Ministerul Educației al Republici Moldova

Universitatea de Stat din Moldova

Facultatea Matematică și Informatică

**Lucrare de laborator Nr. 2**

**La disciplina Baze De Date**

**Tema: Elaborarea unei baze de date cu ajutorul instrumentului Microsoft Access, având la bază un sistem din viața reală, pe exemplul Bibliotecii Universitare**

Efectual de studentul grup TPS11

Ion Filipski

Verificat: Prof Universitar I. Andrieș

Contents

[Scopul lucrării: 3](#_Toc469654565)

[Analiza Domeniului, sinteză: 3](#_Toc469654566)

[Crearea tabelelor 8](#_Toc469654567)

[Refactorizare 10](#_Toc469654568)

[Concluzie: 11](#_Toc469654569)

[BIBLIOGRAFIE: 12](#_Toc469654570)

# Scopul lucrării:

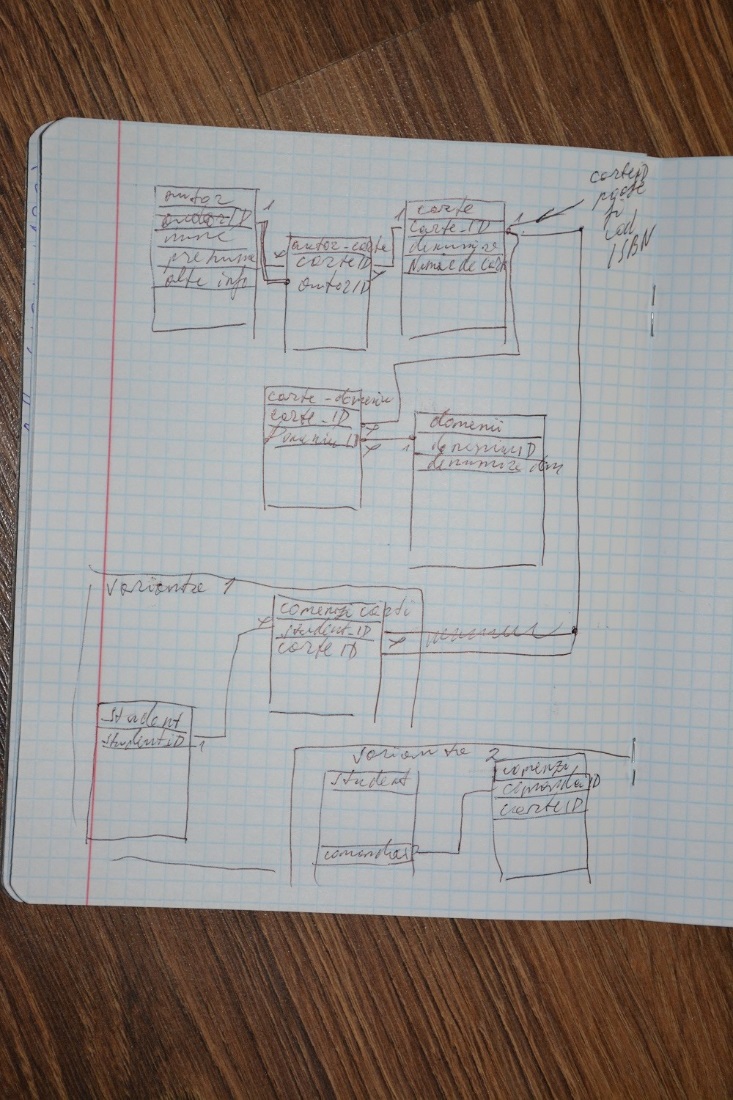
1. Analiza unui sistem real pe exemplul Bibliotecii Universitare și crearea unei baze de date.
2. Elaborarea proiectului sistemului, și eliminarea graduală a erorilor.
3. Crearea unei aplicații flexibile, scalabile și extensibile, lipsită de rigiditate
4. Crearea bazei de date a sistemului
5. Prezentarea flexibilității sistemului prin exemplu de refactoring

# Analiza Domeniului, sinteză:

Proiectarea unui sistem informațional pe exemplul unui sistem real, Biblioteca Universitară.

