd, nazlzd\}$

Odredite ključ za relacijsku shemu **POSDBA** tako da ona zadovoljava 1NF.

$K_{\text{POSDBA}} = \{sif_{cln}, invBrPrim, datPos\}$

posudba							
sif _{cln}	invBrPrim	datPos	dat _{vr}	sifKnj	nazKnj	siflzd	nazlzd
123	11234	3.1.2007	N	567	Kiklop	12	A, M
123	21345	3.1.2007	2.2.2007	351	eto	12	A, M
123	19435	29.1.2007	N	459	Bajke	15	BZ
124	19435	2.1.2007	9.1.2007	459	Bajke	15	BZ
124	23414	2.1.2007	N	398	Svila	15	BZ
234	21345	3.2.2007	N	351	eto	12	A, M

Zadatak 3 2NF

clan

sif _{cln}	prez _{cln}	ime _{cln}	pbr	nazMj	adr _{cln}
123	Novak	lasna	10000	Zagreb	nska 6
124	orvat	Krešo	10020	Zagreb	Siget 8
234	rgić	Ana	10000	Zagreb	Krčka 1

AN zadovoljava 2NF

ZA

AN zadovoljava 2NF
 - ZA O

Zadovoljava li **POSDBA** 2NF?

Postoje li neključni atributi koji ovise o dijelu ključa?

Normalizirajte relacijsku shemu **POSDBA** na 2NF.

posudba							
sif _{cln}	invBrPrim	datPos	dat _{vr}	sifKnj	nazKnj	siflzd	nazlzd
123	11234	3.1.2007	N	567	Kiklop	12	A, M
123	21345	3.1.2007	2.2.2007	351	eto	12	A, M
123	19435	29.1.2007	N	459	Bajke	15	BZ
124	19435	2.1.2007	9.1.2007	459	Bajke	15	BZ
124	23414	2.1.2007	N	398	Svila	15	BZ
234	21345	3.2.2007	N	351	eto	12	A, M

Zadatak 3 2NF

posudba ₁			
sif _{cln}	invBrPrim	datPos	dat _{vr}
123	11234	3.1.2007	N
123	21345	3.1.2007	2.2.2007
123	19435	29.1.2007	N
124	19435	2.1.2007	9.1.2007
124	23414	2.1.2007	N
234	21345	3.2.2007	N

2NF
 K

primjerak				
invBrPrim	sifKnj	nazKnj	siflzd	nazlzd
11234	567	Kiklop	12	A, M
21345	351	eto	12	A, M
19435	459	Bajke	15	BZ
23414	398	Svila	15	BZ

2NF
 K

Zadatak 3 3NF

clan						
sif _{ln}	prez _{ln}	ime _{ln}	pbr	nazMj	adr _{ln}	
123	Novak	lasna	10000	Zagreb	nska 6	
124	orvat	Krešo	10020	Zagreb	Siget 8	
234	rgić	Ana	10000	Zagreb	Krčka 1	

Zadovoljava li **AN** 3NF?

Postoje li neključni atributi koji tranzitivno ovise o ključu?

Normalizirajte relacijsku shemu **AN** na 3NF.

clan ₁						mjesto	
sif _{cln}	prez _{cln}	ime _{cln}	pbr	adr _{cln}		pbr	nazMj
123	Novak	lasna	10000	nska 6		10000	Zagreb
124	orvat	Krešo	10020	Siget 8		10020	Zagreb
234	rgić	Ana	10000	Krčka 1			

3NF
 K

3NF
 K

Zadatak 3 3NF

posudba₁

sif _{ln}	invBrPrim	datPos	dat _{ir}
123	11234	3.1.2007	N
123	21345	3.1.2007	2.2.2007
123	19435	29.1.2007	N
124	19435	2.1.2007	9.1.2007
124	23414	2.1.2007	N
234	21345	3.2.2007	N

POSDBA₁ zadovoljava 3NF
- ZA₁ O

primjerak

invBrPrim	sifKnj	nazKnj	siflzd	nazlzd
11234	567	Kiklop	12	A M
21345	351	eto	12	A M
19435	459	Bajke	15	BZ
23414	398	Svila	15	BZ

Zadovoljava li PRIMERAK 3NF?
Postoje li neključni atributi koji tranzitivno ovise o ključu?

Normalizirajte relacijsku shemu PRIMERAK na 3NF.

Zadatak 3 3NF

primjerak₁

invBrPrim	sifKnj
11234	567
21345	351
19435	459
23414	398

3NF
K

knjiga

sifKnj	nazKnj	siflzd	nazlzd
567	Kiklop	12	A M
351	eto	12	A M
459	Bajke	15	BZ
398	Svila	15	BZ

Zadovoljava li KN₁ A 3NF?

Postoje li u relacijskoj shemi KN₁ A atributi u zavisnom dijelu koji su tranzitivno ovisni o ključu?

Normalizirajte relacijsku shemu KN₁ A na 3NF.

knjiga₁

sifKnj	nazKnj	siflzd
567	Kiklop	12
351	eto	12
459	Bajke	15
398	Svila	15

3NF
K

izdavač

siflzd	nazlzd
12	A M
15	BZ

3NF
K

Zadatak 3 Shema baze podataka u 3NF

$AN_1 = \{sif_{ln}, prez_{ln}, ime_{ln}, pbr, adr_{ln}\}$ $K_{AN_1} = \{sif_{ln}\}$

$MES_O = \{pbr, nazMj\}$ $K_{MES_O} = \{pbr\}$

$PRIMERAK_1 = \{invBrPrim, sifKnj\}$ $K_{PRIMERAK_1} = \{invBrPrim\}$

$KN_{A_1} = \{sifKnj, nazKnj, siflzd\}$ $K_{KN_{A_1}} = \{sifKnj\}$

$IZDA_{A_1} = \{siflzd, nazlzd\}$ $K_{IZDA_{A_1}} = \{siflzd\}$

$POSDBA_1 = \{sif_{ln}, invBrPrim, datPos, dat_{ir}\}$
 $K_{POSDBA_1} = \{sif_{ln}, invBrPrim, datPos\}$

Zadatak 4

Zadana je relacijska shema $R = ABDEF$ i na njoj skup funkcijskih zavisnosti:

$F = \{AB \rightarrow DE, B \rightarrow EF, F \rightarrow B\}$.

Domene atributa sadrže samo jednostavne vrijednosti, vrijednost svakog atributa je samo jedna vrijednost iz domene tog atributa.

Odrediti primarni ključ relacijske sheme (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu), te shemu postupno normalizirati na 2NF i 3NF.

Zadatak 4 1NF

$R = ABDEF$

$F = \{AB \rightarrow DE, B \rightarrow EF, F \rightarrow B\}$

▪ **odrediti primarni ključ relacije.**

$AB \rightarrow DE$

$B \rightarrow EF \Rightarrow AB \rightarrow EF$ (A-2: uvećanje)

$\Rightarrow AB \rightarrow DEF$ (P-1: unija)

postoji li skup $\alpha \subset AB$ za kojeg vrijedi $\alpha \rightarrow R$?

NE

$\Rightarrow \alpha \subset AB \mid K_{\alpha} \subset AB \mid R$ je u 1NF

Zadatak 4 - 2NF

$R = ABDEF$

$K_R = AB$

$F = \{AB \rightarrow DE, B \rightarrow EF, F \rightarrow B\}$

Postoje li atributi iz zavisnog dijela koji nisu potpuno funkcijski ovisni o ključu?



▪ $AB \rightarrow EF$ je nepotpuna FZ, jer vrijedi $B \rightarrow EF$ R nije u 2NF

Odredite relacijske sheme kojima treba zamijeniti relacijsku shemu R. Odredite ključeve.

$R_1 = BEF$

$K_{R_1} = B$

R_1 je u 2NF

$R_2 = ABDE$

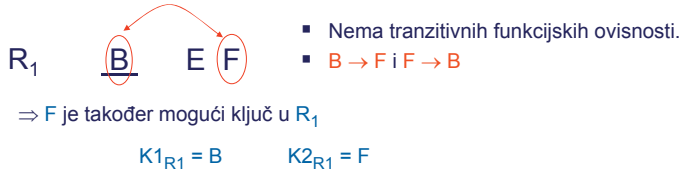
$K_{R_2} = AB$

R_2 je u 2NF

Zadatak 4 - 3N

$R_1 = BEF$ $K_{R_1} = B$
 $R_2 = AB \sqcap D$ $K_{R_2} = AB$
 $F = \sqcap AB \rightarrow \sqcap D, B \rightarrow EF, F \rightarrow B$

- Jesu li R_1 i R_2 u 3NF
- Postoje li u R_1 i R_2 neključni atributi koji tranzitivno ovise o ključu



R_1 i R_2 su u 3NF

Zadatak 5

Zadane su relacijske sheme $\sqcap RED \sqcap A$ i $K \sqcap AR$:

$\sqcap RED \sqcap A = \sqcap mbr \sqcap r, ozn \sqcap r \sqcap r, naz \sqcap r \sqcap r, oznPr, nazPr$
 $K_{\sqcap RED \sqcap A} = \sqcap mbr \sqcap r$
 $K \sqcap AR = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv, op \sqcap rKv, napKv$
 $K_{K \sqcap AR} = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv$

i vrijedi da se za jedan uređaj istog dana može evidentirati više različitih kvarova.

- $mbr \sqcap r$ - matični broj uređaja
- $ozn \sqcap r \sqcap r$ - oznaka vrste uređaja
- $naz \sqcap r \sqcap r$ - naziv vrste uređaja
- $oznPr$ - oznaka proizvođača
- $nazPr$ - naziv proizvođača
- $datKv$ - datum kvara
- $ozn \sqcap rKv$ - oznaka vrste kvara
- $op \sqcap rKv$ - opis vrste kvara
- $napKv$ - napomena uz kvar (napomena uz konkretan kvar na određenom uređaju određenog datuma)

Relacijske sheme $\sqcap RED \sqcap A$ i $K \sqcap AR$ su u 1NF (provjerite!) .
 Normalizirati te relacijske sheme na 2NF i 3NF.

Zadatak 5 - 2N

2NF $\sqcap RED \sqcap A = \sqcap mbr \sqcap r, ozn \sqcap r \sqcap r, naz \sqcap r \sqcap r, oznPr, nazPr$
 $K_{\sqcap RED \sqcap A} = \sqcap mbr \sqcap r$

$\sqcap RED \sqcap A$ zadovoljava 2NF (zašto).

$K \sqcap AR = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv, op \sqcap rKv, napKv$
 $K_{K \sqcap AR} = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv$

Postoje li neključni atributi koji ne ovise o čitavom ključu nego samo o dijelu ključa?
 Normalizirajte relacijsku shemu $K \sqcap AR$ na 2NF.

$\sqcap RS \sqcap AK \sqcap ARA = \sqcap ozn \sqcap rKv, op \sqcap rKv$
 $K_{\sqcap RS \sqcap AK \sqcap ARA} = \sqcap ozn \sqcap rKv$

$K \sqcap AR_1 = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv, napKv$
 $K_{K \sqcap AR_1} = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv$

Zadatak 5 - 3N

Postoje li neključni atributi koji tranzitivno ovise o ključu

$\sqcap RED \sqcap A = \sqcap mbr \sqcap r, ozn \sqcap r \sqcap r, naz \sqcap r \sqcap r, oznPr, nazPr$
 $K_{\sqcap RED \sqcap A} = \sqcap mbr \sqcap r$

Normalizirajte relacijsku shemu $\sqcap RED \sqcap A$ na 3NF.

$\sqcap RS \sqcap A \sqcap RED = \sqcap ozn \sqcap r \sqcap r, naz \sqcap r \sqcap r$ $K_{\sqcap RS \sqcap A \sqcap RED} = \sqcap ozn \sqcap r \sqcap r$

$PROIZ \sqcap OD \sqcap A = \sqcap oznPr, nazPr$ $K_{PROIZ \sqcap OD \sqcap A} = \sqcap oznPr$

$\sqcap RED \sqcap A_1 = \sqcap mbr \sqcap r, ozn \sqcap r \sqcap r, oznPr$ $K_{\sqcap RED \sqcap A_1} = \sqcap mbr \sqcap r$

3NF
 K

Zadatak 5 - 3N

$\sqcap RS \sqcap AK \sqcap ARA = \sqcap ozn \sqcap rKv, op \sqcap rKv$ 3NF OK
 $K_{\sqcap RS \sqcap AK \sqcap ARA} = \sqcap ozn \sqcap rKv$

$K \sqcap AR_1 = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv, napKv$ 3NF OK
 $K_{K \sqcap AR_1} = \sqcap mbr \sqcap r, datKv, ozn \sqcap rKv$

Shema baze podataka u 3NF sastoji se od relacijskih shema:
 $\sqcap RS \sqcap A \sqcap RED$, $PROIZ \sqcap OD \sqcap A$, $\sqcap RED \sqcap A_1$, $\sqcap RS \sqcap AK \sqcap ARA$, $K \sqcap AR_1$

Zadatak 6

Zadane su relacijske sheme $\sqcap INI \sqcap A$ i $PROME$:

$\sqcap INI \sqcap A = \sqcap lin, sifOdr, nazOdr, vrijPol, tr \sqcap oz$ $K_{\sqcap INI \sqcap A} = \sqcap lin$
 $PROME = \sqcap lin, sifPrij, nazPrij, sifAut, tipAut, datPol, brSjed, brKart$
 $K_{PROME} = \sqcap lin, sifPrij, sifAut, datPol$

- lin - broj linije na kojoj se odvija promet
- $sifPrij$ - šifra prijevoznika (poduzeća)
- $nazPrij$ - naziv prijevoznika
- $sifAut$ - šifra autobusa - određuje je prijevoznik
- $tipAut$ - tip autobusa
- $brSjed$ - broj sjedala
- $sifOdr$ - šifra mjesta - odredišta
- $nazOdr$ - naziv mjesta - odredišta
- $datPol$ - datum polaska
- $vrijPol$ - vrijeme polaska
- $tr \sqcap oz$ - trajanje vožnje
- $brKart$ - broj prodanih karata

Relacijske sheme $PROME$ i $\sqcap INI \sqcap A$ su u 1NF (provjeriti!)

Zadatak 6

INI₁A = lin, sifOdr, nazOdr, vrijPol, tr₁oz

K_{INI₁A} = lin

PROME₁ = lin, sifPrij, nazPrij, sifAut, tipAut, datPol, brSjed, brKart

K_{PROME₁} = lin, sifPrij, sifAut, datPol

Normalizirati navedene relacijske sheme na 2NF i 3NF ako vrijedi:

- linija određuje odredište, vrijeme polaska i trajanje vožnje
- istog dana na istoj liniji može prometovati više autobusa (istog ili različitih prijevoznika)
- šifru autobusa određuje prijevoznik (mogu postojati različiti autobusi različitih prijevoznika koji imaju istu šifru)
- autobusi istog tipa imaju jednak broj sjedala

Zadatak 6 2NF

2NF INI₁A = lin, sifOdr, nazOdr, vrijPol, tr₁oz 2NF OK

PROME₁ = lin, sifPrij, nazPrij, sifAut, tipAut, datPol, brSjed, brKart

- šifru autobusa određuje prijevoznik (mogu postojati različiti autobusi različitih prijevoznika koji imaju istu šifru)

sifPrij sifAut → tipAut brSjed

Normalizirajte relacijsku shemu PROME₁ na 2NF.

PRIE₁OZNIK = sifPrij, nazPrij

K_{PRIE₁OZNIK} = sifPrij

A₁OB₁S = sifPrij, sifAut, tipAut, brSjed

K_{A₁OB₁S} = sifPrij, sifAut

PROME₁ = lin, sifPrij, sifAut, datPol, brKart

K_{PROME₁} = lin, sifPrij, sifAut, datPol

Zadatak 6 3NF

INI₁A = lin, sifOdr, nazOdr, vrijPol, tr₁oz

K_{INI₁A} = lin 3NF

Normalizirajte relacijsku shemu INI₁A na 3NF.

ODREDIS₁E = sifOdr, nazOdr

K_{ODREDIS₁E} = sifOdr

INI₁A₁ = lin, sifOdr, vrijPol, tr₁oz

K_{INI₁A₁} = lin

PRIE₁OZNIK = sifPrij, nazPrij

K_{PRIE₁OZNIK} = sifPrij 3NF OK

PROME₁ = lin, sifPrij, sifAut, datPol, brKart

K_{PROME₁} = lin, sifPrij, sifAut, datPol

3NF
OK

Zadatak 6 3NF

A₁OB₁S = sifPrij, sifAut, tipAut, brSjed

3NF

K_{A₁OB₁S} = sifPrij, sifAut

- autobusi istog tipa imaju jednak broj sjedala

Normalizirajte relacijsku shemu A₁OB₁S na 3NF.

IPA₁OB = tipAut, brSjed

K_{IPA₁OB} = tipAut

A₁OB₁S₁ = sifPrij, sifAut, tipAut

K_{A₁OB₁S₁} = sifPrij, sifAut

3NF OK

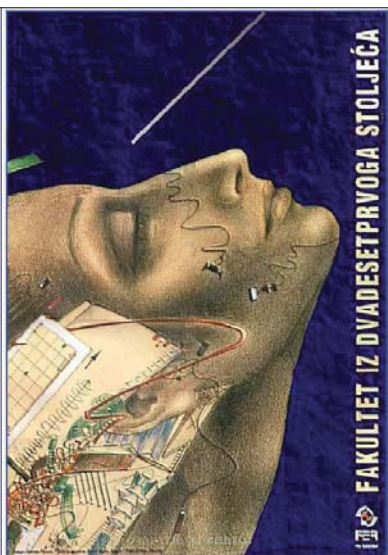
Shema baze podataka u 3NF sastoji se od relacijskih shema:

ODREDIS₁E, INI₁A₁, PRIE₁OZNIK,
PROME₁, IPA₁OB, A₁OB₁S₁

Baze podataka

Predavanja
travanj 2008.

9. Fizička organizacija podataka



Fizička organizacija

- Pojam fizičke organizacije podataka odnosi se na:
 - strukture podataka primijenjene pri pohrani podataka u sekundarnoj memoriji
 - metode pristupa (algoritmi): postupci koji se primjenjuju pri obavljanje operacija nad podacima
- Fizička organizacija podataka ne utječe na rezultate operacija s podacima, ali ima vrlo veliki utjecaj na učinkovitost sustava za upravljanje bazama podataka
 - važna zadaća sustava za upravljanje bazama podataka: obavljati operacije nad velikim količinama podataka na učinkovit način
- SBP skriva od korisnika detalje fizičke organizacije podataka jer za većinu korisnika sustava nisu značajni

Pohrana baze podataka u sekundarnoj memoriji



- **glavna memorija (radni spremnik, RAM)**
 - velike brzina pristupa podacima (10-100 ns), relativno skupa, kapacitet MB
 - neprikladna za pohranu baze podataka jer:
 - ima kapacitet nedovoljan za pohranu baze podataka (previsoka cijena za pohranu velikih količina podataka)
 - nepostojana ($volatile$) memorija: sadržaj memorije se gubi pri gubitku napajanja ili pri pogrešci sustava

Pohrana baze podataka u sekundarnoj memoriji

- karakteristike medija za pohranu podataka uvjetuju da se većina današnjih baza podataka pohranjuje u **sekundarnoj memoriji** (uobičajeno: magnetski diskovi)
- podaci se između sekundarne i primarne memorije prenose u blokovima (tipično 512 B, 1 kB, 2kB, 4kB)

⇒ **dominiraju troškovi** □ (ulazno/izlaznih) **operacija**: vrijeme potrebno za obavljanje □ operacije radi prijenosa bloka podataka između sekundarne i primarne memorije znatno je veće od vremena koje će biti utrošeno za obavljanje operacija nad podacima u primarnoj memoriji

Glavni ciljevi fizičke organizacije

- minimizirati broj  operacija pri pohranu i dohvatu podataka, minimizirati utrošak prostora za pohranu
 - u koji fizički blok pohraniti logički zapis odnosno n-torku
 - koje je dodatne informacije potrebno pohraniti da bi se omogućio učinkovit pristup podacima
- omogućiti različite metode pristupa koje se koriste za pronalaženje fizičke pozicije zapisa (ili fizičke pozicije bloka u kojem se taj zapis nalazi) na temelju vrijednosti ključa pretrage
 - ključ pretrage () ne mora nužno biti primarni ili alternativni ključ. Ključ pretrage može biti bilo koji atribut ili skup atributa relacije (sekundarni ključ).
 - primjenjivost pojedinih metoda pristupa podacima ovisi o primijenjenim strukturama podataka

Strukture podataka i metode pristupa podacima

- Primjena različitih struktura podataka omogućava različite metode pristupa podacima
- Ne postoji najbolja metoda fizičke organizacije, ali dvije se u današnjim sustavima za upravljanje bazama podataka koriste najčešće

- **Neporedana (heap) datoteka**
- Poredana datoteka (*sorted*)
- Raspršena datoteka (*scattered*)
- Indeksno-slijedna organizacija (*indexed sequential*)
- **B-stablo (B-tree)**

Neporedana (*heap*) datoteka

- zapis se upisuje na bilo koje slobodno mjesto u datoteci
- pristup podacima (dohvat podatka sa zadanom vrijednošću ključa pretrage) moguć je isključivo linearnim pretraživanjem



- koristi se za relacije s malim brojem n-torki ili u relacijama čiji se podaci uvijek obrađuju slijedno

Neporedana (*heap*) datoteka

- Dohvat zapisa prema ključu pretrage
 - prema primarnom ili alternativnom ključu
 - u prosjeku je potrebno obaviti $n/2$ ☐ operacija (n predstavlja broj fizičkih blokova u kojima su pohranjeni logički zapisi odnosno n-torke)
 - još gore: u slučaju kada traženi zapis ne postoji, sustav će morati obaviti n ☐ operacija
 - prema ostalim ključevima pretrage ili prema zadanim granicama intervala
 - potrebno je obaviti prijenos n fizičkih blokova

B-Stabla

Literatura:

1. Silberschatz, Korth, Sudarshan: Database System Concepts
2. Garcia-Molina, Elman, Elmasri: Database Systems - The Complete Book

Ovdje će biti opisana varijanta B-stabla koja se naziva **B⁺-stablo**. Opisi ostalih varijanti B-stabala (B⁻-stablo, B⁻-stablo, ...) mogu se pronaći u literaturi.

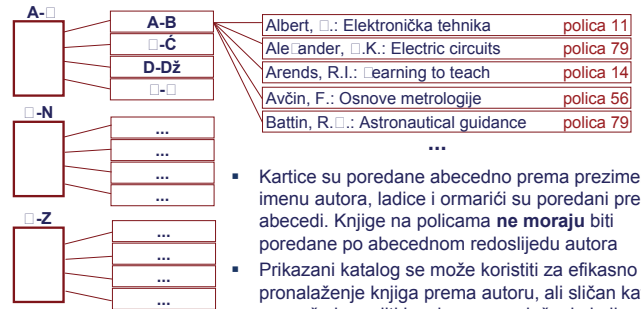
B-stabla

- Idea se temelji na izgradnji indeksnog kazala na više razina: slično kao kod kataloga u papirnatom obliku u knjižnicima (danas se rijetko koriste)

ormarići

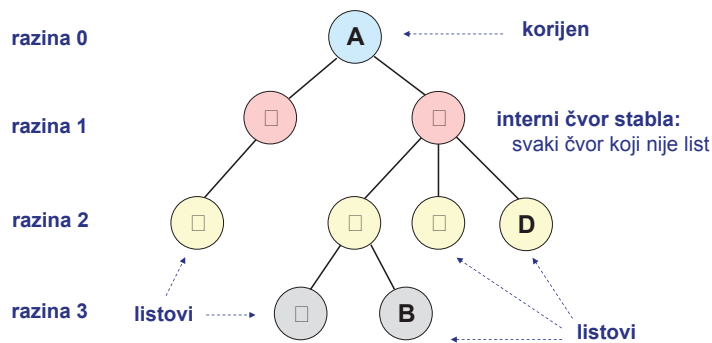
ladice

kartice



- Kartice su poredane abecedno prema prezimenu i imenu autora, ladice i ormarići su poredani prema abecedi. Knjige na policama **ne moraju** biti poredane po abecednom redoslijedu autora
- Prikazani katalog se može koristiti za efikasno pronalaženje knjiga prema autoru, ali sličan katalog se može izgraditi i za brzo pronalaženje knjiga prema naslovu ili prema nekim drugim pojmovima.

Stablo kao struktura podataka

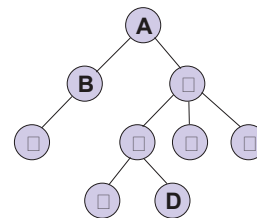


- razina čvora (level):** duljina puta od korijena do čvora
- dubina stabla (depth):** najveća duljina puta od korijena do lista
- red stabla (order):** najveći broj djece koje čvor može imati

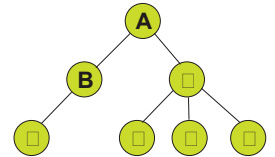
Stablo kao struktura podataka

Stablo je **balansirano (balanced)** ukoliko je duljina puta od korijena do lista jednaka za svaki list u stablu

stablo nije balansirano

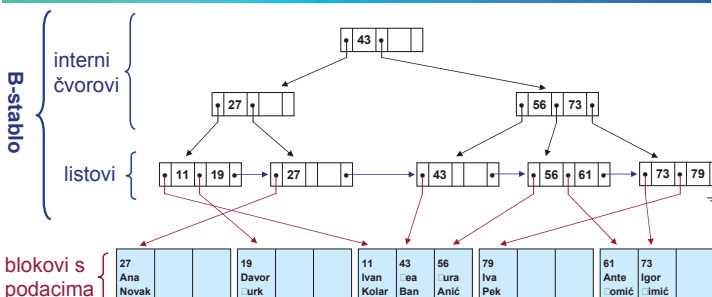


stablo je balansirano



Oznaka **B** u B-stablo znači **balansirano**

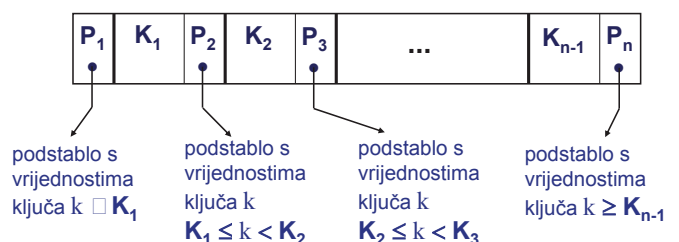
Struktura B⁺-stabla



- Schema: mbr, ime, prezime: B⁺-stablo je izgrađeno za atribut mbr
- Moguće metode pristupa podacima (za bilo koji ključ pretrage):
 - linearnim pretraživanjem (kao kod neporedane datoteke)
 - ako je ključ pretrage mbr, može se koristiti B-stablo

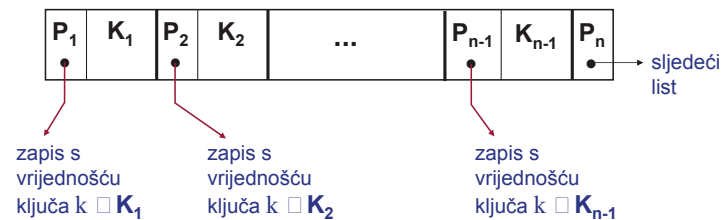
Struktura internog čvora B⁺-stabla

- B⁺-stablu reda **n**, **interni čvor** sadrži:
 - najviše **n** kazaljki
 - najmanje $\lceil n/2 \rceil$ kazaljki → $\lceil a \rceil$ je najmanji cijeli broj $\geq a$
 - ovo ograničenje ne vrijedi za korijen (najmanji broj kazaljki je 2)
 - uz **p** kazaljki u čvoru, broj pripadnih vrijednosti K_i u čvoru je **p-1**
 - K_i je vrijednost ključa

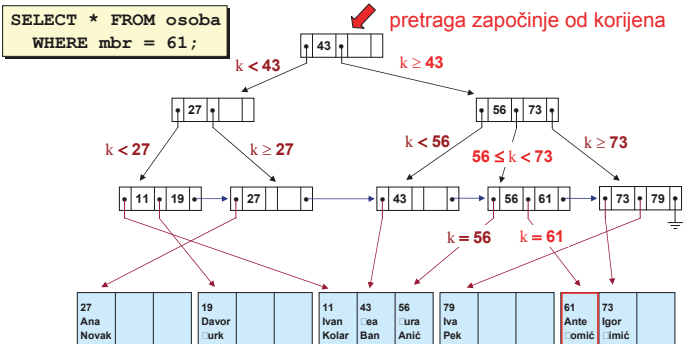


Struktura lista B⁺-stabla

- B⁺-stablu reda n , list sadrži:
 - najviše $n-1$ vrijednosti K_i i pripadnih kazaljki na zapise
 - najmanje $\lceil (n-1)/2 \rceil$ vrijednosti K_i i pripadnih kazaljki na zapise
 - svi listovi sadrže kazaljku na sljedeći list
 - omogućava upite tipa od-do (prema zadanim granicama intervala)



Algoritam za pronalaženje zapisa putem B⁺-stabla

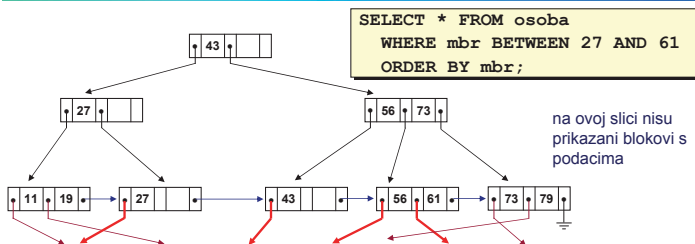


- slijediti odgovarajuću kazaljku do sljedeće razine
- postupak se ponavlja dok se ne dođe do lista u kojem će se naći kazaljka na zapis u bloku s podacima

Algoritam za pronalaženje zapisa putem B⁺-stabla

- algoritam za traženje zapisa s ključem vrijednosti k je rekurzivan
 - cilj je u svakom koraku rekurzije (pretraga i -te razine) pronaći čvor na nižoj, ($i-1$)-voj razini, koji će voditi prema listu u kojem se nalazi ključ čija je vrijednost k
- traženje zapisa započinje od korijena (0-te razine)
- u čvoru i -te razine potrebno je pronaći najveću vrijednost ključa koja je manja ili jednaka traženoj vrijednosti k
 - za prvu kazaljku internog čvora nije navedena vrijednost ključa, pa ona pokriva sve vrijednosti ključeva manje od prve vrijednosti ključa (K_1) navedene u čvoru
- nakon pronalaženja odgovarajuće vrijednosti ključa, slijedi se pripadna kazaljka i time se obavlja pozicioniranje na ($i-1$)-vu razinu
- postupak se ponavlja rekurzivno sve dok se ne dođe do lista. U njemu se mora nalaziti, ukoliko postoji, ključ čija je vrijednost k , te pripadna kazaljka prema traženom zapisu (n -torki)

Dohvat podataka iz intervala i sortiranje

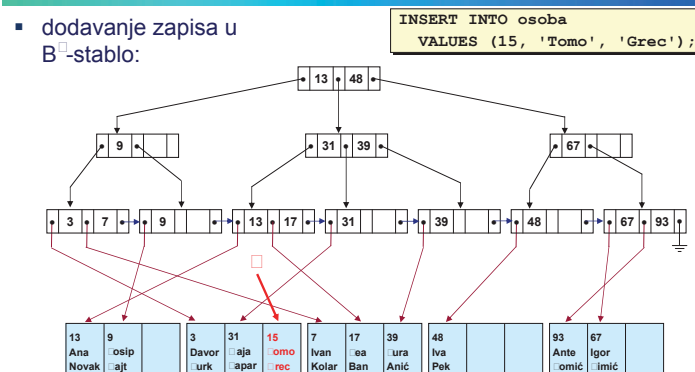


- u listu pronaći kazaljku na zapis s ključem 27
- redom dohvaćati kazaljke i pripadne zapise dok se ne dođe do kazaljke na zapis s ključem 61
 - taj postupak omogućavaju kazaljke među listovima
- dobiveni su svi traženi zapisi, pri tome su poredani prema mbr
- Ako se obavlja `SELECT * FROM osoba WHERE mbr BETWEEN 27 AND 61 ORDER BY mbr;`
 - pronađene zapise jednostavno ispisati obrnutim redoslijedom

Dodavanje i brisanje zapisa

- nakon dodavanja ili brisanja zapisa u bloku s podacima, mijenja se i sadržaj B⁺-stabla
 - koriste se algoritmi za dodavanje i brisanje zapisa u B⁺-stablu
 - algoritmi osiguravaju ispravnu popunjenost internih čvorova i listova B⁺-stabla
 - pri tome se može dogoditi da stablo promijeni dubinu
- operacija izmjene
 - ukoliko se ne mijenjaju vrijednosti atributa za koje je izgrađeno B⁺-stablo, u B⁺-stablu nisu potrebne izmjene
 - ukoliko se mijenjaju vrijednosti atributa za koje je izgrađeno B⁺-stablo, u B⁺-stablu se obavlja algoritam za brisanje zapisa i algoritam za dodavanje zapisa
- važno dobro svojstvo algoritama B-stabla: dubina stabla se automatski prilagođava broju zapisa - čvorovi stabla (osim korijena) su uvijek barem 50% popunjeni

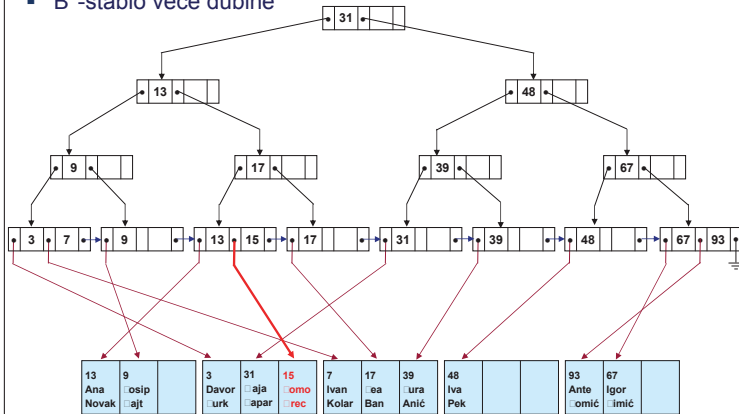
Primjer dodavanja zapisa



- rezultat dodavanja zapisa u B⁺-stablo je na sljedećoj slici:

Rezultat dodavanja zapisa

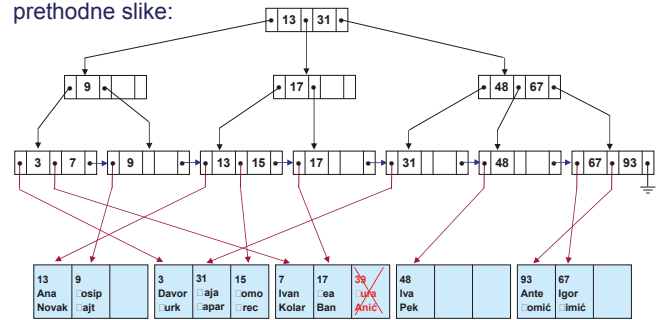
- B⁺-stablo veće dubine



Primjer brisanja zapisa

- obavljanjem operacije nad B⁺-stablom s prethodne slike:

DELETE FROM osoba WHERE mbr = 39;



- B⁺-stablo manje dubine

Činkovitost operacije pretrage u B⁺-stablu

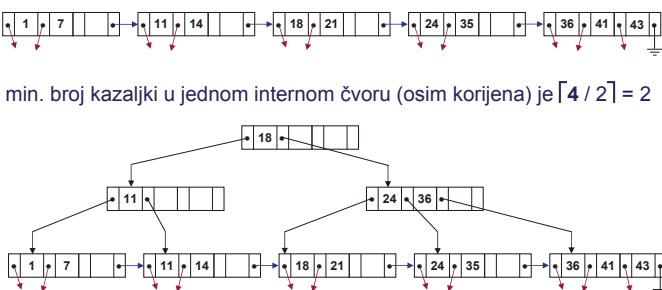
- pretpostavka: stablo reda n sadrži kazaljke na m zapisa podataka
- n se odabire tako da se sadržaj čvora može smjestiti u jedan fizički blok
⇒ za dohvat **jednog** čvora potrebna je **jedna** operacija
- broj operacija u stablu pri traženju zapisa ovisi o broju razina u stablu jer se pri dohvat zapisa mora obaviti po jedna operacija za svaki čvor B-stabla na putu od čvora do lista
- B-stablo ima najveći broj razina onda kada su čvorovi najmanje popunjeni
⇒ moguće je odrediti koliko će operacija biti potrebno obaviti u najlošijem slučaju

Činkovitost operacije pretrage u B⁺-stablu

- Primjer:** za broj n -torki $m = 1\,000\,000$, za **red** stabla $n = 70$, ukupni broj razina (uključujući i razinu korijena) u najlošijem slučaju je 4:
 1. \square točno 1 čvor, najmanje 2 kazaljke
 2. $\square \square$ najmanje 2 čvora, najmanje 70 kazaljki
 3. $\square \square \dots \square \square$ najmanje 70 čvorova, najmanje 2 450 kazaljki
 4. $\square \square \square \dots \square \square \square$ najmanje 2 450 čvorova, najmanje 85 750 kazaljki
 5. $\square \square \square \square \dots \square \square \square \square$ najmanje 85 750 čvorova, najmanje 3 001 250 kazaljki
- B⁺-stablo koje bi imalo ukupno 5 razina, moralo bi imati **najmanje** 3 001 250 kazaljki na zapise. To znači da B-stablo reda 70 čije kazaljke u listovima pokazuju na 1 000 000 n -torki može imati najviše 4 razine.
- ⇒ Za dohvat zapisa prema vrijednosti ključa potrebno je najviše 5 operacija (4 operacije za dohvat lista u kojem se nalazi kazaljka na zapis i 1 operacija za dohvat bloka s podacima)

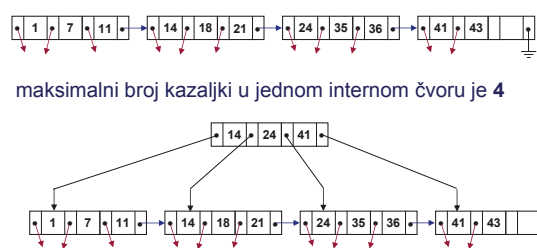
Zadatak 1.

- Relacija $slu\check{z}$ ($\square \square r, pr\check{z}, i\check{z}$) sadrži n -torke sa sljedećim vrijednostima atributa $\square \square r$: 1, 7, 11, 14, 18, 21, 24, 35, 36, 41, 43. Nacrtati B⁺-stablo reda 4 za atribut $\square \square r$ tako da popunjenost stabla bude minimalna.
- min. broj kazaljki na zapise (n -torke) u jednom listu je $\lceil (4 - 1) / 2 \rceil = 2$
- min. broj kazaljki u jednom internom čvoru (osim korijena) je $\lceil 4 / 2 \rceil = 2$



Zadatak 2.

- Relacija $slu\check{z}$ ($\square \square r, pr\check{z}, i\check{z}$) sadrži n -torke sa sljedećim vrijednostima atributa $\square \square r$: 1, 7, 11, 14, 18, 21, 24, 35, 36, 41, 43. Nacrtati B⁺-stablo reda 4 za atribut $\square \square r$ tako da popunjenost čvorova u stablu bude maksimalna.
- maksimalni broj kazaljki na zapise (n -torke) u jednom listu je $4 - 1 = 3$
- maksimalni broj kazaljki u jednom internom čvoru je 4



Zadatak 3.

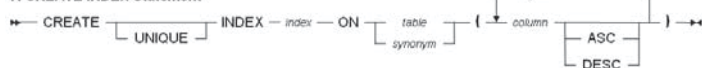
- Koliko n-torki sadrži relacija ako je nad njom izgrađeno B⁺-stablo reda 101, s ukupno 5 razina, s minimalno dopuštenom popunjenošću **svih** čvorova
- min. broj kazaljki u jednom listu je $\lceil (101 - 1) / 2 \rceil = 50$
- min. broj kazaljki u jednom internom čvoru (osim korijena) je $\lceil 101 / 2 \rceil = 51$
- min. broj kazaljki u korijenu je 2
- relacija sadrži $2 \cdot 51 \cdot 51 \cdot 51 \cdot 50 \approx 1.33 \cdot 10^7$ n-torki
- ZAKLJUČAK:** ako je B-stablo reda 101, do svake n-torke u relaciji koja sadrži $\approx 1.33 \cdot 10^7$ n-torki može se pristupiti, u najlošijem slučaju, korištenjem tek 6 \square operacija (5 \square za dohvat lista, 1 \square za dohvat fizičkog bloka u kojem se nalazi n-torka)

Zadatak 4.

