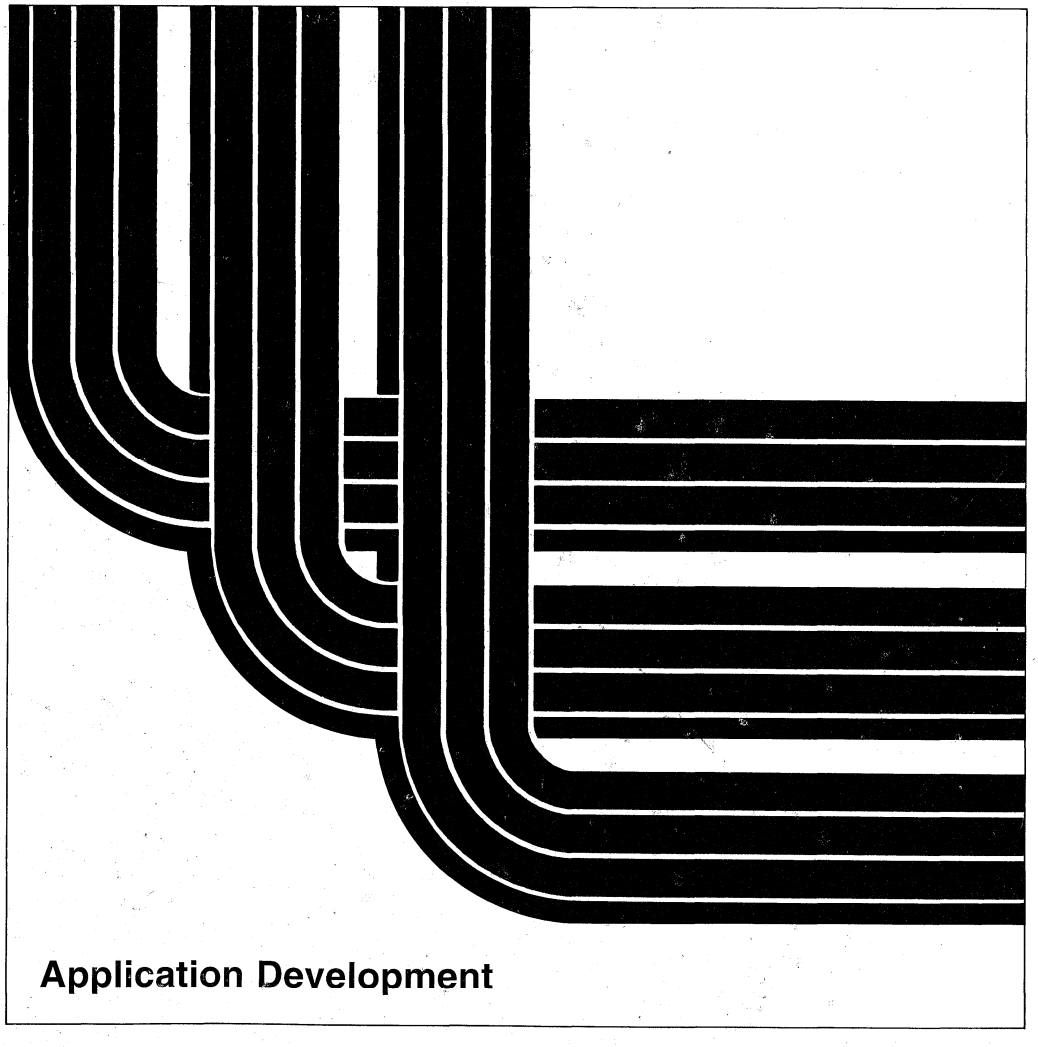
Application System/400

SC41-9620-02

Data Description Specifications Reference

Version 2



Application Development

Application System/400

SC41-9620-02

Data Description Specifications Reference

Version 2

Take Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page xiii.

| **Third Edition (November 1993)**

| This edition applies to the licensed program IBM Operating System/400, (Program 5738-SS1), Version 2 Release 3 Modification 0,
| and to all subsequent releases and modifications until otherwise indicated in new editions. This major revision makes obsolete
| SC41-9620-01. Make sure you are using the proper edition for the level of the product.

Order publications through your IBM representative or the IBM branch serving your locality. Publications are not stocked at the address given below.

A Customer Satisfaction Feedback form for readers' comments is provided at the back of this publication. If the form has been removed, you can mail your comments to:

Attn Department 245
IBM Corporation
3605 Highway 52 N
Rochester, MN 55901-7899 USA

or you can fax your comments to:

United States and Canada: 800+937-3430
Other countries: (+1)+507+253-5192

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you or restricting your use of it.

© **Copyright International Business Machines Corporation 1991, 1993. All rights reserved.**
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Notices xiii
Programming Interface Information xiii
Trademarks and Service Marks xiv

About This Manual xv
Who Should Use This Manual xv

Summary of Changes xvii

Chapter 1. Introduction 1-1
Creating a File Using Data Description Specifications 1-1
Completing the Form 1-1
Entering the Source Statements 1-3
Creating the File 1-3
Syntax Rules 1-4
Keywords and Parameter Values 1-4
DDS Naming Conventions 1-6
Syntax Coding Examples 1-8

Chapter 2. Physical and Logical Files 2-1
Defining a Physical File 2-1
Defining a Logical File 2-2
Simple and Multiple Format Logical Files 2-2
Join Logical Files 2-3
Specifying Record Formats in a Logical File 2-3
Positional Entries (Positions 1 through 44) 2-4
Sequence Number (Positions 1 through 5) 2-4
Form Type (Position 6) 2-5
Comment (Position 7) 2-5
Conditioning (Positions 8 through 16) 2-5
Type of Name or Specification (Position 17) 2-5
Reserved (Position 18) 2-5
Name (Positions 19 through 28) 2-6
Reference (Position 29) 2-21
Length (Positions 30 through 34) 2-22
Data Type (Position 35) 2-25
Decimal Positions (Positions 36 and 37) 2-27
Usage (Position 38) 2-29
Location (Positions 39 through 44) 2-30
Keyword Entries (Positions 45 through 80) 2-30
ABSVAL (Absolute Value) 2-31
ALIAS (Alternative Name) 2-32
ALL (All)—Logical Files Only 2-33
ALTSEQ (Alternative Collating Sequence) 2-33
ALWNULL (Allow Null Value)—Physical Files Only 2-34
CCSID (Coded Character Set Identifier)—Physical Files Only 2-35
CHECK (Check) 2-35
CHKMSGID (Check Message Identifier) 2-36
CMP (Comparison) 2-37
COLHDG (Column Heading) 2-37

© Copyright IBM Corp. 1991, 1993

COMP (Comparison) . . . 2-38
CONCAT (Concatenate)—Logical Files Only . . . 2-40
DATFMT (Date Format) . . . 2-43
DATSEP (Date Separator) . . . 2-44
DESCEND (Descend) . . . 2-45
DFT (Default)—Physical Files Only . . . 2-45
DIGIT (Digit) . . . 2-47
DYNSLT (Dynamic Select)—Logical Files Only . . . 2-48
EDTCDE (Edit Code) and EDTWRD (Edit Word) . . . 2-51
FCFO (First-Changed First-Out) . . . 2-52
FIFO (First-In First-Out) . . . 2-52
FLTPCN (Floating-Point Precision) . . . 2-53
FORMAT (Format) . . . 2-53
JDFTVAL (Join Default Values)—Join Logical Files Only . . . 2-54
JDUPSEQ (Join Duplicate Sequence)—Join Logical Files Only . . . 2-55
JFILE (Joined Files)—Join Logical Files Only . . . 2-57
JFLD (Joined Fields)—Join Logical Files Only . . . 2-58
JOIN (Join)—Join Logical Files Only . . . 2-60
JREF (Join Reference)—Join Logical Files Only . . . 2-62
LIFO (Last-In First-Out) . . . 2-64
NOALTSEQ (No Alternative Collating Sequence) . . . 2-64
PFILE (Physical File)—Logical Files Only . . . 2-65
RANGE (Range) . . . 2-66
REF (Reference)—Physical Files Only . . . 2-68
REFACCPTH (Reference Access Path Definition)—Logical Files Only . . . 2-69
REFFLD (Referenced Field)—Physical Files Only . . . 2-70
REFSHIFT (Reference Shift) . . . 2-71
RENAME (Rename)—Logical Files Only . . . 2-72
SIGNED (Signed) . . . 2-73
SST (Substring)—Logical Files Only . . . 2-74
TEXT (Text) . . . 2-76
TIMFMT (Time Format) . . . 2-76
TIMSEP (Time Separator) . . . 2-77
TRNTBL (Translation Table)—Logical Files Only . . . 2-78
UNIQUE (Unique) . . . 2-79
UNSIGNED (Unsigned) . . . 2-81
VALUES (Values) . . . 2-81
VARLEN (Variable-Length Field) . . . 2-83
ZONE (Zone) . . . 2-84

Chapter 3. Display Files . . . 3-1
Defining a Display File . . . 3-1
Positional Entries (Positions 1 through 44) . . . 3-2
Sequence Number (Positions 1 through 5) . . . 3-2
Form Type (Position 6) . . . 3-2
Comment (Position 7) . . . 3-2
Conditioning (Positions 7 through 16) . . . 3-2
Conditioning a Field for More than One Keyword . . . 3-3
Type of Name or Specification (Position 17) . . . 3-6
Reserved (Position 18) . . . 3-7
Name (Positions 19 through 28) . . . 3-7
Reference (Position 29) . . . 3-8
Length (Positions 30 through 34) . . . 3-9
Data Type/Keyboard Shift (Position 35) . . . 3-10

Decimal Positions (Positions 36 and 37) . . . 3-21
Usage (Position 38) . . . 3-22
Location (Positions 39 through 44) . . . 3-24
Keyword Entries (Positions 45 through 80) . . . 3-26
ALARM (Audible Alarm) . . . 3-28
ALIAS (Alternative Name) . . . 3-28
ALTHELP (Alternative Help Key) . . . 3-29
ALTNAME (Alternative Record Name) . . . 3-30
ALTPAGEDWN/ALTPAGEUP (Alternative Page Down/Alternative Page Up) . . . 3-30
ALWGPH (Allow Graphics) . . . 3-32
ALWROL (Allow Roll) . . . 3-33
ASSUME (Assume) . . . 3-35
AUTO (Auto) . . . 3-36
BLANKS (Blanks) . . . 3-36
BLINK (Blink) . . . 3-38
BLKFOLD (Blank Fold) . . . 3-39
CAnn (Command Attention) . . . 3-39
CFnn (Command Function) . . . 3-41
CHANGE (Change) . . . 3-43
CHCACCEL (Choice Accelerator Text) . . . 3-44
CHCAVAIL (Choice Color/Display Attribute when Available) . . . 3-45
CHCCTL (Choice Control) . . . 3-47
CHCSLT (Choice Color/Display Attribute when Selected) . . . 3-49
CHCUNAVAIL (Choice Color/Display Attribute when Unavailable) . . . 3-50
CHECK (Check) . . . 3-52
CHGINPDFT (Change Input Default) . . . 3-62
CHKMSGID (Check Message Identifier) . . . 3-64
CHOICE (Selection Field Choice) . . . 3-65
CHRID (Character Identifier) . . . 3-67
CLEAR (Clear) . . . 3-67
CLRL (Clear Line) . . . 3-68
CMP (Comparison) . . . 3-70
CNTFLD (Continued-Entry Field) . . . 3-71
COLOR (Color) . . . 3-72
COMP (Comparison) . . . 3-76
CSRLOC (Cursor Location) . . . 3-77
DATE (Date) . . . 3-78
DFT (Default) . . . 3-78
DFTVAL (Default Value) . . . 3-80
DLTCHK (Delete Check) . . . 3-81
DLTEDT (Delete Edit) . . . 3-81
DSPATR (Display Attribute) . . . 3-82
DSPMOD (Display Mode) . . . 3-90
DSPRL (Display Right to Left) . . . 3-91
DSPSIZ (Display Size) . . . 3-92
DUP (Duplication) . . . 3-97
EDTCDE (Edit Code) . . . 3-99
EDTMSK (Edit Mask) . . . 3-104
EDTWRD (Edit Word) . . . 3-105
ENTFLDATR (Entry Field Attribute) . . . 3-108
ERASE (Erase) . . . 3-110
ERASEINP (Erase Input) . . . 3-111
ERRMSG (Error Message) and ERRMSGID (Error Message Identifier) . . . 3-112

ERRSFL (Error Subfile) 3-115
FLDCSRPRG (Cursor Progression Field) 3-116
FLTFIXDEC (Floating-Point to Fixed Decimal) 3-117
FLTPCN (Floating-Point Precision) 3-118
FRCDTA (Force Data) 3-118
GETRETAIN (Get Retain) 3-120
HELP (Help) 3-120
HLPARA (Help Area) 3-121
HLPBDY (Help Boundary) 3-125
HLPCLR (Help Cleared) 3-126
HLPCMDKEY (Help Command Key) 3-127
HLPDOC (Help Document) 3-129
HLPEXCLD (Help Excluded) 3-130
HLPFULL (Help Full) 3-131
HLPID (Help Identifier) 3-132
HLPPNLGRP (Help Panel Group) 3-132
HLPRCD (Help Record) 3-133
HLPRTN (Help Return) 3-134
HLPSCHIDX (Help Search Index) 3-135
HLPSEQ (Help Sequencing) 3-136
HLPTITLE (Help Title) 3-137
HOME (Home) 3-138
INDARA (Indicator Area) 3-139
INDTXT (Indicator Text) 3-139
INVITE (Invite) 3-140
INZINP (Initialize Input) 3-142
INZRCD (Initialize Record) 3-145
KEEP (Keep) 3-146
LOCK (Lock) 3-147
LOGINP (Log Input) 3-147
LOGOUT (Log Output) 3-148
LOWER (Lower) 3-148
MDTOFF (Modified Data Tag Off) 3-148
MLTCHCFLD (Multiple-Choice Selection Field) 3-149
MNUBAR (Menu Bar) 3-151
MNUBARCHC (Menu-Bar Choice) 3-152
MNUBARDSP (Menu-Bar Display) 3-156
MNUBARSEP (Menu-Bar Separator) 3-158
MNUBARSW (Menu-Bar Switch Key) 3-160
MNUCNL (Menu-Cancel Key) 3-161
MSGALARM (Message Alarm) 3-162
MSGCON (Message Constant) 3-163
MSGID (Message Identifier) 3-164
MSGLOC (Message Location) 3-166
NOCCSID (No Coded Character Set Identifier) 3-168
OPENPRT (Open Printer File) 3-168
OVERLAY (Overlay) 3-169
OVRATR (Override Attribute) 3-170
OVRDTA (Override Data) 3-171
PAGEDOWN/PAGEUP (Page down/Page up) 3-172
PASSRCD (Passed Record) 3-173
PRINT (Print) 3-173
PROTECT (Protect) 3-176
PULLDOWN (Pull-Down Menu) 3-177

PUTOVR (Put with Explicit Override) 3-178
PUTRETAIN (Put-Retain) 3-181
RANGE (Range) 3-182
REF (Reference) 3-183
REFFLD (Referenced Field) 3-184
RETKEY (Retain Function Keys) and RETCMDKEY (Retain Command Keys) 3-186
RETLCKSTS (Retain Lock Status) 3-186
RMVWDW (Remove Window) 3-186
ROLLUP/ROLLDOWN (Roll up/Roll down) 3-187
RTNCSRLOC (Return Cursor Location) 3-188
RTNDTA (Return Data) 3-189
SETOF (Set Off) 3-190
SETOFF (Set Off) 3-191
SFL (Subfile) 3-191
SFLCLR (Subfile Clear) 3-193
SFLCSRPRG (Subfile Cursor Progression) 3-194
SFLCSRRRN (Subfile Cursor Relative Record Number) 3-194
SFLCTL (Subfile Control) 3-195
SFLDLT (Subfile Delete) 3-196
SFLDROP (Subfile Drop) 3-197
SFLDSP (Subfile Display) 3-199
SFLDSPCTL (Subfile Display Control) 3-199
SFLEND (Subfile End) 3-200
SFLENTER (Subfile Enter) 3-202
SFLFOLD (Subfile Fold) 3-203
SFLINZ (Subfile Initialize) 3-205
SFLLIN (Subfile Line) 3-206
SFLMODE (Subfile Mode) 3-207
SFLMSG (Subfile Message) and SFLMSGID (Subfile Message Identifier) 3-209
SFLMSGKEY (Subfile Message Key) 3-211
SFLMSGRCD (Subfile Message Record) 3-212
SFLNXTCHG (Subfile Next Changed) 3-213
SFLPAG (Subfile Page) 3-214
SFLPGMQ (Subfile Program Message Queue) 3-216
SFLRCDNBR (Subfile Record Number) 3-218
SFLRNA (Subfile Records Not Active) 3-219
SFLROLVAL (Subfile Roll Value) 3-220
SFLSIZ (Subfile Size) 3-223
SLNO (Starting Line Number) 3-225
SNGCHCFLD (Single-Choice Selection Field) 3-226
SYSNAME (System Name) 3-228
TEXT (Text) 3-228
TIME (Time) 3-229
UNLOCK (Unlock) 3-229
USER (User) 3-231
USRDFN (User-Defined) 3-231
USRDSPMGT (User Display Management) 3-232
USRRSTDSP (User Restore Display) 3-232
VALUES (Values) 3-233
VLDCMDKEY (Valid Command Key) 3-234
WDWBORDER (Window Border) 3-235
WINDOW (Window) 3-238

Chapter 4. Printer Files 4-1
Defining a Printer File 4-1
Positional Entries (Positions 1 through 44) 4-2
Sequence Number (Positions 1 through 5) 4-2
Form Type (Position 6) 4-2
Comment (Position 7) 4-2
Conditioning (Positions 7 through 16) 4-3
Type of Name or Specification (Position 17) 4-4
Reserved (Position 18) 4-4
Name (Positions 19 through 28) 4-4
Reference (Position 29) 4-5
Length (Positions 30 through 34) 4-6
Data Type (Position 35) 4-7
Decimal Positions (Positions 36 and 37) 4-7
Usage (Position 38) 4-8
Location (Positions 39 through 44) 4-8
Keyword Entries (Positions 45 through 80) 4-11
ALIAS (Alternative Name) 4-11
BARCODE (Bar Code) 4-12
BLKFOLD (Blank Fold) 4-15
BOX (Box) 4-16
CDEFNT (Coded Font Name) 4-18
CHRID (Character Identifier) 4-19
CHRSIZ (Character Size) 4-20
COLOR (Color) 4-21
CPI (Characters Per Inch) 4-22
CVTDTA (Convert Data) 4-26
DATE (Date) 4-29
DFNCHR (Define Character) 4-30
DFT (Default) 4-42
DLTEDT (Delete Edit) 4-43
DRAWER (Drawer) 4-44
EDTCDE (Edit Code) 4-45
EDTWRD (Edit Word) 4-50
ENDPAGE (End Page) 4-53
FLTFIXDEC (Floating-Point to Fixed Decimal) 4-54
FLTPCN (Floating-Point Precision) 4-55
FNTCHRSET (Font Character Set) 4-55
FONT (Font) 4-56
GDF (Graphic Data File) 4-59
HIGHLIGHT (Highlight) 4-61
INDARA (Indicator Area) 4-62
INDTXT (Indicator Text) 4-62
LINE (Line) 4-63
LPI (Lines Per Inch) 4-65
MSGCON (Message Constant) 4-67
OVERLAY (Overlay) 4-68
PAGNBR (Page Number) 4-70
PAGRTT (Page Rotation) 4-71
PAGSEG (Page Segment) 4-72
POSITION (Position) 4-74
PRTQLTY (Print Quality) 4-76
REF (Reference) 4-77
REFFLD (Referenced Field) 4-78

SKIPA (Skip After) 4-79
SKIPB (Skip Before) 4-80
SPACEA (Space After) 4-81
SPACEB (Space Before) 4-81
TEXT (Text) 4-82
TIME (Time) 4-82
TRNSPY (Transparency) 4-83
TXTRTT (Text Rotation) 4-84
UNDERLINE (Underline) 4-85

Chapter 5. Intersystem Communications Function Files 5-1
Defining an ICF File 5-1
Positional Entries (Positions 1 through 44) 5-2
Sequence Number (Positions 1 through 5) 5-2
Form Type (Position 6) 5-2
Comment (Position 7) 5-2
Conditioning (Positions 7 through 16) 5-2
Type of Name or Specification (Position 17) 5-3
Reserved (Position 18) 5-4
Name (Positions 19 through 28) 5-4
Reference (Position 29) 5-4
Length (Positions 30 through 34) 5-5
Data Type (Position 35) 5-6
Decimal Positions (Positions 36 and 37) 5-6
Usage (Position 38) 5-6
Location (Positions 39 through 44) 5-7
Keyword Entries (Positions 45 through 80) 5-7
ALIAS (Alternative Name) 5-7
ALWWRT (Allow Write) 5-8
CANCEL (Cancel) 5-8
CNLINVITE (Cancel Invite) 5-9
CONFIRM (Confirm) 5-10
CTLDTA (Control Data) 5-10
DETACH (Detach) 5-11
DFREVOKE (Defer Evoke) 5-12
ENDGRP (End of Group) 5-12
EOS (End of Session) 5-13
EVOKE (Evoke) 5-13
FAIL (Fail) 5-16
FLTPCN (Floating-Point Precision) 5-17
FMH (Function Management Header) 5-17
FMTNAME (Format Name) 5-18
FRCDTA (Force Data) 5-18
INDARA (Indicator Area) 5-19
INDTXT (Indicator Text) 5-20
INVITE (Invite) 5-21
NEGRSP (Negative Response) 5-21
RCVCANCEL (Receive Cancel) 5-22
RCVCONFIRM (Receive Confirm) 5-23
RCVCTLDTA (Receive Control Data) 5-23
RCVDETACH (Receive Detach) 5-24
RCVENDGRP (Receive End of Group) 5-25
RCVFAIL (Receive Fail) 5-25
RCVFMH (Receive Function Management Header) 5-26

RCVNEGRSP (Receive Negative Response) 5-26
RCVTRNRND (Receive Turnaround) 5-27
RECID (Record Identification) 5-27
REF (Reference) 5-31
REFFLD (Referenced Field) 5-32
RQSWRT (Request Write) 5-33
RSPCONFIRM (Respond Confirm) 5-34
SECURITY (Security) 5-35
SUBDEV (Subdevice) 5-36
SYNLVL (Synchronization Level) 5-38
TEXT (Text) 5-38
TIMER (Timer) 5-39
VARBUFMGT (Variable Buffer Management) 5-40
VARLEN (Variable-Length User Data) 5-40

Appendix A. When to Specify REF and REFFLD A-1

Appendix B. Examples B-1
Database Files B-1
Field Reference File B-1
Physical File with a New Record Format B-3
Logical File Specifying Multiple Formats and New Keys B-4
Logical File Specifying a New Record Format B-5
Join Logical File B-6
Device Files B-7
Inquiry Display with Two Record Formats B-7
Subfile with SFLPAG Value Equal to SFLSIZ Value B-11
Subfile with Paging by OS/400 Program and High-Level Language Program B-13
Horizontal Subfile Displayable on Two Display Sizes B-14
Message Subfile B-16
Printer File B-18
ICF File B-19
Example Program Using a Physical, Display, and Printer File B-21
Compiler Listing Example B-24
Debugging Template B-27

Appendix C. DDS Keyword and Value Abbreviations C-1

Appendix D. DDS for 3270 Remote Attachment D-1

Appendix E. Double-Byte Character Set Considerations E-1
General Considerations E-1
Positional Entries E-1
Keyword Entries (Positions 45 through 80) E-2
DBCS Character Strings E-3
DDS Computer Printouts E-4
Database Files E-4
Positional Entry Considerations E-4
Keyword Considerations E-5
Additional Considerations for Describing Database Files E-7
Display Files E-8
Positional Entry Considerations E-8
Keyword Considerations E-9

IGCALTTYP (Alternative Data Type) E-10
IGCCNV (DBCS Conversion) E-11
Additional Considerations for Describing Display Files E-12
Printer Files E-13
Positional Entry Considerations E-13
Keyword Considerations E-14
DFNLIN (Define Line) E-15
IGCALTTYP (Alternative Data Type) E-17
IGCANKCNV (Alphanumeric-to-DBCS Conversion) E-18
IGCCDEFNT (DBCS Coded Font) E-20
IGCCHRRTT (DBCS Character Rotation) E-20
Additional Considerations for Describing Printer Files E-21
ICF Files E-23
Positional Entry Considerations E-23
Additional Considerations for Describing ICF Files E-23

Appendix F. System/36 Environment Considerations F-1
General Considerations F-1
Keyword Considerations F-1
CHANGE F-2
HELP and HLPRTN F-2
MSGID F-2
PRINT(*PGM) F-4
ALTNAME (Alternative Record Name) F-4
RETKEY (Retain Function Keys) and RETCMDKEY (Retain Command Keys) F-5
USRDSPMGT (User Display Management) F-6

Appendix G. CODE128 Character Set G-1

Glossary H-1

Bibliography I-1

Index X-1

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Purchase, NY 10577, U.S.A.

This publication could contain technical inaccuracies or typographical errors.

This publication may refer to products that are announced but not currently available in your country. This publication may also refer to products that have not been announced in your country. IBM makes no commitment to make available any unannounced products referred to herein. The final decision to announce any product is based on IBM's business and technical judgment.

Changes or additions to the text are indicated by a vertical line (|) to the left of the change or addition.

| Refer to the "Summary of Changes" on page xvii for a summary of changes made to the DDS keywords
| and how they are described in this publication.

This publication contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

This publication contains small programs that are furnished by IBM as simple examples to provide an illustration. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. All programs contained herein are provided to you "AS IS". THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY DISCLAIMED.

Programming Interface Information

The *Data Description Specifications Reference*, SC41-9620, is intended to help you code the data description specifications for files that can be described externally. This manual documents General-Use Programming Interface and Associated Guidance Information provided by the OS/400 program.

General-Use programming interfaces allow you to write programs that obtain the services of the OS/400 program.

© Copyright IBM Corp. 1991, 1993

Trademarks and Service Marks

The following terms, denoted by an asterisk (*) in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

Advanced Function Printing
AFP
Application System/400
AS/400
BookManager
C/400
FORTRAN/400
GDDM
IBM
ILE
Integrated Language Environment
Intelligent Printer Data Stream
IPDS
OfficeVision/400
Operating System/400
OS/400
Personal System/2
PrintManager
PS/2
RPG/400
SAA
Systems Application Architecture
400

The following terms, denoted by a double asterisk (**) in this publication, are trademarks of other companies:

Century Schoolbook — American Type Foundry

About This Manual

This manual contains detailed instructions for coding the data description specifications (DDS) for files that can be described externally. These files are the physical, logical, display, printer, and intersystem communications function, hereafter referred to as ICF, files.

You may need to refer to other IBM manuals for more specific information about a particular topic. The *Publications Guide*, GC41-9678, provides information on all the manuals in the AS/400 library.

For a list of related publications, see the Bibliography.

Who Should Use This Manual

This manual is intended for programmers who use the AS/400 system.

To use this manual effectively, you should know the following:

- How to use source entry utility (SEU) or another data-entry function to enter DDS
- How to enter the create file commands on the AS/400 system
- How to use a high-level language (for example, the CL, RPG/400, or COBOL/400 programming languages) to develop an application program

This manual uses the following conventions:

- A *keyword* is a name that identifies a function.
- A *parameter* is an argument shown between the parentheses on a keyword that identifies a value or set of values you can use to tailor the function the keyword specifies.
- A *value* is an actual value that you can use for a parameter.
- In the keyword descriptions, *this field* or *this record format* means the field or record format you are defining.
- The expression *use this file- or record-level keyword* means the keyword is valid only at the file or record level.
- *To specify a keyword* means to code the keyword in the DDS for a file. This contrasts with *to select a keyword* or *when a keyword is in effect*, which both mean that any conditioning (such as one or more option indicators) is satisfied when an application program issues an input or output operation.
- *Current source* or *source you are defining* means the DDS that together make up the description of one file.
- In sample displays, character fields are shown as Xs and numeric fields are shown as Ns.
- The 5250 Work Station Feature is a feature of the OS/2 communications manager that allows the personal computer to perform like a 5250 display station and use functions of the AS/400 system.

© Copyright IBM Corp. 1991, 1993

- *Logical file* includes join logical files, simple logical files, and multiple-format logical files.
- *Page* means to move information up or down on the display. *Roll* means the same as page. *Paging keys* are the same as *roll keys*. The PAGEDOWN keyword is the same as the ROLLUP keyword. The PAGEUP keyword is the same as the ROLLDOWN keyword.

Summary of Changes

New Keywords Added for Display Files

The following keywords were added for display files: CHCACCEL, CHCAVAIL, CHCCTL, CHCSLT, CHCUNAVAIL, CHOICE, CNTFLD, DSPRL, EDTMSK, ENTFLDATR, FLDCSRPRG, HLPID, MLTCHCFLD, MNUBAR, MNUBARCHC, MNUBARDSP, MNUBARSEP, MNUBARSW, MNUCNL, NOCCSID, PULLDOWN, SFLCSRPRG, and SNGCHCFLD. These keywords are described in Chapter 3, "Display Files."

New Parameter on DSPATR Keyword

The following parameter was added to the DSPATR keyword: &program-to-system-field. These parameters are described in "DSPATR (Display Attribute)" on page 3-82.

New Parameter on HLPARA Keyword

The following parameters were added to the HLPARA keyword: *FLD, choice-number, *CNST, and help-identifier. These parameters are described in "HLPARA (Help Area)" on page 3-121.

New Parameter on SFLPGMQ Keyword

The following parameters were added to the SFLPGMQ keyword: 10 or 276. These parameters are described in "SFLPGMQ (Subfile Program Message Queue)" on page 3-216.

New Parameter on WINDOW Keyword

The following parameters were added to the WINDOW keyword: *DFT, *MSGLIN, and *NOMSGLIN. These parameters are described in "WINDOW (Window)" on page 3-238.

Miscellaneous Changes

The glossary has been updated to reflect new and changed terms for this release.

© Copyright IBM Corp. 1991, 1993

Chapter 1. Introduction

A traditional means of describing data attributes (such as the names and lengths of records and fields) is to specify the data attributes in the application programs themselves. However, a convenient and powerful alternative is available on the AS/400* system. Data description specifications (DDS) describe data attributes in file descriptions external to the application program that processes the data. DDS can be used with the following types of files:

- Physical files (DDS is optional)
- Logical files (DDS is required)
- Display files (DDS is optional)
- Printer files (DDS is optional)
- ICF files (DDS is required)

Creating a File Using Data Description Specifications

To create a file using DDS, follow these steps:

1. Complete the data description specifications (DDS) form.
2. Type the source statements into a source file. The source file can be part of the AS/400 database (in a source physical file such as the IBM-supplied QDDSSRC in library QGPL) or it can be on diskettes.
3. Create the file using the appropriate create-file command.

Note: Through the screen design aid (SDA) utility, display files can be created and tested without coding DDS directly, using only the functions that apply to display files. See the *SDA User's Guide and Reference*.

Completing the Form

A sample data description specifications form is printed in reduced size in Figure 1-1 on page 1-2.

The left side of the DDS form (positions 1 through 44) is for fixed-format entries called positional entries. Positional entries define the most common attributes of record formats and fields, such as names and lengths of fields. For a brief description of the most important positional entries, see items **1** through **7** following the figure. For more detailed information, see the positional entries section in Chapter 2, "Physical and Logical Files" through Chapter 5, "Intersystem Communications Function Files."

The right side of the DDS form (positions 45 through 80) is for DDS keywords. DDS keywords define less-common and more-varied attributes of files, record formats, and fields; they follow a subset of the syntax rules for control language. For a brief description of keyword entries, see item **8** following the figure. For more detailed information, see the keyword entries section in Chapter 2, "Physical and Logical Files" through Chapter 5, "Intersystem Communications Function Files."

© Copyright IBM Corp. 1991, 1993

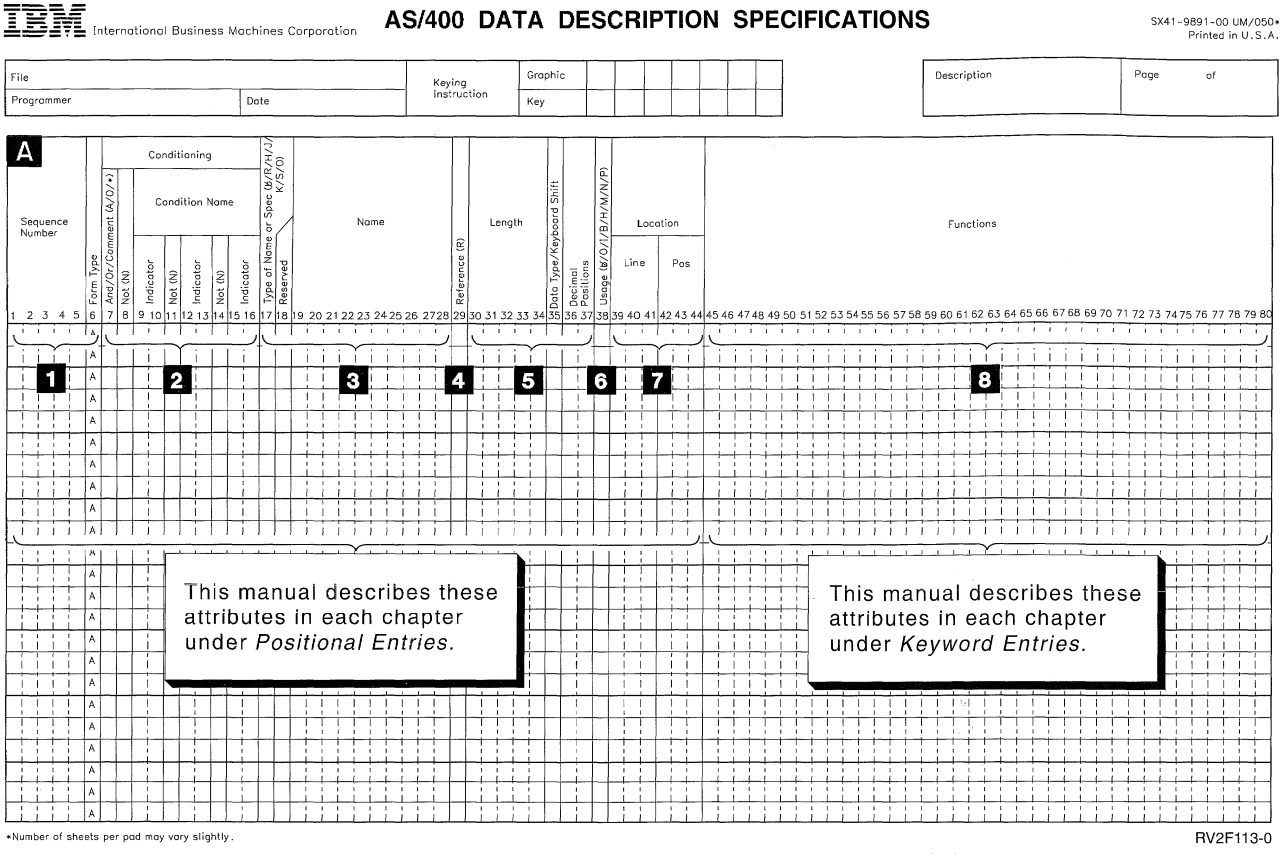


Figure 1-1. Overview of the Positional Entries and Keywords

1 **Sequence Number** and **Form Type** are optional in DDS. The form type identifies the source as DDS source. The entries are valid for all types of files.

2 An asterisk in position 7 makes the entire line a comment. This is valid for all types of files. When A (And), or O (Or), or a blank is in position 7, positions 8 through 16 can provide conditioning for the DDS on or immediately following the current line. Conditioning is not valid in physical or logical files.

3 **Type of Name or Specification** (position 17) identifies the **Name** entry (positions 19 through 28) or the specification:

Name Entry	Description	Type of File
R	Specifies a record format name	All
blank	Specifies a field name	All
K	Specifies a key field name	Physical and logical only
S	Specifies a select field name	Logical only
O	Specifies an omit field name	Logical only
J	Specifies this as a join specification	Join logical only
H	Specifies this as a help specification	Display only

4 R specified in position 29 indicates that the attributes of the field in **Name** (positions 19 through 28) refer to a field specified elsewhere. This is ignored for logical files.

5 **Length, Data Type,** and **Decimal Positions** specify attributes of named fields within record formats. These are valid for all types of files.

6 **Usage** specifies fields as input, output, output/input, neither input nor output, hidden, message, or program-to-system fields. Each type of file has its own restrictions regarding field use.

7 **Location** specifies the location of the field on the display or printed page. This applies for display and printer files only.

8 **Functions** specified through the use of keywords apply at different levels for different file types, as follows:

Keywords Apply at This Level	For These Files
File	All types of files
Record	All types of files
Field	All types of files
Join	Join logical files only
Key field	Physical and logical files
Select or omit field	Logical files only
Help	Display files only

For display and printer files, constants specified within apostrophes become the default values for displayed or printed fields.

Entering the Source Statements

After filling out the forms, enter the source statements into source files. You can enter the source either interactively or in batch.

Entering Source Statements Interactively with SEU. Use the source entry utility (SEU) of the Application Development Tools Licensed Program. See the *SEU User's Guide and Reference* for more information. Type in the Start SEU (STRSEU) command to call SEU.

Entering Source Statements in Batch Using Diskettes. Use one of the following methods:

- Enter an input stream containing DDS source and control language (CL) commands on diskette and start a spooling reader using the Start Diskette Reader (STRDKTRDR) command.
- Enter only source statements on diskette and copy the resulting data file into a source physical file with the Copy File (CPYF) command.
- Enter only source statements on diskette and type a create-file command. Specify the name of the data file on the SRCFILE parameter and *FILE on the SRCMBR parameter of the create-file command.

  Note: This method does not create a source physical file.

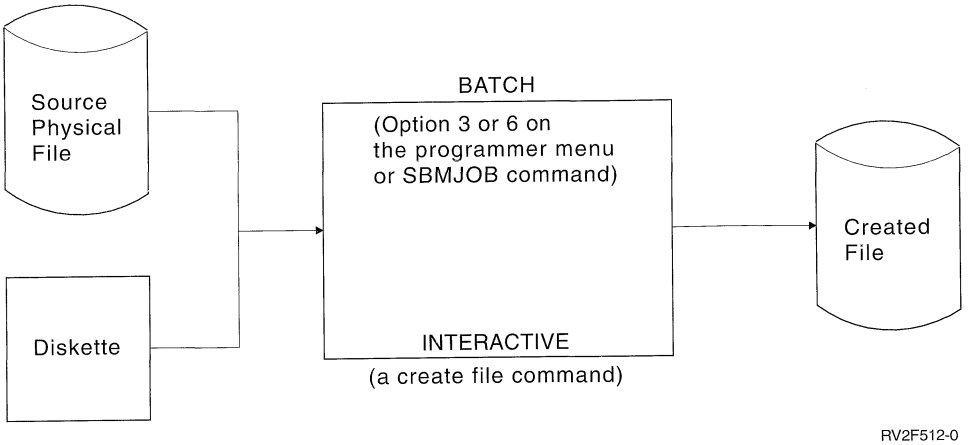
Creating the File

Create the file by issuing a create-file command. The command you use depends on the type of file you are creating. The file types and commands are:

File Type	Command
Physical file	CRTPF
Logical file	CRTLF
Display file	CRTDSPF
Printer file	CRTPRTF
ICF file	CRTICFF

When you issue a create-file command, the DDS is retrieved from the source file and validated, and a file is created as shown in the following figure. The file is created only if there are no errors in the DDS of equal or greater severity than the severity specified on the GENLVL parameter of the create-file command. Thus, you can use the GENLVL parameter to control the allowable error severity when creating the file. Depending on the options you specify on the OPTION and FLAG parameters, a DDS source (or compiler) listing may also be created. The DDS listing contains the data description and error information.

You can use the FLAG parameter to specify the minimum severity of DDS messages which will be printed. For example, you can suppress the warning messages for field overlapping.



See "Compiler Listing Example" on page B-24 for an example of a DDS listing and "Debugging Template" on page B-27 for an example of a debugging template.

Syntax Rules

This section summarizes general information needed to properly code DDS keywords and names.

Keywords and Parameter Values

The DDS coding syntax for keywords and their parameter values is similar to the CL syntax. The DDS syntax rules are:

- Code all DDS entries in uppercase except for character values enclosed in apostrophes and extended names enclosed in quotation marks.
- Code keywords on the same (or subsequent) line as the entry with which they are associated.
- Separate multiple keywords with at least one blank. Parameter values for keywords must be enclosed in parentheses. The initial parenthesis must immediately follow the keyword. For example:

  ```
  KEYWORD(VALUE)
  ```

  This rule is slightly different from that in control language. When coding control language, the parameter values can be positional. Syntax for DDS requires that the keyword be specified, except when specifying either a constant or the parameter value for the DFT (Default) keyword.

- Separate multiple parameter values for the same keyword with at least one blank. For example:

  ```
  KEYWORD(VALUEA VALUEB)
  ```

- Enclose parameter expressions in parentheses. A parameter expression consists of a special value followed by the parameter value. The special value begins with an asterisk and must immediately follow the left parenthesis. The parameter value must immediately precede the right parenthesis. Separate the special value and the parameter value by at least one blank. A parameter expression represents one parameter value and must be separated from other parameter values by at least one blank. For example:

  ```
  KEYWORD(VALUEA (*special-value VALUEB) VALUEC)
  ```

- Use apostrophes to enclose character values. Numeric values appear without apostrophes. See the coding examples for COMP, RANGE, and VALUES keywords. Character values can appear in two places in the syntax:
  - As a parameter value for some keywords. For example, TEXT (all types of files) and COLHDG (database files) require character strings as text description. Other keywords, such as CAnn and CFnn, use character strings as text descriptions for response indicators.
  - As the default value of a constant field (either with or without the DFT keyword) for display and printer files only. In display files, a character constant can also be specified for named fields. Even if you do not specify the DFT keyword, specifying a character constant implies the DFT keyword.

- To specify an apostrophe within a character string, specify two apostrophes so that one apostrophe appears in the output. For example:

  ```
  KEYWORD('Customer''s name')
  ```

  appears as

  ```
  Customer's name
  ```

- Use a plus (+) or a minus (–) sign as a continuation character when a keyword and its parameter values do not fit on a single line. The sign must be the last nonblank character in the functions field. A single statement can be continued for a maximum of 5000 character positions.
  - A **minus (–) sign** means the continuation begins in position 45 of the next line (the first position in the functions field).
  - A **plus (+) sign** means the continuation begins with the first nonblank (first significant) character in the functions field on the next line.

  If you specify a continuation character within a parameter value, any blanks preceding the continuation character are included in the parameter value.

- Specify a plus (+) sign as the last nonblank character on a line to continue conditioning for keywords specified on the next line. This is helpful when a condition includes several option indicators and applies to several keywords.

- The Operating System/400* (OS/400)* program continues a DDS statement until you specify one of the following:
  - A record format name (R in position 17).
  - A field specification (field name or location).
  - For physical or logical files, a key field specification (K in position 17).

- For logical files, a select or omit specification (S or O in position 17).
- For join logical files, a join specification (J in position 17).
- For display files, a help specification (H in position 17).
- For device files, an option indicator or condition name that conditions a keyword, field, or field location.
- The maximum length of a DDS statement (5000 characters). The fixed length entries (positions 1 through 44) of the first line are included in the statement, so the maximum space available for keywords is 4956.

- Keyword descriptions use the following punctuation marks to indicate the syntax for the keyword:

()	Enclosed values are required.
[]	Enclosed values are optional.
[...]	Specify additional values as needed.
{}	The upper value is the default value (see REFFLD).
|	Specify either the value to the left or to the right (may refer to optional values).

DDS Naming Conventions

The naming conventions used in DDS are as follows:

- Qualified names
  - Use a slash to separate the parts of a qualified name. Embedded blanks are not allowed. For example:

    ```
    KEYWORD(library/file)
    ```

  - For most keywords with a qualified name parameter value, you can code *LIBL or *CURLIB for the library name. If you do not specify a library name, *LIBL is used. You cannot code *USRLIBL for the library name. This rule differs from that in CL, which often allows *USRLIBL.
  - Code a maximum of 10 characters for object names. If you enclose the name in quotation marks, you may specify up to 8 characters between the quotation marks. This rule is different from that in CL, which allows a basic name of up to 10 characters to be specified between the quotation marks. Refer to the *CL Reference* manual for syntax rules for object names.
- Record and field names
  - The DDS syntax rules for record and field names are:
    - Names must be 10 characters or less.
    - Names must begin with an alphabetic character (A through Z, @, $, and #). All subsequent characters can be alphanumeric (A through Z, 0 through 9, @, $, #, and _ (underscore)). There can be no embedded blanks.
    - In ICF files, record names cannot start with $$.
  - Specify qualified field names similar to qualified names. For example:

    ```
    KEYWORD(record-name/field-name)
    ```

  High-level languages can impose specific length and value restrictions on the name. Check the appropriate high-level language reference guide for the syntax requirements for your high-level language processor.

- ALIAS (alternative field) names
  - The length of an alternative field name is 1 to 30 characters. The first character must be A through Z. Subsequent characters must be A through Z, 0 through 9, or the underscore (_).
  - Because DDS does not perform any language-specific syntax checking, you must make sure that the alternative field names you specify conform to the naming conventions of the high-level language that uses the names. The high-level language compiler checks the syntax of the names when they are brought into the program.
- Message identifiers
  - Message identifiers must be 7 characters long. The first 3 characters are the message prefix.
  - The first character of the message prefix must be an alphabetic character (A through Z, @, $, and #). The next 2 characters of the message prefix must be alphanumeric (A through Z, @, $, #, _, 0 through 9).
  - The last 4 characters must be a hexadecimal value (0 through 9, A through F).
- Label, document, and folder names
  - An online help information label name must be from 1 to 10 characters long and must begin with an uppercase alphabetic character (A through Z, @, #, or $). The label name cannot contain a comma, an apostrophe, or an embedded blank.
  - A document name (and a simple folder name) must be a 1 to 8 character part. It can be followed by a period and a 1 to 3 character part called an extender. The characters used most often are A through Z, 0 through 9, @, #, $, and _. See the *Using OfficeVision/400* Word Processing* manual for the syntax rules for document names.
  - If a folder name is concatenated, each simple folder name is separated by a forward slash (/). The total length of the folder name cannot exceed 63 characters.
  - In DDS, a document, simple folder name, or online help information label name can be enclosed in apostrophes. The enclosing apostrophes are required when the name contains an opening or closing parenthesis or an apostrophe character. When the name is enclosed in apostrophes, specify two apostrophes for each apostrophe character within the name. If a folder name is concatenated, the enclosing apostrophes, if specified, must be around the entire concatenated name.

Syntax Coding Examples

Figure 1-2 through Figure 1-7 are syntax coding examples. Except for HLPARA, JFILE, JFLD, and PFILE, the keywords shown in these examples are not actual keywords. They simply indicate where you should specify the keywords.

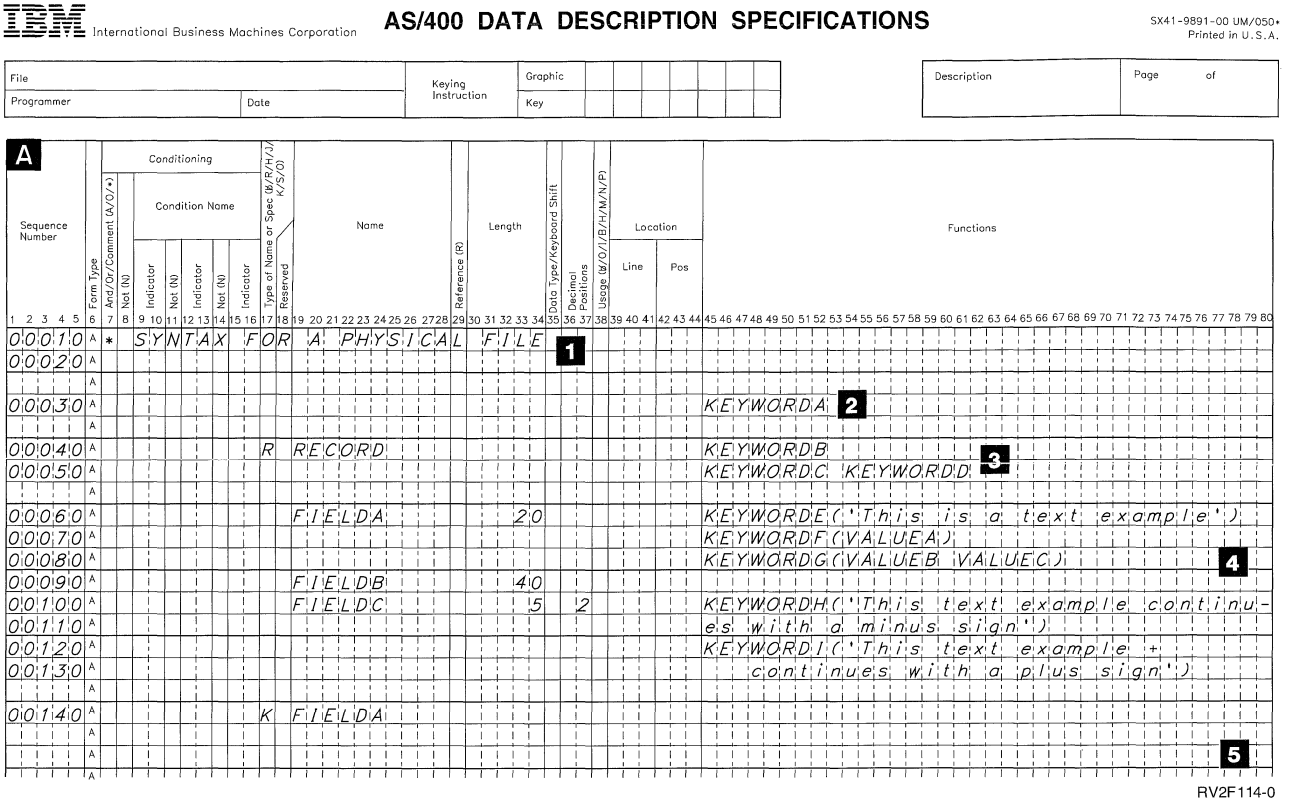


Figure 1-2. Syntax for a Physical File

1 Comments (optional): Comments can appear on any line in DDS. They are identified by an asterisk in position 7.

2 File level (optional): File-level keywords appear before the record format name (RECORD on line 00040).

3 Record level (only one record is allowed in physical files): R in position 17 identifies RECORD as a record format name. The record level continues until the first field is named.

4 Field level (at least one field name is required, unless the FORMAT keyword is specified on the record): For fields in physical files, specify at least a name and length. Other attributes can be specified explicitly or by default.

5 Key field level (optional): K in position 17 identifies the field as a key field. A K must be specified for each key field. Specify the key field level by repeating a field name (here, FIELDA) after the field-level specifications.

IBM International Business Machines Corporation

AS/400 DATA DESCRIPTION SPECIFICATIONS

SX41-9891-00 UM/050• Printed in U.S.A.

File		Keying Instruction	Graphic								Description	Page of
Programmer	Date		Key									

A

Sequence Number	Form Type	And/Or/Comment (A/O/*)	Conditioning: Not (N)	Conditioning: Condition Name: Indicator	Not (N)	Indicator	Not (N)	Indicator	Type of Name or Spec (B/R/H/J/K/S/O)	Reserved	Name	Reference (R)	Length	Data Type/Keyboard Shift	Decimal Positions	Usage (B/O/I/B/H/M/N/P)	Location: Line	Location: Pos	Functions
1 2 3 4 5	6	7	8	9 10	11	12 13	14	15 16	17	18	19 20 21 22 23 24 25 26 27 28	29	30 31 32 33 34	35	36 37	38	39 40 41	42 43 44	45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
00010	A	*									SYNTAX FOR A SIMPLE LOGICAL FILE								1
00020	A	*																	
	A																		
00030	A																		KEYWORDA 2
	A																		
00040	A								R		RECORD1								PFILE(PF1)
00050	A																		KEYWORDC KEYWORDD 3
	A																		
00060	A										FIELDA								KEYWORDE('This is a text example')
00070	A																		KEYWORDF(VALUEA)
00080	A																		KEYWORDG(VALUEB VALUEC) 4
00090	A										FIELDB		40						
00100	A										FIELDC		5	2					KEYWORDH('This text example continu-
00110	A																		es with a minus sign')
00120	A																		KEYWORDI('This text example +
00130	A																		continues with a plus sign')
	A																		
00140	A								K		FIELDA								5
	A																		
00150	A								S		FIELDB								KEYWORDJ 6
	A																		

RV2F115-0

Figure 1-3. Syntax for a Simple Logical File

1 Comments (optional): Comments can appear on any line in DDS. They are identified by an asterisk in position 7.

2 File level (optional): File-level keywords appear before the record format name (RECORD1 on line 00040).

3 Record level (at least one is required): R in position 17 identifies RECORD1 as a record format name. In simple or multiple format logical files, the PFILE keyword is required for every record format. The record level continues until the first field is named.

4 Field level: Field names and field attributes are not required for logical files. See Chapter 2, "Physical and Logical Files" for more information.

5 Key field level (optional): K in position 17 identifies the field as a key field. A K must be specified for each key field. Specify the key field level by repeating one or more field names (such as FIELDA) after the field-level specifications.

6 Select and omit levels (optional): S in position 17 identifies FIELDB as a select field. (O in position 17 identifies a field as an omit field.) The select and omit levels follow the key field level.

Note: To form a multiple format logical file, specify more record formats within the file by repeating items **3** through **6**, or specify more than one file on the PFILE keyword.

IBM International Business Machines Corporation

AS/400 DATA DESCRIPTION SPECIFICATIONS

SX41-9891-00 UM/050*
Printed in U.S.A.

File		Keying Instruction	Graphic							Description	Page of
Programmer	Date		Key								

Sequence Number	Form Type	And/Or/Comment (A/O/*)	Conditioning: Not (N)	Condition Name: Indicator	Condition Name: Not (N)	Condition Name: Indicator	Condition Name: Not (N)	Condition Name: Indicator	Type of Name or Spec (R/F/H/J/K/S/O)	Reserved	Name	Reference (R)	Length	Data Type/Keyboard Shift	Decimal Positions	Usage (I/O/B/H/M/N/P)	Location: Line	Location: Pos	Functions
1 2 3 4 5	6	7	8	9 10	11	12 13	14	15 16	17	18	19 20 21 22 23 24 25 26 27 28	29	30 31 32 33 34	35	36 37	38	39 40 41	42 43 44	45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
A																			
00010	A	*									SYNTAX FOR A JOIN LOGICAL FILE								1
00020	A	*																	
	A																		
00030	A																		KEYWORDA 2
	A																		
00040	A								R		RECORD1								JFILE(PF1 PF2)
00050	A																		KEYWORDC KEYWORDD 3
	A																		
00060	A								J										JFLD(FIELDA FLDA) 4
	A																		
00070	A										FIELDA								KEYWORDE('This is a text example')
00080	A																		KEYWORDF(VALUEA)
00090	A																		KEYWORDG(VALUEB VALUEC) 5
00100	A										FIELDB		40						
00110	A										FIELDC		5		2				KEYWORDH('This text example continu-
00120	A																		es with a minus sign')
00130	A																		KEYWORDI('This text example +
00140	A																		continues with a plus sign')
00150	A								K		FIELDA 6								
	A																		
00160	A								S		FIELDB 7								KEYWORDJ
	A																		
	A																		

*Number of sheets per pad may vary slightly.

RV2F116-0

Figure 1-4. Syntax for a Join Logical File

1 Comments (optional): Comments can appear on any line in DDS. They are identified by an asterisk in position 7.

2 File level (optional): File-level keywords appear before the record format name (RECORD1 on line 00040).

3 Record level (exactly one required): R in position 17 identifies RECORD1 as a record format name. In join logical files, the JFILE keyword is required for the record format. The record level continues until the first join specification.

4 Join level: J in position 17 identifies the beginning of a join specification. At the join level, specify at least one join specification. Each join specification must include at least one JFLD keyword. There must be one JOIN keyword for each join specification in a join logical file if there is more than one join specification in the file. A join specification continues until the next join specification or field name.

5 Field level: At least one field name with usage other than N is required for join logical files.

6 Key field level (optional): K in position 17 identifies the field as a key field. A K must be specified for each key field. Specify the key field level by repeating one or more field names (such as FIELDA) after the field-level specifications.

7 Select and omit levels (optional): S in position 17 identifies FIELDB as a select field. (O in position 17 identifies a field as an omit field.) The select and omit levels follow the key field level.

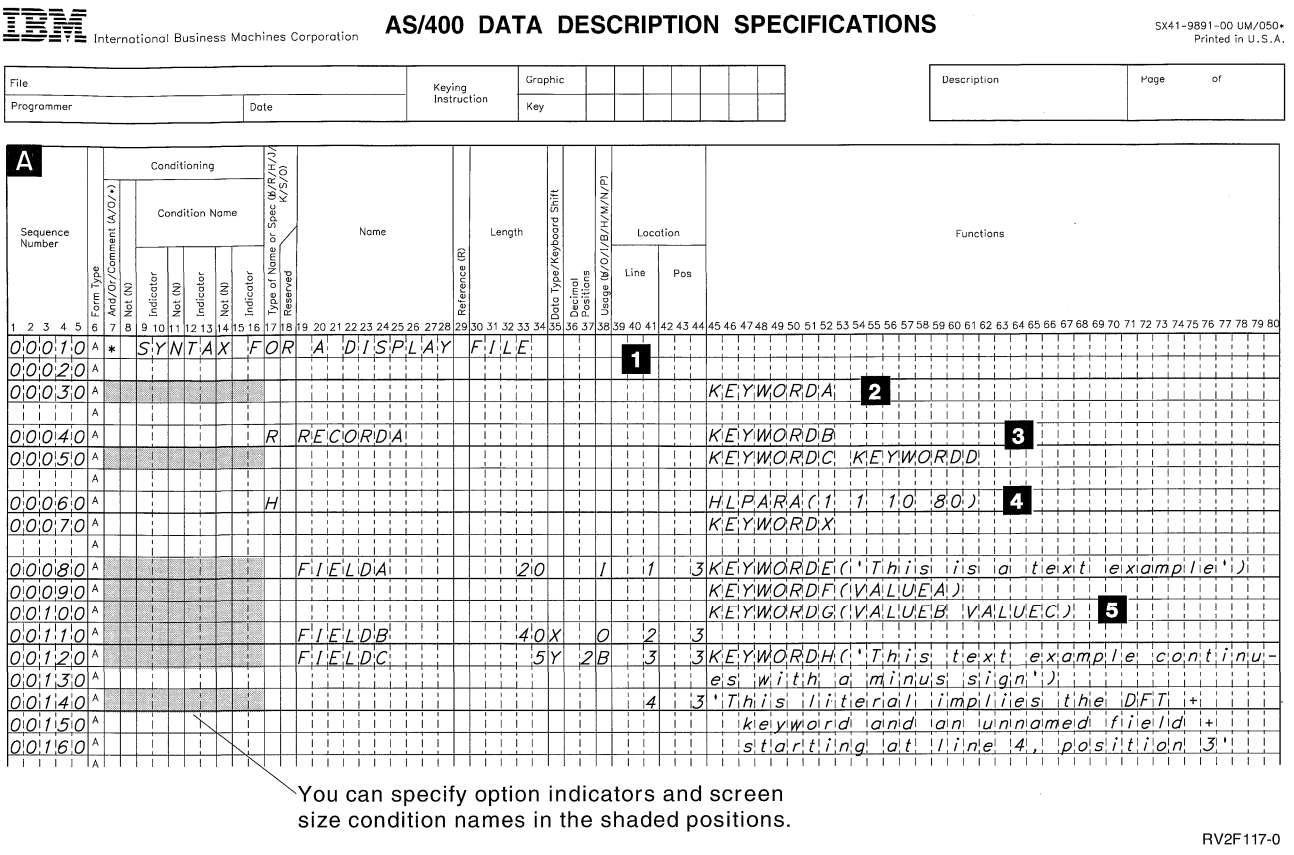


Figure 1-5. Syntax for a Display File

1 Comments (optional): Comments can appear on any line in DDS. They are identified by an asterisk in position 7.

2 File level (optional): File-level keywords appear before the first record format name (RECORDA on line 00040).

3 Record level (at least one required): R in position 17 identifies RECORDA as a record format name. The record level continues until either the first field is specified or the first help specification.

4 Help level (optional): H in position 17 identifies the beginning of a help specification. A help specification continues until the next H in position 17 or until the first field. Each help specification must include at least one HLPARA keyword, and a HLPRCD or HLPDOC keyword.

5 Field level (optional): Display file fields that are passed between the display device and the program must be named fields and must have a length specified. Other attributes can be specified explicitly or by default. Constant (unnamed) fields require only a location and a keyword, as described in DATE, DFT, TIME, and MSGCON keyword descriptions in Chapter 3, "Display Files." Positions 17 through 38 do not apply to constant fields.

Note: Items **3** through **5** can be repeated to specify new record formats within the display file.

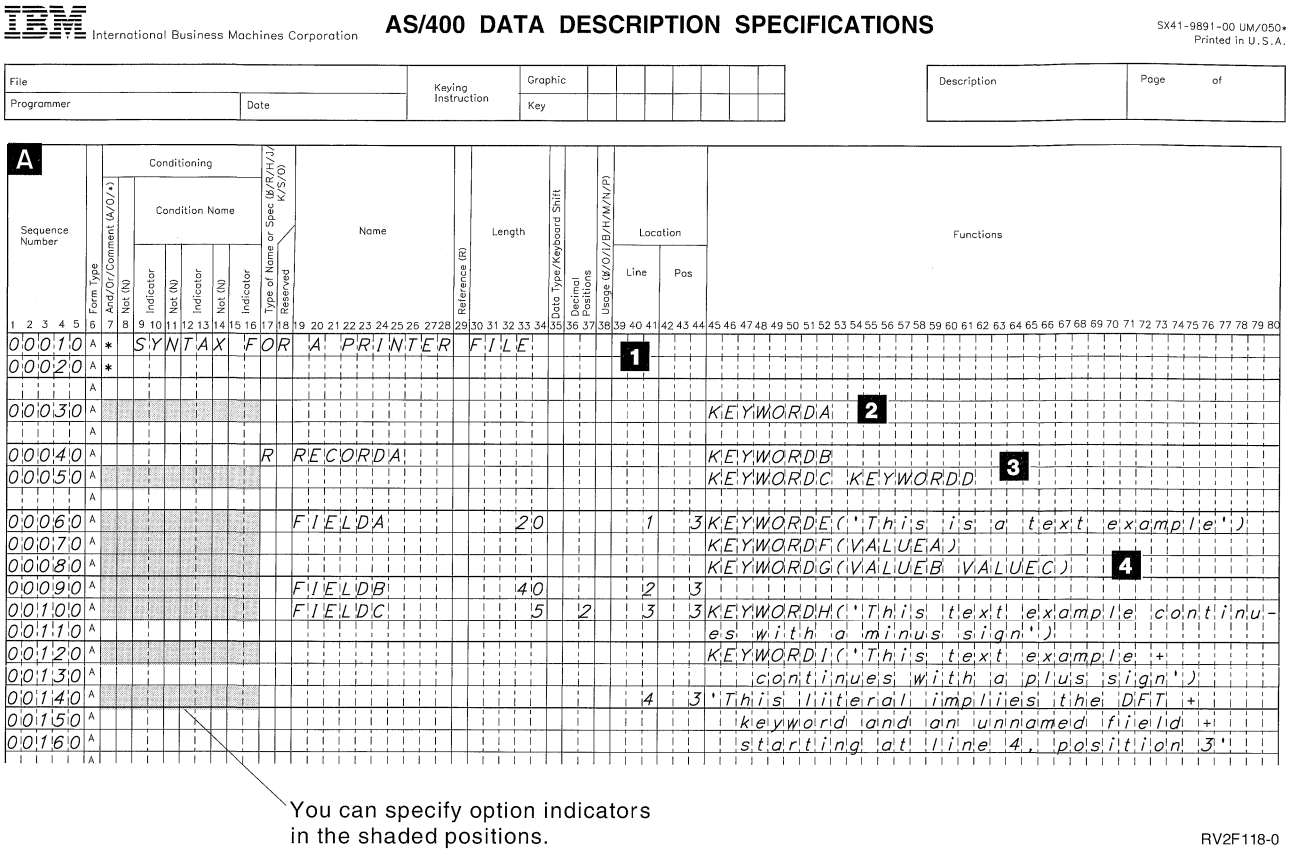


Figure 1-6. Syntax for a Printer File

1 Comments (optional): Comments can appear on any line in DDS. They are identified by an asterisk in position 7.

2 File level (optional): File-level keywords appear before the first record format name (RECORDA on line 00040).

3 Record level (at least one required): R in position 17 identifies RECORDA as a record format name. The record level continues until the first field is specified.

4 Field level (at least one field, whether named or unnamed, is required in each record format in the file): Printer file fields that are passed from the program to the printer must be named fields and must have a length specified. Other attributes can be specified explicitly or by default. Constant (unnamed) fields require only a location and a keyword, as described in the DATE, DFT, PAGNBR, TIME, and MSGCON keyword descriptions in Chapter 4, "Printer Files."

Note: Items **3** and **4** can be repeated to specify new record formats within the printer file.

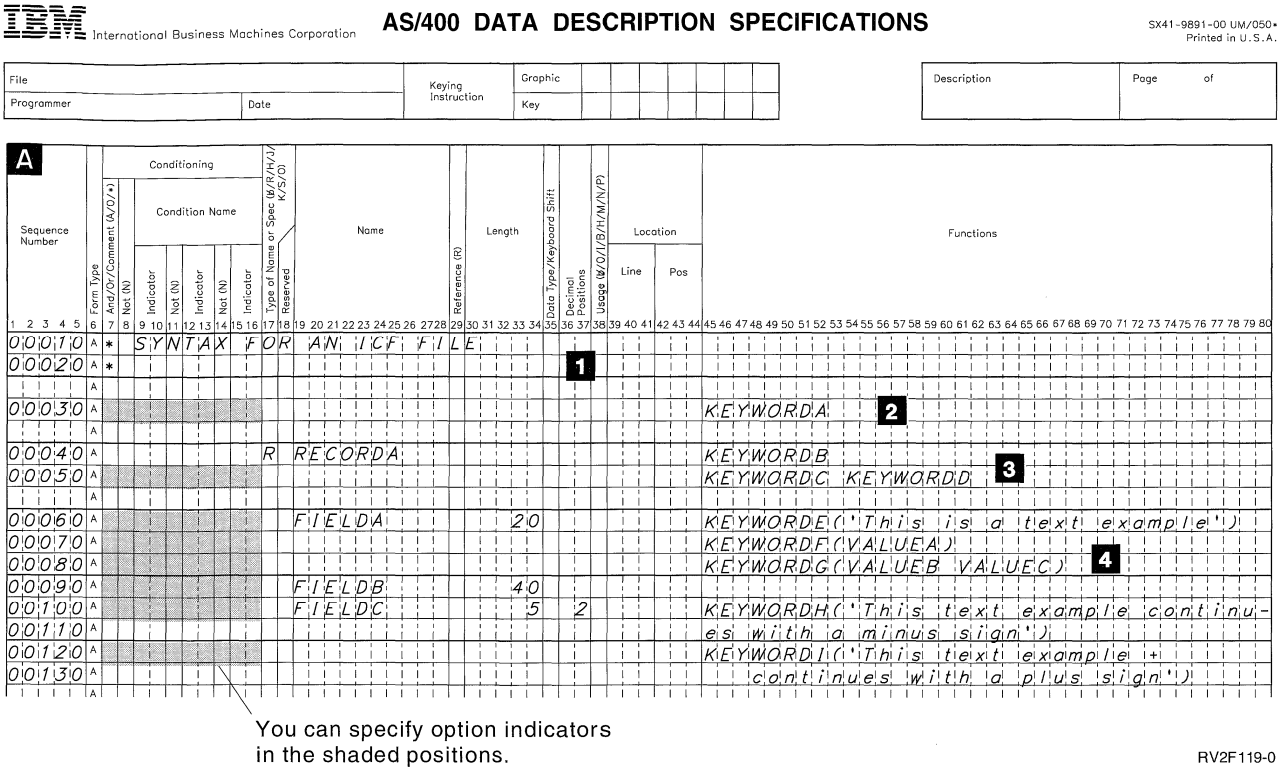


Figure 1-7. Syntax for an ICF File

1 Comments (optional): Comments can appear on any line in DDS. They are identified by an asterisk in position 7.

2 File level (optional): File-level keywords appear before the first record format name (RECORDA on line 00040).

3 Record level (at least one required): R in position 17 identifies RECORDA as a record format name. The record level continues until the first field is specified.

4 Field level (optional): ICF file fields must have at least a name (as in FIELDA) and a length. Other attributes can be specified explicitly or by default.

Note: Items **3** and **4** can be repeated to specify new record formats within the ICF file.

