LIMBAJUL SQL

1. GENERALITĂȚI

Limbajul structurat de interogare SQL (*Structured Query Language*) este limbajul standard pentru bazele de date (BD) relaționale definit de ANSI în 1986 şi adoptat ulterior ca standard internațional de către ISO (1992). Peste o sută de sisteme de gestiune a bazelor de date (SGBD) acceptă recunosc limbajul SQL.

Ca orice limbaj de baze de date, SQL permite:

- Crearea bazei de date relaționale şi structurarea relațiilor prin componenta sa de definire a datelor (DDL)
- Efectuarea operațiilor elementare asupra BD (inserare, ştergere, modificare a datelor) şi a interogărilor asupra BD, prin componenta de manipulare a datelor (DML)

Limbajul SQL conține **comenzi de definire şi regăsire a datelor** (CREATE TABLE, SELECT TABLE, DELETE, INSERT etc.) dar nu conține instrucțiuni pentru controlul fluxului datelor (IF ... THEN ... ELSE, GO TO, DO etc.).

SQL este un limbaj **neprocedural** cu format liber deoarece precizează ce rezultate sunt necesare şi nu procedura prin care se obțin acestea.

Prima implementare comercială a unui SGBD relațional bazat pe SQL a fost realizată de corporația ORACLE. Ulterior au apărut sute de produse de BD bazate pe SQL şi dialecte ale acestuia.

SQL ca standard pentru BD a fost inclus ca şi componentă în arhitecturile de aplicații de BD dezvoltate de marile firme producătoare de soft, cum este IBM, şi adopta pentru prelucrarea informațiilor în sistemul federal al SUA.

Grupul de acces SQL lucrează pentru realizarea interoperabilității dintre limbajul SQL şi alte SGBD disparate.

Îmbunătățirea SQL se face prin definirea de noi componente precum standardul de acces la BD de la distanță (RDA – *Remote Data Access*) şi sistemul de dicționare de resurse informaționale (IRDS – *Information Resource Dictionarz System*). Alte îmbunătățiri includ suportul pentru BD distribuite în rețea, programarea orientată-obiect şi acceptarea extensiilor definite de către utilizator.

2. REGULI SQL

Limbajul SQL foloseşte termenii de tabele, coloane şi rânduri în locul celor de relații, atribute şi înregistrări.

O instrucțiune SQL include cuvinte rezervate şi cuvinte definite de utilizator pentru a denumi tabelele, atributele, indexurile etc., nefiind sensibilă la formatul literei (majusculă sau literă mică). Totuşi SQL devine sensibil la formatul literelor (*case sensitive*) în cazul înregistrărilor din BD, adică două înregistrări "CLIENT" şi "Client" sunt tratate distinct.

Regulile sau convențiile adoptate în mod universal pentru scrierea instrucțiunilor SQL pot fi sintetizate astfel:

a. Fiecare clauză a unei instrucțiuni trebuie scrisă pe o linie nouă, cu separare prin virgulă.
b. Dacă o clauză are mai multe părți, fiecare dintre acestea este scrisă pe o linie nouă şi indentată pentru a indica relația cu linia superioară.
c. Majusculele sunt utilizate pentru cuvintele rezervate (SELECT, INSERT, DELETE, GRANT etc.).
d. Literele mici sunt folosite pentru termenii proprii utilizatorului (denumiri de tabele, coloane, câmpuri etc.).
e. Bara verticală | semnifică operația logică „SAU" adică alegerea dintre două sau mai multe opțiuni.
f. Acoladele indică un element necesar.
g. Parantezele pătrate indică un element opțional.
h. Punctele de suspensie (...) specifică o repetare opțională a unui articol din BD, de zero sau de mai multe ori.
i. În practică, se creează mai întâi structura BD prin definirea tabelelor şi stabilirea formatului datelor, precum şi definirea drepturilor de acces ale utilizatorilor, după care se trece la popularea BD.
j. Fiecare instrucțiune se încheie prin caracterul "punct şi virgulă".
k. Valorile, în general, sunt scrise între paranteze rotunde, separate prin virgule.
l. Valorile literale sunt încadrate de apostroafe.

În instrucțiunile SQL apar diverşi **identificatori** care respectă următoarele constrângeri:

a. Lucrează cu setul de caractere prestabilit de ISO: litere mari (A … Z), litere mici (a … z), cifre (0 … 9) şi liniuța de subliniere (_ *underscore*).
b. Să nu depăşească lungimea maximă impusă (tipic, 128 de caractere)

c. Să înceapă cu o literă şi nu cu alt caracter (cifră, alt semn etc.)

d. Să nu conțină spații libere.

