NORMALIZACIJA
Baze podataka 1

dr Miloš CVETANOVIĆ

Normalizacija

- Formalan postupak dizajna baze podataka
- Primena od samog početka dizajna ili nakon pravljenja modela entiteta i odnosa (EO)
- Ciljevi primene postupka je postizanje dizajna baze podataka koji:
    1. čuva podatke bez nepotrebne redundanse
    2. omogućava lako dohvatanje informacija
    3. obezbeđuje laku proveru određenih ograničenja
- Dizajn relacionih šema koje zadovoljavaju određene kriterijume – normalne forme
- Dizajn zasnovan na funkcijskim (i više-vrednosnim) zavisnostima
- Zavisi od informacija koje postoje u sistemu koji se modeluje, ali ne i u modelu EO

Fudbalski savez

Posmatrani sistem je fudbalski savez koji evidentira utakmice odigrane u toku jedne sezone. Za svaku utakmicu se trajno čuvaju podaci o tome koji timovi su igrali, u kom kolu je ta utakmica odigrana (i koje godine), kakav je ishod utakmice bio, koji su sve igrači igrali i na kojim pozicijama. Za svakog fudbalera se pamti ime i tim za koji igra, dok se za timove pamti naziv i mesto iz koga dolaze. Pretpostavka je da fudbaleri ne mogu da menjaju tim u kome igraju, u toku sezone. Takođe je potrebno trajno evidentirati svaki postignuti gol kao i to koji fudbaler ga je postigao, u kom minutu i koji je to gol bio po redu na posmatranoj utakmici. Pored golova trajno se evidentiraju i svi kartoni, žuti i crveni, dodeljeni na utakmici i to kom fudbaleru i u kom minutu.

FUDBALER -> Šifra fudbalera, Ime, Naziv tima, Mesto

Šifra	Ime	Naziv	Mesto
1	Edwin van der Sar	Manchester United	Manchester
2	Rio Ferdinand	Manchester United	Manchester
3	Nemanja Vidic	Manchester United	Manchester
4	Wayne Rooney	Manchester United	Manchester
5	Tal Ben Haim	Maccabi	Tel Aviv
6	Cesc Fabregas	Arsenal	London
7	Dejan Stankovic	Inter	Milano
8	Lionel Messi	Barselona	Barselona
9	Tal Ben Haim	Beitar	Jerusalem
10	Andres Iniesta	Barselona	Barselona

Čuvanje podataka u jednoj tabeli -> jednostavnije dohvatanje podataka?
Šta ako tim promeni naziv? problem višestrukog ažuriranja …slično i višestrukog brisanja
Da li se može "uštedeti" prostor?

Da li možemo da evidentiramo tim u kome ne igra nijedan fudbaler? … anomalija unošenja
Anomalija brisanja – ako se želi brisanje tima, neophodno je obrisati i sve fudbalere.

Rešenje je dekompozicija relacione šeme

Nije svaka dekompozicija dobra!

Šifra	Ime	Naziv	Mesto
1	Edwin van der Sar	Manchester United	Manchester
2	Rio Ferdinand	Manchester United	Manchester
3	Nemanja Vidic	Manchester United	Manchester
4	Wayne Rooney	Manchester United	Manchester
5	Tal Ben Haim	Maccabi	Tel Aviv
6	Cesc Fabregas	Arsenal	London
7	Dejan Stankovic	Inter	Milano
8	Lionel Messi	Barselona	Barselona
9	Tal Ben Haim	Beitar	Jerusalem
10	Andres Iniesta	Barselona	Barselona

Dekompozicija bez gubitka
pri spajanju, R na R1 i R2
kao skupova atributa R = R1 U R2
samo ako
nema gubitka informacija prilikom
zamene R sa R1 i R2

Neophodna je reverzibilnost.

