

Universitatea Constantin Brâncuși din Târgu Jiu

Facultatea: Inginerie

*Program de conversie profesională a cadrelor didactice din învățământul preuniversitar: Informatică,
Tehnologia Informației și a Comunicațiilor*

Baze de date

Teams: TIC - Baze de date-2022/2023

Adrian Runceanu

Curs 1
Noțiuni introductive
despre
teoria generală a bazelor de date

Câteva precizări

Structura cursului

- 2 ore curs – titular curs

Lector dr. Adrian Runceanu

- 2 ore laborator – titular aplicații practice

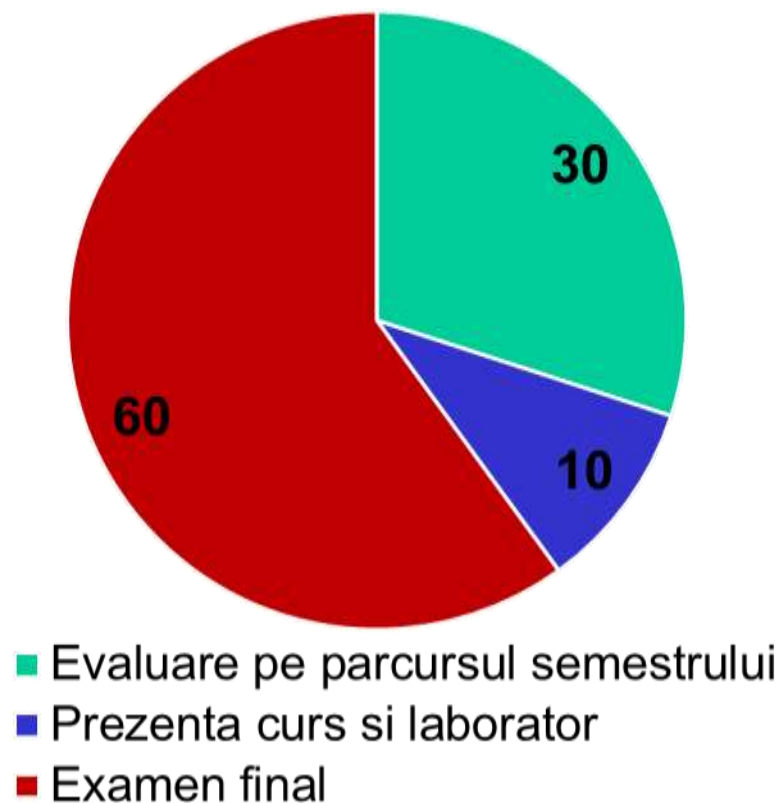
Lector dr. Adrian Runceanu

Câteva precizări

Forme de examinare:

- Examen final (**verificare**)
– 60%
- Evaluare pe parcursul semestrului a activității de laborator – 30%
- Prezența pe parcursul semestrului – 10%

Procentaje evaluare



Câteva precizări

Bibliografia necesară cursului:

1. **Dezvoltarea bazelor de date în Oracle 9i prin exemple**, Dan Hotka, Editura All, 2002.
2. **An Introduction to Database Systems**, C. J. Date, Addison-Wesley Publishing Company, New York, 1995.
3. **Database Management Systems**, Ramakrishnan, R., New York: McGraw-Hill, 1998.
4. **SQL. Dialecte DB2, Oracle, Visual FoxPro**, M. Fotache, ed. Polirom, 2001.
5. **Baze de date – Visual Foxpro 6.0 – îndrumar de laborator**, Marian Popescu, Adrian Runceanu, Editura Academica Brâncuși, Târgu-Jiu, 2007.
6. **Baze de date – o abordare Visual Foxpro**, Adrian Runceanu,
 - Mihaela Runceanu, Editura Academica Brâncuși, Târgu-Jiu, 2009

Conținutul cursului

În cadrul acestui curs se vor studia bazele de date prelucrate cu ajutorul sistemului de gestiune a bazelor de date

ORACLE Database 11g Express Edition



Conținutul cursului

Limbajul de interogare a bazelor de date în care se vor face exemplificările noțiunilor teoretice va fi **SQL (Structured Query Language)**.



Capitolele cursului

1. **Sisteme de baze de date.** Modelul de date relațional. Modelul de date orientate obiect.
2. **Evoluția și facilitățile sistemului ORACLE.** Arhitectura sistemului ORACLE. ORACLE SERVER. Oracle Database 11g Express Edition
3. **Limbajul SQL.** Introducere. Prezentare generală. Cereri SELECT pe o tabelă
4. **Cereri SELECT pe o tabelă.** Clauza WHERE. Clauza ORDER BY
5. **Funcții.** Funcții referitoare la o singură înregistrare
6. **Funcții referitoare la mai multe înregistrări** (Funcții de grup). Clauza GROUP BY. Excluderea grupurilor (clauza HAVING). Imbricarea funcțiilor de grup

Capitolele cursului

7. **SUBQUERIES** (Subinterogări). SINGLE ROW SUBQUERIES. MULTIPLE ROW SUBQUERIES

8. **Cereri din mai multe tabele** (JOIN-uri). JOIN-urile proprietatea ORACLE

9. **Cereri din mai multe tabele** (JOIN-uri). JOIN-urile ANSI/ISO SQL99. Operatorii pe mulțimi

10. **Limbajul de manipulare al datelor** (LMD). Tranzacții(Transactions)