- Koliko n-torki sadrži relacija ako je nad njom izgrađeno B⁺-stablo reda 101, s ukupno 5 razina, s maksimalno popunjenim **svim** čvorovima
- max broj kazaljki u jednom listu je $101 - 1 = 100$
- max broj kazaljki u internom čvoru je **101**
- relacija sadrži $101 \cdot 101 \cdot 101 \cdot 101 \cdot 100 \approx 1.04 \cdot 10^{10}$ n-torki
- ZAKLJUČAK:** ako je B-stablo reda 101, u najboljem slučaju, korištenjem tek 6 \square operacija može se dohvatiti blok s n-torkom koja se nalazi u relaciji koja sadrži čak $\approx 1.04 \cdot 10^{10}$ n-torki

5. Indeksi

7. CREATE INDEX Statement



- Obavljanjem naredbe za kreiranje indeksa nad relacijom, nad blokovima s podacima relacije formira se struktura B-stabla

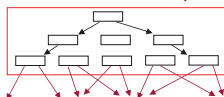
1

```
CREATE TABLE osoba (
  mbr INTEGER
, ime NCHAR(20)
, prez NCHAR(20));
INSERT INTO ...;
```

2

```
CREATE INDEX osoba_prez
ON osoba (prez);
```

B-stablo za osoba.prez



7	17	39	48		93	67
Ivan	ea	ura	Iva		Ante	Igor
Kolar	Ban	Anić	Pek		omić	ilić

5. Indeksi

- Kreiranjem indeksa uz navođenje rezervirane riječi **UNIQUE** osigurava se jedinstvenost vrijednosti navedenog atributa

```
CREATE UNIQUE INDEX osoba_mbr
ON osoba (mbr);
```

B-stablo za osoba.mbr



7	17	39	48		93	67
Ivan	ea	ura	Iva		Ante	Igor
Kolar	Ban	Anić	Pek		omić	ilić

- ukoliko se indeks pokušava kreirati nad relacijom u kojoj već postoje duplikati vrijednosti atributa mbr, sustav će odbiti kreirati indeks i dojaviti pogrešku
- pokuša li se nakon kreiranja ovog indeksa unijeti n-torka s vrijednošću atributa mbr koja već postoji u nekoj n-torci, sustav će odbiti operaciju i dojaviti pogrešku

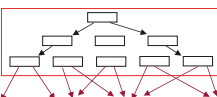
- ništavanje indeksa - primjer:

```
DROP INDEX osoba_mbr;
```

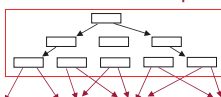
6. Više indeksa nad istom relacijom

- nad istom relacijom može se izgraditi više indeksa

B-stablo za osoba.mbr



B-stablo za osoba.prez



11	43	56	79		61	73
Ivan	ea	ura	Iva		Ante	Igor
Kolar	Ban	Anić	Pek		omić	ilić

- B-stabla (indeksi) prikazani u primjeru omogućuju efikasno obavljanje upita s uvjetima ($=$, $<$, $<=$, $>$, $>=$, $BETWEEN$) i efikasno sortiranje (ASC , $DESC$)
 - za atribut mbr
 - za atribut prez
- prema uvjetima koji sadrže atribut ime, podacima se može pristupiti jedino linearnom pretragom svih blokova

6. Više indeksa nad istom relacijom

- Ako indeksi omogućuju efikasan pristup do n-torki, znači li to da bi indekse trebalo kreirati za svaki atribut u relaciji?

Ne

Zašto?

- indeksi zauzimaju prostor
- operacija unosa ili brisanja n-torke
 - uvijek rezultira promjenama (manjim ili većim) B-stabla
 - npr. ako je nad relacijom izgrađeno 10 različitih indeksa, unosom jedne n-torke u blokove s podacima morat će se unijeti zapisi i u 10 različitih indeksa
- operacija izmjene n-torke
 - izmjena vrijednosti atributa A jedne n-torke rezultirat će brisanjem i dodavanjem zapisa u svim B-stablama za indekse u kojima se koristi atribut A

Za koje attribute treba kreirati indeks

- za attribute koji se često koriste za postavljanje uvjeta selekcije (zašto)
- za attribute prema kojima se obavlja spajanje relacija (zašto)
 - primarni i alternativni ključevi relacije
 - strani ključevi
- za attribute prema kojima se često obavlja sortiranje ili grupiranje (zašto)

Za koje attribute treba kreirati indeks

Primjer:

```
CREATE TABLE stud (  
  mbr INTEGER  
  , ime NCHAR(20)  
  , prez NCHAR(20));
```

```
SELECT * FROM stud  
WHERE mbr = 12345;
```

```
SELECT * FROM stud  
WHERE prez > 'Kolar'  
ORDER BY prez;
```

⇒ Kreirati indeks za mbr i indeks za prez

- to se nakon kreiranja navedenih indeksa dešava pri obavljanju:

```
UPDATE stud SET prez = UPPER(prez)  
WHERE ime = 'Ivan';
```

- n-torke se pronalaze linearnom pretragom (loše), a zbog izmjene vrijednosti atributa prez mora se izmijeniti sadržaj B-stabla za indeks nad atributom prez (loše)

```
UPDATE stud SET ime = UPPER(ime)  
WHERE prez = 'Horvat';
```

- n-torke se pronalaze pomoću B-stabla (dobro), nad atributom ime nije izgrađen indeks, stoga ne postoji B-stablo čiji se sadržaj mora izmijeniti (dobro)

Za koje attribute ne treba kreirati indeks

- ako vrijednosti atributa imaju relativno mali broj različitih vrijednosti
 - npr. atribut spolOsobe s dozvoljenim vrijednostima M, Ž
- ako relaciji predstoji velik broj upisa, izmjena ili brisanja n-torki. Preporuča se u takvim slučajevima postojeće indekse izbrisati, te ih ponovo izgraditi tek nakon obavljenih promjena nad podacima
- ako relacija sadrži relativno mali broj n-torki (sve n-torke su pohranjene u nekoliko blokova). U takvim slučajevima B-stablo ne pridonosi efikasnosti pretrage
 - npr. relacija zupaniya

Složeni indeksi

```
CREATE TABLE stud (  
  mbr INTEGER  
  , ime NCHAR(20)  
  , prez NCHAR(20));
```

```
CREATE INDEX osoba_ime ON osoba (ime);  
CREATE INDEX osoba_prez ON osoba (prez);
```

- efikasno se obavljaju upiti oblika:

```
SELECT * FROM stud  
WHERE prez = 'Kolar';
```

```
SELECT * FROM stud  
WHERE ime = 'Ivan';
```

- upit oblika:

```
SELECT * FROM stud  
WHERE prez = 'Kolar'  
AND ime = 'Ivan';
```

- pomoću indeksa nad atributom prez dohvatit će se n-torke studenata čije je prezime Horvat ali će se u dobivenom skupu n-torki linearnom pretragom morati pronaći oni čije je ime Ivan (ili indeksom po imenu, a onda linearno po prezimenu)

Složeni indeksi

- Prethodni upit se efikasnije obavlja ako se umjesto posebnih indeksa za attribute ime i prezime, kreira složeni indeks:

```
CREATE INDEX osoba_prez_ime ON osoba (prez, ime);
```

- problem: ovaj indeks se koristi za upite oblika:

```
SELECT * FROM stud  
WHERE prez = 'Kolar'  
AND ime = 'Ivan';
```

```
SELECT * FROM stud  
WHERE ime = 'Ivan'  
AND prez = 'Kolar';
```

- također i za upite oblika:

```
SELECT * FROM stud  
WHERE prez = 'Kolar';
```

- ali se ne može koristiti za upite oblika:

```
SELECT * FROM stud  
WHERE ime = 'Ivan';
```

Složeni indeksi

- Kako upotreba složenih indeksa utječe na sortiranje:

```
CREATE INDEX osoba_prez_ime1 ON osoba (prez, ime);
```

- indeks osoba_prez_ime1 se efikasno koristi za sortiranje oblika:

```
SELECT * FROM stud  
ORDER BY prez, ime;
```

```
SELECT * FROM stud  
ORDER BY prez DESC, ime DESC;
```

- ali ne i za:

```
SELECT * FROM stud  
ORDER BY prez DESC, ime;
```

- ako se kreira indeks:

```
CREATE INDEX osoba_prez_ime2 ON osoba (prez DESC, ime);
```

- indeks osoba_prez_ime2 se efikasno koristi za sortiranje oblika:

```
SELECT * FROM stud  
ORDER BY prez DESC, ime;
```

```
SELECT * FROM stud  
ORDER BY prez, ime DESC;
```

Složeni indeksi

- Ako je nad relacijom r ($AB \dots D$) kreiran složeni indeks r_{abc1}

```
CREATE INDEX r_abc1 ON r (A, B, C);
```

tada sljedeće indekse nije potrebno kreirati:

```
CREATE INDEX r_abc2 ON r (A DESC, B DESC, C DESC);
CREATE INDEX r_a1 ON r (A);
CREATE INDEX r_a2 ON r (A DESC);
CREATE INDEX r_ab1 ON r (A, B);
CREATE INDEX r_ab2 ON r (A DESC, B DESC);
```

- Indekse r_{abc2} , r_{a1} , r_{a2} , r_{ab1} i r_{ab2} ne treba kreirati jer $S \dots BP$ može koristiti indeks r_{abc1} u svim slučajevima u kojima bi se koristili indeksi r_{abc2} , r_{a1} , r_{a2} , r_{ab1} i r_{ab2} .

Zadatak 5. zadana je relacija stud (mbr ime prez postBr)

- Za relaciju stud kreirati najmanji mogući broj indeksa koji će omogućiti efikasno obavljanje (pomoću B-stabla) svih navedenih upita:

- SELEKT FROM stud ORDER BY ime DESC, prez
- SELEKT FROM stud ORDER BY ime DESC, prez DESC
- SELEKT FROM stud ORDER BY ime, prez, pbrStan
- SELEKT FROM stud WHERE prez = 'Novak' AND ime = 'Ivo'
- SELEKT FROM stud WHERE pbrStan > 51000 ORDER BY pbrStan DESC

- (ime DESC, prez)
- (ime DESC, prez DESC)
- (ime, prez, pbrStan) - ali sada više nije potreban indeks pod 2.
- može se koristiti indeks pod 3.
- (pbrStan)

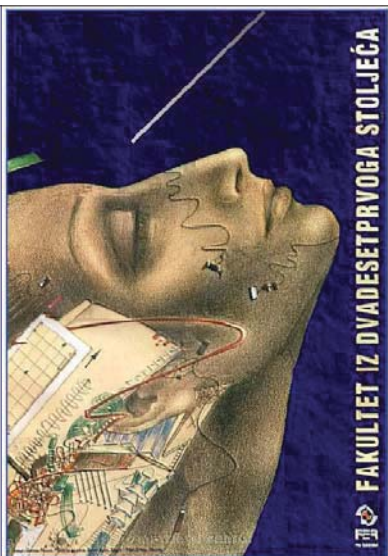
Konačno rješenje - kreirati indekse za: (ime DESC, prez)
(ime, prez, pbrStan)
(pbrStan)

Sve naredbe za kreiranje indeksa napisati za vježbu!

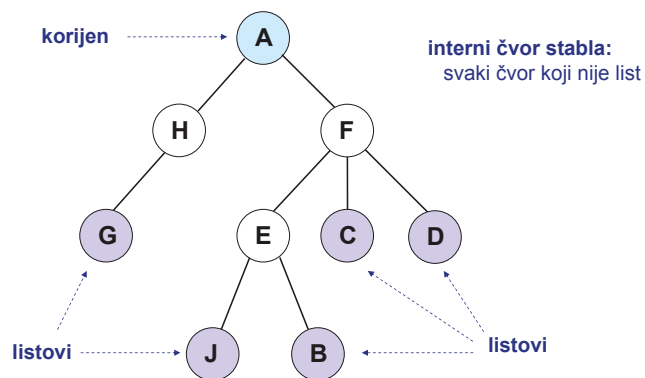
Baze podataka

Dopunski materijali
(za one koji žele znati više)

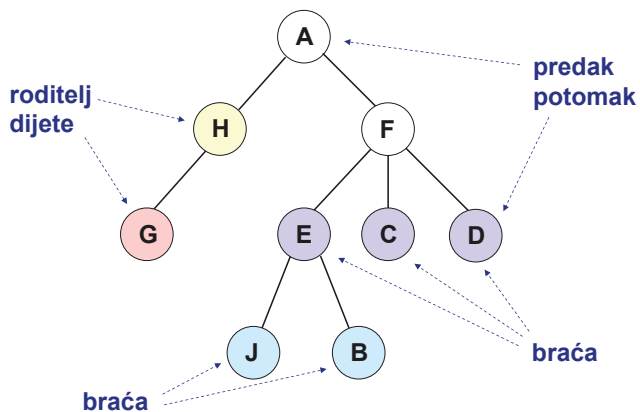
B-stabla



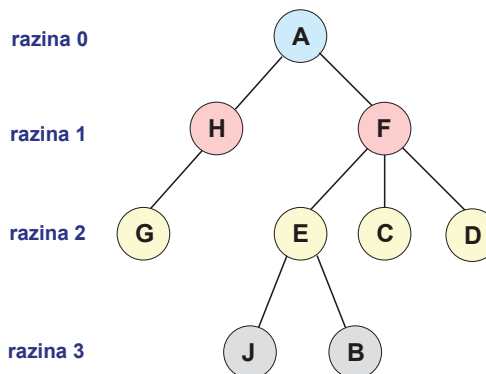
1. Stablo kao struktura podataka



Stablo kao struktura podataka



Stablo kao struktura podataka



razina čvora (level): duljina puta od korijena do čvora
dubina stabla (depth): najveća duljina puta od korijena do lista
red stabla (order): najveći broj djece koje čvor može imati

3 Algoritmi za B-stablo

Algoritam za pretraživanje B-stabla

Algoritam za traženje zapisa s ključem vrijednosti k je rekurzivan

- cilj je u svakom koraku rekurzije pretraživanja i te razine pronaći čvor na nižoj razini koji će voditi prema listu u kojem se nalazi ključ čija je vrijednost k

traženje zapisa započinje od korijena iste razine

u čvoru iste razine potrebno je pronaći najveću vrijednost ključa koja je manja ili jednaka traženoj vrijednosti k

- za prvu kazaljku internog čvora nije navedena vrijednost ključa pa ona pokriva sve vrijednosti ključeva manje od prve vrijednosti ključa navedene u čvoru

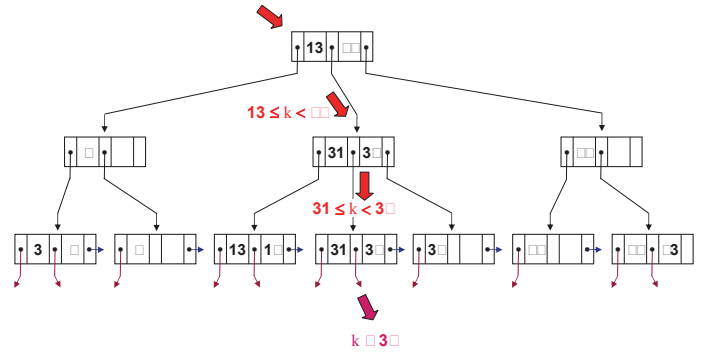
nakon pronalaska odgovarajuće vrijednosti ključa slijedi se pripadna kazaljka i time se obavlja pozicioniranje na listu

postupak se ponavlja rekurzivno sve dok se ne dođe do lista njemu se mora nalaziti ukoliko postoji ključ čija je vrijednost k te pripadna kazaljka prema traženom zapisu u datoteci

□□

Algoritam za pretraživanje B-stabla

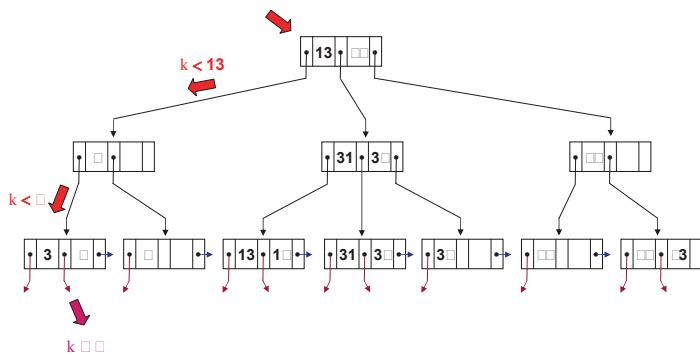
traži se zapis s ključem 3



□□

Algoritam za pretraživanje B-stabla

traži se zapis s ključem k



□3

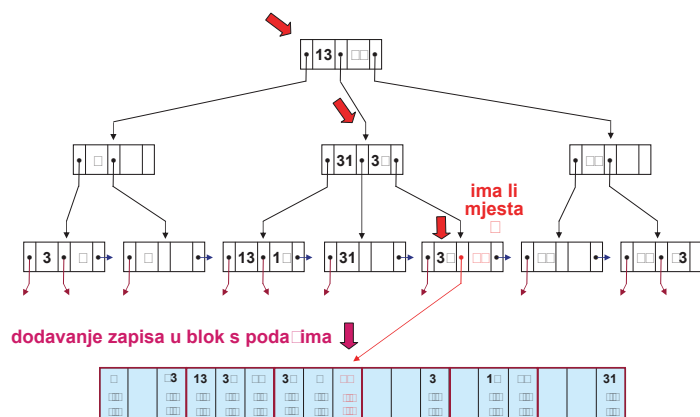
Algoritam za dodavanje zapisa u B-stablo

- zapis (podaci) se upisuje u jedan od slobodnih blokova s podacima
- obavlja se algoritam za pronalazjenje lista u koji pripada vrijednost ključa k
- ako čvor nije popunjen u čvor se dodaje zapis s ključem k i kazaljkom na pripadni zapis u datoteci u očuvanje poretka vrijednosti ključeva
 - nova vrijednost nikad nije prva u čvoru osim u slučaju krajnje lijevog čvora
- ako je čvor popunjen
 - stvara se novi čvor i zapisi među njima se podijele pri čemu svakom od čvorova pripadne polovica zapisa
 - budući da je dodan novi čvor u nadređeni čvor potrebno je dodati zapis s kazaljkom i najmanjom vrijednošću ključa u novom čvoru
 - za dodavanje novog zapisa u nadređeni čvor koristi se ista procedura kao za dodavanje zapisa u čvor na nižoj razini
 - postupak je rekurzivan i mora se obaviti za svaku nadređenu razinu sve dok se ne dođe do korijena ili se na nekoj od razina nađe dovoljno mjesta za upis vrijednosti ključa i kazaljke na dodavanje novih čvorova
 - ako se dođe do korijena može se desiti da u korijenu nema mjesta za novi zapis tada se dodaje novi čvor i formira se novi korijen na višoj razini

□4

Algoritam za dodavanje zapisa u B-stablo

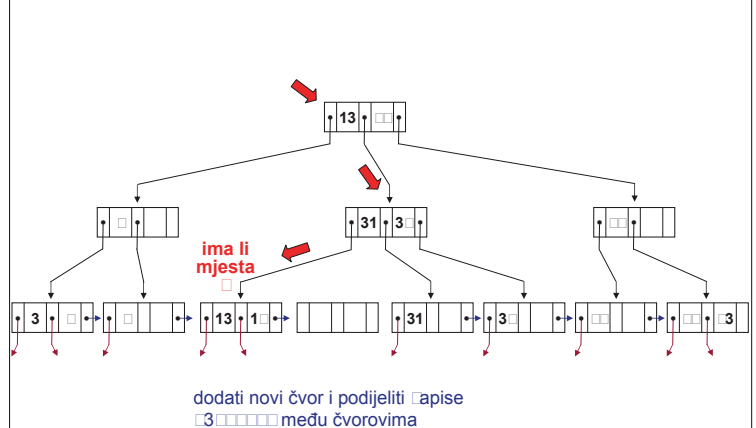
dodavanje zapisa s ključem k



□□

Algoritam za dodavanje zapisa u B-stablo

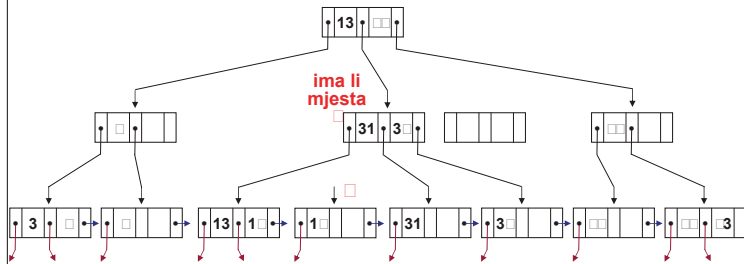
dodavanje zapisa s ključem 1



□□

Algoritam za dodavanje zapisa u B-stablo

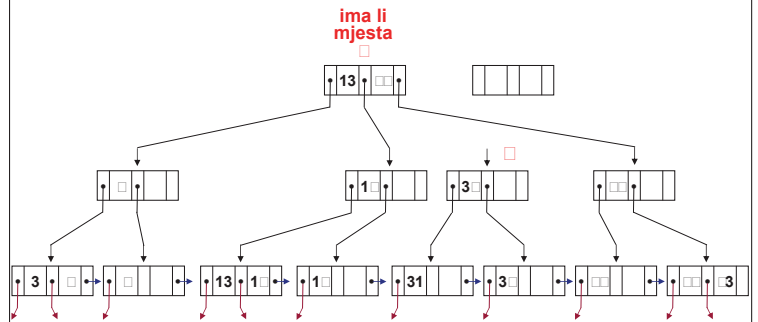
dodavanje zapisa s ključem 1



dodati novi čvor i podijeliti zapise
3 3 3 3 3 među čvorovima

Algoritam za dodavanje zapisa u B-stablo

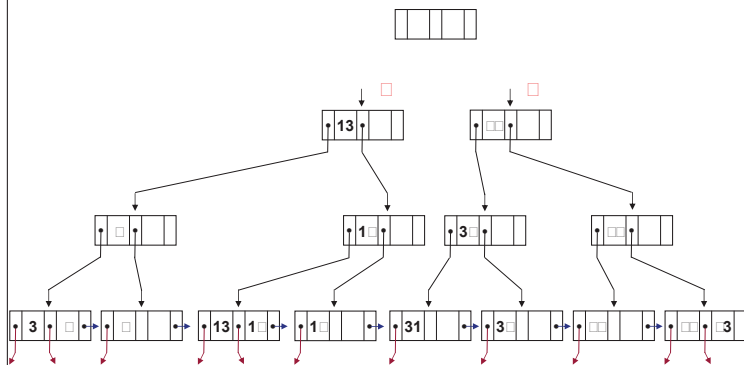
dodavanje zapisa s ključem 1



dodati novi čvor i podijeliti zapise
3 3 3 3 4 među čvorovima

Algoritam za dodavanje zapisa u B-stablo

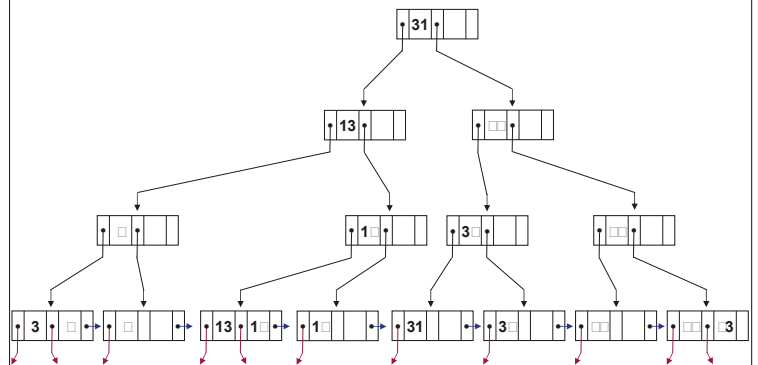
dodavanje zapisa s ključem 1



dodati novi korijen i upisati zapise 3 3 3

Algoritam za dodavanje zapisa u B-stablo

dodavanje zapisa s ključem 1



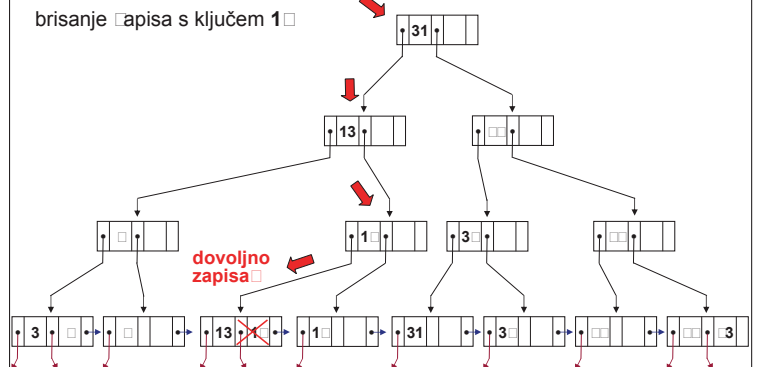
rezultat je balansirano stablo veće dubine

Algoritam za brisanje zapisa iz B-stabla

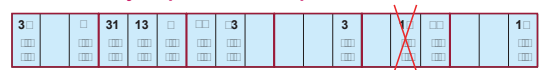
- obavlja se algoritam za pronalaženje lista u kojem se nalazi ključ k
- zapis se briše iz bloka s podacima vrijednost ključa i kataljka iz čvora
- ako čvor sadrži dovoljan broj zapisa
 - ukoliko je potrebno onda kada se iz lista obriše prvi zapis osim u krajnje lijevom listu mijenja se vrijednost ključa u nekom od predaka
- ako čvor nakon brisanja nema dovoljan broj zapisa traži se čvor brat koji se nalazi neposredno s lijeva ili s desna čvoru i ima više od dovoljno broja zapisa ako se takav ne pronađe traži se bilo koji čvor brat koji se nalazi neposredno s lijeva ili s desna čvoru
 - ako odabrani brat ima više od dovoljno broja zapisa tada se zapisi podijele između čvorova i čuvaju se redoslijed zapisa
 - prethodni čvorovi obavlja se potrebne promjene vrijednosti ključeva
 - inače odabrani brat ima upravo dovoljan broj zapisa čvorovi se spajaju u jedan čvor nadređenom čvoru briše se zapis i eventualno mijenja vrijednost ključa u nekom od predaka brisanje zapisa u nadređenom čvoru svodi se na rekursivno izvođenje procedure za brisanje ako se putem prema korijenu dođe u situaciju da treba spojiti jedina dva čvora djeteta korijena tada se oni spajaju postaju novi korijen stabla a stari se korijen briše

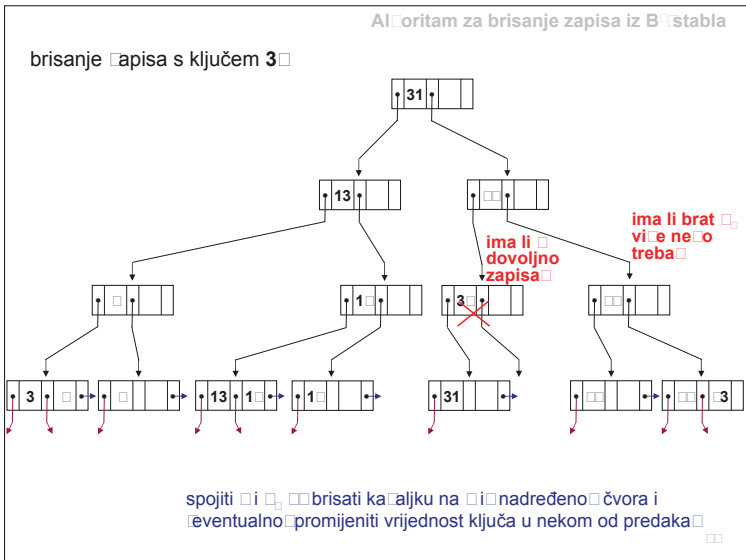
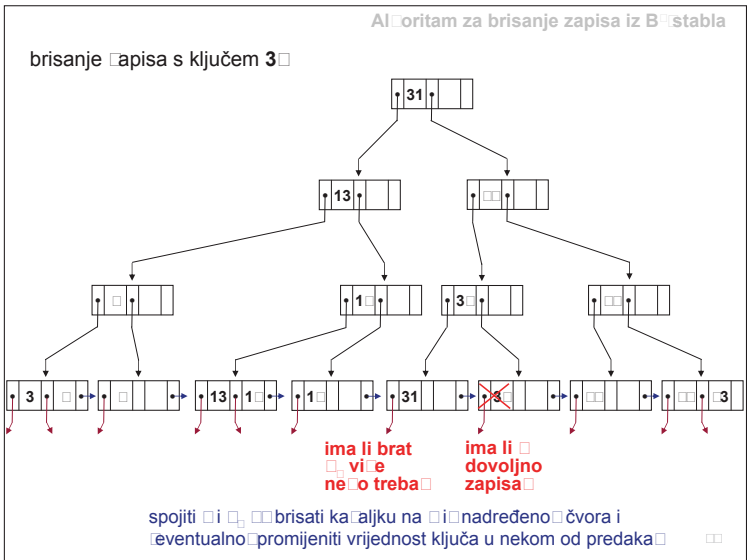
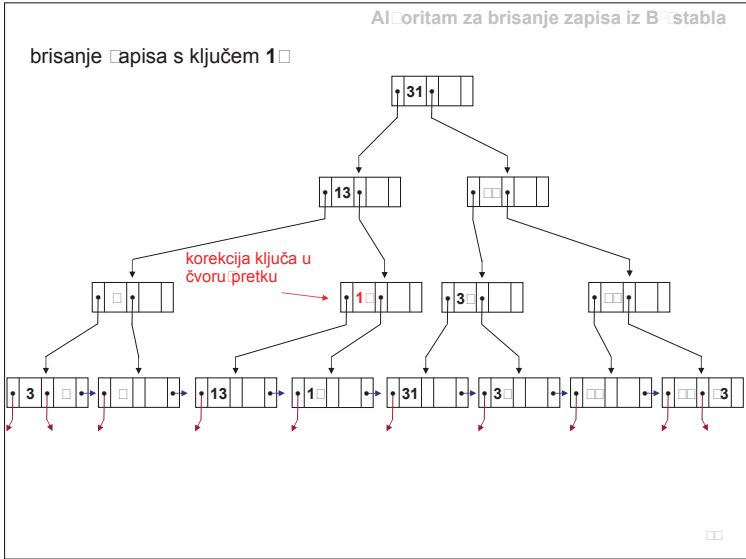
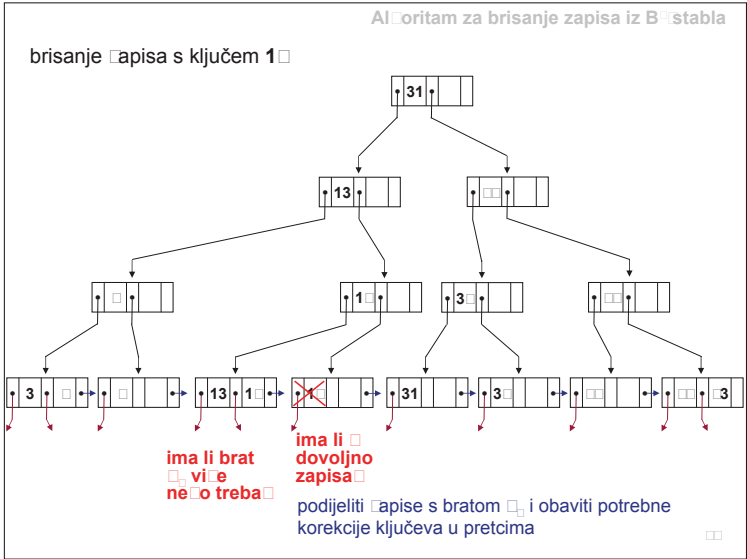
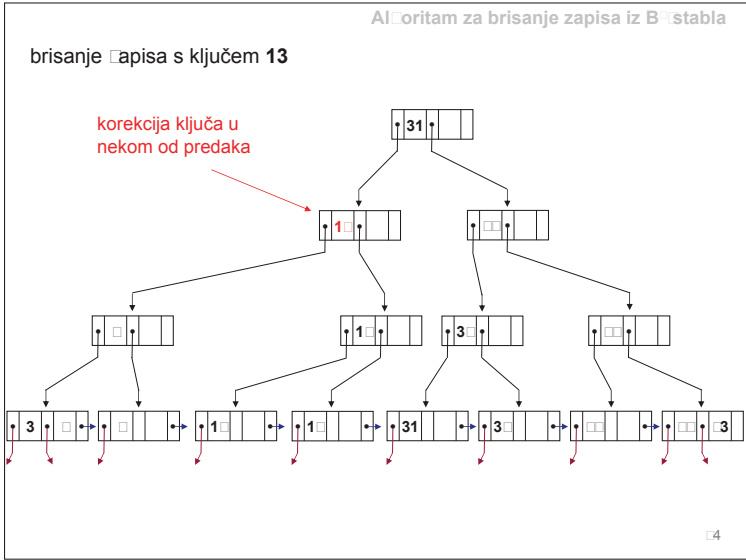
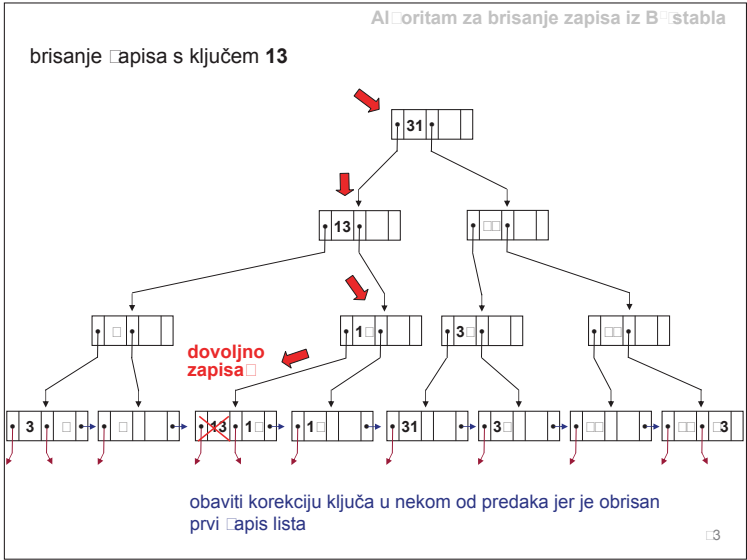
Algoritam za brisanje zapisa iz B-stabla

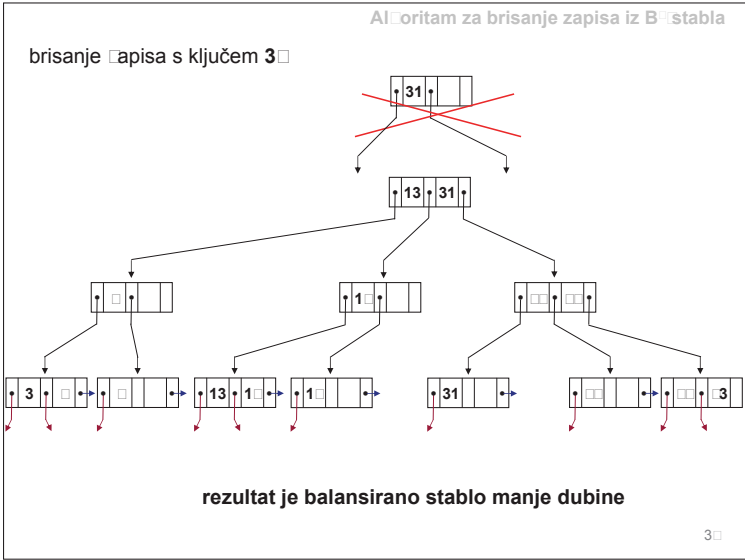
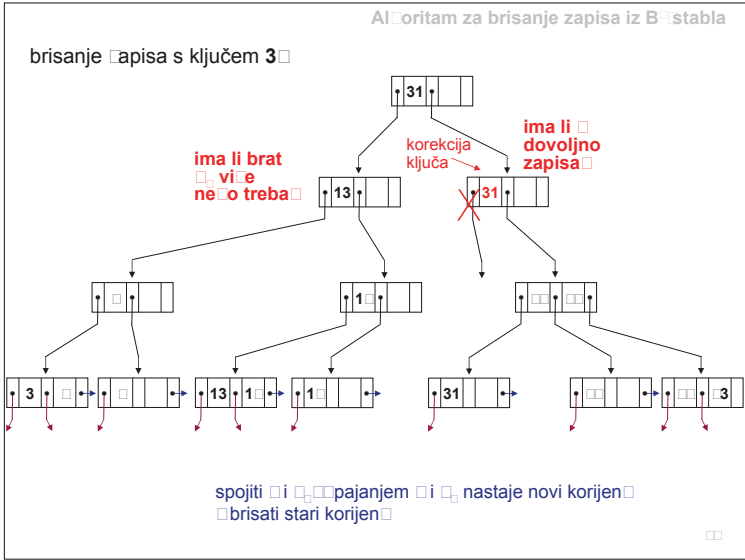
brisanje zapisa s ključem 1



brisanje zapisa iz bloka s podacima







Činkovitost operacije pretrage u B-stablu

- broj operacija u stablu pri traženju zapisa ovisi o broju razina u stablu
- pretpostavka: stablo reda n sadrži m zapisa podataka
- stablo će imati najveći broj razina ako su čvorovi najmanje popunjeni
 - najmanja popunjenost korijena je 2
 - najmanja popunjenost internog čvora $\lceil \frac{n}{2} \rceil$
 - najmanja popunjenost lista $\lceil \frac{n-1}{2} \rceil \approx \lceil \frac{n}{2} \rceil$ ako dovoljno veliki n
- u korijenu razina ima čvor i najmanje 2 kazaljke
- na razini ima najmanje čvora i zato najmanje $2 \lceil \frac{n}{2} \rceil$ kazaljki
- u čvorovima 3 razine ima najmanje $2 \lceil \frac{n}{2} \rceil \lceil \frac{n}{2} \rceil$ kazaljki
- u čvorovima i te razine ima najmanje $2 \lceil \frac{n}{2} \rceil^{i+1}$ kazaljki
- broj zapisa u podatkovnim blokovima stabla koje ima d razina vrijedi
 - $m \geq 2 \lceil \frac{n}{2} \rceil^{d+1}$
- otada slijedi
 - $d \leq \log_{\lceil \frac{n}{2} \rceil} (m/2) + 1$

Činkovitost operacije pretrage u B-stablu

$d \leq \log_{\lceil \frac{n}{2} \rceil} (m/2) + 1$

Primjer: Za $m = 1000000$ i $n = 100$ ukupni broj razina uključujući i razinu korijena u najgorem slučaju je 4

Razina	Broj čvorova	Broj kazaljki
0	1	2
1	$\lceil \frac{1000000}{2} \rceil = 500000$	$2 \lceil \frac{100}{2} \rceil = 100$
2	$2 \lceil \frac{500000}{2} \rceil = 1000000$	$2 \lceil \frac{100}{2} \rceil \lceil \frac{100}{2} \rceil = 10000$
3	$2 \lceil \frac{1000000}{2} \rceil = 1000000$	$2 \lceil \frac{100}{2} \rceil^{i+1} = 1000000$

stablo koje bi imalo ukupno razina moralo bi imati najmanje 3 kazaljke na zapise

Bezbednost podataka

predavanja
travanj 2020.

10. Bezbednost baze podataka

Bezbednost baze podataka

- pojam integriteta baze podataka odnosi se na konzistentnost i ispravnost podataka sadržanih u bazi podataka
- integritet baze podataka može biti narušen zbog
 - slučajne pogreške korisnika kod unosa ili izmjene podataka
 - slučajne pogreške programera ili sustava
- integritetska ograničenja osiguravaju da izmjene podataka koje obavljaju autorizirani korisnici ne rezultiraju narušavanjem konzistentnosti podataka
- integritet baze podataka može biti narušen i kao posljedica djelovanja neautoriziranih korisnika (diverzije ili sabotaze) međutim o tome brine poseban dio sigurnosti koji je zadužen za sigurnost baze podataka

Definicioni dio shema i instanca baze podataka

- Shema baze podataka sastoji se od
 - skupa relacijskih shema
 - $R = \{r_1, r_2, \dots, r_n\}$
 - i skupa integritetskih ograničenja *Integrity constraints*
 - IC = $\{ic_1, ic_2, \dots, ic_m\}$
- Instanca baze podataka stanje baze podataka definirana na shemi baze podataka $R = \{r_1, r_2, \dots, r_n\}$ je skup instanci relacija stanja relacija
 - $r = \{r_1, r_2, \dots, r_n\}$
- Ispravna instanca baze podataka je ona instanca koja zadovoljava **sva** definirana integritetska ograničenja

Definicioni dio integritetska ograničenja

- Primjer:** Shema baze podataka $R = \{r_1, r_2, r_3, r_4, r_5\}$
- relacijske sheme
 - $r_1 = \{mbr, preime, pbr, tan, dat, od, dat, ap\}$
 - mbr - ime mbr
 - $r_2 = \{pbr, na, jesto\}$
 - pbr - ime pbr
- integritetska ograničenja
 - vrijednost atributa mbr je i skupa cijeli brojeva i intervala
 - vrijednost atributa pbr je i skupa cijeli brojeva i intervala
 - ista vrijednost atributa mbr ne smije se pojaviti u dvije ili više niti u relaciji radnik
 - vrijednost atributa mbr je jedinstvena
 - vrijednost atributa mbr ne smije poprimiti vrijednost
 - razlika između dat, ap datum zaposlenja i dat, od datum rođenja ne smije biti manja od godina niti veća od godina
 - itd.

