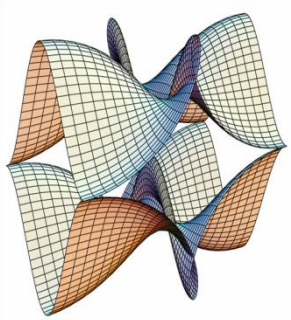
Sveučilište u Zagrebu
PMF – Matematički odsjek

BAZE PODATAKA
Predavanja 2019/2020



Poglavlje 1:
Uvod u baze podataka

Sastavio: Robert Manger
03.03.2020

O čemu je riječ u ovom predmetu

- Bavimo se:
  - trajnim pohranjivanjem
  - većih količina podataka
  - u vanjskoj memoriji računala.
- Trajno pohranjivanje podataka u računalima:
  - Postoji jednako dugo koliko i sama računala,
  - podržana je u programskim jezicima (COBOL, C).
- Kad govorimo o bazama podataka:
  - mislimo na višu razinu rada s podacima od one koju podržavaju klasični programski jezici.

Sadržaj Poglavlja 1

Glavne ideje tehnologije baze podat

- Pojedina aplikacija ne stvara vlastite datoteke.
  - Sve aplikacije rabe *zajedničku i objedinjenu kolekciju podataka.*
- Aplikacija ne pristupa izravno podacima.
  - Ona se koristi uslugama *specijaliziranog softvera* koji je zadužen da se brine za zajedničku kolekciju.

- Zajednička kolekcija podataka naziva se:

  baza podataka.
- Specijalizirani softver koji posreduje između aplikacija i podataka naziva se:

  sustav za upravljanje bazom podataka (DBMS).

Baza podataka i DBMS (1)

- **Baza podataka** je skup međusobno povezanih podataka, pohranjenih u vanjskoj memoriji računala.
  - Podaci su istovremeno dostupni raznim korisnicima i aplikacijskim programima.
  - Ubacivanje, promjena, brisanje i čitanje podataka obavlja se posredstvom posebnog softvera, takozvanog *sustava za upravljanje bazom podataka* (DBMS-a).
  - Korisnici i aplikacije pritom ne moraju poznavati detalje fizičkog prikaza podataka, već se referenciraju na neku idealiziranu logičku strukturu baze.

Baza podataka i DBMS (2)

- **Sustav za upravljanje bazom podataka** (*Data Base Management System* - DBMS) je poslužitelj (server) baze podataka.
  - Oblikuje fizički prikaz baze u skladu s traženom logičkom strukturom.
  - Obavlja u ime klijenata sve operacije s podacima.
  - U stanju je podržati razne baze, od kojih svaka može imati svoju logičku strukturu, no u skladu s istim *modelom*.
  - Brine se za sigurnost podataka, automatizira administrativne poslove s bazom.

Baza podataka i DBMS (3)

- Danas postoji svega nekoliko važnih i široko zastupljenih DBMS-a.
  - **DB2.** Proizvod tvrtke IBM, namijenjen prvenstveno velikim *mainframe* računalima.
  - **Oracle.** Proizvod istoimene tvrtke, pokriva gotovo sve računalne platforme: UNIX, Linux i MS Windows.
  - **MS SQL Server.** Microsoftov proizvod, namijenjen poslužiteljskim računalima s MS Windows.
  - **MySQL.** Besplatni proizvod tvrtke MySQL AB, popularan na raznim platformama, prvenstveno kao podrška web aplikacijama.

Modeli za logičku strukturu baze (1)

- **Model podataka** je skup pravila koja određuju kako sve može izgledati logička struktura baze podataka.
  - Model čini osnovu za oblikovanje i implementiranje baze.
  - Svaki DBMS podržava samo neki od modela.
  - Podaci u bazi moraju biti logički organizirani u skladu s onim modelom koji podržava odabrani DBMS.

Modeli za logičku strukturu baze (2)

- Dosadašnji DBMS-i podržavali su neki od sljedećih modela:
  - **Relacijski model.** Zasnovan je matematičkom pojmu *relacije*. Baza se sastoji od tablica.
  - **Mrežni model.** Baza je predočena mrežom koja se sastoji od *čvorova* i usmjerenih *lukova*.
  - **Hijerarhijski model.** Specijalni slučaj mrežnog. Baza je predočena *stablom* (hijerarhijom) ili skupom stabala.
  - **Objektni model.** Inspiriran OO programskim jezicima. Baza se sastoji od trajno pohranjenih *objekata*.

Ciljevi uporabe baze podataka (1)

- Baze podataka predstavljaju višu razinu rada s podacima u odnosu na programske jezike.

