Baze de date - Anul 1

Laborator 5

Limbajul de definire a datelor (LDD) (partea I)

I. [Obiective]

- Operații de definire (creare, modificare, suprimare) a tabelelor bazei de date
- Definirea secvențelor

II. [LDD]

- În general, instrucţiunile *LDD* sunt utilizate pentru definirea structurii corespunzătoare obiectelor unei scheme: tabele, vizualizări, vizualizări materializate, indecşi, sinonime, clustere, proceduri şi funcţii stocate, declanşatori, pachete stocate etc.
- Aceste instrucţiuni permit:
  - crearea, modificarea şi suprimarea obiectelor unei scheme şi a altor obiecte ale bazei de date, inclusiv baza însăşi şi utilizatorii acesteia (*CREATE*, *ALTER*, *DROP*);
  - modificarea numelor obiectelor unei scheme (*RENAME*);
  - ştergerea datelor din obiectele unei scheme, fără suprimarea structurii obiectelor respective (*TRUNCATE*).
- Implicit, o instrucţiune *LDD* permanentizează (*COMMIT*) efectul tuturor instrucţiunilor precedente şi marchează începutul unei noi tranzacţii.
- Instrucţiunile *LDD* au efect imediat asupra bazei de date şi înregistrează informaţia în dicţionarul datelor.
- Definirea unui obiect presupune: crearea (*CREATE*), modificarea (*ALTER*) şi suprimarea sa (*DROP*).

Reguli de numire a obiectelor bazei de date

- Identificatorii obiectelor trebuie să înceapă cu o literă şi să aibă maximum 30 de caractere, cu excepţia numelui bazei de date care este limitat la 8 caractere şi celui al legăturii unei baze de date, a cărui lungime poate atinge 128 de caractere.
- Numele poate conţine caracterele *A-Z*, *a-z*, 0-9, _, *$* şi #.
- Două obiecte ale aceluiaşi utilizator al *server*-ului *Oracle* nu pot avea acelaşi nume.
- Identificatorii nu pot fi cuvinte rezervate ale *server*-ului *Oracle*.

- Identificatorii obiectelor nu sunt *case-sensitive*.

III. [Definirea tabelelor]

1. Crearea tabelelor

- Formele simplificate ale comenzii de creare a tabelelor sunt:

CREATE TABLE *nume_tabel* **(**
coloana_1 tip_date ***[DEFAULT*** *valoare**]***
*[constrangere_nivel_coloana [constrangere_nivel_coloana]...]**,***
. . . . . .
coloana_n tip_date ***[DEFAULT*** *valoare**]***
*[constrangere_nivel_coloana [constrangere_nivel_coloana]...]**,***
[*constrangeri_nivel_tabel**]***
);
sau
CREATE TABLE *nume_tabel [(coloana_1,..., coloana_n)]*
AS *subcerere;*

- Constrângerile **definite aupra unui tabel pot fi de următoarele tipuri:**
  - *NOT NULL* – coloana nu poate conţine valoarea *Null*; (*NOT NULL*)
  - *UNIQUE* – pentru coloane sau combinaţii de coloane care trebuie să aibă valori unice în cadrul tabelului; (*UNIQUE* (*col1, col2, …*))
  - *PRIMARY KEY* – identifică în mod unic orice înregistrare din tabel. Implică *NOT NULL* + *UNIQUE*; (*PRIMARY KEY* (*col1, col2, …*))
  - *FOREIGN KEY* – stabileşte o relaţie de cheie externă între o coloană a tabelului şi o coloană dintr-un tabel specificat.
    [*FOREIGN KEY nume_col*]
    REFERENCES nume_tabel(*nume_coloana*)
    [*ON DELETE* {*CASCADE*| *SET NULL*}]
    - *FOREIGN KEY* este utilizat într-o constrângere la nivel de tabel pentru a defini coloana din tabelul „copil“;
    - *REFERENCES* identifică tabelul „părinte“ şi coloana corespunzătoare din acest tabel;
    - *ON DELETE CASCADE* determină ca, odată cu ştergerea unei linii din tabelul „părinte“, să fie şterse şi liniile dependente din tabelul „copil“;

- *ON DELETE SET NULL* determină modificarea automată a valorilor cheii externe la valoarea *null*, atunci când se şterge valoarea „părinte“.