Definicioni dio integritetska ograničenja

- definicije integritetskih ograničenja su sastavni dio sheme baze podataka
- definicije integritetskih ograničenja se pojavljuju u **rječnik podataka** baze podataka
 - na taj način pravila definirana integritetskim ograničenjima postaju **neobilažna** za **svako** korisnika sustava
 - provjerava integritetska ograničenja pri obavljanju svake operacije koja mijenja sadržaj baze podataka
 - u trenutku **završetka** operacije nad podacima baze podataka mora biti u stanju u kojem su zadovoljena **sva** integritetska ograničenja
 - odbija** obaviti operacije koje nemaju to svojstvo ili obavlja kompenzacijske akcije koje osiguravaju da su u konačnici **sva** integritetska ograničenja zadovoljena

(Rječnik podataka)

- Data dictionary / Catalogue / Repository**
- opisi podataka (metapodaci) su poranjeni u rječnik podataka
 - ričani su na isti način i može im se pristupiti na isti način kao i običnim podacima
 - korisnici s pravom pristupa nad rječnikom podataka mogu primijeniti relacijski upitni jezik (np. SQL)
- rječnik podataka sadrži
 - opis relacijskih shema
 - opis pravila integriteta
 - opis pravila pristupa (korisnik, objekt, dovoljena akcija)
 - opis poranjenih procedura (poslovni pravila)
 - opis okidača (triggers)
 - itd.

(Rječnik podataka) primjer

- baza podataka u **informacijskom sustavu** sadrži nekoliko desetaka (sistemski) relacija koje čine rječnik podataka i koje se kreiraju automatski prilikom kreiranja baze podataka
- u relacijama **systables** i **syscolumns** poranjeni su metapodaci o relacijama i atributima

```
SELECT *
FROM systables, syscolumns
WHERE systables.tabid = syscolumns.tabid
AND tabname = 'mjesto'
ORDER BY colno;
```

systables						syscolumns					
tabname	owner	tabid	ncols	ncols	created	colname	tabid	colno	coltype	collen	collen
mjesto	bpadmin	3	3	3	2019-01-01 10:00:00	pbr	3	1	int	4	4
mjesto	bpadmin	3	3	3	2019-01-01 10:00:00	na_mjesto	3	2	varchar	40	40
mjesto	bpadmin	3	3	3	2019-01-01 10:00:00	si_tupanja	3	3	int	4	4

Integritetska ograničenja

- Entitetski integritet (*Entity integrity*)
- Integritet ključa (*Key integrity*)
- Domenski integritet (*Domain integrity*)
- Ograničenja vrijednosti (*Constraints on NULL*)
- Referencijski integritet (*Referential integrity*)
- Opća integritetska ograničenja (*General integrity constraints*)

Entitetski integritet

- Imati jedan atribut **primarno** ključa ne smije poprimiti nijednu vrijednost
- Primjer:

siast	jmbast	preast
1	1	1
2	2	2
3	3	3

Primarni ključ označen je s 1

mbrstud	sifpred	datlsp	ocj	siast
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3

⇒ atribut *sifNast* ne smije poprimiti nijednu vrijednost niti u jednoj n-torci relacije *nastavnik(NASTAVNIK)*
⇒ atributi *mbrStud*, *sifPred*, *datlsp* ne smiju poprimiti nijednu vrijednost niti u jednoj n-torci relacije *ispit(ISPIT)*

Integritet ključa

- U relaciji ne smiju postojati dvije n-torke s jednakim vrijednostima ključa. Vrijedi za sve moguće ključeve.
- Primjer:

siast	jmbast	preast
1	1	1
2	2	2
3	3	3

mbrstud	sifpred	datlsp	ocj	siast
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3

⇒ u relaciji *nastavnik(NASTAVNIK)* ne smiju postojati dvije n-torke koje imaju jednake vrijednosti atributa *sifNast*
⇒ u relaciji *nastavnik(NASTAVNIK)* ne smiju postojati dvije n-torke koje imaju jednake vrijednosti atributa *jmbgNast*
⇒ u relaciji *ispit(ISPIT)* ne smiju postojati dvije n-torke koje imaju jednake vrijednosti istu kombinaciju vrijednosti atributa *mbrStud*, *sifPred* i *datlsp*

Domenski integritet

- Atribut može poprimiti samo jednu vrijednost iz domene atributa
 - Primjer:

pbr	najesto
1	1
2	2
3	3
 - domena atributa *pbr* je skup cijelih brojeva u intervalu [1, 1000000000]
- ⇒ vrijednost atributa *pbr* u svakoj n-torci relacije *mjesto(MJESTO)* mora biti cijeli broj u intervalu [1, 1000000000]

Ograničenja vrijednosti

- Za određene attribute se može definirati ograničenje prema kojem vrijednost atributa ne smije poprimiti nijednu vrijednost
- Primjer:

mbrstud	imestud	prestud	adresa
1	1	1	1
2	2	2	2
3	3	3	3

⇒ vrijednost atributa *imeStud* ne smiju poprimiti nijednu vrijednost niti u jednoj n-torci relacije *student(STUDENT)*
⇒ vrijednost atributa *prezStud* ne smiju poprimiti nijednu vrijednost niti u jednoj n-torci relacije *student(STUDENT)*

atribut *mbrStud*

Strani ključ (Foreign key) i referencijski integritet

- Referencijski integritet se odnosi na konzistentnost među n-torkama dviju relacija ili n-torkama iste relacije. Formalno, n-torka iz jedne relacije koja se počinje referencirana drugu relaciju se može pojaviti referencirati samo na postojeće n-torke u toj relaciji
- Primjer:

mbr	pre	pbrtan
1	1	1
2	2	2
3	3	3

mbr	pre	pbrtan
1	1	1
2	2	2
3	3	3

osoba: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837, 838, 839, 840, 841, 842, 843, 844, 845, 846, 847, 848, 849, 850, 851, 852, 853, 854, 855, 856, 857, 858, 859, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876, 877, 878, 879, 880, 881, 882, 883, 884, 885, 886, 887, 888, 889, 890, 891, 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 912, 913, 914, 915, 916, 917, 918, 919, 920, 921, 922, 923, 924, 925, 926, 927, 928, 929, 930, 931, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941, 942, 943, 944, 945, 946, 947, 948, 949, 950, 951, 952, 953, 954, 955, 956, 957, 958, 959, 960, 961, 962, 963, 964, 965, 966, 967, 968, 969, 970, 971, 972, 973, 974, 975, 976, 977, 978, 979, 980, 981, 982, 983, 984, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 995, 996, 997, 998, 999, 1000, 1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018, 1019, 1020, 1021, 1022, 1023, 1024, 1025, 1026, 1027, 1028, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039, 1040, 1041, 1042, 1043, 1044, 1045, 1046, 1047, 1048, 1049, 1050, 1051, 1052, 1053, 1054, 1055, 1056, 1057, 1058, 1059, 1060, 1061, 1062, 1063, 1064, 1065, 1066, 1067, 1068, 1069, 1070, 1071, 1072, 1073, 1074, 1075, 1076, 1077, 1078, 1079, 1080, 1081, 1082, 1083, 1084, 1085, 1086, 1087, 1088, 1089, 1090, 1091, 1092, 1093, 1094, 1095, 1096, 1097, 1098, 1099, 1100, 1101, 1102, 1103, 1104, 1105, 1106, 1107, 1108, 1109, 1110, 1111, 1112, 1113, 1114, 1115, 1116, 1117, 1118, 1119, 1120, 1121, 1122, 1123, 1124, 1125, 1126, 1127, 1128, 1129, 1130, 1131, 1132, 1133, 1134, 1135, 1136, 1137, 1138, 1139, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1150, 1151, 1152, 1153, 1154, 1155, 1156, 1157, 1158, 1159, 1160, 1161, 1162, 1163, 1164, 1165, 1166, 1167, 1168, 1169, 1170, 1171, 1172, 1173, 1174, 1175, 1176, 1177, 1178, 1179, 1180, 1181, 1182, 1183, 1184, 1185, 1186, 1187, 1188, 1189, 1190, 1191, 1192, 1193, 1194, 1195, 1196, 1197, 1198, 1199, 1200, 1201, 1202, 1203, 1204, 1205, 1206, 1207, 1208, 1209, 1210, 1211, 1212, 1213, 1214, 1215, 1216, 1217, 1218, 1219, 1220, 1221, 1222, 1223, 1224, 1225, 1226, 1227, 1228, 1229, 1230, 1231, 1232, 1233, 1234, 1235, 1236, 1237, 1238, 1239, 1240, 1241, 1242, 1243, 1244, 1245, 1246, 1247, 1248, 1249, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1259, 1260, 1261, 1262, 1263, 1264, 1265, 1266, 1267, 1268, 1269, 1270, 1271, 1272, 1273, 1274, 1275, 1276, 1277, 1278, 1279, 1280, 1281, 1282, 1283, 1284, 1285, 1286, 1287, 1288, 1289, 1290, 1291, 1292, 1293, 1294, 1295, 1296, 1297, 1298, 1299, 1300, 1301, 1302, 1303, 1304, 1305, 1306, 1307, 1308, 1309, 1310, 1311, 1312, 1313, 1314, 1315, 1316, 1317, 1318, 1319, 1320, 1321, 1322, 1323, 1324, 1325, 1326, 1327, 1328, 1329, 1330, 1331, 1332, 1333, 1334, 1335, 1336, 1337, 1338, 1339, 1340, 1341, 1342, 1343, 1344, 1345, 1346, 1347, 1348, 1349, 1350, 1351, 1352, 1353, 1354, 1355, 1356, 1357, 1358, 1359, 1360, 1361, 1362, 1363, 1364, 1365, 1366, 1367, 1368, 1369, 1370, 1371, 1372, 1373, 1374, 1375, 1376, 1377, 1378, 1379, 1380, 1381, 1382, 1383, 1384, 1385, 1386, 1387, 1388, 1389, 1390, 1391, 1392, 1393, 1394, 1395, 1396, 1397, 1398, 1399, 1400, 1401, 1402, 1403, 1404, 1405, 1406, 1407, 1408, 1409, 1410, 1411, 1412, 1413, 1414, 1415, 1416, 1417, 1418, 1419, 1420, 1421, 1422, 1423, 1424, 1425, 1426, 1427, 1428, 1429, 1430, 1431, 1432, 1433, 1434, 1435, 1436, 1437, 1438, 1439, 1440, 1441, 1442, 1443, 1444, 1445, 1446, 1447, 1448, 1449, 1450, 1451, 1452, 1453, 1454, 1455, 1456, 1457, 1458, 1459, 1460, 1461, 1462, 1463, 1464, 1465, 1466, 1467, 1468, 1469, 1470, 1471, 1472, 1473, 1474, 1475, 1476, 1477, 1478, 1479, 1480, 1481, 1482, 1483, 1484, 1485, 1486, 1487, 1488, 1489, 1490, 1491, 1492, 1493, 1494, 1495, 1496, 1497, 1498, 1499, 1500, 1501, 1502, 1503, 1504, 1505, 1506, 1507, 1508, 1509, 1510, 1511, 1512, 1513, 1514, 1515, 1516, 1517, 1518, 1519, 1520, 1521, 1522, 1523, 1524, 1525, 1526, 1527, 1528, 1529, 1530, 1531, 1532, 1533, 1534, 1535, 1536, 1537, 1538, 1539, 1540, 1541, 1542, 1543, 1544, 1545, 1546, 1547, 1548, 1549, 1550, 1551, 1552, 1553, 1554, 1555, 1556, 1557, 1558, 1559, 1560, 1561, 1562, 1563, 1564, 1565, 1566, 1567, 1568, 1569, 1570, 1571, 1572, 1573, 1574, 1575, 1576, 1577, 1578, 1579, 1580, 1581, 1582, 1583, 1584, 1585, 1586, 1587, 1588, 1589, 1590, 1591, 1592, 1593, 1594, 1595, 1596, 1597, 1598, 1599, 1600, 1601, 1602, 1603, 1604, 1605, 1606, 1607, 1608, 1609, 1610, 1611, 1612, 1613, 1614, 1615, 1616, 1617, 1618, 1619, 1620, 1621, 1622, 1623, 1624, 1625, 1626, 1627, 1628, 1629, 1630, 1631, 1632, 1633, 1634, 1635, 1636, 1637, 1638, 1639, 1640, 1641, 1642, 1643, 1644, 1645, 1646, 1647, 1648, 1649, 1650, 1651, 1652, 1653, 1654, 1655, 1656, 1657, 1658, 1659, 1660, 1661, 1662, 1663, 1664, 1665, 1666, 1667, 1668, 1669, 1670, 1671, 1672, 1673, 1674, 1675, 1676, 1677, 1678, 1679, 1680, 1681, 1682, 1683, 1684, 1685, 1686, 1687, 1688, 1689, 1690, 1691, 1692, 1693, 1694, 1695, 1696, 1697, 1698, 1699, 1700, 1701, 1702, 1703, 1704, 1705, 1706, 1707, 1708, 1709, 1710, 1711, 1712, 1713, 1714, 1715, 1716, 1717, 1718, 1719, 1720, 1721, 1722, 1723, 1724, 1725, 1726, 1727, 1728, 1729, 1730, 1731, 1732, 1733, 1734, 1735, 1736, 1737, 1738, 1739, 1740, 1741, 1742, 1743, 1744, 1745, 1746, 1747, 1748, 1749, 1750, 1751, 1752, 1753, 1754, 1755, 1756, 1757, 1758, 1759, 1760, 1761, 1762, 1763, 1764, 1765, 1766, 1767, 1768, 1769, 1770, 1771, 1772, 1773, 1774, 1775, 1776, 1777, 1778, 1779, 1780, 1781, 1782, 1783, 1784, 1785, 1786, 1787, 1788, 1789, 1790, 1791, 1792, 1793, 1794, 1795, 1796, 1797, 1798, 1799, 1800, 1801, 1802, 1803, 1804, 1805, 1806, 1807, 1808, 1809, 1810, 1811, 1812, 1813, 1814,

Strani ključ (Foreign key) i referencijski integritet

- Primjer
osoba
mbr pre pbr tan
mbr
pbr na jesto
pbr

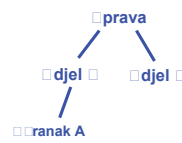
osoba	mbr	pre	pbr tan
	ovrat		
	olar		
	ovak	3	

mjesto	pbr	na jesto
	areb	
	ijeka	

- Kup atributa pbr tan je strani ključ u relaciji osoba koji se poiva na relaciju mjesto
- Relacije osoba i mjesto ne zadovoljavaju pravilo referencijskog integriteta jer u relaciji osoba postoji norka t, s ovak 3 koja u relaciji mjesto ne postoji odovarajuća norka t, s vrijednoću atributa pbr jednakoj 3
norka olar ne naruava referencijski integritet

Strani ključ (Foreign key) i referencijski integritet

- Primjer
si rjed na rjed si ad rjed
si rjed



orjed	si rjed	na rjed	si ad rjed
	prava		
	djel		
3	djel		
4	djel		
	ranak		

- Kup atributa si ad rjed je strani ključ u relaciji orgjed koji se poiva na relaciju orgjed
- Relacija orgjed(ORGJED) ne zadovoljava pravilo referencijskog integriteta jer u relaciji postoji norka t, s 4 djel koja u istoj relaciji ne postoji odovarajuća norka t, s vrijednoću atributa si rjed jednakoj
norka prava ne naruava referencijski integritet

Strani ključ (Foreign key) i referencijski integritet

- Postoje slučajevi u kojima strani ključ i rone smije biti jednak vrijednosti vrijedi a slučaj kad se pravila referencijskog integriteta sukobe s pravilom entitetsko integriteta
strani ključ relacije rone koji je ujedno dio primarno ključa relacije rone smije poprimiti rone vrijednost

- Primjer
mbr tud ime tud pre tud
mbr tud
mbr tud si red dat sp ocj si ast
mbr tud si red dat sp

Kup atributa mbr tud je strani ključ u relaciji ispit(ISPIT) koji se poiva na relaciju stud(STUD) ali je ujedno dio prim ključa u relaciji ispit
⇒ atribut mbrStud u relaciji ispit ne smije poprimiti rone vrijednost

PCA integritetska ograničenja

- PCA integritetska ograničenja su ograničenja opće eneralno oblika
npr poslovna pravila

- Primjer ograničenje odnosa među vrijednostima atributa

mbr pre ime pbr tan dat od dat ap
mbr

⇒ razlika između vrijednosti atributa datZap (datum zaposlenja) i datRod (datum rođenja) ne smije biti manja od godina niti veća od godina
o integritetsko ograničenje proia i ova ovaj primjer imijeno zakonsko ograničenja da se osobe mlađe od godina ili starije od godina ne smiju apojaviti

Implementacija integritetskih ograničenja u SQLu

- Integritetska ograničenja se mogu definirati
u okviru naredbe a kreiranje relacije ili
naknadnim definiranjem pomoću naredbe

Integritetska ograničenja za atribut

```
CREATE TABLE tableName
( {columnName dataType [DEFAULT defaultExpr]
  [columnConstraint] [, ...] | tableConstraint [, ...] )
```

Column constraints:

```
[CONSTRAINT constraintName]
{ NOT NULL | UNIQUE | PRIMARY KEY |
  CHECK (condition) |
  REFERENCES reftable [(refcolumn)] [ ON DELETE action]
  [ ON UPDATE action] }
```

action: NO ACTION, CASCADE, SET NULL, SET DEFAULT
condition - odnosi se samo na dotičnu kolonu

Integritetska ograničenja za tablice

```
CREATE TABLE tableName  
( { columnName dataType [DEFAULT defaultExpr]  
  [columnConstraint] [, ...] | tableConstraint } [, ...] )
```

Table constraints:

```
[CONSTRAINT constraintName]  
{ UNIQUE (columnName [, ...]) |  
  PRIMARY KEY (columnName [, ...]) |  
  CHECK (condition) |  
  FOREIGN KEY (columnName [, ...]) REFERENCES  
    reftable [(refcolumn [, ...])] [ ON DELETE action]  
    [ ON UPDATE action] }
```

SQL: PRIMARY KEY

- Primjer: `CREATE TABLE ispiti (mbrStud INTEGER, sifPred INTEGER, datIsp DATE, ocj SMALLINT, sifNast INTEGER, PRIMARY KEY (mbrStud, sifPred, datIsp));`

```
CREATE TABLE ispiti (  
  mbrStud INTEGER  
, sifPred INTEGER  
, datIsp DATE  
, ocj SMALLINT  
, sifNast INTEGER  
, PRIMARY KEY (  
  mbrStud, sifPred, datIsp  
)  
);
```

- Osigurava entitetski integritet i integritet ključa

vidjeti sintaksne dijagrame 11.1 i 11.2

- entitetski integritet i integritet ključa su primarni ključ se uvijek osigurava pomoću jednog od dva načina

Definiranje integritetskih ograničenja pri definiranju atributa

- U slučajevima kad se integritetsko ograničenje odnosi na samo jedan atribut može se definirati neposredno u definiciju atributa (to vrijedi za sve vrste ograničenja)

- Primjer: `CREATE TABLE nastavnici (sifNast INTEGER, jmbgNast CHAR(13), prezNast CHAR(40), PRIMARY KEY (sifNast), UNIQUE (jmbgNast));`

```
CREATE TABLE nastavnici (  
  sifNast INTEGER PRIMARY KEY  
, jmbgNast CHAR(13)  
, prezNast CHAR(40)  
);
```

vidjeti sintaksne dijagrame 11.1 i 11.4

SQL: UNIQUE

- Primjer: `CREATE TABLE nastavnici (sifNast INTEGER, jmbgNast CHAR(13), prezNast CHAR(40), PRIMARY KEY (sifNast), UNIQUE (jmbgNast));`

```
CREATE TABLE nastavnici (  
  sifNast INTEGER  
, jmbgNast CHAR(13)  
, prezNast CHAR(40)  
, PRIMARY KEY (sifNast)  
, UNIQUE (jmbgNast)  
);
```

- Osigurava integritet ključa

vidjeti sintaksne dijagrame 11.1 i 11.4

```
CREATE TABLE nastavnici (  
  sifNast INTEGER PRIMARY KEY  
, jmbgNast CHAR(13) UNIQUE  
, prezNast CHAR(40)  
... );
```

vidjeti sintaksne dijagrame 11.1 i 11.4

SQL: CHECK

- Domenski integritet je djelomično osiguravan samom definicijom tipa podatka za atribut
 - Upravo definiranjem podatka tipa `INTEGER` određena je njegova domena kao skup cijelih brojeva u intervalu `[-32768 do 32767]`
- Obično je poželjno postići točnije određenje domene atributa

```
CREATE TABLE ispiti (  
  mbrStud INTEGER  
, sifPred INTEGER  
, datIsp DATE  
, ocj SMALLINT  
, sifNast INTEGER  
, PRIMARY KEY (mbrStud, sifPred, datIsp)  
, CHECK (ocj BETWEEN 1 AND 5)  
);
```

- Osigurava domenski integritet

vidjeti sintaksne dijagrame 11.1 i 11.4

```
CREATE TABLE ispiti (  
  ...  
, ocj SMALLINT CHECK (ocj BETWEEN 1 AND 5)  
, sifNast INTEGER  
);
```

vidjeti sintaksne dijagrame 11.1 i 11.4

SQL: CHECK

- Također se može koristiti za definiranje ograničenja odnosa među vrijednostima atributa u istoj tablici (vidjeti primjer za opće pravilo integriteta)

Primjer: razlika između vrijednosti atributa `datZap` (datum zaposlenja) i `datRod` (datum rođenja) ne smije biti manja od 16 godina niti veća od 65 godina

```
CREATE TABLE radnici (  
  mbr INTEGER  
, ime CHAR(40)  
, prez CHAR(40)  
, datRod DATE  
, datZap DATE  
, CHECK (datZap - datRod >= 16*365  
  AND datZap - datRod <= 65*365)  
);
```

razliku je definirano da prestupne godine broje po 366 dana

- Ograničenje koje se tiče odnosa među vrijednostima atributa se ne može napisati neposredno u definiciju atributa

SQL: NOT NULL

- ograničenje vrijednosti se postiže navođenjem rezervirani riječi **NOT NULL** i tipa podatka pri definiciji atributa

- primjer
sifOrgjed
nazOrgjed
sifNadOrgjed
PRIMARY KEY (sifOrgjed)
FOREIGN KEY (sifNadOrgjed) REFERENCES orgjed (sifOrgjed)

```
CREATE TABLE orgjed (  
  sifOrgjed INTEGER  
  , nazOrgjed CHAR(40) NOT NULL  
  , sifNadOrgjed INTEGER  
  , PRIMARY KEY (sifOrgjed)  
  , FOREIGN KEY (sifNadOrgjed)  
    REFERENCES orgjed (sifOrgjed)  
);
```

- može li dopustiti da vrijednost atributa sifOrgjed bude
- može li dopustiti da vrijednost atributa sifNadOrgjed bude

SQL: FOREIGN KEY

- primjer
 - primarni ključ u relaciji student je mbrStud
 - primarni ključ u relaciji predmet je sifPred
 - primarni ključ u relaciji nastavnik je sifNast

```
CREATE TABLE ispit (  
  mbrStud INTEGER  
  , sifPred INTEGER  
  , datIsp DATE  
  , ocj SMALLINT  
  , sifNast INTEGER  
  , PRIMARY KEY (mbrStud, sifPred, datIsp)  
  , FOREIGN KEY (mbrStud) REFERENCES student (mbrStud)  
  , FOREIGN KEY (sifPred) REFERENCES predmet (sifPred)  
  , FOREIGN KEY (sifNast) REFERENCES nastavnik (sifNast)  
);
```

osigurava referentnost strani ključ u relaciji ispit skup atributa sifNast poivava se na primarni ključ u relaciji nastavnik skup atributa sifNast
odražuje se da su u relacijama student-predmet i nastavnik pomoću deklarativna ograničenja referentnosti i integritet ključa

SQL: FOREIGN KEY

```
CREATE TABLE ispit (  
  mbrStud INTEGER REFERENCES student (mbrStud)  
  , sifPred INTEGER REFERENCES predmet (sifPred)  
  , datIsp DATE  
  , ocj SMALLINT  
  , sifNast INTEGER REFERENCES nastavnik (sifNast)  
  , PRIMARY KEY (mbrStud, sifPred, datIsp)  
);
```

vidjeti sintaksne dijagrame
i

SQL: FOREIGN KEY

- pri definiciji ograničenja referencijskog integriteta moguće je specificirati da li će pri pokušaju narušavanja ograničenja **brisanjem poivane nitorke**

- odbiti operaciju brisanja poivane nitorke

- ovakvo ponašanje

- obaviti operaciju brisanja poivane nitorke ali pri tome obaviti i kompenzacijske akcije koje će rezultirati time da integritetsko ograničenje u konačnici bude zadovoljeno oduče akcije su

- vrijednosti strano ključa u norkama koje se poivaju na obrisani norku postaviti na vrijednosti

- ovakvo ponašanje

- vrijednosti strano ključa u norkama koje se poivaju na obrisani norku postaviti na default vrijednosti

- ovakvo ponašanje

- obrisati poivajuće nitorke

- ovakvo ponašanje

SQL: FOREIGN KEY

- pri definiciji ograničenja referencijskog integriteta također je moguće specificirati da li će pri pokušaju narušavanja ograničenja **imjenom primarno ključa u poivanoj ntorci**

- odbiti operaciju imjene poivane nitorke

- ovakvo ponašanje

- obaviti operaciju imjene poivane nitorke ali pri tome obaviti i kompenzacijske akcije koje će rezultirati time da integritetsko ograničenje u konačnici bude zadovoljeno oduče akcije su

- vrijednosti strano ključa u norkama koje se poivaju na imijenjenu norku postaviti na vrijednosti

- ovakvo ponašanje

- vrijednosti strano ključa u norkama koje se poivaju na obrisani norku postaviti na default vrijednosti

- ovakvo ponašanje

- vrijednosti strano ključa u norkama koje se poivaju na imijenjenu norku postaviti na novu vrijednost primarno ključa poivane nitorke

- ovakvo ponašanje

SQL: FOREIGN KEY

student				ispit					
mbr	pre	ime		mbr	sifred	datisp	ocj	sifast	
34	ovak	ivan							
34	olar	petar							

predmet				nastavnik			
sifred	na	red		sifast	pre	ast	
	at				aić		
	at				etić		
33				3333	orvat		

peracije koje bi narušile referencijski integritet:

- unos ispita na nepostojeće studenta
- unos ispita na nepostojeće predmeta
- unos ispita kod nepostojeće nastavnika
- imjene u tablici ispit
 - mbr se mijenja na neku vrijednost koja ne postoji u tablici student
 - sifred se mijenja na neku vrijednost koja ne postoji u tablici predmet
 - sifast se mijenja na neku vrijednost koja ne postoji u tablici nastavnik

odgovara li nam to **DA**

SQL: FOREIGN KEY

student	mbr	pre	ime
	0000	ovak	van
	034	olar	etar

predmet	sifred	na	red
	0000	at	
	0003	at	

nastavnik	sifast	pre	ast
	0000	al	ic
	0000	rneti	c
	3333	orvat	

ispit	mbr	sifred	datisp	ocj	sifast
	0000	0000	00000000	0	0000
	0000	0000	00000000	0	0000
	0000	0000	00000000	0	0000
	0000	0003	03000000	3	3333
	0000	0000	00000000	4	0000
	0034	0000	00000000	3	0000

perakcije koje bi također naruile referencijski integritet:

- brisanje podataka o studentu koji se ispisao s fakulteta npr 0000 ovak van
- brisanje podataka o predmetu koji više ne postoji u novom nastavnom programu
- brisanje nastavnika koji je otišao u mirovinu
- 0000 0000 00000000 00 0000 00

primjedbe

eljeli bismo arhivirati podatke o studentima i nastavnicima koji su napustili fakultet i ibrisati i iaktualne bae podataka

SQL: FOREIGN KEY

- aličite reakcije na pokušaj naruavanja referencijskog integriteta brisanjem poivani ntorke
 - odbijanje operacije a strani ključ sifred
 - obavljanje kompenacijski akcija
 - u kaskadno brisanje a strani ključ mbr
 - u postavljanje na 0000 vrijednosti a strani ključ sifast

```
CREATE TABLE ispit (
  mbr      INTEGER
, sifPred  INTEGER
, datIsp   DATE
, ocj      SMALLINT
, sifNast  INTEGER
, PRIMARY KEY (mbr, sifPred, datIsp)
, FOREIGN KEY (mbr) REFERENCES student (mbr)
  ON DELETE CASCADE
, FOREIGN KEY (sifPred) REFERENCES predmet (sifPred)
, FOREIGN KEY (sifNast) REFERENCES nastavnik (sifNast)
  ON DELETE SET NULL
);
```

efektivni integritet definiran uz odbijanje operacije

koliko se pokušaju obrisati ntorke i tablice predmet na čije se matične brojeve poivaju ntorke i tablice ispit

```
CREATE TABLE ispit (
  mbr      INTEGER
, sifPred  INTEGER
, datIsp   DATE
, ocj      SMALLINT
, sifNast  INTEGER
, PRIMARY KEY (mbr, sifPred, datIsp)
, FOREIGN KEY (mbr) REFERENCES student (mbr)
  ON DELETE CASCADE
, FOREIGN KEY (sifPred) REFERENCES predmet (sifPred)
, FOREIGN KEY (sifNast) REFERENCES nastavnik (sifNast)
  ON DELETE SET NULL
);
```

operacija brisanja ntorke i tablice predmet će biti odbijena korisnik ili aplikacija će dobiti poruku o pogrešci

efektivni integritet definiran uz kaskadno brisanje

koliko se pokušaju obrisati ntorke i tablice student na čije se matične brojeve poivaju ntorke i tablice ispit

```
CREATE TABLE ispit (
  mbr      INTEGER
, sifPred  INTEGER
, datIsp   DATE
, ocj      SMALLINT
, sifNast  INTEGER
, PRIMARY KEY (mbr, sifPred, datIsp)
, FOREIGN KEY (mbr) REFERENCES student (mbr)
  ON DELETE CASCADE
, FOREIGN KEY (sifPred) REFERENCES predmet (sifPred)
, FOREIGN KEY (sifNast) REFERENCES nastavnik (sifNast)
  ON DELETE SET NULL
);
```

brisaat će se ntorke i tablice student i sve ntorke i tablice ispit koje se poivaju na obrisane ntorke i tablice student

efektivni integritet definiran uz postavljanje na 0000 vrijednosti

koliko se pokušaju obrisati ntorke i tablice nastavnik na čije se matične brojeve poivaju ntorke i tablice ispit

```
CREATE TABLE ispit (
  mbr      INTEGER
, sifPred  INTEGER
, datIsp   DATE
, ocj      SMALLINT
, sifNast  INTEGER
, PRIMARY KEY (mbr, sifPred, datIsp)
, FOREIGN KEY (mbr) REFERENCES student (mbr)
  ON DELETE CASCADE
, FOREIGN KEY (sifPred) REFERENCES predmet (sifPred)
, FOREIGN KEY (sifNast) REFERENCES nastavnik (sifNast)
  ON DELETE SET NULL
);
```

brisaat će se ntorke i tablice nastavnik a vrijednosti strana ključa sifast u tablici ispit koje se poivaju na obrisane ntorke će se postaviti na 0000

SQL: imenovanje integritetskog ograničenja

- naiv integritetsko ograničenje **CONSTRAINT constraint** se navodi opcionalno ako se navede korisnik ili aplikacija će pri pokušaju obavljanja naredbe koja naruava integritetsko ograničenje dobiti informaciju o kojem se točno integritetskom ograničenju radi

```
CREATE TABLE ispit (
  mbrStud  INTEGER
, sifPred  INTEGER
, datIsp   DATE
, ocj      SMALLINT
, sifNast  INTEGER NOT NULL
, PRIMARY KEY (mbrStud, sifPred, datIsp)
, FOREIGN KEY (mbrStud) REFERENCES stud (mbrStud)
, FOREIGN KEY (sifPred) REFERENCES pred (sifPred)
, FOREIGN KEY (sifNast) REFERENCES nast (sifNast)
);
```

- primjer uklanjanja definiranog integritetskog ograničenja

```
ALTER TABLE ispit DROP CONSTRAINT ocjNotNull;
```

Svojstva: Ograničenja

- isključivo** u onim slučajevima kada baza ne podržava mogućnost definiranja ograničenja tipa *PRIMARY KEY* i *UNIQUE*
 - entitetski integritet se može osigurati specifikiranjem ograničenja *NOT NULL* u atributu primarno ključa
 - integritet ključa se može osigurati kreiranjem indeksa *UNIQUE INDEX* nad ključem
 - po jedan takav indeks se kreira za svaki moguć ključ i ne svaki atribut ključa
- većina današnjih sustava za upravljanje bazama podataka podržava mogućnost definiranja tipova ograničenja

Svojstva: Ograničenja

- većina sustava za upravljanje bazama podataka automatski kreira *UNIQUE* indekse pri definiranju sljedećih ograničenja
 - primarno ograničenje
 - automatski kreira indeks na primarno ograničenje
 - ograničenje *UNIQUE*
 - automatski kreira indeks na ograničenje
- neki sustavi (npr. Informatica) također automatski kreiraju indekse pri definiranju ograničenja referencijskog integriteta
 - referencijski integritet
 - automatski kreira indeks na referencijski integritet

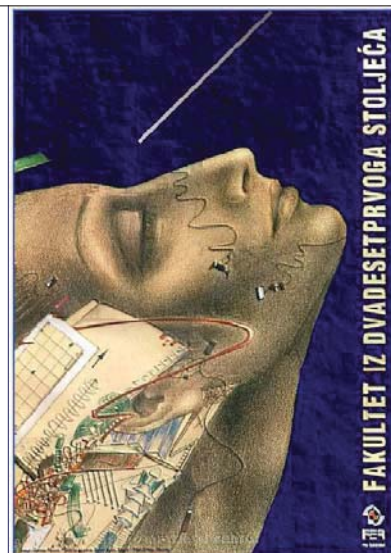
Svojstva: Ograničenja

- neki informacije o definiciji referencijskog integriteta
 - nisu podržane opcije
 - ograničenje *ON DELETE CASCADE*
 - ograničenje *ON DELETE SET NULL*
 - ograničenje *ON DELETE SET DEFAULT*
 - ograničenje *ON DELETE NO ACTION*
 - ograničenje *ON DELETE RESTRICT*
 - podržane su opcije
 - ograničenje *ON DELETE CASCADE*
 - ograničenje *ON DELETE SET NULL* ali se ne navodi jer se podrazumijeva u slučaju kad nije navedena opcija *ON DELETE SET DEFAULT*
 - ograničenje *ON DELETE SET DEFAULT* ali se ne navodi jer se podrazumijeva

Baze podataka

Predavanja
travanj 2008.

10. Integritet baze podataka - dodatni primjer -



Strani ključ (Foreign key) i referencijski integritet

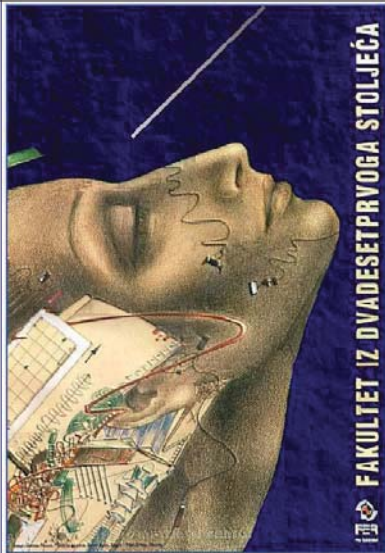
- Zadane su relacije $r(R)$ s primarnim ključem PK_R i $s(S)$ s primarnim ključem PK_S . Skup atributa FK , $FK \subseteq R$, je strani ključ u relaciji $r(R)$ koji se poziva na relaciju $s(S)$ ukoliko vrijedi:
 - atributi u skupu FK imaju domene jednake domenama korespondentnih atributa u skupu PK_S
 - za svaku n-torku $t_1 \in r(R)$
 - postoji n-torka $t_2 \in s(S)$ takva da je $t_2[PK_S] = t_1[FK]$
- ili
 - barem jedna vrijednost atributa iz $t_1[FK]$ je NULL vrijednost**

Strani ključ (Foreign key) i referencijski integritet

	osoba(OSOBA)				mjesto(MJESTO)			drzava(DRZAVA)	
	mbr	prez	oznDrz	pbr	oznDrz	pbr	nazMj	oznDrz	nazDrz
t_1	101	Horvat	HR	10000	HR	10000	Zagreb	HR	Hrvatska
t_2	102	Jones	GB	51000	GB	51000	Leeds	GB	V. Britanija
t_3	103	Kolar	NULL	NULL	HR	51000	Rijeka		
t_4	104	Smith	GB	NULL	GB	10000	Bristol		
t_5	105	Novak	NULL	47000					
t_6	106	Clark	USA	NULL					
t_7	107	Adams	GB	47000					
t_8	108	Wilson	USA	10000					

- $FK_1 = \{ \text{oznDrz}, \text{pbr} \}$ je strani ključ u relaciji *osoba* koji se poziva na relaciju *mjesto*
- $FK_2 = \{ \text{oznDrz} \}$ je strani ključ u relaciji *osoba* koji se poziva na relaciju *drzava*
- $FK_3 = \{ \text{oznDrz} \}$ je strani ključ u relaciji *mjesto* koji se poziva na relaciju *drzava*
- n-torke t_1, t_2, t_3, t_4, t_5 zadovoljavaju pravila referencijskog integriteta za FK_1 i FK_2
- n-torka t_6 ne zadovoljava pravilo referencijskog integriteta za FK_2
- n-torka t_7 ne zadovoljava pravilo referencijskog integriteta za FK_1
- n-torka t_8 ne zadovoljava pravila referencijskog integriteta za FK_1 i FK_2

11. Privremene i virtualne relacije



Vrste relacija (tablica)

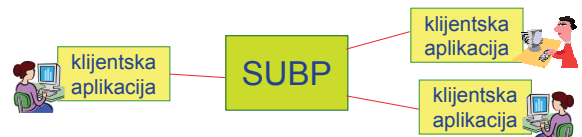
- **Temeljna relacija (base relation)**
 - relacija korespondentna skupu entiteta u konceptualnoj shemi, čija su shema i sadržaj trajno pohranjeni u bazi podataka
- **Privremena relacija (temporary relation)**
 - relacija čija su shema i sadržaj u bazu podataka pohranjeni privremeno
- **Virtualna relacija (virtual relation)**
 - relacija kojoj su shema i sadržaj definirani izrazom relacijske algebre čiji su operandi temeljne ili virtualne relacije
 - u praksi, shema i sadržaj virtualne relacije opisuju se u obliku SQL upita
 - sadržaj virtualne relacije dinamički se određuje u trenutku obavljanja operacije nad virtualnom relacijom: ovisi o trenutnom stanju temeljnih relacija

Temeljna relacija

- Obavljanjem naredbe CREATE TABLE u rječnik podataka se pohranjuju metapodaci
 - naziv relacije
 - nazivi i tipovi atributa
 - integritetska ograničenja
 - ostali metapodaci (vrijeme kreiranja, vlasnik, primijenjena fizička organizacija, itd.)
- shema i sadržaj temeljne relacije su **postojani**: pohranjeni su u bazi podataka na neograničeno vrijeme
 - mijenjaju se tek u slučaju obavljanja eksplicitnih operacija za izmjenu sadržaja (UPDATE, DELETE, INSERT) ili sheme relacije (ALTER TABLE)

(SQL-sjednica)

- SQL-sjednica (SQL-session) je kontekst u kojem jedan korisnik obavlja niz SQL naredbi putem jedne veze (SQL-Connection) prema sustavu za upravljanje bazama podataka
 - SQL-sjednica započinje u trenutku kada korisnik ostvari vezu (connect) sa sustavom za upravljanje bazama podataka
 - npr. u trenutku kada korisnik uporabom klijentske aplikacije Aqua Data Studio ostvari vezu s IBM Informix sustavom za upravljanje bazama podataka
 - SQL-sjednica završava u trenutku kada korisnik prekine vezu (disconnect) prema sustavu za upravljanje bazama podataka



Privremena relacija

- Privremena relacija se kreira obavljanjem naredbe CREATE TEMP TABLE
 - sintaksa preostalog dijela naredbe je identična sintaksi naredbe CREATE TABLE, uz određena ograničenja
 - npr. nije moguće definirati ograničenje referencijskog integriteta
- Privremena relacija je u dosegu ("vidljiva je") isključivo u okviru SQL-sjednice tijekom koje je kreirana
 - "svaka SQL-sjednica koristi svoje privremene relacije"
- Privremene relacije se koriste kao pomoćni objekti, npr. za pohranu međurezultata pri obavljanju složenijih upita
 - **zašto temeljne relacije nisu prikladne za tu namjenu?**
- Privremena relacija se uklanja iz baze podataka:
 - obavljanjem naredbe DROP TABLE *nazivPrivremeneRelacije* ili
 - završetkom SQL-sjednice tijekom koje je ta privremena relacija kreirana

Primjer

- Ispisati najmanju i najveću prosječnu ocjenu predmeta u obliku:

minPros	maksPros
3.00	4.00

```
CREATE TEMP TABLE prosjek (
  sifPred INTEGER
, prosOcJ DECIMAL(3,2));
```

prosiek
sifPred prosOcJ

```
INSERT INTO prosjek
SELECT sifPred, AVG(ocj)
FROM polozeniIspit
GROUP BY sifPred;
```

prosiek
sifPred prosOcJ
1001 4.00
1002 3.50
1003 3.00

```
SELECT MIN(prosOcJ) AS minPros
, MAX(prosOcJ) AS maksPros
FROM prosjek;
```

minPros	maksPros
3.00	4.00

polozeniIspit		
mbr	sifPred	ocj
100	1001	5
101	1001	4
102	1001	3
100	1002	2
101	1002	5
100	1003	3
101	1003	3

Za vježbu riješiti bez korištenja privremene relacije!