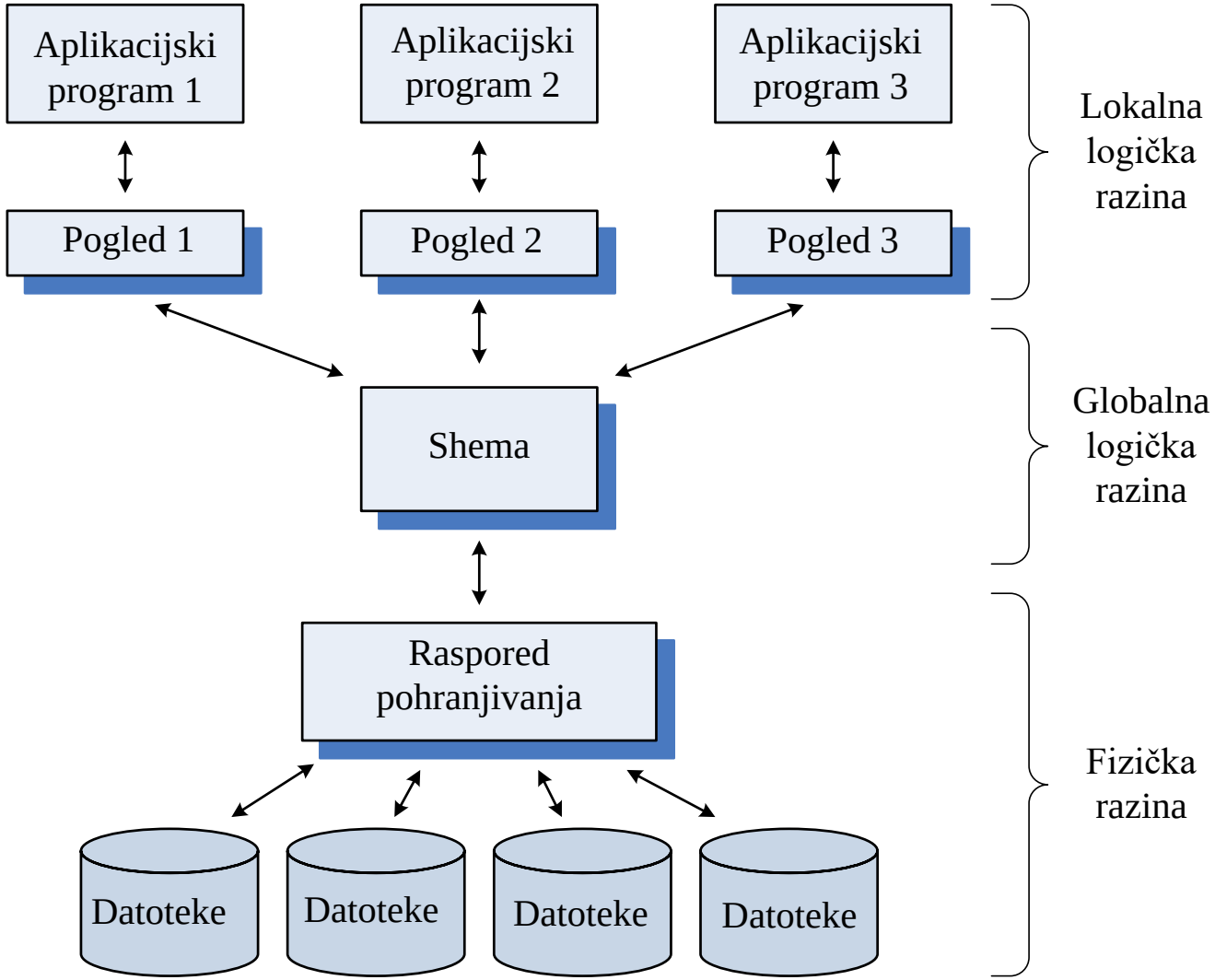
- Ta viša razina rada očituje se u tome što tehnologija baza podataka nastoji (i u velikoj mjeri uspijeva) ispuniti sljedeće ciljeve.
  - Fizička nezavisnost podataka
  - Logička nezavisnost podataka
  - Fleksibilnost pristupa podacima

Ciljevi uporabe baze podataka (2)

- Istovremeni pristup podacima
- Čuvanje integriteta
- Mogućnost oporavka nakon kvara
- Zaštita od neovlaštene uporabe
- Zadovoljavajuća brzina pristupa
- Mogućnost podešavanja i kontrole

- Sve ove ciljeve zaista u potpunosti uspijevaju ispuniti jedino relacijske baze.

Arhitektura baze podataka (1)



Arhitektura baze podataka (2)

- Arhitektura se sastoji od tri „sloja“ (razine apstrakcije) i sučelja među slojevima.
  - **Fizička razina:** fizički prikaz podataka na jedinicama vanjske memorije.
    - *Raspored pohranjivanja* opisuje kako se elementi logičke definicije baze preslikavaju na fizičke uređaje.
  - **Globalna logička razina:** logička struktura cijele baze.
    - Opis globalne logičke definicije naziva se *shema* (engleski također *schema*).
  - **Lokalna logička razina:** logička predodžba o dijelu baze koji rabi pojedina aplikacija.
    - Opis jedne lokalne logičke definicije zove se *pogled* (*view*) ili *pod-shema*.

Arhitektura baze podataka (3)

- *Fizička neovisnost* podataka postiže se time što se pravi razlika između fizičke i globalne logičke razine.
- *Logička neovisnost* postiže razlikovanjem lokalne logičke razine i globalne logičke razine.
- Za stvaranje baze potrebno je zadati samo *shemu* i *poglede*.
  - DBMS tada automatski generira potrebni raspored pohranjivanja i fizičku bazu.
  - Projektant odnosno administrator može *samo donekle* utjecati na fizičku građu baze.

