



# SQL DDL – Limbaj de definire a datelor

---

Seminar 1



# Limbajul SQL: DDL

---

- DDL = Data Definition Language (Limbaj de definire a datelor) – conține comenzi pentru creare/modificare/ștergere bază de date, tabele, indecși, stabilire relații între tabele, constrângeri
- Exemple de comenzi DDL:
  - Pentru baze de date:

`CREATE DATABASE`

`ALTER DATABASE`

`DROP DATABASE`



# Limbajul SQL: DDL

---

- Pentru tabele:

- CREATE TABLE

- ALTER TABLE

- DROP TABLE

- Pentru indecși:

- CREATE INDEX

- ALTER INDEX

- DROP INDEX



# Limbajul SQL: DDL

---

- Instrucțiunea **CREATE DATABASE** se folosește pentru a crea o bază de date

- Sintaxa:

```
CREATE DATABASE database_name;
```

- Exemplu:

```
CREATE DATABASE World;
```

- Instrucțiunea **ALTER DATABASE** se folosește pentru a modifica o bază de date

- Exemplu de modificare a numelui unei baze de date:

```
ALTER DATABASE World  
MODIFY Name=People;
```



# Limbajul SQL: DDL

---

- Instrucțiunea **DROP DATABASE** se folosește pentru a șterge o bază de date

- Sintaxa:

```
DROP DATABASE database_name;
```

- Exemplu:

```
DROP DATABASE People;
```



# Limbajul SQL: DDL

---

- O bază de date conține cel puțin un tabel identificat prin nume
- Tabelele conțin înregistrări cu date
- Exemplu:

Id	Nume	Prenume	Localitate
1	Pop	Oana	Sibiu
2	Porumb	Sebastian	Oradea
3	Georgescu	Ana	București

- Un tabel cu patru coloane și trei înregistrări



# Limbajul SQL: DDL

---

- Instrucțiunea **CREATE TABLE** se folosește pentru a crea un tabel într-o bază de date
- Sintaxa:

```
CREATE TABLE table_name  
(  
    column_name1 data_type,  
    column_name2 data_type,  
    ...  
);
```





# Limbajul SQL: DDL

---

- Dorim să creăm un tabel numit *Persoane* care conține câmpurile *id*, *nume*, *prenume*, *localitate*

```
CREATE TABLE Persoane
(
    id INT,
    nume VARCHAR(30),
    prenume VARCHAR(30),
    localitate VARCHAR(30)
);
```





# Limbajul SQL: DDL

---

- Instrucțiunea **ALTER TABLE** se folosește pentru a modifica structura unui tabel
- Sintaxa instrucțiunii pentru adăugarea unei coloane într-un tabel:

```
ALTER TABLE table_name  
ADD column_name datatype;
```

- Exemplu de adăugare a unei coloane într-un tabel:

```
ALTER TABLE Persoane  
ADD data_nașterii DATE;
```



# Limbajul SQL: DDL

---

- Sintaxa instrucțiunii pentru schimbarea tipului de date al unei coloane dintr-un tabel:

```
ALTER TABLE table_name
```

```
ALTER COLUMN column_name datatype;
```

- Exemplu de schimbare a tipului de date al unei coloane dintr-un tabel:

```
ALTER TABLE Persoane
```

```
ALTER COLUMN data_nașterii DATETIME;
```



# Limbajul SQL: DDL

---

- Sintaxa instrucțiunii pentru ștergerea unei coloane dintr-un tabel:

```
ALTER TABLE table_name  
DROP COLUMN column_name;
```

- Exemplu de ștergere a unei coloane dintr-un tabel:

```
ALTER TABLE Persoane  
DROP COLUMN data_nașterii;
```



# Limbajul SQL: DDL

---

- Instrucțiunea **DROP TABLE** se folosește pentru a șterge un tabel dintr-o bază de date

