Forme normale pentru schemele de relație

prof. dr. ing. Mircea Petrescu

Folosirea formelor normale conduce la eliminarea multora din problemele de redondanțe şi anomalii enunțate anterior.

Fie o schemă R; un atribut A din R este *prim*, dacă A face parte din orice cheie pentru R. Altfel A este *neprim*.

Exemplu. În schema R=OSC sunt prime toate atributele, deoarece fiind date dependențele OS→C şi C→O, sunt chei atât OS cât şi SC. Pe de altă parte, în schema ABCD cu dependențele AB→C, B→D şi BC→A, singurele chei sunt AB şi BC, deci A, B, C sunt prime, iar D neprim.

Prima formă normală (FN1)

În prima formă normală, domeniul fiecărui atribut este construit din valori indivizibile. Cu alte cuvinte, în FN1 domeniile atributelor nu conțin mulțimi de valori sau tupluri, luate din domenii „mai elementare". În general, pentru noi o relație este echivalentă cu FN1, prin definiție.

A doua formă normală (FN2)

Se defineşte mai jos.

A treia formă normală (FN3)

O schemă de relație R este în FN3 dacă nu există o cheie X pentru R, o mulțime de atribute $Y \subseteq R$ şi un atribut neprim A în R, dar $A \notin X$, $A \notin Y$, astfel încât:

a) X→Y să fie respectată în R;
b) Y→A să fie respectată în R, dar
c) Y→X să nu fie respectată în R.

Dacă Y este o submulțime a lui X şi ca urmare a condiției c), Y este o submulțime proprie a lui X, spunem că R are o *dependență parțială*. Dacă Y nu este o submulțime a lui X, atunci R admite o *dependență tranzitivă*.

Spunem că R este în FN2 atunci când R satisface condițiile de mai sus ori de câte ori $Y \subseteq X$, dar nu în mod necesar altfel.

Exemplu: schema de relație R=FDAP, cu dependențele FA→P şi F→D nu respectă FN3 (de asemenea nu respectă FN2). Într-adevăr, fie X=FA, Y=F. Atributul D (adresa!) este neprim, deoarece singura cheie este FA. Atunci X→Y şi Y→D sunt dependențele valabile, pe când Y→X (adică F→FA) nu este valabilă. Vom observa că în acest caz, X→Y şi Y→D nu numai că „funcționează" în R, ci ele sunt dependențe date.

În general, însă, este suficient ca X→Y şi Y→D să decurgă dintr-o mulțime dată de dependențe, chiar dacă ele sunt date ca atare.

Un alt exemplu: schema de relație R=OSC este în FN3, condițiile corespunzătoare acestei forme fiind îndeplinite de la sine.

Exemplu de schemă de relație în FN2, dar nu în FN3: R=MARS, unde M=magazin, A=articol, R=nr. raion, S=şef de raion.

Presupunem că funcționează următoarele dependențe funcționale:
MA→R (fiecare articol în fiecare magazin este vândut de cel mult un raion)
MR→S (fiecare raion în fiecare magazin are un şef)

Schema de relație are o singură cheie: MA. Dacă notăm X=MA şi Y=MR, iar A=S, atunci regulile care definesc FN3 nu sunt îndeplinite. Remarcăm totuşi că nu există dependențe parțiale deoarece nicio submulțime proprie a cheii MA nu determină funcțional atributele M sau S.

Necesitatea FN3

Aşa cum am spus, prin FN3 se evită multe din probleme legate de redondanță şi de anomaliile de actualizare. Să adâncim această idee.

Putem, astfel, presupune că dependențele funcționale X→Y nu reprezintă numai o restricție de integritate asupra relațiilor, ci reprezintă, în acelaşi timp, o legătură (asociere) pe care BD *are intenția* să o memoreze. Cu alte cuvinte, dacă cu atributele din X este asociată o mulțime de valori, considerăm important să ştim ce valoare pentru fiecare atribut din Y este asociată cu această „asignare" de valori pentru atributele din X. Dacă avem o dependență parțială Y→A, X fiind o cheie iar Y o submulțime proprie a lui X, atunci în fiecare tuplu folosit pentru a asocia o valoare X (din mulțimea asociată cu X) cu valori pentru alte atribute în afară de A şi de atributele din X, trebuie să apară aceeaşi asociere între X şi A. Această situație este uşor evidențiată în schema FDAP, în care F→D este o dependență parțială, iar adresa furnizorului trebuie să fie repetată odată pentru fiecare articol livrat de furnizor. Evident că FN3 elimină această posibilitate, precum şi redondanțele respective şi anomaliile de actualizare.