11. **Constrângeri**(Constraints)

12. **Vederi** (Views)

13. **Alte obiecte din baza de date**. Gestiunea utilizatorilor

Orice firmă utilizează baze de date pentru păstrarea și gestionarea informațiilor. Câteva astfel de aplicații sunt spectaculoase:

1. **bazele de date ale liniilor aeriene** care sunt accesate simultan din sute de agenții pentru a realiza rezervări și vânzări de locuri pentru date și zboruri diferite
2. **bazele de date ale băncilor** care permit realizarea a mii de tranzacții zilnic
3. **bazele de date ale supermagazinelor** care sunt accesate atât de la casele de marcaj cât și de la echipamentele de inventariere
4. **bazele de date ale bibliotecilor** care păstrează milioane de titluri și permit localizarea unei lucrări folosind diferite criterii (cuvinte cheie, titlu, autori, domeniu)

Pentru realizarea unei aplicații care folosește baze de date se poate proceda în două moduri:

- a) Se creează baza de date cu ajutorul unei aplicații de tip server de baze de date și se scriu apoi aplicațiile care accesează baza de date într-un limbaj care posedă funcțiile necesare accesării server-ului (frecvent se folosesc limbajele C++, Java, C# sau Visual Basic)
- b) Se folosește o aplicație de tip sistem de gestiune de baze de date (S.G.B.D. sau D.B.M.S. - DataBase Management System).
- c) Un astfel de sistem oferă un *ansamblu de instrumente software cu ajutorul cărora se crează atât baza de date cât și aplicațiile prin care aceasta este exploatată.*
- d) Pentru utilizatorii sistemului de operare Windows cele mai cunoscute sisteme de acest fel sunt Access și Visual FoxPro.

*Noțiuni introductive
despre
teoria generală a bazelor de date*

1.1. Sisteme de baze de date

1.2. Modelul de date relațional

1.3. Modelul de date orientate obiect

1.4. Modelul de date obiect-relațional

1.5. Modelul de date ierarhic

1.6. Modelul de date rețea

1.1. Sisteme de baze de date

Un **sistem de baze de date** este un sistem computerizat de evidență a informațiilor.

Informația într-un sistem de baze de date consta atât din date cât și din **informații despre date (metadate)** cum ar fi relațiile dintre date.



1.1. Sisteme de baze de date

Un sistem de baze de date poate fi considerat ca având patru părți:

1. date



2. utilizatori



3. hardware



4. software



1.1.Sisteme de baze de date

Datele:

Sunt informații pe care diferiți utilizatori (firme, agenții, sau simpli utilizatori) le colectează pentru a-și îndeplini scopurile sau misiunile.

Datele individuale sunt stocate în mulțimi de date relaționate (legate) numite **înregistrări**.

O colecție de înregistrări dependente se numește **bază de date**.



1.1.Sisteme de baze de date

Utilizatorii:

Diferite persoane sau grupuri de persoane care folosesc informațiile sunt definite ca utilizatori.



Hardware: De obicei noțiunea de hardware constă din device-uri fizice, cum ar fi harddisk-uri, imprimante, interfețe de intrare/ieșire și procesorul de date cu memoria sa asociată.



1.1. Sisteme de baze de date

The Oracle logo, consisting of the word "ORACLE" in white, uppercase letters on a red rectangular background.

Software: Interfața dintre datele fizice și utilizatorul se numește **Sistemul de Gestiune a Bazelor de Date (SGBD)**.

SGBD-ul este un sistem software, dar poate conține și hardware specializat pentru a gestiona mai eficient datele.

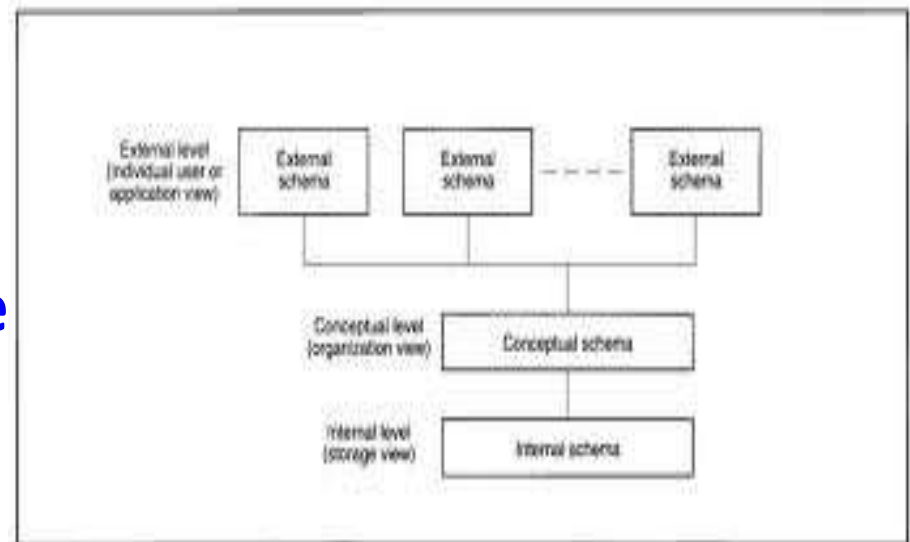
Aceste componente hardware pot fi harddisk-uri speciale care permit un acces mai rapid la date, sau multiprocesoare care permit procesarea paralelă de date.

