Interogarea Bazelor de Date

Procesul de interogare a bazelor de date înseamnă regăsirea unui subset de date după un anumit criteriu de căutare. Interogarea bazelor de date se face cu ajutorul limbajului **SQL** – **Structured Query Language**.

Comenzile limbajului sunt:

Describe <nume tabel>; – afişează structura unui tabel

SELECT * FROM <nume tabel>; - returnează toate înregistrările dintr-un tabel

SELECT <nume coloană1, nume coloană2, etc> FROM <nume tabel> WHERE <condiție>; - returnează un anumit subset de date după condiția dată

INSERT INTO < nume tabel> VALUES (valoare1, valoare2, valoare3, etc); - inserați o înregistrare în tabel

INSERT INTO < nume tabel> (<nume coloană1, nume coloană2, etc>) VALUES (valoare 1, valoare 2, valoare 3, etc); - inserați doar anumite valori în tabel conform coloanelor specificate

ALTER TABLE <nume tabel> ADD (<nume noua coloană> <tipul de dată>); - adăugarea unei noi coloane într-un tabel

ALTER TABLE <nume tabel> DROP COLUMN <nume coloană>; - eliminarea unei coloane dintr-un tabel

DELETE from <nume tabel>; - şterge toate datele din tabel. NU ŞTERGE TABELA!

DELETE from <nume tabel> WHERE <nume coloană> = ' valoare‘; - ştergerea de date (o înregistrare) conform condiției specificate

Drop TABLE <nume tabel>; - şterge tabela

UPDATE <nume tabel> SET <nume coloană> = <valoare> WHERE <condiție>; - modifică valori în tabel după condiția dată

Pentru crearea unei tabele sintaxa poate diferi puțin în funcție de serverul de baze de date folosit.
Pentru **ORACLE** un exemplu este următorul – vezi www.oracle.com

```
CREATE TABLE my_music
        ( MUSICID NUMBER(2) NOT NULL,
        ARTIST_NAME VARCHAR2(20),
        TYPE VARCHAR2(13),
        CONSTRAINT MY_MUSIC_PRIMARY_KEY PRIMARY KEY (MUSICID));
```

Pentru **MYSQL** un exemplu este următorul – vezi dev.mysql.com

```
CREATE TABLE example(
        id INT NOT NULL AUTO_INCREMENT, PRIMARY KEY(id),
```

```
        name VARCHAR(30),  age INT)
```

Pentru **Microsoft SQL Server** – vezi http://msdn.microsoft.com

```
CREATE TABLE person
   (
     num             INT            NOT NULL  ,
     firstname       VARCHAR(20)        NULL  ,
     lastname        VARCHAR(30)        NULL  ,
     gender_code     VARCHAR(1)         NULL  ,
     birth_dttm      DATETIME           NULL  ,
     inactive_date   DATETIME           NULL  ,
     CONSTRAINT PK_Person PRIMARY KEY CLUSTERED (num ASC)
     ON [PRIMARY]
   )
```

Analiza instrucțiunii SELECT

Instrucțiunea Select este cea mai folosită deoarece cu ajutorul ei este interogată baza de date. Sintaxa comenzii SELECT este următoarea:

```
SELECT [ALL | DISTINCT]
   [<alias>.]<select_item>
      [AS <column_name>]
      [, [<alias>.]<select_item>
      [AS <column_name>] ...]
   FROM <table> [<local_alias>]
   [, <table> [<local_alias>] ...]
   [[INTO <destination>]
      | [TO FILE <file>
         [ADDITIVE]
      | TO PRINTER [PROMPT]
      | TO SCREEN]]
   [WHERE <joincondition>
   [AND <joincondition> ...]
   [AND | OR <filtercondition>
   [AND | OR <filtercondition> ...]]]
   [GROUP BY <groupcolumn>
      [, <groupcolumn> ...]]
   [HAVING <filtercondition>]
   [UNION [ALL] <SELECT command>]
   [ORDER BY <order_item>
      [ASC | DESC]
      [, <order_item>
      [ASC | DESC] ...]]
```

[ALL | DISTINCT]
Clauza SELECT specifică câmpuri, constante sau expresii ce vor fi afişate ca rezultat al interogării. Implicit toate liniile (ALL) sunt afişate. Includeți DISTINCT pentru a exclude liniile duplicate din rezultatul interogării.
DISTINCT poate fi folosit o singură dată pentru o clauza SELECT.