În caz că există o dependență tranzitivă X→Y→A, atunci nu putem asocia o valoare –Y cu o valoare –X, dacă nu există o valoare –A asociată cu valoarea Y. Această situație conduce la anomalii de inserare şi de eliminare, acolo unde nu putem insera o asociere X-la-Y fără o asociere Y-la-A, iar dacă eliminăm valoarea A asociată cu o valoare –Y dată, vom „pierde urma" unei asocieri X-la-Y. De exemplu, în schema MARS, cu dependențele MA→R şi MR→S, nu putem înregistra un raion oarecare, dacă acel raion nu are şef.

Forma normală Boyce-Codd (FNBC)

Fie schema de relație R, cu mulțimea de dependențe F. Dacă ori de câte ori X→A este valabilă în R iar A∉X, mulțimea de atribute X conține o cheie pentru R, spunem că R este în FNBC. Cu alte cuvinte, singurele dependențe netriviale sunt acelea în care o cheie determină funcțional unul sau mai multe alte atribute.

Exemplu: schema R=OSC, cu dependențele OS→C şi C→O nu este în FNBC, deşi este în FN3; cauza pentru care R nu este în FNBC: C→O este valabilă (dependență dată!), dar C nu este cheie pentru schema OSC.

Notă. Am văzut că o schemă în FN3 poate să nu fie în FNBC. Pe de altă parte, orice schemă de relație în FNBC este exprimată şi în FN3.

Avantajul aducerii unei scheme în FNBC este, desigur, cel al evitării pericolului redondanțelor şi a anomaliilor de inserare şi de eliminare. În exemplul imediat anterior se vede cum nu putem înregistra un oraş căruia îi corespunde un anumit cod. Aşadar, FNBC elimină unele anomalii care nu pot fi combătute prin FN3.

Teoremă. Dacă o schemă de relație R cu mulțimea de dependențe F este în FNBC, atunci este şi în FN3. Fără demonstrație.

Descompunerea cu joncțiune fără pierderi în FNBC

Se poate arăta că orice schemă de relație are o descompunere cu joncțiune fără pierderi în FNBC; de asemenea, orice schemă admite o descompunere în FN3, descompunere care are o joncțiune fără pierderi şi păstrează dependențele.

Pe de altă parte, nu orice descompunere a unei scheme de relație în FNBC păstrează dependențele. De exemplu, schema R=OSC nu este în FNBC deoarece este valabilă dependența C→O; dacă însă OSC este descompusă în orice mod, astfel încât OSC să nu fie una din schemele care fac parte din descompunere, atunci OS→C nu este implicată de dependențele proiectate.