FUDBALER (IdFud, Ime, Naziv, Mesto)

FUDBALER 1 (IdFud, Ime)
FUDBALER 2 (Ime, Naziv, Mesto)

“spajanjem“ FUDBALER 1 i FUDBALER 2
…višak podataka …gubitak informacija

Šta ako dva fudbalera imaju isto ime, a igraju u različitim klubovima? npr. Tal Ben Haim

Dekompozicija

- Postupak dolaska do skupa relacionih šema od kojih je svaka u „dobroj“ formi
  odnosno, nema ponavljanja informacija
- Više različitih „dobrih“ formi – normalnih formi
- Svaka relaciona šema koja nije u željenoj „dobroj“ formi se dekomponuje

- Dodatne ograničenja/pravila u sistemu koji se modeluje – najčešće su to funkcijske zavisnosti
- Funkcijske zavisnosti, poput:
  1. Svaki fudbaler se jedinstveno identifikuje svojom šifrom
  2. Svaki fudbaler i tim imaju samo jedno ime, odnosno naziv
  3. Svaki tim ima samo jedan grad u kome se nalazi

- Najčešće se ograničenja sistema mogu predstaviti formalno kao ključevi
  (kandidat ključ, primarni ključ)
  pojam super ključa?

- Ključevi - omogućavaju jedinstvenu identifikaciju entiteta
- Funkcijske zavisnosti - omogućavaju jedinstvenu identifikaciju vrednosti određenih atributa

- Funkcijska zavisnost $X \rightarrow Y$, gde su X i Y skupovi atributa relacione šeme R
  govori da ako dva entiteta koji imaju jednake vrednosti za atribute X
  onda ta dva entiteta moraju imati jednake vrednosti za atribute Y

$$\forall t_1 t_2 \, ((t_1 \in r \land t_2 \in r \land t_1[X] = t_2[X]) \Rightarrow t_1[Y] = t_2[Y])$$

FUDBALER (IdFud, Ime, Naziv, Mesto)

Funkcijska zavisnost Naziv → Mesto (jer za svaki tim postoji jedinstven grad u kome se nalazi)

Par atributa IdFud, Naziv formira superključ jer: IdFud, Naziv → Ime, Mesto

Međutim, ne postoji funkcijska zavisnost Ime → Naziv

Šifra	Ime	Naziv	Mesto
1	Edwin van der Sar	Manchester United	Manchester
2	Rio Ferdinand	Manchester United	Manchester
3	Nemanja Vidic	Manchester United	Manchester
4	Wayne Rooney	Manchester United	Manchester
5	Tal Ben Haim	Maccabi	Tel Aviv
6	Cesc Fabregas	Arsenal	London
7	Dejan Stankovic	Inter	Milano
8	Lionel Messi	Barselona	Barselona
9	Tal Ben Haim	Beitar	Jerusalem
10	Andres Iniesta	Barselona	Barselona

Na primer:

IdFud → Ime, Naziv
Naziv → Mesto

Logički implicira: IdFud → Mesto

Trivijalne funkcijske zavisnosti: X → X ili XY → X su zadovoljene od strane svih relacija

Skup funkcijskih zavisnosti F nad relacionom šemom R logički implicira F^+ (to je zatvarač F)

Funkcijske zavisnosti mogu pokazati da je dekompozicija bez gubitaka pri spajanju

Dekompozicija relacione šeme R na relacione šeme R1 i R2 je bez gubitaka
ako je zadovoljeno da $R1 \cap R2 \rightarrow R1$ ili $R1 \cap R2 \rightarrow R2$

FUDBALER (IdFud, Ime, Naziv, Mesto)

F= {IdFud → Ime, Naziv; Naziv → Mesto}

FUDBALER (IdFud, Ime, Naziv)
TIM (Naziv, Mesto)

FUDBALER ∩ TIM = {Naziv}, a pri tom važi Naziv → Mesto odnosno Naziv → TIM
Dakle dekompozicija je bez gubitaka pri spajanju

