2

Für jeden Anlass ein Subfile-Typ

Wie in Kapitel 1 erwähnt wurde, gibt es drei verschiedene Arten von Subfiles. Der bisher beschriebene Typ war eine „Load-All"-Subfile, bei der alle Datensätze auf einmal in die Subfile geladen werden können, bevor die Subfile angezeigt wird. Ein derartiges Programm ist deshalb eine Load-All"-Subfile, weil OS/400 die Seitenverarbeitung für Sie übernimmt und alle Datensätze zu Beginn des Programms angezeigt werden, bevor der Benutzer die Daten sieht. Wie Sie sehen, ist im RPG-Programm kein Code vorhanden, der es Ihnen ermöglicht, in der Subfile nach oben oder nach unten zu blättern. Es gibt nur eine Routine, die alle Daten aus Ihrer Datendatei lädt, bis das Ende der Datei erreicht wird oder bis die Maximalzahl der in der Subfile erlaubten Datensätze (500 in diesem Beispiel) erreicht ist.

Dieser Typ Subfile kann problemlos entwickelt werden und ermöglicht Ihnen, in kürzester Zeit eine Anwendung zu programmieren. Es ist hervorragend, genau zu wissen, wieviele Datensätze Sie anzeigen möchten. Da OS/400 für Sie die Seitenverarbeitung übernimmt, können Sie sich ganz auf das Laden und die Anzeige konzentrieren. Der Benutzer kann durch die Daten blättern, und Sie müssen dafür in Ihrer RPG keine einzige Codezeile schreiben.

Aber wie immer klingt das alles zu schön, um wahr zu sein. Das Problem bei allen „Load-All"-Subfiles entsteht mit zunehmenden Datenmengen. Wenn mehr Datensätze geladen werden müssen, wird das Programm zwar nicht explodieren - wir haben für diesen Fall einige Zeilen in der DOW-Schleife kodiert - aber die Daten werden nicht angezeigt. Ich könnte die DDS modifizieren und das Schlüsselwort SFLIZ auf den Wert 9.999 setzen, und anschließend in das RPG-Programm wechseln und die Konstante SUBFILE_SIZE ändern, um dem Schlüsselwort SFLSIZ zu entsprechen. Nun werden alle Daten geladen - vorausgesetzt, dass die Anzahl der Datensätze 9.999 nicht übersteigt. Aber denken Sie nur an den armen Benutzer, der durch endlose Seiten mit Daten blättern muss, um das Gewünschte zu finden.

Das ist ein weiteres potenzielles Problem für das „Load-All"-Programm und seiner OS/400-gesteuerten Seitenverarbeitung. Der Benutzer muss eventuell durch mehrere Seiten blättern, um die gewünschten Daten zu erhalten. Das kann sehr zeitraubend sein, besonders, wenn sich die Daten eher am Ende der Liste befinden. Ein letzter Fallstrick dieser Technik sind die möglichen Leistungsprobleme, die auftreten, wenn derart große Datenmengen auf einen Schlag geladen werden. Nicht nur dauert es eine ganze Weile, bis diese Tausende Datensätze in eine Subfile geladen werden, sondern der Prozess beansprucht außerdem die Systemressourcen.

Der Einsatz der „Load-All“-Technik muss sorgfältig geplant werden. Sie müssen berücksichtigen, wieviele Datensätze angezeigt werden sollen, wieviele Seiten der Benutzer durchblättern kann, um die gewünschten Informationen zu erhalten, und Sie dürfen die möglichen Leistungsprobleme nicht vergessen, die mit dem Laden der maximal zulässigen Datenmenge verbunden sind. Nehmen wir also an, Sie wissen nicht genau, wieviele Datensätze angezeigt werden, oder Sie kennen zwar die Anzahl, aber diese ist relativ hoch (z.B. 1.000). Wie schreiben Sie am besten ein Subfile-Programm, das sowohl alle Datensätze anzeigt, als auch den leichten Zugriff durch den Benutzer ermöglicht? Dies geschieht, indem Sie den zweiten der drei Subfile-Typen verwenden: Selbsterweiternde Subfiles.

Selbsterweiternde Subfiles

Da Sie nach wie vor nur bis zu 9.999 Datensätze laden können (dies soll ein deutlicher Hinweis darauf sein, dass es eventuell eine Möglichkeit gibt, mehr als 9.999 Datensätze zu laden), funktioniert die selbsterweiternde Subfile auf die gleiche Weise wie eine Load-All-Subfile. Sie unterscheidet sich jedoch in mehreren anderen Punkten. Eine selbsterweiternde Subfile erlaubt dem Benutzer das Hinzufügen von Datensätzen zur Subfile nur bei Bedarf. Normalerweise geschieht dies jeweils für eine Seite, obwohl das nicht unbedingt sein muss. Ein Vorteil der selbsterweiternden Subfile gegenüber der „Load-All"-Subfile besteht darin, dass die Subfile-Leistung konsistenter ist. Indem jeweils nur eine bestimmte Datenmenge geladen wird, ist eher zu erwarten, dass Ihre Subfile wie erwartet ausgeführt wird. Eine Load-All-Subfile kann einmal 50 Datensätze und ein andermal 500 Datensätze laden, abhängig von Ihren Daten. Ein weiterer Vorteil liegt darin, dass ich, indem ich eine selbsterweiternde Subfile und eine Positionierungstechnik verwende, die Navigation innerhalb der Subfile flexibler gestalten kann.

Sehen wir uns die DDS und das RPG-Programm an und vergleichen wir diese mit der bereits besprochenen Load-All-Technik. (Die komplette DDS und RPG für die hier verwendeten Programme finden Sie am Ende dieses Kapitels. Ich werde Ihnen nur die Teile des Codes zeigen, die für frühere Beispiele relevant sind oder sich von ihnen unterscheiden.)

Als erstes beschäftigen wird uns mit der DDS. SFL002DF sieht nicht viel anders als die „Load-All"-DDS aus, mit Ausnahme von wenigen Änderungen. Abbildung 2.1 zeigt den Subfile-Steuerdatensatz der DDS, der der einzige Teil der DDS ist, der Änderungen aufweist.

```
A             R SF1CTL                    SFLCTL(SFL1)
A*
A                                         SFLSIZ(0018)
A                                         SFLPAG(0017)
A                                         OVERLAY
A                                         ROLLUP(27)
A N32                                     SFLDSP
A N31                                     SFLDSPCTL
A        31                               SFLCLR
A        90                               SFLEND(*MORE)
A             RRN1            4S 0H       SFLRCDNBR
A                                       4  2'Nachname'
A                                          DSPATR(HI)
A                                       4 26'Vorname'
A                                          DSPATR(HI)
A                                       4 50'MI'
A                                          DSPATR(HI)
A                                       4 55'Rufname'
A                                          DSPATR(HI)
A                                       1  2'SFL002RG'
A                                       1 71DATE
A                                          EDTCDE(Y)
A                                       2 71TIME
A                                       1 25'Selbsterweiterndes Subdatei-Programm'
A                                          DSPATR(HI)
```

Abbildung 2.1: DDS Steuerdateidatensatz für eine selbsterweiternde Subfile (SFL002DF).

Wie Sie sehen, habe ich SFLSIZ von 500 auf 18 geändert. Das bedeutet nicht viel, sowohl die Load-All- als auch die selbsterweiternden Subfiles erfordern, dass SFLSIZ größer als SFLPAG ist. Was wirklich eine Rolle spielt, ist die Tatsache, wie die Subfile in den RPG-Code geladen wird. Ich hätte auch hier 500 verwenden können, aber ich gebe bei selbsterweiternden Subfiles standardmäßig einen um eins größeren Wert für SFLSIZ als für SFLPAG an und lade in mein RPG-Programm nur jeweils eine Seite mit Datensätzen.

Außerdem habe ich das Schlüsselwort ROLLUP hinzugefügt. Damit wird es meinem RPG-Programm möglich, den Zeitpunkt zu steuern, zu dem der Subfile weitere Datensätze hinzugefügt werden. Bei einer selbsterweiternden Subfile ist es wichtig, zu beachten, dass die Steuerung des Blätterns durch die Daten vom RPG-Programm und OS/400 gemeinsam übernommen werden. Durch Hinzufügen des Schlüsselworts ROLLUP habe ich OS/400 angewiesen, dass jedes Mal, wenn der Benutzer bei der Anzeige der letzten Seite der Subfile die Taste Bild Ab drückt, um mehr Daten zu erhalten, mein Programm die Steuerung übernimmt und die Seitenverarbeitung steuert. Wenn der Benutzer sich nicht auf der letzten Seite befindet oder wann immer der Benutzer durch die Daten blättert, möchte ich, dass OS/400 die Seitenverarbeitung übernimmt. Darüber erfahren Sie mehr, wenn ich den RPG-Code bespreche.

Nachdem wir uns mit der DDS beschäftigt haben und Sie gesehen haben, dass nur wenige Änderungen erforderlich waren, um eine selbsterweiternde Subfile zu erstellen, untersuchen wir als nächstes das RPG-Programm (SFL002RG). Abbildung 2.2 zeigt die F- und D-Spezifikationen für das Programm.

```
*============================================================
Fsfl002df           cf   e         workstn
F                                              sfile(sfl1:rrn1)
Fsfl001lf           if   e         k disk

Dsflpag                      C                 const(17)

Dlstrrn                      S     4           0 inz(0)
```

Abbildung 2.2: Standard F- und D-Spezifikationen in den Subfile-Programmen (SFL002RG)

Wie Sie sehen, habe ich ein Standalone-Feld mit der Bezeichnung LSTRRN vorgesehen. Dieses Feld wird für die Aufnahme der letzten relativen Datensatznummer verwendet werden, die in die Subfile während der Subfile-Laderoutine geschrieben wird. Wenn der Benutzer die Taste Seite nach unten drückt, weiß das Programm, wo es in der Subfile mit dem Schreiben der Datensätze beginnen muss.

Beachten Sie auch, dass ich das Bereinigen der Subfile vom Laden der Subfile getrennt habe, indem ich die Aufgaben in zwei separate Subroutinen gestellt habe. Dies geschieht in einem selbsterweiternden Subfile-Programm, da ich die Subfile nicht jedes Mal, wenn ich Datensätze laden möchte, neu bereinigen möchte. Da der Subfile Datensätze hinzugefügt werden soll, wenn der Benutzer die Taste Seite nach unten drückt, möchte ich die Datei nicht zuerst bereinigen. Wir kommen gleich zum Hauptteil des Programms, aber sehen wir uns zuerst die Subfile-Routine für die Bereinigung an, wie in Abbildung 2.3 dargestellt.

```
*****************************************************************
*        CLRSFL - Bereinigen der Subdatei
*****************************************************************
*
C         clrsfl       begsr
*
* Relative Datensatznummern und Subdatei bereinigen
*
C                      eval        rrn1 = *zero
C                      eval        lstrrn = *zero
C                      eval        *in31 = *on
C                      write       sf1ctl
C                      eval        *in31 = *off
C                      eval        *in32 = *off

C                      endsr
```

Abbildung 2.3: Bereinigungsroutine für die selbsterweiternde Subfile (SFL002RG)

In der CLRSFL-Subroutine setze ich die relative Datensatznummer (Relative Record Number oder RRN1) auf 0 und setze außerdem die letzte relative Datensatznummer (Last Relative Record Number - LSTRRN) ebenfalls auf 0. Ich werde LSTRRN verwenden, um den letzten relativen Datensatz der Subfile aufzunehmen, so dass ich beim Hinzufügen von Datensätzen zur Subfile weiß, wenn ich das Ende erreicht habe.

```
*****************************************************************
*         SFLBLD - Erstellung der Liste
*****************************************************************
*
C         sflbld       begsr
*
* Macht RRN1 = zur letzten relativen Datensatznummer der Subdatei,
* so dass der Ladeprozess die Datensätze korrekt ganz unten anhängt
*
C                      eval        rrn1 = lstrrn
*
* Laden der Subdatei mit einer Seite Daten bis zum Ende der Datei
*
C                      do          sflpag
C                      read        sfl001lf
C                      if          *in90
C                      leave
C                      endif
C                      eval        rrn1 = rrn1 + 1
C                      write       sfl1
C                      enddo
*
C                      if          rrn1 = *zero
C                      eval        *in32 = *on
C                      endif
*
C                      eval        lstrrn = rrn1
*
C                      endsr
```

Abbildung 2.4: Laderoutine für die selbsterweiternde Subfile (SFL002RG).