Chapter 2. Physical and Logical Files

This chapter provides the following information regarding physical and logical files:

- Definition
- Positional entries
- Keyword entries

"Positional Entries (Positions 1 through 44)" on page 2-4 gives rules and examples for filling in positions 1 through 44 of the data description specifications (DDS) form. "Keyword Entries (Positions 45 through 80)" on page 2-30 gives rules and examples for specifying DDS keywords. The keywords are described in alphabetical order.

For more information about choosing positional entries and keywords for physical and logical files, see the *Database Guide*.

Defining a Physical File

A physical file can contain only one record format. Specify the record format in either of two ways:

- Define a new record format. Specify field and key field specifications as desired for the new record format.
- Share an existing record format. Use the FORMAT keyword to specify that the OS/400 program is to use a previously defined record format from another physical file. When the FORMAT keyword is used, key field level specifications must be specified again (if a keyed access path is desired) even if they were specified on the existing record format.

Specify the entries in the following order to define a physical file:

1. File-level entries (optional)
2. Record-level entries
3. Field-level entries
4. Key field-level entries (optional)

Note: The file name is specified through the Create Physical File (CRTPF) command, not through DDS.

You can find an explanation of file-, record-, field-, and key field-level specifications in Chapter 1, "Introduction."

You can find the syntax rules for specifying DDS keywords in "Syntax Rules" on page 1-4.

The maximum number of fields in a record format is 8000. If any of the fields in the record format are date, time, timestamp, variable length, or allows the null value, then the actual maximum number of fields can be less than 8000. The maximum number of fields can vary depending on the number of fields and combinations of fields that occur within the record format. The maximum number of bytes in a record format is 32 766 if variable length fields are not included and 32 740 if variable length fields are included. Figure 2-17 on page 2-23 describes rules for determining the total length of the record format.

© Copyright IBM Corp. 1991, 1993

Defining a Logical File

A logical file determines how data records are selected and defined when read by an application program. A logical file can be a *simple*, *multiple format*, or *join* logical file. A simple logical file contains one record format and has one file specified on the PFILE keyword. A multiple format logical file either contains more than one record format or has more than one file specified on the PFILE keyword. A join logical file contains one record format and has up to 32 files specified on the JFILE keyword.

Simple and Multiple Format Logical Files

You must specify the PFILE keyword at the record level for simple and multiple format logical files. In a multiple format logical file, a record format can use only the fields common to all the physical files specified on the PFILE keyword for that record format.

Specify the entries in the following order to define a simple or multiple format logical file:

1. File-level entries (optional)
2. Record-level entries
3. Field-level entries (optional)
4. Key field-level entries (optional)
5. Select and omit-field level entries (optional)

Repeat steps 2 through 5 for each record format in the file.

Figure 2-1 shows a multiple format logical file coding example.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* LOGICAL FILE EXAMPLE
00020A* INVENTORY FORMAT
00030A          R INVFMT                    PFILE(INVENTORY)
00040A          K ITEM
00050A*
00060A* ORDER FORMAT
00070A          R ORDFMT                    PFILE(ORDER)
00080A                                      TEXT('ORDER ANALYSIS')
00090A            ITEM
00100A            ORDER         10
00110A            SUPPLY        +2
00120A            SHPDAT                    CONCAT(SHPMO SHPDA SHPYR)
00130A            QTY            5P         RENAME(QTYDUE)
00140A          K ITEM
00150A          K SHPYR
00160A          K SHPMO
00170A          K SHPDA
00180A          O QTYDUE                    CMP(LT 1)
00190A*
00200A* ACCOUNTING FORMAT
00210A          R ACTFMT                    PFILE(ACCOUNTS)
00220A                                      FORMAT(ACCOUNTL)
00230A          K ITEM
     A
```

Figure 2-1. Multiple Format Logical File Coding Example

Join Logical Files

Join logical files combine different fields from more than one physical file into a single record. You must specify the JFILE keyword at the record level for join logical files.

Specify the entries in the following order to define a join logical file:

1. File-level entries (optional)
2. Record-level entries
3. Join-level entries
4. Field-level entries
5. Key field-level entries (optional)
6. Select/omit-field level entries (optional)

Because only one record format is allowed in a join logical file, specify these entries only once.

Figure 2-2 shows a join logical file coding example.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* Joins fields from two physical files into one record format
00020A          R RECORD1                   JFILE(PF1 PF2)
00030A          J                           JOIN(PF1 PF2)
00040A                                      JFLD(NAME NAME)
00050A            NAME                      JREF(1)
00060A            ADDR
00070A            PHONE
     A
```

Figure 2-2. Join Logical File Coding Example

You can find an explanation of file-, record-, join-, field-, key field-, and select/omit field-levels in Chapter 1, "Introduction."

You can find the syntax rules for specifying DDS keywords in "Syntax Rules" on page 1-4.

For more information on using simple, multiple format, and join logical files, see the *Database Guide*.

Specifying Record Formats in a Logical File

If there is more than one record format specified in a logical file, you must specify the PFILE keyword for each record format.

There are three ways to specify the fields in a record format:

- Specify the record format name and the PFILE keyword.
- Specify the record format name, the PFILE or JFILE keyword, and at least one individual field.
- Specify the record format name, the PFILE keyword, and the FORMAT keyword.

Figure 2-1 on page 2-2 illustrates the three ways to specify fields.

For each of the three ways to specify fields in a record format, you can have one of the following *access path* specifications:

- Specify no key fields (arrival sequence access path). You cannot specify select/omit fields unless you specify the DYNSLT keyword. You can specify only one record format with one physical file on the PFILE keyword for the logical file.
- Specify one or more key fields (keyed sequence access path). If you specify more than one record format in the logical file, each record format must have at least one key field specified. You can specify select/omit fields for any of the record formats in the file.
- Specify the REFACCPTH keyword (keyed sequence access path). The access path information from another physical or logical file is copied into the file you are defining.

The maximum number of fields in a record format is 8000. If any of the fields in the record format are date, time, timestamp, variable length, or allows the null value, then the actual maximum number of fields can be less than 8000. The maximum number of fields can vary depending on the number of fields and combinations of fields that occur within the record format. The maximum number of bytes in a record format is 32 766 if variable length fields are not included and 32 740 if variable length fields are included. See Figure 2-17 on page 2-23 for rules on determining the total length of the record format.

Positional Entries (Positions 1 through 44)

This section describes how to specify the first 44 positions of the data description specifications form for physical and logical files. To code the remainder of the form, see "Keyword Entries (Positions 45 through 80)" on page 2-30.

Figure 2-3 shows some positional entries for physical files.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* PHYSICAL FILE CODING EXAMPLE
00020A                                     REF(INVENCTL/INVENTORY)
00030A                                     UNIQUE
00040A          R ORDFMT                   TEXT('Format for Purchase Orders')
00050A            ORDNBR         7  0      COLHDG('Order' 'Number')
00060A            ITMNBR    R    10
00070A            SUPNBR    R    +2        REFFLD(SUPID SUPLIB/SUPMST)
00080A            QTYORD         5B
00090A          K ORDNBR
00100A          K ITMNBR                   ABSVAL
     A
```

Figure 2-3. Physical File Coding Example

Figure 2-1 on page 2-2 and Figure 2-2 on page 2-3 show positional entries for multiple format and join logical files.

Sequence Number (Positions 1 through 5)

Use these positions to specify a sequence number for each line on the form. The sequence number is optional and is used for documentation purposes only.

Form Type (Position 6)

Type an A in this position to designate this as a DDS form. The form type is optional and is for documentation purposes only.

Comment (Position 7)

Type an asterisk (*) in this position to identify this line as a comment. Use positions 8 through 80 for comment text. A blank line (no characters specified in positions 7 through 80) is treated as a comment. Comments can appear anywhere in DDS and are kept only in the source file. Comments are printed on the source computer printout but not on the expanded source computer printout.

Conditioning (Positions 8 through 16)

These positions do not apply to physical or logical files. Leave these positions blank unless you use them for comment text.

Type of Name or Specification (Position 17)

Type a value in this position to identify the type of name or, for logical files, the type of specification. If you specify a name type, the name is specified in positions 19 through 28.

The valid entries for physical files are:

Entry	Meaning
R	Record format name
Blank	Field name
K	Key field name

Note: Specify only one R because a physical file can contain only one record format.

The valid entries for logical files are:

Entry	Meaning
R	Record format name
J	Join specification
Blank	Field name or select/omit AND condition
K	Key field name
S	Select field name
O	Omit field name

For more information on types of names, see "Name (Positions 19 through 28)" on page 2-6. For more information on join specifications, see "JOIN (Join)—Join Logical Files Only" on page 2-60.

Reserved (Position 18)

This position does not apply to any file type. Leave this position blank unless you use it for comment text.

Name (Positions 19 through 28)

Use these positions to specify names of the following:

- The record format for this physical file or formats for this logical file
- The field or fields that make up the record format (unless you specify the FORMAT or PFILE keyword at the record level)
- The field or fields used as key fields
- For logical files, the field or fields to be used for select/omit specifications

Note: The file name is specified through the Create Physical File (CRTPF) command, not in the DDS.

Refer to "Syntax Rules" on page 1-4 for rules to use when specifying record or field names in DDS.

Names must begin in position 19.

You must specify the name type in position 17, unless you are specifying a field name or select/omit AND condition.

Figure 2-3 on page 2-4 shows how to code names for a physical file. Figure 2-1 on page 2-2 and Figure 2-2 on page 2-3 show how to code names for logical files.

Record Format Name

When you specify R in position 17, the name specified in positions 19 through 28 is a record format name.

Physical Files: Only one record format name is allowed for a physical file. Specify the record format name in one of two ways:

- As the name of a *new* record format with field names specified in this physical file. The name of the record format can be the same as the file name specified in the Create Physical File (CRTPF) command. However, a warning message appears if the names are not unique, because some high-level language processors do not allow record format and file names to be the same. RPG III is such a high-level language. The record format name and field names do not have to be unique to the system; the same names can exist in another file.
- As the name of a record format *previously defined* in another physical file. The FORMAT keyword must be specified. Field names and attributes are not specified. See "FORMAT (Format)" on page 2-53 for an explanation of the FORMAT keyword.

Simple and Multiple Format Logical Files: You can specify more than one record format name. However, each name must be unique within the file. See the appropriate high-level language manual for exceptions.

Specify the record format name in one of three ways:

- As the record format name in the first physical file specified on the PFILE keyword. This is required if you do not specify the FORMAT keyword and do not identify individual fields by naming them in this record format.
- As the name of a *new* record format with field names specified in this logical file. Every field must be identified by name. No unnamed physical file fields are part of this logical file record format. Physical file fields that are parameters

of RENAME and CONCAT keywords are part of the logical file record format. Physical file fields that are parameters of SST keywords are not part of the logical file record format unless specified elsewhere.

- As the name of a record format *previously described* in a physical or logical file. Field names and attributes are not specified and the FORMAT keyword must be specified. For a description of how to specify the FORMAT keyword, see "FORMAT (Format)" on page 2-53.

The record format name can be the same as the file name specified in the create file command. However, a warning message is sent if the names are not unique. Some high-level language processors, such as RPG III, do not allow record format and file names to be the same.

Use the PFILE keyword in conjunction with the record format name to specify the physical files with which the record format is to be associated. A record format can have more than one physical file specified on the PFILE keyword. If no fields are defined and the FORMAT keyword is not specified, the format of the first file specified in the PFILE keyword is used as the format for all the physical files. (This format is used for field attribute references and attribute and name checking.)

Join Logical Files: Only one record format name can be specified. Specify the record format name as the name of a new record format with field names specified in this logical file. Every field in the record format for a join logical file must be identified by the name in positions 19 through 28. Physical file fields that are parameters of the RENAME, CONCAT, and SST keywords are part of the logical file record format only if you specify the field names elsewhere in the record format.

The JFILE keyword is required at the record level. It specifies the physical files that the record format joins.

Field Name

When position 17 is left blank, the name specified in positions 19 through 28 is a field name. You cannot specify field names if you specify the FORMAT keyword.

Physical files require that each field be named. These names must be unique within the record format. The field names appear in the physical buffer in the same order that they are specified in the DDS.

If you are describing a simple or multiple format logical file, you can use the record format as it exists in the physical file on which this logical file is based, and you do not have to specify field names.

If you do not use the record format as it exists in the physical file, you must name each field specified in a logical file. In a simple or multiple format logical file, each field name must be unique within the record format and must correspond to a field in the physical file record format. The field name order is the order in which the fields appear to programs using the logical file.

The name you give to a field in a logical file record format is usually the same as the corresponding field name in the physical file record format. If different, the two names must be equated by using the RENAME keyword. A field in a logical file record format can also represent the concatenation of two or more fields from the physical file (see "CONCAT (Concatenate)—Logical Files Only" on page 2-40).

The SST keyword can also be used to describe a substring of a field from the physical file in the logical file format.

Note: The sequence in which the field names are specified in the logical file is important. If the same physical field is specified more than once in a record format in the logical file (by using either RENAME or CONCAT), the sequence in which the fields are specified in the logical file is the sequence that the data is moved to the physical file. Thus, the value of the field the last time the field is specified in the logical file is the value in the physical record.

Key Field Name

When you specify K in position 17, the name specified in positions 19 through 28 is a key field name. It must be one of the field names within the physical file record format. The contents of this field are used to sequence the records for retrieval from the database. Specifying a key is optional. If no key field is specified, the default sequence is arrival sequence (the order that the records were put into the file). See the *Database Guide* for information on access paths and the effects they have on save/restore operations.

Use key fields (and optionally, select/omit fields) to define a keyed sequence access path for record formats in the logical file member. The logical file member includes the physical file members specified on the DTAMBRS parameter on the Create Logical File (CRTLF) or Add Logical File Member (ADDLFM) commands.

You can change the sequence of records as they are read from the file by specifying a sequencing keyword. The sequencing keywords are ALTSEQ, NOALTSEQ, SIGNED, UNSIGNED, ABSVAL, ZONE, DIGIT, DESCEND, FCFO, FIFO, and LIFO. Refer to the discussion of each of these keywords for more information.

When you do not specify any sequencing keywords for a key field, the default sequence for that key field is ascending order. The default for character, hexadecimal, date, time, and timestamp fields is the UNSIGNED attribute. The default for numeric fields is the SIGNED attribute, except for zoned decimal fields (S specified in position 35) in the following cases:

- When you specify ALTSEQ at the file level, all zoned decimal key fields in the file default to UNSIGNED.
- When you specify DIGIT or ZONE for a zoned decimal key field, the field defaults to UNSIGNED.

If you specify more than one record format for a logical file or more than one physical file for the PFILE keyword, you must specify at least one key field for all record formats of that logical file.

A key can have more than one key field. This is called a *composite key*. In a composite key, specify the key field names in the order of importance (major to minor), and specify each key field name on a separate line.

Figure 2-4 on page 2-9 shows a multiple format logical file with two record formats, one of which uses a composite key. In this example, RECORD1 has a single key field, FIELD1. RECORD2 has a composite key that includes FIELD4 and FIELD5.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   PFILE(PF1)
00020A            FIELD1
00030A            FIELD2
00040A            FIELD3
00050A          K FIELD1
00060A*
00070A          R RECORD2                   PFILE(PF2)
00080A            FIELD4
00090A            FIELD5
00100A          K FIELD4
00110A          K FIELD5
     A
```

Figure 2-4. Specifying a Multiple Format Logical File with Two Record Formats

If you do not specify a key field for a logical file, the file you are defining has an arrival sequence access path.

The number of fields that make up a key is restricted to 120. The total key length cannot exceed 2000 bytes. (If the FCFO keyword is specified, the total key length cannot exceed 1995 bytes.) The total key length includes the length of each key field. If any of the key fields allow the null value, add 1 byte for each key field that allows the null value. The OS/400 program uses the extra byte to determine whether the key contains the null value. If any of the key fields is variable length, add 2 bytes for each variable-length key field. The OS/400 program uses the extra 2 bytes to store the allocated length of the field.

When you specify more than one record format in a logical file, an additional byte for the first *NONE key field position is required. An additional byte also may be required for each additional key field position. The OS/400 program uses the extra bytes when records from different physical files have duplicate key values.

For example, suppose a key consists of fields named FIELDA, FIELDB, and FIELDC (in that order). The DDS appears as shown in Figure 2-5.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* SAMPLE COMPOSITE KEY (PHYSICAL FILE)
00020A          R RECORD
00030A            FIELDA         3  0
00040A            FIELDB         3  0
00050A            FIELDC         3  0
00060A            FIELDD         3  0
00070A          K FIELDA
00080A          K FIELDB
00090A          K FIELDC
     A
```

Note: Lines 00070 to 00090 make up the composite key.

Figure 2-5. Composite Key

The records are sequenced in the following order:

- They are sequenced according to the contents of FIELDA.
- If two or more records with the same value in FIELDA exist, the OS/400 program sequences those records according to the values in FIELDB.
- If two or more of those records have the same value in both FIELDA and FIELDB, they are sequenced according to the values in FIELDC.

Consider the following file:

Record	FIELDA	FIELDB	FIELDC
1	333	99	67
2	444	10	45
3	222	34	23
4	222	12	01
5	222	23	45
6	111	06	89
7	222	23	67

Assuming ascending sequencing for all fields, the records are retrieved in this order:

Record	FIELDA	FIELDB	FIELDC
6	111	06	89
4	222	12	01
5	222	23	45
7	222	23	67
3	222	34	23
1	333	99	67
2	444	10	45

The following information applies:

- Because records 3, 4, 5, and 7 have the same contents in FIELDA, FIELDB becomes the determining field.
- Within those four records, 5 and 7 have the same values in FIELDB. For these two records, FIELDC becomes the determining field.
- If FIELDC also contains duplicate values, the records are retrieved in first-in first-out (FIFO), last-in first-out (LIFO), or first-changed first-out (FCFO) order. To guarantee the order, specify the FIFO keyword, the LIFO keyword, or the FCFO keyword. Specify the UNIQUE keyword to prevent duplicate key values.

See "SIGNED (Signed)" on page 2-73 for an example that includes a key field with negative (–) contents.

Special restrictions apply to key field specifications when either FILETYPE(*SRC) is used on the Create Physical File (CRTPF) command or for the Create Source Physical File (CRTSRCPF) command. See the *Database Guide* for information about key field specifications for source files.

For logical files, the following rules apply to fields that you specify as key fields:

- For simple and multiple format logical files, the following search order is used to match key field names with defined fields:

  1. Fields specified in DDS positions 19 through 28
  2. Fields specified as parameters on the CONCAT or RENAME keywords

  If the field name is specified more than once, the first occurrence is used.

  The field name on a CONCAT or RENAME keyword and the associated field name in positions 19 through 28 cannot both be specified as key fields.

  The parameter name on the SST keyword is not valid as a key field unless it is defined elsewhere in the logical file format.

- For join logical files, the key field name you specify must be specified at the field level in positions 19 through 28 and must be a field described in the primary file (the first physical file specified on the JFILE keyword).

  Note: If you specify a field as a parameter value on the CONCAT, RENAME, or SST keyword, but do not specify the field in positions 19 through 28 of the join logical file, you cannot specify the field as a key field.

If you are concatenating numeric with either character or hexadecimal, you cannot specify the numeric fields as key fields. If you are concatenating zoned decimal and fields of any other numeric data type, you cannot specify the fields of the other data types as key fields.

Figure 2-6 illustrates which concatenated fields can and cannot be used as key fields.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   PFILE(PF1)
00020A            FLD1
00030A            FLD2
00040A            Z                         CONCAT(ZFLD PFLD)
00050A            A                         CONCAT(AFLD NFLD)
00060A          K ZFLD
00070A          K AFLD
     A
```

Figure 2-6. Correct and Incorrect Concatenated Fields

In physical file PF1, ZFLD is zoned decimal and PFLD is packed decimal. Therefore Z is zoned decimal, and PFLD cannot be used as a key field. ZFLD and Z can be used as key fields but not in the same record format.

In physical file PF1, AFLD is a character field and NFLD is a numeric field. Therefore A is character, and NFLD cannot be used as a key field. AFLD and A can be used as key fields but not in the same record format.

Access Path Keywords: You can specify one or more access path keywords to affect the way the OS/400 program builds and uses key values. The access path keywords are:

File Level	Key Field Level
ALTSEQ	DESCEND
FCFO	DIGIT
FIFO	SIGNED
LIFO	UNSIGNED
REFACCPTH	ZONE
UNIQUE	

Different key fields within a composite key can have different access path keywords.

Logical Files with More than One Record Format: When you specify more than one record format in a logical file, you must specify at least one key field for every record format in the logical file. It is not necessary to specify the same number of key fields in each key. Also, key fields specified in one record format must have the same field attributes and access path keywords as the corresponding key fields in other record formats in the same logical file. For variable-length key fields, a

variable-length key field will not be allowed to align with a fixed-length key field, even if the field types and lengths are the same.

A key is required for every record format so that the logical file members can have a single access path sequencing records of each record format. When records are returned from the various members of the physical file on which the logical file is based, they are merged according to the values of the key fields in the access path for the logical file member.

When records of a logical file member are sequenced, the OS/400 program builds a *key value* for each record by concatenating the values in its key fields. The key value is then used to build the access path for use by your program. See the section on database records in the *Database Guide* for information about the I/O operations permitted by the OS/400 program.

Each key field in a composite key has a *key position*. The first key field specified is in position 1, the second key field specified is in position 2, and so on. During I/O operations to a logical file, the OS/400 program compares the key values of the records written to or read from the database. When you create a logical file that has more than one record format (with or without different key fields specified), the OS/400 program performs *key position attribute checking*. For key position attribute checking to succeed, key fields of different record formats that are in the same key positions must have the same data type, length, decimal positions, and access path keywords specified at the key field level. This ensures a meaningful record sequence from the comparisons made during an I/O operation.

Floating-point fields used as key fields must have the same data type and precision but need not have the same length and decimal positions.

In Figure 2-7, FIELD1, FLD1, and F1 must have the same attributes, and FIELD2, FLD2, and F2 must have the same attributes. FIELD1, FLD1, and F1 are in key position 1; FIELD2, FLD2, and F2 are in key position 2. One record format can have more key fields than another, and the additional fields do not need key position attribute checking. FLD3 is such a field.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                     PFILE(PF1)
00020A            FIELD1          5  0
00030A            FIELD2         10
00040A            FIELD3         10
00050A          K FIELD1
00060A          K FIELD2                      DESCEND
00070A*
00080A          R RECORD2                     PFILE(PF2)
00090A            FLD1            5  0
00100A            FLD2           10
00110A            FLD3           20
00120A          K FLD1
00130A          K FLD2                        DESCEND
00140A          K FLD3
     A*
     A          R RECORD3                     PFILE(PF3)
     A            F1             5  0
     A            F2            10
     A            F3            30
     A          K F1
     A          K F2                          DESCEND
     A
     A
```

Figure 2-7. Key Field Attribute Checking

For examples of key fields in a logical file with more than one record format, refer to Figure 2-1 on page 2-2. In Figure 2-1 on page 2-2, fields named ITEM are specified in each key. For record formats INVFMT and ACTFMT, ITEM is the only key field specified. For record format ORDFMT, a composite key is specified. This composite key includes ITEM, SHPYR, SHPMO, and SHPDA. Each of the fields used in a key must also exist at the field level. Therefore, ITEM must exist in the record format for the physical file INVENTORY so that it can be copied into this logical file for INVFMT. Also, ITEM must exist in the record format for the logical file ACCOUNTL so that it can be copied into this logical file for ACTFMT. ITEM must also exist in physical file ACCOUNTS.

Using *NONE in the Key Field: Two conditions occur in which key fields having the same key position should not be compared. The two conditions are:

- The key fields do not have the same field attributes (data type, length, decimal positions, or access path keywords at the field level).
- The key fields have the same attributes, but you do not want them to be merged and sequenced together.

To avoid unwanted comparisons between key fields, specify *NONE in place of one of them and move the displaced key field to the next key position. The OS/400 program compares the values of key positions before and after *NONE, but retrieves the affected records in the order in which the record formats are specified in the DDS for the logical file.

You can specify *NONE two or more times on the following lines to displace a key field to a key position for which a comparison of key field attributes is relevant to your application.

Figure 2-8 on page 2-14 shows *NONE as the key field.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                    PFILE(PF1)
     A            FIELD1          6A
     A            FIELD2          4A
     A            FIELD3         10A
     A          K FIELD1
     A          K FIELD2
     A          K FIELD3
     A
     A          R RECORD2                    PFILE(PF2)
     A            FLD1            3A
     A            FLD2            4A
     A            FLD3           12A
     A          K *NONE
     A          K FLD2
     A
     A          R RECORD3                    PFILE(PF3)
     A            F1              6A
     A            F2              4A
     A            F3             10A
     A          K F1
     A          K F2
     A          K F3
     A
```

*Figure 2-8. Specifying *NONE as the Key Field*

In Figure 2-8, the attributes for FIELD2, FLD2, and F2, must be identical. Since you specified *NONE for the first key field of the second record, then FIELD1 and F1 (first key field of the first record and first key field of the third record) must have identical attributes. FIELD3 and F3 must also have identical attributes; there is no corresponding field in the second record format.

Figure 2-9 through Figure 2-12 on page 2-18 illustrate how to use the key field.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CLSHST                    PFILE(CLSHSTP)
00020A          K EMPNBR  1
00030A          K CLSDTE  2
00040A*
00050A          R JOBHST                    PFILE(JOBHSTP)
00060A          K EMPNBR  1
00070A          K JOBDTE  2
     A
```

Figure 2-9. Specifying the Key Field (Example 1)

Record Format	Key Positions	
	1	2
CLSHST	EMPNBR	CLSDTE
JOBHST	EMPNBR	JOBDTE

Example 1: In Figure 2-9, a logical file views records of two physical files through two different record formats: CLSHST (class history) and JOBHST (job history). In the logical file, the records from the two physical files can be merged together and sequenced by employee identification number (EMPNBR) by specifying EMPNBR in key position 1.

All records that have the same key value for EMPNBR pertain to the same employee. To merge and sequence all records for a given employee into a single history of classes and job assignments, specify CLSDTE (date of class) and

JOBDTE (date of job assignment) in key position 2 for the two record formats, as shown in Figure 2-9.

Suppose that the job assignment dates and class dates are the dates (month/year) that the class or assignment started. Records for three students are retrieved in the following order:

EMPNBR	CLSDTE	JOBDTE	Description
1005	3/79		Completed class
1005	4/79		Left to begin new job
1005		4/79	Completed job
1005	6/79		Completed class
1006		1/79	Completed job
1006		2/79	Completed job
1006	3/79		Completed class
1006	5/79		Transferred to new location
1007		1/79	Completed job
1007		4/79	Completed job
1007		7/79	Completed job
1007	8/79		Left because of illness

The above report provides a continuous history for each student.

Example 2: In Figure 2-10, another logical file views the same two physical files as in Example 1, but the second record format in the logical file has *NONE specified in key position 2.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CLSHST                    PFILE(CLSHSTP)
00020A          K EMPNBR   1
00030A          K CLSDTE   2
00040A*
00050A*
00060A          R JOBHST                    PFILE(JOBHSTP)
00070A          K EMPNBR   1
00080A          K *NONE    2
00090A          K JOBDTE   3
     A
```

Figure 2-10. Specifying the Key Field (Example 2)

Record Format	Key Positions		
	1	2	3
CLSHST	EMPNBR	CLSDTE	*NONE
JOBHST	EMPNBR	*NONE	JOBDTE

As in Figure 2-9, all records from the two physical files are first merged and sequenced together on employee number (EMPNBR). However, the records for each student are merged and sequenced first on class date (CLSDTE) and then on job assignment date (JOBDTE). The set of records used for Figure 2-9 are now retrieved as follows:

EMPNBR	CLSDTE	JOBDTE	Description
1005	3/79		Completed class
1005	4/79		Left to begin new job
1005	6/79		Completed class
1005		4/79	Completed job
1006	3/79		Completed class
1006	5/79		Transferred to new location

EMPNBR	CLSDTE	JOBDTE	Description
1006		1/79	Completed job
1006		2/79	Completed job
1007	8/79		Left because of illness
1007		1/79	Completed job
1007		4/79	Completed job
1007		7/79	Completed job

When several adjacent record formats have *NONE in the same key position, they form a set, relative to record formats specified before and after them, that functions in sequencing as an individual record format. Key fields specified after *NONE serve to merge and sequence records of the formats within the set. The following example shows how several record formats function as a set.

Example 3: In Figure 2-11, consider a logical employee file over five physical files.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R EMPMST                    PFILE(EMPMSTP)
00020A          K EMPNBR   1
00030A*
00040A          R CLSREG                    PFILE(CLSREGP)
00050A          K EMPNBR   1
00060A          K CLSDTE   2
00070A*
00080A          R CLSHST                    PFILE(CLSHSTP)
00090A          K EMPNBR   1
00100A          K CLSDTE   2
00110A*
00120A          R JOBHST                    PFILE(JOBHSTP)
00130A          K EMPNBR   1
00140A          K *NONE    2
00150A          K JOBDTE   3
00160A*
00170A          R ACTHST                    PFILE(ACTHSTP)
00180A          K EMPNBR   1
00190A          K *NONE    2
00200A          K ACTDTE   3
     A
```

Figure 2-11. Specifying the Key Field (Example 3)

Record Format	Key Positions		
	1	2	3
EMPMST	EMPNBR	*NONE	*NONE
CLSREG	EMPNBR	CLSDTE	*NONE
CLSHST	EMPNBR	CLSDTE	*NONE
JOBHST	EMPNBR	*NONE	JOBDTE
ACTHST	EMPNBR	*NONE	ACTDTE

The records are merged and sequenced as follows:

1. All records are merged and sequenced by employee number.
2. For a given employee, records are sequenced by:
   a. The master record (of the EMPMST format)
   b. Records of the CLSREG and CLSHST formats, merged and sequenced together on values of CLSDTE (key position 2)

c. Records of the JOBHST and ACTHST formats, merged together and sequenced together on values of JOBDTE and ACTDTE (key position 3)

Specifying *NONE in the key definitions achieves this sequencing as follows:

- *NONE and a field name, CLSDTE, appear in the second key position of the adjacent formats, CLSHST and JOBHST. This effectively causes a split between the two formats after the preceding key position (position 1). Records of formats above the split are merged and sequenced with records of formats below the split only on values of EMPNBR.
- An implicit *NONE in the second key position of the format EMPMST forces a similar split.
- With *NONE in key position 2, the JOBHST and ACTHST formats form a set in which the values of JOBDTE and ACTDTE are compared in order to merge and sequence records of these two formats only.

The record sequence defined by the previous key specifications is totally dependent on the order in which the formats are specified. For example, if JOBHST had been specified before CLSHST, key position 2 would read:

```
*NONE, CLSDTE, *NONE, CLSDTE, *NONE
```

Here, the values of CLSDTE within CLSREG would not have been sequenced with the values of CLSDTE within CLSHST, and JOBDTE would not have been sequenced with ACTDTE.

Example 4: In Figure 2-12 on page 2-18, assume that an employee has repeated a class. To sequence two records with the same values for EMPNBR and CLSDTE, a third key field, DATE, is specified in record format CLSHST. However, DATE cannot be specified in the next available key position (position 3) because JOBDTE and ACTDTE appear in that position for other formats. If DATE is specified in this position, the attributes of DATE are compared with the attributes of CLSHST and JOBHST, and the key definitions are rejected.

To obtain the sequencing necessary, specify *NONE before DATE, displacing DATE to key position 4.

The DATE field can be shown in position 4 as in Figure 2-12 on page 2-18.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R EMPMST                    PFILE(EMPMSTP)
00020A          K EMPNBR
00030A*
00040A          R CLSREG                    PFILE(CLSREGP)
00050A          K EMPNBR
00060A          K CLSDTE
00070A*
00080A          R CLSHST                    PFILE(CLSHSTP)
00090A          K EMPNBR
00100A          K CLSDTE
00110A          K *NONE   1
00120A          K DATE    1
00130A*
00140A          R JOBHST                    PFILE(JOBHSTP)
00150A          K EMPNBR
00160A          K *NONE
00170A          K JOBDTE
00180A*
00190A          R ACTHST                    PFILE(ACTHSTP)
00200A          K EMPNBR
00210A          K *NONE
00220A          K ACTDTE
     A
```

Figure 2-12. Specifying the Key Field (Example 4)

Record Format	Key Positions			
	1	2	3	4
EMPMST	EMPNBR	*NONE	*NONE	*NONE
CLSREG	EMPNBR	CLSDTE	*NONE	*NONE
CLSHST	EMPNBR	CLSDTE	*NONE	DATE
JOBHST	EMPNBR	*NONE	JOBDTE	*NONE
ACTHST	EMPNBR	*NONE	ACTDTE	*NONE

Specifying DATE in key position 4 enables records from physical file CLSHSTP with identical values for EMPNBR and CLSDTE to be merged and sequenced according to the value for DATE. See the *Database Guide* for detailed information regarding records with duplicate key values.

Note: Since values are actually placed in the keys to ensure the sequencing in the previous examples, duplicate key values are not always predictable when *NONE is needed for logical files with more than one record format.

Select/Omit Field Name

Use select/omit fields to tell the OS/400 program how to select or omit records when your program retrieves them using this record format. The only records affected are those from the physical file(s) specified for the PFILE or JFILE keyword for this record format.

The following rules apply to select/omit fields in logical files:

- You can specify select/omit fields only if you also specify key fields or if you also specify the DYNSLT keyword for the file. You can also specify *NONE as a key field to satisfy the requirement for a key field when your application requires no key fields.
- For simple and multiple-format logical files, the OS/400 program uses the following search order to match select/omit field names with defined fields:
  - Fields specified in DDS positions 19 through 28

– Fields specified as parameters on the CONCAT or RENAME keywords

If the field name is specified more than once, the first occurrence is used.

The field name on a CONCAT or RENAME keyword and the associated field name in positions 19 through 28 cannot both be specified as select/omit fields.

The parameter name on the SST keyword is not valid as a select/omit field unless it is defined elsewhere in the logical file record format.

For join logical files, the select/omit field name you specify must be specified at the field level in positions 19 through 28.

When using the select/omit fields, specify either S or O in position 17. By specifying either S or O, the select and omit comparison statements are ORed together. The system treats the ORed select and omit comparison statements independently from one another. That is, if the select or omit comparison condition is met, the record is either selected or omitted. If the condition is not met, the system proceeds to the next comparison.

By specifying a blank in position 17, the select and omit comparison statements are ANDed together. The combined comparisons must be met before the record is selected or omitted. See Figure 2-13 on page 2-20 and Figure 2-14 on page 2-20. In positions 19 through 28, specify a field name whose contents at processing time determine whether the record is to be selected or omitted based on the select/omit keyword specified for this field. The select/omit keywords are COMP, RANGE, and VALUES. The last select/omit specification can be made with the ALL keyword, but a field name is not permitted.

The field must appear in both the physical file record format and the logical file record format. Select/omit statements must follow all field and key field level entries for the record format. You can specify both select and omit for the same record format. The following information applies:

- If you specify both select and omit for a record format, the order in which you specify them is important. The select/omit statements are processed in the order they are specified; if a record satisfies a statement, the record is either selected or omitted as specified, and remaining select/omit statements are not examined for that record. See Figure 2-15 on page 2-21.
- If you specify both select and omit statements, you can indicate whether records not meeting any of the values specified are to be selected or omitted. See “ALL (All)—Logical Files Only” on page 2-33 for more information.
- If you do not specify the ALL keyword, the action taken for the records that do not meet the values is the converse of the type of the last statement specified. Records that do not meet selection values are omitted, and records that do not meet omission values are selected.

There are limits to the number of select/omit statements you can specify in a single logical file. If you specify many select/omit statements and you cannot create the file, reduce the overhead for the file through the following changes in the specifications, in decreasing order of importance:

- Reduce the number of record formats in the file.
- Reduce the number of physical files specified on the PFILE or JFILE keyword.
- Reduce the number of fields used (single occurrences) in the select/omit specifications.

You cannot specify a floating-point field as a select/omit field.

It is possible to have an access path with select/omit and process the file in arrival sequence. For example, CPYF can be specified with FROMRCD(1) or the high-level language may not request keyed processing. In this case, the processing is the same as if the DYNSLT keyword had been specified.

Figure 2-13 shows how to specify the select/omit field using ANDed select statements.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                 PFILE(PARTS)
00020A            PNO
00030A            DSC
00040A            UPR
00050A            QOH
00060A          K PNO
00070A          S UPR                     COMP(GT 5.00)
00080A            QOH                     COMP(LT 10)
00090A          O                         ALL
     A
```

Figure 2-13. Specifying the Select/Omit Field (Example 1)

In Figure 2-13, records are selected only if they satisfy two select statements: the first statement selects records in which the value of field UPR is greater than 5.00, and the second statement selects records in which the value of field QOH is less than 10. S is not specified in position 17 for field QOH. Therefore, these select statements are ANDed together. For a record to be read by a program, both conditions specified must be true.

Figure 2-14 shows how to specify the select/omit field using an omit statement ORed with two select statements ANDed together.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                 PFILE(PARTS)
00020A            PNO
00030A            DSC
00040A            UPR
00050A            QOH
00060A          K PNO
00070A          O DSC                     COMP(EQ 'HAMMER')
00080A          S UPR                     COMP(GT 5.00)
00090A            QOH                     COMP(LT 10)
00100A          O                         ALL
     A
```

Figure 2-14. Specifying the Select/Omit Field (Example 2)

In Figure 2-14, records are supplied to the program if they pass both of the following tests:

- The DSC field is not equal to HAMMER.
- The UPR field is greater than 5.00 and the QOH field is less than 10.

Figure 2-15 on page 2-21 shows several ways to specify the same select/omit logic.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          S ST                        COMP(EQ 'NY')
00020A            REP                       COMP(EQ 'JSMITH')   1
00030A            YEAR                      COMP(LT 78)
00040A          O                           ALL
     A
00050A          O YEAR                      COMP(GE 78)
00060A          S ST                        COMP(EQ 'NY')       2
00070A            REP                       COMP(EQ 'JSMITH')
00080A          O                           ALL
     A
00090A          O REP                       COMP(NE 'JSMITH')
00100A          O ST                        COMP(NE 'NY')       3
00110A          S YEAR                      COMP(LT 78)
00120A          O                           ALL
     A
```

Figure 2-15. Specifying the Select/Omit Field (Example 3)

In Figure 2-15, you want to select all the records before 1978 for a sales representative named JSMITH in the state of New York. There are three ways to code this example.

1 All records must be compared with the select fields ST, REP, and YEAR before they can be selected or omitted.

2 All records in and after 1978 are omitted in the first comparison. Then, only the records before 1978 are compared with ST and REP. Only two select fields must be satisfied. This way is more efficient than method **1**.

3 All records that are not associated with JSMITH in the state of New York are omitted in the first and second comparisons. Then, all records left are compared to YEAR. This is more efficient than method **1** or method **2**.

Reference (Position 29)

For a logical file, leave this position blank. All logical files automatically provide the reference capability for all specified fields. Any attributes that are not specified explicitly in the logical file are furnished from the corresponding field in the physical file record format.

For a physical file, specify R in this position to refer to the attributes of a previously defined named field (called the *referenced field*). You must specify the REF or the REFFLD keyword. The referenced field can be previously defined in either the physical file you are defining or a previously created database file. The field attributes referred to are the length, data type, and decimal positions of the field, as well as the ALIAS, COLHDG, DATFMT, DATSEP, FLTPCN, REFSHIFT, TEXT, TIMFMT, TIMSEP, VARLEN, editing, and validity checking keywords.

If R is not specified, you must specify the field attributes for this field.

Note: If the DATFMT keyword is overridden on a reference field to *ISO, *EUR, *USA, or *JIS, the DATSEP keyword is not referenced.

Position 29 must be blank at the file and record levels.

The referenced field name cannot be the same as the field you are defining if that field is in the file you are defining. If the names are the same, specify the name of the file defining the referenced field as a parameter value with the REF or REFFLD keyword. If the names are different, specify the name of the referenced field with

the REFFLD keyword. For more information, see "REF (Reference)—Physical Files Only" on page 2-68, "REFFLD (Referenced Field)—Physical Files Only" on page 2-70, and Appendix A, "When to Specify REF and REFFLD."

To override specific attributes of the referenced field, specify those attributes for the field you are defining. In addition:

- If you specify Edit Code (EDTCDE) or Edit Word (EDTWRD) on the field, no editing specifications are copied from the referenced field.
- If you specify CHECK (AB, ME, MF, M10, M10F, M11, M11F, VN, or VNE), CHKMSGID, COMP, RANGE, or VALUES on the field, no validity checking specifications are copied from the referenced field.
- If you specify data type, field length, or decimal positions for the field you are defining, then neither editing nor validity checking keywords are copied from the referenced field.

Note: After the physical file is created, the referenced file can be deleted or changed without affecting the field descriptions in the physical file. To incorporate changes made in the referenced file, delete and re-create the physical file.

Length (Positions 30 through 34)

For a physical file, use these positions to specify the field length for each named field (unless you copy it from a referenced field). Specify the number of digits for a numeric type field, or specify the number of characters for a character type field.

For a logical file, use these positions to specify the length of a logical field. Specify the length only to override or change the length of the corresponding field in the physical file on which this logical file is based. If you leave this position blank, the field you are defining has the same length as the corresponding field in the physical file(s) on which the logical file(s) is based. If the field in the physical file is variable length and you leave the length blank, the field is also variable length in the logical file. If you do specify a length, the field in the logical file is fixed length unless you also specify the VARLEN keyword.

If you specify length, it must be right-justified; leading zeros are optional.

Figure 2-16 shows correct and incorrect field length specifications for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5
00010A            FIELD1       7
     A
00020A            FIELD2         7
     A
00030A            FIELD3    R   +7
     A
```

Note: FIELD1 shows the field length specified incorrectly. FIELD2 and FIELD3 show the field length specified correctly.

Figure 2-16. Correct and Incorrect Length Specifications for Physical Files

Valid length specifications are:

Data Type	Valid Lengths
Character	1 through 32 766 characters

Data Type	Valid Lengths
Hexadecimal	1 through 32 766 bytes
Binary	1 through 9 digits
Zoned decimal	1 through 31 digits
Packed decimal	1 through 31 digits
Floating-point (single precision)	1 through 9 digits
Floating-point (double precision)	1 through 17 digits
Date	6, 8, or 10 characters
Time	8 characters
Timestamp	26 characters

The length for fields with data type L (date), T (time), or Z (timestamp) is determined by the system. You should not enter a field length in positions 30 through 34.

The field length for date and time includes the separator.

A timestamp has a fixed format that has the following form:

```
YYYY-MM-DD-hh.mm.ss.uuuuuu
```

Type in a maximum of 9 digits for single precision and 17 digits for double precision. The OS/400 program supports a floating-point accuracy of 7 digits for single precision and 15 digits for double precision.

The total number of bytes occupied by all the fields in a record must not exceed 32 766 (in storage). See Figure 2-17 for rules on determining the total length of the record format. The system determines the number of bytes actually occupied in storage as follows:

Data Type	Bytes Occupied in Storage
Character	Number of characters
Hexadecimal	Number of bytes
Binary	
1 through 4 digits	2 bytes
5 through 9 digits	4 bytes
Zoned decimal	Number of digits
Packed decimal	(Number of digits/2) + 1 (truncated if fractional)
Floating-point (single precision)	4 bytes
Floating-point (double precision)	8 bytes
Date	4 characters
Time	3 characters
Timestamp	10 characters

Note: The system performs arithmetic operations more efficiently for a packed decimal than for a zoned decimal data type.

Figure 2-17 describes the rules for determining total format length.

Figure 2-17. Rules for Determining Total Format Length

Situation	Action
Does the record format contain any variable-length fields?	1. Add an extra 24 bytes to the total format length. 2. Add an extra 2 bytes to the format length for each field that is variable length.
Does the record format contain any fields that allow the null value?	Divide the total number of fields in the format by 8, round up to the next highest whole byte, then add to format length.

To override the length of a referenced field (R in position 29) of a physical file or the length of the field in a logical file, either specify a new value or a change in length. To increase the length, specify +n, where n is the amount of increase. To decrease the length, specify –n, where n is the amount of decrease. For example, type +4 to indicate that a numeric field is to be 4 digits longer than the referenced field. See Figure 2-3 on page 2-4 for an example showing how to override the field length for a physical file. Figure 2-1 on page 2-2 shows how to change and override the field length for a logical file.

If the corresponding field in the physical file record format has a data type of binary with decimal positions greater than zero, the length cannot be overridden in the logical file. If the field you are describing is a concatenation of fields from the associated physical record format, you cannot specify the length in the logical file. The sum of the physical field lengths is calculated by the system.

If you specify a value in positions 30 through 34, your program sees the specified length. However, the length of the field in the corresponding physical file field does not change. This can cause data conversion errors. When attempting to add a member to a file or to open a member of a file, the OS/400 program may send a mapping error message. The OS/400 program may also send a mapping error message to your program in the following cases:

- When reading from a logical file that reduces the length specified in the corresponding physical file
- When writing to a logical file that increases the length specified in the corresponding physical file

For example, if the physical file field is defined as 4 characters long and the logical file field decreases the length to 2 characters, a value of ABCD in the physical file cannot be read by the program, although a value of AB can. In this case, the program can always write successfully. For character fields, the data is left-justified and filled with blanks in the physical file field. For numeric fields, the data is right-justified and filled with zeros in the physical file field.

Positions 30 through 34 are valid only for field specifications. You must leave these positions blank at the key field, select/omit field, join, record, and file level.

Note: High-level languages can impose restrictions on the field length. Any length restrictions should be observed for files used by these high-level languages.

Data Type (Position 35)

For a physical file, use this position to specify the data type of the field within the database.

Specify data type in a logical file only to override or change the data type of the corresponding field in the physical file on which this logical file is based. If you leave this position blank, the field you are defining has the same data type as the corresponding field in the physical file(s) on which the logical file(s) is based.

Valid data type entries are as follows:

Entry	Meaning
P	Packed decimal
S	Zoned decimal
B	Binary
F	Floating-point
A	Character
H	Hexadecimal
L	Date
T	Time
Z	Timestamp

Figure 2-3 on page 2-4 and Figure 2-1 on page 2-2 show how to code the data type.

For physical files, if you do not specify a data type or duplicate one from a referenced field, the OS/400 program assigns the following defaults:

- A (character) if the decimal positions 36 through 37 are blank.
- P (packed decimal) if the decimal positions 36 through 37 contain a number in the range 0 through 31.

Notes:

1. Specify 0 in position 37 to indicate an integer numeric field for packed decimal, zoned decimal, or binary fields.
2. Specify an F in position 35 for a single precision floating-point field. Use the FLTPCN keyword to specify double precision or to change the precision of an already specified floating-point field.
3. Specify an H (hexadecimal) in position 35 to indicate a field whose contents are not interpreted by the system. In most cases, hexadecimal fields are treated as character fields, except the contents of a hexadecimal field are not translated to any character set or code page.

The following table shows what types of data conversion are valid between physical and logical files:

Physical File Data Type	Logical File Data Type								
	Character	Hexadecimal	Zoned	Packed	Binary	Floating Point	Date	Time	Timestamp
Character	Valid	Valid	See Note 1	Not valid	Not valid	Not valid	Not valid	Not valid	Not valid
Hexadecimal	Valid	Valid	See Note 1	Not valid	Not valid	Not valid	Not valid	Not valid	Not valid
Zoned	See Note 1	See Note 1	Valid	Valid	See Note 2	Valid	Not valid	Not valid	Not valid
Packed	Not valid	Not valid	Valid	Valid	See Note 2	Valid	Not valid	Not valid	Not valid
Binary	Not valid	Not valid	See Note 2	See Note 2	See Note 3	See Note 2	Not valid	Not valid	Not valid
Floating Point	Not valid	Not valid	Valid	Valid	See Note 2	Valid	Not valid	Not valid	Not valid
Date	Not valid	Not valid	See Note 4	Not valid	Not valid	Not valid	Valid	Not valid	Not valid
Time	Not valid	Not valid	See Note 4	Not valid	Not valid	Not valid	Not valid	Valid	Not valid
Timestamp	Not valid	Not valid	Not valid	Not valid	Not valid	Not valid	See Note 5	See Note 5	Valid

Notes:

1. Valid only if the number of characters (or bytes) equals the number of digits and the character (or hexadecimal) field is not defined as a variable-length field.
2. Valid only if the binary field has a decimal precision of zero.
3. Valid only if both fields have the same decimal precision.
4. The system generates the field length for you so do not enter a length in columns 30 through 34. The length does not include the separator character.
5. Valid only if the field is input only.

Converting One Numeric Data Type to Another

Any conversion of data types from the physical file record format is permitted within the numeric types. For example, a binary field in the physical file can be converted to zoned decimal in the logical file.

Converting between Zoned Decimal and Character or Hexadecimal

You can convert zoned decimal fields to character or hexadecimal fields and the converse, provided that the field lengths are the same. The data type of the field in your program is the data type specified in the logical file. No error occurs in an I/O operation if the data passed contains only numeric characters (0 through 9). However, your program cannot send an I/O operation that attempts to pass characters other than 0 through 9 from a character or hexadecimal field to a zoned decimal field. The OS/400 program sends a message and the I/O operation cannot be completed.

For example, suppose a field is zoned decimal in the physical file. If you specify character type (A) for presentation to your programs, you must ensure that the field contains only numeric characters (0 through 9) when it is returned through the logical file to the physical file.

In another example, suppose a field is a character field in the physical file. If you specify the field as a zoned decimal field and as a key field in the logical file, you

cannot create the logical file unless all records in the physical file contain only numeric characters (0 through 9).

Converting from Floating Point to Packed Decimal, Zoned Decimal, or Binary

If you are converting a floating-point field (in a physical file) to a packed decimal, zoned decimal, or binary field (in a logical file), you must explicitly specify the length and decimal positions. When converting floating-point data to fixed-point format make sure the values you specify for length and decimal positions are large enough to accommodate the data. Physical file length and decimal positions are presentation values only and do not indicate the magnitude of the number.

Converting Data Types When Concatenating Fields

If the field you are defining is a concatenation of fields from the associated physical file (specified by the CONCAT keyword), you cannot specify the data type. The OS/400 program assigns the data type based on the data types of the fields that are being concatenated. The general rules are:

- If the concatenation contains one or more hexadecimal (H) fields, the resulting data type is hexadecimal (H).
- If the concatenation contains one or more character (A) fields, but no hexadecimal fields, the resulting data type is character (A).
- If the concatenation contains only numeric (S, P, B) fields, the resulting data type is zoned decimal (S).

Converting Data Types When Substringing Fields

If the field you are defining is a substring of a field (specified by the SST keyword) from the logical file or the associated physical file, you must specify the data type as character, and the original field must be character, hexadecimal, or zoned (A, H, or S).

Decimal Positions (Positions 36 and 37)

For a physical file, use these positions to specify the decimal placement within a packed decimal, zoned decimal, binary, or floating-point field. Specify a decimal number from 0 through 31 for the number of decimal positions to the right of the decimal point. (The number must not be greater than the number of digits specified in the field length.) Figure 2-3 on page 2-4 shows how to code the decimal positions field.

The data is actually stored in the system without a decimal point. The decimal point is only implied. For example, the value stored for 1.23 is 123. This is what appears in display or printer files if editing is not specified.

To override the position of a referenced field (R in position 29), either specify a new value or a change in position. To increase the position, specify +n, where n is the amount of increase. To decrease the position, specify –n, where n is the amount of decrease. For example, an entry of +4 indicates there are 4 more digits to the right of the decimal point than were in the referenced field. An error message is sent if the number of decimal positions is greater than the maximum allowed.

For logical files, specify decimal positions only to override or change the decimal positions of the corresponding field in the physical file on which this logical file is based. If you leave these positions blank, the field you are defining has the same

decimal positions as the corresponding field in the physical file on which this logical file is based.

To override or change the placement of the decimal point within a packed decimal or zoned decimal field, specify a number from 0 through 31 to indicate the number of decimal positions to the right of the decimal point. The number here must not be greater than the number of digits specified in the field length. You cannot override or change decimals when the corresponding field in the physical file is binary (data type B) and contains decimal positions greater than zero. When the logical file field is binary and the corresponding field in the physical file is not binary (B specified in position 35 in the logical file), the decimal positions must be zero for the binary field.

You can override the position of the field by specifying a new value or by specifying an increase or decrease in position. To increase the position, specify +n, where n is the amount of increase. To decrease the position, specify –n, where n is the amount of decrease. For example, an entry of +4 indicates there are 4 more digits to the right of the decimal point than were in the referenced field.

If you specify a value in positions 36 through 37 and your program writes or retrieves data through the logical file field to the physical file field, the OS/400 program aligns the data on the decimal point. Depending on the case, this can cause the decimal values to be truncated, or it can cause a data conversion error. Decimal values are truncated in the following cases:

- When reading from a logical file that reduces the number of decimal positions specified in the physical file
- When writing to a logical file that increases the number of decimal positions specified in the physical file

For example, if the physical file field is defined as 4 digits long with 2 decimal positions, and the logical file field decreases the decimal positions to 0 decimal positions, a value of 0.20 in the physical file becomes a value of 0 in the logical file, and a value of 2.52 in the physical file becomes a value of 2 in the logical file.

When decimal values are truncated, the left side of the field is filled with zeros.

A data conversion error can occur in the following cases:

- When writing to a logical file that reduces the number of decimal positions specified in the physical file
- When reading from a logical file that increases the number of decimal positions specified in the physical file

The data conversion error occurs because too many digits would be moved into the space available to the left of the decimal point. For example, if, as in the previous example, the physical file field is defined as 4 digits long with 2 decimal positions and the logical file field decreases the decimal positions to 0 decimal positions, a value of 3322 written to the logical file cannot fit in the physical file because only 2 digits are allowed left of the decimal point in the physical file.

To avoid data conversion errors, increase or decrease the length (positions 30 through 34) of the logical file field by the same amount that you increase or decrease the decimal positions.

If you specify the CONCAT keyword for the field you are defining, you cannot specify decimal positions. A field in the physical file that contains decimal positions cannot be included in a concatenated field.

Note: High-level languages can impose specific length and value restrictions on the decimal positions. Observe these restrictions for files used by those high-level languages.

Usage (Position 38)

Use this field to specify that a named field is to be an input-only, both (both input and output are allowed), or neither field.

For physical files, you can specify the following entries:

Entry	Meaning
Blank	Defaults to B (both input and output allowed)
B	Both input and output allowed

Because the default is the same as the only value, you do not need to make an entry in this field.

Entries in position 38 are not referred to by the REF or REFFLD keywords. Therefore, a B in position 38 for a field in a physical file has no effect when that field is referred to in a display file.

The valid entries for logical files are described as follows:

Blank (Default)

If position 38 is blank, the following occurs:

- For simple and multiple format logical files (PFILE specified at the record level), the field is a both (B) field.
- For join logical files (JFILE specified at the record level), the field is an input-only (I) field.

B (Both)

If position 38 is B, the field is a *both* field and can be used for both input and output operations. That is, your program can read data from the field and write data to the field. Both fields are not valid for join logical files, because join logical files are read-only files.

I (Input-Only)

If position 38 is I, the field is an *input-only* field and can be used for input operations only. That is, your program can read data from the field, but cannot change the field. Typical cases of input-only fields are key fields (to reduce maintenance of access paths), sensitive fields that a user can see but not change (such as, in employee records, salary), and fields for which the SST or TRNTBL keyword is specified.

If your program performs a change to a record format in which you have specified input-only fields, the input-only fields are not updated and no message is sent. If your program performs an output operation to a record format in which you have specified input-only fields, the input-only fields take default values (see "DFT (Default)—Physical Files Only" on page 2-45).

Input-only fields are not valid in physical files.

N (Neither)

If position 38 is N, the field is a *neither* field (neither input nor output) and is valid only for join logical files. A neither field can be used as a join field in a join logical file, but your program cannot read a neither field.

Use neither fields when the attributes of join fields in the physical files do not match. In this case, one or both join fields must be redefined. However, you may not want to include the redefined fields in the record format (that is, you may not want the application program to see the redefined fields). Therefore, code the redefined join fields as N and they do not appear in the record format.

A field with N in position 38 does not appear in the buffer used by your program. However, the field description is displayed with the Display File Field Description (DSPFFD) command.

Neither fields cannot be used as select/omit or key fields.

Entries in position 38 are not referred to using the REF or REFFLD keywords. Therefore, a B or an I in position 38 for a field in a logical file has no effect when that field is referred to in a display file.

Location (Positions 39 through 44)

These positions do not apply to physical or logical files. Leave these positions blank unless you use them for comment text.

Keyword Entries (Positions 45 through 80)

This section contains keyword entries valid for describing physical and logical files. They are typed in positions 45 through 80 (functions). See "Syntax Rules" on page 1-4 for a discussion of the general rules for specifying keywords.

The following keywords are valid for both physical and logical files (except where noted):

ABSVAL
ALIAS
ALL (logical files only)
ALTSEQ
ALWNULL (physical files only)
CCSID (physical files only)
CHECK
CHKMSGID
CMP
COLHDG
COMP
CONCAT (logical files only)
DATFMT
DATSEP
DESCEND
DFT (physical files only)
DIGIT
DYNSLT (logical files only)
EDTCDE
EDTWRD
FCFO
FIFO
FLTPCN
FORMAT
LIFO
NOALTSEQ
RANGE
REF (physical files only)
REFFLD (physical files only)
REFSHIFT
RENAME (logical files only)
SIGNED
SST (logical files only)
TEXT
TIMFMT
TIMSEP
TRNTBL (logical files only)
UNIQUE
UNSIGNED
VALUES
VARLEN
ZONE

The following keywords are valid only for simple and multiple format logical files:

PFILE
REFACCPTH

The following keywords are valid only for join logical files:

JDFTVAL	JFILE	JOIN
JDUPSEQ	JFLD	JREF

When you use DDS to describe a source file (usually created without DDS, using the CRTSRCPF command) or when a logical file is based on a physical file to be used as a source file, you cannot use the following keywords:

ABSVAL	NOALTSEQ
ALTSEQ	SIGNED
DESCEND	UNIQUE
FCFO	VARLEN
FIFO	ZONE
LIFO	

ABSVAL (Absolute Value)

Use this key field level keyword to direct the OS/400 program to ignore the sign of the field when it sequences the values associated with this numeric field.

This keyword has no parameters.

The following example shows six records with a zoned decimal key field:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
1	98	F9F8
2	00	F0F0
3	98-	F9D8
4	97	F9F7
5	20	F2F0
6	99	F9F9

If you do not specify any sequencing keywords or the ALTSEQ keyword, the default sequencing for the key field is the SIGNED attribute. In this case, the records are sequenced in the following order:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
3	98-	F9D8
2	00	F0F0
5	20	F2F0
4	97	F9F7
1	98	F9F8
6	99	F9F9

If you specify the ABSVAL keyword, the absolute value of the negative field is used, and the resulting sequence is:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
2	00	F0D0
5	20	F2F0
4	97	F9F7

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
1	98	F9F8
3	98-	F9D8
6	99	F9F9

The ABSVAL keyword is not valid for a character, date, time, timestamp, and hexadecimal data type field. You cannot use this keyword with the DIGIT, SIGNED, UNSIGNED, or ZONE keywords.

ABSVAL (a key field-level keyword) causes ALTSEQ (a file-level keyword) to be ignored. If you specify ABSVAL for a key field, NOALTSEQ is in effect for that key field, even if ALTSEQ was specified at the file level. This occurs whether or not the NOALTSEQ keyword is specified.

Figure 2-18 shows how to specify the ABSVAL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            ORDAMT         5  0
00020A          K ORDAMT                    ABSVAL
     A
```

Figure 2-18. Specifying the ABSVAL Keyword

ALIAS (Alternative Name)

Use this field-level keyword to specify an alternative name for a field. When the program is compiled, the alternative name is brought into the program instead of the DDS field name. The high-level language compiler in use determines if the ALIAS name is used. Refer to the appropriate high-level language reference manual for information about ALIAS support for that language.

The format of the keyword is:

```
ALIAS(alternative-name)
```

Refer to "Syntax Rules" on page 1-4 for ALIAS naming conventions.

The alternative name must be different from all other alternative names and from all DDS field names in the record format. If a duplicate is found, an error message appears on the field name or alternative name.

An alternative name cannot be used within DDS or any other OS/400 function (for example, as a key field name, as the field name specified for the REFFLD keyword, or as a field name used in the Copy File (CPYF) command).

When you refer to a field that has the ALIAS keyword, the ALIAS keyword is copied in unless the ALIAS keyword is explicitly specified on the referencing field.

Figure 2-19 shows how to specify the ALIAS keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            FIELDA        25A         ALIAS(CUSTOMERNAME)
     A
```

Figure 2-19. Specifying the ALIAS Keyword

In Figure 2-19, the alternative name for FIELDA is CUSTOMERNAME.

ALL (All)—Logical Files Only

Use this select/omit field-level keyword to specify the action to be taken after all other select/omit specifications have been processed for this logical file. Specify ALL with S in position 17 to direct the OS/400 program to select any records that do not meet any of the other select/omit rules. Specify O in position 17 to direct the OS/400 program to omit any records that do not meet any of the other select/omit rules. If specified, ALL must follow the other select/omit keywords. You cannot specify a field name with the ALL keyword.

This keyword has no parameters.

If you do not specify the ALL keyword, the default action taken is the opposite of the last select/omit specification you made for the file. If the last specification was a select, the default is to omit all. If the last specification was an omit, the default is to select all.

Figure 2-20 shows how to specify the ALL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          S ACT                       COMP(EQ 3000)
00020A          S ACT                       COMP(GT 3100)
00030A          O AMT                       COMP(LT 0)
00040A          O                           ALL
     A
```

Figure 2-20. Specifying the ALL Keyword

ALTSEQ (Alternative Collating Sequence)

Use this file-level keyword to direct the OS/400 program to use an alternative collating sequence table when sequencing the records of a file member for retrieval, if you specified a key for this file.

The format of the keyword is:

```
ALTSEQ([library-name/]table-name)
```

The name of the alternative collating sequence table is a required parameter value. The library-name is optional. If you do not specify the library-name, the OS/400 program uses the library list (*LIBL) at file creation time.

The ALTSEQ keyword is not valid under the following conditions:

- When you specify FILETYPE(*SRC) on the Create Physical File (CRTPF) or Create Logical File (CRTLF) commands. See the *Database Guide* for information about the FILETYPE(*SRC) parameter value.
- When key fields have a data type of packed decimal, binary, or floating-point.
- When key fields are specified with ABSVAL or SIGNED. For those fields, NOALTSEQ (a key field-level keyword) is assumed and does not need to be specified. You can specify NOALTSEQ for any field in a composite key that does not require the alternative sequence.

- When you specify a value other than *SRC on the SRTSEQ parameter on the Create Physical File (CRTPF) or Create Logical File (CRTLF) command.

The ALTSEQ keyword cannot be specified with the REFACCPTH keyword.

You must have use authority to the alternative collating sequence table. The alternative collating sequence table is created using the Create Table (CRTTBL) command.

ALTSEQ causes zoned key fields to default to unsigned sequence. You can override the default by specifying the SIGNED keyword for individual key fields.

Figure 2-21 shows how to specify the ALTSEQ keyword for a logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     ALTSEQ(TABLELIB/TABLE1)
00020A          R RECORD1                  PFILE (PF1)
00030A            :
00040A            :
00050A            :
00060A            NAME          20
00070A            :
00080A            :
00090A          K NAME
     A
```

Figure 2-21. Specifying the ALTSEQ Keyword

Records with format RECORD1 are sequenced by key NAME according to the alternative collating sequence table (TABLE1 in library TABLELIB).

ALWNULL (Allow Null Value)—Physical Files Only

Use this field-level keyword to define this field to allow the null value.

This keyword has no parameters.

When you specify the ALWNULL keyword, the maximum length you can specify in positions 30 to 34 is 32 765 bytes (32 739 if the field is also variable length).

For physical files, when you specify the DATFMT keyword with values of *JOB, *MDY, *DMY, *YMD, or *JUL and the field allows null value, you must specify a valid date on the DFT keyword for this field.

Figure 2-22 shows how to specify the ALWNULL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A            FIELD1        75A         ALWNULL
00030A            FIELD2       100A
00040A            FIELD3          L         ALWNULL
00050A                                      DATFMT(*MDY)
00060A                                      DFT('12/25/93')
     A
```

Figure 2-22. Specifying the ALWNULL Keyword

FIELD1 is defined to allow the null value. The default value of FIELD1 is the null value. FIELD2 is defined to not allow the null value. The default value of FIELD2 is blanks.

CCSID (Coded Character Set Identifier)—Physical Files Only

Use this file- or field-level keyword to specify a coded character set identifier for character fields.

The format of the keyword is:

```
CCSID(value)
```

The value is a number up to 5 digits long that identifies a specific set of encoding scheme identifiers, character set identifiers, code page identifiers, and other relevant information that uniquely identifies the coded graphic character representation used for the data in the field.

When specified at the file level, the CCSID keyword applies to each character field in the file except those character fields that also have the CCSID keyword specified.

If the CCSID keyword is not specified at the file level and not all character fields have the CCSID keyword specified, then the fields are assigned the job's default CCSID when the file is created.

For a list of the valid CCSIDs for the AS/400 system, see the *National Language Support Planning Guide*.

Figure 2-23 shows how to specify the CCSID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        CCSID(37)
00020A          R RECORD1
00030A            FIELD1         75A          CCSID(500)
00040A            FIELD2        150A
     A
```

Figure 2-23. Specifying the CCSID Keyword

FIELD1 is assigned a CCSID value of 500. FIELD2 is assigned a CCSID value of 37.

CHECK (Check)

Use this field-level keyword to specify validity checking in display files.

The format of the keyword is:

```
CHECK(edit-check-code [. . .])
```

CHECK does not affect the physical or logical file being defined. When you define an input-capable field in a display file, refer to the field you are now defining by specifying R in position 29 and using the REF or REFFLD keyword. At display file creation, the OS/400 program copies the CHECK keyword and other field attributes from the field in the physical or logical file into the field in the display file. You can override the CHECK keyword (as well as all other validity-checking keywords and

the CHKMSGID keyword) by specifying any validity checking keyword for the field in the display file. See "Reference (Position 29)" on page 3-8 for details.

The rules for specifying this keyword in a physical or logical file are similar to those for a display file. However, only the following codes are allowed in physical or logical files:

Code	Meaning
AB	Allow blank
ME	Mandatory enter
MF	Mandatory fill
M10	IBM* Modulus 10 self-check algorithm
M10F	IBM Modulus 10 self-check algorithm
M11	IBM Modulus 11 self-check algorithm
M11F	IBM Modulus 11 self-check algorithm
VN	Validate name
VNE	Validate name extended

You cannot specify the CHECK(AB), CHECK(VN), CHECK(VNE), CHECK(M10), CHECK(M11), CHECK(M10F), or CHECK(M11F) keywords on a floating-point field (F in position 35). You cannot specify the CHECK keyword on a hexadecimal field (H in position 35). Do not specify the CHECK keyword on a date, time, or timestamp field (L, T, or Z in position 35).

See "CHECK (Check)" on page 3-52 for more information and an example showing how to specify the keyword.

CHKMSGID (Check Message Identifier)

Use this field-level keyword to identify an error message that is associated with validity checking keywords. If the CHKMSGID keyword is not specified, a system-supplied message is used. If the CHKMSGID keyword is specified and the field you are now defining is referred to later during display file creation, the validity checking information and the CHKMSGID keyword are copied into the display file. If a validity checking error is found while checking input from the screen, the error message specified on the CHKMSGID keyword is displayed on the message line.

CHKMSGID does not affect the physical or logical file you are defining.

The format of the keyword is:

```
CHKMSGID(message-id [library/]message-file [message-data-field])
```

If the message-data-field parameter is specified, the field it identifies does not need to be defined in the physical or logical file. However, if the field containing the CHKMSGID keyword is referred to during display file creation, the message data field must be defined in the display file (in the same record format as the field with the CHKMSGID keyword).

CHKMSGID is allowed only on fields that also contain a VALUES, RANGE, CMP, COMP, CHECK(M10), CHECK(M11), CHECK(VN), or CHECK(VNE) keyword.

See "CHKMSGID (Check Message Identifier)" on page 3-64 for more information and an example showing how to specify the keyword.

CMP (Comparison)

This keyword is equivalent to the COMP keyword.

The format of the keyword is:

```
CMP(relational-operator value)
```

The COMP keyword is preferred. See "COMP (Comparison)" on page 2-38 for an explanation of how to use these keywords.

COLHDG (Column Heading)

Use this field-level keyword to specify column headings used as a label for this field by text management, the query utility, the data file utility (DFU), and the screen design aid (SDA).

The format of the keyword is:

```
COLHDG('line-1' ['line-2' ['line-3']])
```

A maximum of three lines of 20 characters each is allowed. Each line of the column heading must be enclosed in apostrophes. Use double apostrophes (' ') to specify apostrophes within column headings. Use one or more blanks to separate the first column heading line from the second and the second from the third.

For a physical file, if you do not specify COLHDG and it is not retrieved from a referenced field, the field name is used. If you do not specify COLHDG for a logical file, the column heading from the physical file is used, except when the field is a concatenation of fields; in this case, the default is the field name.

If you specify COLHDG but do not specify TEXT, 50 positions of column heading information are used as text. For example, COLHDG('Order' 'Date') is equivalent to TEXT('Order Date').

