12.DEPENDENȚE FUNCȚIONALE ȘI CHEI PRIMARE

I.OBIECTIVE

1. Dependențe funcționale la nivelul atributelor unei baze de date
2. Determinarea cheilor primare folosind teoria dependențelor funcționale

II. FUNDAMENTE TEORETICE

2.1.DEPENDENȚE FUNCȚIONALE

Este posibil ca în dezvoltarea unei aplicații conținând o bază de date să apară problema proiectării incorecte a schemelor de relație, caz în care pot să apară o serie de ***anomalii*** ce pot complica programarea aplicațiilor. Testarea corectitudinii unei scheme de relații pentru o bază de date poate fi realizată utilizând conceptele și mecanismele specifice ***dependențelor funcționale .*** Acestea modelează *corelații* care există între datele din lumea reală stocate în baza de date și reprezintă *criterii de corectitudine* a datelor încărcate în baza de date.Dacă o relatie nu are o schemă corectă, capabilă să evite anomaliile,ea trebuie înlocuită cu două sau mai multe relații,operatia fiind numită și ***descompunerea unei scheme de relație*** .

Posibile anomalii rezultate din proiectarea defectuoasă a schemei unei relații vor fi analizate folosind tabela Produse (IdP, NumeP, Cant Idf, NumeF, AdresaF) , acestea sunt :

a. **Redundanța:** Redundanța reprezintă stocarea în mod nejustificat a unei aceleiași informatii de mai multe ori în baza de date. Observăm de exemplu că pentru fiecare produs este stocat numele și adresa furnizorului, deși ele sunt unic determinate de codul acestuia.
b. **Anomalia de actualizare:** In cazul actualizării unei informații redundante, se poate întampla ca operația să modifice unele apariții ale acesteia, iar altele să rămână cu vechea valoare.
c. **Anomalia de inserare:** Nu putem insera date despre un furnizor (numele și adresa sa) decât dacă există în stoc un produs furnizat de acesta.
d. **Anomalia de ştergere:** La ștergerea din relație a ultimului produs al unui furnizor se pierd automat și datele despre acesta.

Aceste anomalii apar deoarece într-o aceeași tabelă au fost stocate date despre *două clase diferite* de obiecte. Un set posibil de scheme de relație este urmatorul , rezultă că informația din relatia Produse trebuie stocată de fapt in două relații :

Furnizor(IdF, NumeF, AdresaF)
Produse(IdP, NumeP, Cant, IdF)

Procesul de divizare a unei tabele care are o structură incorectă în două sau mai multe tabele se numește *descompunerea schemei de relație.* Pentru detectarea relațiilor care trebuie descompuse există o serie de reguli de corectitudine, numite *forme normale.* Definirea acestor forme normale se bazează pe noțiunea de *dependență funcțională.*

Determinarea dependețelor funcționale

Notație:

- R, S, T, . . . : scheme de relații,
- r, s, . . . : instante ale relațiilor R respectiv S,A, B, C, D, . . . : atribute ale unei relații,
- X, Y, Z, W, U, . . . : mulțimi de atribute dintr-o schema de relatie,
- X $\subset$R: Mulțimea de atribute X este inclusă în mulțimea atributelor relației R.
- Y $\subseteq$ X: Mulțimea de atribute Y este inclusă în mulțimea de atribute X
- A $\subseteq$ X: Atributul A aparține mulțimii de atribute X
- t, t1, t2, . . . tupluri ale unei relații,
- t[X]: valorile atributelor din X aflate în tuplul t,
- F, G, . . . : mulțimi de dependențe funcționale atașate unei scheme de relație

Fie R o schemă de relație iar X, Y $\subset$R două mulțimi de atribute ale acesteia. Spunem că X *determină funcțional pe* Y (sau Y este determinat funcțional de X) dacă și numai dacă oricare ar fi două tupluri t1 și t2 din orice instanță a lui R atunci: dacă t1[X] = t2[X] rezultă t1[Y] = t2[Y],
Astfel ,dacă două tupluri au aceleași valori pe atributele X atunci ele au aceleasi valori și pe atributele Y**.**
Notația pentru dependențe funcționale este X $\rightarrow$Y.