- *CHECK* – specifică o condiţie care trebuie să fie adevărată la nivel de coloană sau linie (*CHECK* (*conditie*)).

Observații:

- Constrângerile pot fi create odată cu tabelul sau adăugate ulterior cu o comandă *ALTER TABLE*.
- Constrângerile se pot implementa la nivel de coloană doar dacă nu referă o altă coloană a tabelului.
- În cazul în care o constrângere referă mai multe coloane, ea poate fi definită doar la nivel de tabel. De exemplu, dacă cheia primară (sau o cheie unică) este compusă, ea nu poate fi definită la nivel de coloane, ci doar la nivelul întregului tabel.
- Constrângerea de tip *NOT NULL* se poate declara doar la nivel de coloană.

- Principalele **tipuri de date** pentru coloanele tabelelor sunt următoarele:

Tip de date	**Descriere**
VARCHAR2(n) [BYTE | CHAR]	Defineşte un şir de caractere de dimensiune variabilă, având lungimea maximă de *n* octeţi sau caractere. Valoarea maximă a lui *n* corespunde la 4000 octeţi, iar cea minimă este de un octet sau un caracter.
CHAR(n) [BYTE | CHAR]	Reprezintă un şir de caractere de lungime fixă având *n* octeţi sau caractere. Valoarea maximă a lui *n* corespunde la 2000 octeţi. Valoarea implicită şi minimă este de un octet.
NUMBER(p, s)	Reprezintă un număr având *p* cifre, dintre care *s* cifre formează partea zecimală
LONG	Conţine şiruri de caractere având lungime variabilă, care nu pot ocupa mai mult de 2GB.
DATE	Reprezintă date calendaristice valide, între 1 ianuarie 4712 i.Hr. şi 31 decembrie 9999 d.Hr.

2. Modificarea (structurii) tabelelor

- **Modificarea structurii unui tabel** se face cu ajutorul comenzii ***ALTER TABLE***. Forma comenzii depinde de tipul modificării aduse:
  - Adăugarea unei noi coloane (nu se poate specifica poziţia unei coloane noi în structura tabelului; o coloană nouă devine automat ultima în cadrul structurii tabelului)

ALTER TABLE *nume_tabel*
ADD (*coloana tip_de_date* ***[DEFAULT*** *expr][, ...]****);***

- Modificarea unei coloane (schimbarea tipului de date, a dimensiunii sau a valorii implicite a acesteia; schimbarea valorii implicite afectează numai inserările care succed modificării)

ALTER TABLE *nume_tabel*
MODIFY (*coloana tip_de_date* [***DEFAULT*** *expr*][*, ...*])***;***

- Eliminarea unei coloane din structura tabelului:

ALTER TABLE *nume_tabel*
DROP COLUMN *coloana****;***

Observații:

- Dimensiunea unei coloane numerice sau de tip caracter poate fi mărită, dar nu poate fi micşorată decât dacă acea coloană conţine numai valori *null* sau dacă tabelul nu conţine nici o linie.
- Tipul de date al unei coloane poate fi modificat doar dacă valorile coloanei respective sunt *null*.
- O coloană *CHAR* poate fi convertită la tipul de date *VARCHAR2* sau invers, numai dacă valorile coloanei sunt *null* sau dacă nu se micșorează dimensiunea coloanei.