Figure 2-24 shows how to specify the COLHDG keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00150A            ORDDAT          5  0       COLHDG('Order' 'Date')
00160A            NAME           20          COLHDG('Customer''s Name')
00170A            CITY           20          COLHDG('Customer' 'City' 'Field')
     A
```

Figure 2-24. Specifying the COLHDG Keyword

Decimal positions or data type must be specified for ORDDAT since Order Date is a numeric field (denoted by NNNNN below).

The following display illustrates how the column headings can appear when running text management, query, DFU, or SDA.

```
                                        Customer
Order                                   City
Date       Customer's Name              Field

NNNNN      XXXXXXXXXXXXXXXXXXX          XXXXXXXXXXXXXXXXXXX
```

COMP (Comparison)

Use this field-level keyword to specify validity checking for the field you are defining when it is referred to later during display file creation. For logical files, you can also specify this keyword at the select/omit-field level. COMP is equivalent to CMP.

The format of the keyword is:

```
COMP(relational-operator value)
```

At the select/omit-field level, the format of the keyword is:

```
COMP(relational-operator field-name)
```

Valid relational operators are:

Relational Operator	Meaning
EQ	Equal to
NE	Not equal to
LT	Less than
NL	Not less than
GT	Greater than
NG	Not greater than
LE	Less than or equal to
GE	Greater than or equal to

Specify the value parameter at either the field level or the select/omit field level. Specify the field name parameter only at the select/omit field level.

Specifying COMP at the Field Level

At the field level, COMP does not affect the physical or logical file you are describing. However, when you describe an input-capable field in a display file, you can refer to the field you are now describing by specifying R in position 29 and the REF or REFFLD keyword. During display file creation, the OS/400 program copies the COMP keyword and other field attributes from the field in the logical file into the field in the display file. You can override the COMP keyword (as well as all other validity-checking keywords and the CHKMSGID keyword) by specifying any validity checking keyword for the field in the display file. See "Reference (Position 29)" on page 3-8 for details.

You cannot specify a field name as a parameter value for a field-level COMP keyword.

You cannot specify *NULL as a parameter value for a field level COMP keyword.

You cannot specify the COMP keyword on a floating-point field (F in position 35) or a hexadecimal field (H in position 35). Do not specify the COMP keyword on a date, time, or timestamp field (L, T, or Z in position 35).

The rules for specifying this keyword in a physical or logical file are the same as for a display file. See "COMP (Comparison)" on page 3-76 for information on how to specify this keyword.

Defining a Numeric Field: When a work station user types in data, the OS/400 program aligns the characters typed in according to the number of decimal positions in the field. Leading and trailing blanks are filled with zeros when the field is passed to your program. If you do not type a decimal character, the OS/400 program places a decimal character to the right of the farthest right character typed. For example, for a numeric field with a length of 5 (specified in position 34) and 2 decimal positions (specified in position 37), 1.2 is interpreted as 001.20, and 100 is interpreted as 100.00.

Specifying COMP at the Select/Omit-Field Level

At the select/omit-field level, you can specify a field name, a value, or *NULL for the parameter.

If you specify a value, the following rules apply:

- If you are defining a character field, specify a character constant or a hexadecimal character string.

  Specify character strings with apostrophes (see Figure 2-25 on page 2-40).

  Specify hexadecimal character strings as an X followed by a combination of the digits 0 through 9 and the letters A through F, enclosed in apostrophes. The number of hexadecimal digits in apostrophes must be exactly twice the specified length of the field (see Figure 2-26 on page 2-40).

- If you are defining a numeric field, specify a numeric string (digits 0 through 9 specified without apostrophes) as shown in Figure 2-26 on page 2-40.

- If you are defining a date field, specify a valid date in the same format specified on the DATFMT keyword and use the same separator as specified on the DATSEP keyword. For example, `COMP(EQ '12/15/91')` is the default value if *MDY is specified for DATFMT and '/' is specified for DATSEP.

- If you are defining a time field, specify a valid time in the same format specified on the TIMFMT keyword and use the same separator as specified on the TIMSEP keyword. For example, `COMP(EQ '11.00.00')` is the default value if *ISO is specified for TIMFMT. The default separator for *ISO is a period (.).

- If you are defining a timestamp field, you must specify the default value in the following format:

  ```
  COMP(EQ 'YYYY-MM-DD-HH.MM.SS.UUUUUU')
  ```

If you specify *NULL, the relational operator must be EQ or NE.

COMP selects or omits records retrieved from the physical file on which this logical file is based when your program sends an input operation to the record format you are defining. The OS/400 program selects or omits records as a result of testing the value of the select/omit fields against the value you specify, the value of the field whose name you specify, or the null value (if *NULL was specified).

Figure 2-25 shows how to specify the COMP keyword for character and numeric strings.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD                    PFILE(PF1)
00020A
00030A            FIELDA         1  0       COMP(NE 0)      1
C0040A            FIELDB         1          COMP(NE 'A')    1
00050A            FIELDC
00060A            FIELDD
00070A            FIELDE
00080A          K FIELDB
00090A          S FIELDC                    COMP(EQ FIELDD)   2
00100A          S FIELDA                    COMP(NE 0)        2
00110A          S FIELDE                    COMP(NE *NULL)    2
00120A          O FIELDB                    COMP(GE 'A')      2
     A
```

Figure 2-25. Specifying the COMP Keyword (Example 1)

1 COMP is specified for FIELDA and FIELDB as a validity checking keyword for display files that refer to FIELDA and FIELDB.

2 COMP is specified as a select/omit keyword for FIELDC, FIELDA, FIELDB, and FIELDE. Records from the physical file PF1 are retrieved through this logical file record format depending on the following comparisons:

- FIELDC: Records are selected when FIELDC equals FIELDD.
- FIELDA: Records not meeting FIELDC test are selected only when FIELDA is not equal to zero.
- FIELDE: Records not meeting FIELDA test are selected only when FIELDE is not the null value.

Figure 2-26 specifies the COMP keyword using a hexadecimal character string.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD1                      PFILE(PF1)
00020A            CODEA
00030A            FLD1
00040A            FLD2
00050A          K FLD1
00060A          S CODEA                     COMP(EQ X'51')
     A
```

Figure 2-26. Specifying the COMP Keyword (Example 2)

COMP is specified as a select/omit keyword for CODEA (which is a 1-byte field). Records from physical file PF1 are retrieved through this record format only if the value of field CODEA is hex 51.

CONCAT (Concatenate)—Logical Files Only

Use this field-level keyword when you want to combine two or more fields from the physical file record format into one field in the logical file record format you are defining. The name of this concatenated field must appear in positions 19 through 28.

The format of the keyword is:

```
CONCAT(field-1 field-2...)
```

Specify the physical file field names in the order in which you want them to be concatenated, and separate them by blanks.

If the same physical field is specified more than once in a record format in the logical file (that is, by using either RENAME or CONCAT), the sequence in which the fields are specified in the logical file is the sequence in which the data is moved to the physical file on an update or insert operation. Thus, the value in the last occurrence of the physical field is the value that is put in the physical record and is the value that is used for all keys built over that physical field. All previous values of the same field are ignored.

If you want to use a field defined using the CONCAT keyword or a field specified as a parameter value on the CONCAT keyword as a key field, see "Key Field Name" on page 2-8.

You cannot include a field containing decimal positions other than zero in a concatenated field. You can include a field having decimal positions of zero in which case the field is treated as an integer field.

The OS/400 program assigns the length of the concatenated field as the sum of the lengths (digits and characters) of the fields included in the concatenation.

The OS/400 program assigns the field to be fixed length or variable length based on the fields that are concatenated. The general rules are:

- Concatenation of a variable-length field to either a fixed-length field or another variable-length field results in a variable-length field.
- Concatenation of a fixed-length field to a fixed-length field results in a fixed-length field unless the VARLEN keyword is also specified on the same field as the CONCAT keyword.

  Note: If the result of the concatenation is a variable-length field or a field that allows the null value, the CONCAT field must be input only (I in position 38).

The OS/400 program assigns the data type based on the data types of the fields that are being concatenated. The general rules are:

- If the concatenation contains one or more hexadecimal (H) fields, the resulting data type is hexadecimal (H).
- If the concatenation contains one or more character (A) fields, but no hexadecimal fields, the resulting data type is character (A).
- If the concatenation contains only numeric (S, P, B) fields, the resulting data type is zoned decimal (S).

When concatenating numeric fields, the sign of the farthest right field in the concatenation is used as the sign of the concatenated field. The signs of the other fields are ignored; however, they are present in the concatenated field. Therefore, if a negative value appears in a field other than the last, you must take appropriate action to delete the embedded signs (such as converting the concatenated field to packed decimal).

The maximum length of a concatenated field varies, depending on the data type of the concatenated field and the length of the fields being concatenated. If the concatenated field is zoned decimal (S), its total length cannot exceed 31 bytes. If the

field is character (A) or hexadecimal (H), its total length cannot exceed 32 766 bytes. If the concatenated field is a variable length field, its total length cannot exceed 32 740 (32 739 if the field also allows the null value).

You cannot include a floating-point, date, time, or timestamp field in a concatenated field.

In join logical files, the fields to be concatenated must be from the same physical file. The first field specified on the CONCAT keyword identifies which physical file is used. The first field must, therefore, be unique among the physical files the join logical file is based on, or you must also specify the JREF keyword to specify which physical file to use.

Figures 2-27 through 2-30 show how to specify the CONCAT keyword.

MTH, DAY, and YEAR are fields in the physical file that are concatenated into one field DATE in the logical file, as shown in Figure 2-27.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                 PFILE(PF1)
00020A            DATE                    CONCAT(MTH DAY YEAR)
     A
```

Figure 2-27. Specifying the CONCAT Keyword (Example 1)

In Figure 2-28, if the program changes DATE from 01 03 81 to 02 05 81, the value placed in the physical record does not change because the fields specified last are MTH (value 01), DAY (value 03), and YEAR (value 81). However, if MTH, DAY, and YEAR are changed to new values, the value of DATE in the physical record also changes.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD2                 PFILE(PF1)
00020A            DATE                    CONCAT(MTH DAY YEAR)
00030A            MTH
00040A            DAY
00050A            YEAR
     A
```

Figure 2-28. Specifying the CONCAT Keyword (Example 2)

In Figure 2-29, fields from the physical file are concatenated into more than one field in the logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD3                 PFILE(PF1)
00020A            DATE                    CONCAT(MTH DAY YEAR)
00030A            CMPDAT                  CONCAT(DAY MTH YEAR)
     A
```

Figure 2-29. Specifying the CONCAT Keyword (Example 3)

In Figure 2-30, if the fields from PF1 are:

- FIXED1 is a fixed length field.
- FIXED2 is a fixed length field.
- VARLEN1 is a variable length field.

The resulting fields are:

- FIELD1 is a variable length field.
- FIELD2 is a fixed length field.
- FIELD3 is a variable length field.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD4                   PFILE(PF1)
00020A            FIELD1                    CONCAT(FIXED1 VARLEN1)
00030A            FIELD2                    CONCAT(FIXED1 FIXED2)
00040A            FIELD3                    CONCAT(FIXED1 FIXED2)
00050A                                      VARLEN
     A
```

Figure 2-30. Specifying the CONCAT Keyword (Example 4)

DATFMT (Date Format)

Use this field-level keyword to specify the format of a date field. This keyword is valid only for date fields (data type L) or for logical file zoned fields (data type S) whose corresponding physical file fields are date fields (data type L).

The format of the keyword is:

```
DATFMT(date-format)
```

The date-format parameter specifies the format for the date. The following table describes the valid date formats and their default separator values.

Format Name	Date-Format Parameter	Date Format and Separator	Field Length	Example
Job Default	*JOB			
Month/Day/Year	*MDY	mm/dd/yy	8	06/21/90
Day/Month/Year	*DMY	dd/mm/yy	8	21/06/90
Year/Month/Day	*YMD	yy/mm/dd	8	90/06/21
Julian	*JUL	yy/ddd	6	90/172
International Standards Organization	*ISO	yyyy-mm-dd	10	1990-06-21
IBM USA Standard	*USA	mm/dd/yyyy	10	06/21/1990
IBM European Standard	*EUR	dd.mm.yyyy	10	21.06.1990
Japanese Industrial Standard Christian Era	*JIS	yyyy-mm-dd	10	1990-06-21

If you specify *JOB, the default is the job attribute.

For physical files, if you do not specify the DATFMT keyword, the default is *ISO.

For logical files, if you do not specify the DATFMT keyword, the default is the date format from the physical file.

For logical and physical files, if the DFT keyword is not specified, the default value is the current date.

If you specify the *ISO, *USA, *EUR, or *JIS value, you cannot specify the DATSEP keyword. These date formats have a fixed separator.

The DATFMT keyword overrides the job attribute for a date field. It does not change the system default.

For physical files, if you specify the *JOB, *MDY, *DMY, *YMD, or *JUL values and the field allows the null value, you must specify a valid date on the DFT keyword for this field.

Figure 2-31 shows how to specify the DATFMT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A
00020A          R RECORD
00030A            DATFLD1         L         DATFMT(*JUL)
00040A            DATFLD2         L         DATFMT(*EUR)
     A
```

Figure 2-31. Specifying the DATFMT Keyword

If the current date is June 21, 1990, the current system date format value is MDY, and the current system separator is /, DATFLD1 contains 90/172 (the 172nd day of the year 1990). DATFLD2 contains 21.06.1990.

DATSEP (Date Separator)

Use this field-level keyword to specify the separator character for a date field. This keyword is valid only for date fields (data type L).

The format of the keyword is:

```
DATSEP(*JOB | 'date-separator')
```

The date separator parameter specifies the separator character that appears between the year, month, and day. Valid values are a slash (/), dash (–), period (.), comma (,) or blank ( ). The parameter must be enclosed in apostrophes.

If you specify *JOB, the default is the job attribute.

For physical files, if you do not specify the DATSEP keyword, the default is the job attribute.

For logical files, if you do not specify the DATSEP keyword, the default is the date separator from the physical file. If you did not specify the DATSEP keyword for the physical file field (*ISO, *USA, *EUR, or *JIS was specified on the DATFMT keyword), the default for DATSEP is the job attribute.

If you specify the *ISO, *USA, *EUR, or *JIS date format value on the DATFMT keyword, you cannot specify the DATSEP keyword. These formats have a fixed date separator.

The DATSEP keyword overrides the job attribute. It does not change the system default.

Figure 2-32 on page 2-45 shows how to specify the DATSEP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A
00020A          R RECORD1
00030A            DATFLD2         L         DATFMT(*DMY) DATSEP('-')
00040A            DATFLD4         L         DATSEP(' ')
     A
```

Figure 2-32. Specifying the DATSEP Keyword

If the current date is June 21, 1990, the current system date format value is MDY, and the system date separator value is '/', DATFLD2 contains 21-06-90. DATFLD4 contains 06 21 90.

DESCEND (Descend)

Use this key field-level keyword to specify that the values of this character, hexadecimal, or numeric key field are retrieved in descending sequence. The default is ascending sequence. See "SIGNED (Signed)" on page 2-73 for an example of data sorted using the DESCEND keyword.

This keyword has no parameters.

Figure 2-33 shows how to specify the DESCEND keyword for a logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          K ITEM
00020A          K BALDUE                    DESCEND
     A
```

Figure 2-33. Specifying the DESCEND Keyword

DFT (Default)—Physical Files Only

Use this field-level keyword to specify a default value for a field.

The format of the keyword is:

```
DFT('value' | numeric-value | X'hexadecimal-value' | *NULL)
```

Without this keyword, character and hexadecimal fields default to blanks and numeric fields default to zeros. However, if you specify the ALWNULL keyword for the field, then the character, hexadecimal, and numeric fields default to the null value.

The following rules apply to the specified value:

- If the field being defined is a character field, specify a character constant, hexadecimal value, or *NULL. Specify character strings within apostrophes. If the field is variable length (VARLEN), then the length of the string must be less than or equal to the allocated length.

  Specify hexadecimal values as an X followed by a combination of the digits 0 through 9 and the letters A through F. Enclose the combination in apostrophes. The number of hexadecimal digits in apostrophes must be exactly twice the specified length of the field. If the field is variable length (VARLEN), then the number of hexadecimal digits in apostrophes must be exactly twice the allocated length.

- If the field being defined is a hexadecimal field, specify a character constant, hexadecimal value, or *NULL.

  Note: If a character constant is specified, the hexadecimal representation of the character constant is the default value.

  Specify character strings within apostrophes. If the field is variable length (VARLEN), then the length of the string must be less than or equal to the allocated length.

  Specify hexadecimal values as an X followed by a combination of the digits 0 through 9 and the letters A through F. Enclose the combination in apostrophes. The number of hexadecimal digits in apostrophes must be exactly twice the specified length of the field. If the field is variable length (VARLEN), then the number of hexadecimal digits in apostrophes must be exactly twice the allocated length.

- If you are defining a numeric field, specify a numeric value (digits 0 through 9 specified without apostrophes) or *NULL. For a value other than zero in positions 36 and 37, specify the decimal character with a numeric constant in the appropriate position in the DDS.
- If you specify *NULL, then you must also specify the ALWNULL keyword on the field.
- If you do not specify any value (DFT('')), this indicates a default of a 0 length string and is valid only when the field is variable length (the VARLEN keyword must also be specified).
- If you are defining a date field, specify a valid date in the same format specified on the DATFMT keyword and use the same separator as specified on the DATSEP keyword. For example, `DFT('12/15/91')` is the default value if *MDY is specified for DATFMT and '/' is specified for DATSEP.

  If the DFT keyword is not specified, the default value is the current date.
- If you are defining a time field, specify a valid time in the same format specified on the TIMFMT keyword and use the same separator as specified on the TIMSEP keyword. For example, `DFT('11.00.00')` is the default value if *ISO is specified for TIMFMT. The default separator for *ISO is a period (.).

  If the DFT keyword is not specified, the default value is the current time.
- If you are defining a timestamp field, you must specify the default value in the following format:

  ```
  DFT('YYYY-MM-DD-HH.MM.SS.UUUUUU')
  ```

  If the DFT keyword is not specified, the default value is the current time.

The value specified is assigned to the field in the following cases:

- When the program does an output operation to a logical file based on this physical file and the record format in the logical file does not name this field.
- When you use the Initialize Physical File Member (INZPFM) command for a member in this file.
- When you use the Copy File (CPYF) command with FMTOPT(*MAP) specified and a field in the to-file is not in the from-file.

The specified value is supplied to the program when the program does an input operation to a join logical file and all of the following are true:

- You specify the JDFTVAL keyword for the join logical file.
- The file being defined is specified as a secondary file in the join logical file.
- When the input operation occurs and the link to the secondary file produces no records.

This keyword does not affect the physical file on input operations.

Figure 2-34 shows how to specify the DFT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A            CHARFLD1      20A         DFT('Sample field')
00030A            CHARFLD2       5A         DFT(X'D985955185')
00040A            HEXFLD1        3H         DFT('ABC')
00050A            HEXFLD2        3H         DFT(X'C1C2C3')
00060A            NUMFLD1        5S 0       DFT(99999)
00070A            NUMFLD2        5S 2       DFT(999.99)
00080A            NUMFLD3        5S 2       DFT(999)
00090A            NUMFLD4        5S 2       DFT(*NULL)
00100A                                       ALWNULL
00110A            NUMFLD5        5S 2       DFT(999.99)
00120A                                       ALWNULL
00130A            DATFLD1         L         DATFMT(*MDY) DATSEP('-')
00140A                                       DFT('12-31-91')
00150A            TIMFLD1         T         DFT('11.15.00')
     A
```

Figure 2-34. Specifying the DFT Keyword

The default value for CHARFLD1 is 'Sample field'. The default value for CHARFLD2 is hex D985955185. The default value for HEXFLD1 is C1C2C3 (the hexadecimal representation of the character constant). The default value for HEXFLD2 is C1C2C3. The default value for NUMFLD1 is 99999 (no decimal character is required because the field has zero decimal positions). The default value for NUMFLD2 is 999.99. The default value for NUMFLD3 is 999 (no decimal character is required if you do not need to specify decimal values). The default value for NUMFLD4 is the null value (ALWNULL is a required keyword for the field if DFT(*NULL) is specified). The default value for NUMFLD5 is 999.99; the field also allows the null value. The default value for DATFLD1 is 12-31-91. The default value for TIMFLD1 is 11.15.00 (*ISO format).

DIGIT (Digit)

Use this key field-level keyword to specify that only the digit portion (farthest right 4 bits) of each byte of the key field is used when constructing a value associated with this key field. The zone portion is zero-filled.

This keyword has no parameters.

The DIGIT keyword is applied against the entire key field (not just a position within the field). It is valid only for character, hexadecimal, or zoned decimal type fields.

You cannot use this keyword with the ABSVAL, SIGNED, or ZONE keywords.

If you specify DIGIT for a key field, the value of the field is treated as a string of unsigned binary data, rather than signed data, which is the default for zoned decimal fields.

Figure 2-35 on page 2-48 shows how to specify the DIGIT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00040A          K ORDTYP                    DIGIT
     A
```

Figure 2-35. Specifying the DIGIT Keyword

If ORDTYP is a 3-byte field, the values of the field for three different records could be as follows:

Values	Hexadecimal	Digits Used for Key
C4J	C3F4D1	341
CMA	D3D4C1	341
3D1	F3C4F1	341

DYNSLT (Dynamic Select)—Logical Files Only

Use this file-level keyword to indicate that the selection and omission tests specified in the file (using select/omit specifications) are done at processing time. This keyword specifies dynamic select/omit rather than access path select/omit.

This keyword has no parameters.

As your program does input operations to a logical file with the DYNSLT keyword specified, all the records in the associated physical file are tested by the system to see if they satisfy the select/omit values. Only those records that satisfy the values are supplied to your program. The testing of each record can result in slower I/O performance, but may be more efficient than maintaining an access path for the file. This is particularly likely for files read only occasionally, especially when the physical files they are based on are updated frequently. Using dynamic select/omit is probably also more efficient for files with a high percentage of selected records.

In keyed sequence access files, an access path is created at file creation time and is maintained for the file according to the MAINT parameter on the Create Logical File (CRTLF) or Change Logical File (CHGLF) command. The DYNSLT keyword does not affect the maintenance of access paths for keyed sequence access files.

For all single-format logical files with a DYNSLT keyword, you do not need to specify key fields in order to specify select/omit fields. However, for all multiple-format logical files with a DYNSLT keyword, you do need to specify at least one key field. You can specify *NONE for this key field.

You must use the DYNSLT keyword when you want to select or omit fields and any of the following are true:

- The logical file has arrival sequence (no key fields are specified). See Figure 2-36 on page 2-49.
- The logical file is a join logical file with the JDFTVAL keyword specified.
- The logical file is a join logical file, select/omit fields come from more than one of the physical files the logical file is based on, and one of the following is true:
  - The select/omit fields are on the same select or omit statement. See Figure 2-38 on page 2-50.
  - The select/omit fields are on a mixture of select and omit statements. See Figure 2-39 on page 2-50.

- The select/omit fields are on select statements that are ORed together.
- The select/omit fields are on omit statements that are ANDed together.

You cannot specify the DYNSLT keyword with the REFACCPTH keyword.

For a join logical file, the select/omit fields can occur in any of the physical files specified on the JFILE keyword. Use the JREF keyword in join logical files to qualify the origin of the field and resolve any ambiguities.

The following examples show how to specify the DYNSLT keyword.

Figure 2-36 shows how to specify dynamic select with arrival sequence.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     DYNSLT
00020A          R RECORD1                  PFILE(PF1)
00030A            FLD1
00040A            FLD2
00050A          S FLD1                     COMP(GT 2)
```

Figure 2-36. Specifying the DYNSLT Keyword (Example 1)

The DYNSLT keyword is required because there are no key fields.

The logical file supplies records to your program in arrival sequence. Assume that physical file PF1 has the following records:

FLD1	**FLD2**
1	aaaa
2	dddd
3	jjjj
4	bbbb

As your program does input operations, the system tests the first two records according to the select/omit values, but does not supply them to your program. Your program only sees the last two records:

FLD1	**FLD2**
3	jjjj
4	bbbb

Figure 2-37 shows how to specify dynamic select with keyed sequence access path.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     DYNSLT
00020A          R RECORD1                  PFILE(PF1)
00030A            FLD1
00040A            FLD2
00050A          K FLD1
00060A          S FLD2                     COMP(GT 'bbbb')
     A
```

Figure 2-37. Specifying the DYNSLT Keyword (Example 2)

In Figure 2-37, the DYNSLT keyword is not required. The logical file supplies records to your program in keyed sequence. Assume that physical file PF1 has the following records:

FLD1	FLD2
1	aaaa
2	dddd
3	jjjj
4	bbbb

When your program requests a record, the system tests the value of FLD2 for that record according to the select/omit values. Your program only sees the following records:

FLD1	FLD2
2	dddd
3	jjjj

Figure 2-38 shows how to specify a join logical file with select/omit comparing fields from two physical files.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     DYNSLT
00020A          R RECORD1                  JFILE(PF1 PF2)
00030A          J                          JFLD(FLD1 FLD3)
00040A            FLD1                     JREF(PF1)
00050A            FLD2                     JREF(PF1)
00060A            FLD3                     JREF(PF2)
00070A            FLD4                     JREF(PF2)
00080A          S FLD1                     COMP(GT FLD4)
     A
```

Figure 2-38. Specifying the DYNSLT Keyword (Example 3)

FLD1 and FLD2 come from the primary file (PF1), and FLD3 and FLD4 come from the secondary file (PF2). The select specification compares FLD1 from the primary file with FLD4 from the secondary file. Therefore, the DYNSLT keyword is required.

Figure 2-39 shows how to specify a join logical file with select and omit using fields from more than one physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     DYNSLT
00020A          R JREC                     JFILE(PF1 PF2)
00030A          J                          JOIN(PF1 PF2)
00040A                                     JFLD(FLD1 FLD2)
00050A            FLD1                     JREF(PF1)
00060A            FLD2                     JREF(PF1)
00070A            FLD3                     JREF(PF2)
00080A          K FLD1
00090A          S FLD1                     COMP(GT 0)
00100A          O FLD3                     COMP(GT 4)
     A
```

Figure 2-39. Specifying the DYNSLT Keyword (Example 4)

FLD1 and FLD3 come from different physical files and are specified in a mixture of select and omit statements. Therefore, the DYNSLT keyword is required.

EDTCDE (Edit Code) and EDTWRD (Edit Word)

Use these field-level keywords to specify editing for the field you are defining when the field is referenced later during display or printer file creation. The EDTCDE and EDTWRD keywords do not affect the physical or logical file.

The format of the EDTCDE keyword is:

```
EDTCDE(edit-code [* | floating-currency-symbol])
```

The format of the EDTWRD keyword is:

```
EDTWRD('edit-word')
```

When defining an input-capable field in a display file, refer to the field you are now defining by specifying the letter R in position 29 and the REF or REFFLD keyword. At display file creation, the OS/400 program copies the EDTCDE or EDTWRD keyword and other field attributes from the field in the physical or logical file into the field in the display file. You can override the EDTCDE or EDTWRD keyword by specifying new editing keywords in the display or printer file. Specifying the DLTEDT keyword in the display or printer file deletes all editing for the field. See "Reference (Position 29)" on page 3-8 for details.

You cannot specify the EDTCDE or EDTWRD keyword on a floating-point field (F in position 35) or a hexadecimal field (H in position 35). Do not specify the EDTCDE or EDTWRD keywords on a date, time, or timestamp field (L, T, or Z in position 35).

The rules for specifying these keywords in a physical or logical file are the same as for a display file. For more information on specifying these keywords, see "EDTCDE (Edit Code)" on page 3-99 or "EDTWRD (Edit Word)" on page 3-105.

Figure 2-40 shows how to specify the EDTCDE and EDTWRD keywords for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD
     A
     A            PRICE          5  2       EDTCDE(J)
     A
     A            SALES          7  2       EDTCDE(K $)
     A
     A            SALARY         8  2       EDTCDE(1 *)
     A
     A            BALANCE        7  2       EDTWRD('$    0.  &CR')
     A
     A            DATE           6  0       EDTCDE(Y)
     A
```

Figure 2-40. Specifying the EDTCDE and EDTWRD Keywords

The fields PRICE, SALES, SALARY, and DATE have editing specified. No new editing needs be specified when they are referred to by a display or printer file. This standardizes the editing of these fields for applications that refer to these fields.

FCFO (First-Changed First-Out)

Use this file-level keyword to specify that if records with duplicate key values are retrieved from the same physical or logical file member, the record with the key value that was changed first is the first record retrieved. This is a first-changed first-out (FCFO) order.

This keyword has no parameters.

FCFO is not allowed with an FIFO, LIFO, UNIQUE, or REFACCPTH keyword.

If you do not specify FCFO, LIFO, FIFO, or UNIQUE, records with duplicate key values are retrieved in first-in first-out (FIFO), last-in first-out (LIFO), or first-changed first-out (FCFO) order, but the order in which they are retrieved is not guaranteed.

With the FCFO keyword, the records are ordered by when the record key value is changed. With the FIFO and LIFO keywords, the records are ordered by the relative record number.

At least one key field must be specified in the file containing the FCFO keyword. The FCFO keyword is not valid when you specify FILETYPE(*SRC) on the Create Physical File (CRTPF) or Create Logical File (CRTLF) command.

Figure 2-41 shows how to specify the FCFO keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      FCFO
00020A          R CUSREC                    TEXT('CUSTOMER RECORD')
00030A            CUSNAMEF      10A
00040A            CUSNAMEM       1A
00050A            CUSNAMEL      10A
00060A          K CUSNAMEL
     A
```

Figure 2-41. Specifying the FCFO Keyword

FIFO (First-In First-Out)

Use this file-level keyword to specify that if records with duplicate key values are retrieved from the same physical or logical file member, they are to be retrieved in a first-in first-out (FIFO) order.

This keyword has no parameters.

FIFO is not allowed with an FCFO, LIFO, UNIQUE, or REFACCPTH keyword.

If you do not specify FCFO, LIFO, FIFO, or UNIQUE, records with duplicate key values are retrieved in first-in first-out (FIFO), last-in first-out (LIFO), or first-changed first-out (FCFO) order, but the order in which they are retrieved is not guaranteed.

At least one key field must be specified in a file containing the FIFO keyword. The FIFO keyword is not valid when you specify FILETYPE(*SRC) on the Create Physical File (CRTPF) or Create Logical File (CRTLF) command.

Figure 2-42 on page 2-53 shows how to specify the FIFO keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        FIFO
00020A          R CUSREC                      TEXT('CUSTOMER RECORD')
00030A            CUSNAMEF      10A
00040A            CUSNAMEM       1A
00050A            CUSNAMEL      10A
00060A          K CUSNAMEL
     A
```

Figure 2-42. Specifying the FIFO Keyword

FLTPCN (Floating-Point Precision)

Use this field-level keyword to specify the precision of a floating-point field.

The format of the keyword is:

```
FLTPCN(*SINGLE | *DOUBLE)
```

where *SINGLE is single precision and *DOUBLE is double precision. This keyword is valid for floating-point fields only (data type F).

If you do not specify the FLTPCN keyword, the default is single precision. A single precision field can be up to 9 digits; a double precision field can be up to 17 digits. If you specify a field length greater than 9 (single precision) or 17 (double precision), an error message is sent and the file is not created.

Figure 2-43 shows how to specify the FLTPCN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00090A            FIELDA        17F 4       FLTPCN(*DOUBLE)
     A
```

Figure 2-43. Specifying the FLTPCN Keyword

FIELDA is a floating-point field with double precision.

FORMAT (Format)

Use this record-level keyword to specify that this record format is to share the field specifications for a previously defined record format. The name of the record format you are defining must be the name of the previously defined record format.

The format of the keyword is:

```
FORMAT([library-name/]database-file-name)
```

The database-file-name parameter is required. It is the name of the physical or logical file from which the previously defined record format is taken.

The library-name is optional. If you do not specify the library-name, the library list (*LIBL) in effect at file creation time is used.

If you specify the FORMAT keyword, you cannot specify field specifications for this record format. Specify key specifications and, if necessary, select/omit specifica-

tions if you want them to be in effect for this file. (They can be the same as or different from the previously defined record format.)

The FORMAT keyword is not valid in join logical files and you cannot specify a join logical file as the parameter value on the FORMAT keyword.

If the database file from which you are using the record format is deleted, the record format remains in existence as long as some file is using the record format. For example, RECORD in FILE2 uses the FORMAT keyword to share the specifications of RECORD in FILE1. Both files have been created. If FILE1 is deleted and then re-created with different DDS, RECORD still exists in FILE2. It can be referred to for the original record format by other files using the FORMAT keyword.

You cannot specify a distributed data management (DDM) file on this keyword.

Figure 2-44 shows how to specify the FORMAT keyword for a logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD                    PFILE(FILE2)
00020A                                      FORMAT(FILE1)
     A
```

Figure 2-44. Specifying the FORMAT Keyword

The record format for this logical file is the same as the previously specified record format in file FILE1. The name of this record format (RECORD) must be the same as the name of the record format in FILE1.

JDFTVAL (Join Default Values)—Join Logical Files Only

Use this file-level keyword in a join logical file so the system provides default values for fields when a join to a secondary file does not produce any records. JDFTVAL is valid only for join logical files.

This keyword has no parameters.

The default values for the system are blanks for character and hexadecimal fields and zeros for numeric fields. You can change the default for specific fields by specifying the DFT keyword for the fields in the physical file (see "DFT (Default)—Physical Files Only" on page 2-45).

If you specify JDFTVAL, your program retrieves records for which a secondary file does not have a corresponding record. If you do not specify JDFTVAL, a record in the primary file for which there is no corresponding record in a secondary file is skipped.

If you are joining three or more files, and you specify the JDFTVAL keyword for fields used as join fields, default values of fields missing in secondary files are used in the same way that a field value is used. For example, records are selected and omitted based on the default value. Also, if this field is used as a join field to join to other secondary files, records from the other secondary files are returned to your program based on the default value.

Figure 2-45 on page 2-55 shows how to specify the JDFTVAL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     JDFTVAL
00020A          R RECORD1                  JFILE(PF1 PF2)
00030A          J                          JOIN(PF1 PF2)
00040A                                     JFLD(NAME NAME)
00050A            NAME                     JREF(1)
00060A            ADDR
00070A            BAL
     A
```

Figure 2-45. Specifying the JDFTVAL Keyword

PF1 is the primary file and PF2 is a secondary file. Assume that PF1 and PF2 have the following records:

PF1		PF2	
NAME	**ADDR**	**NAME**	**BAL**
Anne	120 1st St.	Anne	5.00
Doug	40 Pillsbury	Doug	6.50
Mark	2 Lakeside Dr.	Sue	2.00
Sue	120 Broadway		

With JDFTVAL specified in the join logical file, the program reads the following records (shown in arrival sequence):

NAME	ADDR	BAL
Anne	120 1st St.	5.00
Doug	40 Pillsbury	6.50
Mark	2 Lakeside Dr.	0.00
Sue	120 Broadway	2.00

Without JDFTVAL specified in the join logical file, the program can read only three records (no record is found for Mark).

In this example, if you specified JREF(2) instead of JREF(1), the records returned to the program would be different, as follows:

NAME	ADDR	BAL
Anne	120 1st St.	5.00
Doug	40 Pillsbury	6.50
	2 Lakeside Dr.	0.00
Sue	120 Broadway	2.00

JDUPSEQ (Join Duplicate Sequence)—Join Logical Files Only

Use this join-level keyword to specify the order in which records with duplicate join fields are presented when your program reads a join logical file.

The format of the keyword is:

```
JDUPSEQ(sequencing-field-name [*DESCEND])
```

This keyword has no effect on the ordering of unique records. If you do not specify the keyword, the system does not guarantee the order in which records with duplicate join fields are presented.

If more than one JDUPSEQ keyword is specified in one join specification, the order in which you specify the JDUPSEQ keywords determines the order of pres-

entation of duplicate records. This is similar to specifying an additional key field, in that it determines the order in which records with duplicate keys are presented.

This keyword is valid only for join logical files.

In a single join specification, the total length of fields specified as *to* fields on the JFLD keyword and fields specified on the JDUPSEQ keyword cannot exceed 120 bytes.

The sequencing field name must be a field that (1) exists in the *to* file for this join specification and (2) has not been specified as a *to* field on the JFLD keyword for this join specification. The sequencing field name can be a concatenated field or a SST field. The sequencing field name need not be specified in the record format for the join logical file.

Optionally, you can specify *DESCEND to change the order in which duplicate records are presented. Without *DESCEND, duplicate records are presented in the following default sequences:

- Ascending signed order for a numeric sequencing field
- Ascending order for a character sequencing field

Figure 2-46 and Figure 2-47 show how to specify the JDUPSEQ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JREC                      JFILE(PF1 PF2)
00020A          J                           JOIN(PF1 PF2)
00030A                                      JFLD(NAME1 NAME2)
00040A                                      JDUPSEQ(PHONE)
00050A            NAME1
00060A            ADDR
00070A            PHONE
```

Figure 2-46. Specifying the JDUPSEQ Keyword (Example 1)

Figure 2-46, assumes that PF1 and PF2 have the following records:

PF1		PF2	
NAME1	**ADDR**	**NAME2**	**TELEPHONE**
Anne	120 1st St.	Anne	555-1111
Doug	40 Pillsbury	Anne	555-6666
Mark	2 Lakeside Dr.	Anne	555-2222
		Doug	555-5555

There are three records for Anne in PF2, showing three telephone numbers. With JDUPSEQ specified as shown, the records are returned as follows:

NAME	**ADDR**	**TELEPHONE**
Anne	120 1st St.	555-1111
Anne	120 1st St.	555-2222
Anne	120 1st St.	555-6666
Doug	40 Pillsbury	555-5555

The JDUPSEQ keyword only affects the order of records when duplicates exist.

Figure 2-47 assumes the logical file is based on the same physical files as Figure 2-46. There are three records for Anne in PF2, showing three telephone numbers, as shown in Figure 2-47 on page 2-57.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JREC                      JFILE(PF1 PF2)
00020A          J                           JOIN(PF1 PF2)
00030A                                      JFLD(NAME1 NAME2)
00040A                                      JDUPSEQ(PHONE *DESCEND)
00050A            NAME1
00060A            ADDR
00070A            PHONE
     A
```

Figure 2-47. Specifying the JDUPSEQ Keyword (Example 2)

When you specify JDUPSEQ with *DESCEND, the records are returned as follows:

NAME1	ADDR	TELEPHONE
Anne	120 1st St.	555-6666
Anne	120 1st St.	555-2222
Anne	120 1st St.	555-1111
Doug	40 Pillsbury	555-5555

The list shows Anne's telephone numbers in descending order.

JFILE (Joined Files)—Join Logical Files Only

Use this record-level keyword to identify the physical files containing the data to be accessed through the join logical file you are defining.

The format of the keyword is:

```
JFILE([library-name/]physical-file-name [..32])
```

This keyword is similar to the PFILE keyword except it identifies this file as a join logical file. The JFILE keyword is not allowed with the PFILE keyword.

The JFILE keyword is required at the record level in a join logical file. The JFILE keyword requires a minimum of two physical file names. You can specify the same file name more than once.

The first file is called the primary file, which is the file from which the join will begin. All other files are called secondary files. Up to 31 secondary files can be specified (32 total files on the JFILE keyword).

Distributed data management (DDM) files are allowed on the JFILE keyword only when the logical file is being created on a remote system. Refer to the *DDM Guide* for more information.

The following considerations apply to the order in which you specify physical files on the JFILE keyword:

- If the physical files have a different number of records, specify physical files with fewer records toward the left on the JFILE keyword. The primary file should have as many or fewer records than the secondary files. This can improve performance when reading files.
- Primary and secondary files specified in join specifications must be in a specific order. This order depends on the order in which the files are specified on the JFILE keyword. See Figure 2-54 on page 2-62 in the "JOIN (Join)—Join Logical Files Only" keyword description.

- The JOIN and JREF keywords can use relative file numbers to identify files specified by the JFILE keyword. The first file specified on the JFILE keyword has relative file number 1, the second file has relative file number 2, and so on up to 32. If you use relative file numbers instead of file names on the JOIN and JREF keywords, the order of files on the JFILE keyword can affect the way the JOIN and JREF keywords are specified.

Note: If the names in the physical file are not unique, you must specify relative file numbers.

Figure 2-48 and Figure 2-49show how to specify the JFILE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JREC                      JFILE(PF1 PF2)
00020A          J                           JOIN(PF1 PF2)
00030A                                      JFLD(NAME1 NAME2)
     A
```

Figure 2-48. Specifying the JFILE Keyword (Example 1)

In the join logical file, PF1 is the primary file and PF2 is the secondary file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JREC                      JFILE(MYLIBA/PHYSICAL1 +
00020A                                      MYLIBB/PHYSICAL2 MYLIBC/PHYSICAL3)
00030A          J                           JOIN(1 2)
00040A                                      JFLD(FIELD1 FIELD2)
00050A          J                           JOIN(1 3)
00060A                                      JFLD(FIELD1 FIELD2)
     A
```

Figure 2-49. Specifying the JFILE Keyword (Example 2)

In the join logical file, file PHYSICAL1 in library MYLIBA is the primary file. File PHYSICAL2 in library MYLIBB and file PHYSICAL3 in library MYLIBC are secondary files.

JFLD (Joined Fields)—Join Logical Files Only

Use this join-level keyword to identify the *from* and *to* fields whose values are used to join physical files in a join logical file. These fields are both referred to as join fields.

The format of the keyword is:

```
JFLD(from-field-name to-field-name)
```

The join fields must correspond to fields in the physical files identified on the JOIN keyword for this join specification. The name you specify on the JFLD keyword must be the same as the name specified in the physical file unless it was renamed in the join logical file. If you do not specify a JOIN keyword, then the JFILE keyword is used.

This keyword is valid only for join logical files.

At least one JFLD keyword is required for each join specification. A join specification is identified by J in position 17. Since at least one join specification is required in a join logical file, you must have at least one JFLD keyword specified in a join logical file.

These fields need not also be specified as fields in the record format for a join logical file.

To specify additional join fields to use when joining physical files, specify more than one JFLD keyword.

The field names you specify on the JFLD keyword must either be specified at the field level in the join record format or in one of the physical files, which are specified on the JFILE keyword.

The OS/400 program uses the following search order to match join field names with defined fields:

1. Fields specified in the join logical file at the field level in positions 19 through 28.

   Note: Fields that specify the CONCAT, RENAME, or SST keywords are valid as join fields; fields that are specified on CONCAT, RENAME, or SST keywords cannot be join fields.

2. Fields in the physical file specified on the JOIN keyword.

The rules for specifying join fields are as follows:

- The *from* field must be found in the *from* file specified on the JOIN keyword.
- The *to* field must be found in the *to* file specified on the JOIN keyword.
- Join fields are not required to be defined in the join record format.
- *From* and *to* fields must have the same field attributes (length, data type, and decimal positions) but need not have the same name. When the joined fields in the physical files have different definitions, you must redefine one or both fields. If you redefine fields, there is a possibility of data conversion errors. See "Length (Positions 30 through 34)" on page 2-22, "Data Type (Position 35)" on page 2-25, "Decimal Positions (Positions 36 and 37)" on page 2-27, and "Usage (Position 38)" on page 2-29.

  Note: Character fields need not have the same length. The shorter join field is padded with blanks to equal the length of the longer join field.

- In a single join specification, the total length of fields specified as *to* fields on the JFLD keyword and fields specified on the JDUPSEQ keyword can be up to 120 bytes.

Figure 2-50 and Figure 2-51 on page 2-60 show how to specify the JFLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JREC                      JFILE(PF1 PF2)
00020A          J                           JOIN(PF1 PF2)
00030A                                      JFLD(NAME1 NAME2)
     A
```

Figure 2-50. Specifying the JFLD Keyword (Example 1)

In the join logical file, the JFLD keywords specify that NAME1 in physical file PF1 is used to join to NAME2 in physical file PF2.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JREC                      JFILE(PF1 PF2)
00020A          J                           JOIN(PF1 PF2)
00030A                                      JFLD(NAME1 NAME2)
00040A                                      JFLD(ADDR1 ADDR2)
     A
```

Figure 2-51. Specifying the JFLD Keyword (Example 2)

In the join logical file, the JFLD keywords specify that NAME1 and ADDR1 in physical file PF1 are used to join to NAME2 and ADDR2 in physical file PF2.

JOIN (Join)—Join Logical Files Only

Use this join-level keyword to identify which pair of files are joined by the join specification in which you specify this keyword.

The format of the keyword is:

```
JOIN(from-file to-file)
```

This keyword is valid only for join logical files.

You can use file names or relative file numbers to indicate which files are to be joined. You must specify a relative file number if the same file is specified more than once on the JFILE keyword.

If you specify file names, you must select files that you have specified only once on the JFILE keyword. On each JFILE keyword, the *from* file must occur before the *to* file.

If you specify numbers, they correspond to the files specified on the JFILE keyword. The following are the valid values:

File	Valid Values
From-file number	1 through 31
To-file number	2 through 32

The *from-file* number must always be less than the *to-file* number.

Special rules apply to the order in which you specify *from* and *to* files. See Figure 2-54 on page 2-62 for details.

In a join logical file, each secondary file can be a *to* file only once.

Join Specifications

To describe a join specification do the following:

- Specify J in position 17 immediately after the record level (before the first field name in positions 19 through 28). J in position 17 indicates the beginning of a join specification.
- Specify the JOIN keyword. The JOIN keyword is optional when only two files are specified on the JFILE keyword. When more than two physical files are specified on the JFILE keyword, one JOIN keyword is required for each secondary file.
- Specify the JFLD keyword at least once for each join specification.

- The end of the join specification is indicated by another J in position 17 or by a field name specified in positions 19 through 28.

There must be one join specification for each secondary file specified on the JFILE keyword. Therefore, at least one join specification is required in a join logical file.

You can specify the JOIN keyword only once within a join specification.

Figure 2-52, Figure 2-53, and Figure 2-54 on page 2-62 show how to specify the JOIN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   JFILE(PFA PFB PFC)
00020A          J                           JOIN(PFA PFB)
00030A                                      JFLD(NAME1 NAME2)
00040A          J                           JOIN(PFA PFC)
00050A                                      JFLD(NAME1 NAME3)
00060A            NAME1
     A
```

Figure 2-52. Specifying the JOIN Keyword (Example 1)

In Figure 2-52, PFA is joined to PFB and also to PFC.

Figure 2-53 shows how to specify JOIN using relative file numbers.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   JFILE(PFA PFB PFC)
00020A          J                           JOIN(1 2)
00030A                                      JFLD(NAME1 NAME2)
00040A          J                           JOIN(1 3)
00050A                                      JFLD(NAME1 NAME3)
00060A            NAME1
     A
```

Figure 2-53. Specifying the JOIN Keyword (Example 2)

Figure 2-53 is equivalent to Figure 2-52. PFA is the first physical file specified on the JFILE keyword and has relative file number 1. PFB and PFC are the second and third files specified on the JFILE keyword and have relative file numbers 2 and 3, respectively.

Figure 2-54 on page 2-62 shows the order of associated physical files.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R J3                        JFILE(VENDORS PARTS PARTWARE +
00020A                                            WAREHOUSE
                                                  1
00030A          J                           JOIN(1   2)    2
00040A                                      JFLD(VNBR VNUM)
00050A          J                           JOIN(2   3)    3
00060A                                      JFLD(PNBR PNBR)
00070A          J                           JOIN(3   4)    3
00080A                                      JFLD(WNBR WNBR)
00090A            VNAME
00100A            VAD1
00110A            VAD2
00120A            PNBR                      JREF(2)
00130A            WNBR                      JREF(4)
00140A            BIN
00150A            QOH
     A
```

Figure 2-54. Specifying the JOIN Keyword (Example 3)

The join logical file in Figure 2-54 is based on four physical files. The VENDORS file, which is specified first on the JFILE keyword, is the primary file and has relative file number 1. The PARTS, PARTWARE, and WAREHOUSE files, which are secondary files, have relative file numbers 2, 3, and 4, respectively.

Notice the pattern of numbers specified on the JOIN keywords:

1 The first parameter value on the first JOIN keyword (the first *from* file) must be the primary file.

2 The second parameter values specified on the JOIN keywords (*to* files) must reflect the same order as the secondary files on the JFILE keyword. If file names were specified instead of relative file numbers, they would have to be specified in the following order:

```
J    JOIN(VENDORS PARTS)
J    JOIN(PARTS PARTWARE)
J    JOIN(PARTWARE WAREHOUSE)
```

3 On each JOIN keyword, the *from* and *to* files must be specified in ascending order.

Note: A file can be specified as a *from* file more than once. For example, the parameters on the JOIN keywords above could have been specified as follows:

```
J    JOIN(1 2)
J    JOIN(2 3)
J    JOIN(2 4)
```

However, a file can be specified as a *to* file only once.

JREF (Join Reference)—Join Logical Files Only

Use this field-level keyword in join logical files for fields whose names are specified in more than one physical file. This keyword identifies which physical file contains the field.

The format of the keyword is:

```
JREF(file-name | relative-file-number)
```

You can specify either the physical file name or its relative file number. If a physical file is named twice on the JFILE keyword, then you must specify the relative file number. The relative file number corresponds to the physical file name specified on the JFILE keyword. For example, specifying JREF(1) associates a field with the first physical file specified on the JFILE keyword. Specifying JREF(2) associates a field with the second physical file specified on the JFILE keyword. See Figure 2-56.

This keyword is valid only in a join logical file.

Join logical files are based on two or more physical files (up to 32). Field names specified in the record format in a join logical file must uniquely identify only one field from the physical files on which the join logical file is based. For example, if the join logical file is based on two physical files, and each physical file has the field named NAME, you must specify the JREF keyword to identify which physical file the field comes from.

When a field name is unique among the physical files specified on the JFILE keyword, this keyword is optional. For example, if the join logical file is associated with two physical files, and only one of the physical files has a field named NAME1, you do not need to specify the JREF keyword.

If the join logical file is associated with only one physical file (the JFILE keyword names the same file twice), you must specify the JREF keyword on every field.

Figure 2-55 and Figure 2-56 show how to specify the JREF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JOINREC                   JFILE(PFA PFB PFC)
00020A               :
00030A               :
00040A               :
00050A            NAME                      JREF(PFB)
     A
```

Figure 2-55. Specifying the JREF Keyword (Example 1)

In Figure 2-55, the JREF keyword is specified with the file name, and NAME occurs in both PFA and PFB. Specifying JREF (PFB) associates this field with PFB.

Figure 2-56 shows how to use the file reference numbers to specify JREF.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R JOINREC                   JFILE(PFA PFB PFC)
00020A               :
00030A               :
00040A               :
00050A            NAME                      JREF(2)
     A
```

Figure 2-56. Specifying the JREF Keyword (Example 2)

Figure 2-56 is equivalent to Figure 2-55. In Figure 2-56, NAME occurs in both PFA and PFB. Specifying JREF(2) associates this field with PFB (the second of the physical files specified on the JFILE keyword).

LIFO (Last-In First-Out)

Use this file-level keyword to specify that records with duplicate key values from the same physical file member are retrieved in a last-in first-out (LIFO) order.

This keyword has no parameters.

LIFO is not allowed with an FCFO, FIFO, UNIQUE, or REFACCPTH keyword.

If you do not specify FCFO, FIFO, LIFO, or UNIQUE, records with duplicate key values are retrieved in first-in first-out (FIFO), last-in first-out (LIFO), or first-changed first-out (FCFO) order, but the order in which they are retrieved is not guaranteed.

At least one key field must be specified in a file containing the LIFO keyword. The LIFO keyword is not valid when you specify FILETYPE(*SRC) on the Create Physical File (CRTPF) or Create Logical File (CRTLF) command.

Figure 2-57 shows how to specify the LIFO keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     LIFO
00020A          R CUSREC                   TEXT('CUSTOMER RECORD')
00030A            CUSNAMEF      10A
00040A            CUSNAMEM       1A
00050A            CUSNAMEL      10A
00060A          K CUSNAMEL
     A
```

Figure 2-57. Specifying the LIFO Keyword

NOALTSEQ (No Alternative Collating Sequence)

Use this key field-level keyword to specify that the ALTSEQ keyword specified at the file level does not apply to this key field. If you specify ABSVAL or SIGNED for a key field, NOALTSEQ is automatically in effect whether or not the NOALTSEQ keyword is specified for that key field.

This keyword has no parameters.

Figure 2-58 shows how to specify the NOALTSEQ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     ALTSEQ(TABLELIB/TABLE1)
00020A          R DSTR
00030A            :
00040A            :
00050A            CODE           1
00060A            NAME          20
00070A            :
00080A            :
00090A          K CODE
00100A          K NAME                     NOALTSEQ
     A
```

Figure 2-58. Specifying the NOALTSEQ Keyword

Records with the record format DSTR are sequenced by the composite keys CODE and NAME. CODE is sequenced by the alternative collating sequence (TABLE1 in TABLELIB). NAME is sequenced by the EBCDIC collating sequence. NOALTSEQ prevents the sequence of the NAME field from being altered.

PFILE (Physical File)—Logical Files Only

Use this record-level keyword to identify the physical file(s) containing the data to be accessed through the record format you are now defining.

The format of the keyword is:

```
PFILE([library-name/]physical-file-name [.32])
```

The PFILE keyword is required on every record format in a simple or multiple format logical file. This keyword is similar to the JFILE keyword except it identifies this file as a simple or multiple format logical file; the PFILE keyword is not allowed with the JFILE keyword. Up to 32 physical file names can be specified on PFILE keywords in a logical file. If the maximum is being used, 32 physical file names can be specified on one record format (using one PFILE keyword) or 32 physical file names can be distributed among 32 record formats; or, file names can be unevenly distributed among record formats. In any case, the maximum number of physical file names allowed is 32. For restrictions on specifying multiple physical files when creating a logical file, see the appropriate high-level language manual.

For each physical-file-name, a library-name is optional. If the library-name is omitted, the library list (*LIBL) that is in effect at file creation time is used.

If you specify more than one physical file name for one record format in a multiple format logical file, all fields in the record format for the logical file must exist in all physical files specified. This type of file cannot be externally described in RPG III because it results in duplicate format names. If your program requires access to fields that occur in one or more of the physical files specified on the PFILE keyword, but not in all of them, you can do one of the following:

- Specify a join logical file. If you do this, use the JFILE keyword instead of the PFILE keyword.
- Specify a separate logical file record format that includes fields not in the first physical file.

  For instance, if FLD1 and FLD2 occur in physical files PF1, PF2, and PF3, but FLD3 occurs only in PF3, you cannot specify FLD3 in a logical file record format based on PF1 and PF2. To provide access to FLD3, either specify a second logical file record format that includes FLD3 or use a join logical file.

  You cannot use a simple or multiple format logical file to bring together into one record format fields from separate physical files. Use a join logical file to accomplish this. A record read through the use of a record format in a simple or multiple format logical file can contain data from only one physical file, and a record written through the use of a record format in a logical file can be stored only in one physical file.

Distributed Data Management (DDM) files are allowed on the PFILE keyword only when the logical file is being created on a remote system. Refer to the *DDM Guide* for more information.

Figure 2-59 on page 2-66, Figure 2-60 on page 2-66, and Figure 2-61 on page 2-66 show how to specify the PFILE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R LOGRCD1                   PFILE(PF1)
     A
```

Figure 2-59. Specifying the PFILE Keyword (Example 1)

In Figure 2-59, LOGRCD1 can use fields only in PF1.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R LOGRCD2                   PFILE(PF1 PF2)
     A               :
     A               :
00020A          R LOGRCD3                   PFILE(PF1 PF2 PF3)
     A               :
     A               :
     A
```

Figure 2-60. Specifying the PFILE Keyword (Example 2)

In Figure 2-60, LOGRCD2 must use fields common to PF1, and PF2, and LOGRCD3 must use fields common to PF1, PF2, and PF3.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R LOGRCD4                   PFILE(PF1)
     A               :
     A               :
00020A          R LOGRCD5                   PFILE(PF2)
     A               :
     A               :
00030A          R LOGRCD6                   PFILE(LIB1/PF6)
     A
```

Figure 2-61. Specifying the PFILE Keyword (Example 3)

In Figure 2-61, LOGRCD4, LOGRCD5, and LOGRCD6 can have unique fields. LOGRCD6 specifies a qualified physical-file name.

RANGE (Range)

Specify this keyword at the field level, the select/omit-field level, or both.

The format of the keyword is:

```
RANGE(low-value high-value)
```

Specifying RANGE at the Field Level

At the field level, this keyword specifies validity checking for the field you are defining when it is referred to later during display file creation.

RANGE does not affect the physical or logical file you are defining. When you define an input-capable field in a display file, you can refer to the field you are now defining by specifying R in position 29 and the REF or REFFLD keyword. During display file creation, the OS/400 program copies the RANGE keyword and other field attributes from the field in the physical or logical file into the field in the display file. You can override the RANGE keyword (as well as all other validity-checking keywords and the CHKMSGID keyword) by specifying any validity checking

keyword for the field in the display file. See "Reference (Position 29)" on page 3-8 for details.

The rules for specifying this keyword in a physical or logical file are the same as for a display file. See "RANGE (Range)" on page 3-182 for more information on how to specify this keyword.

You cannot specify the RANGE keyword on a floating-point field (F in position 35) or a hexadecimal field (H in position 35). Do not specify the RANGE keyword on a date, time, or timestamp field (L, T, or Z in position 35).

Specifying RANGE at the Select/Omit-Field Level

At the select/omit-field level, this keyword selects or omits records retrieved from the physical file(s) when your program sends an input operation using the record format in which the select/omit field is specified.

The following rules apply:

- If the field you are defining is a character field, you must specify character strings or hexadecimal character strings.

  Specify character strings enclosed in apostrophes (see Figure 2-62).

  Specify hexadecimal character strings as an X followed by a combination of the digits 0 through 9 and the letters A through F, enclosed in apostrophes. The number of hexadecimal digits in apostrophes must be exactly twice the specified length of the field. See Figure 2-63 on page 2-68.

- If you are defining a numeric field, you must specify a numeric value (digits 0 through 9 specified without apostrophes). See Figure 2-62.

- If you are defining a date field, specify a valid date in the same format specified on the DATFMT keyword and use the same separator as specified on the DATSEP keyword. For example, `RANGE('12/15/91' '12/31/91')` is the default value if *MDY is specified for DATFMT and '/' is specified for DATSEP.

- If you are defining a time field, specify a valid time in the same format specified on the TIMFMT keyword and use the same separator as specified on the TIMSEP keyword. For example, `RANGE('11.00.00' '12.00.00')` is the default value if *ISO is specified for TIMFMT. The default separator for *ISO is a period (.).

- If you are defining a timestamp field, you must specify the default value in the following format:

  ```
  RANGE('YYYY-MM-DD-HH.MM.SS.UUUUUU' 'YYYY-MM-DD-HH.MM.SS.UUUUUU')
  ```

Figure 2-62 shows how to specify character and numeric strings for the RANGE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD                    PFILE(PF1)
     A
00020A            FIELDA         1  0       RANGE(2 5)     1
00030A            FIELDB         1          RANGE('2' '5')
00040A            FIELDC
00050A          K FIELDD
00060A          S FIELDA                    RANGE(1   4)   2
     A
```

Figure 2-62. Specifying the RANGE Keyword (Example 1)

In Figure 2-62, RANGE (1) is specified for FIELDA and FIELDB as a validity checking keyword for display files that refer to FIELDA and FIELDB. In the display file, RANGE requires that the work station user type only 2, 3, 4, or 5 in FIELDA or FIELDB. FIELDA is a numeric field and FIELDB is a character field. The type of field you specify depends on the high-level language the program is written in.

RANGE (2) is specified as a select/omit keyword for FIELDA. Records from the physical file PF1 are retrieved through this logical file record format only if the value of FIELDA is 1, 2, 3, or 4.

Figure 2-63 uses hexadecimal character strings when specifying the RANGE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD1                      PFILE(PF1)
00020A            CODEA
00030A            FLD1
00040A            FLD2
00050A          K FLD1
00060A          S CODEA                     RANGE(X'51' X'54')
     A
```

Figure 2-63. Specifying the RANGE Keyword (Example 2)

RANGE is specified as a select/omit keyword for CODEA (which is a 1-byte field). Records from physical file PF1 are retrieved through this record format only if the value of field CODEA is from hex 51 through hex 54.

REF (Reference)—Physical Files Only

Use this file-level keyword to specify the name of the file from which field descriptions are retrieved.

The format of the keyword is:

```
REF([library-name/]database-file-name [record-format-name])
```

REF supplies the field attributes from a previously defined record format. Specify the file name once in the REF keyword instead of on several REFFLD keywords if each field description refers to the same file. To refer to more than one file, use the REFFLD keyword. You can specify the REF keyword only once.

The database-file-name is a required parameter value for this keyword. The library-name and the record-format-name are optional.

If you do not specify the library-name, the library list (*LIBL) in effect at file creation time is used. Specify a record-format-name as a parameter value for this keyword if there is more than one record format. If you do not specify the record-format-name, each record format is searched sequentially. The first occurrence of the field name is used. For information on how the choice of REF and REFFLD keywords controls these searches, see Appendix A, "When to Specify REF and REFFLD."

You can specify a Distributed Data Management (DDM) file on this keyword.

When using a DDM file, the database-file-name and library-name are the DDM file and library names on the source system. The record-format-name is the record format name in the remote file on the target system.

Figure 2-64 on page 2-69 and Figure 2-65 on page 2-69 show how to specify the REF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      REF(FILE1)
00020A          R RECORD
00030A            FLD1       R
     A
```

Figure 2-64. Specifying the REF Keyword (Example 1)

FLD1 has the same attributes as the first (or only) FLD1 in FILE1.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      REF(LIB1/FILE1 RECORD2)
00020A          R RECORD
00030A            FLD1       R
     A
```

Figure 2-65. Specifying the REF Keyword (Example 2)

FLD1 has the same attributes as FLD1 in RECORD2 in FILE1 in LIB1.

REFACCPTH (Reference Access Path Definition)—Logical Files Only

Use this file-level keyword to specify that the access path information for this logical file is to be copied from another physical or logical file. The access path information includes key information, select and omit information, alternative collating sequence information, dynamic select information, and key sequencing information (specified in the FCFO, FIFO, LIFO, and UNIQUE keywords).

The format of the keyword is:

```
REFACCPTH([library-name/]database-file-name)
```

The name of the file defining the access path is the parameter value for the keyword.

The file containing the REFACCPTH keyword cannot contain key, select, or omit fields.

The record format(s) in the file you are defining can contain fewer or more fields than the record format(s) in the physical file on which this logical file is based.

If the file specified on the REFACCPTH keyword is a simple or multiple format logical file, it and the file containing the REFACCPTH keyword must have the same physical files specified in the same order on the PFILE keyword.

The REFACCPTH keyword is not allowed in join logical files. You can specify a join logical file as the parameter value on the REFACCPTH keyword only if all the following are true:

- The file you are creating is a simple logical file.
- The physical file specified on the PFILE keyword is the first file specified on the JFILE keyword in the join logical file.
- The join logical file has key fields specified and does not have select and omit fields specified.

You cannot specify a Distributed Data Management (DDM) file on this keyword.

You cannot specify the REFACCPTH keyword with the DYNSLT, ALTSEQ, FCFO, FIFO, LIFO, or UNIQUE keywords.

Figure 2-66 shows how to specify the REFACCPTH keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A* ORDER HEADER LOGICAL FILE (ORDHDR11)
00040A                                  REFACCPTH(DSTLIB/ORDHDRL)
00050A          R ORDHDR                PFILE(ORDHDRP)
     A
```

Figure 2-66. Specifying the REFACCPTH Keyword

REFFLD (Referenced Field)—Physical Files Only

Use this field-level keyword to refer to a field under one of these three conditions:

- When the name of the referenced field is different from the name in positions 19 through 28
- When the name of the referenced field is the same as the name in positions 19 through 28, but the record format, file, or library of the referenced field is different from that specified with the REF keyword
- When the referenced field occurs in the same DDS source file as the referencing field

The format of the keyword is:

```
REFFLD([record-format-name/]referenced-field-name
[{*SRC | [library-name/]database-file-name}])
```

The referenced-field-name is required even if it is the same as the name of the field being defined. Use the record-format-name when the referenced file contains more than one record format. Use *SRC (rather than the database-file-name) when the field name being referred to is in the same DDS source file as the field being defined. *SRC is the default value when the database-file-name and the library-name are not specified.

Note: When you refer to a field in the same DDS source file, the field being referred to must precede the field being defined.

Specify the database-file-name (with its library-name, if necessary) to search a particular database file.

An R must be in position 29. Some keywords specified with the field being referred to are not included on the field being defined. For more information, see "Reference (Position 29)" on page 2-21.

If you specify REF at the file level and REFFLD at the field level in the same DDS source file, the REFFLD specification is used. The search sequence depends on both the REF and REFFLD keywords. For more information, see Appendix A, "When to Specify REF and REFFLD."

You can specify a Distributed Data Management (DDM) file on this keyword.

When using a DDM file, the database-file-name and library-name are the DDM file and library names on the source system. The referenced-field-name and the record-format-name are the field name and the record format name in the remote file on the target system.

Figure 2-67 shows how to code the REFFLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R FMAT1
00020A            ITEM           5
00030A            ITEM1     R               REFFLD(ITEM)
00040A            ITEM2     R               REFFLD(FMAT1/ITEM)
00050A            ITEM3     R               REFFLD(ITEM FILEX)
00060A            ITEM4     R               REFFLD(ITEM LIBY/FILEX)
00070A            ITEM5     R               REFFLD(FMAT1/ITEM LIBY/FILEX)
00080A            ITEM6     R               REFFLD(ITEM *SCR)
     A
```

Figure 2-67. Specifying the REFFLD Keyword

The default for lines 00030 and 00040 is to search the DDS source file where they are specified because the REF keyword is not specified. In line 00080, the parameter *SRC explicitly specifies this source file. See Appendix A, "When to Specify REF and REFFLD," for explanations of the various specifications.

REFSHIFT (Reference Shift)

Use this field-level keyword to specify a keyboard shift for a field when the field is referred to in a display file or DFU operation.

The format of the keyword is:

```
REFSHIFT(keyboard-shift)
```

When defining an input-capable field in a display file, refer to the field you are now defining by specifying the letter R in position 29 and the REF or REFFLD keyword. At display file creation, the OS/400 program copies the REFSHIFT keyword and other field attributes from the field in the logical file into the field in the display file. You can override the editing specified in the display or printer file by specifying new editing keywords. Specifying the DLTEDT keyword deletes all editing for the field. See "Reference (Position 29)" on page 3-8 for details.

The keyboard shift in the display file (position 35) becomes the parameter value specified on this keyword instead of the data type specified in the database file. When you refer to a field with the REFSHIFT keyword from a physical or logical file, the REFSHIFT keyword is copied into the new field. However, if the field attributes (such as data type) specified for the new field are not compatible with the keyboard shift specified on the REFSHIFT keyword, the keyword is ignored.

This keyword is valid for fields with data types A, S, B, or P. Choose any keyboard shift that is compatible with the data type as a parameter value. The following parameters apply to the data types:

- Character field (A): REFSHIFT(A | X | W | N | I | D | M)
- Numeric fields (S, B, P): REFSHIFT(S | Y | N | I | D)

Refer to "Data Type/Keyboard Shift (Position 35)" on page 3-10 for more information on the parameters.

Figure 2-68 shows how to specify the REFSHIFT keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD
00020A            FIELDA         5          REFSHIFT(X)
00030A            FIELDN         4P         REFSHIFT(N)
     A
```

Figure 2-68. Specifying the REFSHIFT Keyword

Fields FIELDA and FIELDN in the file (FILE1) have the REFSHIFT keyword specified as shown. The REFSHIFT keyword is used when the fields are referred to from a display file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        REF(FILE1)
00020A          R RECORD
00030A            FIELDA    R           1  2
     A            FIELDN    R           2  2
     A
```

The display file references FILE1 (REF keyword). Fields FIELDA and FIELDN in this display file reference fields FIELDA and FIELDN in FILE1. When the REFSHIFT keyword is specified for the fields in FILE1, the keyboard shift specified with the REFSHIFT keyword is used in the display file, and the fields have the following attributes:

- FIELDA has keyboard shift X in position 35.
- FIELDN has keyboard shift N in position 35.

RENAME (Rename)—Logical Files Only

Use this field-level keyword when you want a field name in the logical record format you are defining to be different from its corresponding physical file field name.

The format of the keyword is:

```
RENAME(physical-file-field-name)
```

The name as it appears in the physical file record format is the parameter value for this keyword. One field in the physical file record format can be renamed to more than one field in the record format being described.

You would rename fields in situations similar to the following:

- You want to use programs that were written using a different name for the same field.
- You want to map one field in a physical file record format to two or more fields in a logical file record format.
- You are using a high-level language (such as RPG III) that does not permit two fields having different names to have only one data storage area. By specifying the RENAME keyword, you allow both fields to access the same data storage area.

If you specify the same physical field more than once in a record format in the logical file (that is, by using either RENAME or CONCAT), the sequence in which the fields are specified in the logical file is the sequence in which the data is moved

to the physical file on an update or insert operation. Thus, the value in the last occurrence of the physical field is the value that is put in the physical record and is the value that is used for all keys built over that physical field. All previous values of the same field are written over and have no effect.