1.1. Sisteme de baze de date

SGBD-ul furnizează diferiților utilizatori ai bazei de date, diferite modalități de lucru cu date în funcție de necesitățile fiecăruia.

Aceste diferite modalități de lucru cu datele reprezintă diferite nivele de abstractizare al datelor:

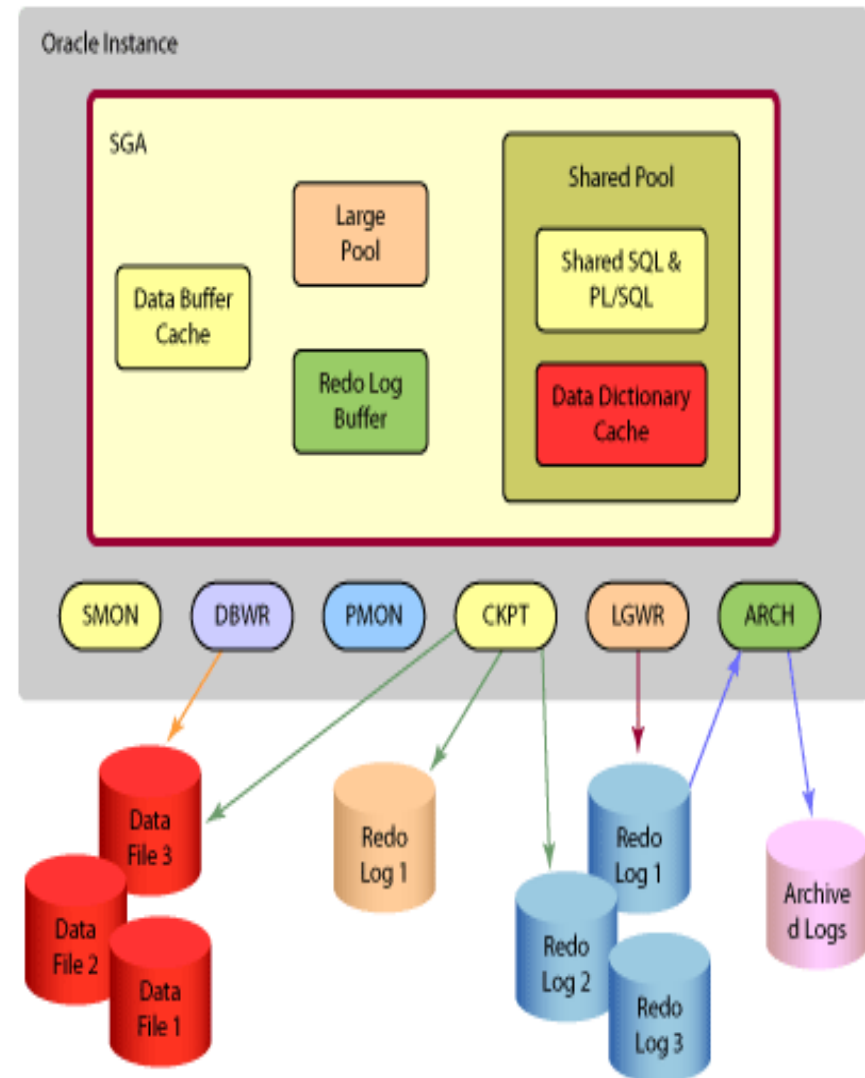
1. Nivelul fizic
2. Nivelul conceptual
3. Nivelul de vizualizare



1.1. Sisteme de baze de date

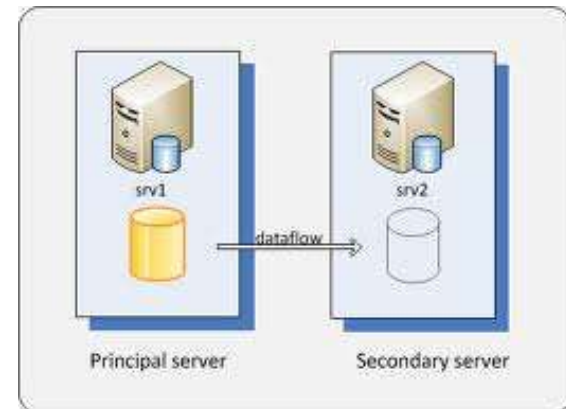
1. **Nivelul fizic** este cel mai de jos nivel de abstractizare.

De obicei, acest nivel este utilizat de programatorii **SGBD**-ului, care sunt interesați de cum anume se memorează datele pe suportul fizic.



1.1.Sisteme de baze de date

2. **Nivelul conceptual** este nivelul de mijloc al abstractizării, și care se concentrează pe descrierea datelor care sunt în baza de date și pe relațiile dintre aceste date.



De acest nivel de abstractizare sunt interesați:

- *Administratorii bazei de date*
- *Administratorii securității bazelor de date*

1.1.Sisteme de baze de date

3. **Nivelul de vizualizare** este cel mai înalt nivel de abstractizare.

