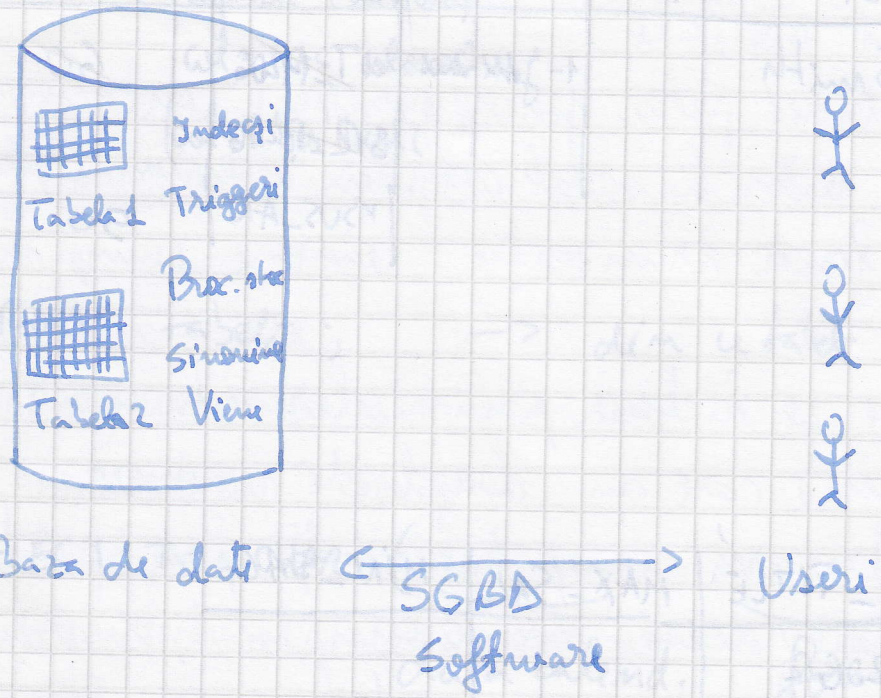
LABORATOR 1

18-02-2013



Activităţi

1) Proiectare

2) Creare

3) Prelucrarea/procesarea datelor

4) Administrare

Exemplu:

Angajat

Cod	Nume	Data Angajării
100	Smith	1-Jan-2010
200	Silver	10-May-2005

- tabele
- coloane
- linii
- tip de date

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	...	HIRE_DATE	JOB_ID	DEPARTMENT_ID
900	Smith		1-JAN-2000	~~IT_PROG~~	~~60~~
				~~SR_PROG~~	
				BUS_AN	50

JOBS

JOB_ID	JOB_TITLE	MAX_SAL	MIN_SAL
IT_PROG	PROGR		
SR_PROG	SENIOR PROGR		
BUS_AN	BUS AN		

JOB-HISTORY

EMPLOYEE_ID	START_DATE	JOB_ID	DEPARTMENT_ID
900	1-JAN-2000	IT_PROG	60
900	1-JUN-2002	SR_PROG	60
900	6-JUN-2006	BUS_AN	50

Conn name : grupa_grupa243
Username: GRUPA43
Password : bazededate

Hostname: 192.168.1.100
Port: 1521
SID: lab223

SELECT { * / col1, col2, col3 etc / expresie (col1-col2) / col1 AS "coloană" / col1 || 'șir' || col2 } -> (proiecție) ce coloane/expr să afișăm din tabelă

FROM tabelă; -> din ce tabelă

SELECT TO_CHAR ('data calend.', 'format șir')

FROM EMPLOYEES

ORDER BY ; -> sortează după anumite criterii liniile din rezultat (implicit ascendent)

Valorile NULL -> considerate cele mai mari

19-02-2013

CURS 1

„Proiectarea bazelor de date" - I. Popescu, L. Velcescu
Edit. Universității din București

Gestiunea (parțială) tranzacțiilor de vânzare-cumpărare de acțiuni la bursă prin intermediul societăților de valori mobiliare (SVM)

În cadrul burselor se încheie tranzacții cu valori

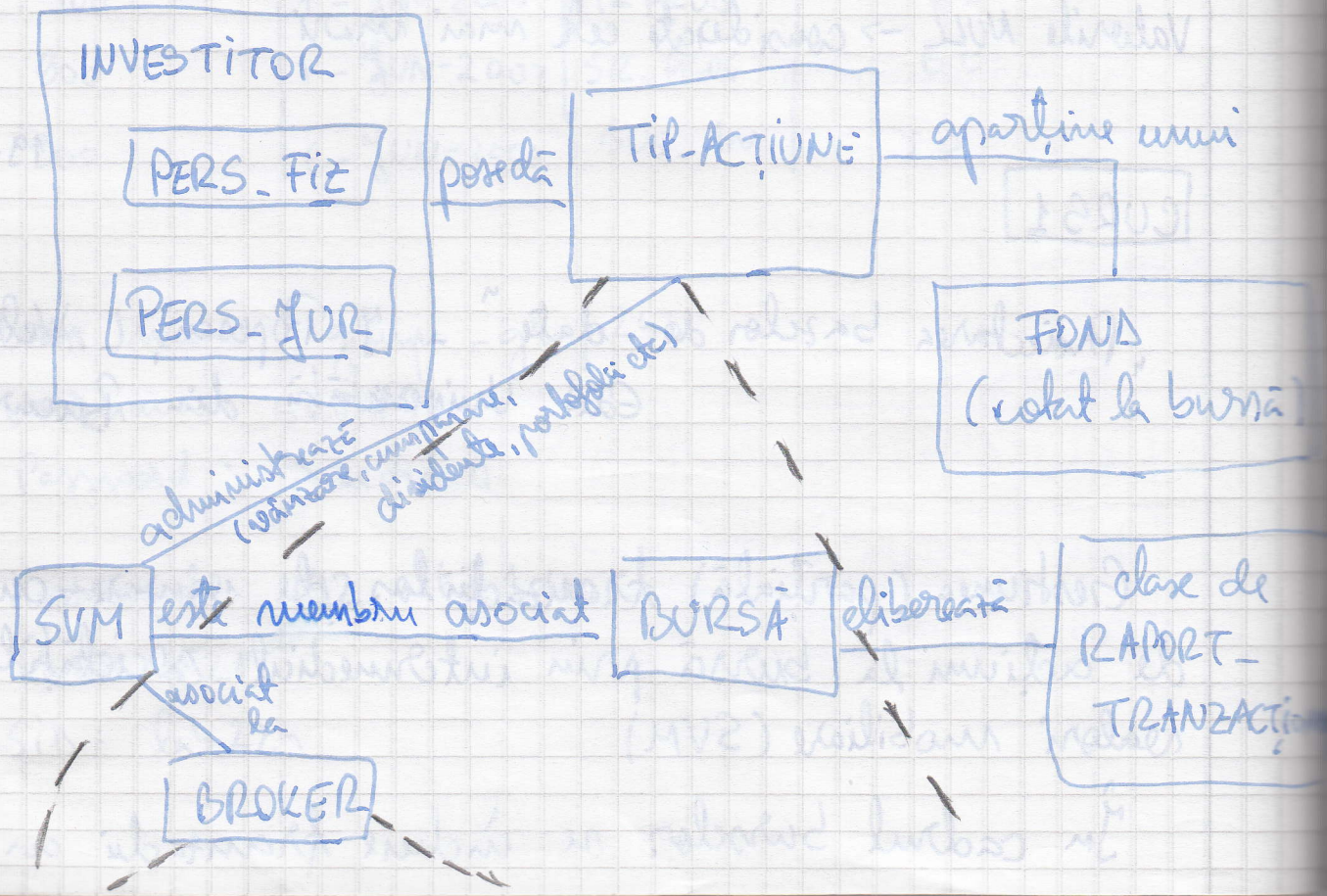
mobiliare, acțiuni, obligațiuni, titluri de rentă, bonuri de tezaur etc. Fiecare au specificul său și constrângeri speciale.

Investitorii (pers. fizice/juridice) contactează o societate de valori mobiliare în eventualitatea că doresc să facă tranzacții vânzare-cumpărare pe bursă.

Operațiile se încheie prin intermed. agenților de bursă (brokeri) care au drept de negociere.

La încheierea unei tranzacții pot participa mai mulți brokeri. De exemplu, X acțiuni sunt vândute de N brokeri, un neapărat distincți, la N momente de timp, fiind cumpărate de P brokeri, un neapărat distincți.

La diferite momente de timp sunt obținute clase de rapoarte de tranzacționare pentru vânzător, cumpărător, bursă, SVM etc.



ORDIN_VERBAL TRANZACTIE

Generalitati despre baze de date

1. Introducere
2. Gestiunea bazelor de date
3. Arhitectura SGBD
4. Evoluția BD-urilor
5. Arhitecturi multi-user pentru SGBD
6. Tehnologia Web și SGBD

1. Introducere

- Baze de date (persistent, integrat, partajat)
- Sisteme de gestiune a BD
- Administrator BD
- Dicționarul datelor (DD)

2. Gestiunea bazelor de date

- Arhitectura
- Cerințe minimale
- Niveluri de abstracție și percepție a datelor
- Independența datelor
- Limbaje pt BD
- Obiective
- Dezavantaje
- Structura unui SGBD

- Arhitectură
  - B.D.
  - SGBD
  - Dicționarul datelor (DD)
  - Mijloace hardware
  - Reglementări administrative destinate bunei funcționări a bazei
  - Personal implicat
    - administratori date și baze de date
    - designeri
    - programatori de aplicații
    - utilizatori finali

- Cerințe minimale

- Niveluri de abstractizare și percepție a datelor
  - 1) fizic
  - 2) conceptual, logic
  - 3) extern

- Independența datelor
  - fizică
  - logică

- Limbaje pt. BD
  - limbajul pt. definirea datelor (LDD)
  - limbajul pt. prelucrarea (manipularea) datelor (LMD)
  - limbajul pt. controlul datelor (LCD)

• LDD: creează, modifică, distruge } structura obiectelor limbajului

- LMD
  - inserare (INSERT)
  - ștergere (DELETE)
  - modificare (UPDATE)
  - consultare (SELECT)

  } non-procedurale

- LCD

- Obiective
- Dezavantaje
- Structura unui SGBD

25-02-2013

LABORATOR 2

Conn Name: X_243
Username: GRUPA43
Password: bazededate
Host: 192.168.1.100
Port: 1521
SID: lab223

} laborator

interogări multi-relație:

- Produs cartezian

SQL 2	SQL 3
SELECT * FROM MEDIC, SECȚIE;	SELECT * FROM MEDIC CROSS JOIN SECȚIE

- JOIN

  - θ - JOIN (THETA-JOIN)
    - EQUIJOIN
      - -> NATURAL JOIN
    - NONEQUIJOIN

  SELF JOIN

  OUTER JOIN (comprende esterni)
  - LEFT OUTER JOIN
  - RIGHT OUTER JOIN
  - FULL OUTER JOIN

- EQUIJOIN:

  SQL2

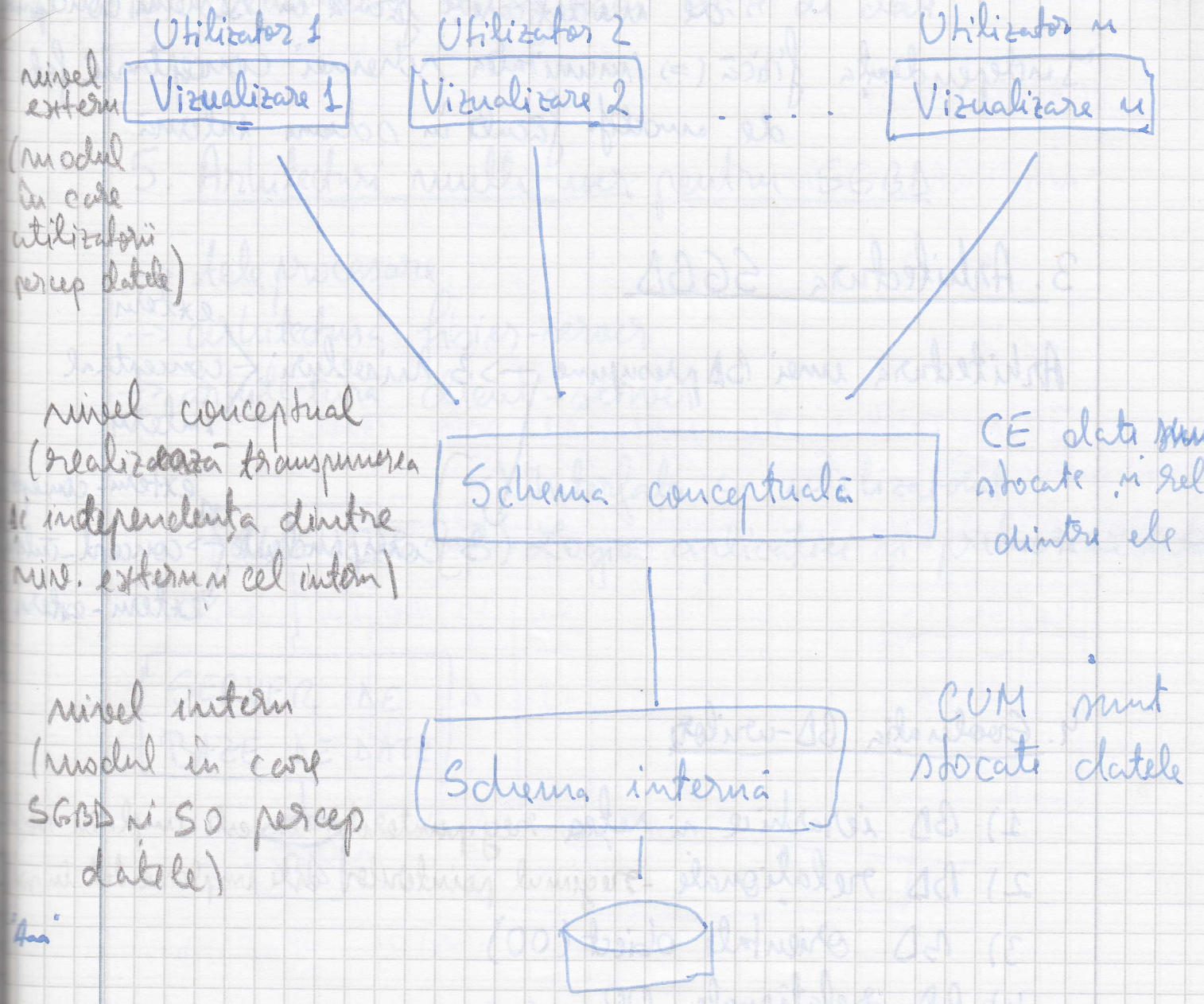
```
SELECT t1.c1, t2.c2, t3.c3
FROM TAB1 t1, TAB t2, TAB t3
WHERE t1.fk = t2.pk AND t2.fk2 = t3.pk AND t3.col=
```

