

CAPITOLUL 5. INCARCAREA SI ACTUALIZAREA DATELOR CU COMENZI SQL

5.1. ADĂUGAREA DE NOI TUPLURI

Actualizarea datelor se referă la adăugarea unor noi rânduri într-o tabelă (cu comanda **INSERT**), la modificarea valorilor uneia sau mai multor valori dintr-un rând (cu comanda **UPDATE**) și la ștergerea unui rând dintr-o tabelă (cu comanda **DELETE**).

În vederea adăugării unor rânduri noi într-o tabelă sau într-o viziune se utilizează comanda:

```
INSERT INTO nume-tabelă  
[(nume-col1,nume-col2,...)]  
{VALUES (valoare 1,valoare2,...) | cerere );
```

Pentru *nume-col1,nume-col2...* precizate în paranteze vor fi furnizate valorile corespunzătoare, iar coloanelor nespecificate le sunt atașate valori nule. Coloanele pot fi precizate în orice ordine, însă trebuie asigurată corespondența între numele coloanelor și valorile furnizate.

În cazul în care anumite coloane nu sunt specificate explicit se impune ca ordinea în care apar valorile în comanda **INSERT** să coincidă cu cea în care coloanele au fost definite la crearea tabelului.

Dacă nu se mai cunoaște ordinea de declarare a coloanelor se folosește comanda **DESCRIBE** care va afișa lista coloanelor definite pentru tabela respectivă, tipul și lungimea lor.

Prin forma **INSERT...VALUES** se introduce în tabelă un singur rând. Cu ajutorul valorii **NULL** se pot introduce valori nule. Pentru a furniza valori pentru o coloană de tip dată calendaristică se poate folosi funcția **TO_DATE** sau cuvântul cheie **SYSDATE**.

Funcția **TO_DATE** permite furnizarea valorilor într-un format diferit de cel standard.

Specificarea cererii din comanda **INSERT** determină copierea unor date dintr-o tabelă în alta pe atâtea rânduri câte au rezultat din cererea SQL.

Exemple:

1) Să se adauge un nou rând în tabela **PRODUSE**.

```
SQL> INSERT INTO PRODUSE VALUES
```

1 (100000,11111,'MESE 15/20',7,'27-JUN-92','BUC')

1 record created.

2) Să se creeze o nouă tabelă, EXEMPLU, cu un singur câmp numeric, identic cu coloana CODY a tabelii PRODUSE. Să se introducă în această nouă tabelă codul produselor din tabela PRODUSE.

SQL> CREATE TABLE EXEMPLU

2 (CODY NUMBER(6) NOT NULL);

Table created.

SQL> INSERT INTO EXEMPLU

2 SELECT CODY FROM PRODUSE;

5 records created.

5.2. MODIFICAREA TUPLURILOR DIN TABELE

În funcție de momentul în care se dorește realizarea modificărilor asupra bazei de date, utilizatorul poate folosi una din următoarele comenzi: **SET AUTOCOMMIT IMM[EDIATE]** (schimbările se efectuează imediat); **SET AUTOCOMMIT OFF** (schimbările sunt păstrate într-un buffer). La execuția comenzii **COMMIT** se permanentizează schimbările efectuate, iar la execuția comenzii **ROLLBACK** se renunță la schimbările realizate.

În scopul modificării datelor dintr-o tabelă se utilizează una din formele sintactice ale comenzii **UPDATE**:

UPDATE nume-tabelă [sinonim]

SET nume-crt=expresie,nume -expresie,...

[WHERE condiție];

UPDATE nume-tabelă [sinonim]

SET (nume-col, nume-col,...)=(subcerere)

[WHERE condiție];

Sunt două posibilitati de modificare. Una constă în furnizarea în mod explicit a fiecărei valori pentru câmpurile care trebuie modificate iar cealaltă posibilitate constă în obținerea valorilor în urma unei cereri SQL.

Dacă nu este specificată clauza **WHERE** se vor modifica toate rândurile tabelii.

(nume-col,nume-col,...)=(subcerere) impune ca cererea să conțină pentru fiecare rând un număr de valori corespunzător numărului de coloane din

paranteza care precede caracterul =. În cazul în care se modifică o singură coloană, parantezele pot fi omise.

Exemple:

1) Să se modifice câmpul NRSAL din tabela SALARIAȚI, pentru depozitul cu codul 130000, atribuindu-i valoarea 11.

```
SQL> UPDATE DEPOZITE  
2 SET NRSAL=11  
3 WHERE CODD= 130000;  
1 record updated.
```

2) Să se modifice data de livrare, cantitatea solicitată și prețul de livrare pentru produsul cu codul 13333 din comanda cu numărul 211111.

```
SQL> UPDATE COMENZI  
2 SET DATAL=SYSDATE,CANT=50, PRET=42000  
3 WHERE CODP=13333 AND  
4 NRCOM=211111;  
1 record updated.
```

3) Să se modifice data de livrare cu data actuală pentru toate produsele cu codul egal cu 13333, din toate comenzile.

```
SQL> UPDATE COMENZI  
2 SET DATAL=SYSDATE  
3 WHERE CODP=13333;  
1 record updated.
```

4) Să se mărească salariul cu 15% pentru salariații care au o funcție identică cu CARMEN ANA.

```
SQL> UPDATE SALARIAȚI  
2 SET SALA=SALA*1.15  
3 WHERE FUNCT IN  
4 (SELECT FUNCT  
5 FROM SALARIAȚI  
6 WHERE NUME='CARMEN ANA');  
11 records updated.
```

5.3. ȘTERGEREA TUPLURILOR DIN TABELE

Ștergerea unor rânduri dintr-o tabelă se realizează cu următoarea comandă:

DELETE FROM nume tabela [WHERE condiție];

Folosirea clauzei **WHERE** determină ștergerea acelor rânduri care îndeplinesc condiția impusă. În această clauză pot fi folosite și subcereri. Dacă nu este specificată nici o condiție, se șterg toate rândurile tabelului. Ștergerile accidentale pot fi omise, restaurându-se valorile inițiale prin comanda **AUTOCOMMIT OFF**.

Exemple:

1) Să se ștergă datele din tabela DEPOZITE.

SQL> DELETE FROM DEPOZITE;

5 records deleted.

2) Să se ștergă datele pentru depozitele care au codul mai mare sau egal cu 100000.

SQL> DELETE FROM DEPOZITE

2 WHERE CODD>=100000;

2 records deleted.

3) Să se scrie comanda pentru ștergerea datelor despre salariații VLAD VASILE. Ștergerile să nu fie efectuate imediat ci ulterior.

SQL> SET AUTOCOMMIT OFF

SQL> DELETE FROM SALARIAȚI

2 WHERE NUME='VLAD VASILE';

1 record deleted.

SQL> COMMIT;

4) Să se ștergă datele salariaților care au aceeași funcție cu a lui PAUL ȘTEFAN. Ștergerile să nu fie realizate imediat. Ulterior să se renunțe la aceste ștergeri.

SQL> SET AUTOCOMMIT OFF

SQL> DELETE FROM SALARIAȚI

2 WHERE FUNCT IN

3 (SELECT FUNCT FROM SALARIAȚI

4 WHERE NUME='PAUL ȘTEFAN');

5 records deleted.