- Comanda *ALTER* permite **adăugarea unei constrângeri într-un tabel existent, eliminarea, activarea sau dezactivarea constrângerilor**.
  - Pentru adăugare de constrângeri, comanda are forma:

ALTER TABLE *nume_tabel*
ADD [***CONSTRAINT*** *nume_constr*] *tip_constr* (*coloana*);

- Pentru eliminare de constrângeri:

ALTER TABLE *nume_tabel*
DROP CONSTRAINT *nume_constr;*

- Pentru activare/dezactivare constrângere:

ALTER TABLE *nume_tabel*
MODIFY CONSTRAINT *nume_constr* ***ENABLE|DISABLE****;*
sau
ALTER TABLE *nume_tabel*
ENABLE| DISABLE CONSTRAINT *nume_constr;*

3. Suprimarea tabelelor

- Ştergerea fizică a unui tabel, inclusiv a înregistrărilor acestuia, se realizează prin comanda:

DROP TABLE *nume_tabel;*

- Pentru ştergerea conţinutului unui tabel şi păstrarea structurii acestuia se poate utiliza comanda:

TRUNCATE TABLE *nume_tabel;*

Observaţie: Fiind operaţie *LDD*, comanda *TRUNCATE* are efect definitiv (spre deosebire de *DELETE* care, fiind o comandă *LMD*, poate fi anulată).

4. Redenumirea tabelelor

- Comanda ***RENAME*** permite redenumirea unui tabel, vizualizare sau secvenţă.

RENAME *nume1_obiect* ***TO*** *nume2_obiect;*

Observaţii:

- În urma redenumirii sunt transferate automat constrângerile de integritate, indecşii şi privilegiile asupra vechilor obiecte.
- Sunt invalidate toate obiectele ce depind de obiectul redenumit, cum ar fi vizualizări, sinonime sau proceduri şi funcţii stocate.

5. Consultarea dicţionarului datelor

- Informaţii despre tabelele create se găsesc în vizualizările din dicționarul datelor:
  - *USER_TABLES* – informaţii complete despre tabelele utilizatorului.
  - *TAB* – informaţii de bază despre tabelele existente în schema utilizatorului.
- Informaţii despre constrângeri găsim în *USER_CONSTRAINTS*, iar despre coloanele implicate în constrângeri în *USER_CONS_COLUMNS*.

IV. [Exerciții – definire tabele]

1. Să se creeze tabelul *ANGAJATI_pnu* (*pnu* se alcatuieşte din prima literă din prenume şi primele două din numele studentului) corespunzător schemei relaţionale:

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)*)

în următoarele moduri:

a) fără precizarea vreunei chei sau constrângeri;
b) cu precizarea cheilor primare la nivel de coloană și a constrângerilor *NOT NULL* pentru coloanele nume şi salariu;
c) cu precizarea cheii primare la nivel de tabel și a constrângerilor *NOT NULL* pentru coloanele *nume* şi *salariu*.

Se presupune că valoarea implicită a coloanei *data_ang* este *SYSDATE*.

Observație*:* Nu pot exista două tabele cu acelaşi nume în cadrul unei scheme, deci recrearea unui tabel va fi precedată de suprimarea sa prin comanda:
DROP TABLE ANGAJATI_pnu;

2. Adăugaţi următoarele înregistrări în tabelul *ANGAJATI_pnu*:

Cod_ang	**Nume**	**Prenume**	**Email**	**Data_ ang**	**Job**	**Cod_ sef**	**Salariu**	**Cod_ dep**
100	Nume1	Prenume1	Null	Null	Director	null	20000	10
101	Nume2	Prenume2	Nume2	02-02-2014	Inginer	100	10000	10
102	Nume3	Prenume3	Nume3	05-06-2010	Programator	101	5000	20
103	Nume4	Prenume4	Null	Null	Inginer	100	9000	20
104	Nume5	Prenume5	Nume5	Null	Programator	101	3000	30

Prima si a patra înregistrare vor fi introduse specificând coloanele pentru care introduceţi date efectiv, iar celelalte vor fi inserate fără precizarea coloanelor în comanda *INSERT*.
Salvaţi comenzile de inserare.

3. Creaţi tabelul *ANGAJATI10_pnu*, prin copierea angajaţilor din departamentul 10 din tabelul *ANGAJATI_pnu*. Listaţi structura noului tabel. Ce se observă?