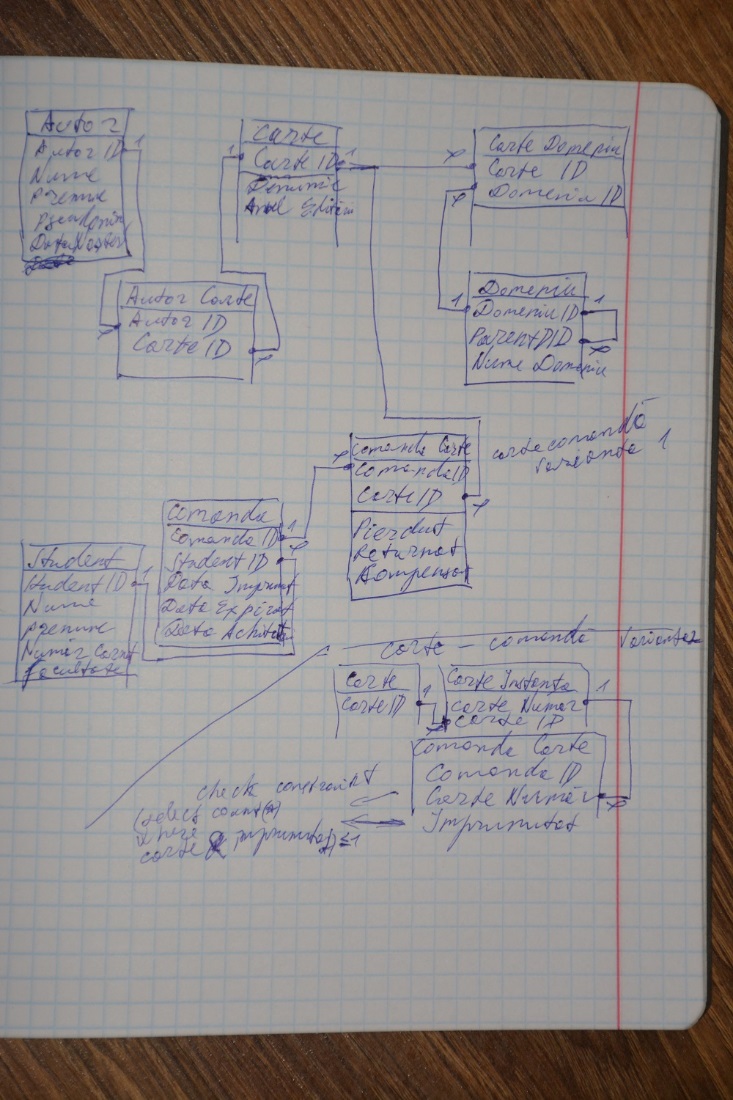
Vom elabora sistemul pe principiul analizei, probelor și lichidării erorilor. Un mod evident de elaborare este modul schematic, pe foaie, înainte de a elabora sisteme reale.

Mai jos vor fi prezentate două scheme de evoluție graduală a conceptului sistemului elaborat. Ambele scheme sunt incorecte, și fiecare schemă nouă prezintă schema precedentă cu eliminarea erorilor.



În prima schemă se văd trei variante de lucru, care creează entități inutile, și în același timp duce carențe de entități necesare. În primul rând acest concept nu permite evidența strictă a cărților. În al doilea rând, nu permite organizarea pe domenii subdomenii. În total au fost elaborate trei variante, vedeți partea de jos a schemei.

Varianta a doua a schemei prezintă îmbunătățirea relației domeniu, subdomeniu. Permite creare comenzi cu mai multe cărți.



Cu o elaborare avansată sistemul ar trebui să permită evidență avansată a cărților. Însă schema elaborată nu reflectă modul real de lucru al bibliotecii, a se vedea mai jos.

Elaborarea unui sistem informațional se face prin analiza proceselor și evidențierea obiectelor. Pe exemplul de lucru al unei biblioteci universitare:

1. În bibliotecă avem cărți, și respectiv cărțile prezintă obiectul principal al bibliotecii.

2. Mai mare interes prezintă anume cărțile care sunt în bibliotecă. Fiecare carte poate exista în inventarul bibliotecii în mai multe exemplare, ori poate să nu existe deloc. Deci, un obiect central va fi inventarul, care ține cont de fiecare carte care există în bibliotecă, împrumutată de cititor, întoarsă în bibliotecă, ori pierdută. Sau absența cărții din cauza că toate exemplarele unei cărți sunt înafara bibliotecii, la cititor. Deci obligator se introduce obiectul Inventar.