In relația *Produse* avem următoarele dependențe funcționale:

IdP $\rightarrow$NumeP, Cant , IdF, NumeF, AdresaF
IdF $\rightarrow$NumeF, AdresaF

Aceste dependențe arată că: dacă două produse au același IdP, este vorba de fapt de același produs și dacă două produse au același IdF (Id furnizor) atunci și valorile pentru numele și adresa acestuia trebuie să fie aceleași. Dependențele funcționale nu se determină din semnificația atributelor acesteia. In exemplul prezentat, a doua dependență funcțională arată că dacă la două produse apare același Id furnizor atunci numele și adresa furnizorului sunt de asemenea aceleași (deoarece nu pot să existe doi furnizori diferiți cu același Id).Pornind de la o mulțime de dependențe funcționale atașate unei scheme de relație se pot deduce alte dependențe functionale valide, bazat pe o multitudine de *reguli de inferență***.** Pentru a se putea realiza o determinare formală a acestora, trei dintre ele sunt **axiomele Armstrong** ,iar celelalte se pot deduce pornind de la ele.

A1. Reflexivitate: Fie R o schema de relatie si X $\subseteq$ R. : daca Y $\subseteq$ X atunci X$\rightarrow$Y

Toate dependențele funcționale care rezultă din această axiomă sunt numite și ***dependențe triviale.*** Ele nu spun nimic în plus față de setul de dependențe inițial ,dar sunt dependențe funcționale valide.

A2. Augmentare: Fie R o schemă de relație și X, Y, Z$\subseteq$ R. :daca X $\rightarrow$Y atunci si XZ $\rightarrow$ YZ

Această axiomă arată că se poate reuni o aceeași mulțime Z în stânga și in dreapta unei dependențe funcționale valide obținând de asemenea o dependență funcțională validă.

A3. Tranzitivitate: Fie R o schemă de relație si X, Y, Z $\subseteq$ R. :daca X $\rightarrow$Y si Y $\rightarrow$ Z atunci si X $\rightarrow$Z

Pe baza acestor axiome se pot demonstra o serie de reguli de inferență pentru dependențe funcționale dintre care cele mai importante sunt următoarele:

R1. Descompunere: Fie R o schemă de relație și X, Y, Z $\subseteq$ R.:dacă X $\rightarrow$ Y și Z $\subseteq$ Y atunci și X $\rightarrow$ Z

Regula descompunerii ne permite să rescriem un set de dependente funcționale astfel încât sa obținem doar dependențe care au în *partea dreapta doar un singur atribut*.

Să presupunem ca avem o dependență funcțională de forma: X $\rightarrow$ A1A2A3…An

Atunci ea poate fi înlocuită cu următoarele **n** dependențe funcționale: X $\rightarrow$ A_1, X $\rightarrow$ A_2 ...X $\rightarrow$ An

R2. Reuniune: Fie R o schemă de relație și X, Y, Z $\subseteq$R. :daca X $\rightarrow$ Y si X $\rightarrow$ Z atunci si X $\rightarrow$YZ

Rezultă și faptul că din cele **n** reguli obținute prin descompunere se poate obține dependența inițială, deci înlocuirea acesteia nu duce la pierderea vreunei corelații existente.

R3. Pseudotranzitivitate: Fie R o schemă de relaţie şi X, Y, Z, W $\subseteq$ R.:daca X $\rightarrow$Y si YZ $\rightarrow$ W atunci și XZ $\rightarrow$ W

2.2.DETERMINAREA CHEILOR UNEI RELAȚII

Pornind de la un set de dependențe funcționale F și utilizând axiomele și regulile obținem o multitudine de alte dependențe, triviale sau nu. Mulțimea tuturor dependențelor funcționale care se pot deduce din F se numește ***închiderea mulțimii de dependențe*** F notată cu F+.

Definiția formală a acestei închideri este următoarea: F+ = {X $\rightarrow$Y | F => X $\rightarrow$ Y }

unde prin => am notat faptul că dependența respectivă se poate deduce din F folosind axiomele și regulile.

Multimea F^+ conține foarte multe dependențe, inclusiv dependențe triviale ca:
ABC $\rightarrow$A, ABC $\rightarrow$B, ABC $\rightarrow$C, ABC $\rightarrow$AB, ABC $\rightarrow$AC, ABC$\rightarrow$ BC sau ABC $\rightarrow$ABC

Ea nu se calculează (este dificil), algoritmii care au nevoie de ea ocolind într-un fel sau altul calculul acesteia. Introducerea acestei noțiuni a fost necesară pentru a explica, în cazul descompunerii unei scheme de relație, care sunt dependențele moștenite de elementele descompunerii de la relatia inițiala și pentru a putea defini formal alte noțiuni.

