

Capitolul 12 Actualizarea și interogarea bazelor de date

Datele din bazele de date Access pot fi extrase și manipulate (introduse, modificate și șterse) cu ajutorul interogărilor. Astfel de interogări pot fi salvate în cadrul bazei de date la secțiunea „Queries”, dar ele pot fi scrise și în cadrul aplicațiilor ce folosesc bazele de date Access.

În funcție de efectul lor, interogările Access pot fi:

- interogări de selecție (Select Query, Make Table Query, Crosstab Query);
- interogări de inserare (Append Query);
- interogări de modificare (Update Query);
- interogări de ștergere (Delete Query).

Interogările se pot defini în modul de lucru *Design View* sau în *modul de lucru asistat* (folosind instrumente de tip *Wizard*). Modul de lucru *Design View* folosește un limbaj bidimensional de tip grafic asemănător limbajului **QBE** (*Query By Example*) din gama instrumentelor **RAD** (*Rapid Application Development*). De asemenea, interogările pot fi definite prin folosirea comenzilor SQL.

Instrumentele de tip wizard și cele de tip RAD folosite în definirea interogărilor sunt potrivite pentru interogări relativ simple, în timp ce, în cazul interogărilor complexe, scrierea comenzilor SQL nu poate fi evitată.

În acest capitol prezentăm posibilitățile de filtrare, ordonare și actualizare directă a datelor, precum și instrumentele de tip Wizard și cele de tip RAD existente în Access.

12.1 Filtrarea și ordonarea directă a datelor dintr-o tabelă

Utilitatea bazelor de date este dată de posibilitatea obținerii informațiilor necesare la momentul oportun. În acest scop trebuie să găsim datele care ne

interesează. Trebuie să ne imaginăm că, în cazuri reale, datele dintr-o bază de date sunt ceva mai multe decât cele luate ca exemplu în această carte, iar efortul de căutare ar fi foarte dificil și îndelungat.

Exemplul nr. 1

Să se identifice clienții din localitatea cu codul 1002 folosindu-se filtrarea directă a datelor.

Rezolvare

Se deschide tabela Clienți dând dublu click pe ea și se dă click dreapta pe coloana Cod localitate în partea de date (pe spațiul alb) și nu pe antetul tabelului. Se obține meniul din figura nr. 12.1.1.

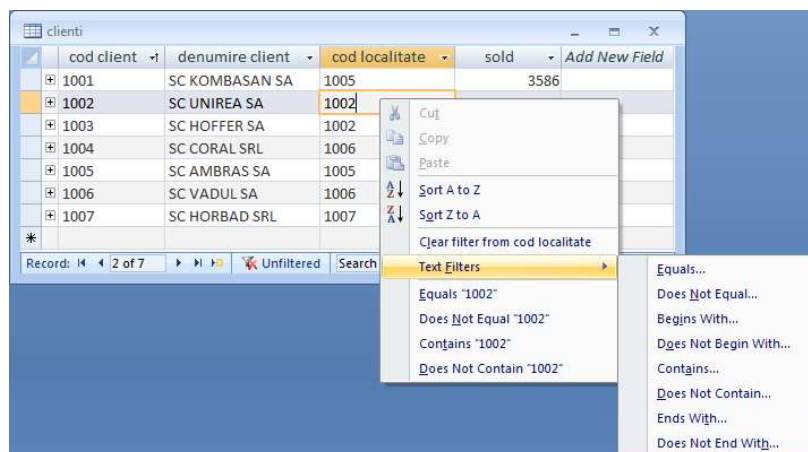


Figura nr. 12.1.1 Meniul specific filtrării directe a datelor

Pentru rezolvarea exemplului nr. 1 putem alege opțiunea *Equals „1002”*, dacă am dat click dreapta chiar pe valoarea 1002 din coloana Cod localitate. O altă variantă, indiferent de valoarea pe care am dat click dreapta este să folosim opțiunea *Text Filters* → *Equals* și să scriem în fereastra care ne apare valoarea căutată.

Ordonările și filtrările pot fi realizate și cu instrumentele din zona *Sort&Filter* din meniul *Home*. Pe lângă funcționalitățile specifice meniului contextual prezentat în figura 12.1.1, în zona *Sort&Filter* mai întâlnim opțiunea *Filter* cu aceeași funcționalitate ca în Excel.

Toate sortările și filtrările prezentate mai sus, nu afectează în niciun fel modul de stocare a datelor, ci doar vizualizarea acestora.

12.2 Actualizarea directă a datelor

După găsirea unui rând ce trebuie modificat, putem introduce direct noile valori fără să fie nevoie să salvăm explicit modificările. Mesajul care poate să apară atunci când închidem o tabelă (vezi figura nr. 12.2.1) face referire nu la salvarea datelor modificate, ci la salvarea modului de afișare a datelor dintr-o tabelă (ordinea datelor și a coloanelor).

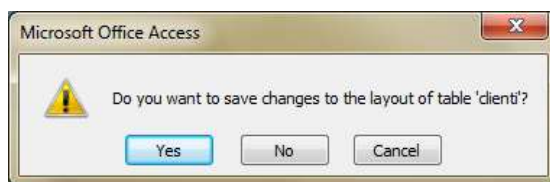


Figura nr. 12.22.1 Mesajul specific salvării modului de afișare a datelor dintr-o tabelă

Introducerea unei noi înregistrări într-o tabelă se realizează în Access prin completarea ultimului rând (înregistrarea de rezervă).

Ștergerea unui rând se poate face cu ajutorul opțiunii *Delete Record* din meniul ce apare când se dă click dreapta pe marginea din stânga a unei tabele, ca în figura nr. 12.2.2.

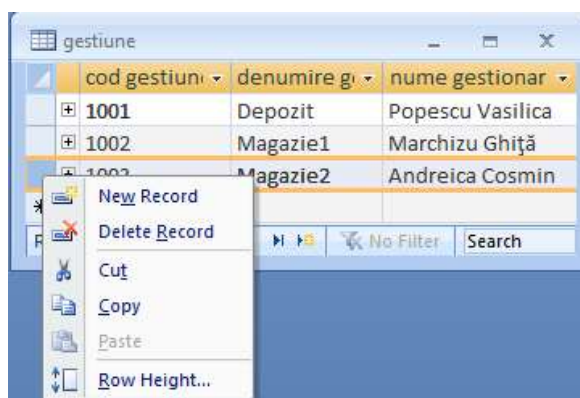


Figura nr. 12.22.2 Meniul pentru gestiunea înregistrărilor dintr-o tabelă

12.3 Crearea de interogări cu instrumente de tip Wizard

Instrumentele de tip wizard din Access dau posibilitatea realizării mai multor tipuri de interogări. Lista acestora se poate vedea folosind, din meniul *Create*, opțiunea *Query Wizard*.

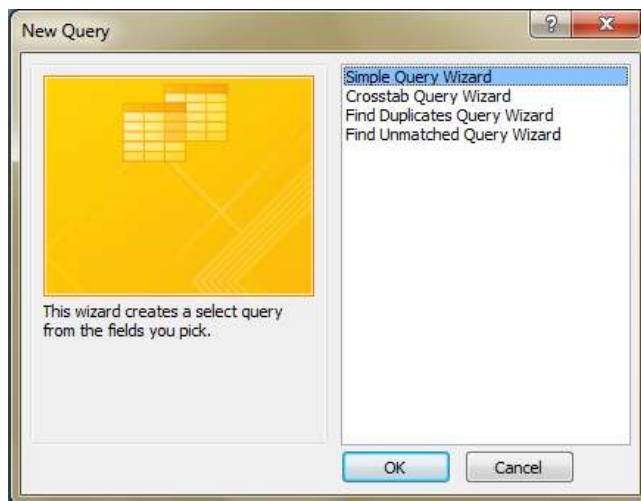


Figura nr. 12.3.1 Lista instrumentelor de tip wizard destinate interogărilor

Opțiunea *Simple Query Wizard* face referire la interogări simple, care nu permit introducerea de filtre.

Opțiunea *Crosstab Query Wizard* se folosește atunci când dorim să realizăm o agregare a datelor într-un tabel cu două dimensiuni: una specificată pentru primul rând și una specificată pentru prima coloană.

Opțiunea *Find Duplicates Query Wizard* ne ajută să identificăm rândurile ce se repetă în anumite tabele sau interogări.

Opțiunea *Find Unmatched Query Wizard* este destinată identificării înregistrărilor dintr-o tabelă care nu are înregistrări corespondente cu altă tabelă cu care se află în legătură.

12.3.1 Crearea unei interogări cu Simple Query Wizard

Cu *Simple Query Wizard* se pot crea interogări simple ce folosesc date din una sau mai multe tabele, interogări ce nu pot conține condiții.

Exemplul nr. 1, cel în care extrăgeam toți clienții din localitatea având codul 1002, nu poate fi realizat cu Simple Query Wizard deoarece presupune și filtrarea datelor în așa fel încât să se extragă doar o parte din clienți.

Exemplul nr. 2

Să se obțină o listă cu toți clienții și toate localitățile din care aceștia fac parte. Lista trebuie să conțină numele clienților și numele localităților.

Rezolvare

În prima fereastră de la *Simple Query Wizard* se aleg câmpurile care să apară în listă. Mai întâi se specifică tabelele în care pot fi găsite aceste câmpuri de la *Tables/Queries* ca în figura nr. 12.3.2.