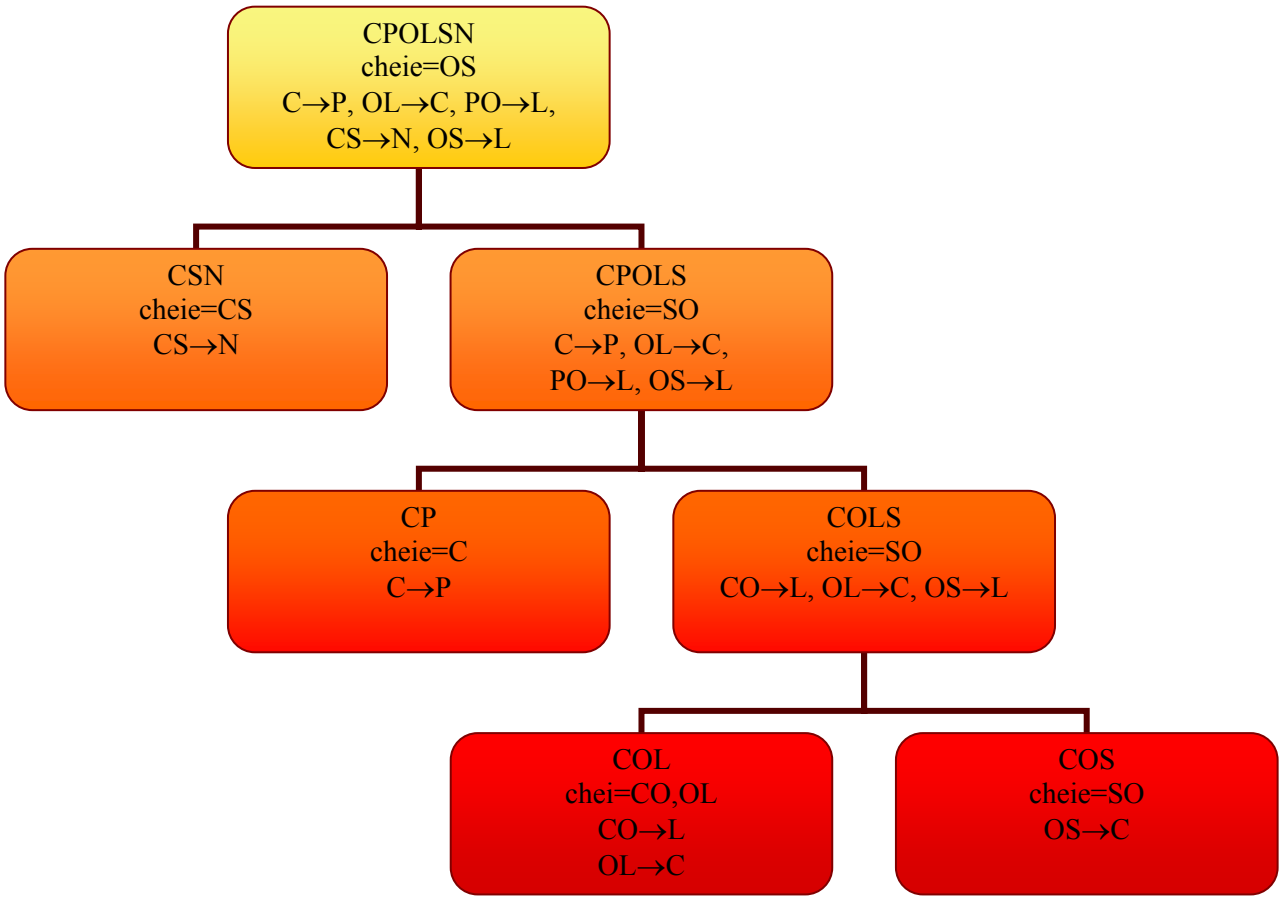
Algoritm de descompunere cu joncțiune fără pierderi în FNBC

Intrare: schema R, mulțimea F.

Ieşire: o descompunere a schemei R cu joncțiune fără pierderi, astfel încât fiecare schemă de relație din descompunere este în FNBC în raport cu proiecția mulțimii F pe acea schemă.

Metoda: se construieşte în mod iterativ o descompunere ρ pentru R. În orice moment al procesului iterativ, ρ va avea o joncțiune fără pierderi față de F. La început, ρ este format numai din R. Dacă S este o schemă de relație din ρ, iar S nu este în FNBC, fie X→A o dependență valabilă în S, astfel încât X nu include o cheie pentru S, iar A∉X. Atunci, trebuie să existe un atribut oarecare din S, care ≠A şi care ∉X, deoarece altfel X va include o cheie pentru S. În continuare, S se înlocuieşte în ρ prin S_1 şi S_2, unde S_1 constă din A şi din atributele lui X, iar S_2 din toate atributele din S, în afară de A.

