EXEMPLU DE PROIECTARE BD EXAMENE STUDENŢI FOLOSIND NORMALIZAREA

1. *Nevoia de normalizare*

Unul dintre cele mai importante argumente în favoarea normalizării este evidenţierea a ceea ce s-ar întâmpla dacă ar lipsi. Să luăm ca exemplu tabela 1, dedicată stocării datelor referitoare la studenţii Facultăţii de Informatică (sau oricare alta), mai ales în ceea ce priveşte situaţia la învăţătură a acestora.
Fiecare student are următoarele caracteristici:

- Posedă un identificator unic **Nr_Matricol**;
- Este înscris la o specializare **Specializare,** într-un an de studii **An;**
- Are un nume şi un prenume, **NumePrenume;**
- Susţine examene la disciplinele din planul de învăţământ, fiecare disciplină având un nume **DenumireDisc** şi un cod unic de identificare **CodDisc**;
- Orice disciplină are asociat un număr de credite prin planul de învăţământ al specializării **NrCredite;**
- Fiecare student poate susţine un examen de mai multe ori până îl promovează (sau este exmatriculat), atributele referitoare la acesta fiind **DataExamen** şi **Nota.** De exemplu, Popa Ion a picat examenul la Baze de date în prima sesiune (pe 29 ianuarie 2020), dar l-a luat în sesiunea de restanţe (pe 20 iunie 2020).

Tabelul 1. Relaţia STUDENTI_EXAMENE supusă tehnicii de normalizare

Nr_Matricol#	NumePrenume	An	Specializare	CodDisc#	DenumireDisc	NrCredite	DataExam#	Nota
IN13455	Popovici I. Vasile	3	Informatică	DI3501	Baze de date	6	29/01/2019	7
IN13456	Popa V. Ion	3	Informatică	DI3501	Baze de date	6	29/01/2022	**4**
IN13457	Ciutac C. Liviu	3	Informatică	DI3501	Baze de date	6	29/01/2019	8
IN13455	Popovici I. Vasile	3	Informatică	DI3502	Programare web	7	01/02/2019	9
IN13456	Popa V. Ion	3	Informatică	DI3502	Programare web	7	01/02/2019	6
IN13457	Ciutac C. Liviu	3	Informatică	DI3502	Programare web	7	01/02/2019	4
IN13456	Popa V. Ion	3	Informatică	DI3501	Baze de date	6	20/06/2022	**6**
IN13457	Ciutac C. Liviu	3	Informatică	DI3502	Programare web	7	01/02/2019	10
IN13458	Spanu A. Maria	3	Informatică	DI3503	Programare O.O.	8	04/02/2019	8
…….								
…….								

Întrucât fiecare linie se referă la la un examen susţinut de către un student la o anumită disciplină şi la o anumită dată, cheia primară a relaţiei este combinaţia (**Nr_Matricol**, **CodDisc**, **DataExam**).

Redundanţe

Este uşor de observat că relaţia de mai sus conţine redundanţe. Astfel, dacă un student susţine pe parcursul celor 6 semestre de studiu un număr de examene(să spunem 30), atunci numărul matricol, numele, anul şi specializarea apar în toate cele 30 de linii. Dacă pentru o disciplină s-au înregistrat în baza de date un număr oarecare de examinări, atunci nu numai codul, ci şi denumirea şi numărul de credite alocat acesteia apar de acelaţi număr de ori.

Cu cât baza de date este mai mare, cu atât spaţiul ocupat este mai mare (risipa de spaţiu). Chiar dacă spaţiul de stocare nu mai este o problemă din punct de vedere al costului, dimensiunea mare a unei tabele ca cea de mai sus poate genera probleme serioase de viteză în exploatare.