Primjer (nastavak)

- Treba primijetiti: sadržaj privremene relacije *prosjeck* neće se "automatski" promijeniti nakon upisa još jedne n-torke u temeljnu relaciju *polozeniIspit*

```
INSERT INTO polozeniIspit VALUES(102, 1003, 2);
```

polozeniIspit	mbr	sifPred	ocj
	100	1001	5
	101	1001	4
	102	1001	3
	100	1002	2
	101	1002	5
	100	1003	3
	101	1003	3
	102	1003	2

Sadržaj privremene relacije se time nije promijenio:

prosjeck	sifPred	prosOcJ
	1001	4.00
	1002	3.50
	1003	3.00

prosjeck za predmet 1003 bi trebao biti 2.67

```
SELECT MIN(prosOcJ) AS minPros,
       MAX(prosOcJ) AS maksPros
FROM prosjeck;
```

minPros	maksPros
3.00	4.00

neispravan rezultat

Virtualna relacija (primjer)

- Problem "zastarijevanja" podataka u privremenim relacijama može se izbjeći uporabom virtualnih relacija

polozeniIspit

mbr	sifPred	ocj
100	1001	5
101	1001	4
102	1001	3
100	1002	2
101	1002	5
100	1003	3
101	1003	3

```
CREATE VIEW prosjeck (sifPred, prosOcJ) AS
SELECT sifPred, AVG(ocj)
FROM polozeniIspit
GROUP BY sifPred;
```

prosjeck	sifPred	prosOcJ
	1001	4.00
	1002	3.50
	1003	3.00

- Tek u trenutku obavljanja upita, SUBP dinamički određuje sadržaj virtualne relacije *prosjeck*

```
SELECT MIN(prosOcJ) AS minPros,
       MAX(prosOcJ) AS maksPros
FROM prosjeck;
```

prosjeck	sifPred	prosOcJ
	1001	4.00
	1002	3.50
	1003	3.00

minPros	maksPros
3.00	4.00

Primjer (nastavak)

- Sadržaj virtualne relacije se ponovno određuje pri izvršavanju svakog upita koji koristi tu virtualnu relaciju

```
INSERT INTO polozeniIspit VALUES(102, 1003, 2);
```

polozeniIspit	mbr	sifPred	ocj
	100	1001	5
	101	1001	4
	102	1001	3
	100	1002	2
	101	1002	5
	100	1003	3
	101	1003	3
	102	1003	2

```
SELECT MIN(prosOcJ) AS minPros,
       MAX(prosOcJ) AS maksPros
FROM prosjeck;
```

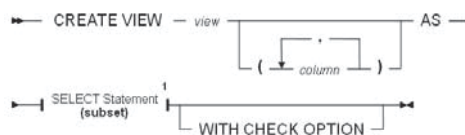
prosjeck	sifPred	prosOcJ
	1001	4.00
	1002	3.50
	1003	2.67

minPros	maksPros
2.67	4.00

Naredba za kreiranje virtualne relacije

- Virtualna relacija se kreira naredbom CREATE VIEW

6. CREATE VIEW Statement



- SELECT Statement* može sadržavati sve prethodno opisane dijelove SELECT naredbe osim
 - ORDER BY
 - FIRST n
- Virtualna relacija se uklanja iz baze podataka naredbom:
 - DROP VIEW nazivVirtualneRelacije;

Svojstva virtualne relacije

- Obavljanjem naredbe CREATE VIEW u rječnik podataka se pohranjuje samo definicija virtualne relacije
 - sadržaj virtualne relacije se određuje tek za vrijeme izvršavanja upita koji koristi virtualnu relaciju
 - odnosno, sadržaj virtualne relacije uvijek odražava sadržaj temeljnih relacija u trenutku izvršavanja upita u kojem se virtualna relacija koristi
- virtualne relacije se u upitima mogu koristiti na svim mjestima gdje se mogu koristiti temeljne relacije
 - između ostalog i za kreiranje novih virtualnih relacija
- za razliku od privremene relacije
 - definicija virtualne relacije je trajno pohranjena u bazi podataka
 - virtualna relacija je u dosegu ("vidljiva je") u svim SQL-sjedicama

Atributi virtualne relacije

- Ukoliko se nazivi atributa u definiciji virtualne relacije ne navedu, nazivi atributa virtualne relacije određeni su nazivima atributa u SELECT naredbi kojom se definira sadržaj virtualne relacije
- tipovi podataka za attribute virtualne relacije proizlaze iz tipova podataka atributa temeljnih relacija koje se koriste u definiciji virtualne relacije

osoba

mbr	ime	prez	pbrStan
101	Ana	Kolar	23000
102	Tomo	Novak	21000
103	Tea	Ban	23000

```
CREATE VIEW zadrani1 AS
SELECT mbr, ime, prez
FROM osoba
WHERE pbrStan = 23000;
SELECT * FROM zadrani1;
```

mbr	ime	prez
101	Ana	Kolar
103	Tea	Ban

```
CREATE VIEW zadrani2 (matBr, imeSt, prezSt) AS
SELECT mbr, ime, prez
FROM osoba
WHERE pbrStan = 23000;
SELECT * FROM zadrani2;
```

matBr	imeSt	prezSt
101	Ana	Kolar
103	Tea	Ban

Atributi virtualne relacije

- Ukoliko se u listi za selekciju pri definiciji virtualne relacije koriste izrazi, nazivi atributa virtualne relacije se moraju eksplicitno navesti

polozeniIsplit		
mbr	sifPred	ocj
100	1001	2
101	1001	4
102	1001	3
100	1002	2
101	1002	5
100	1003	3
101	1003	3

```
CREATE VIEW prosjek (sifPred, prosOcj) AS
SELECT sifPred, AVG(ocj)
FROM polozeniIsplit
GROUP BY sifPred;
```

ispravno

```
CREATE VIEW prosjek AS
SELECT sifPred,
       AVG(ocj)
FROM polozeniIsplit
GROUP BY sifPred;
```

neispravno

```
CREATE VIEW prosjek AS
SELECT sifPred,
       AVG(ocj) AS prosOcj
FROM polozeniIsplit
GROUP BY sifPred;
```

neispravno

Implementacija virtualnih relacija

- Kako sustavi za upravljanje bazama podataka izvršavaju upite koji sadrže virtualne relacije?
 - modifikacijom upita**
 - SUBP ugrađuje elemente definicije virtualne relacije u originalni SQL upit koji koristi virtualnu relaciju - umjesto originalnog SQL upita izvršava se modificirani SQL upit
 - korištenjem materijalizirane virtualne relacije**
 - SUBP fizički pohranjuje sadržaj virtualne relacije. Kada se promijeni sadržaj neke od temeljnih relacija pomoću kojih je virtualna relacija definirana, SUBP automatski mijenja i sadržaj materijalizirane virtualne relacije
 - prednost: virtualne relacije koje se vrlo često koriste, a čiji se sadržaj određuje složenim upitima, ne moraju se svaki puta kada neki korisnik koristi tu virtualnu relaciju ponovno izračunavati
 - nedostatak: ukoliko se temeljne relacije pomoću kojih je virtualna relacija definirana često mijenjaju, pri svakoj izmjeni temeljnih relacija troši se dodatno vrijeme radi izmjene sadržaja virtualne relacije

Implementacija virtualnih relacija modifikacijom upita

- Primjer:

ispit			stud			mjesto	
mbr	predmet	ocj	mbr	ime	prez	pbrStan	nazMjesto
100	Elektronika	3	100	Ivan	Kolar	52100	42000 Varaždin
100	Fizika	2	101	Ana	Horvat	42230	52100 Pula
101	Elektronika	5	102	Jura	Novak	52100	42230 Ludbreg
101	Fizika	2	103	Ana	Ban	52100	
102	Fizika	1					
103	Fizika	5					

studenti koji su položili predmet Fizika

```
CREATE VIEW polFiz AS
SELECT stud.*, ocj
FROM ispit, stud
WHERE ispit.mbr = stud.mbr
AND predmet = 'Fizika'
AND ocj > 1;
```

korisnik obavlja:

```
SELECT * FROM polFiz;
```

SUBP modificira upit

```
SELECT stud.*, ocj
FROM ispit, stud
WHERE ispit.mbr = stud.mbr
AND predmet = 'Fizika'
AND ocj > 1;
```

mbr	ime	prez	pbrStan	ocj
100	Ivan	Kolar	52100	2
101	Ana	Horvat	42230	2
103	Ana	Ban	52100	5

Primjer (nastavak)

- Ispisati prezime, ime i dobivenu ocjenu iz Fizike za studente koji su položili Fiziku, a stanuju u Puli

```
CREATE VIEW polFiz AS
SELECT stud.*, ocj
FROM ispit, stud
WHERE ispit.mbr = stud.mbr
AND predmet = 'Fizika'
AND ocj > 1;
```

korisnik obavlja:

```
SELECT polFiz.prez, polFiz.ime, polFiz.ocj
FROM polFiz, mjesto
WHERE polFiz.pbrStan = mjesto.pbr
AND nazMjesto = 'Pula';
```

SUBP modificira upit

```
SELECT stud.prez, stud.ime, ispit.ocj
FROM ispit, stud, mjesto
WHERE ispit.mbr = stud.mbr
AND predmet = 'Fizika'
AND ocj > 1
AND stud.pbrStan = mjesto.pbr
AND nazMjesto = 'Pula';
```

prez	ime	ocj
Kolar	Ivan	2
Ban	Ana	5

Virtualna relacija: INSERT, UPDATE, DELETE

- virtualne relacije se također mogu koristiti u naredbama INSERT, UPDATE i DELETE

```
CREATE VIEW splitStud AS
SELECT mbr, ime, prez, pbrStan
FROM stud
WHERE pbrStan = 21000;
```

stud			
mbr	ime	prez	pbrStan
100	Ivan	Kolar	31000
101	Ana	Horvat	21000

```
INSERT INTO splitStud
VALUES (102, 'Jure', 'Novak', 21000);
```

```
SELECT * FROM splitStud;
```

mbr	ime	prez	pbrStan
101	Ana	Horvat	21000
102	Jure	Novak	21000

```
INSERT INTO splitStud
VALUES (103, 'Tea', 'Ban', 10000);
```

```
SELECT * FROM splitStud;
```

mbr	ime	prez	pbrStan
101	Ana	Horvat	21000
102	Jure	Novak	21000

n-torka jest unesena u temeljnu relaciju, ali se "ne vidi" u virtualnoj relaciji

```
SELECT * FROM stud;
```

mbr	ime	prez	pbrStan
100	Ivan	Kolar	31000
101	Ana	Horvat	21000
102	Jure	Novak	21000
103	Tea	Ban	10000

Virtualna relacija: INSERT, UPDATE, DELETE

- SUBP ne može promijeniti "sadržaj virtualne relacije" - umjesto toga mora promijeniti sadržaj temeljnih relacija koje se koriste u definiciji te virtualne relacije

ispit		
mbr	predmet	ocj
100	Elektronika	1
100	Fizika	5
101	Elektronika	1
101	Fizika	3

```
CREATE VIEW prosli AS
SELECT * FROM ispit
WHERE ocj > 1;
```

```
CREATE VIEW pali AS
SELECT * FROM ispit
WHERE ocj = 1;
```

korisnik obavlja:

```
UPDATE prosli SET ocj = 4
WHERE mbr = 100
AND predmet = 'Fizika';
```

SUBP modificira upit

```
UPDATE ispit SET ocj = 4
WHERE ocj > 1
AND mbr = 100
AND predmet = 'Fizika';
```

Virtualna relacija: problem migrirajućih n-torki

- n-torka se pojavljuje u virtualnoj relaciji onda kada zadovoljava uvjet iz definicije virtualne relacije
 - n-torka unesena u virtualnu relaciju ili izmijenjena u virtualnoj relaciji može "nestati" iz te virtualne relacije (i eventualno se "pojaviti" u nekoj drugoj virtualnoj relaciji)

ispit

mbr	predmet	ocj
t ₁	100 Elektronika	1
t ₂	100 Fizika	5
t ₃	101 Elektronika	1
t ₄	101 Fizika	3

CREATE VIEW prosli AS

SELECT * FROM ispit

WHERE ocj > 1;

CREATE VIEW pali AS

SELECT * FROM ispit

WHERE ocj = 1;

korisnik obavlja:

UPDATE prosli SET ocj = 1

WHERE mbr = 100

AND predmet = 'Fizika';

INSERT INTO prosli

VALUES (102, 'Elektronika', 1);

- n-torka t₂ je "nestala" iz *prosli* i "pojaviła" se u *pali*
- nova n-torka <102, Elektronika, 1> unesena preko *prosli* se "pojaviła" u *pali*

Virtualna relacija: problem migrirajućih n-torki

- Rješenje:** virtualne relacije koje se koriste u naredbama koje mijenjaju podatke obavezno se kreiraju uz opciju **WITH CHECK OPTION**
 - SUBP tada ne dopušta izmjenu ili unos n-torke putem virtualne relacije ukoliko n-torka nakon obavljanja operacije više ne bi pripadala virtualnoj relaciji putem koje je izmijenjena ili unesena

ispit	mbr	predmet	ocj
	100	Elektronika	1
	100	Fizika	5
	101	Elektronika	1
	101	Fizika	3

```
CREATE VIEW prosli AS
SELECT * FROM ispit
WHERE ocj > 1
WITH CHECK OPTION;
```

```
CREATE VIEW pali AS
SELECT * FROM ispit
WHERE ocj = 1
WITH CHECK OPTION;
```

```
UPDATE prosli SET ocj = 1
WHERE mbr = 100
AND predmet = 'Fizika';
```

pogreška

```
UPDATE prosli SET ocj = 4
WHERE mbr = 100
AND predmet = 'Fizika';
```

O.K.

```
INSERT INTO prosli
VALUES (102, 'Fizika', 1);
```

pogreška

```
INSERT INTO prosli
VALUES (102, 'Fizika', 3);
```

O.K.

```
INSERT INTO pali
VALUES (102, 'Fizika', 3);
```

pogreška

Neizmjenjive virtualne relacije

- SUBP ne može promijeniti "sadržaj virtualne relacije" - umjesto toga mora promijeniti sadržaj temeljnih relacija koje se koriste u definiciji te virtualne relacije
 - ako je virtualna relacija definirana tako da SUBP nije u stanju **jednoznačno** odrediti koje operacije treba obaviti na temeljnim relacijama, tada je virtualna relacija **neizmjenjiva** (*non-updateable*)

polozeniIsipit

mbr	sifPred	ocj
100	1001	5
101	1001	4
102	1001	3
100	1002	2
101	1002	5
100	1003	3
101	1003	3

CREATE VIEW prosjek (sifPred, prosOcJ) AS
SELECT sifPred, AVG(ocj)
FROM polozeniIsipit
GROUP BY sifPred;

SELECT * FROM prosjek;

sifPred	prosOcJ
1001	4.00
1002	3.50
1003	3.00

UPDATE prosjek SET prosOcJ = 4.5
WHERE sifPred = 1001;

?

INSERT INTO prosjek VALUES (1004, 2.5);

?

Izmjenjive virtualne relacije

- Virtualna relacija je izmjenjiva ukoliko u glavnom SELECT dijelu definicije virtualne relacije koristi attribute iz samo jedne temeljne relacije r(R) i pri tome ne sadrži:
 - eliminaciju duplikata pomoću DISTINCT
 - izraze u listi za selekciju (osim izraza koji sadrže samo ime atributa)
 - spajanje ili uniju
 - grupiranje i postavljanje uvjeta nad grupom (GROUP BY i HAVING)
- Prethodno navedena ograničenja se ne odnose na eventualne podupite koji se koriste unutar glavnog SELECT dijela definicije virtualne relacije, ali
 - podupiti ne smiju u svojem FROM dijelu koristiti relaciju r(R)

Primjeri izmjenjivih virtualnih relacija

ispit						stud			
matBr	sifPred	datIsip	ocj	sifNas		matBr	prez	ime	pbrSt
1111	1001	29.01.2006	1	101		1111	Novak	Ivan	10000
1111	1001	05.02.2006	3	101		4444	Ban	Marko	51000
1111	1003	28.06.2006	2	303		1234	Kolar	Petar	23000
1111	1002	27.06.2006	4	202					
1234	1001	29.01.2006	3	202					


```
CREATE VIEW poloziliNista AS  
SELECT * FROM stud  
WHERE NOT EXISTS  
(SELECT * FROM ispit  
WHERE ispit.matBr = stud.matBr  
AND ocj > 1)  
WITH CHECK OPTION;
```

```
CREATE VIEW poloziliBaremDva AS  
SELECT * FROM stud  
WHERE  
(SELECT COUNT(*) FROM ispit  
WHERE ispit.matBr = stud.matBr  
AND ocj > 1) >= 2  
WITH CHECK OPTION;
```

```
CREATE VIEW ispitizAdranal AS  
SELECT matBr  
, sifPred  
, datIsip  
, ocj  
FROM ispit  
WHERE matBr IN (  
SELECT matBr FROM stud  
WHERE pbrSt = 23000)  
WITH CHECK OPTION;
```

Primjeri neizmjenjivih virtualnih relacija

ispit						stud			
matBr	sifPred	datIsip	ocj	sifNas		matBr	prez	ime	pbrSt
1111	1001	29.01.06	1	1111		1111	Novak	Ivan	10000
1111	1001	05.02.06	3	1111		1234	Kolar	Petar	21000
1111	1003	28.06.06	2	3333					
1111	1002	27.06.06	4	2222					
1234	1001	29.01.06	3	2222					


```
CREATE VIEW stud1 (ime_prez) AS  
SELECT ime || prez  
FROM stud;
```

```
CREATE VIEW poloziliNesto AS  
SELECT DISTINCT matBr  
FROM ispit  
WHERE ocj > 1;
```

```
CREATE VIEW boljiOdProsjeaka AS  
SELECT * FROM ispit  
WHERE ocj >  
(SELECT AVG(ocj)  
FROM ispit);
```

```
CREATE VIEW ispitizAdranal2 AS  
SELECT ispit.matBr  
, sifPred  
, datIsip  
, ocj  
FROM ispit, stud  
WHERE ispit.matBr = stud.matBr  
AND pbrSt = 23000;
```

↑

usporediti s izmjenjivom virtualnom relacijom **ispitizAdranal** s prethodne stranice!

Implementacija eksternih shema pomoću virtualnih relacija

- konceptualna shema: baza podataka u banci

klijent	jmbg	ime	prez	uplataIsplata	brRac	vrijeme	valuta	iznos
123456	Ana	Horvat			1001	7.8.2007 08:20	HRK	15.00
654321	Ivan	Novak			1002	9.4.2006 12:31	EUR	-100.21
123654	Tea	Kolar			1001	6.5.2007 14:15	HRK	452.15
racun	brRac	jmbgVI	tipRac	datRac				
1001	123456	kunski	7.2.2007		1004	5.5.2007 16:42	HRK	1200.00
1002	123456	devizni	1.3.2006		1004	9.9.2005 10:15	HRK	-350.50
1003	654321	devizni	4.8.2004		1002	7.2.2007 15:01	EUR	235.20
1004	123654	kunski	8.9.2005		1003	1.4.2005 12:44	USD	2750.00
					1001	1.9.2007 12:19	HRK	-250.35
					1004	8.2.2006 11:55	HRK	420.00

- za različite kategorije korisnika (aplikacija) definiraju se različite eksterne sheme
 - aplikacija za otvaranje računa
 - aplikacija za deviznu uplatu/isplatu
 - aplikacija za kunsku uplatu/isplatu
 - aplikacija za pregled trenutnog stanja sredstava u banci

Implementacija eksternih shema pomoću virtualnih relacija

```
CREATE VIEW devRacun AS
SELECT * FROM racun
WHERE tipRac = 'devizni'
WITH CHECK OPTION;
```

```
CREATE VIEW devUplIspl AS
SELECT * FROM uplataIsplata
WHERE valuta <> 'HRK'
WITH CHECK OPTION;
```

```
CREATE VIEW kunRacun AS
SELECT * FROM racun
WHERE tipRac = 'kunski'
WITH CHECK OPTION;
```

```
CREATE VIEW kunUplIspl AS
SELECT * FROM uplataIsplata
WHERE valuta = 'HRK'
WITH CHECK OPTION;
```

```
CREATE VIEW pregledStanja
(valuta, ukupno) AS
SELECT valuta, SUM(iznos)
FROM uplataIsplata
GROUP BY valuta;
```

- eksterne sheme za aplikacije
 - za otvaranje računa: **klijent**, **racun**
 - za deviznu uplatu/isplatu: **devRacun**, **devUplIspl**
 - za kunsku uplatu/isplatu: **kunRacun**, **kunUplIspl**
 - za pregled trenutnog stanja sredstava: **pregledStanja**

Implementacija eksternih shema pomoću virtualnih relacija

eksterne sheme

otvaranje računa	devizna uplata/isplata	kunska uplata/isplata	pregled stanja
klijent	devUplIspl	kunUplIspl	pregledStanja
jmbg ime prez	brRac vrijeme valuta iznos	brRac vrijeme valuta iznos	valuta ukupno
...	?	?	?
racun	devRacun	kunRacun	
brRac jmbgVI tipRac datRac	brRac jmbgVI tipRac datRac	brRac jmbgVI tipRac datRac	
...	?	?	

preslikavanja: konceptualna ↔ eksterne sheme

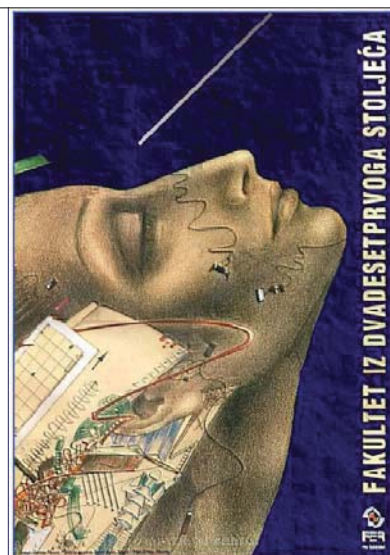
konceptualna shema

racun	klijent	uplataIsplata
brRac jmbgVI tipRac datRac	jmbg ime prez	brRac vrijeme valuta iznos
...

Baze podataka

Predavanja
svibanj 2008.

12. ER model baze podataka (1. dio)



Primjer normalizacije

- Zadana je relacijska shema:
ISPIT = { matBr, prez, ime, sifPred, nazPred, datIspr, ocj, sifNas, prezNas }
i trenutna vrijednost relacije ispit(ISPIT):

ispit (ISPIT)

matBr	prez	ime	sifPred	nazPred	datIspr	ocj	sifNas	prezNas
1111	Novak	Ivan	1001	Mat-1	29.01.06	1	1111	Pašić
1111	Novak	Ivan	1001	Mat-1	05.02.06	3	1111	Pašić
1111	Novak	Ivan	1003	Fiz-1	28.06.06	2	3333	Horvat
1111	Novak	Ivan	1002	Mat-2	27.06.06	4	2222	Brnetić
1234	Kolar	Petar	1001	Mat-1	29.01.06	3	2222	Brnetić

- funkcijske zavisnosti odrediti na temelju značenja podataka
- odrediti primarni ključ relacije (tako da bude zadovoljen uvjet 1NF prema kojem neključni atributi funkcijski ovise o ključu)
- postupno normalizirati relacijsku shemu ISPIT na 2NF i 3NF

Primjer normalizacije

student (STUDENT)

matBr	prez	ime
1111	Novak	Ivan
1234	Kolar	Petar

$K_{STUDENT} = \{ matBr \}$

predmet (PREDMET)

sifPred	nazPred
1001	Mat-1
1003	Fiz-1
1002	Mat-2

$K_{PREDMET} = \{ sifPred \}$

nastavnik (NASTAVNIK)

sifNas	prezNas
1111	Pašić
3333	Horvat
2222	Brnetić

$K_{NASTAVNIK} = \{ sifNas \}$

ispit₃ (ISPIT₃)

matBr	sifPred	datIspr	ocj	sifNas
1111	1001	29.01.06	1	1111
1111	1001	05.02.06	3	1111
1111	1003	28.06.06	2	3333
1111	1002	27.06.06	4	2222
1234	1001	29.01.06	3	2222

$K_{ISPIT_3} = \{ matBr, sifPred, datIspr \}$

- Shema baze podataka STUSLU:

$STUSLU = \{ STUDENT, PREDMET, NASTAVNIK, ISPIT_3 \}$

Implementacija: SQL

```
CREATE TABLE student (  
  matBr INTEGER  
  , prez CHAR(20)  
  , ime CHAR(20)  
  , PRIMARY KEY (matBr));
```

```
CREATE TABLE predmet (  
  sifPred INTEGER  
  , nazPred CHAR(20)  
  , PRIMARY KEY (sifPred));
```

```
CREATE TABLE nastavnik (  
  sifNas INTEGER  
  , prezNas CHAR(20)  
  , PRIMARY KEY (sifNas));
```

```
CREATE TABLE ispit (  
  , matBr INTEGER REFERENCES student (matBr)  
    ON DELETE CASCADE  
  , sifPred INTEGER REFERENCES predmet (sifPred)  
  , datIsp DATE  
  , ocj SMALLINT CHECK (ocj BETWEEN 1 AND 5)  
  , sifNas INTEGER REFERENCES nastavnik (sifNas)  
    ON DELETE SET NULL  
  , PRIMARY KEY (matBr, sifPred, datIsp));
```

OBLIKOVANJE MODELA BAZE PODATAKA

ER model $n:1$ i $1:n$ i $1:1$ i $1:m$ i $m:n$ Model entiteti-veze

- postrelacijski model
- zadržava dobre karakteristike relacijskog modela
- omogućuje eksplicitni prikaz veza koje u sebi sadrže važne semantičke informacije

Literatura:

- P.P.Chen:
The Entity-Relationship Model - Toward a Unified View of Data,
ACM Transactions on Database Systems, Vol. 1, No. 1, 1976
- T. J. Teorey:
Database Modeling & Design, Morgan Kaufmann, 1999

Entiteti, veze, uloge

Entitet

- bilo što, što ima suštinu ili bit, ima jasnoću kao činjenica ili ideja, posjeduje značajke s pomoću kojih se može razlučiti od svoje okoline

Skup entiteta E_i (entityset)

- Slični entiteti se grupiraju u skupove entiteta

Skup veza R_i (relationship set)

- matematička relacija između n entiteta:

$$R_i \subseteq E_1 \times E_2 \times E_3 \times \dots \times E_n$$

ili $R_i = \{ (e_1, e_2, \dots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n \}$

n-torka $(e_1, e_2, e_3, \dots, e_n)$, naziva se vezom.

Uloga (role)

- funkcija koju skup entiteta obavlja u skupu veza.

Skup vrijednosti, atribut

- Informacije o entitetu ili vezi izražavaju se s pomoću parova **atribut-vrijednost**
- Vrijednosti su klasificirane u skupove vrijednosti V_i .
- **Atribut** je funkcija koja preslikava iz skupa entiteta ili skupa veza u skup vrijednosti ili Kartezijev produkt skupova vrijednosti:

$$f: E_i \rightarrow V_i$$

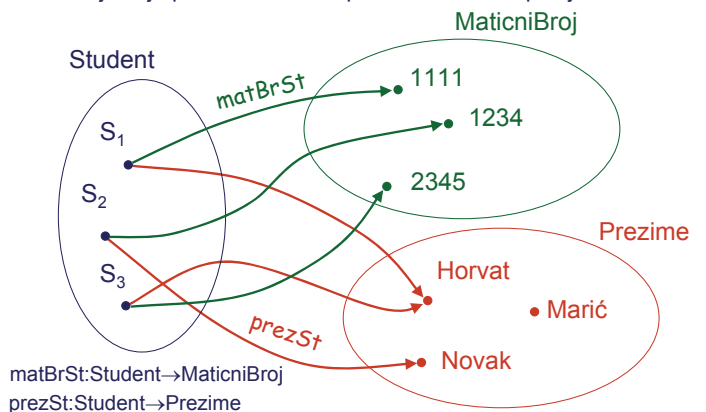
$$f: E_i \rightarrow V_{i_1} \times V_{i_2} \times \dots \times V_{i_n}$$

$$f: R_i \rightarrow V_i$$

$$f: R_i \rightarrow V_{i_1} \times V_{i_2} \times \dots \times V_{i_n}$$

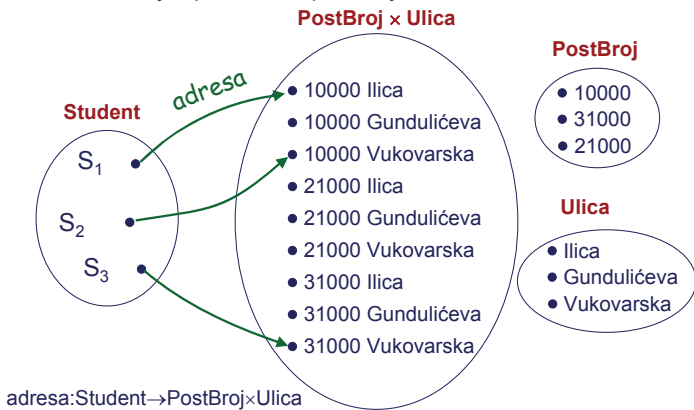
Atributi entiteta

- funkcija koja preslikava sa skupa entiteta na skup vrijednosti ...



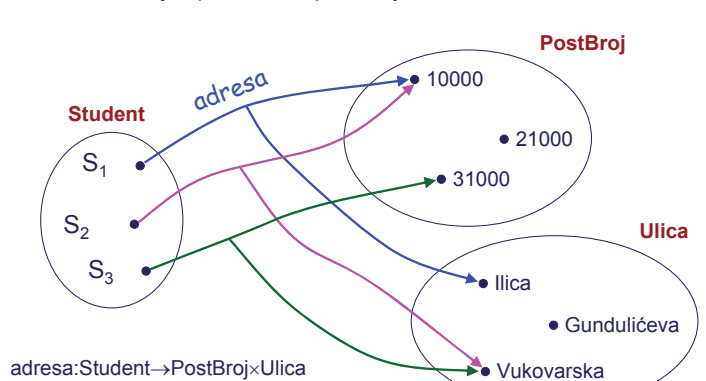
Atributi entiteta

... ili na Kartezijev produkt skupova vrijednosti



Atributi entiteta

... ili na Kartezijev produkt skupova vrijednosti



Terminologija

Chen:

entitet, **skup entiteta**
veza, **skup veza**

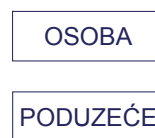
Teorey:

instanca entiteta, **entitet**
(*entity instance*)
(*entity occurrence*)
instanca veze, **veza**
(*relationship instance*)
(*relationship occurrence*)

Grafički prikaz entiteta i veza

- entitet se grafički prikazuje pravokutnikom unutar kojeg se nalazi ime entiteta
- veza se grafički prikazuje romбом unutar kojeg se nalazi ime veze

Entiteti



Veza



Atributi entiteta

- atribut entiteta se grafički prikazuju ovalom unutar kojeg se upisuje ime atributa
- atribut (ili atributi) **primarnog** ključa se potcrtavaju



- povećanjem broja atributa, dijagram postaje nepregledan
 - atributi se tada ne prikazuju grafički - umjesto toga, uz dijagram se prilažu **sheme entiteta**

Shema entiteta:

NASTAVNIK = sifNast, jmbgNast, imeNast, prezNast ili

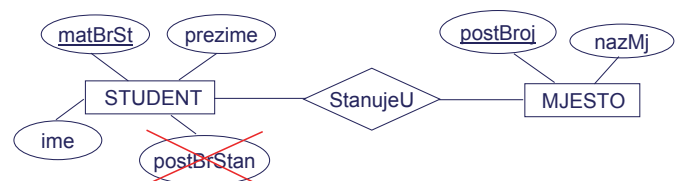
PK = { sifNast }

NASTAVNIK
sifNast
jmbgNast
imeNast
prezNast
 $K_1 = \{ \text{sifNast} \}$
 $K_2 = \{ \text{jmbgNast} \}$
 $PK = K_1$

Vlastiti atributi entiteta

Entiteti se opisuju samo vlastitim atributima

- vlastiti atribut entiteta** je atribut koji opisuje znanja o entitetu koja se pripisuju isključivo samom entitetu, a nikako vezi s drugim entitetima



- isključivo identifikacijski slabi entiteti, osim svojih vlastitih atributa, posjeduju i attribute primarnog ključa entiteta vlasnika

Regularni i slabi entiteti

- regularni** entitet je entitet koji može postojati sam za sebe
- slabi** entiteti (engl. *weak entity*) ne postoje ukoliko ne postoji i neki drugi entitet (entitet vlasnik)
- Slabi entitet se grafički prikazuje dvostruko uokvirenim pravokutnikom, sa strelicom koja dolazi iz smjera veze koja ga povezuje s entitetom vlasnikom

slabi entitet čiji je vlasnik entitet DJELATNIK



- slabi entiteti, osim što su **egzistencijalno slabi**, također mogu biti i **identifikacijski slabi**
 - kod određivanja identifikatora nisu im dovoljni vlastiti atributi
 - za identifikaciju se koriste i ključni atributi entiteta vlasnika

Identifikacijski slabi entiteti (primjer)

- entitet DIJETE, osim što je egzistencijalno slab, također je i identifikacijski slab



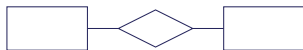
- entitet PUTOVNICA je egzistencijalno slab (nije identifikacijski slab)



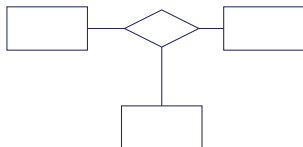
Stupanj veze

- broj entiteta koje povezuje dotična veza
- veza može biti unarna(refleksivna), binarna, ternarna, itd.
 - unarna ili refleksivna veza - veza je definirana nad jednim entitetom koji u vezi ima dvije različite uloge

binarna



ternarna



refleksivna



Spojnost veze (one-to-many)

- spojnost veze** opisuje ograničenje preslikavanja pojedinačnih entiteta koje veza povezuje
- vrijednosti spojnosti: jedan (*one*), više (*many*)
- koriste se oznake 1, N ili rasponi, npr. 0..1, 1..N, 1..2, itd.



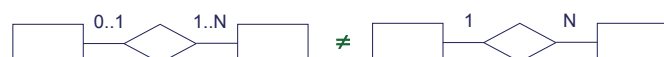
- jedan djelatnik radi na jednom projektu, na jednom projektu radi N djelatnika



- jedan djelatnik radi na nula (niti jednom) ili jednom projektu, na jednom projektu radi između nula (niti jedan) i više djelatnika

Spojnost veze (one-to-many)

- radi pojednostavljenja
 - spojnost 0..N se često označava samo oznakom N
 - spojnost 1..1 se često označava samo oznakom 1

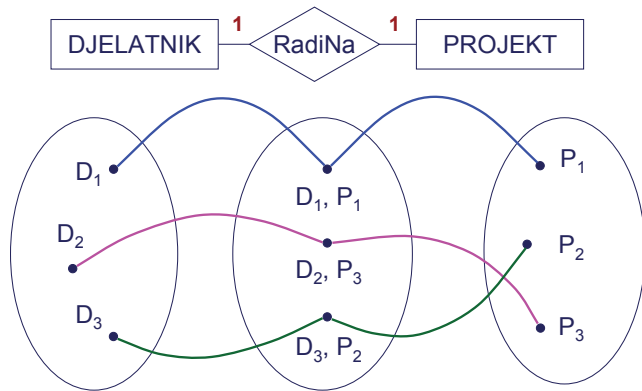


Preslikavanje (mapping)

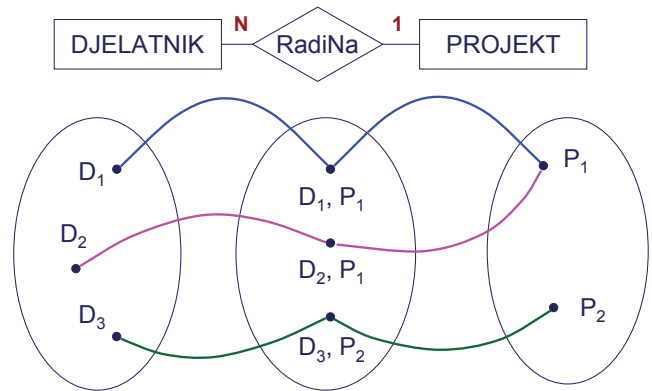
- preslikavanje** - međusobni odnos entiteta u vezi
- kod binarnih veza moguća su **preslikavanja 1:1** (jedan-prema-jedan), **1:N** (jedan-prema-više), **N:1** (više-prema-jedan), **N:N** (više-prema-više).



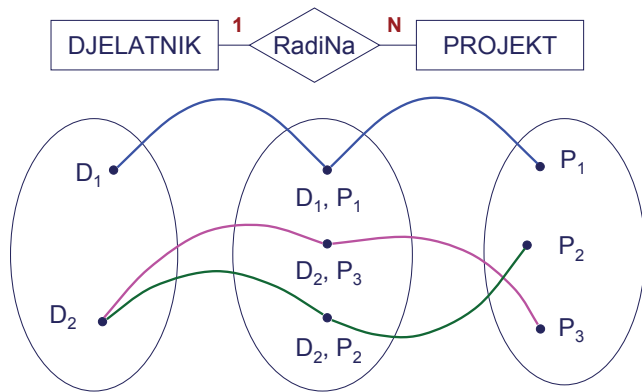
Preslikavanje 1:1



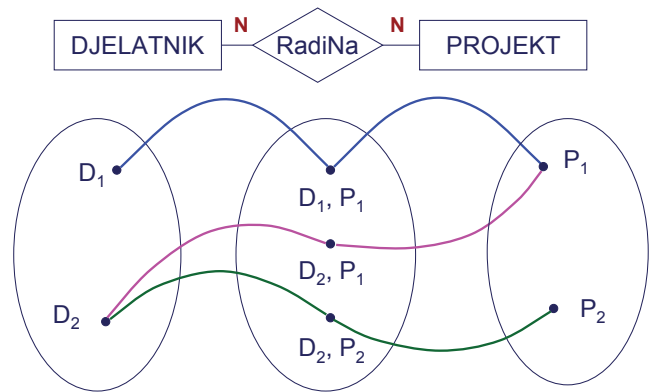
Preslikavanje N:1



Preslikavanje 1:N



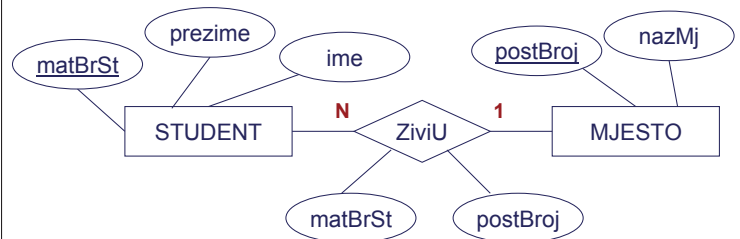
Preslikavanje N:N



Atributi veza

- Schema veze sadrži ključeve entiteta koje povezuje, te vlastite attribute
- Atribut veze se grafički prikazuje ovalom unutar kojeg se upisuje ime atributa

Atributi veza



shema entiteta:

STUDENT = matBrSt, prezime, ime

MJESTO = postBroj, nazMj

shema veze:

ZiviU = matBrSt, postBroj

Koji atributi čine ključ veze?

Ključevi veza

- Povezanost entiteta opisuje se kao odnos među ključevima entiteta
- Ključevi veza definirani su s pomoću ključeva entiteta koje povezuju i njihovih spojnosti

Definicija 1. (Teorey)

U vezi koja povezuje entitete

$$E_1, \dots, E_k, \dots, E_m,$$

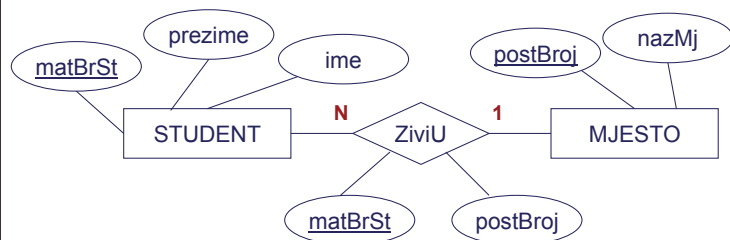
spojnost = 1 entiteta E_k znači da za svaku vrijednost svih entiteta E_1, \dots, E_m , osim E_k , uvijek postoji točno jedna vrijednost od E_k .

