UNIVERZITET METROPOLITAN BEOGRAD

IT250 - BAZE PODATAKA

Primeri dobrih konceptualnih modela; Veza konceptualni – logički- fizički model baze podataka

Lekcija 05

PRIRUČNIK ZA STUDENTE

IT250 - BAZE PODATAKA

Lekcija 05

PRIMERI DOBRIH KONCEPTUALNIH MODELA; VEZA KONCEPTUALNI – LOGIČKI- FIZIČKI MODEL BAZE PODATAKA

- Primeri dobrih konceptualnih modela; Veza konceptualni – logički- fizički model baze podataka
- Poglavlje 1: Principi dobrog modelovanja konceptualnog modela
- Poglavlje 2: Primer izrade konceptualnog modela baze podataka
- Poglavlje 3: Klasni dijgrami kao alternativa ER dijagamima
- Poglavlje 4: Logički model baze podataka
- Poglavlje 5: Fizički model
- Poglavlje 6: Pokazna vežba
- Poglavlje 7: Domaći zadatak
- Zaključak

UVOD

Šta ćemo naučiti u ovoj lekciji?

U ovoj lekciji se najpre govori o principima dobrog modeliranja konceptualnih modela baza podataka koji se odnose na verodostojnost i redundantnost. **Princip verodostojnosti** je jedan od najvažnijih principa prilikom izrade konceptualnih modela baze podataka koji znači da model baze podataka treba da odgovara postavljenim specifikacijama, odnosno zahtevima za izradu aplikacije tj. da tipovi entiteta i njihovi atributi treba da odslikavaju realnost. **Redundantnost i anomalije unosa, ažuriranja i brisanja** su uobičajeni problemi koji se javljaju u ER dizajniranju. U ER modeliranju postoji nekoliko novih mehanizama kojima se može izbeći redundantnost.

Dat je jedan primer izrade konceptualnog modela podataka primenom analize teksta koji je često primenjivana tehnika u izradi konceptualnog modela.

Kao alternativa korišćenja ER modela za predstavljanje konceptualnih modela baza podataka, mogu se primenjivati UML klasni dijagrami.

Na kraju je objašnjeno kako se iz konceptualnog modela može doći do logičkog, a zatim i do fizičkog modela baze podataka, što jeste krajnji cilj modeliranja podataka.

Poglavlje 1

Principi dobrog modelovanja konceptualnog modela

VERODOSTOJNOST

Znači da tipovi entiteta i njihovi atributi treba da odslikavaju realnost

Princip verodostojnosti je jedan od najvažnijih principa prilikom izrade konceptualnih modela baze podataka koji znači da model baze podataka treba da odgovara postavljenim specifikacijama odnosno zahtevima za izradu aplikacije tj. da tipovi entiteta i njihovi atributi treba da odslikavaju realnost. U bazi podataka o filmovima, ne možete dodati atribut *number-of-cylinders* tipu entiteta STARS, jer taj atribut opisuje svojstvo automobila. Ono što je prikazano relacijama u relacionoj bazi podataka treba da ima smisla u odnosu na ono što znamo o delu realnosti koji modelujemo.
***Primer 1**: Ako u bazi podataka o filmovima definišemo relaciju Stars-in između tipova entiteta STARS i MOVIES, to bi trebalo da bude relacija više na prema više. Razlog je očigledan, jer u realnom svetu jedna zvezda filma se može pojaviti u više od jednog filma, dok u filmu može igrati više od jedne zvezde. Ako više na više relaciju predstavimo kao tip entiteta, korektno je da relacija Stars-in bude jedan na prema više u oba smera.*

S druge strane, nekada je manje očigledno šta realni svet zahteva od nas da uradimo u našem ER modelu.
***Primer 2**: Razmotrimo tipove entiteta COURSES i INSTRUCTORS, sa vezom Teaches između njih iz baze podataka univerziteta. Da li relacija između COURSES i INSTRUCTORS više na prema jedan? Odgovor zavisi od politike i namere organizacije za koju se kreira baza podataka. Moguće je da univerzitet ima politiku da može postajati samo jedan instruktor na jednom kursu. Međutim, mogu i više instruktora da timski predaju neki kurs. Ili, namera univerziteta može biti da ima informaciju ne samo o instruktorima koji trenutno drže kurs, već i onima koji su ga bilo kada držali ili koji su u stanju da ga drže. Zavisno od ovih slučajeva, potrebno je da relacija Teaches ima kardinalnost jedan prema jedan, jedan prema više ili više prema više.*

SPREČAVANJE REDUDANTNOSTI

U ER modeliranju postoji nekoliko novih mehanizama kojima se mogu izbeći redudantnost i anomalije

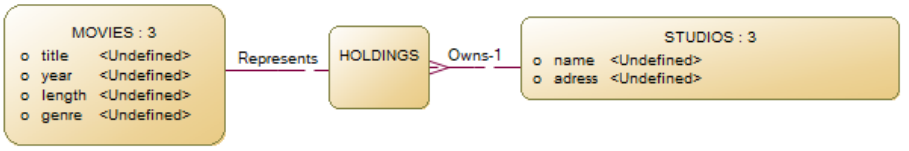
Redundantnost i anomalije unosa, ažuriranja i brisanja su uobičajeni problem koji se javljaju u ER dizajniranju. U ER modeliranju postoji nekoliko novih mehanizama kojima se može izbeći redundantnost.

Primer 1: *između tipova entiteta MOVIES i STUDIOS u bazi podataka filmova, koristili smo relaciju Owens. Umesto toga, možemo izabrati rešenje koje podrazumeva korišćenje atributa studioName u entitetu Movies. Mada u tome nema ništa pogrešnog, to je opasno iz nekoliko razloga.*

1. *Radeći tako nešto, može doći do ponavljanja nekih činjenica što bi rezultiralo potrebom za većim prostorom za čuvanjem podataka kada se E/R dijagram konvertuje u relacije radi konkretne implementacije*
2. *Postoji mogućnost za pojavom anomalije ažuriranja, jer mi možemo promeniti relaciju ali ne i atribut i obrnuto.*

Primer 2: *Pretpostavimo da umesto relacije između tipova entiteta MOVIIES i STUDIOS treba da postavimo relaciju "movie-holdings," koja opisuje studio koji je zakupac filma. Tada možemo kreirati još jedan tip entiteta HOLDINGS (zakupac) i između tipa entiteta HOLDINS i tipa entiteta MOVIES uspostaviti relaciju Represents tipa jedan na prema jedan a takođe i relaciju Owens tipa više na prema jedan između tipova entiteta HOLDINS i STUDIO čime je slika 1.1 kompletirana.*

Tehnički, struktura na slici 1. u potpunosti predstavlja realni svet, međutim, može se reći da tip entiteta HOLDINGS nema neku korisnu svrhu i da se kao takav može izostaviti. On čini program komplikovanijim, dovodi do nepotrebnog zauzeća prostora i povećava mogućnost pojave greške.

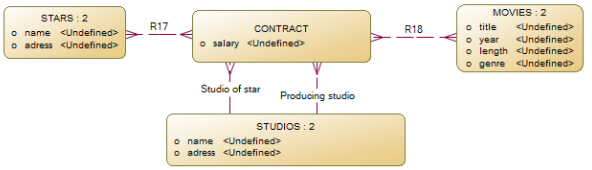


Slika 1.1 Loš dizajn sa suvišnim entitetom [Izvorr: NM IT350-2020/2021.]

KAKO IZABRATI PRAVU RELACIJU - SLUČAJ 1

Dodavanje svih mogućih relacija u dizajn baze podataka nije često dobra ideja.

Tipovi entiteta mogu biti povezani različitim relacijama u relacionom modelu. Međutim, dodavanje svih mogućih relacija u dizajn baze podataka nije često dobra ideja. To može dovesti do redundantnosti, anomalija ažuriranja, anomalija brisanja u smislu da povezan par tipova entiteta za jednu relaciju na dijagramu može biti izveden iz jedne ili više drugih relacija. Ova situacija će biti pokazana na dva primera:

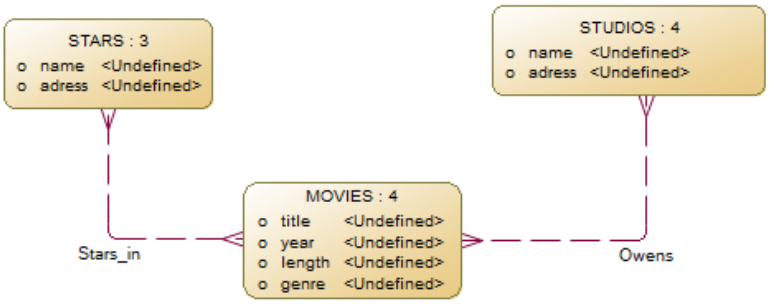


Slika 1.2 Primer relacije CONTRACT [Izvor: NM IT350-2020/2021.]

Slučaj 1: Pogledajmo sliku 1.2 gde smo povezali tipove entiteta MOVIES*, STARS i STUDIOS ternarnom relacijom Contracts koja je zbog ograničenja CASE alata PowerDesigner predstavljena kao tip entiteta CONTRACT.*

Da li je u ovom slučaju potrebno uspostaviti binarne relacije između MOVIES i STARS (Stars-in) i između MOVIES i STUDIOS (Owens)respektivno, kao na slici 1.3.

*Odgovor je: **ne znamo**; to zavisi od naših pretpostavki u pogledu ternarne relacije iz pitanja. Relaciju Stars-in je moguće izvesti na osnovu CONTRACT. Ukoliko neka zvezda može da se pojavi u filmu samo ako postoji ugovor koji se odnosi na tu zvezdu, film i studio, tada nema potrebe za relacijom Stars-in i treba izbaciti sve parove star-movie iz torki star-movie-studio u relaciji CONTRACT. Međutim, ako zvezda može raditi u filmu bez ugovora – ili što je verovatnije ako mi u našoj bazi ne znamo da ugovor postoji – tada bi trebalo da postoje parovi star-movie u Stars-in i ne sme se ukloniti deo torki star-movie-studio u CONTRACT. U tom slučaju, imamo potrebu da zadržimo Stars-in relaciju.*



Slika 1.3 Primer binarnih relacija. [Izvor: NM IT350-2020/2021.]

IZBOR PRAVE RELACIJE - SLUČAJ 2

Da li redudantne veze ponekad mogu da budu opravdane?