Acoperirea unei multțmi de DF: Fie R o schemă de relație și F, G două mulțimi de dependențe pentru R. Se spune ca F ***acopeăa*** pe G dacă și numai dacă G $\subseteq$ F+.

Echivalența a doua mulțimi de dependențe: Fie R o schema de relație si F, G două mulțimi de dependențe pentru R. Se spune ca F ***e echivalentă*** cu G dacă și numai dacă F acoperă pe G și G acoperă pe F (deci G$\subseteq$ F+ și F $\subseteq$ G+ , deci F+ = G+)

Forma canonică a unei mulțimi de DF: Din definițiile de mai sus rezultă că o mulțime de dependențe poate fi înlocuită cu alta echivalentă conținând alte dependențe. In cazul în care aceasta mulțime îndeplinește condițiile următoare se spune că este în ***forma canonică:***

- Orice dependență are î*n partea dreapta un singur atribut*. Acest lucru se poate obține aplicând regula descompunerii prezentată anterior.

- Mulțimea de dependențe este *minimală*, nici una dintre dependente neputând să fie dedusă din celelalte (altfel spus nu există dependențe redundante).

Definiția unei chei folosind dependențe funcționale: Fie R o schemă de relație, F mulțimea de dependențe funcționale asociată și $X \subseteq R$. Atunci X este cheie pentru R , dacă și numai dacă:

- **F => X $\rightarrow$R** (deci $X \subseteq R$ se poate deduce din F) și
- **X este minimală** : oricare ar fi $Y \subseteq X$, $Y \# X$ atunci not (F=> $Y \rightarrow R$) (deci orice submulțime strictă a lui X nu mai indeplinește condiția anterioară).

Deci ,o cheie a unei relații determină funcțional toate atributele relației și este minimală.

Astfel, cunoscându-se valorile pe atributele X , sunt unic determinate valorile pentru toate atributele relației, deci este unic determinat tuplul din relație. In cazul în care doar prima condiție este îndeplinită mulțimea X se numeste ***supercheie*.** Faptul că o supercheie nu este constrânsă de minimalitate nu inseamnă însă ca ea nu poate fi minimală. Rezultă ca orice cheie este în acelasi timp și supercheie, reciproca nefiind însă adevarată.

Fie relația Produse :

Produse = (IdP, NumeP,Cant, IdF, NumeF, AdresaF)

având asociată multimea de dependențe:

F = { IdP$\rightarrow$NumeP, IdP$\rightarrow$Cant, IdP$\rightarrow$ IdF, IdF$\rightarrow$NumeF, IdF$\rightarrow$AdresaF }

Prin reorganizarea acestei relații obținem :

Produse = (IdP, NumeP, Cant, IdF)
Furnizori= (IdF, NumeF, AdresaF)

Atributele relației inițiale se regăsesc fie doar într-una dintre schemele rezultate fie în amândouă. Se pune însa și problema: ce dependențe moștenesc cele două relații de la relația inițială? Soluția este de a defini *proiecția unei mulțimi de dependente pe o mulțime de atribute*.

Fie o relație R, o mulțime asociată de dependențe funcționale F și o submulțime de atribute $S \subseteq R$

Proiecția mulțimii de dependențe F p e S, notată cu

π_S (F) este mulțimea dependențelor din F+ care au și partea stângă și pe cea dreaptă incluse în S.

Formal putem scrie:

$$\pi S\ (F) = \{X \rightarrow Y \ \ F \in + \mid X, Y \subseteq S\ \}$$

Pentru exemplul de mai sus proiecțiile sunt următoarele:

$\pi_{PRODUSE}$(F) = { IdP$\rightarrow$NumeP, IdP$\rightarrow$Cant, IdP$\rightarrow$IdF}

$\pi_{FURNIZORI}$ (F) ={ IdF$\rightarrow$NumeF, IdF$\rightarrow$AdresaF }

Observație: Atunci când descompunem o schemă se poate întampla ca unele dintre dependențele schemei inițiale să se piardă.

Închiderea unei mulțimi de atribute. Fie R o schemă de relatie, F mulțimea de dependențe asociată și $X \subseteq R$.

Se poate defini ***inchiderea mulțimii de atribute X în raport cu F***(notata X+) astfel:

$$X+ = \{ A \mid X \rightarrow A \in F+ \}$$

X^+ conține deci toate atributele care apar în partea dreaptă a unei dependențe din F sau care se poate deduce din F folosind regulile și axiomele. Pentru calculul lui X+ există un algoritm simplu, prezentat în continuare.