BCNF

- Jedna od najpoželjnijih normalnih formi je BCNF (Boyce - Codd Normal Form)
- BCNF eliminiše sve redundantnosti podataka koje se mogu otkriti funkcijskim zavisnostima

- Relacione šema R nad kojom važi skup funkcijskih zavisnosti F je u BCNF
  ako za sve funkcijske zavisnosti $X \to Y$ koje postoji u F^+ takve da je $X \subseteq R$ i $Y \subseteq R$ važi bar jedno:
    1. $X \to Y$ je trivijalna funkcijska zavisnost (odnosno $Y \subseteq X$)
    2. X je superključ za R

FUDBALER (IdFud, Ime, Naziv, Mesto)

F= {IdFud → Ime, Naziv; Naziv → Mesto}

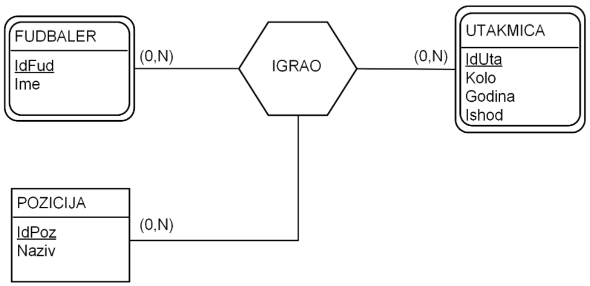
Naziv nije superključ za FUDBALER, te zbog toga FUDBALER nije u BCNF

Međutim, dekompozicija na FUDBALER (IdFud, Ime, Naziv) i TIM (Naziv, Mesto) jeste u BCNF
IdFud je ključ za FUDBALER, a Naziv je ključ za TIM

Postupak dekompozicije relacione šeme R do BCNF
ako postoji $X \to Y$ takva da X nije superključ za R
onda se R zamenjuje sa R1 i R2, takve da R1 sadrži atribute $X \cup Y$, a R2 atribute $R - (Y - X)$

Očuvanje zavisnosti prilikom dekompozicije

- Provera ograničenja (primarni ključ, kandidat ključ, strani ključ, funkcijska zavisnost …)
- Ograničenja čija provera zahteva proveru nad više tabela je „skupa“ operacija
- Dekompozicija do BCNF ne garantuje očuvanje funkcijskih zavisnosti



Fudbaler na više pozicija na jednoj utakmici.

IGRAO (IdFud, IdUta, IdPoz)

Ako želimo da fudbaler može da igra na više pozicija
ali maksimalno na jednoj pozicija na jednoj utakmici
onda je neophodno da važi: IdFud, IdUta → IdPoz

Ako bismo imali dodatni zahtev
da se jedna pozicija može igrati maksimalno na jednoj utakmici
onda je neophodno da važi IdPoz → IdUta
(narušava BCNF u IGRAO)

R1 (IdPoz, IdUta)
R2 (IdPoz, IdFud)

Odustati od zahteva za BCNF?

Zavisnost IdPoz → IdUta se proverava nad R1
Provera zavisnosti IdFud, IdUta → IdPoz zahteva i R1 i R2

3NF

- Može garantovati očuvanje funkcijskih zavisnosti
- Relaksira uslove postavljene za BCNF time što toleriše netrivijalne funkcijske zavisnosti koje nisu superključne

- Relacione šema R nad kojom važi skup funkcijskih zavisnosti F je u 3NF ako za sve funkcijske zavisnosti $X \rightarrow Y$ koje postoji u F^+ takve da je $X \subseteq R$ i $Y \subseteq R$ važi bar jedno:
  1. $X \rightarrow Y$ je trivijalna funkcijska zavisnost (odnosno $Y \subseteq X$)
  2. X je superključ za R
  3. Svaki atribut is skupa Y - X je ključni, odnosno sadržan je u nekom kandidat ključu za R

- Nije neophodno da svi atributi iz skupa Y - X pripadaju jednoj istom kandidat ključu

