dBASE II
程式寫作

蔡文賢・陳光宏　譯

dBASE II
程式寫作

蔡文賢・陳光宏　譯

儒林圖書公司　印行

dBASE Ⅱ 程式寫作

譯　者：蔡文賢、陳光宏
發行人：楊　鏡　秋
出版者：儒　林　圖　書　有　限　公　司
地　址：台北市重慶南路一段 111 號
電　話：3812302　3110883　3140111
郵政劃撥：0106792-1號
吉豐印刷廠有限公司承　印
板橋市三民路二段正隆巷 46 弄 7 號
行政院新聞局局版台業字第 1492 號

中華民國七十四年七月初版

定價新台幣　230　元正

前　言

dBASE II 是一功能極強的資料庫語言。它的特點是能以簡單的命令來建立資料庫結構與增添（ add ），更改（ change ）及刪除（ delete）資料項。dBASE II可讓操作者搜尋（ search ），選取（ select ）與顯示（ display ）資料庫的記錄（ records ），且可簡單地設計與列印報表。對於許多使用者而言，這些功能已足夠應付他們處理簡單的資料庫查詢與報表列印。然而，對另一個層次的使用者來說dBASE II却是一種非常好用的程式語言。

dBASE II問世不久即成爲微電腦領域中廣受歡迎的“程式語言”。對許多人而言，它已取代了BASIC 或PASCAL 。在大企業裏，它甚至取代了COBOL，FORTRAN與PL/1。dBASE II 包含了幾乎解決任何資料處理問題的完整程式語言。因爲dBASE II使用了專門的資料庫技術，所以使得企業程式規劃比其他傳統語言還簡單。

所有平常的程式設計技巧均能以dBASE II來達成，譬如決策（ decision making ），迴圈（ looping ），排序（ sorting ），搜尋（ searching），選擇（ selecting ），顯示（ displaying ），資料處理（ data manipulating ）與顧客報表（ custom reporting ）。對於全螢幕的功能選擇表（ selection menus)與資料輸入螢幕（ data entry screen ），dBASE II 均能簡單地設計與執行。您可將許多資料庫程式命令寫進dBASE II程式中，而系統操作環境我們可以用簡單的命令加以設定。如果程式撰寫完畢後，想要更改檔案記錄描述（ record description），也能簡單地加以更改，而如果是使用非資料庫語言，則這種更改將是非常的複雜。資料庫查詢語言不但可用來排序，搜尋與選擇記錄；也可作爲程式命令，以便取代傳統語言的許多程式敍述列。dBASE II是眞正八十年代的第四代程式語言。

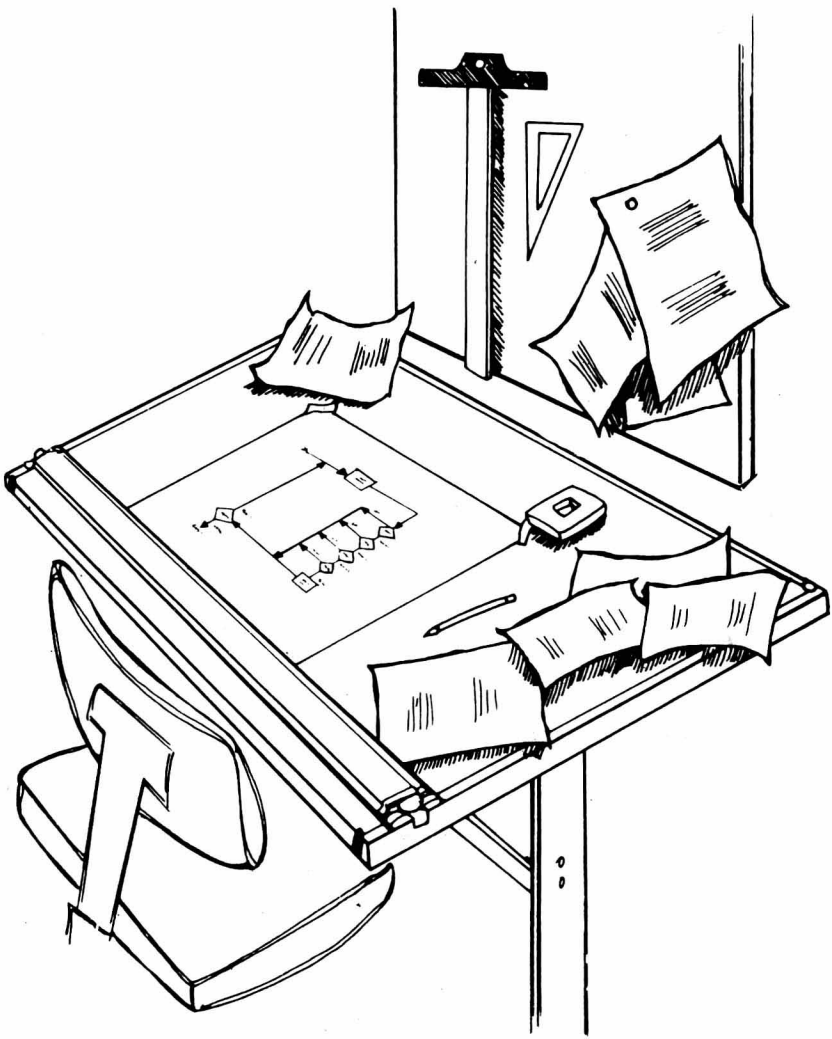
本書專爲電腦初學者以及想再學新語言的老練程式設計師而設計，亦對伙計與老闆均有幫助。希望讀者在閱讀之前先具備粗淺的dBASE II概念，並且有

