

Baze de date - proiect

UNIVERSITATEA DIN CRAIOVA
Facultatea de Automatica, Calculatoare si Electronica
Specializarea ----Automatica si informatica aplicata -----
GrupaAIA 2.1 si 2.2.....

PROIECT la disciplina BAZE DE DATE

Videoteca Multimedia



Echipa DOA
Membri echipei
1. Petrov Daniela
2. Militaru Alexandru-Ionut
3.Gheorghe Stefan

Coordonator,

Prof. Dr. Ing. Viorel Stoian

Craiova, 2024

UNIVERSITATEA DIN CRAIOVA

Facultatea de Automatica, Calculatoare si Electronica

Specializarea ----Automatica si informatica aplicata -----

GrupaAIA 2.1 si 2.2.....

PROIECT la disciplina BAZE DE DATE

Videoteca Multimedia



Echipa DOA

Membri echipei

1. Petrov Daniela
2. Militaru Alexandru-Ionut
- 3.Gheorghe Stefan

Coordonator,

Prof. Dr. Ing. Viorel Stoian

Craiova, 2024

Cuprins

1. INTRODUCERE
2. TEMA DE PROIECT
3. SCHEMA CONCEPTUALA
 - 3.1. Notiuni teoretice
 - 3.2. Schema conceptuala
4. SCHEMA LOGICA
 - 4.1. Notiuni teoretice
 - 4.2. Schema logica
5. NORMALIZAREA BAZEI DE DATE
 - 5.1. Notiuni teoretice
 - 5.2. Normalizarea bazei de date
6. DENORMALIZAREA BAZEI DE DATE
 - 6.1. Notiuni teoretice
 - 6.2. Denormalizarea bazei de date
7. SGBD UTILIZAT
 - 7.1. Notiuni teoretic
 - 7.2. Exemple de interogari SQL pe baza de date creata
8. CONCLUZII
9. BIBLIOGRAFIE

TEMA DE PROIECT

Whatpal este o platforma de streaming si planificare a materialelor video din intreaga lume. Ceea ce diferențieaza platforma noastra de altele este o eficienta baza de date ce ne permite manevrarea a mii de gb de date pentru beneficiul utilizatorilor nostrii/



3. SCHEMA CONCEPTUALA

Notiuni teoretice (3)

În prima fază, o echipă nominalizată colectează (achiziționează) datele corespunzatoare din sistem, apoi urmează faza de organizare a acestora utilizându-se modelul entitate-legătură. Principalele concepte folosite în acest model sunt: entitatea, relația (legătura) și atributul.

Entitatea este un obiect de interes din sistem pentru care trebuie să existe date înregistrate.

Observații:

- Fiecare entitate are o denumire unică în cadrul unui sistem.
- Entitățile sunt reprezentate prin substantive, dar nu orice substantiv folosit în descrierea sistemului este entitate, ci numai acelea care au o semnificație deosebită.
- Fiecare entitate trebuie să fie bine definită și precizată pentru a se evita confuziile.

Relația (legătura) este o asociere (raport) nedirecționată între 2 entități.

Cardinalitatea unei relații indică numarul de instanțe din fiecare entitate care poate participa la relație. Există 3 tipuri de cardinalitate:

- "multî-la-unu" (many-to-one, M:1).

Relația dintre entitățile A și B este de tipul "multî-la-unu" dacă fiecarei instanțe din A îi se poate asocia cel mult o singură instanță din B și fiecarei instanțe din B îi se pot asocia mai multe instanțe din A.

Ex.: "studiază-la"

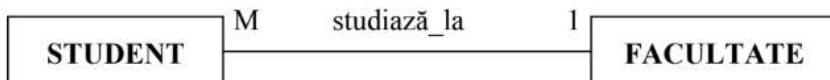


Fig. 2.2. Exemplu de relație "multî-la-unu" (M:1)

- "unu-la-unu" (one-to-one, 1:1).

Relația dintre entitățile A și B este de tipul "unu-la-unu" dacă fiecarei instanțe din A îi se poate asocia cel mult o singură instanță din B și fiecarei instanțe din B îi se poate asocia cel mult o singură instanță din A.

Ex.: "conduce"

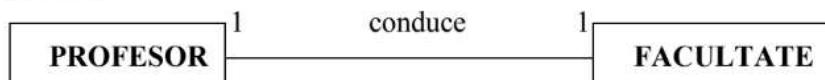


Fig. 2.3. Exemplu de relație "unu-la-unu" (1:1)

- **"mulți-la-mulți"** (**many-to-many**, **M:M**).

Relația dintre entitățile A și B este de tipul "mulți-la-unu" dacă fiecărei instanțe din A i se pot asocia mai multe instanțe din B și fiecărei instanțe din B i se pot asocia mai multe instanțe din A.

Ex.: "predă", "urmează".

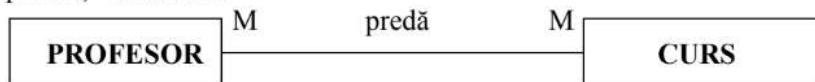


Fig. 2.4. Exemplu de relație "mulți-la-mulți" (M:M)

Atributul este o caracteristică a unei entități sau a unei relații. Fiecare entitate are un anumit număr de atribută despre care sunt înregistrate date.

Observații:

- Numele unui atribut este unic în cadrul unei entități sau al unei relații.
- Atributele sunt întotdeauna substantive, dar nu orice substantiv este atribut.
- Pentru fiecare atribut este necesară o descriere, împreună cu domeniul de valori (întreg, sir de caractere, dată calendaristică etc.).
- Trebuie evitate atributele indirecte.

Chei primare, naturale, artificiale

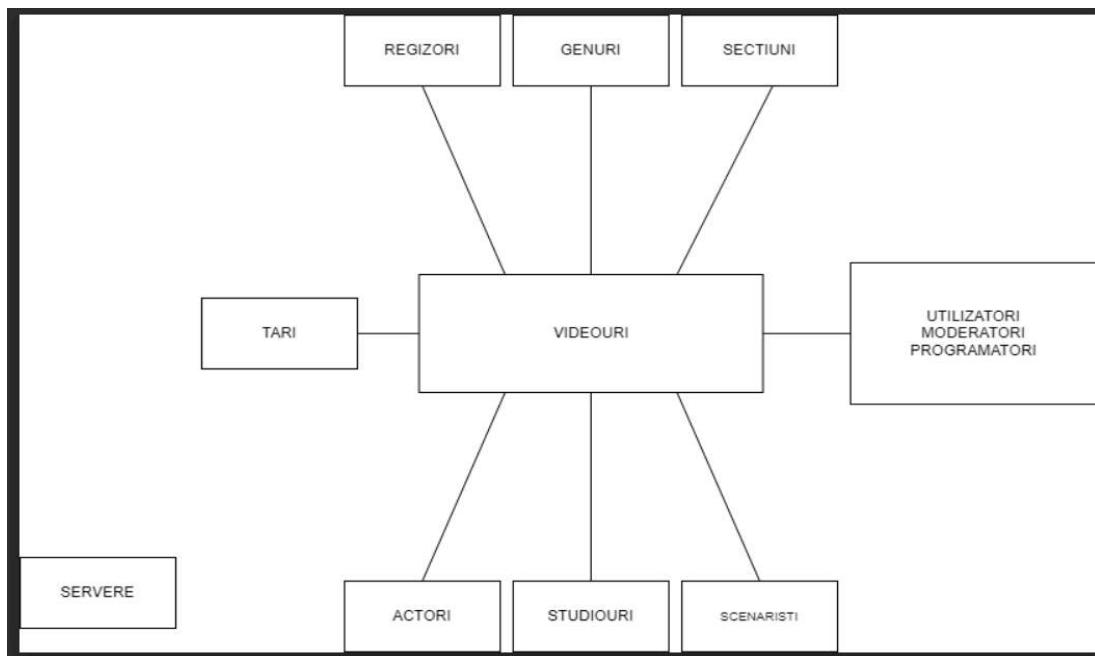
Cheia unei entități este un atribut sau set de atribută care identifică în mod unic o instanță a acelei entități (face distincție între oricare 2 rânduri diferite ale tabelului asociat entității).