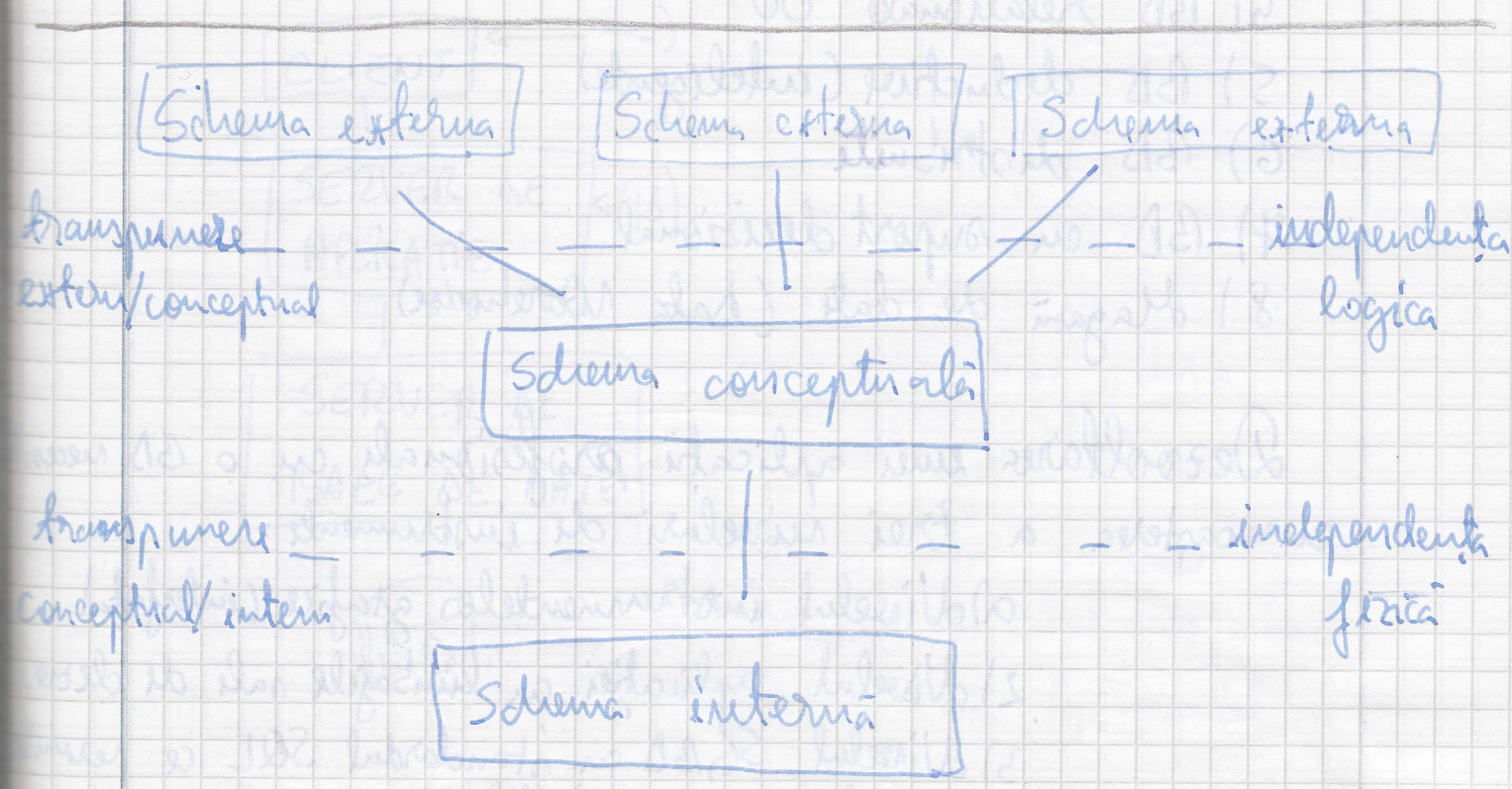
  SQL3

```
SELECT ....
FROM TAB t1 JOIN TAB t2 ON (t1.fk = t2.pk)
        JOIN TAB3 t3 ON (t2.fk2 = t3.pk)
WHERE t3.col = 'Aaa';
```

CURS 2

26-02-2013





Independența logică <=> imunitatea schemelor externe față de modificările făcute în schema conceptuală

Independența fizică <=> imunitatea schemei conceptuale fata de modif. făcute în schema internă

3. Arhitectura SGBD

Arhitectura unei BD presupune → 3 niveluri < extern, conceptual, intern

→ 3 corespondențe < extern-concept, concept-intern, extern-extern

4. Evoluția BD-urilor

1) BD ierarhice și rețea - reg. pointerilor nu este impl. în re
2) BD relaționale - regimul pointerilor este implementat în re
3) BD orientate obiect (OO)
4) BD relaționale OO
5) BD deductive (inteligente)
6) BD distribuite
7) BD cu suport decizional
8) Magazii de date (Data Warehouse)

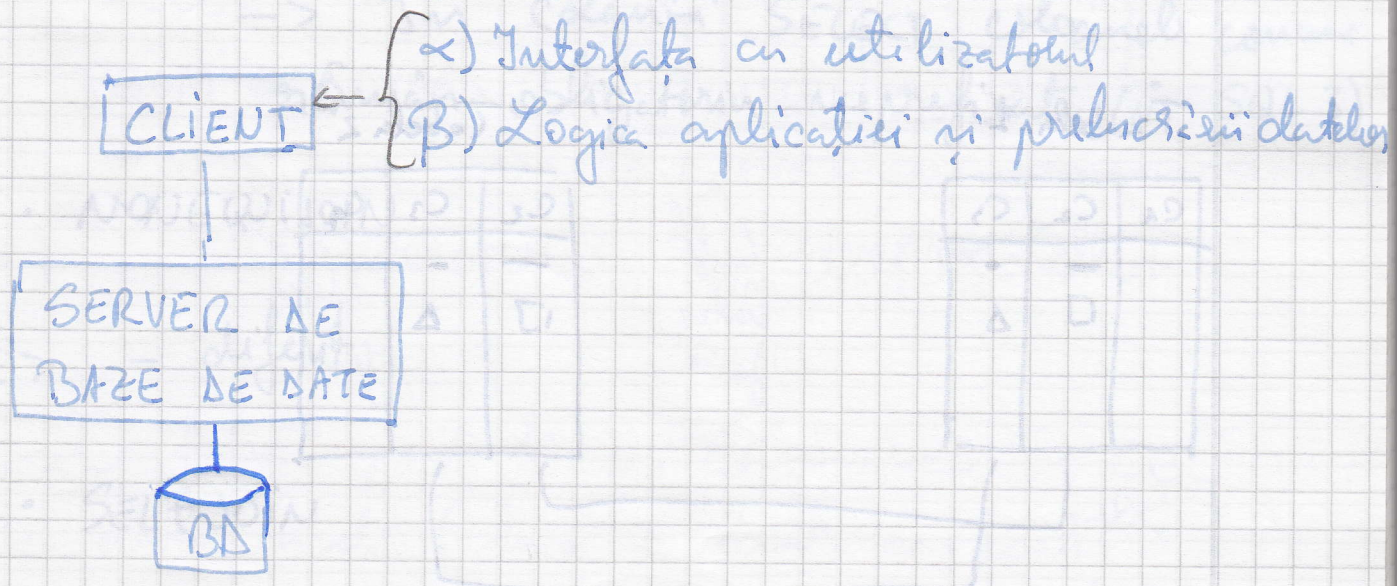
Dezvoltarea unei aplicații profesionale cu o BD nec cunoașterea a trei niveluri de instrumente:

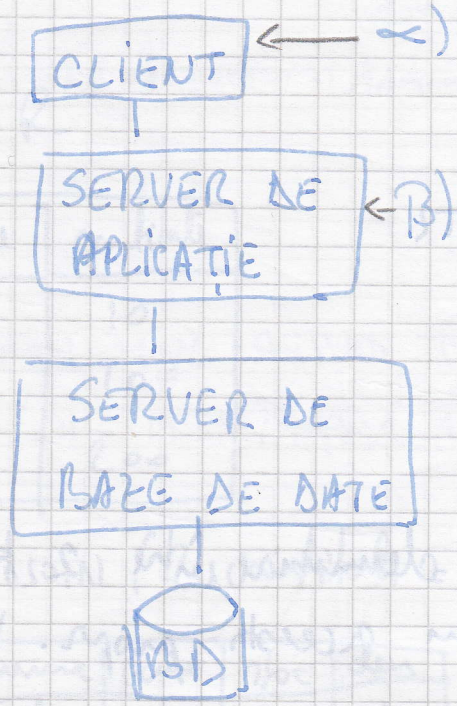
1) Nivelul instrumentelor grafice (interfața)
2) Nivelul aplicației cu limbajele sale de des
3) Nivelul SGBD cu standardul SQL ce per definirea, prelucrarea și controlul BD

9) Multi baze de date (baze de date federale)
10) Calculatoare și mașini baze de date

5. Arhitecturi multi-user pentru SGBD

→ teleprocesare
→ arhitectură fișier-server
→ arhitectură client-server





6. Tehnologia Web și sistemele SGBD

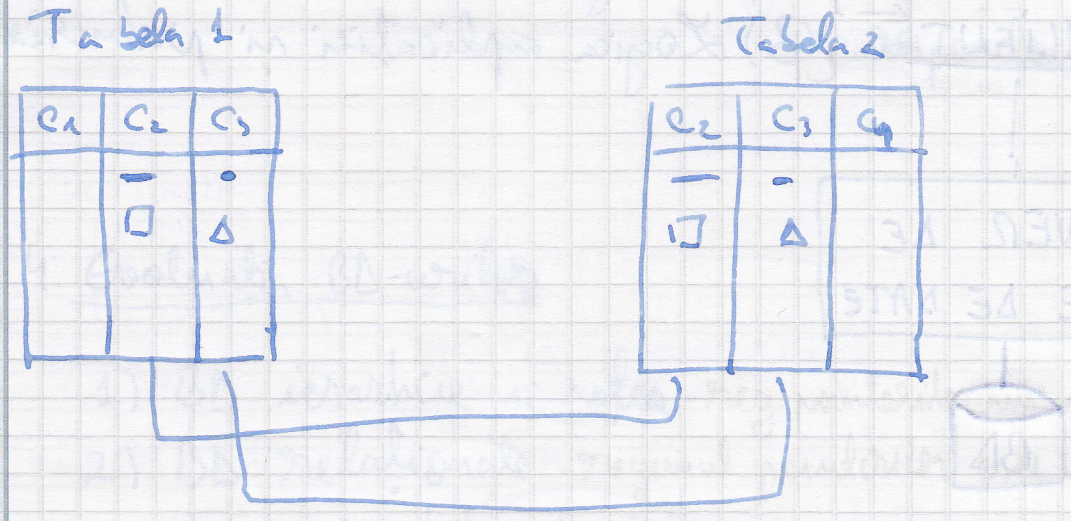
04.03.2013

LABORATOR 3

- NATURAL JOIN

SQL 3:

```
SELECT *
FROM Tabela 1 T1 NATURAL JOIN Tabela T2
```



SQL 2:

```
SELECT *
FROM Tabela1 T1, Tabela T2
WHERE T1.C2 = T2.C2
    AND T1.C3 = T2.C3;
```

Avem coloane cu aceeași denumire între 2 tabele. Unele dintre perechile de coloane cu aceeași propr. Vrem să le folosim pt. JOIN.

```
SELECT *
FROM TABELA1 T1 JOIN TABELA2 T2
    USING (C2, C3);
```

```
SELECT T1.C1, T2.C4, C2, C3
FROM TABELA T1 JOIN TABELA2 T2
    USING (C2, C3);
```

→ în clauza SELECT coloanele comune rămân obligatoriu neprefixate (în SQL 3)

- NONEQUIJOIN

<> - diferit

- SELFJOIN
- OUTER JOIN

R

nume	cantitate
A	10
B	100
C	200

S

stoc	den
10	D
200	E
300	F

EQUIJOIN (R.cantitate = S.stoc)

nume	cant	stoc	den
A	10	10	D
C	200	200	E

LeftOuterJoin (R.cantitate = S.stoc)

A	10	10	D
C	200	200	E
B	100	NULL	NULL

RightOuterJoin (R.cantitate = S.stoc)

A	10	10	D
C	200	200	E
NULL	NULL	300	F

FullOuterJoin = Left Outer Join UNION Right Outer Join

SQL2

```
SELECT ...
FROM EMPLOYEES E, DEPARTMENTS D
WHERE E.DEPARTMENT_ID = D.DEPARTMENT_ID(+)
```

LOJ

SQL 3

```
SELECT ...
FROM EMPLOYEES E LEFT OUTER JOIN DEPARTMENTS D
  ON (E.DEPARTMENT_ID = D.DEPARTMENT_ID)
```

SQL 2

ROJ

E.DEPARTMENT_ID(+) = D.DEPARTMENT_ID

SQL 2 - F.O.J.

```
SELECT ....
FROM EMPLOYEES E, DEPARTMENTS D
WHERE E.DEPARTMENT_ID = D.DEPARTMENT_ID (+)
      UNION E.DEPARTMENT_ID (+) = D.DEPARTMENT_ID;
```

elimină duplicate

SQL 3 - F.O.J.

```
SELECT ...
FROM EMPLOYEES E FULL OUTER JOIN DEPARTMENTS D
                 ON (E.DEPARTMENT_ID = D.DEPARTMENT_ID)
```