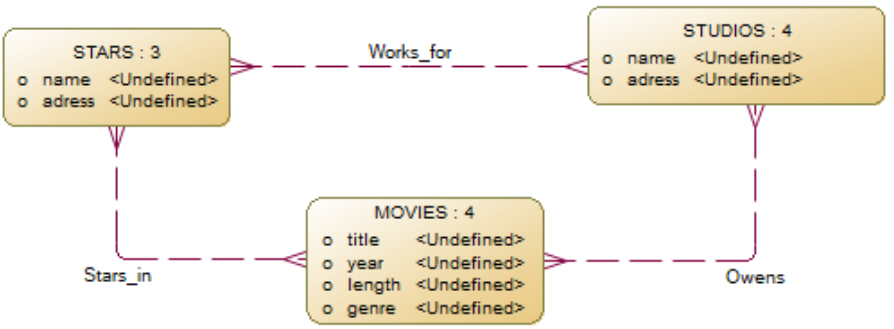
Slično posmatranje se može primeniti i na relaciju Owens sa slike 3. Ako za svaki film postoji bar jedan ugovor koji obuhvata film, njegov studio i neke zvezde filma, tada možemo izostaviti Owns.

Međutim, ako postoji mogućnost da je studio vlasnik filma a još uvek nema zvezdi pod ugovorom za taj film, ili se za takav ugovor ne zna u bazi podataka, tada moramo ostaviti Owens.

Kao zaključak, ne možemo reći da li su date relacije redundantne. To zavisi od onoga šta očekuje onaj ko implementira bazu podataka. Samo tada se može doneti razumna odluka o tome da li ili ne uključiti relacije kao što su Stars-in i Owens.
Slučaj 2: Razmotrimo ponovo dijagram predstavljen na slici 1.3. Na ovom dijagramu ne postoji relacija između STARS i STUDIOS, već kako bi napravili vezu možemo koristiti dve relacije Stars-in i Owns. Tako je zvezda povezana za neke filmove preko Stars-in a ti filmovi su povezani sa studios preko Owens.

Možemo reći da je zvezda povezana za studio koji je vlasnik filma u kojem se pojavljuju zvezde. Treba li na osnovu toga napraviti relaciju Works-for kao što je predstavljeno na slici 1.4?

Još jednom, to ne možemo reći a da ne znamo nešto više o tome. Prvo, šta bi bilo značenje te relacije? Ako ona treba da znači „zvezda se pojavljuje u bar jednom filmu tog studija“, tada verovatno postoji dobar razlog da se ona uključi na dijagram.Umesto toga, ta informacija se može izvesti iz Stars-in i Owens.



Slika 1.4 Dodavanje relacije Works_for između STARS i STUDIOS [Izvor: NM IT350-2020/2021.]

NASTAVAK O ODABIRU PRAVE RELACIJE - SLUČAJ 2

Zašto je nekada opravdano postojanje redudantnih veza - objašnjenje kroz slučaj 2

Međutim, možemo imati druge informacije o zvezdama koje rade za neki studio a koje se ne podrazumevaju vezom preko movie. U tom slučaju, relacija koja povezuje stars direktno sa studios može biti korisna i ne mora da bude redundantna.

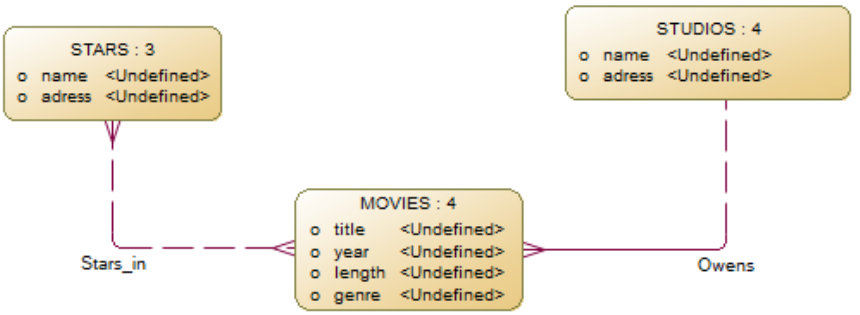
Alternativno, mi možemo koristiti relaciju između stars i studios kako bi označila nešto potpuno drugo. Na primer, ona može predstavljati činjenicu da je zvezda pod ugovorom sa studijom na način koji nije povezan sa filmom. Kao što je sugerisano u primeru, moguće je da zvezda bude pod ugovorom sa jednim studijom ali da takođe radi na filmu koji je u vlasništvu drugog studija. U tom slučaju, informacije koje se nalaze u novoj relaciji Works-for bi bila nezavisne od Stars-in i Owens relacija i sigurno bi bila ne redundantna.
Još jednom, to ne možemo reći a da ne znamo nešto više o tome. Prvo, šta bi bilo značenje te relacije? Ako ona treba da znači „zvezda se pojavljuje u bar jednom filmu tog studija“, tada verovatno postoji dobar razlog da se ona uključi na dijagram. Umesto toga, ta informacija se može izvesti iz Stars-in i Owens.

IZBOR PRAVE VRSTE ELEMENATA - SLUČAJ 1

Ponekada postoji više opcija u pogledu tipova elemenata koji se koriste da predstave koncepte realnog sveta. Atribut je jednostavniji za implementaciju od tipa entiteta ili relacija.

Ponekad, postoji više opcija u pogledu tipova elemenata koji se koriste da predstave koncepte realnog sveta. Mnogi od tih izbora se odnose na izbor između korišćenja atributa i korišćenja tipova entiteta ili relacija. Generalno, atribut je jednostavniji za implementaciju od tipa entiteta ili relacija. Ali, predstaviti sve kao atribut može dovesti do problema.

Slučaj 1: Razmotrimo jedan specifičan problem. Da li smo mudro uradili kada smo na slici 1.5 studio predstavili kao tip entiteta? Da li smo umesto toga ime i adresu studija mogli da pamtimo kao atribute tipa entiteta MOVIES i eliminišemo tip entiteta STUDIOS? Jedan problem koji bi nastao ako bismo tako uradili je da bi morali da ponavljamo adresu studija za svaki film. Takođe, ukoliko bismo promenili adresu za jedan film a za drugi je potrebno da ona ostane ista, javila bi se anomalija ažuriranja, a anomalija brisanja bi se javila ukoliko bi obrisali zadnji film koji je u vlasništvu nekog studija.



Slika 1.5 Izbor prave vrste elemenata - slučaj 1 [Izvor: NM IT350-2020/2021.]

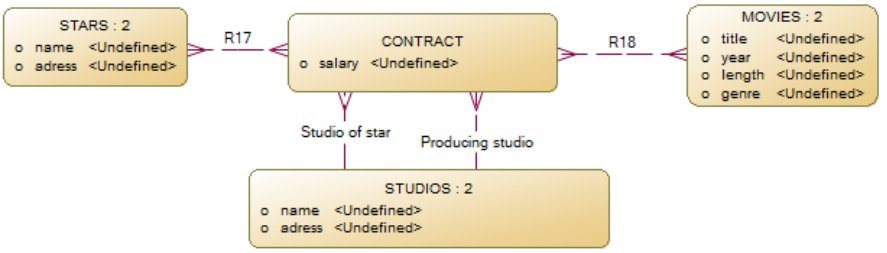
S druge strane, ukoliko ne bismo morali da pamtimo adresu studija, ne bi bilo štete da se ime studija pamti kao atribut filma. U tom slučaju, ne bi bilo anomalija. Moglo bi se reći da ime studija za svaki film nije redundantan podatak, obzirom da moramo da na neki način predstavimo vlasnika filma, pa se može smatrati da je taj način predstavljanja studija razuman.

KAKO ODABRATI PRAVE VRSTE ELEMENATA-SLUČAJ 2

Razmatranje razlike između korišćenja n-arnih relacija i više binarnih relacija kojima one mogu biti zamenjene.

Slučaj 2: *Razmotrimo sada razliku između korišćenja n-arnih relacija i više binarnih relacija kojima one mogu biti zamenjene.*
Mi smo posmatrali kvaternarnu relaciju Contracts između star, movie,
i dva studija na slici 1.6 koji smo konvertovali u tip entiteta CONTRACT.

Pretpostavimo da su ugovorom obuhvaćeni jedna zvezda, jedan film ali skup studija. To je situacija koja je mnogo složenija od one koju smo predstavili na slici 1.6 gde smo imali slučaj da jedan studio ima dve različite role. Međutim, uopšteno, možemo imati bilo koji broj studija - na primer, jedan u kojem se proizvodi (producira) film, jedan za specijalne efekte jedan za distribuciju itd. U tom slučaju ne možemo dodeliti rolu studiju.

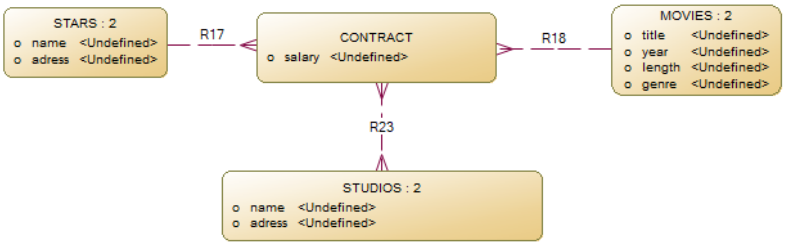


Slika 1.6 Kvaternarna relacija Contract između stars, movies,i dva studija

[Izvor: NM IT350-2020/2021.]

Očigledno je da svako pojavljivanja tipa entiteta CONTRACT mora da sadrži torke u formi (star, movie, set-of-studios) i da sama relacija Contracts obuhvata ne samo uobičajene tipove entiteta Stars i Movies već i novi tip entiteta čija pojavljivanja predstavljaju skup studija. Mada je ovakav pristup moguć, izgleda neprirodno razmišljati o pojavljivanjima studija kao o baznom entitetu, pa se to ne preporučuje.

Bolji pristup je razmišljati o ugovoru kao tipu entiteta a ne kao n-arnoj relaciji koja je na slici 6 predstavljena kao tip entiteta. *Tako je na slici 7 tip entiteta CONTRACT povezan sa* tipovima entiteta STARS*, MOVIES i STUDIOS, pri čemu nema ograničenja u broju studija. Relacija između CONTRACT i STUDIOS je više na prema više umesto jedan na prema više što je prikazano na slici 1.7. Napomenimo da je ugovor povezan sa samo jednom zvezdom i jednim filmom, ali sa proizvoljnim brojem studija.*



Slika 1.7 CONTRACT kao tip entiteta [Izvor: NM IT350-2020/2021.]