Cheile sunt de 2 feluri: naturale (au semnificație reală pentru entitate, ex.: (nume, prenume, data_nașterii)) și artificiale (nu au semnificație reală pentru entitate, ex.: cod_student, cod_facultate).

Avantajele cheilor artificiale primare:

- stabilitatea – Cheile primare artificiale se schimbă rar. Schimbarea unei chei primare presupune schimbarea cheilor străine care fac referire la ea.
- simplitatea – Au un număr de atribută și de caractere mai mic.
- Nu prezintă ambiguități în reprezentare (ex.: liniuțe, spații etc.).
- Elimină apariția valorilor Null. (1)

Schema conceptuala



SCHEMA LOGICA

Notiuni teoretice (1)

Pentru realizarea schemei logice a unei baze de date se pornește de la scheme conceptuală (modelul entitate – legătură) urmărindu-se conversia entităților și a legăturilor în tabele relaționale.

Regulile de conversie ale entităților, legăturilor și atributelor sunt următoarele:

Transformarea entităților

Regulă generală: entitățile se transformă în tabele.

Subcazuri:

a) Entitățile independente devin tabele independente, adică tabele a căror cheie primară nu conține chei străine.

Ex.: Entitatea "STUDENT" devine tabelul "STUDENT" cu cheia primară cod_student.

b) Entitățile dependente devin tabele dependente (tabele detaliu) adică tabele a căror cheie primară conține cheia străină ce face referință la cheia primară a entitatii de care depinde entitatea in cauza.

Ex.: Entitatea "MODUL" devine tabelul "MODUL" a carui cheie primara este formata din cod_curs (care este o cheie straina pentru entitatea "CURS") si nr_modul.

c) Subentitatile devin subtabele, adica tabele a caror cheie primara este cheia straina pentru tabelul superentitate.

Ex.: Subentitatea "PROFESOR" va avea cheia primara cod_personal, aceeasi cu cea a superentitatii "PERSONAL".

- Avantajele supertabelelor: simplificarea programelor de manipulare a datelor.

- Dezavantajele supertabelelor: probleme de integritate, apar valori de Null.

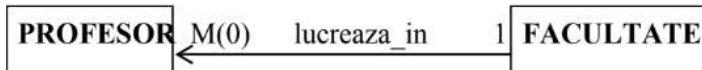
- Avantajele subtabelelor: mai stabile, mai flexibile, ocupa spatiu mai mic, contin mai putine valori de Null.

- Dezavantajele subtabelelor: se ingreuneaza manipularea datelor.

Transformarea relatiilor (legaturilor)

- b) Relatiile M:1 devin chei straine plasate in tabelul care se afla in partea de “multi” a relatiei.

Ex.:



Cazuri:



Cheia strana nu poate avea valoarea Null, iar in cazul entitatilor dependente ea va face parte chiar din cheia primara a tabelului detaliu.



Fig. 2.18

Cheia strana poate avea valoarea Null si nu poate face parte din cheia primara.

- c) O relatie M:M se transforma in 2 relatii M:1. In acest caz se construieste un tabel special numit *tabel asociativ* care are 2 chei straine care fac referinta la cheile primare ale celor 2 tabele aflate in relatia M:M. Cheia sa primara este formata din cele 2 chei straine plus (eventual) alte atribute suplimentare.

Ex.:

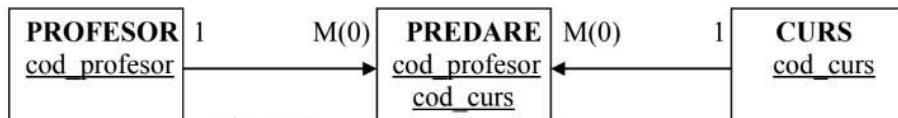
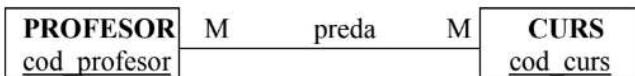


Fig. 2.19

- d) O relatie de tip 3 se transforma intr-un numar de relatii de tip 2, egal cu numarul de tabele asociate. Aceste relatii (legaturi) se stabilesc intre un tabel asociativ si tabelele asociate. Tabelul asociativ are cate o cheie strana pentru fiecare tabel asociat, iar cheia sa primara este formata din toate aceste chei straine plus (eventual) alte atribute suplimentare.

Transformarea atributelor

Regula generală: Atributele se convertesc în coloane ale tabelelor provenite din entități sau chiar în tabele.

Cazuri:

a) Atributele simple ale unei entități devin coloane în tabelul provenit din acea entitate.

b) Toate componentele unui atribut compus devin coloane.

Ex.: *adresa* → *tara, oras, str., nr., bl., sc., et., ap.*

c) Atributele repetitive (multivaloare) ale unei entități devin tabele dependente ce contin fiecare o cheie străină (care face referință la cheia primară a entității) și atributul multivaloare. Cheia primară a unui astfel de nou tabel este formată din cheia străină plus alte coloane suplimentare.

Ex.:



Fig. 2.22.

d) - Atributele simple ale unei relații 1:1 sau M:1 devin coloane ale tabelului care conține cheia străină.

Ex.: atributul *data_inscrierii* al relației *studiaza_la* devine coloana în tabelul STUDENT.

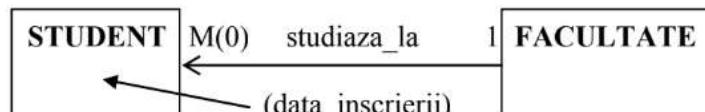


Fig. 2.23.

- Fiecare componentă a unui atribut compus al unei relații 1:1 sau M:1 se va converti în mai multe coloane în tabelul care conține cheia străină.

e) - Atributele simple ale unei relații M:M vor deveni coloane ale tabelului asociativ.

Ex.:

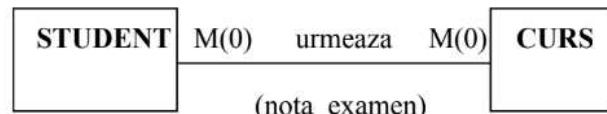


Fig. 2.24.

Atributul “*nota_examen*” al relației “*urmează*” devine coloană în tabelul asociat *URMARE*.

- Fiecare componentă a unui atribut compus al unei relații M:M va deveni o coloană în tabelul asociativ creat.