Exemplu:
Select * from studenti;
Select **distinct** cod_produs from vanzari;

[<alias>.]<select_item> [AS <column_name>]
[, [<alias>.]<select_item> [AS <column_name>] ...]

<select_item> poate fi unul din următoarele:

- numele unui câmp dintr-o tabelă din clauză FROM;
- o expresie ce poate fi numele unei funcţii definite de utilizator.

Orice element specificat prin <select_item> generează o coloană în rezultatul interogării. Dacă selectaţi coloane cu acelaşi nume din tabele diferite calificaţi denumirile identice incluzând aliasul tabelei urmat de '.'.
Exemplu:
Select **s**.nume, **s**.nota, nota*credite **AS** "Credite" from studenti **s** ;

Există două tipuri de funcţii pentru manipularea datelor din baza de date: **Single Row Functions** (incluzând funcţiile de conversie pentru caractere, numere si date calendaristice) si **Multiple Row Functions**(cele descrise mai jos.)

AVG(<select_item>) - Calculează media aritmetică a unei coloane de date.

COUNT(<select_item>) - Calculează numărul de articole selectate într-o coloană.

COUNT(*) – calculează nr. de linii pentru toate coloanele.

MIN(<select_item>) - Determină cea mai mică valoare pentru <select_item> într-o coloana.

MAX(<select_item>) - Similar pentru maximul valorii.

SUM(<select_item>) - Totalizează o coloana de valori numerice.

Aceste funcţii nu pot fi imbricate.

Clauza opţională AS <column_name> specifică antetul fiecarei coloane în rezultatul interogării. <column_name> poate fi şi o expresie de calculat.

FROM <table> [<local_alias>]
[, <table> [<local_alias>] ...]

Clauza FROM cuprinde tabelele din care vor fi selectate datele. <local_alias> este un nume temporar pentru tabela specificată prin <table>.

WHERE <joincondition>
[AND <joincondition> ...]
[AND | OR <filtercondition>
[AND | OR <filtercondition>...]]

WHERE este necesară atunci când sunt selectate date din mai multe tabele. Ea impune ca numai anumite linii să fie incluse în rezultatul interogării.

<joincondition> specifică câmpurile ce pun în legatură tabelele din lista clauzei FROM.

Condițiile de join multiple trebuie conectate cu AND. Orice condiție de join este de forma:
<field1> <comparison> <field2>

unde <field1> este numele unui câmp al unei tabele, <field2> este numele unui câmp din altă tabelă, iar <comparison> este unul din următorii operatori:

Operator	Comparison
=	Like
< >, !=	Not like
= =	Exactly Like
>	More than
> =	More than or equal to
<	Less than
< =	Less than or equal to

<filtercondition> specifică criteriile ce trebuie îndeplinite de liniile ce vor fi incluse în rezultatele interogării. <filtercondition> poate fi una din formele:

<field1> <comparison> <field2>
Ex: evaluare.stud_id= studenti.stud_id

<field> <comparison> <expression>
Ex: studenti.nota <= 10

<field> <comparison> ALL (<subquery>)
Ex: taxe < ALL (SELECT taxe FROM client WHERE judet = "Bacau")

<field> <comparison> ANY | SOME (<subquery>)
Ex: taxe < ANY
(SELECT taxe FROM client WHERE judet = ”Bacau”)

<field> [NOT] BETWEEN <start_range> AND <end_range>
Ex: client.rate BETWEEN 5.50 AND 6.00

[NOT] EXISTS (<subquery>)
Ex: EXISTS
(SELECT * FROM invoices WHERE customer.zip = invoices.zip)
(subinterogarea returnează true)

<field> [NOT] IN <value_set>
Ex: cod NOT IN ("98052","98072","98034")

<field> [NOT] IN (<subquery>)
Ex: nume IN (select nume from student where nota >5)

<field> [NOT] LIKE <expC>
Example: nume NOT LIKE "%A"

Sunt selectate câmpurile ce se potrivesc cu <expC>. Se pot folosi caracterele mască % sau _ ca parte a <expc>. Underscore reprezintă un singur caracter, oarecare.