The following examples show how to specify the RENAME keyword.

In Figure 2-69, the QTYDUE field in the physical file (PF1) is renamed QTY in the logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00050A          R RCD1                    PFILE(PF1)
00060A            QTY                     RENAME(QTYDUE)
     A
```

Figure 2-69. Specifying the RENAME Keyword (Example 1)

In Figure 2-70, the renamed field in the logical file (QTY) is used as a key field.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00050A          R RCD2                    PFILE(PF2)
00060A            :
     A            :
00130A            QTY                     RENAME(QTYDUE)
00140A          K QTY
     A
```

Figure 2-70. Specifying the RENAME Keyword (Example 2)

SIGNED (Signed)

Use this key field-level keyword to specify that when sequencing the values associated with this numeric key field, the OS/400 program is to consider the signs of the values (negative versus positive values).

This keyword has no parameters.

The following example shows six records with a zoned decimal key field:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
1	98	F9F8
2	00	F0F0
3	98–	F9D8
4	97	F9F7
5	20	F2F0
6	99	F9F9

By default (with no sequencing keywords specified and without the ALTSEQ keyword), the key field has the SIGNED attribute. The records are sequenced in the following order:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
3	98–	F9D8
2	00	F0F0
5	20	F2F0
4	97	F9F7

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
1	98	F9F8
6	99	F9F9

If both SIGNED and DESCEND are specified, the records are sequenced in this order:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
6	99	F9F9
1	98	F9F8
4	97	F9F7
5	20	F2F0
2	00	F0F0
3	98-	F9D8

This keyword is not valid for a character, date, time, timestamp, or hexadecimal data type field. You cannot use it with the ABSVAL, DIGIT, UNSIGNED, or ZONE keywords.

SIGNED (a key field-level keyword) causes ALTSEQ (a file-level keyword) to be ignored. If you specify SIGNED for a key field, NOALTSEQ is automatically in effect for that key field even if ALTSEQ is specified at the file level. This occurs whether or not NOALTSEQ is specified.

Figure 2-71 shows how to specify the SIGNED keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD
00020A            FLDA           7S 2
00030A            FLDB
00040A          K FLDA                         SIGNED
     A
```

Figure 2-71. Specifying the SIGNED Keyword

SST (Substring)—Logical Files Only

Use this field-level keyword to specify a character string that is a subset of an existing character, hexadecimal, zoned field, or graphic.

The format of the keyword is:

```
SST(field-name starting-position [length])
```

The field-name parameter specifies the name of the field from which the substring is taken. This field must be defined in the same logical file format prior to the SST field (which is the field you are defining) or it must exist in the physical file specified on the PFILE or JFILE keyword. To find the field, the system searches for a matching field name as follows:

1. First, the system searches the field names specified in positions 19 to 28 in the logical file format prior to the SST field.
2. If no matching field name is found in positions 19 to 28 in the logical file format, the system searches for the field name in the physical file specified on the PFILE or JFILE keyword, according to the following rules:

- If the logical file is a simple or multiple format logical file, the field must exist in all files specified on the PFILE keyword.
- If the logical file is a join logical file and the JREF keyword is specified on the SST field, the field must exist in the JFILE referred to by the JREF keyword.
- If the logical file is a join logical file and the JREF keyword is not specified on the SST field, the field must exist in exactly one JFILE.

The substring begins at the starting position you specify on the SST keyword. Specify its length either as the third parameter on the keyword or on the field length (DDS positions 30 through 34). The starting position is a required parameter; the length is optional.

Note: Both the starting position and length values must be positive integer values and the defined substring must not be greater than the length of the field specified on the SST keyword.

The following rules apply:

- If the field on the SST keyword is hexadecimal, the resulting field is hexadecimal; otherwise, the resulting field is always character. If the data type is not specified in DDS, a default of H or A is assigned.
- The use of the resulting field must be either input-only (I) or neither (N).
- The length of the resulting field is optional. You must specify either the field length or the length parameter on the keyword. If you specify both, they must be equal. If the field length is not specified, it is assigned the length parameter on the keyword.
- You cannot specify this keyword on the same field with the CONCAT, RENAME, or TRNTBL keywords.
- The field specified on this keyword cannot be defined with the CONCAT, TRNTBL, or SST keywords.

Figure 2-72 and Figure 2-73 on page 2-76 show how to specify the SST keyword.

Figure 2-72 shows how to specify the SST keyword on a join logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                   JFILE(PF1 PF2)
     A          J                           JOIN(1 2)
     A                                      JFELD(CITY CITY)
     A            ADDRESS                   JREF(2)
     A            CITY              I       SST(ADDRESS 21 10)
     A                                      JREF(2)
     A            SYEAR             I       SST(SALESDATE 5)
     A            NAME                      JREF(1)
     A            CUSTNAME          I       SST(NAME 11 10) JREF(2)
     A          K SYEAR
     A
```

Figure 2-72. Specifying the SST Keyword on a Join Logical File

Figure 2-72 shows:

- CITY is a substring of ADDRESS from the logical format and is joined with CITY from PF1.

- CUSTNAME is a substring of NAME from PF2, since NAME in the logical file format has a different JREF.
- Since SYEAR is a key field, the unique field name SALESDATE must exist in PF1.
- The usage (position 38) for a field with the SST keyword must be I (input only). Since this is a join logical file, the usage default is I.

Figure 2-73 shows how to specify the SST keyword on a simple or multiple format logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R REC1                        PFILE(PFA)
     A            LASTNAME           I        SST(NAME 10 10)
     A          K LASTNAME
     A
```

Figure 2-73. Specifying the SST Keyword on a Simple or Multiple Format Logical File

The LASTNAME field is a substring of NAME from PFA. The usage I in position 38 must be specified for SST fields in simple or multiple format logical files.

TEXT (Text)

Use this record- or field-level keyword to supply a text description (or comment) for the record format or field that is used for program documentation.

The format of the keyword is:

```
TEXT('description')
```

The text must be enclosed in apostrophes. If the length of the text is greater than 50 positions, only the first 50 characters are used by the high-level language compiler.

Figure 2-74 shows how to specify the TEXT keyword at the record and field levels.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CUSMST                      TEXT('Customer Master Record')
00020A            FLD1           3  0        TEXT('ORDER NUMBER FIELD')
     A
```

Figure 2-74. Specifying the TEXT Keyword

TIMFMT (Time Format)

Use this field-level keyword to specify the format of a time field. This keyword is valid for either time fields (data type T) or zoned fields (data type S) whose corresponding physical file fields are time fields (data type T).

The format of the keyword is:

```
TIMFMT(time-format)
```

The following table describes the valid time formats and their default separators.

Format Name	Time Format Parameter	Time Format and Separator	Field Length	Example
Hours:Minutes:Seconds	*HMS	hh:mm:ss	8	14:00:00
International Standards Organization	*ISO	hh.mm.ss	8	14.00.00
IBM USA Standard	*USA	hh:mm AM or hh:mm PM	8	2:00 pm
IBM European Standard	*EUR	hh.mm.ss	8	14.00.00
Japanese Industrial Standard Christian Era	*JIS	hh:mm:ss	8	14:00:00

If you do not specify the TIMFMT keyword for a physical file, the default is *ISO.

If you do not specify the TIMFMT keyword for a logical file, the default is the time format from the physical file.

If you specify the time-format parameter value *ISO, *USA, *EUR, or *JIS, you cannot specify the TIMSEP keyword. These formats have a fixed separator.

The TIMFMT keyword overrides the job attribute for a time field. It does not change the system default.

Figure 2-75 shows how to specify the TIMFMT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A
00020A          R RECORD
00030A            TIMFLD1         T         TIMFMT(*ISO)
00040A            TIMFLD2         T         TIMFMT(*USA)
     A
```

Figure 2-75. Specifying the TIMFMT Keyword

If the current time is 2 o'clock p.m., the system time format is hhmmss, and the system time separator is ':', TIMFLD1 contains 14.00.00. TIMFLD2 contains 2:00 PM.

TIMSEP (Time Separator)

Use this field-level keyword to specify the separator character used for a time field. This keyword is valid only for time fields (data type T).

The format of the keyword is:

```
TIMSEP(*JOB | 'time-separator')
```

The time-separator parameter specifies the separator character that appears between the hour, minute, and second values. Valid values are a colon (:), period (.), and blank ( ). The parameter must be enclosed in apostrophes.

If you specify *JOB, the default is the job attribute.

For physical files, if you do not specify the TIMSEP keyword, the default is the job attribute.

For logical files, if you do not specify the TIMSEP keyword, the default is the separator character from the physical file. If you did not specify the TIMSEP keyword for the physical file field (*ISO, *USA, *EUR, or *JIS was specified on the TIMFMT keyword), the default is the job attribute.

If you specify *ISO, *USA, *EUR, or *JIS time format on the TIMFMT keyword, you cannot specify the TIMSEP keyword. These formats have a fixed separator.

If the DFT keyword is not specified, the default value is the current time.

The TIMSEP keyword overrides the job attribute for a time field. It does not change the system default.

Figure 2-76 shows how to specify the TIMSEP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A
00020A          R RECORD
00030A            TIMFLD1         T         TIMSEP(' ')
00040A            TIMFLD2         T         TIMSEP('.')
```

Figure 2-76. Specifying the TIMSEP Keyword

If the current time is 2 o'clock p.m., the system time format is hhmmss, and the system time separator is ':', TIMFLD1 contains 14 00 00. TIMFLD2 contains 14.00.00.

TRNTBL (Translation Table)—Logical Files Only

Use this field-level keyword to specify the name of a translation table to be used when passing this field between the physical file on the PFILE or JFILE keyword and your program. The field must be a character field and its length cannot be redefined in the logical file. If the TRNTBL keyword is specified with the CONCAT keyword, the fields specified on the CONCAT keyword must all be character fields.

The format of the keyword is:

```
TRNTBL([library-name/]translation-table-name)
```

The translation-table-name is a required parameter value; the library-name is optional. If you do not specify the library-name, the OS/400 program uses the library list (*LIBL) that is in effect at file creation time.

This keyword is valid only for character fields that are input-only (I specified in position 38) or neither (N specified in position 38) fields.

You cannot specify the TRNTBL keyword on a hexadecimal field (H in position 35). Do not specify the TRNTBL keyword on a date, time, or timestamp field (L, T, or Z in position 35).

You can specify as many as 99 different translation tables for different fields in the same logical file.

Translation occurs when the field is read from the physical file. Therefore, all functions specified in the logical file (such as key field sequencing, select/omit processing, and joining of records) depend on the translated version of the data.

The TRNTBL keyword changes the data in the records returned from the logical file. The ALTSEQ keyword changes only the order of the records returned from the logical file.

The TRNTBL keyword is similar to the CHRID keyword, except for the following:

- The TRNTBL keyword names the translation table to be used; the CHRID keyword does not.
- The TRNTBL keyword changes data on input to the program when your program is reading a logical file. The CHRID keyword changes the data for display or printing on a specific device. Use the TRNTBL keyword when your program will use the changed data (for example, in an IF-THEN-ELSE statement or a COBOL SORT statement). If your program is handling data that comes only from a logical file, you do not need to specify the CHRID keyword in display or printer files used by the program.

The TRNTBL keyword is not valid when you specify FILETYPE(*SRC) on the Create Logical File (CRTLF) command.

Notes:

1. When you use the TRNTBL keyword, the length of the field in the logical file must be the same as the length of the corresponding field in the physical file.
2. At file creation time, you must have use authority to the translation table. The translation table is created using the Create Table (CRTTBL) command.
3. The translation table specified in the TRNTBL keyword is referred to only when the logical file is created. Therefore, a change to a translation table does not affect the logical file until the logical file is re-created.

Figure 2-77 shows how to specify the TRNTBL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   PFILE(PF1)
00020A            CHAR1               I     TRNTBL(LIB1/TBL1)
00030A            CHAR2             A I     TRNTBL(LIB2/TBL2)
00040A            NUM1
00050A            NUM2
     A
```

Figure 2-77. Specifying the TRNTBL Keyword

Field CHAR1 is translated using table TBL1 in library LIB1. Field CHAR2 is translated using table TBL2 in library LIB2. Field CHAR2 was redefined in the logical file as a character field (A in position 35) to allow the TRNTBL keyword to be specified. Fields NUM1 and NUM2 are numeric fields in the physical file PF1 and cannot have the TRNTBL keyword specified for them.

UNIQUE (Unique)

Use this file-level keyword to specify that records with duplicate key values are not allowed within a member of this physical or logical file. You can specify whether null key values are to be considered as duplicates using the parameter. Any

inserts or additions of new records, or updates to existing records, that would result in a duplicate key are rejected. The application program issuing the write or the update operation receives an error message. When a work station user is using DFU, a message is displayed at the work station. A copy file command that would copy records with duplicate keys in this file is not completed.

The format of this keyword is:

```
UNIQUE[(*INCNULL | *EXCNULL)]
```

The parameter is optional. When specified, it determines whether null key values cause duplicates. *INCNULL is the default and indicates to include null values when determining duplicates. *EXCNULL, when specified, indicates to exclude null values when determining duplicates.

When a logical file based on a physical file has the UNIQUE keyword, the physical file member or members cannot have duplicate key values.

When you specify the UNIQUE keyword for a physical or logical file, you must specify the MAINT(*IMMED) parameter value on the Create Physical File (CRTPF) or Create Logical File (CRTLF) command that creates the file. This means that the access path is maintained immediately as changes are made.

If you do not specify the UNIQUE keyword, records with duplicate key values are sequenced in the order you specify. If you specify the FIFO keyword, they are sequenced in first-in first-out order. If you specify the LIFO keyword, they are sequenced in last-in first-out order. If you specify the FCFO keyword, they are sequenced in first-changed first-out order. If you do not specify FIFO, LIFO or FCFO, the order in which the records are sequenced is not guaranteed. For an explanation of how records with duplicate key values are sequenced when those records are not in the same file member, see the section on database records in the *Database Guide*.

You cannot specify the UNIQUE keyword with the FIFO, LIFO, FCFO, or REFACCPTH keywords.

Figure 2-78 shows how to specify the UNIQUE keyword for a logical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A* SAMPLE LOGICAL FILE (CUSMSTL)
00030A*
00040A                                      UNIQUE
00050A          R CUSREC                    PFILE(CUSMSTP)
00060A                                      TEXT('Logical File Master Record')
00070A            CUST
00080A            NAME
00090A            ADDR
00100A          K CUST
     A
```

Figure 2-78. Specifying the UNIQUE Keyword

UNSIGNED (Unsigned)

Use this key field-level keyword to specify that numeric fields are sequenced as a string of unsigned binary data. Character, date, time, timestamp, and hexadecimal fields default to unsigned values.

This keyword has no parameters.

UNSIGNED is valid on key fields in physical or logical files regardless of the data type of the key field. The UNSIGNED keyword is not allowed with the SIGNED and ABSVAL keywords.

The UNSIGNED keyword will be the default in the following situations:

- When you specify ALTSEQ at the file level for a zoned key field
- When you specify ZONE or DIGIT for a zoned key field
- For all character and hexadecimal fields

Note: You can specify UNSIGNED for floating point fields, but the results cannot be predicted.

The following figure shows six records with a zoned decimal key field:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
1	98	F9F8
2	00	F0F0
3	98-	F9D8
4	97	F9F7
5	20	F2F0
6	99	F9F9

If you specify UNSIGNED, the records are sequenced in this order:

Record	Numeric Key Field (Zoned Decimal)	Hexadecimal Representation
2	00	F0F0
5	20	F2F0
3	98-	F9D8
4	97	F9F7
1	98	F9F8
6	99	F9F9

Figure 2-79 shows how to specify the UNSIGNED keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
 00010A         R RECORDA
 00020A           FLDA           7S 2
 00030A           FLDB           5
 00040A         K FLDA                        UNSIGNED
      A
```

Figure 2-79. Specifying the UNSIGNED Keyword

VALUES (Values)

Specify this keyword at the field level, the select/omit-field level, or both.

The format of the keyword is:

```
VALUES(value-1 [value-2...[value-100]])
```

Specifying VALUES at the Field Level

At the field level, this keyword specifies validity checking for the field you are defining when it is referred to later during display file creation.

VALUES does not affect the physical or logical file you are defining. When you define an input-capable field in a display file, you can refer to the field you are now defining by specifying R in position 29 and the REF or REFFLD keyword. During display file creation, the OS/400 program copies the VALUES keyword and other field attributes from the field in the physical or logical file into the field in the display file. You can override the VALUES keyword (as well as all other validity-checking keywords and the CHKMSGID keyword) by specifying any validity checking keyword for the field in the display file. See "Reference (Position 29)" on page 3-8 for details.

The rules for specifying this keyword in a physical or logical file are the same as for a display file. See "VALUES (Values)" on page 3-233 for more information on how to specify this keyword.

You cannot specify the VALUES keyword on a floating-point field (F in position 35) or a hexadecimal field (H in position 35). Do not specify the VALUES keyword on a date, time, or timestamp field (L, T, or Z in position 35).

Specifying VALUES at the Select/Omit-Field Level

At the select/omit-field level, this keyword selects or omits records retrieved from the physical file(s) when your program sends an input operation using the record format in which the select/omit field is specified.

The following rules apply:

- If the field you are defining is a character field, you must specify character strings or hexadecimal character strings.

  Specify character strings enclosed in apostrophes (see Figure 2-80 on page 2-83).

  Specify hexadecimal character strings as an X followed by a combination of the digits 0 through 9 and the letters A through F, enclosed in apostrophes. The number of hexadecimal digits in apostrophes must be exactly twice the specified length of the field. See Figure 2-81 on page 2-83.

- If the field you are defining is a numeric field, you must specify a numeric value (digits 0 through 9 specified without apostrophes). See Figure 2-80 on page 2-83.

- If you are defining a date field, specify a valid date in the same format specified on the DATFMT keyword and use the same separator as specified on the DATSEP keyword. For example, `VALUES('12/15/91' '12/31/91')` is the default value if *MDY is specified for DATFMT and '/' is specified for DATSEP.

- If you are defining a time field, specify a valid time in the same format specified on the TIMFMT keyword and use the same separator as specified on the TIMSEP keyword. For example, `VALUES('11.00.00' '12.00.00')` is the default value if *ISO is specified for TIMFMT. The default separator for *ISO is a period (.).

- If you are defining a timestamp field, you must specify the default value in the following format:

  ```
  VALUES('YYYY-MM-DD-HH.MM.SS.UUUUUU' 'YYYY-MM-DD-HH.MM.SS.UUUUUU')
  ```

The following examples show how to specify the VALUES keyword.

Figure 2-80 uses character and numeric values.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   PFILE(PF1)
00020A            FIELDA         1  0       VALUES(1 6 9)          1
00030A            FIELDB         1          VALUES('A' 'B' 'C')    1
00040A          K FIELDA
00050A          S FIELDB                    VALUES('A' 'B')        2
00060A          S FIELDA                    VALUES(1 6)            2
     A
```

Figure 2-80. Specifying the VALUES Keyword (Example 1)

1 VALUES is specified for FIELDA and FIELDB as a validity checking keyword for display files that refer to FIELDA and FIELDB.

2 VALUES is also specified for FIELDA and FIELDB as a select/omit keyword. Records from the physical file PF1 are retrieved through this logical file record format depending on the values of the following fields:

- FIELDB: Records are selected only when FIELDB equals A or B.
- FIELDA: Records not already selected are selected when FIELDA equals 1 or 6.

Figure 2-81 uses hexadecimal values.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD1                      PFILE(PF1)
00020A            CODEA
00030A            FLD1
00040A            FLD2
00050A          K FLD1
00060A          S CODEA                     VALUES(X'51' X'54' X'AE')
     A
```

Figure 2-81. Specifying the VALUES Keyword (Example 2)

VALUES is specified as a select/omit keyword for CODEA (which is a 1-byte field). Records from physical file PF1 are retrieved through this record format only if the value of field CODEA is hex 51, hex 54, or hex AE.

VARLEN (Variable-Length Field)

Use this field-level keyword to define this field as a variable-length field. Variable-length fields are useful for improving storage when the data for the field typically fits within a certain length, but can occasionally be longer. Specify the maximum length of the field in positions 30 to 34. You can specify the allocated length (or typical length) in the parameter.

The format of the keyword is:

```
VARLEN[(allocated-length)]
```

The allocated-length parameter is optional. Use it to specify the number of bytes allocated for the field in the fixed portion of the file. If you do not specify the allocated-length parameter, the data for this field is stored in the variable length portion of the file.

Valid values for the allocated-length parameter are 1 to the maximum length of the field specified in positions 30 to 34.

The VARLEN keyword has no parameters for a logical file.

The VARLEN keyword is valid only on character fields.

When you specify the VARLEN keyword, the maximum length you can specify in positions 30 to 34 is 32 740 (32 739 if the field allows the null value).

If you specify the DFT keyword for a variable-length field, the length of the default value must be less than or equal to the allocated length for the field. If the default value is longer than the allocated length, an error message is issued when the file is created.

If you specify a hexadecimal value as the default value for a variable-length field, the number of hexadecimal characters must be equal to two times the allocated length for the field.

The DFT keyword is not allowed on the same field as a VARLEN keyword unless you specify a value for the allocated-length parameter.

Do not specify the VARLEN keyword on a date, time, or timestamp field (L, T, or Z in position 35).

Figure 2-82 shows how to specify the VARLEN keyword for a physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A            FIELD1       100A         VARLEN(30)
00030A            FIELD2       200A         VARLEN
     A
```

Figure 2-82. Specifying the VARLEN Keyword

FIELD1 is defined as a variable-length field with a maximum length of 100 and an allocated length of 30. FIELD2 is defined as a variable-length field with a maximum length of 200 and no allocated length.

ZONE (Zone)

Use this key field-level keyword to specify that only the zone portion (farthest left 4 bits) of each byte of the key field is used when constructing a value associated with this key field. The digit portion is filled with zeros.

This keyword has no parameters.

This keyword is applied against the entire key field (not just a position within the field) and is valid only for character, hexadecimal, or zoned decimal type fields.

ZONE is not allowed with the ABSVAL, SIGNED, or DIGIT keywords.

If you specify ZONE for a key field, the value of the field is treated as a string of unsigned binary data rather than signed (which is the default for zoned decimal fields).

Figure 2-83 shows how to specify the ZONE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A         K CODE                       ZONE
     A
```

Figure 2-83. Specifying the ZONE Keyword

If CODE is a 1-byte field, the values of the field for three different records could be as follows:

Values	Hexadecimal	Digits Used for Key
A	C1	C
B	C2	C
E	C5	C

Chapter 3. Display Files

This chapter provides the following information regarding display files:

- Definition
- Positional entries
- Keyword entries

"Positional Entries (Positions 1 through 44)" on page 3-2 gives rules and examples for filling in positions 1 through 44 of the Data Description Specifications (DDS) form. "Keyword Entries (Positions 45 through 80)" on page 3-26 gives rules and examples for specifying DDS keywords. The keywords are described in alphabetical order.

For more information about choosing positional entries and keywords for display files, see the *Database Guide*.

Defining a Display File

Specify the entries in the following order to define a display file:

1. File-level entries (optional)
2. Record-level entries
3. Help-level entries (optional)
4. Field-level entries (optional)

Specify at least one record format in the file. The maximum number of record formats in a display file is 1024. The maximum number of fields in any one record format is 32 763. The maximum number of fields that can be displayed per record is 4095. The maximum combined length of all named fields and indicators in a record format is 32 763 bytes, regardless of the usage (I, O, B, M, H, P). For more information, see "Valid Entries for Usage" on page 3-22. Also, see the *Guide to Programming Displays* for the maximum number of input-capable fields.

Note: Specify the file name through the Create Display File (CRTDSPF) command, not through DDS.

You can find an explanation of file-, record-, help-, and field-level entries as well as syntax rules for specifying DDS keywords in Chapter 1, "Introduction." You can find complete display file examples in Appendix B, "Examples."

Figure 3-1 on page 3-2 shows a display file coding example.

© Copyright IBM Corp. 1991, 1993

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A* DISPLAY FILE EXAMPLE
00101A*
00102A                                        REF(PAYROLL)
00103A          R MENU
00104A          H                             HLPARA(1 1 12 80)
00105A                                        HLPRCD(RECORD1 FILEA)
00106A N01
00107AO 02        FLDA          20I 2O  2  2DSPATR(HI)
00108A            FLDB          22N 2B  3  2
00109A  72 73
00110AO 60 61 62
00111AA 63                                    DSPATR(HI)
00112A            FLDC           7Y 0B  7 20DSPATR(RI PC)
00113A  42 43
00114AO 60 61
00115AO 62                                    9  2'Constant'
00116A            FLDD      R               11  2
```

Figure 3-1. Display File Coding Example

Positional Entries (Positions 1 through 44)

This section describes how to specify the first 44 positions of the data description specifications (DDS) form for display files. See "Keyword Entries (Positions 45 through 80)" on page 3-26 to code the remaining parts of the form.

Figure 3-1 shows some positional entries for display files.

Sequence Number (Positions 1 through 5)

Use these positions to specify a sequence number for each line on the form. The sequence number is optional and is for documentation purposes only.

Form Type (Position 6)

Type an A in this position to designate this as a DDS form. The form type is optional and is for documentation purposes only.

Comment (Position 7)

Type an asterisk (*) in this position to identify this as a comment. Use positions 8 through 80 for comment text. A blank line (no characters specified in positions 7 through 80) is also treated as a comment. Comments can appear anywhere in DDS and are kept only in the source file. Comments are printed on the source computer printout but are not printed on the expanded source computer printout.

Conditioning (Positions 7 through 16)

Positions 7 through 16 are a multiple field area in which you can specify option indicators. Option indicators are 2-digit numbers from 01 to 99. Your program can set option indicators on (hex F1) or off (hex F0) to select a field or keyword. You can use option indicators to select fields to display different data on different output operations instead of defining a different record format for each combination of fields.

A condition is an ANDed grouping of two through nine indicators that must all be in effect (set off if N is specified; set on if N is not specified) before the field or

keyword is selected. You can specify a maximum of nine indicators for each condition and nine conditions for each field or keyword. Therefore, a maximum of 81 indicators can be specified for each field or keyword. An AND condition occurs when you specify a condition requiring that more than one indicator must be on or off before the condition is satisfied. The first indicator you specify, AND the second, AND the third, and so on, must all be in effect before the condition is satisfied and the field or the keyword is selected. You must specify the field or the keyword on the same line as the last (or only) set of indicators specified.

You can also specify several conditions for a field or keyword so that if any one of them is satisfied, the field or the keyword is selected. This is called an OR relationship. In an OR relationship, if the first condition is satisfied, OR the second condition, OR the third condition, and so on, the field or the keyword is selected. Conditions within the OR relationship can consist of just one indicator or can consist of several indicators ANDed together. Indicators can be ANDed to form a condition. Conditions can be ORed to give your program several ways to select the field or keyword.

Position 7 (AND) If you need more than three indicators to form an ANDed condition, specify the indicators on the next line or lines. You can specify an A in position 7 on the second or following lines to continue the ANDed condition, or you can leave it blank because A is the default.

Position 7 (OR) If you specify several conditions that are to be ORed together, each condition must start on a new line and each condition, except the first, must have an O in position 7. An O specified for the first condition produces a warning message, and that position is assumed to be blank.

Position 8, 11, 14 (NOT) If you want an indicator to be off instead of on to satisfy a condition, specify an N in the position just preceding the indicator (position 8, 11, or 14).

Conditioning a Field for More than One Keyword

If you are conditioning a field, the field name (or the constant) and the last (or only) indicator must be on the same line. If you do not select the field for an output operation, no keywords specified for that field are in effect, regardless of how the keywords are conditioned. For example, in Figure 3-1 on page 3-2, FLDA is selected if either indicator 01 is off or indicator 02 is on. If FLDA is not selected, any keyword associated with that field, such as DSPATR(HI), is ignored.

If you want to condition one or more keywords, the last (or only) indicator must appear on the same line as the keywords. If the conditioning applies to keywords on more than one line, you must use keyword continuation for the indicators to apply to all keywords. See Chapter 1, "Introduction," for more information about DDS syntax rules.

Display Size Condition Names: If you want your program to open this file to display devices with display sizes other than 24 lines x 80 characters, specify the DSPSIZ (Display Size) keyword at the file level. You can then condition the use of keywords and the location of fields with the "display size condition names" specified for the DSPSIZ keyword. If you do not specify the DSPSIZ keyword, your program can only open this file to display devices with a 24 x 80 display.

The following table shows the display size condition name for each display device.

Device	Display Size	Display Size Condition Name (See Note 1)
3179 3180 3196 3197 (Models C1 and C2) 3476 3487 (Models HA, HC, HG, and HW) 3488 (See Note 2) 3486 (Models BA and BG) 5251 (Models 11 and 12) 5291 5292	24 lines x 80 characters (1920 characters)	*DS3
3180 3197 (Models D1, D2, W1, and W2) 3477 (Models FA, FC, FD, and FG) 3487 (Models HA, HC, HG, and HW)	27 lines x 132 characters (3564 characters)	*DS4
Note 1	You can specify a user-defined display size condition name instead of *DS3, or *DS4. See the DSPSIZ keyword description for an explanation of how to specify user-defined condition names.	
Note 2	Dependent on monitor attached to display device.	

Figure 3-2 shows how to specify the DSPSIZ keyword and display size condition names.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                                  1           2
00010A                                    DSPSIZ(27 132 *LARGE 24 80 *NORMAL)
00020A          R RECORDA
00030A            FIELDA        10  0   1  2
00040A            FIELDB        10  0   1120
00050A  *NORMAL                           1 49
00060A            FIELDC        10  0  27  1
00070A  *NORMAL                          15  1
     A
```

Figure 3-2. Specifying the DSPSIZ Keyword and Display Size Condition Names

In Figure 3-2, the display size condition name for the primary display size is defined as *LARGE **1** (column 52 to 64) and the display size condition name for the secondary display size is defined as *NORMAL **2** (column 66 to 75). FIELDA appears on line 1, position 2 for both display sizes. FIELDB appears on line 1, position 120 for the primary display size (*LARGE by default), and on line 1, position 49 for the secondary display size (*NORMAL specified in positions 9 through 16). FIELDC appears on line 27, position 1 for the primary display size and on line 15, position 1 for the secondary display size. Only secondary display sizes (in this example, *NORMAL) can be used to condition field locations.

Use display size condition names similar to the way you use option indicators, except that display size condition names do not appear in your program and do not appear in the output record. A display size condition is on if the display file is

opened to the corresponding display size. When you use display size condition names, the following rules apply:

- Specify the DSPSIZ keyword to designate the primary display size and the secondary display size. If you do not specify the DSPSIZ keyword, the default is DSPSIZ(*DS3).
- You can specify only one display size condition name for a condition. You cannot specify AND or OR with other display size condition names or option indicators.
- The display size condition name must start in position 9.
- The display size condition name can be user-defined. See the keyword description for "DSPSIZ (Display Size)" on page 3-92 for more details.
- You can specify N in position 8 to designate a NOT condition (for the primary display size).

  Note: Specifying N in position 8 implies an OR relationship between the remaining display size condition names. For example, N*DS4 implies *DS3 when *DS3 is specified as a secondary display size on the DSPSIZ keyword.

- You must not use display size condition names that alter the line or position sequence of a field within a record. Fields are ordered in the display file by primary locations. A severe error occurs at file creation time if the secondary location alters this primary sequence.

  For example, FLD1 and FLD2 are on the primary display located on line 2, position 2 and line 4, position 2, respectively. You cannot use a display size condition name to display FLD2 ahead of FLD1 on the display (on line 1) for a secondary display size.

- When you specify the location of a field on a secondary display size, you can only specify positions 8 through 16 (conditioning) and 39 through 44 (location).
- If you do not specify a condition name for a keyword for which condition names are valid, the primary condition name specified on the DSPSIZ keyword is the default.

Figure 3-3 shows the correct and incorrect combinations of display size condition names and primary display sizes, when both display sizes are specified on the DSPSIZ keyword and the first one specified varies.

Figure 3-3 (Page 1 of 2). Valid Display Size Condition Specifications

Display Size Condition Name[1]	**24 x 80 DSPSIZ(*DS3...) or DSPSIZ(24 80...) Primary Display Size**	**27 x 132 DSPSIZ(*DS4...) or DSPSIZ(27 132...) Primary Display Size**
*DS3	Error[2]	Valid
*DS4	Valid	Error[2]
N*DS3	Valid	Error[3]

Figure 3-3 (Page 2 of 2). Valid Display Size Condition Specifications

Display Size Condition Name[1]	24 x 80 DSPSIZ(*DS3...) or DSPSIZ(24 80...) Primary Display Size	27 x 132 DSPSIZ(*DS4...) or DSPSIZ(27 132...) Primary Display Size
N*DS4	Error[3]	Valid

Notes:

1. See the DSPSIZ keyword description for user-defined names for these display size condition names.
2. The display size condition names are in error because that display size is the primary display size.
3. These display size condition names are in error because a primary and a secondary location are implied for the same display size. A condition name specified with the NOT condition implies an OR relationship. For example, N*DS4 implies *DS3.

Figure 3-4 and Figure 3-5 show display size conditioning for a keyword (in this case, MSGLOC, Message Location).

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A                                      DSPSIZ(*DS3 *DS4)
00040A  *DS4                                MSGLOC(26)
     A
```

Figure 3-4. Display Size Conditioning (Example 1)

In Figure 3-4, the display size condition name *DS4 is specified, so that the message line is line 26 for a 27 x 132 display and line 25 (the default) for a 24 x 80 display.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00080A                                      DSPSIZ(*DS4 *DS3)
00081A                                      MSGLOC(26)
     A
```

Figure 3-5. Display Size Conditioning (Example 2)

In Figure 3-5, the message line is also line 26 for the 27 x 132 display and line 25 (the default) for the 24 x 80 display, even though no display size condition name is specified, because the primary display size (*DS4) specified with the DSPSIZ keyword is the default.

Type of Name or Specification (Position 17)

Specify a value in this position to identify the type of name in positions 19 through 28. The valid entries for display files are:

Entry	Meaning
R	Record format name
H	Help specification
Blank	Field name

Figure 3-1 on page 3-2 shows you how to code the name type.

For more information on types of names, see the section, "Name (Positions 19 through 28)" on page 3-7. For more information on Help specifications, see "HELP (Help)" on page 3-120.

Reserved (Position 18)

This position does not apply to any file type. Leave this position blank unless you use it for comment text.

Name (Positions 19 through 28)

Use these positions to specify record format names and field names.

Refer to "Syntax Rules" on page 1-4 for rules to use when specifying record or field names in DDS.

The names must begin in position 19.

Figure 3-1 on page 3-2 shows you how to specify record format names and field names.

Record Format Name

When you specify R in position 17, the name specified in positions 19 through 28 is a record format name. You can specify more than one record format for a display file, but each record format name must be unique within that file.

Field Name

When position 17 is left blank, the name you specified in positions 19 through 28 is a field name. Field names must be unique within the record format.

Constant Fields

Constant fields are unnamed fields (positions 19 through 28 must be blank). The following rules apply to constant fields:

- Positions 17 through 38 must be blank.
- The location of the field is required (positions 39 through 44).
- The field can be conditioned using option indicators (positions 7 through 16).
- You can specify secondary display locations using display size condition names (positions 8 through 16). Only the display size condition name and location can be specified. That is, positions 7, 17 through 38, and 45 through 80 must be blank.
- The constant itself is defined in positions 45 through 80 using one of the following entries:
  - Explicit DFT keyword (specify the value within apostrophes with the DFT keyword)
  - Implicit DFT keyword (specify the value within apostrophes without the DFT keyword)
  - DATE keyword (specify no value; see the DATE keyword description)
  - TIME keyword (specify no value; see the TIME keyword description)
  - SYSNAME keyword (specify no value; see the SYSNAME keyword description)
  - USER keyword (specify no value; see the USER keyword description)
  - MSGCON keyword (specify the message description, the message file, the library name, and the length of the message description)

How to Determine the Order of Fields in a Record Format

The order of the name fields that you specify in a record format is the order in which the name fields appear in your program when it is compiled. (Unnamed fields do not appear in your program.)

The locations of named and unnamed fields you specify in positions 39 through 44 determine the order that the fields appear in the display. See "Location (Positions 39 through 44)" on page 3-24. Hidden fields (H in position 38) and Program-to-System fields (P in position 38) do not appear on the display.

Reference (Position 29)

Specify R in this position to use the reference function of the OS/400 program to copy the attributes of a previously defined named field (called the *referenced field*) to the field you are defining. The referenced field can be previously defined in either the display file you are defining or a previously created database file (the database file to be referred to is specified with the REF or REFFLD keyword). The field attributes referred to are the length, data type, and decimal positions of the field, as well as the ALIAS (alternative name) FLTPCN (floating-point precision), TEXT, editing, and validity checking keywords.

If you do not specify R, you cannot use the reference function for this field and you must specify field attributes for this field.

Position 29 must be blank at the file, record, and help levels.

If the name of the referenced field is the same as the field you are defining, you need only specify R in position 29 (in addition to specifying the name of the field you are defining in positions 19 through 28). If the name of the field you are defining is different from the name of the referenced field you must specify the name of the referenced field with the REFFLD (Referenced Field) keyword.

You can specify the name of the file defining the referenced field as a parameter value with the REF (Reference) or the REFFLD keyword. See "REF (Reference)" on page 3-183, "REFFLD (Referenced Field)" on page 3-184, and Appendix A, "When to Specify REF and REFFLD," for explanations of how the OS/400 program finds the referenced field.

You do not need to copy all attributes from the previously defined field to the field you are defining. To override specific attributes of the referenced field, specify those attributes for the field you are defining as follows:

- To override the EDTCDE (Edit Code) or EDTWRD (Edit Word) keywords, specify EDTCDE or EDTWRD for the field you are defining. You can delete these keywords by specifying the DLTEDT (Delete Edit) keyword for the field you are defining.
- To override the CHECK (Check), COMP (Comparison), RANGE (Range), and VALUES (Values) validity checking keywords and the CHKMSGID (Check Message Identifier) keyword, specify any validity checking keyword for the field you are defining. You can delete these keywords by specifying the DLTCHK (Delete Check) keyword for the field you are defining.

When you override some specifications, others are also affected, as follows:

- If you specify keyboard shift attribute, field length, or decimal positions for the field you are defining, neither editing nor validity checking keywords are copied from the referenced field.
- If you override the previously defined data type to character (by specifying M, A, X, or W in position 35), decimal positions are not copied. However, if you specify N, D, or I in position 35 and leave blanks in positions 36 and 37 (decimal positions), the field you define has the decimal positions of the referenced field. For D, the decimal positions must be zero.
- Packed decimal and binary fields are not supported for display files. Therefore, when you refer to fields of these types, the data type assigned is zoned decimal with a keyboard shift as follows:
  - If editing is in effect for the field you are defining, the keyboard shift is numeric only (Y in position 35).
  - If no editing is in effect for the field you are defining, the keyboard shift is signed numeric (S in position 35).
- When the referenced field contains the REFSHIFT (Reference Shift) keyword, the value specified for REFSHIFT is used as the display file keyboard shift. However, if the data type specified for the new field is not compatible with the keyboard shift specified on the REFSHIFT keyword, the keyword is not copied to the new field.

Note: Once the display file is created, you can delete or change the referenced file without affecting the field descriptions in the display file. Delete and create the display file again to incorporate changes made in the referenced file.

Length (Positions 30 through 34)

You must specify a length for each named field unless you are copying the length from a referenced field. The length is the number of bytes of data to pass to or receive from your program when I/O operations are done for the field. This is called the *program length* of the field.

The length of a field when it appears on the display is called the *display length.* The display length is greater than or equal to the program length. The display length of a field is determined by the keyboard shift (specified in position 35) and other field specifications such as decimal positions (positions 36 and 37) and editing functions. See "Data Type/Keyboard Shift (Position 35)" on page 3-10 for more information.

The display length does not include beginning and ending attribute characters of a field. However, you must consider these attribute characters when planning the display layout for field locations. Within a record, the ending attribute character of a field can overlap the beginning attribute character of the next field, requiring only one space between fields. See "Location (Positions 39 through 44)" on page 3-24 for more information.

The maximum length of a character field is equal to the display size minus one. (This allows space for the beginning attribute character.) The maximum length of a numeric (zoned decimal) field is 31 positions. The maximum length of a single-precision floating-point field is 9 digits. The maximum length of a double-precision floating-point field is 17 digits.

You must not specify a field length for a constant field. See the DATE, DFT (Default), MSGCON, or TIME keyword descriptions for explanations of the lengths of constant fields.

If you specify length, it must be right-justified. Leading zeros are optional. Figure 3-6 shows incorrect and correct field length specifications.

```
|...+....1....+....2....+....3....+....4....+....5
00010A            FIELD1       7
     A
00020A            FIELD2         7
     A
00030A            FIELD3    R   +7
     A
```

Note: FIELD1 shows the field length specified incorrectly. FIELD2 and FIELD3 show the field length specified correctly.

Figure 3-6. Incorrect and Correct Length Specifications

If you use a referenced field, you can override the length of the field by specifying a new value or by specifying the increase or decrease in length. To increase the length, specify +n, where n is the increase. To decrease the length, specify –n, where n is the decrease. For example, an entry of +4 for a numeric field indicates a field that is 4 digits longer than the referenced field.

In some cases, some keywords specified with the field in the database file are not included in the display file if you specify a value for length. For more information, see "Reference (Position 29)" on page 3-8.

A field cannot occupy the first position on the display. The first position is reserved for an attribute character. For example, on a 24 by 80 display, an entry of 1 in positions 39 through 41 (line), and 1 in positions 42 through 44 (position) is not allowed for a signed numeric field since the field starts in position 1.

Data Type/Keyboard Shift (Position 35)

The entry you make in position 35 is the data type/keyboard shift attribute for display files. This entry does not determine the data type of the field used in your program. The entry in positions 36 and 37 (decimal positions) determines the data type of the field. See "Decimal Positions (Positions 36 and 37)" on page 3-21.

The keyboard shift attribute automatically shifts 5250 work stations with data-entry keyboards and can, for all keyboards, limit what the work station user can type into a field. However, the keyboard shift attribute does not shift 5250 work stations with the typewriter-like keyboard. Nor does the keyboard shift attribute restrict in any way what your program can write to a field. Your program can write alphabetic characters to a numeric field and, in most cases, read that field and receive those characters. Any restrictions are enforced solely by the programming language used for your program.

Valid Entries for Display Files

Following are valid entries for display files:

Entry Keyboard Shifts	Meaning	Data Type Permitted
Blank	Default	
X	Alphabetic only	Character

Entry Keyboard Shifts	Meaning	Data Type Permitted
A	Alphanumeric shift	Character
N	Numeric shift	Character or numeric
S	Signed numeric	Numeric
Y	Numeric only	Numeric
W	Katakana (for Japan only)	Character
I	Inhibit keyboard entry	Character or numeric
D	Digits only	Character or numeric
M	Numeric only character	Character
Data Type		
F	Floating point	Numeric

Figure 3-1 on page 3-2 and Figure 3-7 on page 3-18 show you how to specify the keyboard shift attribute.

The keyboard shift attributes are defined in detail in the following sections.

Default (Blank): If you leave position 35 blank, the entry in positions 36 and 37 (decimal positions) determines the data type of the field as follows:

- If you make a valid entry in positions 36 and 37, the data type is zoned decimal and the keyboard shift attribute is signed numeric (S) unless you also specify an editing keyword. The keyboard shift attribute is numeric only (Y) when you also specify an editing keyword.
- If you make no entry in positions 36 and 37, the data type is character and the keyboard shift attribute is alphanumeric shift (A).

If you specify the REFSHIFT keyword for a referenced field, the specified value is used. Otherwise, a data type of packed or binary is converted to zoned decimal in the display file. Conversion to or from packed or binary can occur within your program.

Alphabetic Only (X): Both types of keyboards are in lowershift. Only the characters A through Z, comma (,), period (.), dash (–), and space ( ) can be typed in. When you type lowercase characters a through z, uppercase characters are sent to the program. See "LC (Lowercase)" on page 3-58 for an explanation of how to permit typing in lowercase characters for the typewriter-like keyboard on the 5250 work station.

Alphanumeric Shift (A): Both types of keyboards are in lowershift. All characters are valid for entry.

Numeric Shift (N): Each type of keyboard is shifted to allow numeric entry: uppershift for the data-entry keyboard and lowershift for the typewriter-like keyboard. All characters are valid for entry.

The display length for a numeric shift field is one more than the length coded in positions 30 through 34 when the following conditions occur:

- The field is an unedited, input-capable field.
- The value in the decimal positions field is greater than zero.

The extra position in the display length is for the decimal point.

Note: Input numeric shift fields with specified decimal positions (in positions 36 and 37) are processed by data management as numeric-only fields, except

that editing is not supported. For more information, see the keyboard attribute section, "Numeric Only (Y)" on page 3-13.

Signed Numeric (S): Each type of keyboard is shifted to allow numeric entry: uppershift for the data-entry keyboard and lowershift for the typewriter-like keyboard. You can only type the numbers 0 through 9 into the field (no blanks, no plus sign, no minus sign). To leave the field, press the Field Exit key, the Field+ key, the Field- key, or a cursor movement key. If you do not type any data into the field, you can press the Enter key.

You should consider the following differences when you choose between signed numeric (S) and numeric only (Y):

- Signed numeric restricts the characters that you can type into the field to the numbers 0 through 9.
- You cannot specify S in position 35 if you also specify the EDTCDE or EDTWRD keyword.
- Numeric-only performs character removal to remove nonnumeric characters; signed numeric prevents you from typing in these characters at all.

For input-capable fields only, the display length for the field is one more than the length specified in positions 30 through 34. The farthest right position on the display is reserved for a minus sign.

The following considerations apply when the OS/400 program passes the contents of a signed numeric field to your program:

- Your program always sees a numeric, right-justified, zero-filled field.
- The field is displayed as a right-justified, blank-filled field unless you specify CHECK(RZ). If you specify CHECK(RZ), the field is displayed as right-adjusted and zero-filled.
- The OS/400 program does not perform decimal alignment.
- The OS/400 program does not remove characters from the field (as it does for numeric only fields).

When an input-capable signed numeric field displays and you do not specify CHECK(RZ), the OS/400 program performs zero suppression by default (the EDTCDE and EDTWRD keywords are not valid for signed numeric fields).

Negative numbers are handled as follows:

- On input, you must type the number and press the Field- key. The number is right-justified in the displayed field with a minus sign in the farthest right position. The OS/400 program converts the farthest right significant digit to hex Dn where n is the significant digit, before passing the number to your program. For example, if you type 12345 and press the Field- key, 12345- is displayed and your program sees X'F1F2F3F4D5'.
- On output, the OS/400 program converts hex D in the farthest right digit to hex F. This changes the negative number to a positive number for display purposes and displays a minus sign in the farthest right (additional) position in the displayed field. For example, if your program sees X'F1F2F3F4D5', the number appears on the display as 12345-.

For examples of signed numeric fields, and sample data typed into them, see Figure 3-7 on page 3-18.

Numeric Only (Y): Each keyboard is shifted to allow numeric entry: uppershift for the data-entry keyboard and lowershift for the typewriter-like keyboard. You can only type the numbers 0 through 9, plus (+), minus (–), period (.), comma (,), and space ( ) into the field. You can press any key to leave the field.

The display length for a numeric-only field is one more than the program length when both of the following conditions occur (the program length is specified in positions 30 through 34):

- The field is an unedited, input-capable field.
- The value in positions 36 and 37 (decimal positions) is greater than zero.

The extra position in the display length is for the decimal point.

When the OS/400 program passes the contents of the field to your program, the following considerations apply:

- Your program sees a numeric, decimally aligned field.
- To type digits to the right of the decimal, positions 36 and 37 must be greater than zero and you must type the decimal character.
- You cannot type the maximum number of digits, a decimal character, and a sign character, because the display length of the field equals only the program length plus one. You can press the Field+ key or the Field– key to avoid typing a sign character.
- The OS/400 program removes all characters except 0 through 9 (whether typed or supplied through the EDTWRD keyword) and the sign.
- The OS/400 program converts embedded blanks (hex 40) to zeros (hex F0) before decimal alignment. (Embedded blanks are blanks between any significant digits in the field.) Leading blanks, trailing blanks, zeros, plus signs, and minus signs are not treated as significant digits. Embedded ampersands in an edit word are also converted to zeros before decimal alignment.
- All nonnumeric characters are removed before decimal alignment and validity checking for the RANGE, COMP, CMP, VALUES, CHECK(VN), CHECK(M10), CHECK(M11), and CHECK(VNE) keywords. Numeric characters (0 through 9) supplied by the EDTWRD keyword are not removed. Validity checking for the CHECK(M10F) and CHECK(M11F) keywords is performed before the nonnumeric characters are removed.
- The field length in the input buffer is the program length.

When the OS/400 program displays a numeric-only field, the EDTCDE or the EDTWRD keyword, if specified, applies. You can specify EDTCDE and EDTWRD only for numeric-only fields. The display length equals the program length plus the editing characters from the specified edit code or edit word.

Negative numbers are handled as follows:

- The user can type a negative number on input in one of two ways:
  - Type the digits, then a minus. The minus sign (–) appears (hex 60) on the display where it was typed in.
  - Type the digits, then press the Field– key.

If you do not specify CHECK(RZ) or CHECK(RB), a brace (}) is displayed in the farthest right position. This causes an error message to appear at the work station if you specify decimal positions other than zero in positions 36 and 37. If you specify CHECK(RZ) or CHECK(RB), the digits typed in are right-justified. No minus sign appears in either case.

If you specify an EDTCDE keyword that displays a minus sign and you do not specify CHECK(RZ) or CHECK(RB), a brace (}) is displayed in the farthest right position. This does not cause an error message to appear at the work station. A minus sign appears in the farthest right position on output.

If you specify an EDTCDE keyword that displays a minus sign and you also specify CHECK(RZ) or CHECK(RB), the farthest right significant digit is displayed as hex Dn (negative). A minus sign appears on the output.

When a negative number passes to your program, the OS/400 program converts the farthest right significant digit from hex Fn (positive) to hex Dn (negative), where n is the significant digit.

- The sign appears in the farthest right display position on output and takes up one of the positions in the display length. See the keywords "EDTCDE (Edit Code)" on page 3-99 and "EDTWRD (Edit Word)" on page 3-105.

Note: The OS/400 program examines each character of a numeric-only field to remove the nonnumeric characters plus sign (+), minus sign (–), comma (,), and decimal point(.) and nonsignificant digits, and to convert embedded blanks to zeros. This examination and removal can delay response time if a significant number of fields must be processed on an input operation.

Katakana (W): This field attribute designates the Japanese Katakana keyboard shift. All characters are valid for entry.

Inhibit Keyboard Entry (I): A field with this keyboard shift attribute does not accept data typed in from the keyboard, and an error is issued if you press any keys. You can press the field advance key to position the cursor at the start of the field. This field can be used to allow input from feature devices such as a light pen. The Field+, Field Exit, and Dup keys are valid for a field with this attribute, and function the same as if pressed in any input field for which the Display Attribute Protect (DSPATR(PR)) keyword is not in effect.

The display length for an inhibit keyboard entry field is one position greater than the length coded in positions 30 through 34 when the following conditions occur:

- The field is an unedited, input-capable field.
- The value in the decimal positions field is greater than zero.

The extra position in the display length is for the decimal point.

Digits Only (D): Each type of keyboard is shifted to allow numeric entry: uppershift for the data-entry keyboard and lowershift for the typewriter-like keyboard. The numbers-only keyboard shift defines a character or numeric field that allows you to key only the digits 0 through 9 into the field. You cannot key special characters or blanks.

The numbers-only keyboard shift is supported only on devices that are configured on the 6040 or 6041 local controller or the 5294 or 5394 Control Unit. When a digits-only field is sent to a device that is not configured on a valid controller type, the field is processed as alphanumeric (keyboard shift A). Since you can type any

alphanumeric character into the field, a decimal data error can result in the application program.

The Field Exit, Field+ Exit, and Dup keys are allowed. The Field+ Exit is processed as an unsigned Field Exit. The Field– Exit key is not allowed.

Blank and zero are the only supported values for decimal positions (DDS positions 36 and 37). If positions 36 and 37 are blank, the field is considered a character field. If you specify zero, the field is considered numeric.

You can only enter a positive integer value into a D field.

The display length of a digits-only field is always the field length as specified in positions 30 through 34.

Zero suppression is not supported for digits-only fields. EDTCDE and EDTWRD keywords are not valid, and the OS/400 program does not perform zero suppression by default, as it does for signed-numeric fields.

You cannot enter blanks into the field. However, you can move the cursor out of the field after entering part of it. If you move the cursor after entering part of the field, when the OS/400 program passes the contents of the field to a program, the following considerations apply:

- For a numeric field, leading blanks are converted by the OS/400 program to zero, and the field is right-justified prior to passing the field contents to the application program.
- For a character field, blanks are passed to the application program as hex 40, and the field is not right-justified.
- The field length in the input buffer is the program length.

In a database file, you can specify the D keyboard shift on the REFSHIFT keyword if the field data type is numeric or character (S, B, P, or A). For a numeric field, the number of decimal positions must be zero.

Numeric Only Character (M): Each type of keyboard is shifted to allow numeric entry: uppershift for the data-entry keyboard and lowershift for the typewriter-like keyboard. The M keyboard shift defines a character field that allows you to type only the digits 0 through 9, plus (+), minus (–), comma (,), period (.), and blank into the field.

The Field Exit, Field+ Exit, Field– Exit, and Dup keys are allowed. The Field+ Exit is processed as an unsigned Field Exit. The Field– Exit is processed as follows:

- If you do not specify CHECK(RZ) or CHECK(RB), the farthest right position is changed to a brace.
- If you specify CHECK(RZ) or CHECK(RB), the last character you type into the field must be a digit (or a keyboard error is issued). The farthest right digit, n, is converted from hex Fn (positive) to hex Dn (negative).

The display length for an M field is the length coded in positions 30 through 34. You must include any additional positions needed for a sign character or decimal point in the field length.

The field displays as a blank-filled field unless you specify a CHECK(RZ) or CHECK(RB) keyword, in which case the field displays as right-justified, zero, or blank filled respectively.

When the OS/400 program passes the contents of the field to a program, the following considerations apply:

- The program always sees a character field.
- The field length in the input buffer is the program length.
- The field contents are passed directly to the program. The OS/400 program neither converts embedded blanks to zeros nor removes nonnumeric characters, such as sign characters and decimal points.

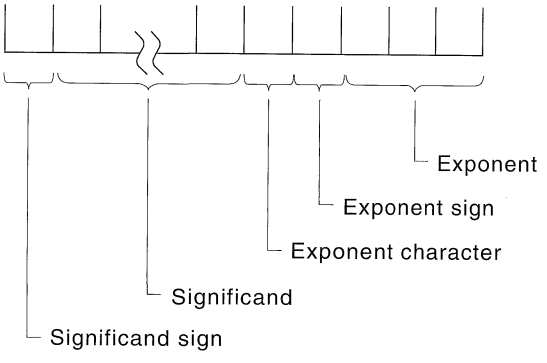
In a database file, you can specify the M keyboard shift on the REFSHIFT keyword if the field data type is character (A).

Floating Point (F): Each type of keyboard is shifted to allow numeric entry: uppershift for the data-entry keyboard and lowershift for the typewriter-like keyboard. You can type any combination of characters in a floating-point field (but only the digits 0 through 9, sign characters (+ or –), E or e, decimal point (.), and comma (,) are valid). An error message is issued if you type any other character in a floating-point field.

A floating-point value consists of five parts:

- Significand sign
- Significand
- Exponent character
- Exponent sign
- Exponent

The following illustration shows the five parts of a floating-point value.



RSLL613-0

The parts in the illustration are as follows:

- Significand sign
  - Use + for a positive value and - for a negative value.
  - On output, the significand sign is not displayed for a positive value.
  - On input, the significand sign is optional. If you do not type a + or –, the significand is assumed to be positive.

- Significand
  - The digits 0 through 9 and a decimal point (.) or a comma (,) are valid.
  - On output, the number of digits in the significand is determined by the length specified (positions 30 through 34). The location of the decimal point or the comma is determined by the decimal positions you specify (positions 36 and 37).
  - On input, you must type the significand. Only the digits 0 through 9 are valid. The decimal point or comma is optional. If you do not specify a decimal point, a decimal point is assumed on the right.
- Exponent character
  - E or e are valid.
  - On output, the exponent character always displays.
  - On input, you must type an exponent character if the floating-point value includes an exponent.
- Exponent sign
  - Use + for a positive value and – for a negative value.
  - On output, the exponent sign always displays.
  - On input, the exponent sign is optional. If you do not type a + or a –, the exponent is assumed to be positive.
- Exponent
  - The digits 0 through 9 are valid.
  - On output, the exponent is always 3 digits.
  - On input, you must type at least one digit if you type the exponent character (E or e). You can type a maximum of 3 digits.

Notes:

1. When a floating-point value displays, embedded blanks are removed. On input, you can type blanks before or after a floating-point value. Within a floating-point value, blanks are allowed between the significand and the exponent character.
2. If you do not type a value in a displayed floating-point field, a positive zero is assumed.
3. A value of negative zero is valid in a floating-point field. Only the first zero to the left of the decimal point displays. A minus sign displays to the left of the first zero.
4. A value of positive zero is valid in a floating-point field. The significand sign (+) does not display. Only the first zero to the left of the decimal point displays.
5. You can type a fixed-point value in a floating-point field.

The display length for a floating-point field is 7 positions greater than the length specified in positions 30 through 34. The 7 extra positions are for the significand sign, the decimal point or comma, the exponent character, the exponent sign, and the 3 exponent digits.

Figure 3-7 and Figure 3-8 show how the data you type is passed to your program.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A*
00200A* KEYBOARD SHIFT ATTRIBUTES
00300A            CHARA          5   I  2  2
00400A            CHARB          5   I  3  2CHECK(RB)
00500A            CHARC          5   I  4  2CHECK(RZ)
00600A            CHARD          5X  I  4 30
00700A            CHARE          5M  I  4 40
00800A            CHARF          5D  I  4 50
00900A            SIGN1          5  0I  5  2
01000A            SIGN2          5  2B  5 30
01100A            NBR1           5Y 0I  6  2
01200A            NBR2           5Y 2I  6 15
01300A            NBR3           5Y 2B  6 30EDTCDE(L)
01400A            NBR4           5N 2I  6 40
01500A            NBRZ           5   H
01600A            FLPT           7F 4I  7  2
     A
```

Figure 3-7. Data Type and Keyboard Shift Coding Example

Three special characters are used in Figure 3-8.

- _ means that you did not type in any character.
- X indicates a blank.
- } is represented internally as hex D0.

Except where indicated, you enter the data only by pressing a command function key. The fourth entry under SIGN1 is an exception. You make that entry by pressing the Field Exit key.

The following table refers to fields defined in Figure 3-7.

Figure 3-8 (Page 1 of 3). Example Data Type and Keyboard Shift Coding Instructions

Field Name (Keyboard Shift)	As Typed in by the Work Station User	As Passed to Your Program
CHARA (Alphanumeric Shift)	1. _ _ _ _ _	x x x x x (X'40')
	2. A B C _ _	A B C x x
	3. A _ C _ _	A x C x x
	4. _ _ _ D E	x x x D E
CHARB (Alphanumeric Shift)	1. _ _ _ _ _	x x x x x
	2. A B C _ _	x x A B C
	3. A _ C _ _	x x A x C
	4. _ _ _ D E	x x x D E
CHARC (Alphanumeric Shift)	1. _ _ _ _ _	0 0 0 0 0
	2. A B C _ _	0 0 A B C
	3. A _ C _ _	0 0 A x C
	4. _ _ _ D E	0 0 0 D E
CHARD (Alphabetic Only)	1. _ _ _ _ _	x x x x x
	2. A B C _ _	A B C x x
	3. 4 _ _ _ _	error message
	4. A B C. $ _	error message

Figure 3-8 (Page 2 of 3). Example Data Type and Keyboard Shift Coding Instructions

Field Name (Keyboard Shift)	As Typed in by the Work Station User	As Passed to Your Program
CHARE (Numeric Only Character)	1. _ _ _ _ _ 2. 516.7 3. 5, 2 _ _ 4. A _ _ _ _	x x x x x 516.7 5, 2 x x error message
CHARF (Digits Only)	1. _ _ _ _ _ 2. 2 3 _ 5 _ 3. 1 2 _ _ _ 4. A _ _ _ _	x x x x x 2 3 x 5 x 1 2 x x x error message
SIGN1 (Signed Numeric)	1. _ _ _ _ _ _ 2. 1 2 3 _ _ _ 3. 1 _ 3 _ _ _ 4. _ _ _ 4 5 _ (Field Exit key) 5. _ _ _ 4 5 (Field + key) 6. _ _ _ 4 5 _ (Field – key) 7. 1 2 3 4 5 _	0 0 0 0 0 0 0 1 2 3 0 0 1 0 3 0 0 0 4 5 0 0 0 4 5 0 0 0 4 N (X'F0F0F0F4D5') 1 2 3 4 5
SIGN2 (Signed Numeric)	1. _ _ _ _ _ _ 2. 1 2 3 4 _ _ 3. 1 2 _ _ _ _ 4. 1 2 _ _ _ _ (Field – key)	0 0 0 0 0 0 1 2 3 4 0 0 0 1 2 0 0 0 1 K (X'F0F0F0F1D2')
NBR1 (Numeric Only)	1. _ _ _ _ _ 2. 0 0 0 0 5 3. 0 0 0 5 _ 4. 0 0 2 _ _	0 0 0 0 0 0 0 0 0 5 0 0 0 0 5 0 0 0 0 2
NBR2 (Numeric Only)	1. _ _ _ _ _ _ 2. 0 0 0 0 5 _ 3. 0 0 5 _ _ _ 4. 5 _ _ _ _ _ 5. 0 5 0 0 _ _ 6. 5 x 5 _ _ 7. 5 x x _ _ _ 8. 5 5 . 1 _ _ 9. 5 0 0 0 0 . _ 10. 5 0. 0 0 0 11. 5 5 – _ _ _ 12. 5 _ – – _ _ 13. 5 _ + _ – _	0 0 0 0 0 0 0 5 0 0 0 0 5 0 0 0 0 5 0 0 5 0 0 0 0 5 0 5 0 0 0 0 5 0 0 0 5 5 1 0 error message (use of decimal error message not valid) 0 5 5 0 } 0 0 5 0 } 0 0 5 0 }
NBR3 (Numeric Only)	Input processing is the same as for NBR2.	
NBR4 (Numeric Shift)	1. _ _ _ _ _ _ 2. 5 – _ _ _ _ 3. _ 5 _ – _ _ 4. 5 _ + _ _ _ 5. 5 _ A B C _ 6. 5 _ K K _ _ 7. 5 _ K A K _ 8. 5 _ K K A _ 9. 1 0 E + 0 3	0 0 0 0 0 0 0 5 0 } 0 0 5 0 } 0 0 5 0 0 0 0 5 0 0 5 0 2 0 } See note. 5 0 2 0 } See note. 0 0 5 0 0 See note. 1 0 0 3

Figure 3-8 (Page 3 of 3). Example Data Type and Keyboard Shift Coding Instructions

Field Name (Keyboard Shift)	As Typed in by the Work Station User	As Passed to Your Program
NBRZ (Hidden Field)	This is a hidden field and does not appear on the display.	
FLPT (Floating point)	1. _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _	+ 0
	2. – 9 9 . 2 _ _ _ _ _ _ _ _ _	– 9 9 . 2
	3. – 9 9 E 0 2 _ _ _ _ _ _ _ _	– 9 9 0 0 .
	4. + 9 9 9 . 9 9 9 9 E + 0 0 3	+ 9 9 9 9 9 9 . 9
	5. A B C _ _ _ _ _ _ _ _ _ _ _ _	error message
	6. _ _ 9 9 0 _ e _ _ _ _ _ _ _	error message

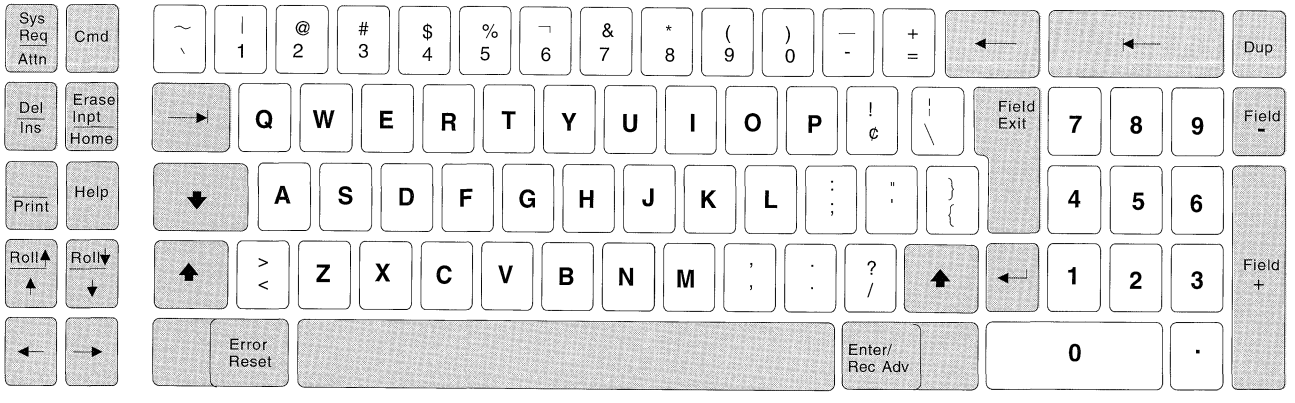
Note: The internal representation of K is hex D2. All nonnumeric characters (including those with hex D in the zone portion) are deleted with no place value. For example, 5_KAK_ becomes 5020}.

Keyboard Types: There are two keyboards on the AS/400 system, a **typewriter-like** keyboard and a **data-entry** keyboard. Display stations can have either the typewriter-like or the data-entry keyboard.

Typewriter-Like Keyboard: The typewriter-like keyboard functions in either uppershift or lowershift. Type the upper symbol (for the keys with two symbols) when in uppershift. Type the lower symbol (for the keys with two symbols) when the keyboard is in lowershift. Type uppercase characters for alphabetic keys (which have only one symbol) when the keyboard is in uppershift. Type uppercase alphabetic characters when the keyboard is in lowershift, unless the Check Lowercase (CHECK(LC)) keyword is specified. If the CHECK(LC) keyword is specified, and you place (or leave) the keyboard in lowershift, you can type lowercase a through z characters.

Note: None of the keyboard shift attributes causes an automatic uppershift for the typewriter-like keyboard.

The following illustration shows the typewriter-like keyboard.



RV2F107-0

Data-Entry Keyboard: The data-entry keyboard functions in either numeric shift (upper) or alphabetic shift (lower). The upper symbol (for the keys with two symbols) is typed when the keyboard is in uppershift. On this keyboard, the numbers 0 through 9 are the upper symbols on alphabetic keys. The lower symbol

(for the keys with two symbols) is typed when the keyboard is in lowershift. The alphabetic characters A through Z are the lower symbols and are always in uppercase. The data-entry keyboard does not support lowercase characters a through z, even if you specify CHECK(LC) keyword.

The following illustrations shows the data-entry keyboard.

RSLL616-0

Note that you type the numbers 0 through 9 by using the lowershift on the typewriter-like keyboard and by using the uppershift (numeric) on the data-entry keyboard. Therefore, when a field has one of the numeric keyboard shift attributes (numeric shift or numeric only), the typewriter-like keyboard is in lowershift and the data-entry keyboard is in uppershift. In both cases, you can type numeric characters without pressing a shift key.

Decimal Positions (Positions 36 and 37)

Use these positions to specify the decimal placement within a zoned decimal field and also to specify the data type of the field as it appears in your program.

If you leave these positions blank, the OS/400 program assigns a data type of character for the field. If you type in a number in these positions, the OS/400 program assigns a data type of zoned decimal for the field. The number you specify is the number of positions to the right of the decimal point. The entry must be less than or equal to the field length, with a maximum of 31 positions.

Packed decimal and binary fields are not supported for display files. Therefore, when you refer to fields of these types using the reference function, the data type assigned is zoned decimal with a keyboard shift as follows:

- If editing is in effect for the field you are defining, the keyboard shift is numeric only (Y in position 35).
- If no editing is in effect for the field you are defining, the keyboard shift is signed numeric (S in position 35).

If you are using a referenced field, you can override or change these positions. Specify the new value to override decimal positions. To change decimal positions, specify the amount you want the field increased or decreased and precede that number with either a plus (+) or minus (–) sign. For example, an entry of +4 indicates four more digits to the right of the decimal point than in the referenced field.

Figure 3-7 on page 3-18 shows how to specify the decimal positions field.

Usage (Position 38)

Use this position to specify that a named field is an output-only, input-only, input/output (both), hidden, program-to-system, or message field. Make no entry in this position for a constant (unnamed) field.

Valid Entries for Usage

The valid entries for display files are:

Entry	Meaning
Blank or O	Output only
I	Input only
B	Output/input (both)
H	Hidden (special output/input field)
M	Message (special output field)
P	Program-to-system (special output field)

Notes:

1. Input-only and output/input fields are input-capable fields.
2. Output-only and output/input fields are output-capable fields.
3. Output-only is the default if you leave the position blank.

