Projektovanje baza podataka

1. april 2026.

Sadržaj

1. Proces projektovanja baza podataka
2. Konceptualno modeliranje
   - Analiza zahteva
   - Modeliranje podataka i odnosa
   - Integrisanje pogleda

Napomena

Slajdovi su nastali obradom slajdova prof. dr Saše Malkova za predmet Projektovanje baza podataka

Projektovanje baza podataka

Konceptualno modeliranje

1. april 2026.

Sadržaj

1. Proces projektovanja baza podataka
2. Konceptualno modeliranje

Mesto projektovanja baza podataka

- Projektovanje baze podataka se odvija u okviru šireg procesa razvoja softvera ili informacionog sistema
- Da bi moglo da se pristupi projektovanju baze podataka pretpostavlja se da je već obavljeno:
  - istraživanje sistema
    - t.j. prikupljanje informacija o domenu čiju sliku će da predstavlja baza podataka koja se projektuje
  - analiza sistema
    - t.j. analiza prikupljenih informacija i formulisanje zahteva
- Već postoje potrebne informacije i definisani zahtevi
  - ali često nešto nedostaje, pa neki elementi istraživanja i analize moraju da se ponove ili dopune u toku projektovanja

Proces projektovanja baza podataka

- Baze podataka se projektuju po nivoima
- Nivoi projektovanja delimično prate nivoe arhitekture
- Granice nivoa projektovanja se ne poklapaju tačno sa granicama arhitekture
- Takvo nepodudaranje je prirodno:
  - Neki nivoi projektovanja su apstraktni, a neki su konkretni
  - Svi nivoi arhitekture su konkretni – postoje u implementaciji

Osnovne celine pri projektovanju

- **Konceptualno projektovanje**
  - odgovara apstrakciji spoljašnje sheme i dela konceptualne sheme
  - najpre se pravi po apstraktan **pojedinačni konceptualni model** za svaku od spoljašnjih shema
  - zatim se od njih pravi jedinstven apstraktan **objedinjeni konceptualni model**
- **Logičko projektovanje**
  - uglavnom odgovara konceptualnoj shemi
  - prilagođavanje konceptualnog modela konkretnom modelu podataka
  - pravi se **logički model** na jeziku modela izabranog SUBP
- **Fizičko projektovanje**
  - pravi se **fizički (implementacioni) model**
  - na kraju se implementiraju i spoljašnje sheme
- **Projektovanje bezbednosti**
  - Uglavnom (ali ne sasvim) ortogonalno u odnosu na ostale korake
  - Prepliće se sa njima, ali se u najvećoj meri odvija u toku i nakon fizičkog projektovanja

Osnovni koraci pri projektovanju

- **Konceptualno projektovanje**
  1. Analiza zahteva
  2. Konceptualno projektovanje pojedinačnih spoljašnjih shema
     - naziva se i *Projektovanje pogleda*
  3. Pravljenje objedinjenog konceptualnog modela
     - naziva se i *Objedinjavanje pogleda*
  4. Grupisanje entiteta
- **Logičko projektovanje**
  5. Prevođenje konceptualnog modela u logički model
  6. Prečišćavanje sheme
- **Fizičko projektovanje**
  7. Fizičko projektovanje implementacije
  8. Fizičko projektovanje spoljašnjih shema
- **Projektovanje bezbednosti**
  9. Projektovanje bezbednosti

* Prvih 8 koraka se izvode redom, a 9. se izvodi paralelno sa njima.

1 - Analiza zahteva

- Cilj:
  - Razumevanje domena
    - elementi domena i odnosi među njima
    - njihovi atributi (podaci)
    - okvirna procena obima podataka i odnosa među podacima
  - Razumevanje aplikacija
    - potrebe za podacima
    - učestalost upita i transakcija
    - performanse

2 - Konceptualno projektovanje pojedinačnih shema

- Cilj:
  - Pravljenje modela domena za svaku spoljašnju shemu posebno
    - naziva se i *modeli podataka visokog nivoa*
  - Opisivanje
    - strukture domena
    - odnosa među elementima domena
    - uslova integriteta
- Sve se opisuje iz ugla korisnika
  - Veoma je važno da se opiše tačna semantika podataka i odnosa
  - Smisao konceptualnog modela je u vernom predstavljanju semantike

3 - Objedinjavanje pogleda

- Cilj:
  - Pravljenje objedinjenog apstraktnog modela domena
  - Ujednačavanje rečnika i opisa domena
    - Dodatno apstrahovanje radi objedinjavanja
  - Integrisanje pogleda u jedinstveni model

