

Baze de Date

Solutii – Laborator 4 – Saptamana 7

LABORATOR 4 - SAPTAMANA 7

--Daca nu ati fost prezenti in laboratorul anterior se ruleaza urmatoarele tranzactii

-- ATENTIE LA DENUMIREA FOLOSITA PENTRU CELE DOUA TABELE EMP SI DEPT

-- trebuie sa utilizati sirul (EMP_PNU/DEPT_PNU), unde pnu inseamna:

-- p - prima litera din prenume, iar nu - primele doua litere din nume

-- apoi se ruleaza cele doua comenzi

```
CREATE TABLE EMP_pnu AS SELECT * FROM employees;
```

```
CREATE TABLE DEPT_pnu AS SELECT * FROM departments;
```

-- IN CONTINUARE SE ADAUGA CONSTRANGERILE DE INTEGRITATE

```
ALTER TABLE emp_pnu
```

```
ADD CONSTRAINT pk_emp_pnu PRIMARY KEY(employee_id);
```

```
ALTER TABLE dept_pnu
```

```
ADD CONSTRAINT pk_dept_pnu PRIMARY KEY(department_id);
```

```
ALTER TABLE emp_pnu
```

```
ADD CONSTRAINT fk_emp_dept_pnu FOREIGN KEY(department_id)
```

```
REFERENCES dept_pnu(department_id);
```

```
ALTER TABLE emp_pnu
```

```
ADD CONSTRAINT fk_emp_emp_pnu FOREIGN KEY(manager_id)
```

```
REFERENCES emp_pnu(employee_id); -- managerul unui angajat
```

```
ALTER TABLE dept_pnu
```

```
ADD CONSTRAINT fk_dept_emp_pnu FOREIGN KEY(manager_id)
```

```
REFERENCES emp_pnu(employee_id); -- managerul de departament
```

-- APOI SE REZOLVA, IN CADRUL LABORATORULUI CURENT,
URMATOARELE EXERCITII

5. Să se insereze departamentul 300, cu numele Programare în DEPT_pnu.

Analizați cazurile, precizând care este soluția corectă și explicând erorile celorlalte variante.

Pentru a anula efectul instrucțiunii(ilor) corecte, utilizați comanda ROLLBACK.

```
DESC DEPT_PNU;
```

```
SELECT * FROM dept_pnu;
```

```
SELECT * FROM emp_pnu;
```

--discutie tipuri de INSERT si erori posibile

--vezi laborator

```
DESC DEPT_PNU;
```

--a) inserare implicita

```
INSERT INTO DEPT_pnu  
VALUES (300, 'Programare');
```

--b) inserare explicita

```
INSERT INTO DEPT_pnu (department_id, department_name)  
VALUES (300, 'Programare');
```

```
SELECT * FROM dept_pnu;
```

--c)

```
INSERT INTO DEPT_pnu (department_name, department_id)
```

```
VALUES (300, 'Programare');
```

--d)

```
INSERT INTO DEPT_pnu (department_id, department_name, location_id)
```

```
VALUES (300, 'Programare', null);    -- se incalca constrangerea de unicitate a  
cheii primare
```

-- varianta corecta

```
INSERT INTO DEPT_pnu (department_id, department_name, location_id)
```

```
VALUES (301, 'Programare', null);
```

```
SELECT * FROM dept_pnu;
```

--e)

```
INSERT INTO DEPT_pnu (department_name, location_id)
```

```
VALUES ('Programare', null);
```

create table

alter table

drop

commit implicit

Baze de Date

<https://www.cezarabenegui.com/>

cezara.benegui@fmi.unibuc.ro

-- Ce se intampla daca executam rollback?

```
SELECT * FROM dept_pnu;
```

```
rollback;
```

-- Executati varianta corecta si permanentizati modificarile.

```
INSERT INTO DEPT_pnu (department_id, department_name)
VALUES (300, 'Programare');
```

```
commit;
```

```
desc dept_pnu;
```

6. Să se insereze un angajat corespunzător departamentului introdus anterior în tabelul EMP_pnu, precizând valoarea NULL pentru coloanele a căror valoare nu este cunoscută la inserare (metoda implicită de inserare).

Determinați ca efectele instrucțiunii să devină permanente.

Atenție la constrângerile NOT NULL asupra coloanelor tabelului!