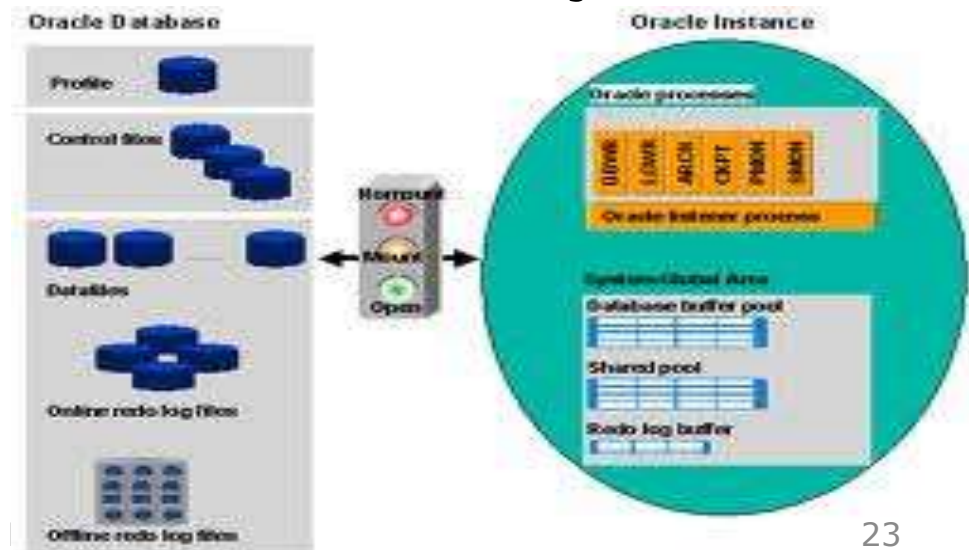
De obicei, acest nivel este modalitatea prin care utilizatorii finali folosesc datele.

Fiecare utilizator final are o vizualizare specifică asupra datelor pe care îl interesează.

Această vizualizare a datelor nu presupune ca utilizatorul să stie sau să înțeleagă caracteristicile interne ale datelor (cum ar modalitatea lor de reprezentare sau de stocare).

1.1.Sisteme de baze de date

Modelele bazelor de date permit diferențierea dintre descrierea bazei de date, care este specificată în schemă, și colecția de conținuturi sau de valori ale datelor din baza de date la un moment dat, care se numește instanță.



1.1.Sisteme de baze de date

Schema bazei de date utilizează un **limbaj de definire a datelor (DDL – Data Definition Language)**.

Manipularea datelor în baza de date (inserare, ștergere, actualizare, sau recuperare de valori de date) se poate face cu ajutorul **limbajului de manipulare datelor (DML – Data Manipulation Language)**.