Anomalii la inserare

Dacă ne vom referi la studenţii din anul I, după Admitere (din iulie şi septembrie) aceştia sunt înmatriculaţi; dacă baza de date are schema relaţiei de mai sus, preluarea este imposibilă până în momentul primului examen susţinut de studentul respectiv. Aceasta deoarece în cheia primară sunt incluse atributele **CodDisc** şi **DataExam**, iar modelul relaţional interzice valorile nule pentru atributele componente(restricţia de entitate). Ori, la data înmatriculării, studenţii anului I nu au nici un examen susţinut, iar prima sesiune este abia în luna ianuarie a anului viitor, aşa că şi **CodDisc** şi **DataExam** sunt în acel moment NULL.

Anomalii la modificare

Presupunem că într-o şedinţă a consiliului facultăţii se hotărăşte ca disciplina *Programare web* să fie redenumită *Programare utilizând tehnologii web*, titulatură sub care va apărea şi în foile matricole ale studenţilor din actualul an 3 care vor absolvi specializarea Informatică. În baza de date sunt sute de înregistrări corespunzătoare studenţilor examinaţi la această disciplină şi toate trebuie modificate. Dacă, din diferite motive, modificarea se face numai pe o parte dintre liniile implicate, datele îşi pierd consistenţa.

După O'Neil & O'Neil[O'Neil, P., O'Neil, E., *Database. Principles, Programming, Performance*, Morgan Kaufmann, San Francisco, 2011] , avem de-a face cu o anomalie de modificare într-o relaţie dacă modificarea valorii unui atribut (aici **DenumireDisc**) atrage obligativitatea actualizării aceleiaşi valori pe mai multe linii.

Anomalii la ştergere

Anomaliile la ştergere apar când prin eliminarea unor linii dintr-o tabelă se pierd involuntar nu numai informaţiile despre entitatea reflectată pe linia respectivă, ci şi alte informaţii. Astfel, dacă ştergem ultima linie din relaţia din tabelul 1, care conţine nota examenului la *Programare O.O.* susţinut de studenta Spanu A. Maria pe 4 februarie 2019, se pierd nu numai datele despre examenul respectiv, ci şi toate informaţiile despre studenta Spanu A. Maria, pentru că este singura linie în care apărea studenta.

Normalizarea îşi propune ca principale obiective:

- Minimizarea spaţiului necesar stocării datelor;
- Minimizarea riscului apariţiei de date inconsistente în cadrul BD;
- Minimizarea anomaliilor ce pot apărea la actualizare (inserarea datelor, dar mai ales la modificare şi ştergere);
- Îmbunătăţirea structurii bazei, reprezentarea diferitelor conexiuni dintre atributele bazei;
- Diminuarea nevoii de reorganizare periodică a modelului.

1.1 Prima formă normalizată(FN1)

Dintre toate formele normalizate (cinci), doar **prima are caracter de obligativitate**. Celelalte forme normale sunt dezirabile pentru diminuarea redundanţelor, reducerea spaţiului ocupat pe disc sau eliminarea anomaliilor manifestate la actualizare, dar nu sunt obligatorii. Prima etapă a normalizării constă în identificarea informaţiilor ce trebuie stocate şi gestionate cu ajutorul bazei de date. Este momentul în care la aplicaţiile complexe intră în scenă **analiştii de sistem**, care observă cum se derulează procesele, operaţiunile ce constituie obiectul aplicaţiei/bazei de date, discută cu toate categoriile de utilizatori, încearcă să identifice soluţii anterioare de la aceeaşi organizaţie sau de la organizaţii similare pentru a putea anticipa problemele cele mai dificile etc. În urma activităţii de analiză va rezulta relaţia universală **R**, care va fi alcătuită din toate atributele identificate şi considerate ca relevante pentru aplicaţie, împreună cu toate tuplurile posibile cu valorile acestor atribute. Prin urmare, **schema relaţiei universale conţine toate atributele bazei de date**.

Etapele de aducere a unei relaţii în forma FN1 sunt:

I. Se elimină atributele compuse (se atomizează) şi se construieşte câte o relaţie pentru fiecare grup repetitiv (dacă apar asemenea grupuri);
II. În schema fiecărei noi relaţii obţinute la pasul I se introduce şi cheia primară a relaţiei R (iniţială) nenormalizate;
III. Cheia primară a fiecărei noi relaţii va fi compusă din atributele chei ale relaţiei R, plus unul sau mai multe atribute proprii.