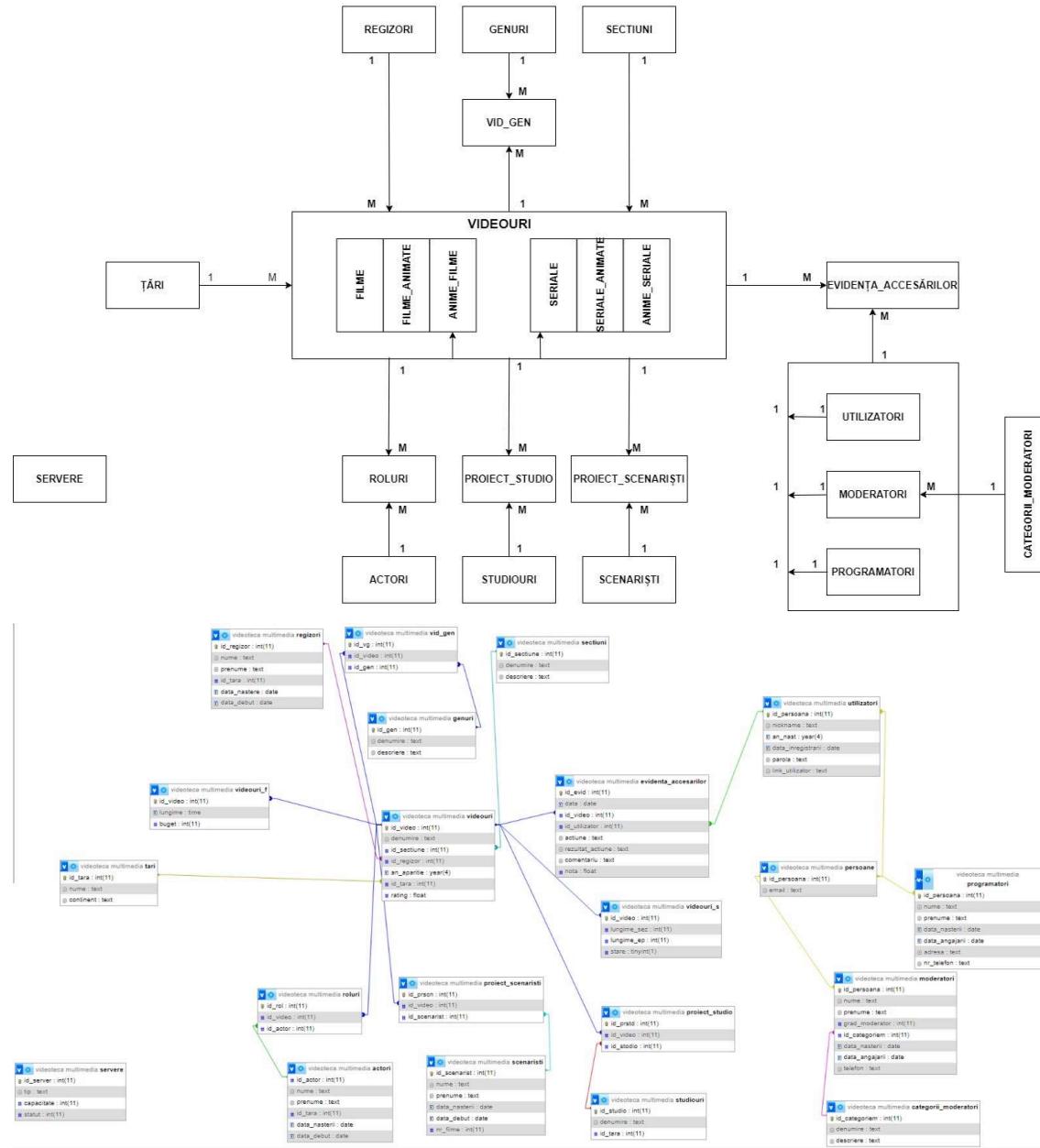
f) - Atributele repetitive (multivaloare) ale unei relații 1:1 sau 1:M devin tabele dependente de tabelul care conține cheia străină.

- Atributele repetitive ale unei relații M:M devin tabele dependente de tabelul asociativ corespunzător relației. Cheia primară a acestor tabele dependente va fi formată din cheia străină respectivă plus una sau mai multe coloane suplimentare.

Diagrama logica a BD si tabelele asociate

Prezentam diagrama logica a videotecii multimedia WathPal care a rezultat in urma transformarilor necesare.

VIDEOTECA MULTIMEDIA



Videoteca Multimedia

TARI (id_tara, nume, continent)
SECTIUNI (id_sectiune, denumire, descriere)
GENURI (id_gen, denumice descriere)
REGIZORI (id_regizor, nume, prenume, id_tara, data_nasterii, data_debut)
VIDEOURI (id_video, denumire, id_sectiune, id_regizor, an_aparitie, id_tara, rating)
VIDEOURI_F (id_video, lungime (in minute), buget)
VIDEOURI_S (id_video, lungime (in sezoane+episoade), stare (ongoing/terminat))
VID_GEN (id_vg, id_video, id_gen)
ACTORI (id_actor, nume, prenume, id_tara, data_nasterii, data_debut)
STUDIOURI (id_studio, denumire, id_tara)
SCENARISTI (id_scenarist, nume, prenume, data_nasterii, data_debut, nr_filme)
ROLURI (id_rol, id_video, id_actor)
PROIECT_STUDIO (id_prstd, id_video, id_studio)
PROIECT_SCENARISTI (id_prscn, id_video, id_scenarist)
PERSOANE (id_persoana, email)
UTILIZATORI (id_persoana, nickname, an_nast, data_inregistrarii, parola, link_utilizator)
EVIDENTA_ACCESARILOR (id_evid, data, id_video, id_utilizator, actiune, rezultat_actiune, comentariu, nota)
MODERATORI (id_persoana, nume, prenume, grad_moderator, id_categoriem, data_nasterii, data_angajarii, telefon)
CATEGORII_MODERATORI (id_categoriem, denumire, descriere)
PROGRAMATORI (id_persoana, nume, prenume, data_nasterii, data_angajarii, adresa, nr_telefon)
SERVERE (id_server, tip, capacitate, statut (principal/rezerva))

Legenda:

MAJUSCULE – tabelele

minuscule – coloanele in tabele

Rosu – tabele de legatura

Oranj – super tabele

Albastru – chei primare

Verde – chei straine

4. NORMALIZAREA BAZEI DE DATE

4.1. Noțiuni introductive

In trecut normalizarea era utilizata pentru proiectarea unei BD. In prezent, proiectare unei BD se realizeaza pe baza celor prezентate anterior (schema conceptuala, schema logica), iar normalizarea intervine asupra tabelelor obtinute pe baza schemei logice eliminand unele probleme care pot apare in procesul de proiectare initial: redundanta in date, anomalii la actualizare.

Definitie: Normalizarea reprezinta procesul de descompunere a unui tabel relational in mai multe tabele care satisfac anumite reguli si care stocheaza aceleasi date ca si tabelul initial astfel incat sa fie eliminate redundanta in date si anomalii la actualizare.

Exemplu: Fie tabelul VANZARI care se foloseste la inregistrarea tranzactiilor unui magazin ce vinde articole la comanda.

VANZARI (cod_client, nume_client, nr_telefon, cod_comanda, data, cod_articol, nume_articol, cost_articol, cantitate)

cod_client	nume_client	nr_telefon	cod_comanda	data	cod_articol	nume_articol	cost_articol	cantitate
A1	Popescu	415355	C1	08.10.01	P1	camasa	400000	2
A1	Popescu	415355	C1	08.10.01	P3	tricou	200000	1
A2	Ionescu	596322	C2	09.10.01	P1	camasa	400000	3
A2	Ionescu	596322	C2	09.10.01	P3	tricou	200000	2
A2	Ionescu	596322	C2	09.10.01	P2	pantaloni	800000	1
A1	Popescu	415355	C3	10.10.01	P3	tricou	200000	3
A3	Marinescu	546229	C4	10.10.01	P1	camasa	400000	1

Tabelul de mai sus prezinta urmatoarele deficiente:

a) redundante in date:

- informatia (P1, camasa, 400000) este specificata de 3 ori,
- informatia (A1, Popescu, 415355) este specificata de 3 ori,
- informatia (A2, Ionescu, 596322) este specificata de 3 ori etc.

b) anomalii la actualizare

- anomalie la insertie

Daca magazinul achizitioneaza un nou articol (P4, pantofi, 980000) informatia nu poate fi introdusa in tabel (un nou tuplu) pentru ca s-ar introduce o valoare Null in cheia primara (cod_comanda).