↳ nu elimină duplicate

→ O tabelă poate fi OUTER JOINED cu CEL MULT o altă tabelă (dacă e lanț se poate)

<u>Pentru Acasă:</u>

Conn name

Username : GRUPA43

Password : bazedate

Hostname : 193.226.51.37

Port : 1521

SID : o11g

Operatori pe mulțimi:

- UNION - elimină duplicate
- UNION ALL - nu elimină duplicate

```
SELECT C1 AS "Camp1", C2 AS "Camp2"
FROM
WHERE

UNION

SELECT C1, C2
FROM
WHERE ;
```

=>

Camp1	Camp2

- DIFFERENCE : MINUS

```
SELECT
...

MINUS
SELECT
...
```

- INTERSECT

Subcereri nesincronizate (necorelate)

```
SELECT T1.*
FROM Tabela1 T1
```

WHERE T1.col OPERATOR (SELECT T2.col5
pt rezultat → >=, <, =, > FROM Tabela2 T2
IN
etc. WHERE T2.C1 = 'A')

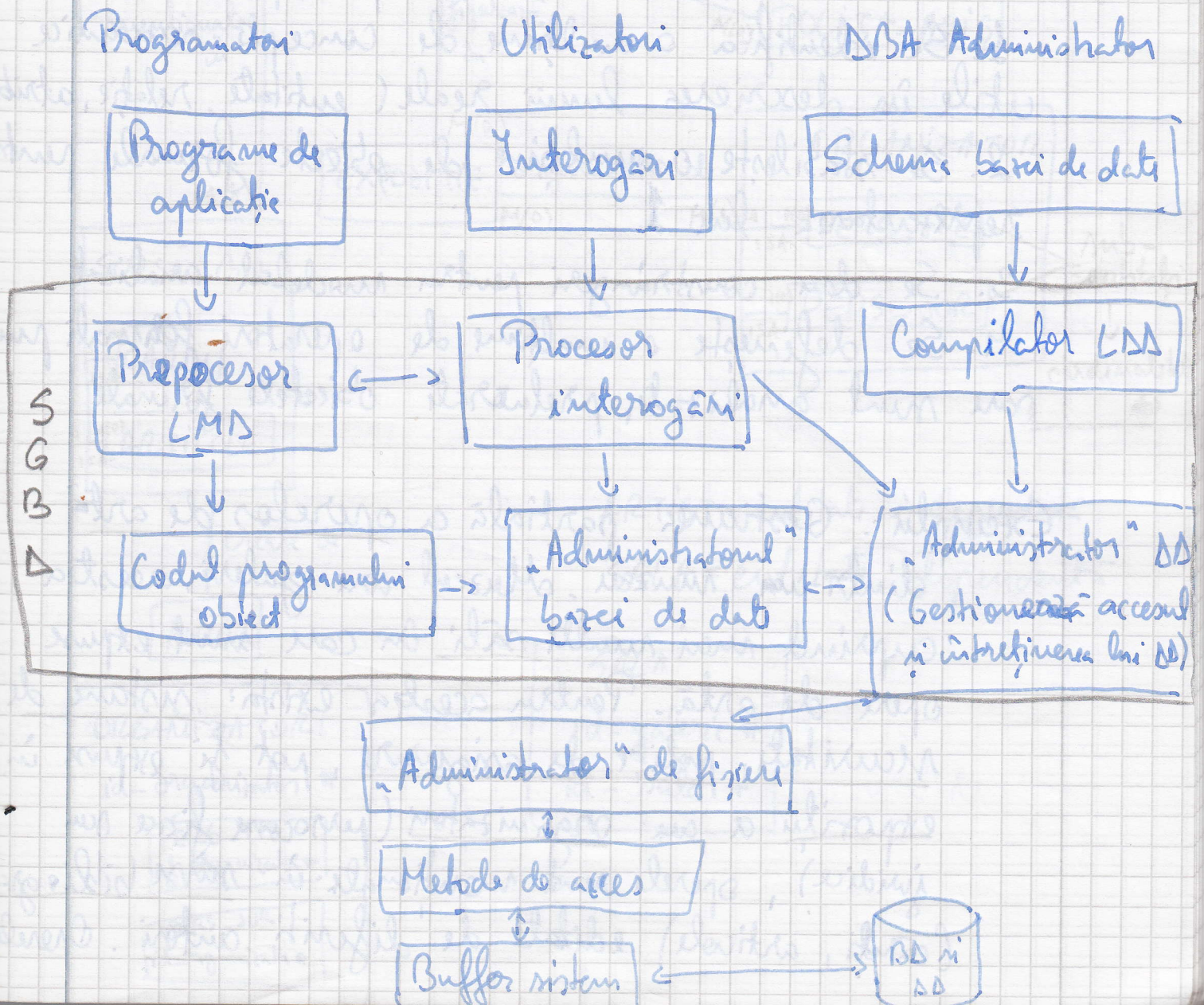
ORDER BY ... ;

→ e necorelată pt că în subcerere
nu este nimic legat de cererea părinte

05-03-2013

CURS 3

Componentele software principale ale unui SGBD



„Administrator BD":

1) Verifică dacă utilizatorul poate efectua operația
2) Procesează comanda
3) Verifică integritatea datelor
4) Determină strategia optimă
5) Administrează tranzacțiile
6) Planifică activitățile concurente

Diagrame E/R (entitate/relație)

Modelarea semantică a datelor presupune:

1. Se identifică o mulțime de concepte semantice utile în descrierea lumii reale (entitate, relație, atribut)
2. Se stabilește o mulțime de obiecte formale pentru reprezentarea lui 1
3. Se dau constrângeri pentru modelul analizat
4. Se definește o mulțime de operatori formali prin care sunt analizate, prelucrate obiectele formale

Exemplu: Gestiunea parțială a operelor de artă dintr-un muzeu. Muzeul are galerii, acestea cuprind mai multe săli în care sunt expuse opere de artă. Pentru acestea există: sisteme de securitate, polițe de asigurare, pot fi expuse în expoziții și au organizatori (persoane fizice sau juridice), operele sunt menționate în surse bibliografice (cărți, articole) editate de diferiți autori. Operele

de artă sunt create de artiști.

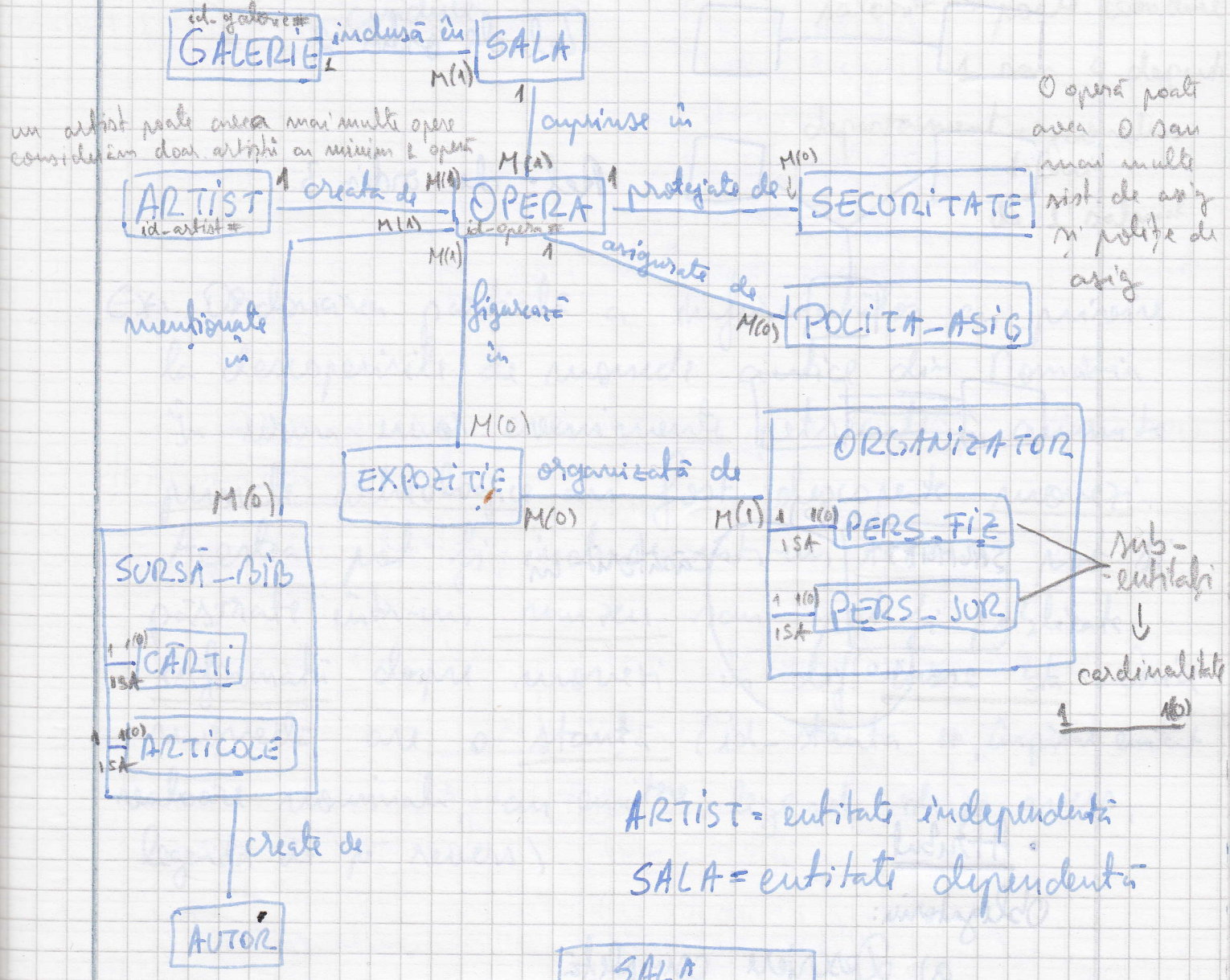
- **Entitate**

Obligatorie:

a) Descriere completă

b) Cheie primară

CHEIE PRIMARĂ { id_galerie #, id_galerie



ARTIST = entitate independentă

SALA = entitate dependentă

SALA
id_galerie #
id_sală #

ORGANIZATORI
id_organizator #
PERS_FIZ
id_organizator #
PERS_JUR
id_organizator #

• Relații

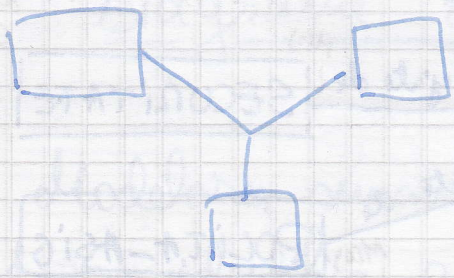
L= comunicare, legătură între 2 sau mai multe entitați

Obligatoriu:

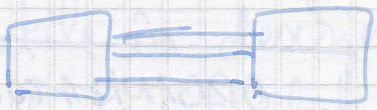
a) Descriere completă

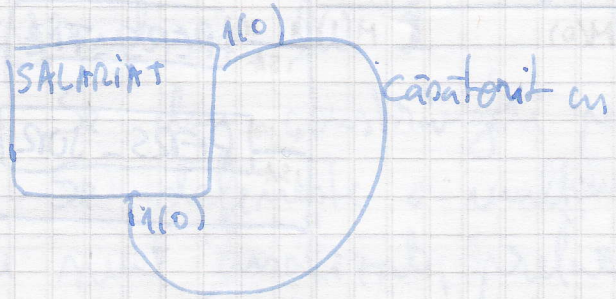
b) Cardinalitate maximă și minimă (jocul cuvintelor „poate" și „trebuie")

Rel. de ordin 2



Rel. de ordin 3





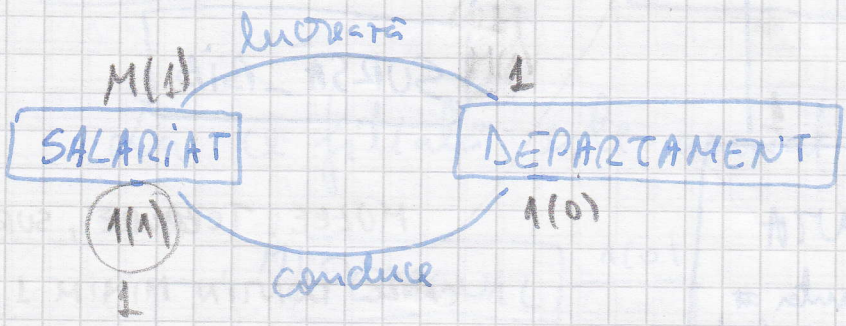
• Atribut

Obligatoriu:

a) Descriere completă

b) Specificarea numelui, tipul fizic, valori implicite, criterii de validare etc.