Poglavlje 2

Primer izrade konceptualnog modela baze podataka

ZAHTEVI ZA KUPOVINU CD-OVA PUTEM ONLINE SITE FIRME CD SELECTIONS

Zahtevi specificirani Use case opisom Plasiranje porudžbenice.

Na slici 2.1 je dat opis procedure kupovine CD-ova putem online site firme CD Selections iskazan korišćenjem Opisa slučaja korišćenja

Ime slučaja korišćenja: Plasiranje porudžbine; **ID**:3; **Nivo važnosti**: Visok

Primarni aktor: Kupac; **Tip UC**: Detaljni, Osnovni

Stejkholderi i interesenti: Kupac-želi da pretaži Web site kako bi kupio CD
EM menadžer-želi da maksimalno zadovolji kupca

Kratak opis: Ovaj UC opisuje kako kupac može pretraživati Web site i plasirati porudžbenicu

Relacije:
1. Asocijacije: Kupac
2. Uključuje: Završetak kupovine, Održavanje porudžbine
3. Proširuje:
4. Generalizacija:

Normalni tok događaja:
1. Kupac šalje (submit-uje) svoj zahtev za pretraživanje sistemu.
2. Sistem prikazuje kupcu listu preporučenih Cd-ova
3. Kupac bira jedan od CD-ova kako bi o njemu našao dodatne informacije
4. Sistem prikazuje kupcu osnovne informacije i preglede koje se odnose na CD
5. Kupac izvršava UC Održavanje porudžbenice
6. Kupac ponavlja korake od 3 do 5 dok ne završi kupovinu
7. Kupac izvršava UC Završavanje kupovine (Check out)
8. Kupac napušta Web site

Alternativni tokovi:
3a-1. Kupac šalje sistemu novi zahtev za pretraživanje.
3a-2. Kupac ponavlja tokove 2 do 3 dok ne bude zadovoljan rezultatima pretraživanja ili ne odustane od kupovine
7a-1. Kupac poništava porudžbenicu

Slika 2.1 Opis slučaja korišćenja za kupovinu CD-ova putem online site firme CD Selections [Izvor: NM IT350-2020/2021.]

ZAHTEVI ZA DIZAJNIRANJE SISTEMA ZA INTERNET PRODAJU U CD SELECTIONS: ODRŽAVANJE INFORMACIJA

Održavanje informacija o CD-ovima, Održavanje CD marketinških informacija.

Funkcionalni zahtevi za projektovanje sistema za internet prodaju u CD Selections su:

1. Održavanje informacija o CD-ovima

- Sistemu za internet prodaju je potrebna baza informacija o CD-ovima koji će se prodavati preko interneta, slično bazi podataka CD-ova u svakoj maloprodajnoj prodavnici (npr. naslov, muzičar, ID broj, količina na zalihama)
- Svakog dana, sistem za internet prodaju prima podatke od sistema za distribuciju koji se koriste za ažuriranje baza podataka o CD-ovima. Neki od CD-ova se dodaju, neki se brišu a neki menjaju (npr. cene)
- Menadžer marketinga za on-line prodaju (pozicija koja tek treba da se kreira) takođe može da ažurira informacije (npr. cena prodaje)

2. Održavanje CD marketinških informacija

- Sistem za internet prodaju pruža dodatne informacije tržištu CD-ova za postojeće i nove kupce. Sistem treba da obezbedi bazu podataka marketinškog materijala o izabranim CD-ovima koji će pomoći Web korisnicima da o njima više saznaju (npr. muzičke časopise, linkove do Web sajtova, informacije o umetnicima i jednostavne muzičke klipove). Kada se prikažu informacije o CD-ovima koje imaju dodane marketinške informacije, tim informacijama može da se pristupi preko linka.
- Da bi bolje promovisali svoje CD-ove, marketinške materijale prvenstveno obezbeđuju sami prodavci CD-ova. Menadžer marketinškog odeljenja će odrediti koji će marketinški materijali biti stavljeni na sistem i biti odgovorni za dodavanje, menjanje i brisanje materijala.

ZAHTEVI ZA DIZAJNIRANJE SISTEMA ZA INTERNET PRODAJU U CD SELECTIONS: POPUNJAVANJE NARUDŽBENICE

Popunjavanje narudžbenice

3. Plasiranje porudžbine za kupovinu CD-ova

- Kupci će pristupiti sistemu za internet prodaju kako bi pregledavali CD-ove za koje su zainteresovani. Neki od kupaca će tražiti CD-ove koji su za njih interesantni po određenim umetnicima, dok će drugi kupci CD-ove pretraživati po kategorijama (npr. rok, džez ili klasična muzika).
- Kada kupac pronađe CD-ove za koje je zainteresovan, on se isključuje (chech out) iz sistema dajući u tom trenutku personalne informacije (npr. ime, e-mail, adresu, kreditnu karticu) i informacije koje se odnose na porudžbine (npr. CD-ovi koji su naručeni i količina za svaku od njih).
- Sistem će informacije o kreditnim karticama korisnika verifikovati preko online centra za odobravanje kreditnih kartica i na osnovu toga će porudžbine biti odbacivane ili prihvatane.
- Kupac će biti u stanju da proveri i vidi da li njegova najbliža prodavnica ima traženi CD na zalihama. Da bi se pronašla najbliža prodavnica treba koristiti zip kod. Ako je CD

raspoloživ u toj prodavnici, kupac može trenutno da pošalje rezervaciju za CD-om koji se nalazi na skladištu a zatim ode u prodavnicu i uzme svoj CD.
- Ako CD nije raspoloživ u najbližoj prodavnici, kupac može tražiti da prodavnica naruči taj CD i da ga kasnije uzme. Kupac će mail-om biti obavešten kada traženi CD stigne u prodavnicu; CD će po dolasku u prodavnicu biti rezervisan (rezervacija ističe posle 7 dana).
- Alternativno, kupac može da porudžbinu za CD pošalje mail-om.

4. Popunjavanje porudžbine preko mail-a

- Kada se CD naručuje preko mail-a, sistem za internet prodaju će mail proslediti sistemu za distribuciju.
- Sistem za distribuciju će obraditi sve pošiljke kupaca CD-ova; on obaveštava sistem za internet prodaju i šalje mail kupcu.
- Menadžer će nedeljno generisati izveštaje za proveru statusa narudžbina.

PRAVILA "ANALIZE TEKSTA"

Koriste se da bi na osnovu tekstualno opisanih zahteva identifikovali kandidate za entitete, atribute i relacije.

Da bi identifikovali kandidate za entitete, atribute i relacije mogu se koristiti pravila "analize teksta" koja glase:

1. ***Zajedničke imenice ili imenice koje nisu vlastite*** **su kandidati za tipove entiteta.**
2. ***Vlastite imenice*** **podrazumevaju instance entiteta.**
3. ***Zbirne imenice*** **podrazumevaju tipove entiteta sačinjenih od grupe drugih entiteta.**
4. ***Pridev*** **podrazumeva atribut tipa entiteta.**
5. ***Glagol "biti"*** **podrazumeva relaciju generalizacije između instanci entiteta i tipa entiteta.**
6. ***Glagol „imati"*** **podrazumeva vezu agregacije ili asocijacije.**

1. Korišćenjem ovih pravila nad opisom normalnog toka događaja Use case-a "Plasiranje porudžbenice", mogu se identifikovati sledeći tipovi entiteta: **Kupac, Zahtev za pretraživanje, CD, Lista CD**, i **Pregled.** Mogu se izdvojiti tri različita tipa zahteva za pretraživanje: **Pretraživanje po naslovu, Pretraživanje po umetniku** i **Pretraživanje po kategoriji**.

2. Tekstualnom analizom kratkog opisa koji je dat u okviru Use case-a "Plasiranje porudžbenice", otkrivena je još jedan kandidat za tip entiteta: **Narudžbenica**.

3. Pregledavanjem glagola koji su sadržani u tekstu UC, može se zaključiti da **Kupac plasira Porudžbenicu** i da **Kupac formira Zahtev za pretraživanje.**

4. Pregledavanjem originalno postavljenih zahteva sistema može se identifikovati skup atributa za tip entiteta **Kupca (ime, adresa, e-mail, kreditna kartica)** i **Narudžbenica**

(CD koji se naručuje i količina) i otkriti dodatni kandidati za tipove entiteta: **Kategorije CD-ova** i **Centar za proveru kreditnih kartica.**

PRIMENA PRAVILA “ANALIZE TEKSTA” NAD OSTALIM ZAHTEVIMA SISTEMA

Primenjena pravila nad ostalim zahtevima sistema.

5. Na kraju, identifikovane su tri podklase tipa entiteta **CD kategorije: Rok, Jazz i Klasika**.

6. Daljom analizom se mogu pronaći dodatni kandidati za tipove entiteta, atribute i relacije, npr. dodatni skup atributa za entitet **CD–naslov**, umetnik koji izvodi kompoziciju, količina na zalihama, cena i kategorija.

7. Što se tiče porudžbenice, treba sačuvati iznos sračunate takse, troškove isporuke i ukupni total.

8. Sadašnji atributi u **Porudžbenici** (CD-ovi koje kupac želi da kupi i količina koja se kupuje) podrazumevaju da kupac može da naruči više kopija istog CD-a, a takođe da se u okviru iste porudžbenice može naći više različitih CD-ova. Zbog toga je kreiran nov tip entiteta **Stavka porudžbenice** koja je povezana i sa tipom entiteta **Porudžbenica** i sa tipom entiteta **CD**.