- Svaka šema koja zadovoljava BCNF automatski zadovoljava i 3NF, obrnuto ne važi

IGRAO (IdFud, IdUta, IdPoz) sa skupom F = {IdFud, IdUta → IdPoz; IdPoz → IdUta}

Zbog IdPoz → IdUta relaciona šema IGRAO nije bila u BCNF
Međutim, IdUta je atribut koji pripada kandidat ključu nad IGRAO, jer IdFud, IdUta → IdPoz

Nedostaci 3NF:
1. Nekada je neophodno tolerisati null da bi se predstavili neki odnosi
2. Provera funkcijskih zavisnosti u SQL (ako nisu superključne) je problematično (tj. „skupo")

Više normalne forme

- Posmatranje funkcijskih zavisnosti u određenim slučajevima nije dovoljno da bi se izbeglo ponavljanje podataka

- Ako bi se relaciona šema FUDBALER (IdFud, Ime) proširila:
  1. Informacijama o brojevima telefona koje fudbaler ima
  2. Informacijama o deci koje fudbaler ima

- Model EO bi kreirao tri relacione šeme:
  FUDBALER (IdFud, Ime) DETE (IdFud, ImeDeteta) TELEFON (IdFud, BrTel)

- Relaciona šema FUDBALER (IdFud, Ime, ImeDeteta, BrTel) je u BCNF jer nema netrivijalnih funkcijskih zavisnosti, ali bi ipak došlo do ponavljanja informacija
- U tabeli FUDBALER bi za svako dete morao da postoji po jedan red za svaki broj telefona

- Normalne forme zasnovane na funkcijskim zavisnostima nisu dovoljne (4NF, PJNF, DKNF)

Teorija funkcijskih zavisnosti

- Zatvarač skupa funkcijskih zavisnosti F označava se sa F^+
- F^+ je logički impliciran od F
- Svaka funkcijska zavisnost implicirana od F uvek važi nad svakom relacijom nad kojom važi F
- Armstrongove aksiome:
  1. Refleksinovst (reflexivity): Ako je $Y \subseteq X$ onda važi $X \rightarrow Y$
  2. Uvećanje (augmentation): Ako važi $X \rightarrow Y$ onda važi i $ZX \rightarrow ZY$
  3. Tranzitivnost (transitivity): Ako važe $X \rightarrow Y$ i $Y \rightarrow Z$ onda važi i $X \rightarrow Z$

- Dodatna pravila:
  1. Unija (union): Ako važe $X \rightarrow Y$ i $X \rightarrow Z$ onda važi i $X \rightarrow YZ$
  2. Dekompozicija (decomposition): Ako važi $X \rightarrow YZ$ onda važe i $X \rightarrow Y$ i $X \rightarrow Z$
  3. Pseudotranzitivnost (pseudotransitivity): Ako važe $X \rightarrow Y$ i $ZY \rightarrow Q$ onda važi i $ZX \rightarrow Q$

- Postupak određivanja F^+ na osnovu aksioma je vremenski zahtevno
- **Postupak određivanja zatvarača X^+** skupa atributa X nad skupom funkcijskih zavisnosti F

```
Rezultat = X;
While (promene u Rezultat) do
        For Each (Y→Z in F) do
            If Y ⊆ Rezultat Then Rezultat = Rezultat U Z;
        End For;
End While;
```

Moguća upotreba za:
1. Određivanje ključa relacije R
2. Proveru da li nad R važi $X \rightarrow Y$
3. Određivanje F^+