4 - Grupisanje entiteta

- Cilj:
  - Pravljenje većeg broja manjih i preglednijih dijagrama modela
  - Grupisanje jače povezanih entiteta u celine
  - Uočavanje centralnih entiteta za odgovarajuće grupe
  - Ovaj korak nije suštinski (tj. ne pravi novi model), ali može da bude značajan za lakše razumevanje složenih modela

5 – Prevođenje konceptualnog u logički model

- Cilj:
  - Svi elementi modela se prevode na jezik logičkog modela
    - model domena se prevodi u model baze podataka
  - Inicijalno pravljenje logičkog modela
  - Osnovno prilagođavanje konceptualnog modela konkretnom implementacionom modelu podataka

6 - Prečišćavanje sheme

- Cilj:
  - Dosledno prilagođavanje modela baze podataka jeziku i pravilima konkretnog modela podataka
  - Prepoznaju se i rešavaju se potencijalni problemi
  - U slučaju relacionog modela osnovni cilj je eliminacija redundantnosti

7 - Fizičko projektovanje implementacije

- Cilj:
  - Modeliranje načina implementiranja baze podataka
    - naziva se i *model podataka*
  - Optimizovanje logičkog modela prema specifičnostima primenjenog SUBP i prema očekivanom modelu upotrebe
  - Određivanje tačne interne sheme baze podataka
    - parametri implementacije tabela baze podataka
    - pomoćne strukture (indeksi, baferi,...)
    - parametri funkcionisanja SUBP
    - projektovanje distribuiranja
    - ...
  - Određivanje mehanizama preslikavanja fizičke u konceptualnu shemu

8 – Fizičko projektovanje spoljašnjih shema

- Cilj:
  - Projektovanje implementacije spoljašnjih shema
  - Projektovanje aplikativnih interfejsa za pristupanje podacima
  - Prilagođavanje interne sheme specifičnostima primenjenih spoljašnjih alata

9 - Projektovanje bezbednosti

- Cilj:
  - Prepoznavanje vrste i uloge svakog korisnika
  - Prepoznavanje vrste aplikacije
  - Definisanje korisničkih grupa i odgovarajućih minimalnih skupova privilegija za sve različite uloge i vrste korisnika i aplikacija
  - Prepoznavanje delova baze podataka koji su posebno osetljivi i kojima potrebno dodatno redukovati pristup
  - Definisanje i implementiranje odgovarajućih bezbednosnih mehanizama

Odnos projektovanja i arhitekture

- Pri projektovanju se dva puta prolazi kroz nivoe arhitekture
  - prvo naniže
    - konceptualno pa logičko projektovanje prvo razmatraju najapstraktnije elemente arhitekture
  - zatim naviše
    - fizičko projektovanje definiše implementacije svih shema, odozdo naviše – najpre fizičke, pa konceptualne pa spoljašnjih shema

Odnos projektovanja i arhitekture

Konceptualno projektovanje

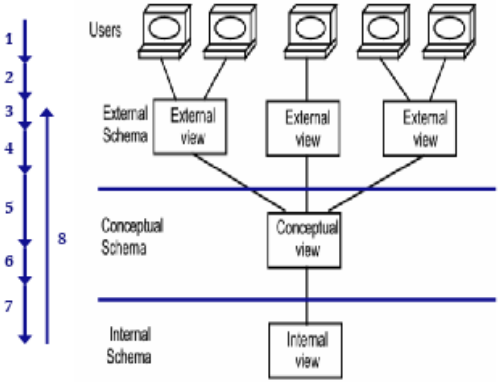
1. Analiza zahteva
2. Konceptualno projektovanje pogleda
3. Objedinjavanje konceptualnog modela
4. Grupisanje entiteta

Logičko projektovanje

5. Prevođenje konceptualnog modela u logički
6. Prečišćavanje sheme

Fizičko projektovanje

8. Optimizovanje logičkog modela i projektovanje implementacije prema modelu upotrebe
9. Implementacija spoljašnje sheme



Opšti i specifični elementi projektovanja

- Konceptualno projektovanje
  - odnosi se na domen i na aplikacije
  - odvija se uglavnom uopšteno, bez mnogo obraćanja pažnje na vrstu i implementaciju SUBP
- Logičko projektovanje
  - odnosi se na konkretan model podataka izabranog SUBP
  - uzimaju se u obzir i neke specifičnosti konkretne implementacije SUBP
- Fizičko projektovanje
  - odnosi se na konkretan SUBP
  - neki elementi su uopšteni za vrstu (model) baza podataka
  - većina elemenata se više ili manje prilagođava konkretnoj implementaciji SUBP