Algoritm de calcul al închiderii mulțimii de atribute
Intrare: R o schema de relație, F mulțimea de dependențe asociată și $X \subseteq R$.
Iesire: X^+
Metoda: se procedează iterativ astfel:

- Se pornește cu $X^{(0)} = X$
- Pentru $i > 1$, $X^{(1)} = X^{(i-1)} \cup \{ A \mid (\exists) Y \rightarrow A \in F \text{ cu } Y \subseteq X^{(i-1)} \}$
- Oprirea se face atunci cand $X^{(i)} = X^{(i-1)}$

Scopul introducerii acestei notiuni este și cel de a putea ocoli in alti algoritmi si definiții calculul lui F^+ . Astfel avem următorul rezultat teoretic: fie R o schemă de relație, F multimea de dependențe asociată și X, $Y \subseteq R$. Atunci $X \rightarrow Y$ se poare deduce din F dacă și numai dacă $Y \in X^{+}$.

Pe baza închiderii mulțimii de atribute, se poate da o altă definiție pentru cheia/supercheia unei relații, bazată nu pe F+ ci pe inchiderea unei mulțimi de atribute.

Definiția cheii : Fie R o schemă de relație, F multimea de dependențe funcționale asociată și $X \subseteq R$. Atunci X este cheie pentru R dacă și numai dacă:

- X+ = R și
- X este minimală: oricare ar fi $Y \subset X$, Y #X atunci Y+ # R (deci orice submulțime strictă a lui X nu mai îndeplinește condiția anterioară).

Dacă numai prima condiție este îndeplinită, atunci X este supercheie pentru R.
Echivalența acestei definitii cu cea anterioară este evidentă:

- $X^+ = R$ inseamnă cf. Propoziției că $X \rightarrow R$
- minimalitatea este de asemenea definită echivalent: NOT ($F \Rightarrow Y \rightarrow R$) este echivalentă NOT ($Y^+ = R$) , adica $Y^+ \# R$

Folosind această definiție se poate defini o euristică de găsire a cheilor unei relații.

2.3.ALGORITM DE DETERMINARE A CHEILOR UNEI RELAȚII

Pentru găsirea cheilor unei relații pornim de la observația că atributele care nu sunt în partea dreaptă a nici unei dependențe nu pot să apară în procesul de închidere a unei mulțimi de atribute și deci ele apartin oricărei chei a relației.

Intrare: R o schemă de relație și F multimea de dependențe funcționale asociată (F în forma canonică).
Iesire: Cheia unica sau cheile alternative ale lui R.
Metoda:

1. Se pornește de la mulțimea de atribute $X \subseteq R$ care nu apar în partea dreaptă a nici unei dependențe

2. Se calculează X^+. Daca X+ = R atunci X este cheia unică minimală a relației R și calculul se oprește aici. Pașii următori se efectuează doar dacă X^+ #R
3. Se adaugă la X câte un atribut din R - X^+ obținându-se o mulțime de chei candidat.
4. Se calculează X^+pentru fiecare dintre candidate. Dacă se obțin toate atributele lui R ,atunci acel X este o cheie a lui R.
5. Se repetă pașii 3 si 4 pornind de la acele mulțimi candidat X care nu sunt găsite ca și chei la pasul anterior. Intre mulțimile candidat nu luăm niciodată în considerare pe cele care conțin o cheie găsită anterior.
6. Procesul se oprește când nu se mai pot face augmentări.

III.PROBLEME REZOLVATE

3.1: Fie R = ABCDE o schemă de relație și F mulțimea de dependențe funcționale asociată, cu F = { AB $\rightarrow$ CDE, C $\rightarrow$ DE } .

Aplicăm regula de descompunere. Obținem:

F = { AB $\rightarrow$ C, AB$\rightarrow$ D, AB $\rightarrow$ E, C $\rightarrow$ D, C $\rightarrow$E }

Mulțimea nu este însă minimală deoarece AB $\rightarrow$D si AB $\rightarrow$ E se pot deduce prin tranzitivitate din AB $\rightarrow$ C împreună cu C $\rightarrow$D, C$\rightarrow$ E.

forma canonică a lui F este:

F = { AB $\rightarrow$ C, C $\rightarrow$ D, C $\rightarrow$ E }

3.2. Fie R = ABCDE și F ={ AB $\rightarrow$C, C $\rightarrow$D, C $\rightarrow$ E }. Atunci AB este cheie pentru R:

- Din AB $\rightarrow$ C, C $\rightarrow$D si C $\rightarrow$ E obținem prin tranzitivitate AB $\rightarrow$ D si AB $\rightarrow$E
- Din AB $\rightarrow$C, AB $\rightarrow$ D si AB $\rightarrow$ E obținem prin reuniune AB $\rightarrow$CDE
- Din AB $\rightarrow$ CDE obtinem (augmentare cu AB) AB $\rightarrow$ ABCDE, deci AB $\rightarrow$R

Rezultă că AB este supercheie pentru R..