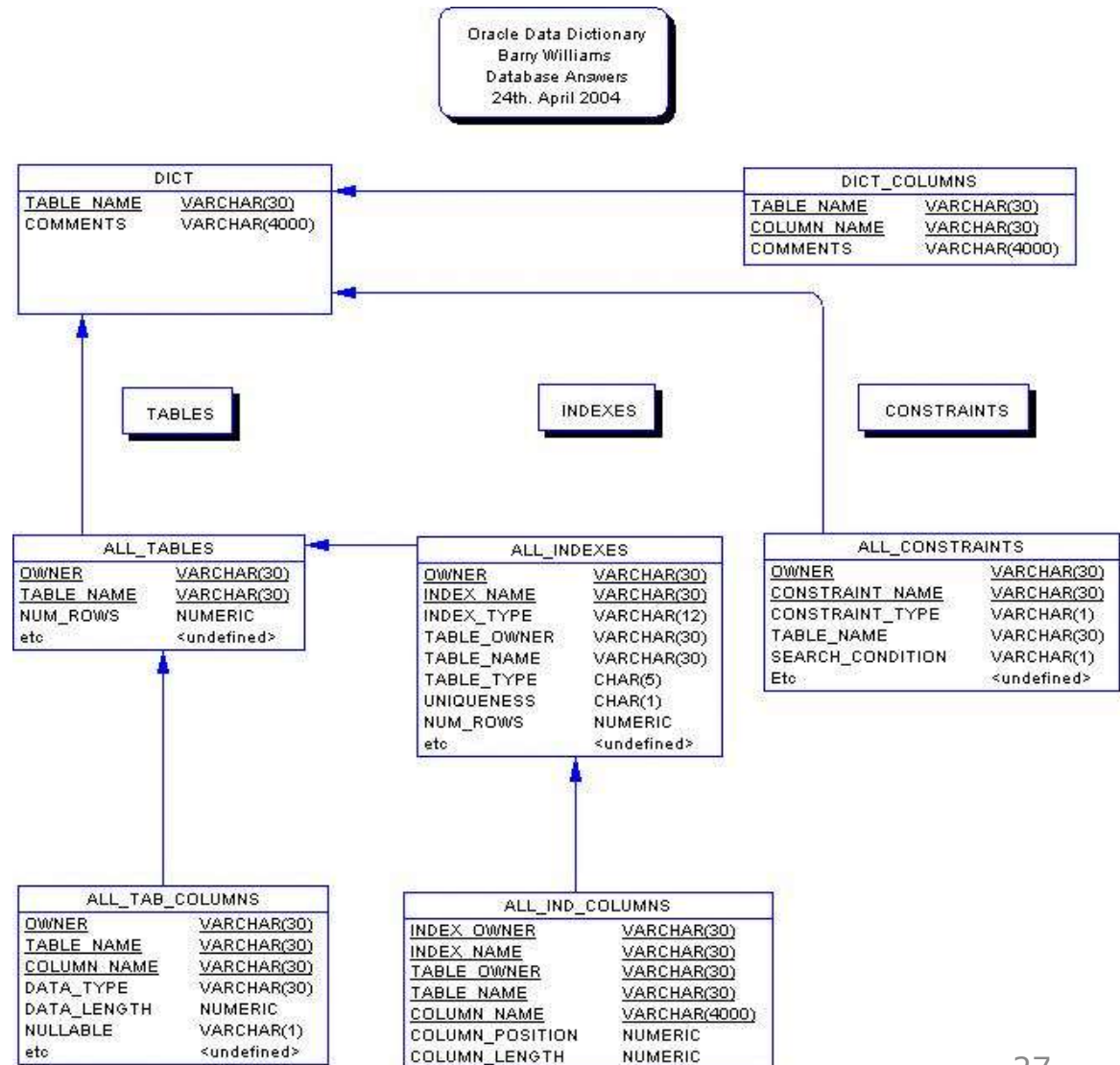
1.1.Sisteme de baze de date

- **precompilatorul limbajului de manipulare a datelor** care transformă instrucțiunile **DML** de la programele de aplicații la limbajul gazdă, și lucrează cu procesorul de interogări;
- **compilatorul limbajului de definiție a datelor** care transformă instrucțiunile **DDL** în tabele de metadate.

1.1.Sisteme de baze de date

Metadatele sunt memorate în **dictionarul de date**, care include:

- **structura bazei de date** sau **schema**
- **constrângerile de integritate**
- **constrângerile de securitate**



Modele de baze de date

Vom prezenta acum două din modele de baze de date care sunt cele mai utilizate:

1. Modelul de date relațional

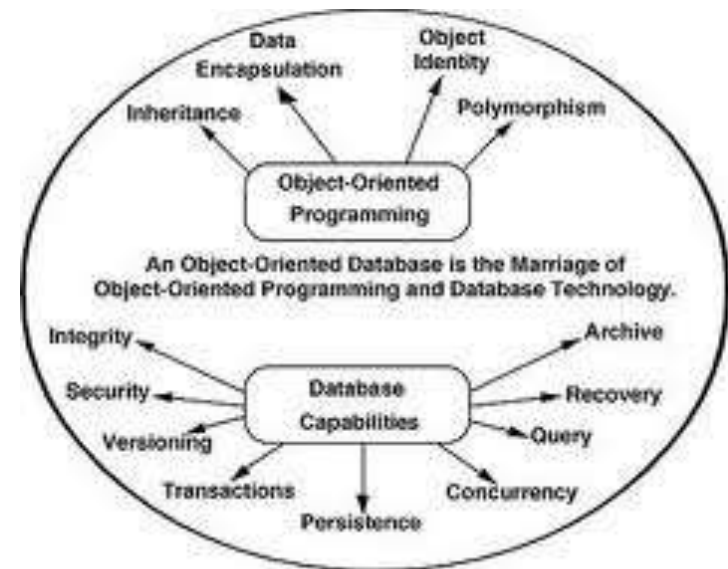
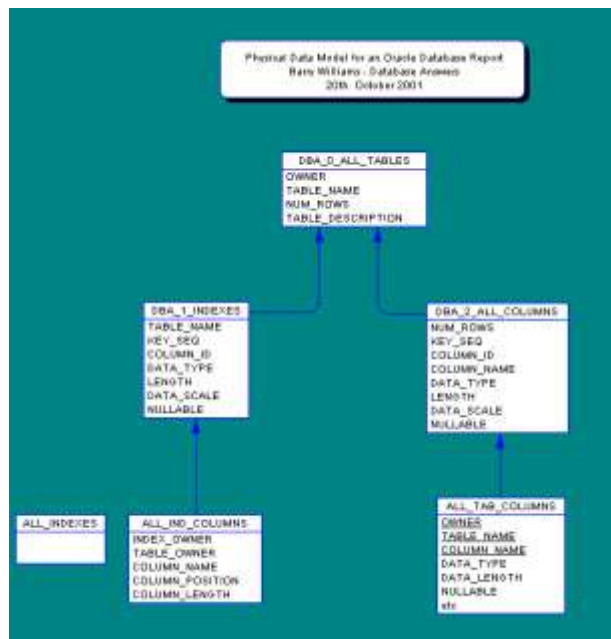


Figure 1. Makeup of an Object-Oriented Database

2. Modelul de date orientate obiect

*Noțiuni introductive
despre
teoria generală a bazelor de date*

1.1. Sisteme de baze de date

1.2. Modelul de date relațional

1.3. Modelul de date orientate obiect

1.4. Modelul de date obiect-relațional

1.5. Modelul de date ierarhic

1.6. Modelul de date rețea

1.2. Modelul de date relațional

O bază de date este reprezentată cu ajutorul **modelului relațional** ca o colecție de tabele.

Mult mai important, este faptul că acest model este direct legat de conceptul matematic de relație și este compus din:

1. **o parte structurală.** Schema bazei de date este o colecție de scheme de relații și o bază de date este o colecție de relații.
2. **o parte de integritate.** Chei primare și chei străine.
3. **o parte de manipulare.** Algebra relațională și calcul relațional.

1.2. Modelul de date relațional

Formal, o relație R este o mulțime, unde D_1, D_2, \dots, D_n sunt **domeniile** a n **attribute** A_1, A_2, \dots, A_n .