Projektovanje relacionih baza podataka

- U skladu sa prethodno predstavljenim konceptima
- Specifičnosti su posledica osobina relacionog modela
- Logičko projektovanje ima za cilj relacioni model
  - konceptualni model se prevodi na jezik relacionog modela
  - prečišćavanje sheme se odnosi prvenstveno na eliminaciju redundantnosti
- Fizičko projektovanje
  - interna shema zavisi od impl. RSUBP, ali ima nekih deljenih koncepata
  - definisanje konceptualne sheme pomoću pogleda (i okidača)
  - definisanje spoljašnjih shema se ostvaruje na isti način

Sadržaj

1 Proces projektovanja baza podataka

2 Konceptualno modeliranje

Konceptualni model

- Opisuje prepoznatu semantiku posmatranog domena
- Posebno važan
  - za razumevanje celine sistema
  - za razmenu informacija sa implementatorima aplikacija
  - za razmenu informacija sa klijentima
- Model se obično oblikuje i zapisuje primenom
  - ER modeliranja
  - UML modeliranja

Osnovne karakteristike

- Ključne osobine konceptualnog modela su
  - Iscrpno opisana semantika
    - od suštinske važnosti
    - mora da bude jasno predstavljena uloga svakog elementa u domenu - entiteti, odnosi, atributi
  - Apstraktnost
    - cilj je lakše i bolje razumevanje semantike celine i delova modela
    - smanjuje se nivo detaljnosti
    - izostavljaju se suvišne informacije - bez nebitnih ili manje bitnih atributa, bez pomoćnih tabela neophodnih za fizičku implementaciju
    - uopštava se predstavljanje odnosa

Koraci

- Pravljenje konceptualnog modela počiva na sledećim koracima:
  - Analiza zahteva
  - Konceptualno modeliranje podataka i odnosa
  - Integrisanje pogleda (lokalnih shema)
  - Grupisanje entiteta

Analiza zahteva

- Analiza zahteva
  - Veoma važna!
  - Veoma zahtevna!
- Tesno povezana za procesom analize zahteva u kontekstu opšteg projektovanja softvera ili informacionog sistema
- Da bi se razumele potrebe za podacima često moraju da se detaljno razumeju svi procesi koji se odvijaju u sistemu

Ciljevi analize zahteva

- Osnovni ciljevi:
  - Prevođenje funkcionalnih zahteva u kontekst trajnih podataka
  - Razumevanje i opisivanje podataka i njihove uloge
    - elemenata domena i njihovih odnosa, tj. struktura odgovarajućih potrebnih podataka i njihovih odnosa, uočava se koji atributi elemenata domena imaju značaja za modeliranje
    - svih pojedinačnih transakcija koje se izvršavaju u domenu - nad elementima domena, odnosno nad podacima o njima
  - Definisanje nefunkcionalnih zahteva
    - integritet
    - performanse
    - bezbednost
    - i razni administrativni aspekti
  - ...

Ciljevi analize zahteva

- Osnovni ciljevi:
  - ...
  - Određivanje i oblikovanje dodatnih uslova implementacije
    - tehnologije
    - hardverske i softverske pretpostavke
    - aplikativni i korisnički interfejsi
    - i drugo
  - Izrada iscrpne dokumentacije za sve navedeno

Ciljevi analize zahteva

- Jedan od najvažnijih ciljeva analize zahteva, u okviru konceptualnog projektovanja, je **dekompozicija sistema na skup slabije povezanih delova / segmenata**
  - Ideja je da se projektovanje složenog sistema podeli na manje celine, koje se lakše izučavaju i modeliraju
    - t.j. na veći broj manjih konceptualnih modela
  - Kasnije, u fazi grupisanja, ti delovi će se spojiti u celinu

Ciljevi analize zahteva

- Dekompozicija obično počinje od prirodnih razlika – različitih uočenih spoljašnjih shema, tj. vrsta korisnika
  - na primer, u slučaju BP fakulteta, možemo da uočimo studente, nastavnike, studentsku službu, finansijsku službu,...
- Nastavlja se, ako je potrebno, uočavanjem manjih funkcionalnih celina i daljim dekomponovanjem
  - uočavamo poslove koji su izdvojeni od ostatka sistema
    - na primer, organizacija upisnih rokova
  - u okviru jedne spoljašnje sheme uočavamo različite poslove
    - npr. poslovi studenta kao što su prijavljivanje ispita, upis godine, ...