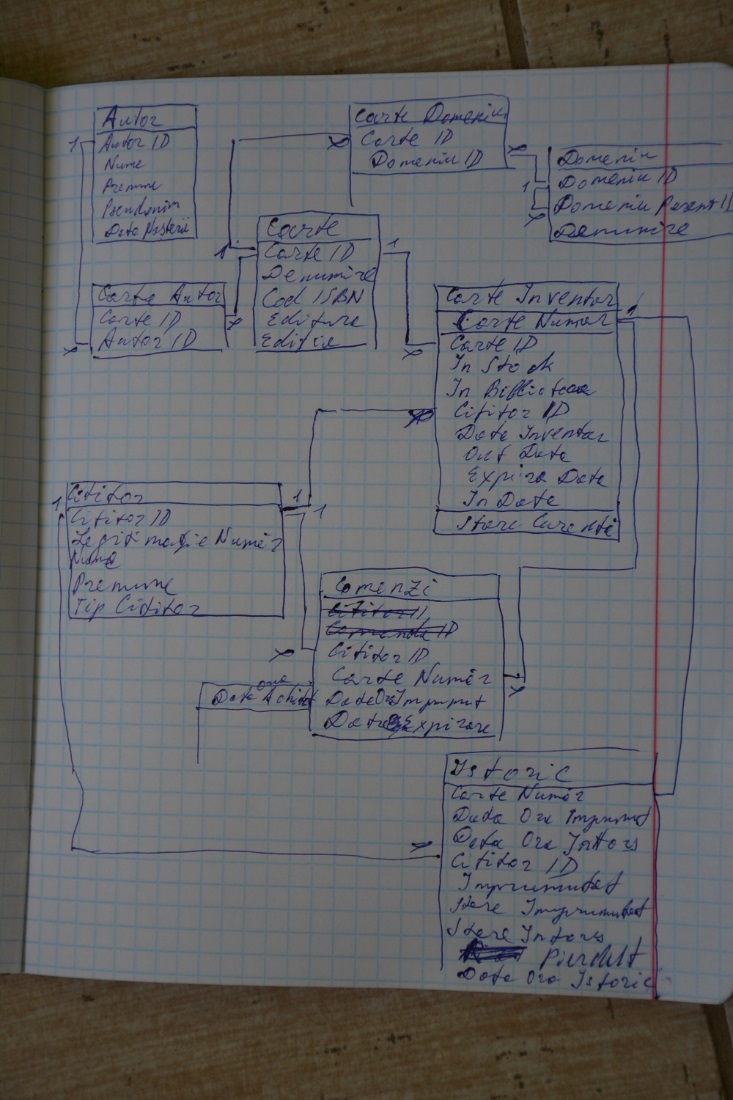
3. Un atribut obligatoriu al unei cărți asta este autorul. Un autor poate avea mai multe cărți publicate, iar o carte poate avea mai mulți autori. Autorii sunt niște atribute a cărților, și nu a inventarului. Orice student poate căuta cartea după autor. Relația dintre cărți și autori este de mai mulți la mai mulți. Respectiv un obiect cheie în ierarhia informațională a bibliotecii va fi autorul.

4. Cititorul poate să nu știe denumirea cărții care o caută, ori autorul, autorii cărților. Însă în toate cazurile cititorul știe exact ce domeniu îl interesează. În acest caz, cititorul va căuta după domeniu. Fiecare domeniu poate fi mai concret, sau mai generic. De exemplu, un domeniu generic este medicina. Un domeniu mai concret ca medicina este chirurgia, dar e mai generic decât microchirurgia ochiului. Aici avem o ierarhie arborescentă de domenii și subdomenii. Studentul poate căuta după toată ierarhia de subdomenii. Iar cartea poate avea mai multe domenii subdomenii. Deci am evidențiat un obiect nou domeniu, și două relații noi, domeniu-domeniu unu la mai mulți, și carte-domeniu, mai mulți la mai mulți.

5. Vorbind despre ce poate face un cititor sau nu poate face, este necesar de elaborat un moment cheie. Biblioteca nu poate exista fără cititor. Fiecare cititor se înregistrează în bibliotecă și primește un identificator unic al bibliotecii. Dar în contextul instituției în care se află biblioteca și cititorul, mai există și niște atribute unice atribuite de instituție, precum număr carnet de student, legitimație de profesor, tip precum cititor, profesor, grad didactic sau științific.