- ✓ Elementele relației sunt n -tuplurile (v_1, v_2, \dots, v_n) cu $v_i \in D_i$, adică valoarea celui de-al i -lea atribut trebuie să fie un element din mulțimea D_i .
- ✓ Elementele dintr-un tuplu sunt numite **câmpuri**.
- ✓ Când un câmp nu conține nici o valoare, vom reprezenta acest fapt cu o valoare specială numită **valoare null**, semnificând că “**nu este nici o intrare**” în loc de “intrarea este necunoscută”.

1.2. Modelul de date relațional

Prezentăm în continuare un exemplu de tabelă pentru gestiunea studenților dintr-o facultate:

CODSTUD	NUMEPREN	AN	FACULTATE	EDADM
101	Popa Ionel	4	Inginerie	10.00
102	Popescu Vasile	1	Drept	9.24
103	Badea George	1	Inginerie	8.79
104	Achim Mimi	2	Economic	9.31
105	Ionescu Mioara	3	Litere	10.00

1.2. Modelul de date relațional

Constrângeri de integritate

Constrângerile de integritate restricționează mulțimea tuplu-rilor teoretice posibile la o mulțime care este în mod practic cu o semnificație.

Fie X și Y două mulțimi cu unul sau mai multe attribute A_i , din schema relațională.

Spunem că Y este dependent funcțional de X , și notăm acest lucru prin $X \rightarrow Y$, dacă și numai dacă nu este posibil să avem două tupluri cu aceeași valoare pentru toate attributele din X dar cu valoare diferită pentru toate attributele din Y .

1.2. Modelul de date relațional

Cele mai importante constrângeri de integritate sunt:

Constrângerea de integritate a entității care stabilește că fiecare tuplu să fie unic identificat printr-o cheie și atributul cheie să nu poată fi **null**;

CODSTUD	NUMEPREN	AN	FACULTATE	GRUPA	MEDADM
101	Popa Ionel	4	Inginerie	145	10.00
102	Popescu Vasile	1	Drept	113	9.24
103	Badea George	1	Inginerie	112	8.79
104	Achim Mimi	2	Economic	126	9.31
105	Ionescu Mioara	3	Litere	132	10.00

1.2. Modelul de date relațional

Constrângerea referențială de integritate

stabilește că un n-tuplu dintr-o relație care se referă la o altă relație, trebuie să se refere la un n-tuplu care există în acea relație; această condiție se referă la cheile străine.

1.2. Modelul de date relațional

O **cheie candidat** a unei relații **R** este o mulțime minimă de attribute de care toate celelalte attribute ale lui **R** sunt dependente funcțional.

Cheia primară a unei relații **R** este una din cheile candidat care a fost desemnată în acest scop.

O **cheie străină** a unei relații **R** este o mulțime de attribute din schema relațională care formează o cheie primară pentru o altă relație.

CODSTUD	NUMEPREN	AN	FACULTATE	GRUPA	MEDADM
101	Popa Ionel	4	Inginerie	145	10.00
102	Popescu Vasile	1	Drept	113	9.24
103	Badea George	1	Inginerie	112	8.79
104	Achim Mimi	2	Economic	126	9.31
105	Ionescu Mioara	3	Litere	132	10.00

*Noțiuni introductive
despre
teoria generală a bazelor de date*

1.1. Sisteme de baze de date

1.2. Modelul de date relațional

1.3. Modelul de date orientate obiect

1.4. Modelul de date obiect-relațional

1.5. Modelul de date ierarhic

1.6. Modelul de date rețea

1.3. Modelul de date orientate obiect

Bazele de date orientate obiect permit crearea unor obiecte complexe din componente mai simple, fiecare având atribute proprii și comportament specific.

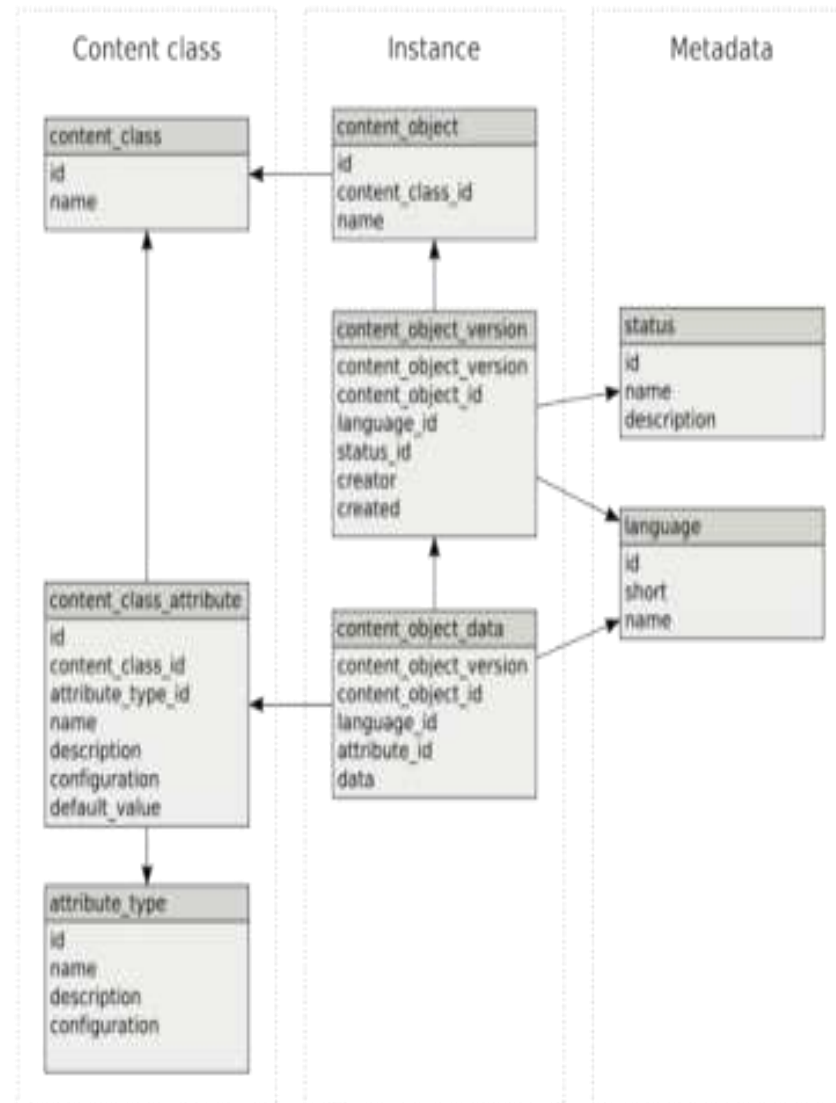
Aceste sisteme combină posibilitatea definirii și manipulării structurilor complexe de date cu funcționalitatea unui limbaj de programare și tehnologia de gestiune a bazelor de date.

