

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, instrucțiunile *LDD* sunt utilizate pentru definirea structurii corespunzătoare obiectelor unei scheme: tabele, vizualizări, vizualizări materializate, indecși, sinonime, clustere, proceduri și funcții stocate, declanșatori, pachete stocate etc.
- Aceste instrucțiuni permit:
 - crearea, modificarea și suprimarea obiectelor unei scheme și a altor obiecte ale bazei de date, inclusiv baza înșăși și utilizatorii acesteia (*CREATE*, *ALTER*, *DROP*);
 - modificarea numelor obiectelor unei scheme (*RENAME*);
 - ștergerea datelor din obiectele unei scheme, fără suprimarea structurii obiectelor respective (*TRUNCATE*).
- Implicit, o instrucțiune *LDD* permanentizează (*COMMIT*) efectul tuturor instrucțiunilor precedente și marchează începutul unei noi tranzacții.
- Instrucțiunile *LDD* au efect imediat asupra bazei de date și înregistrează informația în dicționarul datelor.
- Definirea unui obiect presupune: crearea (*CREATE*), modificarea (*ALTER*) și suprimarea sa (*DROP*).

Reguli de numire a obiectelor bazei de date

- Identificatorii obiectelor trebuie să înceapă cu o literă și să aibă maximum 30 de caractere, cu excepția numelui bazei de date care este limitat la 8 caractere și celui al legăturii unei baze de date, a cărui lungime poate atinge 128 de caractere.
- Numele poate conține caracterele A-Z, a-z, 0-9, _, \$ și #.
- Două obiecte ale aceluiași utilizator al server-ului *Oracle* nu pot avea același 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 asupra unui tabel pot fi de următoarele tipuri:**
 - *NOT NULL* – coloana nu poate conține valoarea *Null*; (*NOT NULL*)
 - *UNIQUE* – pentru coloane sau combinații 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* – stabilește o relație de cheie externă între o coloană a tabelului și 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“ și coloana corespunzătoare din acest tabel;
 - *ON DELETE CASCADE* determină ca, odată cu ștergerea unei linii din tabelul „părinte“, să fie șterse și liniile dependente din tabelul „copil“;

- *ON DELETE SET NULL* determină modificarea automată a valorilor cheii externe la valoarea *null*, atunci când se șterge valoarea „părinte”.
- *CHECK* – specifică o condiție 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]	Definește un șir de caractere de dimensiune variabilă, având lungimea maximă de <i>n</i> octeți sau caractere. Valoarea maximă a lui <i>n</i> corespunde la 4000 octeți, iar cea minimă este de un octet sau un caracter.
CHAR(n) [BYTE CHAR]	Reprezintă un șir de caractere de lungime fixă având <i>n</i> octeți sau caractere. Valoarea maximă a lui <i>n</i> corespunde la 2000 octeți. Valoarea implicită și minimă este de un octet.
NUMBER(p, s)	Reprezintă un număr având <i>p</i> cifre, dintre care <i>s</i> cifre formează partea zecimală
LONG	Conține șiruri de caractere având lungime variabilă, care nu pot ocupa mai mult de 2GB.
DATE	Reprezintă date calendaristice valide, între 1 ianuarie 4712 i.Hr. și 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 poziția 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 micșorată decât dacă acea coloană conține numai valori *null* sau dacă tabelul nu conține 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

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

DROP TABLE *nume_tabel*;

- Pentru ștergerea conținutului unui tabel și păstrarea structurii acestuia se poate utiliza comanda:

TRUNCATE TABLE *nume_tabel*;

Observație: Fiind operație *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 secvență.

RENAME *nume1_obiect TO* *nume2_obiect*;

Observații:

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

5. Consultarea dicționarului datelor

- Informații despre tabelele create se găsesc în vizualizările din dicționarul datelor:
 - *USER_TABLES* – informații complete despre tabelele utilizatorului.
 - *TAB* – informații de bază despre tabelele existente în schema utilizatorului.
- Informații 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 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))

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* și *salariu*;
- c) cu precizarea cheii primare la nivel de tabel și a constrângerilor *NOT NULL* pentru coloanele *nume* și *salariu*.