Jezici za rad s bazama podataka (1)

- Služe za komunikaciju korisnika odnosno aplikacijskog programa i DBMS-a.
- Tradicionalna podjela:
  - **Jezik za opis podataka** (*Data Description Language* - DDL). Služi projektantu baze ili administratoru za zapisivanje sheme ili pogleda.
  - **Jezik za manipuliranje podacima** (*Data Manipulation Language* - DML). Služi programeru za uspostavljanje veze između aplikacijskog programa i baze.
  - **Jezik za postavljanje upita** (*Query Language* - QL). Služi korisniku za interaktivno pretraživanje baze.

Jezici za rad s bazama podataka (2)

- Kod relacijskih baza postoji težnja objedinjavanja svih triju vrsta jezika u jedan sveobuhvatni.
- Primjer takvog *integriranog* jezika je SQL.
- DDL, DML i QL nisu programski jezici, dakle oni nisu dovoljni za razvoj aplikacija.
- Alati za razvoj aplikacija:
  - Klasični programski jezici (COBOL, C) s ugniježđenim DML naredbama.
  - Jezici 4. generacije (*4-th Generation Languages* - 4GL).
  - Objektno orijentirani programski jezici (Java, C++, C#) s unaprijed pripremljenim klasama za rad s bazom.

Sadržaj Poglavlja 1

Općenito o razvojnom ciklusu

- Uvođenje baze podataka složeni je zadatak:
  - Zahtijeva primjenu pogodnih metoda i alata,
  - Zahtijeva timski rad stručnjaka raznih profila.

- Zadatak se dijeli u 5 aktivnosti:
  - Utvrđivanje i analiza zahtjeva
  - Oblikovanje (projektiranje)
  - Implementacija
  - Testiranje
  - Održavanje

Utvrđivanje i analiza zahtjeva

- Proučavaju se tokovi informacija: dokumenti, radni procesi, postojeći softver.
- Razgovara se s korisnicima.
- Uočavaju se *podaci* koje treba pohranjivati i *veze* među njima.
- Prepoznaju se *sinonimi* i *homonimi* u nazivima podataka, usklađuje se terminologija.
- Analiziraju se *transakcije* (postupci, operacije) koje će se obavljati s podacima.
- Rezultat je dokument (obično pisan neformalno u prirodnom jeziku) koji se zove *specifikacija.*

Oblikovanje (1)

- Cilj oblikovanja je da se u skladu sa specifikacijom oblikuje građa baze.
- Analiza zahtjeva otprilike je odredila koje vrste podataka baza treba sadržavati i što se s njima treba moći raditi.
- Oblikovanje predlaže kako da se podaci na pogodan način grupiraju, strukturiraju i povežu.
- Glavni rezultat oblikovanja trebala bi biti *shema* cijele baze, građena u skladu s pravilima rabljenog modela podataka, te zapisana na način da je rabljeni DBMS može razumjeti i realizirati.

Oblikovanje (2)

- Kod većih baza, rezultat oblikovanja mogu biti i *pogledi* (pod-sheme) za potrebe važnijih aplikacija.
- Oblikovanje je složena aktivnost, dijeli se u 3 faze:
  - **Konceptualno oblikovanje.**
    - Glavni rezultat prve faze je *konceptualna shema* cijele baze, sastavljena od entiteta, atributa i veza.
    - Ona zorno opisuje sadržaj baze i načine povezivanja podataka u njoj.
    - Prikaz je jezgrovit, neformalan i pogodan ljudima za razumijevanje.
    - Prikaz je nedovoljno razrađen da bi omogućio izravnu implementaciju.

Oblikovanje (3)

- **Logičko oblikovanje.**
  - Kao glavni rezultat druge faze oblikovanja nastaje *logička shema*, koja je u slučaju relacijskog modela sastavljena od relacija (tablica).
  - Sastavni dio logičkog oblikovanja je i takozvana *normalizacija*, gdje se primjenom posebnih pravila nastoji popraviti logička struktura samih relacija, tako da se ona bolje prilagodi inherentnim osobinama samih podataka.
- **Fizičko oblikovanje.**
  - Glavni rezultat treće faze oblikovanja je *fizička shema* cijele baze, dakle opis njezine fizičke građe.

Oblikovanje (4)

- U slučaju uporabe DBMS-a zasnovanog na jeziku SQL, fizička shema zapravo je niz SQL naredbi kojima se relacije iz logičke sheme realiziraju kao SQL tablice.
- Pritom se dodaju pomoćne strukture i mehanizmi za postizavanje traženih performansi, te čuvanje integriteta i sigurnosti podataka.
- Također se mogu uključiti i SQL naredbe kojima se pogledi (pod-sheme) za pojedine aplikacije realiziraju kao virtualne tablice izvedene iz stvarnih tablica.

- Rezultati oblikovanja opisuju se u dokumentima koji čine *projektnu dokumentaciju* baze.

Implementacija

- Fizička realizacija oblikovane baze na poslužiteljskom računalu.
- Pokreću se SQL naredbe koje čine fizičku shemu baze, te se na disku stvaraju prazne SQL tablice sa svim pratećim strukturama i mehanizmima.
- Dalje slijedi punjenje praznih tablica s početnim podacima.
- Većina DBMS-a ima alate koji nam olakšavaju transfer podataka iz običnih datoteka u bazu.
- Razvijaju se aplikacije koje obavljaju najvažnije transakcije s podacima.