Poslovi u okviru analize zahteva

- Moraju da se obuhvate svi bitni aspekti analize sistema
  - Analiza zahteva obuhvata detaljno izučavanje i modeliranje sistema
  - To može da obuhvati (i često je neophodno) i modeliranje poslovnih i implementacionih aspekata sistema
  - Ova faza se prepliće sa projektovanjem IS i ne može da se iz njega izdvoji, a da pri tome ostane potpuna

Konceptualno modeliranje domena / podataka

- Cilj konceptualnog modeliranja je pravljenje verne slike domena
- Zato se naziva i **modeliranje domena**
- Modeliraju se samo oni elementi domena koji imaju značaja za ciljnu bazu podataka
- Zato se konkretan korak opisivanja elemenata domena naziva i **konceptualno modeliranje podataka**

Konceptualno modeliranje podataka

- U ovom koraku se bavimo konceptualnim modeliranjem jednog dela sistema – tj. jedne spoljašnje sheme
  - uglavnom na sličan način kao što bi se modelirao ceo kada ne bi bio dekomponovan
- Glavni koraci
  - Klasifikacija skupova podataka
    - prepoznavanje entiteta / klasa i atributa
  - Prepoznavanje odnosa
    - asocijacija i složeniji odnosi
    - generalizacija / specijalizacija, hijerarhije

Klasifikacija skupova podataka

- Svaki objekat/predmet u domenu je kandidat za entitet ili atribut
  - tj. svaka imenica u tekstualnom opisu domena ili prikupljenim zahtevima je kandidat za entitet ili atribut
  - nije uvek očigledno šta je entitet (ili klasa) a šta atribut
- Neke smernice
  - Entiteti bi trebalo da sadrže opisne informacije
    - Ako nešto ima višestruku ili složenu vrednost (a ne skalarnu) onda je verovatno entitet
  - Atributi bi trebalo da pripadaju onim entitetima koje najneposrednije opisuju
    - Ako je nešto opisano jednom skalarnom vrednošću, onda je verovatno u pitanju atribut

Sadržaj entiteta

- Entiteti sadrže opisne informacije
  - ako imamo opisne informacije o nekom podatku, onda trebalo da bude entitet
    - *predmet* ima *naziv* i *broj ESPB*, pa bi trebalo da bude entitet
  - ako neki podatak ima samo identifikator (koji može da bude opisnog karaktera ali nema dodatnih opisa) onda bi to trebalo da bude atribut
    - *ocena* se identifikuje svojom vrednošću i nisu potrebni dodatni opisi, pa je to atribut

Sadržaj entiteta

- Primer: *grad*
  - ako nam je dovoljan naziv grada, onda može da bude atribut
  - ali ako su nam važne i neke druge informacije o gradu (na primer lokacija grada ili broj stanovnika), onda je to entitet

Višestruke vrednosti

- Ako nešto može da ima više vrednosti, složenu vrednost, vrednost koja je podložna promenama, onda je to entitet
  - to nekada ne mora da bude neophodno, ali je jednostavnije za analizu razmatranje, a kasnije može da se optimizuje ako je potrebno
- Primer:
  - ako *udruženje studenata* ima samo naziv, onda je to atribut
  - ali ako jedan student može da bude član više udruženja, onda je možda bolje da se udruženje proglasi za entitet
  - takođe, pošto više studenata može da bude u istom udruženju, eventualna promena naziva udruženja može da bude problematična ako je u pitanju samo atribut...

Dodeljivanje atributa entitetima

- Atributi se dodeljuju onim entitetima koje najneposrednije opisuju
  - To nekada nije sasvim očigledno
  - Obično može da se ustanovi iz funkcionalnih zavisnosti
    - ako se pogreši, obično može da se popravi u fazi logičkog projektovanja
  - Nekada odgovarajuća zavisnost postoji samo u domenu ali ne i u podacima
    - tj. vrednost atributa se u načelu određuje na osnovu entiteta, ali ne može da se ustanovi funkcionalna zavisnost
    - u takvim slučajevima se obično radi o (potencijalno prikrivenim) tranzitivnim zavisnostima

Dodeljivanje atributa entitetima

- Na primer:
  - ako za neki atribut A procenimo da pripada entitetu E1
  - a entitet E1 je u odnosu sa entitetima E2 i E3
  - možda je atributu A mesto u nekom od entiteta E2 i E3?