- anomalie la stergere

Daca este anulat articolul P2 in cadrul comenzii C2 se pierde informatia referitoare la numele si costul articolului respectiv.

- anomalie la modificare

Daca se modifica nr. de telefon al unui client modificarea trebuie facuta in toate tuplurile (liniile) unde apare numele acelui client.

In cele ce urmeaza se va realiza o eliminare a deficienelor constatate, dar mai intai,

sunt definite cateva notiuni.

a) Caracterul reversibil al normalizarii.

Prin caracter reversibil al normalizarii se intlege faptul ca descompunerea se face fara pierdere de informatie, adica tabelul initial poate fi reconstituit prin compunerea naturala, pe atribute comune, a tabelelor rezultate.

Pentru un tabel R care se descompune prin proiectie in mai multe tabele: R1, R2, ...

Rn, conditia de descompunere fara pierdere de informatie presupune ca in urma operatiei de compunere naturala a tabelelor R1, R2, ... Rn sa se obtina tabelul R.

Regula Casey-Delobel (caz particular de descompunere fara pierdere de informatie):

Fie un tabel R(X, Y, Z) care se descompune prin proiectie in tabelele R1(X, Y) si R2(X, Z) unde prin X notam setul de coloane comune ale tabelor R1 si R2, iar prin Y si Z, coloanele specifice lui R1, respectiv R2. Conditia de descompunere fara pierdere de informatie presupune ca tabelul R sa fie obtinut prin compunerea naturala a tabelor R1 si R2.

In SQL:

```
SELECT R1.X, R1.Y, R2.Z  
FROM R1, R2  
WHERE R1.X = R2.X
```

b) Dependenta functionala

Definitie: Fie R un tabel relational si X si Y doua submultimi de coloane ale lui R.

Spunem ca X determina functional pe Y sau ca Y depinde functional de X daca nu exista doua

randuri in tabelul R care sa aiba aceleasi valori pentru coloanele din X si sa aiba valori diferite

pentru cel putin o coloana din Y.

Notatie uzuala: $X \rightarrow Y$

unde X = determinant

Y = dependenta

$X \rightarrow Y$ este triviala daca $Y \subseteq X$.

Exemplu: In tabelul Vanzari exista urmatoarele dependente funktionale in care determinantul nu este cheie a tabelului:

(cod_articol) \rightarrow (nume_articol, cost_articol,
(cod_comanda) \rightarrow (data, cod_client, nume_client, nr_telefon)
(cod_client) \rightarrow (nume_client, nr_telefon)

Obs.: Existenta dependentelor funktionale pentru care determinantul nu este cheie a tabelului duce la aparitia redundantei in date si a anomalilor la actualizare in lucrul cu BD.

Definitie: Fie R un tabel relational si fie X si Y doua submultimi de coloane ale lui R.

O dependenta functionala $X \rightarrow Y$ se numeste dependenta functionala totala daca pentru orice

subset de coloane Z al lui X si $Z \subseteq X$, daca $Z \rightarrow Y$ atunci $Z = X$. Deci nu exista nici un subset Z al lui X (cu $Z \neq X$) pentru care $Z \rightarrow Y$.

Definitie: O dependenta functionala care nu este totala se numeste dependenta functionala parciala.

c) Dependenta functionala tranzitiva

Fie R un tabel relational, X o submultime de coloane a lui R si A o coloana a lui R.

Spunem ca A este dependenta tranzitiv de X daca exista o submultime de coloane Y care nu

include coloana A si care nu determina functional pe X astfel incat $X \rightarrow Y$ si $Y \rightarrow A$. Daca in

aceasta definitie se doreste sa se evidentieze si Y atunci se spune ca A depinde functional de
X prin intermediul lui Y si se scrie: $X \rightarrow E Y \rightarrow E A$.

Exemplu: (cod_comanda) $\rightarrow E$ (cod_client) $\rightarrow E$ (nume_client)

d) Descompunerea minimala

Prin descompunerea minimala a unui tabel se intlege o descompunere astfel incat nici o coloana din tabelele rezultate nu poate fi eliminata fara a duce la pierderea de informatii si implicit la pierderea caracterului ireversibil al transformarii. Aceasta inseamna ca nici unul dintre tabelele rezultate nu poate fi continut unul in altul.

Obs.: Este de dorit ca procesul de normalizare sa conserve dependentele funktionale netranzitive dintre date (atat determinantul cat si determinatul sa se regaseasca intr-unul din tabelele rezultate prin descompunere). Dependentele funktionale tranzitive nu trebuie conservate deoarece ele pot fi deduse din cele netranzitive.

Un tabel relational se poate afla in 6 situatii diferite numite forme normale:

- prima forma normala,
- a 2-a forma normala,
- a 3-a forma normala,
- forma normala Boyce-Codd,
- a 4-a forma normala,
- a 5-a forma normala.

Un tabel este intr-o anumita forma normala daca satisface un set de constrangeri, corespunzator acelei forme normale. Constrangerile pentru o anumita forma normala sunt totdeauna mai puternice decat cele pentru formele normale inferioare (tabelele din f.n._i sunt un subset al tabelelor din f.n._i-1). Codd si Fagin au elaborat o teorie matematica a normalizarii.

4.2. Prima forma normala (1NF - First Normal Form)

Definitie: Un tabel relational este in 1NF daca fiecarei coloane ii corespunde o valoare indivizibila (atomica). Deci orice valoare nu poate fi compusa dintr-o multime de valori (atributele compuse) si nu sunt admise atributele repetitive.

Algoritmul 1NF:

1. Se inlocuiesc in tabel coloanele corespunzatoare atributelor compuse cu coloane ce contin componentele acestora.
2. Fiecare grup de atribute repetitive se plaseaza in cate un nou tabel.
3. Se introduce in fiecare tabel nou creat la pasul 2 cheia primara a tabelului din care a fost extras atributul respectiv, care devine cheie strina in noul tabel.
4. Se stabileste cheia primara a fiecarui nou tabel creat la pasul 2. Aceasta va fi compusa din cheia strina introdusa la pasul 3 plus una sau mai multe coloane suplimentare.

Obs.: Algoritmul 1NFA permite aducerea unui tabel R in 1NF prin eliminarea atributelor compuse si atributelor repetitive.

(

4.3. A doua forma normala (2NF - Second Normal Form)

Definitie: Un tabel relational R este in a doua forma normala (2NF) daca si numai daca:

- R este in 1NF
 - Orice coloana care depinde parțial de o cheie a lui R este inclusă în acea cheie.
- Deci, 2NF nu permite dependențele funcționale parțiale fata de cheile tabelului, cu excepția dependențelor triviale, de incluziune.

Regula de descompunere a tabelului:

Fie $R(K_1, K_2, X, Y)$ un tabel relational unde K_1, K_2, X și Y sunt submultimi de coloane ale lui R astfel încât $K_1 \cup K_2$ este o cheie a lui R , iar $K_1 \not\rightarrow X$ este o dependență funcțională totală. Dacă $X \subset K_1$ atunci tabelul este deja în 2NF, altfel tabelul R poate fi descompus prin proiecție în $R_1(K_1, K_2, Y) ->$ având cheia $K_1 \cup K_2$ și $R_2(K_1, X) ->$ având cheia primă K_1 . Aceasta descompunere conservă și datele și dependențele funcționale.