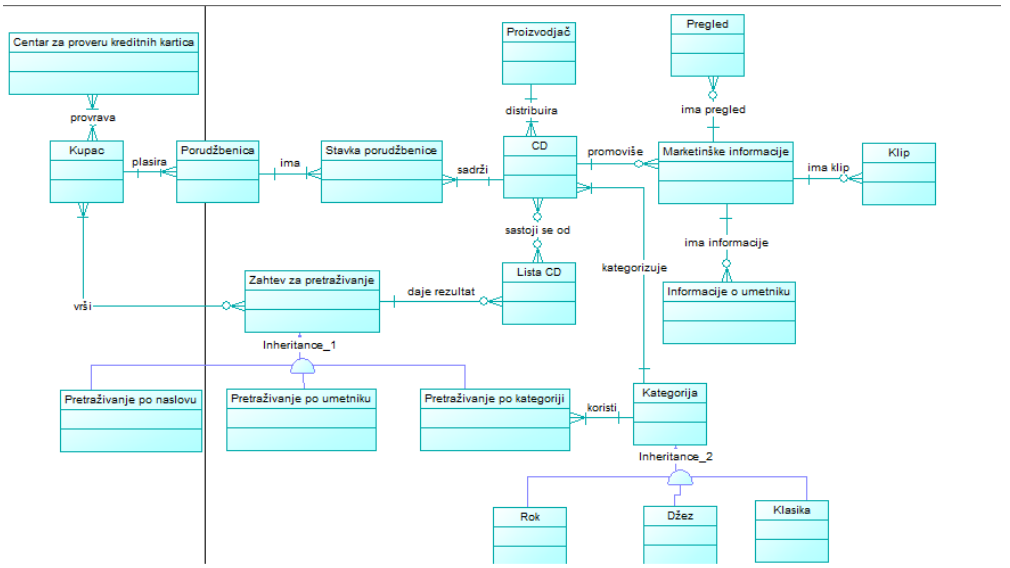
9. Ovaj novi tip entiteta ima samo atribut količina ali ima i dve relacije: jednu sa Porudžbenica i drugu sa CD.

10. Takođe se može utvrditi da marketinške informacije koje daje proizvođač treba da budu poseban tip entitet koji treba pridodati CD-u.

PRELIMINARNI ER DIJAGRAM ZA SISTEM INTERNET PRODAJE

Kreiran dijagram na osnovu primenje analize teksta.

Na slici 2.2 je prikaz preliminarni ER dijagram za sistem Internet prodaje.



Slika 2.2 Preliminarni ER dijagram za sistem Internet prodaje. [Izvor: NM IT350-2020/2021.]

Pažljivim pregledavanjem ER modela se može doći do zaključka da:

1. Tip entiteta CD kategorije i njene podklase nisu neophodne. Za njih nisu definisani nikakvi atributi pa je CD kategorija modelirana kao atribut CD-a

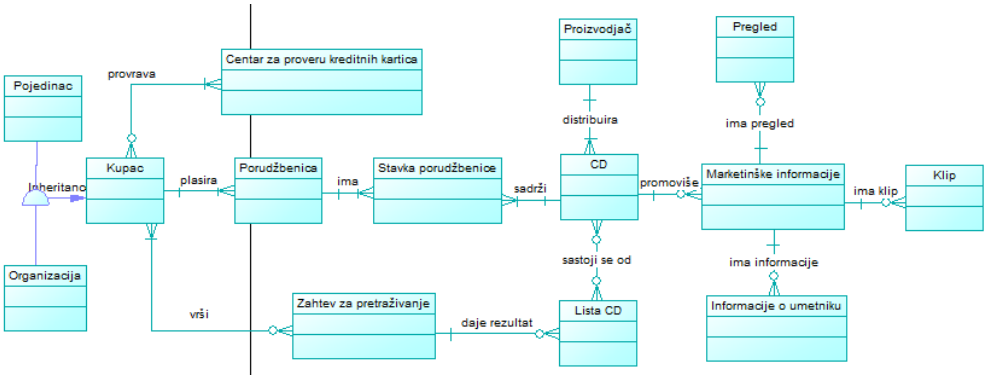
2. Takođe, pregledavanjem tipa entiteta Zahtev za pretraživanje i njegovih podklasa zaključeno je da podklase nisu ništa više nego skup operacija za klasu Zahtev za pretraživanje

3. Daljom analizom se može uvideti da tip entiteta ***Kupac može imati dve podklase: Individualni kupac i Organizacija***.

ER DIJAGRAM ZA SISTEM INTERNET PRODAJE

Kreiran je unapređenjem preliminarnog ER dijagrama.

Na slici 2. 3 je prikazan ER dijagram Sistema za Internet prodaju u CD Selections. On predstavlja konceptualni model koji uključuje sve prethodno opisane izmene.



Slika 2.3 ER dijagram Sistema za Internet prodaju u CD Selections [Izvor: NM IT350-2020/2021.]

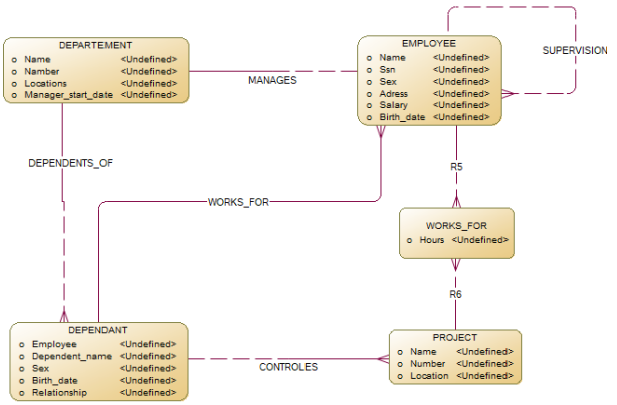
Poglavlje 3

Klasni dijgrami kao alternativa ER dijagamima

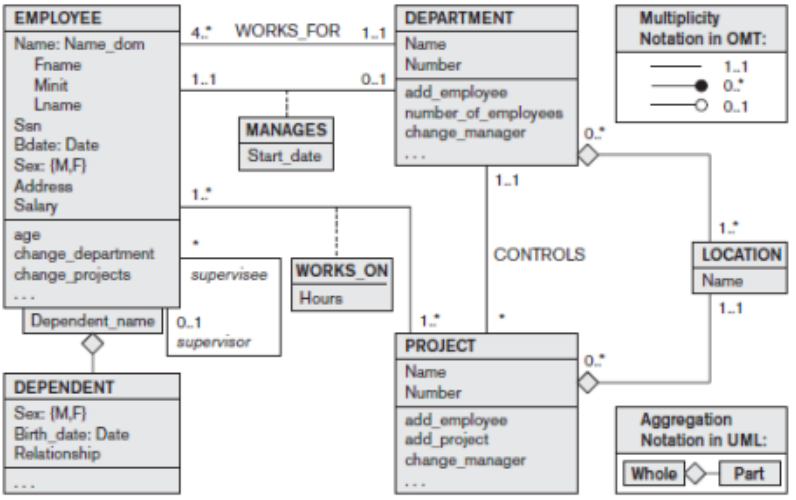
UML KLASNI DIJAGRAMI

Dijagrami klasa mogu se smatrati alternativnom notacijom ER dijagrama

UML metodologija se u velikoj meri koristi u projektovanju softvera i ima mnogo tipova dijagrama za različite svrhe projektovanja softvera. Ovde ćemo ukratko predstaviti osnove UMLklasnih dijagrama i uporediti ih sa ER dijagramima. Na neki način, dijagrami klasa mogu se smatrati alternativnom notacijom ER dijagrama. Slika 3.2 pokazuje kako se shema baze podataka sa slike 3.1 predstavljena ER dijagramima može prikazati koristeći UML oznaku klase dijagrama. Tipovi entiteta na slici 3.1 su modelirani kao klase na slici 3.2. Entitet u ER odgovara objektu u UML-u.



Slika 3.1 Šema baze podataka COMPANY predstavljena E/R dijagramom [Izvor: NM IT350-2020/2021.]



Slika 3.2 UML klasni dijagram za COMPANY. [Izvor: NM IT350-2020/2021.]

PREDSTAVLJANJE TIPOVA ENTITETA I VEZA

Tipovi entiteta se predstavljaju kao klase, a entiteti kao objekti klasa. Veze se nazivaju asocijacijama u UML terminologiji, a ograničenja relacija višestrukost (eng. multiplicities)

U UML dijagramima klasa, klasa (slična tipu entiteta u ER) je prikazana kao pravougaonik (vidi sliku 3.2) koja uključuje tri dela: Gornji deo daje ime klase (slično imenu tipa entiteta); srednji deo uključuje atribute; a poslednji dep obuhvata operacije koje se mogu primeniti na pojedinačne objekte (slično pojedinačnim entitetima u skupu entiteta) klase. Operacije nisu navedene u ER dijagramima. Razmotrite klasu EMPLOYEE na slici 3.2. Njeni atributi su Name, Ssn, Bdate, Sex, Address, and Salary. Projektant može po želji da odredi domen atributa ako želi, postavljanjem dvotočke (:) praćeno nazivom domena ili opisom, kao što je prikazano atributima Name, Sex, and Bdate attributes na EMPLOYEE na slici 3.2.

Složeni atribut je modeliran kao strukturirani domen, kao što je prikazano na atributu Name u EMPLOYEE. Višestruki atribut će uglavnom biti modelovan kao posebna klasa, kao što je ilustrovano LOCATION klasom na slici 3.2.

Tipovi veza se nazivaju asocijacijama u UML terminologiji, a instance veze se nazivaju veze. Binarna asocijacija (binarni tip odnosa) je predstavljena kao linija koja povezuje klase koje učestvuju (tipovi entiteta), i može opciono imati ime. Atribut veze, nazvan link atribut, se postavlja se u okvir koji je povezan sa linijom asocijacije isprekidanom linijom. Oznaka (min, max) se koristi za određivanje ograničenja relacije, koja se nazivaju višestrukost relacija (eng. multiplicities) u UML terminologiji. Višestrukosti su navedene u obliku min..maks, a zvezdica (*) označava da ne postoji maksimalno ograničenje za učešće.

U UML-u, jedna zvezdica označava mnoštvo od 0 .. *, a jedna 1 označava mnoštvo od 1..1. Rekurzivna veza se u UML-u naziva refleksivna asocijacija, a imena uloga - kao i višestrukost i - se postavljaju na suprotnim krajevima asocijacije kada se uporede sa stavljanjem imena uloga na slici 1. U UML-u postoje dva tipa odnosa: asocijacija i agregacija

VEZA AGREGACIJA U UML DIJAGRAMIMA

Agregacija treba da predstavi odnos između celog objekta i njegovih sastavnih delova, i ima različitu shematsku notaciju u odnosu na asocijaciju.

Veza agregacija treba da predstavi odnos između celog objekta i njegovih sastavnih delova, i ima različitu shematsku notaciju. Na slici 3.2, modelirali smo lokacije odeljenja i lokaciju projekta kao agregacije. Međutim, agregacija i asocijacija nemaju različita strukturalna svojstva, a izbor o tome koji tip relacije treba koristi je donekle subjektivan. U ER modelu, oba su predstavljena kao relacije. UML takođe razlikuje jednosmerne i dvosmerne asocijacije (ili agregacije). U jednosmernom slučaju, linija koja povezuje klase se prikazuje sa strelicom da bi ukazala da je potreban samo jedan smer za pristupanje povezanim objektima. Ako nije prikazana nijedna strelica, pretpostavlja se dvosmerni slučaj, što je podrazumevajuća vrednost. Na primer, ako uvek očekujemo pristup menadžeru odeljenja počevši od objekta DEPARTMENT, nacrtali bismo liniju asocijacije koja predstavlja MANAGES asocijaciju sa strelicom od DEPARTMEN do EMPLOYEE.