建立資料庫與編製簡單報表的經驗。初學者將會發現此書是一本簡介程式設計觀念與資料庫技術的好書。老練的程式設計師將發現本書有助於將其他語言轉換成dBASE II。

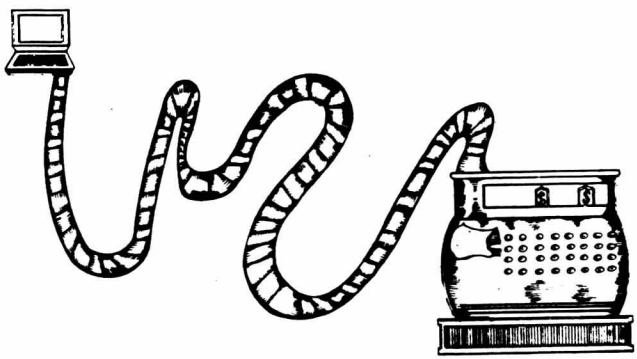
本書內容分成三篇：基本應用系統設計，基本資料庫設計，以dBASE Ⅱ設計應用系統。在每一篇裏，您將遇到本書的主角Fred。Fred擁有一魚市場，而本書藉Fred電腦化的範例，來教您如何以dBASE II來設計應用系統。“基本應用系統設計”該篇將教您如何設計應用系統（不僅僅是教您撰寫程式），並比較dBASE II與傳統程式語言之不同。“基本資料庫設計”該篇解釋資料庫之意義，以及如何有效率地（efficiently）與有效益地（effectively）設計與運用資料庫。最後一篇，“以dBASE Ⅱ設計應用系統”此篇劃分成七章。每一章分別教您如何運用dBASE Ⅱ來執行某一特定功能。每一章都是獨立的課題，並可與前面討論過的課題整合起來。這使得讀者能分別了解各個主題，也能將之組合成完整的程式。不管是初學者或專家，讀完本書就能以dBASE Ⅱ來設計應用系統與撰寫應用程式，以便解決他們各自的問題。

目　錄

第一篇
基本應用系統設計



1



程式設計的基礎

資料庫程式是使用特殊語言敍述來處理資訊。其目的是將所要的結果呈現於報表、圖形或表格上。未經處理的資訊，程式設計師稱之爲資料（data）。

資料與資訊

簡單的說，資料是資訊的原始形態；意即，獨立的事件與數字極少甚或沒有意義。電腦程式的目的便是要將資料整理成更有意義及／或有用的形式。

例如，你看到一列數字 199462482010454，此數字顯然毫無意義（也許有某種含義，端看您如何分割）。當其被割開成爲 199462482-010454 時，此數列仍然是資料，意即此劃分並沒有產生新的資訊。若此數字再劃分成最小部份 199-46-2482-01-04-54，則顯然是社會保險號碼及日期，若這資料再標成：

SNN DATE
199-46-2482/01-04-54

你看剛剛沒有意義的資料，已現出其意義了，無意義的資料已變成資訊，此例即說明資料與資訊間的差異。

資料型態

電腦程式的目的是處理資料，將其轉換成資訊。這些被處理的資料有兩種基本型式：字元的（character）及數值的（numeric）。

字元資料，如名稱、地址、電話號碼及任何以同一型式儲存並用於報告中的其他項目。字元資料項可以多種方式處理，但不能用於數學計算。例如，名字可儲存成姓、名的次序，但顯現在報表上的是“名、姓”（first，last）的格式。此例的處理中包含有組成資料項之字元的重組。任何使用此名字資料的加、減、乘、除，或執行其他型式的數學計算的企圖均是無意義的，且極可能會出現錯誤。

數值資料以稍異的方式儲存，此型的資料可用於數學計算。數值資料項包含有金錢總額、日期、任何事項的計數、或其他可用於計算的資料項。數值資料可用於加總、平均或用於計算以獲得另一資料項。

出現在電腦程式中的字元或數值資料，我們稱之爲運算式（expressions），而不論它們是單獨或與其他資料項結合，運算式只能有一種資料型態。使用字元資料於數值運算式的方式稱爲混合模式（mixing modes），在程式裡是不合法的。

變　數

變數是用於程式裡的記憶區（area of storage），用來保存資料項。變數的使用可賦予電腦程式基本的變通性。由於有了變數，電腦程式才能運算變數所擁有的任何值。

變數可以想成是一“桶子”能夠容納程式所要用的值。於此方式，程式可能要指定一變數稱爲NAME，存入任何值。無需知道其確實內容，NAME變數便可定位於一輸出報表上，顯示於標題部份，或用於其他所需的功能上。若 NAME 變數的最大長度是 25 個字元，程式的輸出（印出的報表或螢幕的顯示爲最普通）必須

能夠容納 25 個字元。

例如一個記錄（record）之內包含一NAME（ 25 個字元）及一 PHONE 號碼（ 10 個字元）。檔案的第一個記錄可以讀成

```
CARY PRAGUE        2035557685
```

第二個記錄讀成

```
JIM HAMMITT        2035556281
```

現在，若要第一個記錄，其NAME的值將是 CARY PRAGUE 而 PHONE 的值是 2035557685 。若程式的目的是印出姓名與電話清單，程式首先讀入一個記錄，印出變數 NAME 及 PHONE 的值，然後再跳回去讀下一個記錄。藉由使用 NAME 及 PHONE 兩變數，此程式變得用途廣泛；即可執行任何設定相同變數（NAME 及 PHONE ）於相同形式的記錄。

使用變數操作的能力，是一般電腦最有用的特性。電腦可以這些值在程式內快速作業，而我們人類却無法如此。

變數的處理是電腦程式最重要的功能，而變數的觀念對於程式運作方式的了解亦極爲重要。當變數被規定爲一特定值，其值將留在變數內，直到被一新值取代或程式運作結束爲止。改變變數值的唯一方法是以一新值取代之。因此，若產生一叫 TITLE 的變數來做爲輸出報表的標題且不再改變它，其將從頭到底一直使用其內容，並維持其值不變。

定　字

另一種情況是使用寫在程式中的定值，這種型態的值稱爲定字（literal），其無法被程式處理。定字是直接書寫於運算式中。

請看下列運算式例子，其計算是將華氏溫度轉換成攝氏溫度：C = (F – 32)*5/9 。

在此運算式裡，F 及 C 是變數（即其值可改變），而 32、5 及 9 是定字，因爲其值不再改變。在如上的數學運算式裡，等號 (=) 是用來指示方程式左右兩半部的

相等；在電腦語言，這種方程式叫做指定敍述（assignment statement），並不用於表明相等性，它是告訴電腦，用敍述右邊的運算值取代左邊的變數值。

敍 述

電腦語言是由敍述所組合而成的。前一段所提的那一個敍述，是指定敍述，因爲其指派一新值到變數C上去。不論F爲何值，當此特殊敍述被執行時，其計算將採用F值及定字值來執行，計算結果即被存入變數C中。此程序並不改變F的值，亦不變動定字值，只有等號左邊的變數被改變。電腦程式內的敍述大都是指定敍述。

在dBASEⅡ的指定敍述是

```
STORE [expression] TO variablename
```

攝氏溫度轉換的計算將成爲：

```
STORE (F-32)*(5/9) TO CELCIUS
```

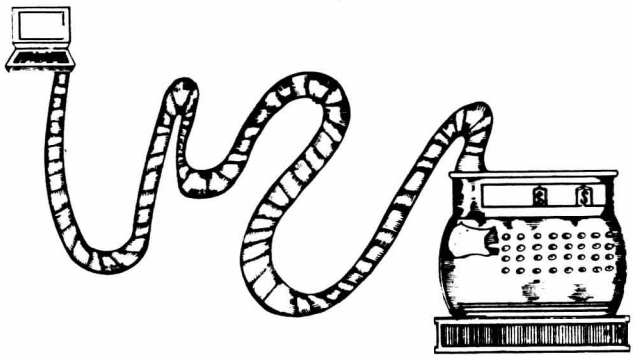
此種格式的指定敍述比前一段所述的傳統格式更具有可讀性。其動作的發生是由左而右： STORE value TO variable.

敍述型態

在電腦語言裡，一般有四種敍述型態。除指定敍述外，尚有決策敍述（decision statements）、廻圈控制敍述（loop control statements）及輸入/輸出敍述（input/output statements）。

電腦語言敍述是用來產生行動的。指定敍述是將新值指派至變數中，這些新值可以爲計算的結果。決策敍述可以依據變數值來執行或跳過一段敍述。廻圈敍述則可以重複執行一系列的敍述，直到某一條件成立爲止，例如一計算器之值超過目標值。輸入/輸出敍述是命令電腦將資料自外部設備送出或讀入。總之，不同的電腦語言敍述則產生不同型式的反應。

2



應用系統發展

程式的最終結果是“輸出”，它是寫程式的首要需求。要確認一程式的撰寫正確，則必須在寫程式之前先定義好程式所要產生的資訊。如同所有解決問題的方法（包括科學方法），其尋求解答的第一步驟是對問題有一明確的定義，此爲以電腦程式解決問題七步驟中的第一步驟。此七步驟是：

1. 問題定義（ Define the problem ）
2. 輸出定義（ Define the output ）
3. 輸入定義（ Define the input ）
4. 程序定義（ Determine the process ）
5. 程式撰寫（ Code the program ）
6. 測試程式（ Test the program ）
7. 評估程式（ Evaluate the program ）

企業問題

企業問題發生於目前一些方法在執行企業活動，不再能夠產生可接受的結果時。則其發生可能有各種原因：人力的減少、工作負荷的增加、立即擷取資料的需求，或改變工作方法的需求。在一公司裡，如果包括總經理只有四個人的話，則其薪水作業可以不需要電腦來處理。然而，如果公司擴充到四百人就非常有必要了，因爲對一小群人而言，當其處理一大團體的工作時，不論採用什麼方法均可能是一繁重的工作。

另一問題的例子將使一小公司需要電腦。譬如那家四人公司與兩百家公司進行交易，而存貨高達數百萬美元之多時，除非那四個人整個星期均做存貨控制而不做其它事，否則會萬事無成。此例可能過份誇大，但其指明一點：處理商務的自動化方法可能有其必要。一旦電腦安裝了，若時間與資源允許，則存貨控制、應付薪工均可實施電腦化。

問題定義

不論企業上所需爲何，尋找解答的第一個正確步驟是定義出應做什麼、應何時做、應做多少等。這目標的整體敍述是保證尋求解答的各步驟能針對企業問題，並可避免錯誤的發生及副作用。

一個問題的定義必須說明爲什麼需要此行動，及期望自解答得到什麼結果。問題的定義亦應避免因爲某些限制而排除某解決方案，亦即不管是否用電腦，均不應含有用什麼工具來得到解答的提示。

預期結果的說明，對問題的定義亦具重要性。這說明不應模糊不清而應盡可能完整，其有助於決定要產生所需的結果時，需要那些資訊。若要用電腦來求解，可能需要大量的資料輸入工作，來使所需的資訊可自電腦系統而取得。在估算新系統的成本時，必須將資料輸入時間也列入計算。

一個書寫完整的問題定義要能減少在臆測上的時間浪費，此可更迅速地得到解