Algoritmul 2NF:

1. Pentru fiecare set de coloane X care depinde funcțional parțial de o cheie K , $K \not\rightarrow X$, și care nu este inclusă în K , se determină $K_1 \subset K$ un subset al lui K , astfel încât dependența $K_1 \not\rightarrow X$ este totală și se creează un nou tabel $R_1(K_1, X)$, adică un tabel format din determinantul K_1 și determinantul X al acestei dependențe.
2. Dacă în tabelul R există mai multe dependențe totale ca mai sus cu același determinant, atunci pentru acestea se creează un singur tabel format din determinant (luat o singura dată) și din toți determinanții dependențelor considerate.
3. Se elimină din tabelul initial R toate coloanele X care formează determinantul dependențelor considerate.
4. Se determină cheia primă a fiecarui tabel nou creat, R_1 . Aceasta va fi K_1 , determinantul dependențelor considerate.
5. Dacă noile tabele create contin alte dependențe parțiale, atunci se merge la pasul 1, altfel algoritmul se termină.

4.4. A treia forma normală (3NF - Third Normal Form)

Definție_1: Un tabel relational R este în a treia forma normală (3NF) dacă și numai dacă:

- R este în 2NF
- Pentru orice coloana A necontinuită în nici o cheie a lui R , dacă există un set de coloane X astfel încât $X \not\rightarrow A$, atunci fie X conține o cheie a lui R , fie A este inclusă în X .

Obs.: A doua condiție din definție interzice dependențele funcționale totale de alte coloane în afara celor care constituie chei ale tabelului. Prin urmare, un tabel este în 3NF

dacă orice coloana care nu este continuată într-o cheie, depinde de cheie, de întreaga cheie și

numai de cheie. Dacă există astfel de dependențe funcționale trebuie efectuate noi descompuneri ale tabelelor. Tinând cont de definirea notiunii de dependență funcțională tranzitivă (4.1.) se poate formula:

Definție_2: Un tabel relational R este în a treia forma normală (3NF) dacă și numai dacă:

- R este în 2NF
- Orice coloana necontinuită în nici o cheie a lui R nu este dependență tranzitivă de nici o cheie a lui R .

Regula de descompunere pentru 3NF fără pierdere de informație:

Fie $R(K, X, Y, Z)$ un tabel relational unde K este o cheie a lui R , iar X, Y și Z sunt

submultimi de coloane ale lui R.

a) Daca exista dependenta tranzitiva $K \rightarrow X \rightarrow Y$, atunci R se poate descompune in $R1(K, X, Z)$ – avand cheia K – si $R2(X, Y)$ – avand cheia X.

b) Fie $K1 \subset K$ o parte a cheii K astfel incat exista dependenta tranzitiva $K \rightarrow X1 \rightarrow Y$, unde $X1 = K1 \cap X$. In acest caz, R poate fi descompus in $R1(K, X, Z)$ – avand cheia K – si $R2(K1, X, Y)$ – avand cheia $K1 \cap X$.

Algoritmul 3NF:

1. Pentru fiecare dependenta functionala tranzitiva $K \rightarrow X \rightarrow Y$, unde K si X nu sunt neaparat disjuncte, se transfera coloanele din X si Y intr-un nou tabel.

2. Se determina cheia primara a fiecarui nou tabel creat la pasul 1, aceasta fiind formata din coloanele din X.

3. Se elimina din tabelul initial coloanele din Y.

4. Daca tabelele rezultate contin alte dependente tranzitive, atunci se merge la pasul 1, altfel algoritmul se termina.

Obs.: - Descompunerile dupa regula de mai sus conserva atat datele cat si dependentele funktionale, determinantul si determinantul dependentelor eliminate regasindu-se in tabelele nou create.

- Algoritmul 3NFA permite aducerea in 3NF a unui tabel relational aflat in 2NF prin eliminarea dependentelor funktionale tranzitive.

4.5. Forma normală Boyce-Codd (BCNF – Boyce-Codd Normal Form)

De obicei bazele de date utilizează tabele care necesită doar algoritmii primelor 3 forme normale pentru o normalizare completă a lor (1NF-3NF). Sunt totuși cazuri în care tabele aflate în 3NF mai prezintă redundanță în date și anomalii la actualizare datorită prezenței unor dependențe funktionale non-cheie, fiind necesară trecerea la un nivel superior

de normalizare. Următorul nivel de normalizare este Forma Normală Boyce-Codd (BCNF – Boyce-Codd Normal Form).

Definiție:

Un tabel relational R este în forma normală Boyce-Codd (BCNF) dacă și numai dacă pentru orice dependență funcțională totală $X \rightarrow A$, unde X este un subset de coloane ale lui R,

iar A este o coloană neconținută în X, X este o cheie a lui R.

Algoritmul BCNF:

- Pentru fiecare dependență non-cheie $X \rightarrow Y$, unde X și Y sunt subseturi de coloane ale lui R, se creează 2 tabele: R1 format din coloanele X și Y și R2 format din coloanele inițiale ale lui R, mai puțin coloanele Y.

- Dacă tabelele rezultate conțin alte dependențe non-cheie, atunci se merge la pasul 1, altfel algoritmul se termină.

Observații:

- Putem spune că un tabel R este în BCNF dacă fiecare determinant al unei dependențe funktionale este cheie candidată a lui R.

- Orice tabel aflat în BCNF este și în 3NF, reciproca fiind falsă.

- Orice tabel care are cel mult 2 coloane este în BCNF.

- Descompunerea prin algoritmul BCNFA se face fără pierdere de informație, dar nu se garantează păstrarea dependentelor funktionale.

4.6. A 4-a formă normală (4NF – Fourth Normal Form)

Definiții:

Fie R un tabel relațional, X și Y două submulțimi de coloane ale lui R și $Z = R - X - Y$ multimea coloanelor din R care nu sunt nici în X nici în Y. Spunem că există o dependență multivaloare Y de X sau că X determină multivaloare pe Y (notația folosită: $X \rightarrow\!\! \rightarrow Y$), dacă, pentru orice valoare a coloanelor lui X, sunt asociate valori pentru coloanele din Y care

nu sunt corelate în nici un fel cu valorile coloanelor lui Z.

	R	X	Y	Z
u	x	y1	z1	
v	x	y2	z2	
s	x	y1	z2	
t	x	y2	z1	

Cu alte cuvinte $X \rightarrow\!\! \rightarrow Y$ dacă și numai dacă oricare ar fi u și v două rânduri ale lui R cu $u(X) = v(X)$, există s și t două rânduri ale lui R a.î.

$s(X) = u(X)$ $s(Y) = u(Y)$ $s(Z) = v(Z)$

$t(X) = u(X)$ $t(Y) = v(Y)$ $t(Z) = u(Z)$

Obs.: - Dacă $X \rightarrow\!\! \rightarrow Y$ atunci și $X \rightarrow\!\! \rightarrow Z$.

- Dependența multivaloare se mai numește și multidependență.

- Orice dependență funcțională este și o dependență multivaloare, reciproca nefiind în general adevărată.

Definiție 4NF

Un tabel relațional R este în 4NF dacă și numai dacă:

- R este în BCNF

- Orice dependență multivaloare $X \rightarrow\!\! \rightarrow Y$ este de fapt o dependență funcțională $X \rightarrow Y$. sau:

Un tabel relațional R este în 4NF dacă și numai dacă pentru orice dependență multivaloare $X \rightarrow\!\! \rightarrow Y$ există o cheie a lui R inclusă în X.

Echivalența definițiilor: Deoarece Orice dependență multivaloare $X \rightarrow\!\! \rightarrow Y$ este de fapt o dependență funcțională $X \rightarrow Y$, iar BCNF a eliminat orice dependență non-cheie înseamnă că

X conține o cheie a lui R.