4. Introduceti coloana *comision* in tabelul *ANGAJATI_pnu*. Coloana va avea tipul de date *NUMBER*(4,2).

5. Este posibilă modificarea tipului coloanei *salariu* în *NUMBER*(6,2)?

6. Setaţi o valoare *DEFAULT* pentru coloana salariu.

7. Modificaţi tipul coloanei *comision* în *NUMBER*(2, 2) şi al coloanei *salariu* în *NUMBER*(10,2), în cadrul aceleiaşi instrucţiuni.

8. Actualizați valoarea coloanei *comision*, setând-o la valoarea 0.1 pentru salariaţii al căror job începe cu litera I. (*UPDATE*)

9. Modificaţi tipul de date al coloanei *email* în *VARCHAR2*.

10. Adăugaţi coloana *nr_telefon* în tabelul *ANGAJATI_pnu*, setându-i o valoare implicită.

11. Vizualizaţi înregistrările existente. Suprimaţi coloana *nr_telefon*.

Ce efect ar avea o comandă *ROLLBACK* în acest moment?

12. Redenumiţi tabelul *ANGAJATI_pnu* în *ANGAJATI3_pnu*.

13. Consultaţi vizualizarea TAB din dicţionarul datelor. Redenumiţi *angajati3_pnu* în *angajati_pnu*.

14. Suprimaţi conţinutul tabelului *angajati10_pnu*, fără a suprima structura acestuia.

15. Creaţi tabelul *DEPARTAMENTE_pnu*, corespunzător schemei relaţionale:
DEPARTAMENTE_pnu (*cod_dep# number(2)*, *nume varchar2(15)*, *cod_director number(4))*
specificând doar constrângerea *NOT NULL* pentru nume (nu precizaţi deocamdată constrângerea de cheie primară).

```
CREATE TABLE departamente_pnu ( … );
DESC departamente_pnu
```

16. Introduceţi următoarele înregistrări în tabelul DEPARTAMENTE_pnu:

Cod_dep	**Nume**	**Cod_director**
10	Administrativ	100
20	Proiectare	101
30	Programare	Null

17. Introduceți constângerea de cheie primară asupra coloanei *cod_dep*, fără suprimarea şi recrearea tabelului (comanda *ALTER*).

Observație:

- Introducerea unei constrângeri după crearea tabelului presupune că toate liniile existente în tabel la momentul respectiv satisfac noua constrângere.
- Specificarea constrângerilor permite numirea acestora.
- In situaţia in care constrângerile sunt precizate la nivel de coloană sau tabel (în *CREATE TABLE*) ele vor primi implicit nume atribuite de sistem, dacă nu se specifică vreun alt nume într-o clauză *CONSTRAINT*.

Exemplu : CREATE TABLE alfa (
X NUMBER CONSTRAINT nn_x NOT NULL,
Y VARCHAR2 (10) NOT NULL
);

18. Să se precizeze constrângerea de cheie externă pentru coloana *cod_dep* din *ANGAJATI_pnu*:

a) fără suprimarea tabelului (*ALTER TABLE*);

b) prin suprimarea şi recrearea tabelului, cu precizarea noii constrângeri la nivel de coloană ({*DROP, CREATE*} *TABLE*). De asemenea, se vor mai preciza constrângerile (la nivel de coloană, în măsura în care este posibil):

- *PRIMARY KEY* pentru *cod_ang*;

- *FOREIGN* KEY pentru *cod_sef*;
- *UNIQUE* pentru combinaţia *nume* + *prenume*;
- *UNIQUE* pentru *email*;
- *NOT NULL* pentru *nume*;
- verificarea *cod_dep* > 0;
- verificarea ca salariul să fie mai mare decât comisionul*100.