3. TIPURI DE DATE SQL

ISO defineşte **cinci tipuri de date scalare**:

I. caracter

CHAR, VARCHAR [lungime]

II. bit

BIT, BIT VARYING [lungime]

III. numeric:

exact:

NUMERIC [precizie [, scala]]

DECIMAL sau DEC [precizie [, scala]]

INTEGER sau INT

SMALL INTEGER sau SMALLINT

aproximativ:

FLOAT [precizia]

REAL

DOUBLE PRECISSION

IV. data şi ora

DATE

TIME [precizie_oră] [WITH TIME ZONE]

TIMESTAMP [precizie_oră] [WITH TIME ZONE]

V. interval

INTERVAL{{câmp_de_start TO câmp_final} câmp_data_ora }

Parametrii unui tip de date se scriu între paranteze rotunde după cuvântul care îl defineşte.

Precizia se exprimă ca număr de cifre din mantisă (partea întreagă). Scala se exprimă ca număr de cifre din exponent (număr de zecimale). Numărul de cifre din câmpul principal este separat prin virgulă de numărul de zecimale prin care se exprimă o valoare.

Se poate impune şi condiția ca obligatoriu un câmp să fie completat, folosind termenul-cheie NOT NULL. Este cazul cheii primare dintr-o relație sau a unei chei alternative.

Câmpurile din instrucțiunea INTERVAL sunt de forma:

YEAR|MONTH|DAY|HOUR|MINUTE [precizie]

Exemplu:

Instrucțiunea SQL:

INTERVAL YEAR(1) TO MONTH

semnifică un interval de la 0 ani şi 0 luni (timpul prezent) la 9 ani şi 11 luni. Numărul de ani poate fi scris în acest caz cu o singură cifră.

4. INSTRUCȚIUNI SQL DE DEFINIRE A BAZEI DE DATE

Instrțiunile de creare, modificare şi distrugere a structurilor din BD sunt următoarele:

CREATE DATABASE		DROP DATABASE
CREATE TABLE	ALTER TABLE	DROP TABLE
CREATE DOMAIN	ALTER DOMAIN	DROP DOMAIN
CREATE SCHEMA		DROP SHEMA
CREATE VIEW		DROP VIEW
CREATE INDEX		DROP INDEX

Formatul de bază dat de ISO al instrucțiunii de creare a unui tabel în BD este următorul:

```
CREATE TABLE nume_tabel
    {(nume_coloană tip_de_date [NOT NULL] [UNIQUE]
    [DEFAULT opțiune_prestabilită] [CHECK (condiție)] [, ...]
    [PRIMARY KEY (listă_de_coloane),]
    {[UNIQUE (listă_de_coloane),] [,...]}
    {[FOREIGN KEY (listă_de_coloane_chei_străine)
    REFERENCES nume_tabel_părinte [(listă_de_coloane_chei_candidat)],
        [MATCH { PARTIAL | FULL}
        [ON UPDATE acțiune_referențială]
        [ON DELETE acțiune_referențială]}
    {[CHECK (condiție)] [,…]})}
```

EXEMPLU:

```
CREATE TABLE agenti(
    cod_agent    DEC(3,0)        NOT NULL        UNIQUE,
    nume         VARCHAR(20)     NOT NULL,
```

```
    prenume     VARCHAR(20)     NOT NULL,
    cnp         DEC(13,0)       NOT NULL,
    filiala     VARCHAR(10),
    salariu         DEC(5,2),
    vechime     SMALLINT        DEFAULT 0
    PRIMARY KEY (cod_agent));

CREATE TABLE proprietati(
    cod_proprietate     DEC(5,0)        NOT NULL    UNIQUE,
    zona                VARCHAR(20)     NOT NULL,
    tip                 VARCHAR(10)     NOT NULL,
    pret                INT,
    cod_agent           DEC(3,0),
    cod_proprietar      DEC(6,0)        NOT NULL,
    PRIMARY KEY (cod_proprietate),
    FOREIGN KEY (cod_agent) REFERENCES agenti ON DELETE SET NULL ON
    UPDATE CASCADE,
    FOREIGN KEY (cod_proprietar) REFERENCES proprietari);
```

Observații:

1. Tipul datelor poate fi declarat separat sub forma unui domeniu de valori şi utilizat pentru mai multe variabile de acelaşi tip:

```
CREATE DOMAIN  nume_domeniu AS tip_de_date
[DEFAULT opțiune_prestabilită]
[CHECK (condiție)];
```

La crearea tabelului se specifică în locul tipului datelor, numele domeniului scris cu majuscule.