➔ može se reći da tada vrijedi funkcijska zavisnost:

$$\bigcup_{j=1}^m K_j \setminus K_k \rightarrow K_k$$

gdje su skupovi K_j , ($j = 1, \dots, m$) ključevi entiteta E_1, \dots, E_m

Ključevi veza



shema entiteta:

STUDENT = matBrSt, prezime, ime

MJESTO = postBroj, nazMj

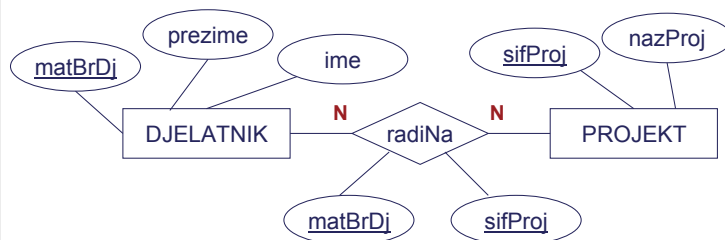
shema veze:

ZiviU = matBrSt, postBroj

Iz definicije 1:

$\text{matBrSt} \rightarrow \text{postBroj}$

Ključevi veza

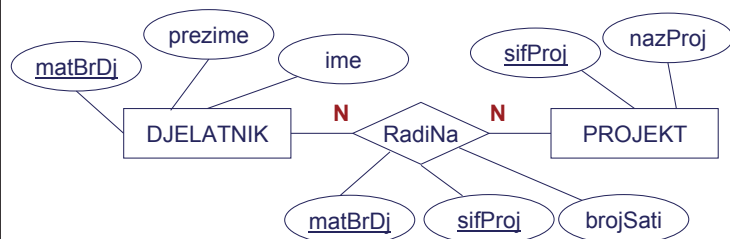


DJELATNIK = matBrDj, prezime, ime

PROJEKT = sifProj, nazProj

RadiNa = matBrDj, sifProj

Vlastiti atributi veza



DJELATNIK = matBrDj, prezime, ime

PROJEKT = sifProj, nazProj

RadiNa = matBrDj, sifProj, brojSati

ključ veze funkcijski određuje
vlastite attribute veze:
 $\text{matBrDj}, \text{sifProj} \rightarrow \text{brojSati}$

Ključ veze - dodatna razmatranje

- iz definicije 1. proizlazi da se ključ veze sastoji isključivo od ključeva entiteta koje povezuje (svih ili samo nekih, ovisno o spojnostima)
- Međutim, u nekim slučajevima ključ može sadržavati i neke druge attribute.



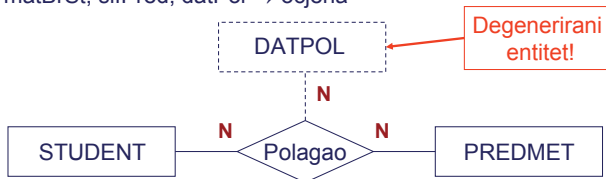
STUDENT = matBrSt, prezime, ime

PREDMET = sifPred, nazPred

Položio = matBrSt, sifPred, ocjena

Ključ veze - dodatna razmatranje

- ako se želi evidentirati sva polaganja ispita
matBrSt, sifPred \nrightarrow ocjena
- potrebno je uvesti atribut datPol (datum polaganja):
matBrSt, sifPred, datPol \rightarrow ocjena



STUDENT = matBrSt, prezime, ime
 PREDMET = sifPred, nazPred
 Polagao = matBrSt, sifPred, datPol, ocjena

Ključ veze - dodatna razmatranje

- druga mogućnost - veza postaje entitet:



STUDENT = matBrSt, prezime, ime
 PREDMET = sifPred, nazPred
 ISPIT = matBrSt, sifPred, datPol, ocjena
 StudIsp = matBrSt, sifPred, datPol
 PredIsp = matBrSt, sifPred, datPol

Veza 1:N \rightarrow preslikavanje u relacijski model



DJELATNIK = matBrDj, prezime, ime
 MJESTO = postBr, nazMjesto
 Stanuje = matBrDj, postBr, adresa

Relacijske sheme opisuju entitete (veze postaju entiteti)

DJELATNIK = matBrDj, prezime, ime
 MJESTO = postBr, nazMjesto
 Stanuje = matBrDj, postBr, adresa

Unija relacijskih shema s jednakim ključevima

DJELATNIK = matBrDj, prezime, ime, postBr, adresa
 MJESTO = postBr, nazMjesto

Veza N:N \rightarrow preslikavanje u relacijski model



DJELATNIK = matBrDj, prezime, ime
 PROJEKT = sifProj, nazProj
 RadiNa = matBrDj, sifProj, brojSati

Relacijske sheme opisuju entitete (veze postaju entiteti)

DJELATNIK = matBrDj, prezime, ime
 PROJEKT = sifProj, nazProj
 RadiNa = matBrDj, sifProj, brojSati

Primjer: zašto je važno ispravno odrediti vlastite attribute entiteta i veza?

- Entiteti se opisuju samo vlastitim atributima: **vlastiti atribut entiteta** je atribut koji opisuje znanja o entitetu koja se pripisuju isključivo samom entitetu, a nikako vezi s drugim entitetima



DJELATNIK = matBrDj, prezime, ime, brojSati \rightarrow nije vlastiti atribut
 PROJEKT = sifProj, nazProj
 RadiNa = matBrDj, sifProj

- Ako se preslikavanje promijeni u N:N



matBrDj \nrightarrow brojSati

atribut brojSati se iz entiteta DJELATNIK mora premjestiti u vezu RadiNa

Primjer: zašto je važno ispravno odrediti vlastite attribute entiteta i veza?



DJELATNIK = matBrDj, prezime, ime
 PROJEKT = sifProj, nazProj
 RadiNa = matBrDj, sifProj, brojSati \rightarrow vlastiti atribut

- Ako se preslikavanje promijeni u N:N



DJELATNIK = matBrDj, prezime, ime
 PROJEKT = sifProj, nazProj
 RadiNa = matBrDj, sifProj, brojSati

Paralelne veze



STUDENT = matBrSt, prezime, ime

MJESTO = postBroj, nazMjesto

Uloge: PREBIVALIŠTE
BORAVIŠTE

Prebiva = matBrSt, postBroj PostBrojPreb

Boravi = matBrSt, postBroj PostBrojBor

Paralelne veze → relacijski model

- Unija shema s jednakim ključevima:

MJESTO = postBroj, nazMjesto

STUDENT = matBrSt, prezime, ime, ~~postBroj~~, ~~postBroj~~

STUDENT = matBrSt, prezime, ime, ~~postBrojBor~~, ~~postBrojPreb~~
+ pravila integriteta

Zadatak: Ispisati prezime i ime studenta, poštanski broj i naziv mjesta boravka te poštanski broj i naziv mjesta prebivališta

```
SELECT student.*
, boraviste.nazMjesto AS nazMjestoBoraviste
, prebivaliste.nazMjesto AS nazMjestoPrebivaliste
FROM student
INNER JOIN mjesto AS boraviste
ON boraviste.postBroj = student.postBrojBor
INNER JOIN mjesto AS prebivaliste
ON prebivaliste.postBroj = student.postBrojPreb
```

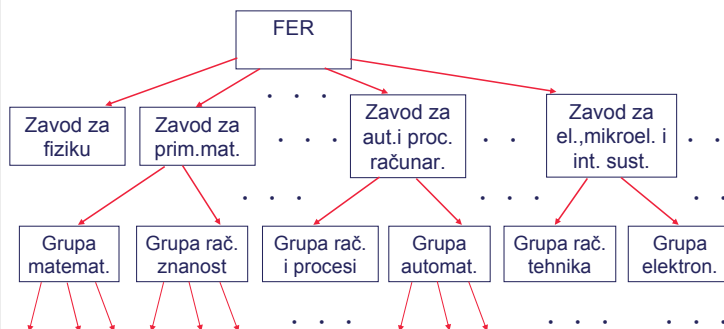
Problem

Kako opisati organizacijsku strukturu poduzeća?

- Organizacijske jedinice opisane su svojom šifrom i nazivom
- Organizacijske jedinice međusobno su povezane
 - kako?
 - među njima postoji hijerarhijski odnos!
 - kolika je dubina stabla (broj razina)?
 - promjenjiva!
- Kako opisati hijerarhiju - stablo promjenjive dubine?
- Čvorovi stabla su opisani na isti način (šifra, naziv)

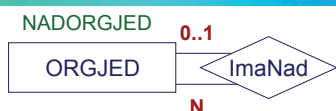
Homogeno stablo

Primjer: Organizacijska struktura



Čvorovi stabla imaju jednaku strukturu: ORGJED= sifOrgJed, nazOrgJed

Refleksivne veze - preslikavanje 1:N



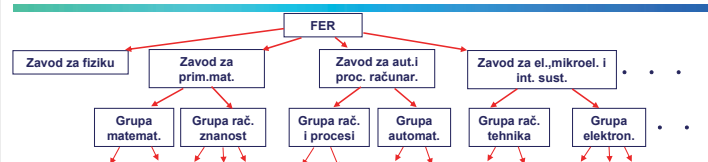
ORGJED = sifOrgJed, nazOrgJed

ImaNad = ~~sifOrgJed~~, ~~sifOrgJed~~

ImaNad = sifOrgJed, sifNadOrgJed

Preimenovati
jedan od atributa !

Homogeno stablo



ORGJED

sifOrgJed	nazOrgJed
1	FER
9	Zavod za prim.mat.
21	Grupa Matematika
33	Grupa Rač. Znanost
49	Zavod za aut. i proc. rač.
53	Grupa Automatika
67	Grupa RASIP
73	Zavod za el.mikroel. i int.
89	Grupa Rač. tehnika

imaNad

sifOrgJed	sifNadOrgJed
9	1
21	9
33	9
49	1
53	49
67	49
73	1
89	73

Refleksivne veze 1:N → relacijski model

- Unija shema s jednakim ključevima:

ORGJED = sifOrgJed, nazOrgJed
imaNad = sifOrgJed, sifNadOrgJed

ORGJED = sifOrgJed, nazOrgJed, sifNadOrgJed
□ pravila integriteta

Zadatak: Ispisati naziv organizacijske jedinice i naziv njezine nadređene organizacijske jedinice (ukoliko postoji)

```
SELECT orjed.nazOrgJed, nadorjed.nazOrgJed
FROM orjed
LEFT OUTER JOIN orjed AS nadorjed
ON orjed.sifNadOrgJed = nadorjed.sifOrgJed
```

to je šifra organizacijske jedinice?

- Govoreća šifra □ šifra koja označava poziciju organizacijske jedinice unutar poduzeća??

npr. □□□ZZZ

□□ □ šifra sektora

□□ □ šifra odjela

ZZZ □ šifra odsjeka

- to se dešava prilikom reorganizacije?

→ moraju se promijeniti šifre organizacijskih jedinica!

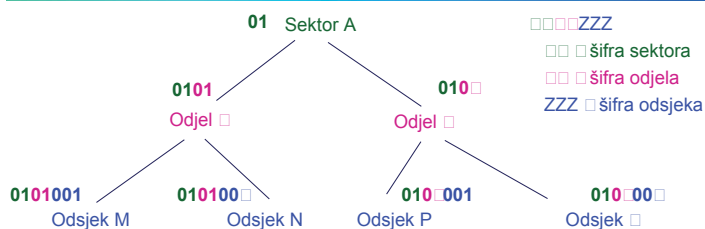
- to se dešava kada broj odjela preraste 100??

→ moraju se promijeniti šifre organizacijskih jedinica!

→ šifra organizacijske jedinice NE SMIJE BITI GOLOREĆA!

→ To vrijedi i za sve ostale šifre i identifikatore!!!

to je šifra organizacijske jedinice?



ORGJED	sifOrgJed	nazOrgJed
01	01	Sektor A
0101	0101	Odjel M
0102	0102	Odjel N
0101001	0101001	Odsjek M
0101002	0101002	Odsjek N
0102001	0102001	Odsjek P
0102002	0102002	Odsjek Q

□ to kada Odsjek P zbog reorganizacije iz Odjela N preseli u Odjel M?

□ to kada broj odjela preraste broj 99?

blikovanje ER modela



Oblikovanje ER modela

- definiranje entiteta**
 - ime, opis, komentar
- definiranje veza**
 - ime, opis, komentar, entiteti koje povezuje, preslikavanje
- definiranje atributa entiteta**
 - za svaki atribut: ime, opis, komentar, domena
 - definirati ključeve, provjeriti da li zadovoljava 3NF
- definiranje atributa veza**
 - za svaki atribut: ime, opis, komentar, domena
 - definirati ključeve, provjeriti da li zadovoljava 3NF

P □ □ □ P □ K □ E □ □ ER □ □ □ □ N □

odel baze podataka

SADRŽAJ OPISE

- entiteta
- veza
- atributa entiteta
- atributa veza

KARAKTERISTIKE DOBROG MODELA

- opisuje suštinu, prirodu stvari, neovisan o postojećem stanju
- sveobuhvatan
- neredundantan
- fleksibilan
- razumljiv - korisnicima i informatičarima

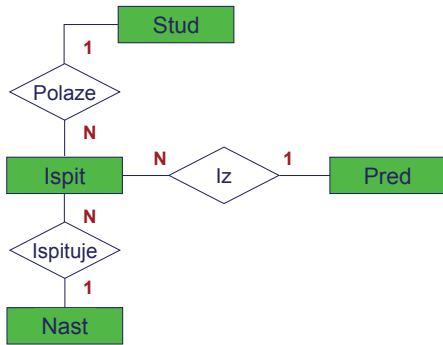
POSEBNO OBRATITI PAŽNJU NA:

- različito shvaćanje istih stvari - kupac, dobavljač → poslovni partner
- praćenje promjena u vremenu - stipendist, djelatnik, penzioner
- jednakost - uopćavanje - različiti odjeli i pojedinci mogu iste ili slične stvari shvaćati različito

Primjer: odel baze podataka za studentsku službu

- Oblikovati model baze podataka koja će omogućiti praćenje podataka o studentima, predmetima, nastavnicima i polaganjima ispita

Primjer (nastavak)

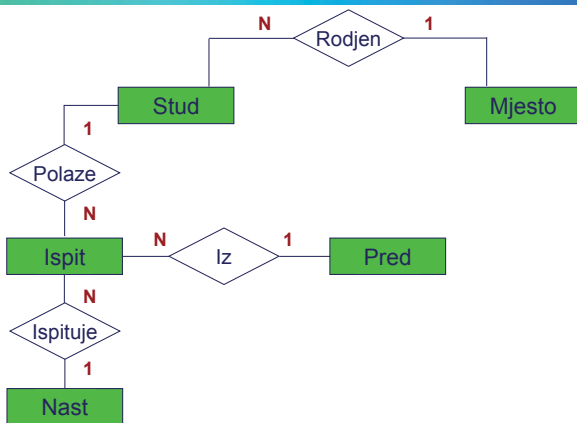


Primjer (nastavak)

Stud = matBrStud, prezStud, imeStud, datRodStud

MJESTO ROĐENJA STUDENTA ???

Primjer (nastavak)



Primjer (nastavak)

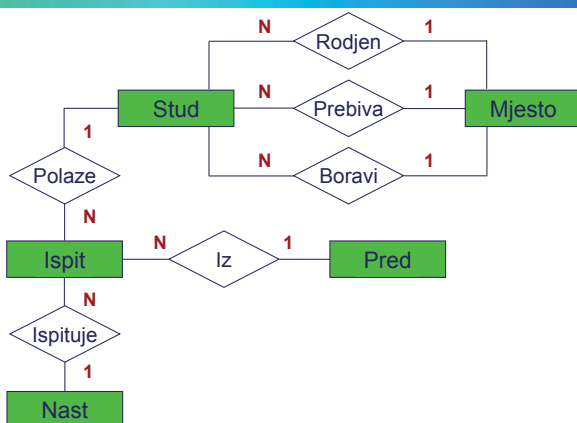
Stud = matBrStud, prezStud, imeStud, datRodStud

Mjesto = pbrMjesto, nazMjesto

PREBIVAJE STUDENTA ???

BORAVIJE STUDENTA ???

Primjer (nastavak)



Primjer (nastavak)

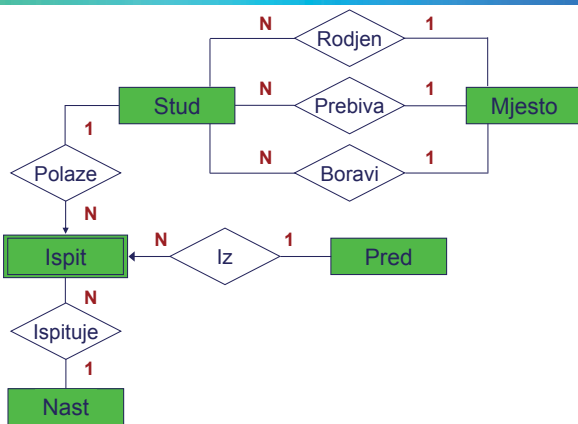
Stud = matBrStud, prezStud, imeStud, datRodStud, datUpisFERStud, rangKlasIspitStud, eMailStud

Mjesto = pbrMjesto, nazMjesto

Ispit = matBrStud, sifraPred, datumIspit, ocjena

SCABI ENTITET !!!!

Primjer (nastavak)



Primjer (nastavak)

Stud = matBrStud, prezStud, imeStud, datRodStud, datUpisFERStud, rangKlasIspitStud, eMailStud

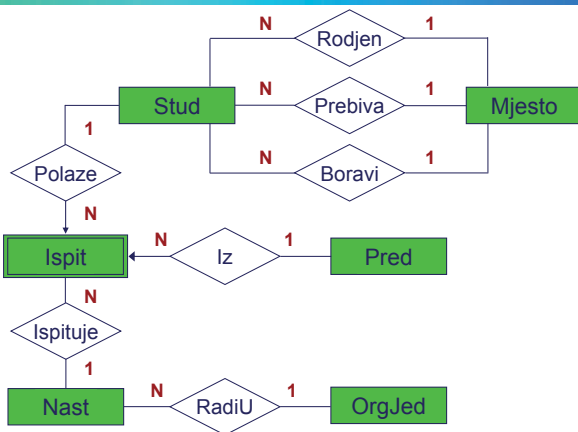
Mjesto = pbrMjesto, nazMjesto

Ispit = matBrStud, sifraPred, datumIspit, ocjena

Nast = sifraNast, prezNast, imeNast

ORGANIZACIJSKA JEDINICA ???

Primjer (nastavak)



Primjer (nastavak)

Stud = matBrStud, prezStud, imeStud, datRodStud, datUpisFERStud, rangKlasIspitStud, eMailStud

Mjesto = pbrMjesto, nazMjesto

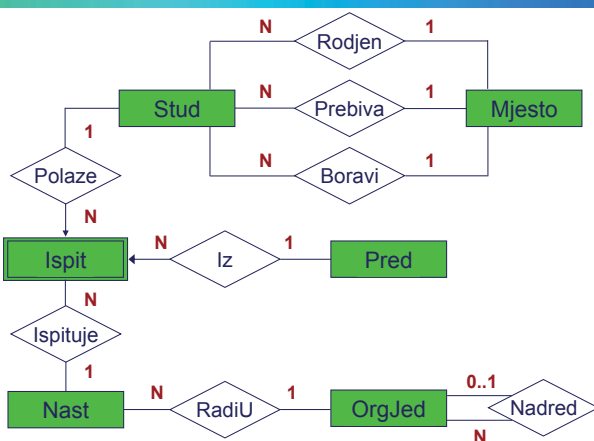
Ispit = matBrStud, sifraPred, datumIspit, ocjena

Nast = sifraNast, prezNast, imeNast

OrgJed = sifraOrgJed, nazivOrgJed

NADREĐENA ORGANIZACIJSKA JEDINICA ???

Primjer (nastavak)



Primjer (nastavak)

Stud = matBrStud, prezStud, imeStud, datRodStud, datUpisFERStud, rangKlasIspitStud, eMailStud

Mjesto = pbrMjesto, nazMjesto

Ispit = matBrStud, sifraPred, datumIspit, ocjena

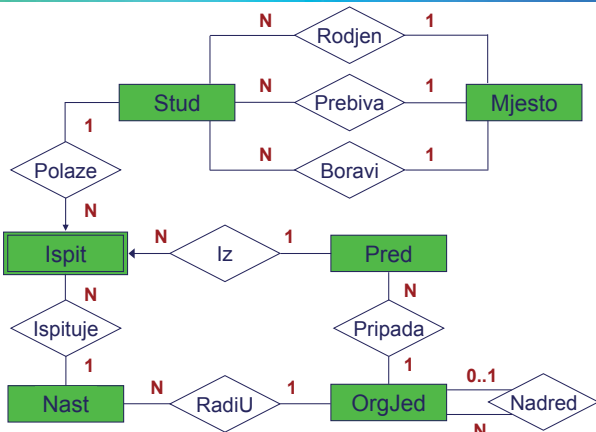
Nast = sifraNast, prezNast, imeNast, eMailNast, URNast

OrgJed = sifraOrgJed, nazivOrgJed

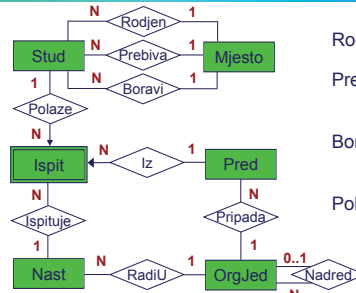
Pred = sifraPred, kraticaPred, nazivPred, URNPred

PREDMET PRIPADA ORGANIZACIJSKOJ JEDINICI ???

Primjer (nastavak)



Primjer (nastavak) - pis veza



Rodjen = matBrStud, postBrMjRodStud
 Prebiva = matBrStud, postBrMjPrebStud, adresaMjPrebStud
 Boravi = matBrStud, postBrMjBorStud, adresaMjBorStud
 Polaze = matBrStud, sifraPred, datumIspit

Iz = matBrStud, sifraPred, datumIspit
 Ispituje = matBrStud, sifraPred, datumIspit, sifraNast
 RadiU = sifraNast, sifraOrgJed
 Pripada = sifraPred, sifraOrgJed
 Nadred = sifraOrgJed, sifraNadOrgJed

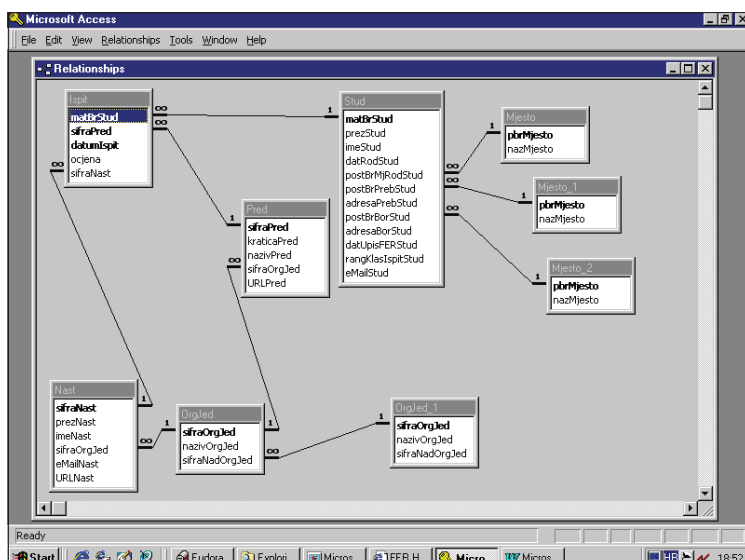
→ Relacijski model

Stud = matBrStud, prezStud, imeStud, datRodStud, datUpisFERStud, rangKlasIspitStud, eMailStud
 Mjesto = pbrMjesto, nazMjesto
 Ispit = matBrStud, sifraPred, datumIspit, ocjena
 Nast = sifraNast, prezNast, imeNast, eMailNast, URNast
 OrgJed = sifraOrgJed, nazivOrgJed
 Pred = sifraPred, kraticaPred, nazivPred, URLPred
 Rodjen = matBrStud, postBrMjRodStud
 Prebiva = matBrStud, postBrMjPrebStud, adresaMjPrebStud
 Boravi = matBrStud, postBrMjBorStud, adresaMjBorStud
 Polaze = matBrStud, sifraPred, datumIspit
 Iz = matBrStud, sifraPred, datumIspit
 Ispituje = matBrStud, sifraPred, datumIspit, sifraNast
 RadiU = sifraNast, sifraOrgJed
 Pripada = sifraPred, sifraOrgJed
 Nadred = sifraOrgJed, sifraNadOrgJed

→ Relacijski model

Unija shema s jednakim ključevima

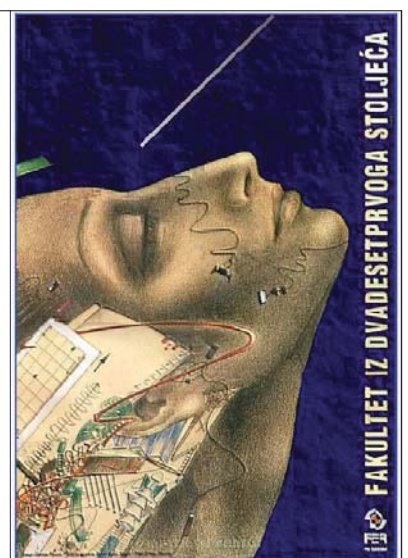
Stud = matBrStud, prezStud, imeStud, datRodStud, datUpisFERStud, rangKlasIspitStud, eMailStud, postBrMjRodStud, postBrMjPrebStud, adresaMjPrebStud, postBrMjBorStud, adresaMjBorStud
 Mjesto = pbrMjesto, nazMjesto
 Ispit = matBrStud, sifraPred, datumIspit, ocjena, sifraNast
 Nast = sifraNast, prezNast, imeNast, eMailNast, URNast, sifraOrgJed
 OrgJed = sifraOrgJed, nazivOrgJed, sifraNadOrgJed
 Pred = sifraPred, kraticaPred, nazivPred, URLPred, sifraOrgJed



Baze podataka

Predavanja
svibanj 2008.

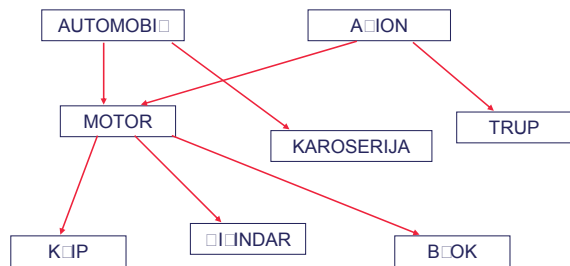
1. ER model baze podataka
(dio)



Homogena mreža

Primjer: Sastavnica

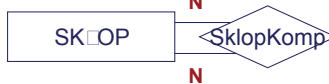
AUTOMOBIL: MOTOR, KAROSERIJA
MOTOR: KLIP, CILINDAR, BLOK
AVION: MOTOR, TRUP



Čvorovi u mreži imaju jednaku strukturu: SKLOP = sifSklop, nazSklop

Refleksivne veze - preslikavanje N:N

KOMPONENTA



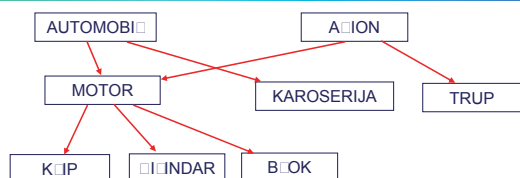
SKLOP = sifSklop, nazSklop

SklopKomp = sifSklop, sifSklop

Preimenovati jedan od atributa !

SklopKomp = sifSklop, sifKomp

Refleksivne veze - preslikavanje N:N



Sklop		sklopKomp	
sifSklop	nazSklop	sifSklop	sifKomp
17	Automobil	17	19
19	Motor	17	21
21	Karoserijska	19	37
37	Klip	19	49
49	Cilindar	19	52
52	Blok	64	19
64	Avion	64	82
82	Trup		

Refleksivne veze N:N → relacijski model

SKLOP = sifSklop, nazSklop

SklopKomp = sifSklop, sifKomp

pravila integriteta

Zadatak: Ispisati naziv sklopa i naziv komponenti od kojih se sklop sastoji (ukoliko komponente sklopa postoje)

```

SELECT s.sklop.nazSklop
, komponenta.nazSklop AS nazKomponenta
FROM s.sklop AS sklop
INNER JOIN s.sklopKomp
ON komponenta.sifSklop = s.sklopKomp.sifKomp
RIGHT OUTER JOIN s.sklop
ON s.sklopKomp.sifSklop = s.sklop.sifSklop
  
```

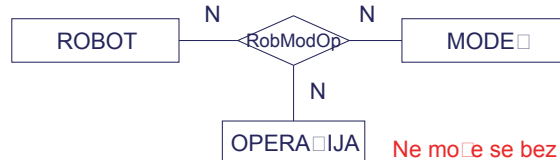
Problem

Model proizvodnje:

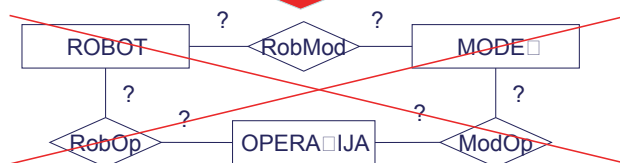
- robot R1 montira prednja lijeva vrata na modelu automobila olvo S40 za 45 sekundi i pri tome utroši 0.8 kWh energije
- robot R2 oboji poklopac motora na modelu automobila olvo S40 za 28 sekundi i pri tome utroši 0.4 kWh energije
- robot R1 montira prednja lijeva vrata na modelu automobila olvo S60 za 52 sekunde i pri tome utroši 0.9 kWh energije
- robot R1 montira poklopac motora na modelu automobila olvo S40 za 25 sekundi i pri tome utroši 0.75 kWh energije
- robot R2 montira prednja lijeva vrata na modelu automobila olvo S40 za 40 sekundi i pri tome utroši 0.6 kWh energije
- robot R2 montira poklopac motora na modelu automobila olvo S40 za 18 sekundi i pri tome utroši 0.7 kWh energije

Ternearne veze

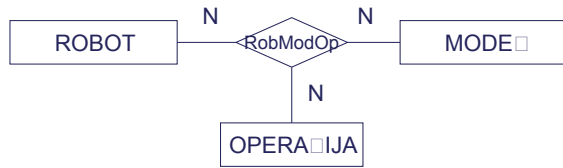
Ternarnom vezom prikazuje se istovremeni odnos triju entiteta.



Ne može se bez gubitaka informacija zamijeniti trima binarnim vezama



Erne veze - preslikavanje N:N:N



ROBOT = sifRobot, nazRobot, □

MODE = sifModel, nazModel, □

OPERA = sifOper, nazOper, □

RobModOp = sifRobot, sifModel, sifOper, utrrijeme, utrEnergija

Erne veze N:N:N → relacijski model

ROBOT = sifRobot, nazRobot, □

MODE = sifModel, nazModel, □

OPERA = sifOper, nazOper, □

□ pravila integriteta

RobModOp = sifRobot, sifModel, sifOper, utrrijeme, utrEnergija

Zadatak: Ispisati naziv robota, naziv modela automobila, naziv operacije, utrošak vremena i energije, za sve operacije koje roboti mogu obaviti

```

SELECT nazRobot, nazModel, nazOper, utrrijeme, utrEnergija
FROM robModOp
INNER JOIN robot
ON robModOp.sifRobot = robot.sifRobot
INNER JOIN model
ON robModOp.sifModel = model.sifModel
INNER JOIN operacija
ON robModOp.sifOper = operacija.sifOper
    
```

Definicija 1. (Teore)

U vezi koja povezuje entitete

$E_1, \dots, E_k, \dots, E_m$,

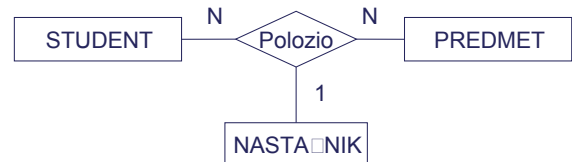
spojnost = 1 entiteta E_k znači da za svaku vrijednost svih entiteta E_1, \dots, E_m , osim E_k , uvijek postoji točno jedna vrijednost od E_k .

→ može se reći da tada vrijedi funkcijska zavisnost:

$$\bigcup_{j=1}^m K_j \rightarrow K_k$$

gdje su skupovi K_j , ($j = 1, \dots, m$) ključevi entiteta E_1, \dots, E_m

Erne veze - preslikavanje N:N:1



STUDENT = matBrSt, prezSt, imeSt

PREDMET = sifPred, nazPred

NASTAVNIK = sifNast, prezNast, imeNast

Polozio = matBrSt, sifPred, sifNast, ocjena

Erne veze N:N:1 → relacijski model

STUDENT = matBrSt, prezSt, imeSt

PREDMET = sifPred, nazPred

NASTAVNIK = sifNast, prezNast, imeNast

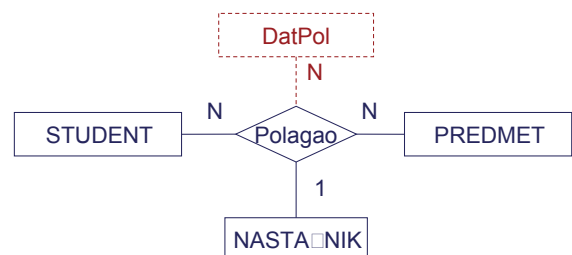
Polozio = matBrSt, sifPred, sifNast, ocjena

□ pravila integriteta

```

SELECT prezSt, imeSt, nazPred, prezNast, imeNast, ocjena
FROM polozio
INNER JOIN student
ON polozio.matBrSt = student.matBrSt
INNER JOIN predmet
ON polozio.sifPred = predmet.sifPred
INNER JOIN nastavnici
ON polozio.sifNast = nastavnici.sifNast
    
```

Erne veze - preslikavanje N:N:1



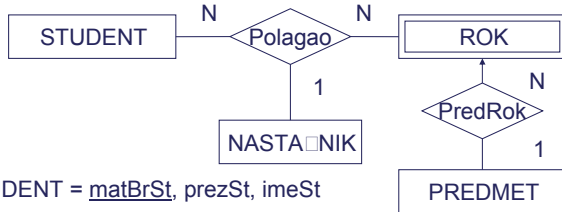
STUDENT = matBrSt, prezSt, imeSt

PREDMET = sifPred, nazPred

NASTAVNIK = sifNast, prezNast, imeNast

Polagao = matBrSt, sifPred, DatPol, sifNast, ocjena

Ernarne veze - preslikavanje N:N:1



STUDENT = matBrSt, prezSt, imeSt

PREDMET = sifPred, nazPred

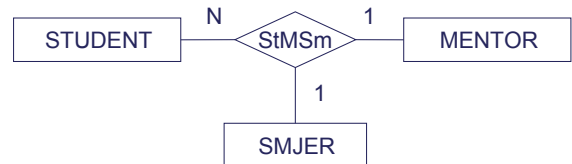
ROK = sifPred, datRok, vrstaRok

NASTAVNIK = sifNast, prezNast, imeNast

PredRok = sifPred, datRok

Polagao = matBrSt, sifPred, datRok, sifNast, ocjena

Ernarne veze - preslikavanje N:1:1



Student može studirati na više smjerova, ali na svakom smjeru mora imati različitog mentora. Student na svakom smjeru ima samo jednog mentora.

STUDENT = matBrSt, prezSt, imeSt

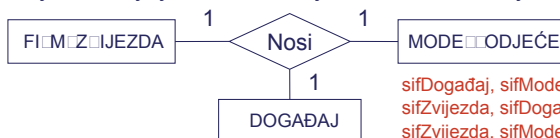
SMJER = sifSmjer, nazSmjer

MENTOR = sifMentor, prezMentor, imeMentor

StMSm = sifSmjer, matBrSt, sifMentor

Ernarne veze - preslikavanje 1:1:1

Na jednom događaju (npr. Dodjela Oscara 2008.) ne smiju dvije ili više filmskih zvijezda nositi isti model odjeće. Tijekom jednog događaja, jedna zvijezda nosi samo jedan model odjeće. Jedna zvijezda smije jedan model odjeće nositi na samo jednom događaju.



sifDogađaj, sifModel → sifZvijezda
sifZvijezda, sifDogađaj → sifModel
sifZvijezda, sifModel → sifDogađaj

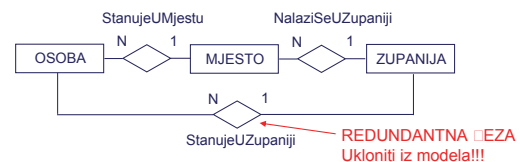
FILMSKIZVJEZDA = sifZvijezda, ime, prezime

MODELODJECE = sifModel, nazModel

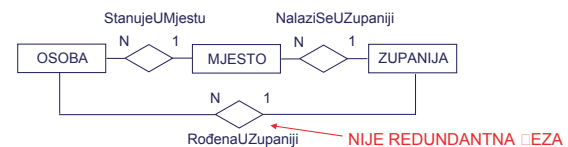
DOGAĐAJ = sifDogađaj, nazDogađaj, datumDogađaj

Nosi = sifZvijezda, sifDogađaj, sifModel

Redundantne veze

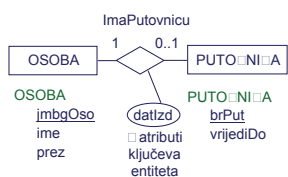


REDUNDANTNA VEZA
Ukloniti iz modela!!!



NIJE REDUNDANTNA VEZA

Primjeri preslikavanja u relacijski model



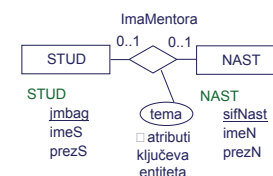
```

CREATE TABLE osoba {
    jmb_Oso ...
    ime ...
    prez ...
    PRIMAR {E} {jmb_Oso}
}
    
```

```

CREATE TABLE putovnica {
    brPut ...
    vrijediDo ...
    datIzd ...
    jmb_Oso ... NOT NULL
    PRIMAR {E} {brPut}
    {NI} {E} {jmb_Oso}
    FOREIGN {E} {jmb_Oso}
    REFERENCES osoba {jmb_Oso}
}
    
```

Primjeri preslikavanja u relacijski model



U ovom primjeru se pretpostavlja da jedan nastavnik može biti mentor najviše jednom studentu (ili niti jednom), te da student može imati najviše jednog mentora (ili niti jednog).

```

CREATE TABLE stud {
    jmba ...
    imeS ...
    prezS ...
    si_Nast ...
    tema ...
    PRIMAR {E} {jmba}
    FOREIGN {E} {si_Nast}
    REFERENCES nast {si_Nast}
}
    
```

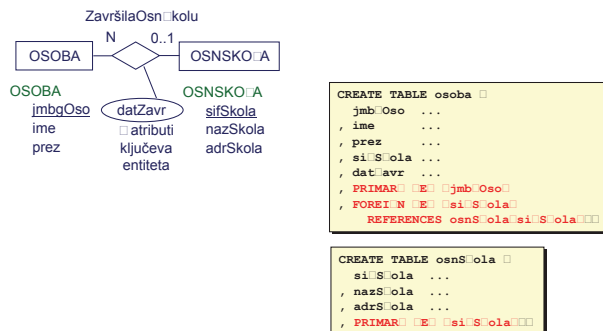
```

CREATE TABLE nast {
    si_Nast ...
    imeN ...
    prezN ...
    jmba ...
    tema ...
    PRIMAR {E} {si_Nast}
    FOREIGN {E} {jmba}
    REFERENCES stud {jmba}
}
    
```

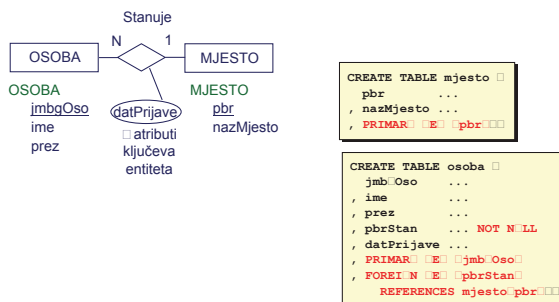
```

CREATE TABLE stud {
    jmba ...
    imeS ...
    prezS ...
    PRIMAR {E} {jmba}
}
    
```

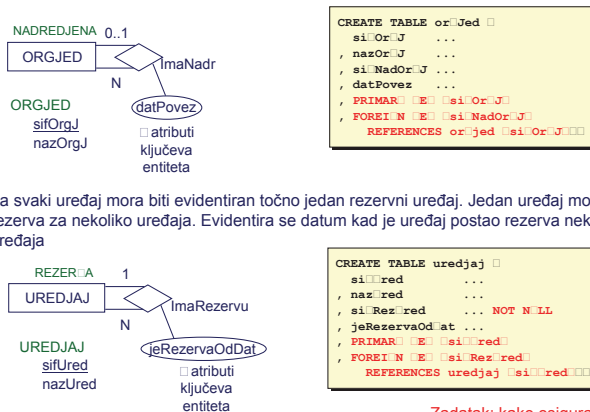
Primjeri preslikavanja u relacijski model



Primjeri preslikavanja u relacijski model



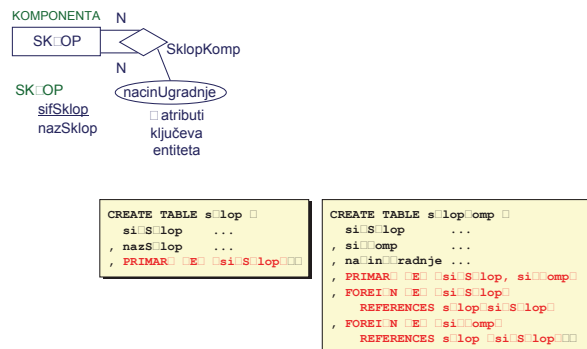
Primjeri preslikavanja u relacijski model



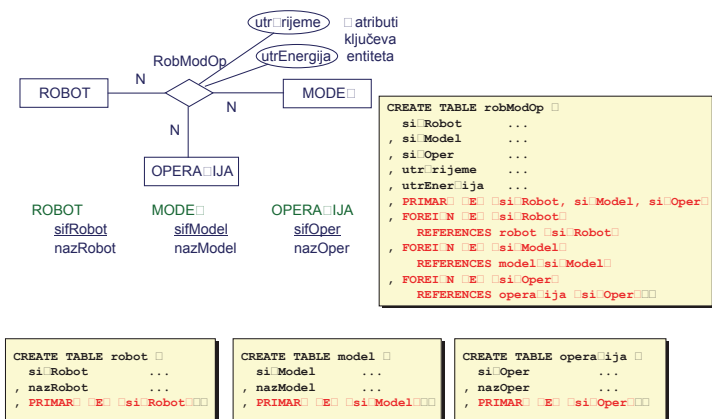
Za svaki uređaj mora biti evidentiran točno jedan rezervni uređaj. Jedan uređaj može biti rezerva za nekoliko uređaja. Evidentira se datum kad je uređaj postao rezerva nekog drugog uređaja

Zadatak: kako osigurati da uređaj ne bude sam sebi rezerva?