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

Observație: Nu pot exista două tabele cu același nume în cadrul unei scheme, deci recrearea unui tabel va fi precedată de suprimarea sa prin comanda:

DROP TABLE ANGAJATI_pnu;

2. Adăugați 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 și a patra înregistrare vor fi introduse specificând coloanele pentru care introduceți date efectiv, iar celelalte vor fi inserate fără precizarea coloanelor în comanda *INSERT*.

Salvați comenzile de inserare.

- 3. Creați tabelul *ANGAJATI10_pnu*, prin copierea angajaților din departamentul 10 din tabelul *ANGAJATI_pnu*. Listați structura noului tabel. Ce se observă?
- 4. Introduceți coloana *comision* în tabelul *ANGAJATI_pnu*. Coloana va avea tipul de date *NUMBER(4,2)*.
- 5. Este posibilă modificarea tipului coloanei *salariu* în *NUMBER(6,2)*?
- 6. Setati o valoare *DEFAULT* pentru coloana *salariu*.
- 7. Modificați tipul coloanei *comision* în *NUMBER(2, 2)* și al coloanei *salariu* în *NUMBER(10,2)*, în cadrul aceleiași instrucțiuni.
- 8. Actualizați valoarea coloanei *comision*, setând-o la valoarea 0.1 pentru salariații al căror job începe cu litera I. (*UPDATE*)
- 9. Modificați tipul de date al coloanei *email* în *VARCHAR2*.
- 10. Adăugați coloana *nr_telefon* în tabelul *ANGAJATI_pnu*, setându-i o valoare implicită.
- 11. Vizualizați înregistrările existente. Suprimați coloana *nr_telefon*.

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

12. Redenumiți tabelul *ANGAJATI_pnu* în *ANGAJATI3_pnu*.

13. Consultați vizualizarea TAB din dicționarul datelor. Redenumiți *angajati3_pnu* în *angajati_pnu*.

14. Suprimați conținutul tabelului *angajati10_pnu*, fără a suprima structura acestuia.

15. Creați tabelul *DEPARTAMENTE_pnu*, corespunzător schemei relaționale:

DEPARTAMENTE_pnu (*cod_dep#* *number(2)*, *nume* *varchar2(15)*, *cod_director* *number(4)*)
specificând doar constrângerea *NOT NULL* pentru *nume* (nu precizați deocamdată
constrângerea de cheie primară).

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

16. Introduceți următoarele înregistrări în tabelul *DEPARTAMENTE_pnu*:

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

17. Introduceți constrângerea de cheie primară asupra coloanei *cod_dep*, fără suprimarea și 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.
- În situația în 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 și 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 combinația *nume + prenume*;
- *UNIQUE* pentru *email*;
- *NOT NULL* pentru *nume*;
- verificarea *cod_dep > 0*;
- verificarea ca salariul să fie mai mare decât comisionul*100.

19. Suprimați și recreați tabelul, specificând toate constrângerile la nivel de tabel (în măsura în care este posibil).

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

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

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

Observație: Pentru a afla informații despre tabelele din schema curentă, sunt utile cererile:

```
SELECT * FROM tab;
```

sau

```
SELECT table_name FROM user_tables;
```

23. a) Listați informațiile relevante (cel puțin nume, tip și tabel) despre constrângerile asupra tabelelor *angajati_pnu* și *departamente_pnu*.

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

Observație: Tipul constrângerilor este marcat prin:

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

b) Aflați care sunt coloanele la care se referă constrângerile asupra tabelelor *angajati_pnu* și *departamente_pnu*.