6. Pentru statistici poate fi necesar un tabel de istoric al utilizării cărților. El va fi construit pe principiu cumulativ, cu sau fără constrângeri de integritate. În el se va conține fiecare apariției a cărții în stoc, istoricum împrumuturilor și returnărilor, precum și a cititorilor. Statisticile pot fi folosite pentru a vedea preferințele cititorillor, gradul de încredere și perseverență, care cărți sumt mai utilizate, autori, domenii, șamd. Asta poate fi util pentru elaborarea strategiilor dezvoltării bibliotecii.

Cititorul face o comandă de mai multe cărți. Însă, la nivel de bibliotecă, de fapt cititorul face comandă de exemplare de cărți din inventar. O comandă poate cuprinde mai multe cărți din inventar. Deci logic s-ar cere un obiect nou, comanda. Un mod schematic de reprezentare prefinal al trebui să arate așa:



Însă modul de lucru al bibliotecii arată multe carențe al acestui principiu. În primul rând, un cititor poate întoarce parțial cărțile. Atunci, care e procedura? Se crează o comandă nouă cu cărțile rămase și se șterge comanda veche? Sau se scot cărțile întoarse din comandă? Nici una dintre aceste procese nu este un proces valid în funcționarea unei biblioteci. În al doilea rând. Dacă un student întoarce exemplarul unei cărți, cum se controlează corectitudinea faptului că această carte devine accesibilă altor cititori? Trebuie să pierdem comenzile vechi cu toată statistica? În al treilea rând. O singură carte din inventar poate fi la un singur cititor într-un moment dat de timp. Și nu poate fi în același timp și la bibliotecă, și la cititor. Deci, noi avem foarte multe constrângeri, legate de toate carențele care pot apărea la o relație, care după cum se vede este una strict unu la unu. Relația unu la unu arată deobicei inutilitatea ei. Iar în cazul bibliotecii arată într-un mod clar inutilitatea obiectului comandă. Deci, toate atributele unui exemplar de carte din inventar, pot fi atribuite chiar cărții din inventar. Deci, prezența cărții în stoc (în bibliotecă sau la cititor), absența cărții în stoc (pierderea de către cititor), prezența exemplarului în bibliotecă, prezența cărții la un cititor înafara bibliotecii, toate aceste atribute aparțin exemplarului de stoc, și se pun în același tabel. Asta corepunde modului real de funcționare a bibliotecii. Dacă elaborăm, observăm, că fiecare exemplar al cărții are o fișă. Când cartea e împrumutată, fișa este scoasă din carte și plasată în fișierul studentului. Asta este echivalent cu prezența cărții în stock, dar în afara bibliotecii în același timp,

Ca rezultat noi obținem o structură a sistemului informatic, care reflectă business logica reală de funcționare a unei biblioteci.

Deci, lista obiectelor elucidate este următoarea

Carte, Inventar, Student, Autor, Domeniu.

Lista referințelor este

Carte – Autor, mai mulți la mai mulți

Carte – Domeniu, mai mulți la mai mulți,

Carte – Inventar, unu la mai mulți,

Inventar – Cititor, unu la mai mulți (un Cititor la mai multe exemplare de inventar)

Domeniu – Domeniu, relație de autoreferire, unu la mai mulți, pentru crearea structurilor arborescente.

Inventar – Cititor – Istoric, relație triplă strictă ori nerestricționată prin constrângeri de integritate. Are rol statistic informativ, și nu rol de business.

Istoricul poate fi utilizat petru elaborarea strategiilor bibliotecii, informarea și încurajarea cititorilor buni cu apariții de ediții noi, pe interese. Și desigur, elucidarea cititorilor care au atitudine nepăsătoare, pierd ori strică cărțile, elaborarea strategiilor de recuperare și reînoire a cărților din stock.

# Crearea tabelelor

create table Autor (

AutorID autoincrement primary key,

Nume char(50),

Prenume char(50),

Pseudonim char(50),

DataNasterii datetime

)

create table Domeniu (

DomeniuID autoincrement primary key,

ParentDomeniuID int,

DenumireDomeniu text(255),

constraint fk\_DomeniuDomeniu foreign key (ParentDomeniuID) references Domeniu (DomeniuID)

)

create table Carte (

CarteID autoincrement primary key,

DomeniuID int,

CodISBN text(14),

Denumire text(255) not null,

Editura text(255),

Editia datetime,

EditieNume text(25),

constraint fk\_CarteDomeniu foreign key (DomeniuID) references Domeniu(DomeniuID)