19. Suprimaţi şi recreaţi tabelul, specificând toate constrângerile la nivel de tabel (în măsura în care este posibil).

20. Reintroduceţi date în tabel, utilizând (şi modificând, dacă este necesar) comenzile salvate anterior.

21. Ce se întâmplă dacă se încearcă suprimarea tabelului departamente_pnu?

22. Analizaţi structura vizualizărilor *USER_TABLES*, *TAB*, *USER_CONSTRAINTS*.

Observaţie: Pentru a afla informaţii despre tabelele din schema curentă, sunt utile cererile:

SELECT * FROM tab;

sau

SELECT table_name FROM user_tables;

23. a) Listaţi informaţiile relevante (cel puţin nume, tip şi tabel) despre constrângerile asupra tabelelor *angajati_pnu* şi *departamente_pnu*.

```
SELECT constraint_name, constraint_type, table_name
FROM     user_constraints
WHERE  lower(table_name) IN ('angajati_pnu ', 'departamente_pnu ');
```

Observaţie*:* Tipul constrângerilor este marcat prin:

- P – pentru cheie primară
- R – pentru constrângerea de integritate referenţială (cheie externă);
- U – pentru constrângerea de unicitate (*UNIQUE*);
- C – pentru constrângerile de tip *CHECK*.

b) Aflaţi care sunt coloanele la care se referă constrângerile asupra tabelelor *angajati_pnu* şi *departamente_pnu*.

```
SELECT table_name, constraint_name, column_name
FROM   user_cons_columns
WHERE LOWER(table_name) IN ('angajati_pnu ', 'departamente_pnu ');
```

24. Introduceţi constrângerea *NOT NULL* asupra coloanei *email*.

25. (Încercaţi să) adăugaţi o nouă înregistrare în tabelul *ANGAJATI_pnu*, care să corespundă codului de departament 50. Se poate?

26. Adăugaţi un nou departament, cu numele Testare, codul 60 şi directorul null în *DEPARTAMENTE_pnu*. *COMMIT*.

27. (Încercaţi să) ştergeţi departamentul 20 din tabelul *DEPARTAMENTE_pnu*. Comentaţi.

28. Ştergeţi departamentul 60 din *DEPARTAMENTE_pnu*. *ROLLBACK*.

29. (Încercaţi să) introduceţi un nou angajat, specificând valoarea 114 pentru *cod_sef*. Ce se obţine?

30. Adăugaţi un nou angajat, având codul 114. Încercaţi din nou introducerea înregistrării de la exerciţiul 29.

Ce concluzii reies din exerciţiile precedente? Care este ordinea de inserare, atunci când avem constrângeri de cheie externă?

31. Se doreşte ştergerea automată a angajaţilor dintr-un departament, odată cu suprimarea departamentului. Pentru aceasta, este necesară introducerea clauzei *ON DELETE CASCADE* în definirea constrângerii de cheie externă. Suprimaţi constrângerea de cheie externă asupra tabelului *ANGAJATI_pnu* şi reintroduceţi această constrângere, specificând clauza *ON DELETE CASCADE*.

32. Ştergeţi departamentul 20 din *DEPARTAMENTE_pnu*. Ce se întâmplă? *Rollback*.

33. Introduceţi constrângerea de cheie externă asupra coloanei *cod_director* a tabelului *DEPARTAMENTE_pnu*. Se doreşte ca ştergerea unui angajat care este director de departament să implice setarea automată a valorii coloanei *cod_director* la *null*.

34. Actualizaţi tabelul *DEPARTAMENTE_PNU*, astfel încât angajatul având codul 102 să devină directorul departamentului 30. Ştergeţi angajatul având codul 102 din tabelul *ANGAJATI_pnu*. Analizaţi efectele comenzii. *Rollback*.
Este posibilă suprimarea angajatului având codul 101? Comentaţi.

35. Adăugaţi o constrângere de tip *check* asupra coloanei salariu, astfel încât acesta să nu poată depăşi 30000.

36. Încercaţi actualizarea salariului angajatului 100 la valoarea 35000.

37. Dezactivaţi constrângerea creată anterior şi reîncercaţi actualizarea. Ce se întâmplă dacă încercăm reactivarea constrângerii?

V. [Definirea secvenţelor]

- Secvenţa este un obiect al bazei de date ce permite generarea de numere întregi unice care pot fi folosite ca valori pentru cheia primară sau pentru coloane numerice

pe care s-a definit o constrângere de unicitate. Secvenţele sunt independente de tabele, astfel încât aceeaşi secvenţă poate fi folosită în mai multe tabele.