Închiderea unei mulțimi de dependențe funcționale F+ a fost introdusă și pentru a putea defini setul de dependențe funcționale moștenite de o schemă de relație obținută prin descompunerea unei scheme incorect proiectată.

3.3. Fie R = ABCD si F = { A$\rightarrow$B, B$\rightarrow$C, C$\rightarrow$D, D$\rightarrow$A }. In cazul in care descompunem R in R1 = AB si R2 = CD atunci:

$F_{R1} = \pi_{R2}$ (F) ={ A$\rightarrow$B, B$\rightarrow$A }

$F_{R2} = \pi_{R1}$ (F) = { C$\rightarrow$D, D$\rightarrow$C }

Observăm însă că dependențele B$\rightarrow$C si D$\rightarrow$A nu mai pot fi obținute nici din F_{R1} nici din F_{R2} nici din reuniunea lor.Va fi prezentată ulterior, o metodă prin care se poate testa daca prin descompunere dependențele inițiale sunt păstrate sau nu.

3.4 Fie R = ABCDE și F = { A $\rightarrow$B, A $\rightarrow$C, D $\rightarrow$E } . Pentru a calcula A+ si AD^+ procedăm astfel:

Calcul A + :

- $X^{(o)}$ = {A}
- Din A $\rightarrow$ B si A $\rightarrow$ C rezultă ca $X^{(1)} = X^{(o)} \cup$ { B, C } = { A } $\cup$ { B, C } = ABC

- Singurele dependențe care au partea dreapta in $X^{(1)}$ sunt tot primele doua deci
  $X^{(2)} = X^{(1)} \cup$ { B, C } = { A, B, C } $\cup$ { B, C } = ABC
- Oprire deoarece $X^{(2)} = X^{(1)}$

Rezulta că $(A)^+$ = ABC

Calcul AD+ :

- $X^{(o)} = \{A, D\}$
- Din $A \rightarrow B$, $A \rightarrow C$ si $D \rightarrow E$ rezultă ca $X^{(1)} = X^{(o)} \cup \{ B, C, E \} = \{ A, D \} \cup \{ B, C , E\} =$ ABCDE

- Oprire deoarece $X^{(1)} = R$ deci oricâte iterații am face nu mai pot să apară noi atribute.
  Rezulta ca $(AD)^+ =$ ABCDE

3.5. Fie R = ABCDE si F = { $A \rightarrow B$, $A \rightarrow C$, $D \rightarrow E$ } .

- Multimea atributelor care nu apar în partea dreapta a nici unei dependențe este X = AD.
- Calculam $(AD)^+$. Obtinem $(AD)^+ =$ ABCDE = R.
- Procesul se oprește. Rezultă ca AD este cheie unică pentru R

3.6 Fie R = ABCDE si F = { $A \rightarrow B$, $B \rightarrow A$, $A \rightarrow C$, $D \rightarrow E$ } .

- Mulțimea atributelor care nu apar în partea dreaptă a nici unei dependențe este X = D.
- Calculăm $(D)^+$. Obținem $(D)^+ =$ DE. Rezulta ca D nu este cheie unică pentru R
- Calculăm mulțimea de candidate: augmentăm D cu atribute din R - $D^+ =$ ABCDE — DE = ABC. Obținem AD, BD și CD
- Calculăm închiderile lor. Obtinem (AD)+ = R, (BD)+ = R și (CD)+ = CDE #R. Rezultă că AD si BD sunt chei ale lui R dar CD nu e cheie.
- Calculăm o nouă mulțime de candidate pornind de ca CD. Putem augmenta CD cu atribute din R - (CD)+ = ABCDE - CDE = AB. Nici una dintre augmentări nu este însă posibila pentru ca atat ACD cat si BCD conțin o cheie găsită anterior (AD respectiv BD).
  - Procesul se oprește. Singurele chei ale lui R rămân AD si BD.