)

create table CarteAutor(

CarteID int not null,

AutorID int not null,

constraint pk\_CarteAutor primary key (CarteID, AutorID),

constraint fk\_CarteAutorCarte foreign key(CarteID) references Carte (CarteID),

constraint fk\_CarteAutorAutor foreign key(AutorID) references Autor (AutorID)

)

create table Cititor (

CititorID autoincrement primary key,

LegitimatieLucruNumar text(10) not null,

LegitimatieBibliotecaNumar text(10) not null,

Nume text (50),

Prenume text (50),

TipCititor text (127)

)

create table CarteInventar

(

NumarInventar int not null primary key,

CarteID int not null,

InStock bit not null,

InBiblioteca bit not null,

CititorID int,

DataInventar datetime,

DataImprumutat datetime,

DataExpiratImprumut datetime,

constraint fk\_CarteInventarCarte foreign key (CarteID) references Carte (CarteID),

constraint fk\_CarteInventarCititor foreign key (CititorID) references Cititor (CititorID)

)

create table InventarIstoric

(

InventarIstoricID autoincrement primary key,

NumarInventar int not null,

CititorID int,

TipEveniment text (255) not null,

NoteAditionale text(255),

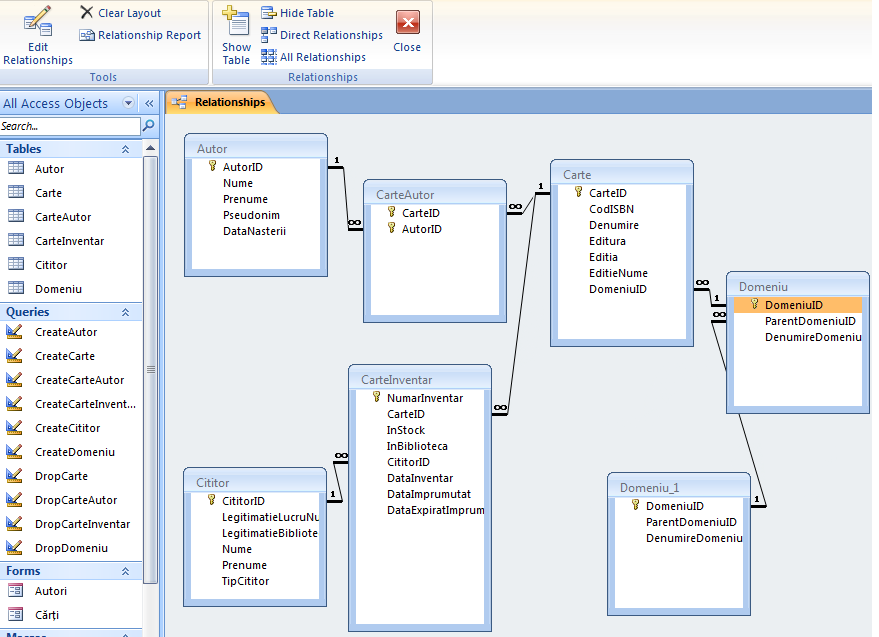
DataEveniment datetime not null,

constraint fk\_InventarIstoricInventar foreign key (NumarInventar) references CarteInventar (NumarInventar),

constraint fk\_InventarIstoricCititor foreign key (CititorID) references Cititor (CititorID)

)

Ca urmare a unei proiectări corecte, obținem schema finală a sistemului informațional, lipsită de neajunsurile elucidate în momentul proiectării:



# Refactorizare

Deoarece orice sistem informațional poate suferi modificări și refactorizări, simulăm un moment normal de lucru, presupunând că a apărut specificație nouă, conform căreia o carte poate avea mai multe domenii.

Pentru asta trebuie să creăm un tabel nou de referințe mai mulți la mai mulți între carte și domeniu

create table CarteDomeniu

(

CarteID int not null,

DomeniuID int not null,

constraint pk\_CarteDomeniu primary key (CarteID, DomeniuID),

constraint fk\_CarteDomeniuCarte foreign key (CarteID) references Carte (CarteID),

constraint fk\_CarteCareteDomeniuDomeniu foreign key (DomeniuID) references Domeniu(DomeniuID)

)

Al doilea pas, copiem datele din tabelul carte în relația nou creată:

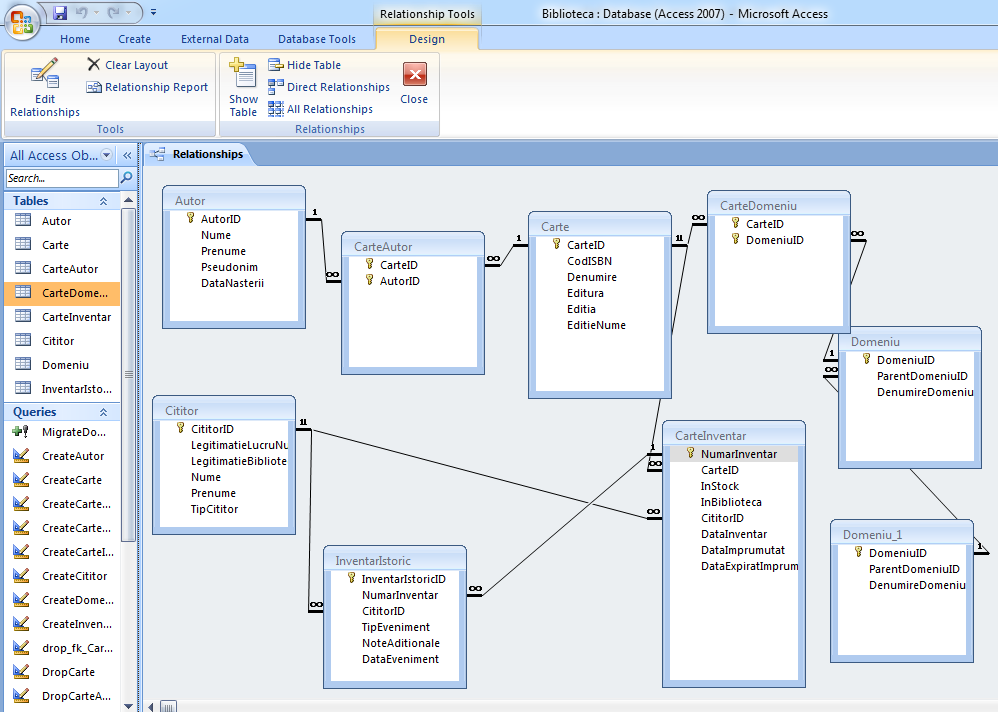
insert into CarteDomeniu select CarteID, DomeniuID from Carte where DomeniuID is not null

Și al treilea pas, ștergem constrângerile vechi și coloana veche din tabelul Carte:

alter table Carte drop constraint fk\_CarteDomeniu

alter table Carte drop column DomeniuID

Ca rezultat obținem următoarea relație



# Concluzie:

Ca urmare a unei analize a obiectelor unui sistem real pe exemplul Bibliotecii am obținut o listă de procese și de obiecte ale sistemului, implementat într-o aplicație Microsoft Access.

Elaborarea arhitecturii avut loc gradual. Fiecare etapă a elaborării prezintă implementarea unei scheme noi, și găsirea neajunsurilor în etapa precedentă. Aceste etape ale analizei și elaborării au duc la apariția unei scheme corecte, scalabile, modificabile și extensibile.

Ca urmare lipsește rigiditatea schemei, de obicei apărute ca urmare a unei analize greșite.

Pentru a demonstra flexibilitatea schemei a fost realizat un exemplu de refactorizare a aplicației, prin schimbarea relației Domeniu – Carte de la unu la mai mulți la mai mulți la mai mulți.

# BIBLIOGRAFIE:

* Vitalie Cotelea: Baze de date și Cunoștințe, UTM Chișinău
* BADEA F. Managementul producţiei industriale Ed. All, Bucureşti
* BARBULESCU C. Managementul producţiei industriale ASE, Bucureşti
* CĂREAN ALEXANDRU Tehnologii de prelucrare cu CNC , Ed. DACIA Cluj-Napoca 2002
* Mereuţă C.,Ciupagea C., Industria prelucrătoare din România 1990-1998. Diagnostic structural, Ed. Libripress, Bucureşti, 2000
* MORAR LIVIU Programarea interactivă a echipamentelornumerice a maşinilor unelte CNC, Ed. UTPRES Cluj-Napoca 2005
* MORAR LIVIU Programarea sistemelor numerice CNC, Ed. UTPRES Cluj-Napoca 2006
* POP ALEXANDRU Maşini şi instalaţii robotizate Ed. DACIA Cluj-Napoca 1999
* Cozma Matei Ionuț Liceul Tehnologic Electromureş Târgu Mureş, http://ccdmures.ro/cmsmadesimple/uploads/file/rev8sp/tehno/tehno10.pdf