1.3. Modelul de date orientate obiect

Modelele de date orientate obiect (MDOO) au fost create pentru a modela lumea reală.

De exemplu, limbajul C++ a fost dezvoltat pentru a crea modelul unui sistem telefonic.

Conceptul fundamental al unui MDOO este **obiectul**.



1.3. Modelul de date orientate obiect

Într-un **MDOO**, orice entitate din lumea reală este un obiect și reciproc, orice obiect reprezintă o entitate a lumii reale.

Un **obiect** reprezintă un grup de date structurate, identificate printr-o referință unică.

Un **obiect persistent** este un obiect stocat în baza de date care are o durată de viață mai mare decât a programului care l-a creat.

Un **obiect tranzitoriu** este un obiect depus în memorie, a cărui durată de viață nu depășește durata de viață a programului care l-a creat.

1.3. Modelul de date orientate obiect

Baza de date orientate obiect (BDOO) este o organizare coerentă de obiecte persistente, partajate de utilizatori concurenți.

Prin urmare, **BDOO** este rezultatul aplicării tehnologiei orientate obiect în domeniul stocării și găsirii informațiilor.

Schema unei **BDOO** trebuie să includă:

- definițiile structurale (**attribute** și **tipuri**)
- definițiile comportamentale (**metode**) ale obiectelor

1.3. Modelul de date orientate obiect

Un **sistem de gestiune al unei baze de date orientate obiect (SGBDOO)** trebuie să îndeplinească cerințele unui **SGBD** și să fie în plus, un sistem orientat pe obiecte.

Aceste două criterii generează o mulțime de caracteristici ale unui **SGBDOO**.

1.3. Modelul de date orientate obiect

Putem accepta ca definiție minimală:

SGBDOO = SGBD + obiect + moștenire + polimorfism

1.3. Modelul de date orientate obiect

Caracteristicile obligatorii ale unui SGBDOO sunt:

1. Manipularea obiectelor atomice și complexe (colecții imbricate).

Un **constructor** este o funcție asociată unei clase care permite crearea și inițializarea unui obiect (în memorie).

Un **destructor** este o funcție asociată unei clase care permite distrugerea unui obiect.

Noțiunea de **obiect complex** s-a născut prin aplicarea de constructori asupra obiectelor simple.

O condiție privind constructorii, referitoare la **MDOO**, o constituie **ortogonalitatea care presupune ca fiecare constructor să fie aplicabil fiecărui obiect.**

1.3. Modelul de date orientate obiect

2. *Persistența obiectelor.* Obiectele pot persista mai mult decât programul care a creat aceste obiecte.

3. *Concurența acceselor.* BDOO poate să fie partajată simultan de către tranzacțiile care o consultă și o modifică.

4. *Fiabilitatea obiectelor.* În cazul unei defecțiuni, obiectele trebuie restaurate la starea pe care au avut-o înainte de defecțiune.

1.3. Modelul de date orientate obiect

5. Ușurința interogării.

Un obiect poate fi găsit utilizând valorile atributelor sale, legăturile cu alte obiecte sau metodele aplicate acestuia.

6. Identitatea obiectelor.

Orice obiect trebuie să aibă un identificator sistem.

1.3. Modelul de date orientate obiect

7. Moștenirea (simplă).

O clasă poate fi specializarea altei clase și, prin urmare, poate să o moștenească.

Moștenirea reduce efortul de programare.

Există mai multe modalități de a moșteni și anume prin:

1. substituție
2. incluziune
3. restricție
4. specializare

1.3. Modelul de date orientate obiect

8. Polimorfismul.

Codul unei metode trebuie ales în funcție de parametrii săi.

9. Extensibilitatea.

SGBDOO trebuie să includă pe lângă clasele sale și tipurile predefinite și instrumentele care să permită utilizatorului definirea unor noi clase și tipuri.