O **subinterogare** este un SELECT în cadrul unui SELECT şi trebuie inclusă între paranteze. Pot fi mai multe subinterogari la acelai nivel (nu imbricate) şi trebuie incluse între paranteze. Pot conține condiții de join multiple.

GROUP BY <groupcolumn> [, <groupcolumn> ...]
Clauza GROUP BY grupează liniile în interogare pe baza valorilor din una sau mai multe coloane. <groupcolumn> poate fi numele unei coloane din clauza SELECT sau o expresie numerică indicând poziția coloanei din rezultatul interogării (numărul celei mai din stânga coloane este 1).

HAVING <filtercondition>
Clauza impune includerea în rezultatul interogării numai a acelor grupuri ce satisfac condiția specificată de <filtercondition>. <filtercondition> nu poate conține subinterogări. Ar trebuie utilizată conjugat cu GROUP BY. În absența unei clauze GROUP BY, efectul ei este cel al unei clauze WHERE.

ORDER BY <order_item> [ASC | DESC]
[, <order_item> [ASC | DESC] ...]
ORDER BY sortează rezultatele interogării pe baza datelor din una sau mai multe coloane. Orice element de ordonare trebuie să corespundă unei coloane din rezultatul interogării principale.

Exercitii:
Creați o tabelă in MAccess după structura:
a) CREATE TABLE my_music
(MUSICID NUMBER(2) NOT NULL,
ARTIST_NAME VARCHAR2(20),
TYPE VARCHAR2(13),
CONSTRAINT MY_MUSIC_PRIMARY_KEY **PRIMARY KEY** (**MUSICID**));

b) Introduceți date in tabel cu ajutorul instrucțiunii Insert din SQL – 9 înregistrări.
c) Ștergeți un artist din tabel folosind instructiunea Delete.
d) Modificați data de natere pentru artistul de pe poziția 3 in tabel.
e) Selectați
- toate datele din tabel;
- artiştii şi genul muzical;
- toți artiştii de pe primele 5 poziții;
- artiştii grupați pe genuri muzicale.

JOIN-URI

Join-urile se folosesc pentru a regăsi date din mai mult de un tabel. Sunt mai multe tipuri de join-uri: equijoin, nonequijoin, şi produs cartezian.

Equijoin sau Inner Join
- combină rîndurile care au valori egale pentru coloanele specificate;
- este denumit deseori join simplu şi este cel mai folosit

Sintaxa este:

```
Select table1.column , table2.column ...
From table1, table2
Where table1.column1 = table2.column2;
```

Exemplu:

```
Select s.nume, s.prenume, c.denumire, e.nota
From student s, cursuri c, evaluare e
Where s.student_id = e.student_id and c.curs_id=e.curs_id
Order by s.nume;

SELECT nume
FROM student INNER JOIN evaluare ON student.id = evaluare.stud_id;
```

Nonequijoin

- combină tabele care nu au coloane care se potrivesc
- acest join este folosit când valorile unei coloane în tabela A sunt într-un interval de valori specificat de 2 coloane din tabela B

Exemplu:

Fie tabelele angajati(id, nume, prenume, salariu) şi nivel_salar(nivel, salar_minim, salar_maxim)

```
Select a.nume, a.prenume, n.nivel
From angajati a, nivel_salar n
Where a.salar between n.salar_minim and n.salar_maxim
```

Produs Cartezian

- atunci când o interogare join nu specifică o condiție în clauza where rezultatul este un produs cartezian
- combină fiecare înregistrare dintr-o tabelă cu fiecare înregistrare din cealaltă tabelă

Exemplu:

```
SELECT nume, nota
FROM student, evaluare
```

Exerciții:

1. Aflați numele şi notele tuturor studenților.
2. Aflați numele profesorilor şi cursurile predate de ei.
3. Studenții, notele lor, disciplina şi profesorii.
4. Aflați nota maximă pentru fiecare disciplină.