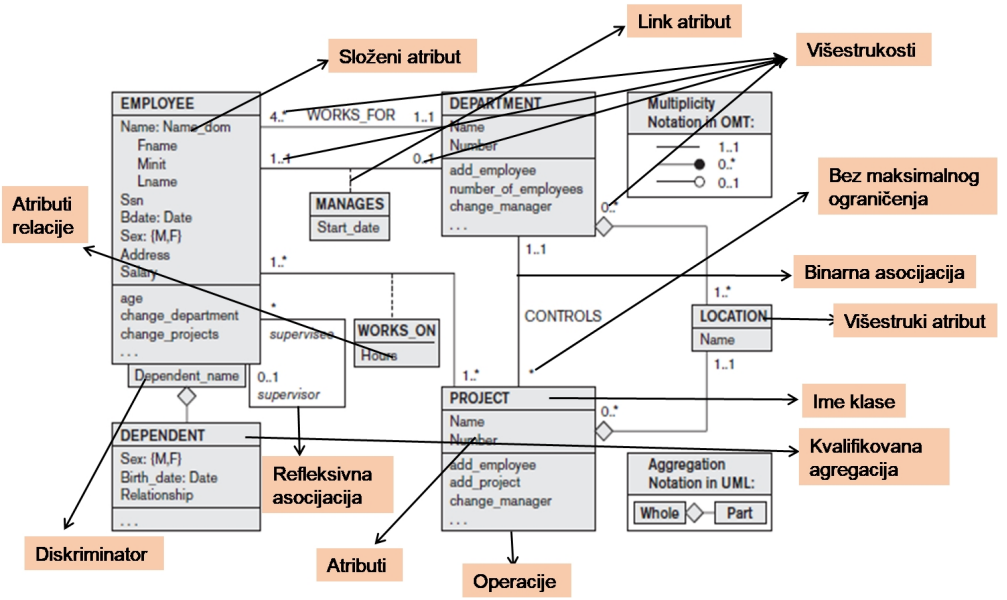
Osim toga, instance veze mogu biti uređene. Na primer, mogli bismo da navedemo da objekti zaposlenog koji su u vezi sa svakim odeljenjem preko WORKS_FOR asocijacije (relacije) treba da budu poređani po njihovoj vrednosti atributa Salary. Imena asocijacija (relacija) su opciona u UML-u, a atributi relacija se prikazuju uokvireno u pravougaoniku koji je priključen isprekidanom linijom na liniju koja predstavlja asocijaciju / agregaciju (pogledajte Start_date i Hours na slici 3.2).

Operacije date u svakoj klasi su izvedene iz funkcionalnih zahteva aplikacije. Dovoljno je navesti početne nazive operacija za logičke operacije za koje se očekuje da će se primenjivati na pojedinačne objekte klase, kao što je prikazano na slici 3.2. Kako se projekat unapređuje, dodaje se više detalja, kao što su tačne vrste argumenata (parametri) za svaku operaciju, kao i funkcionalni opis svake operacije. UML ima opise funkcija i dijagrame sekvenci koje specificiraju neke detalje operacija, ali one su izvan okvira naše diskusije.

Slabi entiteti se mogu modelovati korišćenjem konstrukcije koja se zove kvalifikovana asocijacija (ili kvalifikovana agregacija) u UML-u; ovo može predstavljati i identifikacionu relaciju i parcijalni ključ, koji se nalazi u pravougaoniku koji je pridružena vlasničkoj klasi. Ovo je ilustrovano klasom DEPARTEMENT i njenim kvalifikovanim agregiranjem u EMPLOYEE na slici 3.2. Delimični ključ Dependent_name naziva se diskriminator u UML terminologiji, pošto njegova vrednost razlikuje objekte pridružene (povezane sa) sa istim zaposlenim. Kvalifikovana udruženja nisu ograničena na modeliranje slabih entiteta, i mogu se koristiti za modeliranje drugih situacija u UML-u. Ovaj deo nije zamišljen da bude kompletan opis UML dijagrama klase, već da ilustruje jedan popularan tip alternativne shematske notacije koja se može koristiti za predstavljanje ER modela.

SAŽETI PRIKAZ UML KLASNOG DIJAGRAMA

Prikazan je detaljno objašnjen UML klasni dijagram



Slika 3.3 UML klasni dijagram. Izvor: [Autor]

Poglavlje 4

Logički model baze podataka

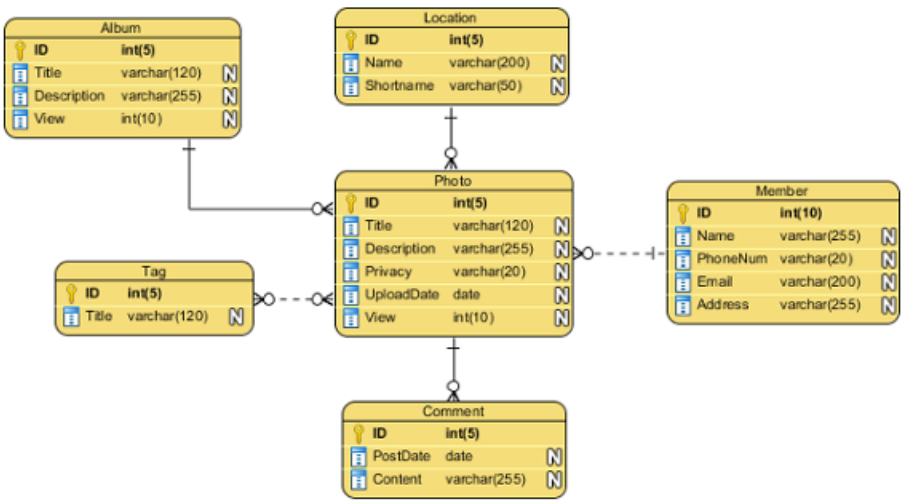
RAZLIKA LOGIČKOG U ODNOSU NA KONCEPTUALNI MODEL

Predstavlja model koji je nezavistan od određenog sistema za upravljanje bazom podataka ali se vezuje za određenu struktura podataka (relacionu, objektnu itd.)

Logički model podataka ili logička shema predstavlja model podataka određenog domena problema koji se izražava nezavisno od određenog sistema za upravljanje bazom podataka (DBMS) ili tehnologije skladištenja (fizički model podataka), ali se vezuje za određenu struktura podataka kao što su relacione baze podataka, objektno orijentisane baze, objektno relacione ili tekstualne baze. Ovo je suprotno konceptualnom modelu podataka, koji opisuje semantiku organizacije bez pozivanja na tehnologiju.

Logički modeli podataka predstavljaju apstraktnu strukturu domena informacija. Oni se često prikazuju preko dijagrama i najčešće se koriste u poslovnim procesima koji pokušavaju da predstave stvari od značaja za jednu organizaciju. Kada je validiran i odobren,logički model podatka može postati osnova fizičkog modela podataka i formirati bazu podataka.
Logički modeli podataka treba da budu zasnovani na strukturama identifikovanim u prethodno urađenom konceptualnom modelu podataka, jer on opisuje semantiku informacija, koju opisuje i logički model. Čak i tako, pošto logički model podataka predviđa implementaciju na određenom računarskom sistemu, sadržaj logičkog modela podatka se prilagođava kako bi se postigla određena efikasnost.

Na slici 4.1 je dat primer logičkog modela.



Slika 4.1 Primer logičkog modela [Izvor: NM IT350-2020/2021.]

ŠTA PREDSTAVLJA LOGIČKI MODEL BAZE PODATAKA?

Logičko projektovanje relacionih baza podataka je transformisanje konceptualnog modela u šemu koji se korišćenjem DDL unosi u SUBP

Logički model podataka definiše strukturu podataka pogodnu za projektovanje i implementaciju baze podataka ili za razvoj aplikacija koje koriste bazu. Logički model se sastoji od potpuno normalizovanih objekata čiji su svi atributi definisani. Pored toga, definisan je tip podataka i domen svih atributa, kao i atributi koji su kandidati za ključeve. Drugim rečima, logički model treba da bude kompletan opis baze podataka na osnovu koga će se razviti željena baza podataka.

Može se reći da je logičko projektovanje relacionih baza podataka postupak kojim se ranije definisani konceptualni model transformiše u šemu, odnosno opis baze podataka u strukturu relacione baze podataka. Cilj je da se dobije potpuno definisan relacioni model podataka koji se onda korišćenjem DDL jezika unosi u sistem za upravljanje bazama podataka (SUBP) i tako definiše fizički model podataka

Ako je konceptualni model već preveden u relacioni model, na primer normalizacijom tabela, nikakva transformacija nije potrebna. Međutim, ako je konceptualni model izrađen kao model objekti – veze, potrebno je izvršiti njegovu transformaciju u relacioni model.

Prednosti izrade logičkog modela podataka kao posrednika između konceptualnog i fizičkog modela, umesto direktnog generisanja fizičkog modela podataka su sledeće:

1. Obzirom da se proizvodi na osnovu konceptualnog modela podataka, logički model podatka odražava zahteve za poslovnim informacijama, u koje nisu ugrađeni zahtevi za performansama već pravila o svojstvima podataka (kao što su funkcionalne zavisnosti). Ova pravila se ne mogu uvek izvesti iz fizičkog modela podataka, koji može biti denormalizovan ili izmenjen na neki drugi način.
2. Ako se baza podataka prenese na drugi DBMS koji podržava slične strukture (npr., drugi relacioni DBMS ili novu verziju istog DBMS-a koji ima različite performanse), logički model podataka se može koristi kao osnova za novi fizički model podataka. Zadatak prevođenja konceptualnog modela podataka u relacioni logični model je sasvim jednostavan, a čak je i za velike modele malo verovatno da će trajati više od nekoliko dana. Zapravo, mnogi CASE alati pružaju mogućnosti da se logički model podataka automatski generiše iz konceptualnog modela.