Testiranje

- Glavni cilj testiranja je otkrivanje i popravak grešaka koje su se mogle potkrasti u svakoj od prethodnih aktivnosti.
- Greške u ranijim aktivnostima imaju teže posljedice jer se provlače kroz kasnije aktivnosti.
- Mjerenjem performansi tijekom testiranja provjerava se jesu li zadovoljeni zahtjevi vezani uz performanse.
- Ako performanse nisu dovoljno dobre, administrator baze može to pokušati kasnije ispraviti podešavanjem parametara fizičke organizacije.

Održavanje

- Odvija se u vrijeme kad je baza ušla u uporabu.
- Kontinuirani proces, gdje su baza i njezini dokumenti podvrgnuti stalnim promjenama.
- Vrste održavanja: *korekcijsko*, *perfekcijsko*, *adaptacijsko, preventivno*.
- Održavanje baze je nužnost na koju se treba pripremiti već tijekom njezinog razvoja.
- Da bi promjene tekle što bezbolnije, važno je da baza od početka ima zdravu građu.
- Kod relacijskih baza, ta zdrava građa znači normaliziranost.

Sadržaj Poglavlja 1

Općenito o dokumentaciji

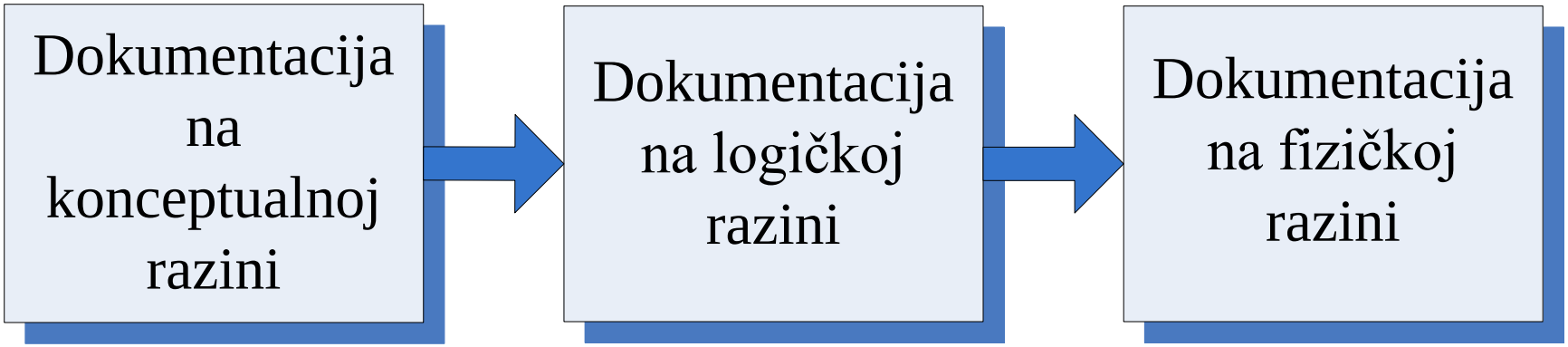
- Razlikujemo:
  - *korisničku* dokumentaciju namijenjenu korisnicima,
  - *razvojnu* dokumentaciju namijenjenu inženjerima.
- Razvojnu dokumentaciju dalje dijelimo na:
  - dokumente koji prate pojedine aktivnosti iz razvojnog ciklusa.
- Mi ćemo se ograničiti na *projektnu* dokumentaciju,
  - dakle na dokumente koji nastaju tijekom oblikovanja.
- Spomenut ćemo:
  - važnost takve dokumentacije,
  - predloške za njezinu izradu,
  - alate koji se rabe za njenu izradu.

Važnost izrade dokumentacije (1)

- Projektna dokumentacija važna je tijekom polaznog razvoja zato jer ona omogućuje:
  - ispravan tijek oblikovanja i implementacije u skladu s pravilima struke,
  - konzistentan prelazak iz pojedine razvojne faze ili aktivnosti u drugu.
- Dokumentacija je neophodna tijekom kasnijeg testiranja i održavanja budući da ona:
  - predstavlja jedini pouzdani izvor informacija o građi baze.
  - čini sponu između ljudi koji su razvijali bazu i ljudi koje je održavaju.

Važnost izrade dokumentacije (2)

- Projektna dokumentacija prati tri faze oblikovanja, i zato se dijeli na tri dijela.
  - **Projektna dokumentacija na konceptualnoj razini.** Opisuje konceptualnu shemu baze.
  - **Projektna dokumentacija na logičkoj razini.** Dokumentira logičku shemu baze.
  - **Projektna dokumentacija na fizičkoj razini.** Sadrži fizičku shemu baze.



Važnost izrade dokumentacije (3)

- Dokumentacija na fizičkoj razini jedina se izravno rabi za implementaciju baze.
  - No sva tri dijela moraju se čuvati zato jer svaki od njih daje korisnu i komplementarnu informaciju o građi baze.
- Nakon što baza uđe u uporabu, projektna dokumentacija mora se čuvati zbog održavanja.
  - Kad god dođe do promjene u građi baze, promjena se mora unijeti i u dokumentaciju.
- Kod manjih baza dovoljno je da projektna dokumentacija odražava ažurno stanje.
  - No kod složenijih baza treba dokumentirati i povijest promjena.