Output-only: Output-only fields pass data from a program to the device when the program writes a record to a display. You can use the DFT (Default) keyword to specify an initial value for a named output field if you also specify the OVRDTA keyword for the field. If the OVRDTA keyword is not in effect, the initial value for the field is used. If the OVRDTA keyword is in effect, the data for the field is used. If the OVRDTA keyword is in effect, the data for the field is taken from the output buffer.

Input-only: Input-only fields pass data from the device to a program when the program reads a record. Input fields can be initialized with a default value (specified on the DFT keyword). If you do not change the field and the field is selected for input, the default value is passed to the program. Input fields are, by default, underlined on the display. You can use the Change Input Default (CHGINPDFT) keyword or the Display Attribute Underline (DSPATR(UL)) keyword to prevent underlining.

Output/Input: Output/Input fields are passed from a program when the program writes a record to the display and are passed to a program when the program reads a record from the display and the field is selected for input. Output/input fields are usually used when the program displays data that you can change. An initial value can be specified for the field on the DFT keyword. When DFT is specified, the OVRDTA keyword is also required and indicates whether the data displayed in the field is taken from the output buffer (OVRDTA in effect) or from the DFT keyword (OVRDTA not in effect). Output/input fields are, by default, underlined on the display.

Hidden: A hidden field is a named, numeric, or alphanumeric field that does not appear on the display. Your program can send data to the field with an output operation, and it can retrieve data from the field with an input operation, but you cannot see or change the contents of the field.

The following rules apply to hidden fields:

- Hidden fields are always named.
- Locations are not valid for hidden fields.
- Specify length, data type, and decimal positions as you would for other named fields.
- You can specify more than one hidden field for a display file.

Since hidden fields are not displayed, they are not considered input-capable or output-capable fields, even though your program can send and receive data from them.

Hidden fields are useful in applications involving subfiles. For example, a subfile record can contain record key information in a hidden field. You cannot see the hidden field, but the field is returned to the program with the subfile record so that the program can return the record to the database.

Message Field: A message field is a named, output-only, character field.

The following rules apply to message fields:

- You can use option indicators to select message fields, but during processing, only one message can be displayed at a time. The message from the first message field selected is displayed, and all others are ignored for that operation.
- When a message field displays, all other fields you specify for that record are processed in the normal manner. The device goes into an error condition (locked keyboard, blinking cursor, and message displayed with the high intensity (HI) display attribute). When you press the Reset key, normal processing continues.
- The text of the message is established when your program moves a value to the message field.
- The location of the message on the display is the message line (the last line on the display unless the MSGLOC keyword is in effect).
- The length you specify for the message field should be less than 79 positions for 24 x 80 work stations, or less than 131 positions for the 27 x 132 work station. Any message text that occupies more than 78 positions on the 24 x 80 work stations, or more than 130 positions on a 27 x 132 work station is truncated to fit the message line.
- The Help key is not supported for message fields. Message help for the message is not displayed when Help is pressed.
- Only the following keywords are valid for a message field:

ALIAS	REFFLD
INDTXT	TEXT
OVRDTA	

- You cannot specify M in position 38 for a field if the field is part of the subfile record format.

Note: It is valid to send an input operation to a record that contains no input-capable fields. This permits pressing a function key as a response to an output record.

Program-to-System: A program-to-system field is a named, numeric, or alphanumeric output-only field that is used to pass data between the program and the system; it does not appear on the display. Your program can send data to the field with an output operation, but the work station user cannot see the contents of the field. Since program-to-system fields are not displayed, they are not considered output-capable fields, even though the program can send data to them.

The following rules apply to program-to-system fields in display files:

- Program-to-system fields are always named.
- Locations are not valid for program-to-system fields.
- Specify length, data-type, and decimal positions as you would for other named fields.
- The program-to-system field must be specified as a parameter on a CHCACCEL, CHCCTL, CHKMSGID, CHOICE, ERRMSGID, MNUBARCHC, MSGID, SFLMSGID, or WINDOW keyword within the same record format. The P-usage field is not valid as a parameter on any other keyword. A severe error is sent if the field is not specified on at least one of these keywords.
- Unlike the P-usage fields in ICF files, P-usage fields in display files can appear anywhere in the buffer. In ICF files, P-fields must be specified after all the data fields (B-usage fields).
- A P-usage field can be specified as the message-identifier, message-file, or library name on a MSGID keyword, provided the field is defined with the proper attributes such as length. For more information, see "MSGID (Message Identifier)" on page 3-164.

The only keywords allowed on a program-to-system field are:

ALIAS
INDTXT
REFFLD
OVRDTA
TEXT

Location (Positions 39 through 44)

Use these positions to specify the exact location on the display where each field begins. You cannot specify a location for hidden, program-to-system, or message fields. The validity of the location is based on the DSPSIZ keyword and the display size condition names. See "DSPSIZ (Display Size)" on page 3-92 for specific examples.

Line (Positions 39 through 41): Use these positions to specify the line on which the field begins. The entry must be right-justified. Leading zeros are optional. The maximum number of lines is 24 or 27. See Figure 3-3 on page 3-5 for more information on display size condition names.

Position (Positions 42 through 44): Use these positions to specify the starting position of the field within the line you specified. Your entry must be right-justified. Leading zeros are optional. The maximum position is 132 for the 3180 device and 80 for all of the remaining display devices.

For fields other than the first field within the record, you can specify the location by specifying a plus value (+n) for positions 42 through 44. The plus value indicates the number of spaces to be left between the end of the previous field and the start

of the field you are defining. The plus value must be in the range of 1 through 99. (A plus value of zero is not valid.)

Beginning Attribute Character: Each field displayed has one attribute character that defines the display attribute of the field on the display. This attribute character is not displayed, but occupies one position on the display immediately preceding the field. Because of the beginning attribute character, you cannot specify that a field is to begin in the first position of the display (line 1, position 1), unless the SLNO (Starting Line Number) keyword is specified and the start line number is greater than 1.

When a field begins in position 1 the beginning attribute character occupies the last position of the preceding line. If such a field is the first field of a record, the preceding line is a part of the record area and displayed as a blank line. Any record format using that line cannot be displayed at the same time as the other record. The last one to be displayed causes the other one to be deleted (unless CLRL(*NO) is specified for the last displayed record).

Ending Attribute Character: The end of a field on the display is indicated by an ending attribute character, unless there is only one space between that field and the next field. In that case, the beginning attribute character of the second field serves as the ending attribute character of the first field. In any case, there must always be at least one space left between fields in a record. When the last position occupied by a field of a record is the last position in a line, the ending attribute character for the field is in position 1 of the next line. However, the next line is not considered part of the first record, and records can be displayed on both lines at once.

Overlapping Fields: Within a record format, you can define fields to overlap portions of other fields or their attribute characters; however, only one of those fields is shown on the display at a time. At run time, when processing overlapping fields within a record, the OS/400 program looks at the fields in line and position sequence. When the OS/400 program finds a field whose conditioning is satisfied or that does not have an option indicator specified, it selects that field for display and ignores the remaining overlapping fields. The first overlapping field that does not have an option indicator specified always stops the search, and any subsequent overlapping fields are never displayed. In the following example, if indicator 01 is set on, FIELD1 is the only field displayed. If indicator 01 is off and indicator 02 is on, FIELD2 is the only field displayed. FIELD3 is displayed when neither of the others is selected.

Figure 3-9 shows how to define overlapping fields.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  01        FIELD1        10      1  5TEXT('ONE')
00020A  02        FIELD2         5      1  5TEXT('TWO')
00030A            FIELD3         2  0   1  5TEXT('THREE')
     A
```

Figure 3-9. Specifying Overlapping Fields

If used incorrectly, this capability can result in problems in user and program communication. In the following example, only one input field (FIELD4) is specified for the record, and according to the field location specification, this field overlaps a preceding output field. The work station user is not able to type in any data because

FIELD1 is always the field selected for display. The other three fields, including FIELD4, are never displayed.

Figure 3-10 shows an example of incorrect field specification entry.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            FIELD1        10      1  5TEXT('ONE')
00020A  21        FIELD2         5      1  5TEXT('TWO')
00030A  12        FIELD3         2  0   1  5TEXT('THREE')
00040A            FIELD4         5Y 2I  1  5TEXT('FOUR')
     A
```

Figure 3-10. Incorrect Field Specification

Display Length: The display length is increased for certain types of fields and must be considered when laying out the display. The display length is increased by the OS/400 program in the following situations:

- For numeric-only fields with editing, the display length is determined from the edit word or the program length and the edit code.
- For input-capable signed numeric fields, the display length is one more than the program length.
- The display length is one more than the program length for numeric shift fields and for numeric-only fields without editing when these fields are input-capable and have decimal positions greater than zero.
- The display length for floating-point fields is 7 more than the length specified in positions 30 through 34. The 7 extra positions are for the significand sign, the decimal point or comma, the exponent character, the exponent sign, and the 3 exponent digits.

For an unsigned numeric field (like FIELD4 in Figure 3-10) with a nonzero decimal position, the system requires a decimal character to be typed into the field when decimal values are typed in as data. In Figure 3-9 on page 3-25, 123 in FIELD4 does not require a decimal character, but 1234 does (123.4). For this field, the display length is 6.

Keyword Entries (Positions 45 through 80)

This section contains the valid keyword entries for defining display files. They are typed in positions 45 through 80 (functions). See "Syntax Rules" on page 1-4 for a discussion of the general rules for specifying keywords.

The following keywords are valid for display files:

ALARM
ALIAS
ALTHELP
ALTNAME
ALTPAGEDWN
ALTPAGEUP
ALWGPH
ALWROL
ASSUME
AUTO
BLANKS
BLINK
BLKFOLD
CAnn
CFnn
CHANGE
CHCACCEL
CHCAVAIL
CHCCTL
CHCSLT
CHCUNAVAIL
CHECK
CHGINPDFT
CHKMSGID
CHOICE
CHRID
CLEAR
CLRL
CMP
CNTFLD
COLOR
COMP
CSRLOC
DATE
DFT
DFTVAL
DLTCHK
DLTEDT
DSPATR
DSPMOD
DSPRL
DSPSIZ
DUP
EDTCDE
EDTMSK
EDTWRD
ENTFLDATR
ERASE
ERASEINP
ERRMSG
ERRMSGID
ERRSFL
FLDCSRPRG
FLTFIXDEC
FLTPCN
FRCDTA
GETRETAIN
HELP
HLPARA
HLPBDY
HLPCLR
HLPCMDKEY
HLPDOC
HLPEXCLD
HLPFULL
HLPID
HLPPNLGRP
HLPRCD
HLPRTN
HLPSCHIDX
HLPSEQ
HLPTITLE
HOME
INDARA
INDTXT
INVITE
INZINP
INZRCD
KEEP
LOCK
LOGINP
LOGOUT
LOWER
MDTOFF
MLTCHCFLD
MNUBAR
MNUBARCHC
MNUBARDSP
MNUBARSEP
MNUBARSW
MNUCNL
MSGALARM
MSGCON
MSGID
MSGLOC
NOCCSID
OPENPRT
OVERLAY
OVRATR
OVRDTA
PAGEDOWN
PAGEUP
PASSRCD
PRINT
PROTECT
PULLDOWN
PUTOVR
PUTRETAIN
RANGE
REF
REFFLD
RETKEY
RETCMDKEY
RETLCKSTS
RMVWDW
ROLLDOWN
ROLLUP
RTNCSRLOC
RTNDTA
SETOF
SETOFF
SFL
SFLCLR
SFLCSRPRG
SFLCSRRRN
SFLCTL
SFLDLT
SFLDROP
SFLDSP
SFLDSPCTL
SFLEND
SFLENTER
SFLFOLD
SFLINZ
SFLLIN
SFLMODE
SFLMSG
SFLMSGID
SFLMSGKEY
SFLMSGRCD
SFLNXTCHG
SFLPAG
SFLPGMQ
SFLRCDNBR
SFLRNA
SFLROLVAL
SFLSIZ
SLNO
SNGCHCFLD
SYSNAME
TEXT
TIME
UNLOCK
USER
USRDFN
USRDSPMGT
USRRSTDSP
VALUES
VLDCMDKEY
WDWBORDER
WINDOW

ALARM (Audible Alarm)

Use this record-level keyword to specify that the OS/400 program is to activate the audible alarm when this record is displayed. The alarm is of short duration.

This keyword has no parameters.

If the Error Message (ERRMSG) or Error Message Identifier (ERRMSGID) keyword is in effect, the ALARM keyword has no effect, even if also selected.

To sound the audible alarm when an active ERRMSG or ERRMSGID keyword is on the record, see "MSGALARM (Message Alarm)" on page 3-162.

Option indicators are valid for this keyword.

Figure 3-11 shows how to specify the ALARM keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00005A          R CUST
00010A  01                                  ALARM
     A
```

Figure 3-11. Specifying the ALARM Keyword

ALIAS (Alternative Name)

Use this field-level keyword to specify an alternative name for a field. When the program is compiled, the alternative name is brought into the program instead of the DDS field name. The high-level language compiler in use determines if the ALIAS name is used. Refer to the appropriate high-level language reference manual for information about ALIAS support for that language.

The format of the keyword is:

```
ALIAS(alternative-name)
```

Refer to "Syntax Rules" on page 1-4 for ALIAS naming conventions.

The alternative-name must be different from all other alternative names and from all DDS field names in the record format. If a duplicate name is found, an error appears on the field name or alternative name.

An alternative name cannot be used within DDS or any other OS/400 function (for example, as a key field name, as the field name specified for the REFFLD keyword, or as a field name used in the Copy File (CPYF) command).

When you refer to a field that has the ALIAS keyword, the ALIAS keyword is copied in unless the ALIAS keyword is explicitly specified on the referencing field.

Option indicators are not valid for this keyword.

Figure 3-12 on page 3-29 shows how to specify the ALIAS keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00070A            FIELDA        25A     1  2ALIAS(CUSTOMERNAME)
     A
```

Figure 3-12. Specifying the ALIAS Keyword

The alternative name for FIELDA is CUSTOMERNAME.

ALTHELP (Alternative Help Key)

Use this file-level keyword to assign a command attention (CA) key as an alternative Help key. When either the Help key or the CA key is pressed, the help function is called.

The format of the keyword is:

```
ALTHELP[(CAnn)]
```

The valid values for the optional parameter are CA01 through CA24. If the parameter is not specified, CA01 is the default.

The HELP keyword must also be specified, either at the file level or on at least one record in the file. ALTHELP applies only to records for which the HELP keyword also applies. If HELP is specified at the file level, it applies to all records in the file; thus, ALTHELP also applies to all records in the file. If HELP is specified at the record level, ALTHELP applies only to those records that have the HELP keyword specified. Refer to "HELP (Help)" on page 3-120 for information on how to specify the HELP keyword.

If you specify a response indicator on the HELP keyword, the response indicator is set on and control is returned to your program when either the Help key or the CAnn key is pressed.

If you specify option indicators on the HELP keyword, the same option indicators apply to the ALTHELP keyword. That is, both the Help key and the CAnn key are active when the indicator condition is true.

The following cannot be specified in a file with an ALTHELP keyword that has no parameter (CA01 default):

ALTPAGEDWN(CF01)	SFLDROP(CA01)
ALTPAGEUP(CF01)	SFLDROP(CF01)
CA01	SFLENTER(CA01)
CF01	SFLENTER(CF01)
MNUCNL(CA01)	SFLFOLD(CA01)
MNUBARSW(CA01)	SFLFOLD(CF01)

Similarly, the following cannot be specified in a file with ALTHELP(CAnn) (where nn is the same number):

ALTPAGEDWN(CFnn)	SFLDROP(CAnn)
ALTPAGEUP(CFnn)	SFLDROP(CFnn)
CAnn	SFLENTER(CAnn)
CFnn	SFLENTER(CFnn)
MNUCNL(CAnn)	SFLFOLD(CAnn)
MNUBARSW(CAnn)	SFLFOLD(CFnn)

You cannot specify RETKEY or RETCMDKEY in a file with the ALTHELP keyword.

Option indicators are not valid for this keyword.

Figure 3-13 shows how to specify the ALTHELP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     ALTHELP
     A                                     HELP(01 'HELP KEY PRESSED')
     A          R RECORD
     A            FIELD1        20A     5  5
     A
```

Figure 3-13. Specifying the ALTHELP Keyword

The CA01 key is an alternative Help key. If the user presses the CA01 key or the Help key, response indicator 01 is set on and control returns to the application program.

ALTNAME (Alternative Record Name)

Use this record-level keyword to specify an alternative name for a record. The alternative name can be specified for I/O operations when using program-described files. The syntax of the alternative record name must be valid for the high-level language compiler in use.

The format of the keyword is:

```
ALTNAME('alternative-name')
```

Refer to Appendix F, "System/36 Environment Considerations," for information on how to specify the ALTNAME keyword.

ALTPAGEDWN/ALTPAGEUP (Alternative Page Down/Alternative Page Up)

Use these file-level keywords to assign command function (CF) keys as alternative Page Down/Page Up keys. When either the Page keys or the CF keys are pressed, the page function is called. The alternative Page Down/Page Up keys function only on the displays defined in the file with the ALTPAGEDWN/ALTPAGEUP keyword. They do not function on system displays such as message help.

The format for each of these keywords is:

```
ALTPAGEDWN[(CFnn)]
ALTPAGEUP[(CFnn)]
```

The valid values for the optional parameters are CF01 through CF24. If the parameter is not specified, CF08 is the default for ALTPAGEDWN and CF07 is the default for ALTPAGEUP.

The PAGEDOWN/PAGEUP keyword should also be specified if you want your program to handle any situation where the user has pressed a Page key or the alternative CFnn key and the OS/400 program cannot move the text lines on the display. If you do not specify the PAGEDOWN/PAGEUP keyword, a message indicating the key is not valid is displayed when either a Page key or the alternative CFnn key is pressed and the OS/400 program cannot move the text lines on the

display. Refer to the PAGEDOWN/PAGEUP keywords for more information on how to specify PAGEDOWN/PAGEUP.

Note: Throughout this keyword description, PAGEDOWN means the PAGEDOWN or the ROLLUP keyword. PAGEUP means the PAGEUP or the ROLLDOWN keyword.

If you specify a response indicator on the PAGEDOWN or PAGEUP keyword, the response indicator is set on and returned to your program when either the Page key or the alternative CFnn key is pressed and the OS/400 program cannot move the text lines on the display.

If you specify option indicators on the PAGEDOWN or PAGEUP keyword, the same option indicators apply to the ALTPAGEDWN or ALTPAGEUP keyword, in the order given. That is, when the indicator condition is true, control will be returned to your program when either the Page key or the CFnn key is pressed and the OS/400 program cannot move the text lines on the display.

The following cannot be specified in a file with an ALTPAGEDWN keyword that has no parameter (CF08 default):

ALTHELP(CA08)
ALTPAGEUP(CF08)
CA08
CF08
MNUCNL(CA08)
MNUBARSW(CA08)
SFLDROP(CA08)
SFLDROP(CF08)
SFLENTER(CA08)
SFLENTER(CF08)
SFLFOLD (CA08)
SFLFOLD (CF08)

The following cannot be specified in a file with an ALTPAGEUP keyword that has no parameter (CF07 default):

ALTHELP(CA07)
ALTPAGEDWN(CF07)
CA07
CF07
MNUCNL(CA07)
MNUBARSW(CA07)
SFLDROP(CA07)
SFLDROP(CF07)
SFLENTER(CA07)
SFLENTER(CF07)
SFLFOLD (CA07)
SFLFOLD (CF07)

Similarly, the following cannot be specified in a file with ALTPAGEDWN(CFnn) or ALTPAGEUP(CFnn) (where nn is the same number):

ALTHELP(CAnn)
CAnn
CFnn
MNUCNL(CAnn)
MNUBARSW(CAnn)
SFLDROP(CAnn)
SFLDROP(CFnn)
SFLENTER(CAnn)
SFLENTER(CFnn)
SFLFOLD (CAnn)
SFLFOLD (CFnn)

Also, you cannot specify the same command function (CF) key on both the ALTPAGEDWN and the ALTPAGEUP keywords.

You cannot specify RETKEY or RETCMDKEY in a file with the ALTPAGEDWN or ALTPAGEUP keyword.

The ALTPAGEDWN and ALTPAGEUP keywords are allowed only in files containing a pageable area (either a subfile or a PAGEDOWN/PAGEUP keyword).

Option indicators are not valid for these keywords.

Figure 3-14 shows how to specify the ALTPAGEDWN and ALTPAGEUP keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      ALTPAGEUP
     A                                      ALTPAGEDWN
     A          R SUBFILE                   SFL
     A            FIELD1        20A    5   5
     A          R CONTROL                   SFLCTL(SUBFILE)
     A                                      SFLSIZ(30)
     A                                      SFLPAG(10)
     A                                      SFLDSP
     A
```

Figure 3-14. Specifying the ALTPAGEDWN and ALTPAGEUP Keywords

The CF07 and CF08 keys are alternative page keys. During display of the subfile, if the user presses the CF07/CF08 key or a Page key, the OS/400 program pages the subfile. If the OS/400 program cannot page the subfile without going beyond the end or beyond the beginning, a message is displayed indicating the key is not valid at that time (PAGEDOWN/PAGEUP not specified).

ALWGPH (Allow Graphics)

This file- or record-level keyword allows the display of graphics and the alphanumeric contents displayed by the record format on a 5292 Model 2 Color Display Station at the same time. The keyword is ignored if it is specified for a file displayed on any other type of display.

This keyword has no parameters.

When a record with this keyword in effect is written to a 5292 Model 2 Color Display Station, the device is placed in graphics display mode, if it was not already in that mode. This is done whether the keyword is in effect for any other record also displayed on the device. The device remains in graphics display mode as long as there is any record displayed with the ALWGPH keyword selected. Displaying a record that does not have the ALWGPH keyword will not cause graphics display mode to end.

To turn the graphics display off, you must delete all records with the ALWGPH keyword in them from the display (or option off the keyword in those records). When the graphics display is turned off, any graphics already on the display *are not* deleted, but are simply no longer displayed. Provided the graphics display is not deleted (through the use of GDDM* functions), it will be displayed again the next time a record with ALWGPH is displayed, including one from a secondary interactive job called from the System Request Menu.

In the graphics display mode:

- The device is automatically placed in reduced line spacing mode, with less space between lines. This is done whether or not the work station user has selected this mode from the keyboard (in local mode), and it cannot be overridden by the user.
- Any graphics that are already on the display when the DDS record format is displayed remain on the display as a background to the alphanumeric characters displayed by the record format.

Option indicators are valid for this keyword.

This keyword cannot be specified with the SFL or the USRDFN keywords.

Figure 3-15 shows how to specify the ALWGPH keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   ALWGPH
00020A                                    23  2'Enter account number:'
00030A            ACCT           5   B    +2
     A
```

Figure 3-15. Specifying the ALWGPH Keyword

In Figure 3-15, RECORD1 can be displayed with graphics. The constant *Enter account number:* and the field ACCT appear on the display. Any graphics also displayed (through the use of GDDM routines) would appear *behind* the alphanumerics. In other words, when a graphics line or pattern crosses the alphanumerics, a portion of the graphics is covered by the alphanumerics.

Figure 3-16 shows how RECORD2 can be displayed with graphics only if option indicator 01 is on.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD2
00020A  01                                    ALWGPH
00030A                                      1 34'Sample Title'
     A
```

Figure 3-16. Specifying an Option Indicator with the ALWGPH Keyword

In Figure 3-16, if option indicator 01 is off, any graphics already displayed are turned off (not deleted) and only the alphanumerics specified through DDS (such as *Sample Title*) appear on the display. On a later output operation, with option indicator 01 on, the graphics reappear with the alphanumerics.

ALWROL (Allow Roll)

Use this record-level keyword to allow your program to page through data in a window on the display when displaying the record format you are defining. The window consists of display lines between and including a start line and end line defined in your program. The number of lines to be paged through and the direction in which to page through them are defined in your program.

This keyword has no parameters.

When your program sends an output operation to this record format, the OS/400 program pages through data already in the window up or down the display and then displays the record format. Data paged past the start line or end line is lost. After you page through, your program cannot send an input operation to record formats that were either partially or completely within the window before the page.

To use the ALWROL function in COBOL, use the WRITE ROLLING statement.

The ALWROL keyword does not allow the display station user to page through data; it only allows your program to page through data on the display. To allow paging of data by the display station user, specify the ROLLUP and ROLLDOWN keywords or specify a subfile with subfile page not equal to subfile size. For more

information, see "SFLROLVAL (Subfile Roll Value)" on page 3-220 and "SFLRCDNBR (Subfile Record Number)" on page 3-218.

To prevent deleting paging records, specify the OVERLAY keyword or the CLRL (Clear Line) keyword with the ALWROL keyword.

Do not specify the PUTRETAIN keyword at the field level when you also specify the ALWROL keyword. If you do so, the OS/400 program sends CPF5014 when your program sends an output operation whether or not PUTRETAIN is selected.

If you specify the ALWROL keyword with the following keywords you must specify option indicators for them:

ERRMSG
ERRMSGID
PUTOVR
PUTRETAIN (at the record-level)

Your program cannot at the same time select one of these keywords and send an output operation that attempts to use the ALWROL function (the OS/400 program sends CPF5014).

The ALWROL keyword cannot be specified with any of the following keywords:

ASSUME
KEEP
SFL
SFLCTL
USRDFN

A warning message appears at file creation time if the ALWROL keyword is specified on a record with the DSPMOD keyword. At run time, the ALWROL keyword is ignored when the display mode changes.

The ALWROL keyword cannot be specified for the record format specified by the PASSRCD keyword.

Option indicators are not valid for this keyword.

Figure 3-17 shows how to specify the ALWROL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A      1  R RECORD1                  ALWROL OVERLAY  2
00020A           FLDA          79   I 23  2CHECK(LC)
00030A  44                                3  ERRMSG('Record not found' 44)
     A
```

Figure 3-17. Specifying the ALWROL Keyword

1 The application program can send an output operation to RECORD1, displaying FLDA on line 23, position 2. In a subsequent output operation, the program can page through RECORD1 (in this case, FLDA) up or down the display or entirely off the display. A normal case would be to page up one line. The originally keyed data is then displayed on line 22, and a new input field is displayed on line 23. The display station user cannot key into, and the program cannot read, the field on line 22. The field on line 22 can be

pushed up the display by subsequent output operations in this way until it passes the start line of the window (as specified in the program) or line 1 of the display.

2 The OVERLAY keyword prevents paged records from being deleted.

3 The ERRMSG keyword is shown to illustrate how ERRMSG affects the ALWROL function. The program cannot at the same time set option indicator 44 on and send an output operation that requests the ALWROL function. If the program does so, the OS/400 program sends a notify message CPF5014.

ASSUME (Assume)

Use this record-level keyword to specify that the OS/400 program is to assume that this record is already shown on the display when this file is opened. Such a record would also be defined, with the KEEP keyword, in another display file. That other display file would be closed before this file (in which you are specifying ASSUME) is opened.

This keyword has no parameters.

Specify the ASSUME keyword for at least one record format within the display file so that the OS/400 program does not erase the display when the file is opened. In addition, specify the OVERLAY keyword with the ASSUME keyword to prevent the OS/400 program from deleting the display when your program sends the first output operation after opening the file.

If you use the ASSUME keyword, at least one field in the record must be able to be displayed. If more than one record with the ASSUME keyword exists, they must occupy unique display lines.

For the OS/400 program to process the data correctly, your program must specify the record format name containing this keyword.

The ASSUME keyword is not needed if the record format you are defining is in a shared file (SHARE(*YES)) parameter specified on the Create Display File (CRTDSPF), Change Display File (CHGDSPF), or Override with Display File (OVRDSPF) command).

This keyword cannot be specified with any of the following keywords:

ALWROL	SLNO
CLRL	USRDFN
SFL	USRDSPMGT

A warning message is issued at file creation time if the ASSUME keyword is specified on a record with the DSPMOD keyword. At run time, the ASSUME keyword is ignored when the display mode changes.

A file with the ASSUME keyword will be opened to the display size of the file with the KEEP keyword.

Option indicators are not valid for this keyword.

Figure 3-18 on page 3-36 shows how to specify the ASSUME keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00005A          R RECORD                    ASSUME
     A
```

Figure 3-18. Specifying the ASSUME Keyword

AUTO (Auto)

The AUTO keyword is equivalent to the CHECK keyword as follows:

AUTO(RA) CHECK(ER)
AUTO(RAB) CHECK(RB)
AUTO(RAZ) CHECK(RZ)

The format of the keyword is:

```
AUTO(RA [RAB | RAZ])  AUTO(RAB | RAZ)
```

The CHECK keyword is preferred. See "Keyboard Control" on page 3-57 in the CHECK keyword description for an explanation of how to use these keywords.

BLANKS (Blanks)

This field-level keyword, when specified for a numeric, input-capable field, enables your program to distinguish when the field is blank and when the field is zero on the display. In either case, your program sees zeros. The BLANKS keyword sets on the specified response indicator when the field is blank on the display. After an input operation, your program can test this indicator to determine that the field (whose program value is zero) is actually blank on the display. The field can contain all blanks (hex 40) or all nulls (hex 00). It still appears blank to the display station user. If the indicator is off, the field is zero on the display.

This keyword is also valid for character fields, but there is generally no need to specify it for them. Your program can test character fields directly to determine what is on the display.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the BLANKS keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
BLANKS(response-indicator ['text'])
```

The response indicator associated with the BLANKS keyword should be unique within the record. That is, the same response indicator should not be used with other keywords such as CHANGE, DUP, or VLDCMDKEY; with any of the keywords for function keys; or with the BLANKS keyword on other fields in the same record. This is because the OS/400 program always turns the response indicator off if the field contains non-blank characters on an input operation. The OS/400 program does this to make sure that when the field appears as all blanks, the response indicator is set on, and that when it does not appear as all blanks, the response indicator is set off.

The optional text is included on the list generated at program compilation time to explain the intended use of the indicator. This text has no function in the file or the

program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program list.

Option indicators are not valid for this keyword.

Specifying the BLANKS Keyword

Figure 3-19 shows how to specify the BLANKS keyword.

Note: Figure 3-20, Figure 3-21 on page 3-38, and Figure 3-22 on page 3-38 show some cases that restrict the BLANKS keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A            QTY1           5Y 0B  5  2BLANKS(01 'ON=QTY1 IS ALL BLANKS')
00040A            QTY2           5Y 0B  6  2BLANKS(02 'ON=QTY2 IS ALL BLANKS')
00050A            QTY3           5Y 0B  7  2BLANKS(03 'ON=QTY3 IS ALL BLANKS')
     A
```

Figure 3-19. Specifying the BLANKS Keyword

Three numeric fields (QTY1, QTY2, and QTY3) are displayed. If the display station user keys values into the fields and presses the Enter key, the following occurs:

Value as Keyed into Fields	Value as Passed to Program	Condition of Response Indicator
100	00100	Off
0	00000	Off
Blanks	00000	On

Note: If the display station user presses a Field Exit key or the Erase Input key, the field appears blank because it contains nulls.

Restricting the BLANKS Keyword

In some cases, the BLANKS keyword does not set the specified response indicator on but rather restricts its function. Figure 3-20, Figure 3-21 on page 3-38, and Figure 3-22 on page 3-38 illustrate these cases.

Note: Other cases occur when the field is a character field, but then it is unnecessary to use the BLANKS keyword.

In Figure 3-20, when an output/input field contains all blanks (hex 40) or all nulls (hex 00) when displayed, and certain keywords affecting the display of the field are also specified, the response indicator is not set on.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* When OVRATR is specified
00020A          R REC1                          PUTOVR
00030A            FLD1          10  0B  2  2BLANKS(50) OVRATR
00040A  78                                    DSPATR(HI)
00050A* When PUTRETAIN is specified
00060A          R REC2                          PUTRETAIN OVERLAY
00070A            FLD2          10  0B  2  2BLANKS(50)
00080A          R REC3                          OVERLAY
00090A            FLD3          10  0B  2  2BLANKS(50)
     A                                        PUTRETAIN
     A
```

Figure 3-20. Specifying the BLANKS Keyword (I/O Fields with Nulls and Blanks)

For all record formats in this example, response indicator 50 is set on as expected the first time the field is read by the program (if the field appears blank on the

display). However, after a subsequent display, response indicator 50 is set on again *only if the display station user again blanks the field.* If the work station user does not again blank the field, response indicator 50 is off.

Figure 3-20 on page 3-37 and Figure 3-22 concern cases when the field is first displayed, then deleted.

In Figure 3-21, when an input-capable field is overlapped by another field, causing the first field to be deleted, the response indicator is not set on (even though the field in the input buffer still contains all blanks or all nulls).

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R REC4                        OVERLAY
00020A  15        FLDA          10  0B  2  2
00030A            FLD4          10  0B  2  5BLANKS(50)
     A
```

Figure 3-21. Specifying the BLANKS Keyword (Field Erased by Overlap)

In Figure 3-21, if option indicator 15 is off when REC4 is first displayed, FLD4 is displayed and FLDA is not. When REC4 is read, response indicator 50 is set on if FLD4 is blank. If option indicator 15 is then set on when REC4 is displayed again, FLDA overlaps FLD4 and deletes it. Response indicator 50 is then off when REC4 is read. (This occurs because the OS/400 program turns it off when displaying the record format and does not turn it back on for a field that is not on the display, even if the field contains blanks or nulls from a previous I/O operation.)

In Figure 3-22, after initial display, an output/input field is not displayed again on a subsequent output/input operation (even though the field in the input buffer still contains all blanks or all nulls).

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R REC5                        ERASEINP OVERLAY
00020A  20        FLD5          10  0B  2  2BLANKS(50)
00030A*
00040A          R REC6                        ERASEINP  OVERLAY  MDTOFF
00050A  20        FLD6          10  0B  2  2BLANKS(50)  DSPATR(MDT)
     A
```

Figure 3-22. Specifying the BLANKS Keyword (No Redisplay)

In Figure 3-22, if option indicator 20 is on when REC5 or REC6 is first displayed, FLD5 or FLD6 is displayed. When REC5 or REC6 is read, response indicator 50 is set on if FLD5 or FLD6 is blank. However, if option indicator 20 is set off on a second display, FLD5 or FLD6 is not displayed.

BLINK (Blink)

Use this record-level keyword to specify that as long as the record being defined is displayed, the cursor is to blink. The blinking is set off by the next output operation of a record that does not have the BLINK keyword specified.

This keyword has no parameters.

Option indicators are valid for this keyword.

Figure 3-23 on page 3-39 shows how to specify the BLINK keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00020A          R MASTER                    BLINK
     A
```

Figure 3-23. Specifying the BLINK Keyword

BLKFOLD (Blank Fold)

Use this field-level keyword for named, output-only fields (but not message or program-to-system fields) that are defined so that they overflow onto subsequent display lines. The keyword causes folding to occur at a blank in the data rather than at the end of the display line. It is used to make long text fields easier to read. The default is for the data to be folded at the end of the physical line.

This keyword has no parameters.

When BLKFOLD is used, the field length is not increased. Therefore, it is possible for a portion of the output data to be truncated.

You cannot specify the BLKFOLD keyword on a floating-point field (F in position 35).

Option indicators are not valid for this keyword.

Figure 3-24 shows how to specify the BLKFOLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A            FIELD1       638      2  1BLKFOLD
     A
```

Figure 3-24. Specifying the BLKFOLD Keyword

CAnn (Command Attention)

Use this file- or record-level keyword to specify that the function key specified in the keyword (CA01 through CA24) is available for use. It is to be used as a command attention (CA) key. No input data is transmitted from the device. Response indicators 01 through 99 are valid.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the CAnn keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
CAnn[(response-indicator ['text'])]
```

If you specify this keyword and the display station user presses the specified function key, the following happens:

- All other function key response indicators in the input buffer are set off (hex F0).
- The response indicator, if specified with the CAnn keyword, is set on (hex F1).
- The OS/400 data management feedback area is updated.

- Data already in the input buffer remains unchanged except that the response indicator (if specified) is set on.
- Control is returned to your program.

If you specify a response indicator and the key is pressed, the response indicator is set on and returned to your program. (The text information is associated with the indicator and is used by high-level language compilers to help in program documentation.)

If the display station user presses a function key and you have not specified it as either a command function (CF) key or a command attention key, the OS/400 program displays a message to the display station user indicating that the key is not valid at that time.

You can use combinations of CA and CF keywords within the same display file, but you cannot specify the same key number as both CA and CF keys. For example, CA02 and CF02 are not valid in the same display file.

Note: File level CA and CF keys are extended to the record level. This must be considered when assigning key numbers. For example, if CA02 is specified at file level and CF02 is specified at record level, CF02 is an error.

If you specify a key in the range 1 through 9, you must supply the leading zero in the keyword (for example, CA04).

Option indicators are valid for this keyword.

Validity Checking Considerations

When the display station user presses a CF key, the data from fields with their MDT set on is placed into the input buffer before validity checking is done. Any errors in the data are then detected, and the appropriate error messages are sent to the display. Because validity checking is not done until after the data is placed in the input buffer, pressing a valid CA key after the CF key can cause incorrect data to be returned to your program. This condition is not a problem as long as your program does not process the input data when the CA key is pressed.

This condition can be prevented in either of two ways:

- Do not allow the use of CA keys. Specify CF keys, which cause validity checking to be done on the data.
- Do not specify any of the following validity checking keywords if CA keys are allowed:

  CHECK(M10)
  CHECK(M11)
  CHECK(VN)
  CHECK(VNE)
  COMP (EQ, NE, LT, NL, GT, NG, LE, GE)
  RANGE
  VALUES

Function Keys Valid at Processing Time

As a general rule, the last output operation determines which function keys are valid. However, the following are exceptions to this rule:

- When an operation sends no data to the display, the validity of various function keys is not changed. Such operations include:
  - An output operation to a subfile record
  - An update to a subfile record
  - An output operation to a subfile control record that only clears, deletes, or initializes a subfile without displaying the subfile or the subfile control record
- An output operation that displays an error message by selecting ERRMSG (Error Message) or ERRMSGID (Error Message ID) can also select a CA or CF key to be valid while the error message is displayed.
- If MNUCNL (Menu Cancel), MNUBARSW (Menu Bar Switch), or SFLDROP (Subfile Drop) is specified for a subfile, the validity of the CA or CF key specified for the SFLDROP keyword is determined by the last output operation. However, as long as the subfile is displayed, the CA or CF key, when valid, acts only as a Drop key.
- If SFLFOLD (Subfile Fold) is specified for a subfile, the validity of the CA or CF key specified for the SFLFOLD keyword is determined by the last output operation. However, as long as the subfile is displayed, the CA or CF key, when valid, acts only as a Fold key.
- If two subfiles using SFLDROP or SFLFOLD are displayed at one time, the same function key should be specified on both the SFLDROP and SFLFOLD keywords. If they are different, only the key specified for the most recently displayed subfile is in effect. Pressing the function key affects the subfile containing the cursor. If the cursor is not positioned in a subfile, the function key affects the upper subfile.
- If two subfiles using SFLENTER (Subfile Enter) are displayed at the same time, the only CA or CF key in effect as an Enter key is the CA or CF key specified for the SFLENTER keyword on the most recently displayed subfile. The cursor position at the time the Enter key is pressed determines which subfile is affected.

Note: The following keywords function like CA keys: CLEAR, HELP, HOME, and PRINT (with response indicator specified).

Figure 3-25 shows how to specify the CAnn keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00011A                                  CA01(91 'End of Program')
00012A                                  CA02(92)
00013A                                  CA03
     A
```

Figure 3-25. Specifying the CAnn Keyword

CFnn (Command Function)

Use this file- or record-level keyword to specify that the function key specified in the keyword (CF01 through CF24) is available for use. It is to be used as a command function (CF) key to transmit changed data as opposed to a command attention

(CA) key, which does not transmit changed data. Response indicators 01 through 99 are valid.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the CFnn keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
CFnn[(response-indicator ['text'])]
```

If you specify this keyword, and the display station user presses the specified function key, the following happens:

- All other function key response indicators in the input buffer are set off (hex F0).
- The response indicator, if specified with the CFnn keyword, is set on (hex F1).
- The OS/400 data management feedback area is updated.
- Data is placed in the input buffer according to data received from the device.
- Control is returned to your program.

If you specify a response indicator and the key is pressed, the response indicator is set on and returned to your program along with the input data. If no response indicator is specified, the input data is returned to your program. (The text information is associated with the indicator and is used by high-level language compilers to help in program documentation.)

If the display station user presses a function key and you have not specified it as either a CF key or a CA key, the OS/400 program displays a message to the display station user indicating that the key is not valid at that time.

You can use combinations of CF and CA keywords within the same display file, but you cannot specify the same key number as both command attention and command function. For example, CA01 and CF01 are not valid in the same display file.

Note: File level CA and CF keys are extended to the record level. This must be considered when assigning key numbers. For example, if CA02 is specified at file level and CF02 is specified at record level, CF02 is an error.

If you specify a key in the range 1 through 9, you must supply the leading zero in the keyword (for example, CF03).

Option indicators are valid for this keyword.

Function Keys Valid at Processing Time

As a general rule, the last output operation determines which function keys are valid. The following are exceptions to this rule:

- When an operation sends no data to the display, the validity of the various function keys is not changed. Such operations include:
  - An output operation to a subfile record
  - An update to a subfile record

  - An output operation to a subfile control record that only clears, deletes, or initializes a subfile without displaying the subfile or the subfile control record

- An output operation that displays an error message by selecting ERRMSG or ERRMSGID can also select a CA or CF key to be valid while the error message is displayed.
- If SFLDROP is specified for a subfile, the validity of the CA or the CF key specified for the SFLDROP keyword is determined by the last output operation. However, as long as the subfile is displayed, the CA or CF key, when valid, acts only as a Drop key.
- If SFLFOLD is specified for a subfile, the validity of the CA or the CF key specified for the SFLFOLD keyword is determined by the last output operation. However, as long as the subfile is displayed, the CA or CF key, when valid, acts only as a Fold key.
- If two subfiles using SFLDROP or SFLFOLD are displayed at one time, the same function key should be specified on both the SFLDROP and SFLFOLD keywords. If they are different, only the key specified for the most recently displayed subfile is in effect. Pressing the function key affects the subfile containing the cursor. If the cursor is not positioned in a subfile, the function key affects the upper subfile.
- If two subfiles using SFLENTER are displayed at the same time, the only CA or CF key in effect as an Enter key is the CA or CF key specified for the SFLENTER keyword on the most recently displayed subfile. The cursor position at the time the Enter key is pressed determines which subfile is affected.

Note: The ROLLUP and ROLLDOWN keywords function like CF keys.

Figure 3-26 shows how to specify the CFnn keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                  CF01(91 'End of Program')
00020A                                  CF02(92)
00030A                                  CF03
     A
```

Figure 3-26. Specifying the CFnn Keyword

CHANGE (Change)

Use this record- or field-level keyword to set on the specified response indicator for an input operation under the following conditions:

- The keyword is specified at the record level, and any input-capable field in the record format has its changed data tag (MDT) set on.
- The keyword is specified for an input-capable field, and that field has its changed data tag (MDT) set on.

Refer to Appendix F, "System/36 Environment Considerations," for information on how to specify the CHANGE keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
CHANGE(response-indicator ['text'])
```

The MDT of an input-capable field is set on when the display station user keys into the field, or when your program selects the display attribute (DSPATR(MDT)) keyword for the output operation that displays the field. If the MDT is set on using the DSPATR(MDT) keyword, the data in the field may not have changed even though the MDT (and hence the response indicator specified for CHANGE) is set on. Also, note that the MDT is set on even if the work station user keys the same data in the field as was initially displayed (such as typing into a blank field and then blanking the field).

Note: The CHANGE response indicator is not set on when a command attention key (CAnn, Help, Print, Home, or Clear) is pressed.

When the OS/400 program detects validity checking errors and displays the record again with an error message, any CHANGE keyword response indicators that have been set on by typing into fields remain on until all validity checks succeed and the record is passed to your program.

The optional text is included on the list created at program compilation time to explain the intended use of the indicator. This text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

Option indicators are not valid for this keyword.

Figure 3-27 shows how to specify the CHANGE keyword at the field level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            FLDX           5   B  8  2CHANGE(67 'FLDX was changed')
     A
00020A            FLDY           3   I  8 30CHANGE(68 'FLDY was entered')
     A
```

Figure 3-27. Specifying the CHANGE Keyword at the Field Level

Figure 3-28 shows how to specify the CHANGE keyword at the record level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R REC1
00020A                                      CHANGE(88 'A field was changed')
00030A*
00040A            FIELD1        10   B  3  2
00050A            FIELD2         5   B  5  2
00060A            FIELD3         6   B  7  2
00070A            FIELD4         3   I  9  2DFT('ABC')
     A
```

Figure 3-28. Specifying the CHANGE Keyword at the Record Level

---

CHCACCEL (Choice Accelerator Text)

Use this field-level keyword on a single-choice selection field in a pull-down record to specify text for the accelerator key on a choice. The accelerator key is defined using the CFnn keyword.

The format of the keyword is:

```
CHCACCEL(choice-number accelerator-text)
```

The choice-number parameter specifies the number of the choice on the single selection field that this keyword applies to. Valid values are 1 to 99.

The accelerator-text parameter specifies the text identifying the accelerator key. The parameter can be specified in one of two forms:

As a character string: 'Accelerator text '
As a program-to-system field: &field-name

The field specified must exist in the same record as the selection field and must be defined as a character field with usage P. This text is placed 3 spaces to the right of the maximum length of the choice text. The maximum length of the accelerator text is determined by the length of the longest choice text. The combination of the two must not exceed the width of the smallest display size specified for the file.

The CHCACCEL keyword is allowed only on single-choice selection fields (SNGCHCFLD keyword specified on the same field) in pull-down records (PULLDOWN keyword specified at the record level).

Option indicators are not valid for this keyword.

Figure 3-29 shows how to specify the CHCACCEL keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8

     A
     A          R PULLEDIT                  CF04 CF06
     A                                      PULLDOWN
     A            F1             2Y 0B  1  2SNGCHCFLD
     A                                      CHOICE(1 '>Undo')
     A                                      CHOICE(2 '>Mark')
     A                                      CHOICE(3 '>Copy')
     A                                      CHCACCEL(1 'F4')
     A                                      CHCACCEL(2 &F6)
     A            F6             2A  P
```

Figure 3-29. Specifying the CHCACCEL Keyword

In this example, choices 1 and 2 have accelerator keys CF04 and CF06 respectively. When the pull-down menu is displayed, the character text F4 appears to the right of the text 'Undo', with 3 spaces in between, and the text in field F6 appears to the right of the text 'Mark', with 3 spaces in between. The longest choice text determines the length of all choice text. The same is true for the ACCEL text. The ACCEL text is then started 3 spaces to the right of the longest choice.

CHCAVAIL (Choice Color/Display Attribute when Available)

Use this field-level keyword to specify the color or display attributes to be used when displaying the available choices in a menu bar or selection field.

The format of the keyword is:

```
CHCAVAIL([color] [display-attributes])
```

One parameter *must* be specified.

The color parameter indicates the color of the choice text specified on either the MNUBARCHC or CHOICE keywords for the field on a color work station. The parameter is specified as an expression of the form (*COLOR value).

The valid values for the color parameter are:

Value	Meaning
BLU	Blue
GRN	Green
PNK	Pink
RED	Red
TRQ	Turquoise
YLW	Yellow
WHT	White

If the color parameter is not specified, the default color for the available choices in a menu bar is green. The default color for the available choices in a selection field is green. This parameter is ignored on a monochrome work station.

The display-attribute parameter indicates the display attributes of the choice text specified on the MNUBARCHC or CHOICE keywords for the field. The parameter is specified as an expression of the form, (*DSPATR value1 <value2 <value3...>>).

The valid values for the display attributes are:

Value	Meaning
BL	Blink
CS	Column separator
HI	High intensity
ND	Nondisplay
RI	Reverse image
UL	Underline

The default display attribute in a menu bar is high intensity. The default display attribute in a selection field is normal (or low) intensity.

Note: Display attributes CS, HI, and BL can cause fields on 5292, 3179, 3197 Models C1 and C2, 3477 Model FC, 3486, 3487 Model HC, and 3488[1] work stations to appear as color fields. Separator lines do not appear when display attributes HI, RI, and UL are used. For more information on the COLOR keyword, see "COLOR (Color)" on page 3-72.

The CHCAVAIL keyword is allowed only on fields containing one or more CHOICE or MNUBARCHC keywords.

Option indicators are valid for this keyword.

Figure 3-30 on page 3-47 shows how to specify the CHCAVAIL keyword:

[1] Dependent on monitor attached to display device.

```
|....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD                    MNUBAR
     A            F1             2Y 0B  1  2
     A                                      MNUBARCHC(1 PULLFILE 'File        ')
     A                                      MNUBARCHC(2 PULLEDIT 'Edit        ')
     A                                      CHCAVAIL((*COLOR YLW))
     A
```

Figure 3-30. Specifying the CHCAVAIL Keyword (Example 1)

In Figure 3-31, the File choice for the menu bar of a color display is yellow, if available. For a monochrome display, the menu bar is the default, high intensity. The available choices for the selection field are displayed with underlines.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD
     A            F1             2Y 0B  2  5SNGCHCFLD CHECK(ER)
     A                                      CHOICE(1 'Choice number 1')
     A                                      CHOICE(2 'Choice number 2')
     A                                      CHCCTL(1 &CHCCTL1)
     A                                      CHCCTL(2 &CHCCTL2)
     A                                      CHCAVAIL((*DSPATR UL))
     A
```

Figure 3-31. Specifying the CHCAVAIL Keyword (Example 2)

CHCCTL (Choice Control)

Use this field-level keyword on a selection field to control the availability of the choices for the field.

The format of the keyword is either one of the following:

```
CHCCTL(choice-number &control-field [msg-id [msg-lib/]msg-file])
                    or
CHCCTL(choice-number &control-field [&msg-id [&msg-lib/]&msg-file])
```

The choice-number parameter is required and it specifies the choice to which this keyword applies. Valid values are 1 to 99.

The control-field parameter is required and it specifies the name of a 1-byte numeric hidden field that, on output, contains the control value for the choice. The field must be defined within the same record as the field you are defining, and must be defined as data type Y (numeric) with length 1, decimal positions 0, and usage H. On input for multiple-choice selection fields, the selection field indicates whether the field was selected.

Following are the control values for the hidden field, and their meaning on input and output:

Figure 3-32. Control Values for Hidden Fields

Control Value	Meaning on Output	Meaning on Input
0	available	unselected
1	selected	selected
2	unavailable	
other	available	

The message-id and message-file parameters are optional and specify a message to be displayed when the user selects an unavailable choice. If these parameters are not specified, the system issues a default message, CPD919B, when the user selects an unavailable choice. If a field is used for the message-id, that field must exist in the record you are defining and it must be defined as data type A, usage P, and length of 7.

The message-file parameter is a required parameter when the message-id parameter is used. If you do not specify the library parameter, *LIBL is used to search for the message file at program run time. If a field is used for the message library or message file, that field must exist in the record you are defining and it must be defined as data type A, usage P, and length of 10.

When the CHCCTL keyword is specified on a field, a CHOICE keyword with the same choice number must also be specified for the field.

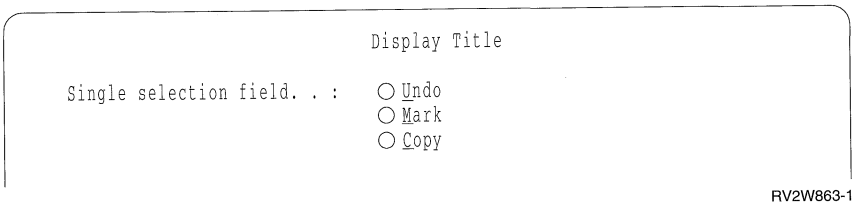
Option indicators are not valid for this keyword.

Figure 3-33 shows how to specify the CHCCTL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      :
     A                                      :
     A            F1             2Y 0B  3 35SNGCHCFLD
     A                                        CHOICE(1 '>Undo       ')
     A                                        CHOICE(2 '>Mark       ')
     A                                        CHOICE(3 '>Copy       ')
     A                                        CHCCTL(1 &CTLUNDO MSG1112 QUSER/A)
     A                                        CHCCTL(2 &CTLMARK &MSG &LIB/&MSGF)
     A                                        CHCCTL(3 &CTLCOPY)
     A            CTLUNDO         1Y 0H
     A            CTLMARK         1Y 0H
     A            CTLCOPY         1Y 0H
     A            MSG             7A  P
     A            MSGF           10A  P
     A            LIB            10A  P
     A
```

Figure 3-33. Specifying the CHCCTL Keyword

When using a graphical display station attached to a controller that supports an enhanced interface for nonprogrammable work stations, the selection field looks like this:



CHCSLT (Choice Color/Display Attribute when Selected)

Use this field-level keyword to specify the color or display attributes to be used when displaying a selected choice in a menu bar. You can also use this keyword to specify the color or display attributes to be used for selected choices in a selection field, if the selection field is in a pull-down menu that has PULLDOWN (*NOSLTIND).

The format of the keyword is:

```
CHCSLT([color] [display-attributes])
```

One parameter *must* be specified.

The color parameter indicates the color of the choice text specified on the MNUBARCHC or CHOICE keywords for the field on a color work station. The parameter is specified as an expression of the form (*COLOR value).

The valid values for the color parameter are:

Value	Meaning
BLU	Blue
GRN	Green
PNK	Pink
RED	Red
TRQ	Turquoise
YLW	Yellow
WHT	White

If the color parameter is not specified, the default color for the selected choice in a menu bar is white. The default color for the selected choices in a selection field in a pull-down menu that does not display selection characters is white. The color parameter is ignored on a monochrome display.

The display-attribute parameter indicates the display attributes of the choice text specified on the MNUBARCHC or CHOICE keywords for the field. The parameter is specified as an expression of the form (*DSPATR value1 <value2 <value3...>>).

The valid values for the display attributes are:

Value	Meaning
BL	Blink
CS	Column separator
HI	High intensity
ND	Nondisplay
RI	Reverse image
UL	Underline

The default display attribute for the selected choice in a menu bar is normal (or low) intensity. The default display attribute for the selected choices in a selection field in a pull-down menu that does not display selection characters is high intensity.

Note: Display attributes CS, HI, and BL can cause fields on 5292, 3179, 3197 Models C1 and C2, 3477 Model FC, 3486, 3487 Models HC, and 3488[2] work stations to appear as color fields. Separator lines do not appear when display attributes HI, RI, and UL are used. For more information on the COLOR keyword, see "COLOR (Color)" on page 3-72.

When this keyword is specified, either the MNUBARCHC keyword or the CHOICE keyword must also be specified on the field. If the CHOICE keyword is specified on the field rather than MNUBARCHC, the record containing this field must have the PULLDOWN keyword specified with the value *NOSLTIND.

Option indicators are valid for this keyword.

Figure 3-34 shows how to specify the CHCSLT keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD                    MNUBAR
     A            F1             2Y 0B  1  2
     A                                      MNUBARCHC(1 PULLFILE 'File        ')
     A                                      MNUBARCHC(2 PULLEDIT 'Edit        ')
     A                                      CHCSLT((*COLOR PNK) (*DSPATR RI))
     A
```

Figure 3-34. Specifying the CHCSLT Keyword

In this example, when the choice is selected on a color display, the menu bar is displayed in pink reverse image.

CHCUNAVAIL (Choice Color/Display Attribute when Unavailable)

Use this field-level keyword to specify the color or display attributes to be used when displaying the unavailable choices in a selection field.

The format of the keyword is:

```
CHCUNAVAIL([color] [display-attributes])
```

One parameter *must be* specified.

The color parameter indicates the color of the choice text specified on the CHOICE keywords for the field on a color display station. The parameter is specified as an expression of the form (*COLOR value).

The valid values for the color parameter are:

Value	Meaning
BLU	Blue
GRN	Green
PNK	Pink

[2] Dependent on monitor attached to display device.

RED	Red
TRQ	Turquoise
YLW	Yellow
WHT	White

If the color parameter is not specified, the default color for unavailable choices in a selection field is blue. This parameter is ignored on a monochrome display.

The display-attribute parameter indicates the display attributes of the choice text specified on the CHOICE keywords for the field. The parameter is specified as an expression of the form (*DSPATR value1 <value2 <value3...>>).

Value	**Meaning**
BL	Blink
CS	Column separator
HI	High intensity
ND	Nondisplay
RI	Reverse image
UL	Underline

The default display attribute for unavailable choices in a selection field on monochrome display stations is normal (or low) intensity. Also, the first character of an unavailable choice on a monochrome display station is overwritten with an asterisk (*).

Note: Display attributes CS, HI, and BL can cause fields on 5292, 3179, 3197 Models C1 and C2, 3486, 3487 Model HC, and 3488[3] work stations to appear as color fields. Separator lines do not appear when display attributes HI, RI, and UL are used. For more information on the COLOR keyword, see "COLOR (Color)" on page 3-72.

This keyword is allowed only on fields containing one or more CHOICE keywords.

Option indicators are valid for this keyword.

Figure 3-35 shows how to specify the CHCUNAVAIL keyword:

```
|....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD
     A            F1             2Y 0B  2  5SNGCHCFLD CHECK(ER)
     A                                      CHOICE(1 'Choice number 1')
     A                                      CHOICE(2 'Choice number 2')
     A                                      CHCCTL(1 &CHCCTL1)
     A                                      CHCCTL(2 &CHCCTL2)
     A                                      CHCUNAVAIL((*COLOR TRQ))
     A            :
     A            :
     A
```

Figure 3-35. Specifying the CHCUNAVAIL Keyword

In this example, the unavailable choices for the selection field are displayed in turquoise on a color display.

[3] Dependent on monitor attached to display device.

CHECK (Check)

Use this keyword to perform the following functions, depending on the parameter values specified:

Function	Valid Parameter Values
Validity checking	AB, ME, MF, M10, M10F, M11, M11F, VN, VNE
Keyboard control	ER, FE, LC, RB, RZ
Cursor control	RL, RLTB

The formats of the keyword are:

```
CHECK(validity-checking-code [. . .])
CHECK(keyboard-control-code [. . .])
CHECK(cursor-control-code)
```

The following CHECK keywords are the preferred form of other DDS keywords:

- CHECK(ER) is equivalent to AUTO(RA)
- CHECK(LC) is equivalent to LOWER
- CHECK(RB) is equivalent to AUTO(RAB)
- CHECK(RZ) is equivalent to AUTO(RAZ)

The following CHECK keyword functions can also be specified using the Change Input Default (CHGINPDFT) keyword at the file, record, or field level:

CHECK Keyword	CHGINPDFT Equivalent
CHECK(FE)	CHGINPDFT(FE)
CHECK(LC)	CHGINPDFT(LC)
CHECK(ME)	CHGINPDFT(ME)
CHECK(MF)	CHGINPDFT(MF)

Option indicators are valid only for CHECK(ER) and CHECK(ME).

Validity Checking

Use CHECK at the field level to specify that the OS/400 program or the device is to check the validity of the data typed into an input-capable (input-only or output/input) field.

CHECK validates the data by applying one or more edit/check algorithms against the data. An error message is displayed if a specified edit/check algorithm is not satisfied.

Note: See "CHKMSGID (Check Message Identifier)" on page 3-64 for information on defining user-specified messages.

The valid edit/check codes are:

Edit/Check Code	Meaning
AB	Allow blanks
ME	Mandatory enter
MF	Mandatory fill
M10	IBM Modulus 10 self-check algorithm
M10F	IBM Modulus 10 self-check algorithm
M11	IBM Modulus 11 self-check algorithm
M11F	IBM Modulus 11 self-check algorithm

VN	Validate name
VNE	Validate name extended

AB (Allow Blanks): Use this code at the file, record, or field level to allow all-blank input to satisfy validity checking for an input-capable field should any associated validity check fail. This enables the passing of data to the program when the work station user has positioned the cursor to the field but left it blank (for instance, by pressing the Erase Input key, the Field Exit key, or the spacebar). For example, FLD1 is an input-capable field with CHECK(M10 ME) in effect. If the work station user accidentally types into the field, the M10 algorithm must be satisfied. Specifying (CHECK(M10 ME AB)) allows the display station user to blank the field to satisfy validity checking.

When specified at the file level, this keyword applies for all input-capable fields in the file for which a validity checking keyword is coded. Likewise, when specified at the record level, this keyword applies for all input-capable fields in the record for which a validity checking keyword is coded. At the field level, always specify this keyword with another validity checking keyword (CHECK(M10, M10F, M11, M11F, VN, VNE), CHKMSGID, COMP, RANGE, or VALUES).

CHECK(AB) should not be specified if SFLROLVAL or SFLRCDNBR is also specified for the field.

CHECK(AB) can be used in database files for reference purposes.

When you consider using CHECK(AB) with other validity checking functions, note that processing occurs in the following order:

1. Any of the following:
   a. The keyboard shift attribute specified in position 35 (such as alphanumeric shift or numeric only) can restrict input keying to certain characters.
   b. If the keyboard shift attribute is numeric shift, the data type (character or numeric) is set by the entry in positions 36 through 37 (decimal positions) and restricts input keying to certain characters.
   c. The CHECK(FE), CHECK(MF), and CHECK(ME) keywords, if specified, restrict input keying.
2. Either of the following:
   a. If CHECK(AB) is specified, data management passes the input data to the program (blanks for a character field and zeros for a numeric field). No further validity checking is done.
   b. If CHECK(AB) is not specified, data management performs the following validity checking functions before passing the data to the program: CHECK(VN), CHECK(VNE), CHECK(M10), CHECK(M10F), CHECK(M11), CHECK(M11F), COMP(. . .), RANGE(. . .), VALUES(. . .).

You cannot specify the CHECK(AB) keyword on a floating-point field (F in position 35).

Option indicators are not valid for this keyword.

ME (Mandatory Enter): This code specifies that at least 1 character of data (a blank is valid) must be typed into the field. Note that when no field currently on the display has been changed, the display station does not enforce mandatory enter.

To enforce mandatory enter, specify DSPATR(MDT) for at least one field in each record on the display. For all other fields in the record, CHECK(ME) is then enforced. However, because the device cannot determine if the user has typed data to a field with both DSPATR(MDT) and CHECK(ME), you should also specify DSPATR(ND) so that this field is not displayed.

Option indicators are valid for this keyword.

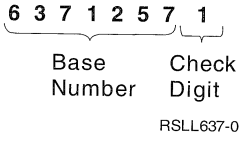
MF (Mandatory Fill): This code specifies that if any part of the field is altered, each position in the field must have a character entered in it. Blanks are considered valid characters.

This code cannot be specified with keyboard control codes RB or RZ.

Option indicators are not valid for this keyword.

M10/M10F or M11/M11F (IBM Modulus 10 or Modulus 11 Algorithm): This code specifies that data typed into the field must satisfy the IBM Modulus 10 (M10 or M10F) or Modulus 11 (M11 or M11F) self-check algorithm. When you specify CHECK(M10) or CHECK(M11), the self-check verifies that the field has a valid Modulus 10 or Modulus 11 number when you press the Enter key or a function key. When you specify CHECK(M10F) or CHECK(M11F), the self-check verifies that the field has a valid Modulus 10 or Modulus 11 number as the user types the data into the field. You cannot specify both the Modulus 10 and the Modulus 11 self-check algorithms for the same field or both formats of the same algorithm for the same field.

A self-check field is composed of two parts: the base number and one check digit. The check digit is the farthest right digit in the field. The base number and the check digit together make up a field in your database (for example, an account number). The following illustration is an example of an 8-digit self-check field.



See the *Guide to Programming Displays* for information on how to use CHECK(M10), CHECK(M10F), CHECK(M11), and CHECK(M11F).

Notes:

1. The OS/400 program supports a maximum length of 31 digits for numeric fields.
2. You cannot specify the CHECK(M10), CHECK(M10F), CHECK(M11), and CHECK(M11F) keywords with the COMP(EQ) keyword.
3. You cannot specify the CHECK(M10), CHECK(M10F), CHECK(M11), and CHECK(M11F) keywords on a floating-point field (F in position 35).
4. You cannot specify the CHECK(M10F) or the CHECK(M11F) keyword in a file containing the USRDSPMGT keyword.
5. You cannot specify the CHECK(M10F) or the CHECK(M11F) keyword on a field containing the CHKMSGID keyword.

For each position in the base number, there is a Modulus 10 weight factor and a Modulus 11 weight factor. Positions are counted from the farthest right digit (not including the check digit).

The Modulus 10 weight factor is 2 for positions 1, 3, 5, . . ., 31. It is 1 for positions 2, 4, 6, . . ., 30. The Modulus 11 weight factors are 2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 6, 7, . . ., 2, 3, 4, 5, 6, 7, 2 for positions 1, 2, . . ., 31, respectively.

To calculate the Modulus 10 self-check digit, do the following:

1. Multiply the units position and every alternate position of the base number by 2.
2. Add the digits in the products to the digits in the base number that were not multiplied.
3. Subtract the sum from the next higher number ending in zero.

The difference is the self-check digit.

For example:

Base number	6	1	2	4	8
Units position and every alternate position	6		2		8
Multiply by the weight factor, 2	x2		x2		x2
Products	12		4		16
Digits not multiplied		1		4	
Add the *digits* of the products and the digits from the base number not used for multiplication	1 + 2 + 4 + 1 + 6 + 1 + 4 = 19				
Next higher number ending in 0	20				
Subtract	–19				
Self-check digit	1				

To calculate the Modulus 11 self-check digit, do the following:

1. Assign a weight factor to each digit position of the base number. These factors are: 2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 6, 7, 2, 3, . . . starting with the units position of the number and progressing toward the high-order digit. For example, the base number 991246351 would be assigned the weight factors as follows:

```
Base number      9 9 1 2 4 6 3 5 1

Weight factors   4 3 2 7 6 5 4 3 2
```

2. Multiply each digit by its weight factor.
3. Add the products.
4. Divide this sum by 11.
5. Subtract the remainder from 11.

The difference is the self-check digit.

For example:

Base number	1	3	7	3	9
Weight factors	x6	x5	x4	x3	x2

Multiply each digit by its weight factor	6 15 28 9 18
Add the products	6 + 15 + 28 + 9 + 18 = 76
Divide the sum by 11	76/11 = 6 plus a remainder of 10
Subtract the remainder from 11	11 - 10 = 1
Self-check digit	1

Note: If the remainder in step 4 is 0, the self-check digit is 0. If the remainder is 1, the base number has no self-check digit; you must make sure that such base numbers are not used in the fields you define as self-check fields.

Option indicators are not valid for CHECK(M10), CHECK(M10F), CHECK(M11), or CHECK(M11F).

VN (Validate Name): Use this code to specify that the data typed into the field must be a valid simple name. The first character must be $, #, @, or A through Z. The remaining characters must be alphanumeric ($, #, @, A through Z, 0 through 9, or underscore (_), and must not contain embedded blanks.

When the CHECK(VN) keyword is specified on a field, the field must be character (keyboard shift of A, N, X, W, or I), and must be input-capable (usage of I or B).

CHECK(VN) cannot be specified with any of the following keywords:

CHECK(M10)	CHECK(VNE)
CHECK(M10F)	COMP
CHECK(M11)	RANGE
CHECK(M11F)	VALUES

Option indicators are not valid for this keyword.

VNE (Validate Name Extended): Use this code to specify that the data typed into the field must be a valid extended name.

When the CHECK(VNE) keyword is specified on a field, the field must be character (keyboard shift of A, N, X, W, or I), input-capable (usage of I or B), and have a maximum length of 255 characters.

If the name is not delimited by double quotation marks:

- The first character must be A through Z, a through z, #, $, or @.
- The remaining characters must be A through Z, a through z, #, $, _, or a period.
- Lowercase letters will be converted to uppercase.

If the name is delimited by double quotation marks:

- Any character is allowed except:

Hex 00 through Hex 3F	(device control)
Hex FF	(device control)
Hex 40	(blank)
Hex 5C	(*)
Hex 6F	(?)

Hex 7D	(')
Hex 7F	(")

- Lowercase letters remain lowercase.
- The system removes quotation marks when they are not needed (if the syntax of the name meets the requirements of an unquoted name, and all letters are uppercase).

CHECK(VNE) cannot be specified with any of the following keywords:

CHECK(M10)	CHECK(VN)
CHECK(M10F)	COMP
CHECK(M11)	RANGE
CHECK(M11F)	VALUES

Option indicators are not valid for this keyword.

Figure 3-36 shows how to specify the validity-checking CHECK keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00060A          R RECORD1                     CHECK(AB)
00070A            FIELD11       10   B  1  2TEXT('CHECK(AB) not propagated to +
00080A                                          this field')
00090A            FIELD21       10   B  1 22CHECK(VN)
00100A                                        TEXT('CHECK(AB) is propagated to +
00110A                                          this field')
00120A            FIELD31       10   B  1 42CHECK(VNE)
00130A                                        TEXT('CHECK(AB) is propagated to +
00140A                                          this field')
00150A*
00160A          R RECORD2
00170A            FIELD12       10   B  2  2CHECK(VN) CHECK(AB)
00180A            FIELD22       10   B  2 22CHECK(VN AB)
00190A            FIELD32        1   B  2 42CHECK(AB) VALUES('A' 'B' 'C')
00200A            FIELD42       10   B  2 62CHECK(VN)
00210A            FIELD52       10   B  3  2CHECK(VNE)
00220A            FIELD62       10   B  3 22CHECK(VNE AB)
00230A            FIELD72       10   B  4  1CHECK(ME MF)
00240A            FIELD82        8  OB  4 22CHECK(M10)
00250A            FIELD92       10  OB  4 42CHECK(M11)
     A
```

Figure 3-36. Specifying the Validity-Checking CHECK Keywords

Keyboard Control

When the CHECK keyword is used with a keyboard control code, it controls certain data-entry aspects. The valid keyboard control codes are:

Keyboard Control Code	Meaning
ER	End of record; equivalent to AUTO(RA)
FE	Field exit check
LC	Lowercase; equivalent to LOWER
RB	Right-justified with blank fill; equivalent to AUTO(RAB)
RZ	Right-justified with zero fill; equivalent to AUTO(RAZ)

ER (End of Record): Use this code so that the work station user does not need to press the Enter key. Whenever the work station user keys a character (including a blank) into the last position of the field, the record is sent from the device just the same as if the Enter key had been pressed. If you also specify DSPATR(SP) for the field, the record is sent from the device as soon as the work station user selects the field. If you use this function, it should be on the last field typed in by the user for this record.

Option indicators are valid for this keyword.

FE (Field Exit Check): This code specifies that the work station user cannot advance to the next input field without pressing one of the field exit keys. The cursor remains under the low-order character position of the field until a valid field exit key has been pressed, even though that character has been keyed in. If the user presses any other key, an error results.

If you want to specify CHECK(FE) for all the input-capable fields in a record format, specify CHGINPDFT(FE) at the record level. If you want to specify CHECK(FE) for all the input-capable fields in a file, specify CHGINPDFT(FE) at the file level.

Field exit keys include the Field Exit, Field+, Field–, and cursor movement keys. Which keys are valid field exit keys depends on the keyboard style being used.

This code applies only to input fields into which the work station user can type.

Option indicators are not valid for this keyword.

LC (Lowercase): Use CHECK(LC) for input-only or output/input fields to permit the work station user to type lowercase a through z. The way the work station user keys in the characters (uppercase or lowercase) is the way the characters appear on the display and are returned to your program.

If you want to specify CHECK(LC) for all the character input-capable fields in a record format, specify CHGINPDFT(LC) at the record level. If you want to specify CHECK(LC) for all the character input-capable fields in a file, specify CHGINPDFT(LC) at the file level.

Your program can display a field that contains both uppercase and lowercase characters.

If you specify this keyword, lowercase a through z remain lowercase. If you do not specify this keyword, lowercase a through z are changed to uppercase.

The CHECK(LC) keyword has no effect on data-entry keyboards. Data-entry keyboards do not support lowercase characters a through z.

Option indicators are not valid for this keyword.

Figure 3-37 shows how to specify the CHECK(LC) keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A           NAME          30   I  3  2CHECK(LC)
     A
```

Figure 3-37. Specifying the CHECK(LC) Keyword

RB (Right Justify with Blank Fill): This code shifts data typed into the field to the farthest right positions and fills the remaining positions with blanks.

For signed numeric fields, you do not need to specify CHECK(RB). Right-justified with blank fill is the default. When the value of a signed numeric field is zero, it appears as all blanks on the display.

The OS/400 program converts blanks to zeros when returning numeric fields to your program.

Option indicators are not valid for this keyword.

RZ (Right Justify with Zero Fill): This code shifts data typed into the field to the farthest right positions and fills the remaining positions with zeros.

For signed numeric fields, if you do not specify CHECK(RZ), CHECK(RB) is the default.

Option indicators are not valid with this keyword.

Programming considerations for CHECK(RB) and CHECK(RZ) include the following:

- You activate right-justification only by pressing the Field Exit, the Field+, or the Field– key. If you use the cursor movement keys to exit from a right-justified field, the field is not right-justified; it is left as is.
- Right-justified fields longer than 15 character positions slow keyboard entries.
- The Dup key fills a right-justified field from the cursor location to the end of the field with the duplication character, but the field is not right-justified.

Figure 3-38 shows how to specify the CHECK keyword for right-justified with blank fill (RB) and for right-justified with zero fill (RZ).

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A* Numeric only
00030A            DATA1          7Y OI  2  2TEXT('No right-adjust')
00040A            DATA2          7Y OI  3  2CHECK(RZ)
00050A* Signed numeric
00060A            DATA3          7S OI  4  2TEXT('CHECK(RB) is the default')
00070A            DATA4          7S OI  5  2CHECK(RZ)
00080A* Character
00090A            DATA5          7   I  6  2TEXT('No right-adjust')
00100A            DATA6          7   I  7  2CHECK(RB)
     A
```

Figure 3-38. Specifying the CHECK Keyword for Right-justified with Zero Fill and Blank Fill

When you specify the CHECK keyword for right-justified with zero or blank fill, fill the following displays:

Field Name	Data Keyed In	Key Pressed	Result on Display	
Numeric only				
DATA1	1. 1 2 3 _ _ _ _	Field Exit	1 2 3 _ _ _ _	
	2. 1 2 3 – _ _ _	Field Exit	1 2 3 – _ _ _	
	3. 0 _ _ _ _ _ _	Field Exit	0 _ _ _ _ _ _	See note.
	4. _ _ _ _ _ _ _	Field Exit	_ _ _ _ _ _ _	See note.
DATA2	1. 1 2 3 _ _ _ _	Field Exit	0 0 0 0 1 2 3	
	2. 1 2 3 – _ _ _	Field Exit	0 0 0 1 2 3 –	
	3. 0 _ _ _ _ _ _	Field Exit	0 0 0 0 0 0 0	
	4. _ _ _ _ _ _ _	Field Exit	0 0 0 0 0 0 0	
Signed Numeric				
DATA3	1. 1 2 3 _ _ _ _ _	Field Exit	_ _ _ _ 1 2 3 _	
	2. 1 2 3 _ _ _ _ _	Field–	_ _ _ _ 1 2 3 –	
	3. 0 _ _ _ _ _ _ _	Field Exit	_ _ _ _ _ _ 0 _	See note.
	4. _ _ _ _ _ _ _ _	Field Exit	_ _ _ _ _ _ _ _	See note.
DATA4	1. 1 2 3 _ _ _ _ _	Field Exit	0 0 0 0 1 2 3 _	
	2. 1 2 3 _ _ _ _ _	Field–	0 0 0 0 1 2 3 –	
	3. 0 _ _ _ _ _ _ _	Field Exit	0 0 0 0 0 0 0 _	
	4. _ _ _ _ _ _ _ _	Field Exit	0 0 0 0 0 0 0 _	

Note: The OS/400 program converts blanks to zeros when returning numeric fields to your program. Therefore, this field is returned to your program as all zeros.

Character				
DATA5	1. 1 2 3 _ _ _ _	Field Exit	1 2 3 _ _ _ _	
	2. 1 2 3 – _ _ _	Field Exit	1 2 3 – _ _ _	
	3. 0 _ _ _ _ _ _	Field Exit	0 _ _ _ _ _ _	
	4. _ _ _ _ _ _ _	Field Exit	_ _ _ _ _ _ _	
	5. A B C _ _ _ _	Field Exit	A B C _ _ _ _	
DATA6	1. 1 2 3 _ _ _ _	Field Exit	_ _ _ _ 1 2 3	
	2. 1 2 3 – _ _ _	Field Exit	_ _ _ 1 2 3 –	
	3. 0 _ _ _ _ _ _	Field Exit	_ _ _ _ _ _ 0	
	4. _ _ _ _ _ _ _	Field Exit	_ _ _ _ _ _ _	
	5. A B C _ _ _ _	Field Exit	_ _ _ _ A B C	

Cursor Control

When the CHECK keyword is used with a cursor control code, it specifies that the cursor is to move from right to left. This feature is designed for languages where information is read right to left.

The OS/400 program does not ensure that right-to-left files are opened only for display stations capable of right-to-left cursor movement. Therefore, all work stations in the same system should be configured with the same language capability and with the same right-to-left capability.

The valid cursor control codes that can be specified for cursor control are:

Cursor Control Code	Meaning
RL	Right-to-left cursor movement within fields
RLTB	Right-to-left, top-to-bottom cursor movement from field to field

The right-to-left capability includes the following restrictions:

- The check digit for modulus checking is the farthest right byte in the field.
- Katakana cannot be used with right-to-left support.
- CHECK(RL) and CHECK(RLTB) cannot be specified with user-defined records (having the USRDFN keyword).
- CHECK(RL) applies only to character fields.

A warning message appears for the following conditions:

- A right-to-left field that also allows magnetic card reader operator identification data (DSPATR(OID) keyword)
- A right-to-left field that spans more than one line
- A right-to-left field that is also a self-check field (CHECK(M10) or CHECK(M11) keyword)
- A right-to-left field for which CHECK(RZ) or CHECK(RB) is specified

Option indicators are not valid with cursor control codes.

RL (Right to Left): Use the CHECK(RL) keyword at the file, record, or field level to specify that the cursor should move from right to left within input-capable character fields. At the file level, specifying CHECK(RL) makes the cursor move from right to left in all input-capable character fields in the file. At the record level, specifying CHECK(RL) makes the cursor move from right to left in all input-capable character fields in the record. At the field level, specifying CHECK(RL) makes the cursor move from right to left in only the field with which it is associated.

Figure 3-39 shows how to specify the CHECK(RL) keyword at the file level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      CHECK(RL)
00020A          R DSPLY
     A
```

Figure 3-39. Specifying the CHECK(RL) Keyword at the File Level

Figure 3-40 shows how to specify the RL cursor control with edit check.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A            :
     A            :
     A          R RECORD1                  CHECK(RL AB)
     A            :
     A            :
     A          R RECORD2
     A            INPFLD         4   I  4 10CHECK(RL MF)
     A            :
     A            :
     A
```

Figure 3-40. Specifying the RL Cursor Control with Edit Check

Note: If you want to specify the RL cursor control code with an edit/check code, you can do so only if the edit/check code is valid at the level you specify. In the example above, CHECK(RL AB) is specified at the record level because AB is valid at that level. CHECK(RL MF) is specified at the field level because MF is valid only at that level.

RLTB (Right to Left, Top to Bottom): Use the CHECK(RLTB) keyword only at the file level. It specifies the direction the cursor is to advance from input-capable field to input-capable field. CHECK(RLTB) specifies that on exiting from a field, the cursor advances by moving from right to left and from top to bottom of the display until it reaches the next input-capable field. You can specify the RLTB cursor control code with only the edit/check code AB, as the others are not valid at the file level.

Note: Specifying CHECK(RLTB) does not change which input-capable field the cursor is positioned in when the display initially appears.

Figure 3-41 shows how to specify the CHECK(RLTB) keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        CHECK(RLTB)
00020A          R PROMPT
     A
```

Figure 3-41. Specifying the CHECK(RLTB) Keyword

CHGINPDFT (Change Input Default)

Use this file-, record-, or field-level keyword to change one or more input defaults for input-capable fields. Without parameter values, this keyword removes the underline for input-capable fields (input only or input/output). With parameter values, this keyword applies the specified display attributes or keyboard controls to the affected input-capable fields.

The format of the keyword is:

```
CHGINPDFT[(input-default1 input-default2 . . .)]
```

Valid parameter values for this keyword are:

Parameter Value	Equivalent DDS Keyword	Meaning
none	DSPATR(UL) specified but not selected	Remove underline
BL	DSPATR(BL)	Blinking field
CS	DSPATR(CS)	Column separators
HI	DSPATR(HI)	High intensity
RI	DSPATR(RI)	Reverse image
UL	DSPATR(UL)	Underline
FE	CHECK(FE)	Field exit
LC	CHECK(LC) or LOWER	Lowercase
ME	CHECK(ME)	Mandatory enter
MF	CHECK(MF)	Mandatory fill

Note: If DSPATR(UL) is specified for a field, the CHGINPDFT keyword cannot control underlining for that field.

The above equivalent DDS keywords apply only to output fields. For input and both fields, DSPATR(UL) also must be specified but not selected in addition to the equivalent keywords. This is because DSPATR(UL) is applied to input and both fields by default when CHGINPDFT is not specified.

Two common ways to use this keyword are to allow lowercase data entry for all input-capable fields in a record format or file and to specify column separators for all input-capable fields in a record format or file.

At the file level, this keyword applies to all input-capable fields in the file. At the record level, this keyword applies to all input-capable fields in the record format. At the field level, this keyword applies only to the fields for which it is specified.

If you specify the CHGINPDFT keyword at more than one level, the lower level keyword overrides the higher level keyword. Thus, specifying CHGINPDFT(BL) at the file level and CHGINPDFT(HI) for a record format causes all input-capable fields in the file except those in that record format to blink. In that record format, all input-capable fields are highlighted.

The CHGINPDFT keyword can be specified with any CHECK or DSPATR keyword. If you specify CHGINPDFT at the file, record, or field level, you can add check codes or display attributes to single fields by specifying CHECK or DSPATR at the field level. For instance, if you specify CHGINPDFT(CS) at the record level and DSPATR(HI) at the field level, the field is displayed with column separators and is highlighted. In addition, the CHECK or DSPATR keyword at the field level controls the check code or display attribute specified with it. For example, if you specify CHGINPDFT(CS) at the record level and DSPATR(CS) with option indicators at the field level, the setting of the option indicators controls the column separators for the field.

If you display a field with UL, RI, and HI in effect, no matter whether specified on the CHGINPDFT keyword, the DSPATR keyword, or a combination of both, the field is not displayed.

When specified at the file or record level, CHGINPDFT(LC) does not apply to numeric fields. If specified for a numeric field, CHGINPDFT(LC) is ignored.

CHGINPDFT(MF) is not allowed with CHECK(RB), CHECK(RZ), AUTO(RAB), and AUTO(RAZ).

Option indicators are not valid for this keyword.

Figure 3-42 shows how to specify the CHGINPDFT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                     CHGINPDFT
00020A            FLD1          10   B  1  2
00030A            FLD2          10      2  2
00040A          R RECORD2                     CHGINPDFT(CS)
00050A            FLD3          10   I  3  2
00060A            FLD4          10   B  4  2
00070A            FLD5          10   B  5  2
00080A  01                                    DSPATR(CS)
00090A            FLD6          10      6  2
00100A          R RECORD3                     CHGINPDFT(CS)
00110A            FLD7          10   I  7  2
00120A            FLD8          10   I  8  2
00130A  02                                    DSPATR(HI)
     A
```

Figure 3-42. Specifying the CHGINPDFT Keyword

CHGINPDFT is specified at the record level for RECORD1, RECORD2, and RECORD3:

- For RECORD1, CHGINPDFT removes the underline for FLD1.
- For RECORD2, CHGINPDFT causes the following:
  - FLD3 and FLD4 have column separators.
  - FLD5 has column separators only when DSPATR(CS) is selected.
  - FLD6 (an output-only field) has no column separators.
- For RECORD3, CHGINPDFT causes the following:
  - FLD7 and FLD8 have column separators.
  - FLD8 is also highlighted when DSPATR(HI) is selected.

CHKMSGID (Check Message Identifier)

Use this field-level keyword to identify an error message that is issued when a validity check error is detected. If the CHKMSGID keyword is not specified, a system-supplied message is used. The associated validity checking rules are specified on a CHECK(M10), CHECK(M11), CHECK(VN), CHECK(VNE), CMP, COMP, RANGE, or VALUES keyword.

The format of the keyword is:

```
CHKMSGID(message-id [library/]message-file [&message-data-field])
```

The message-ID parameter specifies the message description that contains the text to be displayed on the message line.

The message-file and library parameters identify the message file containing the message descriptions. The library name is optional. If it is not specified, the library list (*LIBL) that is in effect at run-time is used to search for the message file.

The message-data-field parameter specifies the name of the field that contains the message replacement text to be displayed on the message line. The format of the message-data-field parameter is &field-name where field-name is the name of the field containing the message replacement text. The field name must exist in the record format, and the field must be defined as a character field (data type A) with usage P.

CHKMSGID is allowed only on fields which also contain a CHECK(M10), CHECK(M11), CHECK(VN), CHECK(VNE), CMP, COMP, RANGE, or VALUES keyword. The field must be input-capable (usage B or I).

Option indicators are not valid for this keyword.

Figure 3-43 shows how to specify the CHKMSGID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     MSGLOC(20)
00020A          R RECORD1
00030A            FIELD1        10A  B  4  2CHECK(VN) CHKMSGID(USR1234 +
00040A                                     QGPL/USRMSGS &MSGFLD1)
00050A            MSGFLD1       12A  P
00060A            FIELD2         1A  I  4 20VALUES('A' 'B' 'I')
00070A                                     CHKMSGID(XYZ9999 APPLMSGS)
00080A            FIELD3         3S 0B  4 25RANGE(023 199)
     A
```

Figure 3-43. Specifying the CHKMSGID Keyword

When RECORD1 is read from the display screen:

- If FIELD1 does not contain a valid name, message USR1234 from the message file USRMSGS in library QGPL with the replacement text specified in MSGFLD1 is displayed on line 20.
- If the data entered into FIELD2 is not the letter A, B, or I, message XYZ9999 from *LIBL/APPLMSGS is displayed on line 20.
- If the data entered into FIELD3 is less than 023 or greater than 199, the system-supplied message CPF5224 (value for the field is not in a valid range) is displayed on line 20 since the CHKMSGID keyword is not specified.

CHOICE (Selection Field Choice)

Use this field-level keyword to define a choice for a selection field.

The format of the keyword is:

```
CHOICE(choice-number choice-text [*SPACEB])
```

The choice-number parameter defines an identification number for this choice. This parameter is required. The choice number returns to the application to indicate which choice in the selection field was selected. On nongraphical displays, the choice number is also displayed to the left of the choice text. Valid values for the choice-number are positive integers greater than 0 and less than or equal to 99. Duplicate choice-number values within a selection field are not allowed.

The choice-text parameter defines the text that appears in the selection field for the choice. This parameter is required. The parameter can be specified in one of two forms:

As a character string: 'Choice text '
As a program-to-system field: &field-name

The field specified must exist in the same record as the selection field and must be defined as a character field with usage P.

The choice text must fit on one line of the display for the smallest display size specified for the file. Therefore, the maximum length for the choice text depends on the:

- Position of the selection field
- Length of the choice number that is displayed to the left of the choice
- Length of the choice text itself

If the smallest display size is 24 x 80, the sum of the position of the field, the length of the input field, the length of the choice number, the spaces in between, plus the choice text must be less than or equal to 80. If the smallest display size specified is 27 x 132, this sum must be less than or equal to 132.

Within the choice text, you can specify a mnemonic for the choice by using a greater than character (>) to indicate the mnemonic character. The character to the right of the > is the mnemonic. The mnemonic is used only on a character-based graphical display attached to a controller that supports an enhanced interface for nonprogrammable work stations, where the choices are rendered using radio buttons. The mnemonic is ignored on displays where the field is rendered using numeric selection, since the system does not support both numeric and mnemonic selection on a selection field. Examples of specifying mnemonics:

Choice Text	Appears as
'>File'	File
'F>inish'	Finish
'Save >As...'	Save As...
'X >= 1'	X = 1

In order to specify > as a character in the text, you must specify it twice, just as you must specify the apostrophe character twice in order to get a single apostrophe character in the text. For example:

Choice Text	Appears as
'X >>= 1'	X >= 1
'X >>>= 1'	X >= 1

Note: It is not possible to specify the > as the mnemonic.

The mnemonic character indicated must be a single-byte character and must not be a blank. Only one mnemonic is allowed in the choice text, and the same mnemonic character cannot be specified for more than one choice.

The *SPACEB parameter is optional and indicates that a blank space (or line) should be inserted before this choice. This parameter is used to specify logical grouping of choices that are numbered consecutively. When the choice numbers are not consecutive, a blank space is automatically inserted between non-consecutive choices.

When the CHOICE keyword is specified on a field, either the SNGCHCFLD or the MLTCHCFLD keyword must also be specified.

Several CHOICE keywords can be specified for one selection field. The maximum number of CHOICE keywords that can be specified depends on the position of the selection field and the display size. Each choice occupies one line, and all choices must fit on the smallest display size specified for the file.

Option indicators are valid for this keyword. When a CHOICE keyword is optioned off, the list of choices is compressed.

Figure 3-44 shows how to specify the CHOICE keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8

     A          R RECORD
     A            F1             2Y 0B  1  2SNGCHCFLD
     A  01                                  CHOICE(1 '>Undo       ')
     A                                      CHOICE(2 &MARKTXT)
     A                                      CHOICE(3 '>Copy       ')
     A            MARKTXT        12A  P
     A
```

Figure 3-44. Specifying the CHOICE Keyword

In this example, three choices are defined for the single-choice selection field F1. The text for choice 2 is contained in field MARKTXT, and the mnemonic for choice 2 must be contained in the text supplied by the application at run time. If indicator 01 is off when the record is written, only choices 2 and 3 are displayed.

CHRID (Character Identifier)

Use this field-level keyword to specify that a named field be translated if the CHRID parameter value for the display file differs from the CHRID parameter value for the work station. This can be important when extended alphabetics (characters such as u with an umlaut or c with a cedilla) are to be displayed or typed in.

This keyword has no parameters.

If the CHRID keyword is not specified for a field and the CHRID value for the display file is not *JOBCCSID, data displayed in that field is displayed in the character set of the device used to type the data. How the data is displayed cannot be predicted and depends on how code points used in the original code page map to the code page used on the device.

For example, an è is displayed as a blank, because hex 51 is an empty code point in the basic U.S. graphic character set.

The CHRID keyword is not valid on constant fields, numeric fields (fields with decimal positions specified in positions 36 through 37), message fields (M specified in position 38), hidden fields (H specified in position 38), or program-to-system fields (P in Position 38).

The CHRID keyword is ignored if the CHRID value for the dislay file is *JOBCCSID.

The CHRID keyword cannot be specified with the DUP (Duplication) keyword.

If you specify the CHRID keyword with the DFT keyword on a field, the initial (default) value of the field is not translated, but data entered into that field is translated.

Option indicators are not valid for this keyword, although option indicators can be used to condition the field for which it is specified.

Figure 3-45 shows how to specify the CHRID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A            TITLE         40      1 20CHRID
     A
```

Figure 3-45. Specifying the CHRID Keyword

The field TITLE is a named field. With the CHRID keyword specified, character translation could occur on both output and input, depending on the conditions described in the *Guide to Programming Displays*.

CLEAR (Clear)

Use this file- or record-level keyword to specify that your program is to receive control if the work station user presses the Clear key, and optionally, that the OS/400 program is to set on the specified response indicator.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the CLEAR keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
CLEAR[(response-indicator ['text'])]
```

The Clear key is processed like a command attention key (no input data is transmitted from the device). The OS/400 program does not clear the display; your program must perform the desired function (such as clearing fields or records from the display).

If you do not specify this keyword and the display station user presses the Clear key, the OS/400 program displays a message indicating that it is not valid at that time.

Note: On display stations with the typewriter-like keyboard, the Clear key is activated by pressing CMD, then pressing the Shift key and the left arrow above the Field Exit key. On work stations with the data-entry keyboard, press CMD, then press the Shift key and the farthest right blank key on the top row.

The optional text is included on the printout created at program compilation time to explain the intended use of the indicator. This text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program printout.

Option indicators are valid for this keyword.

Figure 3-46 shows how to specify the CLEAR keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00101A                                    CLEAR(10 'Clear key pressed')
     A
```

Figure 3-46. Specifying the CLEAR Keyword

CLRL (Clear Line)

Use this record-level keyword to specify that the OS/400 program is to clear (delete) a specific number of lines before the record is displayed. Only those lines are cleared.

Note: As with OVERLAY, other records remain on the display.

See the *Guide to Programming Displays* for information on how to use CLRL in files that are used in the System/36 environment.

The format of the keyword is:

```
CLRL(nn|*END|*NO|*ALL)
```

You can specify the CLRL keyword in one of the following ways:

- Specify nn, where nn is an integer between 1 and 27. The number specified is the number of lines cleared, starting with and including the first line on which

the record is to be displayed. If the SLNO (Starting Line Number) keyword is also specified for this record format, the clearing of lines begins with the starting line number in effect for the record format at the time it is displayed.

Note: When specifying nn, the record must have at least one field defined.

- Specify *END to indicate that all lines starting with and including the first line on which the record is to be displayed are to be cleared. For a 24 x 80 display, lines up to and including line 24 are cleared. For a 27 x 132 display, lines up to and including line 27 are cleared.
- Specify *NO so that no lines on the display are cleared before displaying the record whose format you are defining. The displayed record overlays any data already on the display.
- Specify *ALL so that all of the lines on the display are cleared before displaying the record whose format you are defining. At least one field must be defined in the record format.

When a record format begins with a field in position 1, the beginning attribute byte of the format is in the last position of the previous line. The previous line number is the starting line number in the format. This also applies to a SLNO format with a field defined in the DDS in line 1, position 1.

If the record format for which the CLRL keyword is specified has one or more input-capable fields, any records that are overlaid are no longer recognized by the OS/400 program. That is, any input-capable fields can no longer be typed into, any input operation written to one of those records results in an error, and they cannot be cleared by selecting the ERASE keyword.

If you specify the CLRL(nn) keyword in a record format without input-capable fields, the input-capable fields in the overlapped records remain input-capable. That is, input-capable fields in the overlaid records remain input-capable, and input operations written to those record formats are still valid. If the ROLLUP or ROLLDOWN keywords are specified on the record containing the CLRL keyword, they are ignored. Records with the CLRL keyword and no input-capable fields are not cleared properly when they are overlaid by other records that have the OVERLAY keyword specified. The lines needed for the overlapping record are cleared and the lines not needed for the overlapping record remain on the display.

You can use the CLRL(*NO) keyword to prevent an overlapped record from being cleared when the overlapping record is written to the display. If you use this keyword, any records being displayed that are to be overlapped are not cleared from the display. The new record overlays them entirely or partially. There is a performance advantage to using CLRL(*NO) if you have a display containing constants and data that is repeatedly sent to the display. Sending constants as a separate format and using the CLRL(*NO) keyword for the format containing the data reduces the time required to send the record format to the display.

If the CLRL keyword is not specified and neither OVERLAY nor PUTOVR (Put with Explicit Override) is specified, the entire display is cleared.

If the CLRL keyword is used and the PUTOVR or PUTRETAIN keyword is in effect, the clearing of any lines may conflict with the PUTOVR or PUTRETAIN function. The PUTOVR or PUTRETAIN keyword requires that the fields being overridden be on the display, while the CLRL(nn) or CLRL(*END) keyword may clear those fields

first. If a record becomes unavailable for input because of the CLRL(nn) or CLRL(*END) keyword, the input-capable fields remain input-capable if the PUTOVR keyword is in effect. However, the OS/400 program sends a message if the program attempts to read such a record.

If you specify the CLRL keyword, you should also specify RSTDSP(*YES) on the Create Display File (CRTDSPF) or Change Display File (CHGDSPF) command. Otherwise, data on the display can be lost if the file is suspended.

Preventing Overlapped Records from Being Cleared

The CLRL keyword cannot be specified with any of the following keywords:

ASSUME	SFLCTL
KEEP	USRDFN
SFL	

A warning message appears at file creation time if the CLRL keyword is specified on a record with the DSPMOD keyword. At run time, the CLRL keyword is ignored when the display mode changes.

The CLRL keyword cannot be specified for the record format specified by the PASSRCD keyword.

Option indicators are not valid for this keyword.

Figure 3-47 shows how to specify the CLRL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                     CLRL(5)
00020A            FLD1           5      3  2
00030A            FLD2          10  OB  5  2
00040A            FLD3          10   I  6  2
00050A*
00060A          R RECORD2                     CLRL(*NO)
00070A            FLD1           5  2   2  2
00080A            FLD2           5   H
00090A            FLD3          10   I  4  2
00100A*
00110A          R RECORD3                     CLRL(*END)
00120A            FLD1           5   B  5  2
00130A            FLD2           5   I  8  2
     A
```

Figure 3-47. Specifying the CLRL Keyword

Lines 3, 4, 5, 6, and 7 are cleared before RECORD1 is displayed. In RECORD2, no lines are cleared, and when the record is displayed, it will overlay anything already displayed. Lines 5 through 24 are cleared before RECORD3 is displayed.

CMP (Comparison)

This keyword is equivalent to the COMP keyword.

The format of the keyword is:

```
CMP(relational-operator value)
```

The COMP keyword is preferred. See "COMP (Comparison)" on page 3-76 for an explanation of how to use these keywords.

CNTFLD (Continued-Entry Field)

Use this field-level keyword to define a field as a continued entry field. Continued-entry fields are sets of associated entry fields that are treated by the work station controller as a single field during field-data entry and editing. If the display device is not attached to a controller that supports an enhanced interface for nonprogrammable work stations, each segment of the continued entry field is treated separately when editing is performed on the field.

Figure 3-48 illustrates the use of continued fields to create a rectangular text entry field.

```
Enter Text . . .  _____________________________________
                  _____________________________________
                  _____________________________________
                  _____________________________________
                  _____________________________________
```

Figure 3-48. Continued-Entry Field in Rectangular Arrangement

The text input format is more appealing to the end user than a single input field that wraps across multiple display lines. Even though the last line does not occupy the full width of the column, no other field is allowed in the rectangle. A continued-entry field allows a multiple-row entry field to be defined inside a window.

The format of the keyword is:

```
CNTFLD(width of column)
```

One parameter must be specified.

The width of the column parameter specifies the number of columns to be used for this continued field. This value must fit within the width of the display or window. This value must be less than the length of the field.

The field containing the CNTFLD keyword must be defined as an input-capable field with the data type A. It cannot be defined in a subfile.

The following keywords cannot be specified on a field with the CNTFLD keyword:

- AUTO (RAB, RAZ)
- CHECK(AB, MF, RB, RZ, RLTB)
- CHOICE
- DSPATR(OID SP)
- EDTMSK

Option indicators are not valid for this keyword.

Figure 3-49 shows how to specify the CNTFLD keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD
00020A            F1            90A  B  3  4CNTFLD(30)
```

Figure 3-49. Specifying the CNTFLD Keyword

In this example, a multiple-row entry field is defined. The entry field contains 3 lines and is 30 columns wide.

COLOR (Color)

This field-level keyword specifies the color of a field on a color display (3179, 3197 Models C1 and C2, 3477 Model FC, 3486, 3487 Model HC, 3488[4] or 5292 Color Display Stations only). This keyword is ignored if it is selected for a field displayed on monochrome display stations. You can specify one parameter value for the COLOR keyword, but you can specify more than one COLOR keyword on each field.

The format of the keyword is:

```
COLOR(GRN | WHT | RED | TRQ | YLW | PNK | BLU)
```

The valid parameter values are:

Value	Meaning
GRN	Green
WHT	White
RED	Red
TRQ	Turquoise
YLW	Yellow
PNK	Pink
BLU	Blue

Because green is the default color of the fields on color display stations, you need to specify COLOR(GRN) only to keep the color of a field green. Specifying DSPATR(HI), DSPATR(CS), or DSPATR(BL) for a field changes the color of the field unless you also specify COLOR(GRN).

Option indicators are valid for this keyword.

When you specify the COLOR keyword more than once for a field, you must specify option indicators with each COLOR keyword. If more than one COLOR keyword is in effect for an output operation, the OS/400 program uses the first COLOR keyword that is specified in the DDS (see Figure 3-51 on page 3-75). You cannot specify the same color more than once for a field.

The number of COLOR keywords you can specify in one display file is limited by the maximum size of an internal storage area of the system called the screen attribute array. The maximum size of the screen attribute array is 32 763 bytes for the entire display file. Each COLOR keyword you specify in the file uses up a significant amount of storage within this array.

If you use many COLOR keywords in a file, particularly with conditioning, you should consider the amount of internal storage these keywords will require. If the 32 763-byte limit is exceeded, message CPF0673 (`Too many COLOR or DSPATR keywords specified in file`) is issued during file creation. To determine the amount of storage required for a particular COLOR keyword, use the following algorithm:

[4] Dependent on monitor attached to display device.

```
(# of conditions for the keyword) x 2 + 2 + 2⁹ = # of bytes required
                                                  in the screen
                                                  attribute array
                                                  for the keyword
```

For example, suppose a file contains 8 fields, each field contains 9 COLOR keywords, and each COLOR keyword is optioned using 3 conditions. Using the above algorithm, each COLOR keyword requires 520 bytes in the screen attribute array:

```
3 x 2 + 2 + 2⁹ = 520 bytes
```

Since there are 9 COLOR keywords per field and 8 fields in the file, the total storage required in the screen attribute array is 37 440 bytes (520 x 9 keywords x 8 fields). Since 37 440 is greater than 32 763, message CPF0673 is issued at file creation time.

The COLOR Keyword with the DSPATR Keyword: In some combinations of COLOR and DSPATR, both keywords have effect. Those combinations are:

COLOR	**DSPATR**
Any	RI (reverse image)
Any	UL (underline)
RED	BL (blinking field)
RED	BL and RI
RED	BL and UL
RED	RI and UL
GRN	RI and UL
TRQ	RI and UL
PNK	RI and UL

For example, if COLOR(YLW) and DSPATR(RI) are both in effect, the field appears as black characters on a yellow background.

In some combinations of the COLOR and DSPATR keywords, some of the parameter values are ignored. Those combinations are shown in the following table.

COLOR	DSPATR	Effect
Any	ND (nondisplay)	All colors are ignored
Any	HI (high intensity)	HI is ignored
Any	CS (column separators)	CS is ignored[1]
GRN	BL	BL is ignored[2]
WHT	BL	BL is ignored[2]
TRQ	BL	BL is ignored[2]
YLW	BL	BL is ignored[2]
PNK	BL	BL is ignored[2]
BLU	BL	BL is ignored[2]
RED	RI and BL and UL	UL is ignored[3]
YLW	RI and UL	RI is ignored
BLU	RI and UL	RI is ignored
WHT	RI and UL	RI is ignored

Notes:

1. Turquoise and yellow fields have column separators even if DSPATR(CS) is not specified. (The column separators appear as small blue dots between characters on color displays. They disappear when the display station user sets the color display station for reduced line spacing.)
2. The only color that can blink is red.
3. Underlines are also removed from input-capable fields, which are underlined by default on an AS/400 system.

For example, if COLOR(YLW) and DSPATR(HI) are both selected for an output operation, the field is yellow but not high intensity.

The DSPATR Keyword on Color Displays

When you specify the DSPATR keyword without the COLOR keyword, fields are displayed on color displays with the colors in the following table but without the specified display attributes.

Figure 3-50. DSPATR Keyword on Color Displays

DSPATR(CS) Display Attribute Selected	DSPATR(HI) Display Attribute Selected	DSPATR(BL) Display Attribute Selected	Color Produced on the Color Display Station
			Green (normal)
X			Turquoise[1]
	X		White
		X	Red, no blinking
	X	X	Red, with blinking
X	X		Yellow[1]
X		X	Pink
X	X	X	Blue

Note:

1. Turquoise and yellow fields are displayed with column separators (which are always blue) except when the work station user sets the color display station for reduced line spacing.

For example, if DSPATR(HI) is selected for a field and the COLOR keyword is not specified, the field is white but not highlighted on a color display.

The COLOR keyword is ignored if it is selected for a monochrome display.

Figure 3-51 shows the effects of specifying COLOR and DSPATR for a field.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD                        1
00020A                                        1  2'Column Heading'
00030A                                           DSPATR(HI)
00040A        2   FIELD1          5       3  2
00050A        3   FIELD2          5   I   5  2COLOR(YLW)
00060A        4   FIELD3          5       7  2DSPATR(BL)
00070A        5   FIELD4          5   I   9  2
00080A  42                                       COLOR(YLW)
00090A  43                                       COLOR(TRQ)
00100A  44                                       COLOR(BLU)
     A
```

Figure 3-51. Specifying the COLOR and DSPATR Keywords

1 On color displays, the constant field *Column Heading* is white; on monochrome displays, it is highlighted.

2 On all displays, FIELD1 is green.

3 On color displays, FIELD2 is yellow with blue column separators. On all displays, the field is underlined because it is an input-capable field.

4 On color displays, FIELD3 is red and does not blink; on monochrome displays, it blinks.

5 On color displays, FIELD4 can appear in one of the following colors:

- Green, if no indicators are on
- Yellow, if indicator 42 is on (no matter how other indicators are set)
- Turquoise, if indicator 43 is on and indicator 42 is off
- Blue, if indicator 44 is the only indicator on

On monochrome displays, FIELD4 is green. On all displays, FIELD4 is underlined.

Figure 3-52 shows one way of specifying a field for use as an input-capable field on both color and monochrome displays.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD
00020A 2  1     FIELDA          5   B  2  2COLOR(TRQ)
00030A  44                                  ERRMSG('Record not found' 44)
     A
```

Figure 3-52. Specifying a Field for Color and Monochrome Displays

1 On color displays, FIELDA is turquoise with blue column separators; on monochrome displays, it is green.

2 If option indicator 44 is set on when FIELDA is displayed, the ERRMSG keyword is in effect and causes the following:

- On color displays, FIELDA is turquoise and its image is reversed. (Because of the COLOR keyword, it is not highlighted.) The error message *Record not found* is displayed on the message line in white.

- On monochrome displays, FIELDA is highlighted and its image is reversed. The error message *Record not found* is also highlighted and is displayed on the message line.

COMP (Comparison)

Use this field-level keyword to specify that the OS/400 program is to compare the data that the work station user types into an input or output/input field with the specified value. The relational operator is the criterion for the comparison. If the data typed in this field fails this validity check, the OS/400 program displays an error message. Note that the OS/400 program performs this checking only if the field is changed by the work station user or if its modified data tag (MDT) is set on using DSPATR(MDT).

Note: Refer to the CHKMSGID keyword for information on defining user-specified error messages.

The format of the keyword is:

```
COMP(relational-operator value)
```

You can specify only one operation for the COMP keyword and only one COMP keyword for a field.

The valid entries for the relational operator are:

Relational Operator	Meaning
EQ	Equal
NE	Not equal
LT	Less than
NL	Not less than
GT	Greater than
NG	Not greater than
LE	Less than or equal
GE	Greater than or equal

The specified value must be either numeric or character, depending on the data type (decimal positions entry). Numeric values are expressed by the digits 0 through 9 and a leading sign (+ or –). Character values must be enclosed in apostrophes.

Note: If the field you are defining is numeric, alignment is based on the decimal positions specified (in positions 36 and 37), and leading and trailing blanks are filled with zeros. If no decimal point is typed in, the decimal point is assumed to be to the right of the last (farthest right) digit. For example, for a numeric field with a length of 5 (specified in position 34) and 2 decimal positions (specified in position 37), 1.2 is interpreted as 001.20, and 100 is interpreted as 100.00.

You cannot specify the COMP keyword on a floating-point field (F in position 35).

Option indicators are not valid for this keyword.

Figure 3-53 on page 3-77 shows how to specify the COMP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            FIELD2         6  OI 10 10COMP(EQ +021920)
00020A            FIELD1         3   I 11 11COMP(EQ 'ABC')
     A
```

Figure 3-53. Specifying the COMP Keyword

CSRLOC (Cursor Location)

Use this record-level keyword to specify the cursor location on an output operation to the record format you are defining. Your program sends the output operation after setting the cursor location.

The format of the keyword is:

```
CSRLOC(field-name-1 field-name-2)
```

The parameter values on the keyword specify the names of two fields whose contents are the line number (for field-name-1) and the position number (for field-name-2) of the cursor location. Field-name-1 and field-name-2 are 3-byte, zoned decimal, hidden fields. Your program uses these fields to tell the OS/400 program where to locate the cursor.

The cursor is not positioned to the desired location on an output operation that leaves the keyboard locked. The cursor does not move to the desired position until your program sends an input or an output operation that unlocks the keyboard. If your program sets the cursor location fields to values outside the range of values valid for the display device, this keyword is ignored.

For any one output operation, the CSRLOC keyword overrides any other cursor location specifications, such as DSPATR(PC) and SFLRCDNBR(CURSOR), that are in effect. This keyword is in effect until your program sends another output operation with DSPATR(PC), CSRLOC, or SFLRCDNBR(CURSOR) in effect or until the record in which this keyword is specified is overlaid (OVERLAY keyword) or erased (ERASE keyword), whichever comes first.

On an input operation, the cursor location can be determined by looking at the I/O feedback area. Use the data in the I/O feedback area can be used when you specify the parameter values of the CSRLOC keyword. (See the *Guide to Programming Displays* for more information on the I/O feedback area.)

Specify the CSRLOC keyword only once per record format.

The CSRLOC keyword is not valid for the following record formats.

- Subfile record formats (identified by the SFL keyword)
- User-defined record formats (identified by the USRDFN keyword)

Option indicators are valid for this keyword. Display size condition names are not valid.

Figure 3-54 on page 3-78 shows how to specify the CSRLOC keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   CSRLOC(LINNBR POSNBR)
00020A            TITLE         40   B  1  2
00030A            PAGE           5Y OB  1 60
00040A            TEXT        1760   B  2  1
00050A            LINNBR         3  OH
00060A            POSNBR         3  OH
     A
```

Figure 3-54. Specifying the CSRLOC Keyword

The application program sets the contents of LINNBR and POSNBR before issuing an output operation to RECORD1. When the record is displayed, the fields *Title*, *Page*, and *Text* appear on the display. The cursor could be at some location in a field of RECORD1 where the work station user is to make changes.

DATE (Date)

Use this field-level keyword to display the current job date as a constant (output-only) field 6 bytes long. You can specify the location of the field, the DATE keyword, and, optionally, the EDTCDE, EDTWRD, COLOR, DSPATR, or TEXT keyword. Positions 17 through 38 must be blank.

This keyword has no parameters.

If you specify EDTCDE(Y) for a DATE field, the display length is 8 positions. Specifying EDTCDE(Y) changes the date from

```
mmddyy
```

to

```
mm/dd/yy
```

where the slashes (/) represent the job attribute DATSEP at run time, and the order of month, day, and year is determined by the job attribute DATFMT. (DATFMT can be MDY, DMY, YMD, or JUL, where M=month, D=day, Y=year, and JUL=Julian. DATSEP can be a slash (/), dash (-), period (.), or comma (,).)

Option indicators are not valid for this keyword, although option indicators can be used to condition the field on which it is specified.

Figure 3-55 shows how to specify the DATE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  20                            1 56DATE
00020A  21                            1 56DATE EDTCDE(Y)
     A
```

Figure 3-55. Specifying the DATE Keyword

The date is displayed without editing if option indicator 20 is on. The date is displayed with editing if option indicator 20 is off and option indicator 21 is on.

DFT (Default)

Use this field-level keyword to specify the constant value for constant fields (unnamed fields) and to specify a default value for named fields.

The format of the keyword is:

```
DFT('value') | 'value'
```

The maximum number of characters you can specify in the literal is set by the size of the display on which the field is to be displayed as follows:

Size of Display	Maximum Characters
24 x 80	1919
27 x 132	3563

Constant Fields: The value of a constant field can be specified as a value enclosed by apostrophes. (For other ways to specify a constant field, see the DATE, MSGCON, and TIME keywords.) You can omit the DFT keyword itself, as well as the parentheses, to simplify the DDS. Whether you specify the DFT keyword explicitly or implicitly, the OS/400 program displays the specified value as a constant field on the display. See "Name (Positions 19 through 28)" on page 3-7 for a description of constant fields.

Named Fields: For input-only fields, the specified value is displayed each time the field is displayed. The displayed value can then be changed by the work station user and returned to your program.

For output-only and output/input fields, you must also specify PUTOVR at the record level and OVRDTA at the field level with the DFT keyword. The specified value is displayed only on the first output operation. On subsequent output operations, the program value is displayed. For more information and an example, see "PUTOVR (Put with Explicit Override)" on page 3-178.

The DFTVAL, EDTCDE, and EDTWRD keywords cannot be specified with the DFT keyword.

The DFT keyword is not valid on floating point fields.

Option indicators are not valid for this keyword, although option indicators can be used to condition the field (whether constant or named) on which it is specified.

Figure 3-56 shows how to specify the DFT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A            HOTTYP         1   I  7  9DFT('D')
00040A                                       VALUES('D' 'S')
00050A                                     8  9'ON'
00060A        01                          12  1'HOTEL NAME: ''TERRACE INN'''
00070A                                       TEXT('Constant field is +
00080A                                       conditioned, not the implicit +
00090A                                       DFT keyword')
00100A        02                          12  1'HOTEL NAME: ''RIVER VIEW INN'''
00110A                                       TEXT('Either ''TERRACE INN'' or +
00120A                                           ''RIVER VIEW INN'' could +
00130A                                           appear in line 12, position 1')
     A
```

Figure 3-56. Specifying the DFT Keyword

The constant field *ON*, having no option indicators, is always displayed.

If indicator 01 is on, the following is displayed:

```
HOTEL NAME:  'TERRACE INN'
```

If indicator 02 is on, and indicator 01 is off, the following is displayed:

```
HOTEL NAME:  'RIVER VIEW INN'
```

If you are specifying a constant field for more than one display size, and you are changing the location of the field but not the contents of the field for the different display sizes, then do not repeat the value. Figure 3-57 shows how to do this.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                  DSPSIZ(*DS3 *DS4)
     A                                  :
     A                                  :
00080A                             22   2'Constant data'
00090A                             26   2
     A
```

Figure 3-57. Specifying a Constant Field with More Than One Display Size

The constant field *Constant data* appears on line 22, position 2, on the 24 x 80 display screen, and it appears on line 26, position 2, on the 27 x 132 display.

DFTVAL (Default Value)

Use this field-level keyword to specify a default value for an output-capable field. On the first output operation, the specified value is displayed if the option indicator is on or has not been specified. Otherwise, the program value is used. On subsequent output operations, the program value appears.

The format of the keyword is:

```
DFTVAL('value')
```

This keyword is valid on output-only (O) or output-input (B) fields.

You can only use this keyword to initialize named fields. It is not allowed on constant fields.

Since the maximum number of characters on a DDS statement is 5000, this keyword, along with any other keywords specified on the DDS statement, must contain less than 5000 characters.

You cannot use this keyword in a subfile format (SFL keyword).

You cannot specify the DFTVAL keyword on the same field with a DFT, EDTCDE (Edit Code), or EDTWRD (Edit Word) keyword, or on a floating-point field.

Option indicators are valid for this keyword.

Figure 3-58 on page 3-81 shows how to specify the DFTVAL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1
     A  50                                 PUTOVR
     A            FIELD1         3A  B 12 01DFTVAL('AAA') OVRDTA
     A            FIELD2         3D 00 12 05OVRDTA
     A  10                                 DFTVAL('000')
     A            FIELD3         3D 00 12 09DFTVAL('000') OVRDTA
     A
```

Figure 3-58. Specifying the DFTVAL Keyword

In Figure 3-58, prior to displaying the record, the application program assigns "ZZZ" to FIELD1, "999" to FIELD2, and "456" to FIELD3. On the first output operation, "AAA 000 000" displays if indicator 10 is on; "AAA 999 000" displays if indicator 10 is off.

The work station user types "XXX" into FIELD1. On the second output operation, "XXX 999 456" displays if indicator 50 is on; 'AAA 000 000' displays if indicator 50 is off and indicator 10 was on during the first output operation. 'AAA 999 000' displays if indicator 50 is off and indicator 10 was off during the first output operation.

DLTCHK (Delete Check)

Use this field-level keyword to specify that the OS/400 program is to ignore all validity checking and CHKMSGID keywords that are specified for a referenced field. This keyword is valid only when R is specified in position 29.

This keyword has no parameters.

If you specify *any* new validity checking keywords, DLTCHK is unnecessary. The new validity checking keywords override the referenced validity checking keywords.

Option indicators are not valid for this keyword.

Figure 3-59 shows how to specify the DLTCHK keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A                                 REF(FILE)
00040A          R RECORD
00050A            CODE      R        I  3 20DLTCHK
     A
```

Figure 3-59. Specifying the DLTCHK Keyword

DLTEDT (Delete Edit)

Use this field-level keyword to specify that the OS/400 program is to ignore the EDTCDE or EDTWRD keyword if one of them is specified for a referenced field. This keyword is valid only when R is specified in position 29.

This keyword has no parameters.

If you specify a new editing keyword, DLTEDT is unnecessary. The new editing keyword overrides the referenced editing keyword.

Option indicators are not valid for this keyword.

Figure 3-60 on page 3-82 shows how to specify the DLTEDT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00040A                                      REF(FILEA)
00050A          R RECORD
00060A            AMT       R        B  5 20DLTEDT
     A
```

Figure 3-60. Specifying the DLTEDT Keyword

DSPATR (Display Attribute)

Use this field-level keyword to specify one or more display attributes for the field you are defining. You can specify the DSPATR keyword more than once for the same field, and you can specify more than one attribute for the same keyword. However, each attribute (for example, UL), can be specified only once per field.

The format for the keyword is one of the following:

```
DSPATR(attribute-1 [attribute-2 [attribute-3 [...]]])
                    or
DSPATR(&program-to-system-field)
```

If you specify more than one attribute for the same field, whether in one keyword or in separate keywords, each attribute that is specified (and in effect when the field is displayed) affects the field. For example, if you want a field to be displayed with its image reversed and with high intensity, specify either DSPATR (RI HI), or DSPATR(RI), and DSPATR(HI).

The program-to-system-field parameter is required and specifies that the named field must be defined in the record format, alphanumeric (A in position 35), length of two, and usage P (P in position 38). The program uses this P-field to set the display attribute for the field this DSPATR keyword applies to.

The name P-field is used for multiple fields with the record being defined. One DSPATR P-field is allowed per field. The P-field contains the display attribute and identifies whether the field should be protected. See "Valid P-field Values" on page 3-86.

The following are valid attributes for the first format of the DSPATR keyword:

For All Fields

Display Attribute	Meaning
BL	Blinking field
CS	Column separator
HI	High intensity
ND	Nondisplay
PC	Position cursor
RI	Reverse image
UL	Underline

For Input-Capable Fields Only

Display Attribute	Meaning
MDT	Set changed data tag when displayed
OID	Operator identification
PR	Protect contents of field from input keying
SP	Select by light pen

Notes:

1. If you specify the UL, HI, and RI attributes on the 5250 display station for the same field, the result is the same as if you had specified ND.
2. If OID is specified, then SP should not be specified. Neither OID nor SP can be optioned unless specified with another display attribute.
3. Display attributes BL, CS, HI, RI, and UL can also be specified at the file, record, or field level as parameter values on the CHGINPDFT keyword.
4. Display attributes CS, HI, and BL can cause fields on the 5292, 3477 Model FC, 3487 Model HC, 3179, 3197 Model C1 and C2, and 3488[5] color display stations to appear as color fields. See "COLOR (Color)" on page 3-72 for more information.
5. If you are using an IBM Personal System/2* (PS/2)* computer that is emulating a 5250 display station and you are directly changing the EBCDIC screen buffer, you need to set the MDT attribute. See the *IBM Personal Computer Enhanced 5250 Emulation Program Technical Reference* manual for additional information.
6. If you are using a PS/2 computer and VGA monitor, the UL attribute does not work due to hardware specific limitations in the way buffers are used.

Option indicators are valid for this keyword, except when the attributes OID or SP are the only display attributes specified.

Detailed descriptions of each of the attributes follow the coding example and sample display provided in Figure 3-61 on page 3-84.

[5] Dependent on monitor attached to display device.

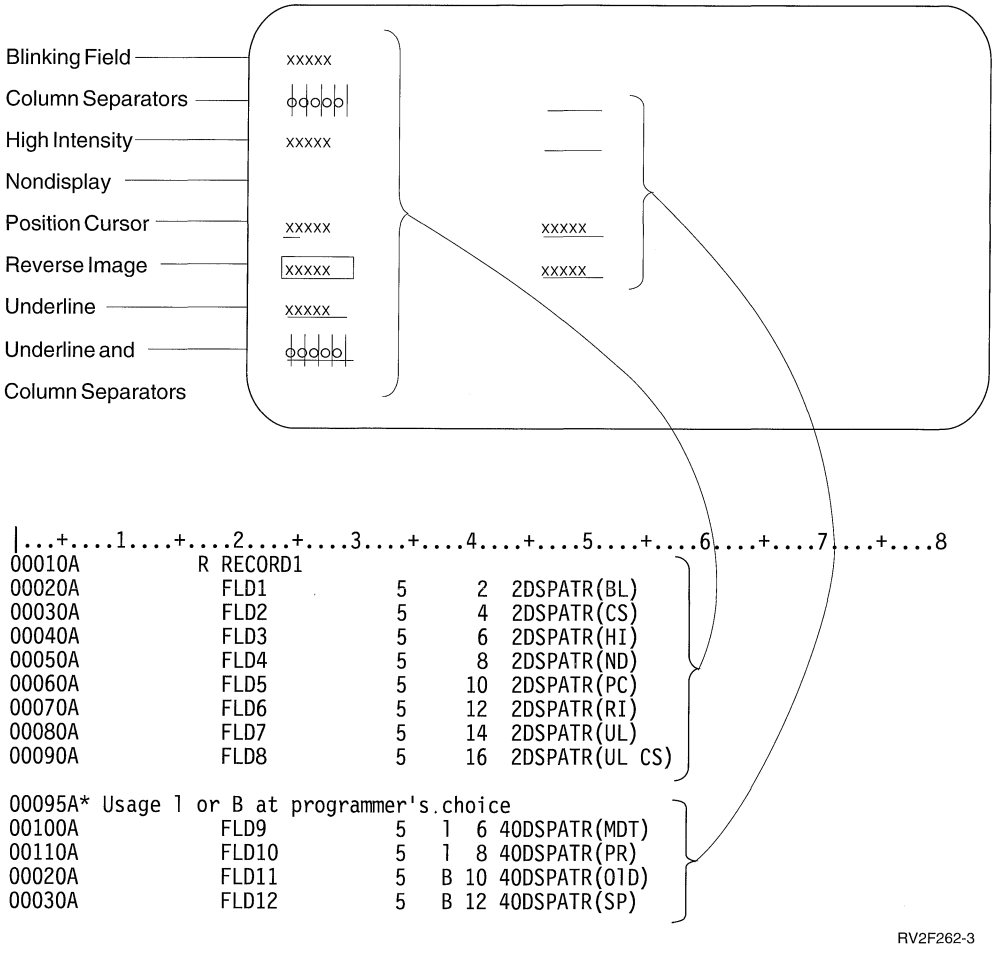


Figure 3-61. A 5-Byte Field Displayed with Various Display Attributes

Display Attributes for All Fields

BL (Blink): Use this attribute to specify that the field is to blink when it is displayed.

CS (Column Separator): Use this attribute to specify that each position of the field is to be displayed with a vertical bar at its left and right edge. When specified for a nondisplay field, the separators are displayed even though there are no characters between them. You can use column separators to precisely indicate cursor positioning within a field and to indicate the length of an otherwise blank field.

HI (High Intensity): Use this attribute to specify that the field is to be intensified (highlighted) when it is displayed on the display.

ND (Nondisplay): Use this attribute to specify that the field is not to be displayed; the display positions for this field appear blank. The attribute can be used for passwords or other security-sensitive data. If the print function (permitted by specifying the PRINT keyword) is performed, nondisplay fields are not printed.

PC (Position Cursor): Use this attribute to position the cursor to the first character position of the field you are defining. You can specify this attribute for several fields, and the cursor will be positioned at the first selected field with this attribute.

Note that the fields within a record are ordered in line/position sequence as they appear on the display and not necessarily in the order you specify them.

RI (Reverse Image): Use this attribute to specify that the image of the field is to be reversed from the other portion of the screen when it is displayed. Whether the screen is light-on-dark or dark-on-light depends on the status of the display prior to displaying the field. This setting is controlled by the work station user.

UL (Underline): Use this attribute to specify that the field is to be underlined when it is displayed. All input-capable fields are underlined by default. See the CHGINPDFT keyword to prevent the default underlining. (If CHGINPDFT is specified, DSPATR(UL) must be specified to underline an input-capable field.) If DSPATR(UL) is specified with option indicators and the option indicators are not satisfied (DSPATR(UL) not selected), the field appears without underline.

Display Attributes for Input-Capable Fields

MDT (Set Changed Data Tag): Use this attribute to specify that the OS/400 program is to set on the changed data tag (MDT) for the field you are defining when the field is written to the display. The attribute ensures that the field is sent from the device when the record is read from the display.

Note: The OS/400 program saves output data for output/input fields or initialized data for fields with the DFT keyword specified. This causes the saved data to be returned on an input operation if no new (changed) data is entered into the field.

OID (Operator Identification): Use this attribute to specify that the OS/400 program is to allow magnetic stripe reader OID data to be entered into this field. If it is to be a nondisplay field also, the DSPATR(ND) attribute must be specified.

A field with the DSPATR(OID) keyword functions like any other input-capable field; data can be entered from either the keyboard or the magnetic stripe reader. The DSPATR(OID) keyword can be specified (but is not required) to indicate that data can be entered using a magnetic stripe reader. You can key into the field unless the keyboard shift Inhibit Keyboard Entry (I) is specified. If both DSPATR(OID) and DSPATR(SP) are specified on the same field, DSPATR(SP) is ignored.

PR (Protect): Use this attribute to specify that the work station user cannot key into the input-capable field you are defining. This attribute is valid for input-capable fields only. Output-only fields and constant fields are protected by definition.

SP (Select by Light Pen): Use this attribute to specify that this input-capable field can be selected by a light pen. The work station user can key into a light pen field unless an I (Inhibit Keyboard Entry) has been specified in position 35 (Data Type/Keyboard Shift) for the field.

When the field is first displayed, the contents of the field are set by your program (output/input field) or in the DDS (input-only field with DFT keyword or character string). If no new data is typed in by the work station user, this output data is returned to your program on an input operation.

A field that can be selected by a light pen should be at least 3 bytes long. The recommended contents of this field are:

- A switch character, either hex 6F (?) or, if the work station user selects the field by a light pen, hex 6E (>)
- A blank (hex 40)
- A target character, which can be any character, such as an asterisk (*)
- Another blank
- Additional data to identify the field to the work station user (1 or more characters)

This attribute is useful only for work stations with a light pen feature for selecting.

Valid P-field Values

The DSPATR P-field does not support the following display attributes:

Display Attribute	Meaning
MDT	Set changed data tag when displayed
OID	Operator identification
PC	Position cursor
SP	Select by light pen

Valid P-field Values (Nonprotect)

Figure 3-62 (Page 1 of 2). P-field values (nonprotect)

Hex	Limited Color	Full Color
20	Normal	Green
21	Reverse image	Green, reverse image
22	High intensity	White
23	High intensity, reverse image	White, reverse image
24	Underscore	Green, underscore
25	Underscore, reverse image	Green, underscore, reverse image
26	Underscore, high intensity	White, underscore
27	Nondisplay	Nondisplay
28	Blink	Red
29	Blink, reverse image	Red, reverse image
2A	Blink, high intensity	Red, high intensity
2B	Blink, high intensity, reverse image	Red, high intensity, reverse image
2C	Blink, underscore	Red, underscore
2D	Blink, underscore, reverse image	Red, underscore, reverse image
2E	Blink, underscore, high intensity	Red, underscore, blink
2F	Nondisplay	Nondisplay
30	Column separator	Turquoise, column separator
31	Reverse image, column separator	Turquoise, column separator, reverse image
32	High intensity, column separator	Yellow, column separator

Figure 3-62 (Page 2 of 2). P-field values (nonprotect)

Hex	Limited Color	Full Color
33	High intensity, reverse image, column separator	White, reverse image, column separator
34	Underscore, column separator	Turquoise, underscore, column separator
35	Underscore, reverse image, column separator	Turquoise, underscore, reverse image, column separator
36	Underscore, high intensity, column separator	Yellow, underscore, column separator
37	Nondisplay	Nondisplay
38	Blink, column separator	Pink
39	Blink, reverse image, column separator	Pink, reverse image
3A	Blink, high intensity, column separator	Blue
3B	Blink, high intensity, reverse image, column separator	Blue, reverse image
3C	Blink, underscore, column separator	Pink, underscore
3D	Blink, underscore, reverse image, column separator	Pink, underscore, reverse image
3E	Blink, underscore, high intensity, column separator	Blue, underscore
3F	Nondisplay	Nondisplay

Valid P-Field Values (Protect)

Figure 3-63 (Page 1 of 2). P-field values (protect)

Hex	Limited Color	Full Color
A0	Normal	Green
A1	Reverse image	Green, reverse image
A2	High intensity	White
A3	High intensity, reverse image	White, reverse image
A4	Underscore	Green, underscore
A5	Underscore, reverse image	Green, underscore, reverse image
A6	Underscore,high intensity	White, underscore
A7	Nondisplay	Nondisplay
A8	Blink	Red
A9	Blink, reverse image	Red, reverse image

Figure 3-63 (Page 2 of 2). P-field values (protect)

Hex	Limited Color	Full Color
AA	Blink,high intensity	Red, high intensity
AB	Blink, high intensity, reverse image	Red, high intensity, reverse image
AC	Blink, underscore	Red, underscore
AD	Blink, underscore, reverse image	Red, underscore, reverse image
AE	Blink, underscore, high intensity	Red, underscore, blink
AF	Nondisplay	Nondisplay
B0	Column separator	Turquoise, column separator
B1	Reverse image, column separator	Turquoise, column separator, reverse image
B2	High intensity, column separator	Yellow, column separator
B3	High intensity, reverse image, column separator	White, reverse image, column separator
B4	Underscore, column separator	Turquoise, underscore, column separator
B5	Underscore, reverse image, column separator	Turquoise, underscore, reverse image, column separator
B6	Underscore, high intensity, column separator	Yellow, underscore, column separator
B7	Nondisplay	Nondisplay
B8	Blink, column separator	Pink
B9	Blink, reverse image, column separator	Pink, reverse image
BA	Blink, high intensity, column separator	Blue
BB	Blink, high intensity, reverse image, column separator	Blue, reverse image
BC	Blink, underscore, column separator	Pink, underscore
BD	Blink, underscore, reverse image, column separator	Pink, underscore, reverse image
BE	Blink, underscore, high intensity, column separator	Blue, underscore
BF	Nondisplay	Nondisplay

Figure 3-64 on page 3-89 shows how to specify the DSPATR(SP) keyword with an input-only field (showing the recommended data contents as a character string).

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00110A           SPFLD         50I  I  5  4'? * OPTION 1'
00120A                                     DSPATR(SP)
     A
```

Figure 3-64. Specifying the DSPATR(SP) Keyword with Input-Only Field

No data can be typed into field SPFLD. When the field is selected with the light pen, the data returned in field SPFLD would be:

```
>_*_OPTION_1
```

where _ represents a blank.

Figure 3-65 shows that when the work station user selects a field with the light pen, both the MDT bit and the first character of that field are changed. When the field is selected, the MDT bit is set on, changing the first character of the field to >. If the same field is selected again, the MDT bit is set off and the first character becomes ?.

By specifying a switch character, your program prevents the first character of data from being changed to > or ? when the field is selected by the light pen. If the MDT bit is on when your program sends an input operation to the record format, the contents of the field are returned to your program as a user-changed field.

If you use DSPATR(MDT) to set on the MDT of a field that can be selected by the light pen, then you should either omit the MDTOFF keyword from other record formats or you should read that field before displaying any record format with MDTOFF in effect.

Figure 3-65 shows what could happen.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A         R LIGHTPEN
00020A           FLD1          10   I  5  2'> * $12.50'
00030A                                     DSPATR(SP MDT)
00040A*
00050A         R RCD2                      OVERLAY MDTOFF
00060A           FLD1          10   B 11  2
     A
```

Figure 3-65. Specifying the DSPATR(MDT) Keyword to Set on a Field for Selection by Light Pen

If the program displays LIGHTPEN, then displays RCD2, then reads LIGHTPEN, and the work station user does not select FLD1 with the light pen, the MDT of FLD1 is turned off by the display of RCD2.

Also, the switch character of FLD1 is returned as ?, even though the field was not selected, and the switch character appears as >. The MDT and the switch character are in opposing states.

Figure 3-66 on page 3-90 shows how to specify the DSPATR keyword with P-field usage:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD                    2DSPATR(&PFLD1)
     A            FLD1           5A     2  6DSPATR(&PFLD2)
     A            FLD2           5A     2  6DSPATR(&PFLD2)
     A            PFLD1          1A  P
     A            PFLD2          1A  P
     A
```

Figure 3-66. Specifying the DSPATR keyword with P-Fields

DSPMOD (Display Mode)

Use this record-level keyword to specify which of two modes (display sizes) you want to use for the 3180, 3477, or the 3197 Model D1, D2, W1, or W2 display station. *DS3 (24 x 80) and *DS4 (27 x 132) are both supported for the 3180, 3477, 3487 Models HA, HC, HG, and HW, 3488, and the 3197 Model D1, D2, W1, or W2 display stations.

The format of the keyword is:

```
DSPMOD(condition-name)
```

This keyword is valid only when both the 24 x 80 and 27 x 132 display sizes are specified on the DSPSIZ keyword. The first of the two display sizes specified on the DSPSIZ keyword is the default display mode. The record is displayed using this mode unless the DSPMOD keyword indicates that the second specified display size should be used.

Note: This keyword is a run-time keyword and not a compile-time keyword.

You can specify the default display size with this keyword only if you do not specify option indicators for this keyword.

The capability to display in the 27 x 132 mode is allowed only on a 3180-2 or a 3197 Model D1, D2, W1, or W2 device attached locally to a 6040 or 6140 controller or remotely to a 5294 or 5394 controller. The DSPMOD keyword is ignored unless these controllers are used.

When a record with DSPMOD causes the mode to be changed, all records currently on the display are deleted. The record with DSPMOD active is then sent to the display. The mode for this record is maintained on the display as long as the DSPMOD keyword is active. Setting DSPMOD off or writing to another record without DSPMOD causes the display mode to be placed back in the primary display size for the device.

The following keywords are ignored if the display modes have changed:

ALWROL
ASSUME
CLRL
ERASEINP/INZINP
ERRMSG
ERRMSGID
KEEP
OVERLAY
PROTECT
PUTOVR
PUTRETAIN
SFLMSG
SFLMSGID

When you create a file specifying any of the above keywords and DSPMOD on the same record, a warning message results at create-time. However, a diagnostic message is *not* issued during processing.

This keyword is not valid for user-defined records (USRDFN keyword).

The DSPMOD keyword cannot be specified on a subfile record (SFL keyword). The subfile is be displayed according to the DSPMOD of the corresponding subfile control record.

Option indicators are valid for this keyword. If the option indicator is on at the time of processing, the display mode you have chosen will be used to display the record. However, if the option indicator is off at the time of processing, the default display mode will be used.

Note: Switching display modes is similar to displaying a record without OVERLAY.

Figure 3-67 and Figure 3-68 show how to specify the DSPMOD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      DSPSIZ(*DS3 *DS4)
     A          R RECORD1                   DSPMOD(*DSP4)
     A          R RECORD2
     A          R RECORD3
     A  03                                  DSPMOD(*DS4)
     A
```

Figure 3-67. Specifying the DSPMOD Keyword (Example 1)

The DSPMOD keyword gives the following results:

- If you write RECORD1, RECORD1 is displayed in *DS4 mode.
- If you write RECORD2, the display is cleared and RECORD2 is displayed in *DS3 mode.
- If you write RECORD3 with indicator 03 off, RECORD3 is displayed in *DS3 mode. RECORD2 remains on the display.
- If you write RECORD3 with indicator 03 on, the display is cleared and RECORD3 is displayed in *DS4 mode.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      DSPSIZ(24 80 *NORM +
     A                                            27 132 *WIDE)
     A          R RECORD1
     A  03                                  DSPMOD(*WIDE)
     A
```

Figure 3-68. Specifying the DSPMOD Keyword (Example 2)

DSPRL (Display Right to Left)

Use this file-level keyword to specify that the records in the display file are written right to left on the display rather than left to right.

This keyword has no parameters.

Option indicators are not valid for this keyword.

Figure 3-69 on page 3-92 shows how to specify the DSPRL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      DSPRL
     A          R RECORD
     A            FIELD1       20A      5  5')Customer Name(:'
     A
```

Figure 3-69. Specifying the DSPRL Keyword

DSPSIZ (Display Size)

Use this file-level keyword to specify the display size to which your program can open this display file.

The formats of the keyword are:

```
DSPSIZ(*DSw [*DSx])
DSPSIZ(lines positions[condition-name-1][lines positions[condition-name-2]])
```

The DSPSIZ keyword is optional. If you do not specify it for a display file, the display file can be opened only to display devices with a 24 x 80 display size. You can specify this keyword in one of two ways:

- Using IBM-supplied display size condition names. Specify up to two parameter values as *DS3 or *DS4 in any order. At least one parameter value is required. You cannot specify a parameter value twice.
- Specifying lines and positions to permit user-defined display size condition names. Instead of the IBM-supplied display size condition names, specify the display size in lines and positions (only 24 x 80, and 27 x 132 are valid). (See Figure 3-71 on page 3-93, Figure 3-73 on page 3-95, and Figure 3-74 on page 3-95.)

  Optionally, you can also define a display size condition name other than *DS3 or *DS4. The display size condition name you define must be from 2 to 8 characters long, and the first character must be an asterisk (*). You can specify these user-defined condition names in positions 7 through 16 (conditioning) on subsequent DDS statements at the field level. (See Figure 3-73 on page 3-95.) If you do not specify user-defined display size condition names, you must use IBM-supplied display size condition names to condition the location of fields.

  If you specify more than one parameter value, see "Primary and Secondary Display Sizes" later in this section.

  Option indicators are not valid for this keyword.

Primary and Secondary Display Sizes

Whether you use IBM-supplied display size condition names or specify lines and positions directly, the first display size you specify is the *primary display size*. The second display size, if specified, is the *secondary display size*. Figure 3-70 shows an example of primary and secondary display size specification.

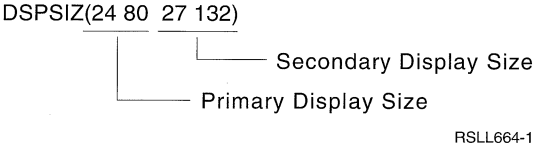


Figure 3-70. Using DSPSIZ to Specify Primary and Secondary Display Sizes

When you specify more than one display size for DSPSIZ, you can specify display size condition names in positions 7 through 16 on subsequent DDS statements at the record and field levels. These display size condition names are then used to condition keywords and the locations of fields. For more information, see "Conditioning (Positions 7 through 16)" on page 3-2.

Note: If you specify user-defined display size condition names for DSPSIZ, you cannot use IBM-supplied display size condition names for conditioning.

The capability to display in the 27 x 132 mode is allowed only on a 3180-2, or a 3197 Model D1, D2, W1, or W2 device attached locally to a 6040 or 6041 controller or remotely to a 5294 or 5394 controller. The display size for the 27 x 132 mode will be ignored for DSPSIZ unless these controllers are used.

The following table shows the valid display sizes.

Display Sizes	Display Device	Meaning
*DS3 or 24 x 80	3179 3180 3196 3197 3476 3486 3487 (Models HA, HC, HG, and HW) 3488 5251 (Models 11 and 12) 5291 5292	24 lines x 80 positions 1920 positions total
*DS4 or 27 x 132	3180 3197 (Models D1, D2, W1, and W2) 3477 (Models FA, FC, FD, and FG) 3487 (Models HA, HC, HG, and HW) 3488 (Use 6040 or 6041 controller locally, or 5294 or 5394 controller remotely for 27 x 132 display capability.)	27 lines x 132 positions 3564 positions total

The display size designated as the primary display size should be the one with which the display file will most often be used. Additional processing is performed when the actual display size is the secondary display size.

The display size condition names let you improve the use of a single display file for any size display. For example, when you are using subfiles, you can specify 24 records per page for a 27 x 132 display and 22 records per page for a 24 x 80 display.

Figure 3-71, and Figure 3-73 on page 3-95 through Figure 3-75 on page 3-95 show how to specify the DSPSIZ keyword and the display size condition names.

Figure 3-71 shows how to specify primary and secondary display sizes using the DSPSIZ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*                                                  1    2
00030A                                      DSPSIZ(27 132 24 80)
00040A          R RECORDA
00050A            FIELDA        10  0   1  2
00060A            FIELDB        10  0   1 81
00070A            FIELDC        10  0  25  1
     A
```

Figure 3-71. Specifying the DSPSIZ Keyword (Example 1)

In Figure 3-71, the primary display size **1** is 27 x 132 and the secondary display size **2** is 24 x 80. FIELDB is beyond position 80 and FIELDC is beyond line 24, so the data description processor gives them a location of *NOLOC in the expanded source printout for secondary display size 24 x 80.

If the data description processor assigns *NOLOC to an input-capable field, that field is processed at run time to the point of setting up the input buffer data to be returned in the user's input buffer. The field itself is not displayed. The work station user cannot enter into or change these fields. No processing of any kind is done for output-only fields.

Figure 3-72 shows a compiler listing for Figure 3-71 on page 3-93.

```
5728SS1 R01 M00 880311               DATA DESCRIPTION             QGPL/DSPSIZ1          01/28/88 14:16:46       PAGE    1
FILE NAME . . . . . . . . . . . . . . . . . . . . :   DSPSIZ1
  LIBRARY NAME  . . . . . . . . . . . . . . . . . :   QGPL
FILE ATTRIBUTE  . . . . . . . . . . . . . . . . . :   DISPLAY
SOURCE FILE CONTAINING DDS  . . . . . . . . . . . :   QDDSSRC
  LIBRARY NAME  . . . . . . . . . . . . . . . . . :   QGPL
SOURCE MEMBER CONTAINING DDS  . . . . . . . . . . :   DSPSIZ1
SOURCE MEMBER LAST CHANGED  . . . . . . . . . . . :   09/04/87  14:48:36
SOURCE LISTING OPTIONS  . . . . . . . . . . . . . :   *SOURCE    *LIST      *NOSECLVL
DDS GENERATION SEVERITY LEVEL . . . . . . . . . . :   20
AUTHORITY . . . . . . . . . . . . . . . . . . . . :   *CHANGE
TEXT  . . . . . . . . . . . . . . . . . . . . . . :   SAMPLE DDS LISTING
COMPILER  . . . . . . . . . . . . . . . . . . . . :   IBM AS/400 DATA DESCRIPTION PROCESSOR
                                  DATA DESCRIPTION SOURCE
SEQNBR  *...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8  DATE
   100  00010A*                                                                          09/04/87
   200  00020A*                                                                          08/10/87
   300  00030A                                DSPSIZ(27 132 24 80)                       08/10/87
   400  00040A          R RECORDA                                                        08/10/87
   500  00050A            FIELDA        10  0   1  2                                  08/10/87
   600  00060A            FIELDB        10  0   1 81                                  08/10/87
   700  00070A            FIELDC        10  0  25  1                                  08/10/87
                        * * * * *  E N D  O F  S O U R C E  * * * * *
```

```
5728SS1 R01 M00 880311               DATA DESCRIPTION             QGPL/DSPSIZ1          01/28/88 14:16:46       PAGE    2
                                      EXPANDED SOURCE
                                                                                    FIELD       BUFFER POSITION
SEQNBR  *...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8 LENGTH       OUT     IN
   300                                   DSPSIZ(27 132 24 80)
   400                R RECORDA
   500                  FIELDA        10S 00  1  2                                    10           1
   600                  FIELDB        10S 00  1 81 }                                  10          11
   600       * *DS3                        *NOLOC  }
   700                  FIELDC        10S 00 25  1 }                                  10          21
   700       * *DS3                        *NOLOC  }
               * * * * *  E N D  O F  E X P A N D E D  S O U R C E  * * * * *
```

FIELDB and FIELDC have no location on *DS3, which corresponds to 24 80.

```
5728SS1 R01 M00 880311               DATA DESCRIPTION             QGPL/DSPSIZ1          01/28/88 14:16:46       PAGE    3
                                          MESSAGES
   ID      SEVERITY  NUMBER
                                    MESSAGE SUMMARY
   TOTAL          INFORMATIONAL      WARNING       ERROR       SEVERE
                      (0-9)          (10-19)      (20-29)      (30-99)
      0                0               0            0            0
* CPC7301      00             MESSAGE . . . . :   FILE DSPSIZ1 CREATED IN QGPL.
                        * * * * *  E N D  O F  C O M P I L A T I O N  * * * * *
```

RSLL911-3

Figure 3-72. Compiler Listing

Figure 3-73 on page 3-95 is another example of specifying the primary and secondary display sizes using the DSPSIZ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      DSPSIZ(27 132 *WIDE 24 80 *NORMAL)
00020A          R RECORDA
00030A            FIELDA        10  0   1  2
00040A            FIELDB        10  0   1 81
00050A  *NORMAL                           1 50
00060A            FIELDC        10  0  25  1
00070A  *NORMAL                          23  1
     A
```

Figure 3-73. Specifying the DSPSIZ Keyword (Example 2)

Figure 3-73 is similar to the example in Figure 3-71 on page 3-93, except that valid secondary locations are specified for FIELDB (line 1, position 50) and for FIELDC (line 23, position 1) on the secondary display size (user-defined as *NORMAL).

Figure 3-74 shows how to reposition a field when the file is opened to different display sizes.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      DSPSIZ(24 80 27 132)
00020A          R RECORDA
00030A            FIELD1        10  0  23  2
00040A
00050A
00060A  *DS4                             26  2
     A
```

Figure 3-74. Specifying the DSPSIZ Keyword (Example 3)

In Figure 3-74, FIELD1 has valid locations on both display sizes. It appears on the next to the last line on each display size.

Figure 3-75 shows that if you do not specify a display size condition name, the display location of a field can still be display size dependent as the result of the *plus* feature of DDS.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      DSPSIZ(*DS4 *DS3)
00020A          R RECORD1
00030A            FIELD1        21      2 70
00040A            FIELD2        10       +10
     A
```

Figure 3-75. Specifying the DSPSIZ Keyword (Example 4)

In Figure 3-75, a line and a position for each field is calculated for each display size specified on the DSPSIZ keyword. If the plus value extends the field location beyond position 80, the field location is dependent on the display size. Figure 3-76 on page 3-96 is a compiler listing for Figure 3-75.

```
5728SS1 R01 M00 880311              DATA DESCRIPTION            QGPL/DSPSIZ4          01/28/88 14:16:52       PAGE    1
FILE NAME . . . . . . . . . . . . . . . . . . . . :   DSPSIZ4
  LIBRARY NAME  . . . . . . . . . . . . . . . . . :   QGPL
FILE ATTRIBUTE  . . . . . . . . . . . . . . . . . :   DISPLAY
SOURCE FILE CONTAINING DDS  . . . . . . . . . . . :   QDDSSRC
  LIBRARY NAME  . . . . . . . . . . . . . . . . . :   QGPL
SOURCE MEMBER CONTAINING DDS  . . . . . . . . . . :   DSPSIZ4
SOURCE MEMBER LAST CHANGED  . . . . . . . . . . . :   09/03/87  10:16:55
SOURCE LISTING OPTIONS  . . . . . . . . . . . . . :   *SOURCE    *LIST      *NOSECLVL
DDS GENERATION SEVERITY LEVEL . . . . . . . . . . :   20
AUTHORITY . . . . . . . . . . . . . . . . . . . . :   *CHANGE
TEXT  . . . . . . . . . . . . . . . . . . . . . . :   SAMPLE DDS LISTING
COMPILER  . . . . . . . . . . . . . . . . . . . . :   IBM AS/400 DATA DESCRIPTION PROCESSOR
                                  DATA DESCRIPTION SOURCE
SEQNBR  *...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8  DATE
   100  00010A                                DSPSIZ(*DS4 *DS3)
   200  00020A          R RECORD1
   300  00030A            FIELD1        21      2 70
   400  00040A            FIELD2        10       +10
                        * * * * *  E N D   O F   S O U R C E  * * * * *
```

```
5728SS1 R01 M00 880311              DATA DESCRIPTION            QGPL/DSPSIZ4          01/28/88 14:16:52       PAGE    2
                                     EXPANDED SOURCE
                                                                                    FIELD       BUFFER POSITION
SEQNBR  *...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8 LENGTH        OUT     IN
   100                                    DSPSIZ(*DS4 *DS3)
   200                R RECORD1
   300                  FIELD1        21A  O  2 70                                  21           1
   400                  FIELD2        10A  O  2101                                10           22
   400          *DS3                         3 21
               * * * * *  E N D   O F   E X P A N D E D   S O U R C E  * * * * *
```

FIELD1 has same location on *DS4 and *DS3
FIELD2 has two locations:

- Line 2, position 101 on *DS4
- Line 3, position 21 on *DS3

```
5728SS1 R01 M00 880311              DATA DESCRIPTION            QGPL/DSPSIZ4          01/28/88 14:16:52       PAGE    3
                                        MESSAGES
   ID      SEVERITY  NUMBER
                                   MESSAGE SUMMARY
   TOTAL          INFORMATIONAL      WARNING        ERROR       SEVERE
                      (0-9)          (10-19)       (20-29)      (30-99)
       0               0               0             0            0
* CPC7301      00             MESSAGE . . . . :   FILE DSPSIZ4 CREATED IN QGPL.
                        * * * * *  E N D   O F   C O M P I L A T I O N  * * * * *
```

RSLL912-3

Figure 3-76. Compiler List

Special Cases: The following are descriptions of special cases you might encounter when specifying DSPSIZ:

- DSPSIZ(*DS3 *DS4). All field locations for display size *DS4 are the same as for display size *DS3.
- All fields of a record can be defined such that none fit on the display size to which the file is opened. In this case, no fields are displayed. The record is handled as it would be for a larger display where the fields fit. The record remains active until it is deleted or overlaid. Active records can be read by your program. The input request is sent to the display device, and the work station user must respond to satisfy the request.
- All fields of a subfile record must fit within the specified subfile page, and the complete page must always fit (vertically) on the display size on which it is displayed at processing time. Specify valid display sizes by conditioning the SFLPAG (Subfile Page) keyword with display size condition names.
- The following records occupy no display space:
  - Records with no fields defined (this is different from none selected)

  - Records with only hidden, message, or program-to-system fields
  - Records that have the CLRL keyword specified and that have no input-capable fields. (These records can remain on the display, but are not recognized by the OS/400 program for input operations, or they can be cleared through the use of the ERASE keyword.)

  For implementation and programming purposes, these records are assumed to be located at 00 (from line 0 to line 0). On an output operation, any record located at 00 overlays a record at that location. When an overlap occurs, the previous record is disregarded and no longer considered active. The new record at location 00 is active and can be read by your program.

- If two fields in a record format have the same display location (line/position), they are treated as overlapping fields. Overlapping fields are not displayed at operation time. The OS/400 program checks each field as it is processed to ensure that it does not overlap a previously processed field.

  If a field does overlap, it is treated as an optioned field and not selected. To allow this processing-time checking, data description specifications must ensure all fields within a record are in primary location sequence, even when condition names are specified. For example, assume only one input field is specified for a record format and, according to the field location specification, this field overlaps a preceding output field. The work station user cannot enter any data because the input field is never displayed.

Note: The primary location sequence as it is seen in the display file must not be changed by specifying a different location sequence for the secondary display size. (A severe error occurs and the file is not created.)

DUP (Duplication)

Use this field-level keyword to activate the Dup key on the display station keyboard. Press the Dup key when the cursor is in this input-capable field. This indicates that data for this field is to be duplicated from the record sent in the previous input operation. The actual duplication is the responsibility of your program (see "Programming for the Dup Key" later in this section).

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the DUP keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
DUP[(response-indicator ['text'])]
```

You cannot specify the DUP keyword on a floating-point field (F in position 35).

You should specify a response indicator for a numeric field defined with this keyword. Hex 1Cs for numeric fields will not be returned to your buffer, but hex F0s will be returned for the remaining field positions.

Option indicators are valid for this keyword.

Programming for the Dup Key: When you press the Dup key, the OS/400 program handles the field as follows:

- If the field is a character field, the data displayed in the field is returned to your

program as is. A hex 1C is placed at the cursor position and in the remaining field positions to the right. (Hex 1C appears as an over-scored asterisk on the display.) The response indicator, if specified, is set on.

- If the field is a numeric field and you specify a response indicator, a hex F0 is placed at the cursor position and in the remaining field positions. The response indicator is set on and returned to your program. If a response indicator is not specified, hex 1Cs are returned to your program.

In your program, you can duplicate entire fields (either character or numeric) with the following procedure:

1. Specify, in DDS, two fields for each input-capable field on the display.
   a. Specify one field as an input-capable field. For this field, specify DUP with a response indicator. You may want to specify DUP with an option indicator that is off on the first display of the field. This prevents the work station user from using the Dup key when the field is first displayed.
   b. Specify the other field as a hidden field (H in position 38).
2. On the first output operation, set off an option indicator for DUP. This prevents the work station user from using the Dup key.
3. On the first input operation, move the input-capable field to the hidden field. This saves the keyed value for later use.
4. On each subsequent output operation, set the option indicator on for the DUP keyword. This allows the work station user to use the Dup key.
5. On each subsequent input operation, test the response indicator specified with DUP. If the response indicator is off, the input data can be moved to the hidden field. If the response indicator is on, you can use the existing value in the hidden field.
6. Repeat steps 4 and 5 for subsequent data entry using the Dup key.

You can also duplicate character fields one character at a time by saving them in arrays, then moving the array one character at a time and checking for the DUP key indication of hex 1C.

You can achieve duplication of numeric fields one digit at a time by defining the field as character and eventually moving it to your numeric field after the hex 1Cs have been removed. You can test whether the Dup key has been pressed:

- For numeric fields, a response indicator is required.
- For character fields, a response indicator is optional.

The field will contain hex 1C at the cursor position and in the remaining positions if the Dup key has been pressed.

Restrictions on Validity Checking: Validity checking keywords (CHECK, COMP, RANGE, and VALUES) can be specified with the DUP keyword. However, they have no effect if the Dup key has been pressed.

If another field in the record format fails validity checking, the OS/400 program tries to read the display again. Whether or not the work station user keys over the DUP characters or presses the Dup key again, the DUP response indicator is still returned to your program in the on condition.

Figure 3-77 on page 3-99 shows how to specify the DUP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A            FLDA           5   I  3  2
00030A  15                                  DUP(16 'FLDA was duped')
00040A            FLDAH          5  2H
     A
```

Figure 3-77. Specifying the DUP Keyword

FLDA is an input-capable character and hexadecimal field for which the work station user can press the Dup key. In your program, you must test the data in FLDA to ensure that it includes only the digits 0 through 9. In RPG III, you can use the TESTN operation code. In COBOL, you can use a numeric class test. In BASIC, you can specify an ON CONV statement earlier in the program than the statement with which you assign FLDA to FLDAH. In PL/I, you can specify an ON statement with the ERROR condition earlier in the program than the statement with which you assign FLDA to FLDAH.

Field FLDAH is a hidden numeric field with the same length as FLDA. When the application program reads RECORD1 and finds response indicator 16 set off, the program moves FLDA to FLDAH and uses FLDAH (which is a numeric field). When the application program reads RECORD1 and finds response indicator 16 set on, the Dup key was pressed and FLDA contains hex 1Cs. The program uses FLDAH without changing its contents (which are the contents from the previous input received from the display device).

EDTCDE (Edit Code)

Use this field-level keyword to edit output-capable numeric fields.

The format of the keyword is:

```
EDTCDE(edit-code [* |floating-currency-symbol])
```

Editing includes the following changes to the appearance of displayed fields, depending on which edit code is specified:

- Leading zeros are suppressed.
- The field can be punctuated with commas and periods to show decimal position and to group digits by threes.
- Negative values can be displayed with a minus sign or CR to the right.
- Zero values can be displayed as zero or blanks.
- Asterisks can be displayed to the left of significant digits to provide asterisk protection.
- A currency symbol (corresponding to the system value QCURSYM) can be displayed immediately to the left of the significant digit that is farthest to the left (called *floating-currency symbol*). For fixed-currency symbols, use the EDTWRD keyword.
- The field can be further edited using a user-defined edit code. See "User-Defined Edit Codes" on page 3-102 for more information.

EDTCDE covers most editing requirements. Use EDTWRD when the EDTCDE keyword is not sufficient.

The EDTCDE keyword is valid only for fields with Y or blank in position 35 (Data Type/Keyboard Shift). The use of this keyword changes the default used for position 35 to a Y.

You cannot specify both EDTCDE and EDTWRD for the same field. If a field previously defined in a database file has EDTCDE specified, you need not specify EDTCDE for that field in the display file. You can specify R in position 29 to refer to the previously defined field. The editing specified for the referenced field is included in the display file. However, if you also specify length, data type, or decimal positions for a display file field, editing specified for the referenced field is not included in the display file, and you must specify editing again in the display file.

The DFT and DFTVAL keywords cannot be specified with the EDTCDE keyword.

Option indicators are not valid for this keyword.

The rules for specifying edit codes and edit words are the same in all types of files. You can specify two kinds of edit codes: OS/400 edit codes and user-defined edit codes.

OS/400 Edit Codes

The OS/400 edit codes are the following:

1 through 4
A through D
J through Q
X through Z

Note: The AS/400 system hardware operates with a preferred sign of F, which is equivalent to using edit code X. Edit code X causes a blank keyboard shift (position 35) to default to numeric-only (attribute Y). The display length of the field is determined by the keyboard shift and not by edit code X (the default numeric only Y attribute may add 1 position to the field for decimals). If the DATE or TIME keyword is specified with edit code X, the separator character is not displayed.

Asterisk Fill or Floating Currency Symbol: You can optionally specify asterisk fill or floating currency symbol with edit codes 1 through 4, A through D, and J through Q.

When you specify asterisk fill, an asterisk (*) is printed for each zero that is suppressed. A complete field of asterisks is printed for a zero balance field.

When you specify floating currency symbol, the symbol appears to the left of the first significant digit. The symbol does not print on a zero balance when an edit code is used that suppresses the zero balance. (The symbol that you specify must match the system value for the currency symbol (QCURSYM). The symbol must match when the file is created. It does not have to match when the file is used.)

Note: If an edit code is changed after a file is created, the editing specified at the time the file was created is used. The new edit code is not used unless the file is re-created.

The following table summarizes the functions provided by OS/400 edit codes.

Figure 3-78. Summary Chart for OS/400 Edit Codes

Edit Codes	Commas[1] Displayed	Decimal Points[1] Displayed	Sign Displayed When Negative Value	Blank Value of QDECFMT System Value	I Value of QDECFMT System Value	J Value of QDECFMT System Value	Leading Zero Suppressed
1	Yes	Yes	No sign	.00 or 0	,00 or 0	0,00 or 0	Yes
2	Yes	Yes	No sign	Blanks	Blanks	Blanks	Yes
3		Yes	No sign	.00 or 0	,00 or 0	0,00 or 0	Yes
4		Yes	No sign	Blanks	Blanks	Blanks	Yes
A	Yes	Yes	CR	.00 or 0	,00 or 0	0,00 or 0	Yes
B	Yes	Yes	CR	Blanks	Blanks	Blanks	Yes
C		Yes	CR	.00 or 0	,00 or 0	0,00 or 0	Yes
D		Yes	CR	Blanks	Blanks	Blanks	Yes
J	Yes	Yes	-(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
K	Yes	Yes	-(Minus)	Blanks	Blanks	Blanks	Yes
L		Yes	-(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
M		Yes	-(Minus)	Blanks	Blanks	Blanks	Yes
N	Yes	Yes	-(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
O	Yes	Yes	-(Minus)	Blanks	Blanks	Blanks	Yes
P		Yes	-(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
Q		Yes	-(Minus)	Blanks	Blanks	Blanks	Yes
Y[2]							Yes
Z[3]							Yes

Notes:

1. The QDECFMT system value determines the decimal point character (period in U.S. usage), the character used to separate groups of three digits (comma in U.S. usage), and the type of zero suppression (depending on comma and period placement).
2. The Y edit code suppresses the farthest left zero of a date field that is three to six digits long, and it suppresses the two farthest left zeros of a field that is seven positions long. The Y edit code also inserts slashes (/) between the month, day, and year according to the following pattern:

   nn/n
   nn/nn
   nn/nn/n
   nn/nn/nn
   nnn/nn/nn

   If the DATE keyword is specified with EDTCDE(Y), the separator character used is the job attribute, DATSEP at run time. The slash (/) is the default DATSEP. If, at file creation time, DATFMT is JUL (Julian), the date is normally formatted as nn/nnn and EDTCDE(Y) is not valid.
3. The Z edit code removes the sign (plus and minus) from a numeric field. The sign of the units position is changed to a hexadecimal F before the field is written.

Note: For more information on the QDECFMT system value, see the *Guide to Programming Displays*.

User-Defined Edit Codes

Edit codes 5 through 9 are user-defined edit codes. A user-defined edit code can do more editing than an OS/400 edit code. For example, you may need to edit numbers that include hyphens (such as telephone numbers) or more than one decimal point. You can use user-defined edit codes for these functions. These edit codes are named QEDIT5, QEDIT6, QEDIT7, QEDIT8, and QEDIT9, and can be referred to in DDS or a high-level language program by number (5, 6, 7, 8, or 9).

A user-defined edit code is an OS/400 object and must exist before display file creation. It is created using the Create Edit Description (CRTEDTD) command. When you create a display file in which a user-defined edit code is specified, editing information is extracted from the previously created edit description. Changing a user-defined edit code after display file creation does not affect the display file unless the display file is re-created.

The following table shows edit codes, unedited source data, and edited output. Zero suppression and decimal characters are determined by the system value QDECFMT. The date separator character is determined by the job attribute DATSEP. In this figure, QDECFMT is assumed to equal x (blank), and DATSEP is assumed to equal / (slash).

Figure 3-79. Valid Edit Codes, Source Data, and Edited Output

Edit Codes	Positive Number– Two Decimal Positions	Positive Number– No Decimal Positions	Negative Number– Three Decimal Positions[1]	Negative Number– No Decimal Positions[1]	Zero Balance– Two Decimal Positions[1]	Zero Balance– No Decimal Positions[1]
Unedited	1234567	1234567	xxxx.125–	xxxx.125–	xxxxxx	xxxxxx
1	12,345.67	1,234,567	.125	125	.00	0
2	12,345.67	1,234,567	.125	125		
3	12345.67	1234567	.125	125	.00	0
4	12345.67	1234567	.125	125		
A	12,345.67	1,234,567	.125CR	125CR	.00	0
B	12,345.67	1,234,567	.125CR	125CR		
C	12345.67	1234567	.125CR	125CR	.00	0
D	12345.67	1234567	.125CR	125CR		
J	12,345.67	1,234,567	.125–	125–	.00	0
K	12,345.67	1,234,567	.125–	125–		
L	12345.67	1234567	.125–	125–	.00	0
M	12345.67	1234567	.125–	125–		
N	12,345.67	1,234,567	–.125	–125	.00	0
O	12,345.67	1,234,567	–.125	–125		
P	12345.67	1234567	–.125	–125	.00	0
Q	12345.67	1234567	–.125	–125		
Y[2]	123/45/67	123/45/67	0/01/25	0/01/25	0/00/00	0/00/00
Z[3]	1234567	1234567	125	125		

Notes:

1. The x represents a blank.
2. The Y edit code suppresses the farthest left zero of a date field that is three to six digits long, and it suppresses the two farthest left zeros of a field that is seven positions long. For more information, see the second footnote in Figure 3-78 on page 3-100.
3. The Z edit code removes the sign (plus or minus) and suppresses leading zeros.

Figure 3-80 shows how to specify the EDTCDE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            PRICE          5  2   1 10EDTCDE(J)
00020A            SALES          7  2   2 10EDTCDE(K $)
00030A            SALARY         8  2   3 10EDTCDE(1 *)
     A
```

Figure 3-80. Specifying the EDTCDE Keyword

The display length for PRICE is 7 because the J edit code is specified, causing the field to contain a decimal point and an ending minus sign. It is edited as:

```
ddd.dd-
```

where d represents a digit.

The display length for SALES is 11 because the K edit code and floating currency symbol are specified. It is edited as:

```
$dd,ddd.dd-
```

The display length for SALARY is 10 because the edit code 1 is specified with asterisk fill. It is edited as:

```
ddd,ddd.dd
```

EDTMSK (Edit Mask)

Use this field-level keyword to define an edit mask for fields with EDTCDE or EDTWRD keywords. When a field is displayed with this keyword, user-specified areas of the field are protected. The EDTMSK keyword is ignored when the work station is not attached to a controller that supports an enhanced data stream.

The format of the keyword is:

```
EDTMSK(edit mask)
```

One parameter *must be* specified.

The edit mask is made of two characters; an ampersand (&), and a blank ( ). The ampersand represents a protected part of the field. A blank represents an unprotected part of the field. The edit mask must equal the display length of the field (after editing), and the number of unprotected positions must at least equal the program length of the field. The user must be careful to protect only nonnumeric data because protected data is not returned to the user if the field is changed.

The field containing the EDTMSK keyword must be usage I or usage B. It must also contain the EDTCDE or EDTWRD keywords.

The following keywords cannot be specified on a field with the EDTMSK keyword:

- AUTO (RAB, RAZ)
- CHECK(AB, MF, RB, RZ, RLTB)
- CHOICE
- CNTFLD
- DSPATR(OID SP)

Option indicators are not valid for this keyword.

If the ENTFLDATR keyword is specified with the EDTMSK keyword and the first position of the field is protected, then the ENTFLDATR keyword is ignored.

Figure 3-81 shows how to specify the EDTMSK keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD
     A            F1            11  0B  3  4EDTWRD('0(   )   -    ')
     A                                       EDTMSK(' &   &   &    ')
     A            F2             6  0B  4  4EDTCDE(Y)
     A                                       EDTMSK('  &  &  ')
     A
```

Figure 3-81. Specifying the EDTMSK Keyword

In this example, the dash and parentheses in field 1 are protected. Also, the data separators in field 2 are protected.

EDTWRD (Edit Word)

Use this field-level keyword to specify an edit word if you cannot obtain the desired editing by using the EDTCDE keyword.

The format of the keyword is:

```
EDTWRD('edit-word')
```

An edit word specifies the form in which the field values are to be displayed and clarifies the data by inserting characters directly, such as decimal points, commas, floating- and fixed-currency symbol, and credit balance indicators. The edit word can also be used to suppress leading zeros and to provide asterisk fill protection.

If a field previously defined in a database file has EDTWRD specified, you need not specify EDTWRD for that field in the display file. You can specify R in position 29 to refer to the previously defined field. The editing specified for the referenced field is included in the display file. However, if you also specify length, data type, or decimal positions for a display file field, editing specified for the referenced field is not included in the display file, and you must specify editing again in the display file.

Parts of an Edit Word

An edit word consists of three parts: the body, the status, and the expansion. Figure 3-82 shows the three parts of an edit word.

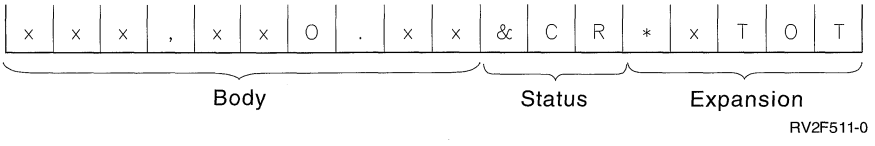


Figure 3-82. Three Parts of an Edit Word

The body is the space for the digits transferred from the data field to the output record. The body begins at the farthest left position of the edit word. The number of blanks (plus one zero or an asterisk) it contains is equal to the number of digits of the data field to be edited. The body ends with the farthest right character that can be replaced by a digit.

The status positions display the sign (+ or –) of the data field. The status continues to the right of the body to either a CR (credit) or – (minus) symbol. These two symbols print only when the field is negative. Edit words without the CR or – symbol have no status positions.

The expansion positions are not changed by the edit operation. The expansion starts at the first position to the right of the status (or body, if status is not specified) and ends with the farthest right character of the edit word.

Forming the Body of an Edit Word

The following characters have special meanings when used in the body of an edit word.

Blank: A blank is replaced with the character from the corresponding position of the data field. A blank position is referred to as a digit position.

Ampersand: An ampersand causes a blank in the edited field. The ampersand is not displayed. Note that ampersands specified in the edit word between blanks can result in incorrect data when specified for an output/input field. This is because embedded blanks in a numeric-only field are converted to zeros.

Zero: A zero stops zero suppression. Place it in the farthest right position where zero suppression is to stop. The zero is replaced with the character from the corresponding position of the data field, unless that character is a zero. Any zeros in the data that appear to the right of the stop-zero-suppression character are displayed. The stop-zero-suppression character is considered a digit position. At least one leading zero is suppressed. Each zero that is suppressed is replaced by a blank. An asterisk replaces zeros with asterisks (asterisk protection). Place the asterisk in the farthest right position where zero suppression is to stop. Each zero that is suppressed is replaced by an asterisk.

Asterisk: An asterisk preceding a zero is interpreted as representing asterisk protection, and in this case, the zero prints as a constant. Any asterisks or zeros to the right of the stop-zero-suppression character are constants.

Currency Symbol: If you code a currency symbol immediately to the left of the zero suppression code, a currency symbol is inserted in the position to the left of the first significant digit. It is called the *floating-currency symbol* when used in this manner.

If you code a currency symbol in the farthest left position of the edit word, it is fixed and prints in the same location each time. When used in this manner, it is called the *fixed-currency symbol*.

The currency symbol is not considered a digit replace position. This symbol must correspond to the system value QCURSYM.

Decimals and Commas: Decimals and commas are printed in the same relative positions in which they are coded in the edit word unless they are to the left of the first significant digit. In that case, they are blanked out or replaced by an asterisk.

All other characters are printed if they are to the right of significant digits in the edit word. If they are to the left of the high-order significant digits in the edit word, they are blanked out or replaced by asterisks if asterisk protection is being used.

If a constant is to be printed in the left most position, the constant must be preceded by a zero and the field length increased by one.

Forming the Status of an Edit Word

The following characters have special meanings when used in the status of an edit word.

Ampersand: Causes a blank in the edited output field. An ampersand cannot be placed in the edited output field.

CR or minus symbol: If the sign in the edited output field is plus (+), these positions are blanked out. If the sign in the edited output field is minus (–), these positions remain undisturbed.

Forming the Expansion of an Edit Word

The characters in the expansion portion of an edit word are always written. The expansion cannot contain blanks. If a blank is required in the edited output field, specify an ampersand in the body of the edit word.

Specifying a Valid Edit Word: Use the following rules to specify a valid edit word:

- The EDTWRD keyword is valid for numeric only fields (Y specified in position 35).
- You cannot specify both EDTWRD and EDTCDE for the same field.
- The DFT and DFTVAL keywords cannot be specified with the EDTWRD keyword.
- Enclose the edit word in apostrophes.
- The sum of the blanks and stop-zero-suppression characters (digit positions) in the edit word must equal the length of the field.
- When you use the floating-currency symbol, the currency symbol is not counted as a digit position. For example, if you specify the floating-currency symbol for a field with a length of 7 and 2 decimal positions, the edit word is:

  ```
  EDTWRD('____$0._)
  ```

  where _ represents a blank.
- If you want to show a negative sign with a negative number, include a sign in the edit word. Use either the minus sign (–) or the letters CR (credit) to the right of the last digit replacement character. These print only if the number is negative.

Option indicators are not valid for this keyword.

Figure 3-83 on page 3-108 shows sample edit words with the program value of the field and the display value of the field (as edited).

Edit Word	Program Value	Displayed As
' , , 0. &CR*'	0000000005-	.05 CR*
' , , $0. CR**'	0000000005+	$0.05 *
' , , $0. CR**'	0034567890-	$345,678.90CR**
'$, , 0. '	0000000000	$ .00
'$& , , 0. &-&GROSS'	1234567890-	$ 12,345,678.90 - GROSS
' , , * . &-'	0000135792	******1,357.92
	0000135792	0000135792
	0000135792-	0000135792
' '	0000000000	
' '	0000135678+	135678
' '	0000135678-	135678
' 0'	0000135678-	135678
'0 '	0000135678+	0001356 78
'$ &-&NET'	0000135678+	$ 135678 NET
'$ &-&NET'	0000135678-	$ 135678 - NET
'$0 -&NET'	0000135678	$ 0001356 78 NET
' $0 &CR*'	0000135678-	$135678 CR*
' $0 &CR*'	1234567809-	$12345678 09 CR*
' *&CR'	0000000000-	*********** CR
' * &CR'	0000000000-	*********00 CR
'* '	0000135678-	*0001356 78
' , , . &CR*&NET'	0000135678-	1,356.78 CR* NET
' , , . &CR*&NET'	0000135678	1,356.78 * NET
' , , $0. '	0000000005	$.05
' , , $0. CR'	0001356789-	$13,567.89CR
' , , *. *CR**'	0000135678+	******1,356.78* **
' , , DOLLARS CENTS'	0000135678	1,356DOLLARS78CENTS
'0 - - '	095140036	95-14-0036
'& ,* 0, '	0130579	**130,579
' - - &LATER'	093076	9-30-76 LATER
1 ' & & &LATER'	093076	9 30 76 LATER
' / / '	100176	10/01/76

1 **Note:** A run time error can occur if this edit word is specified for a both field. (Usage of B) See the description of ampersands earlier in this keyword description.

RV2F481-1

Figure 3-83. Sample Edit Words

Figure 3-84 shows how to specify the EDTWRD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00020A           FIELDA        7  2   5  2EDTWRD('    $0.  ')
     A
```

Figure 3-84. Specifying the EDTWRD Keyword

ENTFLDATR (Entry Field Attribute)

Use this field-, record-, or file-level keyword to define the leading attribute of the field that changes to a specified attribute whenever the cursor is located in the field. When defined at both the field- and record-level, the field-level specification is used

for the field. The ENTFLDATR keyword is ignored when the work station is not attached to a controller that supports an enhanced data stream.

The format of the keyword is:

```
ENTFLDATR[([color] [display attribute] [cursor visible])]
```

The parameters are optional for the keyword.

The color parameter specifies the color the field will change to when the cursor enters the field on a color work station. The parameter is specified as an expression of the form (*COLOR value).

The valid values for the color parameter are:

Value	Meaning
BLU	Blue
GRN	Green
PNK	Pink
RED	Red
TRQ	Turquoise
YLW	Yellow
WHT	White

If the color parameter is not specified, the default is white.

The display-attribute parameter specifies the display attributes of the field when the cursor enters the field. The parameter is specified as an expression of the form (*DSPATR value1 <value2 <value3...>>).

The valid values for the display attributes are:

Value	Meaning
BL	Blink
CS	Column separator
HI	High intensity
ND	Nondisplay
RI	Reverse image
UL	Underline

The default display attribute is HI.

Note: Display attributes CS, HI, and BL can cause fields on 5292, 3179, 3197 Models C1 and C2, 3487 Models HC, and 3488[6] work stations to appear as color fields. Separator lines will not appear when display attributes HI, RI, and UL are used. For more information on the COLOR keyword, see "COLOR (Color)" on page 3-72.

The cursor visible parameter allows the user to specify if the cursor is visible or invisible when it enters the field. *CURSOR means the cursor will stay visible and *NOCURSOR indicates that the cursor will become invisible when it enters the field. *CURSOR is the default. When *NOCURSOR is specified on the ENTFLDATR keyword, the specified field must have an I (inhibit keyword entry) in position 35. If

[6] Dependent on monitor attached to display device.

the field does not have data type I, than the default is used for the visible cursor parameter.

The field containing the ENTFLDATR keyword must be an input-capable field. The ENTFLDATR keyword is ignored for the field with DSPATR(PR).

Option indicators are valid for this keyword.

Figure 3-85 shows how to specify the ENTFLDATR keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
          A
          A     R RECORD
          A       F1            10A  B  3  4ENTFLDATR
          A       F2            10A  B 13  4ENTFLDATR((*COLOR RED))
          A       F3            10I  B 16  4ENTFLDATR(*NOCURSOR (*DSPATR HI RI))
```

Figure 3-85. Specifying the ENTFLDATR Keyword

In this example, the color turns white, the attribute is high intensity, and the cursor is visible for F1. For F2, the color is red, the attribute is high intensity, and the cursor is visible. For F3, the color is white, the attributes are high intensity and reverse image, and the cursor is not visible.

ERASE (Erase)

Use this record-level keyword with the OVERLAY keyword to specify that the records whose names you supplied as parameter values are to be erased from the display when this record is written.

The format of the keyword is:

```
ERASE(record-name-1 [record-name-2 ...[record-name-20]])
```

The record formats specified as parameter values must exist in the file.

ERASE may be specified more than once. The OVERLAY keyword must be specified whenever the ERASE keyword is specified.

If the ERASE and CLRL keywords are both in effect on an output operation, the records specified in the ERASE keyword are erased despite the CLRL keyword.

A record already on the display that has no input-capable fields and that has the CLRL specified cannot be erased by the ERASE keyword on another record (ERASE has no effect).

If the specified record is not on the display, this function is ignored for that record name.

Option indicators are valid for this keyword.

Note: This function requires extra data transmission and should be used only when you do not want to erase all other record formats. You can also specify the OVERLAY keyword with option indicators so that you can select when to erase all other formats.

Figure 3-86 on page 3-111 shows how to specify the ERASE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00020A          R REC1
     A             :
     A             :
     A          R REC2                     OVERLAY
     A             :
     A             :
     A          R REC4                     OVERLAY
     A                                     ERASE(REC1)
     A
```

Figure 3-86. Specifying the ERASE Keyword

ERASEINP (Erase Input)

Use this record-level keyword with the OVERLAY keyword to erase unprotected input-capable fields already on the display. (Unprotected input-capable fields are fields for which the DSPATR(PR) keyword is not in effect.) The fields are erased before the record format you are defining is displayed. Input-capable fields in the record format you are defining are not erased.

See the *Guide to Programming Displays* for information on how to use ERASEINP in files that are used in the System/36 environment.

The format of the keyword is:

```
ERASEINP[(*MDTON | *ALL)]
```

To erase all input-capable fields already on the display, specify the *ALL parameter. To erase only input-capable fields that have their changed data tags (MDTs) set on, specify the *MDTON parameter. Specifying ERASEINP(*MDTON) or ERASEINP has the same effect as pressing the Erase Input key.

The OVERLAY keyword must be specified whenever ERASEINP is specified.

When the MDTOFF keyword is specified on the same record format as ERASEINP, two conditions can occur:

- ERASEINP(*ALL) implies MDTOFF(*UNPR), unless the MDTOFF(*ALL) keyword is specified.
- If the ERASEINP or ERASEINP(*MDTON) is specified with MDTOFF(*ALL), the end effect is as if ERASEINP(*ALL) and MDTOFF(*ALL) are both specified.

If the ERASEINP and PROTECT keywords are both in effect for an output operation, the OS/400 program first erases the input-capable fields specified on the ERASEINP parameter value, then protects all input-capable fields on the display from input keying.

ERASEINP reduces line traffic because record formats already displayed can be reused and do not need to be sent to the display again.

A warning message appears at file creation time if the ERASEINP keyword is specified in a record with the DSPMOD keyword. At run time, the ERASEINP keyword is ignored when the display mode changes.

Option indicators are valid for this keyword.

Figure 3-87 on page 3-112 shows how to specify the ERASEINP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*
00030A*
00040A          R RECORD1                   OVERLAY
00050A              :                       ERASEINP
00060A              :
00070A              :
00080A*           (All changed input-capable fields on the screen are erased)
00090A              :
00100A              :
00110A              :
00120A          R RECORD2                   OVERLAY
00130A              :                       ERASEINP(*ALL)
00140A              :
00150A              :
00160A*           (All unprotected input-capable fields on the screen are erased
00170A*             whether changed or not)
00180A
00190A
00200A          R RECORD3                   OVERLAY
00210A  32                                  ERASEINP(*MDTON)
00220A N32                                  ERASEINP(*ALL)
00230A            FIELD1         5   I      DSPATR(PR)
00240A              :
00250A              :
00260A              :
00270A*           (FIELD1 is never erased)
     A
```

Figure 3-87. Specifying the ERASEINP Keyword

ERRMSG (Error Message) and ERRMSGID (Error Message Identifier)

Use one of these field-level keywords to identify a message to be displayed on the message line and associated with this field.

A warning message appears at file creation time if either of these keywords is specified on a record with the DSPMOD keyword. At run time, these keywords are ignored when the display mode changes.

Option indicators are valid for these keywords.

ERRMSG Keyword

The format of the keyword is:

```
ERRMSG('message-text' [response-indicator])
```

For ERRMSG, the parameters specify the message text and, optionally, a response indicator. The message text is the message to be displayed. (The Help key is not supported. Message help is not displayed when the Help key is pressed.)

If you specify a response indicator, it should be the same as the option indicator used to condition ERRMSG. On the input operation that follows the display of the error message, the OS/400 program turns off the indicator. If the response and the option indicators are the same, they are both turned off. One exception to this rule is if the response indicator is also specified for another keyword such as CHANGE,

CAnn, or CFnn. In that case, the on/off setting of the response indicator is based on the results of the function provided by the CHANGE or CFnn keyword. When a response indicator is specified, the first 50 characters of the message are also used as indicator text. Separate response indicator text is not valid for the ERRMSG keyword.

ERRMSGID Keyword

The format of the keyword is:

```
ERRMSGID(msgid [library-name/]msg-file [response-indicator] [&msg-data])
```

For ERRMSGID, the parameters specify:

- The message identifier for the message to be displayed
- The message file and, optionally, the library
- Optionally, a response indicator
- Optionally, a msg-data field name

The response indicator, if specified, should be the same as the option indicator used to condition the ERRMSGID keyword. On the subsequent input operation, after the display of the error message, the OS/400 program turns off the indicator. However, if the response indicator is also specified on another keyword such as CHANGE, CAnn, or CFnn, the on/off setting of the response indicator is based on the results of the function provided by the CHANGE, CAnn, or CFnn keyword.

Note: Indicator text cannot be specified on the ERRMSGID keyword.

The msg-data field, if specified, contains the replacement text for the specified message. The field must exist in the record format and the field must be defined as a character field (data type A) with usage P. For more information on how replacement text works, refer to the Send Program Message CL command in the *CL Reference* manual.

Figure 3-88 shows how to specify the ERRMSG and ERRMSGID keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CUSMST
00020A              :
00030A              :
00040A              :
00050A            QTYORD        10A  I  5  3
00060A  61                                 ERRMSG('No stock available' 61)
00070A  62                                 ERRMSG('Partial stock available' +
00080A                                     62)
00090A  63                                 ERRMSGID(MSG2000 CONSOLEMSG 63 +
00100A                                     &RPLTXT)
00110A            RPLTXT        78A  P
     A
```

Figure 3-88. Specifying the ERRMSG and ERRMSGID Keywords

Priority among Selected Keywords: You can specify ERRMSG and ERRMSGID more than once for a single field. During program processing, use option indicators to select a particular message to be displayed.

Only one message can be displayed at one time even if messages are in effect for several fields on the same output operation. The field whose message is displayed is the first field for which the program selected a message.

If several keywords are in effect for one field on an output operation, the message to be displayed is the first of the following:

- ERRMSG (If more than one ERRMSG keyword is selected, the first one the program selects is displayed.)
- ERRMSGID (If more than one ERRMSGID keyword is selected, the first one the program selects is displayed.)

A message field is displayed only if no error message keywords are also to be displayed.

For a list of priorities including the SFLMSG and SFLMSGID keywords, see "Priority among Selected Keywords" on page 3-210.

Conditions Occurring during Message Display: The displaying of a message using ERRMSG and ERRMSGID is similar to the displaying of messages by the OS/400 program when field validation errors are detected.

When a message is displayed because of either the ERRMSG or the ERRMSGID keyword, all fields on the display are kept, including the field the message is associated with. Except for option indicators, data in the output buffer is ignored (that is, any new data from the program is not sent to the display).

The function keys valid following display of a message are:

- Function keys specified at the file level
- Function keys specified for the record format for which a message is displayed, if selected when the message is displayed

When the message is displayed, the following conditions occur:

- For all errors:
  - The message is highlighted.
  - The cursor is blinked and the keyboard locked until the work station user presses the Reset key.
- For errors associated with input-capable fields:
  - All fields in error are displayed with their images reversed.

    If a field in error has both the underline (UL) display attribute and the highlight (HI) attribute or the underline (UL) attribute and COLOR(BLU, WHT, or YLW) specified, its image is not reversed.
  - The cursor is repositioned to the first displayed field that is in error.
- For errors associated with output-only fields:
  - The display attribute of the field is not changed.
  - The cursor is not positioned to the field (it does not change position).

Note: Some display attributes can cause fields on the IBM Color Display Station to appear as color fields. See "COLOR (Color)" on page 3-72 earlier in this chapter.

Restoration of Reversed Image Fields: Fields are displayed with their images reversed because of system-detected typing errors or because of the ERRMSG or the ERRMSGID keyword. Generally, the OS/400 program restores the image on

the next I/O operation to the display, usually the next request from your program. The restoration is done before the requested function is performed. The following are exceptions where requests from your program do not cause the OS/400 program to restore reversed image fields:

- An input request with cancel (canceling a read operation with NOWAIT)
- A close request when the KEEP keyword is in effect
- Any request to a subfile record (no data is sent to the device)
- An output operation to a subfile control record format that does not display the subfile control record or subfile records (for example, clearing, deleting, or initializing the subfile)

Restrictions and Notes

The following apply to ERRMSG:

- When an ERRMSG or ERRMSGID keyword is in effect, no processing other than the processing for these keywords is performed for the record. If neither keyword is in effect, the record is processed in the normal manner.
- ERRMSG and ERRMSGID are valid for output-only, input-only, or output/input fields. These two keywords cannot be specified for a constant, hidden, program-to-system, or message field.
- For input-capable fields, ERRMSG and ERRMSGID are in effect only if the record containing the field for which they are specified is already on the display.
- ERRMSG and ERRMSGID cannot be specified in a subfile record format (SFL keyword specified). To display error messages for a subfile, see "SFLMSG (Subfile Message) and SFLMSGID (Subfile Message Identifier)" on page 3-209.
- ERRMSG and ERRMSGID are ignored if the variable start line number (SLNO(*VAR) specified) has changed since the last output operation.
- If you specify ERRMSG or ERRMSGID, you should also specify RSTDSP(*YES) on the Create Display File (CRTDSPF) or Change Display File (CHGDSPF) command. Otherwise, data on the display can be lost if the file is suspended.
- On an output operation that causes the display modes to be changed, ERRMSG and ERRMSGID are ignored.

ERRSFL (Error Subfile)

Use this file-level keyword to specify that messages should be displayed using a system-supplied error subfile. Messages that display in the error subfile are system validity check messages and messages associated with the following keywords:

ERRMSG
ERRMSGID
SFLMSG
SFLMSGID
CHECK(M10)
CHECK(M11)
CHECK(VN)
CHECK(VNE)
COMP
RANGE
VALUES

Validity check messages associated with the following input errors are also displayed in the error subfile:

- Errors in floating point operations

- Decimal position entry errors

This allows you to page through all the error messages issued when a record is written to the display, and all the validity check error messages issued when a record is read from the display. The system displays the error subfile on the message line. If the message line overlaps a record already displayed on the screen, the ERRSFL keyword is ignored.

This keyword has no parameters.

Option indicators are not valid for this keyword.

Figure 3-89 shows how to specify the ERRSFL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        MSGLOC(24)
00020A                                        ERRSFL
00030A          R RCD1
00040A            FIELD1        5A   B  2  3
00050A  10                                    ERRMSGID(MSG0001 MSGF1 10 &MDTA)
00060A            FIELD2        5A   B  5  7
00070A                                        ERRMSG('ERROR MSG 1' 11)
00080A            FIELD3        4S   I  7  7RANGE(1000 9999)
00090A                                        CHKMSGID(MSG0002 MSGF1 &MDTA1)
00100A            FIELD4       10A   B  8  7CHECK(VN)
00110A            MDTA         78A   P
00120A            MDTA1         4A   P
     A
```

Figure 3-89. Specifying the ERRSFL Keyword

In this example, when RCD1 is read from the display, any previous messages in the error subfile are cleared. Then, if FIELD3 does not contain a value in the range 1000 to 9999 and FIELD4 does not contain a valid name, the system places the message MSG0002 and the system message associated with CHECK(VN) in the error subfile and displays the error subfile on line 24 of the display. The user can view the messages by pressing the Page Up and Page Down keys.

When RCD1 is read again from the display, the previous messages in the error subfile are cleared. Then, if FIELD3 and FIELD4 are valid, control returns to the application. If FIELD1 and FIELD2 are not valid, when the application writes RCD1 to the display with indicators 10 and 11 on, the system places the message MSG0001 and the text `ERROR MSG 1` in the error subfile and displays the error subfile on line 24 of the display. The user can view the messages by pressing the Page Up and Page Down keys.

FLDCSRPRG (Cursor Progression Field)

Use this field-level keyword to define the field to which the cursor moves when exiting this field. The FLDCSRPRG keyword is ignored when the work station is not attached to a controller that supports an enhanced data stream.

The format of the keyword is:

```
FLDCSRPRG(name of a field)
```

One parameter *must be* specified.

The field containing the FLDCSRPRG keyword is defined as an input-capable field. It cannot be defined in a subfile.

The field name is a name of an input-capable field that is defined inside the same record that this field is in.

Option indicators are not valid for this keyword.

The FLDCSRPRG keyword is not allowed with the SNGCHCFLD or MLTCHCFLD keywords.

Figure 3-90 shows how to specify the FLDCSRPRG keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
                R RECORD
                  F1           10A  B   3  4FLDCSRPRG(F3)
                  F2           10A  B  13  4FLDCSRPRG(F1)
                  F3           10A  B  16  4FLDCSRPRG(F2)
```

Figure 3-90. Specifying the FLDCSRPRG Keyword

In this example, the cursor moves from field 1 to field 3, then from field 3 to field 2, and finally from field 2 to field 1.

FLTFIXDEC (Floating-Point to Fixed Decimal)

Use this field-level keyword to display a number in an output-capable (usage B or O) floating point field in fixed decimal notation.

This keyword has no parameters.

The floating-point number is first converted to the equivalent number with an exponent of zero. If the resulting number (digits and exponent) will fit in the field defined by the length and decimal positions values, the number is displayed with the exponent suppressed and aligned at the decimal point. If the number will not fit in the field defined by the length and decimal position values, the number is displayed in standard floating point form, n.nnnnnnE+nnn. When FLTFIXDEC is specified, the display length of the field is the DDS length plus 2 (the sign and the decimal point). The minimum length of the field is 6.

When the number is too large or small for the fixed-point form specified by FLTFIXDEC with the total digits and fractional digits specified for the field, a floating point form is displayed that presents the significand as follows. (The significand is the string of digits including the sign and decimal point to the left of the exponent character E.)

- Total significand decimal digits: DDS total digits minus 5
- Fractional significand digits: DDS total digits minus 6

FLTFIXDEC has no effect on the input format of the data. Numbers can be typed into the field in either fixed point or floating point format. When displayed again, however, FLTFIXDEC will be used to determine the display format.

Option indicators are not valid for this keyword.

Figure 3-91 on page 3-118 shows how to specify the FLTFIXDEC keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECFMT1
     A            FIELD1        10F 3B  1  2FLTFIXDEC
     A                                     FLTPCN(*DOUBLE)
     A
```

Figure 3-91. Specifying the FLTFIXDEC Keyword

The output numbers for the example would be converted as follows:

Output Number	Displayed As
-4.99994321000000E-004	'-4.9999E-004'
-5.00010000000000E-004	' -0.001'
-2.69123400000000E-002	' -0.027'
-0.00000000000000E+000	' 0.000'
0.00000000000000E+000	' 0.000'
2.71828182845900E++003	' 2718.282'
3.14159000000000E-052	'3.1416E-052'
9.87654321012345E+006	'9876543.210'
9.99999999960000E+006	'1.0000E+007'

FLTPCN (Floating-Point Precision)

Use this field-level keyword to specify the precision of a floating-point field.

The format of the keyword is:

```
FLTPCN(*SINGLE | *DOUBLE)
```

The valid parameters are *SINGLE (single precision) and *DOUBLE (double precision). This keyword is valid for floating-point fields only (data type F).

A single precision field can be up to 9 digits; a double precision field can be up to 17 digits. If you specify a field length greater than 9 (single precision) or 17 (double precision), an error message appears and the file is not created.

Option indicators are not valid for this keyword.

Figure 3-92 shows how to specify the FLTPCN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00090A            FIELDA        17F 4   1  5FLTPCN(*DOUBLE)
     A
```

Figure 3-92. Specifying the FLTPCN Keyword

FIELDA is a floating-point field with double precision.

FRCDTA (Force Data)

Use this record-level keyword to display a record format immediately, without waiting for the next input or input/output operation. When the buffer is partially full, the FRCDTA keyword can be used to clear the buffer.

Note: If this keyword is used after each write statement, performance problems will occur.

This keyword has no parameters.

When this keyword is in effect for a record format, the record format is displayed as if you had specified DFRWRT(*NO) on the Create Display File (CRTDSPF) command or the Change Display File (CHGDSPF) command.

You can use this keyword when DFRWRT(*YES) is in effect for the display file and your program does several output operations before doing an input operation. With DFRWRT(*YES) specified, none of the record formats is displayed until the input operation. There might be a long delay for the work station user while the program completes its processing. You can specify FRCDTA for a record format that is displayed first. This record format tells the work station user that the delay is normal. For a step-by-step description, see the example below.

FRCDTA can be specified once for each record format.

Option indicators are valid for this keyword.

Figure 3-93 shows how to specify the FRCDTA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* Following record format displays in progress message
00020A          R INPROG                      LOCK
00030A                                        FRCDTA
00040A                                   12 21'Please wait; +
00050A                                        operations in progress'
     A
00060A* Following record format uses upper part of screen
00070A          R RCD1                        OVERLAY
00080A                                     1 34'Sample Title'
00090A            FLD1           8  0   3  2
00100A            FLD2          20      4  2
     A
00110A* Following record format uses middle part of screen
00120A          R RCD2                        OVERLAY
00120A            FLDA           8     11  2
00140A            FLDB          18     12  2
     A
00150A* Following record format uses lower part of screen
00160A          R RCD3                        OVERLAY
00170A            FLDC           8   B 15  2
00180A            FLDD           8   B 16  2
     A
```

Figure 3-93. Specifying the FRCDTA Keyword

Three record formats (RCD1, RCD2, and RCD3) are used to create a single display; each of these record formats uses only a part of the display. Record format INPROG prepares the work station user for the delay while the other three record formats are prepared. The program does the following:

- Displays record format INPROG. With FRCDTA specified, this displays the in-progress notice immediately. The keyboard remains locked because the LOCK keyword is specified.

- Continues processing to prepare the other three record formats (RCD1, RCD2, and RCD3), then displays them. Since they overlap record format INPROG, it is erased.
- Reads record format RCD3. This unlocks the keyboard and the work station user can respond to the complete display.

GETRETAIN (Get Retain)

Use this record-level keyword with the UNLOCK keyword to specify that the OS/400 program is not to erase input-capable fields on input operations as described under the UNLOCK keyword.

This keyword has no parameters.

Using GETRETAIN reduces input keying when successive records contain mostly identical input. With GETRETAIN specified, the work station user needs only to change certain input fields rather than rekeying the entire record.

You must specify the UNLOCK keyword without any parameters when using GETRETAIN. The UNLOCK(*MDTOFF) specification provides the same function as GETRETAIN. See "UNLOCK (Unlock)" on page 3-229 for additional details.

Option indicators are not valid for this keyword.

Figure 3-94 shows how to specify the GETRETAIN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00101A          R REC1                      GETRETAIN
     A                                      UNLOCK
     A
```

Figure 3-94. Specifying the GETRETAIN Keyword

HELP (Help)

Use this file- or record-level keyword to enable the Help key.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the HELP keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
HELP[(response-indicator ['text'])]
```

If you specify a response indicator, the response indicator is set on and returned to your program. No input data is transmitted from the device. Processing is similar to that of a command attention key.

The optional text is included on the listing created at program compilation time to explain the intended use of the indicator. This text has no function in the file or program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program list.

One of the following actions occurs when the Help key is pressed:

- Second level text for a message is displayed if the cursor is located in a message subfile or on a field which specified either the ERRMSGID or SFLMSGID keyword.
- Online help information associated with either the entire display or an area on the display. This function is indicated by H specifications (refer to "HLPARA (Help Area)") or a file level HLPRCD, HLPPNLGRP, or HLPDOC keyword.
- Control is returned to the user's program. This occurs when there are no H specifications or file-level HLPDOC, HLPPNLGRP, or HLPRCD keywords in the file.

If this keyword is not specified and the Help key is pressed, the OS/400 program issues an error message indicating that the Help key is not valid at that time.

The HLPRTN keyword allows you to use option indicators to select when online help information is displayed, and when control is returned to the program. Refer to "HLPRTN (Help Return)" on page 3-134 for more information.

When a response indicator is specified on the HELP keyword, no H specifications or HLPRCD, HLPPNLGRP, HLPDOC, or HLPRTN keywords can be specified in the file.

HELP (with no response indicator) is required if the file contains H specifications or HLPRCD, HLPPNLGRP, HLPDOC, or HLPRTN keywords.

Option indicators are valid for this keyword.

Figure 3-95 shows how to specify the HELP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00024A                                        HELP
     A          R RECORD1
     A
```

Figure 3-95. Specifying the HELP Keyword

HLPARA (Help Area)

Use this help-specification-level keyword to define a rectangular area on the display. If the cursor is located in this area when you press the Help key, the online help information specified for that help (H) specification (on a HLPDOC, HLPPNLGRP, or HLPRCD keyword) is displayed.

The format of the keyword is any one of the following:

```
HLPARA(top-line left-position bottom-line right-position)
                    or
HLPARA(*RCD)
                    or
HLPARA(*NONE)
                    or
HLPARA(*FLD field-name [choice-number])
                    or
HLPARA(*CNST help-identifier)
```

Observe the following rules when the line and position values are specified as parameters:

- The line and position values must be within the display size.
- If you do not specify HLPARA for a secondary display size, the HLPARA of the primary display size is used if it is valid for the secondary display size. HLPARA(*NONE) is used if the HLPARA of the primary display size is not valid for the secondary display size.
- The top line must not exceed the bottom line and the left position must not exceed the right position.
- If you specify the SLNO(n) keyword on the record, the top-line and bottom-line values are adjusted, and any errors are diagnosed at creation time. If you specify the SLNO(*VAR) keyword on the record, the top line and bottom line are adjusted at processing time.

The special value *RCD indicates that the help area is the area of the record containing the H specification. This area includes all display positions in every line occupied by the record.

HLPARA(*RCD) is not valid for subfile control (SFLCTL) or user-defined (USRDFN) record formats.

If you specify HLPARA(*RCD) on an H specification, the record format containing the H specification must contain at least one displayable field for the primary display size. Hidden (H in position 38), message (M in position 38), and program-to-system (P in position 38) fields and fields that specify a SFLPGMQ or SFLMSGKEY keyword are not displayable.

The special value *NONE indicates that no help area is associated with the help information defined on this H specification. If the help information is defined using the UIM (HLPPNLGRP keyword), it is not displayed as item-specific help when the Help key is pressed, but might be displayed as extended help. If the help information is defined using DDS (HLPRCD keyword), it is not displayed as primary help when the Help key is pressed, but it might be displayed as secondary help when the Page Up or Page Down key is pressed on another help display. The *NONE value is not useful when the help information is defined in a document (HLPDOC keyword), since this information would never be displayed when the Help key is pressed.

The special value *FLD indicates that the help area is the area of a field. If the field occupies only one line, the help area consists of the first and last characters of the line and all the characters in between.

If the field wraps from one line to another, the help area consists of the entire length of all lines in the field. For example, if a field starts on line 3, position 4 and ends on line 5, position 10, the help area starts in line 3, position 1 and ends in line 5, position 80.

If the field is a choice or a continued-entry field, the help area consists of the rectangular area occupied by the choice or continued-entry field.

The field-name parameter specifies the name of the field for which the help area is defined. The field must exist in the record containing the H specification.

If the choice-number parameter is specified, the help area is the area of the choice within the field specified. When a choice number is specified, the field name must be the name of a menu-bar field or a selection field, and the choice number you specify must also be specified on a MNUBARCHC or CHOICE keyword for that field. Valid values for the choice number are positive integers greater than 0 and less than 100.

The *CNST special value indicates that the help area is the area of a constant field. This area includes the beginning and ending attribute bytes of the field.

The help-id parameter is a number that identifies the constant field for which this help area is defined. The constant field must exist in the record containing the H specification, and it must have the HLPID keyword specified with the same help-identifier.

You must specify at least one HLPARA keyword on an H specification. When you specify multiple HLPARA keywords for each H specification, you must use display size conditioning.

Help areas can overlap when multiple H specifications are specified on a record. When multiple H specifications are specified, the first H specification with both of the following characteristics is used:

- The help area specified on the HLPARA keyword contains the current cursor location.
- The option indicator on the HLPRCD, HLPPNLGRP, or HLPDOC keyword was in effect when the application record was written to the display.

The following rules apply to H specifications:

- An H in position 17 denotes the start of an H specification. The H specification must be located in the DDS after the record-level keywords and before the first field in that record.
- Each H specification must have exactly one HLPRCD, HLPPNLGRP, or HLPDOC keyword, up to one HLPBDY or HLPEXCLD keyword, and at least one HLPARA keyword.
- The end of the H specification is denoted by another H in position 17 or the first field.
- You cannot use H specifications in subfile (SFL keyword) record formats. H specifications are not allowed in subfile control formats associated with message subfiles (SFLMSGRCD keyword).

Option indicators are not valid for this keyword.

Figure 3-96 on page 3-124 and Figure 3-97 on page 3-124 show how to specify the HLPARA keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     HELP
     A                                     HLPRCD(DFTHELP)
     A          R RECORD1
     A          H                          HLPARA(1 5 3 15)
     A                                     HLPDOC(FLDHELP DOC1 FOLDER1)
     A
     A          H                          HLPARA(*RCD)
     A                                     HLPRCD(HELPRCD1)
     A
     A          H                          HLPARA(*NONE)
     A                                     HLPRCD(HELPRCD2)
     A            FIELD1        10A     2  5
     A            FIELD2        40A    10 10
     A
```

Figure 3-96. Specifying the HLPARA Keyword (Example 1)

In this example, the HLPARA keyword on the first H specification indicates that the area from line 1, position 5 to line 3, position 15 is to be associated with the online help information document DOC1. If the cursor is located in this area when the Help key is pressed, document DOC1 is displayed beginning at label FLDHELP.

The HLPARA keyword on the second H specification indicates that the area occupied by RECORD1 (lines 2 through 10) is to be associated with the online help information record HELPRCD1. If the cursor is located anywhere on lines 2 through 10 (outside the area defined by the first H specification) when the Help key is pressed, record HELPRCD1 is displayed.

The HLPARA keyword on the third H specification indicates that no area is to be associated with the online help information HELPRCD2. HELPRCD2 can only be displayed by pressing the Page Up or Page Down key from a online help information display.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     HELP HLPRCD(DFTHELP)
     A          R RECORD
     A          H                          HLPARA(*FLD F1 1)
     A                                     HLPRCD(UNDOHLP HLPLIB/HLPFILE)
     A          H                          HLPARA(*FLD F1 2)
     A                                     HLPRCD(MARKHLP HLPLIB/HLPFILE)
     A          H                          HLPARA(*FLD F1 3)
     A                                     HLPRCD(COPYHLP HLPLIB/HLPFILE)
     A          H                          HLPARA(*FLD F2)
     A                                     HLPRCD(F2HLP HLPLIB/HLPFILE)
     A          H                          HLPARA(*CNST 1)
     A                                     HLPRCD(TITLEHLP HLPLIB/HLPFILE)
     A                                  1 37'Title' HLPID(1)
     A            F1             2Y 0B 10  2SNGCHCFLD
     A                                     CHOICE(1 'Choice 1')
     A  01                                 CHOICE(2 'Choice 2')
     A                                     CHOICE(3 'Choice 3')
     A            F2            10A  B 10 30
     A
```

Figure 3-97. Specifying the HLPARA Keyword (Example 2)

The HLPARA keyword on the first three H specifications indicates the areas occupied by Choice 1, Choice 2, and Choice 3 respectively, to be associated with online help information. If Choice 2 is optioned off so that Choice 3 moves up one line, the help area for Choice 3 automatically moves with the choice.

The HLPARA keyword specified on the 4th H specification indicates the area of F2 to be associated with online help information. This area is line 10, from position 29 through position 40.

The HLPARA keyword specified on the 5th H specification indicates the area of the constant title to be associated with online help information. This area is line 1, from position 36 through position 42.

HLPBDY (Help Boundary)

Use this help-specification-level keyword to limit the online help information available when online help information is displayed.

This keyword has no parameters.

If the HLPBDY keyword is not specified, the online help information associated with all active H specifications (accumulated for all records on the display) is accessible to the user. Specifying the HLPBDY keyword partitions the list into sublists by defining help boundaries. Each sublist contains the H specifications specified between help boundaries. The H specification that has the HLPBDY keyword is considered to be before the boundary. If the help information is defined using DDS (HLPRCD keyword), the user has access only to the help information in the sublist containing the H specification selected when the Help key is pressed. If the help information is defined using the UIM (HLPPNLGRP keyword), the sublists determine the extended help. The extended help consists of the file level HLPPNLGRP followed by the sublist containing the H specification selected for item-specific help.

Option indicators are valid for this keyword.

Figure 3-98 on page 3-126 shows how to specify the HLPBDY keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      HELP
     A                                      HLPTCD(DFTHELP)
     A          R RECORD1
     A*
     A* This is H-spec 1
     A*
     A          H                           HLPARA(1 5 3 15)
     A                                      HLPRCD(HELPRCD1)
     A*
     A* This is H-spec 2
     A*
     A          H                           HLPARA(*NONE)
     A                                      HLPRCD(HELPRCD2)
     A                                      HLPBDY
     A*
     A* This is H-spec 3
     A*
     A          H                           HLPARA(4 5 6 15)
     A                                      HLPRCD(HELPRCD3)
     A  90                                  HLPBDY
     A*
     A* This is H-spec 4
     A*
     A          H                           HLPARA(8 5 10 15)
     A                                      HLPRCD(HELPRCD4)
     A                                      HLPBDY
     A            FIELD1        10A     1 10
     A
```

Figure 3-98. Specifying the HLPBDY Keyword

The list of H specifications for RECORD1 is divided into two or three sublists, depending on the condition of indicator 90 at the time the Help key is pressed. If indicator 90 is off, there are two sublists. The first sublist contains H specifications 1 and 2, and the second sublist contains H specifications 3 and 4. If indicator 90 is on, there are three sublists. The first sublist contains H specifications 1 and 2, the second sublist contains H specification 3, and the third sublist contains H specification 4.