IV.PROBLEME PROPUSE

4.1. Fie schema de relaţie R = ABCDEF, cu mulţimea de dependenţe funcţionale :
F = {A-→B, A-→F, B-→E, D-→B, E-→A }. Se cere :

a. să se determine o cheie minimală pentru R. În mod obligatoriu se va arăta de unde plecaţi în determinarea cheii, se va calcula închiderea acesteia în raport cu F şi se va arăta dacă mai există sau nu alte chei minimale pentru R.

b. Să se găsească pentru R o descompunere ρ, astfel încât fiecare schemă a descompunerii să fie în Forma Normală Boyce-Codd (FNBC). Se va explica modul în care este aplicat algoritmul de descompunere. De asemenea, pentru fiecare sub-schemă se demonstrează că aceasta satisface efectiv definiţia FNBC. În acest scop, pe fiecare pas al aplicării algoritmului de descompunere se vor calcula proiecţiile corespunzătoare ale mulţimii de dependenţe funcţionale valabile pe sub-schema care conduce la pasul curent.

c. Se va analiza conservarea dependenţelor funcţionale iniţiale de către descompunerea ρ obţinută.

d. Pentru aceeaşi schemă de relaţie R şi aceeaşi mulţime de dependenţe funcţionale, să se verifice dacă descompunerea σ = (AB, BCD, AEF, CDE) are proprietatea de joncțiune fără pierderi. Sunteţi rugaţi să explicaţi în ce mod folosiţi algoritmul de verificare a proprietăţiii menţionate.

4.2. Fie schema de relaţie

Biblioteca = (Idlucrare, Copienr, Titlu, Subiect, Numeautor, Datanastautor, Adrautor, Editia, Mediu, Limba, Loc) şi mulţimea de dependenţe funcţionale:

Numeautor → Datanastautor, Adrautor
Idlucrare →Titlu
Idlucrare, Copienr -→Mediu, Editia, Limba
Idlucrare, Copienr -→Loc
Idlucrare → Subiect

Schema de relaţie Biblioteca admite o descompunere în FNBC cu joncţiune fără pierderi şi conservarea dependenţelor funcţionale, cu următoarele sub-scheme:

Autori = (Numeautor, Datanastautor, Adrautor)
Lucrari = (Idlucrare, Titlu, Subiect)
Copii = (Idlucrare, Copienr, Editie, Mediu, Limba, Loc)
Scrisade = (Idlucrare, Copienr, Numeautor).

Se consideră amulţimea de dependenţe DF= {Idlucrare →Numeautor}.
Se cere:

a. să fie găsită o relaţie cu schema Scrisade, care îndeplineşte condiţiile din F dar nu satisface condiţiile din D.
b. Relaţiei de la punctul „a" de mai sus să i se adauge tupluri, astfel încât noua relaţie care se obţine să respecte condiţiile din D.

4.3.Fie date schemele de relaţie:

Studenţi (ids:integer, numes:string, medie:integer, varsta:real)
Cursuri (idc:integer, numec:string, nprofesor:string)
Examene (ids:integer, idc:integer, ziua:date).

Mai sus, au fost subliniate cheile relaţiilor. Se cere :

a. să fie găsite numele studenţilor care au programat examneul la disciplina cu numărul de identificare idc=120.
b. Să se determine numele studenţilor care au planifricat examenul cu profesorul Paul.
c. Să se găsească numele profesorilor care predau cursurile la care va fi examinat Radu.

4.4. Fie schema de relaţie R= ABCDEF, cu mulţimea de dependenţe funcţionale
F = {A→B, A→F, B→E, D→B, E→A}. Se cere să fie determinată o cheie pentru R Găsirea cheii va fi fundamentată mai întâi logic, iar apoi se va calcula închiderea cheii. Să se arate dacă mai există sau nu alte chei pentru R.

4.5. Este dată mulţimea de dependenţe funcţionale F = {A→B, AB→D, ABC→E}. Să se arate dacă din aceste dependenţe funcţionale decurg BC →E şi A→D.

4.6. Fie R(ABCDEFGH) ce satisfice următoarele dependențe funcționale A → B, CH → A, B → E, BD → C, EG → H, DE → F . Care din dependențele următoare sunt deasemenea garantate a fi satisfăcute de relația R?

a. CEG → AB
b. BDG → AE
c. ADE → CH
d. CDE → AF