Se observă că relaţia noastră iniţială (notată cu **R** în paşii de mai sus) **STUDENTI_EXAMENE** este deja în FN1, pentru că nu avem atribute compuse şi nici grupuri repetitive. Avem însă redundanţe şi anomalii la actualizare. De aceea vom continua cu procesul de normalizare.

1.2 A doua formă normalizată(FN2)

FN2 este în strânsă legătură cu noţiunea de dependenţă funcţională(DF). *Dependenţele funcţionale* reprezintă dependenţa între date prin care se poate identifica un atribut/grup de atribute prin intermediul altui atribut/grup de atribute.

Dacă X şi Y sunt două subansamble de atribute ale atributelor relaţiei R, spunem că între X şi Y există o *dependenţă funcţională*, notată $X \rightarrow Y$, dacă şi numai dacă:

(i) fiecare valoare a lui X poate fi asociată unei singure valori din Y, şi

(ii) două valori distincte ale lui X nu pot fi asociate aceleiaşi valori ale lui Y.

X se numeşte *determinantul* (*sursa*) dependenţei, iar Y se numeşte *determinatul* (*destinaţia*) dependenţei.

Exemple: Următoarele atribute se află în dependenţă funcţională:

- cod_poştal→localitate, deoarece unui cod poştal îi corespunde o singură localitate (sensul dependenţei este foarte important, deoarece, viceversa ar însemna: o localitate are un singur cod_poştal);
- nr_factură→data_factură, deoarece cunoaşterea numărului facturii determină cu exactitate data facturii.

Două atribute *nu sunt în dependenţă funcţională*, notată X $\not\rightarrow$ Y, atunci când cunoaşterea unei valori a primului atribut fie nu permite cunoaşterea nici uneia dintre valorile celui de al doilea atribut, fie permite cunoaşterea mai multor valori ale celui de al doilea atribut.

Exemplu: Următorul atribut nu se află în dependenţă funcţională: **c**np $\not\rightarrow$ nr_factură, deoarece pentru o persoană se pot întocmi mai multe facturi (aferente fiecărei vânzări).

În cadrul modelului relaţional, o dependenţă funcţională este reprezentată printr-o săgeată ce porneşte din sursă şi se termină în destinaţie.

Revenind, o relaţie se află în FN2 dacă:

1. Se află în FN1;
2. Fiecare atribut care nu face parte din cheie este dependent de întreaga cheie primară (nu există DF parţiale între atributele cheie şi celelalte atribute).

Problema trecerii unei relaţii din FN1 la FN2 se pune numai dacă cheia primară a relaţiei este compusă din mai multe atribute.

Etapele de aducere a unei relaţii din forma FN1 în FN2 sunt:

I. Se identifică posibila cheie primară a relaţiei universale aflată în FN1;
II. Se identifică toate dependenţele dintre atributele relaţiei, cu excepţia acelora în care destinaţia este un atribut component al cheii primare;
III. Se identifică toate dependenţele care au ca sursă un atribut sau subansamblu de atribute din cheia primară;
IV. Pentru fiecare atribut (sau subansamblu) al cheii de la pasul III se creează o relaţie care va avea cheia primară atributul (subansamblul) respectiv, iar celelalte atribute vor fi cele care apar ca destinaţie în dependenţele de la etapa III.
V. Din relaţia iniţială sunt eliminate toate atributele destinaţie (noncheie) ale DF găsite la pasul III.