-- inserare prin metoda IMPLICITA de inserare

-- dorim sa inseram un angajat in depart 300

DESC emp_pnu;

SELECT * FROM emp_pnu;

INSERT INTO emp_pnu

VALUES (250, NULL, 'nume250', 'email250', NULL, SYSDATE, 'IT_PROG',
NULL, NULL, NULL, 300);

-- Cum permanentizam efectul actiunii anterioare?

commit;

SELECT * FROM emp_pnu;

-- De ce varianta urmatoare nu functioneaza?

INSERT INTO emp_pnu

VALUES (250, NULL, 'nume251', 'email251', NULL, SYSDATE, 'IT_PROG',
NULL, NULL, NULL, 300);

-- Anulati inserarea anterioara

rollback;

SELECT * FROM emp_pnu;

-- De ce varianta urmatoare nu functioneaza?

INSERT INTO emp_pnu

VALUES (251, NULL, 'nume251', 'email251', NULL, to_date('03-10-2023',
'dd-mm-yyyy'),

'IT_PROG', NULL, NULL, NULL, 300);

SELECT * FROM emp_pnu;

-- De ce varianta urmatoare nu functioneaza?

INSERT INTO emp_pnu

VALUES (252, NULL, 'nume252', 'email252', NULL, SYSDATE,

'IT_PROG', NULL, NULL, NULL, 310); -- nu exista valoarea cheii externe
in tabelul parinte

INSERT INTO emp_pnu

VALUES (252, NULL, 'nume252', 'email252', NULL, SYSDATE,

'IT_PROG', NULL, NULL, NULL, 300);

-- IN CELE DIN URMA PASTRAM IN BAZA DE DATE ANGAJATUL CU ID-UL 250 IN DEPART. 300

rollback;

7. Să se mai introducă un angajat corespunzător departamentului 300, precizând după numele tabelului lista coloanelor în care se introduc valori (metoda explicita de inserare).

Se presupune că data angajării acestuia este cea curentă (SYSDATE).

Salvați înregistrarea.

desc emp_pnu;

--inserare prin metoda EXPLICITA de inserare

INSERT INTO emp_pnu (hire_date, job_id, employee_id, last_name, email, department_id)

VALUES (sysdate, 'sa_man', 278, 'nume_278', 'email_278', 300);

COMMIT;

SELECT * FROM emp_pnu;

8. Creați un nou tabel, numit EMP1_PNU, care va avea aceeași structură ca și EMPLOYEES,

dar fara inregistrari. Copiați în tabelul EMP1_PNU salariații (din tabelul EMPLOYEES)

al căror comision depășește 25% din salariu (se accepta omiterea constrangerilor).

-- crearea tabelului

```
CREATE TABLE emp1_pnu AS SELECT * FROM employees;
```

-- eliminarea inregistrarilor

```
DELETE FROM emp1_pnu;
```

-- adaugarea noilor valori (inserarea randurilor)

```
INSERT INTO emp1_pnu
```

```
SELECT *
```

```
FROM employees
```

```
WHERE commission_pct > 0.25;
```

```
SELECT * FROM emp1_pnu;
```

-- Ce se intampla daca executam un rollback?

rollback; -- anuleaza cele doua comenzi - insert si delete

-- SA SE ANALIZEZE EXERCITIILE 9, 10 SI 11

9. Să se creeze un fișier (script file) care să permită introducerea de înregistrări

în tabelul EMP_PNU în mod interactiv.

Se vor cere utilizatorului: codul, numele, prenumele si salariul angajatului.

Câmpul email se va completa automat prin concatenarea primei litere din prenume

și a primelor 7 litere din nume.

Executati script-ul pentru a introduce 2 inregistrari in tabel.

```
INSERT INTO emp_pnu (employee_id, first_name, last_name, email,
hire_date, job_id, salary)
```

```
VALUES(&cod, '&&prenume', '&&nume', substr('&prenume',1,1) ||
substr('&nume',1,7),
```

```
sysdate, 'it_prog', &sal);
```

```
UNDEFINE prenume;
```

```
UNDEFINE nume;
```

```
SELECT * FROM emp_pnu;
```

```
rollback;
```

10. Creați 2 tabele emp2_pnu și emp3_pnu cu aceeași structură ca tabelul EMPLOYEES,

dar fără înregistrări (acceptăm omiterea constrângerilor de integritate).