- **Postupak određivanja kanoničnog pokrivača F_c** skupa funkcijskih zavisnosti F koji logički implicira F (može postojati više F_c za dato F)
- Uklanjanje suvišnih atributa sa leve i desne strane svake funkcijske zavisnosti $X \rightarrow Y$ u skupu F
  1. Atribut $A \in X$ je suvišan ukoliko F logički implicira $(F - \{X \rightarrow Y\}) \cup \{(X - A) \rightarrow Y\}$
  2. Atribut $A \in Y$ je suvišan ukoliko skup $(F - \{X \rightarrow Y\}) \cup \{X \rightarrow (Y - A)\}$ logički implicira F

```
Fc = F;
While (promene u Fc) do
        Pravilom unije zameniti sve zavisnosti u Fc tako da X→Y1 i X→Y2 postane X→Y1Y2
        For Each (X→Y in Fc) do
                If postoji suvišni atribut A ili u X ili u Y Then ukloni A iz X→Y u Fc
        End For;
End While;
```

Postupak sa dva prolaza:

1. Ukloniti sve suvišne atribute A sa leve strane svake od zavisnosti $X \rightarrow Y$ tako što se računa zatvarač skupa atributa $(X-A)^+$ nad F i proverava da li sadrži Y

2. Ukloniti sve suvišne atribute A sa desne strane svake od zavisnosti $X \rightarrow Y$ tako što se računa zatvarač skupa atributa $(X)^+$ nad F' i proverava da li sadrži Y pri čemu je $F' = (F - \{X \rightarrow Y\}) \cup \{X \rightarrow (Y - A)\}$

Pre 2. prolaska treba primeniti pravilo dekompozice na sve zavisnosti i nakon toga ukloniti sve duplikate i trivijalne zavisnosti

- **Postupak provere očuvanja skupa funkcijski zavisnosti F** nakon dekompozicije D relacione šeme R na relacione šeme $R_1 \dots R_n$

- Projekcija skupa F na R_i je skup F_i svih zavisnosti iz F^+ koje sadrže isključivo atribute iz R_i

```
Izračunati F+;
For Each Ri in D do
      Fi = projekcija F+ na Ri
End For;
F' = Ø
For Each Fi do
      F' = F' ∪ Fi
End For;
Izračunati F'+ ;
If (F'+ = F+) then return TRUE
Else return FALSE;
```

```
For Each X→Y in F do
      Rezultat = X
      While (promene u Rezultat) do
            For Each Ri in D do
                  t = (Rezultat ∩ Ri)+ ∩ Ri
                  Rezultat = Rezultat ∪ t
            End For;
      End While;
      If Y ⊈ Rezultat then return  FALSE
      Else continue
End For;
Return TRUE;
```

Zatvarač skupa atributa se određuje nad F

- **Postupak dekompozicije do BCNF**

```
Rezultat = {R};
Kraj = FALSE;
While (not Kraj) do
     If (postoji Ri in Rezultat koja nije u BCNF) Then
            neka je X→Y netrivijalna koja važi nad Ri
            i koja je takva da X+ ne sadrži Ri i X ∩ Y = ∅;
            Rezultat = (Rezultat - Ri) ∪ (Ri - Y) ∪ (X, Y);
    Else Kraj = TRUE;
End While;
```

- **Postupak dekompozicije do 3NF**

```
Neka je Fc kanonični pokrivač za F
i := 0
For Each functional dependency X→Y in Fc do
     i = i + 1;
    Ri = (X,Y);
End For;
If nijedna Rj, 1 ≤ j ≤ i ne sadrži neki kandidat ključ za R Then
    i := i + 1;
    Ri:= neki kandidat ključ za R;
Repeat                  Radi na 5.času
    If bilo koja šema Rj sadržana u drugoj šemi Rk Then
        Rj = Ri;
         i = i - 1;
Until nema više Rj koje mogu biti obrisane
Return (R1, R2, ..., Ri)
```

Primer 1 – određivanje skupa kandidat ključeva

- Data je relacija R (A, B, C, D, E) i skup funkcijskih zavisnosti F ={ A→BC, CD→E, B→D, E→A }

- Najpre za svaki od atributa nadjemo zatvarač skupa atributa. Zatim isto to radimo za sve parove, trojke, četvorke atributa. Ukoliko se u zatvaraču pojave svi ostali atributi, taj atribut ili skup atributa funkcijski određuje sve ostale atribute u relaciji i samim tim je kandidat ključ.