În cazul relaţiei **STUDENTI_EXAMENE** aflată deja în FN1, cheia primară fiind combinaţia (**Nr_Matricol**, **CodDisc**, **DataExam**), apar automat următoarele dependenţe funcţionale (DF) totale:

(1) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ NumePrenume
(2) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ An
(3) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ Specializare
(4) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ DenumireDisc
(5) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ NrCredite
(6) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ Nota

Relaţia **STUDENTI_EXAMENE** ar fi în FN2 dacă nici una din aceste 6 dependenţe nu ar fi parţială. Fără a face un efort prea mare de gândire, putem să afirmăm că numărul matricol **Nr_Matricol** este desemnat să identifice în mod unic fiecare student. De aici apar trei dependenţe funcţionale parţiale:

(7) **Nr_Matricol** ⟶ NumePrenume
(8) **Nr_Matricol** ⟶ An
(9) **Nr_Matricol** ⟶ Specializare

Şi codul unei discipline **CodDisc** identifică cu certitudine fiecare "materie" parcursă de studenţi. Vom mai adăuga următoarele două dependenţe funcţionale parţiale:

(10) **CodDisc** ⟶ DenumireDisc
(11) **CodDisc** ⟶ NrCredite

Din DF (7) – (9) construim relaţia **STUDENTI** [**Nr_Matricol#**, NumePrenume, An, Specializare], iar din DF (10) şi (11) construim relaţia **DISCIPLINE** [**CodDisc#**, DenumireDisc, NrCredite].
Din relaţia iniţială **STUDENTI_EXAMENE** [**Nr_Matricol#**, **CodDisc#**, **DataExam#**, NumePrenume, An, Specializare, DenumireDisc, NrCredite, Nota], rămâne relaţia **EXAMENE** [**Nr_Matricol#**, **CodDisc#**, **DataExam#**, Nota].

La acest moment, **redundanţele** au fost eliminate:

- numele, specializarea şi anul de studiu al fiecărui student apar o singură dată (în tuplul corespunzător studentului din relaţia **STUDENTI**);
- tot o singură dată apar şi denumirea unei discipline şi numărul de credite alocat ei (vezi relaţia **DISCIPLINE**).

Lucrurile stau la fel de bine şi în ceea ce priveşte **diminuarea anomaliilor manifestate la actualizarea datelor**:

- *anomaliile de inserare* — din momentul înscrierii în anul I, până la cel al susţinerii primului examen, studentul poate fi inserat în tabela **STUDENTI**, urmând ca, odată cu participarea la prima sesiune de examene, să-i fie introduse înregistrări în tabela **EXAMENE**.
- *anomaliile de modificare*— modificarea denumirii unei discipline sau a numărului de credite sau corectarea unei eventuale erori apărute în numele studentului sau al specializării/anului, toate se fac într-un singur loc, pe tuplul corespunzător din relaţia **STUDENTI** sau **DISCIPLINE**, ceea ce înseamnă că s-au rezolvat şi anomaliile de modificare.
- *anomaliile la ştergere*— se atrăgea atenţia asupra riscului de-a pierde informaţii utile la ştergerea unei linii din relaţia iniţială **STUDENTI_EXAMENE**; nici acest pericol nu mai este de actualitate în noua structură (putem şterge notele unui student, fără a şterge şi studentul, operând doar în tabela EXAMENE, evident dacă are legal un asemenea demers).

1.3 A treia formă normalizată(FN3)

O relaţie este în *forma normală trei FN3* dacă:

1. se găseşte în FN2 şi
2. fiecare atribut care nu este cheie (nu participă la o cheie) depinde direct de cheia primară.

A treia regulă de normalizare cere ca toate câmpurile din tabele să fie independente între ele.

Etapele de aducere a unei relaţii de la FN_2 la FN_3 sunt:

I. Se identifică toate atributele ce nu fac parte din cheia primară şi sunt surse ale unor dependenţe funcţionale;
II. Pentru aceste atribute, se construieşte câte o relaţie în care cheia primară va fi atributul respectiv, iar celelalte atribute, destinaţiile din DF considerate;
III. Din relaţia de la care s-a pornit se elimină atributele destinaţie din DF identificată la pasul I, păstrându-se atributele surse.

Pentru a exemplifica FN3 şi a nu complica structura relaţiei iniţiale **STUDENTI_EXAMENE,** presupunem că după ce am ajuns cu normalizarea în FN2, analistul vine şi afirmă că trebuie să stocăm în baza de date şi informaţiile legate de profesorii cu care studenţii Facultăţii de Informatică susţin examenele şi a grupelor din care studenţii fac parte (a omis aceste cerinţe). Se impune să introducem în relaţia iniţială **STUDENTI_EXAMENE** noi atribute:

Grupa --- Grupa de studiu a unui student;
CodProf --- Codul profesorului;
NumeProf --- Numele profesorului;
GradProf --- Gradul didactic al profesorului:
P- profesor universitar; C- conferenţiar, L- lector,
A- asistent, R- preparator.

