

Lucrare laborator 2

Crearea, ștergerea și modificarea tabelelor. Utilizarea comenzilor CREATE TABLE, DESCRIBE table, ALTER TABLE, DROP TABLE, RENAME TABLE, SHOW TABLES

După cum s-a discutat în lucrarea de laborator anterioara, o baza de date este o colecție de informații dintr-un anumit domeniu. Informația într-o baza de date este structurată sub forma de tabele.

În limbajul SQL lucrul cu tabele este materializat printr-o serie de instrucțiuni pentru creare, modificare, ștergere și interogare a tabelor și înregistrărilor din tabele.

În lucrarea de laborator de astăzi vom parcurge câteva instrucțiuni de creare, ștergere, modificare și redenumire de tabele.

1. Instrucțiunea CREATE TABLE.

Formatul general al acestei instrucțiuni este:

```
CREATE TABLE [IF NOT EXISTS] tbl_nume [(definiții de creare,...)]  
[PRIMARY KEY] [reference_definition]
```

Instrucțiunea CREATE TABLE, crează un tabel cu numele specificat în *tbl_nume* iar crearea se face în baza de date curentă (selectată cu comanda USE). De reținut este faptul că numele tabelului poate să cuprindă orice caracter cu excepția caracterelor \, /, ., :, *, ", ?, <, >.

Un mesaj de eroare va fi afișat în cazul în care nici o baza de date nu a fost selectată anterior utilizării. Ca și în cazul creării bazelor de date, clauza IF NOT EXISTS va preveni apariția mesajului de eroare dacă un alt tabel cu numele *tbl_nume* a fost creat anterior.

Definițiile de creare trebuie să se încadreze în următorul format:

```
col_nume type [NOT NULL | NULL] [DEFAULT default_value]  
[AUTO_INCREMENT]
```

Unde

- *col_nume* este numele coloanei ce urmează a fi definită în tabel
- *type* este tipul de date ce definește *col_nume*
- opțiunile NULL sau NOT NULL specifică dacă o valoare trebuie sau nu specificată în coloana respectivă. Dacă nici una din valori nu este

specificată, atunci se va considera valoarea NULL a fi implicita, adică nu este necesară specificarea unei valori.

- DEFAULT value reprezintă o valoare ce va fi înregistrată în coloana respectivă în situația în care nici o valoare nu este specificată. Valoarea precizată în default_value trebuie să fie o constantă și nu poate fi o funcție sau o expresie.
- pentru tipurile de date numerice, opțiunea AUTO_INCREMENT specifică faptul că la inserarea unei noi înregistrări în tabel, valoarea coloanei va fi incrementată. Coloana marcată ca AUTO_INCREMENT trebuie să fie cheia primară în tabel și trebuie specificată în definiția cheii primare
- PRIMARY KEY arată că, coloana specificată este cheia unică în tabel. De reținut este faptul că, coloana marcată ca PRIMARY KEY trebuie specificată ca NOT NULL.

Ca un exemplu pentru această instrucțiune, vom reconsidera exemplul din lucrarea de laborator anterioară pentru crearea unui tabel cu datele studenților din facultate.

```
CREATE DATABASE IF NOT EXISTS facultate;
USE facultate;
CREATE TABLE IF NOT EXISTS student (
    nrleg int NOT NULL AUTO_INCREMENT primary key,
    nume varchar(20) not null,
    prenume varchar(20) not null,
    init char(1),
    sex char(1) default 'm' not null,
    datanastere date);
```

Tabelul creat în acest exemplu cuprinde următoarele câmpuri: nrleg, nume, prenume, init, sex și datanastere.

Pentru:

- nrleg, tipul de date este întreg ce nu poate lua valori nule și se incrementează automat iar acest câmp este cheia primară în tabel;
- nume, tipul de date este șir de caractere cu lungimea de 20 ce nu poate lua valori nule;
- prenume, tipul de date este șir de caractere cu lungimea de 20 ce nu poate lua valori nule;
- init, tipul de date este caracter cu lungimea de 1 care are ca valoare implicită valoarea 'm'
- datanastere, tipul de date este dată calendaristică de forma YYYY-MM-DD

2. Instrucțiunea DESCRIBE.

Formatul general al acestei instrucțiuni este:

```
{DESCRIBE | DESC} tbl_num
```

Instrucțiunea DESCRIBE, afisează informații despre coloanele tabelului specificat in *tbl_num*.

Instrucțiunea poate fi scrisa prescurtat DESC și are același efect.