Prin intermediul unei singure comenzi, copiați din tabelul EMPLOYEES:

- în tabelul EMP1_PNU salariații care au salariul mai mic decât 5000;
- în tabelul EMP2_PNU salariații care au salariul cuprins între 5000 și 10000;
- în tabelul EMP3_PNU salariații care au salariul mai mare decât 10000.

Verificați rezultatele, apoi ștergeți toate înregistrările din aceste tabele.

--VEZI INSERARI MULTI-TABEL IN LABORATORUL 4

```
CREATE TABLE emp1_pnu AS SELECT * FROM employees;
```

```
DELETE FROM emp1_pnu;
```

```
SELECT * FROM emp1_pnu;
```

```
CREATE TABLE emp2_pnu AS SELECT * FROM employees;
```

```
DELETE FROM emp2_pnu;
```

```
CREATE TABLE emp3_pnu AS SELECT * FROM employees;
```

```
DELETE FROM emp3_pnu;
```

```
INSERT ALL
```

```
  WHEN salary < 5000 THEN
```

```
    INTO emp1_pnu
```

```
  WHEN salary >= 5000 AND salary <= 10000 THEN
```

```
    INTO emp2_pnu
```

```
ELSE
```

```
  INTO emp3_pnu
```

```
SELECT * FROM employees;
```

```
SELECT * FROM emp1_pnu;
```

```
SELECT * FROM emp2_pnu;
```

```
SELECT * FROM emp3_pnu;
```

11. Să se creeze tabelul EMP0_PNU cu aceeași structură ca tabelul EMPLOYEES

(fără constrângeri), dar fără înregistrări.

Copiați din tabelul EMPLOYEES:

- în tabelul EMP0_PNU salariații care lucrează în departamentul 80;
- în tabelul EMP1_PNU salariații care au salariul mai mic decât 5000;
- în tabelul EMP2_PNU salariații care au salariul cuprins între 5000 și 10000;
- în tabelul EMP3_PNU salariații care au salariul mai mare decât 10000.

Dacă un salariat se încadrează în tabelul emp0_pnu atunci acesta nu va mai fi inserat

și în alt tabel (tabelul corespunzător salariului său);

```
CREATE TABLE emp0_pnu AS SELECT * FROM employees;
```

```
DELETE FROM emp0_pnu;
```

```
INSERT FIRST
```

```
    WHEN department_id = 80 THEN
```

```
        INTO emp0_pnu
```

```
    WHEN salary < 5000 THEN
```

```
        INTO emp1_pnu
```

```
    WHEN salary >= 5000 AND salary <= 10000 THEN
```

```
        INTO emp2_pnu
```

```
    ELSE
```

```
        INTO emp3_pnu
```

```
SELECT * FROM employees;
```

```
SELECT * FROM emp0_pnu;
```

```
SELECT * FROM emp1_pnu;
```

```
SELECT * FROM emp2_pnu;
```

```
SELECT * FROM emp3_pnu;
```

-- COMANDA UPDATE - VEZI LABORATOR (pentru notiunile teoretice)

12. Măriți salariul tuturor angajaților din tabelul EMP_PNU cu 5%.

Vizualizați, iar apoi anulați modificările.

```
UPDATE emp_pnu
```

```
SET salary = salary * 1.05;
```

```
SELECT * FROM emp_pnu;
```

```
ROLLBACK;
```

13. Schimbați jobul tuturor salariaților din departamentul 80 care au comision, în 'SA_REP'.

Anulați modificările.

```
UPDATE emp_pnu
```

```
SET job_id = 'SA_REP'
```

```
WHERE department_id = 80 and commission_pct IS NOT NULL;
```

```
SELECT * FROM emp_pnu;
```

```
ROLLBACK;
```

14. Să se promoveze Douglas Grant la manager în departamentul 20 (tabelul dept_pnu),

având o creștere de salariu cu 1000\$.

-- verificari

SELECT *

FROM emp_pnu

WHERE lower(last_name||first_name) = 'grantdouglas'; -- 199

SELECT * FROM dept_pnu

WHERE department_id = 20;

-- solutia problemei

UPDATE dept_pnu

SET manager_id = (SELECT employee_id

FROM emp_pnu

WHERE lower(last_name||first_name) = 'grantdouglas'

)

WHERE department_id = 20;

```
update emp_pnu  
set salary = salary + 1000  
WHERE lower(last_name||first_name) = 'grantdouglas';
```

```
rollback;
```

-- COMANDA DELETE - VEZI LABORATOR (pentru notiunile teoretice)

15. Ștergeți toate înregistrările din tabelul DEPT_PNU.