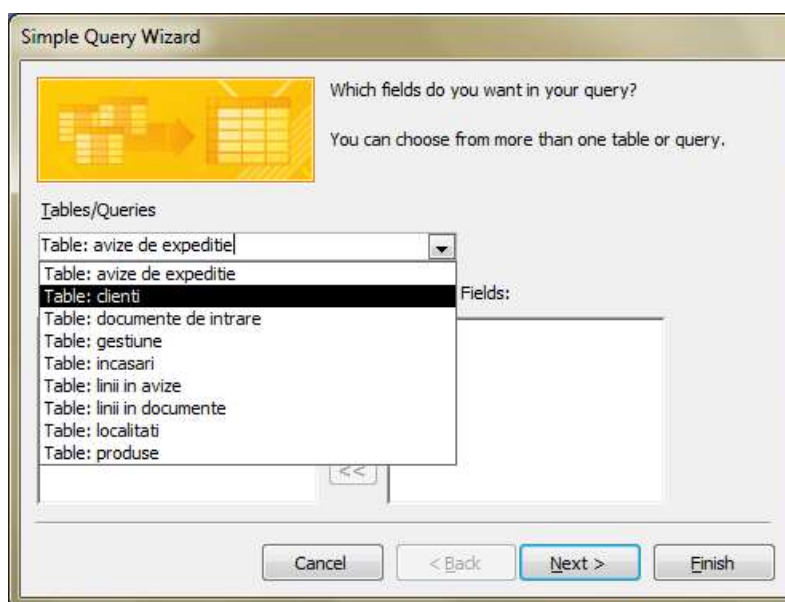


Figura nr. 12.3.2 Alegerea tabelor cu Simple Query Wizard

După alegerea unei tabele în lista *Available Fields* se găsesc toate câmpurile din tabela selectată. Cu butoanele „>”, „>>”, „<”, „<<” se includ sau se exclud în/din lista *Selected Fields* câmpurile de introdus în interogare. În cadrul exemplului nostru din tabela Clienți se alege câmpul Denumire client și din tabela Localități se alege câmpul Denumire localitate. Când creăm interogări de tip wizard cu date din mai multe tabele este important ca anterior să fi creat legăturile dintre aceste tabele, altfel vom obține un produs cartezian (pe exemplu nostru, am obține o mulțime care are un număr de rânduri egal cu numărul de clienți înmulțit cu numărul de localități, altfel spus fiecare client va apare de mai multe ori, pentru fiecare localitate în parte).

Continuând exemplul numărul 2, după ce se apasă pe butonul Next, este afișată fereastra din figura nr. 12.3.3.

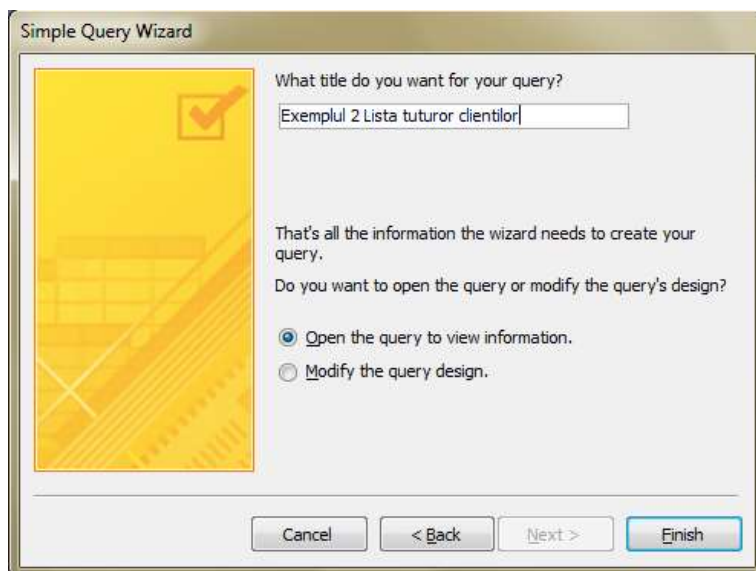


Figura nr.12.3.3 Specificarea titlului interogării

După apăsarea butonului Finish, interogarea este afișată în modul de vizualizare a rezultatelor (ca în figura nr. 12.3.4) sau este deschisă în Query Design (ca în figura nr. 12.3.5) pentru modificare.

denumire client	denumire localitate
SC UNIREA SA	Iași
SC HOFFER SA	Iași
SC KOMBASAN SA	Bârlad
SC AMBRAS SA	Bârlad
SC CORAL SRL	Galați
SC VADUL SA	Galați
SC HORBAD SRL	Tecuci

Figura nr. 12.3.4 Rezultatul interogării, exemplul nr. 2

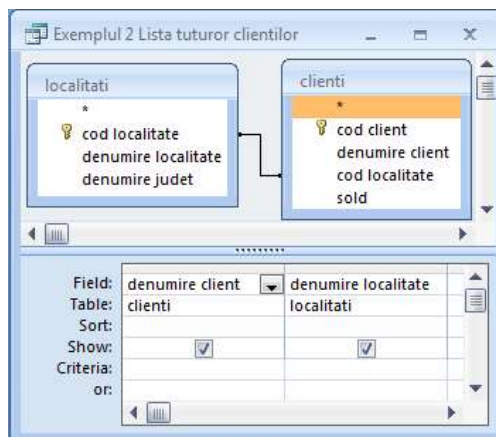


Figura nr. 12.3.5 Interogarea deschisă în Query Design

12.3.2 Crearea unei interogări cu Crosstab Query Wizard

Cu *Crosstab Query Wizard* pot fi create interogări încrucișate, ce au atât pe rânduri, cât și pe coloane valori dintr-o tabelă sau dintr-o altă interogare. Dezavantajul Crosstab Query Wizard este acela că nu se pot folosi direct date din mai multe tabele. Acest dezavantaj poate fi depășit prin realizarea unei interogări intermediare care să extragă câmpurile necesare din mai multe tabele. Crosstab Query reprezintă o tehnică specifică analizei datelor prin care se pot stabili mai

multe perspective sub care se interpretează rezultatele (de exemplu, clienți – pe linie, tipuri de documente – pe coloane).

Exemplul nr. 3

Să se creeze o listă cu încasările de la fiecare client pe tipuri de documente. Lista trebuie să conțină pe prima coloană clienții și câte o coloană pentru fiecare tip de document. La intersecția fiecărui client cu fiecare tip de document trebuie să apară suma încasărilor.

Rezolvare

Se are în vedere următoarea succesiune de pași:

- Selectarea opțiunii *Crosstab Query Wizard* din meniul *Create*, opțiunea *Query Wizard*;
- Din fereastra afișată se alege tabela sau interogarea din care se vor extrage datele (tabela Încasări, în cazul exemplului nostru), ca în figura 12.3.6.

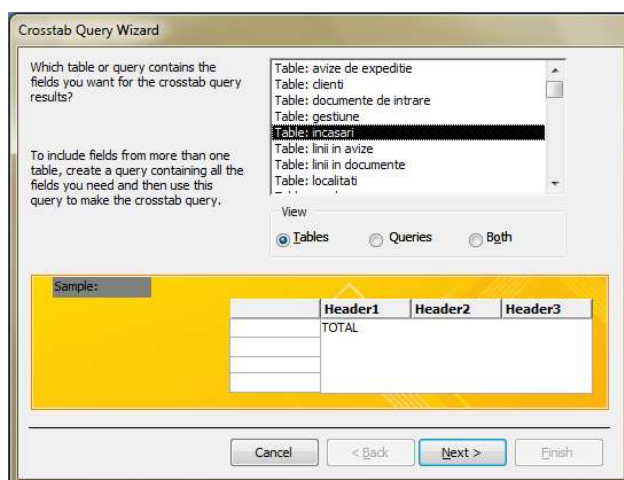


Figura nr.12.3.6 Prima fereastră de la Crosstab Query Wizard

- Selectarea câmpului sau câmpurilor care vor conferi dimensiunea verticală a datelor (câmpul Cod client din exemplu), ca în figura nr. 12.3.7.

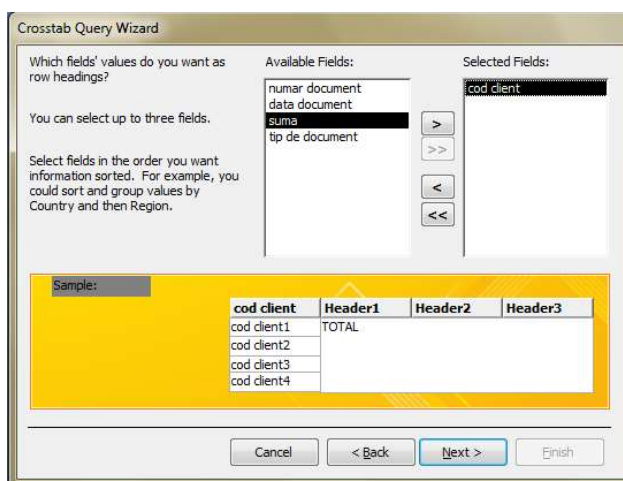


Figura nr. 12.3.7 Stabilirea dimensiunii verticale a datelor (Cod Client)

- Selectarea dimensiunii orizontale a datelor, altfel spus se alege câmpul din care se vor lua valorile cu care se vor crea noi coloane în tabelul rezultat (în exemplu câmpul Tip de document), ca în figura nr. 12.3.8.

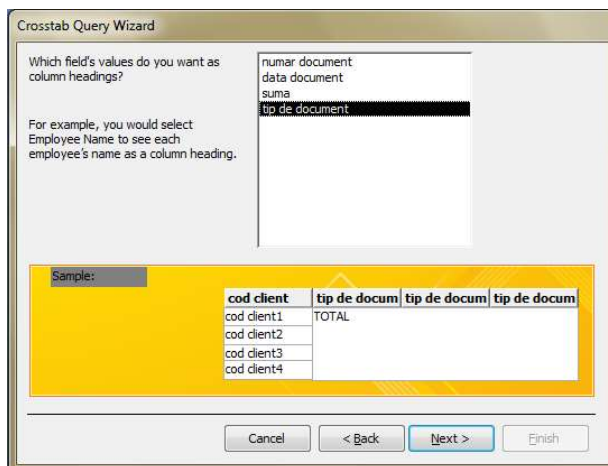


Figura nr. 12.3.8 Stabilirea dimensiunii orizontale a datelor (tip de document)

- Selectarea câmpului căruia i se aplică o funcție agregată (funcția Sum având ca argument câmpul Suma), ca în figura nr. 12.3.9. Tot la acest pas se poate alege apariția unei eventuale coloane de total prin utilizarea opțiunii „Yes, include row sums”.

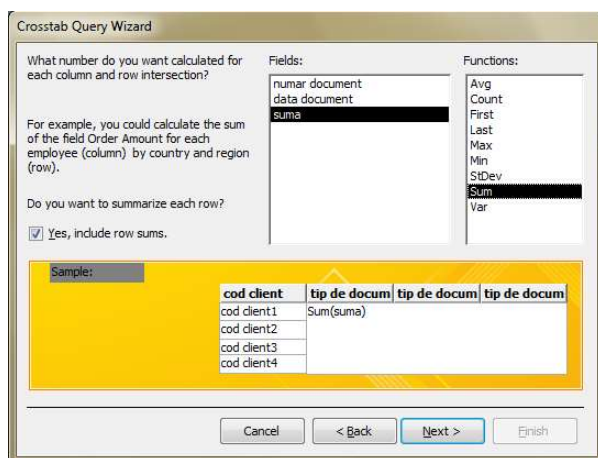


Figura nr. 12.3.9 Alegerea câmpului căruia i se aplică o funcție agregat (SUM(Suma))

- Specificarea numelui interogării. Rezultatul interogării este prezentat în figura nr. 12.3.10.

cod client	Total Of suma	chitanță	ordin de plată
1001	1100	1100	
1002	100	100	
1003	250	250	

Figura nr. 12.3.10 Rezultatul interogării Crosstab

12.4 Definirea interogărilor prin Query Design

Interogările cu un grad redus de complexitate pot fi create și modificate în mod vizual, cu ajutorul utilitarului *Query Design*. El poate fi deschis din meniul *Create* → grupul *Other* → opțiunea *Query Design*.

La deschiderea utilitarului *Query Design* trebuie specificate tabelele din care se vor extrage date sau în care se vor face actualizări.

Cu Query Design se pot realiza următoarele tipuri de interogări:

- interogări de selecție (Select Query);
- interogările încrucișate sau, altfel spus, interogări de orizontalizare a datelor (Crosstab Query);
- interogările de creare a unor tabele rezultat (Make Table Query);
- interogări de adăugare de noi rânduri (Append Query);
- interogări de modificare a rândurilor deja existente (Update Query);
- interogări de ștergere de rânduri (Delete Query).

Tipul implicit de interogare este cel de selecție, în așa fel încât, dacă se dorește realizarea unei interogări de un alt tip, imediat după specificarea tabelor de inclus în interogare, trebuie specificat tipul interogării.

Tipul unei interogări poate fi stabilit din meniul principal, opțiunea *Design*, grupul *Query Type*, sau din meniul ce se obține la click dreapta pe fereastra *Query Design*, submeniul *Query Type*.

Pentru rezolvarea interogărilor realizate cu Query Design, Microsoft Access mai întâi scrie o comandă SQL specifică limbajului neprocedural și apoi dă în execuție această comandă. Conținutul comenzii SQL poate fi vizualizat cu opțiunea *SQL View*, ce o putem găsi atât în meniul principal, la submeniul *Design*, cât și în meniul ce apare la click dreapta pe fereastra *Query Design*.

Interogările din Microsoft Access pot fi folosite:

- direct, în mod vizual, cu gestionarul bazei de date;
- de alte interogări Access (în cazul interogărilor de selecție);
- din ferestrele, rapoartele și modulele VBA (Visual Basic for Applications) ale bazei de date.

12.4.1 Interogări de selecție

Interogările de selecție din Microsoft Access sunt obiecte ale bazei de date folosite pentru a extrage și afișa o parte din datele din baza de date, care îndeplinesc anumite condiții.

Exemplul nr. 4

Să se întocmească lista clienților din localitatea cu codul 1002. Lista va cuprinde numai denumirea clienților.

Rezolvare

Se deschide Query Design și se specifică în fereastra Show Table tabelele sau interogările din care se vor extrage date. O tabelă poate fi introdusă în cadrul unei interogări de mai multe ori, dar nu este cazul exemplului curent.

Dacă o tabelă este introdusă din greșeală în plus în cadrul unei interogări, ea poate fi scoasă:

- prin selectare și apoi apăsarea tastei DELETE;
- prin click dreapta pe tabelă și apoi folosirea opțiunii Remove Table.

Dacă o tabelă nu a fost introdusă și a fost închisă fereastra Show Table, aceasta din urmă poate fi redeschisă explicit dând click dreapta în spațiul în care sunt afișate tabelele, altfel spus în partea superioară a ferestrei principale a utilitarului Query Design.

În exemplul nostru trebuie să selectăm doar tabela *Clienți*.

În partea inferioară a ferestrei principale a utilitarului Query Design se specifică (Figura nr. 12.4.1):

- *Field* – câmpurile folosite în cadrul interogării, indiferent dacă apar sau nu rezultatul final; aici se includ și câmpurile folosite doar în stabilirea unor filtre (așa cum sunt câmpurile Denumire client și Cod localitate din exemplu);
- *Table* – tabelele din care fac parte câmpurile mai sus menționate (tabela *Clienți* în exemplu);
- *Sort* – ordonarea crescătoare sau descrescătoare care să fie aplicată pe câmpurile respective;
- *Show* – dacă respectivul câmp apare sau nu în lista rezultat (în exemplu se marchează doar câmpul Denumire client);
- *Criteria* – condițiile de filtrare a datelor (în exemplu se scrie 1002 pe coloana câmpului Cod localitate, iar Access va încadra singur între ghilimele această valoare, fiind vorba de constantă de tip text). Semnul „=” este opțional în cazul condițiilor de egalitate.

Sub rândul *Criteria* pot fi specificate și alte condiții, ce vor fi despărțite de prima condiție cu operatorul logic *OR* (sau). Expresiile de la zona *Criteria* pot conține, pe același rând, mai multe condiții despărțite de operatori logici SQL, cum ar fi: *AND*, *NOT*, *BETWEEN*, *LIKE*, *IN* etc.

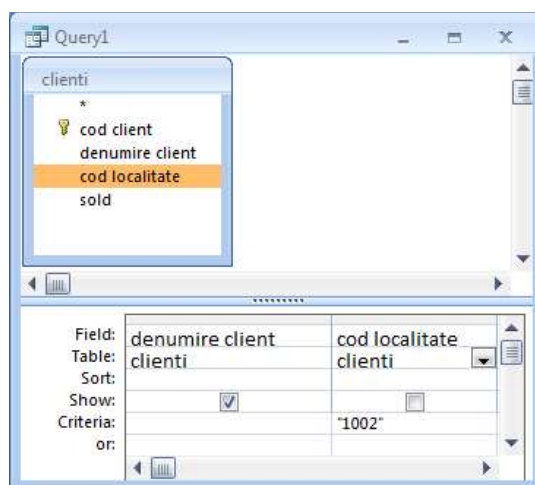


Figura nr. 12.4.1 Fereastra utilitarului Query Design completată pentru exemplul nr. 4

Interogările se pot da în execuție și folosind opțiunea *Run* din meniul *Design*. Interogarea mai poate fi data în execuție și prin selectarea unui alt mod de vizualizare a acesteia. Până acum, pentru creare a fost folosit modul Design (*Design View*), dar acest mod de vizualizare poate fi modificat din meniul *Design*, submeniul *View* în:

- *Datasheet View* - echivalentă cu darea în execuție (vezi figura nr. 12.4.2);
- *PivotTable View* - pentru vizualizarea unui tabel pivot cu datele obținute în urma rulării interogării;
- *PivotChart View* - pentru vizualizarea unui grafic pivot cu datele obținute în urma rulării interogării;
- *SQL View* - pentru vizualizarea comenzii SQL echivalente;

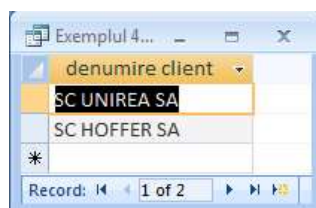


Figura nr. 12.4.2 Rezultatul interogării de la exemplul nr. 4

Interogarea se poate salva cu opțiunea *Save* din linia de instrumente pentru acces rapid (stânga sus) sau cu combinația de taste CTRL + S.

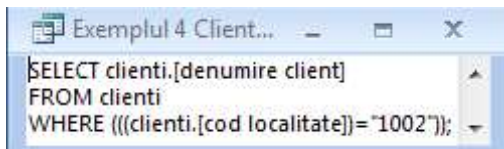


Figura nr. 12.4.3 Comanda SQL de la exemplul nr. 4

Exemplul nr. 4 poate fi rezolvat și cu ajutorul parametrilor, în sensul că, interogarea poate propune introducerea codul localității pentru care dorește o listă a clienților și apoi, afișează rezultatul obținut.

În acest caz, pentru filtrul privind codul localității se va crea un parametru.

Pentru gestionarea parametrilor, se selectează opțiunea *Parameters*:

- din meniul principal *Design* → *Parameters*;
- din meniul ce apare la click dreapta pe fereastra *Query Design*.

În fereastra *Parameters* se completează numele și tipurile de dată ale parametrilor, în exemplul nostru *Cod_Loc* de tip *text* (același tip ca și câmpul *cod localitate* din tabela *clienti*). Este de preferat ca parametrii să aibă nume diferite de cele ale câmpurilor tabelului sau tabelurilor interogării.

În fereastra *Query Design*, spre deosebire de rezolvarea interogării fără parametri, se completează la câmpul *Cod localitate*, în loc de valoarea „1002”, numele parametrului scris între paranteze drepte. Dacă numele parametrului se scrie fără a fi încadrat între paranteze drepte, Access va crede că este vorba de valoarea „Cod_Loc” ce trebuie căutată printre valorile din câmpul *Cod localitate*.

La darea în execuție a unei interogări cu parametri apare fereastra în care trebuie introdusă valoarea de atribuit parametrului pentru respectiva execuție (1002 în cazul exemplului nr. 4):

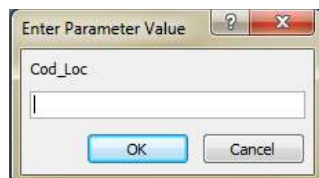


Figura nr. 12.4.4 Fereastra specifică introducerii valorii unui parametru

Instrucțiunea SQL scrisă automat de Query Design pentru rezolvarea cu ajutorul parametrilor a interogării curente este cea din figura nr. 12.4.5.

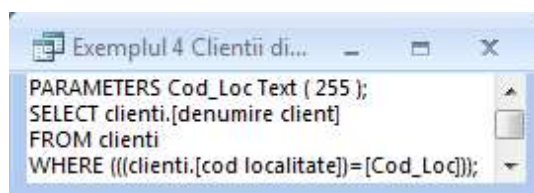


Figura nr. 12.4.5 Comanda SQL ce folosește parametri

Parametrii interogărilor pot fi definiți și implicit, prin simpla lor utilizare între paranteze drepte în zona Criteria. Și la definirea implicită a parametrilor trebuie evitată confuzia de nume, ce poate să apară atunci când aceștia sunt denumiți cu aceleași nume ca și câmpurile tabelor.

Tot în cadrul exemplului nr. 4, dacă dorim crearea unei tabele cu rezultatul obținut se poate alege tipul interogării ca fiind Make-Table în loc de Select. Interogările de creare a unei tabele presupun completarea în fereastra Query Design a acelorași rubrici ca în cazul interogărilor normale de selecție. Diferența constă în faptul că imediat după alegerea tipului de interogare Make-Table apare o fereastră de dialog în care trebuie introdus numele tabelei rezultat (Clienți_din_localitate pentru tabela rezultat din exemplul nr. 4).

Când o interogare de creare de tabele este rulată de mai multe ori, altfel spus când există deja tabela respectivă, este posibil să apară mesajul următor, pentru a se confirma ștergerea vechii versiuni a tabelei.

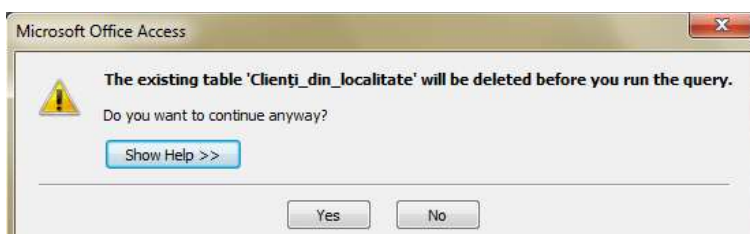



Figura nr. 12.4.6 Mesajul ce apare la rularea repetată a interogării Make-Table

În cazul în care în  → *Access Options* → *Advanced* → *Confirm* este marcată opțiunea *Action queries*, atunci la rularea succesivă a unei interogări de creare de tabele vor apare mai multe mesaje de confirmare.

Pentru interogările de creare de tabele Microsoft Access construiește tot o comandă **SELECT**, în care include clauza **INTO** pentru a specifica tabela rezultat.

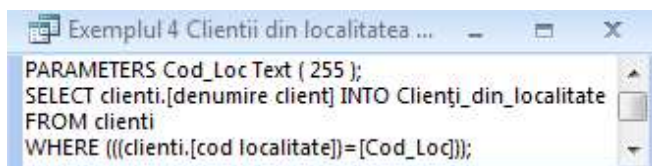


Figura nr. 12.4.7 Comanda SQL a unei interogări de creare a unei tabele

Exemplul nr. 5

Care este conținutul avizului de expediție cu numărul 1002. Lista va conține câmpurile: data aviz, denumire gestiune, denumire client, denumire localitate (localitatea clientului), denumire produs, cantitate, procent TVA, pret.

Rezolvare

Se deschide Query Design și se specifică în fereastra Show Table sursa de date a interogării:

- Avize de expediție (cel puțin pentru câmpul data aviz);
- Gestiune (pentru câmpul denumire gestiune);
- Clienți (pentru câmpul denumire client);
- Localități (pentru câmpul denumire localitate);
- Linii în avize (pentru câmpurile cantitate, procent TVA, pret);
- Produse (pentru denumire produs).

În partea inferioară a ferestrei Query Design se specifică:

- ce câmpuri sunt utilizate de interogare (opțiunea *Field*);
- din ce tabele (opțiunea *Table*);
- cum sunt ordonate datele în rezultat (opțiunea *Sort*);
- dacă respectivul câmp apare sau nu în rezultat (opțiunea *Show*);
- condiții de filtrare (zona *Criteria* alcătuită din rândul *Criteria* și rânduri ce urmează).

Completarea câmpurilor interogărilor poate fi realizată:

- prin scrierea directă a numelor de atribut;
- prin alegerea lor din lista disponibilă la mutarea cursorului pe rândul Field (e mai ușor să se aleagă, mai întâi, numele tabelului și apoi câmpul dorit);
- prin tragerea unui câmp din tabelă în partea inferioară a ferestrei (drag and drop);
- prin dublu click pe numele câmpurilor din zona superioară.

În lista de câmpuri nu trebuie uitate câmpurile folosite în cadrul filtrării, chiar dacă aceste câmpuri nu trebuie să apară în listă (în exemplu, câmpul *numar aviz*). În mod implicit, un câmp are activată opțiunea *Show* și, din acest motiv, pentru câmpurile ce nu apar în listă, ea trebuie deselectedată.

În figura nr. 12.4.8, apare rezolvarea interogării din exemplul nr. 5 cu ajutorul Query Design. Lista de câmpuri ale interogării continuă în partea din dreapta, ele neîncăpând toate în imagine.

Comanda SQL realizată pentru rezolvarea interogării din exemplul nr. 5 este cea prezentată în figura nr. 12.4.9.

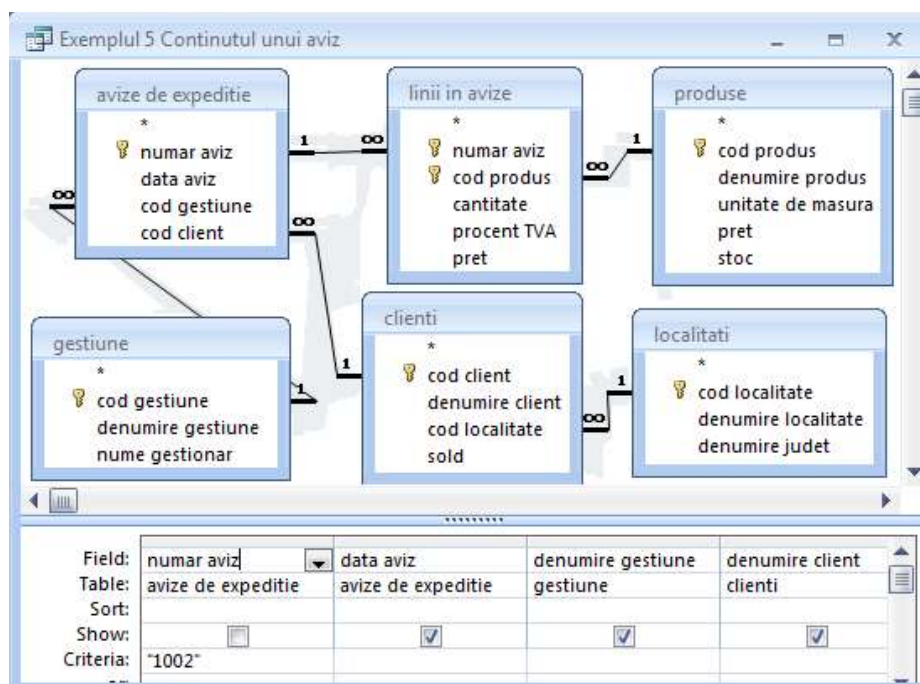


Figura nr. 12.4.8 Rezolvarea exemplului nr. 5 în Query Design

```

SELECT [avize de expeditie].[data aviz], gestiune.[denumire gestiune],
clienti.[denumire client], localitati.[denumire localitate], produse.[denumire
produs], [linii in avize].cantitate, [linii in avize].[procent TVA], [linii in avize].pret
FROM produse
INNER JOIN ((gestiune
INNER JOIN ((localitati
INNER JOIN clienti
ON localitati.[cod localitate] = clienti.[cod localitate])
INNER JOIN [avize de expeditie]
ON clienti.[cod client] = [avize de expeditie].[cod client])
ON gestiune.[cod gestiune] = [avize de expeditie].[cod gestiune])
INNER JOIN [linii in avize]
ON [avize de expeditie].[numar aviz] = [linii in avize].[numar aviz])
ON produse.[cod produs] = [linii in avize].[cod produs]
WHERE ((([avize de expeditie].[numar aviz])="1002"));

```

Figura nr. 12.4.9 Comanda SQL de la exemplul nr. 5

Reluarea exemplului nr. 3

Să se creeze o listă cu încasările de la fiecare client pe tipuri de documente. Lista trebuie să conțină pe prima coloană clienții și câte o coloană pentru fiecare tip de document. La intersecția fiecărui client cu fiecare tip de document trebuie să apară suma încasărilor.

Rezolvarea exemplului nr. 3 cu ajutorul Query Design

În exemplul nr. 3 utilizăm Crosstab Query Wizard pentru a obține o listă cu încasările de la fiecare client pe tipuri de documente, în condițiile în care, pe prima coloană, trebuie să avem toți clienții de la care s-a încasat ceva și pe primul rând (cel de explicație a coloanelor) fiecare tip de document utilizat. La intersecția fiecărui client cu fiecare tip de document trebuie să apară suma încasărilor respective.

Se creează o nouă interogare cu Query Design și după introducerea tabeli Încasări, se alege opțiunea Crosstab Query din meniul principal, submeniul Design sau din meniul ce se obține la click dreapta pe fereastra Query Design, submeniul Query Type.

Spre deosebire de o interogare de selecție normală, în Query Design, în partea inferioară a ferestrei, apar și rubricile Total și Crosstab.

La rubrica Crosstab se specifică dacă respectivele câmpuri sunt folosite:

- pe primele coloane, ca explicație sau total al fiecărui rând (Row Heading);
- pe primul rând, ca explicație a fiecărei coloane (Column Heading);
- la intersecția rândurilor cu coloanele (Value).

La rubrica Total se completează:

- cu Group By pentru câmpurile cu explicații ale rândurilor și ale coloanelor;
- cu o funcție de agregare (Sum, Avg, Min, Max, Count etc.) pentru celelalte câmpuri.

În figura nr. 12.4.10 se arată modul de completare a ferestrei Query Design pentru rezolvarea exemplului nr. 3.

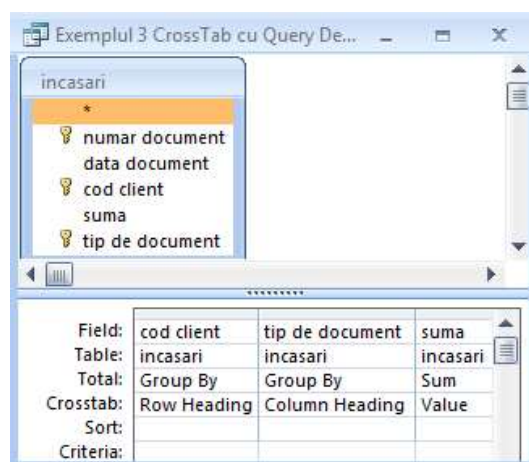


Figura nr. 12.4.10 Rezolvarea exemplului nr. 3 cu Query Design

12.4.2 Interogări de adăugare

Interogările de adăugare (Append Query) presupun completarea unor rubrici diferite de cele de la interogările de selecție, deoarece Access construiește, pe baza datelor din fereastra Query Design, o altă comandă decât SELECT, respectiv, comanda INSERT.

Când vrem să introducem date într-o tabelă cu o interogare de adăugare și deschidem Query Design nu selectăm din fereastra Show Table tabela respectivă, ci doar tabelele din care eventual extragem date pentru a fi introduse în tabela afectată.

Fereastră în care se specifică numele tabelii în care se vor introduce datele apare după selecția tipului de interogare.

Exemplul nr. 6

Introduceți, în tabela Clienți, clientul SC BENNY SRL cu codul 1008, din localitatea cu codul 1001, ce are soldul 777.

Rezolvare

După ce deschidem Query Design, nu selectăm nici o tabelă din fereastra Show Table, dar selectăm tipul interogării ca fiind Append Query (click dreapta pe interogare, selectăm Query Type și Append Query). Selectăm în fereastra Append numele tabelii Clienți.

În rubricile din partea inferioară a ferestrei Query Design, nu vom scrie nimic pe rândul Table, deoarece nu selectăm datele din nici o altă tabelă, iar pe rândul Field vom scrie valori, nu denumiri de câmpuri. Valorile de tip text, ce conțin spații, se vor scrie obligatoriu sub formă de constantă text, între caracterele ghilimele (cum este denumirea clientului, din acest caz).

Implicit, Query Design va atribui nume generate câmpurilor cărora le-au fost specificate valorile pe rândul Field. Pentru fiecare valoare introdusă la Field, pe rândul Append To vom specifica la ce câmp din tabela Clienți facem referire. Fereastra Query Design ar fi completată în acest caz ca în figura nr. 12.4.11.



Figura nr. 12.4.11 Rezolvarea exemplului nr. 6 cu Query Design

Comanda SQL generată automat de Query Design pentru rezolvarea acestei interogări este INSERT, cu o mică diferență față de standardul SQL: se specifică o clauză SELECT, în locul clauzei VALUES pentru specificarea valorilor de inserat. Comanda generată în exemplul nr. 6 este prezentată în figura nr. 12.4.12.

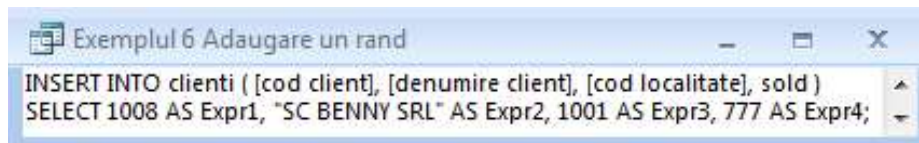


Figura nr. 12.4.12 Comanda SQL corespunzătoare exemplului nr. 6

Rularea de mai multe ori a interogării din exemplul nr. 6 nu este permisă, deoarece s-ar încălca restricția de cheie primară din tabela Clienți. Nu sunt admise mai multe înregistrări cu același cod client.

12.4.3 Interogări de modificare

Interogările de modificare (Update Query), realizate cu Query Design, presupun completarea tuturor informațiilor necesare creării unei comenzi UPDATE, conform standardului SQL.

Exemplul nr. 7

Să se modifice, pentru clientului cu codul 1008, soldul la valoarea 888.

Rezolvare

După deschiderea utilitarului Query Design se selectează tabela Clienti din fereastra Show Table și, apoi, se selectează tipul interogării ca fiind „Update”. În partea inferioară a ferestrei Query Design, se completează atât cu câmpurile în care se vor face modificări (sold), cât și cu câmpurile folosite pentru identificarea rândurilor de actualizat (cod client).

În figura nr. 12.4.13 se prezintă modul de completare a ferestrei Query Design pentru rezolvarea exemplului nr. 7.



Figura nr. 12.4.13 Rezolvarea exemplului nr. 7 cu Query Design

Comanda generată în exemplul nr. 7 este prezentată în figura nr. 12.4.14.

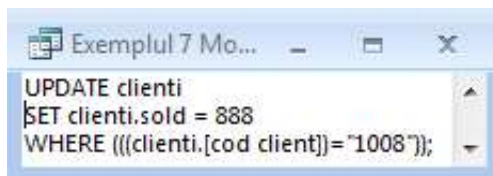


Figura nr. 12.4.14 Comanda SQL corespunzătoare exemplului nr. 7

12.4.4 Interogări de ștergere

Interogările de ștergere, realizate cu Query Design, presupun completarea tuturor informațiilor necesare identificării rândului sau rândurilor de șters.

Exemplul nr. 8

Ștergeți clientul care are codul 1008.

Rezolvare

După deschiderea utilitarului Query Design, se selectează tabela Clienti, din fereastra Show Table și apoi se selectează tipul interogării ca fiind „Delete”. În partea inferioară a ferestrei Query Design se completează cu câmpurile folosite pentru identificarea rândurilor de șters (cod client).

În figura nr. 12.4.15 se prezintă modul de completare a ferestrei Query Design pentru rezolvarea exemplului nr. 8.



Figura nr. 12.4.15 Rezolvarea exemplului nr. 8 cu Query Design

Comanda generată în exemplul nr. 8 este prezentată în figura nr. 12.4.16.

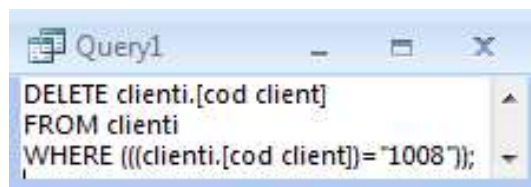


Figura nr. 12.4.16 Comanda SQL corespunzătoare exemplului nr. 8

Capitolul 13 Interogări SQL

Interogările SQL din Access sunt comenzi care, datorită apropierii de specificațiile standardului SQL, sunt asemănătoare din punct de vedere sintactic și semantic, comenzilor existente în alte SGBD-uri. Comenzile SQL din Microsoft Access pot descrie atât operațiuni de extragere a datelor, cât și operațiuni de actualizare a datelor (inserare, modificare, ștergere): SELECT, INSERT, UPDATE, DELETE.

13.1 Primele argumente ale interogărilor SQL

Formatul simplificat al unei interogări SQL de selecție este:

```
SELECT câmp_1, câmp_2, ..., câmp_n  
FROM tabel_1, tabel_2, ..., tabel_n  
WHERE condiție  
ORDER BY câmp_i, câmp_j
```

Doar clauza SELECT a comenzii SELECT este obligatorie în Access. Celelalte argumente sunt opționale, lista lor fiind mai mare decât cea prezentată în acest prim format simplificat. Amintim în acest sens și clauzele: GROUP BY, HAVING, DISTINCT.

Primele argumente ale comenzii SELECT specifică lista de câmpuri din tabelele bazei de date care vor fi extrase în rezultat. Dacă în locul listei câmpurilor se utilizează simbolul *, atunci rezultatul va fi reprezentat de toate câmpurile din tabela sau tabelele specificate în clauza FROM.

Imediat după cuvântul SELECT ar putea urma clauza DISTINCT, dar nu în mod obligatoriu. Ea este utilizată pentru eliminarea, din rezultat, a înregistrărilor care se repetă. Clauza FROM specifică tabelele din care vor fi extrase câmpurile prevăzute în lista de după SELECT.

Clauza WHERE specifică diferite condiții de selecție, în funcție de care sunt extrase doar o parte din rândurile tabelor prezentate la clauza FROM. În formatul comenzii SELECT prezentat mai sus legăturile dintre tabele se specifică tot la clauza WHERE.

Clauza ORDER BY realizează ordonarea liniilor din rezultatul unei interogări. Ordonarea crescătoare este implicită, dar ea se poate realiza și explicit cu clauza ASC (abrevierea pentru cuvântul „ascending”). Pentru o ordonare descrescătoare trebuie folosită clauza DESC (abrevierea cuvântului „descending”).

Pentru a scrie o nouă interogare în Access direct specificând comanda SQL trebuie deschis tot Query Design, dar fără a specifica nimic în fereastra Show Table, se alege opțiunea SQL View din:

- meniul principal, submeniul *Design*;
- meniul contextual al interogării deschise în Query Design (click dreapta în partea superioară a ferestrei Query Design).

Interogările salvate deja se pot edita dând click dreapta pe ele în fereastra de navigare prin obiectele bazei de date și alegând opțiunea Design View. Modul de editare implicit al interogărilor existente (Design View sau SQL View) este cel folosit la ultima editare a acestora.

Mărimea fontului utilizat în fereastra SQL View poate fi modificată de la: butonul Office (stânga sus) → Access Options → Object Designers → Query Design → Size.

Exemplul nr. 1

Lista clienților din localitatea ce are codul 1002. Lista va cuprinde numai denumirea clienților.

```
SELECT [denumire client]  
FROM clienti  
WHERE [cod localitate]="1002"
```

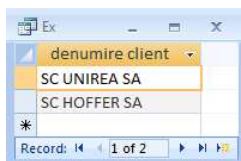


Figura nr. 13.1.1 Rezultatul interogării, exemplul nr. 1

Observații:

- Codul localității 1002 a fost scris ca o constantă de tip text, delimitat la stânga și la dreapta de *ghilimele*, deoarece câmpul cod localitate din tabela Clienti este un câmp de tip text. Dacă s-ar fi scris interogarea fără ghilimele ar fi apărut următoarea eroare:



Figura nr. 13.1.2 Eroarea specifică folosirii neadecvate a unui tip de dată

- ca delimitator pentru constantele de tip text poate fi folosit și caracterul *apostrof* (');
- dacă închidem și redeschidem interogarea putem sesiza faptul că Access a adăugat un caracter *punct și virgulă* (;) la interogare. Nu ne va deranja cu nimic în acest exemplu. Există situații, în practică, în care la interogări complexe Access modifică singur interogări corecte în interogări incorecte sintactic;
- numele de câmpuri și de tabele care conțin spații se scriu între *paranteze drepte*.

Exemplul nr. 2

Care sunt clienții din localitatea având codul 1006, care au soldul mai mare de 1.000? Lista va cuprinde: denumire client, cod client.

```
SELECT [denumire client], [cod client]
FROM Clienti
WHERE [cod localitate] = "1006"
AND Sold >1000
```

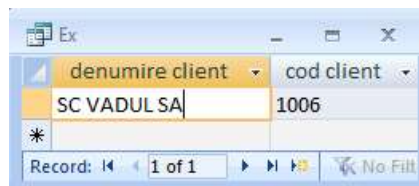
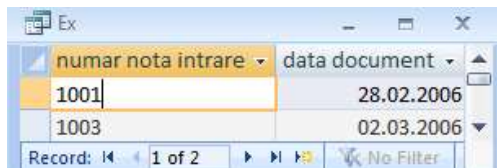


Figura nr. 13.1.3 Rezultatul interogării, exemplul nr. 2

Exemplul nr. 3

*Care sunt documentele de intrare emise între 15 februarie și 15 martie 2006?
Lista va cuprinde numărul și data documentului.*

```
SELECT [numar nota intrare], [data document]
FROM [documente de intrare]
WHERE [data document] >= #2006/02/15#
AND [data document] <= #2006/03/15#
```



numar nota intrare	data document
1001	28.02.2006
1003	02.03.2006

Figura nr. 13.1.4 Rezultatul interogării, exemplul nr. 3

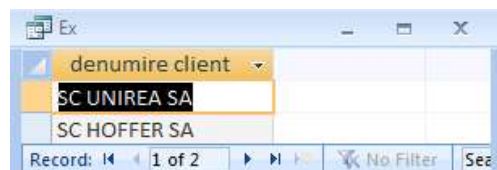
Observații:

- data calendaristică s-a scris în format ISO: YYYY/MM/DD;
- fără a face nici o setare, interogarea ar fi putut folosi și formatele de dată calendaristică: DD/MM/YYYY sau MM/DD/YYYY. Access încearcă să determine ce format de dată a fost folosit. În cazul în care ziua din data calendaristică este mai mică sau egală cu 12, adică s-ar putea confunda cu o lună, atunci se consideră a fi utilizat formatul american MM/DD/YYYY;
- în formatul unei date, ca separator, poate fi folosit și caracterul liniuță (*minus*).

Exemplul nr. 4

*Care sunt clienții din județul ce are în câmpul [denumire judet] valoarea „IS”?
Lista va cuprinde doar denumirile clienților.*

```
SELECT [denumire client]
FROM clienti, localitati
WHERE clienti.[cod localitate]=localitati.[cod localitate]
AND [denumire judet]='IS'
```



denumire client
SC UNIREA SA
SC HOFFER SA

Figura nr. 13.1.5 Rezultatul interogării, exemplul nr. 4

Observații:

- la clauza FROM pot fi scrise mai multe tabele. Lista de tabele folosește ca separator caracterul *virgulă*;
- când folosim date din mai multe tabele, între care există relații, la operatorul WHERE trebuie specificate aceste relații. O altă soluție este operatorul INNER JOIN:

```
SELECT [denumire client]
FROM clienti INNER JOIN localitati
ON clienti.[cod localitate]=localitati.[cod localitate]
WHERE [denumire judet]='IS'
```

- când, în tabelele specificate în clauza FROM, există câmpuri cu același nume, la folosirea unui astfel de câmp se adaugă un prefix format din numele tabelului din care face parte urmat de caracterul punct. Nerespectarea acestei reguli duce, în Access, la eroarea următoare:

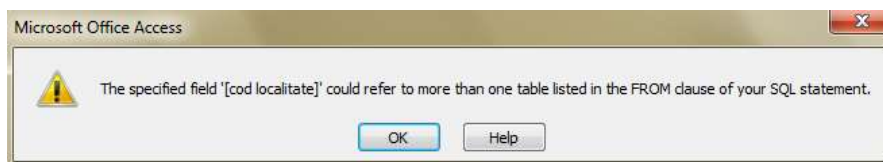


Figura nr. 13.1.6 Câmpul specificat (FROM) poate referi mai mult de o tabelă

- fiecare tabel poate avea un al doilea nume (alias):

```
SELECT [denumire client]
FROM clienti c, localitati l
WHERE c.[cod localitate]=l.[cod localitate]
AND [denumire judet]='IS'
```

În exemplul de mai sus, tabela **Clienți** a fost denumită mai scurt c, iar tabela **Localități** l. Avantajele acestei facilități sunt vizibile mai ales în cazul interogărilor complexe. Pe lângă avantajul scrierii a mai puține caractere, trebuie evidențiată și ușurința în urmărirea provenienței câmpurilor dacă alias-urile sunt folosite pentru toate câmpurile din interogare.

Exemplul nr. 5

Întocmiți lista clienților din județul Iași în ordinea alfabetică a denumirii acestora. Lista va cuprinde: denumirea clientului și soldul acestuia.

```
SELECT [denumire client], [sold]
FROM clienti c, localitati l
WHERE c.[cod localitate]=l.[cod localitate]
```

```
AND [denumire judet]='IS'
ORDER BY [denumire client]
```

denumire client	sold
SC HOFFER SA	658
SC UNIREA SA	65,5

Figura nr. 13.1.7 Rezultatul interogării, exemplul nr. 5

Exemplul nr. 6

Întocmiți lista clienților din județul Iași ordonată după valorile descrescătoare ale câmpului sold. Lista va cuprinde: denumire client și sold.

```
SELECT [denumire client], [sold]
FROM clienti c, localitati l
WHERE c.[cod localitate]=l.[cod localitate]
AND [denumire judet]='IS'
ORDER BY [sold] DESC
```

denumire client	sold
SC HOFFER SA	658
SC UNIREA SA	65,5

Figura nr. 13.1.8 Rezultatul interogării, exemplul nr. 6

13.2 Utilizarea principalilor operatori

Expresiile logice din Access pot conține operatori cum ar fi: =, <, >, <=, >=, AND, OR, BETWEEN (cuprins între), LIKE (ca și, la fel ca), IN (în, existent în mulțimea).

Dialectul de SQL al Microsoft Access include și operatorul de reuniune: UNION (reunit cu).

Reluarea exemplului nr. 3

Interogarea de la exemplul nr. 3 putea fi scrisă, mai simplu, utilizând operatorul BETWEEN:

```
SELECT [numar nota intrare], [data document]
FROM [documente de intrare]
WHERE [data document] BETWEEN #2006/02/15# AND #2006/03/15#
```

Exemplul nr. 7

Ce încasări s-au făcut, în perioada 1 ianuarie 2006 – 31 martie 2006, cu valori între 1.000 și 10.000, de la clienții a căror denumire începe cu una din literele de la A la S?

```
SELECT i.*  
FROM Incasari i, Clienti c  
WHERE i.[cod client] = c.[cod client]  
      AND [data document] BETWEEN #2006/01/01# AND #2006/03/31#  
      AND [suma] BETWEEN 1000 AND 10000  
      AND MID([denumire client], 1, 1) BETWEEN 'A' AND 'S'
```



numar docu	data doc	cod client	suma	tip de document
1001	28.02.2006	1001	1100	ordin de plată

Figura nr. 13.2.1 Rezultatul interogării, exemplul nr. 7

Observații:

- Operatorul BETWEEN poate fi folosit pentru mai multe tipuri de câmpuri (caracter, numeric, dată calendaristică etc.);
- Funcția MID(*șir_de_caractere*, *caracterul_de_început*, *lungimea_subșirului*) returnează un subșir extras dintr-un șir inițial. În cazul nostru, din șirul de caractere dat de denumirea clientului se extrage, începând cu primul caracter (parametrul al doilea = 1), un subșir format dintr-un singur caracter (parametrul al treilea = 1);
- Când e vorba de funcții de prelucrare a șirurilor de caractere nucleul de SQL din Access preferă să folosească numele de funcții specifice Visual Basic în detrimentul celor din standardul SQL (MID în loc de SUBSTRING, UCASE în loc de UPPER, LCASE în loc de LOWER). Ba mai mult, toate funcțiile din VBA pot fi folosite în interogările Access, inclusiv funcțiile VBA create de utilizatori.

Aceeași interogare mai poate fi scrisă și astfel:

```
SELECT i.*  
FROM Incasari i, Clienti c  
WHERE i.[cod client] = c.[cod client]  
      AND MONTH([data document]) BETWEEN 1 AND 3  
      AND YEAR([data document])  
      AND [suma] BETWEEN 1000 AND 10000  
      AND LEFT([denumire client], 1) BETWEEN 'A' AND 'S'
```

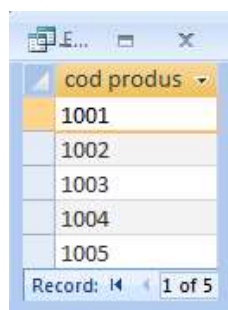
Observații:

- Funcția MONTH(*data*) returnează luna din datele calendaristice folosite ca argument;
- Funcția YEAR(*data*) returnează anul din datele calendaristice folosite ca argument;
- Funcția LEFT(*șir_de_caractere*, *lungimea_subșirului*) returnează un subșir dintr-un șir inițial începând cu primul caracter din partea stângă. Funcția LEFT echivalează cu funcția MID atunci când al doilea caracter al acesteia din urmă (*caracterul_de_început*) este 1.

Exemplul nr. 8

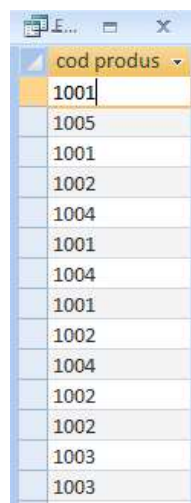
Care sunt codurile produselor ce apar în documentele de intrare sau în avizele de expediție?

```
SELECT [cod produs] FROM [linii in documente]
UNION
SELECT [cod produs] FROM [linii in avize]
```



cod produs
1001
1002
1003
1004
1005

Figura nr. 13.2.2 Rezultatul interogării, exemplul nr. 8



cod produs
1001
1005
1001
1002
1004
1001
1004
1001
1002
1004
1002
1002
1003
1003

Figura nr.13.2.3 Rezultatul interogării în cazul în care s-ar fi folosit UNION ALL, exemplul 8

Observații:

- Cu ajutorul operatorului UNION se realizează reuniunea rezultatelor a două sau a mai multor interogări. Conform operației de reuniune din algebra relațională

se elimină automat liniile identice din rezultat. Dacă se dorește obținerea tuturor linilor din respectivele mulțimi, chiar dacă acestea se repetă, se folosește clauza ALL (vezi figura nr. 13.2.3).

- Pentru operatorul reuniune, indiferent de SGBD-ul utilizat, se impune condiția ca frazele SELECT respective să furnizeze același număr de câmpuri. Dacă această condiție nu este îndeplinită, în Access apare eroarea din figura nr. 13.2.4.

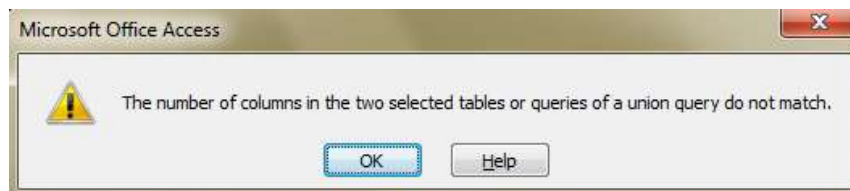


Figura nr. 13.2.4 Eroare ce apare când numărul de coloane de reunit nu este același

Exemplul nr. 9

Întocmiți fișa produsului având codul 1001 (fișa produsului conține stocul inițial, intrările și ieșirile pentru produsul respectiv).

```
SELECT 'Stoc inițial' AS TipDoc, SPACE(8) AS NrDoc, Stoc AS Cantitate
FROM Produe
WHERE [Cod Produs] = '1001'
UNION
SELECT 'Nota intrare', [numar nota intrare], cantitate
FROM [linii in documente]
WHERE [cod produs]='1001'
UNION
SELECT 'Aviz de expeditie', [numar aviz], cantitate
FROM [linii in avize]
WHERE [cod produs]='1001'
```

TipDoc	NrDoc	Cantitate
Aviz de expeditie	1001	4
Aviz de expeditie	1002	6
Nota intrare	1001	100
Nota intrare	1002	200
Stoc inițial		500

Figura nr. 13.2.5 Rezultatul interogării, exemplul nr. 9

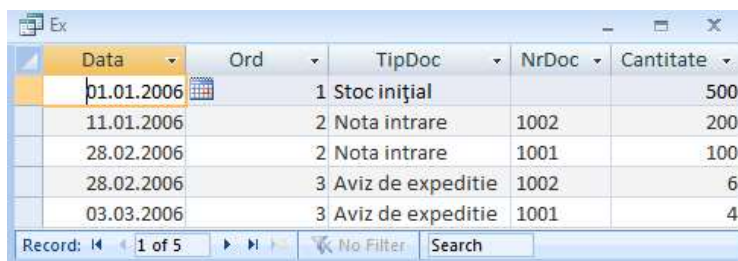
Observații:

- Prima subinterogare folosește alias-uri pentru câmpuri cu ajutorul clauzei AS;
- Prima subinterogare dă și structura finală a rezultatului reuniunii (numele câmpurilor și mărimea lor);
- Funcția SPACE returnează un șir de caractere spațiu de o anumită lungime. Ea a fost folosită, în acest exemplu, deoarece stocul inițial nu apare în nici un document justificativ.
- Ordinea implicită a liniilor din rezultat este ordinea crescătoare după valorile returnate de primul, al doilea și al treilea câmp.

Exemplul nr. 10

Întocmiți fișa produsului având codul 1001, în care ordinea liniilor din rezultat să fie ordinea cronologică de apariție a documentelor (prima linie va fi ocupată de soldul inițial).

```
SELECT #2006-01-01# as Data, 1 as Ord, 'Stoc inițial' AS TipDoc, SPACE(8) AS
NrDoc, Stoc AS Cantitate
FROM Prodeuse
WHERE [Cod Prodeus] = '1001'
UNION
SELECT [data document], 2, 'Nota intrare', i.[numar nota intrare], cantitate
FROM [linii in documente] L, [documente de intrare] i
WHERE L.[numar nota intrare] = i.[numar nota intrare]
AND [Cod Prodeus] = '1001'
UNION
SELECT [data aviz], 3, 'Aviz de expeditie', a.[numar aviz], cantitate
FROM [linii in avize] L, [avize de expeditie] a
WHERE L.[numar aviz] = a.[numar aviz]
AND [cod produs]='1001'
```



Data	Ord	TipDoc	NrDoc	Cantitate
01.01.2006	1	Stoc inițial		500
11.01.2006	2	Nota intrare	1002	200
28.02.2006	2	Nota intrare	1001	100
28.02.2006	3	Aviz de expeditie	1002	6
03.03.2006	3	Aviz de expeditie	1001	4

Figura nr. 13.2.6 Rezultatul interogării, exemplul nr. 10

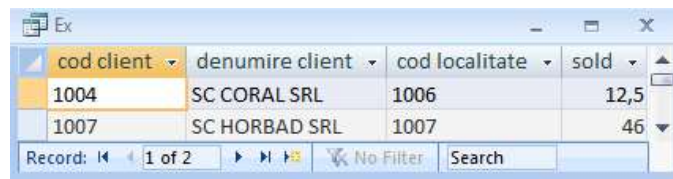
Observație:

Al doilea câmp (Ord) este folosit doar pentru ordonarea documentelor din aceeași dată în așa fel încât mai întâi să apară documentele de intrare și apoi avizele de expediție.

Exemplul nr.11

Care sunt clienții de tip SRL?

```
SELECT * FROM Clienti
WHERE [denumire client] LIKE '*SRL*'
```



cod client	denumire client	cod localitate	sold
1004	SC CORAL SRL	1006	12,5
1007	SC HORBAD SRL	1007	46

Figura nr. 13.2.7 - Rezultatul interogării, exemplul nr. 11

Observații:

- Operatorul LIKE permite căutarea unor șiruri de caractere;
- În sintaxa ANSI 89, caracterul *asterix* (*), inclus în formatul specificat cu operatorul LIKE, semnifică posibilitatea existenței unui număr nelimitat de caractere, iar *semnul întrebării* (?) semnifică existența unui singur caracter;
- În sintaxa ANSI 92, caracterul *procent* (%) semnifică posibilitatea existenței unui număr nelimitat de alte caractere, iar *liniuța de subliniere* (_) semnifică existența unui singur caracter;
- Trecerea de la o sintaxă la alta se face de la butonul Office (stânga sus) → Access Options → Object Designers → Query Design → SQL Server Compatible Syntax (ANSI 92) → This database;
- Trecerea la o altă sintaxă **nu** presupune modificarea automată a interogărilor realizate în precedenta notație, așa încât, o parte din acestea pot deveni incorecte;

Exemplul nr. 12

Care sunt gestionarii ce au numele de familie format din 7 caractere și se termină în 'escu'?

```
SELECT * FROM Gestione
WHERE UCASE([nume gestionar]) LIKE '?????ESCU *'
```



cod gestiune	denumire gestiune	nume gestionar
1001	Depozit	Popescu Vasilica

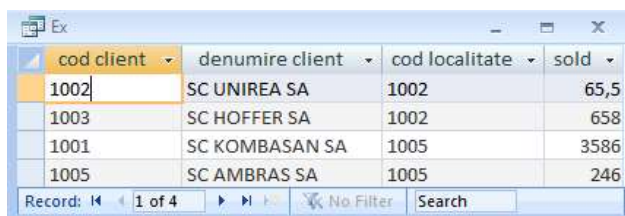
Record: 1 of 1 | No Filter | Search

Figura nr. 13.2.8 - Rezultatul interogării, exemplul nr. 12

Exemplul nr.13

Întocmiți lista clienților din județele Iași, Vaslui, Suceava și Botoșani.

```
SELECT c.*
FROM Clienti C, Localitati L
WHERE c.[cod localitate] = L.[cod localitate]
AND L.[denumire judet] IN ('IS','VS','SV','BT')
```



cod client	denumire client	cod localitate	sold
1002	SC UNIREA SA	1002	65,5
1003	SC HOFFER SA	1002	658
1001	SC KOMBASAN SA	1005	3586
1005	SC AMBRAS SA	1005	246

Record: 1 of 4 | No Filter | Search

Figura nr. 13.2.9 - Rezultatul interogării, exemplul nr. 13

Observații:

- Operatorul IN este folosit pentru a testa dacă o expresie se regăsește într-o listă de valori;
- Această problemă mai are și o altă soluție în care clauza IN este înlocuită cu operatorul logic OR:

```
SELECT c.*
FROM Clienti C, Localitati L
WHERE c.[cod localitate] = L.[cod localitate]
AND (L.[denumire judet] = 'IS'
OR L.[denumire judet] = 'VS'
OR L.[denumire judet] = 'SV'
OR L.[denumire judet] = 'BT')
```

- Mulțimea în care caută operatorul IN nu este neapărat o listă de valori, ci poate fi mulțimea obținută ca rezultat al unei subinterogări. Interogarea mai poate fi scrisă și astfel:

```
SELECT *
FROM Clienti
WHERE [cod localitate] IN
```

```
(SELECT [cod localitate]
FROM Localitati
WHERE [denumire judet] IN ('IS','VS','SV','BT'))
```

13.3 Funcții de agregare

Funcțiile de agregare permit efectuarea de calcule asupra valorilor luate de un atribut sau a valorilor luate de o expresie: COUNT, SUM, AVG, MIN, MAX.

Dacă într-o frază SQL se folosește o funcție de agregare, dar lipsește clauza GROUP BY (vezi paragraful următor), rezultatul va avea întotdeauna o singură linie.

Exemplul nr. 14

Câți clienți aveau soldul inițial mai mare ca zero?

```
SELECT COUNT(*) FROM Clienti WHERE Sold > 0
```

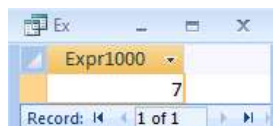


Figura nr. 13.3.1 Rezultatul interogării, exemplul nr. 14

Exemplul nr. 15

Câți clienți sunt din județul Iași?

```
SELECT COUNT(*)
FROM Clienti
WHERE [cod localitate] IN (SELECT [cod localitate]
FROM Localitati
WHERE [denumire judet]='IS')
```

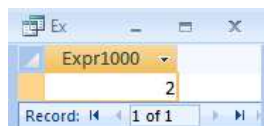


Figura nr. 13.3.2 Rezultatul interogării, exemplul nr. 15

Exemplul nr. 16

Câte documente de intrare s-au întocmit în gestiunea Popescu în anul 2006?

```
SELECT COUNT(*)
FROM gestiune g, [documente de intrare] d
WHERE g.[cod gestiune] = d.[cod gestiune]
```

```
AND UCASE([nume gestionar]) LIKE '*POPESCU*'
AND YEAR([data document])=2006
```

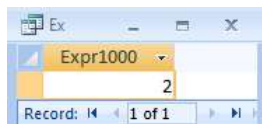


Figura nr.13.3.3 Rezultatul interogării, exemplul nr. 16

Observații:

- Funcția UCASE(*șir_caractere*) transformă un șir de caractere în majuscule; ea a fost folosită aici pentru că nu se știe cum au fost scrise numele gestionarilor în baza de date. De exemplu, gestionarul căutat putea fi scris Popescu sau POPESCU;
- Dacă există mai mulți gestionari cu numele Popescu atunci, pentru a obține rezultatul dorit, mai avem nevoie și de alte informații, cum ar fi prenumele lui.

Exemplul nr. 17

Care este valoarea încasărilor de la clientul cu codul 1001?

```
SELECT SUM(Suma) as Total
FROM Incasari
WHERE [cod client] = '1001'
```

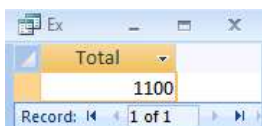


Figura nr.13.3.4 Rezultatul interogării, exemplul nr. 17

Exemplul nr. 18

Care a fost suma totală a soldurilor clienților?

```
SELECT SUM(Sold) as Total From Clienti
```

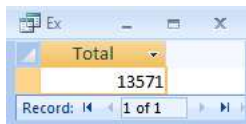
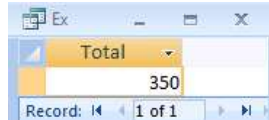


Figura nr.13.3.5 Rezultatul interogării, exemplul nr. 18

Exemplul nr. 19

Care este suma încasată de la clienții din municipiul Iași?

```
SELECT SUM(i.Suma) as Total
FROM Incasari i, Clienti C, Localitati L
WHERE i.[cod client] = C.[cod client]
      AND C.[cod localitate] = L.[cod localitate]
      AND UCASE([denumire localitate]) like 'IASI'
```



Total
350

Record: 1 of 1

Figura nr.13.3.6 Rezultatul interogării, exemplul nr. 19

Alte soluții pentru aceeași interogare:

1. folosind o subinterogare și operatorul IN:

```
SELECT SUM(Suma) as Total
FROM Incasari
WHERE [cod client] IN
      (SELECT [cod client]
       FROM Clienti C, Localitati L
       WHERE C.[cod localitate] = L.[cod localitate]
            AND UCASE([denumire localitate]) like 'IASI')
```

2. tot folosind o subinterogare și operatorul IN, dar o subinterogare mai mică:

```
SELECT SUM(i.Suma) as Total
FROM Incasari i, Clienti C
WHERE i.[cod client] = C.[cod client]
      AND C.[cod localitate] in (SELECT [cod localitate] FROM Localitati
                                WHERE UCASE([denumire localitate]) like 'IASI')
```

3. folosind o subinterogare și operatorul NOT IN:

```
SELECT SUM(Suma) as Total
FROM Incasari
WHERE [cod client] NOT IN
      (SELECT [cod client]
       FROM Clienti C, Localitati L
       WHERE C.[cod localitate] = L.[cod localitate]
            AND UCASE([denumire localitate]) NOT like 'IASI')
```

4. folosind două subinterogări imbricate:

```
SELECT SUM(Suma) as Total
```

```

FROM Incasari
WHERE [cod client] IN
(SELECT [cod client]
FROM Clienti C
WHERE [cod localitate] IN
(SELECT [cod localitate]
FROM Localitati
WHERE UCASE([denumire localitate]) like 'IA?I'))

```

5. folosind funcția IIF ce returnează una din două valori în funcție de îndeplinirea sau neîndeplinirea unei anumite condiții:

```

SELECT SUM(IIF([denumire localitate] = 'Iași', i.Suma, 0)) as Total
FROM Incasari i, Clienti C, Localitati L
WHERE i.[cod client] = C.[cod client]
AND C.[cod localitate] = L.[cod localitate]

```

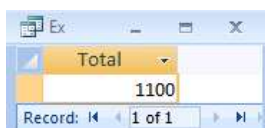
Exemplul nr. 20

Care volumul încasărilor din luna februarie 2006?

```

SELECT SUM(i.Suma) as Total
FROM Incasari i
WHERE MONTH([data document]) = 2
AND YEAR([data document]) = 2006

```



Total
1100

Record: 1 of 1

Figura nr. 13.3.7 Rezultatul interogării, exemplul nr. 20

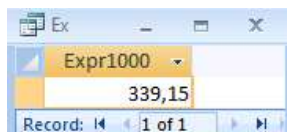
Exemplul nr. 21

Care este totalul valorii vânzărilor din luna februarie 2006?

```

SELECT SUM(Cantitate * Pret * (1 + [procent TVA]/100))
FROM [avize de expeditie] a, [linii in avize] l
WHERE a.[numar aviz] = l.[numar aviz]
AND MONTH([data aviz]) = 2
AND YEAR([data aviz]) = 2006

```



Expr1000
339,15

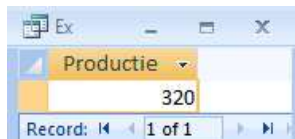
Record: 1 of 1

Figura nr. 13.3.8 Rezultatul interogării, exemplul nr. 21

Exemplul nr. 22

Care este totalul valorii producției din luna februarie 2006?

```
SELECT SUM(Cantitate*Pret) as Productie
FROM [documente de intrare] d, [linii in documente] l, Produse p
WHERE d.[numar nota intrare]=l.[numar nota intrare]
      AND l.[cod produs]=p.[cod produs]
      AND MONTH([data document]) = 2
      AND YEAR([data document]) = 2006
```



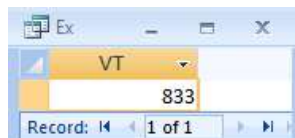
Productie
320

Figura nr. 13.3.9 Rezultatul interogării, exemplul nr. 22

Exemplul nr. 23

Care este valoarea totală a avizului de expediție cu numărul 1001?

```
SELECT SUM(Cantitate*Pret*(1+[procent TVA]/100)) as VT
FROM [linii in avize]
WHERE [numar aviz]='1001'
```



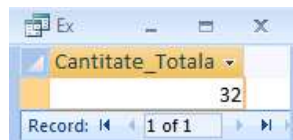
VT
833

Figura nr. 13.3.10 Rezultatul interogării, exemplul nr. 23

Exemplul nr. 24

Ce cantitate de lapte s-a vândut pe piața municipiului Iași în anul 2006?

```
SELECT SUM(La.Cantitate) as Cantitate_Totala
FROM Localitati L, Clienti c, [avize de expeditie] a, [linii in avize] La,
Produse p
WHERE L.[cod localitate] = c.[cod localitate]
      AND c.[cod client] = a.[cod client]
      AND a.[numar aviz] = la.[numar aviz]
      AND la.[cod produs] = p.[cod produs]
      AND UCASE(l.[denumire localitate]) LIKE 'IA?I'
      AND UCASE(p.[denumire produs]) LIKE '*LAPTE*'
```



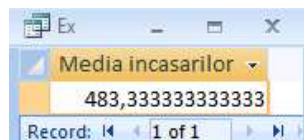
Cantitate_Totale
32

Record: 1 of 1

Figura nr. 13.3.11 Rezultatul interogării, exemplul nr. 24

Exemplul nr. 25*Care este valoarea medie a încasărilor?*

```
SELECT AVG(Suma) as [Media încasarilor] FROM Incasari
```



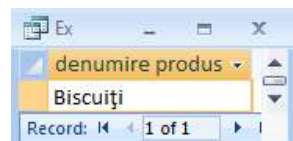
Media încasarilor
483,333333333333

Record: 1 of 1

Figura nr. 13.3.12 Rezultatul interogării, exemplul nr. 25

Exemplul nr. 26*Care este produsul care avea la începutul anului cel mai mare stoc?*

```
SELECT [denumire produs]
FROM Produse
WHERE Stoc IN (SELECT MAX(Stoc) FROM Produse)
```



denumire produs
Biscuiți

Record: 1 of 1

Figura nr.13.3.13 Rezultatul interogării, exemplul nr. 26

Exemplul nr. 27*Care este cea mai mare încasare din anul 2006?*

```
SELECT * FROM Incasari
WHERE Suma IN (
    SELECT MAX(Suma)
    FROM Incasari
    WHERE YEAR([data document]) = 2006)
AND YEAR([data document]) = 2006
```

numar document	data document	cod client	suma	tip de document
1001	28.02.2006	1001	1100	ordin de plată

Figura nr.13.3.14 Rezultatul interogării, exemplul nr. 27

13.4 Gruparea înregistrărilor

Clauza GROUP BY determină grupuri pe baza valorilor luate de unul sau mai multe câmpuri. Această clauză are sens doar dacă se folosește în interogare cel puțin o funcție de agregare. În exemplele ce conțineau funcții de agregare, dar fără să conțină și clauza GROUP BY, respectivele funcții de agregare se executau o singură dată. Cu clauza GROUP BY putem face ca aceste funcții de agregare să se execute de mai multe ori, o dată pentru fiecare grup creat.

Clauza HAVING este asemănătoare cu clauza WHERE, numai că ea operează cu funcții de agregare asupra grupului.

Exemplul nr. 28

Câți clienți sunt din fiecare localitate? Lista va cuprinde: codul localității și numărul de clienți.

```
SELECT [cod localitate], Count(*) AS NrCienti
FROM clienti
GROUP BY [cod localitate]
```

cod localitate	NrCienti
1002	2
1005	2
1006	2
1007	1

Figura nr.13.4.1 Rezultatul interogării, exemplul nr. 28

Observații:

- La clauza SELECT, pe lângă expresiile ce conțin funcții de agregare, mai pot fi specificate doar câmpurile de la clauza GROUP BY. Dacă, în interogarea de mai sus, am mai adăuga la clauza SELECT câmpul *cod client*, am primi mesajul de eroare din figura nr. 13.4.2.

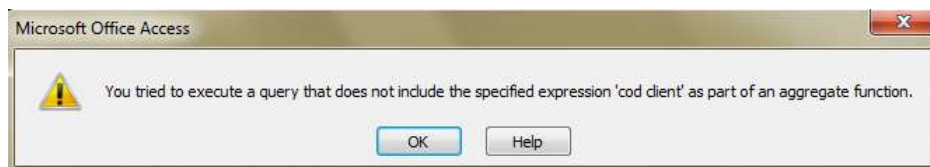


Figura nr.13.4.2 Eroare ce poate apare interogările cu funcții de agregare

- În exemplul curent, funcția de agregare COUNT se execută câte o dată pentru fiecare localitate întâlnită în tabela *Clienți* și NU câte o dată pentru fiecare localitate din tabela *Localități*.

Exemplul nr. 29

Câți clienți sunt din fiecare județ?

```
SELECT l.[denumire județ], Count(*) AS NrCienti
FROM localitati AS l INNER JOIN clienti AS c
ON l.[cod localitate] = c.[cod localitate]
GROUP BY l.[denumire județ]
```

denumire județ	NrCienti
GL	3
IS	2
VS	2

Figura nr.13.4.3 Rezultatul interogării, exemplul nr. 29

Exemplul nr. 30

Care sunt documentele de intrare cu cele mai multe linii (pot exista mai multe documente cu același număr maxim de linii)? Lista va cuprinde doar numerele documentelor de intrare.

```
SELECT [numar nota intrare]
FROM [linii in documente]
GROUP BY [numar nota intrare]
HAVING count(*)=
(SELECT MAX(câte) FROM
(SELECT count(*) as câte
FROM [linii in documente]
GROUP BY [numar nota intrare]
))
```