Dodeljivanje atributa entitetima

- Konkretan primer:
  - pretpostavimo da atribut *Ime* pripada entitetu *Student*
  - ako student upiše dva st. programa (bilo u isto vreme ili jedan posle drugog), onda će se njegovi lični podaci ponavljati, uključujući i *Ime*
  - možemo da prepoznamo skrivenu zavisnost Student -> Osoba -> Ime
  - može da ima smisla izdvajanje novog entiteta *Osoba*, tako da više *Studenata* može da se odnosi na jednu istu *Osobu*
  - *** u praksi se ovo često ne radi, zato što ime (i drugi lični podaci) mogu da se promene između upisivanja različitih programa, pa bi moglo da bude problema u vezi sa zvaničnim dokumentima
    - različite reference na fizički istu osobu se povezuju na neki drugi način

Prepoznavanje odnosa

- Za svaki odnos je potrebno prepoznati
  - učesnike
    - a sa njima i red odnosa (tj. broj učesnika)
  - kardinalnost
    - broj takvih odnosa jednog entiteta
    - opcionost / obaveznost
  - dodatne atribute koji opisuju odnos
  - jasan naziv i semantiku odnosa u domenu
    - kao i odgovarajuće funkcionalne zavisnosti

Prepoznavanje generalizacije

- Generalizacija = Svaki *poseban* slučaj *je* istovremeno i *opšti* slučaj
  - npr. svaki kvadrat je i pravougaonik
- Ustanovljavanje smera generalizacija nije uvek očigledno
  - U slučaju OO dizajna osnovni kriterijum je ponašanje:
    - kvadrat je specijalan slučaj pravougaonika
  - Kod baza podataka smo često u iskušenju da ignorišemo ponašanje i posmatramo samo strukturu:
    - struktura pravougaonika je dopuna (tj. specijalan slučaj) strukture kvadrata – imamo jedan podatak više

Prepoznavanje generalizacije

- Pri građenju hijerarhija ipak ne sme da se potpuno zapostavi ponašanje
  - Fokusiranje samo na strukturu *može da prođe* samo u slučaju plitkih hijerarhija
  - U slučaju iole dubljih hijerarhija eventualno donesene pogrešne odluke mogu da imaju neugodne posledice

Prepoznavanje generalizacije

- Što apstraktnije – to bolje:
  - Atribute je potrebno *popeti* što više uz hijerarhiju
    - Ključne atribute je potrebno vezati za baznu klasu (entitet)
  - I odnose je potrebno *popeti* što više uz hijerarhiju

Redundantnost odnosa

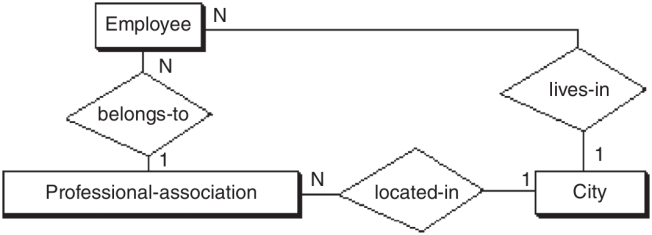
- Redundantni odnosi
  - imaju za rezultat teško otklonjivu redundantnost u modelu implementaciji
  - ometaju konzistentnost i normalizaciju
- Potrebno (poželjno) je eliminisati redundantne odnose

Eliminisanje redundantnih odnosa

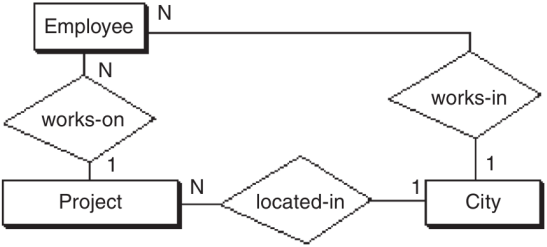
- Na nivou konceptualnog modela redundantnost ne mora uvek da se otklanja, ali mora da bude dokumentovana
  - Ako redundantnost olakšava razumevanje konceptualnog modela, onda može da nam bude od koristi da ostavimo redundantan odnos, ako ga pri tome jasno dokumentujemo
  - Ako nije dokumentovana, može da prođe neprimećeno pri pravljenju logičkog modela, što ne bi smelo da se desi
- OPREZ!
  - *isti* odnosi mogu biti redundantni i neredundantni
  - redundantnost zavisi od semantike i domena

Redundantni odnosi, primer, ER

Neredudantni odnosi



Redudantni odnosi



Odnosi višeg reda

- Odnos je **višeg reda** ako obuhvata više od dva učesnika
- OPREZNO sa odnosima višeg reda
  - uvek je potrebno da se dodatno razmisli da li su neophodni
  - često je bolje (i semantički i tehnički) da se odnos višeg reda podeli na više jednostavnijih odnosa
- Uputstvo:
  - Uvek pokušati sa više binarnih odnosa
  - Ako se ispostavi da je nemoguće, tek onda koristiti odnose reda

