**Normalizarea relaţiilor**

Obiectivul normalizării relaţiilor este de a valida corectitudinea schemei relaționale a bazei de date. O schemă relațională incorectă conduce la apariția unor efecte nedorite în timpul încărcării, exloatării și întreținerii bazei de date. Eliminarea problemelor care apar în urma procesului inițial de proiectare a bazei de date se face prin normalizare, care constă în descompunerea unei relații (tabel relațional) în mai multe relații, care satisfac anumite reguli, fară pierdere de informație.

Un model relațional este normalizat, dacă îndeplinește anumite constrângeri numite **forme normale**. Formele normale sunt bazate pe dependențele funcționale existente între atribute.

Pentru exemplificare considerăm următoarea relație:

factura (NrFactura, DataFactura, CodClient, DenumireClient, AdresaClient, CodProdus, DenumireProdus, Cantitate, Pret, CodProdus, DenumireProdus, Cantitate, Pret, …).

**Prima formă normala - FN1**

**Definiție.** O relație este în FN1 dacă fiecărui atribut îi corespunde o valoare indivizibilă (atomică), în plus nu trebuie să existe atribute sau grupuri de atribute repetitive.

Relația **factura** nu este în FN1 deoarece *AdresaClient* nu este atribut atomic, în plus există și grupuri de atribute repetitive(*CodProdus, DenumireProdus, Cantitate, Pret*).

**Algoritmul FN1:**

1. se înlocuiesc în relație atributele compuse cu atribute ce conțin componentele elementare ale acestora;
2. se plasează grupurile de atribute repetitive, fiecare grup în câte un o nouă relație;
3. se introduce în fiecare relație nou creată la pasul 2 cheia primară a relației din care a fost extrasă care devine astfel și cheie străină;
4. se stabilește cheia primară a fiecărei noi relații, din cheia straină plus atribute adiționale.

Aplicând algoritmul FN1 pentru relația *factura* obținem:

FacturaFN1 (NrFactura, DataFactura, CodClient, DenumireClient, JudetClient, LocalitateClient, RestAdresa)

DetaliiFactura (#NrFactura, CodProdus, DenumireProdus, Cantitate, Pret)

**A doua formă normală - FN2**

**Definiție**. Fie *R* o relație și fie *K*, *X* două submulțimi de atribute ale lui *R*. Vom spune ca *K* **determină funcțional** pe *X* sau că *X* **depinde funcțional** de *K* dacă nu există două instanțe ale relației *R*, care să aibe aceleași valori pentru atributele din *K* și valori diferite pentru cel putin un atribut din *X*. Cu alte cuvinte dacă exista două instanțe cu aceleași valori pentru atributele din *K* atunci și atributele din *X* au aceleași valori în cele două instanțe. Dependența funcțională se noteaza cu *K**X*. Ea pote fi totală sau parțială.

**Definiție**. Un atribut (sau un ansamblu de atribute) *X* este **dependent funcțional parțial** ansamblul de atribute *K* dacă există subansamblul K1 inclus în *K* cu proprietatea ca *X* este dependent functional de *K1*.

**Definiție**. O relație este în a doua formă normală dacă este în *FN1* și fiecare atribut, care nu face parte din cheia primară este dependent de întreaga cheie primară.

Relația *DetaliiFactura* nu este în *FN2* deoarece *DenumireProdus* depinde parțial de cheia primară, *DenumireProdus* depinde total doar de *CodProdus*.

**Algoritmul FN2:**

1. Pentru fiecare atribut *X* dependent funcțional parțial de cheia *K (KX)* și neinclus în *K* se determină *K1 K*, astfel încât dependența *K1 X* este totală și se crează o nouă relaţie *R1 (K1, X)*;
2. dacă mai există dependente totale *K1Y* în relaţia inițială, se adugă *Y* relaţiei *R1*;
3. *K1* devine cheie primară în noua relaţie și cheie straină în relaţia inițială.