Predlošci za izradu dokumentacije (1)

- Dokumentacija na konceptualnoj razini:
  - Uglavnom je grafička, oslanja se na dijagrame.
  - Opisuje konceptualnu shemu: entitete, atribute i veze.
- Tri najvažnija predloška:
  - **Izvorni Chen-ov dijagram.**
    - Kao grafički elementi pojavljuju se pravokutnici (entiteti), rombovi (veze), „mjehurići“ (atributi) i spojnice među njima.
    - U dijagram su ubačena imena entiteta, veza i atributa, te oznake takozvanih kardinalnosti veza.
  - **Reducirani Chen-ov dijagram.**
    - Nacrtani su samo pravokutnici (entiteti), rombovi (veze) i spojnice među njima.
    - Prisutna su imena entiteta i veza, te oznake kardinalnosti veza.
    - Nedostatak informacije o atributima na dijagramu nadomještava se tekstom uz dijagram.

Predlošci za izradu dokumentacije (2)

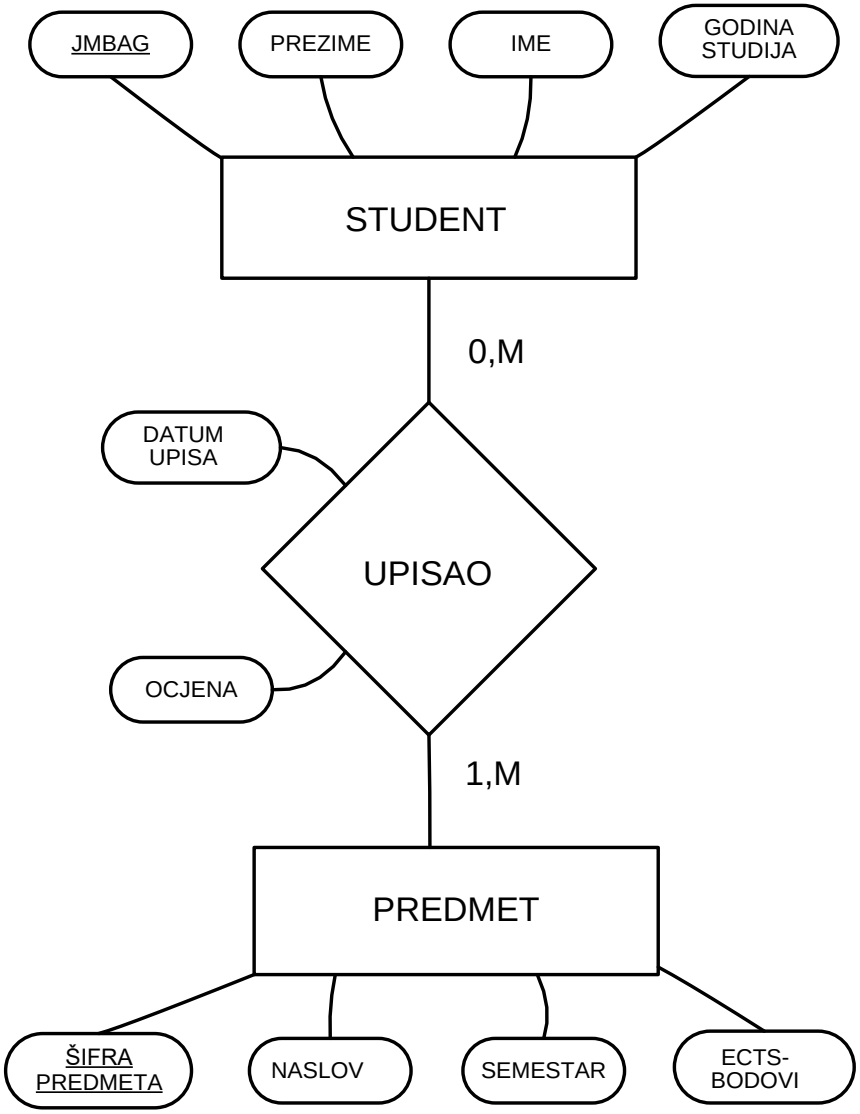
– **UML-ov *class* dijagram.**

- UML je standardizirani grafički jezik koji se rabi u objektno-orijentiranim metodama za razvoj softvera.
- *Class* dijagram je standardni UML dijagram i on originalno služi za prikaz klasa objekata i veza između tih klasa.
- *Class* dijagram možemo upotrijebiti za prikaz konceptualne sheme baze tako da entitet interpretiramo kao klasu koja ima atribute ali nema operacije.
- Entitet se tada crta kao pravokutnik s imenom entiteta na vrhu i imenima atributa u sredini.
- Veza se crta se kao spojnica između pravokutnika s upisanim imenom na sredini i upisanim oznakama kardinalnosti na krajevima. Dijagram sadrži svu potrebnu informaciju.