Exemplu:

```
DESCRIBE student;
```

Această comandă va afișa următorul rezultat:

Field	Type	NULL	Key	Default	Extra
nrleg	int(11)		PRI	[NULL]	auto_increment
nume	varchar(20)				
prenume	varchar(20)				
init	char(1)	YES		[NULL]	
sex	char(1)	YES		m	
datanastere	date	YES		[NULL]	

3. Instrucțiunea ALTER TABLE.

Formatul general al instrucțiunii ALTER TABLE este:

```
ALTER TABLE tbl_num alter_spec [, alter_spec ...]
```

Această instrucțiune da posibilitatea utilizatorului de a modifica structura unui table deja creat. Pentru exemplu, această instrucțiune poate adăuga sau șterge coloane, crea sau distruge indexi, să schimbe tipul unei coloane existente sau să redenumască coloane sau tabelul însuși.

Formatul specificațiilor de tabel este:

alter_spec:

```
ADD [COLUMN] create_definition [FIRST | AFTER column_name ]  
sau ADD [COLUMN] (create_definition, create_definition,...)  
sau ADD INDEX [index_name] (index_col_name,...)  
sau ADD PRIMARY KEY (index_col_name,...)  
sau ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}  
sau CHANGE [COLUMN] old_col_name create_definition  
sau MODIFY [COLUMN] create_definition [FIRST | AFTER column_name]  
sau DROP [COLUMN] col_name
```

sau RENAME [TO] new_tbl_name

După executarea comenzii ALTER TABLE, se va crea o copie a tabelului original iar alterarea este executată pe copie. În momentul terminării execuției, tabelul original este șters și copia va fi redenumită cu numele său. Aceasta tehnică este folosită pentru a preveni orice eroare de modificare a tabelului.

Descrierea specificațiilor instrucțiunii ALTER TABLE

- opțiunea **ADD [COLUMN] create_definition [FIRST | AFTER column_name]** – adaugă o coloană nouă în tabelul specificat în *tbl_nume*. Coloana adăugată poate fi introdusă la începutul tabelului cu opțiunea FIRST sau după coloana specificată în *column_name* din opțiunea AFTER

Ex:

```
ALTER TABLE student ADD new INT DEFAULT 0 AFTER nrleg;
```

Comanda de mai sus va adăuga în tabelul student o nouă coloană cu numele new de tip întreg cu valoare implicită 0, iar coloana new va fi introdusă după coloana nrleg.

- opțiunea **ADD INDEX [index_name] (index_col_name,...)** – are ca efect adăugarea unui nou index tabelului specificat în *tbl_nume*.

Ex:

```
ALTER TABLE student ADD INDEX new_index (new);
```

Comanda de mai sus va adăuga un nou index tabelului prin câmpul new iar indexul se va numi new_index.

- opțiunea **ADD PRIMARY KEY (index_col_name,...)** – va adăuga o cheie primară tabelului specificat în *tbl_nume*.

Ex:

```
CREATE TABLE IF NOT EXISTS student1 (  
    nrleg int NOT NULL,  
    new int NOT NULL,  
    nume varchar(20) not null,  
    prenume varchar(20) not null,  
    init char(1),  
    sex char(1) default 'm' not null,  
    datanastere date);
```

```
ALTER TABLE student1 ADD PRIMARY KEY (new);
```

Comenzile de mai sus vor crea în prima linie un nou tabel cu numele student1 și apoi va adăuga coloana new ca fiind cheie primară a tabelului student1.

- opțiunea **ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}** – adaugă, modifică sau șterge o valoare default pentru câmpul specificat în col_name din tabelul tbl_nume.

Ex:

```
ALTER TABLE student ALTER nrleg SET DEFAULT 5
```

Comanda de mai sus va seta în tabelul student pentru câmpul nrleg valoarea implicită 5

```
ALTER TABLE student ALTER nrleg SET DEFAULT 2
```

Comanda de mai sus va modifica în tabelul student pentru câmpul nrleg valoarea implicită 2

```
ALTER TABLE student ALTER nrleg DROP DEFAULT
```

Comanda de mai sus va șterge în tabelul student pentru câmpul nrleg valoarea implicită.

- opțiunea **CHANGE [COLUMN] old_col_name create_definition** – schimbă numele sau tipul coloanei specificate în old_col_name.