Odnosi višeg reda, primer

- Na primer:
  - U rasporedu časova, složen odnos je *nastavnik A drži čas iz predmeta B u učionici V u terminu G*
  - Međutim, odnos nastavnika i predmeta je važan i nezavisno od rasporeda i može da se izdvoji u koncept grupe
    - *grupa D = nastavnik A drži predmet B* može da ima i dodatne atribute, npr. oznaku; može više nastavnika da drži različite grupe za isti predmet ; može da bude više grupa istog nastavnika i predmeta; može da ima i odnos sa studentima koji čine grupu
    - početni odnos se onda pojednostavljuje (iako i dalje složen) - *grupa D ima čas u učionici V u terminu G*

Razdvojeni pogledi

- Konceptualno modeliranje se često odvija po delovima
  - odvojeno za različite aplikacije
  - ili delove aplikacije
- Osnovna motivacija:
  - analiza i modeliranje manjih delova domena se izvode jednostavnije i efikasnije
  - manji modeli se lakše oblikuju i razumeju
- Svaki pojedinačan model se naziva **pogledom** ili **lokalnom shemom**
  - nema neposrednog dodira sa pogledima u jeziku SQL, mada oni kasnije mogu da se koriste za implementaciju

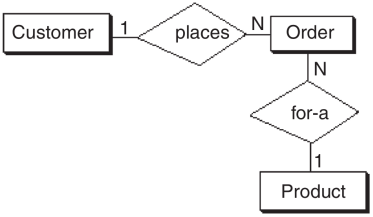
Razdvojeni pogledi, primer

- IS fakulteta
  - iz ugla studenta
    - potrebno je da se zna da je neki nastavnik zadužen za neki predmet
    - nije bitno kada je ko izabran u koje zvanje, pa ni u kom je zvanju
  - iz ugla kadrovske službe
    - nije bitno ko je zadužen za koji predmet
    - važno je kada je ko izabran u koje zvanje, kada je bio na odsustvima, koji je period izbora i drugo

Integrisanje pogleda (lokalnih shema)

- Opšta (globalna) konceptualna shema se dobija integrisanjem lokalnih shema (pogleda)
- To je nekada jednostavno ali može da bude i veoma složeno
- Zato se *integrisanje pogleda* obično opisuje kao poseban korak
  - sastavni deo konceptualnog modeliranja
- VAŽNO: Termini *lokalna shema* i *globalna shema* u kontekstu konceptualnog projektovanja imaju drugačiji smisao nego u kontekstu arhitekture distribuiranih baza podataka

Primer 2, različiti modeli za isti domen



(a)

(b)

(c)

Koraci pri integrisanju pogleda

- Osnovni koraci:
  - Poređenje shema i prepoznavanje konflikata
  - Preuređivanje shema i razrešavanje konflikata
  - Spajanje i restrukturiranje shema

Prepoznavanje konflikata

- Vrste konflikata
  - konflikti imena
  - strukturni konflikti
  - konflikti ključeva
  - konflikti zavisnosti

Konflikti imena

- Neujednačeno imenovanje entiteta i odnosa
  - rečnici entiteta i odnosa moraju biti ujednačeni
- Sinonimi
  - različiti nazivi za iste koncepte
  - prvi i najlakše rešiv problem
  - Primer: *Product* i *Item*
- Homonimi
  - isti naziv se koristi za različite koncepte
  - može da predstavlja ozbiljan problem ako je različito tumačena specifikacija zahteva
  - teže rešivo od sinonima i zato ima prioritet pri rešavanju
  - često je povezan sa strukturnim konfliktima
  - Primer: *product-number* može značiti: *model proizvoda* ili *serijski broj*

Konflikti imena

Primer sinonima

- Entitet koji predstavlja prijavu i polaganje ispita se u jednom pogledu *Polaganje*, a u drugom *Ispit*

Primer homonima

- Imenom *Ispit* se naziva
  - u jednom pogledu entitet koji predstavlja položen ispit
  - u drugom pogledu entitet koji predstavlja prijavu i polaganje ispita

Strukturni konflikti

- Različiti strukturni elementi se upotrebljavaju za modeliranje istih koncepata
  - (primer sa narudžbenicama)
- Da bi pogledi mogli da se spoje, odgovarajući koncepti moraju da imaju identične strukture