Predlošci za izradu dokumentacije (3)

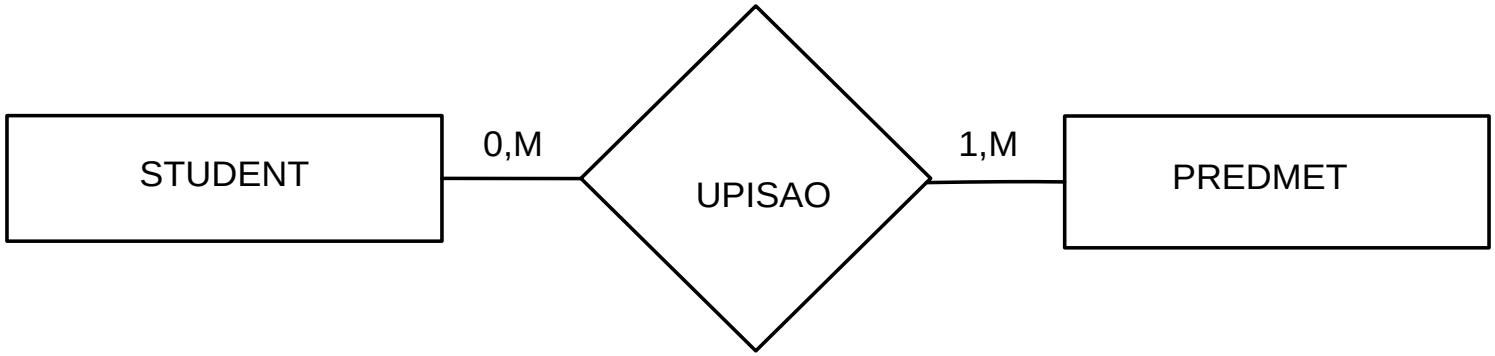
- Prikaz konceptualne sheme jednostavne baze o studentima i predmetima pomoću izvornog Chen-ovog dijagrama.

  Pamti se koji student je upisao koji predmet.



Predlošci za izradu dokumentacije (4)

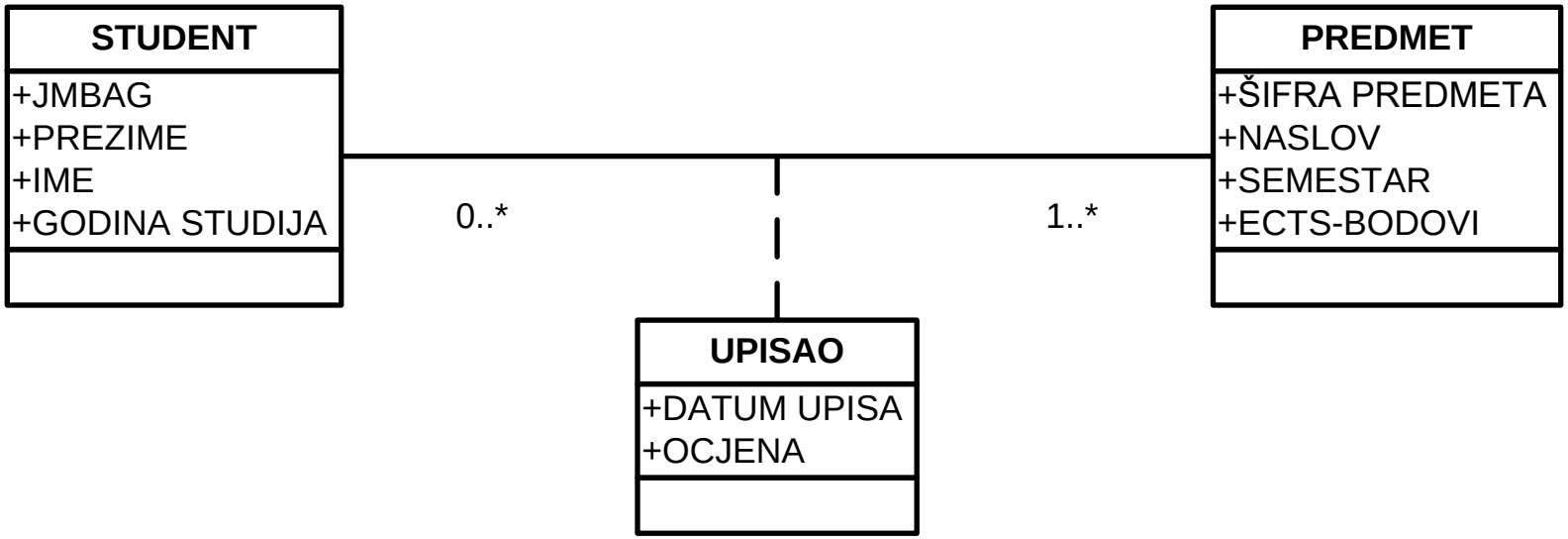
- Konceptualna shema iste baze prikazana pomoću reduciranog Chen-ovog dijagrama s popratnim tekstom.



Tip entiteta STUDENT ima atribute:
JMBAG, PREZIME, IME, GODINA STUDIJA
Tip entiteta PREDMET ima atribute:
ŠIFRA PREDMETA, NASLOV, SEMESTAR, ECTS-BODOVI.
Veza UPISAO ima atribute:
DATUM UPISA, OCJENA.

Predlošci za izradu dokumentacije (4)

- Konceptualna shema iste baze prikazana pomoću UML-ovog *class* dijagrama.