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

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

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

26. Adăugați un nou departament, cu numele Testare, codul 60 și directorul null în *DEPARTAMENTE_pnu*. *COMMIT*.
27. (Încercați să) ștergeți departamentul 20 din tabelul *DEPARTAMENTE_pnu*. Comentați.
28. Ștergeți departamentul 60 din *DEPARTAMENTE_pnu*. *ROLLBACK*.
29. (Încercați să) introduceți un nou angajat, specificând valoarea 114 pentru *cod_sef*. Ce se obține?
30. Adăugați un nou angajat, având codul 114. Încercați din nou introducerea înregistrării de la exercițiul 29.
- Ce concluzii reies din exercițiile precedente? Care este ordinea de inserare, atunci când avem constrângeri de cheie externă?**
31. Se dorește ștergerea automată a angajaților dintr-un departament, odată cu suprimarea departamentului. Pentru aceasta, este necesară introducerea clauzei *ON DELETE CASCADE* în definirea constrângerii de cheie externă. Suprimați constrângerea de cheie externă asupra tabelului *ANGAJATI_pnu* și reintroduceți această constrângere, specificând clauza *ON DELETE CASCADE*.
32. Ștergeți departamentul 20 din *DEPARTAMENTE_pnu*. Ce se întâmplă? *Rollback*.
33. Introduceți constrângerea de cheie externă asupra coloanei *cod_director* a tabelului *DEPARTAMENTE_pnu*. Se dorește ca ștergerea unui angajat care este director de departament să implice setarea automată a valorii coloanei *cod_director* la *null*.
34. Actualizați tabelul *DEPARTAMENTE_PNU*, astfel încât angajatul având codul 102 să devină directorul departamentului 30. Ștergeți angajatul având codul 102 din tabelul *ANGAJATI_pnu*. Analizați efectele comenzii. *Rollback*.
Este posibilă suprimarea angajatului având codul 101? Comentați.
35. Adăugați o constrângere de tip *check* asupra coloanei *salariu*, astfel încât acesta să nu poată depăși 30000.
36. Încercați actualizarea salariului angajatului 100 la valoarea 35000.
37. Dezactivați constrângerea creată anterior și reîncercați actualizarea. Ce se întâmplă dacă încercăm reactivarea constrângerii?

V. [Definirea secvențelor]

- Secvența 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. Secvențele sunt independente de tabele, astfel încât aceeași secvență poate fi folosită în mai multe tabele.

- Crearea secvențelor 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 secvențe se pot specifica:
 - numele secvenței
 - diferența dintre 2 numere generate succesiv, implicit fiind 1 (*INCREMENT BY*);
 - numărul inițial, implicit fiind 1 (*START WITH*);
 - valoarea maximă, implicit fiind 10^{27} pentru o secvență ascendentă și -1 pentru una descendentă;
 - valoarea minimă, implicit fiind 1 pentru o secvență ascendentă și -10^{27} pentru o secvență descendentă;
 - dacă secvența ciclează după ce atinge limita; (*CYCLE*)
 - câte numere să încarce în *cache*, implicit fiind încărcate 20 de numere (*CACHE*).
- Informații despre secvențe găsim în dicționarul datelor. Pentru secvențele utilizatorului curent, interogăm *USER_SEQUENCES*. Alte vizualizări utile sunt *ALL_SEQUENCES* și *DBA_SEQUENCES*.
- Pseudocoloanele *NEXTVAL* și *CURRVAL* permit lucrul efectiv cu secvențele.
 - *Nume_secv.NEXTVAL* – returnează următoarea valoare a secvenței, o valoare unică la fiecare referire. Trebuie aplicată cel puțin o dată înainte de a folosi *CURRVAL*;
 - *Nume_secv.CURRVAL* – obține valoarea curentă a secvenței.

Observație: 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 comandă *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*.
- Ștergerea secvențelor se face cu ajutorul comenzii *DROP SEQUENCE*.

DROP SEQUENCE nume_secventa;

VI. [Exerciții – secvențe]

38. Creați o secvență pentru generarea codurilor de departamente, *SEQ_DEPT_PNU*.
Secvența va începe de la 400, va crește cu 10 de fiecare dată și va avea valoarea maximă 10000, nu va cicla și nu va încărca nici un număr înainte de cerere.

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

40. Creați o secvență pentru generarea codurilor de angajați, *SEQ_EMP_PNU*.

41. Să se modifice toate liniile din *EMP_PNU* (dacă nu mai există, îl recreați), regenerând codul angajaților astfel încât să utilizeze secvența *SEQ_EMP_PNU* și să avem continuitate în codurile angajaților.

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

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

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

44. Ștergeți secvența *SEQ_DEPT_PNU*.