VIDEO

How to convert an ER diagram to the Relational Data Model

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 5

Fizički model

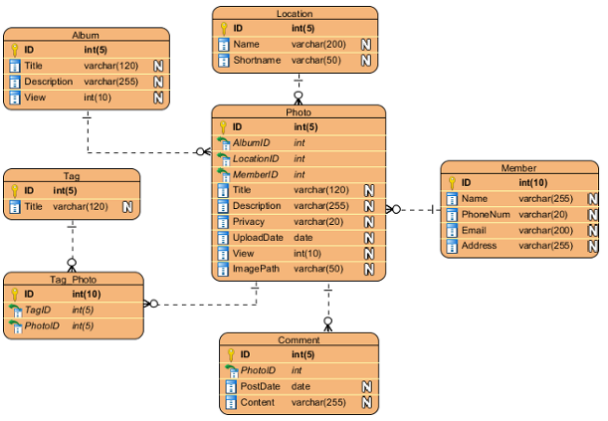
ŠTA JE FIZIČKI MODEL PODATAKA?

Predstavlja način na koji treba struktuirati i povezati podatke u određenom DBMS-u, tako da je važno razmotriti konvenciju i ograničenje DBMS-a koji se koristi

Fizički model podataka predstavlja način na koji treba struktuirati i povezati podatke u određenom DBMS-u, tako da je važno razmotriti konvenciju i ograničenje DBMS-a koji se koristi prilikom projektovanjafizičkog E/R dijagrama. To znači da je za kolone entiteta potrebno specificirati tačne tipove podataka, i izbeći korišćenje rezervisanih reči u imenima entiteta i kolona. Osim toga, dizajneri baze podataka mogu dodati i primarne ključeve, strane ključeve i ograničenja dizajna.

Na slici 5.1 je dat primer fizičkog modela za prethodno urađen logički model podataka.

Većina CASE alata koja prati RDBMS vrše automatsku transformaciju modela entiteti-veze u relacioni model i generišu bazu podataka u izabranom RDBMS. Mada se transformacija logičkog modela u fizički model može izvršiti automatski korišćenjem CASE alata, potrebno je znati pravila transformacije koja će ukratko biti opisana u ovom, a detaljnije u sledećem predavanju.



Slika 5.1 Fizički model podataka [Izvor: NM IT350-2020/2021.]

Osim iz logičkog modela, fizički model baze podataka se može dobiti i inverznim (reverznim) inženjeringom. Reverzni inženjering za baze podataka je proces analize postojeće baze podataka kako bi se dobili detalji o njenom dizajnu, strukturi i funkcionalnostima. Ovaj postupak se obično koristi kada imamo bazu podataka, ali nemamo ili imamo samo delimične informacije o njenom izvornom modelu ili dokumentaciji.

Cilj reverznog inženjeringa je razumevanje kako je baza podataka izgrađena kako bi se olakšali potrebni popravci, unapređenja ili integracije sa drugim sistemima.

Evo nekoliko koraka u procesu reverznog inženjeringa za baze podataka: prikupljanje informacija, analiza podataka, modeliranje baze podataka, generisanje SQL skripti, provera konzistentnosti, identifikacija poslovnih pravila.

Reverzni inženjering za baze podataka može biti složen proces, posebno ako nemate dovoljno informacija o izvornoj bazi. Međutim, to može biti dragoceno sredstvo za održavanje i unapređenje postojećih sistema, kao i za integraciju sa drugim aplikacijama ili bazama podataka.

TRANSFORMACIJA LOGIČKOG U FIZIČKI MODEL BAZE PODATAKA

Transformacija logičkog modela u fizički model podrazumeva transformisanje koncepata logičkog modela u fizičke objekte

Fizički model podataka se kreira da bi se izvršila transformacija logičkog modela u fizičku implementaciju baze podataka korišćenjem nekog RDBMS-a. Ovaj model služi administratoru baze podataka da implementira specificiranu bazu.

Da bi se uspešno kreirao fizički model potrebno je dobro poznavati osobine izabranog RDBMS, kao što su:

- koje objekte podržava RDBMS i koje su fizičke strukture i datoteke potrebne za podršku tih objekata,
- kako je podržano indeksiranje i održavanje referencijalnog integriteta
- koji su tipovi podataka na raspolaganju
- koji su tipovi ograničenja na raspolaganju

Prvi korak u transformisanju logičkog modela u fizički model je transformisanje koncepata logičkog modela u fizičke objekte. Ovo podrazumeva sledeće aktivnosti:

1. **Transformacija entiteta u tabele**
2. **Transformacija atributa u kolone**
3. **Transformacija domena u tipove podataka i ograničenja**
4. **Specifikacija primarnih ključeva**
5. **Specifikacija redosleda kolona**
6. **Definisanje referencijalnih ograničenja za sve veze**
7. **Definisanje prostora tabela (tablespace)**

KORACI U IMPLEMENTACIJI FIZIČKOG MODELA

Transofrmacija relacija u tabele, atributa u kolone, domena u tipove podataka i ograničenja, specficikacija primarnih ključeva, specifikacija redosleda kolona

1. Transformacija entiteta u tabele. Fizički pandan relacijama, odnosno tipovima entitetima, je tabela. Svaki entitet odnosno instanca tipa entiteta će se transformisati u jedan red tabele. U većini slučajeva, svakoj relaciji iz logičkog modela će odgovarati jedna tabela u fizičkom modelu. Međutim, postoje slučajevi kada je neke tabele potrebno spojiti ili razdvojiti da bi se postigle bolje performanse baze podataka.

2. Transformacija atributa u kolone. Fizički pandan atributima su kolone u tabelama. Atributi svakog objekta treba da budu preslikani u kolone odgovarajuće tabele.

3. Transformacija domena u tipove podataka i ograničenja. Za svaki logički domen atributa je potrebno odabrati adekvatan tip podataka iz seta tipova koji nudi SUBP. Na primer, za godinu rođenja se može izabrati tip podataka *integer*, a za matični broj, koji ima čak 13 cifara, je možda bolje izabrati tip podataka *text*. Neki tipovi podataka zahtevaju bliže definisanje podataka. Na primer tip podataka *text* zahteva da se specificira maksimalni broj karaktera, mada neki SUBP dozvoljavaju da se specificira da tekst može biti promenljive dužine.

Pored tipa podataka, ovde je potrebno definisati i ograničenja vrednosti podataka. Na primer, atribut BrojRadnihSatiDnevno se može ograničiti na vrednosti između 0 i 12, jer su takvi poslovni zahtevi. Ovakva ograničenja znatno smanjuju mogućnost unosa pogrešnih vrednosti atributa. Takođe treba definisati da li atribut može imati Null (nema vrednosti) vrednost, ili neku podrazumevajuću vrednost koja se automatski upisuje kao vrednost atributa prilikom kreiranja novog zapisa.

4. Specifikacija primarnih ključeva. Prilikom izrade logičkog modela primarni ključevi su definisani i treba ih preslikati u fizički model. U slučajevima kada to iz nekih razloga nije moguće treba izabrati surogat ključ.

5. Specifikacija redosleda kolona. Sa aspekta tačnosti izvršenja operacija redosled kolona u tabelama je irelevantan. Rezultati upita će biti isti bez obzira da li je neka kolona treća ili osma po redu. Međutim, redosled kolona može bitno da utiče na brzinu izvršavanja operacija, pa je u nekim slučajevima potrebno promeniti redosled koji nudi logički model podataka.

JOŠ DVA KORAKA U IMPLEMENTACIJI FIZIČKOG MODELA

Definisanje referencijalnih ograničenja i prostora tabela

6. Definisanje referencijalnih ograničenja za sve veze. Da bi se definisala referencijalna ograničenja potrebno je definisati primarni ključ u jednoj tabeli i strani ključ u zavisnoj tabeli. Referencijalna ograničenja povezuju primarni ključ za strani.Za svako referencijalno ograničenje je potrebno definisati set pravila koja određuju status kolone sa stranim ključem

prilikom insertovanja i ažuriranja, kao i status zavisne vrste kada se primarni ključ briše. Na primer, kada se briše primarni ključ koji referencira na postojeći strani ključ, ovo pravilo definiše da li će SUBP izbeći brisanje primarnog ključa ili će obrisati i primarni i strani ključ, ili će se kao vrednost stranog ključa upisati Null vrednost. Ovakvim pravilima se garantuje referencijalni integritet baze podataka.

7. Definisanje prostora tabela (tablespace). Iako su podaci relacionog modela fizički organizovani u tabele, odgovarajuće datoteke u kojima su smešteni podaci nisu jednostavno preslikane kolone i vrste. Tabele se preslikavaju u fizičke strukture koje se nazivaju prostor tabela (tablespace). Baza podataka se sastoji iz jednog ili više prostora tabela. Svaki prostor tabela može da sadrži jednu ili više tabela. Pored planiranja prostora tabela, potrebno je proceniti potreban fizički prostor na spoljnim memorijama potrebnim za čuvanje podataka i indeksa. Takođe treba definisati potrebu za kompresijom podataka - metod koji će se koristiti.

Poglavlje 6

Pokazna vežba

ORGANIZACIJA POKAZNIH VEŽBI

Način organizacije pokaznih vežbi

Vežba je organizovana kroz uvod deo i deo za samostalni rad studenata.

1. U uvodnom delu individualnih vežbi se daje pokazni primer koji studentima treba da pomogne u samostalnom rešavanju zadataka.
2. Zadatke koji su zadati za samostalni rad student samostalno rešava uz pomoć asistenta.

6.1 Pokazni primer

ER DIJAGRAM UNIVERZITETSKE BIBLIOTEKE - (10 MIN.)

Opis problema vezanog za rad univerzitetske biblioteke za koji streba napraviti ER dijagram

Neophodno je projektovati strukture podataka za podršku informacionom sistemu jedne savremene univerzitetske biblioteke.

Studenti, profesori i ostalo osoblje fakulteta imaju pravo da koriste usluge biblioteke. Svaki član biblioteke dobija člansku karticu, koja je smart kartica, odnosno kartica slična ATM kartici koju koriste korisnici usluga bankomata. Na kartici je pohranjen lični identifikacioni broj (PIN) člana.

Univerzitetska biblioteka poseduje pultove (uređaji) gde studenti mogu proveravati koje knjige su dostupne, a koje nisu i zadavati zahteve za izdavanje knjige. Student se na sistemu identifikuje tako što ubacuje svoju člansku karticu u čitač kartica, nakon čega se nudi forma za identifikaciju, odnosno za unos PIN koda. Nakon uspešne identifikacije članu biblioteke se nude izbor dostupnih opcija: pretraživanje naslova, podizanje knjiga, vraćanje knjiga u biblioteku.