- Crearea secvenţelor se realizează prin comanda *CREATE SEQUENCE*, a cărei sintaxă este:

CREATE SEQUENCE *nume_secv*

[INCREMENT BY *n]*

[START WITH *n]*

[{MAXVALUE *n* | ***NOMAXVALUE}]***

[{MINVALUE *n* | *NOMINVALUE****}]***

[{CYCLE | ***NOCYCLE}]***

[{CACHE *n* | ***NOCACHE}]***

- La definirea unei secvenţe se pot specifica:
  - numele secvenţei
  - diferenţa dintre 2 numere generate succesiv, implicit fiind 1 (*INCREMENT BY*);
  - numărul iniţial, implicit fiind 1 (*START WITH*);
  - valoarea maximă, implicit fiind 10^{27} pentru o secvenţă ascendentă şi –1 pentru una descendentă;
  - valoarea minimă, implicit fiind 1 pentru o secvenţă ascendentă şi -10^{27} pentru o secvenţă descendentă;
  - dacă secvenţa ciclează după ce atinge limita; (*CYCLE*
  - câte numere să încarce în *cache*, implicit fiind încărcate 20 de numere (*CACHE*).
- Informaţii despre secvenţe găsim în dicţionarul datelor. Pentru secvenţele utilizatorului curent, interogăm *USER_SEQUENCES*. Alte vizualizări utile sunt *ALL_SEQUENCES* şi *DBA_SEQUENCES*.
- Pseudocoloanele *NEXTVAL* şi *CURRVAL* permit lucrul efectiv cu secvenţele.
  - *Nume_secv.NEXTVAL* – returnează următoarea valoare a secvenţei, o valoare unică la fiecare referire. Trebuie aplicată cel puţin o dată înainte de a folosi *CURRVAL*;
  - *Nume_secv.CURRVAL* – obţine valoarea curentă a secvenţei.

Observaţie: Pseudocoloanele se pot utiliza în:

- lista *SELECT* a comenzilor ce nu fac parte din subcereri;
- lista *SELECT* a unei cereri ce apare într un *INSERT*;
- clauza *VALUES* a comenzii INSERT;
- clauza *SET* a comenzii *UPDATE*.

Observație: Pseudocoloanele nu se pot utiliza:

- în lista *SELECT* a unei vizualizări;
- într-o comanda *SELECT* ce conține *DISTINCT*, *GROUP BY*, *HAVING* sau *ORDER BY*;
- într-o subcerere în comenzile *SELECT*, *UPDATE*, *DELETE*
- în clauza *DEFAULT* a comenzilor *CREATE TABLE* sau *ALTER TABLE*.

- Ştergerea secvenţelor se face cu ajutorul comenzii *DROP SEQUENCE*.

  DROP SEQUENCE *nume_secventa;*

VI. [Exerciţii – secvențe]

38. Creaţi o secvenţă pentru generarea codurilor de departamente, *SEQ_DEPT_PNU*. Secvenţa va începe de la 400, va creşte cu 10 de fiecare dată şi va avea valoarea maximă 10000, nu va cicla şi nu va încărca nici un număr înainte de cerere.

39. Să se selecteze informaţii despre secvenţele utilizatorului curent (nume, valoare minimă, maximă, de incrementare, ultimul număr generat).

40. Creaţi o secvenţă pentru generarea codurilor de angajaţi, *SEQ_EMP_PNU*.

41. Să se modifice toate liniile din *EMP_PNU* (dacă nu mai există, îl recreaţi), regenerând codul angajaţilor astfel încât să utilizeze secvenţa *SEQ_EMP_PNU* şi să avem continuitate în codurile angajaţilor.

42. Să se insereze câte o înregistrare nouă în *EMP_PNU* şi *DEPT_PNU* utilizând cele 2 secvenţe create.

43. Să se selecteze valorile curente ale celor 2 secvenţe.

```
SELECT seq_emp_pnu.currval
FROM dual ;
```

44. Ştergeţi secvenţa *SEQ_DEPT_PNU*.