Ex:

```
ALTER TABLE student CHANGE new new1 varchar(3);
```

Comanda de mai sus va schimba în tabelul student numele coloanei new în new1 iar tipul coloanei din integer va deveni șir de caractere de 3.

```
ALTER TABLE student CHANGE new1 new1 INT;
```

Comanda de mai sus va schimba de aceasta dată numai tipul de date al câmpului new1 din șir de caractere în întreg.

- opțiunea **MODIFY [COLUMN] create_definition [FIRST | AFTER column_name]** – modifică definiția de tip a câmpului specificat în create_definition. Cu opțiunile FIRST sau AFTER se poate muta coloana specificată în prima poziție respectiv după coloana specificată în column_name.

Ex:

```
ALTER TABLE student MODIFY nume varchar(10) FIRST
```

Comanda de mai sus modifică tipul câmpului nume pe prima poziție din tabel și schimbă tipul acesteia în șir de caractere cu lungimea de 10.

- opțiunea **DROP [COLUMN] col_name** – cauzează ștergerea coloanei specificată în col_name din tabelul tbl_nume.

Ex:

```
ALTER TABLE student DROP nume;
```

Comanda de mai sus șterge coloana nume din tabelul student.

- opțiunea **RENAME [TO] new_tbl_name** – cauzează redenumirea tabelului `tbl_nume` cu numele `new_tbl_name`

Ex:

```
ALTER TABLE student RENAME student_new
```

Comanda de mai sus modifică numele tabelului `student` în `student_new`

4. Instrucțiunea DROP TABLE.

Formatul general al acestei instrucțiuni este:

```
DROP TABLE [IF EXISTS] tbl_name [, tbl_name,...]
```

Această instrucțiune șterge unul sau mai multe tabele specificate în `tbl_name`.

Clauza `IF EXISTS` asociată acestei instrucțiuni permite ocolirea mesajului de eroare în situația în care `tbl_name` nu există în baza de date curentă.

Ex:

```
DROP TABLE IF EXISTS student1, student2, student3;
```

Comanda de mai sus șterge (dacă există în baza de date curentă) tabelele `student1`, `student2` și `student3`.

5. Instrucțiunea RENAME TABLE.

Formatul general al acestei instrucțiuni este:

```
RENAME TABLE tbl_name TO new_tbl_name[, tbl_name2 TO new_tbl_name2,...]
```

Această instrucțiune redenumeste tabelul `tbl_name` în `new_tbl_name`. Spre deosebire de comanda `alter` cu opțiunea `rename` care face o copie a tabelului înaintea executării instrucțiunii, această instrucțiune modifică direct numele tabelului cu noul nume.

Ex:

```
RENAME TABLE student_new TO student;
```

Comanda de mai sus redenumeste tabelul `student_new` cu numele `student`.

6. Instrucțiunea SHOW TABLES

Formatul general al acestei instrucțiuni este simplu `SHOW TABLES` fără nici un argument. Efectul acestei instrucțiuni este acela de a afișa toate tabelele create din baza de date curentă.

Teme de laborator

1. Să se creeze o bază de date pentru Facultatea de Automatică ce va cuprinde următoarele informații:

- a. informații despre fiecare student în parte,
- b. repartizarea pe grupe și subgrupe

Observație: Pentru declararea tipurilor de date din tabele se vor folosi următoarele:

int - pentru valori întregi

char(1) – pentru valori de tip caracter

varchar(marime) – pentru șiruri de caractere cu lungimea “marime”

date – pentru valori de tip dată calendaristică

După crearea tabelelor ce vor construi baza de date, se va alege la întâmplare un tabel în care:

- a. se va adăuga un câmp cu numele **camp_nou** de tipul întreg nenul care să fie penultima coloană din tabel cu valoare implicită 4;
- b. se va adăuga un index cu numele index1 la tabelul ales pentru coloana camp_nou;
- c. se va șterge valoarea default a câmpului nou creat;
- d. se va schimba coloana creată pentru a fi pe ultima poziție din tabel;
- e. se va redenumi coloana camp_nou cu camp_redenumit iar după redenumire se va schimba tipul în șir de caractere de 10;
- f. se va șterge penultima coloană din tabel;
- g. se va crea un nou tabel cu structura identică cu tabelul modificat la punctele anterioare și acesta va avea numele primului + _nou;
- h. tabelului nou creat i se va adăuga o nouă coloană cu numele descriere_tabel_nou;
- i. cu ajutorul unei singure linii de comandă se vor interschimba numele celor două tabele;
- j. cu ajutorul unei singure linii de comandă se vor șterge ambele tabele.