SalaEx --- Sala de desfăşurare a examenului.

Dacă am reface normalizările FN1 şi FN2, pentru relaţia universală (iniţială) modificată, pentru a reflecta noua structură, **STUDENTI_EXAMENE** [**Nr_Matricol#**, **CodDisc#**, **DataExam#**, NumePrenume, An, Specializare, DenumireDisc, NrCredite, Nota, CodProf, NumeProf, GradProf, SalaEx] , vor apărea la FN2 (vezi 1.2) următoarele DF totale:

(1) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ NumePrenume
(2) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ An
(3) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ Specializare
(4) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ DenumireDisc
(5) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ NrCredite
(6) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ Nota
(7) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ Grupa
(8) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ CodProf
(9) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ NumeProf
(10) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ GradProf
(11) (**Nr_Matricol**, **CodDisc**, **DataExam**) ⟶ SalaEx

Relaţia **STUDENTI_EXAMENE nouă** este în FN1. Pentru a demonstra că nu este în FN2 trebuie să găsim în ansamblul DF (1) – (11) măcar o DF parţială.

Acestea vor fi:

(12) **Nr_Matricol** ⟶ NumePrenume
(13) **Nr_Matricol** ⟶ An
(14) **Nr_Matricol** ⟶ Specializare

(15) **Nr_Matricol** ⟶ Grupa
(16) **CodDisc** ⟶ DenumireDisc
(17) **CodDisc** ⟶ NrCredite
Aplicând paşii descrişi la 1.2, pentru DF (12) – (17), se vor obţine relaţiile următoare aflate în FN2, din relaţia universală **STUDENTI_EXAMENE**, cu structura de mai sus :

STUDENTI [**Nr_Matricol#**, NumePrenume, An, Grupa, Specializare]
DISCIPLINE [**CodDisc#**, DenumireDisc, NrCredite]
EXAMENE [**Nr_Matricol#**, **CodDisc#**, **DataExam#**, SalaEx, Nota, CodProf, NumeProf, GradProf].

Cele trei relaţii sunt în FN2. Aducerea în FN3 echivalează cu eliminarea dependenţelor tranzitive din relaţiile **STUDENTI**, **DISCIPLINE** şi **EXAMENE**.
În relaţia **STUDENTI**, singurele DF existente sunt (12) - (15), adică DF totale, deci această relaţie este în FN3.
Nici relaţia **DISCIPLINE** nu conţine DF tranzitive, aşadar şi ea este în FN3.
În schimb relaţia **EXAMENE** conţine DF tranzitive. CodProf identifică în mod unic un profesor, aşa că vor apărea DF:
(18) CodProf ⟶ NumeProf
(19) CodProf ⟶ GradProf

Cum CodProf, NumeProf şi GradProf nu fac parte din cheia primară, aplicând 1.3, punctul II, rezultă că relaţia **EXAMENE** se descompune în două relaţii în FN3:

PROFESORI [**CodProf#,** NumeProf, GradProf] şi
NOTE [**Nr_Matricol#**, **CodDisc#**, **DataExam#**, SalaEx, Nota, CodProf].
Astfel, în FN3, relaţia universală **STUDENTI_EXAMENE** are o schemă alcătuită din patru relaţii, după cum urmează:
STUDENTI [**Nr_Matricol#**, NumePrenume, An, Grupa, Specializare],
DISCIPLINE [**CodDisc#**, DenumireDisc, NrCredite],
PROFESORI [**CodProf#,** NumeProf, GradProf] şi
NOTE [**Nr_Matricol#**, **CodDisc#**, **DataExam#**, SalaEx, Nota, CodProf].

Diagrama E-R a modelului conceptual va arăta ca în figura 1: