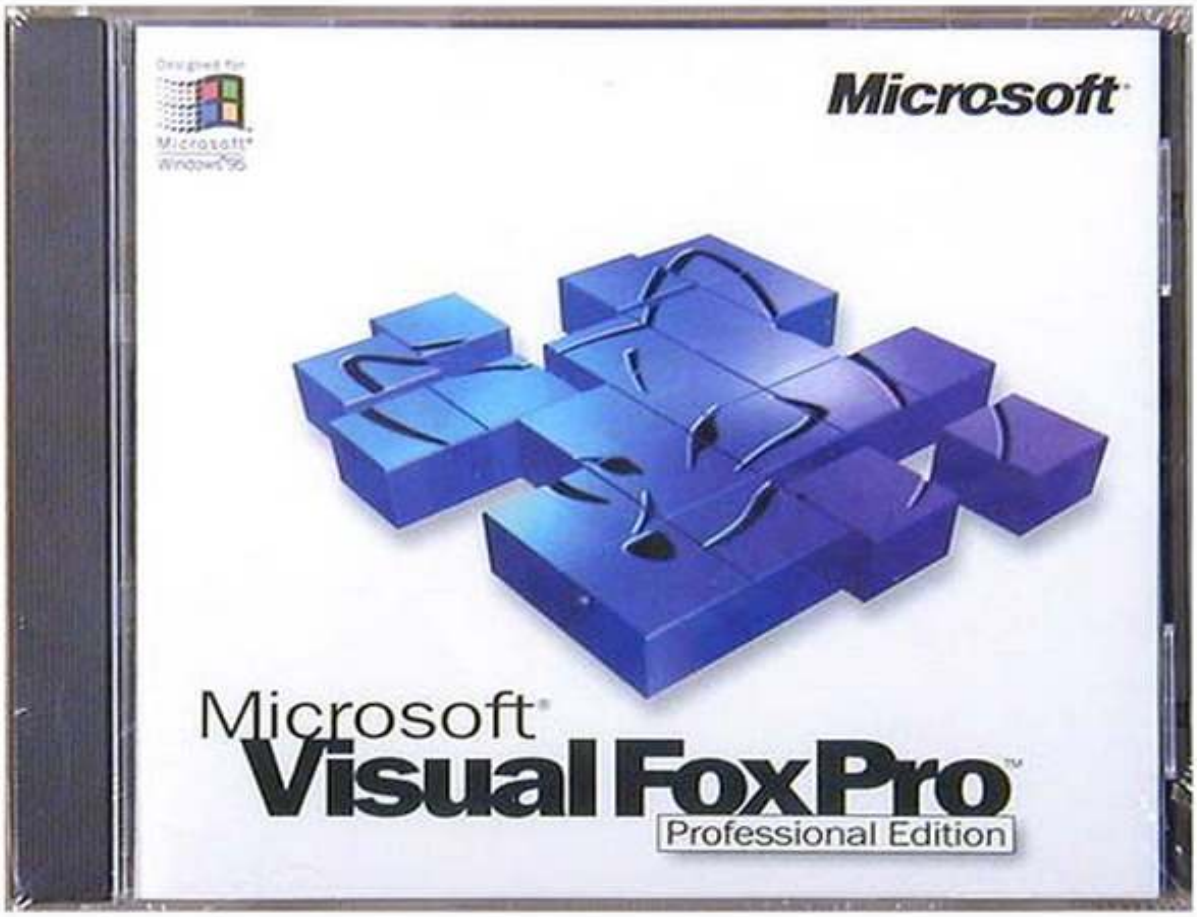
GIÁO TRÌNH
VISUAL FOXPRO



Mục lục

Chương 1:

GIỚI THIỆU VỀ HỆ QUẢN TRỊ CSDL VISUAL FOXPRO

1.1 Tổng quan về FoxPro và Visual FoxPro

1.1.1 Giới thiệu

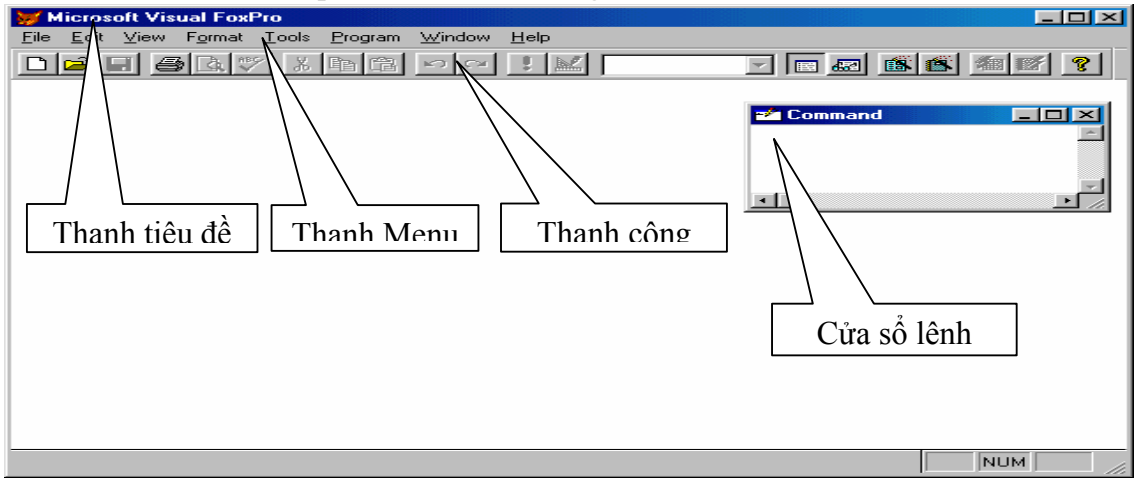
Foxpro là hệ quản trị cơ sở dữ liệu dùng để giải quyết các bài toán trong các lĩnh vực quản lý. FoxPro được thừa kế và phát triển trên phần mềm DBASE III PLUS và DBASE IV, những sản phẩm nổi tiếng của hãng ASTON-TATE. Khi các công cụ lập trình và các ứng dụng trên môi trường Windows ngày nhiều thì Microsoft cho ra đời các phiên bản FoxPro 2.6, chạy được trên hai môi trường DOS và Windows. Visual Foxpro là sản phẩm của hãng Microsoft, nó được kế thừa từ Foxpro for Windows, là một trong những công cụ tiện lợi để giải quyết các bài toán trong lĩnh vực quản lý cho những người chuyên nghiệp và không chuyên nghiệp. Từ khi phát triển đến nay, Hảng Microsoft đã cho ra đời nhiều phiên bản Visual Foxpro 3.0, 4.0, 5.0, 6.0.

1.1.2 Khởi động Visual Foxpro.

Sau khi đã cài đặt Visual FoxPro, ta có thể khởi động nó bằng cách thực hiện file FoxProw.exe hoặc file vfp.exe đối với Visual Foxpro theo các cách sau:

+ Kích chuột vào biểu tượng của FoxPro hoặc Visual Foxpro trên Desktop
+ Chọn menu Start/Program, chọn Microsoft Visual Foxpro và kích chuột vào đó.

Màn hình Visual Foxpro sau khi khởi động:



Visual FoxPro có 2 chế độ làm việc; chế độ tương tác (interactive) và chế độ chương trình (program).

Chế độ tương tác: Là chế độ trả lời từng câu lệnh một của người sử dụng, trong chế độ này có 2 hình thức đưa câu lệnh:

* Đưa câu lệnh qua menu hệ thống (system menu).

* Đưa câu lệnh từ cửa sổ lệnh (command window).

Chế độ chương trình: Các câu lệnh trong cửa sổ lệnh có thể tập trung thành một file và lưu trên đĩa (gọi là file chương trình nguồn). Khi muốn thực hiện các lệnh trong chương trình nầy, tại cửa sổ lệnh đưa vào các câu lệnh: DO < tên chương trình >

Để thoát khỏi Visual FoxPro, tại cửa sổ lệnh sử dụng lệnh QUIT

1.2 Các khái niệm cơ bản

1.2.1 Kiểu dữ liệu

Đối tượng xử lý của V. FOXPRO là dữ liệu, để quản lý và khai thác tốt các dữ liệu này, tuỳ theo tính chất, V.FOXPRO phải chia dữ liệu thành nhiều kiểu dữ liệu khác nhau: kiểu số (numberic), kiểu chuỗi (character), kiểu ngày tháng (date), kiểu lý luận (logical), kiểu bộ nhớ (memo), kiểu hình ảnh (picture).

a. **Kiểu số - Numeric (N)**: dùng để biểu diễn các số liệu mang giá trị số học và có nhu cầu tính toán như trong kế toán, quản lý, ... Mỗi dữ liệu kiểu số chiếm tối đa 20 chữ số gồm cả phần nguyên, phần thập phân và dấu chấm thập phân.

b. **Kiểu số - Float (F):** Dùng để biểu diễn số là các số có dấu chấm động như: 2.03e5 (2.03 x 10^5), thường được sử dụng trong các chương trình thuộc lĩnh vực khoa học kỹ thuật, ...

c. **Kiểu chuỗi - Charater (C):** Chứa các số liệu là tổ hợp một số bất kỳ các ký tự ASCII như tên, họ hoặc là số nhưng không có nhu cầu tính toán như số chứng minh, địa chỉ, số phòng, ... Mỗi dữ liệu kiểu chuỗi có độ dài tối đa 255 ký tự (mỗi ký tự chiếm 1 byte trong bộ nhớ).

d. **Kiểu ngày tháng - Data (D)**: Dùng cho những số liệu dạng ngày tháng như ngày sinh, ngày đến,.... Đó là những số nguyên dạng "yyyymmdd" khi hiển thị ra bên ngoài sẽ được chuyển thành dạng ngày tháng bình thường như mm-dd-yy, dd-mm-yyyy,... tuỳ theo yêu cầu của người

lập trình. Độ dài cố định của dữ liệu kiểu ngày là 8 ký tự.

e. **Kiểu logic - Logical (L)**: Dùng cho những dữ liệu chỉ có một trong hai trường hợp hoặc đúng (T) hoặc sai (F) như giới tính, đối tượng ưu tiên, ... Độ dài cố định của dữ liệu kiểu lý luận là 1 ký tự.

f. **Kiểu ghi nhớ - Memo (M):** Dữ liệu kiểu ghi nhớ là một đoạn văn bản có độ dài lớn hơn 255 ký tự, như khen thưởng, lý kịch, quá trình công tác,... Độ dài khai báo là 10 nhưng nội dung thực sự của kiểu ghi nhớ là tuỳ ý, chúng được lưu trữ trong một tập tin khác có cùng tên nhưng phần mở rộng là .FPT (FoxPro Text).

g. **Kiểu tổng quát - General (G)**: Dùng để chứa dữ liệu như bảng tính, âm thanh,...

h. **Kiểu hình ảnh - Ficture (P)**: Dữ liệu lưu dưới dạng hình ảnh .BMP, thường được dùng trong các chương trình "quản lý như sự", "nhận dạng",...

1.2.2 Các phép toán

a. **Phép toán số học**: Được thực hiện trên các dữ liệu kiểu số, gồm các phép toán:

Phép toán	Ý nghĩa	Ví dụ
-, +	dấu âm và dương	+5, -7
** hay ^	luỹ thừa	5**2,5^2
, /	nhân, chia	25, 5/7
%	phần dư (modulo)	25%5
+, -	cộng, trừ	10-2, 45+4

Độ ưu tiên các phép toán theo thứ tự đã nêu ở trên, có thể thay đổi thứ tự tính toán bằng cách đặt chúng trong 2 dấu ngoặc đơn ( ) như các quy tắc tính toán số học thông thường.

b. **Phép toán chuỗi**: Dùng để xử lý các dữ liệu kiểu chuỗi.

- Phép toán ghép nối (+): dùng để ghép 2 chuỗi cạnh nhau, kết quả của phép toán là một dữ liệu kiểu chuỗi.

Ví dụ: Trung tâm' + 'Tin học' -----> 'Trung tâm Tin học'

• Phép toán ghép nối (-): dùng để ghép 2 chuỗi cạnh nhau và di chuyển các dấu cách ở chuỗi thứ nhất (nếu có) ra cuối chuỗi tạo thành.

Ví dụ: 'Trung tâm ' - ' Tin học' -----> 'Trung tâm Tin học '

• Phép toán $: kiểm tra chuỗi bên trái có nằm trong chuỗi bên phải không. Kết quả của phép toán có kiểu logic.

Ví dụ: 'ab' $ "ABab" cho giá trị .T. nhưng 'ab $ "AaBb" cho giá trị .F.

c. **Phép toán ngày**: Hai dữ liệu kiểu ngày có thể trừ (-) cho nhau để cho khoảng cách đại số giữa 2 ngày.

Ví dụ: {01/08/2003} - {05/09/2003} -------> - 35

{01/08/2003} - {05/07/2003} -------> 25

Một dữ liệu kiểu ngày có thể cộng (+) hay trừ (-) một số nguyên để cho kết quả là một dữ liệu kiểu ngày.

Ví dụ: {01/08/2003}+ 10 -------> {11/08/2003}

{01/08/2003}- 20 -------> {12/07/2003}

Chú ý: • Hai dữ liệu kiểu ngày không thể cộng (+) cho nhau.

• Một số không thể trừ (-) với một dữ liệu kiểu ngày.

Việc diễn tả thứ tự ngày (D), tháng (M), năm (Y) trong một dữ liệu kiểu ngày còn phụ thuộc vào thời điểm hiện tại đang theo hệ thống ngày tháng nào.

(1) Lệnh SET DATE FRENCH |AMERICAN| JAPAN: Cho phép thiết lập dữ liệu dạng ngày theo kiểu Pháp|Mỹ|Nhật.

(2) SET CENTURY ON|OFF: Quy ước năm có một dữ liệu dạng ngày được biểu diễn theo dạng hai số (mặc định) hay dạng bốn số. Nếu SET CENTURY ON thì năm được biểu diễn theo dạng bốn con số, nếu SET CENTURY OFF (dạng mặc định) thì năm được biểu diễn theo dạng hai con số.

(3) Lệnh SET MARK TO <bthức C>: để ấn định ký tự phân cách ngày tháng, năm là <bthức C>. Dùng lệnh SET MARK TO để trở về ký tự phân cách ngày tháng mặc định.

d. **Phép toán quan hệ**: dùng để so sánh hai giá trị của hai biểu thức cùng kiểu

Phép toán	Ý nghĩa	Phép toán	Ý nghĩa
<	nhỏ hơn	<>, !	khác
>	lớn hơn	<=	nhỏ hơn hay bằng
=	bằng	=>	lớn hơn hay bằng
= =	bằng chính xác		

Hai dữ liệu kiểu số được so sánh dựa theo biểu diễn của chúng trên trục số.

Hai dữ liệu kiểu ngày được so sánh dựa theo biểu diễn của chúng theo chiều của thời gian.

Trong kiểu logic, Visual FoxPro quy ước: .T.<.F.

Hai dữ liệu kiểu chuỗi có độ dài bằng nhau được so sánh dựa theo nguyên tắc sau: đầu tiên so sánh 2 mã ASCII của 2 ký tự đầu của hai chuỗi, nếu bằng nhau thì so sánh tiếp.

Ví dụ: 'ABCD' < 'ABCE' ----------> .T. 'a' < 'A' ----------> .F.

Trường hợp hai chuỗi có độ dài khác nhau, thì việc so sánh dựa vào việc thiết lập môi trường SET EXACT ON/OFF, nghĩa là:

Nếu SET EXACT ON thì 'AB' = 'AB ' ----------> .F.

Nếu SET EXACT OFF thì 'ABCD' = 'AB' ----------> .T.

e. **Phép toán logic**: Visual FoxPro có 3 phép toán logic: NOT; AND; OR

NOT hay ! : phủ định của toán hạng theo sau.

AND : cho giá trị .T. nếu cả hai toán hạng đều .T.

OR : cho giá trị .F. nếu cả hai toán hạng đều .F.

1.2.3 Toán hạng

Toán hạng là các dữ liệu tham gia vào các phép toán.

Ví dụ: del=b^2 - 4*a*c thì b,2,4,a,c là các toán hạng.

1.2.4 Hằng

Là đại lượng có giá trị không đổi trong thời gian chương trình thực hiện. Trừ kiểu dữ liệu memo thì mỗi kiểu dữ liệu đều có hằng của nó.

Hằng kiểu số: như -2.5, 100, 4.14

Hằng kiểu chuỗi: hằng loại nầy phải để trong hai dấu "..." hoặc '...' hoặc [...], có độ dài tối đa không quá 253 kí tự.

Ví dụ: "abc"; tổng hợp', '123',.....

Hằng kiểu ngày: phải được đặt trong cặp dấu {...}

Ví dụ: {01/01/96}; {}: ngày rỗng.

Hằng logic: chỉ có 2 giá trị .T. và .F.

1.2.5 Biến

Biến là đại lượng dùng để lưu trữ dữ liệu trong quá trình tính toán. Biến có hai đặc trưng chính: tên biến và giá trị của biến. Tên biến được đặt theo nguyên tắc: dài không quá 10 kí tự, bắt đầu phải là chữ cái hoặc dấu _ phần còn lại là tổ hợp của bất kỳ các chữ cái, chữ số hoặc dấu _. Tên biến không nên đặt trùng tên các từ khoá của Visual FoxPro, tên biến có thể viết bằng chữ in hoa hay chữ thường. Visual FoxPro hiểu kiểu của biến là kiểu của giá trị mà nó đang mang. Số lượng tối đa của biến được phép sử dụng là 2048 biến.

Visual FoxPro chia biến làm 3 loại:

a. ***Biến bộ nhớ***: Gọi chung là biến, do người sử dụng tạo ra trong bộ nhớ, khi không sử dụng nữa có thể giải phóng để tiết kiệm bộ nhớ.

Ví dụ: hsl = 3.12

ngaysinh = {01/01/88}

b. ***Biến hệ thống***: Được tạo ra ngay từ khi khởi động Visual FoxPro. Có tên bắt đầu bằng dấu gạch nối (_) thường được sử dụng trong vấn đề in ấn, người sử dụng không thể giải phóng biến loại nầy.

c. ***Biến trường***: Tên các trường trong tập tin CSDL , nó chỉ có ý nghĩa khi tập tin chứa nó được mở ra để sử dụng.

Nếu có một biến đặt trùng với một biến trường thì biến trường được ưu tiên thực hiện trước.

Nếu tồn hại hai biến trường và biến bộ nhớ trùng tên nhau, để truy nhập đến chúng mà không sợ nhầm lẫn, bạn sử dụng quy cách sau cho biến bộ nhớ:

M.<tên trường> hay M -> <tên trường>

1.2.6 Hàm

Hàm là những đoạn chương trình được viết sẵn nhằm thực hiện một công việc nào đó. Các hàm nầy thường cho ra một giá trị, nhưng cũng có hàm chỉ thi hành một việc nào đó mà không cho ra một trị nào cả. Về hình thức hàm được đặc trưng bởi tên hàm và theo sau là cặp dấu ( ) dùng để bao các đối số, các đối số nầy đặt cách nhau bởi dấu phẩy. Một hàm có thể có nhiều đối số hoặc không có đối số nào cả nhưng phải có ( ) theo sau.

Ví dụ: Date ( ): cho biết ngày tháng năm hệ thống.

Sqrt(x): căn bậc 2 của x.

Có 2 loại hàm: Hàm có sẵn của Visual FoxPro và hàm tự tạo do người sử dụng tạo ra. Chúng ta sẽ nghiên cứu vấn đề nầy kỹ hơn ở chương sau.

1.2.7 Biểu thức

Biểu thức là tập hợp của một hay nhiều thành phần như hằng, hàm, biến, phép toán, dấu ngoặc tròn. Sau khi tính toán biểu thức sẽ cho một trị duy nhất. Trị của biểu thức thuộc về một trong 4 kiểu: N, C, D, L. Một biểu thức có thể rất phức tạp, trị của biểu thức được tính theo nguyên tắc sau:

* Trong ( ) tính trước, ngoài ( ) tính sau,

* Phép toán ưu tiên cao tính trước.

* Bên trái tính trước, bên phải tính sau.

1.2.8 Từ khoá

Từ khoá là những từ được Visual FoxPro sử dụng vào một mục đích riêng, người sử dụng không được đặt tên trùng với các từ khoá nầy. Thông thường từ khoá là những động từ động từ của lệnh thực hiện. Nếu từ khoá có nhiều hơn 4 ký tự thì khi sử dụng chỉ cần ghi 4 ký tự đầu.

Ví dụ: Câu lệnh MODIFY COMMAND LUONG.PRG có 2 từ khoá là MODIFY và COMMAND có thể viết gọn là: MODI COMM LUONG.PRG

1.2.9 Lệnh và chương trình

Lệnh là những yêu cầu để thực hiện một nhiệm vụ nào đó. Lệnh trong Visual FoxPro thường là một động từ, cũng có trường hợp là một kí hiệu như: !. ?, ... Tập hợp các lệnh nhằm đạt được một mục tiêu đề ra gọi là chương trình.

Trong Visual FoxPro có 3 cách để ban hành lệnh:

a. ***Dùng cửa sổ lệnh***:

Lệnh được đưa vào cửa sổ lệnh, sau khi ấn Enter lệnh được thi hành ngay. Thi hành xong một lệnh thì lệnh cũ được lưu lại trên cửa sổ lệnh có thể sử dụng cho lần sau. Cách nầy thường dùng trong những tính toán đơn giản để kiểm tra kết quả của lệnh.

b. *Dùng menu*:

Lệnh được ban hành bằng cách kích hoạt menu tương ứng, sau khi thi hành xong câu lệnh cũng được lưu lại trên cửa sổ lệnh. Cách nầy chỉ hạn chế trong một số lệnh thông thường trên tập tin CSDL.

c. Dùng chương trình: Soạn thảo trước một chương trình gồm nhiều lệnh thích hợp. Chương trình được lưu trên đĩa dưới tên một tập tin có phần mở rộng PRG. Để thực hiện chương trình này, tại cửa sổ lệnh đưa câu lệnh DO <tên file.PRG>. Sau khi ấn Enter chương trình được nạp vào bộ nhớ và từng lệnh được thực hiện theo thứ tự.

Bài thực hành chương 1

1. Giả sử có tập tin HSNV.DBF (có cấu trúc như đã mô tả ở bài 1, thực hành hai) trong đó có ít nhất 15 mẫu tin.

a. Dùng lệnh SORT để sắp xếp lại tập tin HSNV.DBF sang một tập tin mới HSNVSX.DBF theo chỉ tiêu: Các mẫu tin được sắp xếp theo từng đơn vị (giảm dần), trong mỗi đơn vị thứ tự tên, họ được sắp xếp tăng dần.

b. Mở tập tin HSNVSX.DBF

- Sử dụng lệnh LIST liệt kê các trường HOLOT, TEN, NGSINH, M_LUONG, MADV.
- Sử dụng lệnh USE để đóng tập tin lại.

c. Lập 3 tập tin chỉ mục: FMASO.IDX theo trường MASONV, FDONVI.IDX theo trường MADV, FLUONG.IDX theo trường M_LUONG giảm dần.

- Bằng cách thay thế tập tin chỉ mục chủ, hãy liệt kê các mẫu tin theo MASONV tăng dần, theo MADV tăng dần, theo M_LUONG giảm dần.

2. Trong tập tin HSNV.DBF

a. Dùng lệnh LOCATE:

- Tìm người có họ tên là 'LE VAN NAM' (giả sử có họ tên này trong tập tin HSNV.DBF). Dùng lệnh DISPLAY cho hiện nội dung của mẫu tin này, rồi dùng lệnh EDIT để sửa lại.

- Tìm những người ở phòng Hành chính (MADV='HC'), cho hiện đầy đủ thông tin của những người này.

- Tìm những người có mức lương > 310.

b. Dùng lệnh SEEK để tìm kiếm người có MASONV='TCH01' (giả sử mã này có trong tập tin HSNV.DBF). Cho hiện nội dung của mẫu tin này.

c. Cho biết địa chỉ của người có Họ tên là 'HO VAN HAO', sinh ngày 10/11/58 (bằng hai cách: LOCATE và SEEK).

---

1. CHƯƠNG 2: THAO TÁC VỚI BẢNG DỮ LIỆU

1.1. 2.1. KHÁI NIỆM

Bảng dữ liệu chứa dữ liệu theo dạng dòng và cột, mỗi dòng được gọi là một mẫu tin (record), mỗi cột được gọi là một trường (field) của bảng.

Mỗi bảng dữ liệu được lưu trử trên đĩa với tên file có phần mở rộng mặc định là DBF, mỗi bảng dữ liệu có hai phần: cấu trúc và nội dung của bảng.

Ví dụ: bảng nhân viên (nhanvien.dbf) có cấu trúc sau:

Fieldname	Type	Width	Decimal
Hoten	Character	30	
Gioitinh	Logic	1	
Ngaysinh	Date	8	
NamLV	Numberic	4	
Lylich	Memo	10	

Nội dung của NHANVIEN.DBF

Hoten	Gioitinh	Ngaysinh	NamLV	Lylich
Nguyen van A	.T.	10/15/75	1999	Memo
Le thi Nhan	.F.	06/15/70	1995	Memo
.....	.....	.....	.....	.....

1.2. 2.2 FILE VÀ KIỂU FILE TRONG VISUAL FOXPRO

2.2.1 Các kiểu file chính của Foxpro

FoxPro có các kiểu file sau:

*.dbf: File dữ liệu

*.idx: File chỉ mục

*.prg: File chương trình

*.dbc: File cơ sở dữ liệu

*.dll: File thư viện liến kết động

*.pjx: File dự án

*.scx: File Form

*.vcx: File thư viện

2.2.2. Cách tổ chức một file dữ liệu

a. File dữ liệu: Là tập hợp dữ liệu phản ánh về một tập hợp các đối tượng quản lý thông qua các thuộc tính của nó.

b. Bản ghi (Record): Là một bộ giá trị các thuộc tính phản ánh về một đối tượng quản lý.

c. Trường (Field): Là một thuộc tính trong file dữ liệu, mỗi trường được xác định bởi tên trường, kiểu trường và kích thước trường.

+ Tên trường (Field name): Tên trường dài tối đa 10 ký tự bao gồm chữ cái, chữ số, ký tự gạch dưới, ký tự đầu tiên của tên trường phải là chữ cái.

+ Kiểu trường (Field type): Kiểu trường có các dạng sau:

C: Charater	N:Numberic	L:Logic
D:Date	M:Memo	G:General

.....

+ Kích thước trường (Field Width): Là khoảng bộ nhớ cần thiết để lưu trữ các giá trị của trường, kích thước của trường phụ thuộc vào kiểu trường:

Kiểu C: Tối đa 254 Byte

Kiểu N: Tối đa 20 Byte kể cả dấu thập phân

Kiểu L: Chiếm 1 Byte

Kiểu D: Chiếm 8 Byte

Kiểu M: độ dài tuỳ ý, chiếm 10 Byte khi khai báo

Currency: Chiếm 8 byte

+ Cấu trúc file: Mỗi tổ hợp trường sắp xếp theo thứ tự nhất định gọi là cấu trúc của file dữ liệu, mỗi file dữ liệu chỉ có một cấu trúc cụ thể.

2.2.3. Nguyên tắc hoạt động

Chúng ta chỉ có thể truy nhập đến các phần tử của một file DBF nếu file đó đã được mở bằng lệnh USE <tên file DBF>.

Ở mỗi thời điểm bất kỳ, mỗi file DBF đang mở sẽ có một mẫu tin hiện thời, mẫu tin hiện thời

là mẫu tin có thể truy nhập vào thời điểm đó. Mẫu tin hiện thời được trỏ đến bởi con trỏ mẫu tin (record pointer). Mỗi mẫu tin đang mở có 2 vị trí đặc biệt chú ý: đầu file và cuối file. Để biết được con trỏ mẫu tin ở đầu hay ở cuối file ta dùng các hàm logic sau:

. Hàm BOF( ) (begin of file) cho giá trị .T. nếu con trỏ mẫu tin cuối file DBF đang mở, ngược lại hàm cho giá trị .F.

. Hàm BOF( ) (end of file) cho giá trị .T. nếu con trỏ mẫu tin cuối file DBF đang mở, ngược lại hàm cho giá trị .F.

. Số thứ tự của mẫu tin (record number - recno): mô tả số thứ tự vật lý của mẫu tin trong tập tin cơ sở dữ liệu DBF. Số thứ tự nầy do FoxPro qui định một cách tuần tự, được đánh số từ 1 đến mẫu tin cuối cùng. Trong khi làm việc, nếu xoá mọt mẫu tin thì số thứ tự này cũng tự động được cập nhật theo cho phù hợp.

. Hàm RECOUNT( ) dùng để biết số mẫu tin của một tập tin DBF đang mở.

. Hàm RECSIZE( ) dùng để biết được kích thước của một mẫu tin.

. Hàm RECNO( ) cho biết số thứ tự của mẫu tin hiện thời.

. Kích thước của các mẫu tin của một file DBF đều bằng nhau.

1.3. 2.3. CÁC LỆNH CƠ BẢN TRÊN FILE DBF

1.4. 2.3.1 Dạng lệnh tổng quát

Lệnh là một chỉ thị cho máy thực hiện một thao tác cụ thể. Một lệnh trong Foxpro nói chung có cú pháp tổng quát như sau:

Lệnh [phạm vi] [FIELDS <dsách trường>] [FOR <btL1>] [WHILE <btL2>] [FROM <tên file> / ARRAY <tên mảng>] [TO print/tên file/dsách biến]

Trong đó,

Lệnh: một từ khoá, cho biết mục đích của công việc, phải viết đầu tiên và có thể viết 4 kí tự đầu nếu lệnh có nhiều hơn 4 ký tự.

Ví dụ: DISPLAY FIELDS HOTEN, HSLUONG

DISP FIEL HOTEN, HSLUONG

Phạm vi (Scope): chỉ định phạm vi các mẫu tin chịu sự tác động của lệnh, phạm vi có thể là:

- ALL: tất cả các mẫu tin trong file dữ liệu đều bị tác động của lệnh (nếu có sử dụng FOR thì phạm vi được hiểu là ALL).
- NEXT <n>: n mẫu tin tiếp theo tính từ mẫu tin hiện thời bị tác động của lệnh.
- RECORD <n> Lệnh chỉ tác động đến mẫu tin thứ n
- REST Lệnh sẽ tác động từ mẫu tin hiện thời cho đến hết.

FIELDS <dsách trường>: lệnh chỉ có tác dụng trên những trường có tên được nêu trong <dsách trường>.

FOR <btL1>: mẫu tin nào thoả mãn <btL1> mới bị tác động bởi lệnh.

WHILE <btL2>: chừng nào <btL2> còn đúng thì lệnh còn hiệu lực. Nghĩa là, lệnh sẽ tác động lên các bản ghi thoả mãn biểu thức logic đi kèm (có giá trị là .T.) cho đến khi gặp một bản ghi không thoả mãn biểu thức logic (có giá trị .F.) hoặc đến hết file dữ liệu. Nếu điều kiện sai thì lệnh được dừng ngay. Trong lệnh nếu vừa có FOR vừa có WHILE thì mệnh đề WHILE ưu tiên thực hiện trước.

FROM <tên file>: tên của file mà từ đó lệnh lấy số liệu để sử dụng cho file DBF đang mở.

TO PRINT/tên file/dsách biến: chuyển kết quả sau khi thực hiện lệnh đến máy in/file/biến.

Các mệnh đề theo sau lệnh có mặt hay không tuỳ trường hợp và không cần phải viết theo thứ tự như đã nêu.

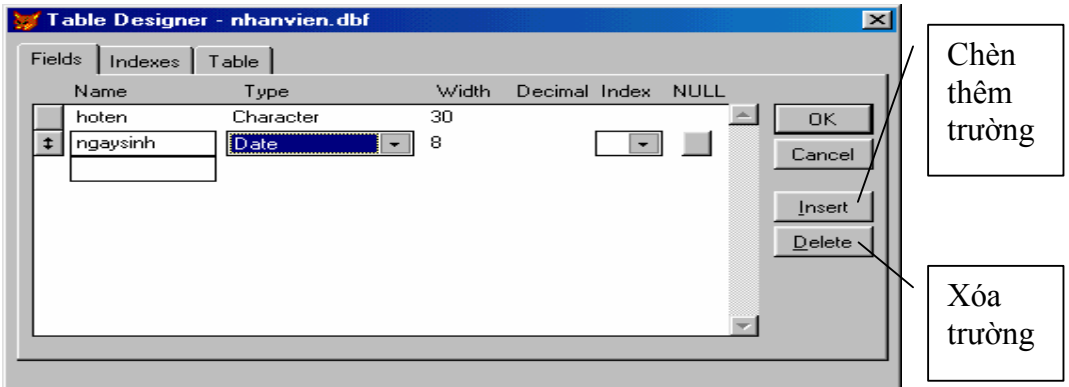
1.5. 2.3.2 Tạo bảng dữ liệu

Cú pháp: Create <tên file DBF> ↵ hoặc

Chọn File/New/<chọn loại file dữ liệu DBF>

Ví dụ: create nhanvien ↵ && tạo bảng nhanvien

Lúc này, màn hình sẽ xuất hiện hộp thoại để ta tạo cấu trúc bảng



Trong đó: Name: Tên trường

Type: Kiểu trường

Width: Độ rộng của trường

Decimal: Số chữ số lẻ sau phần dấu chấm thập phân, phần này chỉ sử dụng cho dữ liệu kiểu số.

Chú ý: - Các tên trường không được trùng nhau, không được trùng với từ khoá.

- Đối với dữ liệu kiểu số nếu có phần thập phân thì độ rộng của phần thập phân phải nhỏ hơn độ rộng của trường ít nhất là 2 đơn vị.

Để kết thúc việc nhập cấu trúc ta ấn đồng thời phím Ctrl+W, lúc này sẽ nhận được hộp thoại

Lúc này: trả lời <No> thì sẽ quay lại cửa sổ lệnh, trả lời <Yes> để tiến hành nhập các bản ghi. Khi chọn Yes sẽ tiếp tục xuất hiện hộp thoại để nhập dữ liệu.

Khi kết thúc việc nhập dữ liệu, nhấn tổ hợp <Ctrl+W> để lưu file dữ liệu lên đĩa. Khi đó file dữ liệu sẽ có dạng <tên file>.DBF

Chú ý: Để nhập dữ liệu cho trường MEMO, ta đưa con trỏ đến hộp memo rồi nhấn tổ hợp phím Ctrl_PgUp, lúc đó sẽ xuất hiện cửa sổ nhập dữ liệu cho trường này. Sau khi kết thúc việc nhập dữ liệu cho nó, ta ấn tổ hợp Ctrl+W để ghi lại.

1.6. 2.3.3 Định vị con trỏ đến một bản ghi

a. Định vị tuyệt đối

Cỳ phỏp: GO <N>|[TOP]|[BOTTOM]

Tác dụng: Dùng để chuyển con trỏ bản ghi đến bản ghi có số hiệu <n> được chi định trong câu

lệnh.

+ GO TOP: Dùng để chuyển con trỏ bản ghi về đầu file dữ liệu.

+ GO BOTTOM: Dùng để chuyển con trỏ bản ghi về cuối file dữ liệu.

b.Định vị tương đối

Cú pháp: Skip [+|-] [<n>]

Tác dụng: Di chuyển con trỏ bản ghi về trước (-) hay sau (+) so với bản ghi hiện thời.

Chú ý: Khi chỉ gõ lệnh Skip ↵ thì con trỏ bản ghi sẽ được di chuyển về sau bản ghi hiện thời một đơn vị.

1.7. 2.3.4 Lấy dữ liệu từ bảng

a. Lệnh Display

Cú pháp: display [<phạm vi>] [fields<danh sách trường>]

[For<bthức logic>] [While<bthức logic>] [on|off]

Tác dụng: Hiển thị nội dung của các bản ghi trong <phạm vi> được chỉ định và thoả mãn điều kiện của các biểu thức logic đi sau FOR và WHILE nếu có.

Theo mặc định thì tất cả các trường trong bảng dữ liệu sẽ được hiển thị, nếu có [field<danh sách trường>] thì những trường được chỉ ra trong danh sách này mới được hiển thị lên màn hình. <phạm vi> mặc định là bản ghi hiện thời.

Ví dụ: 1. Hiển thị tất cả các bản ghi của bảng dữ liệu nhanvien:

Use nhanvien ↵

Display all ↵

2. Hiển thị tất cả những người có năm làm việc (namlv) trước 1980

Display for namlv <1980 ↵

Chú ý: Trong câu lệnh của Fox, nếu có mệnh đề FOR thì phạm vi mặc định là ALL.

b. Lệnh LIST

Cú pháp: List [<phạm vi>] [fields<danh sách trường>] [For<bthức logic>] [While<bthức logic>] [on|off]

Tác dụng: Hiển thị nội dung của các bản ghi như lệnh Display nhưng mặc định của lệnh này là ALL

Ví dụ:1. Hiển thị tất cả các bản ghi của bảng dữ liệu nhanvien:

Use nhanvien ↵
List ↵

2. Hiển thị tất cả những người có năm làm việc (namlv) trước 1980

List for namlv <1980 ↵

c. Lệnh ?|??

Cú pháp: ?|?? <danh sách biểu thức>

Tác dụng: Lệnh này tính toán và cho hiển thị kết quả của danh sách biểu thức lên màn hình.

Chú ý: lệnh ? trước khi in dữ liệu thì xuống dưới 1 dòng, còn lệnh ?? thì không.

Ví dụ: Cho hiển thị họ tên của người có số hiệu là 5 lên màn hình:

Go 5 ↵
? 'ho ten', hoten ↵

Chú ý: danh sách biểu thức trong Fox được viết cách nhau bởi dấu phẩy ",".

Ví dụ: cho biết họ tên, năm làm việc của người có số hiệu là 2:

Go 2 ↵
?'ho ten:',hoten,'nam lam viec:',namlv↵

1.8. 2.3.5 Chèn, bổ sung bản ghi

a. Chèn bản ghi

Cú pháp: INSERT [BEFORE][BLANK]

Tác dụng: Chèn một bản ghi ngay sau bản ghi hiện thời (nếu có [Before]) với nội dung được nhập vào. Nếu có [BLANK] thì sẽ chèn một bản ghi trắng.

Ví dụ: chèn một bản ghi vào sau bản ghi thứ 3:

Go 3 ↵

Insert

b. Bổ sung bản ghi

Cú pháp: APPEND [BLANK]

Tác dụng: Để chèn 1 bản ghi vào cuối bảng dữ liệu (giá trị được nhập vào), nếu có tham số [BLANK] thì sẽ bổ sung một bản ghi trắng.

2.3.6 Sửa chữa nội dung bản ghi

a. Lệnh BROWSE:

CÚ PHÁP: BROWSE [FIELD<DSÁCH TRƯờNG>] [FREEZE<DSÁCH TRƯờNG>][NODELETE]

[NOEDIT] [FOR<BTHứC LOGIC>]

TÁC DụNG: HIểN THị NộI DUNG CủA BảNG Dữ LIệU, MỗI BảN GHI ĐƯợC THể HIệN TRONG MộT HÀNG (DÒNG), TA CÓ THể XEM VÀ DI CHUYểN HộP SÁNG Từ TRƯờNG NÀY QUA TRƯờNG KHÁC, BảN GHI NÀY SANG BảN GHI KHÁC VÀ CÓ THể SửA ĐổI NộI DUNG CủA TừNG MẫU TIN TRONG BảN GHI.

VÍ Dụ:

USE NHANVIEN ↵

BROWSE ↵

[FIELD<DSÁCH TRƯờNG>]: CHO PHÉP CÁC TRƯờNG TRONG DANH SÁCH NÀY ĐƯợC HIểN THị TRÊN MÀN HÌNH, NếU KHÔNG CÓ THAM Số NÀY THÌ TấT Cả CÁC TRƯờNG TRONG BảNG Dữ LIệU Sẽ ĐƯợC HIểN THị.

[FREEZE<DSÁCH TRƯờNG>]: CHO PHÉP CÁC TRƯờNG TRONG DANH SÁCH NÀY LUÔN ĐƯợC HIểN THị TRÊN MÀN HÌNH.

[NODELETE]: KHÔNG CHO PHÉP XOÁ

[NOEDIT]: KHÔNG CHO PHÉP SửA ĐổI.

VÍ Dụ: HIểN THị NộI DUNG CủA CÁC TRƯờNG HOTEN, NAMLV Để TIếN HÀNH SửA ĐổI.

BROWSE FIELD HOTEN,NAMLV FREEZE NAMLV

[FOR<BTHứC LOGIC>]: CHỉ CHO PHÉP NHữNG BIểU THứC THOả MÃN ĐIềU KIệN CủA BIểU THứC LOGIC MớI ĐƯợC HIểN THị.

b. Lệnh Edit

CÚ PHÁP: EDIT [<PHạM VI>] [FIELD<DSÁCHTRƯờNG>][NOAPPEND][NODELETE] [NOEDIT]

[FOR<BTHứC LOGIC>] [WHILE<BTHứC LOGIC>]

TÁC DụNG: TƯƠNG Tự NHƯ LệNH BROWSE NHƯNG CÁC BảN GHI ĐƯợC XUấT HIệN NHƯ ở LệNH APPEND.

c. Lệnh REPLACE

CÚ PHÁP: REPLACE [<PHạMVI>]<TRƯờNG 1>WITH<BTHứC 1>[ADDITIVE]

[,<TRƯờNG 2> WITH <BTHứC 2> [ADDITIVE]...][FOR<BTHứC LOGIC>]

[WHILE<BTHứC LOGIC>]

TÁC DụNG: DÙNG Để THAY THế NộI DUNG CÁC TRƯờNG ĐƯợC CHỉ RA CủA CÁC BảN GHI NằM TRONG <PHạM VI> VÀ THỏA MÃN ĐIềU KIệN CủA <BIểU THứC LOGIC> ĐI SAU FOR HOặC WHILE BởI CÁC BIểU THứC TƯƠNG ứNG. PHạM VI MặC ĐịNH LÀ BảN GHI HIệN THờI.

CHÚ Ý: KIểU Dữ LIệU CủA <BIểU THứC> VÀ CủA <TRƯờNG> TƯƠNG ứNG PHảI TƯƠNG ĐƯƠNG NHAU, NếU KHÔNG THÌ FOX Sẽ THÔNG BÁO LỗI KIểU Dữ LIệU "DATA TYPE MISMATCH".

VÍ Dụ: 1. THAY THế Họ TÊN CủA NHÂN VIÊN TRONG FILE NHANVIEN BằNG CHữ IN

REPLACE ALL HOTEN WITH UPPER(HOTEN) ↵

2. NÂNG LƯƠNG CủA NHữNG NHÂN VIÊN Nữ LÊNH THÊM 50000 ĐồNG

REPLACE LUONG WITH LUONG+50000 FOR !GIOITINH ↵

1.9. **2.3.7 Xoá bản ghi**

VIệC XOÁ MộT BảN GHI TRONG BảNG Dữ LIệU ĐƯợC THựC HIệN THEO HAI BƯớC:

BƯớC 1: ĐÁNH DấU BảN GHI MUốN XOÁ:

CÚ PHÁP: DELETE [<PHạM VI>] [FOR<BTHứC LOGIC>] [WHILE<BTHứC LOGIC>]

TÁC DụNG: LệNH NÀY ĐÁNH DấU TấT Cả CÁC BảN GHI THOả MÃN ĐIềU KIệN ĐƯợC NÊU, MặC ĐịNH LÀ BảN GHI HIệN THờI.

KHI THựC HIệN LệNH NÀY CÁC BảN GHI ĐƯợC CHỉ ĐịNH ĐÁNH DấU XOÁ Sẽ XUấT HIệN DấU * ở TRƯớC CÁC BảN GHI. LÚC NÀY TA CÓ THể PHụC HồI LạI CÁC BảN GHI ĐÓ ĐƯợC.

VÍ Dụ: ĐÁNH DấU XOÁ NHữNG NHÂN VIÊN CÓ NĂM LÀM VIệC TRƯớC 1951.

DELETE FOR NAMLV <1950↵

BƯớC 2. XOÁ CÁC BảN GHI. CÁC BảN GHI SAU KHI ĐÃ ĐƯợC ĐÁNH DấU XOÁ NếU QUYếT ĐịNH THậT Sự MUốN XOÁ NÓ THÌ THựC HIệN LệNH PACK, NGƯợC LạI NếU KHÔNG MUốN XOÁ NÓ THÌ THựC HIệN LệNH RECALL.

A. LệNH XOÁ CÁC BảN GHI Bị ĐÁNH DấU XÓA (PACK)

CÚ PHÁP: PACK

TÁC DụNG: XOÁ CÁC BảN GHI TRONG BảNG Dữ LIệU ĐÃ ĐƯợC ĐÁNH DấU XOÁ BằNG LệNH DELETE.

B. LệNH PHụC HồI CÁC BảN GHI ĐÃ ĐƯợC ĐÁNH DấU XOÁ (RECALL):

CÚ PHÁP: RECALL [<PHạM VI>] [FOR<BTHứC LOGIC>] [WHILE<BTHứC LOGIC>]

TÁC DụNG: PHụC HồI LạI CÁC BảN GHI MÀ TRƯớC ĐÓ ĐÃ ĐƯợC ĐÁNH DấU XOÁ BởI LệNH DELETE. PHạM VI MặC ĐịNH CủA LệNH NÀY LÀ BảN GHI HIệN THờI.

C. LệNH XÓA Dữ LIệU TRÊN FILE DBF.

CÚ PHÁP: ZAP

TÁC DụNG: XÓA TấT Cả CÁC BảN GHI TRONG MộT FILE DBF ĐANG Mở.

1.10. **2.3.8 Lọc dữ liệu**

Để HạN CHế Số LƯợNG CÁC BảN GHI THAM GIA VÀO QUÁ TRÌNH Xử LÝ, TA CÓ THể LọC CÁC BảN GHI TRONG BÀNG Dữ LIệU THOả MÃN ĐIềU KIệN CHO TRƯớC.

CÚ PHÁP: SET FILTER TO <BTHứC LOGIC>↵

SAU KHI THựC HIệN LệNH LọC THÌ CÁC LệNH TIếP THEO SAU LệNH NÀY CHỉ CÓ TÁC DụNG ĐốI VớI CÁC BảN GHI THOả MÃN ĐIềU KIệN LọC.

MUốN HUỷ Bỏ VIệC LọC Dữ LIệU TA THựC HIệN LệNH: SET FILTER TO ↵

VÍ Dụ:

1. CHỉ HIểN THị NHữNG NHÂN VIÊN Nữ:

```
SET FILTER TO !GIOITINH↵
LIST↵

.....
```

2. CHÍ XÉT NHữNG NHÂN VIÊN CÓ QUÊ QUÁN LÀ HUế

```
SET FILTER TO QUEQUAN=="HUE" ↵
LIST↵

.........
```

1.11. **2.3.9 THAO TÁC VỚI CấU TRÚC BẢNG:**

a. Xem cấu trúc bảng (List|Display structure)

CÚ PHÁP: LIST | DISPLAY STRUCTURE↵

TÁC DụNG: HIểN THị CấU TRÚC CủA BảNG Dữ LIệU ĐANG ĐƯợC Mở, BAO GồM: TÊN TRƯờNG, KIểU VÀ Độ RộNG CủA TRƯờNG.

VÍ Dụ:

USE NHANVIEN↵
LIST STRUCTURE

b. Sửa đổi cấu trúc bảng dữ liệu

CÚ PHÁP: MODIFY STRUCTURE↵

TÁC DụNG: HIểN THị VÀ CHO PHÉP SửA ĐổI CấU TRÚC BảNG Dữ LIệU, KếT THÚC LệNH NÀY NHấN Tổ HợP PHÍM CTRL+W.

VÍ Dụ: USE NHANVIEN↵
MODIFY STRUCTURE↵

c. Sao lưu cấu trúc bảng dữ liệu

CÚ PHÁP: COPY STRUCTURE TO <TEN FILE.DBF> [FIELDS<DANH SÁCH TRƯờNG>]↵

TÁC DụNG: Để SAO CHÉP CấU TRÚC CủA BảNG Dữ LIệU ĐANG ĐƯợC Mở SANG MộT BảNG MớI CÓ TÊN ĐƯợC CHỉ RA TRONG <TEN FILE.DBF> VớI CÁC TRƯờNG ĐƯợC CHỉ RA TRONG MụC [FIELD<DANH SÁCH TRƯờNG>]. MặC ĐịNH CủA LệNH NÀY LÀ TấT Cả CÁC TRƯờNG CÓ TRONG BảNG Dữ LIệU ĐANG ĐƯợC Mở.

VÍ Dụ: SAO LƯU CấU TRÚC CủA NHANVIEN THÀNH FILE CÓ TÊN LÀ LUU.DBF NHƯNG CHỉ GồM CÁC TRƯờNG: HOTE, GIOITINH, NĂMLV.

USE NHANVIEN↵

COPY STRUCTURE TO LUU FIELDS HOTEN, GIOITINH, NAMLV↵

CHÚ Ý: BảNG MớI ĐƯợC TạO RA CHỉ CÓ CấU TRÚC, KHÔNG CÓ NộI DUNG.

1.12. 2.3.10 Sao chép bảng

CÚ PHÁP: COPY TO <TÊN BảNG ĐÍCH> [<PHạM VI>] [FIELDS <DANH SÁCH TRƯờNG>] [FOR<BTHứC LOGIC>] [WHILE<BTHứC LOGIC>]↵

TÁC DụNG: LệNH DÙNG Để TạO BảNG MớI CÓ TÊN ĐƯợC CHỉ RA <TÊN BảNG ĐÍCH> VớI NộI DUNG ĐƯợC LấY Từ BảNG Dữ LIệU ĐANG ĐƯợC Mở. MặC ĐịNH LệNH NÀY LÀ TấT Cả CÁC BảN GHI ĐềU ĐƯợC SAO CHÉP, NếU CÓ PHạM VI VÀ CÁC BIểU THứC LOGIC THÌ NHữNG BảN GHI THOả MÃN ĐIềU KIệN MớI ĐWOCJ SAO CHÉP. DANG SÁCH TRƯờNG Để CHỉ ĐịNH CÁC TRƯờNG ĐƯợC SAO CHÉP.

VÍ Dụ: TạO BảNG Dữ LIệU CÓ TÊN LÀ NU.DBF Từ FILE NHANVIEN.DBF GồM CÁC TRƯờNG HOTEN, NGAYSINH, NAMLV.

```
USE NHANVIEN↵
COPY TO NU FIELDS HOTEN, NGAYSINH, NAMLV FOR !GIOITINH↵
USE NU↵     &&Mở Để XEM KếT QUả
LIST↵
```

1.13. 2.4. MỘT SỐ HÀM THÔNG DỤNG

2.4.1. Các hàm về ngày tháng

A. HÀM DATE(): **CHO NGÀY, THÁNG, NĂM HIệN TạI CủA Hệ THốNG. THứ Tự NGÀY, THÁNG, NĂM CủA LệNH NÀY PHụ THUộC VÀO LệNH SET DATE.**

VÍ Dụ:

NếU TA THựC HIệN LệNH: SET DATE FRENCH↵

RồI THựC HIệN LệNH DATE()↵ THÌ NGÀY HIệN HÀNH CủA Hệ THốNG Sẽ ĐƯợC HIệN RA THEO THứ Tự LÀ NGÀY, THÁNG, NĂM.

B. HÀM YEAR(<BTHứC DATE>): **CHO NĂM (CÓ 4 CHữ Số) CủA <BTHứC DATE>.**

VÍ Dụ: YEAR(DATE()) → CHO NĂM HIệN TạI CủA NGÀY Hệ THốNG.

C. HÀM MONTH(<BTHứC DATE>): **CHO THÁNG HIệN TạI CủA BIểU THứC NGÀY**

VÍ Dụ: MONTH(DATE())→ CHO THÁNG CủA NGÀY Hệ THốNG.

D. HÀM DAY(<BTHứC DATE>): CHO NGÀY CủA BIểU THứC NGÀY.

VÍ Dụ: DAY(DATE())↵ → CHO NGÀY HIệN TạI.

2.4.2. Các hàm về chuỗi

A. HÀM LEN(<BTHứC C>): **CHO CHIềU DÀI CủA BIểU THứC, TÍNH BằNG BYTE CủA <BTHứC C>, CHUỗI RỗNG CÓ CHIềU DÀI LÀ 1.**

B. HÀM LEFT(<BTHứC C>, <N>): **TRÍCH RA MộT CHUỗI Từ <BTHứC C> GồM <N> KÝ Tự TÍNH Từ BÊN TRÁI SANG.**

VÍ Dụ: ?LEFT("NGUYEN VAN AN", 6) → CHO KếT QUả LÀ "NGUYEN".

C. HÀM RIGHT(<BTHứC C>, <N>): **TRÍCH RA MộT CHUỗI Từ <BTHứC C> GồM <N> KÝ Tự TÍNH Từ BÊN PHảI SANG.**

*D. HÀM SUBSTR (<BTHứC C>, <N1>, <N2>):***TRÍCH RA MộT CHUỗI CON CủA <BTHứC C> Từ Vị TRÍ <N1> VÀ GồM <N2>) KÝ Tự.**

VÍ Dụ: ? SUBSTR ("NGUYEN VAN AN", 8, 3" KếT QUả CHO CHUỗI "VAN".

E. HÀM ALLTRIM (<BTHứC C>): **CHO KếT QUả LÀ MộT CHUỗI SAU KHI ĐÃ LOạI Bỏ CÁC KÝ Tự TRắNG ở HAI BÊN (NếU CÓ) CủA<BTHứC C>.**

VÍ Dụ: ?ALLTRIM("NGUYEN VAN AN ") → "NGUYEN VAN AN"

F. HÀM UPPER(<BTHứCC>): **CHO KếT QUả LÀ CHUỗI IN HOA CủA <BTHứC C>.**

VÍ Dụ: ?UPPER ("NGUYEN VAN AN") → "NGUYEN VAN AN"

G. HÀM LOWER <BTHứC C>: **NGƯợC LạI CủA HÀM UPPER.**

2.4.3. Các hàm số học

A. ASB(X): **CHO GIÁ TRị TUYệT ĐốI CủA X.**

B. INT(X): **CHO PHầN NGUYÊN CủA X.**

C. ROUND(X,<N>): **LÀM TRÒN X VớI N Số Lẽ.**

E. SIN(X): **CHO GIÁ TRị SIN X**

F. COS (X): **CHO GIÁ TRị COS X.**

Bài thực hành chương 2

1. Tạo tập tin DBF.

Dùng lệnh Create từ cửa sổ lệnh để tạo cấu trúc cho tập tin HSNV.DBF như sau: (Chỉ tạo cấu trúc, không nhập dữ liệu).

2. **Field Name**	**Field Type**	**Width**	**Dec**	**Phần ghi chú**
MASONV	Character	5		Mã số nhân viên
HOLOT	Character	20		Họ lót
TEN	Character	7		Tên
PHAI	Logic	1		Phái (Nam, Nữ)
DIACHI	Character	30		Địa chỉ
NGSINH	Date	8		Ngày sinh
TDVH	Numeric	2		Trình độ văn hoá
M_LUONG	Numeric	3		Mức lương
NGAYLL	Date	8		Ngày lên lương

Ghi chú: Trình độ văn hoá được đánh giá qua các mã sau:

0: Mù chữ, 1-12: Phổ thông, 13: Đại học, 14, Cao học, 15: Tiến sĩ

b. Cho biết công dụng của phím F5 và F6

c. Thêm vào tập tin vừa tạo ra hai trường mới.

3. **Field Name**	**Field Type**	**Width**	**Dec**	**Phần ghi chú**
MADV	Character	2		Mã đơn vị
HOHANG	Memo	10		Họ hàng

d. Ở trường PHAI sửa lại tên là NU có kiểu Logic.

e. Nhập số liệu 10 nhân viên vào tập tin HSNV.BDF này.

Ghi chú: Để nhập dữ liệu vào vùng HOHANG (kiểu Memo) dùng phím Ctrl_Home và kết thúc bằng Ctrl_W.

2. Dùng menu hệ thống tạo cấu trúc tập tin HOCVIEN.DBF sau đây:

4. **Field Name**	**Field Type**	**Width**	**Dec**	**Phần ghi chú**
MASONV	Character	4		Mã số nhân viên
HO	Character	20		Họ lót
TEN	Character	7		Tên
NAM	Logic	1		Nam: .T., Nữ: .F.
NGSINH	Date	8		Ngày sinh
NOISINH	Character	2	5	Nơi sinh
DIACHI	Character	20		Địa chỉ
MALOP	Character	4		Mã lớp
MAGV	Character	3		Mã giáo viên
DIEMLT	Numeric	5	2	Điểm lý thuyết
DIEMTH	Numeric	5	2	Điểm thực hành
UUTIEN	Logic	1		Ưu tiên
GHICHU	Date	10		Ghi chú

Nhập số liệu 10 học viên đầu tiên

b. Dùng lệnh DIR ở cửa sổ lệnh để xem tập tin có trên đĩa hay không, số mẫu tin vừa nhập và dung lượng đĩa còn trống?

c. Gõ lệnh: Use để đóng tập tin HOCVIEN.DBF rồi thoát khỏi FoxPro.

C□p nh□t d□ li□u

1. Mở tập tin HOCVIEN.DBF

- Dùng lệnh LIST hay DISPLAY ALL để xem nội dung các mẫu tin của tập tin HOCVIEN.DBF và chỉ xem các vùng tin MAHV, HO, TEN, NAM, MALOP, MAGV, DIEMLT, DIEMTH, nếu có sai sót hãy điều chỉnh cho đúng.

2. Gõ lệnh SET STATUS ON để xem thanh trạng thái.

- Nếu thanh trạng thái bị che khuất bởi cửa sổ lệnh thì ấn Ctrl_F7 để di chuyển cửa sổ đến vị trí khác.

- Nếu bóng mờ dưới cửa sổ lệnh che lấp thanh trạng thái thì gõ SET SHADOW OFF để tắt đi.

3. Dùng lệnh APPEND để thêm hai mẫu tin mới rồi ấn Ctrl_W để ghi lại.

4. Dùng hàm RECNO() cho biết số hiệu của mẫu tin hiện hành dời con trỏ đến đầu tập tin.

5. Sử dụng lệnh EDIT để sửa nội dung các mẫu tin tuỳ ý thích của bạn, sửa xong ấn Ctrl_W để ghi lại.

6. Gõ lệnh: BROWSE. Quan sát màn hình rồi thử các động tác sau:

a. Đưa vệt sáng đến mẫu tin thứ nhất tại vùng ghi chú, rồi ấn Ctrl_Home để xem phần ghi chú có những nội dung gì? Gõ thêm một ghi chú tuỳ ý rồi ấn Ctrl_W để ghi lại.

b. Ấn Alt+B để gọi MENU của BROWSE, sau đó gọi APPEND, nhập thêm 1 mẫu tin rồi ấn Ctrl_W để ghi lại.

c. Gõ lại lệnh BROWSE lần nữa, ấn Ctrl_N, FoxPro sẽ thêm mẫu tin trắng ở cuối, nhập số liệu cho mẫu tin này.

d. Đưa vệt sáng đến vùng DIEMLT, ấn Alt+B để gọi MENU phụ, sau đó chọn Move rồi chuyển vệt sáng đến vùng DIEMTH, ấn Enter. Kết quả hai cột DIEMLT và DIEMTH sẽ được chuyển cho nhau.

e. Đưa vệt sáng đến vùng NOISINH, ấn Alt+B để gọi menu SIZE, dùng mũi tên trái thu hẹp cột này còn 10 Bytes thôi, sau đó gõ: “Vĩnh lợi-Huế” vào mẫu tin thứ tư.

7. Gõ lệnh DELETE ALL FOR DIEMTH < 7 rồi xem có bao nhiêu mẫu tin bị đánh dấu xoá?

8. Gõ lệnh BROWSE để quan sát, sau đó đưa vệt sáng đến một mẫu tin bị đánh dấu xoá rồi ấn Ctrl_T xem dấu xoá có còn hay không, ấn lại Ctrl_T lần nữa để xem điều gì xảy ra, sau đó ấn Ctrl_W để thoát ra.

9. Gõ lệnh RECALL ALL để phục hồi các mẫu tin bị đánh dấu xoá rồi đóng tập tin HOCVIEN.DBF lại.

10. Mở tập tin HSNV.DBF

a. Dùng lệnh REPLACE để tăng lương gấp đôi cho tất cả nhân viên, sau đó tăng thêm riêng cho các nữ nhân viên 10% nữa.

b. Thêm vào cấu trúc tin HSNV.DBF một trường LOAIBC (loại biên chế: BC/HD) và dùng BROWSE nhập dữ liệu cho vùng tin này, ấn Ctrl_B để thoát khỏi BROWSE.

c. Gõ lệnh DISPLAY STRUCTURE (hay F5) để xem lại cấu trúc.

d. Đánh dấu xoá các nhân viên mù chữ và trình độ phổ thông cho liệt kê trên màn hình những mẫu tin không bị đánh dấu xoá.

e. Nhập thêm hai mẫu tin vào giữa tập tin HSNV.DBF.

- Một mẫu tin sau mẫu tin có STT=5

- Một mẫu tin trước mẫu tin có STT=3

f. Gõ lệnh RECALL ALL để phục hồi các mẫu tin bị đánh dấu xoá.

g. Liệt kê danh sách các nhân viên theo dạng.

MNV	HOLOT	TEN	NU	NGSINH	HSL	TDVH

h. Liệt kê theo ạng câu g những nhân viên nam.

i. Liệt kê theo dạng câu g những nhân viên nam từ 18 đến 30 tuổi.

j. Liệt kê theo dạng câu g những nhân viên nữ có trình độ đại học.

k. Liệt kê theo dạng câu g những nhân viên có tên bắt đầu bằng vần ‘H’

l. Gõ lệnh Use để đóng tập tin HXNV.DBF rồi thoát khỏi FoxPro.

5. CHƯƠNG 3: SẮP XẾP-TÌM KIẾM-THỐNG KÊ

5.1. *3.1. SẮP XẾP*

3.1.1. Khái niệm

Trong m□t b□ng d□ li□u, chỳng ta cú th□ s□p x□p cỏc m□u tin theo m□t tiờu chu□n n□o □ú tu□ theo yờu c□u c□a vi□c khai thỏc thụng tin.

3.1.2. Sắp xếp theo chỉ mục

a. Khái niệm về chỉ mục

Ta đã biết mỗi bảng dữ liệu chứa các bản ghi và mỗi bản ghi đều được đánh số hiệu theo số thứ tự từ 1 đến n.

Ví dụ: bảng NHANVIEN.DBF có dang sau:

Record#	HOTEN	NGAYSINH	GIOITINH	NAMLV
1	**NGUYễN VĂN A**	02/10/75	.T.	1985
2	Lê thị nhàn	05/23/75	.F.	1980
3	Nguyễn An	10/26/80	.T.	1982
4	Trần Hạnh	09/25/70	.T.	1981

Số hiệu các bản ghi

Khi x□ lý thụng tin trong b□ng d□ li□u, ta truy xu□t chỳng theo tr□t t□ c□a s□ hi□u b□n ghi.

Ví dụ: use NHANVIEN ↵
list ↵

Kết quả in ra sẽ như sau:

Record#	HOTEN	NGAYSINH	GIOITINH	NAMLV
1	**NGUYễN VĂN A**	02/10/75	.T.	1985
2	Lê thị nhàn	05/23/75	.F.	1980
3	Nguyễn An	10/26/80	.T.	1982
4	Trần Hạnh	09/25/70	.T.	1981

Sắp xếp bảng dữ liệu theo chỉ mục là tạo ra một file mới (có phần mở rộng mặc định là .IDX) chỉ có hai trường: trường khoá sắp xếp và trường số hiệu bản ghi. Thứ tự của bản ghi ở đây là thứ tự sắp xếp.

Vớ d□: file ch□ m□c c□a b□ng nhanvien theo th□ t□ t□ng d□n c□a n□m lẩm vi□c nh□ sau:

file chỉ mục theo namlv

Namlv	Record#
1980	2
1981	4
1982	3
1985	1

Lúc này, khi truy xuất dữ liệu của bảng, thứ tự của các bản ghi là thứ tự được quy định trong file chỉ mục này.

Vớ d□: Trong b□ng nhanvien, s□ d□ng ch□ m□c theo tr□□ng namlv.idx ta cú th□ t□ truy xu□t:

Record#	Hoten	ngaysinh	gioitinh	namlv
2	Lê thị nhàn	05/23/75	.F.	1980
4	Trần Hạnh	09/25/70	.T.	1981
3	Nguyễn An	10/26/80	.T.	1982
1	Nguyễn văn A	02/10/75	.T.	1985

b. Lập chỉ mục IDX cho bảng dữ liệu

Cú pháp: INDEX ON <bthức khoá> TO <tên file idx>

[FOR<bthức logic>] [UNIQUE]↵

Tác dụng: Lệnh sắp xếp file dữ liệu theo chiều tăng dần của <Bthức khoá> của các bản ghi thoả mãn <Bthức logic> sau FOR, mặc định là tất cả các bản ghi. Nếu có từ khoá [UNIQUE] thì các bản ghi nào có <Bthức khoá> trùng nhau sẽ bị bỏ qua trên file chỉ mục.

Ví dụ 1: Hiển thị theo thứ tự tăng dần của namlv của các nhân viên.

```
use NHANVIEN↵
index on NAMLV to CMNAMLV↵
list↵
```

Ví dụ 2: Hiển thị theo thứ tự tăng dần của hoten

index on HOTEN to CMHOTEN↵

list↵

Chú ý: Lệnh luôn sắp xếp theo thứ tự tăng dần của <bthức khoá>, do vậy khi lựa chọn <bthức khoá> thì phải chọn cho phù hợp.

Ví dụ 1: Hiển thị theo thứ tự giảm dần của namlv của các nhân viên.

use NHANVIEN↵

index on -NAMLV to CMNAMLVG↵

list↵

Ví dụ 2: Hiển thị theo thứ tự giảm dần của ngaysinh.

use NHANVIEN↵

index on date()-NGAYSINH to CMNSINHG↵

list↵

c. Một số lệnh liên quan

+ SET INDEX TO <file chỉ mục>: Dùng để mở file chỉ mục sau khi đã mở một bảng dữ liệu.

+ SET INDEX TO: Dùng để đóng file chỉ mục.

+ REINDEX: Dựng □□ c□p nh□t l□i file ch□ m□c sau khi cú s□ s□a □□i trờn b□ng d□ li□u.

5.2. *3.2. TÌM KIẾM*

3.2.1. Tìm kiếm tuần tự

a. Lệnh Locate:

Cú pháp:

LOCATE [<phạm vi>] FOR<bthức logic> [WHILE<bthức logic>]

Tác dụng: Lệnh sẽ duyệt tuần tự các bàn ghi trong bảng dữ liệu và tìm đến bản ghi đầu tiên trong <phạm vi> thoả mãn điều kiện của <bthức logic>. Nếu tìm được, hàm FOUND() sẽ cho giá trị .T., hàm EOF() có giá trị .F.

Ví dụ: Tìm nhân viên đầu tiên trong bảng dữ liệu sinh năm 1970 trong bảng nhanvien

```
use NHANVIEN↵
Locate for year(NGAYSINH) = 1970↵
Display↵
```

b. Lệnh continue

Cú pháp : CONTINUE

Chức năng : Theo sau lệnh LOCATE, dùng để tìm bảng ghi kế tiếp sau thỏa mãn điều kiện đã nêu.

Ví dụ : Tìm 2 nhân viên đầu tiên sinh năm 1970

```
use NHANVIEN
locate for year ( NGAY SINH) = 1970
display
continue
display
```

3.2.2. Tìm kiếm sau khi đã lập chỉ mục

Cú pháp : SEEK <biểu thức>

Chức năng : sau khi đã lập chỉ mục theo <bthức khóa> đề tìm bản ghi nào thỏa mãn một điều kiện dựa vào <bthức khóa>

Ta sử dụng lệnh SEEK theo sau là <giá trị> của biểu điều kiện cần tìm. nếu tìm thấy thì hàm FOUND() có giá trị .T. và hàm EOF () có giá trị .F.

Ví dụ: 1. Sắp xếp theo thứ tự tăng dần của Họ Tên, tìm nhân viên có tên "Nguyen Van AN".

use NHANVIEN

index on upper(HOTEN) to CMHOTEN

seek “Nguyen Van An”

disp

2. Sắp xếp theo thứ tự giảm dần của NAMLV, tìm nhân viên có năm làm việc 1981.

use NHANVIEN

index on - NAMLV to CMNAMLVG

list

seek -1981

disp

3.3. THỐNG KÊ

3.3.1. Đếm số lượng bản ghi

Cú pháp

COUNT [<phạm vi>][FOR<btlogic>] [WHILE<btlogic>] [TO<biến nhớ>]

Chức năng :lệnh dùng để đếm số mẫu tin trong bảng dữ liệu hiện hành thỏa mãn điều kiện các <bthức logic> nằm trong phạm vi được chỉ ra. Kết quả được đưa ra màn hình hay đưa vào <biến nhớ> nếu có TO.

Ví dụ: Cho biết có bao nhiêu nhân viên có NAMLV là 1980

use NHANVIEN

count for NAMLV = 1980 to songuoi

?’ có songuoi: ‘, songuoi, ‘ làm việc năm 1980’

3.3.2. Tính tổng giá trị các trường kiểu số

Cú pháp: SUM [<phạm vi>] [<dsách bt>] [TO <ds biến>]

[FOR <bt logic>] [WHILE <btlogic>]

Chức năng : Lệnh sẽ lấy tổng theo các biểu thức được xây dựng dựa trên các trường kiểu số, của các bản ghi trong bảng dữ liệu; nằm trong <phạm vi> và thỏa mãn điều kiện của các <bthức logic>. Nừu không có <ds biểu thức> thì các trường kiểu số đều được lấy tổng.

Mặc định, kết quả được đưa ra màn hình; nếu có TO <dsbiến> thì kết quả của các <biểu thức> sẽ được đưa vào các <biến> tương ứng.

Chú ý : Phải tương ứng 1-1 giữa <ds biểu thức> và <ds biến>.

Ví dụ: Dựa vào bảng NHANVIEN, cho biết tổng LUONG phải trả và tổng PHUCAP là bao nhiêu.

```
use NHANVIEN
sum LUONG, PHUCAP to tongluong, tongpc
?’ tong luong la:’ , tong luong
?’ tong phu cap la:’ , tongpc
```

3.3.3. Tính trung bình cộng các trường kiểu số

Cú pháp: AVERAGE [<phạm vi>] [<ds biểu thức>] [TO <ds biến >] [FOR <bt logic>] [WHILE <bt logic>]

Chức năng : giống như lệnh SUM ở trên nhưng sau khi lấy tổng, lệnh sẽ lấy giá trị đó đem chia cho tổng số bản ghi tham gia vào câu lệnh.

Ví dụ: dựa vào bảng NHANVIEN, cho biết trung bình mỗi nhân viên nhận được bao nhiêu LUONG, PHU CAP.

```
use NHANVIEN
average LUONG, PHUCAP  to tbluong, tbphucap
?’ trung binh luong:’ , tbluong
?’ trung bình phu cap:’ , tbphucap
```

3.3.4. Tính tổng các trường số theo nhóm

CÚ PHÁP: TOTAL ON <BT KHÓA> TO <TÊN BảNG MớI.DBF>[<PHạM VI>]

[FIELD <dstrường>][FOR <biểu thức L>][WHILE < biểu thức L>]

Chức năng: Lệnh sẽ cộng dồn các trường kiểu số theo từng nhóm bản ghi có <bt khóa> giống nhau và đưa vào bảng mới có tên được chỉ ra ở <tên bảng .DBF>. Mặc định thì tất cả các trường kiểu số đều được cộng dồn, nếu có FIELDS <danh sách trường> thì chỉ có các trường liệt kê mới được cộng. Lệnh chỉ tác động đến các bàn ghi nằm trong (phạm vi) và thoả mãn điều kiện đi sau các mệnh đề FOR, WHILE.

Chú ý: Trước khi dùng lệnh này, bảng dữ liệu phải định sắp xếp theo khoá.

Ví dụ: Dựa vào bảng VATTV, hãy thống kê xem mỗi mặt hàng đã xuất hay nhập một số lượng là bao nhiêu.

```
use     VATTV
index  on  MAXN + MAVT   to  CMTK
total on   MAXN + MAVT   to THONGKE fields SOLUONG
use   THONGKE
?  'chi tiet la :'
list  MAXN, MAVT, SOLUONG, DONGIA
```

Giả sử bảng VATTU sau khi sắp xếp là:

MAXN	SOCT	MAVTU	SOLUONG	DONGIA
N	9	A01	145	5
N	4	A01	203	500
N	1	F01	123	200
N	2	F01	345	200
N	10	F01	654	180

Kết quả của bảng THONGKE.DBF là:

MAXN	SOCT	MAVTU	SOLUONG	DONGIA
N	9	A01	348	500

N	1	F01	469	200
N	10	F01	654	180

6. CHƯƠNG 4: LẬP TRÌNH TRÊN VISUAL FOXRO

6.1. *4.1. CHƯƠNG TRÌNH*

4.1.1 Khái niệm

Là một dãy lệnh liên tiếp được tổ chức vào 1 file chương trình, file chương trình mặc định có phần mở rộng là *. PRG.

Trong một chương trình, mỗi lệnh được viết trên một hàng và mỗi hàng chỉ chứa một lệnh tại một cột bất kỳ.

4.1.2. Soạn thảo chương trình

Để soạn thảo chương trình, từ cữa sổ lênh đưa vào lệnh;

MODIFY COMAND < tờn file ch□□ng trỡnh >

Lúc này xuất hiện cửa sổ chương trình để ta có thể đưa các lệnh vào cho nó.

Một chương trình foxpro thường có 3 phần.

a) Tạo môi trường làm việc : thường chứa các lệnh sau:

SET DATE FRENCH: đặt ngày tháng năm theo dạng DD-MM-YY

SET CURENCY ON : đặt năm có 4 chữ số

SET TALK OFF/ON : ẩn hiện các kết quả thực hiện lệnh

SET DEFAULT TO <đường dẫn> : đặt đường dẫn hiện thời

CLEAR: xoá màn hình hiển thị kết quả

CLOSE ALL: đóng các bảng dữ liệu, các file cơ sở dữ liệu,...

b) Phần thân chương trình:

Thực hiện các công việc mà chương trình yêu cầu như :

+ Cập nhập dữ liệu

+ xử lý, tính toán

+ Kết xuất thông tin

c) Kết thúc chương trình

+ Đóng các tập tin CSDL, các bảng dữ liệu đang sử dụng

+ Giải phóng biến nhớ

+ Trả lại các chế độ cho hệ thống.

d) Chú thích trong chương trình

Là các giải thích được thêm vào để làm rõ cho chương trình, phải được bắt đầu bởi dấu * hay &&

* : Bắt đầu một dòng

&& : Viết sau một lệnh

6.2. *4.2. BIẾN NHỚ*

4.2.1. Khai báo biến

a) Lệnh gán =

Cú pháp: <biến> = <biển thức>

Ví dụ: a = 5

ngay = Date()

b) Lệnh STORE

Cú pháp: STORE <bthức> to <ds biến>

Công dụng: Gán giá trị <bthức> cho <ds biến> ; nếu <biến> chưa tồn tại nó sẽ khai báo, nếu đã có thì thay thế bởi giá trị mới.

Ví dụ: STORE 0 To a, b, c

4.2.2. Nhập giá trị cho biến từ bàn phím

a) Lệnh ACCEPT

Cú pháp ACCEPT <bthức chuỗi> to <biến chuỗi>

Chức năng : Dùng để nhập một chuỗi từ bàn phím, kết thúc bởi phím ↵, giá trị nhận được sẽ đưa cho <biến>.

Ví dụ:

ACCEPT ‘nhap ho ten’ to bhoten

? ‘Ho ten vua nhap', bhoten

<Bthức chuỗi> là một câu nhắc nhở người sử dụng.

b. Lệnh INPUT

Cú pháp: INPUT <Bthức chuỗi> to <biến>

Tác dụng: Tương tự lệnh trên nhưng có thể nhận dữ liệu theo từng kiểu:

Kiểu Charater: Phải được đặt trong cặp dấu ' ... ' hay "... ".

Kiểu Numberic: Nhập dữ liệu kiểu số.

Kiểu Date: Phải được để trong dấu {}.

Kiểu Logic: Nhập giá trị .T. hay .F.

Ví dụ:

INPUT 'Nhap ngay sinh' TO bngaysinh

INPUT 'Nhap diem" TO bdiem

Chú ý: Trong hai lệnh trên, nếu biến chưa có thì nó sẽ tự khai báo, nếu đã có thì nó sẽ thay giá trị của biến bởi giá trị vừa nhập.

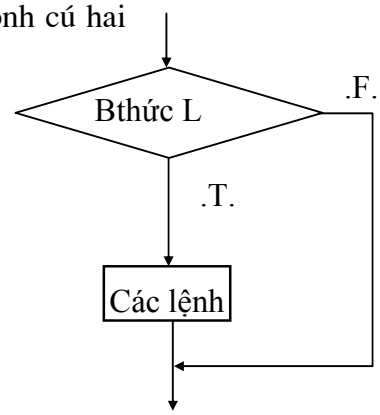
4.3. CÁC CẤU TRUẽC ĐIỀU KHIỂN CHƯƠNG TRÌNH

4.3.1. Cấu trúc tuần tự

Quy ước: Chương trình được thực hiện từ trên xuống dưới.

4.3.2. C□u trỳc r□ nhỏnh

Cũn g□i l□ c□u trỳc ch□n l□a. C□u trỳc r□ nhỏnh cú hai d□ng: d□ng khuy□t v□ d□ng □□y □□:



a. Dạng khuyết:

Cú pháp:

IF <Bthức L>

<các lệnh>

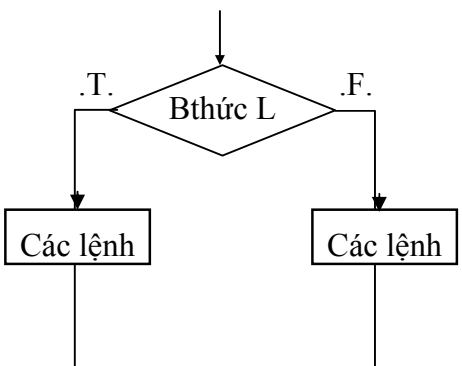
ENDIF

Tác dụng: Khi gặp cấu trúc này, <Bthức L> sẽ được tính, nếu có giá trị .T. thì <các lệnh> sẽ được thực hiện, ngược lại thực hiện các lệnh tiếp theo sau.

Vớ d□: Vi□t ch□□ng trỡnh nh□p v□o hai s□, cho bi□t s□ l□n nh□t

```
set talk off
clear
Input :Nhap so thu nhat' to so1
Input :Nhap so thu hai' to so2
max=so1
If max < so2
    max=so2
Endif
? "So lon nhat la:', max
set talk on
return
```

b. Dạng đầy đủ:



Cú pháp:

```
IF <Bthức L>
        <các lệnh 1>
ELSE
        <các lệnh 2>
ENDIF
```

Tác dụng: Khi gặp cấu trúc này, <Bthức L> sẽ được tính. Nếu có giá trị .T. thì <các lệnh 1> sẽ được thực hiện, ngược lại (có giá trị .F.) thì sẽ thực hiện <các lệnh 2>. Sau đó tiếp tục thực hiện các lênh tiếp theo trong chương trình.

Ví dụ: Dựa vào bảng nhanvien, hãy nhập vào một họ tên nhân viên, tìm xem có đúng là nhân viên của công ty hay không, nếu đúng thì thông báo năm sinh và năm làm việc, ngược lại thì thông báo là không phải nhân viên của công ty.

```
set talk off
clear
close all
use NHANVIEN
accept  'nhap ho ten nhan vien:' to bhoten
locate for HOTEN =bhoten
if found()
    ? ' ngay sinh là:', ngaysinh'
    ?'nam lam viec:', namlv
else
    ?'khong phai la nhan vien cua cong ty'
endif
set talk on
return
```

4.3.2.1. Lựa chọn một trong nhiều trường hợp

Cú pháp

```
    DO CASE
```

```
        CASE <Bthức L1>
                <các lệnh 1>
        CASE <Bthức L2>
                <các lệnh 2>
        .....
        CASE <Bthức Ln>
                <các lệnh n>
        [OTHERWISE
                <Các lệnh n+1>]
ENDCASE
```

Tác dụng: Khi gặp cấu trúc DO CASE, các <Bthức L> điều kiện sẽ được tính. Nếu <Bthức L> điều kiện nào đó có giá trị .T. thì nhóm lệnh tương ứng sẽ được thực hiện và kết thúc cấu trúc này, rồi tiếp tục thực hiện các lênh sau ENDCASE. Trong trường hợp không có <bthức L> nào từ 1 đến n có giá trị .T. thì <nhóm lệnh n+1> (nếu có) sẽ được thực hiện.

Ví dụ: Viết chương trình nhập vào một năm (có 4 chữ số), sau đó nhập thêm một tháng, cho biết tháng này có bao nhiêu ngày.

```
set takl off
clear
input 'nhap vao mot nam' to bnam
input 'nhap vao mot thang' to bthang
do case
        case bthang=4 or bthang = 6 or bthang = 9 or bthang=11
                songay=30
        case bthang=12
                if (mod(bnam, 4)=0 and (mod(bnam, 100)<>0)
                        songay=29
                else
                        songay=28
                endif
        otherwise
                songay=31
```

endcase

? 'thang', bthang, 'nam', bnam, 'co', so ngay, 'ngay'

set talk on

return

4.3.3. Cấu trúc lặp

4.3.3.1. Cấu trúc DO WHILE

Cú pháp:

DO WHILE <Bthức L>

<các lệnh>

[LOOP]

[EXIT]

ENDDO

Tác dụng: Khi gặp cấu trúc này thì <Bthức L> sẽ được tính, nếu có giá trị .F. thì sẽ dừng và thực hiện các lệnh sau ENDDO. Nếu có giá trị .T. thì các lệnh trong thân vòng lặp sẽ được thực hiện và lại quay về kiểm tra điều kiện trong <bthức L> và cứ thế tiếp tục.

[LOOP]: Khi gặp lệnh này, Foxpro sẽ quay về kiểm tra điều kiện logic mà bỏ qua các lệnh phía sau [LOOP].

[EXIT]: Khi gặp lệnh này thĩ sẽ thoát ra khỏi chương trình.

Ví dụ: Cho biết dang sáchhọ tên của các nhân viên trong công ty.

set takl off

clear

close all

use NHANVIEN

? 'danh sach ho ten hoc vien la:'

do while !eof()

?HOTEN

skip

enddo

set talk on

return

Chú ý: Khi sử dụng cấu trúc này, các lệnh trong thân vòng lặp phải thay đổi được giá trị của <Bthức L> để đảm bảo tính kết thúc.

Ví dụ: Nhập vào một năm, hãy thông báo danh sách họ tên, ngày sinh của những nhân viên làm việc trong năm đó, nếu không có thì thông báo là không có.

```
set takl off
set date french
clear
close all
use NHANVIEN
input 'nhap nam lam viec" to bnam
set filter to NAMLV = bnam
count to dem
if dem = 0
        ? ' khong co nhan vien nao'
else
        go top
        ? 'danh sach nhan vien lam viec năm', bnam
        ? "HO TEN                    NGAY SINH'
        do while !eof()
                ?HOTEN, NGAYSINH
                skip
        enddo
endif
set filter to
set talk on
return
```

4.3.3.2. Cấu trúc SCAN

Cú pháp

SCAN [<phạm vi>] [FOR<Bthức L>] [WHILE<bthức L>]

<các lệnh>

[LOOP]

[EXIT]

END SCAN

Tác dụng: Dùng để duyệt lần lượt các bản ghi trong bảng dữ liệu hiện hành nằm trong <phạm vi> được chỉ ra và thoả mãn điều kiện của các <Bthức L> sau FOR hoặc WHILE. Tương ứng với một bản ghi tìm được. <các lệnh> sẽ được thực hiện.

Cấu trúc SCAN sẽ dừng khi nào duyệt đến bản ghi cuối cùng của bảng dữ liệu đang xét.

Vớ d□: Vi□t ch□□ng trỡnh nh□p n□m (b□n ch□ s□), hi□n th□ nh□ng n□ nhõn viờn sinh n□m □ú.

set takl off

set date french

clear

close all

use NHANVIEN

input 'nhap nam lam viec" to bnam

input 'nhap nam:' to bnam

? 'DANH SACH CAC NU NHAN VIEN, SINH NAM', bnam

scan for !GIOITINH and year(NGAYSINH)=bnam

?HOTEN, NGAYSINH

endscan

set talk on

return

Bài thực hành chương 4

7. **Field Name**	**Field Type**	**Width**	**Dec**	**Phần ghi chú**
MASV	Character	5		Mã số nhân viên
HOLOT	Character	20		Họ lót
TEN	Character	7		Tên

NGSINH	Date	8		Ngày sinh
QUEQUAN	Character	20		Quê quán
DOS	Numeric	2		Điểm môn Dos
VRES	Numeric	2		Điểm VRER
FOX	Numeric	2		Điểm FoxPro
DTB	Numeric	4	2	Điểm trung bình
XEPLOAI	Character	4		Xếp loại

Nhập vào 10 mẫu tin cho các vùng: MASV, HOLOT, TEM, NGSINH, QQUAN, DOS, VRES, FOR theo mẫu dưới đây. Các vùng tin còn lại sẽ tính sau:

MASV	**HOLOT**	**TEN**	**NGSINH**	**QQUAN**	**DOS**	**VRES**	**FOR**
CK001	Le Van	Hung	12-04-1972	Da Nang	8	9	10
NT001	Ho Thi	Lan	10-05-1969	Hue	7	6	9
NT002	Tran Van	Long	06-12-1968	Da Lat	5	4	5
CK002	Le	Tung	05-06-1967	TP.HCM	7	7	6
KT001	Ng. Thi	Hoa	10-10-1967	TP.HCM	8	4	7
KT002	Le Van	Chau	05-04-1968	Da Nang	7	6	9
CK003	Vo	Anh	02-10-1969	Hue	9	9	10
CK004	Ho Duc	Tuan	10-02-1968	Da Nang	7	6	9
NT003	Vo Thi	Lan	02-01-1969	Hue	8	9	9
NT004	Le Van	Huy	05-06-1968	Da Nang	8	5	2

2. Dùng lệnh COPY FILE để chép tập tin KETQUA1.DBF thành KQ1.DBF. Sau đó có thể dùng lệnh USE KQ1 để mở tập tin KQ1 không? tại sao?

3. Mở tập tin KETQUA1.DBF

a. Tính (điểm trung bình), biết rằng DOS có hệ số hai, VRES có hệ số 1, FOX có hệ số 3.

b. Xếp loại, biết rằng:

DTB>=9 : Xếp loại ‘GIOI’

7<=DTB<9 : Xếp loại ‘KHA’

5<=DTB<7 : Xếp loại 'TB'

DTB<5 : Xếp loại 'YEU'

c. Sắp xếp giảm dần theo DTB và ghi vào tập tin SX_DTB.DBF. Mở tập tin SX_DTB.DBF rồi dùng lệnh BROWSE để xem.

d. Đổi dữ liệu của trường QQUAN thành chữ hoa.

e. Tính trung bình cộng của các môn học cho toàn bộ các mẫu tin, cho từng nhóm có MASV bắt đầu bằng CK, NT, KT.

f. Cho biết số sinh viên có ít nhất hai môn có điểm >=8.

Bài tập chương 5

1. Tạo tập tin NHAPVT.DBF có cấu trúc như sau:

8. **Field Name**	**Field Type**	**Width**	**Dec**	**Phần ghi chú**
MAVT	Charater	5		Mã số vật tư
NGAYNHAP	Date	8		Ngày nhập
MANX	Charater	1		Nhập: N, xuất: X
SL	Numeric	6		Số lượng
DONGIA	Numeric	8		Đơn giá
THANHTIEN	Numeric	9		Thành tiền

Nhập vào 10 mẫu tin theo mẫu dưới đây:

MAVT	**NGAYNHAP**	**MANX**	**SL**	8.1. **DONGIA**
TV01	01-01-1998	N	12	3850000
TL01	04-01-1998	N	10	4700000
ML01	08-01-1998	X	40	5100000
BU01	04-05-1998	N	30	220000
QB01	05-01-1998	N	28	350000
MG01	05-06-1998	X	12	4000000
ND01	06-06-1998	N	20	650000

HD02	10-10-1998	N	12	13000000
HD02	01-01-1998	N	10	16000000
XD01	01-01-1998	X	30	1200000

2. Tập tin TONKHO98.DBF có cấu trúc như sau:

9. **Field Name**	**Field Type**	**Width**	**Dec**	**Phần ghi chú**
MAVT	Charater	5		Mã số vật tư
TONDAU	Numeric	10		Tồn đầu kỳ
SLN	Numeric	10		Số lượng nhập
SLX	Numeric	10		Số lượng xuất
TONCUOI	Numeric	10		Tồn cuối kỳ

Nhập vào các mẫu tin sau:

MAVT	**TONDAU**
TV01	12
TL01	30
ML01	50
BU01	40
QB01	50
MG01	55
ND01	100
HD02	50
HD02	45
XD01	100

3. Tạo tập tin DMVTU.DBF có cấu trúc sau:

10. **Field Name**	**Field Type**	**Width**	**Dec**
MAVT	Charater	5	

TENVT	Charater	20

- Lấy MAVT trong tập tin TONKH98.DBF thay thế vào trường MAVT
- Nhập vào trường TENVT các dữ liệu sau:

MAVT	NGAYNHAP
TV01	Tivi mau SHAP 14
TL01	Tu lanh TOSHIBA 1401
ML01	May lanh 1.5 HP
BU01	Ban ui Philip
QB01	Quat ban Hitachi
MG01	May giat SANYO 40
ND01	Noi com dien SANYO
HD02	Xe cub 86
XD01	Xe dap NHAT

4. Tính giá trị trường THANHTIEN của tập tin NHAPVT.DBF

5. Tính tổng số tiền nhập của mỗi loại vật tư có chữ cái đầu tiên bên trái giống nhau.

6. Tính SLN, SLX, TONCUOI, sau thời gian nhập xuất trên.

7. Tạo tập tin TONKHO99.DBF có cấu trúc giống như TONKHO98.DBF. Lấy TONCUOI của tập tin TONKHO94.DBF để bỏ vào TONDAU của TONKHO99.DBF

8. Liệt kê danh sách Nhập vật tư gồm các mục sau:

MAVT TENVT NGAYNHAP MANX SL

9. Liệt kê danh sách TONKHO gồm các mục sau:

MAVT TENVT TONDAU TONCUOI

11. CHƯƠNG 5: FORMS

11.1. *5.1. KHÁI NIỆM VỀ LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG:*

Thiết kế và lập trình hướng đối tượng là một sự thay đổi đối với phong cách lập trình cũ, lập trình hướng thủ tục.Ở đây thay vì nghĩ đến các chức năng của chương trình ta chỉ cần nghĩ đến các đối tượng đang tạo: là các thành phần độc lập của một ứng dụng cới chức năng riêng của nó. Mỗi một đối tuợng đều có một bộ thuộc tính mô tả đối tượng; các phương thức là những đoạn trình chứa trong điều khiển, cho điều khiển biết cách thức để thực hiện một đoạn công việc nào đó; và tập hợp những sự kiện đó là những phản ứng của đối tượng.

Trong Visual Foxpro, các form và control là các đối tượng được dùng để xây dựng các ứng dụng

5.1.1. Thuộc tính của đối tượng (Properties)

Để chỉ đến một thuộc tính của đối tượng nào ta dùng cú pháp sau:

<tên đối tượng>.<thuộc tính>

Ví dụ: Myform.caption= “Chương trình ứng dụng”

Các thuộc tính thông dụng:

- Left: Vị trí cạnh trái của đối tượng so với vật chứa nó.
- Top: Vị trí trên của đối tượng so với vật chứa nó.
- Height: Chiều cao của đối tượng.
- Width: Chiều rộng của đối tượng.
- Name: Tên để chỉ đối tượng.
- Enable: Giá trị logic:

    True: có quyền làm việc.

    False: Không có quyền làm việc.

- Visible: Giá trị logic:

    True: Thấy được đối tượng;

False: Không thấy được đối tượng.

5.1.2. Phương thức của đối tượng (Methods)

Để gọi đến phương thức của một đối tượng, ta dùng cú pháp:

<tên đối tượng>.<phương thức>

Ví dụ: Myform.show

Một số phương thức thường dùng:

- Refresh: Làm tươi lại đối tượng.
- Show: Hiện đối tượng.
- Hide: ẩn đối tượng.
- Release: Giải phóng đối tượng.
- SetFocus: Thiết lập "tầm ngắm" cho đối tượng.

5.1.3. Sự kiện của đối tượng

Để chỉ đến sự kiện của đối tượng, ta dùng cú pháp sau:

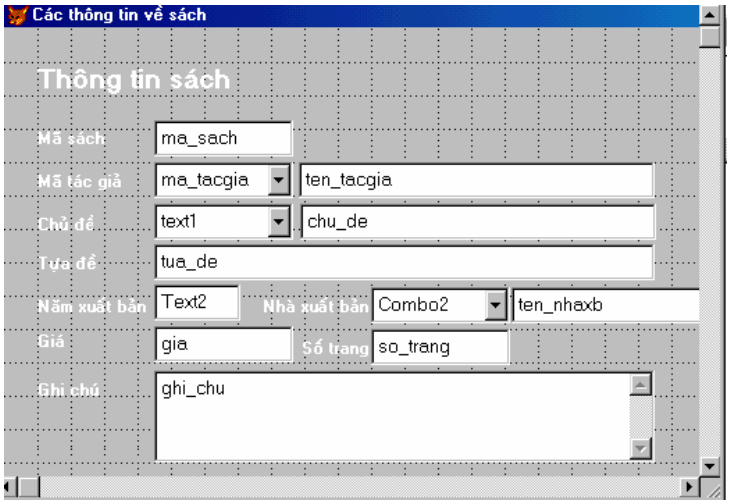
<tên đối tượng>.<sự kiện>

Một số sự kiện thường dùng:

- Click: Được gọi khi kích chuột vào đối tượng.
- DbClick: Được gọi khi kích đúp chuột vào đối tượng.
- MouseMove: Được gọi khi di chuyển chuột trên bề mặt của đối tượng.
- KeyPress: Được gọi khi nhấn một phím kích chuột vào đối tượng.
- Got focus: Được gọi khi đưa đối tượng vào tầm ngắm.
- Lostfocus: Được gọi khi đưa đối tượng ra khỏi tầm ngắm
- Change: Được gọi khi có sự thay đổi nội dung dữ liệu kiểu chuỗi của đối tượng.

11.2. *5.2. FORM*

5.2.1. Giới thiệu



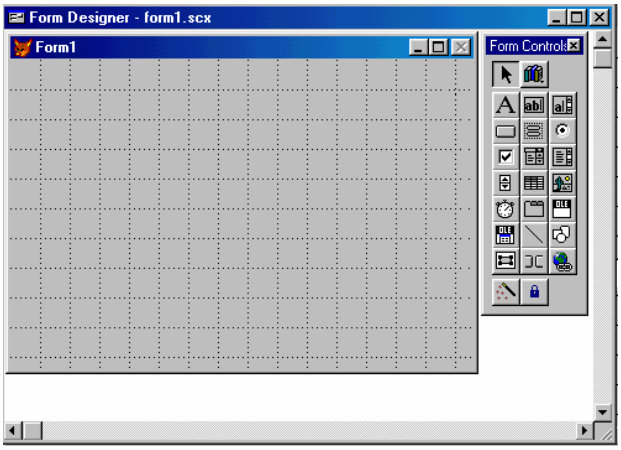
Form được dùng để làm giao diện nhập, hiển thị thông tin, nó cung cấp một tập hợp các đối tượng để đáp lại những thao tác của người dùng làm cho ứng dụng ra dáng chuyên nghiệp.

Ví dụ: Giao diện của một Form nhập dữ liệu

5.2.2. Tạo form thông qua Wizard

Từ menu Tools, chọn wizard, chọn form, xuất hiện giao diện wizard selection, rồi thông qua hướng dẫn.

5.2.3. Tạo form thông qua thiết kế



Để tạo form thông qua thiết kế, từ cửa sổ lệnh ta thực hiện lệnh sau:

CREATE FORM <tờn form>

Khi đó ta được màn hình thiết kế form như sau:

a. Quản lý form

- *Lưu Form*: Từ menu file, chọn save để lưu vào <tên form>, mặc định phần mở rộng là scx.
- *Chạy form*: Từ cửa sổ lệnh, thực hiện lệnh sau:

DO FORM <tên form>

- *Đóng form* (giải phóng khỏi bộ nhớ)

RELEASE <tên form>

b. Tuy cập đến các đối tượng trên form

+ Muốn chỉ đến một đối tượng nào trên form, ta dùng:

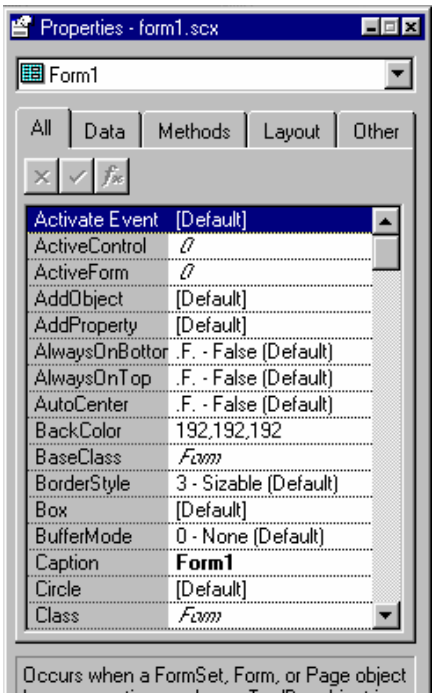
<tên form>. <đối tượng>: nếu <đối tượng> không cùng với form đang thao tác.

<this form>.<đối tượng>: nếu đối tượng nằm trên form đang thao tác.

+ Muốn thay đổi giá trị các thuộc tính trên form, ta dùng:

<tên form>.<thuộc tính>= <giá trị>: nếu muốn thay đổi thuộc tính của form không phải là form hiện hành.

<This form>.<thuộc tính>=<giá trị>: nếu muốn thay đổi các thuộc tính của form hiện hành.



c. Các thuộc tính, phương thức, sự kiện thường dùng trên form.

- *Thuộc tính:*

- BackColor: Màu nền
- BorderStyle: Dạng đường viền
- Caption: Tiêu đề của form
- FillColor: Màu để tô đối tượng
- Fontname: Font chữ cho các đối tượng chứa văn bản
- Fontsize: Kính thước fontname
- Moveable: Cho phép di chuyển hay không

.....

- *Tình huống*

- MoveMouse: Đáp ứng khi di chuyển chuột trên bề mặt form
- Destroy: Đáp ứng khi giải phóng form
- Load: Đáp ứng khi nạp form vào bộ nhớ

5.3. Thanh công cụ Control Toolbar

- Muốn đưa đối tượng trên thanh Control vào form: 4 bước

- Kích chuột vào đối tượng cần đưa

- Vẽ nó trên form để xác định vị trí

- Thiết lập các thuộc tính thích hợp

- Viết mã lệnh cho các tình huống tương ứng

- Quy ước đặt tên cho các đối tượng

Loại đối tượng	Tên
Form	Bắt đầu bởi Frm
Command	Bắt đầu bởi cmd
Edit box	Bắt đầu bởi Edb
Grid	Bắt đầu bởi Grd
Image	Bắt đầu bởi Img
Label	Bắt đầu bởi Lbl
Textbox	Bắt đầu bởi Txt
Timer	Bắt đầu bởi Tmr

5.3.2. Một số đối tượng trên Controls

a. Label : Dùng để thể hiện các chuỗi trên form.

Các thuộc tính thường dùng:

- Caption: Chuỗi thể hiện

- Autosize: Giá trị logic, cho phép kích thước của Label có tự động chỉnh sửa theo độ dài của caption hay không.

b. Command Bottom: Dùng để thể hiện các nút lệnh trên form.

Các thuộc tính thường dùng:

- Caption: tên xuất hiện trên nút lệnh
- Picture: Hình xuất hiện trên nút lệnh
- Enable: giá trị Logic, cho phép chọn nút lệnh hay không

Các Sự kiện thường dùng:

- Click: Khi kích chuột vào nút lệnh thì sự kiện này được gọi.

c. TextBox: Dùng để xem, chỉnh sửa dữ liệu từ các trường trong bảng dữ liệu không phải kiểu memo.

Các thuộc tính thường dùng.

- ControlSource: Tên của trường hay biến mà giá trị của nó được hiện trong textbox
- Value: Giá trị hiện thời của textbox.

Sự kiện thường dùng:

- Change: Khi có sự thay đổi của thuộc tính value
- KeyPress: Khi có phím bất kỳ được ấn.

d. Editbox: Tương tự như textbox, được dùng để chỉnh sửa dữ liệu từ các trường memo.

Các thuộc tính thường dùng.

- Control Source: Tên của trường mà giá trị của nó được thể hiện trong editbox.
- ScrollBars: Có hiện thanh cuộn trong khung editbox hay không.
- ReadOnly: Cho phép có được chỉnh sửa nội dung hay không

Sự kiện thường dùng:

- Change: Khi có sự thay đổi của thuộc tính value.
- Keypress: Khi có phím bất kỳ được ấn.

e. Images: Dùng để đưa các hình ảnh trên form.

Các thuộc tính thường dùng.

- Picture: Xác định file hình ảnh

- Stretch: Xác định cách thức thể hiện hình ảnh (phóng to, thu nhỏ, nguyên mẫu).

f. Timer: Dùng để thiết lập các công việc thực hiện đều đặn sau một khoảng thời gian.

Các thuộc tính thường dùng.

- Enabled: Xác định xem Timer có hiệu lực hay không

- Interval: Quy định khoảng thời gian xác định cho tình huống timer.

Sự kiện thường dùng:

- Timer: Được kích hoạt đều đặn sau một khoảng thời gian xác định ở thuộc tính Interval.

g. Grid: Dùng để thể hiện dữ liệu theo dạng bảng.

Các thuộc tính thường dùng.

- Row Source: Xác định bảng dữ liệu cần thể hiện.

- ColumnCount: Xác định số cột của Grid.

Chú ý:

Nếu Row Source không được chỉ ra thì lấy bảng dữ liệu hiện hành.

Nếu Column count không chỉ ra thì mặc định là tất cả các trường trong bảng dữ liệu (Column count=-1).

Control Source: Được xác định cho từng cột, dùng để khai báo nguồn dữ liệu cho cột đó.

- Allow Addnew: Cho phép thêm các bản ghi mới hay không.

Chú ý:

Muốn thay đổi các thuộc tính trên Grid thì chuyển Grid sang dạng edit bằng cách nhấn phím phải chuột lên Rrid, chọn Properties. Grid đang ở chế độ Edit có một đường viền bao quanh.

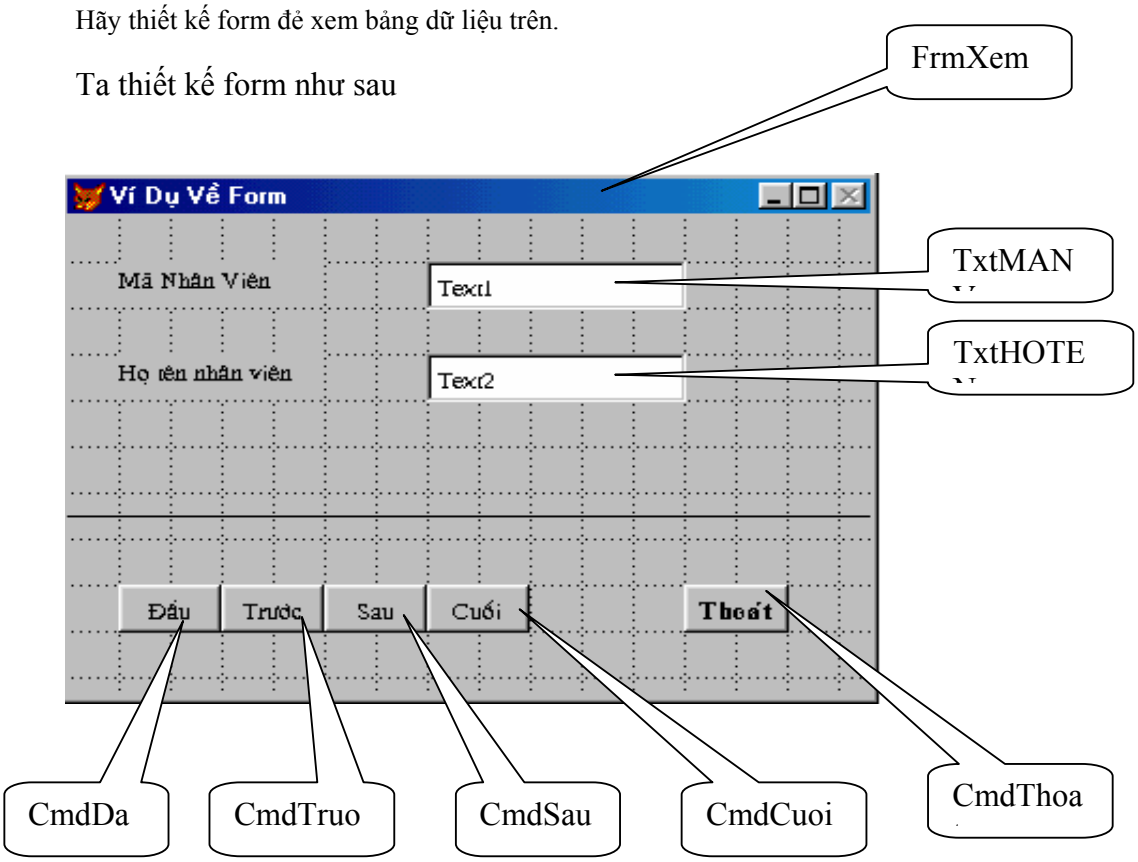
12. **5.4. Ví dụ**

Giả sử có bảng dữ liệu với cấu trúc:

MANV	C	5
HOTEN	C	30

Hãy thiết kế form đẻ xem bảng dữ liệu trên.

Ta thiết kế form như sau



13. **Các** thuộc tính chính :

14. + TxtMANV các thuộc tính ControlSource là MANV

15. + TxtHOTEN các thuộc tính ControlSource là HOTEN

16. Mã lệnh của các đối tượng trên Form là:

+ FrmXem.load

use hoso

+ CmdDau.Click

go top

thisform.refresh

+ CmdCuoi.Click

go bottom

thisform.refresh

+ CmdTruoc.Click

if not bof()

skip -1

endif

thisform.refresh

+ CmdSau.Click

if not eof()

skip

endif

thisform.refresh

+ CmdThoat.Click

use

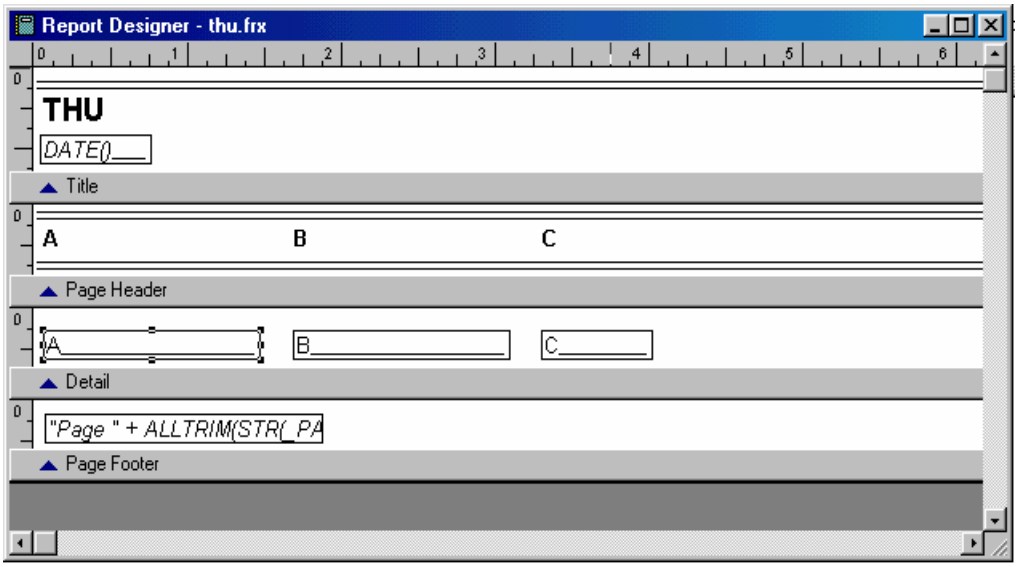
thisform.release

17. CHƯƠNG 6 REPORTS

17.1. *6.1. KHÁI NIỆM*

Reports là công cụ để trình bày và tóm tắt dữ liệu trong một văn bản khi in. Report có hai thành phần cơ bản cấu thành: dữ liệu nguồn, thông thường là các bảng dữ liệu và hình thức trình bày là dạng thức của report sẽ định dạng cách kết xuất dữ liệu.

Màn hình thiết kế Report



17.2. *6.2. CÁC BƯỚC ĐỂ TẠO REPORT*

Ta có thể thiết kế report để thể hiện dữ liệu ở nhiều dạng thức khác nhau trên giấy khi in. Quá trình thiết kế gồm 4 bước chính như sau:

1. Xác định loại Report cần tạo: Tức là quyết định chọn dạng thức mà report hiển thị kết quả.

2. Tạo Report layout: Có thể sử dụng report wizard hay report designer. Report layout được lưu trên đĩa với phần mở rộng của file là FRX: Lưu trử chi tiết của report.

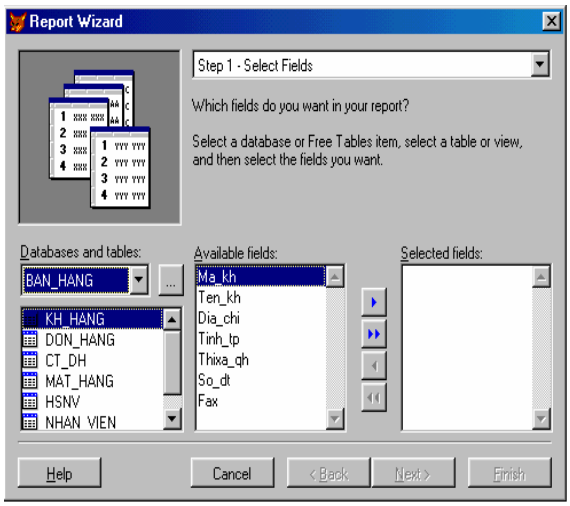
3. Sửa đổi layout của report.

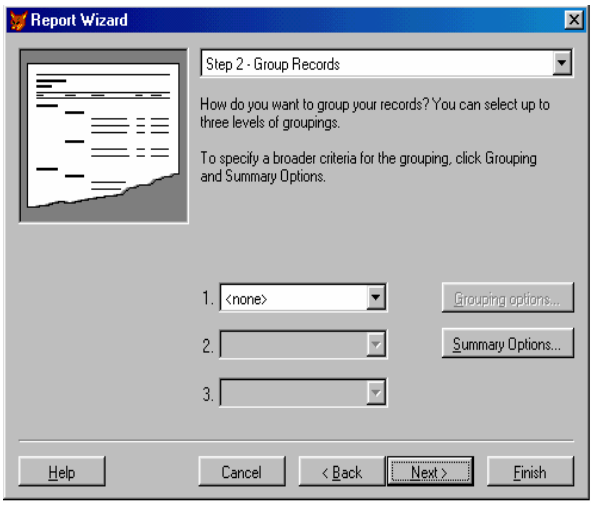
4. Xem và in report.

17.3. *6.3. TẠO REPORT BẰNG WIZARD.*

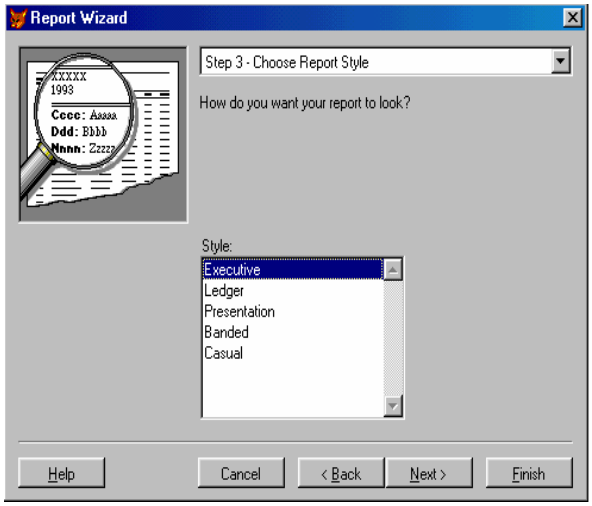
Từ menu Tools, chọn Wizard, chọn Report sau đó làm theo các bước hướng dẫn.

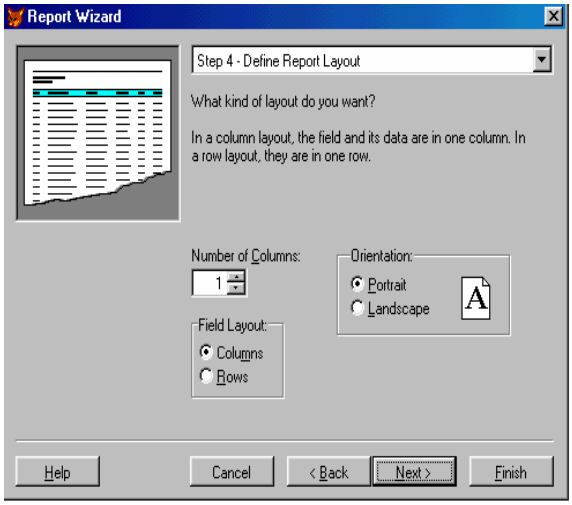
Bước 1: Chọn bảng dữ liệu và các trường cần thể hiện





Bước 2: Tạo nhóm dữ liệu kết xuất

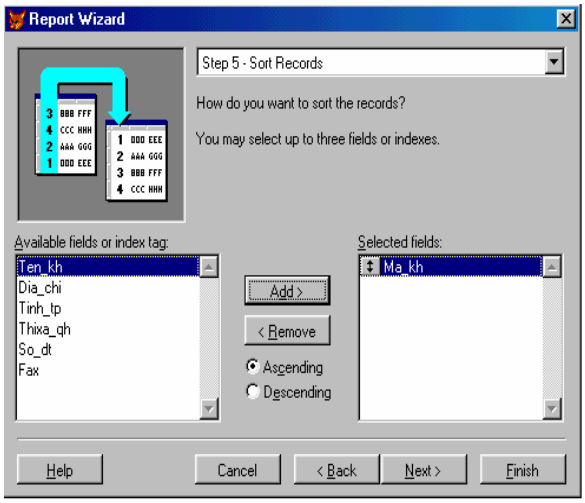


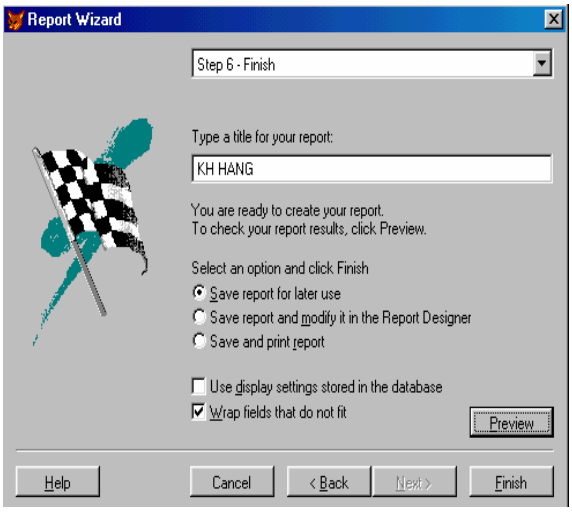


Bước 3: Chọn kiểu Report thể hiện

Bước 4: Chọn cách trình bày trên giấy in

Bước 5: Chọn trường Sắp xếp





Bước 6: Đặt tựa đề, kết thúc

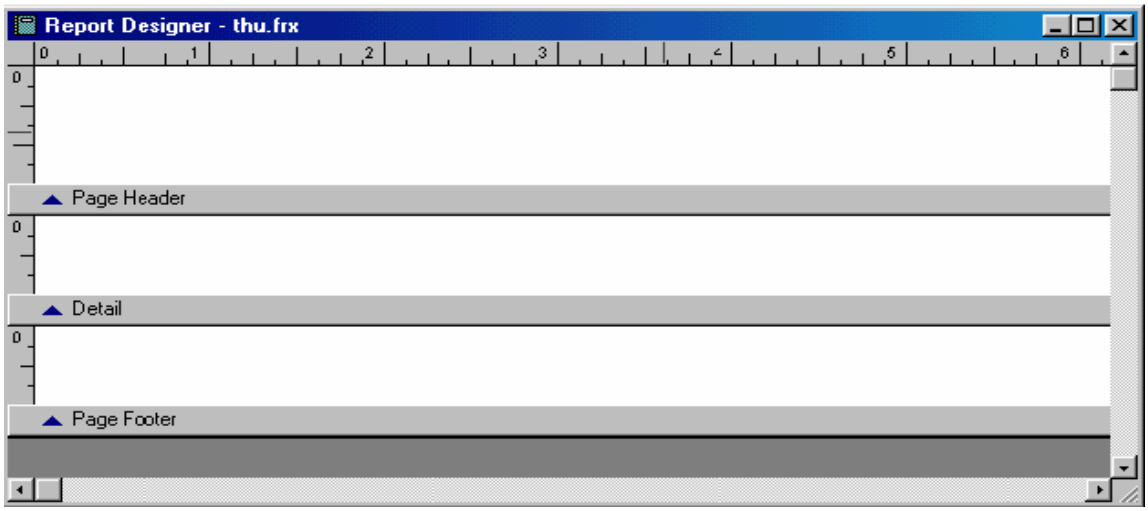
17.4. *6.4. TẠO REPORT BẰNG REPORT DESIGNER*

6.4.1. Quản lý Report

- Tạo mới Report: CREATE REPORT <tên Report>

Ví dụ: create report THU

Lúc này màn hình xuất hiện hộp thoại report

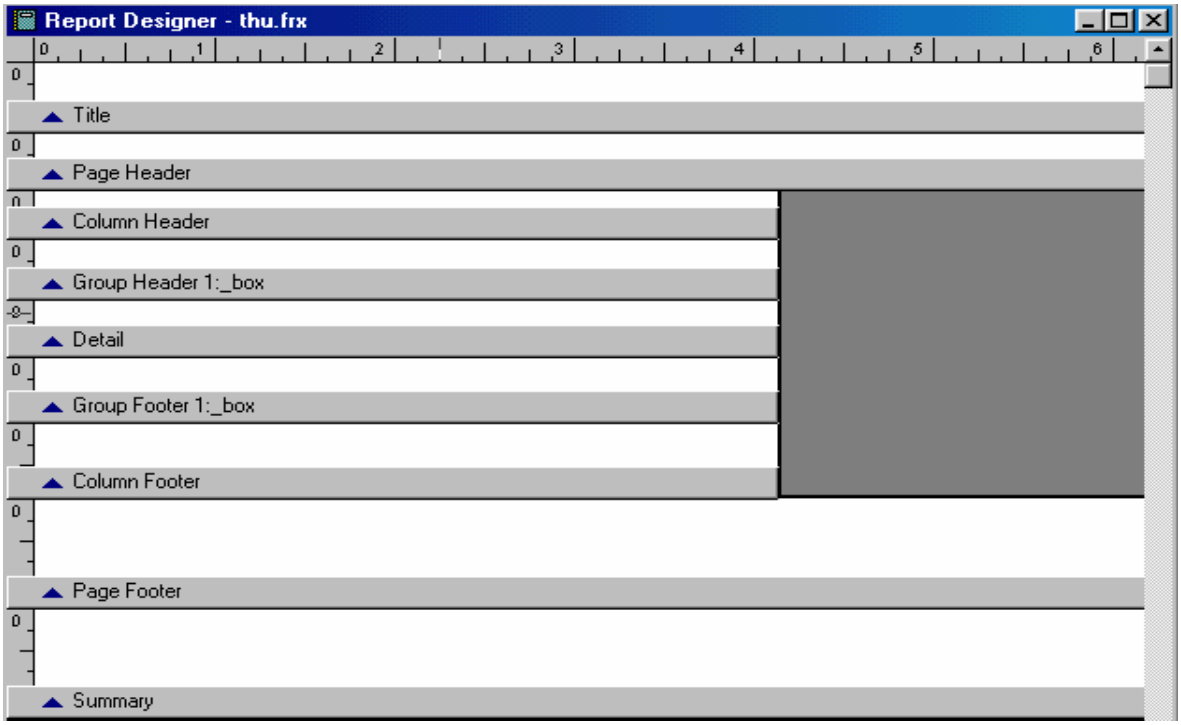


- Mở một report sẵn có: MODIFY <tên report>

- Xem trước khi in: REPORT FORM <tên report> PREVIEW
- Xem trước khi in có điều kiện:

  REPORT FORM <tên report> PREVIEW <điều kiện>
- In report: REPORT FORM <tên report> TO PRINTER

6.4.2. Các thành phần trên Report



- Title: Dùng để in trên mỗi report: Từ menu report, chọn title summary
- Page Header: Để in trên mỗi header của mỗi trang in.
- Column header: Để in tên header của mỗi cột. Để chọn, từ menu file chọn page setup, chọn giá trị cho column number lớn hơn 1.
- Group header: Xuất hiện mỗi khi bắt đầu nhóm mới. Để chọn, từ menu report chọn data grouping.
- Detail: phần chi tiết trên mỗi record (ứng với từng record trên bảng dữ liệu).
- Group footer: In phần Footer của mỗi nhóm. Để chọn, từ menu report chọn data grouping.

- Column footer: In phần Footer của mỗi cột. Để chọn, từ menu file, chọn page setup, chọn giá trị cho column nimber lớn hơn 1.
- Page Footer: In phần Footer của mỗi trang.
- Sumary: Phần tốm tắt của mỗi report.

6.4.3. Các control trên Report

Thanh công cụ Report Control

Chức năng của các control:

Field trong bàng dữ liệu, biến và các biểuthức toán	Field
Text thuần tuý	Label
Đường kẻ	Line
Hộp và đồng khung	Rectangle
Hình tròn, elip	Rounded Rectangle
Hình ảnh hoặc field General	Picture

6.4.4. Đưa các control vào report

Thực hiện các bước sau:

+ Chọn control thích hợp

+ Kéo rê chuột trên report để xác định vị trí của nó trên report

+ Hiệu chỉnh các control

a. Đưa field vào report: abl

+ Kích chuột vào

+ Trong hộp report Expression, chọn nút lệnh sau hộp Expression.

+ Trong hộp field, hcọn tên trường hay biến thích hợp.

+ Chọn OK.

b. Đưa label vào report:

+ Chọn 

+ Gỏ nội dung của label

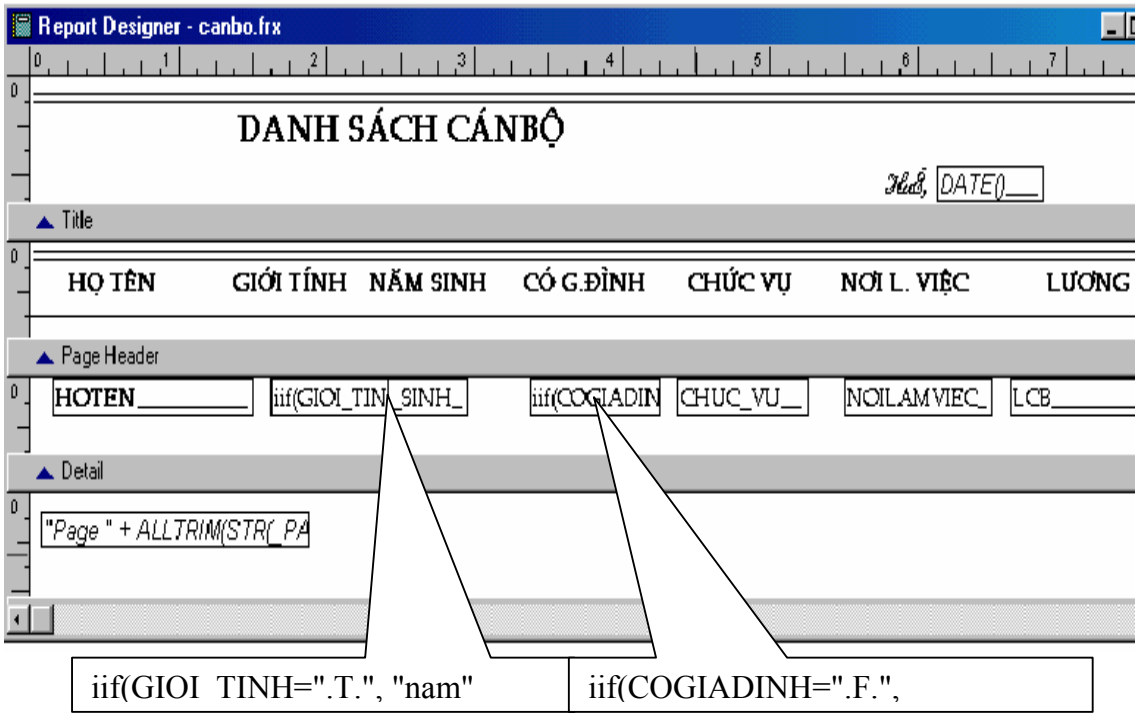
c. Đưa Picture bound control vào report:

+ Chọn picture bound control

+ Xuất hiện hộp hội thoại report picture, chọn file, nếu muốn chèn hình ảnh từ file, chọn field nếu muốn chọn trường General.

+ Chọn Ok

6.5. Ví dụ Thiết kế Report như sau (dựa vào Bảng CANBO.DBF ở bài tập 2):



18. CHƯƠNG 7. TẠO MENU VÀ QUẢN LÝ ĐỀ ÁN

18.1. *7.1. TẠO MENU*

18.2. *7.1.1. GIỚI THIỆU*

Menu cung cấp một phương thức có cấu trúc và giao diện với người dùng để tác động lên những câu lệnh trong ứng dụng.

Việc sắp xếp và thiết kế menu thích hợp sẽ giúp cho người dùng được thuận lợi khi sử dụng hệ thống menu của bạn.

18.3. *7.1.2. CÁC BƯỚC TẠI MỘT MENU HỆ THỐNG*

1. Sắp xếp và thiết kế: Quyết định menu nào bạn cần chúng xuất hiện ở vị trí nào trên màn hình, cần những menu con nào?

2. Sử dụng menu designer, tạo menu và các Submenu.

3. Gắn các câu lệnh tương ứng với công việc.

4. Biên dịch menu

5. Tiến hành chạy thử, kiểm tra.

18.4. *7.1.3. TẠO MENU HỆ THỐNG*

7.1.3.1. Quản lý menu hệ thống

Menu hệ thống được lưu trử tên đĩa với file có phần mở rộng là *.MNX

- Tạo menu bằng công cụ Designer Menu: Thực hiện lệnh:

CREATE MENU <tên menu> ↵

- Mở menu đã có: MODIFY MENU <tên menu>

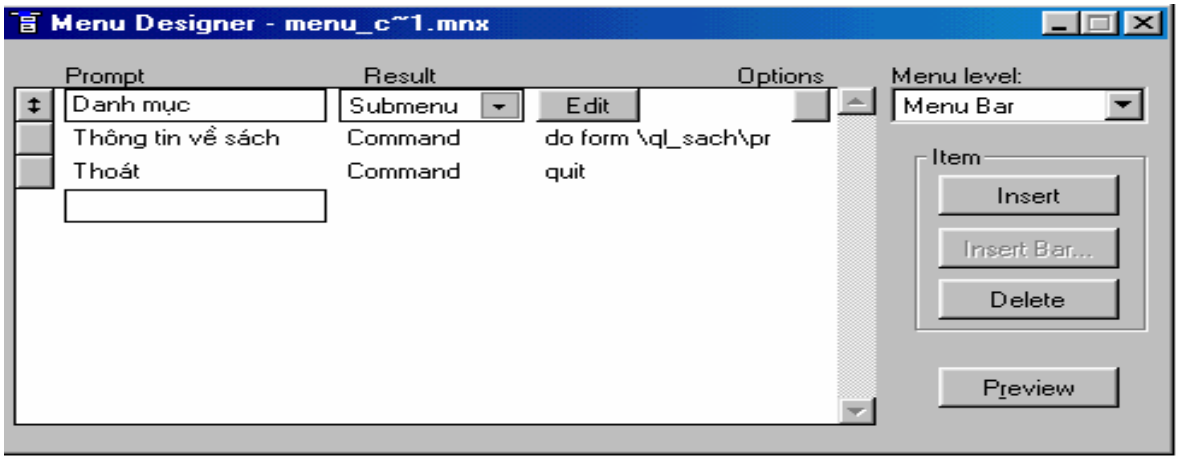
- Dịch file Menu: Để dịch file menu, từ màn hình Menu Designer chọn lệnh Generate.

File menu sau khi dịch sẽ có phần mở rộng là MPR.

7.1.3.2. Tạo menu hệ thống thông qua Menu Designer

Sau khi thực hiện lệnh Create menu, ta được màn hình giao diện Menu: Designer như

sau:



+ Trong hộp Prompt, ta đưa vào tên cần hiển thị trên giao diện.

+ Trong hộp Result, chọn:

- Submenu nếu muốn tạo menu con.

- Procedure nếu muốn thi hành thủ tục

- Command n□u mu□n th□c hi□n m□t l□nh.

+ Kết thúc, ấn Ctrl_W.

18.5. *7.2.QUẢN LÝ ĐỀ ÁN*

18.6. *7.2.1. KHÁI NIỆM ĐỀ ÁN*

Đề án là tên gọi để chỉ đến ứng dụng mà bạn đang xây dựng. Thông thường các thành phần của một đề án bao gồm:

+ Các bảng dữ liệu (table).

+ Các file cơ sở dữ liệu (database)

+ Các form

+ Các report

+ Các query

+ Các file khác như âm thanh, hình ảnh, tài liệu, hình ảnh con trỏ,...

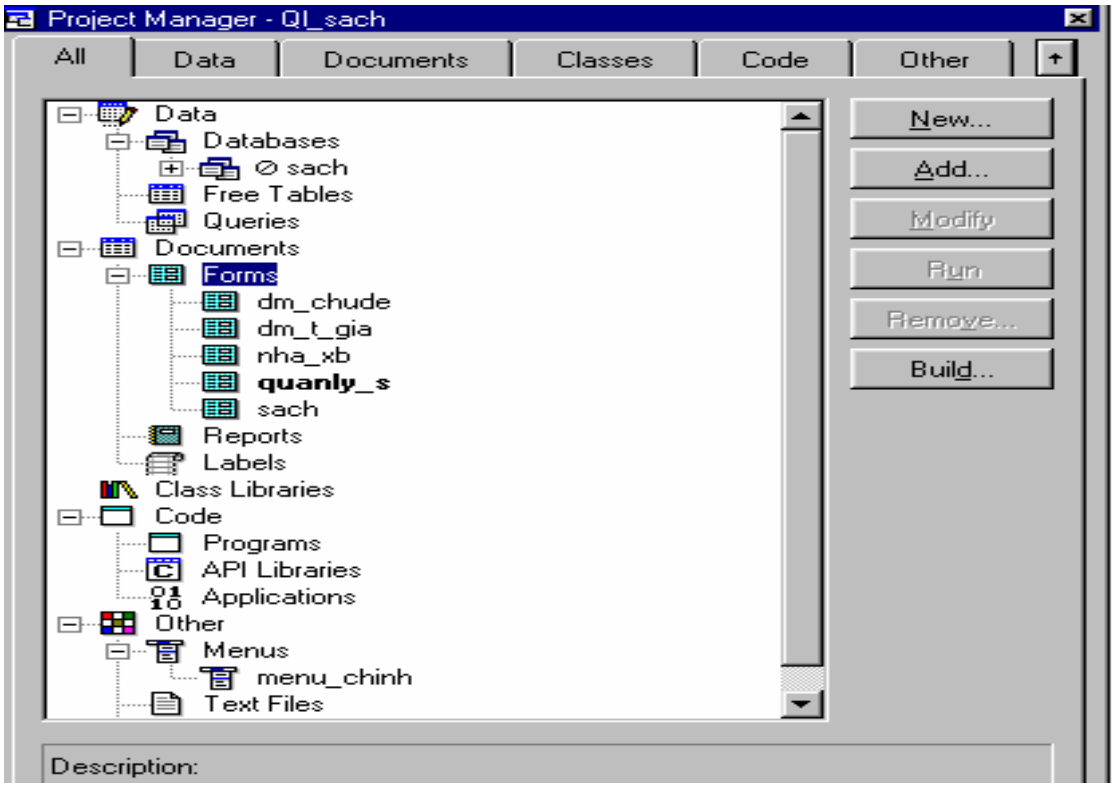
18.7. *7.2.2. QUẢN LÝ ĐỀ ÁN*

Một đề án trong Visual Foxpro được lưu trữ trên file có phần mở rộng mặc định là *.PRJ.

7.2.2.1. Tạo mới các đề án

Thực hiện lệnh: CREATE PROJECT <tên đề án>

Lúc này xuất hiện cửa sổ quản lý đề án Project Manager như ở trên.



Database: Bao gồm các:

Table: Các bảng dữ liệu có liên kết với nhau hay các bảng tự do.

Query: Là cấu trúc để lấy thông tin từ các bảng table.

View: Là các Query chuyên dụng mà ta có thể truy xuất dữ liệu cục bộ và từ xa cho phép cập nhật các nguồn dữ liệu bằng cách làm thay đổ Report bởi quyre.

+ Documents: Chứa các tài liệu sử dụng cho đề án; bao gồm các form và report.

+ Class: Liệt kê các thư viện được sử dụng.

+ Code: và những file khác: Liệt kê các file chương trình và các file khác được sử dụng trong chương trình.

Để chỉnh sửa bất kỳ một thành phần nào trong đề án ta chọn nó rồi chọn nút Modify.

Để thêm bất kỳ một file nào cho đề án ta kích nút add (nếu chọn file đã có) hoặc nút new (nếu tạo mới).

Muốn loại bỏ bất kỳ một thành phần nào của đề án ta chọn nó rồi chọn nút remove.

7.2.2.2. Mở một đề án đã có

Thực hiện lệnh: MODIFY PROJECT <tên đề án>

7.2.2.3. Dịch đề án

+ Dịch sang APP: Khi này, để chọn đề án phải có một bản sao của Visual Foxpro.

Dùng lệnh BUILD <tên đề án>

+ Dịch sang file có phần mở rộng là exe: Khi này, người dùng không cần có Visual Foxpro nhưng phải cung cấp hai file: vfp6r.dll và vfp6renu.dll được cài đặt trong đường dẫn hoặc trong cùng thư mục với ứng dụng.

Dùng lệnh: BUILD EXE <tên đề án>

7.2.2.4. Chạy đề án

Sau khi đã dịch, ta có thể chạy đề án thông qua lệnh: DO <tên ứng dụng>

18.8. ***7.2.3. ĐẶT STARTING POINT CHO ĐỀ ÁN***

Khi ứng dụng được thi hành, có một điểm bắt đầu, đó là Starting point.

Để chọn một thành phần của dự án là Starting point:

+ Chọn thành phần được đặt là Starting point.

+ Từ Menu Project, chọn Set main.

Thông thường, Starting point là một chương trình khởi động chứa các thành phần:

Do setup.prg

Do mainmenu.mpr

Read Events

Do cleanup.prg

a. Do Setup.prg: L□ th□c hi□n ch□□ng trỡnh □□ thi□t l□p mụi tr□□ng cho h□ th□ng.

b. Do mainmenu.mpr: Chạy file menu chính để thiết lập giao diện cho hệ thống.

c. Read Events: Bắt đầu thực hiện vòng lặp để thực hiện công việc.

d. Do cleanup.prg: Chạy chương trình dọn dẹp môi trường, trả lại môi trường cho hệ thống và thoát khỏi hệ thống. ổ đây, phải có lệnh Clear Events để thoát khỏi vòng lặp đã được thiết lập bới lệnh Read Events.