2. **Instrucțiunea DROP** elimină articole din BD şi poate avea două opțiuni:

```
DROP ARTICOL nume_articol [RESTRICT|CASCADE]
```

Opțiunea RESTRICT nu va permite ştergerea articolului dacă de acesta depind alte date din BD. Se evită astfel pierderea de date.

Opțiunea CASCADE este una extremă care determină ştergerea acelui articol din BD precum şi a tuturor datelor care depindeau de acesta. Este utilă pentru actualizarea structurii BD după o perioadă mai lungă de timp sau atunci când se reproiectează aceasta.

3. **Indexul** este o structură care oferă acces accelerat la înregistrările din BD pe baza valorilor dintr-una sau mai multe coloane ale unui tabel, îmbunătățind astfel performanțele de interogare:

```
CREATE [UNIQUE] INDEX nume_index
ON nume_tabel (coloana [ASC|DESC] [, …])
```

Utilizarea indexurilor trebuie făcută cu oarecare rezerve întrucât solicită mai multe resurse din partea serverului de BD.

4. Pentru modificarea structurii de coloane a unui tabel deja creat, se foloseşte instrucțiunea:

```
ALTER TABLE nume_tabel
  [ADD [COLUMN] nume_coloană tip_de_date [NOT NULL] [UNIQUE]
    [DEFAULT opțiune_prestabilită] [CHECK (condiție)]
  [DROP  [COLUMN] nume_coloană [RESTRICT | CASCADE]]
  [ADD [CONSTRAINT [nume_constrângere]] definiție_constrângere]
  [DROP CONSTRAINT nume_constrângere [RESTRICT | CASCADE]]
  [ALTER [COLUMN] SET DEFAULT opțiune_ prestabilită]
  [ALTER [COLUMN] DROP DEFAULT]
```

5. INSTRUCȚIUNI SQL DE MANIPULARE A BAZEI DE DATE

Instrucțiunile de manipulare a datelor din BD sunt următoarele:

SELECT	pentru interogarea BD;
INSERT	pentru introducere de noi înregistrări în BD;
UPDATE	pentru reactualizarea BD;
DELETE	pentru ştergerea de înregistrări din BD.

Fiecare dintre aceste instrucțiuni conține după cuvântul-cheie de definiție diverse clauze cu multiple opțiuni. De aceea le vom studia pe fiecare în parte.

5.1 INSTRUCȚIUNEA SELECT

Instrucțiunea SELECT de interogare a BD are forma următoare:

```
SELECT  [DISTINCT|ALL] {*|expresie_coloana [AS nume_nou]] [, …]}
FROM nume_tabel_sau_vedere [alias] [,…]
```

[WHERE condiție]

[GROUP BY lista_de_coloane] [HAVING condiție]

[ORDER BY lista_de coloane];

Exemplu:

SELECT *

FROM agenti

WHERE filiala = centru;

Rezultatul acestei comenzi SQL va fi afişarea tuturor înregistrărilor din tabelul „agenti" corespunzătoare filialei „centru".

Cuvântul-cheie DISTINCT elimină în cadrul interogării eventualele dubluri din BD.

În **expresiile** incluse în comenzile SQL se folosesc **operatori scalari şi funcții specifice**:

- operatorii aritmetici: +,-,*,/
- funcția de lungime: BIT_LENGTH, OCTET_LENGTH, CHAR_LENGTH
- operatorul de transformare a tipului de date: CAST(tip1 AS tip2)
- concatenarea de şiruri: ||
- identificarea utilizatorului curent: USER
- identificarea sesiunii: SESSION_USER
- identificarea sistemului: SYSTEM_USER
- scrierea cu litere mici: LOWER
- scrierea cu majuscule: UPPER
- data sau timp curent: CURRENT_TIME,CURRENT_DATE, CURRENT_TIMESTAMP.

Limbajul SQL mai acceptă cinci **funcții de grup** care se aplică pe o coloană întreagă şi generează o singură valoare:

- COUNT numărare
- SUM sumare
- AVG calculul valorii medii
- MIN deducerea valorii minime
- MAX deducerea valorii maxime.

Opțiunea GROUP BY permite gruparea înregistrărilor cu aceleaşi valori pe coloanele precizate în lista de coloane şi aplicarea funcțiilor de grup pe aceste grupuri.

O instrucțiune SELECT integrată în altă interoare SELECT într-o clauză WHERE sau HAVING se numeşte **subinterogare.**

Pentru a selecta date din mai multe tabele, în cadrul clauzelor WHERE şi HAVING, se folosesc **operațiile cu mulțimi**: UNION, INTERSECT, EXCEPT.