Nachdem ich den Indikator 31 aktiviert habe („on“), um das SFLCLR-Schlüsselwort bedingt auszuführen, schreibe ich in das Subfile-Steuerdatensatzformat, um die Subfile zu bereinigen. Anschließend deaktiviere ich Indikator 31, der das Schlüsselwort SFLDSP bedingt. Meine Subfile-Laderoutine (siehe Abbildung 2.4) unterscheidet sich etwas von der Laderoutine in der Load-All-Subfile. Der Grund dafür ist, dass ich nur jeweils eine Seite von Daten laden werde und nicht alle verfügbaren Datensatz von meiner Datenbankdatei. In diesem Fall wurde das Feld SFLPAG als eine Konstante in meinen D-Spezifikationen festgelegt und wird die gleiche Zahl enthalten, die auch mit dem Schlüsselwort SFLPAG in der DDS verwendet wird. Wenn Sie sich dafür entscheiden, das Schlüsselwort in der DDS zu ändern, müssen Sie die D-Spezifikation im RPG-Programm ebenfalls ändern. Lassen Sie sich nicht von der Tatsache täuschen, dass sie den gleichen Namen haben, das ist nur eine meiner üblichen Arbeitstechniken.

Wenn das Ende der Datei erreicht ist, kommt Indikator 90 ins Spiel, und ich verlasse die Schleife. Indikator 90 bedingt auch SFLEND. Ansonsten inkrementiere ich den relativen Datensatz mit eins und schreibe in das Subfile-Datensatzformat. Sobald die Schleife ihre Arbeit beendet, indem sie entweder die vorgeschriebene Anzahl von Datensätzen in die Subfile schreibt (in diesem Fall 17) oder auf das Dateiende trifft, prüfe ich, ob ich die Subfile durch den Bedingungsindikator 32 anzeigen und anschließend LSTRRN entsprechend konfigurieren sollte.

Zurück in der Hauptroutine sehen Sie eine komplexere DOU-Schleife, die in der Load-All-Subfile enthalten ist. Abbildung 2.5 zeigt den Haupt-Code.

```
*
* Bereinigen und anschließend Erstellung der initialen Subdatei
*
C                   exsr              clrsfl
C                   exsr              sflbld
*

* Schleife für die Verarbeitung der Subdatei, bis F3 oder F12 gedrückt
wird
*
C                   dou               (*inkc = *on) or (*inkl = *on)
*
C                   write             fkey1
C                   exfmt             sf1ctl
*
C                   select
*
* Hinzufügen weiterer Datensätze zur Subdatei, bis sich der Benutzer
ganz unten befindet
*
C                   when              *in27 and not *in32
C                   exsr              sflbld

C                   endsl
C                   enddo

C                   eval              *inlr = *on

*
```

Abbildung 2.5: Hauptroutine für die sich selbsterweiternde Subfile (SFL002RG).

Das war nicht sehr schwierig, ich habe einfach den Code für die Verarbeitung der Taste Seite nach unten dem ursprünglichen Load-All-Teil hinzugefügt. Vergessen Sie nicht, dass ich mit dem Schlüsselwort ROLLUP in der DDS den Wert 27 angegeben habe, was bedeutet, dass beim Drücken der Bild Ab (ROLLUP bzw. „Seite nach unten“) Indikator 27 aktiviert wird. Wenn Indikator 27 aktiviert ist und die Subfile nicht leer ist (*IN32 wurde mit „off“ deaktiviert), führe ich einfach die SFLBLD-Subroutine aus, die das Hinzufügen von Datensätzen zur Subfile vom letzten Punkt aus beginnt.

Kehren wir für einen Moment zum Befehl EXFMT zurück. Wie Sie sich sicherlich erinnern, sagte ich, dass bei einer selbsterweiternden Subfile die Seitenverwaltung von Ihrem Programm und OS/400 gemeinsam übernommen werden. An dieser Stelle findet die Entscheidung statt. Ihr Programm wird solange in dieser Codezeile bleiben, bis Sie eine gültige Funktionstaste drücken (wie in Ihrer DDS definiert) oder bis die Eingabetaste gedrückt wird. Wenn jedoch die Taste Bild Ab gedrückt wird, ist OS/400 intelligent genug, zu wissen, ob es durch Daten blättern wird, die sich bereits in der Subfile befinden, ein Fall, in dem OS/400 die Verarbeitung übernehmen kann. Wenn die letzte Seite der Subfile bereits angezeigt wird, weiß OS/400, dass die Steuerung an Ihr Programm zurückgegeben werden muss.

Bei selbsterweiternden Dateien übernimmt immer OS/400 die Verarbeitung für den Befehl Bild Auf (ROLLDOWN bzw. „Seite nach oben“). Grundsätzlich bedeutet das, dass die erste Seite der Daten angezeigt wird, wenn Sie Ihr Programm zum ersten Mal aufrufen. Wenn Sie die Taste Bild Ab drücken, um mehr Datensätze zu erhalten, wird die Steuerung an Ihr Programm zurückgegeben und 17 weitere Datensätze werden in die Subfile geschrieben. Damit sind 34 Datensätze in Ihrer Subfile, und Seite zwei

(Datensätze 18 bis 34) wird angezeigt. Wenn Sie die Taste Bild Auf drücken, um zur ersten Seite der Daten zurückzukehren, wird die Navigation durch OS/400 ausgeführt, und da diese Datensätze bereits in der Subfile vorhanden sind, unternimmt das Programm nichts. Wenn Sie beschließen, wieder nach unten zu blättern, um zur zweiten Seite zu gelangen, übernimmt OS/400 auch das für Sie, da die Datensätze 18 bis 34 bereits vorhanden sind. Wenn Sie eine dritte Seite sehen möchten - Sie haben es erraten - wird die Steuerung an Ihr Programm zurückgegeben, das Ihrer Subfile 17 neue Datensätze hinzufügt. Die selbsterweiternde Subfile ist der Typ Subfile, der vermutlich am häufigsten verwendet wird. Sie verteilt die Zuständigkeit für die Verarbeitung gleichmäßig zwischen OS/400, das einen Teil der Verarbeitung übernehmen darf, und dem Programmierer, dem erlaubt wird, etwas Code hinzuzufügen, um die Flexibilität für den Benutzer zu erhöhen.

ETWAS PEPP GEFÄLLIG?

Nachdem Sie nun die Grundlagen für selbsterweiternde Subfiles beherrschen, möchte ich Ihnen einige Techniken zeigen, die Sie verwenden können, um Ihre Subfile-Programme für den Benutzer flexibler zu machen und die Programmierung für den Programmierer einfacher und leichter verständlich zu gestalten.

Ich z.B. stelle in meinen Subfile-Programmen meinen Benutzern immer gern ein zusätzliches Navigationstool zur Verfügung. Es kann z.B. vorkommen, dass ein Benutzer einen Namen sucht, der mit „V“ beginnt, und die Datei Tausende von Namen enthält, die alphabetisch aufgelistet sind, was dazu führt, dass der Prozess sehr lange dauert. Es wäre schön, wenn der Benutzer sofort zu den Namen wechseln könnte, die mit „V“ beginnen, um von diesem Punkt aus alle Seiten zu durchsuchen. Noch besser wäre es, wenn ein Benutzer, der den genauen Namen kennt, sofort zu dieser Seite wechseln könnte, ohne die Seitentasten zu verwenden. Das Positionierungsfeld „Position-to“ ermöglicht diese Flexibilität.

Dieses Feld ist nicht unbedingt erforderlich, aber ich füge bei der Erstellung einer Subfile in das Subfile-Steuerdatensatzformat immer ein Positionierungsfeld ein. Für mich ist das Schreiben eines Subfile-Programms ohne Positionierungsfunktionalität das gleiche, wie wenn man dazu gezwungen wird, im Lift in jeder Etage eines Hochhauses einen Halt einzulegen, obwohl man im Erdgeschoß wohnt. Das Hinzufügen der Positionierungsfunktionalität gibt Ihnen die Fähigkeit, direkt in das Erdgeschoss zu wechseln, in dem Sie aussteigen möchten.

Das Positionierungsfeld bietet dem Benutzer die Fähigkeit, die exakte Position in der Subfile auszuwählen, zu der er navigieren möchte. Wenn die ”A”s in meiner Namensliste angezeigt werden, kann ich in mein Positionierungsfeld einen Namen oder den Teil eines Namens eingeben, der mit „Z“ beginnt, und die Subfile springt zum „Z“, oder zu dem Namen, der sich in direkter Nähe von „Z“ befindet, wenn in der Datei keine „Z“s vorhanden sind. Sehen wir uns noch einmal das Subfile-Steuerformat in SFL002RG an. Dieses Mal habe ich ein Positionierungsfeld mit dem Namen PTNAME hinzugefügt. Abbildung 2.6 zeigt das erweiterte Steuer-Datensatzformat.

```
A              R SF1CTL                    SFLCTL(SFL1)
A*
A                                          SFLSIZ(0016)
A                                          SFLPAG(0015)
A                                          OVERLAY
A                                          ROLLUP
A N32                                      SFLDSP
A N31                                      SFLDSPCTL
A         31                               SFLCLR
A         90                               SFLEND(*MORE)
A              RRN1           4S 0H        SFLRCDNBR
A                                       6  2‘Nachname’
A                                          DSPATR(HI)
A                                       6 26‘Vorname’
A                                          DSPATR(HI)
A                                       6 50‘MI’
A                                          DSPATR(HI)
A                                       6 55‘Rufname’
A                                          DSPATR(HI)
A                                       1  2‘SFL002RG1’
A                                       1 71DATE
A                                          EDTCDE(Y)
A                                       2 71TIME
A                                       1 24‘Subdatei-Programm mit Positionierung’’
A                                          DSPATR(HI)
A                                       4  2‘Positionierung auf Nachname. . .’
A              PTNAME         20     B   4 30CHECK(LC)
```

Abbildung 2.6: Subfile-Steuerformat mit Feld „Positionieren in“ PTNAME (SFL002DF1)..

Um für das Positionierungsfeld Platz zu schaffen, habe ich das Schlüsselwort SFLPAG von 17 auf 15 geändert und SFLSIZ gemäß meinem Standard für selbsterweiternde Subfiles, SFLSIZ = SFLPAG + 1, von 18 auf 16 reduziert. Jetzt habe ich genügend Platz, um das Positionierungsfeld PTNAME in Zeile 4 des Bildschirms unterzubringen. Ich definiere dieses Feld als 20-Byte-Feld, was der Feldlänge des Felds NACHNAME in meiner Datendatei entspricht, und ermögliche die Eingabe in das Feld, indem ich in Position 38 ein „B“ eingebe. Das Schlüsselwort CHECK mit dem Parameter LC erlaubt dem Benutzer die Eingabe von Kleinbuchstaben (Lower Case).

Die einzige Logik, die im RPG-Programm erforderlich ist, ist ein IF-Block in der Hauptroutine, der feststellt, ob im Positionierungsfeld Eingaben vorgenommen wurden. Wenn dies der Fall ist, wird die Subfile mit der Routine CLRSFL bereinigt, die Datendatei wird mit einer SETLL-Operation unter Verwendung des Felds PTNAME positioniert, die Subfile wird geladen, indem die Routine SFLBLD aufgerufen wird und das Feld PTNAME wird bereinigt, bevor die Subfile auf dem Bildschirm angezeigt wird. Abbildung 2.7 zeigt eine Beispiel des Code-Blocks, den Sie hierfür verwenden können.

```
C                     if          ptname    *blanks
C         ptname      setll       sfl001lf
C                     exsr        clrsfl
C                     exsr        sflbld
C                     clear                 ptname
C                     endif
```

Abbildung 2.7: Haupt-Logik zur Verarbeitung des Positionierungsfelds (SFL002RG1)

Gleichgültig, an welcher Stelle Sie sich in Ihrer Subfile befinden, die Subfile wird gelöscht, wenn Sie in das Feld PTNAME Daten eingeben, und 15 neue Datensätze werden geschrieben und angezeigt, abhängig davon, was in das Feld PTNAME eingegeben wurde. Die Abbildungen 2.8 und 2.9 zeigen, was geschieht, wenn das Positionierungsfeld verwendet wird.

Abbildung 2.8 zeigt die Datei, die mit dem Nachnamen ”Anthony” beginnt. Wenn Sie die Datei auf ”Vandever” positionieren möchten, geben Sie einfach „Vandever“ in das Positionierungsfeld ein und drücken die Eingabetaste. Die Liste wird neu positioniert, wie in Abbildung 2.9 dargestellt.

```
SFL002RG              Subfile Program with Position To              1/20/00
                                                                   18:08:30

Position to Last Name . . . Vandever

Last Name                First Name               MI   Nick Name
Anthony                  Tony                     A    Triple A
Bert                     Al                       C    Alphabet Man
Coker                    Jim                      L    Da AS/400 Guru
Harrison                 Harry                    H    Happy
Johnson                  John                     J    JJ
Naisium                  Jim                      B    Sweaty
Patterson                Gary                     R    The All-Knowing One
Saint                    Louis                    A    Missouri
Samuelson                Sam                      S    The snake
Simpson                  Othello                  K    Don't call me OJ
Stevenson                Steve                    S    Mike
Tessential               Quinn                    C    Important
Thompson                 Tom                      T    Triple T
Vandever                 Corina                   B    Wife of Subfile Man
Vandever                 Felicia                  R    Cub
                                                                    More...
F3=Exit   F12=Cancel
```

Abbildung 2.8: Subfile-Beispiel mit einer Liste von Namen.

```
SFL002RG              Subfile Program with Position To              1/20/00
                                                                   18:10:31

Position to Last Name . . .

Last Name              First Name             MI   Nick Name
Vandever               Corina                 B    Wife of Subfile Man
Vandever               Felicia                R    Cub
Vandever               Kalia                  M    Koo
Vandever               Kelly                  M    FaderHead
Vandever               Kevin                  M    Subfile Man
Vannerberg             Kern                   X    kevin
Williamson             William                W    The web
Zachery                Zak                    Z    The snoozer









                                                                    Bottom
F3=Exit    F12=Cancel
```

Abbildung 2.9: Die Datei wird neu auf "Vandever" positioniert.

Noch mehr aufgepeppt

Eine letzte Technik, die ich Ihnen zeigen möchte, bevor wir dieses Thema abschließen, ist die Optimierung der Handhabung der Eingabe- und Funktionstasten in Ihrem Programm. Wenn Sie sich Abbildung 2.5 noch einmal ansehen, werden Sie feststellen, dass die Tasten F3 and F12 jeweils durch *INKC und *INKL dargestellt werden, und dass die Taste Bild Ab durch *IN27 repräsentiert wird. Diese Bezeichnungen sind nicht sehr intuitiv und sind als Hinweis auf die Funktion des Codes nur wenig geeignet. Wie Sie auch sicherlich festgestellt haben, wird die Eingabetaste überhaupt nicht dargestellt. Die DDS enthält kein Schlüsselwort für die Eingabetaste und keinen Indikator, der damit verbunden werden kann. Deshalb ist es in einem Anzeigeprogramm erst dann üblich, anzunehmen, dass die Eingabetaste gedrückt wurde, nachdem alle anderen gültigen Funktionstasten und Seitenverarbeitungstasten geprüft wurden. Das ist nicht gerade die effizienteste Methode für die Abwicklung des Anzeigeprogramms, da es sich bei der Eingabetaste um die Taste handelt, die am häufigsten gedrückt wird. Wäre es nicht schön, wenn Sie anschaulichere Bezeichnungen für die Funktionstasten wählen könnten, und eine Methode für die explizite Handhabung der Eingabetaste vor der Prüfung der anderen Tasten möglich wäre? Nun, genau das können Sie tun, wenn Sie die Dateiinformations-Datenstruktur in Ihr Programm aufnehmen und das Attention-Indikator-Byte verwenden, um festzustellen, welche Taste gedrückt wurde.

```
Fsfl002df1 cf            e            workstn
F                                              sfile(sfl1:rrn1)
F                                              infds(info)
Fsfl001lf           if   e            k disk

Dinfo                     ds
D cfkey                        369   369

Dexit                     C                    const(X'33')
Dcancel                   C                    const(X'3C')
Denter                    C                    const(X'F1')
Drollup                   C                    const(X'F5')
Dsflpag                   C                    const(15)
```

Abbildung 2.10: F- und D-Spezifikationen für eine Dateiinformationsstruktur mit einem Attention-Indikator-Byte (SFL002RG1)

Nehmen wir noch einige weitere Änderungen an unserer selbsterweiternden Subfile vor, um diese neue Technik umzusetzen. Als erstes müssen Sie als einzige Änderung an der DDS den Parameter 27 vom Schlüsselwort ROLLUP entfernen. Die Verknüpfung eines Indikators mit ROLLUP ist nicht mehr erforderlich. Nun sehen wir uns die Zusätze an, mit denen das RPG-Programm ergänzt wurde. In Abbildung 2.10 sehen Sie, wie das Attention-Indikator-Byte definiert wird.

Achten Sie auf die zusätzlichen D-Spezifikationen und auf die zusätzliche F-Spezifikation. Die zusätzliche F-Spezifikation definiert eine Datenstruktur (INFO) als die Datei-Informationsdatenstruktur für die Anzeigedatei SFL002DF1. Die INFO-Datenstruktur in den D-Spezifikationen enthält nur ein Feld, CFKEY. Die Dateiinformationsstruktur enthält alle möglichen Daten zur verknüpften Datei, aber ich interessiere mich nur für ein einziges Byte. Wenn der Benutzer eine gültige Taste drückt und die Steuerung zurück an das RPG-Programm gegeben wird, enthält CFKEY eine hexadezimale Darstellung der gedrückten Taste.

Als nächstes konfiguriere ich Konstanten, um den hexadezimalen Werten aussagekräftigere Namen zu geben. Nun kann ich statt INKC als Bedingungscode beim Drücken von F3 das Wort „Beenden“ verwenden. Sie werden verstehen, was ich meine, wenn wir uns mit dem Code näher beschäftigen. Bitte beachten Sie jedoch, dass die soeben erklärten neuen F- und D-Spezifikationen nicht Subfilespezifisch sind. Wenn Sie bereits eine Methode für die Verarbeitung von Anzeigedateien kennen, verwenden Sie diese. Ich möchte an dieser Stelle nur meine Programmiertechniken vorstellen, die für Sie auch dann nützlich sein können, wenn sie sich nicht spezifisch auf Subfiles beziehen.

Abbildung 2.11 zeigt den Zusatz des Positionierungsfelds und die Verwendung der neuen Konstanten mit der ursprünglichen selbsterweiternden Hauptroutine. Beachten Sie die erste Verwendung des Werts CFKEY. Nachdem die Funktionstastenzeile geschrieben wurde und die Subfile wie gezeigt entwickelt wurde, habe ich eine SELECT-Routine kodiert, um die möglichen Benutzerantworten zu berücksichtigen. Ich verwende SELECT und WHEN-Klauseln häufig, da diese mein Programm modularer gestalten - und damit einfacher lesbar machen. Sobald eine WHEN-Klausel erfüllt ist, werden alle übrigen Klauseln ignoriert. Deshalb ist es sinnvoll, die Tasten in der Reihenfolge ihrer möglichen Verwendung zu validieren. Wenn Sie glauben, dass die Eingabetaste am häufigsten gedrückt werden wird, prüfen Sie in Ihrer ersten WHEN-Klausel das Vorhandensein der Eingabetaste. Das ist einer der Vorteile des Attention-Indikator-Bytes, das es Ihnen ermöglicht, eine spezifische Eingabetaste-Prüfung auszuführen. Die Antworten, die ich in der SELECT-Logik berücksichtige, sind Eingabetaste, Seite nach unten (Bild Ab), F3 und F12. Wenn die Eingabetaste gedrückt wird, möchte ich ebenfalls wissen, ob Daten in das Positionierungsfeld PTNAME ein-

gegeben wurden. Wenn die Eingabetaste gedrückt wurde, nachdem der Benutzer Daten in das Positionierungsfeld eingegeben hat, werde ich diese Daten verwenden, um die Datei mit einer SETLL-Operation zu positionieren, die CLRSFL-Routine auszuführen; die Subfile auf Grundlage des Positionierungseintrags zu laden und zum Schluss den Inhalt des Positionierungsfelds (PTNAME) zu bereinigen, bevor die Subfile erneut angezeigt wird.

```
* Schleife für die Verarbeitung der Subdatei, bis F3 oder F12 gedrückt wird
*
C                     dou         (cfkey = exit) or (cfkey = cancel)
*
C                     write       fkey1
C                     exfmt       sf1ctl
*
* Verarbeitung der eingegebenen Positionierungsinformationen, Bereinigen
* und Neuerstellung der Subdatei. Zuletzt Löschen des Positionierungsfelds
*
C                     select
C                     when        (cfkey = enter) and (ptname   *blanks)
C          ptname     setll       sfl001lf
C                     exsr        clrsfl
C                     exsr        sflbld
C                     clear       ptname
*
* Hinzufügen weiterer Datensätze, wenn sich der Benutzer ganz unten befindet
*
C                     when        (cfkey = rollup) and (not *in32)
C                     exsr        sflbld

C                     endsl
C                     endde
```

Abbildung 2.11: Der Hauptcode unter Verwendung der neuen Techniken

Achten Sie darauf, wie ich das Attention-Indikator-Byte verwende, um die Eingabetaste, die Taste also, die normalerweise am häufigsten gedrückt wird, explizit verarbeiten zu können. Beachten Sie auch, um wie viel einfacher der Code lesbar ist, wenn statt *IN27 das Schlüsselwort ROLLUP für die Taste Seite nach unten (Bild Ab), Enter für die Eingabetaste, Beenden für die Taste F3 etc. verwendet wird. Auch hier ist die Verwendung des Attention-Indikator-Bytes in der Dateiinformationsstruktur und ein Positionierungsfeld in einem Subfile-Programm nicht erforderlich.

Ich stelle Ihnen diese Techniken vor, da ich sie in den folgenden Kapiteln einsetzen werde (ebenso wie in allen meinen Subfile-Anwendungen) und weil ich glaube, dass diese Informationen für Sie beim Lernen der Programmierung von Anzeigeprogrammen wertvoll sein können. Wenden wir uns nun wieder den Subfiles zu.

Subfile für die seitenweise Anzeige

Die Subfile für die seitenweise Anzeige ist der dritte Typ Subfile. Sie erfordert zwar etwas mehr Programmierung, bietet aber dem Benutzer größere Flexibilität. Die Theorie, die sich hinter der Subfile für seitenweise Anzeige verbirgt, lautet, dass nie mehr als jeweils eine Seite Daten in Ihrer Subfile vorhanden sind. Der größte Vorteil dieses Typs Subfile besteht darin, dass Sie nicht auf die 9.999-Datensatzgrenze der Load-All und selbsterweiternden Subfiles beschränkt sind.

Da die Subfile beim Drücken jeder Seitentaste (Bild Auf oder Bild Ab) bereinigt und neu geladen wird, können Sie durch eine unbegrenzte Anzahl von Datensätzen blättern. Ich habe einen zweiten Vorteil geschaffen, indem ich ein Positionierungsfeld hinzugefügt habe. Da Sie die Subfile mit jeder gedrückten Seitentaste neu laden, können Sie nun vom Anfang der Subfile-Liste nach oben blättern (solange das nicht der Anfang Ihrer Datendatei ist).

Als Sie bei der selbsterweiternden Subfile das Positionierungsfeld verwendeten, bestimmten die Daten, die in das Positionierungsfeld eingegeben wurden, die Position des Anfangs Ihrer Liste. Die einzige Methode, um die Daten vor diesem Listenanfang abzurufen, bestand in der erneuten Verwendung des Positionierungsfelds. Bei der Subfile für die seitenweise Verarbeitung können Sie, wenn Sie das Positionierungsfeld verwenden, vom Anfang der Liste nach oben blättern, um die früheren Daten zu erhalten. Dies ist möglich, weil Ihr Programm sowohl das Blättern nach unten als auch das Blättern nach oben übernimmt. Dieser Typ Subfile bietet dem Benutzer die größte Flexibilität, erfordert aber auch vergleichsweise die umfangreichsten Programmierarbeiten und kann der leistungsintensivste Typ Subfile sein.

Wenn wir uns die DDS für unsere Subfile für die seitenweise Verarbeitung (SFL003DF) ansehen, sehen wir, dass lediglich zwei Änderungen an der selbsterweiternden Subfile vorgenommen wurden. Die erste Änderung besteht darin, dass für SFLPAG und SFLSIZ gleiche Werte angegeben werden, was bedeutet, dass Ihre Subfile niemals mehr als eine Seite von Daten enthalten wird.

Wenn Sie versuchen, die von SFLSIZ spezifizierten Werte zu überschreiten, erhalten Sie einen Fehler - tun Sie das also nicht. Als zweite Änderung wurde das Schlüsselwort ROLLDOWN hinzugefügt. Damit wird OS/400 mitgeteilt, dass jetzt Ihr RPG-Programm die gesamte Seitenverarbeitung übernimmt.

Abbildung 2.12 zeigt den Subfile-Steuerdatensatz. Der Rest der DDS bleibt gleich.

```
A                    R SF1CTL                    SFLCTL(SFL1)
A*
A                                                SFLSIZ(0015)
A                                                SFLPAG(0015)
A                                                OVERLAY
A                                                ROLLUP
A                                                ROLLDOWN
A N32                                            SFLDSP
A N31                                            SFLDSPCTL
A             31                                 SFLCLR
A             90                                 SFLEND(*MORE)
A                    RRN1            4S 0H       SFLRCDNBR
A                                              6  2'Nachname'
A                                                 DSPATR(HI)
A                                              6 26'Vorname'
A                                                 DSPATR(HI)
A                                              6 50'MI'
A                                                 DSPATR(HI)
A                                              6 55'Rufname'
A                                                 DSPATR(HI)
A                                              1  2'SFL002RG'
A                                              1 71DATE
A                                                 EDTCDE(Y)
A                                              2 71TIME
A                                              1 24'Subdatei-Programm mit Positionierung'
A                                                 DSPATR(HI)
A                                              4  2'Positionierung auf Nachname . . .'
A                    PTNAME          20    B 4 30
```

Abbildung 2.12: Subfile-Steuerdatensatz für die Subfile mit seitenweiser Verarbeitung (SFL003DF).

```
Dinfo                     ds
D cfkey                        369    369

Dsvlnam                   S              like(dblnam)
Dsvfnam                   S              like(dbfnam)

Dexit                     C              const(X'33')
Dcancel                   C              const(X'3C')
Denter                    C              const(X'F1')
Drollup                   C              const(X'F5')
Drolldn                   C              const(X'F4')
Dsflpag                   C              const(15)
Dsflpag_plus_1            C              const(16)
```

Abbildung 2.13: D-Spezifikationen für Subfiles mit seitenweiser Verarbeitung (SFL003RG).

Das RPG-Programm ist nun zwar etwas komplexer (siehe Abbildung 2.12 bis 2.14), aber es ist nicht übermäßig kompliziert. Wir haben die D-Spezifikationen mit einer Reihe von Standalone-Feldern ergänzt, SVLNAM und SVFNAM. Ich werde dieser Felder verwenden, um den Vornamen und den Nachnamen des ersten Subfile-Datensatzes zu speichern. Da diese Felder den Schlüssel zur von mir verwendeten Datenbank bilden, verwende ich sie, wenn der Benutzer durch die Daten blättern möchte (siehe Abbildung 2.13).

Der Hauptcode wurde ebenfalls geändert. Sehen wir uns einmal die WHEN-Klausel einen Moment lang an. Wenn der Benutzer die Taste Bild Auf und die Subfile angezeigt wurde (Indikator 32 ist deaktiviert), wird die Subroutine GOBACK ausgeführt. Diese Subroutine wird für nichts anderes verwendet als für die richtige Einstellung des Zeigers in der für das Laden der Subfile verwendeten Datendatei. Ich verwende den ersten Datensatz in meiner Subfile, um in meiner Datendatei SFL001LF Untergrenzen (Lower Limits - SETLL) zu setzen. Dabei führe ich eine DO-Schleife aus, die die früheren Datensätze in SFL001LF lesen, bis der

Wert des SFLPAG-Parrameters um eins überschritten ist. Wenn ich vor Ende meiner Schleife auf das Ende der Datendatei treffe, setze ich mit *LOVAL Untergrenzen, um den Zeiger auf den Anfang der Datei zu positionieren und die Schleife verlassen zu können. Sobald die Schleife beendet wurde, befindet sich mein Datendateizeiger entweder am Anfang der Datei oder an einem Punkt der Datei, der davon abhängt, welche Informationen von SAVKEY und SFLPAG_PLUS_1 vorgegeben werden.

Nach Ausführung der Subroutine GOBACK bereinige ich die Subfile und lade eine neue Seite basierend auf dem Punkt, an dem ich zuvor die GOBACK-Subroutine verlassen habe. Der Benutzer kann nun von einem beliebigen Punkt in der Subfile nach oben oder nach unten blättern, während die einzigen Grenzen der Anfang und das Ende der Datendatei sind. Damit wird die 9.999-Datensatzgrenze beseitigt, was in vielen Fällen hilfreich sein kann.

Wenn während der Routine SFLBLD die relative Datensatznummer (RRN1) der Subfile 1 entspricht, verschiebe ich DBLNAM zu SVLNAM und DBFNAM zu SVFNAM. Damit werden diese Felder für die spätere Verwendung vorgesehen. Sie werden auch bemerken, dass ich eine Konstante ROLLDN hinzugefügt habe. Mein Programm übernimmt so das Blättern nach oben und nach unten in der Subfile. Die letzte verwendete ebenfalls neue D-Spezifikation wird beim Bild-Auf-Prozess eingesetzt werden. Die Hauptroutine dieses Programms erinnert an bereits bekannte Routinen mit Ausnahme der Bild-Auf-Verarbeitung, die durch das Hinzufügen des Schlüsselworts ROLLDN in der DDS erreicht wird. Abbildung 2.14 zeigt die Hauptroutine.

```
*
*****************************************************************
*        Hauptroutine
*****************************************************************
*
* Bereinigen, anschließend initiale Subdatei erstellen
*
C                       exsr      clrsfl
C                       exsr      sflbld
*
* Schleife für die Verarbeitung der Subdatei, bis F3 oder F12 gedrückt wird
*
C                       dou       (cfkey = exit) or (cfkey = cancel)
*
C                       write     fkey1
C                       exfmt     sf1ctl
*
* Verarbeitung der eingegebenen Positionierungsinformationen, Bereinigen
* und Neuerstellung der Subdatei. Zuletzt Bereinigen des Positionierungsfelds.
*
C                       select
C                       when      (cfkey = enter) and (ptname   *blanks)
C         ptname        setll     sfl001lf
C                       exsr      clrsfl
C                       exsr      sflbld
C                       clear     ptname
*
* Bereinigen und Neuerstellung der Subdatei, wenn Benutzer Bild Ab/Seite nach
unten verwendet
*
C                       when      (cfkey = rollup) and (not *in90)
C                       exsr      clrsfl
C                       exsr      sflbld
*
* Positionierung Datendatei auf eine Seite vor aktuellen Subdateidaten, dann
* Bereinigen und Neuerstellung der Subdatei, um diese letzte Seite anzuzeigen
*
C                       when      (cfkey = rolldn) and (not *in32)
C                       exsr      goback
C                       exsr      clrsfl
C                       exsr      sflbld

C                       endsl
C                       enddo

C                       eval      *inlr = *on
```

Abbildung 2.14: Hauptroutine für die seitenweise Subfile (SFL003RG).

Der Code für die Erstellungs- und Bereinigungsroutinen ist der gleiche Code wie bei den zuvor besprochenen Programmen, aber wir haben eine neue Routine für das Rückwärtsblättern hinzugefügt. Dieser Code ist recht unkompliziert. Zuerst konfiguriere ich SETLL für die Datendatei SFL001LF, wobei ich die Vor- und Nachnamen des ersten Subfile-Datensatzes verwende (erinnern Sie sich an meine Verwendung von SVLNAM und SVFNAM?). Anschließend lese ich mit der Operation READP rückwärts in der Datei, solange, bis der Wert der Subfile-Seitengröße +1 erreicht wurde, oder bis ich zum Anfang der Datei gelange. Zu diesem Punkt befindet sich mein Datendateizeiger entweder ganz oben in der Datei oder im Datensatz, der sich am Anfang der vorherigen Subfile-Seite befand. Anschließend rufe ich die Routine SFLBLD auf, um die vorherige Seite zu erstellen und anzuzeigen. Abbildung 2.15 zeigt den Code.

```
*
*****************************************************************
*         GOBACK - eine Seite zurück
*****************************************************************
*
C         goback       begsr
*
* Positionierung Datendatei auf ersten Datensatz der Subdatei
*
C         savkey       setll       sfl001lf
*
* Neupositionieren des Zeigers in der Datei für Rückwärtsblättern. Wenn der Anfang
* der Datei erreicht wird vor Ende, Zeigerposition. auf ersten Datensatz der
* Datendatei
C                      do          sflpag_plus_1
C                      readp       sfl001lf
C                      if          %eof
C         *loval       setll       sfl001lf
C                      leave
C                      endif
C                      enddo
*
C                      endsr
```

Abbildung 2.15: Rückwärtsblättern in einer Subfile für das seitenweise Blättern (SFL003RG).

Alternativen für das Blättern

An diesem Punkt wissen Sie nun, wie Sie mit Hilfe der Seitenverarbeitungstasten Bild Ab (Seite nach unten) und Bild Auf (Seite nach oben) durch die Subfile blättern können. Sie haben eventuell auch bemerkt, dass Sie bei jedem Drücken der Seitentaste eine komplette neue Seite mit Daten erhalten. Das muss nicht unbedingt der Fall sein. Sie besitzen etwas Flexibilität bei der Entscheidung, wie durch die Daten geblättert wird und welche Tasten verwendet werden sollen. Zwei Subfile-Schlüsselwörter, SFLENTER und SFLROLVAL, bieten Ihnen diese Flexibilität.