Atribut
- simplu (ex: salariu)
- compus (ex: adresa)
- cu valori multiple (ex: nr. telefon)
- derivat (ex: vârsta se obț. din data angajării)



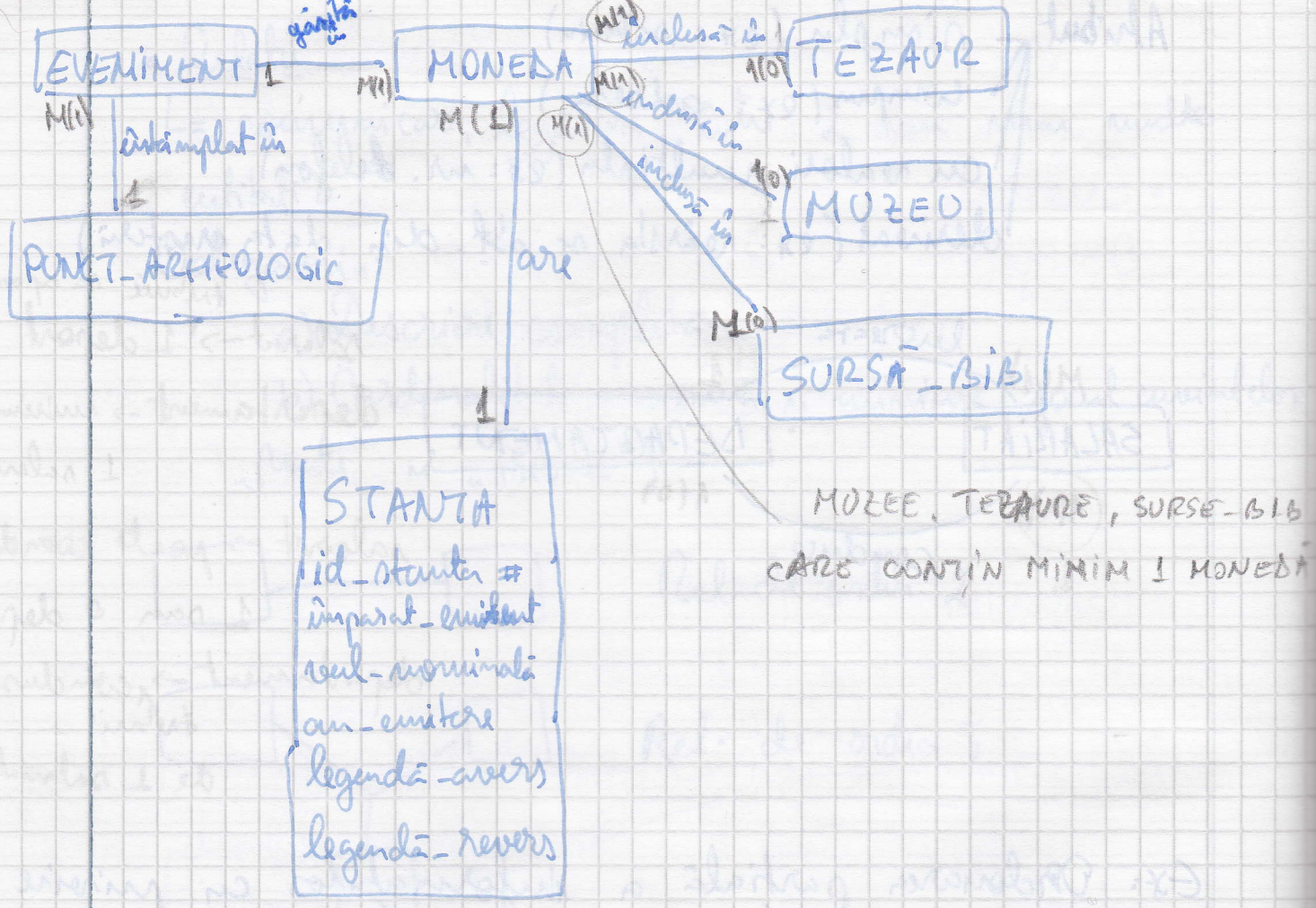
salariat → trebuie să aparțină 1 departament

departament → minim 1 salariat

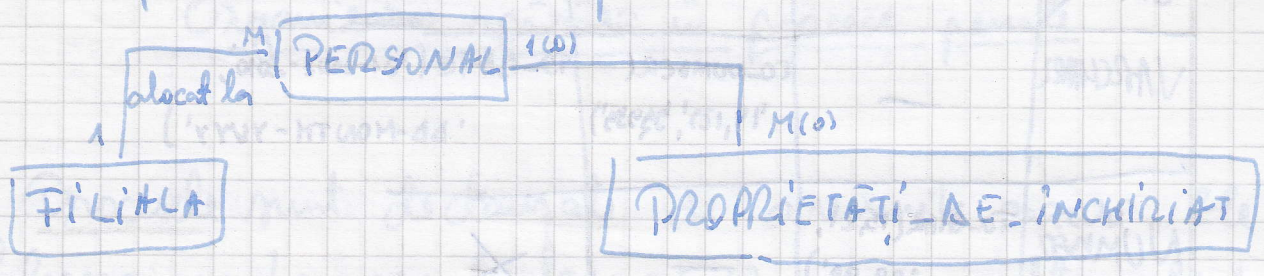
salariat → poate conduce 1 sau 0 departamente

departament → condus trebuie de 1 salariat

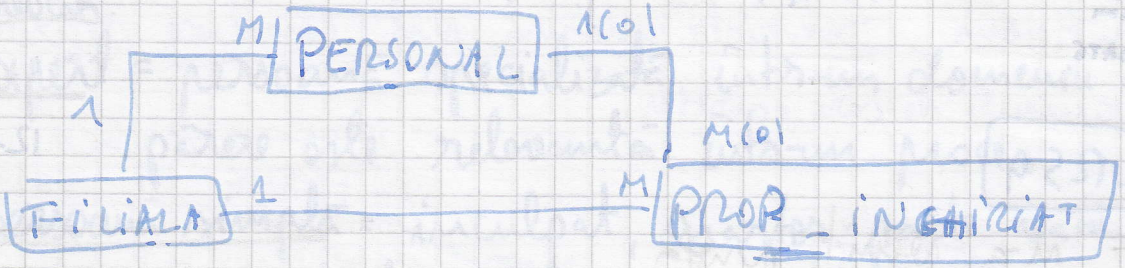
Ex: Ordonarea parțială a informațiilor cu privire la descoperirile de monede antice din România. În urma unor evenimente petrecute în anumite puncte arheologice au fost descoperite monezi. Acestea pot fi incluse într-un tezaur, pot fi păstrate într-un muzeu sau pot fi publicate informații despre monezi în dif. reviste sb. Orice monedă are o stantă (id_stanta #, împărat emitent, valoare nominală, an emitere, legenda de pe avers, legenda de pe revers)


EVENIMENT
găsită în
1
M(1)
MONEDA
M(1)
M(1)
M(1)

2) Capcană de întrerupere



La ce filială este asociat proprietatea XYZ?



↳ se adaugă atributul „id_filială" la proprietăți de închiriat

11-03-2013

LABORATOR 4

Funcții SQL

- conversie implicită — VARCHAR 2 <—> DATE
  VARCHAR 2 <—> NUMBER

- conversie explicită — TO_CHAR
  TO_DATE
  TO_NUMBER ('253.91', '999.99')
  format

Din \ Către	VARCHAR2	NUMBER	DATE
VARCHAR2	—	TO_NUMBER('14,123', '99,999')	TO_DATE('01-JUNE-2010', 'DD-MONTH-YYYY')
NUMBER 15,21	TO_CHAR(15,21, '99,99')	—	×
DATE HIRE_DATE sau SYSDATE	TO_CHAR(SYSDATE, 'DD-MM-YYYY')	×	—

CURS 4

12-03-2013

m-a venit profa

18-03-2013

LABORATOR 5

INSTR(şir, subşir, startpoz, a_câta_apariţie)

SUBSTR(şir, startpoz, un caract) = subşir

de tip compatibil

NVL(expr, val)

↳ dacă expr. este NULL se returnează val
altfel se returnează expr

COALESCE(e_1 ... e_n) → returnează prima valoare nenulă dintre e_1 ... e_n.

CURS 5 19-03-2013

Organizarea apărării în procese penale

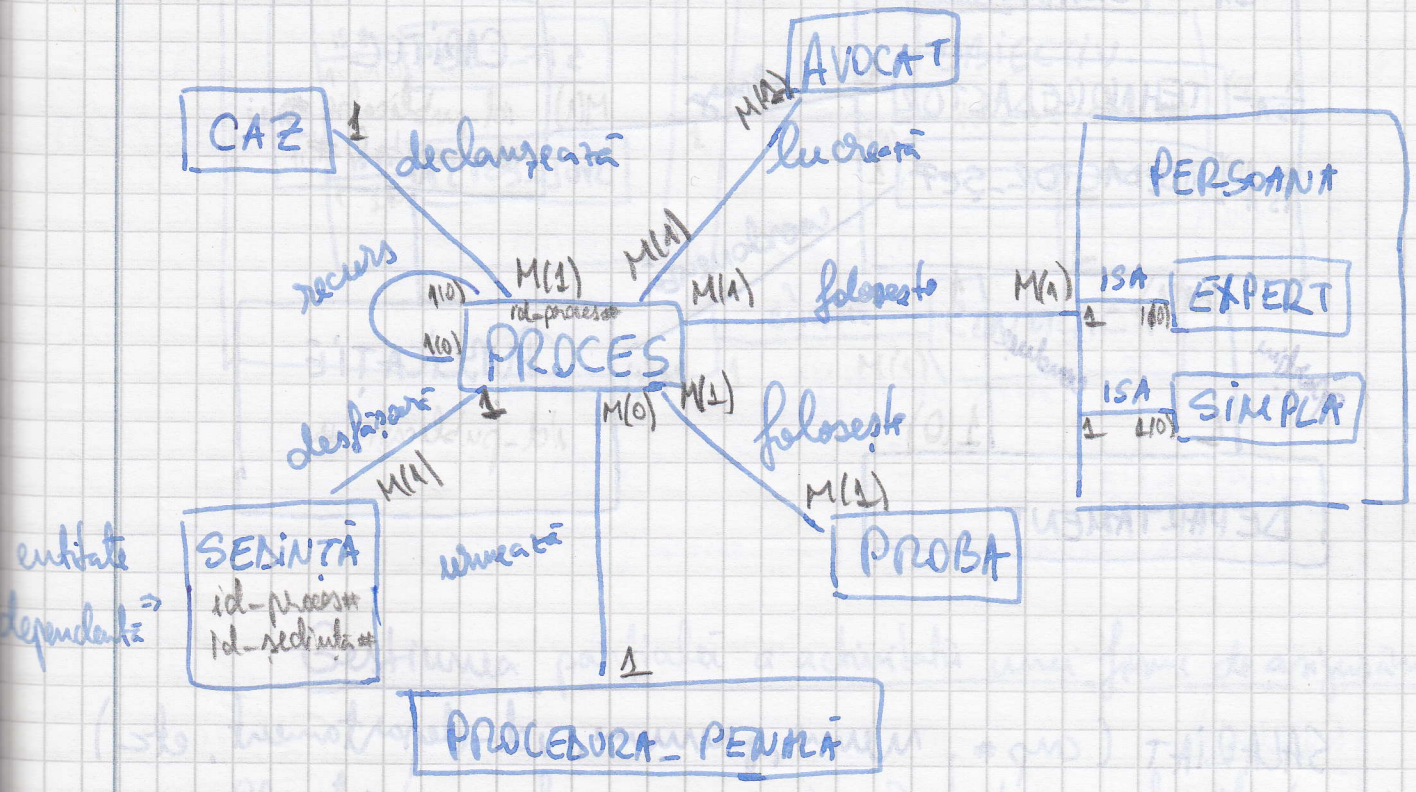
Procesele sunt declanșate de anumite cazuri. Ele folosesc probe, se desfășoară în mai multe ședințe, urmează o procedură penală. Deasemenea, procesul implică persoane speciale (expert, persoană simplă), avocați.

expert = persoană specializată într-un domeniu a cărui părere este relevantă într-un proces

persoană simplă = inculpat, martor, parte vătămată etc.

caz = orice situație reală care declanșează un proces

ședință = audiere într-un proces specificat



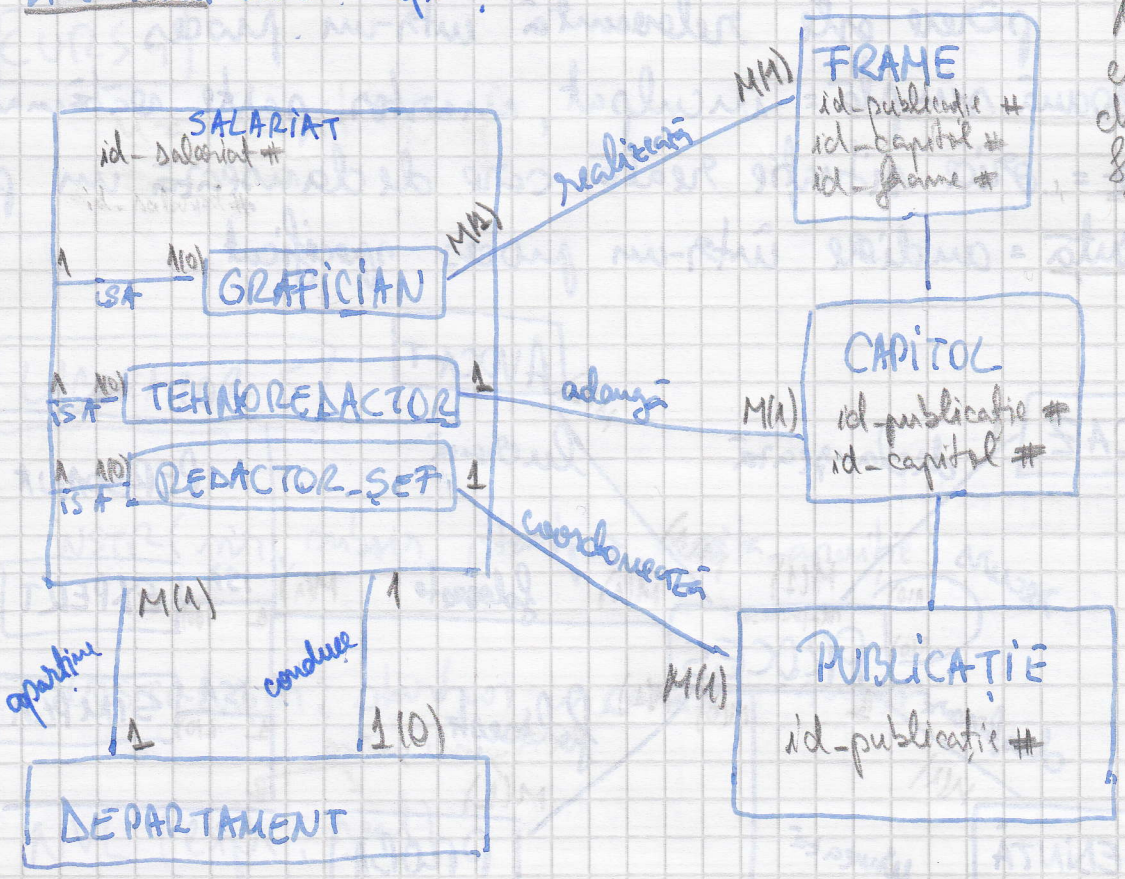
AVOCAT (id_persoană #, nume, prenume, id_casa_avocaturii, id_localizare, ...)

id_localizare → cere o nouă entitate LOCALIZARE

id_casa_avocaturii → cere o nouă entitate

Gestiunea parțială a activităților de editare dintr-o editură

Editura realizează publicații ce sunt coordonate de redactori șefi. O publicație poate cuprinde capitole editate în întregime de tehnoredactori. Capitolele conțin frame-uri ce pot fi realizate de unul sau mai mulți graficieni. Salariații editurii pot conduce diferite departamente de care aparțin.