A+ = A|
Iz A->BC sledi da A+ = A|BC
Prolazeći dalje kroz skup F sledi A+ = A|BC|D pa potom A+ = A|BC|D|E

- Algoritam staje kada više nije moguće ubacivati nove atribute, ili ako su u zatvaraču skupa već svi atributi iz relacije – u tom slučaju je pronađen jedan od kandidat ključeva.
Dakle: KK = {A}

- Potom za ostale atribute
B+ = B|D

C+ = C

D+ = D

E+ = E|A   Dakle: KK = {A, E}

Primer 1 – određivanje skupa kandidat ključeva

- Treba nastaviti i obraditi sve kombinacije od dva, tri i više atributa. Pritom nema smisla proveravati superključne kombinacije, tj. kombinacije koje sadrže kandidat ključ

(BC)+ = BC

(BC)+ = BC|D

(BC)+ = BC|D|E   Dakle: KK = {A, E, BC}

(BD)+ = BD   D je funkcijski određeno sa B, ova kombinacija neće ništa doprineti

(CD)+ = CD|E   Dakle: KK = {A, E, BC, CD}

- Primetiti da:

1. Ako se u datom skupu funkcijskih zavisnosti atribut pojavljuje samo sa leve strane (jedne ili više zavisnosti), do njega se ne može doći iz bilo kog drugog ključa. To znači da on obavezno ulazi u sastav nekog od kandidat ključeva

2. Ako se u datom skupu funkcijskih zavisnosti atribut pojavljuje samo sa desne strane (jedne ili više zavisnosti), do njega se uvek može doći iz nekog kog drugog ključa. To znači da on neće biti deo nekog od kandidat ključeva

Primer 2 – dekompozicija, gubitak pri spajanju i očuvanje funkcijskih zavisnosti

- Data je relacija R (A, B, C, D, E) i skup funkcijskih zavisnosti F ={ A→BC, CD→E, B→D, E→A }
- Primer 1 je odredio KK = {A, E, BC, CD}

- Data je dekompozicija D1={ R_1(A, B, C) , R_2(A, D, E) }

$R_1 \cap R_2 \rightarrow A$ pritom A ∈ KK, dakle A → R, odnosno, dekompozicija bez gubitaka pri spajanju

Treba proveriti da li je došlo do gubitaka funkcijskih zavisnosti
Potrebno je odrediti projekcije skupa F na R_1 i R_2, odnosno odrediti F_1 i F_2

R_1(A, B, C)	R_2(A, D, E)
A+ = A|BC|D|E	A+ = A|BC|D|E
B+ = B|D	D+ = D
C+ = C	E+ = E|A...
(BC)+ = BC|D|E|A	
F_1 = { A → BC, BC → A }	F_2 = { A → DE, E → AD }

Levu stranu funkcijiske zavisnosti čine atributi nad kojima se pravi zatvarač,
a desnu čine svi ostali atributi iz preseka (tj. nema trivijalnih zavisnosti tipa C→C)

Primer 2 – dekompozicija, gubitak pri spajanju i očuvanje funkcijskih zavisnosti

- Uniju F_1 i F_2 upoređujemo sa F ={ A→BC, CD→E, B→D, E→A }

F_1 U F_2 = F_{D1} = { A → BCDE, BC → A , E → AD }

Razlika je u CD→E i B→D (za ove funkcijske zavisnosti kažemo da su prividno izgubljene)

Da bi proverili da li su zaista izgubljene, treba da napraviti zatvarač nad unijom F_1 U F_2 (ne nad originalnim skupom F).

Zavisnosti kod kojih se traženjem zatvarača leve strane dolazi do desne strane nisu izgubljene.