Regulă de descompunere:

Fie $R(X,Y,Z)$ un tabel relațional în care există o dependență multivaloare $X \rightarrow\!\! \rightarrow Y$ astfel încât X nu conține nici o cheie a lui R. Atunci tabelul R poate fi descompus prin proiecție în 2 tabele $R1(X, Y)$ și $R2(X, Z)$.

Algoritmul 4NF:

1. Se identifică dependențele multivaloare $X \rightarrow\!\! \rightarrow Y$ pentru care X și Y nu conțin toate coloanele lui R și X nu conține nici o cheie a lui R.

2. Se înlocuiește tabelul inițial R cu 2 tabele: unul format din coloanele (X, Y), iar

celălalt format din toate coloanele inițiale, mai puțin coloanele Y.

3. Dacă tabelele rezultate conțin alte dependențe multivaloare se revine la pasul 1, altfel algoritmul se termină.

Obs.: 4NF elimină relațiile M:M independente (elimină dependențele funcționale multivaloare).

4.7. A 5-a formă normală (5NF – Fifth Normal Form)

Se întâlnește destul de rar în practică și are rolul de a elimina relațiile M:M dependente care pot introduce redundanțe în date.

Regulile de descompunere stabilite pentru formele normale 1NF-4NF permit să se descomponă prin proiecție unui tabel relațional R în 2 tabele relaționale R1 și R2.

Există

însă tabele relaționale care nu pot fi descompuse în 2 tabele, ci în 3 sau mai multe, fără

pierdere de informație. Astfel de situații sunt tratate de 5NF. S-a introdus conceptul de join-dependență sau dependență la compunere sau dependență de uniune.

Definiție pentru join-dependență:

Fie R un tabel relațional și R1, R2, ..., Rn o mulțime de tabele relaționale care nu sunt disjuncte (au coloane comune) a. I. reuniunea coloanelor din R1, R2, ..., Rn este mulțimea coloanelor din R. Spunem că există join-dependență notată *(R1, R2, ..., Rn) dacă R se descompune prin proiecție în R1, R2, ..., Rn fără pierdere de informație, adică tabelul inițial poate fi reconstruit prin compunerea naturală pe atributele comune ale tabelelor rezultante.

Join-dependență este o generalizare a dependenței multivaloare. Dependență multivaloare corespunde join-dependenței cu 2 elemente.

- Dacă avem R(X, Y, Z) și există dependență multivaloare XÆÆY atunci există și join-dependență *((X ∪ Y), X ∪ Z)).

- Dacă avem join-dependență *(R1, R2) atunci există și dependență multivaloare (R1 ∩ R2) ÆÆ (R1 – (R1 ∩ R2)).

Definiție pentru 5NF:

Un tabel relațional R este în 5NF dacă și numai dacă orice join-dependență *(R1, R2, ..., Rn)

este consecința cheilor candidate ale lui R, adică fiecare dintre R1, R2, ..., Rn include o cheie

candidată a lui R.

6. DENORMALIZAREA BAZEI DE DATE

6.1 Notiuni teoretice

Denormalizarea unei BD reprezinta procesul invers operatiei de normalizare si duce la cresterea redundantei datelor. Prin aceasta se doreste, in principal, cresterea performantei si simplificarea programelor de interogare a datelor.

Obs.: - Denormalizarea se face numai dupa o normalizare corecta.

- Denormalizarea se face printr-o selectare strategica a structurilor care aduc avantaje semnificative

- Denormalizarea trebuie insotita de masuri suplimentare de asigurare a integritatii datelor.

a) Cresterea performantei

Un caz frecvent intalnit in interogarea BD este cazul unor operatii sau calcule foarte des utilizate.

Ex.: Fie tabelele care modeleaza tranzactiile unui magazin care vinde articole la comanda.:

VANZARI_2 (cod_comanda, cod_articol, cantitate)

ARTICOL (cod_articol, nume_articol, cost_articol)

COMANDA_1 (cod_comanda, data, cod_client)

CLIENT (cod_client, nume_client, nr_telefon)

Sa presupunem ca majoritatea rapoartelor cerute de conducerea magazinului se refera la cantitatea totala vanduta intr-o luna pentru fiecare articol. In acest caz este util un tabel suplimentar:

ARTICOL_LUNA (cod_articol, luna, cantitatea_totala)

Utilizarea acestui tabel suplimentar ceara un lucru important: sa se realizeze interogari usoare si rapide, dar si dezavantajul cresterii redundantei datelor fiind, in acelasi timp, periclitarea integritatii datelor. Solutia gasita este atasarea la tabelele VANZARI_2 si COMANDA_1 de declansatoare (trigger-e) care se activeaza dupa fiecare modificare data de INSERT, UPDATE, DELETE. Un efect produs: incetinirea operatiilor de actualizare.

b) Simplificarea codului pentru manipularea datelor

Exemplu: Fie tabelul STOCURI utilizat de o firma pentru inregistrarea cantitatilor de materiale existente in fiecare din depozitele sale.

STOCURI (cod_depozit, cod_material, nume_material, cantitate)

Se cere aflarea tuturor depozitelor in care exista ciocolata.

Dupa normalizare avem:

STOCURI_1 (cod_depozit, cod_material, cantitate)

MATERIAL (cod_material, nume_material)

Interogarea in SQL va fi:

- varianta nenormalizata:

SELECT cod_depozit, cantitate

FROM stocuri

WHERE nume_material = "ciocolata";

- varianta normalizata:

SELECT cod_depozit, cantitate

FROM stocuri_1, material

WHERE stocuri_1.cod_material = material.cod_material

AND nume_material = "ciocolata";

O solutie care rezolva problema conta in a crea vederi bazate pe tabele normalizate.

Ex: CREATE VIEW stocuri AS

```
SELECT cod_depozit, stocuri_1.cod_material, nume_material, cantitate
FROM stocuri_1, material
WHERE stocuri_1.cod_material = material.cod_material;
```

6.2. Denormalizarea bazei de date

In cazul bazei noastre de date nu este nevoie.

Exemple interogari.

Care sunt cei mai vechi utilizatori ai aplicatiei?

The screenshot shows a MySQL Workbench interface. At the top, there's a toolbar with icons for Edit, Copy, Delete, and other functions. Below the toolbar is a table with columns: id_persoana, nickname, and data_inregistrari. The data shows 14 rows of user information. To the right of the table is a text area containing two SQL queries:

```
Press Ctrl+Enter to execute query
=SELECT g.Denumire AS Gen, COUNT(v.id_video) AS Numar_Aparitii FROM Genuri g JOIN Videouri v ON g.id_gen = v.id_gen GROUP BY g.Denumire ORDER BY Numar_Aparitii DESC;
>SELECT id_persoana, nickname, data_inregistrarii FROM utilizatori ORDER BY data_inregistrarii ASC;
> INSERT INTO actori (id_actor, nume, prenume, id_tara, data_nasterii, data_debut)
VALUES
(29, 'Marion', 'Cotillard', 10, '1975-09-30', '1993-01-01')]
```

Care sunt cei mai fideli utilizatori ai aplicatiei?

The screenshot shows a MySQL Workbench interface. At the top, there's a toolbar with icons for Print, Copy to clipboard, Export, Display chart, and Create view. Below the toolbar is a table with columns: id_utilizator and Numar_Aparitii. The data shows 14 rows of user information. To the right of the table is a text area containing two SQL queries:

```
Press Ctrl+Enter to execute query
=SELECT g.Denumire AS Gen, COUNT(v.id_video) AS Numar_Aparitii FROM Genuri g JOIN Videouri v ON g.id_gen = v.id_gen GROUP BY g.Denumire ORDER BY Numar_Aparitii DESC;
>SELECT id_persoana, nickname, data_inregistrarii FROM utilizatori ORDER BY data_inregistrarii ASC;
>SELECT id_utilizator, COUNT(id_utilizator) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_utilizator ORDER BY Numar_Aparitii DESC LIMIT 1;
>
SELECT id_utilizator, COUNT(id_utilizator) AS Numar_Aparitii
FROM evidenta_accesarilor
GROUP BY id_utilizator
ORDER BY Numar_Aparitii DESC
LIMIT 1;
```

Care sunt cele mai populare videouri din aplicatie?