Kada član pronađe željene knjige, odlazi u skladište knjiga (police u bilbioteci) uzima knjige i vraća se ispred pulta gde prinosi svaku knjigu bar kod čitaču. Sistem automatski beleži koje knjige su pozajmljene i korisnik prekida korisničku sesiju.

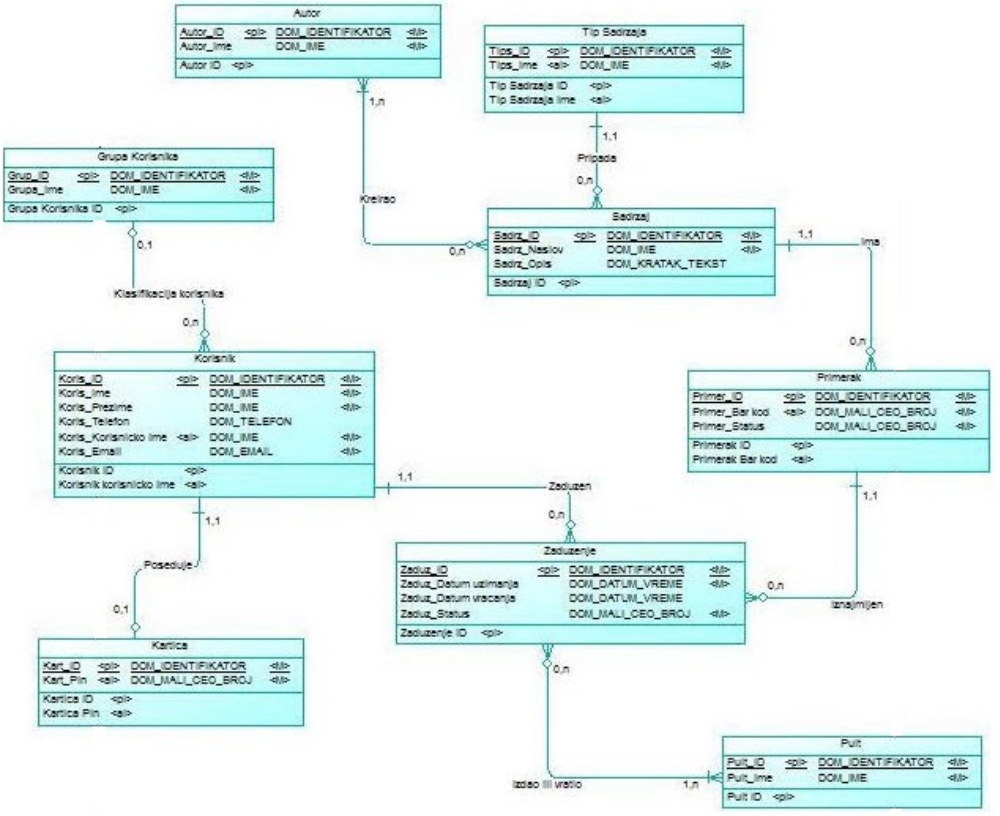
Član biblioteke vraća knjige u biblioteku tako što bira opciju za vraćanje knjiga, nakon čega prinosi svaku knjigu bar kod čitaču. Sistem automatski označava izdanja knjiga kao vraćena, a korisnik ostavlja knjige na pultu.

Zadatak:

1. Napraviti konceptualni model baze podataka biblioteke
2. Od konceptualnog modela generisati logički model baze podataka kako je to objašnjeno u lekciji 3 (grupna vežba koja opisuje upotrebu Power Designer-a)

ER DIJAGRAM UNIVERZITETSKE BIBLIOTEKE - REŠENJE - (10 MIN.)

U nastavku sledi rešenje prvog zadatka. Proverite svoje rešenje sa priloženim rešenjem



Slika 6.1.1 Konceptualni model baze podataka univerzitetske biblioteke predstavljen E/R dijagramom [Izvor: NM IT350-2020/2021.]

ER DIJAGRAM UNIVERZITETSKE BIBLIOTEKE - OPIS REŠENJA - (10 MIN.)

Opis rešenja

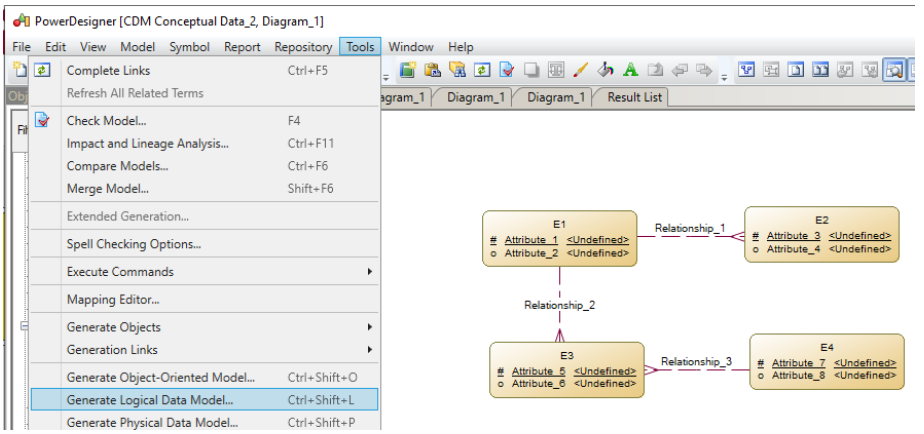
Jedno od mogućih rešenja predstavljeno je na prethodnoj slici.

1. Tip entiteta KORISNIK: služi za čuvanje podataka o korisnicima biblioteke koji su opisani svojim atributima.
2. Tip entiteta KARTICA: služi za čuvanje podataka o karticama koje se izdaju korisnicima biblioteke. Kako svaki korisnik ima samo jednu karticu (zbog čega je relacija između entiteta KORISNIK i KARTICA 1:1, ovaj tip entiteta treba izostaviti i umesto njega tipu entiteta korisnik dodati atribut kartica.
3. Tip entiteta GRUPA KORISNIKA označava grupu kojoj korisnik može pripadati, s tim što jednu grupu sačinjava više korisnika
4. Tip entiteta SADRZAJ označava bibliografsku jedinicu (knjiga, časopis) koji se nalazi u biblioteci i koju korisnik može iznajmiti. On se karakteriše svojim atributima naslov i opis
5. Tip entiteta AUTOR predstavlja autora bibliografske jedinice (sadržaja) koji se nalazi u biblioteci. Svaka bibliografska jedinica može imati više autora (prvi autor, drugi autor itd.) a svaki autor takođe može imati više sadržaja koji se nalaze u biblioteci.
6. Tip entiteta TIP SADRZAJA označava vrstu sadržaja kojem bibliografska jedinica pripada (knjiga, časopis, serija itd.)
7. Tip entiteta PRIMERAK služi da označi svaki pojedinačni primerak sadržaja (u biblioteci se može naći više primeraka iste knjige na primer). Ovaj tip entiteta ne označava ukupan broj primera već sadrži bar kod i status svakog pojedinačnog primerka kako bismo mogli da znamo koliko je primeraka nekog sadržaja prisutno u biblioteci a koliko je izdato korisnicima
8. Tip entiteta ZADUZENJE služi za čuvanje evidencije o tome koji je korisnik pozajmio koji sadržaj: Pamti se datuma kada je sadržaj iznajmljen, kada je vraćen i njegov status (izdat ili vraćen). Tip entiteta ZADUZENJE je nastao kao rezultat veze više-prema-više između tipova entiteta KORISNIK i SADRZAJ jer jedan korisnik može da se zaduži sa više sadržaja a jedan sadržaj može biti zadužen od strane više korisnika.
9. Tip entiteta PUT služi da se označi koji je put korisnik iznajmio određeni sadržaj. Ovaj tip entiteta nije neophodan.

GENERISANJE LOGIČKOG IZ KONCEPTUALNOG MODELA BAZE PODATAKA - (15 MIN.)

Način generisanje logičkog iz konceptualnog modela baze podataka u Power Designer-u

Prilikom rešavanja 2. zadatka treba primeniti princip generisanje logičkog iz konceptualnog modela baze podataka u CASE alatu Power Designer koji je prikazan na slici 7.2.

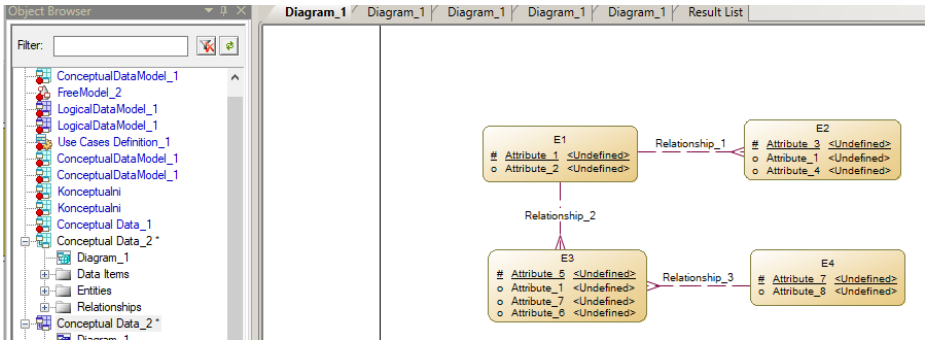


Slika 6.1.2 Generisanje logičkog iz konceptualnog modela baze podataka [Izvor: NM IT350-2020/2021.]

Kao što se slike može videti, najpre treba izabrati opciju Tool a zatim opciju Generate Logical data model.

U trenutku generisanja logičkog iz konceptualnog modela potrebno je imati specificirane sve primarne identifikatore tipova entiteta, u suprotnom prilikom generisanje logičkog modela javiće se greška. Razlozi za to su objašnjeni u predavanju.

Izgled logičkog modela podataka za primer konceptualnog modela podataka sa slike 7.1 prikazan je na slici 7.3.



Slika 6.1.3 Logički model podataka za konceptualni model podataka sa slike 2. [Izvor: NM IT350-2020/2021.]

6.2 Pokazna vežba - samostalni rad

1. ZADATAK ZA SAMOSTALNI RAD

Opis zadatka 1 - student bira jedan od zadatka 1 ili 2 - Potrebno vreme za izradu zadatka 25 minuta.

1. Projektovati konceptualnu šemu baze podataka na osnovu sledećih podataka i zahteva (koristeći Power Designer). Analizirati zahteve i podatke i na osnovu njih odrediti entitete, atribute, veze:
   - Kablovski operater uvodi novu mogućnost u okviru svoje ponude - Gledanje na zahtev. Na zahtev se mogu gledati sve emisije, filmovi, serije, ... koji su se prikazivali u prethodnom periodu.
   - Takođe, postoje i filmovi i serije koji uvek dostupni na serveru operatera
   - Svaki korisnik se mora registrovati za ovu opciju jer se ona naplaćuje dodatno.
   - Registracija se vrši za gledanje jedne emisije, a registracija važi 24 sata. Emisija se može gledani nekoliko puta u roku od 24 časa
   - Za filmove i serije registracija važi za jedno gledanje. Nakon gledanja određenog filma ili serije, isti se može pogledati tek za 48 časova.

2. Na osnovu konceptualne šeme, napraviti logičku šemu baze podataka i sačuvati je za sledeći čas kada iz logičkog modela treba napraviti fizički model baze podataka i na osnovu njega generisati .sql file za generisanje skripti.

Potrebno vreme za izradu zadatka 25 minuta.

2. ZADATAK ZA SAMOSTALNI RAD

Opis zadatka 2 - student bira jedan od zadatka 1 ili 2 - Potrebno vreme za izradu zadatka 25 minuta.

1. Projektovati konceptualnu šemu baze podataka na osnovu sledećih podataka i zahteva (koristeći Power Designer). Analizirati zahteve i podatke i na osnovu njih odrediti entitete, atribute, veze:
   - Studenti svake akademske godine upisuju godinu studija. Pri tom biraju neke od ponuđenih izbornih predmeta, tako da zbir njihovih bodova u svakom semestru bude barem 30. Predmeti moraju biti izabrani najkasnije do 15. oktobra.
   - Fakultet je organizovan u departmane. Svaki profesor/asistent/nastavnik je član tačno jednog departmana. Nastavnici iz istog departmana između sebe biraju šefa departmana. Da bi bolje upravljao ljudskim resursima, šef departmana može menjati plate svojih nastavnika. Šef departmana može biti samo profesor.
   - Jedna od zadataka departmana je da se brine o nastavi. Svake akademske godine departmana nudi nekoliko izbornih predmeta za studente i osigurava nastavnike koji će predavati te predmete. Pri tom svaki predmet ima samo jednog nastavnika.
   - Takođe, departman brine i o ispitima. U toku akademske godine organizuje se nekoliko ispitnih rokova, sa određenom satnicom.
   - Departman zakazuje ispite, određuje mesto održavanja ispita i o tome obaveštava profesore i studente. Svaki profesor je u obavezi da dežura na ispitu.
   - Na kraju akademske godine fakultet pohvaljuje studente koji su postigli najbolje ocene iz upisanih predmeta.

2. Na osnovu konceptualne šeme, napraviti logičku šemu baze podataka i sačuvati je za sledeći čas kada iz logičkog modela treba napraviti fizički model baze podataka i na osnovu njega generisati .sql file za generisanje skripti.

Potrebno vreme za izradu zadatka 25 minuta.

3. ZADATAK ZA SAMOSTALNI RAD

Opis zadatka 3 - svi studenti rade zadatak 3 - Potrebno vreme za izradu zadatka 20 minuta

1. Projektovati konceptualnu šemu baze podataka na osnovu sledećih podataka i zahteva (koristeći Power Designer). Analizirati zahteve i podatke i na osnovu njih odrediti entitete, atribute, veze:
   - Zdravstvena ustanova je odlučila da uvede elektronski sistem praćenja istorija bolesti svojih pacijenata, ali i da prati rad svoj doktora. (Elektronski zdravstveni karton).
   - Prema novom zakonu, svaki pacijent mora da ima svog izabranog lekara. Takođe, po tom zakonu, lekovi se izdaju za maksimalno 14 dana.
   - U okviru ustanove, postoji i apoteka koja izdaje prepisane likove i terapije i vodi računa o lekovima u apoteci.
   - Da bi se olakšao rad doktorima i smanjilo čekanje na pregled pacijentima, uvedeno je i zakazivanje pregleda online i telefonom.
   - Svaki doktor je u obavezi da nakon pregleda ubeleži podatke o obavljenom pregledu, kao i koja je terapija prepisana pacijentu. Ukoliko ima potrebe, doktor zakazuje i kontrolu pacijentu u određenom periodu ili zakazuje pregled kod specijaliste.
   - Apotekari ne mogu izdavati lekove bez recepta, tako da moraju da imaju uvid u podatke o terapiji odrađenog pacijenta. Ukoliko se neki od prepisanih lekova naplaćuje, apotekar izdaje račun pacijentu.
   - Da bi se olakšao rad lekara i apotekara, uvode se i jedinstveni šifarnici za bolesti kao i za lekove.
2. Na osnovu konceptualne šeme, napraviti logičku šemu baze podataka i sačuvati je za sledeći čas kada iz logičkog modela treba napraviti fizički model baze podataka i na osnovu njega generisati .sql file za generisanje skripti.

Potrebno vreme za izradu zadatka 20 minuta.

Poglavlje 7

Domaći zadatak

DOMAĆI ZADATAK 5 - VREME IZRADE 90 MIN.

Uradite sledeće zadatke:

Za konceptualni model baze podataka koji si imao zadatak da uradiš u prethodnom zadatku i koji si predstavio E/R dijagramom, treba uradi odgovarajući klasni dijagram tako što će se:

1. umesto tipova entiteta i njihovih atributa koristiti odgovarajuće klase
2. umesto relacija između tipova entiteta asocijacije za koje treba odrediti odgovarajuću višestrukost.

Obavezno je korišćenje specijalnih tipova asocijacija kao što su agregacija i kompozicija a generalizaciju i specijalizaciju koristiti samo ako je to potrebno.

Rešenje zadatka snimite u dokument IT250-DZ04-BrIndeksa-Ime_Prezime.cdm, (gde su Ime, Prezime i brIndexa vaši podaci a ekstenzija file .cdm se dobija kao izlaz i konceptualnog modeliranje u PowerDesigneru).
Prilikom slanja domaćih zadatka, neophodno je da ispunite sledeće:
- Subject mail-a mora biti IT250-DZbr (u slučaju kada šaljete domaći za ovu nedelju to je IT250-DZ05)
- U prilogu mail-a treba da se nalazi arhiviran projekat koji se ocenjuje imenovan na sledeći način: IT250-DZbr-BrojIndeksa-Ime Prezime. Na primer, IT250-DZ05-1234-VeljkoGrkovic
- Telo mail-a treba da ima pozdravnu poruku
- Arhivu sa zadatkom poslati na adresu predmetnog asistenta:
milica.vlajkovic@metropolitan.ac.rs (studenti u Beogradu i online studenti) ili
tamara.vukadinovic@metropolitan.ac.rs (studenti u Nišu).
Svi poslati mail-ovi koji ne ispunjavaju navedene uslove napomene biti pregledavani. Za sva pitanja ili nedoumice u vezi zadatka, možete se obratiti asistentu

ZAKLJUČAK

Šta smo naučili u ovoj lekciji?

Dobro modeliranje konceptualnog modela je ključno za uspešan dizajn i implementaciju baze podataka. Evo nekoliko principa koji bi trebalo da se uzmu u obzir prilikom kreiranja konceptualnog modela:
Jednostavnost: Konceptualni model treba da bude jednostavan i razumljiv kako bi ga mogli pratiti svi zainteresovani članovi tima, uključujući klijente i programere. Izbegavajte prekomernu složenost koja bi mogla dovesti do konfuzije.
Jasnoća: Definišite entitete, njihove atribute i odnose između njih jasno i precizno. Svaki entitet i atribut treba da ima jasno značenje kako bi se izbegle nesporazumi kasnije tokom razvoja.

Skalabilnost: Model treba da bude skalabilan kako bi se mogao prilagoditi potrebama i zahtevima koji mogu nastati u budućnosti. Razmislite o mogućnosti dodavanja novih entiteta i atributa kako bi se osigurala dugoročna funkcionalnost baze podataka.

Nezavisnost od tehnologije: Konceptualni model treba da bude nezavisan od tehnologije koja će se koristiti za implementaciju. Fokusirajte se na suštinske informacije o podacima, a ne na tehničke detalje poput baze podataka ili programskog jezika.

Konzistentnost: Vodite računa da bude doslednosti u imenovanju entiteta i atributa. Ovo olakšava razumevanje modela i omogućava lakšu implementaciju.

Pravilni odnosi: Pažljivo definišite odnose između entiteta, kao što su jedan-na-jedan, jedan-na-više ili više-na-više. Razumeti ove odnose pomaže u pravilnom oblikovanju baze podataka i održava integritet podataka.

Održivost: Model treba da bude održiv tokom vremena i ne bi trebalo da se menja često, osim ako za tim postoje opravdani razlozi. Prečesta promena modela može uzrokovati probleme u razvoju i održavanju sistema.

Uključenost interesnih strana: Uključite sve ključne interesne strane, uključujući klijente, programere i analitičare podataka, kako biste osigurali da model odražava sve potrebne informacije i funkcionalnosti.

Dokumentacija: Redovno dokumentujte model kako biste zadržali jasnoću i olakšali timski rad. Dobra dokumentacija pomaže u rešavanju nesporazuma i brzom uvođenju novih članova u tim.
Govorili smo kako se tipovi entiteta, njihovi atributi i relacije mogu predstaviti u slučajevima kada umesto E/R dijagrama koristimo UML klasne dijagrame, što je čest slučaj obzirom da se UML danas koristi za projektovanje softvera primenom objektno orijentisanog pristupa.
Na kraju su dati primeri transformacije konceptualnog modela podataka najpre u logički a

zatim i u fizički model baze podataka na osnovu koga se mogu generisati .sql skripte za kreiranje tabela baze podataka.

LITERATURA

Za pisanje ove lekcije je korišćena sledeća literatura:

1. http://corpgov.crew.ee/Materjalid/Database%20Systems%20-%20Design,%20Implementation,%20and%20Management%20(9th%20
2. Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom, DATABASE SYSTEMS –The Department of Computer Science, Stanford University, 2009 by Pearson Education Inc.
3. David M. Kroenke, Database Processing – fundamentals, design and implementation, Prentice Hall, 2004.
4. C. J. Date, An introduction to Database Systems, Addison-Wesley Publishing Company, 1990
5. https://www.visual-paradigm.com/support/documents/vpuserguide/3563/3564/85378_conceptual,l.html
6. https://airbrake.io/blog/sdlc/conceptual-model