Capitolul 12 Actualizarea şi interogarea bazelor de date

Datele din bazele de date Access pot fi extrase şi manipulate (introduse, modificate şi şterse) cu ajutorul interogărilor. Astfel de interogări pot fi salvate în cadrul bazei de date la secţiunea „*Queries*”, dar ele pot fi scrise şi în cadrul aplicaţiilor ce folosesc bazele de date Access.

În funcţie de efectul lor, interogările Access pot fi:

- interogări de selecţie (Select Query, Make Table Query, Crosstab Query);
- interogări de inserare (Append Query);
- interogări de modificare (Update Query);
- interogări de ştergere (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 foloseşte 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 şi 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 posibilităţile de filtrare, ordonare şi actualizare directă a datelor, precum şi instrumentele de tip Wizard şi cele de tip RAD existente în Access.

12.1 Filtrarea şi ordonarea directă a datelor dintr-o tabelă

Utilitatea bazelor de date este dată de posibilitatea obţinerii informaţiilor 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 şi îndelungat.

Exemplul nr. 1

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

Rezolvare

Se deschide tabela Clienţi dând dublu click pe ea şi se dă click dreapta pe coloana Cod localitate în partea de date (pe spaţiul alb) şi nu pe antetul tabelului. Se obţine meniul din figura nr. 12.1.1.

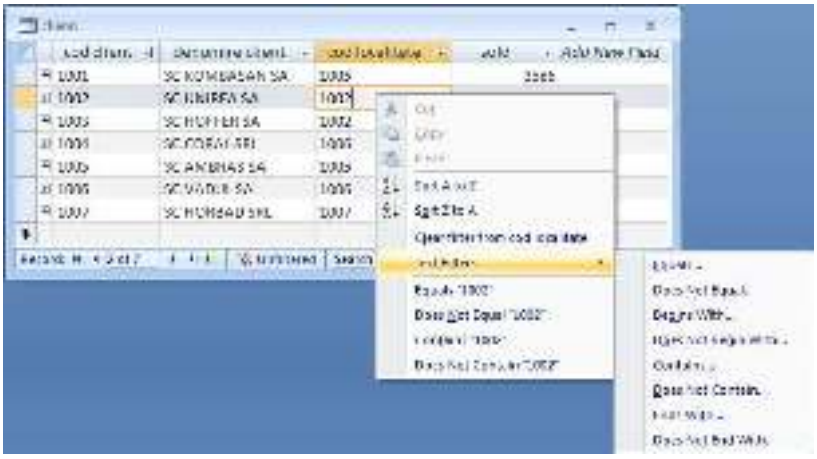


Figura nr. 12.1.1 Meniul specific filtrării directe a datelor

Pentru rezolvarea exemplului nr. 1 putem alege opţiunea *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 opţiunea *Text Filters* → *Equals* şi să scriem în fereastra care ne apare valoarea căutată.

Ordonările şi filtrările pot fi realizate şi cu instrumentele din zona *Sort&Filter* din meniul *Home*. Pe lângă funcţionalităţile specifice meniului contextual prezentat în figura 12.1.1, în zona *Sort&Filter* mai întâlnim opţiunea *Filter* cu aceeaşi funcţionalitate ca în Excel.

Toate sortările şi 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 afişare a datelor dintr-o tabelă (ordinea datelor şi a coloanelor).

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

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

Ştergerea unui rând se poate face cu ajutorul opţiunii *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.

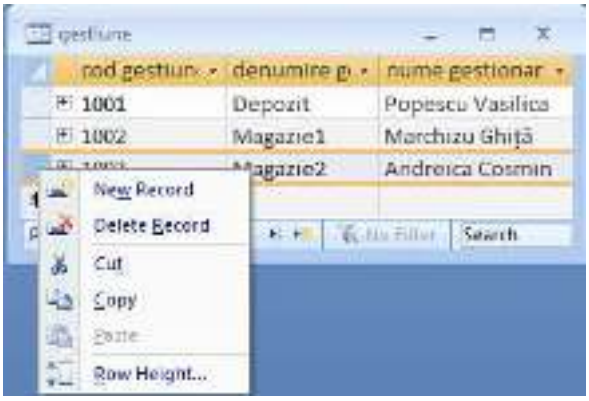


Figura nr. 12.22.2 Meniul pentru gestiunea înregistrărilor dintr-o tabelă

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*, opţiunea *Query Wizard*.

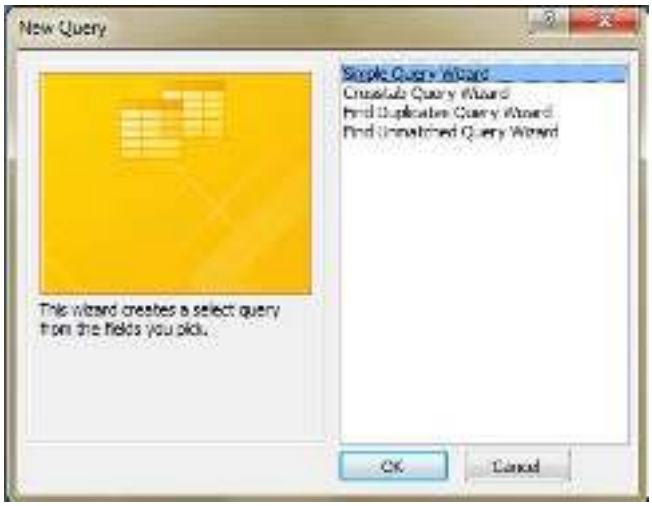


Figura nr. 12.3.1 Lista instrumentelor de tip wizard destinate interogărilor

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

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

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

Opţiunea *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 conţine condiţii.

Exemplul nr. 1, cel în care extrăgeam toţi clienţii din localitatea având codul 1002, nu poate fi realizat cu Simple Query Wizard deoarece presupune şi filtrarea datelor în aşa fel încât să se extragă doar o parte din clienţi.

Exemplul nr. 2

Să se obţină o listă cu toţi clienţii şi toate localităţile din care aceştia fac parte. Lista trebuie să conţină numele clienţilor şi numele localităţilor.

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 tabelelor 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 Clienţi se alege câmpul Denumire client şi din tabela Localităţi 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 obţine un produs cartezian (pe exemplu nostru, am obţine o mulţime care are un număr de rânduri egal cu numărul de clienţi înmulţit cu numărul de localităţi, 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 afişată fereastra din figura nr. 12.3.3.

Figura nr.12.3.3 Specificarea titlului interogării

După apăsarea butonului Finish, interogarea este afişată 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.

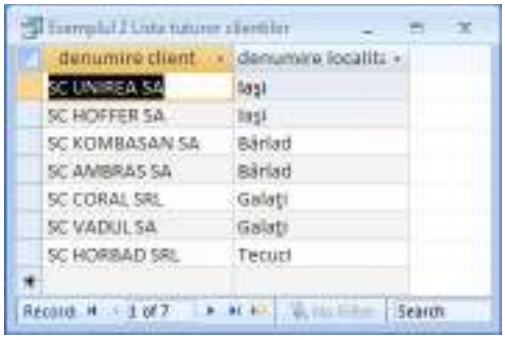


Figura nr. 12.3.4 Rezultatul interogării, exemplul nr. 2

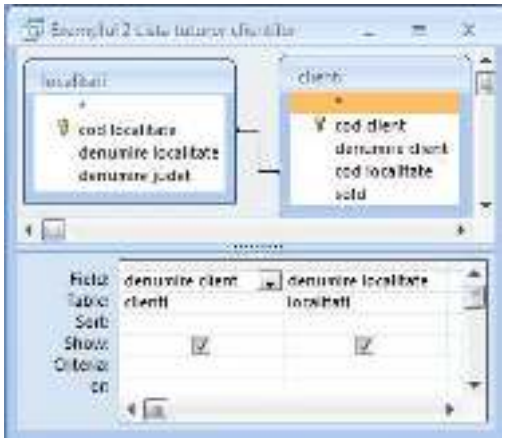


Figura nr. 12.3.5 Interogarea deschisă în Query Design

12.3.2 Crearea unei interogări cu Crosstab Query Wizard

Cu *Crosstab Query Wizard* pot fi create interogări încrucişate, ce au atât pe rânduri, cât şi 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 depăşit prin realizarea unei interogări intermediare care să extragă câmpurile necasare 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, clienţi – 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 conţină pe prima coloană clienţii şi câte o coloană pentru fiecare tip de document. La intersecţia fiecărui client cu fiecare tip de document trebuie să apară suma încasărilor.

Rezolvare

Se are în vedere următoarea succesiune de paşi:

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

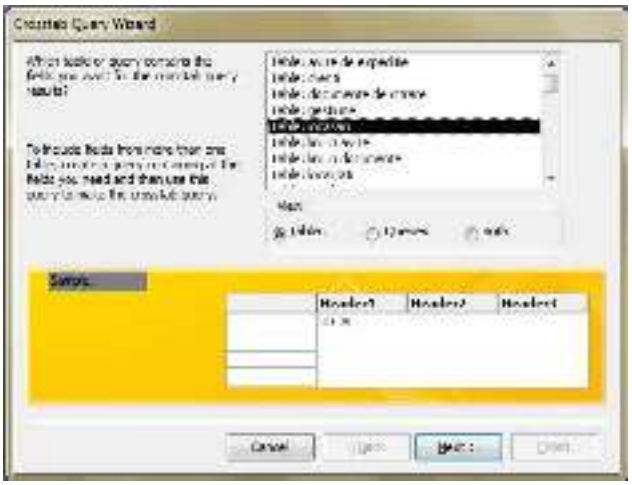


Figura nr.12.3.6 Prima fereastră de la Crosstab Query Wizard

- 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.

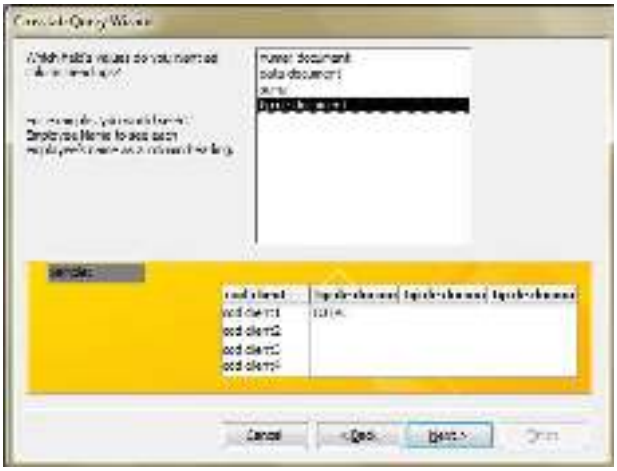


Figura nr. 12.3.8 Stabilirea dimensiunii orizontale a datelor (tip de document)

- Selectarea câmpului căruia i se aplică o funcţie agregată (funcţia Sum având ca argument câmpul Suma), ca în figura nr. 12.3.9. Tot la acest pas se poate alege apariţia unei eventuale coloane de total prin utilizarea opţiunii „*Yes, include row sums*".

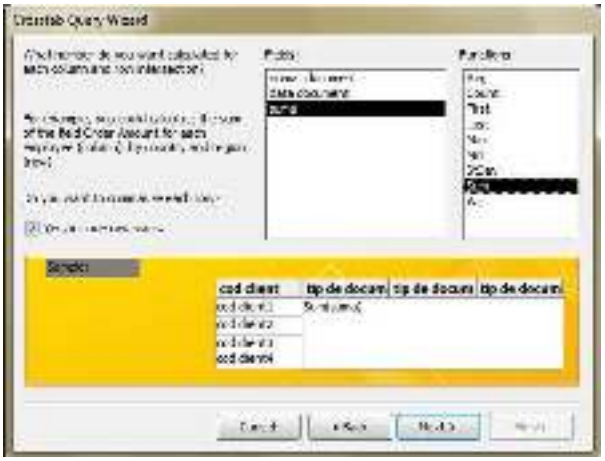


Figura nr. 12.3.9 Alegerea câmpului căruia i se aplică o funcţie agregat (SUM(Suma))

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

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 şi modificate în mod vizual, cu ajutorul utilitarului *Query Design*. El poate fi deschis din meniul *Create* → grupul *Other* → opţiunea *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 selecţie (Select Query);
- interogările încrucişate 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 ştergere de rânduri (Delete Query).

Tipul implicit de interogare este cel de selecţie, în aşa fel încât, dacă se doreşte realizarea unei interogări de un alt tip, imediat după specificarea tabelelor de inclus în interogare, trebuie specificat tipul interogării.

Tipul unei interogări poate fi stabilit din meniul principal, opţiunea *Design*, grupul *Query Type*, sau *din meniul ce se obţine 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 şi apoi dă în execuţie această comandă. Conţinutul comenzii SQL poate fi vizualizat cu opţiunea *SQL View*, ce o putem găsi atât în meniul principal, la submeniul *Design*, cât şi 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 selecţie);
- din ferestrele, rapoartele şi modulele VBA (Visual Basic for Applications) ale bazei de date.

12.4.1 Interogări de selecţie

Interogările de selecţie din Microsoft Access sunt obiecte ale bazei de date folosite pentru a extrage şi afişa o parte din datele din baza de date, care îndeplinesc anumite condiţii.

Exemplul nr. 4

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

Rezolvare

Se deschide Query Design şi 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 greşeală în plus în cadrul unei interogări, ea poate fi scoasă:

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

Dacă o tabelă nu a fost introdusă şi a fost închisă fereastra Show Table, aceasta din urmă poate fi redeschisă explicit dând click dreapta în spaţiul 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 *Clienţi*.

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 şi câmpurile folosite doar în stabilirea unor filtre (aşa cum sunt câmpurile Denumire client şi Cod localitate din exemplu);
- *Table* – tabelele din care fac parte câmpurile mai sus menţionate (tabela Clienţi 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* – condiţiile 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 opţional în cazul condiţiilor de egalitate.

Sub rândul *Criteria* pot fi specificate şi alte condiţii, ce vor fi despărţite de prima condiţie cu operatorul logic *OR* (sau). Expresiile de la zona *Criteria* pot conţine, pe acelaşi rând, mai multe condiţii despărţite 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 execuţie şi folosind opţiunea *Run* din meniul *Design.* Interogarea mai poate fi data în execuţie şi 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 execuţie (vezi figura nr. 12.4.2);
- *PivotTable View* - pentru vizualizarea unui tabel pivot cu datele obţinute în urma rulării interogării;
- *PivotChart View* - pentru vizualizarea unui grafic pivot cu datele obţinute în urma rulării interogării;
- *SQL View* - pentru vizualizarea comenzii SQL echivalente;

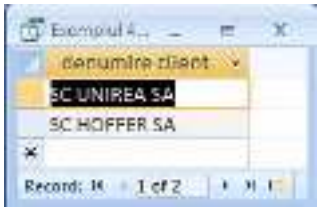


Figura nr. 12.4.2 Rezultatul interogării de la exemplul nr. 4

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

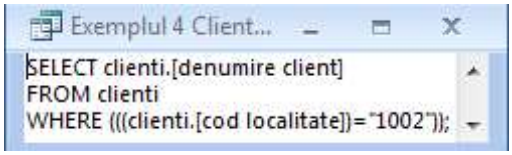


Figura nr. 12.4.3 Comanda SQL de la exemplul nr. 4

Exemplul nr. 4 poate fi rezolvat şi cu ajutorul parametrilor, în sensul că, interogarea poate propune introducerea codul localităţii pentru care doreşte o listă a clienţilor şi apoi, afişează rezultatul obţinut.

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

Pentru gestionarea parametrilor, se selectează opţiunea Parameters:

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

În fereastra Parameters se completează numele şi tipurile de dată ale parametrilor, în exemplul nostru Cod_Loc de tip text (acelaşi tip ca şi câmpul *cod localitate* din tabela *clienti*). Este de preferat ca parametrii să aibă nume diferite de cele ale câmpurilor tabelei sau tabelelor 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 execuţie a unei interogări cu parametri apare fereastra în care trebuie introdusă valoarea de atribuit parametrului pentru respectiva execuţie (1002 în cazul exemplului nr. 4):

Figura nr. 12.4.4 Fereastra specifică introducerii valorii unui parametru

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

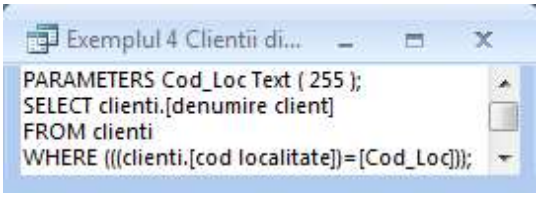


Figura nr. 12.4.5 Comanda SQL ce foloseşte parametri

Parametrii interogărilor pot fi definiţi şi implicit, prin simpla lor utilizare între paranteze drepte în zona Criteria. Şi la definirea implicită a parametrilor trebuie evitată confuzia de nume, ce poate să apară atunci când aceştia sunt denumiţi cu aceleaşi nume ca şi câmpurile tabelelor.

Tot în cadrul exemplului nr. 4, dacă dorim crearea unei tabele cu rezultatul obţinut 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 aceloraşi rubrici ca în cazul interogărilor normale de selecţie. Diferenţa constă în faptul că imediat după alegerea tipului de interogare Make-Table apare o fereastră de dialog în care trebuie introdus numele tabelei rezultat (Clienţi_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 ştergerea 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 opţiunea *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 construieşte tot o comandă **SELECT**, în care include clauza **INTO** pentru a specifica tabela rezultat.

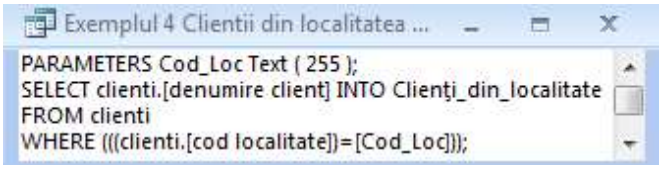


Figura nr. 12.4.7 Comanda SQL a unei interogări de creare a unei tabele

Exemplul nr. 5

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

Rezolvare

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

- Avize de expediţie (cel puţin pentru câmpul data aviz);
- Gestiune (pentru câmpul denumire gestiune);
- Clienţi (pentru câmpul denumire client);
- Localitaţi (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 (opţiunea *Field*);
- din ce tabele (opţiunea *Table*);
- cum sunt ordonate datele în rezultat (opţiunea *Sort*);
- dacă respectivul câmp apare sau nu în rezultat (opţiunea *Show*);
- condiţii de filtrare (zona *Criteria* alcătuită din rândul *Criteria* şi rânduri ce urmează).

Completarea câmpurilor interogărilor poate fi realizată:

- prin scrierea directă a numelor de atribute;
- prin alegerea lor din lista disponibilă la mutarea cursorului pe rândul Field (e mai uşor să se aleagă, mai întâi, numele tabelei şi 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 opţiunea Show şi, din acest motiv, pentru câmpurile ce nu apar în listă, ea trebuie deselectată.

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 neîncâpând toate în imagine.

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

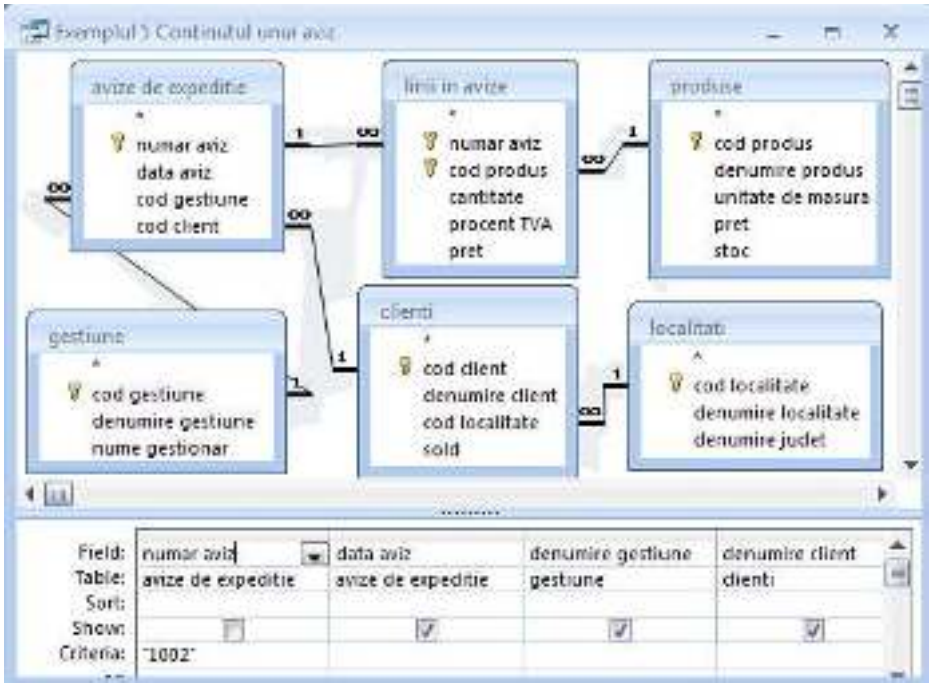


Figura nr. 12.4.8 Rezolvarea exemplului nr. 5 în Query Design

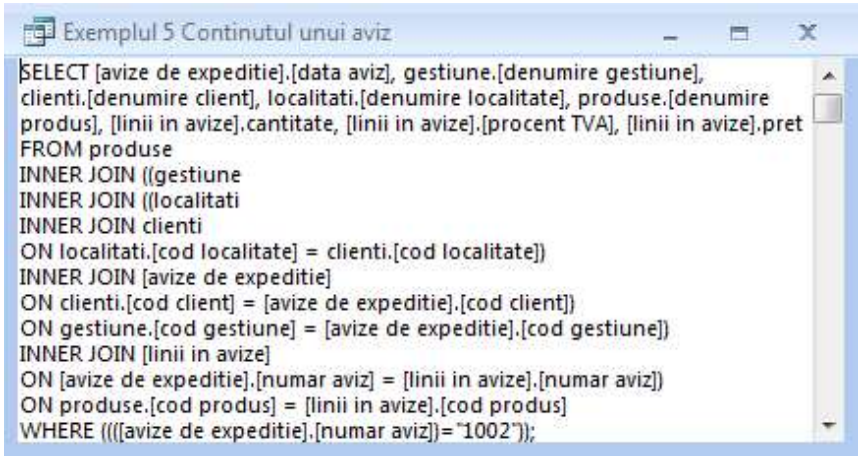


Figura nr. 12.4.9 Comanda SQL de la exemplul nr. 5

Reluarea exemplului nr. 3

Să se creeze o listă cu încasările de la fiecare client pe tipuri de documente. Lista trebuie să conţină pe prima coloană clienţii şi câte o coloană pentru fiecare tip de document. La intersecţia fiecărui client cu fiecare tip de document trebuie să apară suma încasărilor.

Rezolvarea exemplului nr. 3 cu ajutorul Query Design

În exemplul nr. 3 utilizam Crosstab Query Wizard pentru a obţine o listă cu încasările de la fiecare client pe tipuri de documente, în condiţiile în care, pe prima coloană, trebuie să avem toţi clienţii de la care s-a încasat ceva şi pe primul rând (cel de explicaţie a coloanelor) fiecare tip de document utilizat. La intersecţia fiecărui client cu fiecare tip de document trebuie să apară suma încasărilor respective.

Se creează o nouă interogare cu Query Design şi după introducerea tabelei Încasări, se alege opţiunea Crosstab Query din meniul principal, submeniul Design sau din meniul ce se obţine la click dreapta pe fereastra Query Design, submeniul Query Type.

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

La rubrica Crosstab se specifică dacă respectivele câmpuri sunt folosite:

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

La rubrica Total se completează:

- cu Group By pentru câmpurile cu explicaţii ale rândurilor şi ale coloanelor;
- cu o funcţie 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.

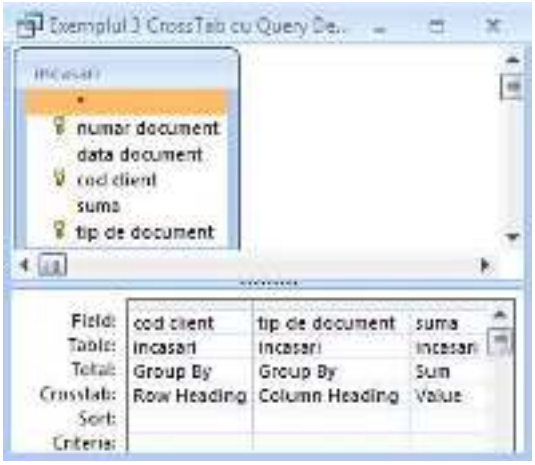


Figura nr. 12.4.10 Rezolvarea exemplului nr. 3 cu Query Design

12.4.2 Interogări de adăugare

Interogările de adăugare (Append Query) presupun completarea unor rubrici diferite de cele de la interogările de selecţie, deoarece Access construieşte, 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 şi 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 tabelei în care se vor introduce datele apare după selecţia tipului de interogare.

Exemplul nr. 6

Introduceţi, în tabela Clienţi, 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 şi Append Query). Selectăm în fereastra Append numele tabelei Clienţi.

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 conţin spaţii, 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 Clienţi 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 diferenţă faţă 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.

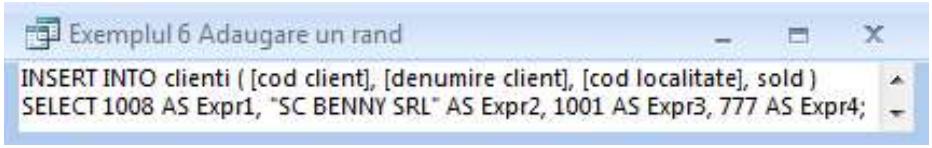


Figura nr. 12.4.12 Comanda SQL corespunzătoare exemplului nr. 6

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

12.4.3 Interogări de modificare

Interogările de modificare (Update Query), realizate cu Query Design, presupun completarea tuturor informaţiilor 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 şi, 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 şi 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.

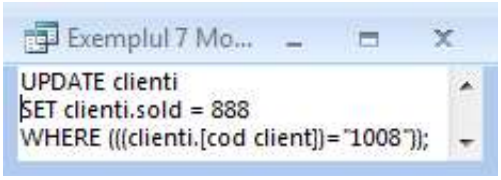


Figura nr. 12.4.14 Comanda SQL corespunzătoare exemplului nr. 7

12.4.4 Interogări de ştergere

Interogările de ştergere, realizate cu Query Design, presupun completarea tuturor informaţiilor necesare identificării rândului sau rândurilor de şters.

Exemplul nr. 8

Ştergeţi clientul care are codul 1008.

Rezolvare

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

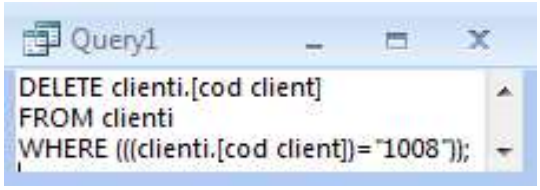


Figura nr. 12.4.16 Comanda SQL corespunzătoare exemplului nr. 8

Capitolul 13 Interogări SQL

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

13.1 Primele argumente ale interogărilor SQL

Formatul simplificat al unei interogări SQL de selecţie este:

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

Doar clauza SELECT a comenzii SELECT este obligatorie în Access. Celelalte argumente sunt opţionale, lista lor fiind mai mare decât cea prezentată în acest prim format simplificat. Amintim în acest sens şi 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 condiţii de selecţie, în funcţie de care sunt extrase doar o parte din rândurile tabelelor 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 şi 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 opţiunea 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 şi alegând opţiunea 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 clienţilor din localitatea ce are codul 1002. Lista va cuprinde numai denumirea clienţilor.

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

Figura nr. 13.1.1 Rezultatul interogării, exemplul nr. 1

Observaţii:

- Codul localităţii 1002 a fost scris ca o constantă de tip text, delimitat la stânga şi 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 şi caracterul *apostrof* (');
- dacă închidem şi redeschidem interogarea putem sesiza faptul că Access a adăugat un caracter *punct şi virgulă* (;) la interogare. Nu ne va deranja cu nimic în acest exemplu. Există situaţii, în practică, în care la interogări complexe Access modifică singur interogări corecte în interogări incorecte sintactic;
- numele de câmpuri şi de tabele care conţin spaţii se scriu între *paranteze drepte*.

Exemplul nr. 2

Care sunt clienţii 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
```

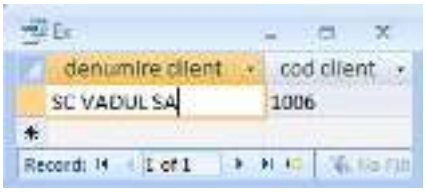


Figura nr. 13.1.3 Rezultatul interogării, exemplul nr. 2

Exemplul nr. 3

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

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

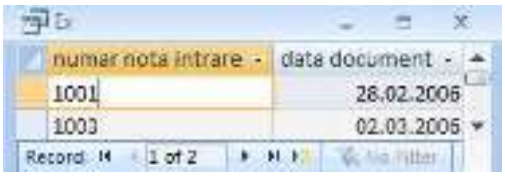


Figura nr. 13.1.4 Rezultatul interogării, exemplul nr. 3

Observaţii:

- data calendaristică s-a scris în format ISO: YYYY/MM/DD;
- fără a face nici o setare, interogarea ar fi putut folosi şi 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 şi caracterul liniuță (*minus*).

Exemplul nr. 4

Care sunt clienţii din judeţul ce are în câmpul [denumire judet] valoarea „IS”? Lista va cuprinde doar denumirile clienţilor.

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

Figura nr. 13.1.5 Rezultatul interogării, exemplul nr. 4

Observaţii:

- la clauza FROM pot fi scrise mai multe tabele. Lista de tabele foloseşte ca separator caracterul *virgulă*;
- când folosim date din mai multe tabele, între care există relaţii, la operatorul WHERE trebuie specificate aceste relaţii. O altă soluţie 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 acelaşi nume, la folosirea unui astfel de câmp se adaugă un prefix format din numele tabelei 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 **Clienti** a fost denumită mai scurt c, iar tabela **Localitati** l. Avantajele acestei facilităţi sunt vizibile mai ales în cazul interogărilor complexe. Pe lângă avantajul scrierii a mai puţine caractere, trebuie evidenţiată şi uşurinţa în urmărirea provenientei câmpurilor dacă alias-urile sunt folosite pentru toate câmpurile din interogare.

Exemplul nr. 5

Întocmiţi lista clienţilor din judeţul Iaşi în ordinea alfabetică a denumirii acestora. Lista va cuprinde: denumirea clientului şi 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]
```

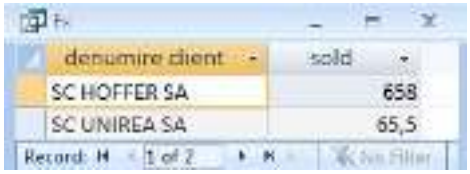


Figura nr. 13.1.7 Rezultatul interogării, exemplul nr. 5

Exemplul nr. 6

Întocmiţi lista clienţilor din judeţul Iaşi ordonată după valorile descrescătoare ale câmpului sold. Lista va cuprinde: denumire client şi 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
```

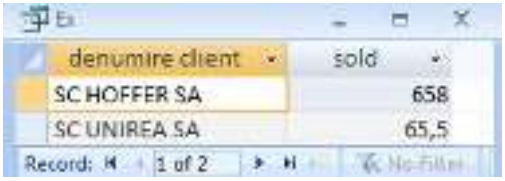


Figura nr. 13.1.8 Rezultatul interogării, exemplul nr. 6

13.2 Utilizarea principalilor operatori

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

Dialectul de SQL al Microsoft Access include şi 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 şi 10.000, de la clienţii 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'
```

Figura nr. 13.2.1 Rezultatul interogării, exemplul nr. 7

Observaţii:

- Operatorul BETWEEN poate fi folosit pentru mai multe tipuri de câmpuri (caracter, numeric, dată calendaristică etc.);
- Funcţia MID(*şir_de_caractere*, *caracterul_de_început*, *lungimea_subşirului*) returnează un subşir extras dintr-un şir iniţial. În cazul nostru, din şirul de caractere dat de denumirea clientului se extrage, începând cu primul caracter (parametrul al doilea = 1), un subşir format dintr-un singur caracter (parametrul al treilea = 1);
- Când e vorba de funcţii de prelucrare a şirurilor de caractere nucleul de SQL din Access preferă să folosească numele de funcţii 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 funcţiile din VBA pot fi folosite în interogările Access, inclusiv funcţiile VBA create de utilizatori.

Aceeaşi interogare mai poate fi scrisă şi 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'
```

Observaţii:

- Funcţia MONTH(*data*) returnează luna din datele calendaristice folosite ca argument;
- Funcţia YEAR(*data*) returnează anul din datele calendaristice folosite ca argument;
- Funcţia LEFT(*şir_de_caractere*, *lungimea_subşirului*) returnează un subşir dintr-un şir iniţial începând cu primul caracater din partea stângă. Funcţia LEFT echivalează cu funcţia 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 expediţie?

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

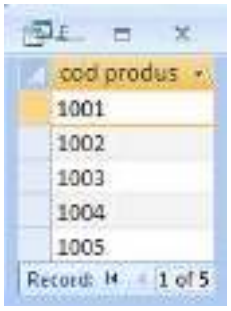


Figura nr. 13.2.2 Rezultatul interogării, exemplul nr. 8

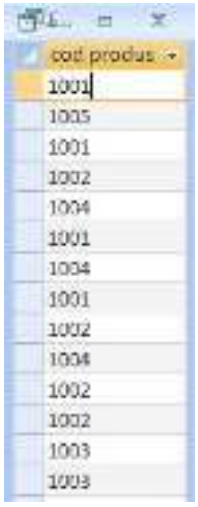


Figura nr.13.2.3 Rezultatul interogării în cazul în care s-ar fi folosit UNION ALL, exemplul 8

Observaţii:

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

se elimină automat liniile identice din rezultat. Dacă se doreşte obţinerea tuturor linilor din respectivele mulţimi, chiar dacă acestea se repetă, se foloseşte clauza ALL (vezi figura nr. 13.2.3).

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

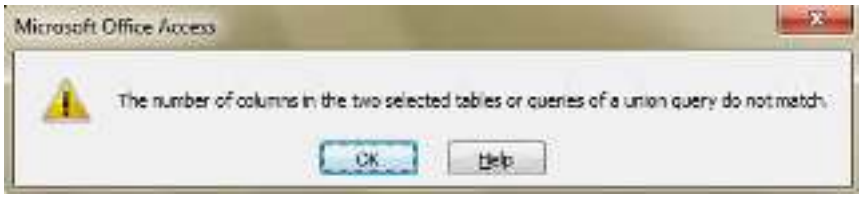
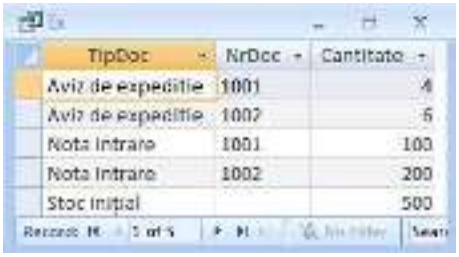


Figura nr. 13.2.4 Eroare ce apare când numărul de coloane de reunit nu este acelaşi

Exemplul nr. 9

Întocmiţi fişa produsului având codul 1001 (fişa produsului conţine stocul iniţial, intrările şi ieşirile pentru produsul respectiv).

```
SELECT 'Stoc initial' AS TipDoc, SPACE(8) AS NrDoc, Stoc AS Cantitate
FROM Produse
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 initial		500

Figura nr. 13.2.5 Rezultatul interogării, exemplul nr. 9

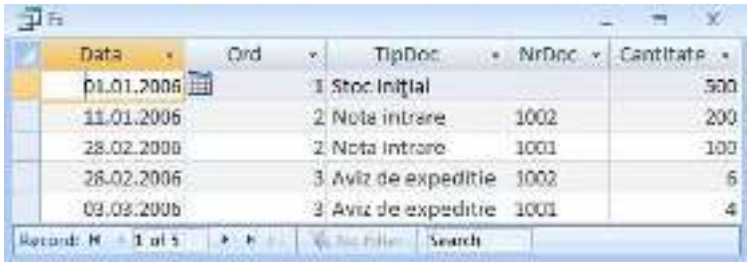
Observaţii:

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

Exemplul nr. 10

Întocmiţi fişa produsului având codul 1001, în care ordinea liniilor din rezultat să fie ordinea cronologică de apariţie a documentelor (prima linie va fi ocupată de soldul iniţial).

```
SELECT #2006-01-01# as Data, 1 as Ord, 'Stoc iniţial' AS TipDoc, SPACE(8) AS
NrDoc, Stoc AS Cantitate
FROM Produse
WHERE [Cod Produs] = '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 Produs] = '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 iniţial		500
11.01.2006	2	Nota intrare	1002	200
28.02.2006	2	Nota intrare	1001	100
26.02.2006	3	Aviz de expeditie	1002	6
09.03.2006	3	Aviz de expeditie	1001	4

Figura nr. 13.2.6 Rezultatul interogării, exemplul nr. 10

Observaţie:

Al doilea câmp (Ord) este folosit doar pentru ordonarea documentelor din aceeaşi dată în aşa fel încât mai întâi să apară documentele de intrare şi apoi avizele de expediţie.

Exemplul nr.11

Care sunt clienţii 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

Record: 1 of 2 Search

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

Observaţii:

- Operatorul LIKE permite căutarea unor şiruri de caractere;
- În sintaxa ANSI 89, caracterul *asterix* (*), inclus în formatul specificat cu operatorul LIKE, semnifică posibilitatea existenţei unui număr nelimitat de caractere, iar *semnul întrebării* (?) semnifică existenţa unui singur caracter;
- În sintaxa ANSI 92, caracterul *procent* (%) semnifică posibilitatea existenţei unui număr nelimitat de alte caractere, iar *liniuţa de subliniere* (_) semnifică existenţa 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 notaţie, aşa încât, o parte din acestea pot deveni incorecte;

Exemplul nr. 12

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

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

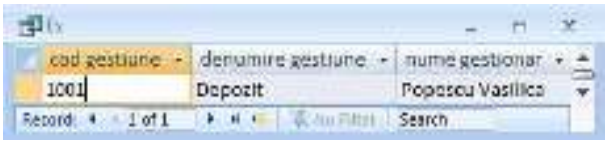


Figura nr. 13.2.8 - Rezultatul interogării, exemplul nr. 12

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')
```

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 Funcţii de agregare

Funcţiile 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 foloseşte o funcţie de agregare, dar lipseşte clauza GROUP BY (vezi paragraful următor), rezultatul va avea întotdeauna o singură linie.

Exemplul nr. 14

Câţi clienţi aveau soldul iniţial mai mare ca zero?

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

Figura nr. 13.3.1 Rezultatul interogării, exemplul nr. 14

Exemplul nr. 15

Câţi clienţi sunt din judeţul Iaşi?

```
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

Observaţii:

- Funcţia UCASE(*şir_caractere*) transformă un şir de caractere în majuscule; ea a fost folosită aici pentru că nu se ştie cum au fost scrise numele gestionarilor în baza de date. De exemplu, gestionarul căutat putea fi scris Popescu sau POPESCU;
- Dacă există mai mulţi gestionari cu numele Popescu atunci, pentru a obţine rezultatul dorit, mai avem nevoie şi de alte informaţii, 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'
```

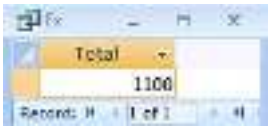


Figura nr.13.3.4 Rezultatul interogării, exemplul nr. 17

Exemplul nr. 18

Care a fost suma totală a soldurilor clienţilor?

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

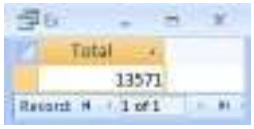


Figura nr.13.3.5 Rezultatul interogării, exemplul nr. 18

Exemplul nr. 19

Care este suma încasată de la clienţii din municipiul Iaşi?

```
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'
```

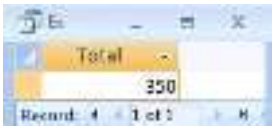


Figura nr.13.3.6 Rezultatul interogării, exemplul nr. 19

Alte soluţii pentru aceeaşi interogare:

1. folosind o subinterogare şi 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 şi 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 şi 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 funcţia IIF ce returnează una din două valori în funcţie de îndeplinirea sau neîndeplinirea unei anumite condiţii:

```
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
```

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
```

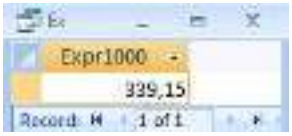


Figura nr. 13.3.8 Rezultatul interogării, exemplul nr. 21

Exemplul nr. 22

Care este totalul valorii producţiei 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
```

Figura nr. 13.3.9 Rezultatul interogării, exemplul nr. 22

Exemplul nr. 23

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

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

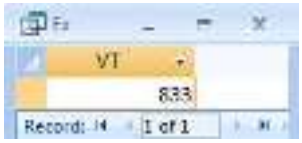


Figura nr. 13.3.10 Rezultatul interogării, exemplul nr. 23

Exemplul nr. 24

Ce cantitate de lapte s-a vândut pe piaţa ***municipiului*** *Iaşi 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*'
```

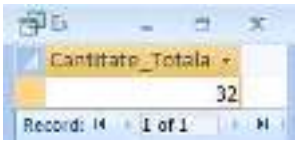


Figura nr. 13.3.11 Rezultatul interogării, exemplul nr. 24

Exemplul nr. 25

Care este valoarea medie a încasărilor?

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

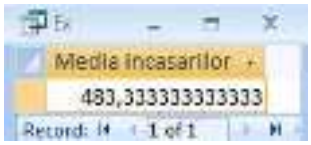


Figura nr. 13.3.12 Rezultatul interogării, exemplul nr. 25

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)
```

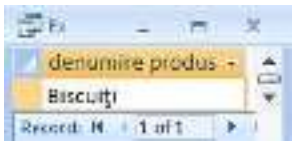


Figura nr.13.3.13 Rezultatul interogării, exemplul nr. 26

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
```

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 foloseşte în interogare cel puţin o funcţie de agregare. În exemplele ce conţineau funcţii de agregare, dar fără să conţină şi clauza GROUP BY, respectivele funcţii de agregare se executau o singură dată. Cu clauza GROUP BY putem face ca aceste funcţii 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 funcţii de agregare asupra grupului.

Exemplul nr. 28

Câţi clienţi sunt din fiecare localitate? Lista va cuprinde: codul localităţii şi numărul de clienţi.

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

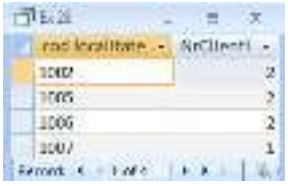


Figura nr.13.4.1 Rezultatul interogării, exemplul nr. 28

Observaţii:

- La clauza SELECT, pe lângă expresiile ce conţin funcţii 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.

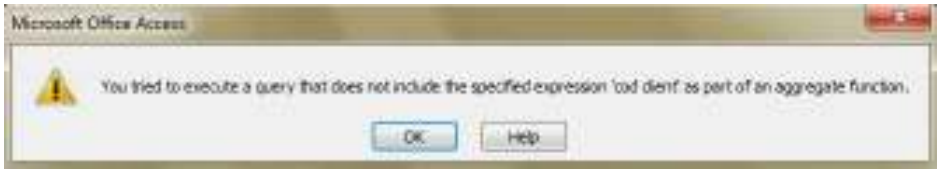


Figura nr.13.4.2 Eroare ce poate apare interogările cu funcţii de agregare

- În exemplul curent, funcţia de agregare COUNT se execută câte o dată pentru fiecare localitate întâlnită în tabela *Clienţi* şi NU câte o dată pentru fiecare localitate din tabela *Localităţi*.

Exemplul nr. 29

Câţi clienţi sunt din fiecare judeţ?

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

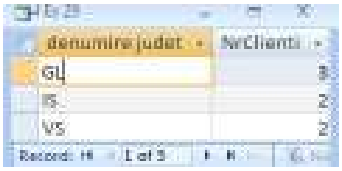


Figura nr.13.4.3 Rezultatul interogării, exemplul nr. 29

Exemplul nr. 30

Care sunt documentele de intrare cu cele mai multe linii (pot exista mai multe documente cu acelaşi 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]
))
```