Niciodată entitate cu cheie primară formată din 3 atribute!

SALARIAT (cmp #, nume, prenume, id_departament, etc)

Listă probleme

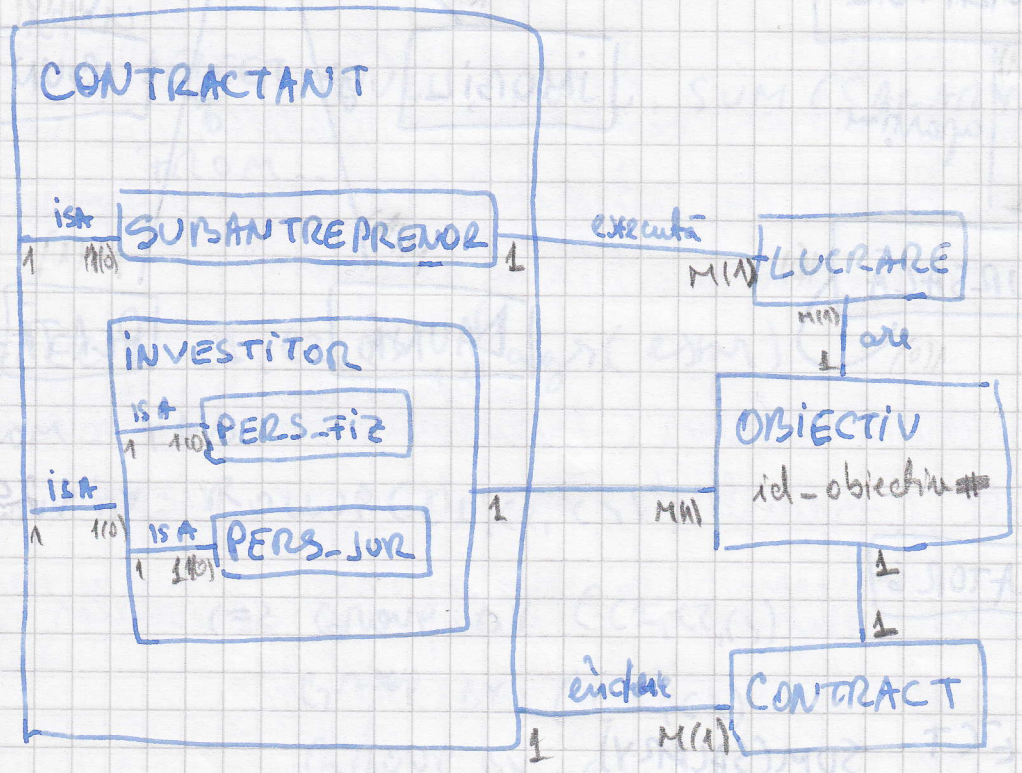
- Organizarea unui spital
- Organizații ecolog. și activ. privind mediul
- firma de const.
- repartiz. în căminele din România

TEMĂ

- societăți de asigurări
- gestiunea active. într-un stup de albine

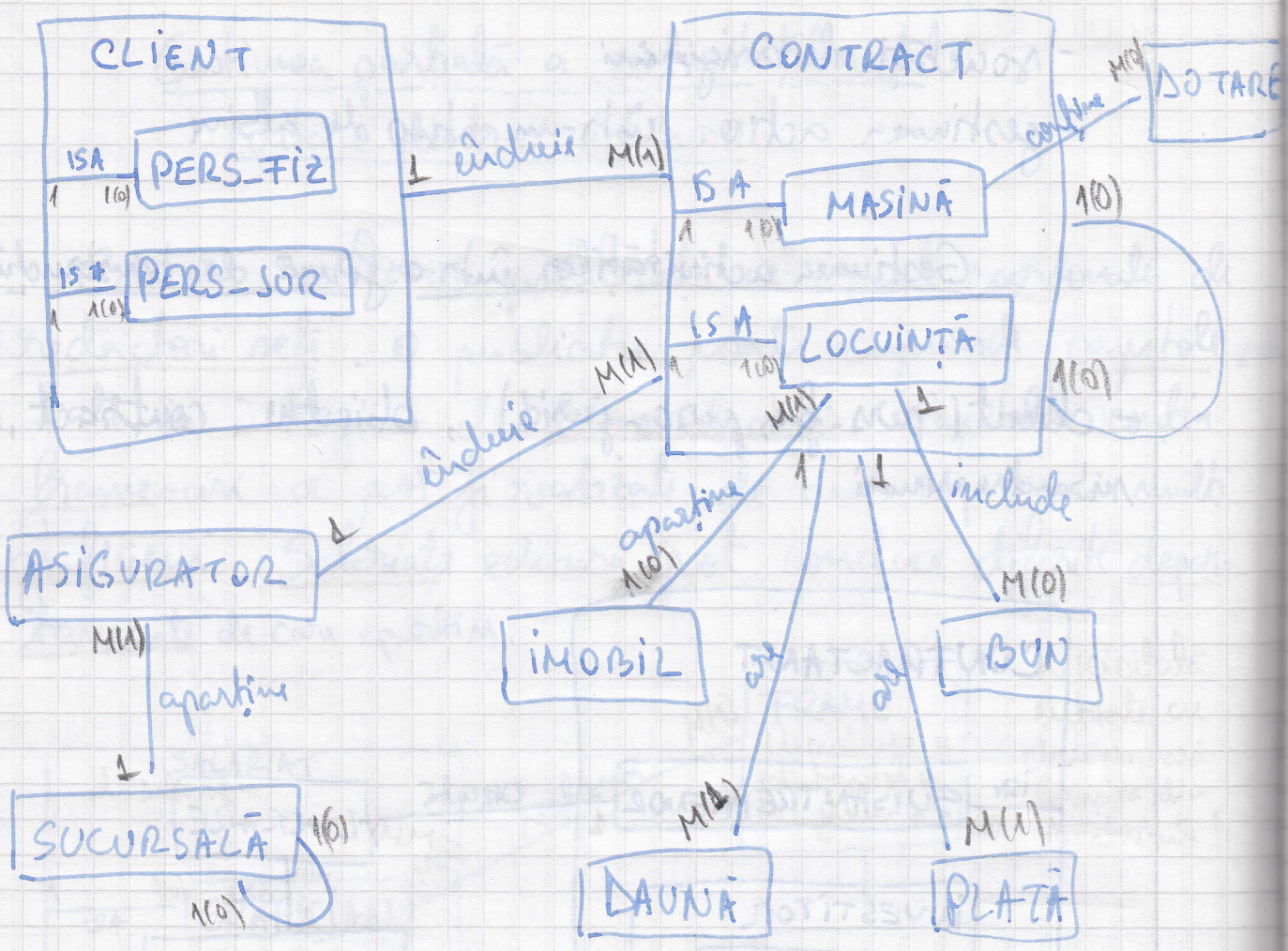
Gestiunea activităților într-o firmă de construcții

- client (pers. fiz, pers. juridi), obiective, contract, lucrări, subantreprenori



Gestiunea parțială a activității unei firme de asigurări

Clienții (pers. fiz, pers. jur) încheie contracte (mașină, locuință). Locuința aparține unui imobil și poate conține bunuri. Contractul este încheiat de un asigurator ce aparține unei succursale. CASCO poate conține niște dotări speciale. Contractele au asociate daune și plăți. Contractele pot fi reînnoite.



25-03-2013

LABORATOR 6

```
SELECT SUM(SALARY)
FROM EMPLOYEES E
HAVING SUM(SALARY) > 200000;
```

ROLLUP

```
SELECT DEPARTMENT_ID, JOB_ID, SUM(SALARY)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID, JOB_ID;
```

↳ înlocuim cu ROLLUP ——"—— → force subtotaluri

```
<=>   SELECT
      FROM
      GROUP BY DEPARTMENT_ID, JOB_ID
UNION
      SELECT DEPARTMENT_ID, NULL, SUM(SALARY)
      FROM EM
      GROUP BY DEPARTMENT_ID
UNION
      SELECT NULL, NULL, SUM(SALARY)
      FROM...
```

```
SELECT C1, C2, C3, f_agr(expr)
FROM TAB
GROUP BY ROLLUP(C1, C2, C3);
```

```
<=> GROUP BY (C1, C2, C3)
    GROUP BY (C1, C2)
    GROUP BY C1
    GROUP BY () -> na lipsacă clauza
                   GROUP BY
```

CUBE

CUBE(dep_id, job_id) -> toate combinațiile

```
<=> GROUP BY dep_id, job_id
    GROUP BY dep_id
    GROUP BY job_id
    GROUP BY () -> lipsa clauzei
```

26-03-2013

CURS 6

Modelul relațional:

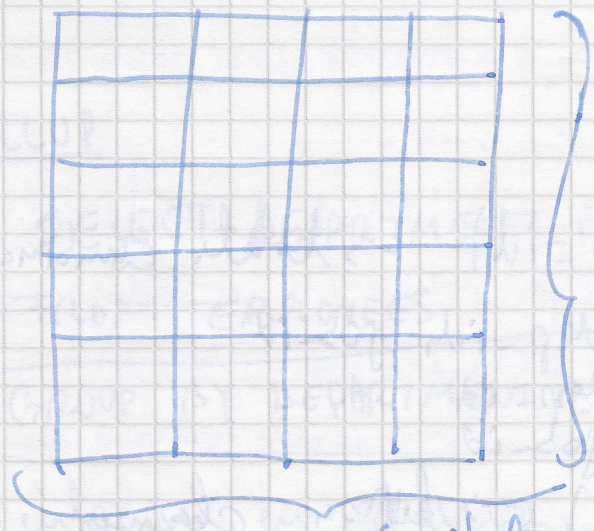
1. Caracteristici
2. Proiectare
3. Algebra relațională
4. Prelucrarea și optimizarea cererilor
5. Regulile lui Codd

1. a) structura relațională a datelor
   b) reguli de integritate
   c) operatorii modelului relațional

Structura datelor (Domeniu, Relație, Schemă relațională, Cardinalitate, Aritate, NULL, view)

Schema relațională = mulțimea atributelor corespunzătoare unei relații

STUDENT (#id_student, cnp, nume, prenume, data_naștere, id_adresă, id_grupă, ...)



m (cardinalitate = nr de linii)

n (aritate = nr. de coloane)

VIEW (vizualizare, filtru, relație virtuală, vedere)

1) Caracteristicile unei relații:

1) Denumire unică
2) Nu există dublurile tuplurilor
3) Fiecare celulă conține o valoare atomică
4) Fiecare atribut are nume unic
5) Ordinea atributelor nu are importanță
6) Toate valorile unui atribut aparțin aceluiași domeniu

Reguli de integritate

- Cheie primara, cheie candidat, supercheie, cheie secundară, index

O cheie identifică o linie și este diferită de un index, care localizează o linie

1. Cheia primară trebuie să fie unică și minimală
2. Atributele cheii primare trebuie să fie diferite de NULL

R (A#, B, C, ...)

P (X#, ..., A, ...)

