Comenzi FoxPro

Sistem de gestiune a bazei de date *(SGBD – Data Base Management System) :*
-un produs software care asigură interacțiunea cu o bază de date, permițând definirea, consultarea şi actualizarea datelor din baza de date.
Baza de date *:*
-este un ansamblu structurat de date coerente, fără redondanță inutilă, astfel încât acestea pot fi prelucrate eficient de mai mulți utilizatori într-un mod concurent.

Operatii asupra unui tabel:

1.Crearea structurii unui tabel se realizează :
- prin meniu (*File* ↵ *New*);
-dupa crearea (sau deschiderea) unei baze de date, prin butonul *New Table* din *Database Designer*;
- prin comanda *CREA*[*TE*].

Pentru **deschiderea unui tabel** existent se foloseşte comanda:
USE <nume_fisier>.

Pentru **închiderea unui tabel** deschis în zona de lucru curentă se foloseşte comanda *USE* (fără parametri).

2.Introducerea de înregistrări într-un tabel se poate face :
-prin comanda *APPEND*;
-prin comanda *BROWSE* (cu această comandă se poate realiza orice operație de adăugare, modificare sau ştergere într-un tabel), urmată de selectarea opțiunii *Add New Record* din meniul *Table* (echivalenta cu combinatia de taste *Ctrl* + *Y*).

Pentru a completa un câmp de tip *memo:CTRL+Page Down* pentru deschidere şi *CTRL* + ↵ pentru închidere.

Pentru a completa un câmp de tip *general*:
- poziționare pe câmp; de 2 ori click;
- *Edit* ↵ *Insert Object*;
- se alege *Object Type* şi *OK* (de exemplu *paintbrush* sau *file*);
- se crează sau se alege desenul;
- *File* ↵ *Save As* ↵ (se da nume - *alfa.bmp* şi *OK*); *CLOSE*;

2.Modificarea structurii unui tabel se realizează:
- prin meniu (*View* ↵ *Table Designer*);
- prin comanda *MODI*[*FY*] *STRU*[*CTURE*];
- prin click dreapta pe tabel în *Database Designer* şi selectarea opțiunii *Modify*….

Obs: În mediul FoxPro, este suficienta introducerea primelor 4 caractere ale unei comenzi. În continuare, nu vom mai semnala acest lucru în mod explicit (prin [ ]).

Obs: Comanda *CLOSE ALL* determină închiderea bazei de date şi a tuturor tabelelor şi indecşilor deschişi.

3.Vizualizarea structurii unui tabel se realizează prin *LIST STRUCTURE*
DISPLAY STRUCTURE.

4.Ştergerea de înregistrări:
-ştergerea logică a înregistrărilor (înregistrările sunt doar marcate pentru ştergere), comanda *DELETE*;
- ştergerea fizică a înregistrărilor marcate, realizată prin comanda *PACK*.

Dacă se doreşte anularea marcajului efectuat prin *DELETE*, se utilizează comanda *RECALL ALL*.
Obs: *RECALL* anuleaza doar marcajul de stergere al înregistrarii curente.

Comanda *ZAP* sterge toate liniile din tabelul curent, dar structura tabelului se pastrează.

- **5.Comenzi utile:**

-DISPLAY | LIST STRUCTURE - afişează structura tabelului deschis în zona de lucru curentă.
-BROWSE, *LIST, DISPLAY ALL*- afişează conținutul tabelului deschis în zona de lucru curentă.
- *DISPLAY* -afişează înregistrarea curentă;
- *CHANGE*, *EDIT* - afişează câmpurile tabelului, în scopul editării
-GO{ n | TOP | BOTTOM} -poziționarea pe a *n*-a, prima, respectiv ultima înregistrare
- *SKIP n* - saltul peste *n* înregistrări; *n* poate fi număr negativ.
-REPLACE <câmp> WITH <valoare> FOR <conditie>
- permite actualizarea unei valori a unui câmp din tabel.
- *LOCATE FOR <conditie>* - caută înregistrarea care îndeplineste condiția

- **Funcții:**apelate precedându-le cu simbolul "?", pentru afişare. *Exemplu:* ? *reccount*()).

- *RECNO*() - furnizează numărul înregistrării curente din tabel.
- *RECCOUNT*() - furnizează numărul de înregistrări din tabel.
- *FOUND*() - întoarce *TRUE* sau *FALSE* după cum ultima comanda *LOCATE* lansată a găsit o înregistrare îndeplinind condiția cerută.
- *DATE()* - întoarce data curentă.
-{*LEFT | RIGHT*}(<*sir_caractere*>, *n*)
-întoarce cele mai din stânga, respectiv din dreapta, *n* caractere din argumentul *sir_caractere*.
-*SUM*(), *AVG*(), *COUNT*(), *MIN*(), *MAX*()
-întorc suma, media, numărul, minimul,maximul valorilor unei coloane
Calculul şi afişarea acestor valori se pot realiza :

calculate sum(sSalariu)

Obs: Dacă se dă comanda *SET TALK OFF*, *calculate* nu afişează nimic.
În acest caz, se poate proceda astfel: calculate sum(sSalariu) to x
?x

Obs: Există şi comenzile *COUNT, AVERAGE, SUM*.*exemplu count for <conditie> to x*
?x

Zone de lucru. Sortare şi indexare.

I. Zone de lucru

În aplicații, apare necesitatea de a lucra cu mai multe tabele în paralel. Pentru aceasta, mediul Foxpro dispune de zone de lucru, iar în fiecare dintre acestea se poate deschide câte un tabel. Aceste zone de lucru sunt numerotate cu numere naturale de la 1 la 32767, iar primele 10 pot fi referite şi prin literele A-J. De asemenea, o zonă de lucru poate fi referită prin tabelul deschis în cadrul acesteia.