Clauza WHERE poate fi urmată de unul din următoarele 5 **predicate**:

- **Compararea valorilor** folosind **operatorii de comparare** (=, <, >, <=, >=, <> ISO, !=) şi/sau operatorii logici (AND, OR, NOT);
- **Testarea domeniului de valori** al unei expresii (BETWEEN/NOT BETWEEN);
- **Testarea apartenenței la o mulțime de valori** (IN/ NOT IN);
- **Corespondența la un anumit model** (LIKE/NOT LIKE);
- **Testarea condiției de null** (IS NULL/IS NOT NULL)**.**

Testele LIKE/NOT LIKE folosesc simbolul "procent" (%) pentru reprezentarea unui şir de zero sau mai multe caractere şi caracterul "liniuță de subliniere" pentru reprezentarea oricărui caracter singular.

Exemplu: Clauza:

WHERE nume='A%'

selectează toate înregistrările al căror nume începe cu litera A.

Clauza ORDER BY poate ordona înregistrările crescător sau descrescător, alfabetic sau numeric, pe baza uneia sau a mai mutor coloane. Prima coloană constituie **cheia majoră de sortare,** iar următoarele sunt **chei minore de sortare.**

5.2 INSTRUCȚIUNEA INSERT

Instrucțiunea INSERT de introducere a uneia sau a mai multor înregistrări în BD foloseşte următoarele două formate:

I. INSERT INTO nume_tabel [(listă_de_coloane)]
VALUES (listă_de_valori);

II. INSERT INTO nume_tabel [(listă_de_coloane)]
SELECT …;

Cel de al doilea format reprezintă o **instrucțiune combinată INSERT … SELECT** deci poate folosi toate clauzele instrucțiunii SELECT pentru copierea mai multor înregistrări din tabele ale BD printr-o singură comandă SQL.

Exemple:

```
INSERT INTO agenti (nume, prenume, cnp, filiala)
VALUES ('popescu', 'marius', '1900102111111', 'iasi');
```

```
INSERT INTO personal
VALUES ('ionescu', 'max', '1900102111111', 'iasi', '0');
```

5.3 INSTRUCȚIUNEA UPDATE

Pentru actualizarea datelor din BD se foloseşte instrucțiunea UPDATE cu următorul format:

```
UPDATE nume_tabel
SET coloana_1 = valoarea_1[, coloana_2 = valoarea_2 …]
[WHERE condiție];
```

Exemplu:

Pentru majorarea cu 5% a salariilor tuturor agenților cu vechime de minimum 3 ani se scrie comanda SQL:

```
UPDATE agenti
SET salariu = salariu*1,05
WHERE vechime >= 3;
```

5.4 INSTRUCȚIUNEA DELETE

Pentru ştergerea unor înregistrări din BD se foloseşte instrucțiunea DELETE cu următorul format:

```
DELETE FROM nume_tabel
[WHERE condiție];
```

6. VEDERI

Prin definiție, o **vedere** este o relație virtuală produsă la cerere prin operații relaționale, folosind relațiile existente în baza de date.

O vedere este creată prin instrucțiunea CREATE VIEW:

```
CREATE VIEW nume_vedere [(nume_coloană [, …])]
```

AS SELECT [WITH [CASCADED|LOCAL] CHECK OPTION];

Pentru a crea vederea, utilizatorul trebuie să aibă drepturi de interogare (SELECT) asupra tuturor tabelelor implicate în subselecție şi drepturi de utilizare (USAGE) asupra tuturor coloanelor solicitate.

O vedere care restrânge accesul la înregistrările selectate dintr-unul sau mai multe tabele fără restricționarea coloanelor, se numeşte **vedere orizontală.**

O vedere verticală restrânge accesul la anumite atribute (coloane) dintr-unul sau mai multe tabele. De exemplu, salariile sunt confidențiale şi nu pot fi vizualizate de către agenți într-o vedere verticală. Acest lucru devine posibil într-o vedere orizontală care îi permite fiecăruia să citească propria înregistrare din BD.

Instrucțiunea CREATE VIEW se combină cu instrucțiunea SELECT şi cu clauzele acesteia.