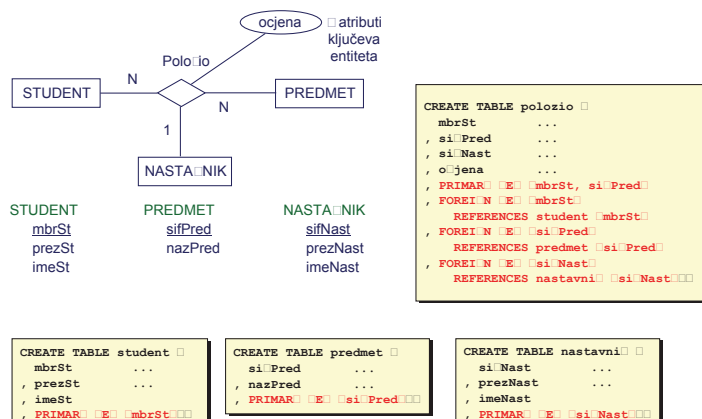
Primjeri preslikavanja u relacijski model



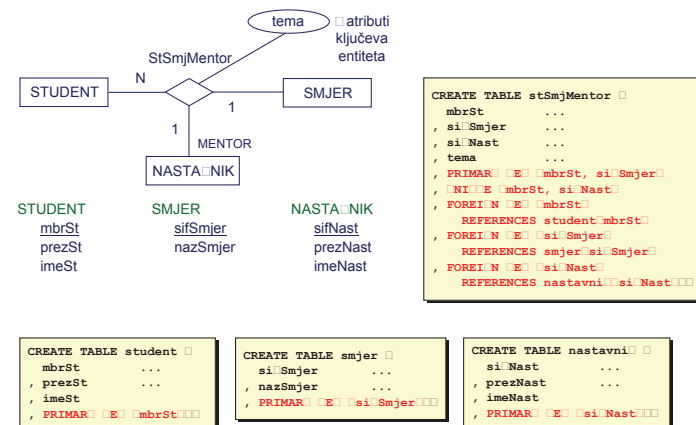
Primjeri preslikavanja u relacijski model



Primjeri preslikavanja u relacijski model

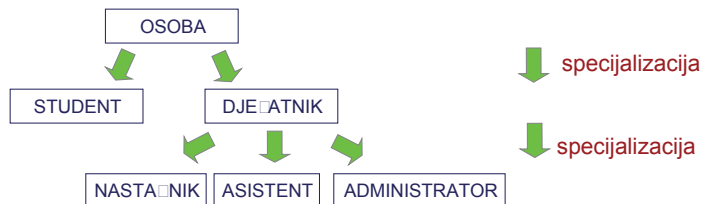


Primjeri preslikavanja u relacijski model



Specijalizacija

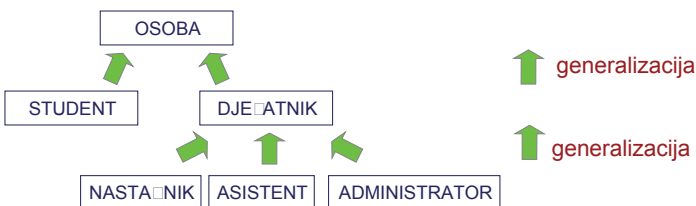
Entiteti jednog skupa entiteta mogu se temeljem njihovih karakterističnih svojstava klasificirati u zasebne skupove entiteta, postupkom koji se naziva **specijalizacija**



Skupovi entiteta dobiveni postupkom **specijalizacije** nazivaju se podklase (*subclasses*) ili specijalizacije

Generalizacija

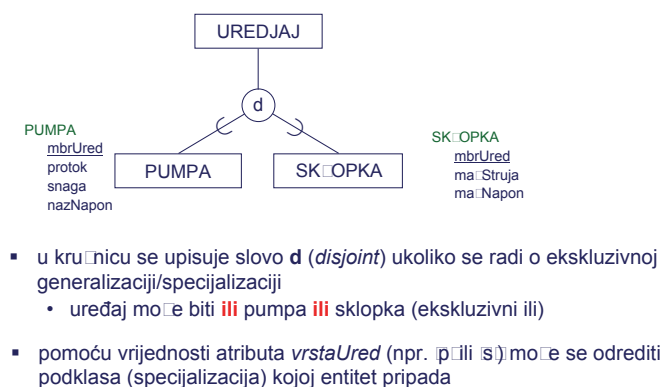
Entiteti iz nekoliko skupova entiteta sa sličnim svojstvima mogu se grupirati u zajednički skup entiteta, postupkom koji se naziva **generalizacija**



Skupovi entiteta dobiveni postupkom **generalizacije** nazivaju se nadklase (*superclasses*) ili generalizacije

Postupak generalizacije je inverzan postupku specijalizacije

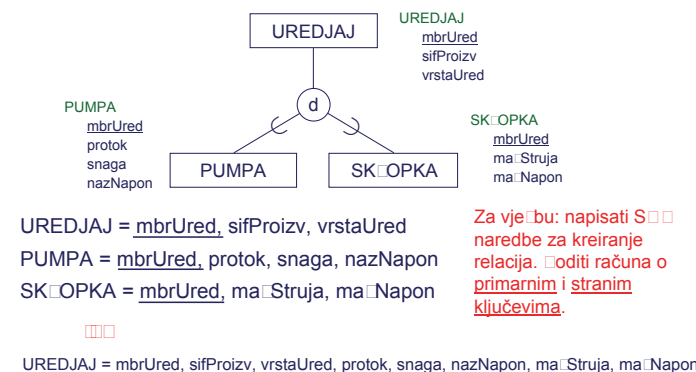
Generalizacija i specijalizacija - ER dijagram



- u kružnicu se upisuje slovo **d** (*disjoint*) ukoliko se radi o ekskluzivnoj generalizaciji/specijalizaciji
 - uređaj može biti **ili** pumpa **ili** sklopka (ekskluzivni ili)
- pomoću vrijednosti atributa *vrstaUred* (npr. *p* ili *s*) može se odrediti podklasa (specijalizacija) kojoj entitet pripada

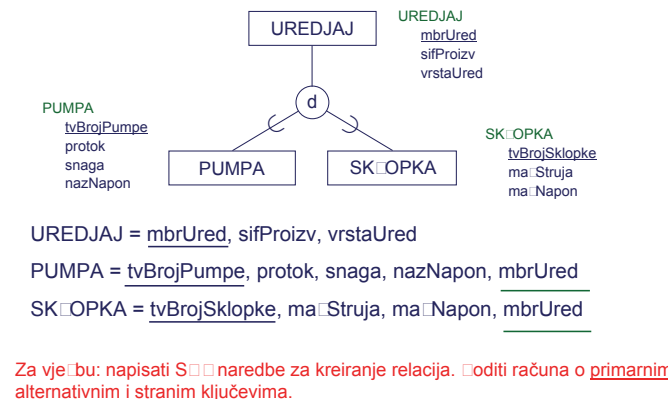
Preslikavanje u relacijski model

- specijalizacije nemaju vlastite ključeve



Preslikavanje u relacijski model

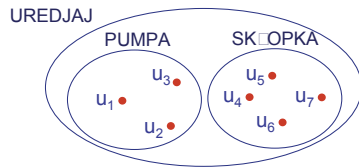
- specijalizacije imaju vlastite ključeve



Neekskluzivna generalizacija/specijalizacija

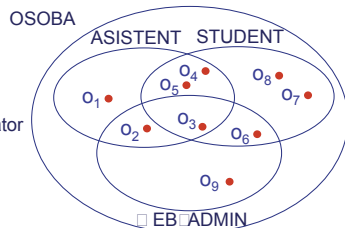
- ekskluzivna generalizacija/specijalizacija

Uređaj može biti pumpa ili sklopka

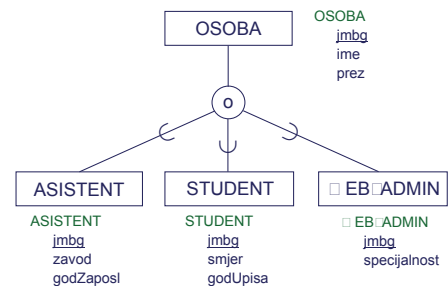


- neekskluzivna generalizacija/specijalizacija

Osoba može biti asistent i/ili student na poslijediplomskom studiju i/ili administrator web stranica fakulteta

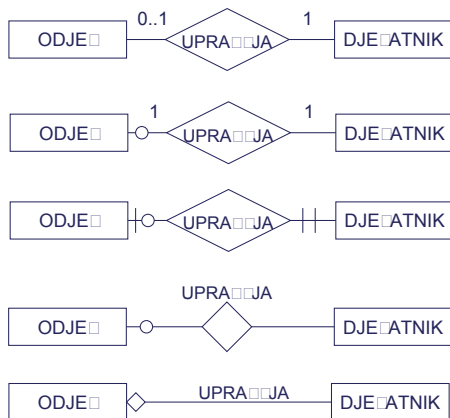


Neekskluzivna generalizacija/specijalizacija

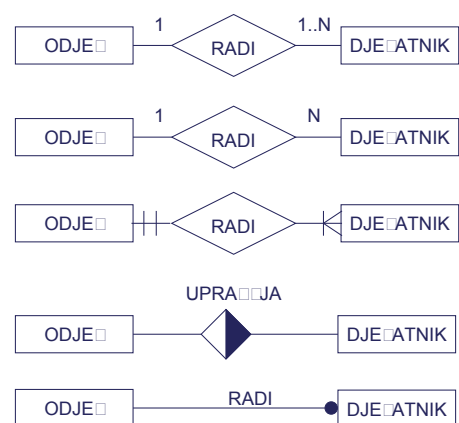


- ukoliko se radi o neekskluzivnoj generalizaciji/specijalizaciji u kružnicu se upisuje slovo **o** (*overlapping*)
 - osoba može biti asistent i/ili student i/ili administrator web stranica
- preslikavanje u relacijski model: jednako kao kod ekskluzivne generalizacije/specijalizacije (također su moguće specijalizacije s vlastitim i bez vlastitih ključeva)

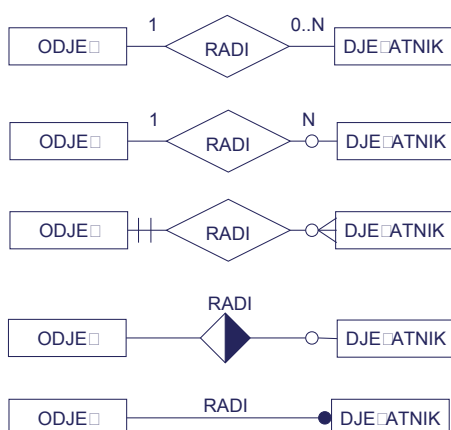
ER model - različita notacija



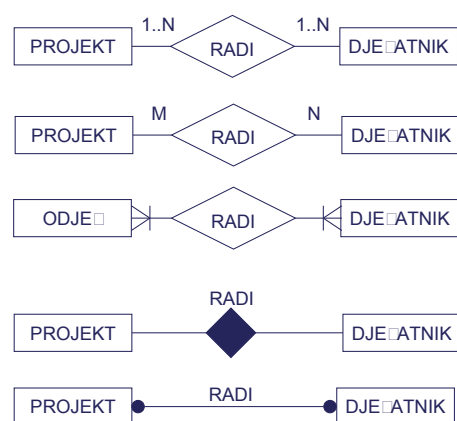
ER model - različita notacija



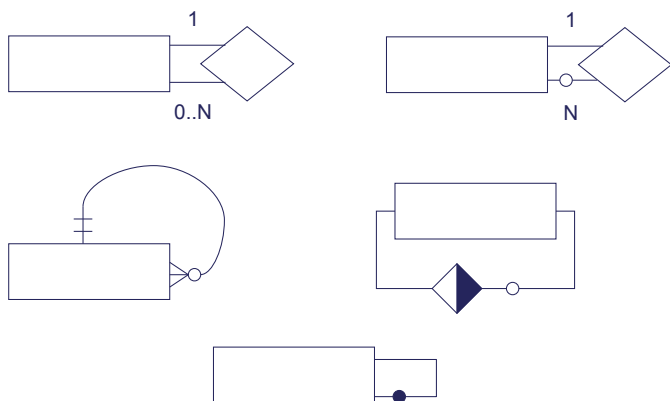
ER model - različita notacija



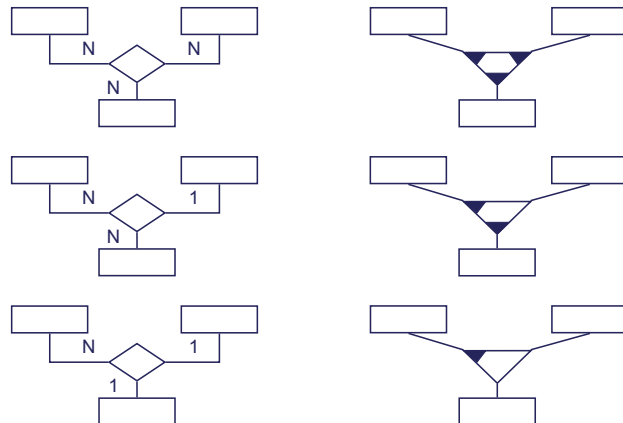
ER model - različita notacija



ER model - različita notacija



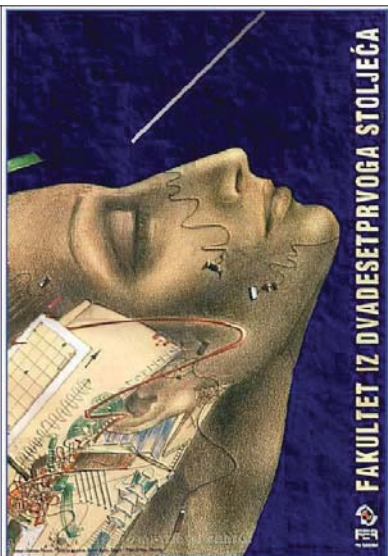
ER model - različita notacija



Baze podataka

Predavanja
lipanj 2008.

1. ER model baze podataka (1. dio - primjeri)



1. Model baze podataka za razredbeni ispit

Potrebno je evidentirati podatke o kandidatima: JMBG, prezime, ime, završenu srednju školu, mjesto rođenja i mjesto stanovanja. Pretpostavlja se da je kandidat završio samo jednu srednju školu. Za svaku srednju školu treba evidentirati šifru koja ju jedinstveno identificira, naziv, adresu i mjesto u kojem se škola nalazi. Za mjesto treba evidentirati poštanski broj, naziv mjesta i županiju u kojoj se mjesto nalazi. Županija ima svoju šifru i naziv.

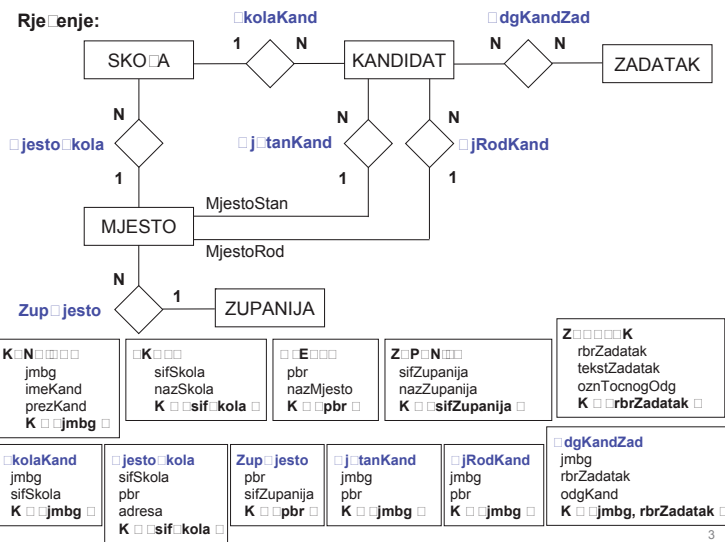
Treba evidentirati podatke o zadacima na testu: redni broj zadatka, tekst zadatka, oznaku točnog odgovora (može biti A, B, C, D ili E).

Za svakog kandidata evidentirati odgovore koje je dao na zadatke (mogući odgovori kandidata su A, B, C, D, E ili ništa).

Nacrtati ER model i opisati entitete i veze. Sve sheme moraju zadovoljavati 3NF.

Opisati relacijski model u obliku SQL naredbi za kreiranje relacija s opisanim integritetskim ograničenjima. Odabrati prikladne tipove podataka.

Rješenje:



K N
jmbg
imeKand
prezKand
K jmbg

Ako bi neki entitet imao više mogućih ključeva, shema entiteta bi se mogla opisati npr. ovako:

K N
jmbg
sifKand
imeKand
prezKand
PK K jmbg
K sifKand

Relacijski model u obliku SQL naredbi za kreiranje relacija:

```
CREATE TABLE zupanija (
    sif_zupanija SMALLINT,
    naz_zupanija CHAR(40),
    PRIMAR KEY sif_zupanija
)

CREATE TABLE mjesto (
    pbr INTEGER,
    naz_mjesto CHAR(40),
    sif_zupanija SMALLINT NOT NULL,
    PRIMAR KEY pbr,
    FOREIGN KEY sif_zupanija REFERENCES zupanija(sif_zupanija)
)

CREATE TABLE skola (
    sif_skola INTEGER,
    naz_skola CHAR(40),
    pbr INTEGER NOT NULL,
    adresa CHAR(40),
    PRIMAR KEY sif_skola,
    FOREIGN KEY pbr REFERENCES mjesto(pbr)
)

CREATE TABLE zadaca (
    rbr_zadaca INTEGER,
    tekst_zadaca CHAR(40),
    ozn_todno_od CHAR(40),
    PRIMAR KEY rbr_zadaca
)
```

5

Relacijski model u obliku SQL naredbi za kreiranje relacija (nastavak):

```
CREATE TABLE kandidat (
    jmb CHAR(40),
    ime_and CHAR(40),
    prez_and CHAR(40),
    pbrRod INTEGER NOT NULL,
    pbrStan INTEGER NOT NULL,
    sif_skola INTEGER NOT NULL,
    PRIMAR KEY jmb,
    FOREIGN KEY pbrRod REFERENCES mjesto(pbr),
    FOREIGN KEY pbrStan REFERENCES mjesto(pbr),
    FOREIGN KEY sif_skola REFERENCES skola(sif_skola)
)

CREATE TABLE odabrad (
    jmb CHAR(40),
    rbr_zadaca INTEGER,
    odabrad CHAR(40),
    PRIMAR KEY jmb, rbr_zadaca,
    FOREIGN KEY jmb REFERENCES kandidat(jmb),
    FOREIGN KEY rbr_zadaca REFERENCES zadaca(rbr_zadaca)
)
```

6

Model baze podataka za videoteku

Za film se evidentira šifra (identificira film), naslov filma, te osobe i njihove funkcije u filmu.

Funkcije koje osoba može imati u filmu predstavljene su kraticom i nazivom (npr. G - glumac, RED, redatelj, S - scenarist, itd.). Za svaku se osobu evidentira šifra osobe (identificira osobu), prezime i ime.

Treba uočiti da ista osoba može u istom filmu imati različite funkcije, npr:

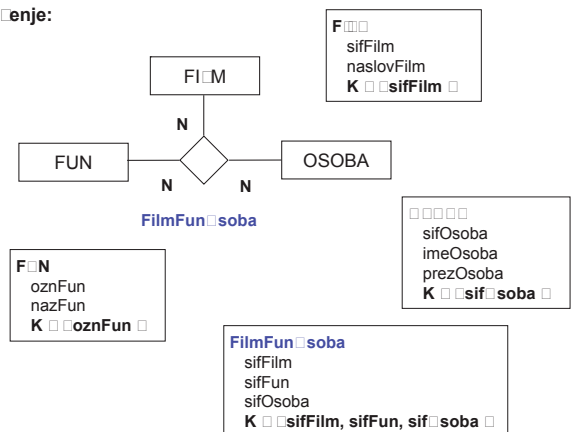
relacija FilmFunOsoba

sif Film	naslovFilm	ozn Fun	nazFun	sif Osoba	ime	prezime
1	Nepomirjljivi	RED	redatelj	10	Lint	Eastwood
1	Nepomirjljivi	G	glumac	10	Lint	Eastwood
1	Nepomirjljivi	G	glumac	20	Morgan	Freeman
2	Mostovi okruga Madison	RED	redatelj	10	Lint	Eastwood
2	Mostovi okruga Madison	G	glumac	40	Merill	Streep
2	Mostovi okruga Madison	G	glumac	10	Lint	Eastwood
3	Priljavi arr	RED	redatelj	30	Don	Siegel
3	Priljavi arr	G	glumac	10	Lint	Eastwood

7

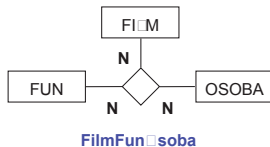
Nacrtati ER model i opisati entitete i veze. Sve sheme moraju zadovoljavati 3NF.

Rješenje:



8

iskusija:



Relacije koje nastaju transformacijom ER modela na slici:

ključevi relacija su podcrtani

FILM	FUN	OSOBA
<u>sifFilm</u>	<u>oznFun</u>	<u>sifOsoba</u>
naslovFilm	nazFun	ime, prezime
1	RED	10
2	G	10
3	G	20
2	RED	10
2	G	40
2	G	10
3	RED	30
3	G	10
	RED	
	glumac	

9

SQL upit kojim se dohvaćaju osobe i njihove funkcije u filmu Priljavi arr

```
SELECT osoba.*, fun.*
FROM osoba, film, fun, filmFunOsoba
WHERE osoba.sifOsoba = filmFunOsoba.sifOsoba
AND film.sifFilm = filmFunOsoba.sifFilm
AND fun.oznFun = filmFunOsoba.oznFun
AND film.naslovFilm = Priljavi arr
```

sifOsoba	ime	prezime	oznFun	nazFun
10	Lint	Eastwood	G	glumac
30	Don	Siegel	RED	redatelj

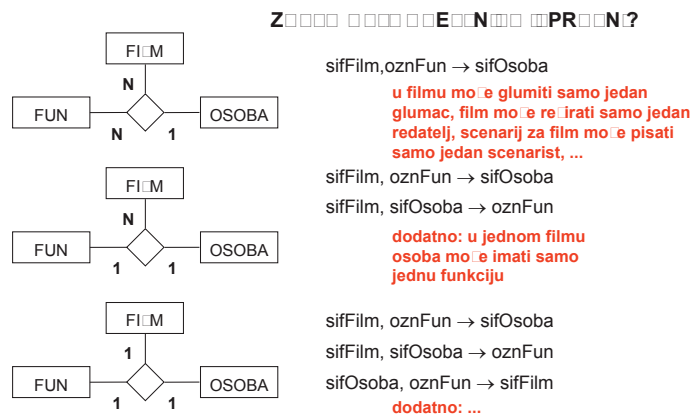
relacija FilmFunOsoba

sif Film	naslovFilm	ozn Fun	nazFun	sif Osoba	ime	prezime
1	Nepomirjljivi	RED	redatelj	10	Lint	Eastwood
1	Nepomirjljivi	G	glumac	10	Lint	Eastwood
1	Nepomirjljivi	G	glumac	20	Morgan	Freeman
2	Mostovi okruga Madison	RED	redatelj	10	Lint	Eastwood
2	Mostovi okruga Madison	G	glumac	40	Merill	Streep
2	Mostovi okruga Madison	G	glumac	10	Lint	Eastwood
3	Priljavi arr	RED	redatelj	30	Don	Siegel
3	Priljavi arr	G	glumac	10	Lint	Eastwood

10

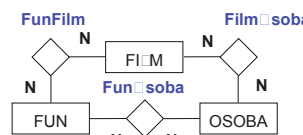
(**teorema**): U vezi koja povezuje entitete $E_1, \dots, E_k, \dots, E_m$, spojnost = 1 entiteta E_k znači da (...) odnosno, vrijedi funkcijska zavisnost

$\bigcup_{j=1}^m K_j \rightarrow K_k$ gdje su skupovi K_j , ($j = 1, \dots, m$), ključevi entiteta E_1, \dots, E_m



11

Značenje: Funkcija $E \rightarrow N$ **PR** $E \rightarrow N$?



Relacije koje nastaju transformacijom ER modela na slici:

FILM
sifFilm
1
2
3

naslovFilm
Nepomirljivi
Mostovi okruga M.
Prljavi Harr

FunFilm
oznFun sifFilm
RED 1
RED 2
RED 3

FilmOsoba
sifFilm sifOsoba
1 10
1 20
2 10

FunOsoba
oznFun sifOsoba
G 10
G 20
G 40

ključevi relacija su podcrtani

OSOBA
sifOsoba ime, prezime
10 Eastwood
20 M. Freeman
30 D. Siegel
40 M. Strep

FUN
oznFun nazFun
RED redatelj
G glumac

FunOsoba
oznFun sifOsoba
G 10
G 20
G 40

FunOsoba
oznFun sifOsoba
RED 10
RED 30

12

SQL upit kojim se dohvaćaju osobe i njihove funkcije u filmu (Prljavi Harr)

```
SELECT osoba.*, fun.*
FROM osoba, film, fun, filmOsoba, funOsoba, funFilm
WHERE osoba.sifOsoba = filmOsoba.sifOsoba
AND filmOsoba.sifFilm = film.sifFilm
AND film.sifFilm = funFilm.sifFilm
AND funFilm.oznFun = fun.oznFun
AND fun.oznFun = funOsoba.oznFun
AND funOsoba.sifOsoba = osoba.sifOsoba
AND film.naslovFilm = Prljavi Harr
```

sifOsoba	ime	prezime	oznFun	nazFun
10	Clinton	Eastwood	G	glumac
10	Clinton	Eastwood	RED	redatelj
30	Don	Siegel	RED	redatelj

Clinton Eastwood nije redatelj filma Prljavi Harr
→ gubitak informacije

Dekompozicija relacije IDEOTEKA nije obavljena bez gubitka informacije. Uvjet za dekompoziciju bez gubitka informacija opisan je u predavanjima.

13

odel baze podataka za poduzeće za održavanje plinskih instalacija

Uređaji koje poduzeće evidentira su brojila, ventili i reduktori. Za svaki pojedini uređaj treba evidentirati vrstu uređaja (Brojilo ili R), proizvođača uređaja, tvornički broj i godinu proizvodnje uređaja. Za proizvođače uređaja evidentiraju se njihove šifre i nazivi. Ne postoje dva uređaja istog proizvođača koji imaju jednake tvorničke brojeve.

Dodatno, ovisno o vrsti uređaja, treba evidentirati njima svojstvene, posebne ili specijalističke podatke.

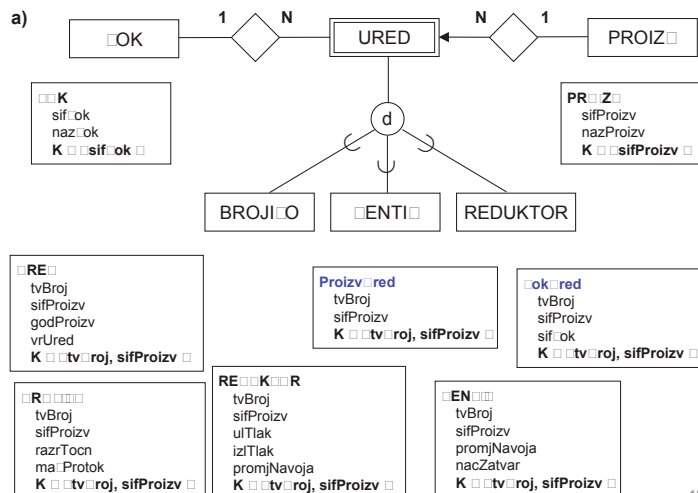
za brojila:	za ventile:	za reduktore:
razred točnosti	promjer navoja	ulazni tlak plina
maks. protok plina	način zatvaranja	izlazni tlak plina
		promjer navoja

Potrebno je evidentirati popis lokacija (šifra i naziv) na kojima uređaji mogu biti instalirani. Evidentirati trenutnu lokaciju na kojoj je uređaj instaliran.

- Nacrtati ER model i opisati entitete i veze. Sve sheme moraju zadovoljavati 3NF. Opisati relacijski model u obliku SQL naredbi za kreiranje relacija s ugrađenim pravilima integriteta.
- Što treba promijeniti u ER modelu iz a) kako bi se omogućilo evidentiranje povijesti premještanja uređaja. Kakve su posljedice na relacijski model?

14

Rješenje:



15

Rješenje: Relacijski model

```
CREATE TABLE proizvod (
    sifProizv INTEGER,
    nazProizv CHAR(50),
    PRIMAR KEY (sifProizv)
)

CREATE TABLE lok (
    sifLo INTEGER,
    nazLo CHAR(50),
    PRIMAR KEY (sifLo)
)

CREATE TABLE ured (
    tvBroj CHAR(10),
    sifProizv INTEGER,
    godProizv SMALLINT,
    vrUred CHAR(10),
    sifLo INTEGER NOT NULL,
    PRIMAR KEY (tvBroj, sifProizv),
    FOREIGN KEY (sifProizv) REFERENCES proizvod (sifProizv),
    FOREIGN KEY (sifLo) REFERENCES lok (sifLo)
)

CREATE TABLE brojilo (
    tvBroj CHAR(10),
    sifProizv INTEGER,
    razrTozn DECIMAL(5,2),
    maxProto DECIMAL(5,2),
    PRIMAR KEY (tvBroj, sifProizv),
    FOREIGN KEY (tvBroj, sifProizv) REFERENCES ured (tvBroj, sifProizv)
)
```

16

Rješenje: Relacijski model (nastavak)

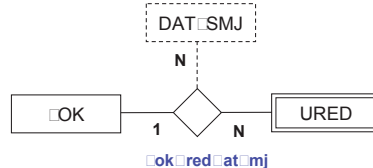
```
CREATE TABLE ventil
(
    tvBroj CHAR(10),
    siProizv INTEGER,
    promjNavoja DECIMAL(10,2),
    naatvar DECIMAL(10,2),
    PRIMAR KEY (tvBroj, siProizv),
    FOREIGN KEY (tvBroj, siProizv)
    REFERENCES ured (tvBroj, siProizv)
);
```

```
CREATE TABLE reduktor
(
    tvBroj CHAR(10),
    siProizv INTEGER,
    ulTla DECIMAL(10,2),
    izlTla DECIMAL(10,2),
    promjNavoja DECIMAL(10,2),
    PRIMAR KEY (tvBroj, siProizv),
    FOREIGN KEY (tvBroj, siProizv)
    REFERENCES ured (tvBroj, siProizv)
);
```

17

Rješenje:

- b) Pretpostavi li se da jedan uređaj ne može biti premješten više nego jedan puta na dan, segment ER modela će se promijeniti na sljedeći način:



U odnosu na rješenje pod a), u novom relacijskom modelu potrebno je izbaciti atribut sifok iz relacije ured, te dodati novu relaciju lokUredDatSmj

```
CREATE TABLE lokUredDatSmj
(
    tvBroj CHAR(10),
    siProizv INTEGER,
    datSmjestaj DATE,
    siLo INTEGER NOT NULL,
    PRIMAR KEY (tvBroj, siProizv, datSmjestaj),
    FOREIGN KEY (tvBroj, siProizv)
    REFERENCES ured (tvBroj, siProizv),
    FOREIGN KEY (siLo)
    REFERENCES loksiLo
);
```

18

odel baze podataka automehaničarske radionice

Evidentirati podatke o automobilima. Automobil je identificiran tvorničkim brojem (ne postoje dva automobila s istim tvorničkim brojem). Za automobil treba evidentirati godinu proizvodnje i model automobila. Modeli automobila identificirani su proizvođačem i nazivom modela (međusobno različiti proizvođači mogu svoje modele nazivati istim imenom - npr. Renault može imati svoj model naziva Europa, a Opel može imati sasvim drugi model koji se također naziva Europa). Za model automobila evidentira se godina u kojoj je model prvi puta proizveden. Proizvođač ima naziv, a identificiran je svojom šifrom.

U radionici je napravljen popis vrsta poslova koji se mogu obavljati na automobilima. Vrste poslova su šifrirane, a osim šifre i opisa vrste posla (npr. Izmjena ulja, Podešavanje ventila itd.), za svaku vrstu posla se evidentira normativom zadano trajanje izrađeno u minutama (koliko bi vremena mehaničar trebao utrošiti obavljajući posao te vrste). Vrste poslova su nezavisne od modela automobila - npr. Izmjena ulja je uvijek jednak posao neovisno od modela automobila na kojem se obavlja.

19

Za mehaničare zaposlene u radionici evidentira se jmbg, prezime i ime.

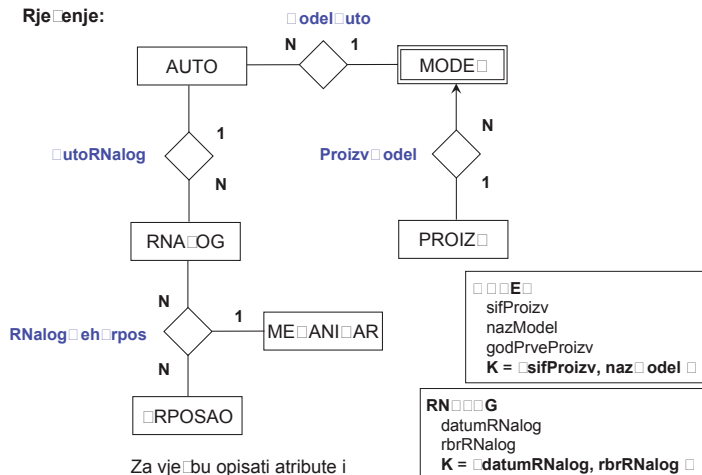
Za svaki dolazak automobila u radionicu otvara se jedan Radni nalog na kojem se evidentira automobil i datum dolaska automobila u radionicu. Isti automobil može biti primljen u radionicu više puta (čak i istog dana), ali se svaki put otvara novi Radni nalog. Radni nalog nema šifru. Radni nalog pri otvaranju dobiva svoj redni broj, pri čemu svakog dana redni brojevi naloga započinju ponovo s brojem jedan. Za isti datum ne postoje dva Radna naloga s istim brojem.

Uz Radni nalog se evidentira koji mehaničari će obaviti koje vrste poslova na automobilu. Poslove koji su zadani na Radnom nalogu može obaviti jedan ili nekoliko mehaničara, ali jedan zadani posao će jedan mehaničar obaviti sam od početka do kraja. Mehaničari na raznim Radnim nalogima mogu obavljati različite vrste poslova. Mehaničar odmah po obavljenom poslu na nekom automobilu evidentira koliko je vremena u minutama zaista utrošio na obavljanje tog posla (to se vrijeme može razlikovati od normativom zadanog vremena).

Nacrtati ER model i opisati entitete i veze. Sve sheme moraju zadovoljavati 3NF.

20

Rješenje:



21

odel baze podataka za izlobo pasa

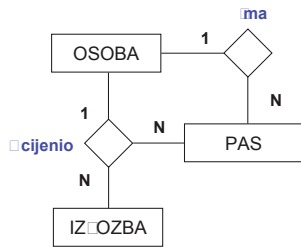
Za svaku se osobu evidentira jmbg, prezime i ime. Za psa se evidentira broj markice koja identificira psa, ime psa, datum okota i osoba koja je vlasnik tog psa. Pretpostavlja se da jedna osoba može imati više pasa, a pas pripada samo jednoj osobi.

Neki vlasnici vode svoje pse na izlobo pasa. Za izlobu se evidentira šifra izlobo koja ju jedinstveno identificira i datum izlobo. Za jednog psa na jednoj izlobi treba evidentirati samo jednu ocjenu i osobu koja ga je ocjenjivala. Ista osoba na jednoj izlobi može ocijeniti više pasa. Ista osoba može ocjenjivati istog psa na više različitih izlobi. Za osobe koje ocjenjuju pse također se evidentiraju jmbg, prezime i ime. Te osobe mogu istovremeno biti i vlasnici pasa.

Nacrtati ER model i opisati entitete i veze. Sve sheme moraju zadovoljavati 3NF.

22

Rješenje:



OSOBA
jmbg
imeOso
prezOso
K = jmbg

PAS
brMarkice
imePas
datOkota
K = brarkice

IZOZBA
sifzlozba
datizlozba
K = sifzlozba

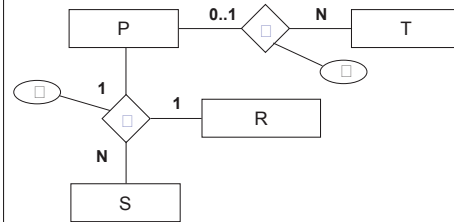
ima
brMarkice
jmbg
K = brarkice

cijenio
brMarkice
sifzlozba
jmbg
ocjena
K = sifzlozba, brarkice

23

Zadan je ER model i pripadne sheme entiteta. Na slici su prikazani samo vlastiti atributi veza.

Definirati sheme veza. Napisati SQL naredbe za kreiranje relacija relacijskog modela. Tipove podataka ne treba navoditi. Naredbe moraju sadržavati definicije integritetskih ograničenja.



P
a
b
c
d
PK = K₁ = a, b
K₂ = c

R
e
f
PK = e

T
g
h
PK = g

S
i
j
k
PK = i, j

24

Rješenje: heme veza

a
b
e
g
PK = K₁ = a, b, g
K₂ = e, g

i
j
a
b
PK = i, j

Relacijski model

```
CREATE TABLE p
(a ...
, b ...
, d ...
, PRIMAR (a, b)
, NI (e))
```

```
CREATE TABLE r
(e ...
, PRIMAR (e))
```

```
CREATE TABLE s
...
, PRIMAR (e))
```

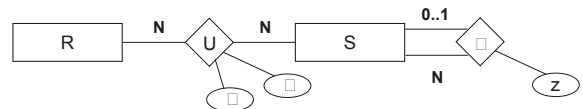
```
CREATE TABLE t
(i ...
, j ...
, a ...
, b ...
, PRIMAR (i, j)
, FOREIGN (a, b)
REFERENCES p (a, b))
```

```
CREATE TABLE v
(a ...
, b ...
, e ...
, PRIMAR (a, b, e)
, NI (e)
, FOREIGN (a, b)
REFERENCES p (a, b)
, FOREIGN (e)
REFERENCES r (e)
, FOREIGN (e)
REFERENCES s (e))
```

25

Zadan je ER model i pripadne sheme entiteta. Na slici su prikazani samo vlastiti atributi veza.

Definirati sheme veza. Napisati SQL naredbe za kreiranje relacija relacijskog modela. Tipove podataka ne treba navoditi. Naredbe moraju sadržavati definicije integritetskih ograničenja.



R
a
b
c
PK = a, b

S
d
e
PK = d

26

Rješenje:

heme veza

a
b
d
PK = a, b, d

d
d1
z
PK = d

Preimenovati jedan od atributa

Relacijski model

```
CREATE TABLE r
(a ...
, b ...
, PRIMAR (a, b))
```

```
CREATE TABLE u
(a ...
, b ...
, d ...
, PRIMAR (a, b, d)
, FOREIGN (a, b) REFERENCES r (a, b)
, FOREIGN (d) REFERENCES s (d))
```

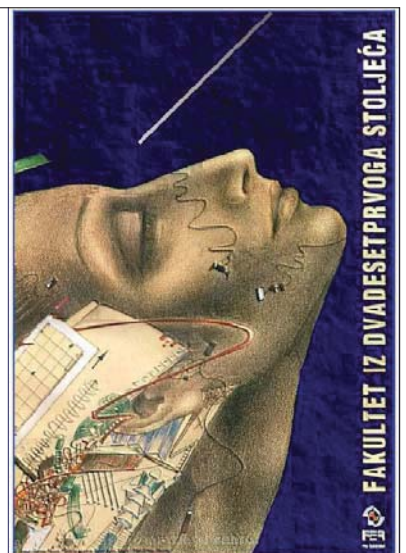
```
CREATE TABLE s
(d ...
, d1 ...
, e ...
, z ...
, PRIMAR (d)
, FOREIGN (d1) REFERENCES s (d))
```

27

Baze podataka

Predavanja
lipanj 2008.