↳ cheie externă (străină)

SALARIAT (id_sal # ..., id_dep, ...) → cheie externă

DEPARTAMENT (cod #, ..., id_manager, ...) → cheie externă

nu trebuie să aibă aceleași nume

3. O cheie externă trebuie să fie ori NULL în întregime ori să corespundă unei valori a cheii primare asociate.

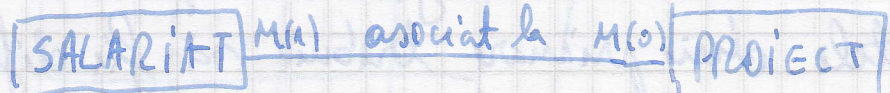
2) Proiectarea modelului relațional (crearea diagramei conceptuale)
(9 reguli → vezi curs)

X = poziția cheii externe

X̲ = această cheie externă este conținută în cheia primară

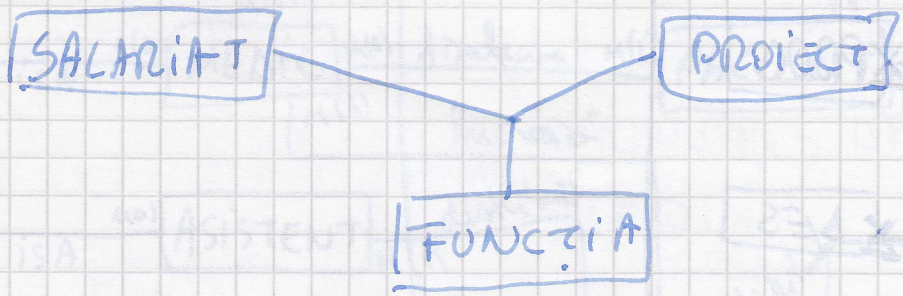
• Transformarea relațiilor

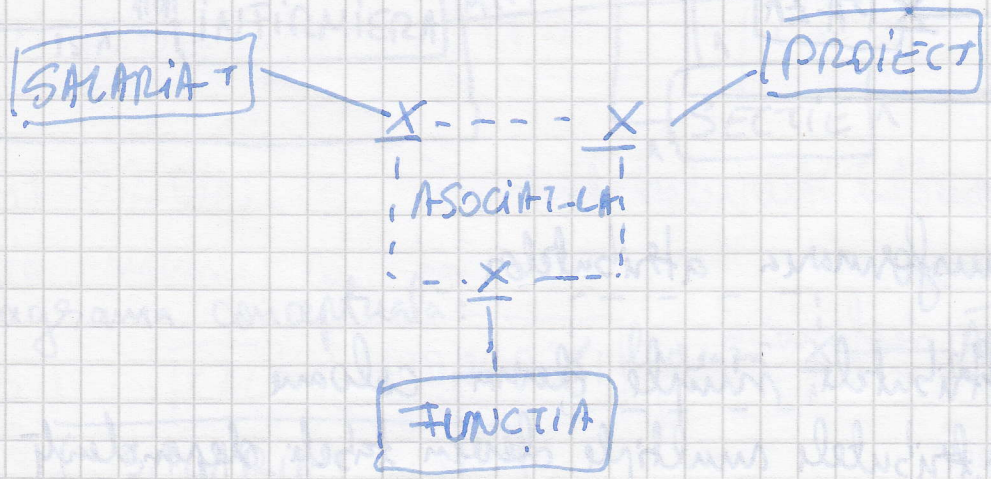
1) Rel. M:M devin tabele asociative (se desenează punctat). Cheia primară a tabelului asociativ este compunerea cheilor primare ale entităților pe care le leagă, plus eventuale atribute adiționale



ASOCIAT_LA(id_sal #, id_proiect #, data #, ...)

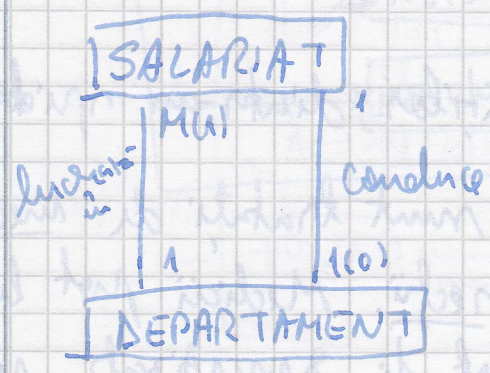
2) Rel. de tip 3 devin tabele asociative

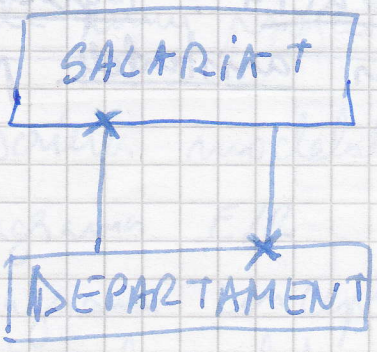


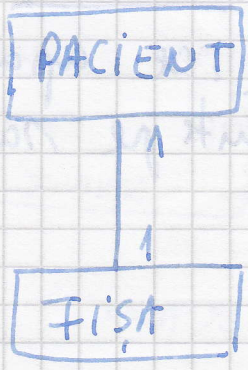


3) Rel. 1:1 și 1:M devin doi externe

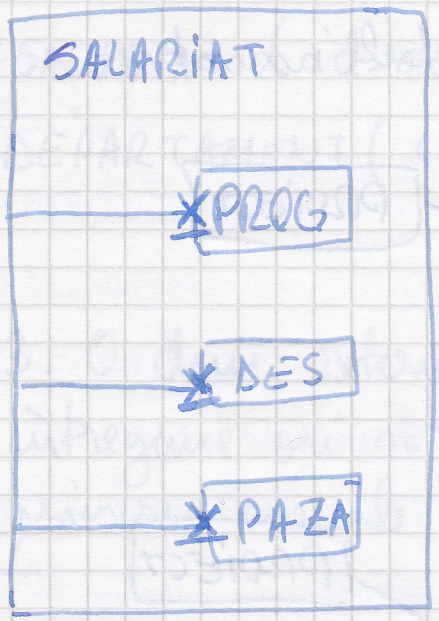
↳ unde e M punem FK







punem FK în ambele



• Transformarea atributelor

1) Atributele simple devin coloane
2) Atributele multiple devin tabele dependente
3)

Gestiunea parțială a activităților dintr-un spital

Pacienții (care au fișă med.) sunt tratați de medici care aparțin unor anumite secții. Medicii pot lucra și pe ambulanță. Pacienții pot fi repartizați pe saloane, ce aparțin de secții. Infirmierele sunt distribuite pe saloane, iar asistentele pe secții.

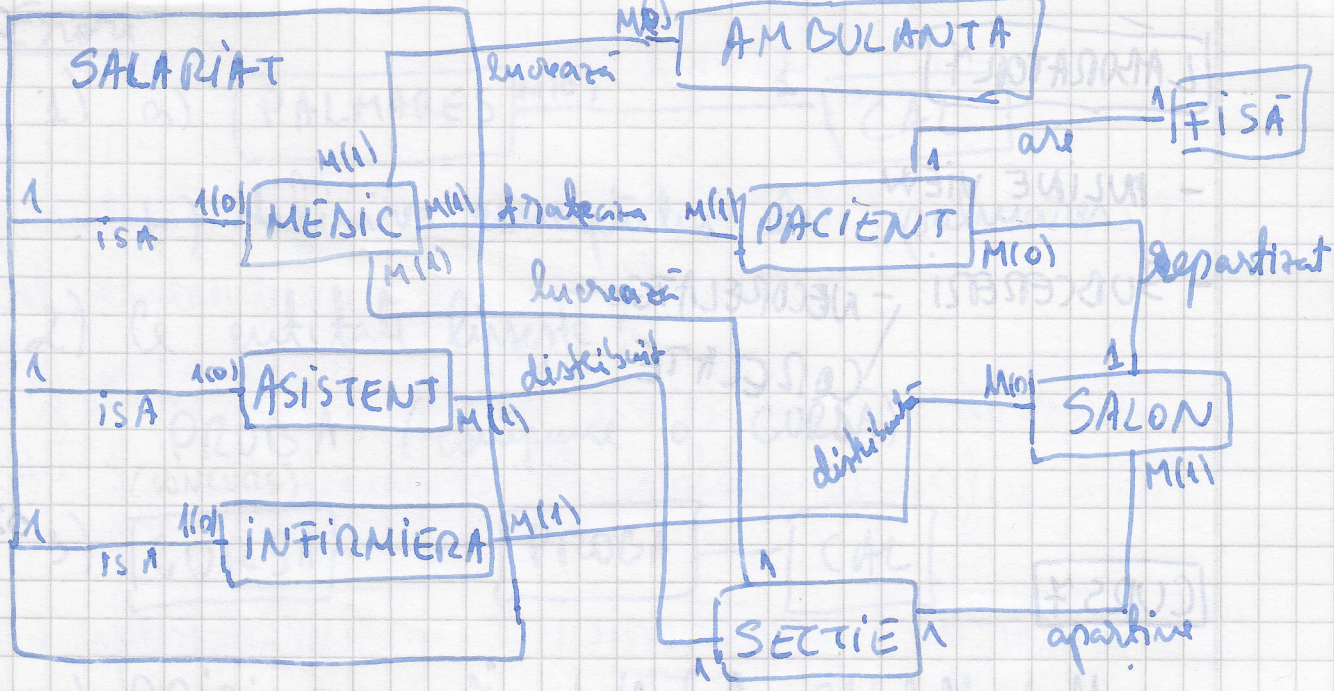
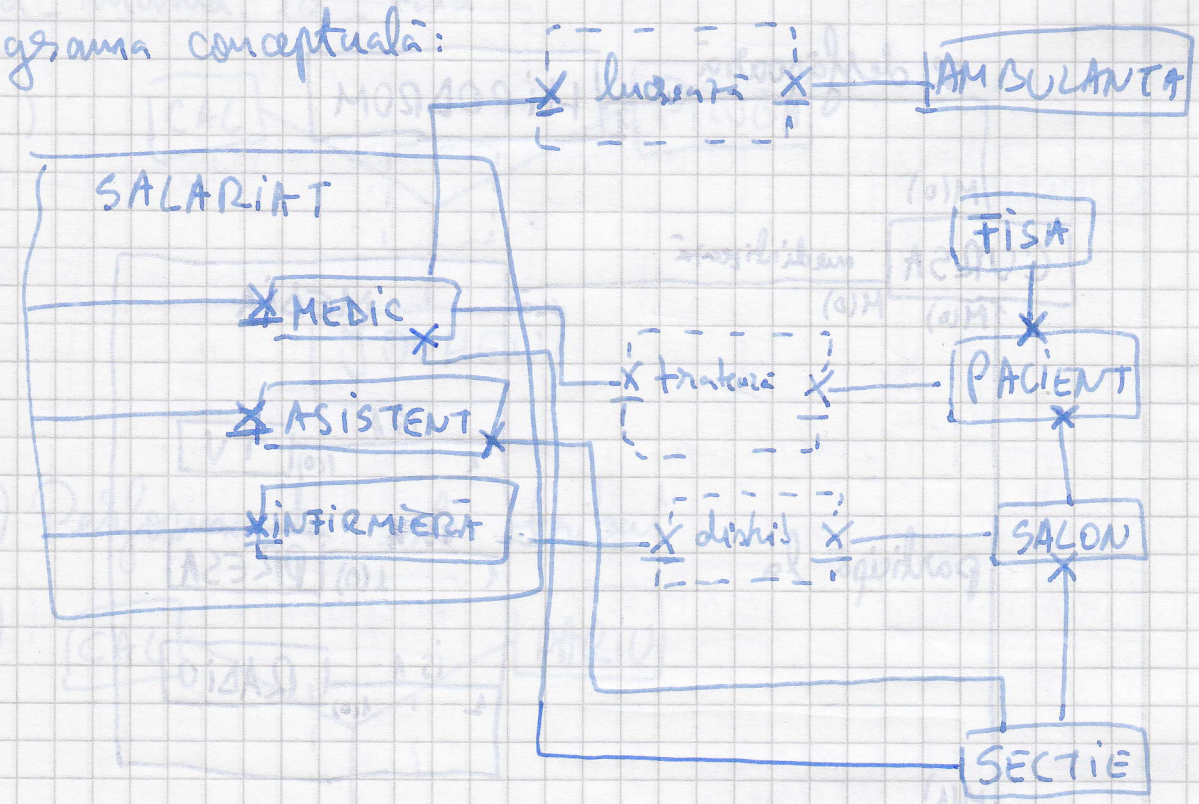


Diagrama conceptuală:



Etapele unui proiect

a) Descrierea modelului real
b) Diagrama E/R
c) Diagrama conceptuală
e) Schemele relaționale
f) Tabele

01-04-2013

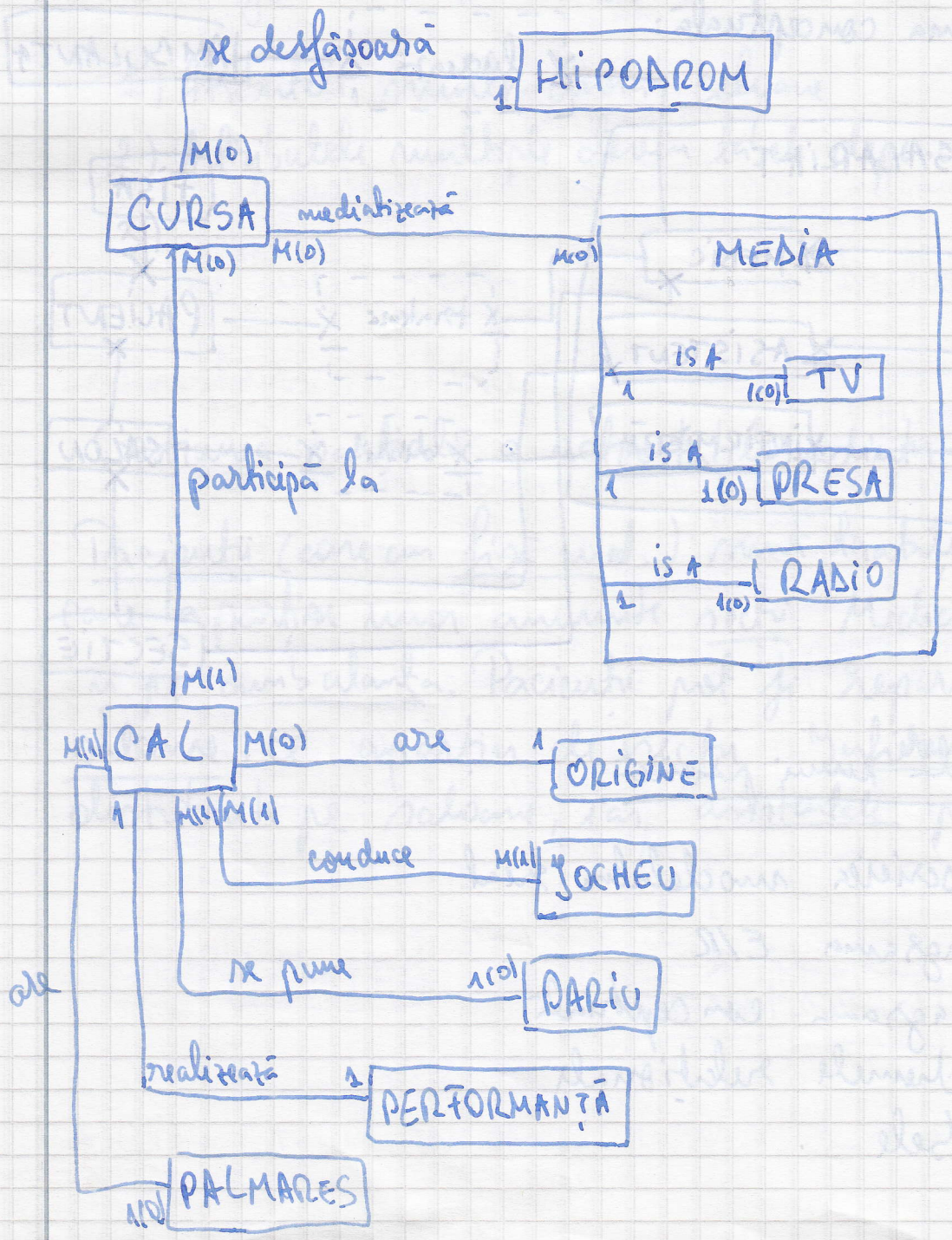
LABORATOR 7

- INLINE VIEW
- SUBCERERI
  - NECORELATE
  - CORELATE

02-04-2013

CURS 7

Jurnalul curselor de cai



Erori

1) a) PALMARES (M10) —— 1 — CAL

b) definirea conceptului de „palmares"