Exemplu:

Managerul agentiei doreşte să cunoască proprietățile gestionate de toți agenții, fără detalii specifice legate de cnp, salariu etc. Presupunem că în BD există un tabel "agenți" şi un tabel "proprietăți":

Agenti (cnp, nume, prenume, filiala, salariu, vechime);

Proprietati (nr_proprietate, zona, tip, suprafata, pret, adresa, cod_proprietar, cnp)

Vederea este creată prin comanda:

```
CREATE VIEW agenti_proprietati
AS SELECT nr_proprietate, nume_agent, prenume_agent, filiala
        FROM agenti nume, agenti prenume, agenti filiala, proprietati nr_proprietate
        WHERE agenti.cnp = proprietati.cnp
        GROUP BY agenti.filiala;
```

O vedere este reactualizabilă dacă SGBD este capabil să urmărească orice rând sau coloană până la relația-sursă.

Vederile pot fi folosite pentru a crea noi vederi.

O vedere care face apel la mai multe tabele se numeşte **vedere unificată.**

O vedere care utilizează clauza GROUP BY se mai numeşte şi **vedere grupată.**

Distrugerea unei vederi se face prin instrucțiunea:

DROP VIEW nume_vedere [RESTRICT | CASCADE]

Opțiunea CASCADE determină ştergerea tuturor vederilor bazate pe vederea eliminată.

Înregistrările dintr-o vedere care în urma reactualizării BD sau inserării de noi date satisfac sau nu mai satisfac clauza WHERE vor intra sau vor ieşi din acea vedere, fiind numite şi **rânduri**

migratoare. Clauza WITH CHECK OPTION migrarea unui rând în afara vederii, ceea ce asigură o mai bună securitate a datelor incluse în vedere decât în tabelele BD.

7. TRANZACȚII

ISO definește un model de tranzacții bazat pe două instrucțiuni SQL: COMMIT (executare) şi ROLLBACK (revenire).

Tranzacția este o unitate logică de lucru care conține una sau mai multe comenzi SQL.

Inițierea tranzacției poate fi făcută de către o persoană sau un program printr-o comandă de inițiere de tip SELECT; INSERT; UPDATE.

Până la completarea tranzacției, efectele ei nu sunt vizibile.

Încheierea tranzacției se poate realiza în unul din următoarele 4 moduri:

a. Prin instrucțiunea COMMIT, modificările din BD sunt permanente.
b. Prin instrucțiunea ROLLBACK, se abandonează modificările inițiate şi BD rămâne nemodificată.
c. Finalizarea cu succes a programului încheie tranzacția şi modificările au efect chiar dacă nu s-a executat instrucțiunea COMMIT.
d. Abandonarea tranzacției fără rularea instrucțiunii ROLLBACK atunci când se termină anormal programul respectiv.

Formatul unei tranzacții este următorul:

```
SET TRANSACTION
        [READ ONLY |READ WRITE] |
        [ISOLATION LEVEL READ UNCOMMITTED | READ UNCOMMITTED |
                REPEATABLE READ | SERIALIZABLE];
```

8. CONTROLUL ACCESULUI

Limbajul SQL foloseşte două instrucțiuni pentru controlul accesului la BD: GRANT şi REVOKE.

Mecanismul de securitate al BD se bazează pe conceptele de:

a. identificator de autorizație
b. posesiune

c. privilegiu.

Fiecărui utilizator la BD i se alocă un identificator de autorizație, asociat cu o parolă, utilizat pentru a determina drepturile de acces ale acestuia la obiectele din BD.

Fiecare obiect din BD este proprietatea celui care l-a creat (drept de posesiune).

În clauza AUTHORIZATION din schema căreia îi aparține obiectul apare identificatorul proprietarului.

Prin privilegii se înțeleg acțiunile care îi sunt permise unui utilizator al BD:

SELECT
INSERT [(nume_coloană [, ...])]
UPDATE [(nume_coloană [, ...])]
DELETE
REFERENCES [(nume_coloană [, ...])]
USAGE

Formatul instrucțiunii GRANT este următorul:

```
GRANT {listă_de_privilegii | ALL PRIVILEGES}
ON    nume_obiect
TO    {listă_de_identificatori | PUBLIC}
[WITH GRANT OPTION]
```

Cuvântul cheie PUBLIC îi desemnează pe toți utilizatorii BD.

Clauza WITH GRANT OPTION permite transmiterea privilegiilor spre alți utilizatori.

Formatul instrucțiunii REVOKE este următorul:

```
REVOKE [GRANT OPTION FOR] {listă_de_privilegii | ALL PRIVILEGES}
ON    nume_obiect
FROM {listă_de_identificatori | PUBLIC}   [RESTRICT | CASCADE]
```

Opțiunea GRANT OPTION FOR permite retragerea separată a drepturilor acordate altor utilizatori prin clauza WITH GRANT OPTION.