(CD)+ = CD   nismo došli do desne strane, došlo je do stvarnog gubitaka CD→E

B+ = B   nismo došli do desne strane, došlo je do stvarnog gubitaka B→D

Prividno izgubljene zavisnosti se mogu proveriti spajanjem tabela dekompozicije

Stvarno izgubljene zavisnosti se ne mogu proveriti

Primer 3 – dekompozicija do BCNF, gubitak funkcijskih zavisnosti

- Data je relacija R (A, B, C) i skup funkcijskih zavisnosti $F=\{B \rightarrow C, AC \rightarrow B\}$

- Dekomponovati šemu R do BCNF, a potom navođenjem konkretnih relacija nad šemama dobijenih dekompozicijom pokazati da je došlo do gubitka zavisnosti.

$B \rightarrow C$ narušava BCNF te se R dekomponuje na R_1 (B, C) i R_2 (A, B)

R_1 (B, C)

B+ = B|C

C+ = C

$F_1 = \{ B \rightarrow C \}$

B	C
B1	C1
B2	C1

R_2(A, B)

A+ = A

B+ = B|C

$F_2 = \{ \}$

A	B
A1	B1
A1	B2

Ako pokušamo da rekonstruičemo tabelu dobićemo (spajanje po B):

A	B	C
A1	B1	C1
A1	B2	C1

$B \rightarrow C$ je zadovoljena

$AC \rightarrow B$ je narušena! (AC se preslikava u dve različite vrednosti B)

Primer 4 – dekompozicija do BCNF

- Data je relacija R (A, B, C, D, E) i skup funkcijskih zavisnosti $F=\{AB \rightarrow C, C \rightarrow DE, D \rightarrow E\}$
- Dekomponovati šemu R do BCNF

$C \rightarrow DE$ narušava BCNF te se R dekomponuje na R_1 (C, D, E) i R_2 (A, B, C)

$R_1(C, D, E)$	$R_2(A, B, C)$
$C+ = C\|D\|E$	$A+ = A$
$D+ = D\|E$	$B+ = B$
$(DE)+ = D\|E$	$C+ = C\|D\|E$
	$(AB)+ = AB\|C$
$F_1 = \{C \rightarrow DE, D \rightarrow E\}$	$(AC)+ = AC$
	$(BC)+ = BC$
	$F_2 = \{AB \rightarrow C\}$

$D \rightarrow E$ narušava BCNF te se R_1 (C, D, E) dekomponuje na R_{11}(D, E) i R_{12}(C, D)

Dakle, polazno R (A, B, C, D, E) je dekomponovana na tri šeme (A, B, C), (D, E), (C, D)

Primer 5 – dekompozicija do 3NF, određivanje kanoničnog pokrivača

- Dati su: Relacija R(A, B, C, D, E, F, G, H)
  Skup funkcijskih zavisnosti F={ABH→C, A→D, C→E, BGH→F, F→AD, E→F, BH→E}
  Skupa kandidat ključeva KK = {BGH}

- Dekomponovati šemu R do 3NF, algoritmom koji garantuje očuvanje funkcijskih zavisnosti

Određivanje kanoničnog pokrivača F_c

Prvi prolaz: uklanjanje suvišnih atributa sa leve strane funkcijskih zavisnosti

ABH→C ima tri atributa sa leve strane; gledamo da li sa leve strane ima suvišnih atributa.

A: (BH)+ = BH|E|F|A

Ukoliko se u zatvaraču pojavi atribut za koji proveravamo da li je suvišan, onda je on višak; takođe, ako se u zatvaraču dođe do desne strane (u ovom slučaju C), atribut je višak.
U ovom slučaju A je suvišan i treba ga uklaniti.

B: (H)+ = H
H: (B)+ = B

Ni B ni H nisu suvišni.

Zavisnosti koje imaju samo jedan atribut sa leve strane nema potrebe proveravati u ovom prolazu (one će biti uklonjene kada se budu uklanjali atributi sa desne strane).