Comenzi şi funcții utile :

- *USE <nume_tabel>* [*IN <zona>*] deschide tabelul *T* în zona de lucru *n* . Dacă nu se precizează zona de lucru, tabelul va fi deschis în zona curentă.
- *USE* (fără parametri) determină închiderea tabelului deschis în zona de lucru curentă.
- *SELECT n* | *alias_tabel* determină ca zona de lucru identificată prin *n* sau *alias_tabel* să devină zona curentă . Un *alias* este un nume atribuit unui tabel ; un tabel poate fi referit prin alias-ul său sau prin zona de lucru în care este deschis.
- *SELECT 0* selectează (determină să fie zonă curentă) zona de lucru *n*, unde *n* este cel mai mic număr de zonă nefolosită.
- *SELECT([0 | 1 | alias_tabel])*
  SELECT(0) - întoarce numărul zonei curente
  SELECT(1) - întoarce un număr *n*, unde *n* este cel mai mare număr corespunzător unei zone neutilizate.
  SELECT(alias_tabel) - întoarce numărul zonei în care este deschis tabelul cu aliasul respectiv.
  SELECT() - întoarce numărul zonei curente dacă *SET COMPATIBLE* este *OFF* şi numărul minim al zonei neutilizate dacă *SET COMPATIBLE* este *ON.*
- *USED([<zona>])* –întoarce *.T.* sau *.F.*, după cum zona de lucru specificată este utilizată sau nu (i.e. există un tabel deschis în aceasta). Dacă nu se specifică zona, este vorba de cea curentă.
- *DBF(alias_tabel | zona)* – întoarce numele tabelului care are *alias*-ul respectiv sau care este deschis în zona de lucru specificată.

Pentru observarea zonelor de lucru şi a tabelelor deschise în cadrul acestora, se utilizează opțiunea *Data Session* din meniul *Window*.

II. Sortarea tabelelor

Pentru situațiile practice, un tabel ordonat este mult mai folositor.
Ordonarea unui tabel presupune:

- alegerea unui câmp al tabelului drept criteriu de sortare;
- alegerea ordinii sortării: crescătoare sau descrescătoare.

Dacă există mai multe înregistrări pentru care câmpul-criteriu are aceeaşi valoare, ordinea lor poate să nu fie importantă, dar, de multe ori, se doreşte ca înregistrările cu aceeaşi valoare a câmpului criteriu sa fie ordonate la rândul lor după alt criteriu. De exemplu, la un examen de admitere, primul criteriu este nota obtinuta, iar al doilea, este numele concurenților.

Dezavantajele sortării:

- se creează noi fişiere pe disc;
- este costisitoare ca timp.

Avantajul sortării: căutarea foarte rapidă care se poate efectua într-un tabel sortat.

Comanda SORT ***are următoarea sintaxă:***

SORT TO <nume_fisier>

ON <camp1>[/A|/D][/C] [, <camp2>,…]

[ASCENDING|DESCENDING][<domeniu>]

[FOR…] [WHILE…] [FIELDS <lista_campuri>]

Efectul este crearea unui nou tabel cu numele specificat. În acest tabel sunt copiate înregistrările din tabelul activ, în ordinea valorilor din câmpul <*camp1*>, în ordine crescătoare. Înregistrările pentru care <*camp1*> are aceeaşi valoare sunt la rândul lor sortate dupa <*camp2*> , etc.

Implicit, sortarea după fiecare din aceste câmpuri se face în ordine crescătoare. Modul de sortare se poate preciza pentru fiecare câmp în parte, incluzând după numele său una din opțiunile "/A" (pentru sortare crescătoare) sau "/D" (pentru sortare descrescătoare).

Obs.: Pentru câmpurile de tip logic, se consideră ***.F. <.T.***

Nu se poate face sortare după câmp *memo*.

Dacă dorim ca un tabel să fie sortat după un câmp de tip caracter fără a se face distincție între literele mari şi mici, vom adăuga după câmpul respectiv opțiunea "***/C***".

Clauze:

- *ASCENDING*| *DESCENDING* determină ordinea de sortare pentru toate câmpurile-criteriu pentru care nu s-a specificat una din opțiunile *"/A"* sau *"/D".* Ordinea implicită este *ASCENDING;*
- *FIELDS <lista_campuri>* - din structura noului tabel vor face parte numai câmpurile specificate. Informația din celelalte câmpuri nu este copiată în noul tabel. Numele din listă se separă prin virgulă.
- *FIELDS LIKE<lista sabloane>* - în noul tabel sunt copiate câmpurile ale căror nume se potrivesc cu unul din şabloane.
- *FIELDS EXCEPT<lista sabloane>* - în noul tabel sunt copiate toate câmpurile în afară de cele ale căror nume se potrivesc cu unul din şabloane.
- <*Domeniu*> - poate fi *ALL, REST, NEXT <numar>* sau *RECORD <numar>* şi face ca în noul tabel creat să fie copiate înregistrările din domeniul specificat. Domeniul implicit este *ALL*- toate înregistrările sunt sortate.

Exemplu:

Fie tabelul *CLIENT (ccod, cnume, cprenume, coras, cadresa, ctelefon, cafaceri).* Sa se sorteze descrescator dupa cifra de afaceri.

use client
list
sort on cafaceri/D to fclient
use fclient
list

III. Indexarea tabelelor

Indexarea este posibilitatea de ***ordonare logică*** a unui tabel permițând parcurgerea într-o anumită ordine a înregistrărilor fişierului prin sistemul pointerilor.

Indexarea presupune extragerea din fiecare înregistrare a fişierului de bază, a cheii care determină ordinea. Această cheie împreună cu numărul logic al înregistrării respective vor fi aranjate într-un ***fişier index*** (simplu indexat, cu extensia .IDX).

Un index este în esență un tablou cu 2 coloane. O coloană se referă la înregistrarea din tabel prin valorile date de cheie, iar cealaltă coloană conține numărul înregistrării ce dă poziția inițială în tabel. Ca efect deci, un fişier index este o ***sortare virtuală a unui tabel***, în timp ce înregistrările acestuia rămân neschimbate.

Exemplu: Iniţial:

1	Oprean
2	Dragomir
3	Albu

Fişier sortat după nume:

1	Albu
2	Dragomir
3	Oprean

Fişier indexat dupa nume:

3	Albu
2	Dragomir
1	Oprean

Modul de lucru cu un tabel indexat este urmatorul:

1. mai întâi se creează fişierul index asociat tabelului (indexarea tabelului); se dau cheile şi criteriile de indexare;
2. dacă se doreşte utilizarea tabelului indexat anterior, se deschide tabelul şi o dată cu el se deschid şi fişierele index asociate, fie automat de FoxPro, fie manual de utilizator;
3. se realizează operațiile dorite asupra tabelului (stergere, modificare, listare), înregistrările fiind văzute în ordinea dată de indexul activ. Modificarea conținutului tabelului implică actualizarea automată a fişierelor index **deschise** pentru tabelul respectiv;
4. după ce se termină lucrul cu tabelul, acesta se închide împreuna cu fişierele index asociate.

➢ ***Fisierele index*** asociate unui tabel pot fi:

a. ***simplu indexate*** (fişiere index simple, cu extensia .IDX) ce contin un singur criteriu de ordonare;
b. ***multiplu indexate*** (fişiere index compuse, cu extensia .CDX), care memorează mai multe criterii de ordonare, doar unul singur fiind activ la un moment dat.

Comanda INDEX ***are următoarea sintaxă:***

INDEX ON <criteriu>

TO <fisier.idx> | TAG <nume_tag> [OF<Fisier.cdx>]

[FOR<conditie>] [COMPACT] [ASCENDING | DESCENDING] [UNIQUE] [ADDITIVE]

Clauze:

- *UNIQUE* Pentru cazul in care se produc valori duplicat pentru campul de indexare, Visual FoxPro va construi intotdeauna indecşi secundari. Daca dorim un index primar, atunci se specifica această clauză.
- *ADDITIVE* permite crearea unui fişier index pentru un tabel, în condițiile în care fişierele index deschise anterior rămân deschise.

➢ ***Fişierele compuse*** sunt de 2 tipuri:

- ***structurale***, deschise şi asociate automat tabelului odată cu deschiderea acestuia, folosind comanda *USE*. Ele au acelaşi nume cu tabelul, au extensia *.CDX* şi sunt create folosind clauza *TAG* fara *OF…*
- ***nestructurale***, nu sunt deschise automat cu tabelul. Au nume diferit de numele tabelului şi va fi specificat în clauza *OF…*

Cum se deschid sau se activează fişierele index?

- Odată cu deschiderea unui tabel, se pot **deschide şi fişiere index** asociate acestuia (care anterior au fost create cu o comanda *INDEX*):

USE [<fisier>|?]……
[INDEX <lista fisiere index>|?]
[ORDER [<expresie_numerica>|<fisier.IDX>|[TAG]<nume tag>[of<fisier.CDX>]]]
[ASCENDING|…]……

- Dacă nu se dă *ORDER*, atunci primul fişier index din listă va fi cel activ.

<*expresie_numerica*> este numărul de ordine al fişierului index simplu sau al tagului. Numărarea se face astfel:

- mai întâi fişierele.*IDX* (în ordinea din listă)
- apoi tag-urile din fişierele structurale (în ordinea definirii lor)
- apoi tag-urile din fişierele nestructurale în ordinea apariției lor.

- Dacă se doreşte deschiderea unor fişiere index pentru tabelul activ (după ce acesta a fost deschis) se utilizează comanda:

SET INDEX TO [<lista_fisiere_index> | ?]
[ORDER <expN> | <idx index file> | [TAG] <tag name> [OF <cdx file>]
[ASCENDING | DESCENDING]]
[ADDITIVE]

Se vor deschide toate fişierele din listă. Noua listă de fişiere index o va înlocui pe cea veche, dacă nu se precizează clauza *ADDITIVE*.

- Pentru a schimba ordinea de accesare, pentru a selecta un anumit fişier sau tag activ utilizăm:

SET ORDER TO [<expN1> | <idx index file> | [TAG] <tag name> [OF <cdx file>]
[IN <expN2> | <expC>]
[ASCENDING | DESCENDING]]

[IN <expN2> | <expC>] permite activarea unui index dintr-o altă zonă de lucru decât cea activă. Este posibilă şi folosirea de alias-uri în locul numărului zonei de lucru.

Cum se închid sau se dezactivează fişierele index?

- Pentru a închide toți indecşii, în afară de cel compus structural, vom da comanda:

SET INDEX TO

- *SET ORDER TO* dezactivează toți indecşii, fără a-i închide.

- *CLOSE INDEX* – închide toate fişierele index din zona de lucru curentă, în afara celui structural.

Fişierele index pot fi închise şi cu *CLOSE ALL, USE, CLOSE DATABASES*.

Vizualizarea indexului activ la un moment dat se realizează prin:

LIST STATUS sau

DISPLAY STATUS

Pot apărea situații când două sau mai multe înregistrări corespund la ***aceeaşi valoare a cheii de indexare***:

SET UNIQUE ON | *OFF* – din mulțimea de înregistrări cu aceeaşi valoare a cheii de indexare, va putea fi accesată numai prima dintre acestea.

Ce se întâmplă la modificarea conținutului unui tabel?

Modificarea conținutului unui tabel determină actualizarea fişierelor index deschise pentru tabelul respectiv, dar cele care nu sunt deschise în momentul executării modificărilor nu vor fi reactualizate, deci vor memora o stare anterioară a tabelului. Apar astfel discrepanțe între tabel şi fişierul index respectiv, care trebuie reactualizat cu noul conținut al tabelului, operație care poartă numele de reindexare. Reindexarea este necesară şi când se modifică tipul indexării (cu acces unic sau acces multiplu – *SET UNIQUE*). Comanda folosită este:

REINDEX [COMPACT] - determină reactualizarea tuturor fişierelor *.IDX* sau a tag-urilor din *.CDX* deschise curent pentru tabelul activ.

Funcții referitoare la indexarea tabelelor:

- *NDX(), CDX()*- dau numele fişierelor index deschise într-o zonă de lucru.
- Funcția *TAG* – întoarce numele unui index simplu sau al unei etichete dintr-un index compus.
- Numele fişierului simplu indexat activ sau al tag-ului activ cu funcția *ORDER.*
- Ordinea de accesare a înregistrărilor unui fişier indexat este determinată de cheia de indexare. Cheia de indexare se poate afla cu functia *KEY*.

Comenzi referitoare la indexare

a) trecerea de la un fişier simplu indexat la un tag dintr-un fişier index compus cu:
COPY INDEXES <lista de fisiere index> | ALL [TO <fisier.CDX>]
(tag-urile vor avea acelaşi nume cu fisierul *.IDX*)

b) operația inversă: transformare tag în fişier *.IDX*:
COPY TAG <nume tag> [of <fisier.CDX>] TO <fisier.IDX>

c) după b), tag-ul rămâne totuşi în lista de indecşi a tabelului. Pentru a şterge tag-ul din fişierul compus:
DELETE TAG……

Obs: Dacă dintr-un fişier compus se şterg toate tag-urile, fişierul este şters în întregime de pe disc.

d) căutarea unei înregistrări într-un tabel indexat se face cu:

- comanda *SEEK <expresie>*; dacă este găsită, pointerul se poziționează pe ea, iar *FOUND()* ia valoarea *.T.*; altfel pointerul se poziționează după ultima înregistrare.
- funcția *SEEK(expresie,[<expN>|<expC>])* întoarce o valoare logică: *.T.* dacă este găsită o înregistrare (prima din ele) pentru care valoarea cheii de indexare este egală cu expresia (din argumentul funcției); prin al doilea argument al funcției, se specifică tabelul în care se face căutarea (prin zona de lucru sau alias).

Obs: Funcția *SEEK* înlocuieşte combinația: comanda *SEEK* şi funcția *FOUND()*.

Observații:

1) Presupunem că avem 2 câmpuri numerice *cafaceri, cavere* şi dorim indexare după *cafaceri* şi in interiorul său după *cavere*.

```
use client
index on cafaceri+cavere to alfa
list
```

Sistemul adună valorile din câmpurile *cafaceri* şi *cavere*, iar apoi face indexarea, deci se obtine un rezultat eronat.
Operatorul "+" are un sens pentru câmpuri, valori numerice, şi alt sens pentru caractere.

Rezolvarea problemei se face prin: a) *sort on cafaceri, cavere to gama* *SAU*
b) *index on str(cafaceri)+str(cavere) to delta*

2) Litera mare este diferită de litera mică, prin urmare se pot folosi pentru căutări funcțiile:
UPPER – transforma literele mici in litere mari
LOWER –transforma literele mari in litere mici.

Relații între tabele. Programe. Filtre

II. Comanda BROWSE

Comanda *BROWSE* poate avea următoarele clauze:

i. FOR <conditie>
ii. FREEZE <camp> - permite modificarea doar a valorilor campului specificat
iii. NOAPPEND – interzice inserarea de inregistrari
iv. NODELETE – interzice marcarea pentru stergere in fereastra BROWSE
v. NOMODIFY, NOEDIT – interzic modificarea inregistrarilor existente, dar permit adaugarea si stergerea de inregistrari
vi. TITLE<string> - permite schimbarea titlului ferestrei BROWSE
vii. VALID <conditie> permite verificarea corectitudinii inregistrarilor introduse
viii. WHEN <conditie> - se evalueaza ori de cate ori pozitionam cursorul pe o noua inregistrare; in cazul in care conditia este adevarata, este permisa modificarea liniei respective, altfel nu.
ix. FIELDS – permite vizualizarea si modificarea numai unei parti din campuri; se pot crea si campuri "imaginare" (campuiri calculate)

Clauza FIELDS are la randul ei o serie de parametri:

x. :R – campul vizualizat nu poate fi modificat
xi. :V=<conditie> - permite efectuarea de validari suplimentare asupra datelor
xii. :E=<string> - mesajul de eroare ce va fi afisat in status bar in cazul in care conditia din :V este falsa.
xiii. :B=<lim_inf>, <lim_sup> - permite incadrarea corecta a datelor introduse.

III. Relații între tabele

Între tabele apar relații, adică legături orientate care determină ca orice repoziționare a cursorului în primul tabel (tabel-părinte) să atragă după sine o repoziționare a cursorului în cel de-al doilea tabel (tabel-copil). Pentru ca să se poată realiza o relație, este necesar ca între cele două tabele să existe un câmp comun.O relație în care unei înregistrări din tabelul părinte îi corespunde exact o înregistrare în tabelul copil, se numeşte relație one-to-one. O relație în care unei înregistrări din tabelul părinte îi pot corespunde zero, una sau mai multe înregistrări din tabelul copil relația se numeşte one-to-many.

- ***SET RELATION*** – creează relații one-to-one între tabele.

SET RELATION TO
[<expr1> *INTO* <expN1> | <expC1>
[, <expr2> *INTO* <expN2> | <expC2> ...]
[*ADDITIVE*]]

Exemplu: Indexati tabelul copil (*SALARIAT*) dupa câmpul comun (*sdep*):

INDEX ON sdep TO idep.idx

Următoarea secvență de instructiuni creează o relatie one-to-one :

```
CLOSE DATABASES
USE salariat IN 1 INDEX idep.idx      && tabel copil
USE departament IN 2                  && tabel parinte
SELECT 2
SET RELATION TO dcod INTO 1
select 1
BROWSE NOWAIT
SELECT 2
BROWSE NOWAIT
```

- ***SET SKIP*** – creează relații one-to-many. Comanda se utilizează împreună cu *SET RELATION*.

SET SKIP TO [<alias1>[, <alias2>] ...]
!!! Tabelul copil trebuie indexat dupa cheia externă (câmpul comun – *sdep*). Dacă indexarea se face prin index compus structural, înainte de crearea relației se va executa comanda:

INDEX ON sdep TAG depart ADDITIVE

Altfel, dacă se indexează prin index simplu, vezi exemplul anterior.

```
CLEAR
CLOSE DATABASES
USE departament in 0                  && tabel parinte
USE salariat ORDER TAG depart in 0     && tabel copil
SELECT departament
SET RELATION TO dcod INTO salariat              && stabilire relatie one-to one
SET SKIP TO salariat                        && relatie one-to-many
SELECT salariat
BROWSE NOWAIT
SELECT  departament
BROWSE NOWAIT
```

- ***RELATION()*** – Returnează câmpul pe care s-a facut legatura.

RELATION(<expN1>[, <expN2> | <expC>]). ExpN1 – numarul de ordine al relatiei. ExpC – alias-ul tabelului. ExpN2 – aria de lucru. Implicit foloseşte aria de lucru curentă

?relation(1)

- ***TARGET()*** – Returnează alias-ul tabelului care este ținta unei relații (specificat în clauza *SET RELATION* după *INTO*)

TARGET(<expN1>[, <expN2> | <expC>]) – la fel ca mai sus

? target(1)

Instrucțiuni FoxPro: *IF, FOR, DO WHILE, SCAN, DO CASE, EXIT, LOOP*. Comenzile *ACCEPT, WAIT, INPUT*.

a) *IF* <expresie_logica>
<set de instructiuni>
[*ELSE*
< set de instructiuni >]
ENDIF

b) *FOR <memvar> = <expN1> TO <expN2> [STEP <expN3>]*
<set de instructiuni>
[EXIT]
ENDFOR
memvar = variabilă de memorie; *expN*= expresie numerică.

Exemplu: (afişarea numerelor impare de la 1 la 10)

```
CLOSE DATABASES
CLEAR
FOR mcount = 1 TO 10 STEP 2
        ? mcount
ENDFOR
```

Exemplu: (afişarea numelor tipărite cu majuscule, utilizând *FOR*)

```
SET TALK OFF
USE SALARIAT
GO TOP
STORE RECNO() TO I   && STORE – comandă pentru atribuire; este utilă atunci când
                     && stocăm aceeaşi valoare în mai multe variabile care vor apărea
                     && în comandă separate prin virgule
STORE RECCOUNT() TO J
FOR mcount = I TO J
        GOTO mcount
        DISPLAY UPPER (snume)
ENDFOR
USE
```

c) *DO WHILE <expr_logica>*
<set instructiuni>
[LOOP]
[EXIT]
ENDDO

Exemplu: (afişarea primelor 10 numere Fibonacci)

```
CLOSE DATABASES
CLEAR
STORE 1 TO t0
STORE 1 TO t1
?'t0=',t0
?'t1=',t1
i = 1
DO WHILE i<10
    temp = t0 +  t1
    t0 = t1
    t1 = temp
    ?'t'+alltrim(str(i,2)),'=', str(t1,2)
     i=i+1
ENDDO
```

d) *DO CASE*

CASE <expL1> *&& expL* = expresie logică
<set instructiuni>
[CASE <expL2>
< set instructiuni >
...
CASE <expLN>
< set instructiuni >*]*
[OTHERWISE
< set instructiuni >*]*
ENDCASE

Exemplu: (stabilirea în funcție de luna curentă, a anotimpului în care ne aflăm)

```
STORE CMONTH(DATE( )) TO luna
DO CASE
        CASE INLIST(luna,'March','April','May')
                STORE 'Este primavara!' TO anotimp
        CASE INLIST(luna,'June','July','August')
                STORE 'Este vara!' TO anotimp
        CASE INLIST(luna,'September', 'October','November')
                STORE 'Este toamna!' TO anotimp
        OTHERWISE
                STORE 'Este iarna!' TO anotimp
ENDCASE
WAIT WINDOW anotimp NOWAIT
```

e) SCAN – mută pointerul in tabelul current si executa un bloc de comenzi pentru fiecare inregistrare care indeplineste conditiile.

SCAN

```
[<scope>]
[FOR <expL1>]
[WHILE <expL2>]
        [<set_instructiuni>]
[LOOP]
[EXIT]
ENDSCAN
```

Exemplu: (afişarea salariaților şi acordarea unei măriri de 250 la salariul celor din departamentul 30, care câştigă mai puțin de 500)

```
close database
clear
use salariat
go 1
display all
scan all for scopii>1
   if ssalariu<500
     REPLACE  salariat.ssalariu WITH 250
   endif
endscan
display all
```

f) *EXIT* apare in blocuri de tipul *DO WHILE, FOR* sau *SCAN*. Transfera controlul la instructiunea imediat urmatoare blocului.

g) *LOOP* apare in blocuri de tipul *DO WHILE, FOR* sau *SCAN.* Poate fi plasata oriunde in interiorul acestor blocuri. Transfera controlul direct clauzei *DO WHILE, FOR* sau *SCAN,* fără să se execute instrucțiunile care urmează după *LOOP*.

Aplicatie: Parcurgeti tabela de angajati. Sumati salariile celor care lucrează în departamentul 30 si au salariul mai mic decat media pe unitate. (Indicatie : calcul medie, parcurgere si pentru fiecare inregistrare, daca salariul e mai mare ca media sau nu lucrează în departamentul 30, salt la sfarsitul ciclului, altfel continuare cu calculul sumei)

h) *ACCEPT* – citire variabila de la tastatura (sir de caractere)

ACCEPT
[<expC>]
TO <memvar>

Exemplu:

```
ACCEPT 'Dati numele unui nou client: ' TO var_nume
? var_nume
```

i) WAIT [<expC>]
TO <memvar>
[WINDOW [NOWAIT]]

Exemplu :

```
WAIT WINDOW “numele noului client este: ”;
        + var_nume  NOWAIT
```

j) *INPUT* – intoduce date de la tastatura; asemănător cu *ACCEPT*

Exemplu :

```
INPUT ‘Dati numele noului client: ' TO var_nume
&&textul dat de la tastatura se scrie  intre ghilimele
?
```

4.Filtre (Query)

Interogari – modalitati de consultare a informatiilor din mai multe tabele. Se creaza o interogare (FILE -> NEW -> Query sau *CREATE QUERY* <nume_query>).

QBE (Query By Example) este un limbaj de interogare interactiv, care va permite sa spuneti sistemului ce doriti, si nu cum sa obtineti date din mai multe tabele (neprocedural). In linia de comanda poate scrie *CREATE QUERY* si se deschide editorul pentru query.

Reamintiti-va posibilitatile meniului *Query*, atunci cand este deschis un Query. Observați opțiunile de adăugare sau eliminare de tabele din interogare – clauza *FROM* (a instrucțiunii *SELECT* din *SQL*), condiții de filtrare a înregistrarilor din rezultat – clauza *WHERE*, lista câmpurilor de afişat – lista *SELECT*, clauza de ordonare *ORDER BY*, clauza de grupare *GROUP BY*, clauza *HAVING* care pune conditii asupra gruparilor, butonul 'View SQL' (din toolbar-ul Query Designer sau din meniu) unde se poate consulta comanda SQL generata de ecran, si optiunea 'Run Query' din meniul Query.

Comenzi şi funcții utile

SET PATH TO
CLOSE DATABASES | *TABLES* | *ALL*
EOF

RAPOARTE

Rapoartele reprezinta o modalitate de prezentare a datelor. Acestea contin informatii extrase din bazele de date si prezentate intr-o anumita forma pe ecran, la imprimanta sau intr-un fisier. Ca si ecranele, rapoartele pot fi create fie prin program, fie folosind generatorul de rapoarte.

- **Crearea unui raport** se poate realiza prin:
  - comanda ***CREATE REPORT <nume raport>***

CREATE REPORT ? – permite alegerea raportului pe care dorim sa îl modificăm din lista celor existente.

  - meniu: ***File -> New -> Report***

După aceasta, apare o fereastră care va conține modelul raportului. Fiecare model de raport este memorat într-un fişier cu extensia *.FRX*. Dacă un raport cu numele specificat există deja, el va fi deschis pentru modificare.

In bara de meniu superioara, va apărea submeniul ***Report***. Opțiunile submeniului ***View*** sunt şi ele "adaptate" lucrului cu rapoarte.

- **Modificarea unui raport** se realizează prin:
  - Comanda ***MODIFY REPORT <nume raport>***
  - Meniu: File -> Open -> Report.
- **Vizualizarea** înainte de tipărire a raportului se poate realiza prin opțiunea ***Preview*** a submeniului ***View***.
- **Tipărirea unui raport** se poate realiza prin:
  - ***Report -> Run Report*** (opțiunea *Run Report* a submeniului *Report*);
  - comanda ***REPORT FORM <nume_raport.frx> -*** rezultatul acestei ciomenzi este tipărirea pe ecran, în fereastra principală *Foxpro*, a înregistrărilor în formatul descries în fişierul *.FRX*.

Dintre clauzele acestei comenzi amintim:

- TO PRINTER – tipăreşte raportul la imprimantă.
- ENVIRONMENT – înainte de listarea informațiilor, este încărcat environment-ul salvat în fişierul .FRX. Clauza este utilă atunci când tabelele necesare raportului nu sunt deschise.
- TO FILE <nume> - efectuează tipărirea într-un fişier.
- NOCONSOLE - suprimă tipărirea raportului pe ecran.
- PREVIEW – deschide pagina Page Preview.
- <Domeniu>- selectează domeniul de înregistrări care vor fi listate (ALL, REST, NEXT <numar>, RECORD <numar>).
- FOR <conditie>, WHILE <conditie>.

- **Fereastra de configurare a rapoartelor** conține, la inceput, 3 secțiuni (benzi):
  - *Header* şi *Footer*: sunt aşezate la începutul, respectiv la sfârşitul fiecărei pagini şi pot conține informații ca: numărul de pagină, data creării raportului etc.
  - *Detail* este cea mai importantă, pentru că în ea va fi inclusă câte o dată pentru fiecare înregistrare care face parte din raport; această secțiune cuprinde informațiile de fond ale raportului.

Pe langa aceste benzi, in functie de optiunile utilizatorului, in raport pot fi incluse şi alte secțiuni, prin intermediul submeniului *Report*:

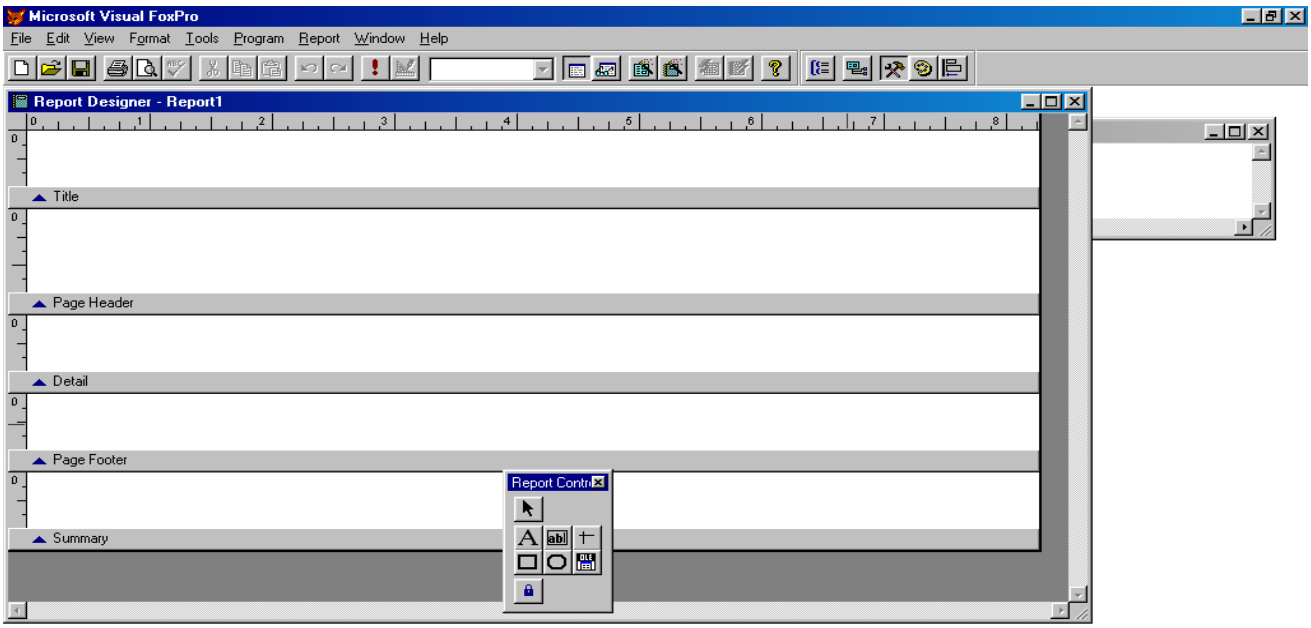
  - *Title* şi *Summary*: Inserarea uneia dintre ele (sau ambelor) se face prin submeniul *Report -> Title/Summary*. Secțiunea *Title* este inclusă o singură dată, la începutul raportului generat şi poate fi folosită pentru a afişa un titlu generic, data generării raportului etc.

Conținutul secțiunii Summary apare o singură dată, la sfârșitul raportului și în general conține date statistice: medii, sume, maxime și minime referitoare la datele din raport.

➢ *Group Header (continutul acestei benzi apare în partea superioara a fiecărui grup); Group Footer (continutul ei apare in partea inferioara a fiecarui grup),*

- Pentru a realiza un **raport grupat**, tabelul din care se extrag datele trebuie sa fie sortat sau indexat după cheia care dă criteriul de grupare.

Crearea unui raport grupat se realizează prin opțiunea ***Data Grouping*** din submeniul ***Report***. Aici se vor alege criteriile de grupare ale raportului, iar fiecărui grup îi vor corespunde benzi *Group Header, Group Footer* specifice.



- **Toolbar**-ul ***Report Controls*** conține cursorul de selecție, butonul de text (*Label*), butonul pentru introducerea câmpurilor (*Fields*), butonul pentru trasarea liniilor (*Line*), pentru crearea de dreptunghiuri (*Rectangle, Rounded Rectangle*), pentru inserarea de imagini (*Picture/ActiveX Bounded Control*). Controlul *Button Lock* permite crearea succesivă a mai multor controale de un anumit tip.

Unele caracteristici ale raportului se stabilesc prin intermediul instrumentelor din toolbar-ul ***Layout***, care se deschide prin optiunea ***Layout Toolbar*** din submeniul ***View***.

- **Crearea câmpurilor de intrare / ieşire**. Introducerea unui camp deschide o fereastră de dialog, în care se poate specifica:
  - expresia câmpului introdus
  - formatul de afișare a câmpului
  - valoarea statistică pe care o conține câmpul: prin butonul ***Calculations*** indicăm că nu dorim să reținem o simplă valoare în camp, ci o sumă, o medie, un maxim etc. Valoarea statistică poate fi calculată pentru întregul raport sau pentru fiecare pagină în parte.

- Condiția care trebuie îndeplinită pentru ca respectivul camp să fie afişat: câmpul poate fi vizibil doar în unele înregistrări, prin utilizarea butonului ***Print When…***

Forme (ecrane)

O ***formă*** permite schimbul de informații între utilizator şi sistemul Visual Foxpro, cu ajutorul unei interfețe familiare pentru vizualizarea şi introducerea datelor în baza de date. Forma afişează pe ecran, într-o anumită configurație, ferestre şi obiecte specifice, txt, linii, chenare, obiecte de control, asupra cărora acționează utilizatorul cu ajutorul tastaturii sau al mouse-ului, în vederea transmiterii opțiunilor sale sistemului.

Formele furnizează o mulțime vastă de ***obiecte*** care pot răspunde la ***evenimente*** (utilizator sau sistem) astfel încât managementul informației devine cât se poate de simplu şi intuitiv.

Un ***obiect*** este o instanță a unei clase, având astfel atât ***date*** cât şi ***metode***. De exemplu, un control (buton, listă etc.) într-o formă în execuție este un obiect.

Un eveniment este o acțiune, recunoscută de un obiect, pentru realizarea căruia se scrie un cod (program) corespunzător. Evenimentele sunt generate fie de către acțiunea utilizatorului (ex: click cu mouse-ul), fie de către system (cu timer-e).

Cum se creează o formă?

- se deschide fereastra de configurare a formei (*Form Designer*), în care se definesc obiectele care alcătuiesc ecranul şi se stabilesc caracteristicile globale ale acestuia. Această fereastră se deschide:
  - prin comanda ***CREATE FORM***;
  - prin meniul sistem: *File -> New -> Form*.
- se salvează fişierul *.SCX*.
- se rulează prin meniul *Program -> Do,* prin butonul *Run* de pe bara de instrumente sau prin comanda ***DO FORM*** *nume_fisier.scx*.

Ce pune la dispoziție fereastra Form Designer?

Zona gri a ferestrei este exact forma pe care o vom proiecta, deci dacă forma trebuie să aibă altă dimensiune este necesară redimensionarea acestei ferestre.

Deschiderea ***Form Designer*** conduce la apariția meniului ***Form*** în bara de meniu superioară şi la apariția unor opțiuni specifice în cadrul meniului ***View***.

În *Form designer* se pot deschide următoarele ***toolbar***-uri (ultimele 3 se pot deschide din meniul ***View***):

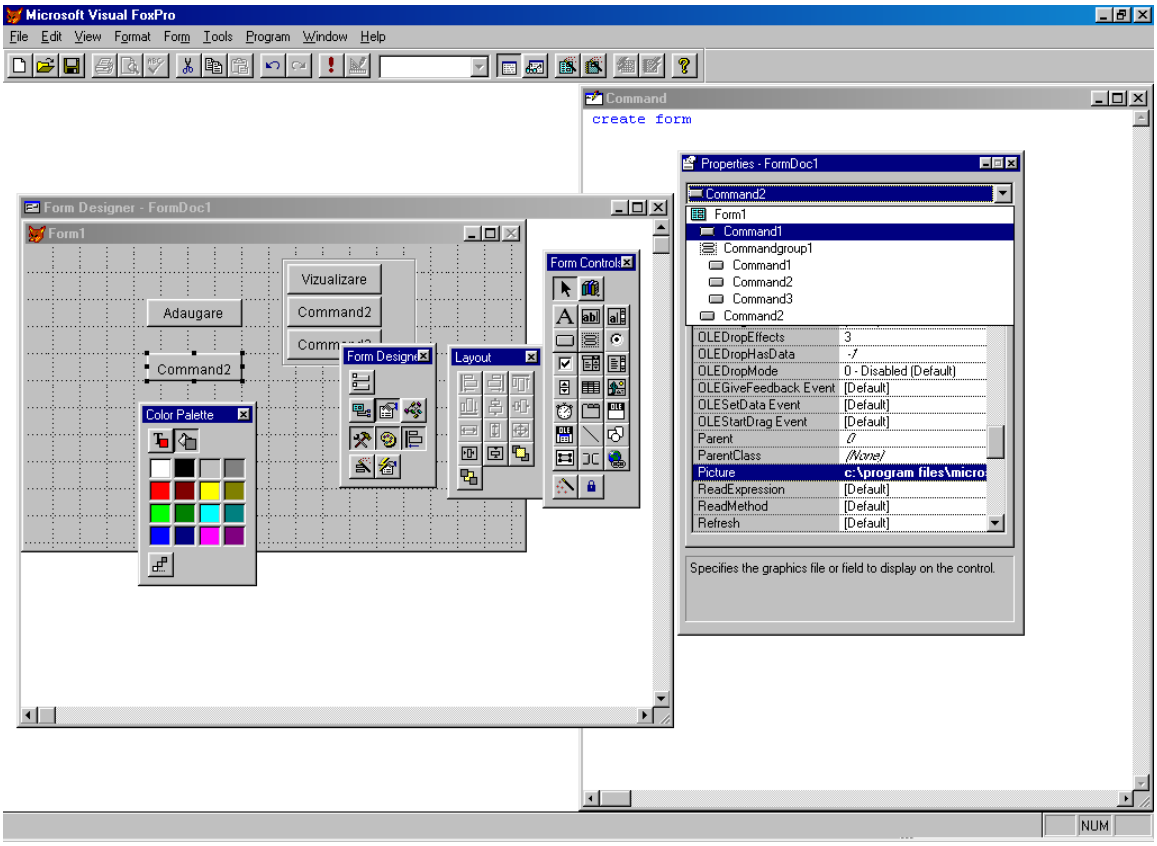
- ***Form designer;***
- ***Form Controls;***
- ***Layout;***
- ***Color Palette***

Toolbar-ul **Form Controls** *cuprinde diversele tipuri de obiecte pe care le putem insera în ecran:*

- text (*Label*);
- câmpuri de intrare/ieşire (*Fields*);
- linii (*Line*), dreptunghiuri (*Rectangle, Rounded Rectangle*), imagini (*Image*, *ActiveX Control*) diverse forme geometrice (*Shape*);
- butoane (*Command Button*) sau grupuri de butoane (*Command Group*);
- butoane radio (*Option Group*);
- *checkbox*-uri;
- *Combo Box*-uri;
- liste (*List Box*);
- *spinner*-e;
- grile (*Grid*);
- pagini (*Page Frame*).

În fereastra ***Properties*** a formei avem:

- o listă ascunsă în care putem vizualiza obiectele create;
- o listă a proprietăților obiectului selectat (toate – *All* sau grupate după specificul lor – *Data, Method, Layout, Other*). Obiectul selectat poate fi întreaga formă (caz în care se pot seta proprietăți generale ale acesteia, de exemplu titlul) sau un control din cadrul formei.



*Stabilirea mediului de lucru (*Data Environment*)*

Fiecare formă are un mediu de lucru asociat. Acesta este un obiect care include tabelele utilizate de formă şi relațiile între ele. Mediul de lucru poate fi proiectat în ***Data Environment Designer*** şi se salvează odată cu forma.

Data Environment Designer se deschide din meniul ***View***, iar apoi, prin *click dreapta* şi selectarea opțiunii ***Add*** se pot deschide tabelele necesare.

Relațiile dintre tabele se pot stabili prin *drag and drop* asupra câmpurilor comune (pe baza cărora se realizează relația), dinspre tabelul primar (care face referință la câmpurile altui tabel) către cel referit.

Controale şi containere

Containerele *pot conține alte containere sau controale, având rol de obiect părinte pentru alte obiecte. De exemplu, o formă este obiectul printe al unui obiect din forma respectivă.*

Controalele *pot fi plasate în containere, dar nu pot fi părinți ai altor obiecte. De exemplu, un buton nu poate conține un alt obiect.*

Exemple de containere:

- *grupurile de butoane – pot conține mai multe butoane;*
- *forma – poate conține obiecte de tip* page frame, grid, *orice control;*
- *obiectele* grid *– pot conține coloane;*
- *obiectele* option group *– conține mai multe butoane radio;*
- *obiectele* page frame ***-*** *conține pagini;*
- *obiectele* page *– obiecte* grid*, orice control.*

- *Butoane*

La crearea unui buton, vom avea nevoie de setarea următoarelor proprietăți:

- ***caption*** (mesajul afişat pe el);
- ***name*** - numele butonului: fiecare obiect trebuie să aibă un nume asociat prin care celelalte obiecte pot să comunice cu el;
- ***cancel*** – specifică dacă obiectul respectiv este unul de anulare;
- ***terminate read*** – specifică dacă forma este dezactivată la acționarea butonului (opțiunea este adecvată unui buton *Exit*);
- ***picture*** – fişierul sau câmpul grafic care ar urma să fie plasat pe control;
- putem face ca butonul să fie inițial dezactivat, urmând sa-l activăm ulterior (prin program);
- ***when event*** – putem specifica în ce condiții se permite apăsarea butonului. Codul scris trebuie să fie o funcție care returnează o valoare logică. Dacă valoarea întoarsă este .T., butonul se poate apăsa în mod obişnuit, altfel nu are nici un effect;
- ***valid event*** – explicăm ce se întâmplă atunci când este apăsat butonul;
- ***message event*** – apare încă pentru compatibilitate cu versiunile anterioare: rezultatul întors de codul scris la procedura *Message* va fi trecut în *status bar* la momentul apăsării butonului. În loc de acesta, se recomandă utilizarea proprietății ***StatusBarText***.
- este posibilă introducerea de comentarii, pentru claritate (*comment*).

Comenzi utile

➢ Comenzi pentru transferul datelor între variabile şi câmpuri:

IV. SCATTER MEMVAR [MEMO]

- transferă o înregistrare dintr-un tabel într-un grup de variabile de memorie;
- pentru fiecare câmp se creează cîte o variabilă de memorie, cu acelaşi nume, tip de dată, lungime şi conținut.

GATHER MEMVAR [MEMO]

- se atribuie fiecărui câmp al înregistrării curente valoarea variabilei cu acelaşi nume.

➢ ***Thisform.refresh***

- actualizează toate obiectele folosite de aplicație

Închiderea unei forme active

Pentru a permite utilizatorilor închiderea formei active prin click pe butonul *close* sau prin opțiunea close din meniu, se setează proprietatea *Closeable* a formei la *true* iar apoi se utilizează comanda ***RELEASE nume_forma***.

De exemplu, pentru un buton *Quit* codul ar putea fi:

V. THISFORM.Release

Obs: THIS, THISFORM se utilizează pentru a face referință la forma, respectiv la obiectul curent.

Ascunderea unei forme se realizează prin metoda ***HIDE***.

Indicație (idee pentru codul procedurii *Valid* la butoane):

➢ Butonul “Inceput”:

```
go top
scatter memvar
thisform.refresh
```

wait "Inceputul fisierului" window nowait

- Butonul “Precedent”

if bof()
wait "Inceputul fisierului" window nowait
else
skip-1
scatter memvar
thisform.refresh
endif

- Butonul “Urmator”

if eof()
wait "Sfarsitul fisierului" window nowait
else
skip+1
scatter memvar
thisform.refresh
endif

- Butonul “Ultimul”

go bottom
scatter memvar
thisform.refresh
wait "Sfarsitul fisierului" window nowait

- Butonul “Adaugare”

go bottom
append blank
scatter memvar memo
thisform.refresh

- Butonul “Stergere”

Pos=recno()
use &&inchidere tabel “salariat”
use salariat exclusive && redeschidere tabel in mod “exclusive”
&&(altfel nu e posibila stergerea)
go pos
delete
pack
if pos>1
go pos-1
else
go top
endif

- Butonul “Salvare”

if reccount() = 0
append blank
endif
gather memvar
thisform.refresh

- Butonul “Iesire”

thisform.release