Ce înregistrări se pot șterge? Anulați modificările.

```
DELETE FROM dept_pnu;
```

```
SELECT * FROM dept_pnu;
```

```
SELECT * FROM emp_pnu;
```


16. Suprimați departamentele care nu au angajati. Anulați modificările.

-- prima data afisam departamentele care nu au angajati

-- SOLUTIA 1 – MINUS

```
SELECT department_id
FROM departments
```

MINUS

```
SELECT department_id
FROM employees;
```

-- SOLUTIA 2 - JOIN

-- trebuie sa afisam departamentele care NU AU angajati

-- DECI afisam TOATE departamentele chiar daca au sau nu angajati

-- si din acestea le vom pastra doar pe cele fara angajati

```
SELECT d.department_id
FROM employees e RIGHT JOIN departments d ON(e.department_id =
d.department_id)
WHERE employee_id is null;
```

-- apoi stergem departamentele care nu au angajati;

```
delete from dept_pnu
where department_id IN
  (SELECT department_id
   FROM departments
```

MINUS

```
  SELECT department_id
   FROM employees
);
```

```
SELECT * FROM dept_pnu;
rollback;
```

17. Să se mai introducă o linie in tabelul DEPT_PNU.

```
desc dept_pnu;
```

```
INSERT INTO dept_pnu  
VALUES(320, 'dept_nou', NULL, NULL);
```

```
SELECT * FROM dept_pnu;
```

18. Să se marcheze un punct intermediar in procesarea tranzacției (SAVEPOINT p).

```
SAVEPOINT p;
```

19. Să se șteargă din tabelul DEPT_PNU departamentele care au codul de departament

cuprins între 160 și 200. Listați conținutul tabelului.

```
DELETE FROM dept_pnu  
WHERE department_id BETWEEN 160 AND 200;
```

```
SELECT * FROM dept_pnu;
```

20. Să se renunțe la cea mai recentă operație de ștergere, fără a renunța la operația precedentă de introducere.

Determinați ca modificările să devină permanente;

```
SELECT * FROM dept_pnu;
```

```
ROLLBACK TO p;
```

```
COMMIT;
```

LABORATOR 5 - SAPTAMANA 7

-- Limbajul de definire a datelor (LDD)

--COMENZI CARE FAC PARTE DIN LDD:

CREATE, ALTER, DROP, TRUNCATE, RENAME

--Ce comanda LCD se executa dupa instructiunile de tip LDD?

commit;

1. Crearea tabelelor (vezi notiunile in laborator 5)

-- EXERCITII

1. Să se creeze tabelul ANGAJATI_pnu

(pnu se alcatuiește din prima literă din prenume și primele două din numele studentului)

corespunzător schemei relaționale:

```
ANGAJATI_pnu(cod_ang number(4), nume varchar2(20), prenume
varchar2(20), email char(15),
            data_ang date, job varchar2(10), cod_sef number(4), salariu number(8,
2),
            cod_dep number(2)
);
```

a) cu precizarea cheilor primare la nivel de coloană

si a constrangerilor NOT NULL pentru coloanele nume și salariu;

```
CREATE TABLE angajati_pnu
```

```
( cod_ang number(4) constraint pk_angajat primary key,
  nume varchar2(20) constraint nume_ang not null,
  prenume varchar2(20),
  email char(15) unique,
```

```
data_ang date default sysdate,  
job varchar2(10),  
cod_sef number(4),  
salariu number(8,2) not null,  
cod_dep number(2)  
);
```

--commit implicit

```
SELECT * FROM angajati_pnu;  
DESC angajati_pnu;
```

b) cu precizarea cheii primare la nivel de tabel

si a constrângerilor NOT NULL pentru coloanele nume și salariu.

```
DROP TABLE angajati_pnu;
```

rollback; -- nu are niciun efect

```
CREATE TABLE angajati_pnu
```

```
( cod_ang number(4),  
  nume varchar2(20) constraint nume_ang not null,  
  prenume varchar2(20),
```

```
email char(15)unique,  
data_ang date default sysdate,  
job varchar2(10),  
cod_sef number(4),  
salariu number(8, 2) constraint salariu_ang not null,  
cod_dep number(2),  
constraint pkey_ang primary key(cod_ang) --constrangere la nivel de tabel  
);
```