Primer 5 – dekompozicija do 3NF, određivanje kanoničnog pokrivača

BGH→F ima tri atributa sa leve strane

B: (GH)+ = GH

G: (BH)+ = BH|E|F

U zatvaraču se došlo do cele leve strane. G je suvišno i treba ga uklaniti.

H: (B)+ = B

Ni B ni H nisu suvišni atributi.

BH→E ima dva atributa sa leve strane

B: (H)+ = H

H: (B)+ = B

Ni B ni H nisu suvišni atributi.

Time je završen prvi prolaz:

F_{c1} = { BH→C, A→D, C→E, BH→F, F→AD, E→F, BH→E }

U ovom koraku se mogu, ali ne moraju sažimati relacije BH→CFE (te ih nećemo sažimati)

Primer 5 – dekompozicija do 3NF, određivanje kanoničnog pokrivača

Drugi prolaz: uklanjanje suvišnih atributa sa desne strane funkcijskih zavisnosti

Najpre treba dekomponovati funkcijske zavisnosti tako da na desnoj strani bude jedan atribut
Treba ukloniti sve trivijlane zavisnosti i sve duplikate

F_{c1} = { BH→C, A→D, C→E, BH→F, F→A, F→D, E→F, BH→E }

Na početku $F_{c2} = F_{c1}$

Za svaku od zavisnosti X→Y se radi sledeći postupak:
1. Ukloniti X→Y privremeno iz skupa F_{c2}
2. Odrediti zatvarač leve strane (X) uklonjene zavisnosti nad skupom F_{c2} bez uklonjene relacije
3. Ako se u zatvaraču dođe do desne strane (Y), onda treba X→Y trajno izbaciti iz skupa F_{c2}
4. Ako se u zatvaraču ne može doći do desne strane (Y), onda se X→Y ne uklanja iz skupa F_{c2}

BH→C:
(BH)+ = BH|FE|A|D   C nije suvišan atribut

A→D:
(A)+ = A   D nije suvišan atribut

C→E:
(C)+ = C   E nije suvišan atribut

Primer 5 – dekompozicija do 3NF, određivanje kanoničnog pokrivača

BH→F:

(BH)+ = BH|CE|F F je suvišan atribut i treba ukloniti BH→F

F_{c2} = { BH→C, A→D, C→E, F→A, F→D, E→F, BH→E }

F→A:

(F)+ = F|D A nije suvišan atribut

F→D:

(F)+ = F|A|D D je suvišan atribut i treba ukloniti F→D

F_{c2} = { BH→C, A→D, C→E, F→A, E→F, BH→E }

E→F:

(E)+ = E F nije suvišan atribut

BH→E:

(BH)+ = BH|CF|E E je suvišan atribut i treba ukloniti BH→E

F_{c2} = { BH→C, A→D, C→E, F→A, E→F }

Napraviti uniju funkcijskih zavisnosti sa istom levom stranom (npr. BH→C, BH→F u BH→CF)

F_c = { BH→C, A→D, C→E, F→A, E→F }

Primer 5 – dekompozicija do 3NF, određivanje kanoničnog pokrivača

- Primena algoritma koji garantuje očuvanje funkcijskih zavisnosti (generiše dekompoziciju)

$F_c = \{ BH \to C, A \to D, C \to E, F \to A, E \to F \}$

$R_1(B,H,C)$, $R_2(A,D)$, $R_3(C,E)$, $R_4(F,A)$, $R_5(E,F)$

Ni jedna od relacija nije sadržana u nekoj drugoj.

Ni u jednoj od relacija nije sadržan neki od kandidat ključeva,
dodajemo jednu relaciju u koju stavljamo proizvoljan kandidat ključ (u primeru je to BGH)

$R_1(B,H,C)$, $R_2(A,D)$, $R_3(C,E)$, $R_4(F,A)$, $R_5(E,F)$, $R_6(B,G,H)$