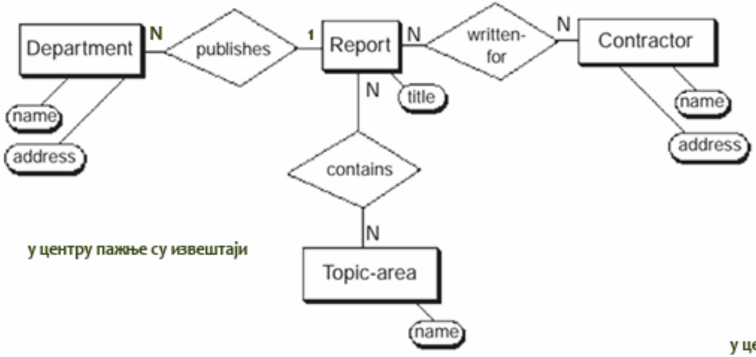
Konflikti ključeva

- Istom entitetu su u različitim pogledima dodeljeni različiti ključevi
- Neophodno je da se ključevi ujednače
- U suprotnom preti opasnost od redundantnih ključeva

Konflikti zavisnosti

- Isti odnosi se u različitim pogledima modeliraju na različite načine
  - različite vrste odnosa (slabi i jaki entitetski odnosi, agregacija, asocijacija i slično)
  - različite kardinalnosti
  - mogu da budu i suštinski različiti odnosi
- Rešavaju se ujednačavanjem karakteristika odnosa
  - u nekim slučajevima i dodavanjem novih odnosa ili čak menjanjem strukture

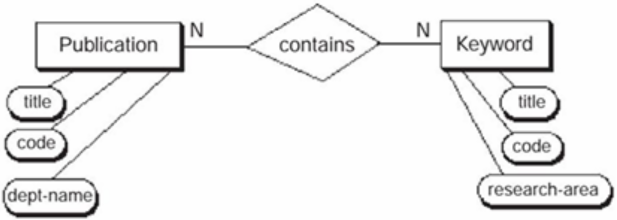
Primer, veći konceptualni konflikti



у центру пажње су извештаји

у центру пажње су публикације

Разлика је наизглед велика,
као да су у питању
потпуно различити домени



Razrešavanje konflikata

- Može da zahteva dodatnu ili ponovljenu analizu zahteva
- Može da zahteva ozbiljno modifikovanje pogleda
- Dobro je da učestvuju projektanti konfliktnih pogleda
- Tehnike razrešavanja konflikata su iste kao i za pravljenje konceptualnog modela

Razrešavanje konflikata

- U primeru se u dva pogleda koriste različiti termini za istu stvar (konflikt sinonima):
  - *Topic-area* i *Keyword*
  - štaviše, imaju i različite atribute (konflikt strukture)
- Konflikt se može rešiti tako što se:
  - u oba modela koristi naziv Topic-area
  - zadrže nazivi atributa iz modela publikacija

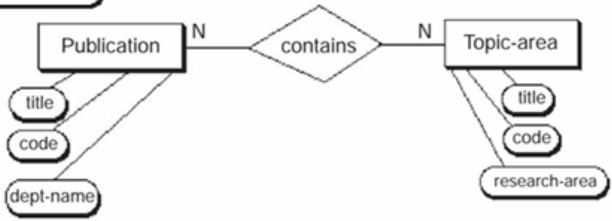
Primer, veći konceptualni konflikt

у центру пажње су извештаји

1. Разрешавање синонима

2. Разрешавање синонима и уједначавање скупа атрибута

у центру пажње су публикације



Razrešavanje konflikata

- Dalje, vidimo da se u publikaciji departman pominje samo atribut *dept-name* a u izveštajima predstavlja entitet
- To je strukturni konflikt – potrebno je da se uvede odgovarajući entitet koji je u odnosu sa publikacijom