Aplicând algoritmul FN2 relația *DetaliiFactura* devine:

DetaliiFacturaFN2 (#NrFactura, #CodProdus, Cantitate)

Produs (CodProdus, DenumireProdus, Pret)

**A treia forma normală - FN3**

**Definiție**. Fie *R* o relație și fie *X*, *Y* două submulțimi de atribute ale lui *R* și *A* un atribut al lui *R* neinclus în *Y*. Dacă *X* detrmină funcțional pe *Y* și *Y* determină funcțional pe *A* atunci spunem că *A* este **dependent tranzitiv** de *X*.

**Definiție**. O relație *R* este în FN3 dacă este în FN2 și orice atribut neconținut în cheia primară a lui *R* depinde direct și nu tranzitiv de cheia primară a lui *R*.

FN3 elimină dependentele tranzitive.

**Alg. FN3:**

1. Pentru fiecare dependență funcțională *KXY* a relației *R, K* și *X* nu neapărat disjuncte, se construiește o nouă relație *R1(X, Y);*
2. *X* devine cheie primară în noua relaţie;
3. *Y* se elimină din *R* și *X* devine cheie straină în *R*.

De exemplu relația *FacturaFN1* nu este în *FN3* deoarece *DenumireClient* depinde de *CodClient* şi prin urmare tranzitiv de *NrFactura*. Aplicând algoritmul FN3 relația *FacturaFN1* se transformă în:

FacturaFN3(NrFactura, DataFactura, #CodClient)

Client (CodClient, Denumire, JudetClient, LocalitateClient, RestAdresa)

**Forma normală Boyce-Codd - FNBC.**

Dacă FN3 elimină dependențele funcționale parțiale și tranzitive pentru cheia primară, forma normală Boyce-Codd ia în considerare celelalte chei posibile(chei alternante) în cazul în care acestea există.

Remarcă: Dacă o relație nu conține chei alternante și este în FN3, atunci este de asemnea în *FNBC*.

**Definiție**: O relație *R* este în FNBC dacă este în *FN3* și niciun atribut ce compune cheia primară a relației *R* nu depinde funcțional de un alt atribut neinnclus în cheia primară.

O relație este în *FNBC* dacă și numai dacă singurele dependențe funcționale sunt cele în care atributele non-cheie sunt determinate funcțional de cheia primară a relației.

Această formă normală elimină o serie de lacune ale relațiilor ce sunt în *FN3* dar în care există un atribut non-cheie care determină funcțional o parte a cheii primare.

Exemplu: Se dorește realizarea unei statistici a echipelor campioane la fotbal din europa. Propunem pentru modelarea acestei probleme schema relațională următoare:

Campioane(Tara, An, Echipa, Buget)

|  |  |  |  |
| --- | --- | --- | --- |
| Tara | An | Echipa | Buget |
| Spania | 2014 | FCB | 8 |
| Spania | 2015 | FCB | 7 |
| … | … | … | … |

Relația are două chei candidat (Tara*,* An) și (An*,* Echipa).

Cheia primară aleasă este (Tara*,* An).

Putem remarca faptul că schema relației *Campioane* este în *FN3* dar nu este în FN2 în raport cu cheia alternantă *(An, Echipa)* deoarece atributul *Tara* depinde parțial și nu total de cheia alternantă *(An, Echipa).*

Dependențele funcționale ale relației sunt:

Algoritmul FNCB presupune aplicarea succesivă a algoritmului FN2 pentru fiecare cheie candidat care va deveni cheie primară.

Pentru exemplul analizat, vom obține următoarea descompunere

R1(Echipa, Tara)

R2(An, Echipa, Buget)

**A patra formă normală - FN4**

FN4, ca și FN5, operează asupra cheilor compuse pentru a înlătura redundanțele generate de dependențele nonfuncționale dintre atributele cheilor candidat.

**Definiție**. Definiție: Fie *R* *(X, Y, Z)* o schemă relație.

Spunem că există o **dependență multivaloare** *Y* de *X* sau că *X* **determină multivaloare** pe *Y* și notăm *X →→Y* dacă și numai dacă pentru orice tupluri (x, y, z) și (x, y', z') din *R* vom avea de asemenea în R și tuplurile (x, y, z') și (x, y', z).