- Sintaxa:

```
DROP TABLE table_name;
```

- Exemplu:

```
DROP TABLE Persoane;
```



# Limbajul SQL: DDL

---

- În limbajul SQL fiecare coloană, variabilă locală, expresie sau parametru are un tip de date
- Un tip de date este un atribut care specifică ce fel de valori pot fi stocate în obiectul respectiv
- Exemple:  
int, tinyint, smallint, bigint, decimal, float, real, money, nchar, varchar, datetime, date, time



# Limbajul SQL: DDL

---

- **Constrângerile** se folosesc pentru a asigura integritatea datelor pe care le introducem într-un tabel
- Integritatea datelor se poate asigura în mod declarativ, ca parte din definiția tabelului sau în mod procedural prin proceduri stocate sau triggers
- Constrângerile se pot specifica la crearea tabelului (în instrucțiunea CREATE TABLE), dar și după ce tabelul a fost creat (cu ajutorul instrucțiunii ALTER TABLE)



# Limbajul SQL: DDL

---

– Constrângeri:

NOT NULL

UNIQUE

PRIMARY KEY

FOREIGN KEY

CHECK

DEFAULT





# Limbajul SQL: DDL

---

- În mod implicit un tabel permite inserarea de valori NULL
- Dacă nu dorim să permitem introducerea de valori NULL pentru o coloană, aplicăm **constrângerea NOT NULL** pe coloana respectivă
- Ca rezultat, nu vom putea insera sau actualiza înregistrări care nu specifică o valoare pentru coloana respectivă



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri NOT NULL la crearea unui tabel:

```
CREATE TABLE Studenți  
(  
    cod_s INT NOT NULL,  
    nume VARCHAR(50),  
    prenume VARCHAR(50),  
    oraș VARCHAR(50)  
);
```



# Limbajul SQL: DDL

---

- **Constrângerea UNIQUE** se definește pe coloanele în care nu dorim să permitem valori duplicate
- Se pot defini mai multe constrângeri **UNIQUE** în același tabel
- Se poate defini pe una sau mai multe coloane
- În cazul în care o constrângere **UNIQUE** este definită pe mai multe coloane, combinația de valori din coloanele respective trebuie să fie unică la nivel de înregistrare
- Definirea unei constrângeri **UNIQUE** creează în mod automat un index **UNIQUE** corespunzător



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri UNIQUE pe o coloană la crearea unui tabel:

```
CREATE TABLE Studenți  
(  
    cod_s INT UNIQUE,  
    nume VARCHAR(50),  
    prenume VARCHAR(50),  
    oraș VARCHAR(50)  
);
```



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri UNIQUE pe mai multe coloane la crearea unui tabel:

```
CREATE TABLE Studenți  
(  
    cod_s INT NOT NULL,  
    nume VARCHAR(50),  
    prenume VARCHAR(50),  
    oraș VARCHAR(50),  
    CONSTRAINT uc_StudentID UNIQUE (cod_s, nume)  
);
```



# Limbajul SQL: DDL

---

- Definirea unei constrângeri UNIQUE după ce tabelul a fost creat se face cu ajutorul instrucțiunii ALTER TABLE

- Exemplu de definire a unei constrângeri UNIQUE pe o singură coloană:

```
ALTER TABLE Studenți  
ADD UNIQUE(cod_s);
```

- Exemplu de definire a unei constrângeri UNIQUE pe mai multe coloane:

```
ALTER TABLE Studenți  
ADD CONSTRAINT uc_StudentID UNIQUE(cod_s, nume);
```



# Limbajul SQL: DDL

---

- O constrângere poate fi eliminată cu ajutorul instrucțiunii **DROP CONSTRAINT**

- Sintaxa:

```
ALTER TABLE table_name  
DROP CONSTRAINT constraint_name;
```

- Exemplu:

```
ALTER TABLE Studenți  
DROP CONSTRAINT uc_StudentID;
```





# Limbajul SQL: DDL

---

- Fiecare tabel trebuie să aibă o singură **cheie primară**
- **Cheia primară** identifică în mod unic fiecare înregistrare din tabel
- Nu permite introducerea valorilor duplicate sau NULL în coloana pe care este definită
- Poate fi definită pe o singură coloană sau pe o combinație de coloane
- În cazul în care este definită pe mai multe coloane, combinația de valori din acele coloane trebuie să fie unică
- Se poate defini **o singură constrângere de tip cheie primară** (primary key) într-un tabel