Primer, veći konceptualni konflikt

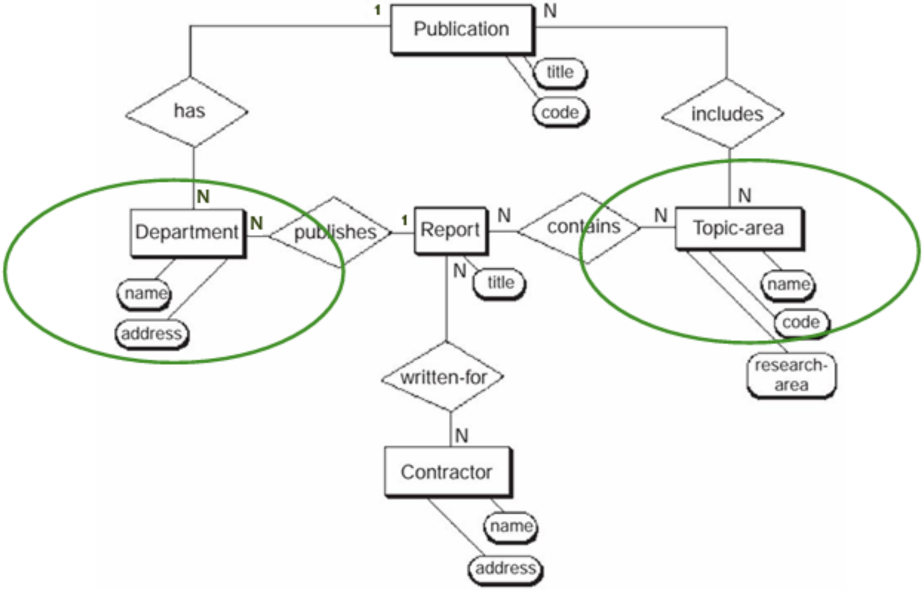
Spajanje i restrukturiranje shema

- Nastupa nakon usaglašavanja konflikata
- Rukovodi se principima
  - **potpunosti** – svi koncepti iz lokalnih shema moraju očuvani i u potpunosti prenesu u globalnu shemu
  - **minimalnosti** – svi redundantni koncepti (i entiteti odnosi) moraju da se uklone iz globalne sheme
  - **razumljivosti** – globalna shema mora da bude razumljiva svim korisnicima

Princip potpunosti

- Nakon razrešavanja konflikata najpre spajamo sve odgovarajuće koncepte i ostvarujemo potpunost

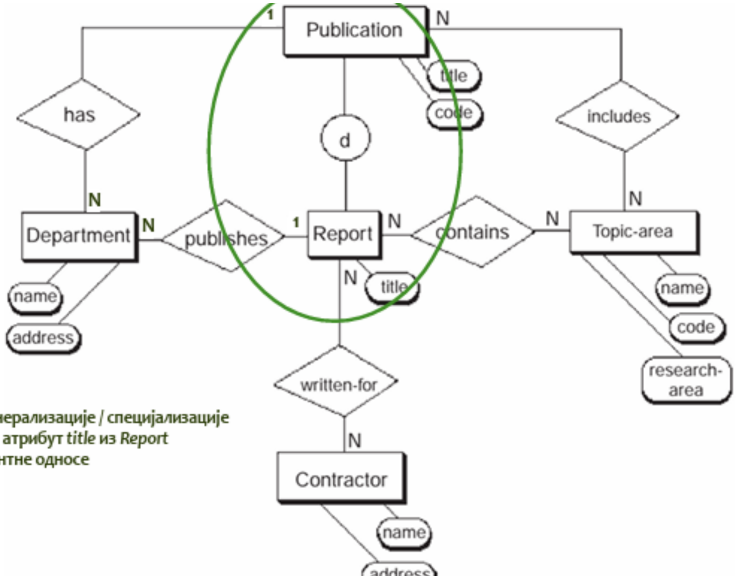
Primer spajanja shema – potpuno, ali ne i minimalno



Minimalnost

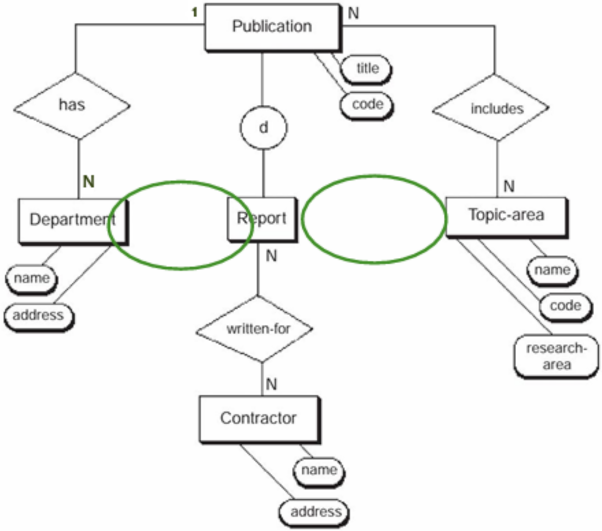
- Prepoznajemo redundantne koncepte
- Pokušavamo da ih razrešimo
  - uvođenjem generalizacije
  - uklanjanjem redundantnih odnosa

Primer spajanja shema – uvodimo generalizaciju između publikacije i izveštaja



1. Уводимо однос генерализације / специјализације
2. Бришемо сувишан атрибут *title* из *Report*
3. Бришемо редундантне односе

Primer spajanja shema – uklanjamo redundantne odnose sa izveštajem



Razumljivost

- Po potrebi dodatno uređujemo model da bi bio razumljiviji

Na narednom času...

- Grupisanje entiteta
- Pre toga ćemo prvo da detaljnije upoznamo model entiteta i odnosa