Extra options

id_video	Numar_Aparitii
21	1
46	1
24	1

Open results operations Bookmarks Options History Clear

```
Press Ctrl+Enter to execute query
>SELECT g.Denumire AS Gen, COUNT(v.id_video) AS Numar_Aparitii FROM Genuri g JOIN Videouri v ON g.id_gen = v.id_gen GROUP BY g.Denumire ORDER BY Numar_Aparitii DESC;
>SELECT id_persoana, nickname, data_inregistrarii FROM utilizatori ORDER BY data_inregistrarii ASC;
>SELECT id_utilizator, COUNT(id_utilizator) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_utilizator ORDER BY Numar_Aparitii DESC LIMIT 1;
>SELECT id_video, COUNT(id_video) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_video ORDER BY COUNT(id_video) DESC LIMIT 3;
> SELECT id_video, COUNT(id_video) AS Numar_Aparitii
  FROM evidenta_accesarilor
  GROUP BY id_video
  ORDER BY COUNT(id_video) DESC
  LIMIT 3;
```

Activate Windows

Care videouri sunt considerate cele mai bune, după părerea utilizatorilor?

Denumire Nota

Denumire	Nota
Edit Copy Delete Breaking Bad	9.5
Edit Copy Delete The Shawshank Redemption	9.3
Edit Copy Delete The Godfather	9.2
Edit Copy Delete Hunter x Hunter	9.1

Bookmarks Options History

```
Press Ctrl+Enter to execute query
>SELECT g.Denumire AS Gen, COUNT(v.id_video) AS Numar_Aparitii FROM Genuri g JOIN Videouri v ON g.id_gen = v.id_gen GROUP BY g.Denumire ORDER BY Numar_Aparitii DESC;
>SELECT id_persoana, nickname, data_inregistrarii FROM utilizatori ORDER BY data_inregistrarii ASC;
>SELECT id_utilizator, COUNT(id_utilizator) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_utilizator ORDER BY Numar_Aparitii DESC LIMIT 1;
>SELECT id_video, COUNT(id_video) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_video ORDER BY COUNT(id_video) DESC LIMIT 3;
>SELECT Denumire, Nota FROM Videouri ORDER BY Nota DESC LIMIT 5;
> SELECT Denumire, Nota
  FROM Videouri
  ORDER BY Nota DESC
  LIMIT 5;
```

Câte roluri a jucat scorul X? Inclusiv și rolurile ca actor vocal

id_actor

id_actor
1

Show all Number of rows: 25 Filter rows: Search this table Bookmarks Options History

Console

```
Press Ctrl+Enter to execute query
>SELECT g.Denumire AS Gen, COUNT(v.id_video) AS Numar_Aparitii FROM Genuri g JOIN Videouri v ON g.id_gen = v.id_gen GROUP BY g.Denumire ORDER BY Numar_Aparitii DESC;
>SELECT id_persoana, nickname, data_inregistrarii FROM utilizatori ORDER BY data_inregistrarii ASC;
>SELECT id_utilizator, COUNT(id_utilizator) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_utilizator ORDER BY Numar_Aparitii DESC LIMIT 1;
>SELECT id_video, COUNT(id_video) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_video ORDER BY COUNT(id_video) DESC LIMIT 3;
>SELECT Denumire, Nota FROM Videouri ORDER BY Nota DESC LIMIT 5;
>SELECT COUNT(*) AS Numar_Roluri FROM Roluri WHERE id_actor = (SELECT id_actor FROM Actori WHERE nume = 'Hiiragi');
> SELECT COUNT(*) AS Numar_Roluri
  FROM Roluri
  WHERE id_actor = (SELECT id_actor FROM Actori WHERE nume = 'Hiiragi');
```

Activate Windows

În cazul în care serverele cedează, cine trebuie să se coordoneze eliminarea acestei probleme?

Nume Prenume

Console

```
Press Ctrl+Enter to execute query
>SELECT g.Denumire AS Gen, COUNT(v.id_video) AS Numar_Aparitii FROM Genuri g JOIN Videouri v ON g.id_gen = v.id_gen GROUP BY g.Denumire ORDER BY Numar_Aparitii DESC;
>SELECT id_persoana, nickname, data_inregistrarii FROM utilizatori ORDER BY data_inregistrarii ASC;
>SELECT id_utilizator, COUNT(id_utilizator) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_utilizator ORDER BY Numar_Aparitii DESC LIMIT 1;
>SELECT id_video, COUNT(id_video) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_video ORDER BY COUNT(id_video) DESC LIMIT 3;
>SELECT Denumire, Nota FROM Videouri ORDER BY Nota DESC LIMIT 5;
>SELECT COUNT(*) AS Numar_Roluri FROM Roluri WHERE id_actor = (SELECT id_actor FROM Actori WHERE nume = 'Hiiragi');
>SELECT Denumire AS Gen, COUNT(id_video) AS Numar_Aparitii FROM Genuri JOIN Videouri ON id_gen = id_gen GROUP BY Denumire ORDER BY Numar_Aparitii DESC;
>SELECT Nume, Prenume FROM Moderatori WHERE Grad_moderator = 'Sef';
> SELECT Nume, Prenume
  FROM Moderatori
 WHERE Grad_moderator = 'Sef';

Activate Windows
Go to Settings to activate Windows.
```

Care este cel mai popular proiect de multiplicare al studiai X?

Denumire Nota

	Denumire	Nota
<input type="checkbox"/>	Gravity Falls	8.9
<input type="checkbox"/>	Narcos	8.8
<input type="checkbox"/>	BoJack Horseman	8.7
<input type="checkbox"/>	Gomorrah (Gomorra: La serie)	8.7
<input type="checkbox"/>	Adventure Time	8.6

Console

```
Press Ctrl+Enter to execute query
>SELECT g.Denumire AS Gen, COUNT(v.id_video) AS Numar_Aparitii FROM Genuri g JOIN Videouri v ON g.id_gen = v.id_gen GROUP BY g.Denumire ORDER BY Numar_Aparitii DESC;
>SELECT id_persoana, nickname, data_inregistrarii FROM utilizatori ORDER BY data_inregistrarii ASC;
>SELECT id_utilizator, COUNT(id_utilizator) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_utilizator ORDER BY Numar_Aparitii DESC LIMIT 1;
>SELECT id_video, COUNT(id_video) AS Numar_Aparitii FROM evidenta_accesarilor GROUP BY id_video ORDER BY COUNT(id_video) DESC LIMIT 3;
>SELECT Denumire, Nota FROM Videouri ORDER BY Nota DESC LIMIT 5;
>SELECT COUNT(*) AS Numar_Roluri FROM Roluri WHERE id_actor = (SELECT id_actor FROM Actori WHERE nume = 'Hiiragi');
>SELECT Denumire AS Gen, COUNT(id_video) AS Numar_Aparitii FROM Genuri JOIN Videouri ON id_gen = id_gen GROUP BY Denumire ORDER BY Numar_Aparitii DESC;
>SELECT Nume, Prenume FROM Moderatori WHERE Grad_moderator = 'Sef';
>SELECT Denumire, Nota FROM Videouri WHERE id_video IN (SELECT id_video FROM Proiect_studio WHERE id_studio = (SELECT id_studio FROM Studiouri WHERE denumire = 'X')) ORDER BY Nota DESC;
>SELECT Denumire, Nota FROM Videouri WHERE id_video IN (SELECT id_video FROM Proiect_studio WHERE id_studio = (SELECT id_studio FROM Studiouri WHERE denumire = 'X')) ORDER BY Nota DESC;
>SELECT Denumire, Nota FROM Videouri WHERE id_video IN (SELECT id_video FROM Proiect_studio WHERE id_studio = (SELECT id_studio FROM Studiouri WHERE denumire = 'Fresh TV')) ORDER BY Nota DESC;
> SELECT Denumire, Nota
  FROM Videouri

Activate Windows
Go to Settings to activate Windows.
```