Predlošci za izradu dokumentacije (5)

- Dokumentacija na logičkoj razini opisuje logičku shemu baze.
  - U slučaju relacijske baze to je relacijska shema: skup relacija (tablica) građenih od atributa (stupaca).
- Dva predloška:
  - **Tekstualni prikaz relacijske sheme.**
    - Građa jedne relacije prikazuje se jednim retkom teksta: ime relacije, okrugle zagrade unutar kojih su nanizana imena atributa odvojena zarezima.
    - Građa cijele baze prikazana je nizom ovakvih redaka: koliko relacija toliko redaka.
    - Poželjno je unutar redaka rabiti bogatu tipografiju, na primjer za primarni ključ relacije.
    - Poželjno priložiti rječnik podataka: popis svih atributa s neformalnim opisom njihova značenja i tipa vrijednosti.

Predlošci za izradu dokumentacije (6)

- **Grafički prikaz relacijske sheme.**
  - Građa baze opisuje se dijagramom: pravokutnici i strelice.
  - Jedan pravokutnik prikazuje jednu relaciju, te sadrži ime relacije i imena atributa jedno ispod drugog.
  - Ključ je označen odgovarajućom ikonom ili kraticom.
  - Strelice označavaju referencijalni integritet: one spajaju atribut u jednoj relaciji koji je ujedno ključ u drugoj relaciji.
  - Opet je poželjan rječnik podataka.

- Tekstualni prikaz relacijske sheme baze o studentima i predmetima:

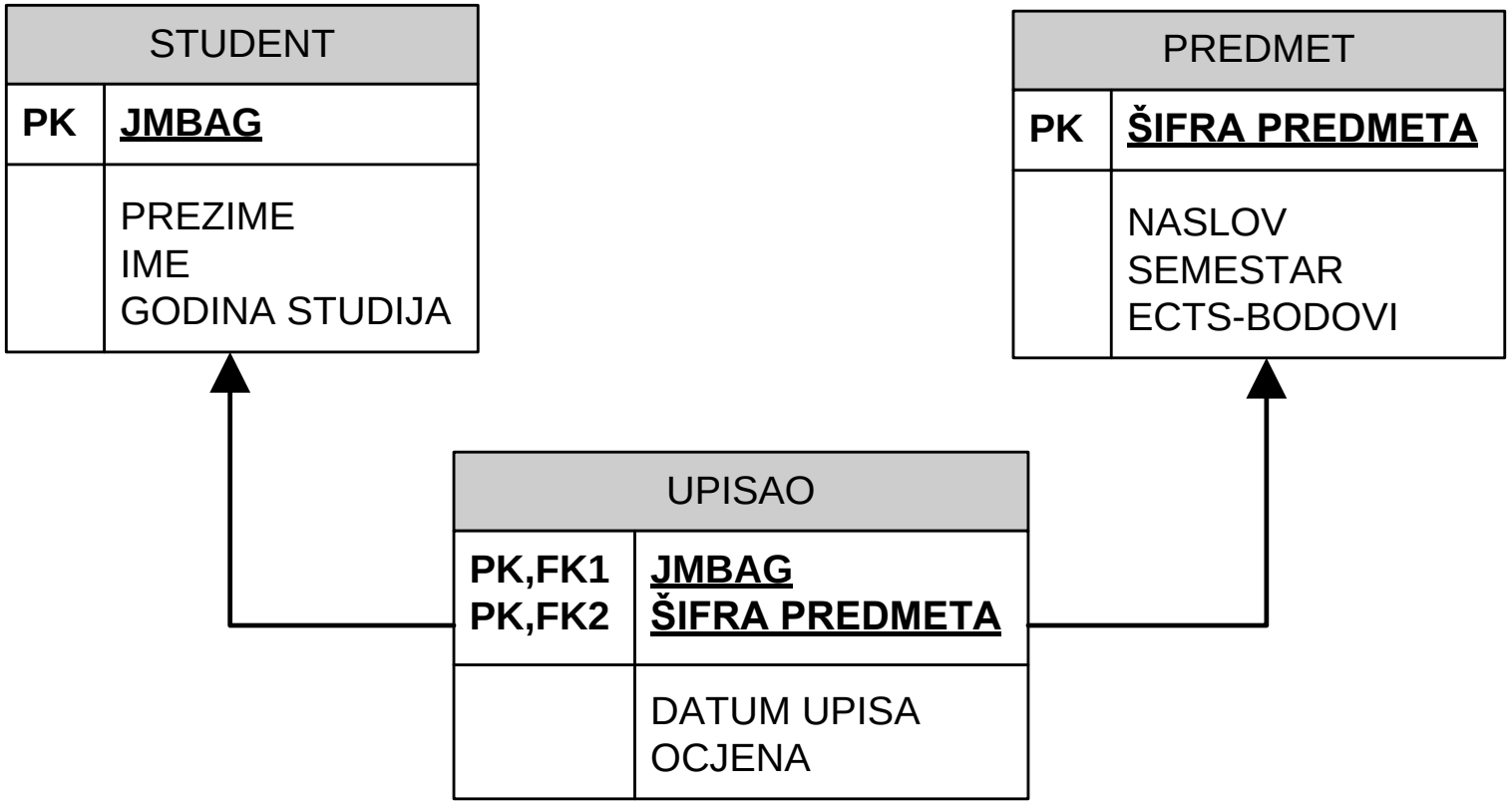
STUDENT(JMBAG, PREZIME, IME, GODINA STUDIJA)