Potrivit unei teoreme anterioare, descompunerea schemei S în S_1 şi S_2 admite o joncțiune fără pierderi în raport cu mulțimea de dependențe proiectată pe S, deoarece $S_1 \cap S_2 = X$, iar $X \rightarrow (S_1 - S_2) = A$. Se mai poate arăta, de asemenea, că dacă o descompunere ρ admite o joncțiune fără pierderi, atunci va avea aceeaşi proprietate şi dacă o schemă S din ρ va fi înlocuită cu S_1 şi S_2. Deoarece S_1 şi S_2 au fiecare mai puține atribute decât S, se va ajunge la un moment dat în o situație în care fiecare schemă de relație din ρ va fi în FNBC. În această fază finală ρ va poseda în continuare o joncțiune fără pierderi, deoarece a avut această proprietate în etapa inițială, când a fost constituită numai din R, iar proprietatea s-a păstrat în fiecare etapă a aplicării algoritmului.



Exemplu: fie schema de relație CPOLSN, unde C=cursul (numerorat, codificat), P=profesorul, O=ora, L=locul (sala), S=studentul, N=nota obținută. De asemenea, admitem următoarele dependențe funcționale: C→P (fiecare curs – un profesor), OL→C (doar un curs se poate ține în o sală la o oră dată), OP→L (un profesor poate fi numai în o sală la o oră dată), CS→N (fiecare student obține o notă la fiecare curs), OS→L (fiecare student se poate găsi numai în o sală la o oră dată).

Singura cheie acceptată de schema relației este OS. Aşa cum se va arăta, descompunerea schemei va conduce la următorul rezultat, reprezentat printr-un arbore.

În efectuarea descompunerii schemei de relație CPOLSN în FNBC (forma normală Boyce-Codd), putem începe cu examinarea dependenței CS→N, care nu îndeplineşte condiția din definiția FNBC, deoarece CS nu conține o cheie. Potrivit algoritmului de descompunere, secționăm CPOLSN în CSN şi CPOLS. Apoi, pentru realizarea descompunerilor care urmează, trebuie să calculăm închiderea F^+ şi să o proiectăm pe CSN şi CPOLS. Calculul închiderii F^+ şi al proiecțiilor este însă, în general, un consumator important de timp, deoarece $\#F^+$ poate fi ~exp(#F), unde # este cardinalul unei mulțimi.

Chiar şi în cazul relativ simplu tratat, F^+ va conține, fireşte, toate dependențele triviale care rezultă prin reflexivitate, precum şi dependențele netriviale ca CO→L, OS→C, OL→P, în afară de dependențele din F!

După ce se calculează F^+, se selectează acele dependențe care includ numai C, S, N. Aceasta va fi, de fapt, mulțimea dependențelor F proiectate pe CSN; mulțimea va avea o acoperire minimală care constă numai din CS→N; toate celelalte dependențe din mulțime decurg din aceasta prin axiomele lui Armstrong.

De asemenea, proiectăm F^+ pe CPOLS. Mulțimea proiectată are o acoperire minimală formată din dependențele: C→P, OL→C, PO→L, OS→L, iar schema CPOLS acceptă o singură cheie: OS.

Faptul că schema CSN este în FNBC se verifică uşor. În continuare, se procedează la descompunerea schemei CPOLS, în subschemele CP şi COLS, pornind de la dependența C→P. În subschema CP, pentru dependențele proiectate, avem o singură acoperire minimală: C→P. În subschema COLS, acoperirile minimale sunt CO→L, OS→L, OL→C, iar singura cheie a subschemei este OS. Mai remarcăm că dependența CO→L este necesară în o acoperire pentru COLS, deşi în CPOLS a decurs din C→P şi PO→L.

Subschema CP este în FNBC; în continuare, descompunerea subschemei COLS în COL şi COS, folosind dependența CO→L, ne conduce la rezultatul dorit pentru schema bazei de date.

Descompunerea finală a schemei R=CPOLSN, în FNBC, este dată de ρ=(CSN, CP, COL, CNS). Remarcăm că am ajuns la o structură bună de BD, deoarece din cele patru scheme de relație din ρ ne dau, printre altele:

a) notele studenților la diferite cursuri,
b) profesorul pentru fiecare curs,
c) orele la care au loc cursurile şi locurile (sălile de clasă) pentru fiecare oră,
d) orarul cursurilor şi orelor pentru fiecare student.