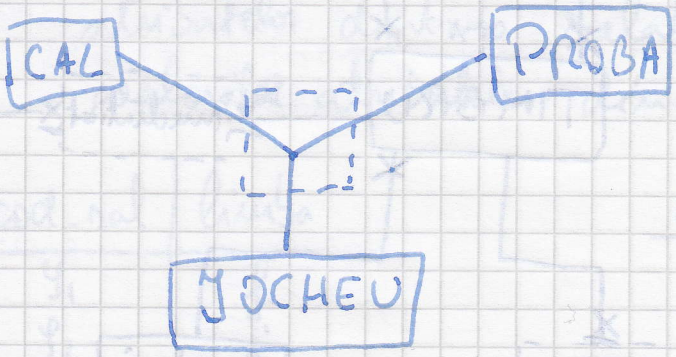
2) Ce entitate lipsește?

PROBA (compune o CURSĂ)
(CONCURS)

3) CURSA —— PROBA — CAL

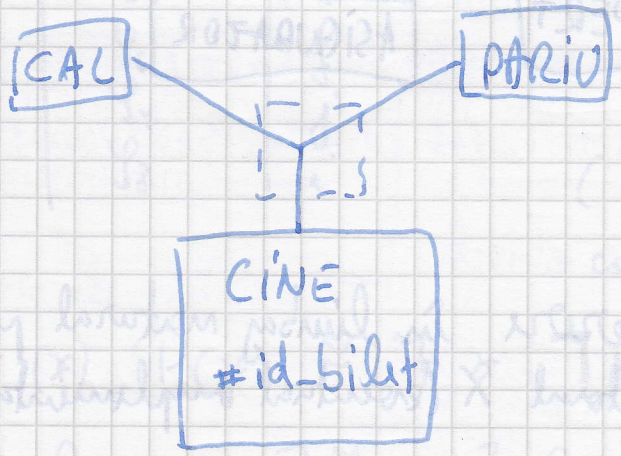
4) ORIGINE -> În tabelul CAL trebuie să avem id_mama, id_tata

5)



6) Performanța este atribut

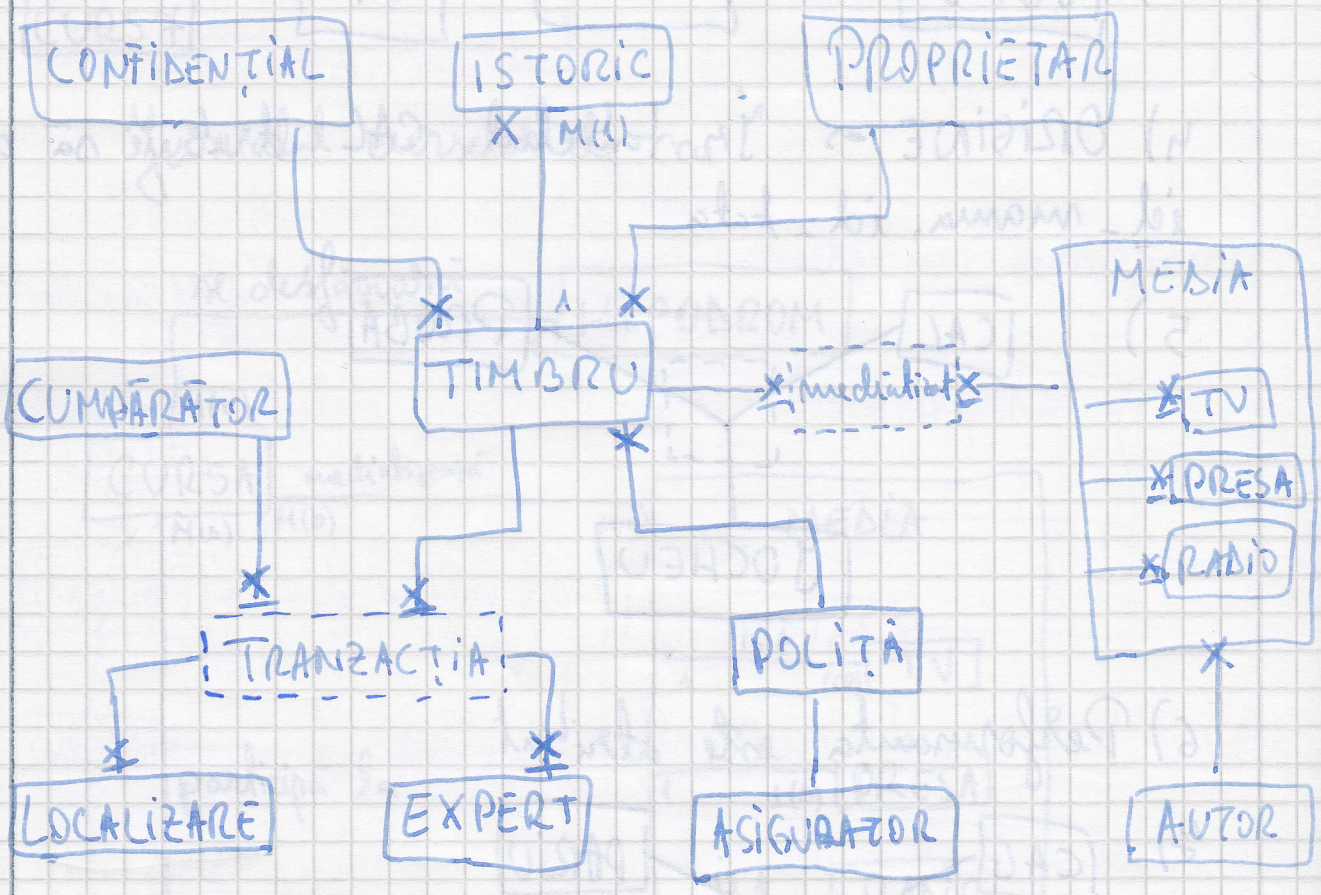
7)



Tranzacții filaterice

TRANZACȚIA (ce timbru, cine cumpără, valoare pornire, valoare finală, ce expert, data, localitate)

Timbrul are un istoric, are proprietar, poate fi asigurat, există informații confidențiale despre el, poate fi mediatizat



3) Algebra relațională

Subiect examen: Cerere în limbaj natural prin care implementați operatorul X (eventual implementarea în SQL)

1. Scopul fundamental al algebrei relaționale este construirea expresiilor relaționale, de fapt o reprezentare formală simbolică a intențiilor utilizatorului

2. Operatorii sunt numai pentru citire, nu actualizează operanzii

1) SELECT

2) PROJECT

3) UNION

4) REUNION

5) DIFFERENCE

$\Pi_{oraș}$ (FILIALĂ) − $\Pi_{oraș}$ (PROPRIETATI)

6) INTERSECT

7) PRODUCT

8) DIVISION (Definește o relație ce conține valorile atributelor dintr-o relație, care apar în toate valorile atributelor din cealaltă relație)

INFO	cod_sal	limba
	φ_1	f
	φ_2	j
	φ_3	e
	φ_4	f
	φ_5	e
	φ_6	f
	φ_7	j
	φ_8	i

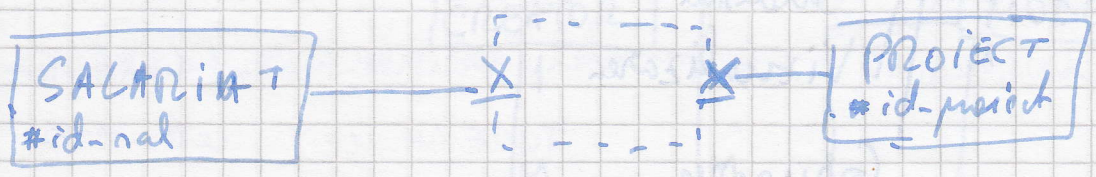
L	limba-s
	f
	j

INFO :- L	
	φ_1
	φ_4

(codurile salariaților care cunosc toate limbile)

$\neg \forall (x)\ P(x) \approx \exists x\ \neg P(x)$

$\forall (x)\ P(x) \approx \neg \exists x\ \neg P(x)$



08-04-2013

LABORATORUL 8

SELECT

FROM Tab T1, Tab T3

WHERE EXISTS (SELECT FROM TAB T2

WHERE T2.col = T1.col)

DIVISION - cuantificatorul universal

$$\forall x \; P(x) \equiv \neg \exists x \; \neg P(x)$$

2 mod implementare: 1) folosind de 2 ori NOT EXISTS

2) folosind count

09-04-2013

CURS 8

Pseudocoloane

SQL
- LDD : CREATE, ALTER, DROP
- LMD : SELECT, INSERT, UPDATE, DELETE
- LCD : COMMIT, ROLLBACK, SAVEPOINT

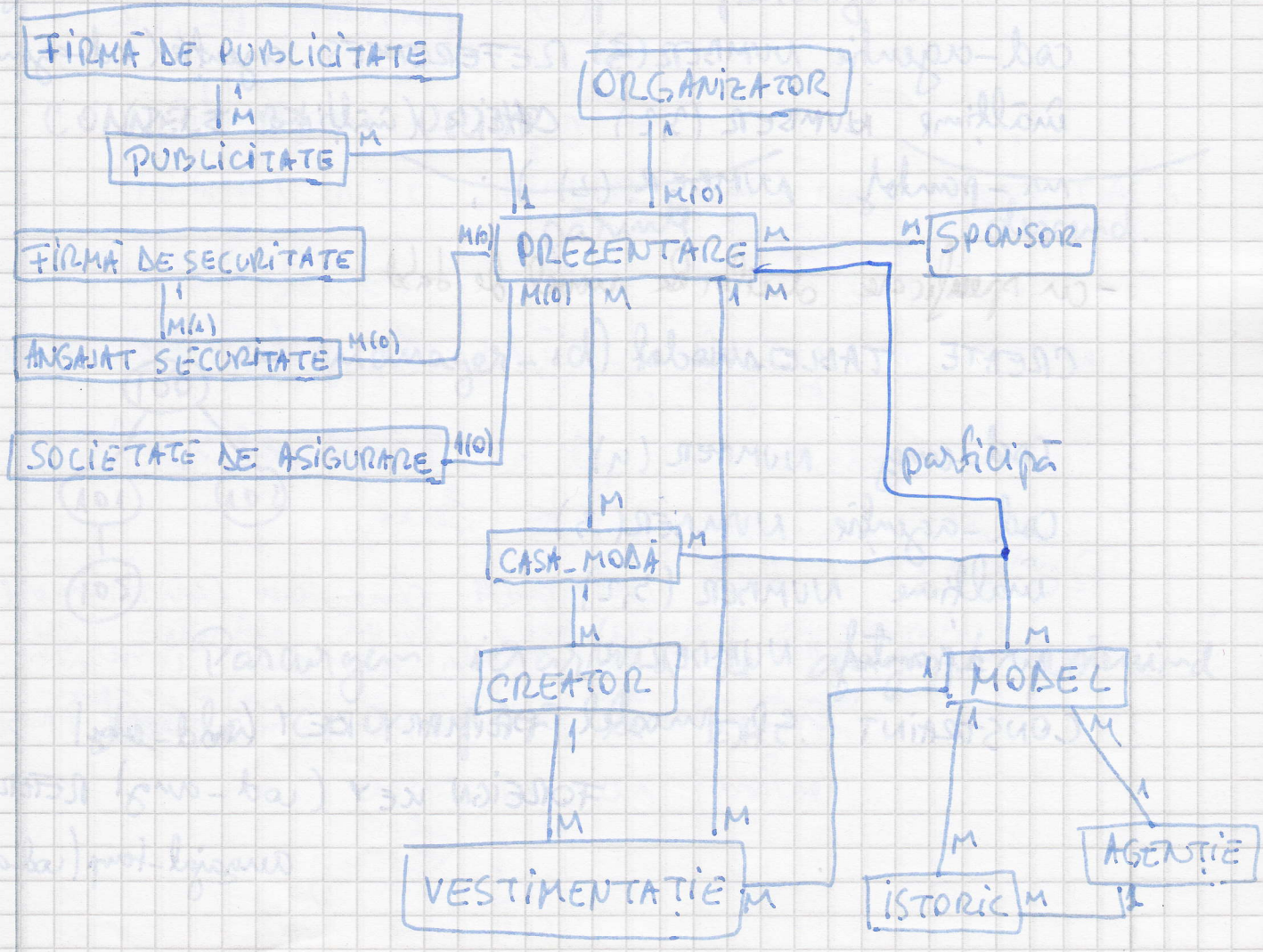
BD = o mulțime de obiecte

Obiecte
- Tabelul
- Secvența
- Indexul
- Vizualizarea
- Grupurile

Comenzi pe obiecte: Rename, Comment, truncate, flashback, purge, audit, analize

Să se creeze diagrama conceptuală privind prezentarea de modă. Va conține:

- aspecte privind organizatorii prezentărilor
- societățile de asigurare
- societățile de pază și protecție angajați
- firmele de publicitate
- sponsorii care finanțează prezentările
- casele de modă care participă
- creatorii de modă care lucrează
- istoricul modelelor legat de agențiile la care au lucrat
- modelele



Integritatea structurală

1. Unicitatea cheii : C.P. - unică și minimală
2. Atributele C.P. ≠ NULL
3. C.E : NULL toate sau să corespundă unei valori a C.P. asociate

Constrângeri

- → de domeniu : CHECK, UNIQUE, NOT NULL, DEFAULT (la nivel de coloană)
- → de integritate a entității : PRIMARY KEY
- → de integritate a referirii : FOREIGN KEY

- chei la nivel de coloană

```
CREATE TABLE model(
cod_ang NUMBER(4) PRIMARY KEY, REFERENCES angajat_temp
                                (cod_angajat),
cod_agenție NUMBER(3) REFERENCES agenție(cod_agenție),
înălțime NUMBER(3,2) CHECK (înălțime > 1.5),
nr_pantof NUMBER(2) );
```

- cu specificare cheilor la nivel de tabel

```
CREATE TABLE model(
cod_ang NUMBER(4)
cod_agenție NUMBER(3)
înălțime NUMBER(3,2)
nr_pantof NUMBER(2)
CONSTRAINT cp-model PRIMARY KEY (cod_ang)
                    FOREIGN KEY (cod_ang) REFERENCES
                                angajat_temp(cod_ang
```

FOREIGN KEY (cod_agentii) REFERENCES agenzie (cod_agentii)

CHECK (mălțime >1.5));

- prin copiere

CREATE TABLE model2
AS SELECT * FROM model;

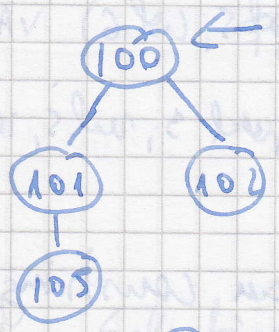
15-04-2013

LABORATOR 9

- cereri ierarhice
- LMD
- ex. recapitulative

START WITH -> cond. ce dă rădăc. nodurilor din care începem parcurgerea

CONNECT BY PRIOR =
părinte descend.



← manager_id is NULL

Parcurgem ierarhia de angajați pornind de la CEO până la exec.

```
SELECT emp_id, last_name, level
FROM employees
START WITH manag_id is NULL
CONNECT BY PRIOR emp_id = manag_id;
```

LMD
- INSERT
- UPDATE
- DELETE

```
CREATE TABLE
  EMP_JR
AS SELECT *
FROM EMPLOYEES;
```

→ copiază structura
păstrează const. NOT NULL

NU păstrează PK, FK

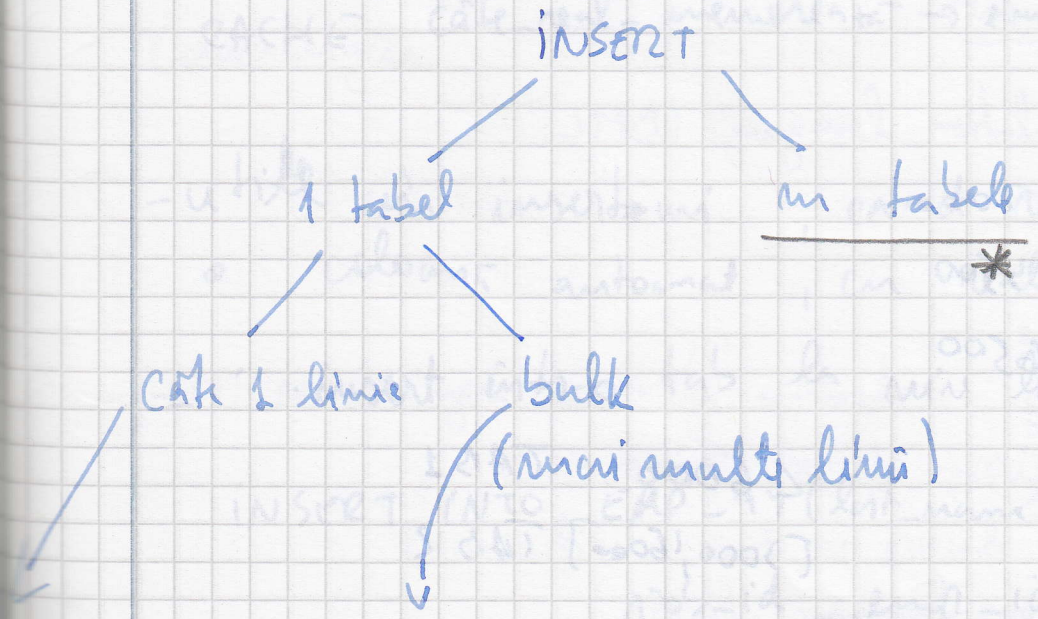
copiază conținutul

```
INSERT INTO EMP_JR (col1, col3, col5, col6) VALUES
                   (val1, val3, val5, val6)
```

REGULĂ! Toate coloanele care au constrângere NOT NULL în tabela în care inserăm trebuie să primească prin ~~[illegible]~~ INSERT valori NOT NULL

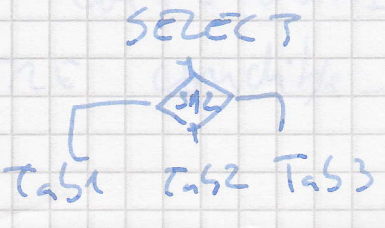
COMMIT - permanentizează efectul și îl face vizibil și pt. ceilalți useri

-> dacă lipsește enumerarea coloanelor -> la valori furnizăm valori pt. toate coloanele din tabelă, în ordinea din DESCRIBE




```
INSERT INTO TABELA (col1, col5, col3, col4)
    SELECT V1, V5, V3, V4
    FROM emp e, dep d
    WHERE e.dep_id = d.dep_id
      AND d.d_name = 'Shipping'
```

* Selectăm toate info despre angajați și în fct. de salariu să fie inserate astfel



INSERS { FIRST / ALL }

WHEN ...

WHEN ...

ELSE ...

FIRST -> la prima condizione true se operata

ALL -> inserisce tutte rispettive true

101	X	2000
102	Y	3500
103	Z	6500

< 6000 TAB 1

[3000, 6000] TAB 2

≥ 15000 TAB 3

FIRST :

X -> tab 1

Y -> tab 1

Z -> tab 2

ALL :

X -> tab 1

Y -> tab 1, tab 2

Z -> tab 2

Cuanul ou secvențe (SEQUENCE)

↳ pt a genera întregi unici dintr-un sistem multi-user

↳ sunt independente de tabele

CREATE SEQUENCE numesecv_NP

START WITH 10 -> val. de start

INCREMENT BY 1 -> pasul

CACHE câte_val_memorate -> implicit 20

- utile în inserțiuni, când vrem să populăm o coloană automat, cu valori unice

- insert într-o tab la mai linii

INSERT INTO EMP_YY (last_name, email, hire_date, job_id, emp_id)

VALUES ('abc', 'abc@yahoo.com', ..., 'IT_PROG', numesecv_NP.NEXTVAL);

UPDATE -> actualizăm/modificăm date

↳ cu atribuire separată pt. fiecare col. actualizată

UPDATE Tabela

SET Col1 = Val1, Col5 = Val5, Col3 = Val3

WHERE condiție_linii_actualizate;

↳ cu subcereri (SELECT)

Stergere → linii (date)

DELETE FROM Tabela
WHERE cond_linie_sterse

16-04-2013

CURS 9

SQL Statements

- Data manipulation language (DML)
  - SELECT
  - INSERT
  - UPDATE
  - DELETE
  - MERGE
- Data definition language (DDL)
  - CREATE
  - ALTER
  - DROP
  - RENAME
  - TRUNCATE
  - COMMENT
- Transaction control
  - COMMIT
  - ROLLBACK
  - SAVEPOINT

• Data control language (DCL)

GRANT

REVOKE

Constrângeri declarative:

- constrângeri de domeniu
- constrângerea de integritate a entității
- constrângerea de integritate referențială

Constrângeri curs

CURS

->O constrângere poate fi amânată - DEFERRABLE

↳ restricția va fi verificată doar la sfârșitul tranzacției, când se execută COMMIT

↳ implicit: NOT DEFERRABLE

Modificarea structurii unui tabel ->ALTER TABLE

↳ realizează modificarea structurii tabelului (la nivel de coloană sau la nivel de tabel), dar nu și modificarea conținutului tabelului

ALTER TABLE - ADD

MODIFY

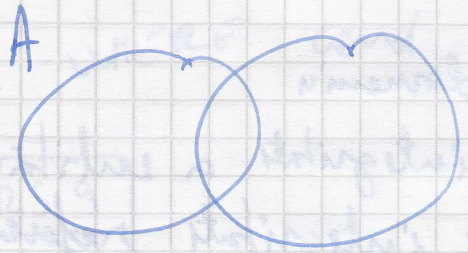
DROP

ENABLE/DISABLE

~~20~~ 5 IUNIE - EXAMEN

22-04-2013

LABORATOR 10



proiectele angajatului curent

proiectele lui 102

- în toate pr lui 102
- cel puțin în pr. lui 102 } NOT EXISTS (A \ B)
- cel mult în pr. lui 102 } NOT EXISTS (B \ A)
- exact în pr lui 102 { NOT EXISTS (A \ B) AND NOT EXISTS (B \ A)
- în niciun pr. al lui 102 NOT EXISTS (A ∩ B)

23-04-2013

CURS 10

```
CREATE OR REPLACE VIEW VIZ1
AS SELECT EMP_ID, LAST_NAME, FIRST_NAME, EMAIL
FROM EMPLOYEES;
+ WHERE LOWER(LAST_NAME) LIKE '%a'
```

WITH CHECK OPTION;

CREATE OR REPLACE VIEW2
AS SELECT EMP_ID, LAST_NAME, SALARY*12
FROM EMPLOYEES;

29-04-2013

LABORATOR 11

VIEWS

13-05-2013

LABORATOR 12

LDS = limbaj de def. al datelor

- create ob
- modif struct ob.

crearea - secvente

CREATE SEQUENCE s1
START WITH 1
INCREMENT BY 10;

INSERT INTO DEPARTMENTS
VALUES (s1.NEXTVAL, 'APROV', 101, 1700)

Create - view

```
CREATE OR REPLACE VIEW V1
    AS SELECT *
        FROM EMPLOYEES;
```

- Create - tabelelor

Met 1) crearea prin copierea structurii a datelor dintr-o tab existente
- se păstr. constr. NOT NULL; se pierd FK, PK

~~Met 2)~~

Tipuri de date

VARCHAR2(n) → șiruri de lung. variab., maxim lung n caract.

CHAR(n) → șiruri de lung. fixă egală cu n

NUMBER(p,s) → nr. cu p cifre din care s repr. partea zecimală

DATE → dată calend

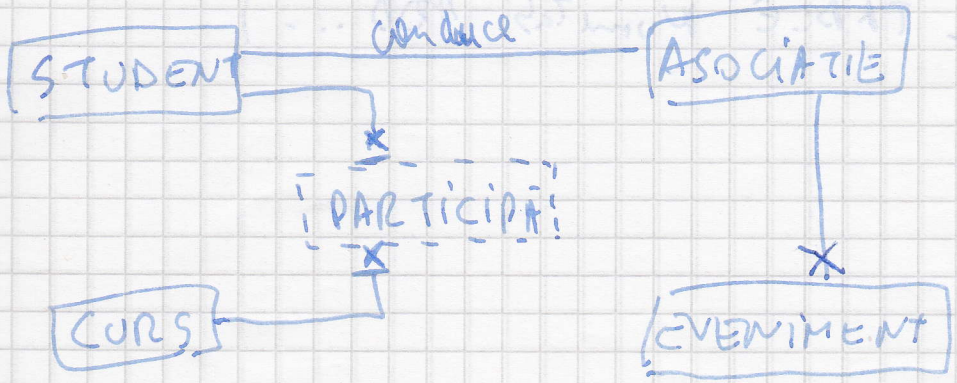
Met 2) Crearea tabelă cu specif. datelor la nivel de coloană

CREATE TABLE TEST1_JR

```
(   idt      NUMBER PRIMARY KEY
    nm       VARCHAR(10),
    prn      CHAR(10),
    data     DATE
)
```

Constrângeri la nivel de coloană -> implică doar col. resp.

- cheie primară -> PRIMARY KEY
- cheie externă -> REFERENCES ...
- NOT NULL -> împiedică val NULL pt. coloana resp.
- UNIQUE -> val de pe col. resp. trebuie să fie unice (∄ duplicate)
- CHECK (colresp > val)
  <
  =
  <>



Met 3) Crearea tabelei constă la nivel de tabelă

Met 4) Crearea tabelei fără const.
adăugăm constrângerile

• Ștergere tabelă

-> ștergem obiectul în total cu structură

DROP TABLE Numetab

-> ștergem doar datele, menținem structura

CMD + CCD { DELETE FROM Numetab;
COMMIT;

TRUNCATE TABLE Numetab;
CDD:

• Modificarea structurii tabelei create

ALTER TABLE Numetab ADD ...)

LMD	LDD
! op. asupra datelor - insert - update - delete	! op asupra struct. ob create alter

Ştergere col:

ALTER TABLE NumeTab DROP COLUMN NumeCol

Ştergere constr.: ——— " ——— " ——— CONSTRAINT

Modif. unei col:

ALTER TABLE NUMETAB MODIFY
(Sediu varchar2(50))