PREDMET (ŠIFRA PREDMETA, NASLOV, SEMESTAR, ECTS-BODOVI)

UPISAO (JMBAG, ŠIFRA PREDMETA, DATUM UPISA, OCJENA)

Predlošci za izradu dokumentacije (7)

- Grafički prikaz relacijske sheme baze o studentima i predmetima



Predlošci za izradu dokumentacije (7)

- Rječnik podataka kao prilog relacijskoj shemi baze o studentima i predmetima

IME PODATKA	TIP	OPIS
JMBAG	Niz od točno 10 znamenki	Šifra koja jednoznačno određuje studenta
PREZIME	Niz znakova	Prezime studenta
IME	Niz znakova	Ime studenta
GODINA STUDIJA	Cijeli broj između 1 i 5	Godina koju je student upisao
ŠIFRA PREDMETA	Niz od točno 5 znamenki	Šifra koja jednoznačno određuje predmet
NASLOV	Niz znakova	Naslov predmeta
SEMESTAR	„Z“ ili „L“ (zimski ili ljetni)	Semestar u kojem se predmet predaje
ECTS-BODOVI	Mali cijeli broj	Bodovi koje student dobiva ako položi predmet
DATUM UPISA	Datum	Datum kad je određeni student upisao određeni predmet
OCJENA	Cijeli broj između 2 i 5	Ocjena koju je određeni student dobio iz određenog predmeta

Predlošci za izradu dokumentacije (8)

- Dokumentacija na fizičkoj razini sadrži fizičku shemu baze.
  - U slučaju uporabe DBMS-a zasnovanog SQL-u, fizička shema zapravo je niz naredbi u SQL-u ili nekom dodatnom komandnom jeziku.

- Postoji samo jedan predložak:
  - **Tekst sastavljen od ASCII znakova.**
    - Sadržaj tog teksta može se razlikovati ovisno o DBMS-u.
    - Svaki DBMS koristi se donekle različitom sintaksom SQL-a te raspolaže drukčijim komandnim jezikom.

Predlošci za izradu dokumentacije (9)

- Fizička shema baze o studen-tima i pred-metima.

Inačica SQL-a iz MySQL.

```
CREATE TABLE STUDENT
  (JMBAG NUMERIC(10) UNSIGNED NOT NULL,
   PREZIME CHAR(20),
   IME CHAR(20),
   GODINA_STUDIJA ENUM('1','2','3','4','5'),
   PRIMARY KEY(JMBAG))
   ENGINE=INNODB;
CREATE TABLE PREDMET
  (SIFRA_PREDMETA NUMERIC(5) UNSIGNED NOT NULL,
   NASLOV CHAR(80),
   SEMESTAR ENUM('Z','L'),
   ECTS_BODOVI NUMERIC(2) UNSIGNED,
   PRIMARY KEY(SIFRA_PREDMETA))
   ENGINE=INNODB;
CREATE TABLE UPISAO
  (JMBAG NUMERIC(10) UNSIGNED NOT NULL,
   SIFRA_PREDMETA NUMERIC(5) UNSIGNED NOT NULL,
   DATUM_UPISA DATE,
   OCJENA ENUM('2','3','4','5'),
   PRIMARY KEY(JMBAG,SIFRA_PREDMETA),
   INDEX R_JMBAG_IND (JMBAG),
   INDEX R_SP_IND (SIFRA_PREDMETA),
   FOREIGN KEY (JMBAG) REFERENCES STUDENT(JMBAG),
   FOREIGN KEY (SIFRA_PREDMETA)
     REFERENCES PREDMET(SIFRA_PREDMETA))
   ENGINE=INNODB;
```

Uporaba CASE-alata i drugih sw (1)

- Osnovni alat za izradu projektne dokumentacije je tekst procesor poput MS Word.
- Za izradu umetnutih dijagrama potreban nam je dodatni alat, i on se može odabrati na dva načina.
  - **Specijalizirani CASE alat.**
    - Paket koji služi softverskim inženjerima za razvoj softvera u skladu s određenom razvojnom metodom.
    - Podržano je crtanje svih dijagrama koje ta metoda propisuje, te je automatizirana izrada odgovarajuće razvojne dokumentacije.
    - Većina današnjih CASE alata omogućuje crtanje UML dijagrama, a neki imaju uključene i dodatne dijagrame koji su specifični za oblikovanje baze podataka.
    - Primjer: *Visual Paradigm.*

Uporaba CASE-alata i drugih sw (2)

- **Općeniti alat za crtanje dijagrama.**
  - Paket opće namjene za crtanje raznih vrsta dijagrama, koji sadrži razne grafičke predloške bez obzira na njihovo značenje.
  - Primjer: *MS Visio*. U njemu već postoje predlošci za većinu UML dijagrama i za grafički prikaz logičke sheme baze podataka, te također i elementi za sastavljanje Chen-ovih dijagrama.

- CASE alati su pogodniji za veće projekte na kojima rade iskusniji projektanti.
- Općeniti alati za crtanje bolji su za male projekte.