Observăm, însă, că nu orice descompunere produce o schemă de BD care să se potrivească aşa de bine cu ideile noastre despre modul în care trebuie tabulată informația în BD. De exemplu, dacă în ultima fază a procesului de descompunere am fi folosit nu dependența CO→L, ci OL→C, am fi ajuns la schema OLS în loc de COS; subschema OLS reprezintă sala de clasă în care un student poate fi găsit la o oră dată, nu anul (sau grupa) din care face parte (reprezentate prin COS). Este evident că COS conține o informație „mai fundamentală" decât OLS.

O altă problemă: descompunerea efectuată nu păstrează dependența PO→L. Cu alte cuvinte, proiecția mulțimii de dependențe F pe descompunerea ρ=(CSN, CP, COL, COS), proiecție care

poate fi reprezentată prin acoperirea CS→N, OL→C, C→P, OS→C, CO→L, care a fost determinată luând acoperirile minimale în fiecare din frunzele arborelui de descompunere, nu implică dependența PO→L. Ca exemplu, observăm că următoarea valoare curentă (relație) a schemei CPOLSN:

C	P	O	L	S	N
c_1	p	o	l_1	s_1	n_1
c_2	p	o	l_2	s_2	n_2

nu satisface dependența PO→L, deşi proiecțiile sale pe ρ satisfac toate dependențele proiectate.

Notă. S-a arătat (Beeri şi Bernstein, 1979) că problema verificării apartenenței unei relații la FNBC este NP-completă. Prin urmare, este greu de presupus că se poate găsi un algoritm de descompunere în FNBC care să fie executat într-un interval de timp mai scurt decât un interval exponențial.

Descompunerea în FN3 cu păstrarea dependențelor

Algoritm de descompunere:

Intrare: schema de relație R şi mulțimea de dependențe funționale F (presupunem că F este o acoperire minimală, fără pierderea generalității).

Ieşire: o descompunere care conservă dependențele, astfel încât fiecare schemă de relație este în FN3 în raport cu proiecția mulțimii F pe acea schemă.

Metoda: dacă în R sunt atribute care nu fac parte din nici o dependență din F, nici în părțile stângi, nici în părțile drepte, atunci un astfel de atribut poate forma, în principiu, el singur, o schemă de relație şi îl vom elimina din R. Dacă una din dependențele din F conține toate atributele din R, atunci se extrage chiar R – ca ieşire. Altfel, descompunerea ρ căutată va consta din schemele de tip XA pentru fiecare dependență X→A din F.

Dacă însă $X\rightarrow A_1$, $X\rightarrow A_2$, ..., $X\rightarrow A_n$ sunt dependențele din F, putem folosi schema $XA_1A_2...A_n$ în loc de XA_i pentru $1\leq i\leq n$; această substituție este de obicei preferabilă.

Exemplu. Dependențele C→P, OL→C, PO→L, CS→N, OS→L, care țin de schema de relație anterioară R=CPOLSN, au acoperirea minimală C→P, OL→C, PO→L, CS→N, OS→L (chiar F!). Algoritmul dat mai sus ne va conduce la descompunerea: ρ=(CP, COL, OLP, CNS, OLS), care păstrează dependențele şi este în FN3.

Descompunerea unei scheme de relație în FN3, cu păstrarea dependențelor şi cu joncțiune fără pierderi

Fie R o schemă de relație şi fie ρ o descompunere în FN3, obținută aplicând algoritmul anterior. De asemenea, fie X o cheie a schemei R. Se poate atunci arăta că $\tau=\rho\cup\{X\}$ este o descompunere a schemei R, în care toate subschemele de relație sunt în FN3, iar descompunerea păstrează dependențele şi are o joncțiune fără pierderi.

Exemplu. În exemplul anterior, s-a găsit o descompunere ρ a schemei R=CPOLSN, care are subschemele în FN3. Cheia schemei R este OS, care însă face parte din OLS, iar $OLS\in\rho$; ca urmare, OS este eliminată, iar τ=ρ=(CP, COL, OLP, CNS, OLS). Se poate verifica faptul că schemele din τ conservă dependențele, iar descompunerea admite joncțiune fără pierderi.