Notă: Se observă că dacă *X →→Y* atunci *X →→Z*.

Cu alte cuvinte spunem că *X* determină multivaloare pe *Y (X →→Y)* dacă unei valori a lui *X* îi sunt asociate mai multe valori din *Y* independente(necorelate în niciun fel) de valorile atributelor .

Exemplu:

Fie relaţia R(Curs,GrupaStudenti,Tema):

|  |  |  |
| --- | --- | --- |
| CURS | GrupaStudenti | TEMA |
| Programare | Grupa1 | Tema1 |
| Programare | Grupa1 | Tema 2 |
| Programare | Grupa1 | Tema 3 |
| Programare | Grupa2 | Tema 1 |
| Programare | Grupa2 | Tema 2 |
| Programare | Grupa2 | Tema3 |

Avem dependențele multivaloare CURS→→GrupaStudenti și CURS→→ TEMA.

Curs

GrupaStudenti

Tema

n

n

n

n

Se observă că apar anomalii legate de redundanță, adăugare și stergerea de rânduri.

**Definiție**. O relație R este în FN4 dacă și numai dacă este în FNBC și nu conține două sau mai multe dependențe multivaloare.

FN4 elimină redundanțele datorate relațiilor independente de tip n:n ce apar între atributele ce formează cheia compusă.

**Algoritmul FN4**

Se identifică o dependență multivaloare X→→Y. Se înlocuiește relația R cu două relații

R1(X,Y)

R2=R\Y.

Dacă relațiile rezultate conțin dependențe multivaloare se reia Algoritmul FN4.

Aplicând Algoritmul FN4 relației R(Curs, GrupaStudenti, Tema), obținem:

R1(Curs, GrupaStudenti)

R2(Curs, Tema).

**A cincea formă normală FN5**

**Definiție**. Fie R (X, Y, Z) o schemă relație și R1(X,Y)=∏X,Y(R), R2(X,Z)=∏X,Z(R), R3(Y,Z)=∏Y,Z(R),

Dacă:

spunem că *R* satisface **dependența de uniune**(joncţiune, join dependency).

Exemplu:

R:

|  |  |  |
| --- | --- | --- |
| FIRMA | AGENT | PRODUS |
| F1 | A1 | P1 |
| F1 | A1 | P2 |
| F2 | A1 | P1 |
| F2 | A2 | P1 |
|  |  |  |

Dacă o firmă este reprezentată de un agent de vânzări și această firmă propune la vânzare un nou produs, atunci agentul trebuie să vândă și acest nou produs pentru firma producătoare.

Firmă

Agent

Produs

n

n

n

n

n

n

FN5 elimină redundanțele generate de dependențele de tip n:n ce apar între atributele cheii.

Eliminarea redundanțelor se face prin descompunerea prin proiecție în trei relații:

R1(X,Y)=∏X,Y(R), R2(X,Z)=∏X,Z(R), R3(Y,Z)=∏Y,Z(R).

Pentru exemplul de mai sus avem descompunerea:

R1 :

|  |  |
| --- | --- |
| FIRMA | AGENT |
| F1 | A1 |
| F2 | A1 |
| F2 | A2 |

R2:

|  |  |
| --- | --- |
| FIRMA | PRODUS |
| F1 | P1 |
| F1 | P2 |
| F2 | P1 |

R3 :

|  |  |
| --- | --- |
| AGENT | PRODUS |
| A1 | P1 |
| A1 | P2 |
| A2 | P1 |

Relația *R* se poate obține prin compunerea celor trei proiecții.

Dependenţa de uniune este proprietatea ce garantează că nu se generează înregistrări false la reunirea prin compunere naturală a relaţiilor obţinute prin descompunere.

Dacă *X* reprezintă atributele comune relațiilor *R1* și *R2*, atunci compunerea naturală a celor două relații se efectuează astfel:

Se calculează produsul cartezian *R1xR2*

Se selectează acele tupluri din *R1xR2* pentru care valoarea lui *X* din *R1* este identică cu valoarea lui *X* din *R2*.