1. Sigurnost baze podataka



Integritet i sigurnost baze podataka

- Pojmovi integritet i sigurnost baze podataka se često spominju zajedno, međutim radi se o dva različita aspekta zaštite podataka
 - Integritet baze podataka (*database integrity*) - operacije nad podacima koje korisnici obavljaju **su ispravne** (tj. uvijek rezultiraju konzistentnim stanjem baze podataka)
 - podaci se štite od ovlaštenih korisnika
 - Sigurnost baze podataka (*database security*) - korisnici koji obavljaju operacije nad podacima **su ovlašteni** za obavljanje tih operacija
 - podaci se štite od neovlaštenih korisnika

Među ovim pojmovima postoje i sličnosti. U oba slučaja:

- moraju biti definirana **pravila** koja korisnici ne smiju narušiti
- pravila se pohranjuju u rječnik podataka
- SUBP nadgleda rad korisnika - osigurava poštivanje pravila

Oblici narušavanja sigurnosti i moguće posljedice

- Oblici narušavanja sigurnosti baze podataka su:
 - neovlašteno čitanje podataka
 - neovlaštena izmjena podataka
 - neovlašteno uništavanje podataka
- Moguće posljedice su:
 - krađa ili prijevarena
 - gubitak tajnosti
 - odnosi se na podatke kritične za funkcioniranje organizacije
 - npr. krađa recepture - rezultira gubitkom konkurentnosti na tržištu
 - gubitak privatnosti
 - odnosi se na osobne podatke
 - npr. krađa podataka o zdravstvenom stanju osobe - rezultira sudskim procesom protiv vlasnika baze podataka
 - gubitak raspoloživosti
 - npr. uništenjem dijela podataka

Protumjere

- sigurnost baze podataka se osigurava zaštitom na nekoliko razina
 - zaštita na razini OS/DBMS**
 - spriječiti pristup bazama podataka ili onim dijelovima baza podataka za koje korisnici nisu ovlašteni
 - zaštita na razini operacijskog sustava**
 - spriječiti pristup radnoj memoriji računala ili datotekama u kojima SUBP pohranjuje podatke
 - zaštita na razini računalne mreže**
 - spriječiti presretanje poruka (*sniffing*) na internetu i intranetu
 - fizička zaštita**
 - fizički zaštititi lokaciju računalnog sustava
 - zaštita na razini korisnika**
 - spriječiti da ovlašteni korisnici nepoželjnom ili namjerno (npr. u zamjenu za mito ili druge usluge) omoguće pristup podacima neovlaštenim osobama

Aspekti zaštite podataka

- zakonski, socijalni i etički aspekt**
 - ima li vlasnik baze podataka zakonsko pravo na prikupljanje i korištenje podataka
 - npr. smije li zdravstvena ustanova koja, u skladu sa zakonom prikuplja podatke o pacijentima, te iste podatke koristiti pri donošenju odluke hoće li svog bivšeg pacijenta zaposliti
- strategijski aspekt**
 - tko definira pravila pristupa - tko određuje kakve ovlasti ima pojedini korisnik baze podataka, ...
- operativni aspekt**
 - kako osigurati poštivanje pravila - kojim mehanizmima se osigurava poštivanje definiranih pravila, na koji način su lozinke zaštićene, koliko često se mijenjaju, ...

Članak 77. RH - Zakon o zaštiti osobnih podataka

Svakom se jamči sigurnost i tajnost osobnih podataka. Bez privole ispitanika, osobni se podaci mogu prikupljati, obrađivati i koristiti samo uz uvjete određene zakonom.

Zakonom se uređuje zaštita podataka te nadzor nad djelovanjem informatičkih sustava u Republici.

Zabranjena je uporaba osobnih podataka suprotna utvrđenoj svrsi njihovoga prikupljanja.

- Zakon o zaštiti osobnih podataka

Korisnici OS/DBMS i ovjera autentičnosti

- administrator sustava (operacijskog sustava ili SUBP) omogućuje korisniku pristup sustavu (operacijskom sustavu ili SUBP) definiranjem jedinstvenog identifikatora korisnika (*user name*, *user ID*, *login ID*) i pripadne lozinke (*password*) koja je poznata samo dotičnom korisniku i sustavu
- korisnik koji pristupa sustavu (operacijskom sustavu ili SUBP) poznavanjem lozinke ovjerava svoju autentičnost (*authentication*)
- za ovjeru autentičnosti korisnika SUBP može koristiti
 - mehanizme operacijskog sustava ili
 - vlastite mehanizme

Autorizacija i modeli kontrole pristupa

- Autorizacija je postupak kojim se određenom korisniku dodjeljuje dozvola za obavljanje određenih vrsta operacija (čitanje, izmjena, brisanje, ...) nad određenim objektima baze podataka (relacija, pogled, atribut, ...)
 - podaci o dodijeljenim dozvolama pohranjuju se u rječnik podataka
- Prije obavljanja svake operacije, SUBP provjerava ima li korisnik dozvolu za obavljanje operacije nad objektom
 - kontrola pristupa (*access control*)
- Današnji SUBP podržavaju dva različita modela kontrole pristupa podacima
 - mandatna kontrola pristupa** (MAC-Mandatory Access Control)
 - diskrecijska kontrola pristupa** (DAC-Discretionary Access Control)

Mandatna kontrola pristupa

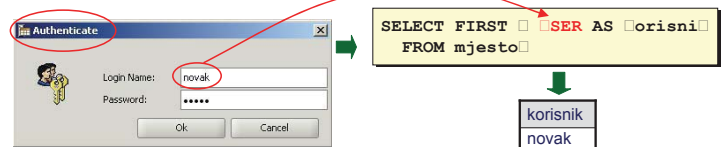
- manji broj SUBP podržava mandatnu kontrolu pristupa
 - koristi se relativno rijetko u odnosu na diskrecijsku kontrolu pristupa
- mandatna kontrola pristupa je primjenjiva u sustavima u kojima se dozvole dodjeljuju na temelju pozicije korisnika u hijerarhiji neke organizacije (vojska, državna uprava, ...)
- svaki **objekt** dobiva oznaku klasifikacijske razine (*classification level*), npr. povjerljivo, tajno, vrlo tajno, ...
- svakom **korisniku** dodjeljuje se oznaka razine ovlasti (*clearance level*)
 - korisnici mogu obavljati operacije nad onim objektima za koje imaju odgovarajuću razinu ovlasti

Diskrecijska kontrola pristupa

- većina današnjih SUBP podržava diskrecijsku kontrolu pristupa
 - diskrecijska kontrola pristupa je podržana SQL standardom
- određenom korisniku se eksplicitno dodjeljuje dozvola za obavljanje određene operacije nad određenim objektom
 - dozvole su opisane trojkama (korisnik, objekt, vrsta operacije)
 - (horvat, ispit, čitanje)
 - (horvat, ispit, izmjena)
 - (horvat, predmet, čitanje)
 - (novak, predmet, čitanje)
 - kada korisnik novak pokuša obaviti operaciju čitanja objekta (relacije) predmet, SUBP provjerava postoji li dozvola u obliku trojke (novak, predmet, čitanje)
- u preostalom dijelu predavanja razmatrat će se diskrecijska kontrola pristupa

Korisnici u SQL-u

- korisnik s određenom identifikacijskom oznakom (userID)**
 - pri uspostavljanju SQL-sjednice korisnik se prijavljuje svojim identifikatorom korisnika, te lozinkom ovjerava svoju autentičnost
 - funkcija USER vraća vrijednost identifikatora korisnika koji se koristi u dotičnoj SQL-sjednici



- bilo koji korisnik (PUBLIC)**
 - dodjelom dozvole (korisniku) PUBLIC, dozvolu za obavljanje operacije dobivaju svi sadašnji i budući korisnici

Objekti i vlasnici objekata u SQL-u

- objekti**
 - relacija (tablica, *table*)
 - atribut (stupac tablice, *column*)
 - virtualna relacija (pogled, *view*)
 - baza podataka**
- vlasnik objekta (object owner)**
 - vlasnik objekta je korisnik koji je kreirao objekt, npr:
 - vlasnik baze podataka je korisnik koji je kreirao bazu podataka
 - vlasnik relacije je korisnik koji je kreirao relaciju
 - vlasnik objekta implicitno dobiva dozvole za obavljanje svih vrsta operacija nad objektom, uključujući dozvole za:
 - dodjeljivanje svih vrsta dozvola nad tim objektom drugim korisnicima
 - uništavanje objekta

Vrste dozvola u SQL-u na razini baze podataka (dbPrivilege)

- Različiti SUBP imaju različita rješenja za dodjeljivanje dozvola na razini baze podataka. Ovdje je prikazano rješenje koje se koristi u sustavu IBM Informatica
 - SQLNNE**
 - uspostavljanje SQL-sjednice i obavljanje operacija nad objektima za koje je korisnik dobio dozvolu od vlasnika objekta ili je njihov vlasnik, kreiranje virtualnih i privremenih relacija
 - RESQLR**
 - SQLNNE kreiranje novih relacija u bazi podataka
 - SQL**
 - RESOURCE neovisno o vlasništvu i dozvolama nad objektima u bazi podataka: sve vrste operacija nad svim objektima, uništavanje svih objekata (uključujući i bazu podataka)
 - korisnik koji kreira bazu podataka je vlasnik te baze podataka i implicitno dobiva DBA (*Database administrator*) dozvolu

Vrste dozvola u MySQL-u na razini virtualne relacije (tablePrivilege)

- **SELECT (columnList)**
 - čitanje n-torki (ili vrijednosti navedenih atributa) virtualne relacije
- **UPDATE (columnList)**
 - izmjena n-torki (ili vrijednosti navedenih atributa) virtualne relacije
- **INSERT**
 - unos n-torki virtualne relacije
- **DELETE**
 - brisanje n-torki virtualne relacije
- **REFERENCES (columnList)**
 - korištenje relacije (ili samo navedenih atributa kao pozivane relacije pri definiranju stranog ključa)
- **INDEX**
 - kreiranje indeksa nad relacijom
- **ALTER**
 - izmjena strukture relacije i definiranje integritetskih ograničenja
- **PROXY GRANT**
 - sve do sada navedene vrste operacija nad virtualnom relacijom

MySQL naredbe za dodjeljivanje i ukidanje dozvola

- **GRANT dbPrivilege TO PUB userList**
- **REVOKE dbPrivilege FROM PUB userList**
- **GRANT tablePrivilegeList ON tableName viewName TO PUB userList roleList WITH GRANT OPTION**
- **REVOKE tablePrivilegeList ON tableName viewName FROM PUB userList roleList CASCADE RESTRICT**

Primjer 1:

student					ispit			
matBr	ime	prez	pbr	adresa	matBr	nazPred	datIspr	ocj
100	Ana	Ivić	51000	Korzo 2	100	Fizika	1.5.2004	3
102	Ivan	Perić	10000	Ilica 20	102	Matematika	7.9.2003	1
105	Matija	Matić	31000	Unska 7	102	Matematika	9.2.2004	5
107	Tea	Bilić	10000	Plaćka 5	107	Fizika	5.4.2006	4

- kreirati bazu podataka studBaza i relacije student i ispit
 - vlasnik baze podataka i relacija treba biti korisnik bpadmin
- korisnik horvat treba dobiti dozvole:
 - pregled svih podataka u relacijama student i ispit
 - unos, izmjena, brisanje svih podataka u relaciji ispit
- korisnik novak treba dobiti dozvole:
 - pregled svih podataka u relaciji student
 - izmjena poštanskog broja i adrese u relaciji student
- korisnik kolar treba dobiti dozvolu:
 - pregled svih podataka u relaciji student, osim adrese

Primjer 1 (nastavak):

bpadmin ← naredbe obavlja korisnik bpadmin

```
CREATE DATABASE studBaza;
CREATE TABLE student (...);
CREATE TABLE ispit (...);

GRANT CONNECT TO orvat;
GRANT CONNECT TO nova;
GRANT CONNECT TO kolar;

GRANT SELECT ON student
TO orvat;
GRANT SELECT, INSERT,
UPDATE, DELETE ON ispit
TO orvat;

GRANT SELECT ON student
TO nova;
GRANT UPDATE pbr, adresa
ON student TO nova;

GRANT SELECT matBr, ime,
prez, pbr
ON student TO kolar;
```

→ korisnik bpadmin je vlasnik baze podataka studBaza i relacija student i ispit. Posjeduje DBA dozvolu na razini baze podataka

→ dozvole za uspostavljanje Sjednice

→ dozvole korisniku horvat za pregled podataka u relaciji student

→ dozvole korisniku horvat za pregled, unos, izmjenu i brisanje podataka u relaciji ispit

→ dozvola korisniku novak za pregled podataka u relaciji student

→ dozvola korisniku novak za izmjenu vrijednosti atributa u relaciji student

→ dozvola korisniku kolar za pregled svih podataka u relaciji student, osim adrese

Primjer 2:

bpadmin

```
CREATE DATABASE studBaza;
GRANT RESOURCE TO orvat;
GRANT CONNECT TO nova;
```

→ korisnik bpadmin kreira bazu podataka studBaza. Kao vlasnik baze podataka implicitno dobiva DBA dozvolu na razini baze podataka

orvat

```
CREATE TABLE zupanja (
siup INTEGER,
nazup CHAR(20),
PRIMARY KEY(siup));
GRANT SELECT, INSERT, UPDATE
ON zupanja TO nova;
```

→ može jer ima RESOURCE dozvolu

→ može jer je vlasnik relacije zupanja

nova

```
SELECT * FROM zupanja;
INSERT INTO zupanja ...;
UPDATE zupanja ...;
```

→ može jer ima barem SELECT dozvolu (bez SELECT dozvole ne bi mogao uspostaviti Sjednicu), te dozvole koje je dobio od vlasnika relacije zupanja

Primjer 2 (nastavak):

nova

```
DROP TABLE zupanja;
```

→ ne može jer nije vlasnik objekta niti ima DBA dozvolu

kolar

```
SELECT * FROM zupanja;
```

→ ne može jer nema niti SELECT dozvolu (ne može uspostaviti Sjednicu)

orvat

```
GRANT CONNECT TO kolar;
```

→ ne može jer nema DBA dozvolu

bpadmin

```
GRANT CONNECT TO kolar;
```

→ može jer ima DBA dozvolu

orvat

```
GRANT SELECT
ON zupanja TO kolar;
```

→ može jer je vlasnik relacije zupanja

kolar

```
SELECT * FROM zupanja;
```

→ može jer ima barem SELECT dozvolu, te dozvolu za obavljanje operacije SELECT nad relacijom zupanja

Primjer (nastavak):

nova
`CREATE TABLE mjesto ...` → ne može jer nema RESOURCE dozvolu

orvat
`GRANT RESOURCE TO nova` → ne može jer nema DBA dozvolu

bpadmin
`GRANT DBA TO orvat` → može jer ima DBA dozvolu

orvat
`GRANT RESOURCE TO nova` → može jer ima DBA dozvolu

nova
`CREATE TABLE mjesto ...
 REFERENCES zupanja ...` → ne može jer nema dozvolu za kreiranje stranog ključa koji se poziva na primarni ključ relacije zupanja (mogao bi kreirati relaciju bez stranog ključa jer ima RESOURCE dozvolu)

Primjer (nastavak):

orvat
`GRANT REFERENCES ON zupanja TO nova` → može jer ima DBA dozvolu (ali čak i da nema DBA dozvolu, vlasnik je relacije zupanja)

nova
`CREATE TABLE mjesto ...
 REFERENCES zupanja ...` → može jer ima RESOURCE dozvolu i dozvolu za kreiranje stranog ključa koji se poziva na primarni ključ relacije zupanja

orvat
`GRANT CONNECT TO PUBLIC` → može jer ima DBA dozvolu

- sada svaki korisnik (sadašnji ili budući) koji uspije ovjeriti svoju autentičnost može uspostaviti Sjednicu s bazom podataka studBaza

nova
`GRANT SELECT ON mjesto TO PUBLIC` → može jer je vlasnik relacije mjesto

- sada svaki korisnik (sadašnji ili budući) koji uspostavi Sjednicu s bazom podataka (uz prethodnu ovjeru autentičnosti) može obavljati operaciju SELECT nad relacijom mjesto

odjeljivanje prenosivih dozvola

- ukoliko se korisniku dozvola dodijeli uz navođenje opcije `WITH GRANT OPTION`, korisnik će moći dodjeljivati tu istu dozvolu ostalim korisnicima (unatoč tome što nije vlasnik objekta)

Primjer:

orisni
`CREATE TABLE ispit ...
 GRANT SELECT ON ispit TO orisni WITH GRANT OPTION
 GRANT SELECT ON ispit TO orisni WITH GRANT OPTION`

orisni
`GRANT SELECT ON ispit TO orisni WITH GRANT OPTION
 GRANT SELECT ON ispit TO orisni`

orisni
`GRANT SELECT ON ispit TO orisni`

orisni
`GRANT SELECT ON ispit TO orisni`

orisni
`GRANT SELECT ON ispit TO orisni`

Diagram: vlasnik korisnik1 → korisnik2, korisnik3, korisnik4, korisnik5, korisnik6. Legenda: `GRANT OPTION`

kidanje dozvola

- korisnik koji je dozvolu dodijelio, tu istu dozvolu može ukinuti naredbom `REVOKE`

Primjer:

- vlasnik baze podataka studBaza je korisnik bpadmin
- vlasnik relacije mjesto je korisnik orvat

orvat
`GRANT SELECT, UPDATE ON mjesto TO nova WITH GRANT OPTION`

nova
`GRANT SELECT, UPDATE ON mjesto TO orlar`

- npr. naredbu: `REVOKE UPDATE ON mjesto FROM orlar`
- može obaviti korisnik novak jer je novak korisnik koji je dozvolu dodijelio

kidanje dozvola dodijeljenih temeljem WITH GRANT OPTION

- ukidanjem dozvole korisniku (koji je dozvole dalje dodjeljivao temeljem ovlasti stečene pomoću `WITH GRANT OPTION`) uz primjenu opcije `CASCADE`, dozvola se ukida i svim ostalim korisnicima koji su dotičnu dozvolu stekli od korisnika (neposredno ili posredno)

Primjer: **orisni**
`REVOKE SELECT ON ispit FROM orisni CASCADE`

Diagram: korisnik1 → korisnik2, korisnik3, korisnik4, korisnik5, korisnik6. Legenda: `GRANT OPTION`

- obavljanjem naredbe dozvolu gube korisnik2, korisnik4 i korisnik6
- korisnik5 će izgubiti dozvolu koju je dobio od korisnika2, ali će zadržati dozvolu koju je dobio od korisnika1
- ukoliko se opcija `CASCADE` ne navede, naredba `REVOKE` djeluje na jednak način kao kada je opcija `CASCADE` navedena

kidanje dozvola dodijeljenih temeljem WITH GRANT OPTION

- ukidanjem dozvole korisniku uz primjenu opcije `RESTRICT`, dozvola će biti ukinuta jedino u slučaju kada korisnik nije dalje dodjeljivao ovlasti temeljem ovlasti stečene pomoću `WITH GRANT OPTION`

Primjer:

Diagram: korisnik1 → korisnik2, korisnik3, korisnik4, korisnik5, korisnik6. Legenda: `GRANT OPTION`

orisni
`REVOKE SELECT ON ispit FROM orisni RESTRICT`
 SUBP odbija obaviti naredbu (dodjeljuje pogrešku)

orisni
`REVOKE SELECT ON ispit FROM orisni RESTRICT`
 SUBP odbija obaviti naredbu (dodjeljuje pogrešku)

orisni
`REVOKE SELECT ON ispit FROM orisni RESTRICT`
 SUBP obavlja naredbu (korisnik3 ostaje bez dozvole)

Primjena virtualnih relacija

ispit			
mbrSt	nazPred	datIspr	ocj
100	Fizika	1.5.2004	3
102	Matematika	7.9.2003	1
102	Matematika	9.2.2004	5
107	Fizika	5.4.2006	4

- vlasnik relacije ispit je korisnik horvat
- korisniku novak omogućiti pregled samo prosječnih ocjena po predmetima
- korisniku kolar omogućiti pregled, unos, izmjenu i brisanje samo za ispite iz predmeta Fizika

horvat

```
CREATE VIEW prosje AS
SELECT nazPred, AVG(ocj)
FROM ispit
GROUP BY nazPred;

GRANT SELECT ON prosje TO nova;

CREATE VIEW ispitFizika AS
SELECT * FROM ispit
WHERE nazPred = 'Fizika';

GRANT SELECT, INSERT, UPDATE, DELETE
ON ispitFizika TO kolar;
```

- zašto je nužno virtualnu relaciju ispitFizika kreirati uz opciju WITH CHECK OPTION?

odjeljivanje kontekstno ovisnih dozvola

ispit				nast				predaje		
mbrSt	sifPred	datIspr	ocj	sifNast	imeN	prezN	userId	sifNast	sifPred	
100	100	1.5.2004	3	1001	Slavko	Kolar	kolar	1001	100	
102	200	7.9.2003	1	1002	Ivo	Ban	ban	1001	200	
102	200	9.2.2004	5	1003	Ana	Novak	novak	1002	200	
107	300	5.4.2006	4					1003	200	
								1003	300	

- vlasnik relacija je korisnik horvat
- svakom nastavniku (korisnicima kolar, ban, novak) omogućiti pregled i izmjenu ispita samo iz predmeta koje predaju

horvat

CREATE VIEW

```
CREATE VIEW kolarIspiti AS
SELECT * FROM ispit
WHERE sifPred IN (
SELECT sifPred FROM predaje
WHERE sifNast = 1001);

GRANT SELECT, UPDATE ON kolarIspiti TO kolar;
```

- ponoviti za svakog nastavnika: banIspiti, novakIspiti, ...
- nova virtualna relacija za svakog novog nastavnika (≈150 na FER-u)
- svaki nastavnik upit nad relacijom ispit mora pisati na drugačiji način

odjeljivanje kontekstno ovisnih dozvola

ispit				nast				predaje		
mbrSt	sifPred	datIspr	ocj	sifNast	imeN	prezN	userId	sifNast	sifPred	
100	100	1.5.2004	3	1001	Slavko	Kolar	kolar	1001	100	
102	200	7.9.2003	1	1002	Ivo	Ban	ban	1001	200	
102	200	9.2.2004	5	1003	Ana	Novak	novak	1002	200	
107	300	5.4.2006	4					1003	200	
								1003	300	

horvat

CREATE VIEW

```
CREATE VIEW ispitNastavnici AS
SELECT * FROM ispit
WHERE sifPred IN (
SELECT sifPred FROM predaje, nast
WHERE predaje.sifNast = nast.sifNast
AND userId = 1001);

GRANT SELECT, UPDATE ON ispitNastavnici TO kolar;
GRANT SELECT, UPDATE ON ispitNastavnici TO ban;
GRANT SELECT, UPDATE ON ispitNastavnici TO nova;
```

- sadržaj virtualne relacije ovisit će o identifikatoru nastavnika koji je ostvario sifNast
- smije li se nastavnicima dozvoliti izmjena vrijednosti atributa userId u relaciji nast ili sadržaj relacije predaje?!

potreba sinonima

PROBLEM:

- nastavnici (odnosno aplikativni ili primjenski programi koje nastavnici koriste) moraju u upitima o ispitima koristiti virtualnu relaciju ispitZaNastavnike

```
SELECT * FROM ispitZaNastavnici WHERE ocj = 5
```

- dekan (npr. korisnik s identifikatorom novosel), za razliku od nastavnika, dobiva sve dozvole nad relacijom ispit. U upitima o ispitima mora koristiti relaciju ispit

```
SELECT * FROM ispit WHERE ocj = 5
```

- kada korisnik novosel prestane biti dekan, ukinut će mu se dozvola nad relacijom ispit, a dodijeliti dozvola nad virtualnom relacijom ispitZaNastavnike. U svojim upitima morat će koristiti virtualnu relaciju ispitZaNastavnike

```
SELECT * FROM ispitZaNastavnici WHERE ocj = 5
```

potreba sinonima

RJEŠENJE:

- Kreirati sinonime: alternativna imena za relacije ili virtualne relacije

```
CREATE PRIVATE SYNONYM kolar.ispitZaNastavnici FOR ispitZaNastavnici;
CREATE PRIVATE SYNONYM ban.ispitZaNastavnici FOR ispitZaNastavnici;
... sinonimi za ostale nastavnike i sinonim za deкана
CREATE PRIVATE SYNONYM novosel.ispitZaNastavnici FOR ispitZaNastavnici;
```

- sada i dekan i nastavnici mogu koristiti isto ime objekta kada postavljaju upite o ispitima

```
SELECT * FROM ispitZaNastavnici WHERE ocj = 5
```

- kada korisnik novosel prestane biti dekan

```
CREATE PRIVATE SYNONYM novosel.ispitZaNastavnici FOR ispitZaNastavnici;
GRANT SELECT, UPDATE ON ispitZaNastavnici TO novosel;
```

korisnik s DBA dozvolom

```
CREATE PRIVATE SYNONYM novosel.ispitZaNastavnici FOR ispitZaNastavnici;
```

- korisnik novosel će i dalje u svojim upitima moći koristiti ime objekta ispitZaNastavnici, ali će kao rezultat dobivati samo one podatke na koje, sada u svojstvu nastavnika, ima pravo

odjeljivanje istih dozvola velikom broju korisnika

PROBLEM:

- svakom nastavniku treba dodijeliti dozvole za
 - pregled, unos i izmjenu podataka o ispitima za predmete koje predaje, pregled podataka iz relacije nast, iz relacije predaje, itd.
 - 150 nastavnika ⇒ 150 puta treba obaviti niz naredbi za dodjelu dozvola:

```
GRANT SELECT, INSERT, UPDATE ON ispitZaNastavnici TO kolar;
GRANT SELECT ON predmet TO kolar;
GRANT SELECT ON nast TO kolar;
...
ponoviti za sva 150 od 150 nastavnika
```

- za svakog novog zaposlenog nastavnika ponoviti postupak
- kada nastavnik ode u mirovinu, mora se obaviti niz REVOKE naredbi
- ako se promijene pravila pristupa (npr. odluči se da nastavnici mogu brisati svoje ispite), promjena se mora provesti za svakog nastavnika posebno:

```
GRANT DELETE ON ispitZaNastavnici TO kolar;
ponoviti za sva 150 od 150 nastavnika
```


Podjeljivanje istih dozvola velikom broju korisnika

ROLE:

- definiše se uloga (*role*), npr. nastavnik
- dozvole se, umjesto direktno korisnicima-nastavnicima, dodjeljuju ulozi

```
CREATE ROLE nastavnik;  
GRANT SELECT, INSERT, UPDATE ON ispiti TO nastavnik;  
GRANT SELECT ON nast TO nastavnik;  
GRANT SELECT ON predaje TO nastavnik;  
...
```

- svakom nastavniku, umjesto cijelog niza dozvola, dovoljno je dodijeliti dozvolu za korištenje uloge nastavnik

```
GRANT nastavnik TO olar;  
GRANT nastavnik TO ban;  
...
```

- ako nastavnik s identifikatorom korisnika ban ode u mirovinu

```
REVOKE nastavnik FROM ban;
```

- ako nastavnici trebaju dobiti dozvolu za brisanje svojih ispita

```
GRANT DELETE ON ispiti TO nastavnik;
```

Korištenje dozvola dobivenih putem uloga

- nakon uspostavljanja skupine, korisnik posjeduje sljedeće dozvole:
 - sve dozvole koje su dodijeljene PUBLIC korisniku
 - sve dozvole koje su dodijeljene izravno dotičnom korisniku
 - sve dozvole nad objektima kojima je dotični korisnik vlasnik
 - dozvole na razini baze podataka (npr. ako korisnik ima DBA dozvolu, dopušteno mu je obavljanje svih operacija nad svim objektima)
- ako korisnik namjerava koristiti i dozvole dodijeljene nekoj ulozi, mora obaviti naredbu (npr.):

```
SET ROLE nastavnik;
```

 - od tog trenutka, korisnik će (osim dozvola navedenih pod 1-4) imati i dozvole dodijeljene ulozi nastavnik.
- korisniku može biti dodijeljena više nego jedna uloga, ali u jednom trenutku može koristiti samo jednu od njih. Npr. nakon obavljanja naredbe:

```
SET ROLE studentskiSavjetnik;
```

 - korisnik će (osim dozvola navedenih pod 1-4) imati i dozvole dodijeljene ulozi studentskiSavjetnik (ali ne i ulozi nastavnik).
- naredbu

```
SET ROLE NONE;
```

 korisnik koristi onda kad ne želi koristiti niti jednu ulogu

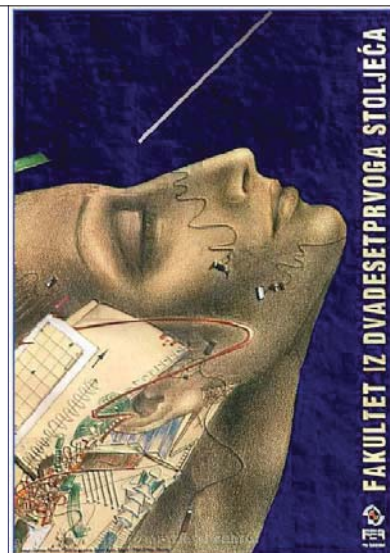
Praćenje rada korisnika (*auditing*)

- evidentirati svaki pristup osjetljivim podacima u posebnoj datoteci za praćenje rada korisnika (*Audit Trail*)
- tipičan zapis datoteke sadrži sljedeće informacije:
 - SQL naredba koja se izvršava (*statement source*)
 - mjesto s kojeg je upućen zahtjev (terminal, IP adresa računala)
 - identifikator korisnika koji je pokrenuo operaciju
 - datum i vrijeme operacije
 - n-torke, atributi na koje se zahtjev odnosi
 - stara vrijednost n-torke
 - nova vrijednost n-torke
- sama činjenica da se prati i trag obavljenih operacija nad podacima, često je dovoljna za sprečavanje zlorabe

Baze podataka

Predavanja
lipanj 2008.

1. Poshranjene procedure i okidači



Poshranjene procedure

Primjer 1:

osoba	CHAR(13)	CHAR(13)
	jmbg	prez
	1234567890123	Horvat
	4567890123456	olar
	7890123456789	eler
	0123456789012	Nova

- smatra se da su ispravne one vrijednosti atributa jmbg u kojima postoji točno 13 znamenaka
- smatra se da su ispravna ona prezimena u kojima ne postoji niti jedna znamenka
- ispisati podatke o osobama s neispravnim jmbg ili prezimenom
- kad bi barem postojala SQL funkcija CountDigits(nizZnakova)

```
SELECT * FROM osoba  
WHERE CountDigits(jmbg) < 13  
OR CountDigits(prez) > 13
```

Pohranjene procedure (pohranjene funkcije)

- Pohranjena procedura ili pohranjena funkcija je potprogram koji je pohranjen u rječniku podataka i koji se izvršava u kontekstu sustava za upravljanje bazama podataka
 - može se promatrati kao procedura ili funkcija kojom se proširuje skup SQL funkcija ugrađenih u SUBP
 - procedura je potprogram koji u pozivajući program ne vraća rezultat
 - funkcija je potprogram koji u pozivajući program vraća rezultat

Primjer 1 (nastavak):

- Funkcija koja u zadanom nizu znakova broji koliko ima znakova koji su znamenke (broji znakove iz intervala '0' ... '9'). Pretpostavlja se da duljina zadanog niza znakova ne premašuje 255 bajtova

```
CREATE FUNCTION brojZnamenki (niz CHAR(255))
RETURNS SMALLINT AS broj
BEGIN
    DECLARE broja INT;
    SET broja = 0;
    FOR i = 1 TO CHAR_LENGTH(niz)
        IF SUBSTR(niz, i, 1) BETWEEN '0' AND '9' THEN
            SET broja = broja + 1;
        END IF;
    END FOR;
    RETURN broja;
END FUNCTION;

GRANT EXECUTE ON brojZnamenki TO PUBLIC;
```

- funkciju brojZnamenki svaki (sadašnji i budući) korisnik može koristiti na jednak način kao što se koriste standardne SQL funkcije

Primjer 1 (nastavak):

osoba	CHAR(13)	CHAR(13)
	jmbg	prez
	1234567890123	Horvat
	1234567890123	Polar
	1234567890123	Hefer
	12AB345.67890	Nova

- funkcija brojZnamenki se može iskoristiti za ispis onih osoba u čijem jmbg nema točno 13 znamenaka ili u prezimenu postoje znamenke

```
SELECT *, brojZnamenki(jmbg) AS br1, brojZnamenki(prez) AS br2
FROM osoba
WHERE brojZnamenki(jmbg) <> 13 OR brojZnamenki(prez) > 0
```

jmbg	prez	br1	br2
1234567890123	Polar	13	0
1234567890123	Hefer	13	0
12AB345.67890	Nova	12	1

Primjer 1 (nastavak):

- Pohranjena funkcija se iz interaktivnih alata (npr. Aqua Data Studio) može pozvati na sljedeći način:

```
EXECUTE FUNCTION brojZnamenki('12345678901234567890')
```

broj
6

```
CREATE FUNCTION brojZnamenki (niz CHAR(255))
RETURNS SMALLINT AS broj
...
```

Primjer 2:

- Korisnik novak je službenik u banci kojem je potrebno omogućiti obavljanje **isključivo** jedne vrste bankovne transakcije: prebacivanje iznosa s jednog na drugi račun

racun	brRacun	stanje
	1001	1250.15
	1002	-300.00
	1003	10.25

- Zadatak se ne može riješiti dodjelom dozvole za obavljanje operacije UPDATE nad relacijom racun korisniku novak (**zašto?**)

Dozvole za pohranjene procedure/funkcije

- SQL naredbe za dodjeljivanje i ukidanje dozvola za izvršavanje procedura

```
GRANT EXECUTE ON {procName | funName}
TO {PUBLIC | userList | roleList}
[WITH GRANT OPTION]
```

```
REVOKE EXECUTE ON {procName | funName}
FROM {PUBLIC | userList | roleList}
[CASCADE | RESTRICT]
```

Primer na zbirku

```
CREATE PROCEDURE prebraci saRacunaBr LIKE racun.brRacun  
    naRacunBr LIKE racun.brRacun  
    iNos LIKE racun.stanje  
  
-- prenesi aani iNos  
UPDATE racun SET stanje = stanje - iNos  
WHERE brRacun = saRacunaBr  
UPDATE racun SET stanje = stanje + iNos  
WHERE brRacun = naRacunBr  
END PROCEDURE  
GRANT EXECUTE ON prebraci TO noa
```

Primer na zbirku

racun	brRacun	stanje
	1001	1250.15
	1002	-300.00
	1003	10.25

```
noa UPDATE racun SET stanje = stanje - 0.30  
WHERE brRacun = 1001
```

[Error] No UPDATE permission

```
noa EXECUTE PROCEDURE prebraci 1001 1002 0.30
```

racun	brRacun	stanje
	1001	1119.85
	1002	-230.00
	1003	10.25

- Problem: što će se dogoditi ako korisnik pri pozivu procedure kao broj prvog računa zada postojeći a kao broj drugog računa zada nepostojeći broj računa?

```
EXECUTE PROCEDURE prebraci 1001 1005 30.15
```

Značajke Exceptions

- ukoliko SUBP tijekom obavljanja operacije utvrdi da se dogodila pogreška (*error condition*) obavljanje operacije se prekida a stanje pogreške se signalizira iznimkom (*exception*)

```
SELECT stanje - stanje - 10.25 FROM racun
```

[Error] An attempt was made to divide by zero.

```
SELECT * FROM ispit
```

[Error] No SELECT permission.