SGBD UTILIZAT

Sistemele de gestiune a bazelor de date (DBMS – DataBase Management System) sunt sisteme informaticе specializate în stocarea și prelucrarea unui volum mare de date, numărul prelucrărilor fiind relativ mic. Termenul de bază de date se va referi la datele de prelucrat, la modul de organizare a acestora pe suportul fizic de memorare, iar termenul de gestiune va semnifica totalitatea operațiilor ce se aplică asupra datelor din baza de date. În arhitectura unui sistem de baze de date SGBD ocupă locul central. Un SGBD este ansamblul software interpus între utilizatori și baza de date și este un interpretor de cereri de acces sau regăsire de date în baza de date, execută cererea și returnează rezultatul. SGBD este un sistem de programe care facilitează procesul definirii, construcției, organizării și manipulării datelor pentru diverse aplicații. Utilizatorul are acces la SGBD prin intermediul unei interfețe (aplicație) cu ajutorul căreia stabilește parametrii interogării și se primește răspuns; întreg ansamblul este descris în Figura 3.

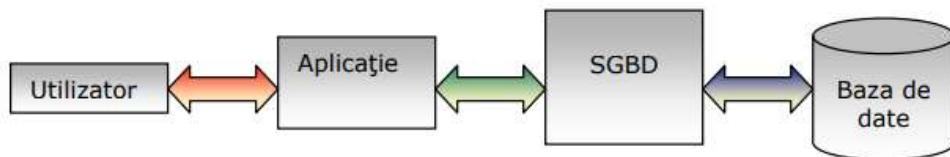


Fig. 3

Organizarea pe trei niveluri a sistemelor de baze de date este strâns legată de conceptul de independentă a datelor, în sensul că sistemul bazei de date poate fi modificat la orice nivel fără a afecta nivelurile superioare.

Independentă datelor poate fi privită în două moduri, corespunzătoare nivelurilor conceptual (logic) și intern (fizic). Independentă logică a datelor reprezintă capacitatea modificării schemei conceptuale fără a provoca modificări în schema externă sau în programele de aplicație. Schema conceptuală se poate modifica prin mărirea bazei de date datorită adăugării de noi tipuri de înregistrări (o nouă coloană într-o tabelă) sau date (înregistrări) sau micșorarea bazei de date în cazul ștergerii unor înregistrări.

Independentă fizică a datelor este dată de capacitatea de schimbare a schemei interne fără modificarea schemei conceptuale sau externe.

Funcționarea unui SGBD se realizează prin comenzi specifice limbajului SQL. Nivelele conceptual și intern nefiind distinct delimitate sunt adresate printr-un limbaj comun numit DDL – Data Definition Language, utilizat pentru administrarea și proiectarea bazei de date în definirea ambelor scheme. Dacă SGBD are o delimitare clară între nivelul conceptual și cel intern, atunci DDL se folosește pentru comenzi la nivel conceptual, iar pentru specificarea schemei interne se folosește limbajul SDL – Storage Definition Language. Pentru cel de al treilea nivel, extern, se folosește limbajul VDL – View Definition Language, destinat utilizatorilor și pentru interfața acestora cu nivelul conceptual. Pentru operațiile tipice legate de căutare, inserare, ștergere și modificarea datelor, SGBD dispune de un limbaj de manipulare numit DML - Data Manipulation Language.

1.5.1. Interfețe SGBD

Un SGBD este un ansamblu complex de programe care asigură interfață între o bază de date și utilizatorii acesteia. SGBD este componenta software a unui sistem de baze de date care interacționează cu toate celelalte componente ale acestuia asigurând legătura și interdependența între ele. Un SGBD oferă interfețele corespunzătoare tuturor categoriilor de utilizatori pentru a facilita legătura acestora cu baza de date. Principalele tipuri de interfețe:

- Interfețe pe bază de meniuri – care oferă utilizatorilor o listă de opțiuni (meniuri) pentru formularea interogărilor.

Interfețe grafice – afișează utilizatorului un set de diagrame, cererile sunt formulate prin manipularea acestor diagrame. De cele mai multe ori interfețele grafice sunt asociate cu meniurile.

- Interfețe bazate pe videoformate – se utilizează pentru introducerea de noi date, actualizarea bazei de date și căutare.
- Interfețe în limbaj natural – acceptă comenzi scrise în limba engleză sau alte limbi de circulație internațională. Interpretarea cererilor se face pe baza unui set de standard de cuvinte cheie ce sunt interpretate pe baza schemei interne.
- Interfețe specializate pentru cereri repetitive (limbaj de comandă) – sunt destinate unei anumite categorii de utilizatori, de exemplu pentru angajații unei bănci se implementează un mic set de comenzi prescurtate pentru a micșora timpul necesar introducerii comenzi.
- Interfețe pentru administrarea bazei de date – se utilizează pentru comenziile privilegiate utilizate de administratorii bazei de date și se referă la crearea de conturi, parole, setarea parametrilor sistemului, autorizarea intrării pe un anumit cont, reorganizarea structurii de stocare a datelor din baza de date, accesul la e, înregistrări.

1.5.2. Obiectivele unui SGBD

Un SGBD are rolul de a furniza suportul software complet pentru dezvoltarea de aplicații informaticice cu baze de date. El trebuie să asigure:

- minimizarea costului de prelucrare a datelor,
- reducerea timpului de răspuns,
- flexibilitatea aplicațiilor și • protecția datelor

(6)

CONCLUZII

1. Evaluarea Bazei de Date (BD)

Baza de date proiectată pentru biblioteca video a demonstrat eficacitate în gestionarea informațiilor despre titlurile video, categorii, utilizatori și împrumuturi. Structura bazei de date s-a dovedit a fi coerentă și ușor de gestionat, oferind o fundație solidă pentru stocarea și recuperarea datelor relevante.

2. Utilitatea SGBD-ului în Cadrul Bibliotecii Video:

Sistemul de Gestire a Bazelor de Date (SGBD) utilizat, cum ar fi MySQL sau MariaDB, a furnizat un mediu robust pentru gestionarea eficientă a datelor. Caracteristicile specifice ale SGBD-ului au contribuit semnificativ la implementarea și utilizarea eficientă a bazei de date în cadrul bibliotecii video.

3. Consistența și Integritatea Datelor:

SGBD-ul a fost esențial în menținerea consistenței și integrității datelor în întreaga bază de date. Utilizarea restricțiilor și a tranzacțiilor a asigurat că datele sunt mereu valide și complete.

BIBLIOGRAFIE

1. https://drive.google.com/file/d/1ISnkm_dDAXxfg3tTfUB4bcpDWWdXKJBm/view
2. <https://danielanicolae.com/files/SGBD.pdf>
3. https://drive.google.com/file/d/1ISnkm_dDAXxfg3tTfUB4bcpDWWdXKJBm/view
4. <https://drive.google.com/file/d/15tJ3Er47Oe3SeWXp9VE38UfVB5vYIGaK/view>