1.3. Modelul de date orientate obiect

Dintre caracteristicile opționale ale unui **SGBDOO** amintim:

- ✓ ***Distribuția obiectelor.*** Această distribuție permite gestionarea obiectelor în diferite stații.
- ✓ ***Modelarea tranzacțiilor evaluate.*** Ideea este de a accepta tranzacții imbricate care pot fi descompuse în subtranzacții.
- ✓ ***Versiuni ale obiectelor.*** Plecând de la un anumit obiect, prin modificări succesive sau paralele, pot fi obținute mai multe versiuni ale obiectului.

1.3. Modelul de date orientate obiect

- ✓ **Moștenirea multiplă.** O clasă (subclasă) poate fi specializarea directă a unor supraclase și să moștenească proprietățile acestora.
- ✓ **Mesajele de eroare.** Este vorba de un mecanism de detectare și tratare a erorilor care implică faptul că dacă într-o metodă apare o eroare, este trimis un mesaj unei clase speciale definită anterior, care o va înregistra și o va trata corespunzător.

Noțiuni introductive despre teoria generală a bazelor de date

1.1. Sisteme de baze de date

1.2. Modelul de date relațional

1.3. Modelul de date orientate obiect

1.4. Modelul de date obiect-relațional

1.5. Modelul de date ierarhic

1.6. Modelul de date rețea

1.4. Modelul de date obiect-relațional

Modelul de date obiect-relațional (Object-Relational Model) reprezintă extinderea modelului relațional cu caracteristici ale modelului obiect, extindere necesară pentru realizarea bazelor de date care definesc și prelucrează tipuri de date complexe.

1.4. Modelul de date obiect-relațional

- ✓ În esență, modelul obiect-relațional păstrează structurarea datelor în **relații** (reprezentate ca tabele), dar adaugă *posibilitatea definirii unor noi tipuri de date*, pentru domeniile de valori ale atributelor.
- ✓ *Tipurile de date definite de utilizator pot fi extinse prin mecanismul de moștenire* și pentru fiecare tip sau subtip se pot defini metode pe care le pot executa obiectele de acel tip.

Noțiuni introductive despre teoria generală a bazelor de date

1.1. Sisteme de baze de date

1.2. Modelul de date relațional

1.3. Modelul de date orientate obiect

1.4. Modelul de date obiect-relațional

1.5. Modelul de date ierarhic

1.6. Modelul de date rețea

1.5. Modelul de date ierarhic

- ✓ În modelul de date ierarhic (Hierarchical Model) o bază de date se reprezintă printr-o *structură ierarhică de înregistrări de date (records) conectate prin legături (links)*.
- ✓ Modelul ierarhic a fost primul model folosit pentru dezvoltarea bazelor de date.
- ✓ Schema conceptuală a unei baze de date în modelul ierarhic se reprezintă printr-un număr oarecare de scheme ierarhice.

1.5. Modelul de date ierarhic

- ✓ O schemă ierarhică este un *arbore direcționat*, reprezentat pe mai multe niveluri, în care **nodurile sunt tipurile de înregistrări**, iar **arcele sunt tipurile de legături**.
- ✓ Fiecare nod (cu excepția nodului rădăcină) are o singură legătură către un nod de pe un nivel superior (nodul părinte) și fiecare nod (cu excepția nodurilor frunză) are una sau mai multe legături către noduri de pe nivelul imediat inferior (noduri fii).

Noțiuni introductive despre teoria generală a bazelor de date

- 1.1. Sisteme de baze de date**
- 1.2. Modelul de date relațional**
- 1.3. Modelul de date orientate obiect**
- 1.4. Modelul de date obiect-relațional**
- 1.5. Modelul de date ierarhic**
- 1.6. Modelul de date rețea**

1.6. Modelul de date rețea

Modelul de date rețea (Network Model) folosește o **structură de graf** pentru definirea schemei conceptuale a bazei de date:

- **nodurile grafului** sunt tipuri de entități (înregistrări, records),
- iar **muchiile grafului** reprezintă în mod explicit asocierile (legăturile, links) dintre tipurile de entități.

1.6. Modelul de date rețea

- ✓ La fel ca și modelul ierarhic, **dezavantajul** principal al modelului rețea este acela că *fiecare interogare trebuie să fie prevăzută încă din faza de proiectare*, prin memorarea explicită a legăturilor între tipurile de entități.
- ✓ În plus, complexitatea reprezentării datelor în modelul rețea este deosebit de ridicată, iar programatorii trebuie să o cunoască pentru a putea realiza aplicațiile necesare.

Sisteme de baze de date

Alte clasificari ale **sistemelor de baze de date**:

1. Clasificare după modelul de date
2. Clasificare după numărul de utilizatori
3. Clasificare este cea după numărul de stații pe care este stocată baza de date

Clasificare dupa numărul de utilizatori

1. Majoritatea sistemelor de baze de date sunt **sisteme multiutilizator**, adică permit accesul concurent (în același timp) a mai multor utilizatori la aceeași bază de date.
2. Există și un număr redus de **sisteme monoutilizator**, adică suportă accesul doar al unui utilizator (la un moment dat).

Clasificare după numărul de stații pe care este stocată baza de date

✓ Există două categorii de sisteme de baze de date:

1. centralizate

2. distribuite

1. **Un sistem de baze de date centralizat (Centralized Database System)** este un sistem de baze de date în care datele și sistemul de gestiune sunt stocate pe un singur calculator.
2. **Un sistem de baze de date distribuit (Distributed Database System)** poate avea atât datele, cât și sistemul de gestiune, distribuite pe mai multe calculatoare interconectate printr-o rețea de comunicație.

Întrebări?