- pogreške koje SUBP nije u stanju prepoznati (jer ih ne smatra pogreškama) mogu se signalizirati naredbom **RAISE EXCEPTION** u poboljšanoj proceduri **prebraci** signalizira se pogreška u slučaju kad ne postoji neki od zadanih brojeva računa

```
EXECUTE PROCEDURE prebraci 1001 1005 30.15
```

[Error] Ne postoji krući račun

Primer na zbirku

```
CREATE PROCEDURE prebraci saRacunaBr LIKE racun.brRacun  
    naRacunBr LIKE racun.brRacun  
    iNos LIKE racun.stanje  
  
-- proveri postoje li aani i rođeni računa  
IF (SELECT COUNT(*) FROM racun  
    WHERE brRacun = saRacunaBr = 0 THEN  
    RAISE EXCEPTION -1000 0 'Ne postoji prvi račun'  
END IF  
IF (SELECT COUNT(*) FROM racun  
    WHERE brRacun = naRacunBr = 0 THEN  
    RAISE EXCEPTION -1000 0 'Ne postoji drugi račun'  
END IF  
-- prenesi aani iNos  
UPDATE racun SET stanje = stanje - iNos  
WHERE brRacun = saRacunaBr  
UPDATE racun SET stanje = stanje + iNos  
WHERE brRacun = naRacunBr  
END PROCEDURE  
GRANT EXECUTE ON prebraci TO noa
```

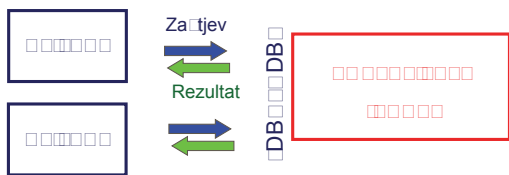
Stored Procedure Language

- Proizvođači SUBP koriste vlastite inačice jezika za definiranje počinjenih procedura (standard postoji ali je rijetko gdje implementiran)
 - IBM Informatica SPL (Stored Procedure Language)
 - Oracle: PL/SQL (Procedural Language/Structured Query Language)
 - Microsoft SQL Server: Transact-SQL
- avedeni jezici proširuju mogućnosti SQL jezika proceduralnim elementima koji se koriste u strukturiranim jezicima (Java, Python, ...)
 - sim SQL naredbi počinjene procedure omogućuju korištenje
 - varijabli
 - naredbi za kontrolu toka programa (*if, for, while, ...*)
 - naredbi za rukovanje iznimkama (*exception handling*)

Redno i upotreba pohranjenih procedura

- proširenje mogućnosti SQL jezika
- omogućena je zaštita podataka na razini funkcije (a ne samo objekta)
- omogućena je uporaba klijent-poslužitelj arhitekture oslonjene na poslužitelj:
 - postigne se veća učinkovitost SUBP
 - SUBP ne mora ponavljati prevođenje i optimiranje SQL upita
 - postigne se veća produktivnost programera i smanjuje mogućnost pogreške
 - programski kod potreban za obavljanje nekog postupka koji čini logičku cjelinu implementira se i testira na samo jednom mjestu

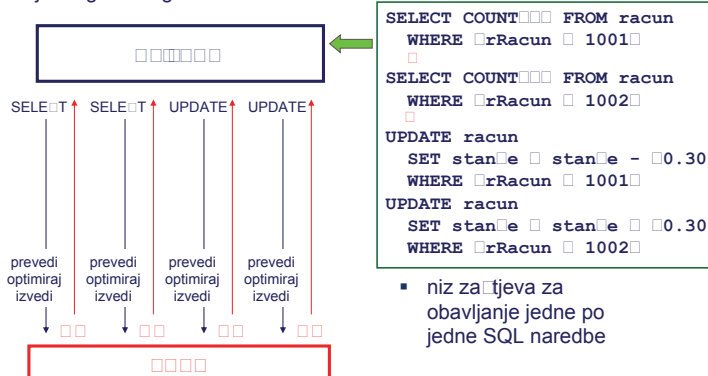
klasični model arhitekture



- sustav obuhvaća dvije komponente
 - klijent i poslužitelj (*client-server*)
- koncept zahtjev-odgovor (*request-response*): klijent postavlja zahtjev, poslužitelj odgovara
- komunikacija između klijenta i poslužitelja se odvija preko dobro definiranih standardnih programskih sučelja: npr. *ODBC* (*Open Database Connectivity*), *JDBC* (*Java Database Connectivity*)

klasični model arhitekture proširen na klijenta

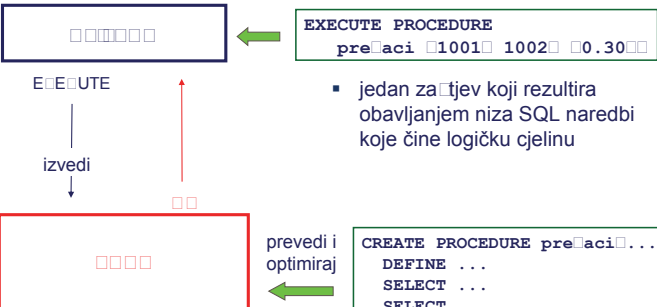
- provjeri postoje li zadani brojevi računa, ako postoje prebaci iznos s jednog na drugi račun



- niz zahtjeva za obavljanje jedne po jedne SQL naredbe

klasični model arhitekture proširen na poslužitelja

- provjeri postoje li zadani brojevi računa, ako postoje prebaci iznos s jednog na drugi račun



- jedan zahtjev koji rezultira obavljanjem niza SQL naredbi koje čine logičku cjelinu

- U rječnik podataka se pohranjuje:
 - izvorni kod procedure
 - prevedeni i optimirani kod procedure

```

CREATE PROCEDURE prebaci...
DEFINE ...
SELECT ...
SELECT ...
UPDATE ...
UPDATE ...
END PROCEDURE
    
```

okidači

primjer

racun	brRac	siKlijent	stanje	uplata	splata	brRac	vrijeme	iznos
001	9828	2	80			001	7.8.2007 08:20	0.00
002	897	1	99			002	9.9.2007 12:00	-0.02
001	2	2	27	0.00		001	11.2007 11:00	2.00
001			8	9.00		001	9.9.2007 10:00	-0.00
						002	7.2.2007 11:00	2.20
						001	11.2007 12:00	27.00
						001	19.2007 12:00	-2.00

- u relaciju *uplata_splata* upisuju se promjene na računima
- tijekom godina evidentiran je vrlo veliki broj uplata i isplata
- stanje na određenom računu moglo bi se izračunati zbrajanjem iznosa u relaciji *uplata_splata* koji se odnose na dotični račun
- u ovom primjeru uz svaki račun se redundantno pohranjuje trenutno stanje računa koje u svakom trenutku mora odgovarati stanju koje bi se dobilo zbrajanjem iznosa u relaciji *uplata_splata*
- kako osigurati da se pri svakoj relevantnoj promjeni podataka (unos, brisanje, izmjena iznosa) u relaciji *uplata_splata* izmijeni i odgovarajuće stanje u relaciji *racun*?

ključne baze podataka

- konvencionalni SUBP je pasivan
 - operacije nad podacima se izvršavaju isključivo na temelju eksplicitnog zahtjeva korisnika/aplikacije
- aktivni SUBP i aktivne baze podataka
 - aktivni SUBP autonomno reagira na određene događaje (*events*)
 - u aktivnim bazama podataka neke operacije nad podacima se izvršavaju automatski reakcijom na određeni događaj ili stanje
- željeno ponašanje sustava postiže se definiranjem aktivnih pravila (*active rules*)
- najčešće korištena paradigma za opisivanje aktivnih pravila u današnjim SUBP je događaj-uvjet-akcija (*ECA: Event-Condition-Action*)
 - okidači (*triggers*)

on **event**
if **condition** then **action**

- događaj (**event**): ako se dogodi izračunava se uvjet

□ općenito □ događaji mogu biti:

- unos □ izmjena ili brisanje podatka
- čitanje podatka
- uspostavljanje SQL-sjednice
- protok određene količine vremena □ dostizanje trenutka u vremenu □□□

- uvjet (**condition**): ako je rezultat izračunavanja uvjeta istina □ obavljaju se akcije

□ zadaje se u obliku predikata (slično kao u □ □ERE dijelu SQL naredbi)

- akcije (**action**): niz operacija □ najčešće operacije nad podacima

□ SQL naredbe □ INSERT □ UPDATE □ DELETE □ poziv procedure □□□

riječ naavak

- kako osigurati da se pri svakoj relevantnoj promjeni podataka (unos □ brisanje □ izmjena iznosa) u relaciji **uplata:splata** izmijeni i odgovarajuće stanje u relaciji **racun**?

racun	brRac	si	lijent	stanje	uplata:splata	brRac	vrijeme	iznos
	000		9828	2	80	000	7.8.2007 08:20	000
	002		897		99	002	9.2007 2:00	-002
						000	2007	2

- potrebno je utvrditi koji događaji mogu uzrokovati neispravnu vrijednost atributa stanje u relaciji racun □ te pod kojim uvjetima treba obaviti koje akcije kako bi se očuvao integritet podataka □ npr □

- događaj: obavljanje operacije □ INSERT nad relacijom uplata:splata
- uvjet: iznos □ 0.00
- akcija: pribrojiti vrijednost atributa iznos unesene n-torke u odgovarajuće stanje

riječ naavak

- događaj: obavljanje operacije □ INSERT nad relacijom uplata:splata
- uvjet: iznos □ 0.00
- akcija: pribrojiti vrijednost atributa iznos unesene n-torke u odgovarajuće stanje

```
CREATE TRIGGER insUplataIsplata
INSERT ON uplataIsplata
REFERENCING NEW AS no□aUplataIsplata
FOR EACH ROW
WHEN □no□aUplataIsplata.i□nos □ 0
UPDATE racun SET stanje □ stanje □ no□aUplataIsplata.i□nos
WHERE □rRac □ no□aUplataIsplata.□rRac
```

- kad god se obavi naredba □ INSERT nad relacijom uplata:splata SUBP obavlja
 - nakon unosa svake n-torke (jednom □ INSERT naredbom može se unijeti više n-torki) provjerava uvjet □no□aUplataIsplata.i□nos □ 0
 - na sadržaj unesene n-torke može se referencirati koristeći ime □ n-torke koje je zadano pomoću REFERENCING NEW AS □no□aUplataIsplata
 - ako je uvjet zadovoljen (za dotičnu n-torku) □ obavlja izmjenu stanja u relaciji racun

riječ naavak

- događaj: brisanje n-torke iz relacije uplata:splata
- uvjet: iznos □ 0.00
- akcija: oduzeti vrijednost atributa iznos unesene n-torke od odgovarajućeg stanja

```
CREATE TRIGGER delUplataIsplata
DELETE ON uplataIsplata
REFERENCING OLD AS □risanaUplataIsplata
FOR EACH ROW
WHEN □□risanaUplataIsplata.i□nos □ 0
UPDATE racun SET stanje □ stanje - □risanaUplataIsplata.i□nos
WHERE □rRac □ □risanaUplataIsplata.□rRac
```

- ukoliko je potrebno □ moguće je navesti više SQL naredbi □ međusobno odijeljeni □ zarezima
- SQL naredbe koje se mogu koristiti za opisivanje akcije:
 - INSERT
 - UPDATE
 - DELETE
 - EXECUTE PROCEDURE

riječ naavak

- događaj: izmjena vrijednosti atributa iznos u relaciji uplata:splata
- uvjet: nova vrijednost iznosa □ stara vrijednost iznosa
- akcija: u odgovarajuće stanje pribrojiti razliku između nove i stare vrijednosti atributa iznos

```
CREATE TRIGGER up□i□nosUplataIsplata
UPDATE OF i□nos ON uplataIsplata
REFERENCING OLD AS staraUplataIsplata NEW AS no□aUplataIsplata
FOR EACH ROW
WHEN □no□aUplataIsplata.i□nos □ staraUplataIsplata.i□nos
UPDATE racun SET stanje □ stanje □
no□aUplataIsplata.i□nos - staraUplataIsplata.i□nos
WHERE □rRac □ staraUplataIsplata.□rRac
```

- UPDATE OF i□nos ON uplataIsplata: događaj izmjene vrijednosti atributa iznos u relaciji uplata:splata
- UPDATE OF a□ □ c ON relacija: događaj izmjene vrijednosti bilo kojeg od atributa a□ □ a□ □ u relaciji
- UPDATE ON relacija: događaj izmjene vrijednosti bilo kojeg atributa u relaciji

naredba □□□□□□□□□□

- oblik naredbe za kreiranje okidača propisan je SQL standardom □ ali SUBP koriste uglavnom vlastite inačice
- jedna od važniji □ mogućnosti koje su na raspolaganju pri de□iniciji okidača:
 - moguće je speci□ificirati da li se akcije navedene u okidaču obavljaju:
 - po jednom za svaku n-torku na koju je djelovala operacija koja je aktivirala okidač (operacija koja je uzrokovala događaj)
 - FIRST □ R□□
 - samo jednom □ nakon što se obavi operacija koja je aktivirala okidač
 - AFTER □ INSERT □ AFTER UPDATE □ AFTER DELETE
 - samo jednom □ prije nego se obavi operacija koja je aktivirala okidač
 - BEFORE □ INSERT □ BEFORE UPDATE □ BEFORE DELETE
 - uništavanje okidača: DROP TRIGGER imeOkidača

rijenja okidača

- implementacija integritetskog ograničenja
 - okidače treba koristiti onda kada integritetska ograničenja nije moguće opisati na drugi način (PRIMARY KEY, FOREIGN KEY, UNIQUE, ...)
 - obavljanjem korektivne akcije koja bazu podataka dovodi u konzistentno stanje (primjer)
 - odbijanjem operacije koja narušava integritetsko ograničenje (primjer)
- praćenje rada korisnika (primjer)
- sustavi obavještanja (primjer)
- itd.

rijer

- u relaciji ispit osigurati integritetsko ograničenje prema kojem je promjena ocjena dopuštena samo ako se mijenja na nižu ocjenu
 - dopušteno je ocjenu izvrstan promijeniti u dobar
 - nije dopušteno ocjenu dovoljan promijeniti u vrlo dobar

ispit	matBr	siPred	datSp	ocj	siAst
00	00	29.0.200			
00	00	0.02.200			
0	002	27.0.200	2	2222	
02	00	29.0.200			2222

- očito je da ne postoji korektivna akcija koja bi bazu podataka mogla dovesti u konzistentno stanje nakon što korisnik obavi naredbu:

```
UPDATE ispit SET ocjena = 2
WHERE ocjena = 1
```

- jedini način na koji se može osigurati navedeno integritetsko ograničenje jest: odbiti izvršavanje takve naredbe

rijer naavak

```
CREATE PROCEDURE oaiPoresuUecaneOcene
RAISE EXCEPTION - '0' Ocjena se ne smije ućati
END PROCEDURE
```

```
CREATE TRIGGER upOcIspit
UPDATE OF oc ON ispit
REFERENCING OLD AS stariIspit NEW AS noiIspit
FOR EACH ROW
WHEN (noiIspit.oc = stariIspit.oc)
EXECUTE PROCEDURE oaiPoresuUecaneOcene
```

- što se dešava pri izvršavanju naredbe
- nakon promjene prve n-torke akcije iz okidača se neće obaviti jer uvjet za obavljanje akcije nije ispunjen
- nakon promjene druge n-torke aktivirat će se akcija iz okidača
 - poziva se procedura
 - procedura signalizira pogrešku
 - budući da se naredba mora obaviti u cjelosti ili uopće ne sustav poništava i promjenu prve n-torke korisniku prikazuje opis pogreške

```
UPDATE ispit SET oc = 2
WHERE matBr = 100
```

rijer

- pretpostavi li se da je izmjena podataka u relaciji racun naročito osjetljiva operacija - potrebno je pratiti rad korisnika (audit trail)

```
CREATE TABLE auditTrailaRacun
(orisni CHAR(32),
rieme DATETIME(6) EAR TO SECOND,
rRac1 ...,
sifKliEnt1 ...,
stan1 ...,
rRac2 ...,
sifKliEnt2 ...,
stan2 ...)

```

```
CREATE TRIGGER upRacun
UPDATE ON racun
REFERENCING OLD AS stari NEW AS noi
FOR EACH ROW
-- u et se mo e ispustiti
INSERT INTO auditTrailaRacun VALUES
(USER CURRENT, stari.rRac, stari.sifKliEnt, stari.stan1,
noi.rRac, noi.sifKliEnt, noi.stan2)
```

rijer naavak

```
...
FOR EACH ROW
INSERT INTO auditTrailaRacun VALUES
(USER CURRENT, stari.rRac, stari.sifKliEnt, stari.stan1,
noi.rRac, noi.sifKliEnt, noi.stan2)
```

- što se dešava obavljanjem naredbe

```
noa
UPDATE racun
SET stan1 = stan2 + 10
WHERE rRac BETWEEN 1002 AND 1003
```

racun	brRac	siIlijent	stanje
00	9828	2	80
002	897		99
00	2	2	27:0:00
00			8:9:00

- osim promjene u relaciji racun u relaciju auditTrailZaRacun bit će dodane dvije n-torke

auditTrailZaRacun								
korisnik	vrijeme	brRac	siIlijent	stanje	brRac2	siIlijent2	stanje2	
novak	2007-02-27 17:17:00	002	897	99	002	897	99	
novak	2007-02-27 17:17:00	00	2:2:2	27:0:00	00	2:2:2	27:0:00	

rijer

- postoji poranjena procedura saljiPostu(adresa, tekst)
- u relaciji artikl nalaze se podaci o artiklima na skladištu. Za svaki artikl prati se trenutno stanje (količina) artikla
- kada stanje artikla padne ispod optimalne količine potrebno je na e-mail adresu djelatnika zaduženog za nabavu tog artikla poslati poruku

artikl	siArt	stanje	optKol	adresaZaduzenog
00	2	0	pero	tvrtka
002	00	200	joza	tvrtka
00	0	0	jura	tvrtka

```
CREATE TRIGGER upArti1
UPDATE OF stanje ON arti1
REFERENCING OLD AS stari NEW AS noi
FOR EACH ROW
WHEN (stari.stanje = stari.optKol
AND noi.stanje < stari.optKol)
EXECUTE PROCEDURE saljiPostu(stari.adresa, aueno,
Naai arti1, stari.sifArt)
```


riječ naavak

artikl	si.Art	stanje	opt.ol	adresaZaduzenog
000	20	0	0	pero tvrtka
002	00	200	0	joza tvrtka
000	0	0	0	jura tvrtka

- rezultat obavljanja naredbe

```
UPDATE artikl
SET stanje = stanje - 150
```

- dvije poruke

artikl	si.Art	stanje	opt.ol	adresaZaduzenog
000	00	0	0	pero tvrtka
002	20	200	0	joza tvrtka
000	00	0	0	jura tvrtka

pero tvrtka: abavi artikl: 00
jura tvrtka: abavi artikl: 00

- ako se nakon toga obavi naredba

```
UPDATE artikl
SET stanje = stanje - 100
```

- poruka

artikl	si.Art	stanje	opt.ol	adresaZaduzenog
000	0	0	0	pero tvrtka
002	0	200	0	joza tvrtka
000	200	0	0	jura tvrtka

joza tvrtka: abavi artikl: 002

RATPRUČ ZA SPL

reiranje pohranjene procedure

- Procedura se kreira SQL naredbom oblika:

```
CREATE PROCEDURE imeProcedure (eventualni argumenti)
tielo procedure
END PROCEDURE
```

```
CREATE FUNCTION imeFunkcije (eventualni argumenti)
tielo funkcije
END FUNCTION
```

- Eventualne pogreške u sintaksi naredbe sustav će dojaviti za vrijeme obavljanja naredbe (na isti način kao i pogreške za vrijeme obavljanja ostalih SQL naredbi)

- Brisanje (uništavanje) procedure

```
DROP PROCEDURE imeProcedure
DROP FUNCTION imeFunkcije
```

- izmjena procedure: brisanjem starog objekta i deiniranjem novog objekta pod istim imenom

```
DROP PROCEDURE imeProcedure
CREATE PROCEDURE imeProcedure ...
```

rukura pohranjene procedure

```
CREATE PROCEDURE imeProcedure (eventualni argumenti)
definicija arija
nareja
nareja
...
END PROCEDURE
```

```
CREATE FUNCTION imeProcedure (eventualni argumenti)
definicija arija
nareja
nareja
...
END FUNCTION
```

- naredbe procedure završavaju znakom (točka-zarez)

Definicija varijabli

- Definicije varijabli se navode na početku procedure. Sadržaj varijable je nedeiniran dok mu se ne pridruži neka vrijednost
- Tipovi varijabli mogu biti deinirani eksplicitno:

```
CREATE PROCEDURE imeProcedure (eventualni argumenti)
DEFINE ime CHAR(20)
DEFINE ocjena brojila SMALLINT
...
```

- ili implicitno prema tipovima atributa u relacijama baze podataka

```
CREATE PROCEDURE imeProcedure (eventualni argumenti)
DEFINE ime LIKE student.imeStu
DEFINE ocjena LIKE ispit.ocjena
...
```

- kad god je moguće tipove varijabli treba deinirati implicitno
 - u slučaju promjene tipa podatka nekog atributa u relaciji sve što je potrebno obaviti jest ponovo prevesti procedure

naredba

- koristi se za pridruživanje vrijednosti varijablama

```
CREATE PROCEDURE ...
DEFINE r porsina DECIMAL(10,5)
DEFINE roIspita SMALLINT
DEFINE sumaOcjena INTEGER
DEFINE prosje DECIMAL(3,2)
DEFINE ronam SMALLINT

LET r = 10
LET porsina = 3.115 r r

LET roIspita = SELECT COUNT FROM ispit
LET sumaOcjena = SELECT SUMOCJENA FROM ispit
LET prosje = sumaOcjena / roIspita
LET ronam = ronameni123a/c
...
```

- rezultat obavljanja SELECT naredbe koja vraća jednu jednostavnu vrijednost (skalar) može se koristiti na svim mjestima na kojima se koriste izrazi. SELECT naredba mora biti unutar okruglih zagrada

Naredbe IF, WHILE, FOR

- naredba za jednostranu, dvostranu ili višestranu selekciju
- naredba za realizaciju petlje s ispitivanjem uvjeta na početku
- naredba za realizaciju petlje s unaprijed utvrđenim brojem ponavljanja

```
IF u_let THEN
  naredbe
ELSEIF u_let THEN
  naredbe
ELSEIF u_let THEN
  naredbe ...
ELSE
  naredbe
END IF
```

```
WHILE u_let
  naredbe
  EXIT WHILE
CONTINUE WHILE
END WHILE
```

kao break u jeziku
kao continue u jeziku

```
FOR i = m TO n STEP
  naredbe
  EXIT FOR
CONTINUE FOR
END FOR
```

kao break u jeziku
kao continue u jeziku

Naredbe u pohranjeni procedura

- U pohranjenim procedurama mogu se koristiti (gotovo) sve do sada prikazane SQL naredbe (izuzetak je npr. DROP DATABASE)

```
...
DELETE FROM stu
WHERE preStu LIKE
UPDATE stu SET p_rMestoStan = 10000
WHERE p_rMestoStan = 1000
INSERT INTO m_esto VALUES (31000, 'Osi')
...
```

- Rezultat SELECT naredbe može se pohraniti u varijablu npr.

```
DEFINE imeStu LIKE student.imeStu
DEFINE preStu LIKE student.preStu
...
SELECT imeStu, preStu
INTO imeStu, preStu
FROM student
WHERE m_rStu = 12345
```

- broj i tipovi varijabli moraju odgovarati broju i tipovima izraza iz liste za selekciju
- SELECT naredba smije vratiti samo jednu n-torku

Porazna varijabli u naredbama

- varijable se slobodno mogu koristiti na svim mjestima na kojim se u SQL naredbama koriste izrazi npr.
 - u izrazima u SELECT listi
 - u WHERE dijelu SQL naredbe
 - u VALUES listi INSERT naredbe
 - u izrazima u SET dijelu UPDATE naredbe

```
CREATE PROCEDURE ...
  DEFINE i_nos DECIMAL(3,2)
  DEFINE datum DATE
  DEFINE s INTEGER
  DEFINE n CHAR(20)
  LET coef = (SELECT MAX(coef) FROM nastani)
  LET s = 100 LET n = Primorsko-Oranska
  LET datum = TODAY - 3*5*20
  SELECT AVG(ocjena) AS coef INTO i_nos FROM ispit
  WHERE datIspit = datum
  UPDATE stu SET datRoStu = datum
  WHERE datRoStu = datum
  INSERT INTO upani_a VALUES(s, n)
  ...
```

Proceduralne pohranjene procedure

```
CREATE PROCEDURE imeProcure(imeAr, tip, imeAr, tip) ...
```

- tipovi podataka ulazni argumenta procedure mogu kao i varijable biti definirani eksplicitno

```
CREATE FUNCTION površina(sirina INTEGER, visina INTEGER)
  ...
```

- ili implicitno

```
CREATE PROCEDURE postaviAresu(p_m_rStu LIKE stu.m_rStu,
  p_a_resa LIKE stu.a_resa)
  ...
```

- jednako kao u drugim programskim jezicima argumenti se u tijelu procedure/unkcije mogu koristiti na jednak način kao i varijable

Rezultati funkcije

- tipovi rezultata koje funkcija vraća moraju se deklarirati

```
CREATE FUNCTION imeFunkcije(imeAr, tip, imeAr, tip) ...
  RETURNING INTEGER AS ime CHAR(20) AS ime DATE AS ime
  DEFINE ...
  ...
END FUNCTION
```

- tipovi rezultata mogu se deklarirati jedino eksplicitno (nije moguće koristiti oblik RETURN atribut kao pri definiciji argumenta ili varijabli)
- ime rezultata se navodi opcionalno: korisno je deklarirati ime rezultata jer se npr. pri pozivu funkcije iz interaktivnog alata rezultat prikazuje zajedno s deklariranim imenom

Proceduralne rezultate funkcije u pozivajući proceduru

- koristi se naredba RETURN slična naredbi RETURN u ostalim programskim jezicima RETURN naredba se u tijelu procedure može pojaviti više puta naredbom je u pozivajući program moguće vratiti jednu ili više vrijednosti

```
CREATE FUNCTION opsegPovrsina(radijus DECIMAL(10,5))
  RETURNING DECIMAL(10,5) AS opseg
  DECIMAL(10,5) AS površina
  DEFINE o p DECIMAL(10,5)
  LET o = 2 * radijus * 3.1415
  LET p = radijus * radijus * 3.1415
  RETURN o p
END FUNCTION
```

```
EXECUTE FUNCTION opsegPovrsina(5)
```

opseg	povrsina
28.27	78.5

Načini poziva procedure i funkcije

- Interaktivni alati npr. Aqua Data Studio

```
EXECUTE PROCEDURE prebraci 1001 1002 0.30
```

- procedura ne vraća rezultat (eventualno signalizira pogrešku)

```
EXECUTE FUNCTION opsePo površina 5
```

- funkcija vraća rezultat (eventualno signalizira pogrešku)

opseg	površina
28.27	720

Načini poziva procedure i funkcije

- počranjene procedure ili funkcije

```
CREATE PROCEDURE p ...
DEFINE ro_nam ...
DEFINE opse površina ...
...
-- procedure ne vraća rezultat
EXECUTE PROCEDURE prebraci 1001 1002 0.30
...
-- funkcije vraća rezultat
LET ro_nam = ro_nameniaac123
CALL ro_nameniaac123 RETURNING ro_nam
...
-- funkcije vraća rezultat
CALL opsePo površina 5 RETURNING opse površina
```

Načini poziva funkcije

- orištenje funkcija u SQL naredbama
 - funkcije koje vraćaju točno jednu vrijednost mogu se u SQL naredbama koristiti na svim mjestima na kojima se mogu koristiti ugrađene SQL funkcije

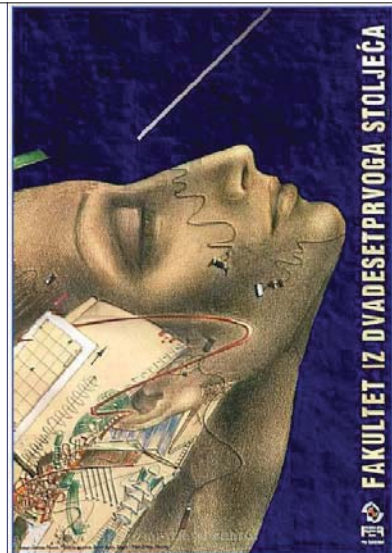
```
SELECT ro_nameniaresa AS ro_nam
FROM osoba
WHERE ro_nameniaresa = 0

DELETE FROM osoba
WHERE ro_namenim = 13
```

Baze podataka

Predavanja
lipanj 2008.

uav za
upravljanje
baza a podataka



Uav za upravljanje baza a podataka

- Database Management System
 - skriva od korisnika detalje fizičke pohrane podataka
 - omogućuje definiciju i rukovanje s podacima
 - obavlja optimiranje upita
 - obavlja funkciju zaštite podataka
 - integritet podataka
 - pristup podacima - autorizacija, sigurnost
 - osigurava potporu za upravljanje transakcijama
 - obnova u slučaju pogreške ili uništenja baze podataka
 - kontrola paralelnog pristupa

Uav za upravljanje baza a podataka

- jedinica rada nad bazom podataka
- sastoji se od niza logički povezanih izmjena
- početak transakcije - **START TRANSACTION**
- završetak transakcije:
 - COMMIT** - uspješan završetak - potvrđivanje transakcije
 - ROLLBACK** - neuspješan završetak - poništavanje transakcije - poništavanje svih izmjena koje je transakcija obavila

Primer transakcije

```
CREATE PROCEDURE prienos sracuna INTEGER
    nracun INTEGER
    i nos DECIMAL (10,2)
DEFINE pom salo DECIMAL (10,2)
BEGIN WORK
    UPDATE racun SET salo = salo + i nos
    WHERE rracun = sracuna
    UPDATE racun SET salo = salo + i nos
    WHERE rracun = nracun
    SELECT salo INTO pom salo FROM racun
    WHERE rracun = sracuna
    IF pom salo = 0 THEN
        ROLLBACK WORK
    ELSE
        COMMIT WORK
    END IF
END PROCEDURE
```

Implikacije granice transakcija

- Ukoliko granice transakcije nisu eksplicitno definirane naredbama BEGIN/COMMIT/ROLLBACK, tada se granice transakcije određuju implicitno:

- početkom transakcije smatra se početak programa
 - uspješan završetak programa - potvrda transakcije
 - neuspješan završetak programa - poništavanje transakcije

ili

- svaka SQL naredba se smatra transakcijom za sebe
 - naročito važno: UPDATE, DELETE, INSERT u slučajevima kada djeluju nad skupom n-torki

Svojstva transakcije

ACID

- Atomicity** - nedjeljivost transakcije (atomarnost) - transakcija se mora obaviti u cijelosti ili se uopće ne smije obaviti
- Consistency** - konzistentnost - transakcijom baza podataka prelazi iz jednog konzistentnog stanja u drugo konzistentno stanje
- Isolation** - izolacija - kada se paralelno obavljaju dvije ili više transakcija, njihov učinak mora biti jednak kao da su se obavljale jedna iza druge
- Durability** - izdržljivost - ukoliko je transakcija obavila svoj posao, njezini efekti ne smiju biti izgubljeni ako se dogodi kvar sustava, čak i u situaciji kada se kvar desi neposredno nakon završetka transakcije

Nedjeljivost transakcije

```
CREATE PROCEDURE prienos sracuna INTEGER nracun INTEGER
    i nos DECIMAL (10,2)
DEFINE pom salo DECIMAL (10,2)
BEGIN WORK
    UPDATE racun SET salo = salo + i nos
    WHERE rracun = sracuna
    UPDATE racun SET salo = salo + i nos
    WHERE rracun = nracun
    SELECT salo INTO pom salo FROM racun
    WHERE rracun = sracuna
    ...
    Kvar se dogodio za vrijeme obavljanja druge UPDATE naredbe
    • sustav mora osigurati poništavanje efekata prve UPDATE naredbe!
```

- Sa stanovišta krajnjeg korisnika transakcija je **nedjeljiva**
 - nije bitno što se moraju obaviti dvije ili više zasebnih operacija nad bazom podataka
- Korisnik mora biti siguran da je zadatak **obavljen po-puno i ta-o jedno** (ili ništa nije obavljeno)

Zdržljivo transakcije

```
...
BEGIN WORK
    UPDATE racun SET salo = salo + i nos
    WHERE rracun = sracuna
    UPDATE racun SET salo = salo + i nos
    WHERE rracun = nracun
    SELECT salo INTO pom salo FROM racun
    WHERE rracun = sracuna
    IF pom salo = 0 THEN
        ROLLBACK WORK
    ELSE
        COMMIT WORK
    END IF
```

Kvar se dogodio nakon potvrđivanja transakcije

- efekti transakcije ne smiju biti izgubljeni

- Bez obzira u kojem se trenutku nakon potvrđivanja transakcije dogodio kvar, sustav mora osigurati da su njezini efekti trajno pohranjeni

Database Recovery

- dovesti bazu podataka u najnovije stanje za koje se pouzdano zna da je bilo ispravno

- Velike baze podataka – dijeljene, višekorisničke – nužno moraju posjedovati mehanizme obnove
- Male, jednokorisničke baze podataka obično imaju malu ili uopće nemaju potporu obnovi – obnova se prepušta korisnikovoj odgovornosti – podrazumijeva se da korisnik periodički stvara "backup" kopiju pomoću koje u slučaju potrebe obnavlja bazu podataka

Izvorci pogrešaka

- pogreške opreme
- pogreške operacijskog sustava
- pogreške sustava za upravljanje bazama podataka
- pogreške operatera
- kolebanje izvora energije
- požar, sabotaza, ...

Procenjeno pravilo koje omogućuje obnovu

- **Redundancija** - svaki se podatak mora moći rekonstruirati iz nekih drugih informacija redundantno pohranjenih negdje drugdje u sustavu (na traci, na drugom disku, na zrcalnom disku, ...)

Procenjeno opipovanje koji omogućuje obnovu

- 1 Periodičko kopiranje sadržaja baze podataka na arhivski medij (traka)
(1 × dnevno, 1 × tjedno - ovisno o učestalosti promjena)
- 2 Svaka izmjena u bazi podataka evidentira se u **logičkom dnevniku izmjena (logical log, journal)**
 - stara vrijednost zapisa, nova vrijednost zapisa
 - korisnik, vrijeme, ...
 - izmjena se **prvo zapisuje u dnevnik, a tek se onda provodi!**
- **dnevnici izmjena omogućuju**
 - poništavanje transakcija (važno radi svojstva nedjeljivosti)
 - ponovno obavljanje transakcija (važno radi svojstva izdržljivosti)

Dnevnik izmjena

Transakcija A **beginA A1 A2 commitA**

Transakcija B **beginB B1 commitB**

Transakcija C **beginC C1 C2 commitC**



Poništavanje transakcije pomoću dnevnika izmjena

Transakcija A **beginA A1 A2 commitA**

Transakcija B **beginB B1 commitB**

Transakcija C **beginC C1 C2 rollbackC**



Tipovi pogrešaka

- 1 Pogreške transakcija (*transaction failure*) - pogreške koje otkriva sama aplikacija ili pogreške koje su posljedica neplaniranog prekida transakcije
- 2 Pogreška računalskog sustava (*system failure*) - baza podataka nije fizički uništena
- 3 Kvar medija za pohranu (*media failure*) - baza podataka je fizički uništena

Slučaj 1 - pomoću dnevnika izmjena poništavaju se efekti transakcije, kao da transakcija nikada nije započela s radom
Slučaj 2 - transakcije koje su se obavljale u trenutku prekida se nakon ponovnog pokretanja poništavaju
Slučaj 3 - baza podataka se obnavlja pomoću arhivske kopije i pripadnog dnevnika izmjena

Što reke ranakcija koje otkriva aplikacija

- Slučajevi u kojima aplikacija predviđa obavljanje naredbe

□ □ □ □ □ □ □ □ □ □ □ □ □ □

```
...
IF pom[sal] = 0 THEN
    ROLLBACK WORK
ELSE
    COMMIT WORK
END IF
...
```

☐ o ☐ re ☐ ke ☐ ran ☐ akcija koje ne o ☐ kriva aplikacija

- ako se dogodi pogreška za koju program nema pretpostavljenu reakciju, program završava na neplanirani način, SUBP automatski obavlja **ROLLBACK WORK**
- Primjer: pokušaj unosa zapisa čiji ključ već postoji u bazi:

```
CREATE TABLE oso_a (
    m_r          INTEGER
    pre_time     CHAR(20)
    PRIMARY KEY (m_r))
```

```
početa pro Rama
BEGIN WORK
INSERT INTO oso a VALUES (1'D'etlić Pero
INSERT INTO oso a VALUES (2'Marić Maro
INSERT INTO oso a VALUES (1'Katić Kata
INSERT INTO oso a VALUES (Matić Mato
COMMIT WORK
a r eta pro Rama
```

Pogreška!

Core računalno uva

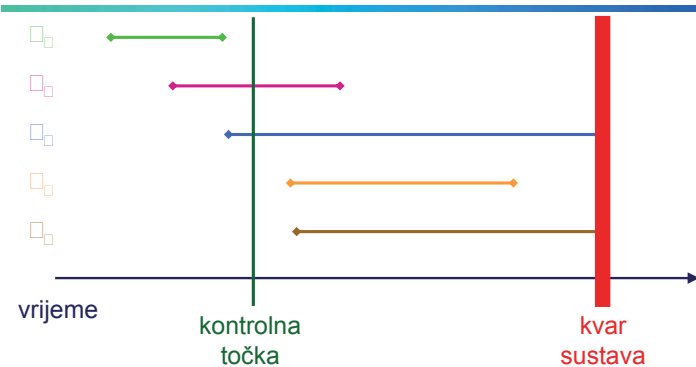
- **Čaza nije uničena**

- **Uve ranakcije koje u e odvijale u renu ku kvara oraju ii poni ene** jer nisu kompletne!
- pretraživanjem dnevnika od početka identificiraju se transakcije za koje postoji **ii** i ne postoji **iii**
 ii
 ■ takav postupak bi predugo trajao
 ■ u određenim intervalima (obično svakih 5 minuta) određuje se **konrolna očka** (checkpoint)

api kontrolne točke adri

- listu svih aktivnih transakcija
- za svaku transakciju - adresu najnovijeg zapisa u datoteci dnevnika










ri jer



Transakcije ☐ i ☐ treba poništiti

Transakcije ☐ i ☐ treba ponovo obaviti

roce o nove

- Stvara se:
 - lista za poništavanje - na početku sadrži sve transakcije koje su bile aktivne u kontrolnoj točki
 - lista za ponovo obavljanje - na početku je prazna
- Pretražuje se dnevnik od kontrolne točke
 - transakcija za koju se pronađe     dodaje se u listu za poništavanje
 - transakcija za koju se pronađe      prebacuje se iz liste za poništavanje u listu za ponovo obavljanje
- Ponovo se obavljaju transakcije iz liste za ponovo obavljanje
- Poništavaju se transakcije iz liste za poništavanje

□□□□ ne □ o□e prihva□i□ ni □ jedan zah□jev dok □e ne zavr□i
proce□ o□nove□

☐ var ☐ edija za pohranu

- **čaza je fizički uništena i npr: zlo i kvara diča**
- obnova sadržaja baze pomoću najnovije arhivske kopije
- pomoću najnovijeg dnevnika obavljaju se transakcije koje su bile provedene od trenutka arhiviranja
 - ako je najnovija arhivska kopija “pokvarena”
 - uzima se predzadnja arhivska kopija
 - dnevnik izmjena od predzadnje arhive do zadnje arhive
 - dnevnik izmjena nastalih nakon zadnje arhive

PREPORUKE:

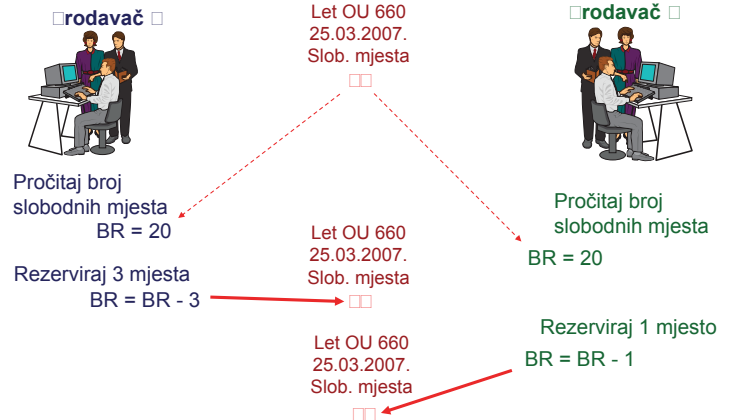
- čuvati najmanje tri posljednje arhive i pripadne dnevnike
- dnevnik se ne nalazi na istom disku na kojem je baza podataka

□□□□□□□□ □□□□□□□□ □□□□□□□□

- u višekorisničkom radu **više programa** može **istovremeno pristupiti istim podacima**
- Rezultat transakcije ne smije ovisiti o tome da li se istodobno odvijaju i neke druge transakcije!!!
- SUBP mora spriječiti sljedeće:
 - dva (ili više) programa "istovremeno" mijenjaju isti podatak – problem: **izgubljene izmjene (lost update)** - posljednji pobjeđuje
 - neki programi pregledavaju podatak dok ga neki drugi program mijenja – problemi: **prljavo čitanje (dirty read)**, **neponovljivo čitanje (nonrepeatable read)**, **sablasi (phantoms)**

Primer zbuđena izmena **Lost update**

Rezervacija zrakoplovnih karata



Prljavo čitanje **Dirty read**

osoba	mbr	prez	ime
	1111	Novak	Ivan
	2222	Kolar	Iva

Transakcija

```
INSERT INTO osoba  
VALUES (3333,  
urić,  
Ana,  
.  
.  
.  
.  
.  
ROLLBACK WORK
```

Transakcija

```
SELECT * FROM osoba
```

1111	Novak	Ivan
2222	Kolar	Iva
3333	urić	Ana

n-torka koja
nikad nije
stvarno postojala
u bazi podataka

Neponovljivo čitanje i nepravilne n-torke

- Ista transakcija obavljanjem istog upita mora dobiti uvijek isti rezultat (osim ako sama nije promijenila podatke čije čitanje ponavlja)

Transakcija

```
SELECT salo FROM racun  
WHERE rRacun = 2  
  
-- A ne mijenja salo  
-- racun s roem 2
```

Rezultat: 400.00

```
SELECT salo FROM racun  
WHERE rRacun = 2
```

Rezultat **mora** (opet) biti:
400.00

Primer neponovljivo čitanje **Nonrepeatable read**

Transakcija

```
SELECT salo FROM racun  
WHERE rRacun = 2  
  
-- A ne mijenja salo  
-- racun s roem 2  
  
SELECT salo FROM racun  
WHERE rRacun = 2
```

Rezultat: 400.00

Transakcija

```
UPDATE racun  
SET salo = salo * 1.1  
WHERE rRacun = 2
```

Rezultat: 440.00

- Ista transakcija obavljanjem istog upita dobije drugačiji rezultat

Primer nepravilne n-torke **Phantom rows**

Transakcija

```
SELECT COUNT(*)  
FROM racun  
WHERE salo > 100  
  
-- A ne mijenja racun  
  
SELECT COUNT(*)  
FROM racun  
WHERE salo > 100
```

Rezultat: 2

Transakcija

```
INSERT INTO racun  
VALUES (3, 100.00)
```

Rezultat: 3 !!!

- Ista transakcija obavljanjem istog upita dobije drugačiji rezultat - zbog toga što je u međuvremenu transakcijom B unesena n-torka koja zadovoljava kriterij upita

zaključavanje **Locking**

- transakcija može zaključati podatak (podatke)
 - sprečava druge transakcije da pristupe podatku dok ga ona ne otključa
- podaci koji su se mijenjali tijekom transakcije **ostaju zaključani do kraja transakcije**
- dio SUBP (**locking manager**) zaključava zapise i prosuđuje u slučajevima kad postoji više zahtjeva za zaključavanjem istog podatka

zaključavanje

rezervacija zrakoplovnih karata

prodavač



zaključaj norku

Pročitaj broj slob.mjesta

BR = 20

Rezerviraj 3 mjesta

BR = BR - 3

zaključaj norku

Let OU 660
25.03.2007.
Slob. mjesta



Let OU 660
25.03.2007.
Slob. mjesta



Let OU 660
25.03.2007.
Slob. mjesta



prodavač



zaključaj norku čekaj

čekaj →
Pročitaj broj slob.mjesta
BR = 17
Rezerviraj 1 mjesto
BR = BR - 1
zaključaj norku

potpuni zastoj **Deadlock**

transakcija

zaključaj

zaključaj

transakcija

zaključaj

zaključaj

.....

- izbjegavanje potpunog zastoja:
 - transakcija zatraži sva zaključavanja odjednom (npr. na početku) - **zaključa sve ili ništa**
 - zahtijeva se da transakcije zaključavaju podatke u nekom **određeno poretku**
- u slučaju da se dogodi potpuni zastoj:
 - barem jedna transakcija se mora prekinuti - poništavaju se njezini efekti

vrste zaključavanja

- ključ za pisanje/izmjenu
 -
- ključ za čitanje
 -D.....D.....

granulacija zaključavanja

- Veličina objekta koji se zaključava
 - baza podataka
 - tablica/relacija
 - memorijska stranica
 - n-torka
- Zaključavanje većih objekata
 - manji broj ključeva ⇒ manji utrošak proc. vremena i memorije
 - manja dostupnost podataka (često se zaključa više nego što je potrebno)
- Zaključavanje manjih objekata
 - veći broj ključeva ⇒ veći utrošak proc. vremena i memorije
 - veća dostupnost podataka (zaključavaju se samo objekti koje je zaista potrebno zaključati)

zaključak

- Zaštita integriteta i sigurnost baze podataka temelji se na pravilima pohranjenim u rječniku podataka
- Pravila pohranjena u rječniku podataka
 - nezaobilazna su za sve korisnike
 - ne opterećuju aplikacijske programe
- Obnova baze podataka bez gubitka informacija moguća je jedino ako se:
 - redovito izrađuju arhivske kopije
 - vodi briga o dnevnicima izmjena