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri PRIMARY KEY la crearea unui tabel:

```
CREATE TABLE Studenți  
(  
    cod_s INT PRIMARY KEY,  
    nume VARCHAR(50),  
    prenume VARCHAR(50),  
    oraș VARCHAR(50)  
);
```



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri PRIMARY KEY pe mai multe coloane la crearea unui tabel:

```
CREATE TABLE Studenți  
(  
    cod_s INT,  
    nume VARCHAR(30),  
    prenume VARCHAR(50),  
    oraș VARCHAR(50),  
    CONSTRAINT pk_Student PRIMARY KEY (cod_s, nume)  
);
```



# Limbajul SQL: DDL

---

- Pentru a putea crea o cheie primară după crearea tabelului, coloana sau coloanele pe care dorim să le includem în cheia primară trebuie să aibă definită o constrângere NOT NULL

- Exemplu de definire a unei constrângeri PRIMARY KEY după crearea tabelului:

```
ALTER TABLE Studenți
```

```
ADD CONSTRAINT pk_Student PRIMARY KEY(cod_s, nume);
```

- Exemplu de eliminare a unei constrângeri PRIMARY KEY:

```
ALTER TABLE Studenți
```

```
DROP CONSTRAINT pk_Student;
```

# Limbajul SQL: DDL

- Un foreign key (cheie străină) pointează la un primary key (cheie primară) dintr-un alt tabel

- Tabelul Clienți

IDClient	Nume	Prenume	Localitate
1	Pop	Oana	Cluj-Napoca
2	Rus	Andrei	Sibiu

- Tabelul Comenzi

IDCom	NrCom	IDClient
1	3455	2
2	3456	1



# Limbajul SQL: DDL

---

- Coloana *IDClient* din tabelul *Comenzi* pointează spre coloana *IDClient* din tabelul *Clienți*
- Coloana *IDClient* din tabelul *Comenzi* este **FOREIGN KEY**, iar coloana *IDClient* din tabelul *Clienți* este PRIMARY KEY
- Constrângerea **FOREIGN KEY** este folosită pentru a preveni acțiuni care ar distruge legăturile dintre cele două tabele, dar și pentru a împiedica introducerea unor date invalide care nu se regăsesc în coloana care este PRIMARY KEY
- Nu se pot face modificări în tabelul care conține cheia primară (primary key) dacă aceste modificări distrug legături spre date din tabelul care conține cheia străină (foreign key)



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri FOREIGN KEY la crearea unui tabel:

```
CREATE TABLE Comenzi  
(  
  IDCom INT PRIMARY KEY,  
  NrCom INT,  
  IDClient INT FOREIGN KEY REFERENCES Clienți(IDClient)  
);
```





# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri FOREIGN KEY cu numele *fk\_Client* la crearea unui tabel:

```
CREATE TABLE Comenzi  
(  
  IDCom INT PRIMARY KEY,  
  NrCom INT,  
  IDClient INT,  
  CONSTRAINT fk_Client FOREIGN KEY (IDClient) REFERENCES  
  Clienți(IDClient)  
);
```



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri FOREIGN KEY după crearea tabelului:

```
ALTER TABLE Comenzi  
ADD FOREIGN KEY (IDClient)  
REFERENCES Clienți(IDClient);
```

SAU

```
ALTER TABLE Comenzi  
ADD CONSTRAINT fk_Client FOREIGN KEY (IDClient)  
REFERENCES Clienți(IDClient);
```



# Limbajul SQL: DDL

---

- Se pot specifica acțiuni care vor fi efectuate în cazul în care un utilizator încearcă să șteargă sau să modifice un key spre care pointează un foreign key
- Următoarele acțiuni pot fi specificate în acest caz:

NO ACTION

CASCADE

SET NULL

SET DEFAULT



# Limbajul SQL: DDL

---

- NO ACTION – motorul bazei de date afișează o eroare și actualizarea sau ștergerea eșuează
- CASCADE – se șterge sau se actualizează înregistrarea din tabelul care conține cheia referită împreună cu înregistrările corespunzătoare din tabelul care conține foreign key-ul
- SET NULL – se va seta valoarea null pentru toate valorile care alcătuiesc foreign key-ul atunci când înregistrarea corespunzătoare din tabelul care conține cheia referită este actualizată sau ștearsă
- SET DEFAULT – toate valorile care alcătuiesc foreign key-ul sunt setate pe valoarea default (cu condiția să fie definite valori default pe coloana sau coloanele respective) atunci când înregistrarea corespunzătoare din tabelul care conține cheia referită este actualizată sau ștearsă



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri FOREIGN KEY cu acțiuni care au loc în caz de modificare sau ștergere:

```
CREATE TABLE Comenzi  
(  
  IDCom INT PRIMARY KEY,  
  NrCom INT,  
  IDClient INT FOREIGN KEY REFERENCES Clienti(IDClient)  
  ON DELETE CASCADE  
  ON UPDATE CASCADE  
) ;
```



# Limbajul SQL: DDL

---

- **Constrângerea CHECK** se folosește pentru a limita intervalul de valori ce se pot introduce pentru o anumită coloană
- Se poate defini pe o coloană, iar în acest caz limitează valorile ce pot fi introduse pentru coloana respectivă
- Se poate defini pe mai multe coloane



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri CHECK pe o coloană la crearea tabelului:

```
CREATE TABLE Clienți  
(  
    IDClient INT PRIMARY KEY CHECK(IDClient>0),  
    Nume VARCHAR(50) NOT NULL,  
    Prenume VARCHAR(50),  
    Localitate VARCHAR(50)  
);
```





# Limbajul SQL: DDL

---

- Exemplu de constrângere CHECK definită pe mai multe coloane la crearea unui tabel:

```
CREATE TABLE Clienți
(
    IDClient INT PRIMARY KEY,
    Nume VARCHAR(50) NOT NULL,
    Prenume VARCHAR(50),
    Localitate VARCHAR(50),
    CONSTRAINT ck_IDClient CHECK(IDClient>0 AND
    Localitate IN ('Cluj-Napoca', 'Sibiu'))
);
```



# Limbajul SQL: DDL

---

- Exemplu de adăugare a unei constrângeri CHECK după crearea tabelului:

```
ALTER TABLE Clienți  
ADD CHECK (IDClient>0);
```

- Exemplu de adăugare și stabilire a unui nume pentru o constrângere CHECK după crearea tabelului:

```
ALTER TABLE Clienți  
ADD CONSTRAINT ck_Client  
CHECK (IDClient>0 AND Localitate IN ('Cluj-Napoca',  
'Sibiu'));
```



# Limbajul SQL: DDL

---

- **Constrângerea DEFAULT** se folosește pentru a insera o valoare implicită într-o coloană
- Valoarea implicită va fi adăugată pentru toate înregistrările noi dacă nu se specifică o altă valoare
- Se poate folosi și pentru a insera valori sistem obținute prin apelul unor funcții
- Exemplu de definire a unei constrângeri DEFAULT după crearea unui tabel:

```
ALTER TABLE Clienți
```

```
ADD CONSTRAINT d_Localitate DEFAULT 'Cluj-Napoca' FOR  
Localitate;
```

- Eliminarea unei constrângeri DEFAULT

```
ALTER TABLE Clienți
```

```
DROP CONSTRAINT d_Localitate;
```



# Limbajul SQL: DDL

---

- Exemplu de definire a unei constrângeri DEFAULT la crearea unui tabel:

```
CREATE TABLE Comenzi  
(  
  IDCom INT PRIMARY KEY,  
  NrCom INT NOT NULL,  
  IDClient INT,  
  DataCom DATE DEFAULT GETDATE()  
);
```