HLPCLR (Help Cleared)

Use this record-level keyword to clear the list of active help specifications. When this record is displayed, only the online help information defined on the current record format or on the file level is accessible.

This keyword has no parameters.

If the HLPCLR keyword is not specified, the help specifications are accumulated for all records on the display and are active until the record containing the help specification either is cleared from the display or is completely overlapped by another record.

Option indicators are allowed on this keyword.

The record specifying the HLPCLR keyword must contain at least one help specification. Since help specifications are ignored for an override operation with USRDSPMGT specified, the use of HLPCLR with USRDSPMGT and PUTOVR will

result in no online help being available for display. To maintain the available online help, HELPCLR should be optioned off when PUTOVR is in effect.

Figure 3-99 shows how to specify the HLPCLR keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                   USRDFN
     A                                      HLPCLR
     A          H                           HLPARA(1 10 1 30)
     A                                      HLPRCD(RECORDA FILE1)
     A          H                           HLPARA(4 10 4 30)
     A                                      HLPDOC(A B C)
     A
```

Figure 3-99. Specifying the HLPCLR Keyword

When RECORD1 is displayed, only the online help information associated with the two help specifications defined in RECORD1 is available for display.

HLPCMDKEY (Help Command Key)

Use this record-level keyword to return control to your application program after a command attention (CA) or command function (CF) key is pressed on an application help record format. This keyword is specified on the application help record format. For control to return, the command key must be specified on both the application record format and the application help record format.

This keyword has no parameters.

A CA or CF key must be specified either at the file level or on the help record containing the HLPCMDKEY keyword. If no CAnn or CFnn keys are specified at the file level or on the help record, a warning message (severity 10) is issued. If all the CAnn and CFnn keys specified at the file level and on the same help record as the HLPCMDKEY keyword have option indicators, a warning message (severity 10) is issued. When a response indicator is specified on a CA or CF key on the application help record format, a warning message (severity 10) is issued and the response indicator will be ignored.

If you specify this keyword on the help record and the display station user presses one of the command keys that is specified on both the application record and the application help record, the following happens:

- If the command key is a CAnn key, no input data from the application record format is transmitted to the application program.
- If the command key is a CFnn key, input data from the application record format is transmitted to the application program.
- The command key will be returned to the application program.

The command key must be specified on both the application record format and the application help record format. If the command key is specified only on the application record format, the command key will not be allowed when the application help record format is displayed. If the command key is specified only on the application help record format, the command key will function as the Enter key and control will not return to the application program.

You cannot specify HLPCMDKEY on subfile (SFL keyword), subfile control (SFLCTL keyword), or user-defined (USRDFN keyword) record formats.

You cannot specify the HLPCMDKEY keyword in a file containing the USRDSPMGT keyword.

Option indicators are not valid for this keyword.

Figure 3-100 shows how to specify the HLPCMDKEY keyword. The first record is the application record format and the second record is the application help record format.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R APPRCD                    CA01
00020A                                      CA03
00030A                                      CF12(12)
00040A                                      CA04
00050A                                      HELP
00060A          H                           HLPRCD(HELPRCD)
00070A                                      HLPARA(1 1 24 80)
00080A                                    8  2'THIS IS THE APPLICATION'
00090A                                    9  2'RECORD FORMAT'
00100A            INPUT1        10    B 12 10
00110A            INPUT2        10    B 13 10
00120A            INPUT3        10    B 14 20
00130A*
00140A          R HELPRCD                   HLPCMDKEY
00150A                                      CA01(11)
00160A                                      CF03
00170A                                      CF05
00180A                                      CF12
00190A                                    5  8'SPECIFY COMPANY NAME'
00200A                                    6  9'SPECIFY STREET'
00210A                                    7 10'SPECIFY CITY, STATE, ZIP'
     A
```

Figure 3-100. Specifying the HLPCMDKEY Keyword

If the user is on the application help screen, the following occurs:

- If the CMD1 key is pressed, control is returned to the application program, but no data from the application record format is transmitted to the application program. Response indicator 11 is not set on since it was specified on the help record format rather than the application record format.
- If the CMD12 key is pressed, control is returned to the application program and data from the application record format is transmitted to the application program. Response indicator 12 is on.
- If the CMD5 key is pressed, it functions as the Enter key. The same CA or CF keys must be specified on both the application record format and the application help record format for control to be returned to the application program.
- If the CMD3 key is pressed, it functions as the Enter key. The corresponding CA or CF keys must be specified on both the application help record format and the application record format for control to be returned to the program.
- If the CMD4 key is pressed, it results in a message indicating that the CMD4 key is not allowed. The same CA or CF keys must be specified on both the application record format and the application help record format in order for control to be returned to the application program.

HLPDOC (Help Document)

Use this file- or help-specification-level keyword to define the document to use as help information text for a specific location on the display.

The format of the keyword is:

```
HLPDOC(online-help-information-text-label-name document-name folder-name)
```

The online-help-information-text-label-name parameter corresponds to a label in the online help document and marks the point at which the document's display begins.

The document-name parameter identifies the online document containing the online help information.

The folder-name parameter identifies the folder containing the document specified. Because folders can reside inside other folders, and because any given folder or document name is only unique within its containing folder, you may be required to concatenate several folder names together to identify a document/folder. The folder name you specify on the HLPDOC keyword can be a simple folder name, which follows the same syntax rules as the document name, or you can specify the folder name as a concatenated name.

Refer to "Syntax Rules" on page 1-4 for document, folder, and label naming conventions.

The document specified on an H specification level HLPDOC keyword is displayed if both of the following are true:

- The cursor is located in the help area (defined by HLPARA) for that H specification.
- The H specification is active. (The option indicator on the H specification level HLPDOC keyword determines if the H specification is active.)

The document specified on a file level HLPDOC keyword is displayed when no help area for the active records contains the current cursor location.

You cannot specify HLPDOC with HLPBDY, HLPPNLGRP, or HLPRTN.

Option indicators are valid for this keyword.

Figure 3-101 shows how to specify the HLPDOC keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      HELP
     A                                      HLPDOC(START GENERAL.HLP HELP.F1)
     A          R REC1                      OVERLAY
     A          H                           HLPDOC(LBL1 HELP#1 HELP.F1)
     A                                      HLPARA(10 3 12 50)
     A          H                           HLPARA(15 9 17 61)
     A  90                                  HLPDOC(LBL2 HELP#2 HELP/FLD)
     A          H                           HLPARA(15 9 17 61)
     A N90                                  HLPDOC(LBL3 HELP#3 HELP.F1/FLD)
     A
```

Figure 3-101. Specifying the HLPDOC Keyword

The HELP keyword enables the Help key.

The HLPDOC keyword at the file level specifies that the GENERAL.HLP document in the HELP.F1 folder will be displayed starting at the START help label if the Help key is pressed and the cursor is not in a help area defined by a HLPARA keyword at the H specification level.

At the H specification level:

- The first H specification indicates that the HELP#1 document in the HELP.F1 folder will be displayed starting at the LBL1 help label if the Help key is pressed and the cursor is in positions 3 through 50 of lines 10, 11, or 12.
- The second H specification indicates that the HELP#2 document in the HELP/FLD folder will be displayed starting at the LBL2 help label if the REC1 record is put with indicator 90 on, the Help key is pressed, and the cursor is in positions 9 through 61 of lines 15, 16, or 17.
- The third H specification indicates that the HELP#3 document in the HELP.F1/FLD folder will be displayed starting at the LBL3 help label if the REC1 record is put with indicator 90 off, the Help key is pressed, and the cursor is in positions 9 through 61 of lines 15, 16, or 17.

HLPEXCLD (Help Excluded)

Use this help-specification-level keyword to indicate that the online help information associated with this help specification should not be displayed as extended help, but is available as item-specific help.

This keyword has no parameters.

If you do not specify this keyword, extended help consists of the online help information associated with both the file-level HLPPNLGRP keyword (if any) and the HLPPNLGRP keyword on all active help specifications.

This keyword is allowed only on help-specifications that specify a HLPPNLGRP keyword.

At least one instance of each parameter on the HLPPNLGRP keyword should not have the HLPEXCLD keyword specified. If all help specifications specifying a particular help panel group name are excluded, an error message is issued at run-time if the Help key is pressed when the cursor is located in a help area associated with that help panel group name.

Option indicators are valid for this keyword.

Figure 3-102 on page 3-131 shows how to specify the HLPEXCLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1
     A
     A          H                           HLPARA(1   12   8   14)
     A                                      HLPPNLGRP(R1  PNLA)
     A          H                           HLPARA(1   18   8   19)
     A                                      HLPPNLGRP(R2  PNLA)
     A          H                           HLPARA(1   35   8   37)
     A                                      HLPPNLGRP(R1  PNLA)
     A                                      HLPEXCLD
     A          H                           HLPARA(1   49   8   50)
     A                                      HLPPNLGRP(R2  PNLA)
     A                                      HLPEXCLD
     A
```

Figure 3-102. Specifying the HLPEXCLD Keyword

In Figure 3-102, the HLPEXCLD keywords prevent help modules R1 and R2 from being displayed twice as extended help.

HLPFULL (Help Full)

Use this file-level keyword to indicate that the help panel group help text for the application should be displayed using the full screen rather than using windows.

This keyword has no parameters.

If you do not specify this keyword, the online help information is displayed in a window unless the *HLPFULL option is specified for the user profile.

When you specify the HLPFULL keyword, you must specify the HLPPNLGRP keyword either at the file level or at the help specification level.

Option indicators are not valid for this keyword.

Figure 3-103 shows how to specify the HLPFULL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      HELP
     A                                      HLPPNLGRP(GENERAL LIBA/PNL1)
     A                                      HLPFULL
     A                                      HLPTITLE('Sample Screen')
     A          R RECO01H
     A          H                           HLPARA(4  10  4  29)
     A                                      HLPPNLGRP(NAMETAG LIBA/PNL1)
     A                                  1 10'Sample Screen'
     A            NAME1         20A  B  2 10
     A
```

Figure 3-103. Specifying the HLPFULL Keyword

When the Help key is pressed in cursor location line 4 positions 10 through 29, the help module NAMETAG from LIBA/PNL1 is displayed using the full screen. If the Help key is pressed in any other location, the help module GENERAL from LIBA/PNL1 is displayed using the full screen.

HLPID (Help Identifier)

Use this constant field-level keyword to specify an identifier for the constant in the field-level help. The identifier you specify can be used on the HLPARA keyword to link help text to this constant field.

The format of the keyword is:

```
HLPID(help-identifier)
```

The help-identifier parameter is required and can be only a numeric value from 1 to 999. The value you specify must be unique within the record you are defining.

Option indicators are not valid for this keyword.

Figure 3-104 shows how to specify the HLPID keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD                    HELP
     A          H                           HLPARA(*CNST 1)
     A                                      HLPRCD(HLPCNST1 LIB1/FILE1)
     A          H                           HLPARA(*CNST 2)
     A                                      HLPRCD(HLPCNST2 LIB1/FILE1)
     A                                  2  4'Constant field 1' HLPID(1)
     A                                  4  4'Constant field 2' HLPID(2)
     A
```

Figure 3-104. Specifying the HLPID Keyword

In this example, if the cursor is located on the text `'Constant field 1'` on the display and the Help key is pressed, record HLPCNST1 in file FILE1, library LIB1, is displayed as help text. If the cursor is located on the text `'Constant field 2'` on the display and the Help key is pressed, record HLPCNST2 in file FILE1, library LIB1 is displayed as help text.

HLPPNLGRP (Help Panel Group)

Use this file- or help-specification-level keyword to specify the source of UIM-defined online help information to be displayed when the Help key is pressed.

The format of the keyword is:

```
HLPPNLGRP(help-module-name [library-name/]panel-group-name)
```

The help module name is 1 to 32 characters long. Valid values for the first character in the name are alphabetic characters A through Z. Valid values for subsequent characters are alphabetic characters A through Z, numeric characters 0 through 9, and slash (/). If the name contains a slash character, the name must be enclosed in apostrophes.

The panel group name specifies the UIM-panel group object that contains the help module. It does not have to exist when the display file is created. If you do not specify the library name, *LIBL is used to search for the panel group object.

Every help specification must contain a HLPRCD, HLPDOC, or HLPPNLGRP keyword, but a display file cannot contain both HLPPNLGRP and HLPRCD keywords, nor HLPPNLGRP and HLPDOC keywords.

If you specify the HLPPNLGRP keyword at the file level, you must specify the HELP keyword at the file level. If there are no help specifications in the file, you must also specify the HLPTITLE keyword at the file level.

If you specify the HLPPNLGRP keyword at the help specification level, you must specify a HELP and a HLPTITLE keyword either at the file level or on the current record.

Option indicators are valid for this keyword.

Figure 3-105 shows how to specify the HLPPNLGRP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     HELP
     A                                     HLPPNLGRP(GENERAL LIBA/PANEL1)
     A                                     HLPTITLE('Sample Screen')
     A                                     HLPSCHIDX(QHSS1)
     A          R REC001
     A          H                          HLPARA(4  10  4  29)
     A                                     HLPPNLGRP(NAMETAG LIBA/PANEL1)
     A          H                          HLPARA(5  10  5 19)
     A                                     HLPPNLGRP(OPTION1TAG PANEL2)
     A                                  1 10'Sample Screen'
     A            NAME1         20A  B  4 10
     A            OPTION1       10A  B  5 10
     A
```

Figure 3-105. Specifying the HLPPNLGRP Keyword

If the cursor is located on line 4 between positions 10 and 29 when the Help key is pressed, help module NAMETAG from UIM panel group PANEL1 in LIBA is displayed. If the cursor is located on line 5 between positions 10 and 19, help module OPTION1TAG from UIM panel group PANEL2 in the library list is displayed. If the cursor is located anywhere else when the Help key is pressed, help module GENERAL in panel group PANEL1 in LIBA is displayed.

HLPRCD (Help Record)

Use this file- or help-specification-level keyword to specify the record format containing the online help information to be displayed when the Help key is pressed.

The format of the keyword is:

```
HLPRCD(record-format-name [[library-name/]file-name])
```

The record format can exist either in the file being defined or in the file specified on HLPRCD. If you do not specify the file name, the record format must exist in the file being defined.

The file-name parameter identifies the file containing the record format. The current library list (*LIBL) at program run time is used if you do not specify the library name.

The record specified on an H specification level HLPRCD keyword is displayed if both of the following are true:

- The cursor is located in the help area (defined by HLPARA) for that H specification.

- The H specification is active. (The option indicator on the H specification level HLPRCD keyword determines whether the H specification is active.)

The record specified on a file-level HLPRCD keyword is displayed when no help area for the active records contains the current cursor location.

Option indicators are valid for this keyword.

Figure 3-106 shows how to specify the HLPRCD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      HELP
     A                                      HLPRCD(DFTHELP HELPFILE)
     A          R RECORD1
     A          H                           HLPARA(1 1 24 80)
     A  99                                  HLPRCD(ERRHELP)
     A          H                           HLPARA(1 1 1 80)
     A                                      HLPRCD(HELPRCD1 HELPFILE)
     A            FIELD1        10A     1 10
     A
```

Figure 3-106. Specifying the HLPRCD Keyword

When indicator 99 is set on, the online help information in the record ERRHELP (which must exist in this display file) displays when the Help key is pressed. If indicator 99 is set off and the cursor is located on the first line when the Help key is pressed, the record HELPRCD1 in file HELPFILE will display. Otherwise, the online help information from the record specified on the file-level HLPRCD keyword will display when the Help key is pressed.

HLPRTN (Help Return)

Use this file- or record-level keyword to return control to your program when you press the Help key. If HLPRTN is not specified, online help information associated with the current cursor location is displayed. (Refer to "HLPARA (Help Area)" on page 3-121 for more information on specifying help areas.)

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the HLPRTN keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
HLPRTN[(response-indicator ['text'])]
```

If you specify a response indicator, the response indicator is set on and returned to your program. No input data is transmitted from the device. Processing is similar to that of a command attention key.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text has no function in the file or program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

HLPRTN at either the file or record level takes priority over any HLPRCD, HLPPNLGRP, or HLPDOC keywords. Any HLPRTN keyword found in the file is processed before any other applicable help keyword.

When you specify HLPRTN, control may or may not return to your program, depending on whether you use an option indicator:

- If you specify HLPRTN without an option indicator, control returns to your program when you press the Help key. A warning message appears at creation time if you specify an unoptioned HLPRTN keyword on a file or record containing H specifications.
- If you specify HLPRTN with an option indicator, control returns to your program if the option indicator is on at the time the record is displayed. The H specifications are used if the option indicator is off.

Option indicators are valid for this keyword.

Figure 3-107 and Figure 3-108 show how to use the HLPRTN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      HELP
     A                                      HLPRCD(GENERAL)
     A          R RECORD1
     A  02                                  HLPRTN
     A          H                           HLPARA(1 1 3 10)
     A                                      HLPRCD(HELPREC1)
     A            FIELD1        10A  B  2  2
     A          R RECORD2                   HLPRTN
     A            FIELDA         5A  B 10  7
     A
```

Figure 3-107. Specifying the HLPRTN Keyword (Example 1)

If indicator 02 is on when RECORD1 is written to the display, control will return to the user program when the Help key is pressed. If indicator 02 is off, online help information record HELPREC1 or GENERAL will be displayed when the Help key is pressed, depending on the position of the cursor. When RECORD2 is displayed, control will return to the user program when the Help key is pressed.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      HELP
     A  01                                  HLPRTN
     A                                      HLPRCD(GENERAL)
     A          R RECORD1
     A          H                           HLPARA(1 1 3 10)
     A                                      HLPRCD(HELPREC1)
     A            FIELD1        10A  B  2  2
     A          R RECORD2
     A            FIELDA         5A  B 10  7
     A
```

Figure 3-108. Specifying the HLPRTN Keyword (Example 2)

If indicator 01 is on, control returns to your program when you press the Help key regardless of the cursor position.

HLPSCHIDX (Help Search Index)

Use this file-level keyword to enable the index search function (F11 on the Help display) and specify the search index object used for the index search.

The format of the keyword is:

```
HLPSCHIDX([library-name/]search-index-object)
```

The search index object, created using the CRTSCHIDX command, contains the data to be made available when the assigned function key (F5) is pressed to start the index search function.

If you do not specify a library name, *LIBL is used to search for the search index object. The search index object does not have to exist when the display file is created.

HLPSCHIDX is valid only when at least one HLPPNLGRP keyword is specified in the file.

Option indicators are not valid for this keyword.

Figure 3-109 shows how to specify the HLPSCHIDX keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     HELP
     A                                     HLPTITLE('Sample Screen')
     A                                     HLPPNLGRP(GENERAL LIBA/PANEL1)
     A                                     HLPSCHIDX(LIBA/SEARCH1)
     A          R REC001H
     A          H                          HLPARA(4  10  4  29)
     A                                     HLPPNLGRP(NAMETAG LIBA/PNL1)
     A                                   1 10'Sample Screen'
     A            NAME1         20A  B  4 10
     A
```

Figure 3-109. Specifying the HLPSCHIDX Keyword

HLPSEQ (Help Sequencing)

Use this record-level keyword to define the sequencing of text records for Page key processing.

The format of the keyword is:

```
HLPSEQ(group-name sequence-number)
```

The group name is a 1- to 10-character name used to associate the primary help format with the secondary help formats in the help file. When a Page Up or Page Down key is pressed on an online help information display, those record formats in the help file that have the same help group name as the record currently displayed as online help information are displayed.

The sequence number is a numeric value (0 to 99) used to order the record formats within the help group. This order determines the sequence in which the record formats are displayed as secondary online help information.

Duplicate numbers within a group are not allowed.

A help record format that does not have the HLPSEQ keyword coded is considered to be the only record in its group.

You cannot specify HLPSEQ on subfile (SFL keyword) or user-defined (USRDFN keyword) record formats.

Option indicators are not valid for this keyword.

Figure 3-110 on page 3-137 shows how to specify the HLPSEQ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                   HLPSEQ(HGROUP1 10)
     A                                    5  1'Help text ...'
     A
```

Figure 3-110. Specifying the HLPSEQ Keyword

RECORD1 is in the help group HGROUP1 and has a sequence number of 10.

HLPTITLE (Help Title)

Use this file- or record-level keyword to define the default title of panel group online help information that is displayed using the full screen. This should be the name of the display you were on when you pressed the Help key. Use this keyword only on a full-screen help display and when no help title is specified in the help source.

The format of the keyword is:

```
HLPTITLE('text')
```

The text can be up to 55 characters long.

If you specify the HLPTITLE keyword in a file, the file must contain at least one HLPPNLGRP keyword at either the file or help specification level.

If you specify a file-level HLPPNLGRP keyword and no help specifications are defined in the file, the HLPTITLE keyword is required at the file level.

If you do not specify a HLPTITLE keyword at the file level, at least one HLPTITLE keyword is required on every record that contains help specifications. The HLPTITLE keyword is not valid on records that do not contain help specifications.

Option indicators are not valid on a file-level HLPTITLE keyword. Option indicators are allowed on record-level HLPTITLE keywords and must be specified on each HLPTITLE keyword if the record contains multiple HLPTITLE keywords. You can specify a maximum of 15 HLPTITLE keywords on a record if all have option indicators. At run time, the first HLPTITLE keyword in effect is used. If no HLPTITLE keyword is in effect for the record, a message is issued.

Figure 3-111 shows how to specify the HLPTITLE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      HELP
     A                                      HLPPNLGRP(GENERAL LIBA/PANEL1)
     A          R REC001
     A  90                                  HLPTITLE('Sample Screen 1')
     A N90                                  HLPTITLE('Sample Screen 2')
     A          H                           HLPARA(4  10  4  29)
     A                                      HLPPNLGRP(NAMETAG LIBA/PANEL1)
     A          H                           HLPARA(6  10  6  19)
     A  10                                  HLPPNLGRP(OPTION2TAG PANEL2)
     A  90                                1 10'Sample Screen 1'
     A N90                                1 10'Sample Screen 2'
     A            NAME1         20A  B  4 10
     A  10        OPTION2       10A  B  6 10
     A
```

Figure 3-111. Specifying the HLPTITLE Keyword

Two titles are associated with the record, so the HLPTITLE keyword is specified at the record level, where option indicators are used. The state of option indicator 90 determines which title is displayed on both the application display and the online help information display. Using an indicator and its complement guarantees that one of the two HLPTITLE keywords is in effect.

HOME (Home)

Use this file- or record-level keyword to specify that you want to recognize and handle the Home key through your program.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the HOME keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
HOME[(response-indicator ['text'])]
```

If you press the Home key and the cursor is not already at the home position, the cursor returns to the home position, whether or not you specify the HOME keyword.

If the cursor is already at the home position when the Home key is pressed, the OS/400 program returns control to your program as it does when a command attention key is pressed (no data is received from the device). In this situation, if you have not specified the HOME keyword, the OS/400 program sends a message indicating that the key is not valid at that time.

The home position is one of the following (in order of priority):

- The cursor position specified by the last output operation
- The first unprotected input field
- Position 1, line 1

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text's only function in the file or the program is as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

Option indicators are valid for this keyword.

Figure 3-112 shows how to specify the HOME keyword so that if the cursor is in the home position when the Home key is pressed, control returns to the program with response indicator 95 set on.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A                                  HOME(95 'Home key')
     A
```

Figure 3-112. Specifying the HOME Keyword

INDARA (Indicator Area)

Use this file-level keyword to remove option and response indicators from the buffer (also called the record area) and place them in a 99-byte separate indicator area.

This keyword has no parameters.

Specifying INDARA provides the following advantages:

- Simplifies COBOL programming when both option and response indicators are used. If the same indicator is used as a response indicator and as an option indicator, both indicators always have the same value, regardless of the order in which they are specified in the DDS.
- Assists the RPG III programmer using program-described WORKSTN files.

If you specify INDARA, you can add, change, or delete option and response indicators in the DDS and re-create the file without re-creating the high-level language program. This is true because the field locations in the buffer have not changed and therefore the level check data has not changed. However, if the program is to take advantage of the new indicators, it would still need to be changed and re-created.

If you specify INDARA, some high-level languages may require that you specify in your program that a separate indicator area is to be used. See the appropriate high-level language manual.

If you specify INDARA, the LOGINP and LOGOUT keywords do not log response or option indicators when your program sends I/O operations. This is because response and option indicators do not appear in the input or output buffers.

Option indicators are not valid for this keyword.

Figure 3-113 shows how to specify the INDARA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     INDARA
00020A                                     CF01(01 'End program')
00030A          R PROMPT
00040A            ACTNBR        10   B  2  2
00050A  41                                 ERRMSG('Account number +
00060A                                     not found' 41)
     A
```

Figure 3-113. Specifying the INDARA Keyword

INDARA has been specified; option indicator 41 and response indicators 01 and 41 are removed from the buffer for record format PROMPT and placed in the separate indicator area. Only ACTNBR, a named output/input field, remains in the buffer for record format PROMPT.

INDTXT (Indicator Text)

Use this file-, record-, or field-level keyword to associate descriptive text (indicating intent or usage) with a specific response or option indicator. You can specify it once for each response and option indicator.

The format for the keyword is:

```
INDTXT(indicator 'indicator-text')
```

If you specify this keyword, indicator-text is a required parameter value. The text is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text's only function in the file or the program is a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

Option indicators are not valid for this keyword.

Note: INDTXT by itself does not cause the specified indicator to appear in either the input or the output record area. It merely provides text to be associated with the indicator. If the indicator has not been specified elsewhere (as either an option indicator or a response indicator), then the text is lost without a diagnostic. Also, once an indicator has been given a text assignment (either by this keyword or the response indicator text), no other text assignment is allowed.

Figure 3-114 shows how to specify the INDTXT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        INDTXT(02 'Alternate month')
00020A          R MASTER
00030A            MTH                   2 10
00040A  02        ALTMTH                2 10
     A
```

Figure 3-114. Specifying the INDTXT Keyword

INDTXT describes the use of option indicator 02. In a compiler computer printout for a high-level language, 'Alternate month' would be printed as a comment with the description of indicator 02.

INVITE (Invite)

Use this file- or record-level keyword to invite the device for a later read. To send an invite to a specific device, your program sends an output operation to the device with the INVITE keyword in effect. If the record format used has output-capable fields, the data is sent to the device before the device is invited.

This keyword has no parameters.

INVITE must be used if the display file can have multiple acquired devices and your program does read from invited devices operations. This is because a read from invited devices to a multiple device display file only returns a record from one of the devices that was invited. If you want all the acquired devices to be able to return data, an output operation with INVITE in effect must be sent to each device before the read from invited devices. Even if there is only one device acquired on the multiple device file, the device must be invited via INVITE before a read from invited devices.

INVITE also gives you the ability to create a subset of acquired devices that are eligible to respond on a read from invited devices. For example, if ten devices are currently acquired to the file but only three devices were invited, the next read from

invited devices operation returns a record from one of the three invited devices. This is true even if the other devices have data available.

INVITE provides some performance improvement. Normally a read request is issued to a device when your program sends an input operation. However, INVITE allows you to request the read when you issue the output operation. After the output operation is complete, your program can do other processing while the device is issuing data and the OS/400 program is processing the received data. This may be a significant improvement if the device is remote.

For specific instructions on when an invite operation is necessary and how to specify a read from invited devices operation in your program, see the appropriate high-level language manual.

INVITE cannot be specified at both the file and record level and cannot be specified with the subfile keyword (SFL).

Option indicators are valid for this keyword.

Special Considerations

The following are special considerations when using the INVITE keyword:

- An input operation sent to a specific device does not require an invite. Input operations with a specified record format name or device are directed to one device. If that device has an invite outstanding at the time of the input operation, the invite is deleted after the input operation is completed.
- Once an invite has been sent to a device, the only valid operations (in addition to a read from invited devices) are the following:
  - An input operation to a specific device.
  - An output operation with data that tries to cancel the invite. If the cancel is successful, the data is written. If INVITE was in effect on the output operation, the device is invited again. If the cancel is not successful (because the data has already been received by the system), the output operation fails. Your program must perform an input operation to process the data. The input operation erases the invite for that device.
- On a read from invited devices operation to a display file, only data from devices with an outstanding invite are considered. The input operation waits for data from any of the invited devices. (See the WAITRCD parameter on the Create Display File (CRTDSPF) and Change Display File (CHGDSPF) commands.) If none of the invited devices responds before the wait time ends, a notify message is sent and no data is returned. All invited devices remain invited.
- If more than one device acquired to the display file has an invite outstanding, a read from invited devices operation returns the next available record from one of the invited devices. If records are received from more than one device before this input operation, the other records are kept for subsequent input operations.
- When a read from invited devices operation to a display file returns a record to your program from an invited device, the invite for that device is deleted. Other devices that have an invite outstanding remain invited. The device that sent the record your program read must be invited again if you want to receive data from this device on a later read from invited devices operation.

- If no device was invited or if a device was invited but the job was canceled with the controlled option, a read from invited devices operation to a display file results in a notify message and no data is returned to your program. All invited devices remain invited.
- If you want to invite a device but have no data to send it, issue an output operation using a record format containing no output-capable fields with INVITE in effect.
- After the first record is received from an invited display device, the device should not be re-invited until all the record formats on the display with input-capable fields are read by your program. Your program can read those other record formats if you specify the record format name and the device name on the read operation.
- If your display file has the delayed write option (DFRWRT(*YES) parameter on the (Create Display File (CRTDSPF) and Change Display File (CHGDSPF) commands), an output operation with the INVITE keyword in effect causes the delayed output to appear on the display before the device is invited.

Figure 3-115 shows how to specify the INVITE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  01                                  INVITE
00020A          R RCD1
00030A            FLD1          10      2  2
00030A            FLD2           5      2 24
     A
```

Figure 3-115. Specifying the INVITE Keyword

INVITE is in effect only when option indicator 01 is set on.

INZINP (Initialize Input)

Use this record-level keyword to initialize output/input fields without necessarily sending the initialized data to the display when the PUTOVR and ERASEINP(*ALL) keywords are both in effect.

This keyword has no parameters.

INZINP is particularly useful for applications that perform data entry from remote work stations. It can reduce line traffic between the system unit and the remote work station.

The following steps describe how this keyword is used:

1. An output operation displays an output/input field for which the OVRDTA or OVRATR keyword is in effect. The system initializes the input save area to the program value of the field. For example, if the program sets NAME1 to the name Bob, the input save area contains the name Bob.
2. An input operation places data from the work station in an input buffer. If the work station user did not key into the input-capable field and the DSPATR(MDT) keyword is not in effect for the field, the field does not return data from the work station. The system retrieves data from the input save area and places it in the input buffer for use by the program. Thus, all input-capable fields have data in the input buffer, either from the work station or from the input save area.

3. On another output operation, the following could occur:
   - If the INZINP and OVRDTA keywords are not in effect, the input save area is unchanged, even if the program changed the field value. For example, if the program changed the field value to the name Tom, the input save area would still have either the value entered by the work station user or the name Bob, the earlier program value.
   - If INZINP is in effect, the input save area would have the current program value. The current program value is sent to the display for fields for which the OVRDTA keyword is in effect.

     If the OVRDTA keyword is not in effect on an output operation, the program must clear the output buffer for the fields with the OVRDTA keyword specified to ensure that the input save area matches the fields on the display (which are all blanks after the output operation).

Note: If the ERASEINP(*ALL) keyword is in effect for the output operation, input-capable fields are cleared at the display device (on the display), but the input save area is not cleared. For the contents of the input save area, see Figure 3-116 on page 3-143 and Figure 3-117 on page 3-144. Fields with the DFT keyword specified are initialized with the value specified for the DFT keyword even if the fields are not selected for display. The values are maintained unless the application program selects the fields for display, then changes their data values. This keyword requires the PUTOVR, OVERLAY, and ERASEINP(*ALL) keywords to be specified at the record level.

The following tables show the effect of the ERASEINP(*ALL) and PUTOVR keywords with and without the INZINP keyword.

Figure 3-116. INZINP Output/Input Fields

OVRATR or OVRDTA Keyword	INZINP Keyword	Contents of Input Save Area
Not specified	Does not apply	Previous contents
Specified but not in effect	Specified but not in effect	Previous contents
Specified but not in effect	Specified and in effect	Program value (not sent to display)
OVRATR specified and in effect	Does not apply	Program value (also sent to display)
OVRATR specified and in effect; OVRDTA not specified or not in effect	Does not apply	Previous contents (not sent to display)

Figure 3-117. INZINP Input Only Fields

OVRATR Keyword	INZINP Keyword	Contents of Input Save Area
Not specified	Does not apply	Previous contents
Specified and in effect	Does not apply	Previous contents
Specified but not in effect	Does not apply	Character fields: blanks Numeric fields: zeros

Do the following to set the input save area to blanks and zeros to match the fields cleared at the work station by the ERASEINP(*ALL) keyword:

1. Specify the same option indicators for INZINP as for ERASEINP(*ALL), PUTOVR, and OVERLAY keywords.
2. Specify OVRDTA or OVRATR for all output/input fields. (Set option indicators off for these keywords if you do not want to send data or attributes to the device. If you enable OVRATR, also enable OVRDTA.)
3. Specify the OVRATR keyword for all input-only fields. (Set option indicators off for this keyword if you do not want to send attributes to the device.)
4. Set all output/input fields to blanks (for character fields) or zeros (for numeric fields) before the output operation.

A warning message appears at file creation time if the INZINP keyword is specified on a record with the DSPMOD keyword. At run time, the INZINP keyword is ignored when the display mode changes.

Option indicators are valid for this keyword.

Figure 3-118 shows how to specify the INZINP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R FMT1
00020A  77                                  PUTOVR OVERLAY ERASEINP(*ALL)
00030A  77                                  INZINP
00040A                                    7  8'CUSTOMER NUMBER'
00050A            CUSNBR         6   I  7 26
00060A N77                                  OVRATR
00070A                                    9 12'CUSTOMER NAME'
00080A            NAME          25   B  9 30
00090A N77                                  OVRATR
00100A                                   10  2'CUSTOMER ADDRESS LINE 1'
00110A            ADDR1         25   B 10 30
00120A N77                                  OVRATR
00130A                                   11  2'CUSTOMER ADDRESS LINE 2'
00140A            ADDR2         25   B 11 30
00150A N77                                  OVRATR
00160A                                   12  9'NEW CREDIT LIMIT'
00170A            LIMIT          4  0I 12 30
00180A N77                                  OVRATR
     A
```

Figure 3-118. Specifying the INZINP Keyword

This example illustrates the following:

1. For the first output operation, the user program sets off option indicator 77. Therefore, the PUTOVR, ERASEINP, and INZINP keywords are not in effect. This causes the following to take place:
   - Fields NAME, ADDR1, and ADDR2 are sent to the work station.
   - The input save area contains:

CUSNBR	All blanks
NAME	The program value (which appears on the display)
ADDR1	The program value (which appears on the display)
ADDR2	The program value (which appears on the display)
LIMIT	All zeros

2. For the second output operation, the user program sets on option indicator 77. This causes the following to take place:
   - All input-capable fields are cleared at the work station.
   - No fields are sent to the work station.
   - The input save area contains:

CUSNBR	All blanks
NAME	The program value (not sent to the display)
ADDR1	The program value (not sent to the display)
ADDR2	The program value (not sent to the display)
LIMIT	All zeros

Note: If fields NAME, ADDR1, and ADDR2 had been set to blanks before this second output operation, the input save area would contain all blanks and zeros.

INZRCD (Initialize Record)

Use this record-level keyword to specify that if this record is not already on the display, it is to be written to the display before an input operation is sent from the program specifying this record name. If this record is already on the display, the keyword is ignored. The performance of this implicit output operation is OS/400-program-initiated; its only purpose is to format the display when an input operation is performed.

This keyword has no parameters.

This keyword does not apply to output operations.

If the INZRCD keyword is not specified, your program receives an error if it tries to read a record that is not on the display.

When the INZRCD keyword is processed, the following special conditions exist:

- For output-only fields, no user data is available. The field appears on the display as blanks. Any editing specified is ignored. The BLKFOLD keyword does not affect the display.
- For output/input fields, no user data is available. The field appears on the display as blanks. The input save area is initialized in the same way as uninitialized input-only fields (blanks or zeros, depending on the data type).
- Constants and input-only fields appear the same as when displayed using an explicit output operation.
- Hidden fields are returned on an input operation as blanks or zeros.

- Message and program-to-system fields are ignored.
- The LOGOUT keyword is ignored because there is no output buffer to log.
- The ERRMSG and ERRMSGID keywords are ignored because the record format is not already on the display.
- The SFLMSG and SFLMSGID keywords are ignored.
- All other optioned keywords and fields are processed as if they were optioned.

Note: In order for the INZRCD function to be performed, your program must specify a record format name when issuing an input operation that contains this keyword. The record format used for the input operation must specify the INZRCD keyword.

Option indicators are not valid for this keyword.

Figure 3-119 shows how to specify the INZRCD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00274A          R REC4                      INZRCD
     A
```

Figure 3-119. Specifying the INZRCD Keyword

KEEP (Keep)

Use this record-level keyword to keep the display from being deleted when the display file is closed. The entire display is kept if any of the records on the display have KEEP specified. The default causes the entire display to be deleted when the file is closed.

In addition, the name of the first, uppermost record on the display that has the KEEP attribute is saved by the OS/400 program for possible use by subsequent programs. The name kept can be used by a subsequent program that does not specify a record name on its first input operation. This keyword enables you to leave data on the display for review after your program ends, or use that data as input for subsequent programs.

This keyword has no parameters.

This keyword cannot be specified with the following keywords:

ALWROL
CLRL
SLNO

A warning message appears at file creation time if the KEEP keyword is specified on a record with the DSPMOD keyword. At run time, the KEEP keyword is ignored when the display mode changes.

Option and response indicators are not valid for this keyword.

Figure 3-120 on page 3-147 shows how to specify the KEEP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00061A          R REC46                     KEEP
     A
```

Figure 3-120. Specifying the KEEP Keyword

LOCK (Lock)

Use this record-level keyword to specify that the keyboard is to remain locked after an output operation. Normally, the keyboard automatically unlocks after an output operation. The LOCK keyword is used when there are several consecutive output operations that contain input fields.

This keyword has no parameters.

If this keyword is not specified, the work station user could key data into a field when a subsequent output operation sends data to the display. In this case, the cursor location could be changed and the key entry data lost.

Note: The default on an output operation is to unlock the keyboard. If the keyboard is locked when an input operation is sent, it is automatically unlocked.

This keyword is independent of other keywords that affect the output operation.

Option indicators are valid for this keyword.

Figure 3-121 shows how to specify the LOCK keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00011A          R REC1                      LOCK
     A
```

Figure 3-121. Specifying the LOCK Keyword

LOGINP (Log Input)

Use this record-level keyword to specify that the input buffer for this record format is to be written to the job log each time the OS/400 program performs an input operation for the record. The data logged includes the values of input-capable fields, hidden fields, and response indicators specified in the record format you are defining. It also includes output fields if the record format is a subfile record format. (Response indicators are not logged if the INDARA keyword is specified for the file you are defining.) This keyword is used for debugging and other exception conditions. The job log cannot be read by your program.

This keyword has no parameters.

The OS/400 program ignores LOGINP for either of the following conditions:

- There are no input-capable fields, no hidden fields, and no response indicators in the record format.
- The record format is a subfile record format for a message subfile.

Option indicators are not valid for this keyword.

Figure 3-122 on page 3-148 shows how to specify the LOGINP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00072A          R REC24                     LOGINP
     A
```

Figure 3-122. Specifying the LOGINP Keyword

LOGOUT (Log Output)

Use this record-level keyword to specify that the output buffer for this record format is to be written to the job log each time the OS/400 program performs an output operation for the record. The data logged includes the values of output-capable fields, hidden fields, and option indicators specified in the record format you are defining. (Option indicators are not logged if the INDARA keyword is specified for the file you are defining.) The LOGOUT keyword is used for debugging and other exception conditions. The job log cannot be read by your program.

This keyword has no parameters.

The OS/400 program ignores LOGOUT for either of the following conditions.

- There are no output-capable fields, no hidden fields, and no option indicators in the record format.
- The record format is a subfile record format for a message subfile.

Option indicators are valid for this keyword.

Figure 3-123 shows how to specify the LOGOUT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00021A          R REC25                     LOGOUT
     A
```

Figure 3-123. Specifying the LOGOUT Keyword

LOWER (Lower)

The LOWER keyword is equivalent to the CHECK(LC) keyword; the CHECK keyword is preferred. See "CHECK (Check)" on page 3-52 for an explanation of how to use a LOWER keyword.

MDTOFF (Modified Data Tag Off)

Use this record-level keyword with the OVERLAY keyword to set off modified data tags (MDTs) for input-capable fields in record formats already on the display.

The format of the keyword is:

```
MDTOFF[(*UNPR | *ALL)]
```

The MDTs are set off when your program sends an output operation to the record format you are defining.

To set off MDTs for unprotected fields only (those without the DSPATR(PR) keyword in effect), specify the *UNPR parameter value (this is also the default if no parameter value is specified). To set off MDTs for all input-capable fields, specify the *ALL parameter value.

Your program can select the DSPATR(MDT) keyword for fields in the same record format for which it selects MDTOFF (any parameter). If so, these fields are displayed with their MDTs set on.

The ERASEINP(*ALL) keyword implies MDTOFF(*UNPR) unless MDTOFF(*ALL) is specified.

If the ERASEINP(*MDTON) keyword is specified with MDTOFF(*ALL), the end effect is as if the ERASEINP(*ALL) keyword and MDTOFF(*ALL) are both specified. This is also true if the ERASEINP keyword is specified with no parameter value.

Option indicators are valid for this keyword.

MDTOFF is not valid for the subfile record format (identified by the SFL keyword). It is valid for all other record formats for which OVERLAY keyword is also specified.

Figure 3-124 shows how to specify the MDTOFF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                     OVERLAY MDTOFF
00020A            FLD1          6   B  2  2
00030A            FLD2          6   B  3  2
00040A*
00050A          R RECORD2                     OVERLAY MDTOFF(*UNPR)
00060A            FLD21         6   B  4  2
00070A            FLD22         6   B  5  2
00080A*
00090A          R RECORD3                     OVERLAY MDTOFF(*ALL)
00100A            FLD31         6   B  6  2
00110A            FLD32         6   B  7  2
00120A            FLD33         6   B  8  2DSPATR(PR)
     A
```

Figure 3-124. Specifying the MDTOFF Keyword

RECORD1 and RECORD2 have equivalent MDTOFF keyword specifications. When RECORD1 or RECORD2 is displayed, the MDT of each input-capable field already on the display is set off, unless the field has the DSPATR(PR) keyword in effect (FLD33, when displayed, is such a field). When RECORD3 is displayed, the MDTs of each input-capable field already on the display is set off even if the DSPATR(PR) keyword is in effect for the field.

MLTCHCFLD (Multiple-Choice Selection Field)

Use this field-level keyword to define a field as a multiple-choice selection field. A multiple-choice selection field is a field that contains a fixed number of choices from which a user can select multiple choices. The field appears as a vertical list of choices, with a field or a check box at the left of each choice.

If you see an input field, instead of a check box to the left of each choice, the selection character default value is a slash (/). Message CPX5A0C contains the country designated selection characters. The value can be changed to allow alternate selection characters for a multiple-choice selection field. The country designated selection characters are for all multiple-choice selection fields in the display file. The characters are defined when the display file is created.

This keyword has no parameters.

A field containing the MLTCHCFLD keyword must contain one or more CHOICE and CHCCTL keywords defining the choices for the field.

The field containing the MLTCHCFLD keyword must be defined as an input-capable field with the data type Y and length of two. The position specified for the field is the position of the input field displayed to the left of the first choice or of the upper-most check box. If *NOSLTIND is used on the PULLDOWN keyword and the device is attached to a controller that supports an enhanced interface for nonprogrammable work stations, the position is the first character of the text for the first choice. On input, the field contains the number of the choices selected, or 0 if no choice was selected. On output, the value of the field is ignored.

The following keywords can be specified on a field with the MLTCHCFLD keyword:

ALIAS
AUTO(RA)
BLANKS
CHANGE
CHCAVAIL
CHCUNAVAIL
CHCSLT[1]
CHCCTL
CHECK(ER, FE)[2]
CHOICE
CHGINPDFT
COLOR[3]
FLDCSRPRG
DSPATR(RI UL BL CS HI ND PC)[3]
ERRMSG
ERRMSGID
INDTXT
OVRATA
OVRATR
PUTRETAIN
TEXT

Notes:

1. CHCSLT functions only if the multiple-choice selection field is displayed in a pull-down menu that does not display selection indicators, when PULLDOWN(*NOSLTIND) is specified.
2. Check(FE) applies only to a display attached to a controller that does not support an enhanced interface.
3. If the COLOR or DSPATR keyword is specified for a field with the MLTCHCFLD keyword, it applies only to the input field portion of the selection field on character-based displays.

Option indicators are not valid for this keyword.

Figure 3-125 on page 3-151 shows how to specify the MLTCHCFLD keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD
     A            F1             2Y 0B  3 35MLTCHCFLD
     A  01                                  CHOICE(1 '>Undo       ')
     A                                      CHOICE(2 &MARKTXT)
     A                                      CHOICE(3 '>Copy       ')
     A                                      CHCCTL(1 &CTLONE1 MSG1111 QUSER/A)
     A                                      CHCCTL(2 &CTLTWO1 &MSG1 &LIB/&MSGF)
     A                                      CHCCTL(3 &CTLTHR1)
     A            CTLONE1         1Y 0H
     A            CTLTWO1         1Y 0H
     A            CTLTHR1         1Y 0H
     A            MSG1            7A  P
     A            LIB            10A  P
     A            MARKTXT        10A  P
     A
```

Figure 3-125. Specifying the MLTCHCFLD Keyword

The CHCCTL keyword is required for each CHOICE keyword used for the MLTCHCFLD.

On input, the hidden field for the CHCCTL keyword indicates whether or not the choice was selected. 0 indicates the choice was not selected; 1 indicates the choice was selected. On output, the hidden field controls the availability of the choice, and is used to set a default selection of a choice. 0 indicates the choice should be available, 1 indicates the choice should be selected by default, and 2 indicates the choice is unavailable. Other values such as 0 are truncated.

MNUBAR (Menu Bar)

Use this record-level keyword to define a menu bar. A menu bar is a horizontal list of choices that is followed by a separator line. The choices represent groups of related actions that the application user can select. For example, a group of actions appears in a pull-down menu when the user selects a menu-bar choice. A menu-bar record contains a field with one or more MNUBARCHC keywords that define the menu-bar choices. The separator line is generated by the system.

This keyword has no parameters.

A record with the MNUBAR keyword specified must contain one and only one menu bar field (a field with one or more MNUBARCHC keywords), and cannot contain any displayable fields other than the menu bar field.

The following keywords are allowed on a record containing the MNUBAR keyword:

CAnn
CFnn
CLEAR
CLRL
CSRLOC
DSPMOD
HELP
HLPCLR
HLPCMDKEY
HLPRTN
HLPTITLE
HOME
INDTXT
INVITE
KEEP
LOCK
MNUBARDSP
MNUBARSEP
MNUBARSW
MNUCNL
OVERLAY
PAGEDOWN/PAGEUP
PRINT
PROTECT
ROLLUP/ROLLDOWN
TEXT
UNLOCK
VLDCMDKEY

Note: These keywords are ignored on the menu-bar record if the menu-bar record is displayed by the system (for example, if MNUBARDSP is specified on a record other than this menu-bar record).

Option indicators are not valid for this keyword.

Figure 3-126 shows how to specify the MNUBAR keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                   MNUBAR
     A            MNUFLD         2Y 0I  1  2
     A                                       MNUBARCHC(1 RCDFILE 'File')
     A                                       MNUBARCHC(2 RCDEDIT 'Edit')
     A                                       MNUBARCHC(3 RCDVIEW 'View')
     A                                       MNUBARCHC(4 RCDOPT 'Options')
     A                                       MNUBARCHC(5 RCDHELP 'Help')
```

Figure 3-126. Specifying the MNUBAR Keyword

In this example, RECORD1 is defined as a menu-bar record. When RECORD1 is displayed, the menu-bar choices defined on the field MNUFLD are displayed as a menu bar. The menu-bar choices are followed by a separator line. On character-based displays, the separator line is made up of blue dashes. On a graphical display attached to a controller that supports an enhanced interface for nonprogrammable work stations, the separator line is a solid line.

MNUBARCHC (Menu-Bar Choice)

Use this field-level keyword to define a choice for a menu-bar field. A menu-bar choice represents a group of related actions that the application user can select. A group of actions appears in a pull-down menu when the user requests a menu-bar choice.

The format of the keyword is:

```
MNUBARCHC(choice-number pull-down-record choice-text
          [&return-field])
```

The choice-number parameter is required and specifies an identification number. The choice number is returned to the application to indicate which choice in the menu bar was selected. Valid values for the choice number are integers 1 to 99. Duplicate values within a single menu-bar field are not allowed.

The pull-down-record parameter is required and specifies the name of the pull-down record that is displayed when the user selects this choice. The record specified must exist within the file and must contain a PULLDOWN keyword.

The choice-text parameter is required and defines the text that appears in the menu bar for the choice. The parameter can be specified in one of two forms:

As a character string: 'Choice text '

As a program-to-system field: &field-name

The field-name specified must exist in the menu bar record and must be defined as a character field with usage P.

The choice text must fit on one line of the display for the smallest display size specified for the file. Since the text for the first menu-bar choice on a line begins at

position 3 and a trailing blank is always inserted after the choice text, the maximum length of the choice text is 76 if the smallest display size for the file is 24 x 80 and 128 if the smallest display size for the file is 27 x 132.

When the choice text contained in the character string or the program-to-system field is displayed, trailing blanks in the text are truncated and 3 blank spaces are inserted between choices. However, the number of lines that the menu-bar field occupies on the display is determined by the sum of the lengths of the choice-text parameters (either the length of the character string or the length of the program-to-system field) on all the MNUBARCHC keywords for the menu bar, including trailing blanks, plus 3 blank spaces between each choice. The maximum number of lines that a menu bar field may occupy is 12 lines (this includes the separator line).

Within the choice text, you can specify a mnemonic for the choice by using a greater than character (>) to indicate the mnemonic character. The character to the right of the > is the mnemonic. Examples:

Choice Text	**Appears as**
'>File'	File
'F>inish'	Finish
'Save >As...'	Save As...
'X >= 1'	X ≥ 1

To specify a > as a character in the text, you must specify it twice, just as you must specify the apostrophe character twice in order to get a single apostrophe character in the text.

Choice Text	**Appears as**
'X >>= 1'	X >= 1
'X >>>= 1'	X >≥ 1

Note: It is not possible to specify the > as the mnemonic character.

The mnemonic character indicated must be a single-byte character and must not be a blank. Only one mnemonic is allowed in the choice text, and the same mnemonic character cannot be specified for more than one choice.

The return-field parameter is optional and specifies whether or not control is returned to the application because a menu bar choice was selected. This parameter specifies the name of a hidden field in the menu-bar record that contains the number of the choice selected when control is returned to the application. The hidden field is defined as a data type Y (numeric), the length of the field is two, and decimal positions are 0. The presence of a choice number in this field indicates that control has been returned to the application because a menu-bar choice was selected. The next operation of the application updates (if necessary) and writes the pull-down record associated with that choice; that is, the pull-down record specified on the MNUBARCHC keyword for the choice. When a choice number is returned in this field, zero is returned in the field that contains the choice number after pull-down input has been received. Likewise, when pull-down input has been received, zero is returned in this field, and the presence of a choice number in menu-bar field or the choice field in the application record indicates that the application should process the pull-down input.

The menu bar field contained in the MNUBARCHC keywords is defined as an input-capable field with data type Y (numeric). The length of the field is two and decimal positions 0. If the menu bar record is read, the number of the choice

selected (if any) is returned in the menu-bar field. The menu-bar field must always be defined as starting in row 1, column 2.

When MNUBARCHC is specified on a field, the MNUBAR keyword is required at the record level.

Multiple MNUBARCHC keywords can be specified for one menu bar field. The number of MNUBARCHC keywords that can be specified is limited only by the lengths of the choice text parameters and the 12 line limit for a MNUBAR. All the choices defined for a menu bar field must fit on the screen, allowing for 3 spaces between each choice.

The following keywords can be specified on a field with the MNUBARCHC keyword:

ALIAS	INDTXT
CHCAVAIL	MNUBARSEP
CHCSLT	TEXT

Option indicators are valid for this keyword.

Figure 3-127, Figure 3-128 on page 3-155, and Figure 3-129 on page 3-156 show how to specify the MNUBARCHC keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8

     A                                      DSPSIZ(*DS3 *DS4)
     A          R MENUBAR                   MNUBAR
     A            MNUFLD         2Y 0B  1  2
     A                                      MNUBARCHC(1 PULLFILE +
     A                                         '>File          ')
     A  01                                  MNUBARCHC(2 PULLEDIT +
     A                                         '>Edit          ')
     A                                      MNUBARCHC(3 PULLVIEW +
     A                                         '>View          ')
     A                                      MNUBARCHC(4 PULLOPT +
     A                                         '>Options       ' &RTNFLD)
     A                                      MNUBARCHC(5 PULLHELP +
     A                                         '>Help          ')
     A            RTNFLD         2Y 0H
```

Figure 3-127. Specifying the MNUBARCHC Keyword (Example 1)

In Figure 3-127, five menu bar choices (File, Edit, View, Options and Help) are defined in a menu bar. If option indicator 01 is on and the menu bar record is written before the system displays it, the Edit choice is displayed when the system displays the menu bar. If option indicator 01 is off or the menu-bar record is not written before the system displays it, the Edit choice is not displayed. If the Edit choice is not displayed, the list of choices are compressed and the View choice will follow the File choice, with 3 blank spaces in between.

If the File choice is selected, record PULLFILE is displayed as a pull-down menu beneath the File choice. If the Options choice is selected, control is returned to the application. The application can update the PULLOPT record before the system displays it as a pull-down menu.

On displays capable of a single-character underline, the mnemonic for each choice is the first character in the text. If the menu-bar record is read, the menu-bar field

MNUFLD contains the number of the choice selected, or 0 if no choice was selected.

The text for each choice has been specified as a character string, with 15 spaces available for text. Thus, the maximum space required for the menu bar is 87 positions (15 possible text positions for each of the 5 choices plus 3 spaces between choices). Because the smallest display size is 24x80 (*DS3), the menu-bar choices occupy 2 lines. The menu-bar separator occupies one more line, with the entire menu bar occupying 3 lines. However, the actual text coded above for each choice does not use all 15 positions. When the menu bar is displayed, the trailing blanks are truncated and 3 blanks are inserted between each choice. Thus, the actual space in which the menu bar is displayed is 36 positions. The choices are displayed on one line followed by the separator on the next line. Because the menu bar actually occupies 3 lines, the line following the separator is blank.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
                                        DSPSIZ(*DS3 *DS4)
     A          R MENUBAR                MNUBAR
     A            MNUFLD         2Y 0B  1  2
     A                                     MNUBARCHC(1 PULLFILE +
     A                                        &FILETXT)
     A  01                                 MNUBARCHC(2 PULLEDIT +
     A                                        &EDITTXT)
     A                                     MNUBARCHC(3 PULLVIEW +
     A                                        &VIEWTXT)
     A                                     MNUBARCHC(4 PULLOPT +
     A                                        &OPTTXT &RTNFLD)
     A                                     MNUBARCHC(5 PULLHELP +
     A                                        &HELPTXT)
     A            FILETXT       15A  P
     A            EDITTXT       15A  P
     A            VIEWTXT       15A  P
     A            OPTTXT        15A  P
     A            HELPTXT       15A  P
     A            RTNFLD         2Y 0H
     A
```

Figure 3-128. Specifying the MNUBARCHC Keyword (Example 2)

Figure 3-128 is the same as Figure 3-127 on page 3-154, except that the choice text has been specified using program-to-system fields.

At run time, the choice text to be displayed for each choice is retrieved from the program-to-system fields. Any mnemonics must be specified in the text supplied by the application at run time. As Figure 3-127 on page 3-154 when the menu-bar record is read, the menu-bar field MNUFLD contains the number of the choice selected, or 0 if no choice was selected.

As in Figure 3-127 on page 3-154, the number of spaces available for the text for each choice is 15. The maximum space required for the menu bar is 78 positions (15 possible text positions for each of the 5 choices plus 3 spaces between choices). Because the smallest display size is 24x80 (*DS3), the menu-bar choices occupy 2 lines. The menu-bar separator occupies one more line, so the entire menu bar occupies 3 lines. However, the actual number of lines that is used to display the choices depends on the text that is contained in the program-to-system fields. When the menu bar is displayed, the trailing blanks in the P-fields are truncated, and 3 blanks are inserted between each choice.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R MENUBAR                   MNUBAR
     A            MNUFLD         2Y 0B  1  2
     A                                      MNUBARCHC(1 PULLFILE +
     A                                         '>File        ')
     A                                      MNUBARCHC(2 PULLEDIT +
     A                                         '>Edit        ' &RTNFLD)
     A            RTNFLD         2Y 0H
```

Figure 3-129. Specifying the MNUBARCHC Keyword (Example 3)

In Figure 3-129, if choice 2 in the menu bar is selected, control is returned to the application and field RTNFLD contains the number 2. Field MNUFLD contains 0, indicating no pull-down input received. The application must read record MENUBAR in order to get the contents of field RTNFLD. The application must then write record PULLEDIT. The system resumes control of the menu-bar interaction. If input is then entered in record PULLEDIT, control is returned to the application, and field MNUFLD contains number 2. Field RTNFLD contains 0, indicating control has been returned because pull-down input was received.

If choice 1 is selected, the system displays pull-down record PULLFILE. If input is entered in PULLFILE, control is returned to the application, and field MNUFLD contains number 1. Field RTNFLD contains 0, indicating control has been returned because pull-down input was received.

MNUBARDSP (Menu-Bar Display)

Use this record-level keyword to display a menu bar. The MNUBARDSP keyword has two formats: one for records that contain the MNUBAR keyword and one for records that do not contain a MNUBAR keyword.

The format for MNUBARDSP when specified on a record that is not a menu bar record is:

```
MNUBARDSP(menu-bar-record &choice-field [&pull-down-input])
```

The format for MNUBARDSP when specified on a menu bar record is:

```
MNUBARDSP[(&pull-down-input)]
```

The menu-bar-record parameter specifies the menu-bar record that is to be displayed when this record is written. The menu-bar record must exist in the same file as the record you are defining.

The &choice-field parameter specifies the name of a hidden field, which on input contains the number of the choice (if any) selected from the menu bar. The field must exist in the record you are defining and it must be defined as numeric Y in position 35, usage H, length 2, and decimal positions 0.

The &pull-down-input parameter is optional and specifies the name of a hidden field that contains the input from the pull-down menu when the pull-down menu contains only a single-choice selection field. The field must exist in the record you are defining and it must be defined as a length 2, decimal positions 0, and zoned (S in position 35) field with usage H (hidden). On input, this field contains one of the following values:

Value	Meaning
0	No selection made

n	Choice n in the pull-down menu was selected
-1	Pull-down record contains something other than the one single-choice selection field. You must read the pull-down record to receive its contents

Option indicators are valid for the MNUBARDSP keyword, and more than one MNUBARDSP keyword can be specified on the record if all are optioned. If more than one MNUBARDSP keyword is in effect when the record is written, the first one in effect is used.

Figure 3-130 shows how to specify the MNUBARDSP keyword on a record that is not a menu bar.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD1
     A  01                                  MNUBARDSP(MENURCD &MNUCHOICE &INPUT)
     A            FIELD1         10A  B 10 12
     A            FIELD2          5S 0B 14 12
     A            MNUCHOICE       2Y 0H
     A            INPUT           2S 0H
     A          R MENURCD                   MNUBAR
     A            F1              2Y 0B  1  2
     A                                      MNUBARCHC(1 PULLFILE 'File')
     A                                      :
     A                                      :
     A
```

Figure 3-130. Specifying the MNUBARDSP Keyword (Example 1)

In this example, if option indicator 01 is on when record RECORD1 is written to the display, the system displays the menu bar in record MENURCD. When the menu bar is activated and the pull-down menu is selected, the number of the menu-bar choice is returned in field MNUCHOICE. If the pull-down menu selected contains one single-choice selection field, the choice made for that field is returned in field INPUT. Otherwise, field INPUT contains -1, indicating that the application must read the pull-down record to receive the pull-down input.

Figure 3-131 shows how to specify the MNUBARDSP keyword on a menu-bar record.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R MENURCD                   MNUBAR
     A  01                                  MNUBARDSP
     A            F1              2Y 0B  1  2
     A                                      MNUBARCHC(1 PULLFILE 'File')
     A                                      :
     A                                      :
     A
```

Figure 3-131. Specifying the MNUBARDSP Keyword (Example 2)

If option indicator 01 is on when record MENURCD is written to the display, the system displays the menu bar defined in MENURCD. If the user selects a pull-down menu from the menu bar, the number of the menu-bar choice selected is returned in F1 field.

MNUBARSEP (Menu-Bar Separator)

Use this field-level keyword on a menu-bar field to specify the color, display attributes, or character used to form the menu-bar separator line.

The format of the keyword is:

```
MNUBARSEP([color] [display-attribute] [character])
```

One parameter must be specified.

The color parameter specifies the color of the separator characters on a color work station. The parameter is specified as an expression of the form (*COLOR value).

The valid values for the color parameter are:

Value	Meaning
BLU	Blue
GRN	Green
PNK	Pink
RED	Red
TRQ	Turquoise
YLW	Yellow
WHT	White

If the color parameter is not specified, the default is blue. This parameter is ignored if it is specified for a menu bar on a monochrome display.

The display-attribute parameter specifies the display attributes of the separator characters. The parameter is specified as an expression of the form (*DSPATR value1 <value2 <value3...>>).

The valid values for the display attributes are:

Value	Meaning
BL	Blink
CS	Column separator
HI	High intensity
ND	Nondisplay
RI	Reverse image
UL	Underline

The default display attribute for the menu-bar separator is normal (or low) intensity.

Note: Display attributes CS, HI, and BL can cause fields on 5292, 3179, 3197 Model C1 and C2, 3487 Models HC, and 3488[7] work stations to appear as color fields. Display attributes HI, RI, and UL cause a separator line not to be displayed. For more information on the COLOR keyword, see "COLOR (Color)" on page 3-72.

The character parameter specifies the character that makes up the separator line. The parameter is specified as an expression of the form (*CHAR

[7] Dependent on monitor attached to display device.

'separator-character'). The separator-character value is one character. If this parameter is not specified, the default separator character is a dash (-) or on a graphical device this shows as a solid line. Although any displayable character may be specified as the separator character, it is recommended that you use invariant characters.

Figure 3-132 shows the invariant characters:

Figure 3-132. Character Set for System Data

Hexadecimal	Character	Description
40		Blank
4B	.	Period
4C	<	Less-than sign
4D	(	Left parenthesis
4E	+	Plus sign
50	&	Ampersand
5C	*	Asterisk
5D	)	Right parenthesis
5E	;	Semicolon
60	-	Minus sign
61	/	Slash
6B	,	Comma
6C	%	Percent sign
6D	_	Underline
6E	>	Greater-than sign
6F	?	Question mark
7A	:	Colon
7D	'	Apostrophe
7E	=	Equal sign

Note:

In addition, you can use any of the following:

- Uppercase English alphabetic characters: A through Z
- Numeric characters: 0 through 9

When the MNUBARSEP keyword is specified on a field, the MNUBAR keyword must also be specified on the associated record.

Option indicators are valid for this keyword.

If more than one COLOR keyword is specified, the color parameter, display attribute, and separator character are taken from the first keyword that was specified.

Figure 3-133 on page 3-160 shows how to specify the MNUBARSEP keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R MENUBAR                    MNUBAR
     A            MNUFLD          2Y 0B  1  2
     A                                       MNUBARSEP((*COLOR PNK) +
     A                                       (*DSPATR RI) (*CHAR  ' '))
     A                                       MNUBARCHC(1 PULLFILE +
     A                                          'File         ')
     A                                       MNUBARCHC(2 PULLEDIT +
     A                                          'Edit         ')
     A
```

Figure 3-133. Specifying the MNUBARSEP Keyword

In this example, the menu-bar separator is made up of pink blanks displayed in reverse image.

MNUBARSW (Menu-Bar Switch Key)

Use this file- or record-level keyword to assign a CA key to be the Switch-to-menu-bar key. If the menu-bar switch key is active and a menu bar is displayed, pressing the CA key will do one of the following:

- If the cursor is located on the application record, pressing the switch key moves the cursor to the first field in the menu bar.
- If the cursor has been moved to the menu bar using the switch key, pressing the switch key again moves the cursor back to the location the cursor was when the switch key was pressed to move the cursor into the menu bar.
- If the cursor has been moved to the menu bar manually (for example, using the cursor movement keys), pressing the switch key moves the cursor to the first input-capable field in the application record.

The format of the keyword is:

```
MNUBARSW [(CAnn)]
```

The CAnn parameter is optional. If not specified, the default is CA10. Valid values for the CAnn parameter are CA01 through CA24.

Within a record, the CAnn key specified by the MNUBARSW keyword cannot be specified again using another keyword (such as MNUCNL). Because MNUBARSW at the file level extends to all records in the file, this must be considered when assigning a CAnn key.

If the MNUBARSW keyword is specified on the record, the CAnn key or default CA10 key can be used only as a CA key on other records, not as a CF key.

The MNUBARSW keyword is allowed only in a file containing a menu-bar record.

Option indicators are valid for this keyword.

Figure 3-134 on page 3-161 shows how to specify the MNUBARSW keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     MNUBARSW(CA10)
     A          R MENUBAR                  MNUBAR
     A            MNUFLD         2Y 0B  1  2
     A                                     MNUBARCHC(1 PULLFILE +
     A                                         '>File        ')
     A  01                                 MNUBARCHC(2 PULLEDIT +
     A                                         '>Edit        ')
     A
     A          R PULLEDIT                 PULLDOWN
     A            F1             1D 0B  1   2RANGE(1 3)
     A                                1  5'1. Undo        '
     A                                2  4'2. Mark        '
     A                                3  4'3. Copy        '
     A            :
     A            :
     A          R APPSCR                   MNUBARDSP(MENUBAR &MNUCHOICE)
     A            FIELD1         10A  B 10 12
     A            FIELD2          5S 0B 14 12
     A                                   24  1'F12=Cancel '
     A            MNUCHOICE       2S 0H
     A
```

Figure 3-134. Specifying the MNUBARSW Keyword

In this example, CA10 is defined as the Switch-to-menu-bar key for all records in the file. When the cursor is located anywhere except in the menu bar and CA10 is pressed, the cursor is moved to the File choice on the menu bar. If CA10 is pressed again while the cursor is located anywhere in the menu bar, the cursor is moved back to its previous location within the APPSCR record.

MNUCNL (Menu-Cancel Key)

Use this file- or record-level keyword to assign a CA key to be the cancel key for menu bars or pull-down menus. When the MNUCNL keyword is active and a pull-down menu is displayed, pressing the CA key cancels the pull-down menu and returns the cursor to the choice in the menu bar. If no pull-down menu is displayed and the cursor is located in the menu bar, pressing the CA key cancels the menu bar and returns the cursor to the application screen. If no pull-down menu is displayed and the cursor is on the application screen, pressing the CA key returns control to the application.

The format of the keyword is:

```
MNUCNL[(CAnn [response-indicator])]
```

The CAnn parameter is optional. If not specified, the default is CA12. Valid values are CA01 through CA24.

The response-indicator parameter is optional. The parameter is set on when the MNUCNL keyword is active on a record other than a menu bar or pull-down menu, and control is being returned to the application.

Within a record, the CAnn key specified by the MNUCNL keyword cannot be specified again using another keyword (such as MNUBARSW). Because MNUCNL at the file-level extends to all records in the file, this must be considered when assigning a CAnn key.

If the MNUCNL keyword is specified on the record, the CAnn key or default CA12 key can be used only as a CA key on other records, not as a CF key.

The MNUCNL keyword is allowed only in a file containing a menu-bar record.

Option indicators are valid for this keyword.

Figure 3-135 shows how to specify the MNUCNL keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     MNUCNL(CA12 12)
     A          R MENUBAR                  MNUBAR
     A            MNUFLD         2Y 0B  1  2
     A                                     MNUBARCHC(1 PULLFILE +
     A                                        '>File        ')
     A  01                                 MNUBARCHC(2 PULLEDIT +
     A                                        '>Edit        ')
     A
     A          R PULLEDIT                 PULLDOWN
     A            F1             1D 0B  1  2RANGE(1 3)
     A                                   1  5'1. Undo        '
     A                                   2  4'2. Mark        '
     A                                   3  4'3. Copy        '
     A            :
     A            :
     A          R APPSCR                   MNUBARDSP(MENUBAR &MNUCHOICE)
     A            FIELD1        10A  B 10 12
     A            FIELD2         5S 0B 14 12
     A                                  24  1'F12=Cancel '
     A            MNUCHOICE      2S 0H
```

Figure 3-135. Specifying the MNUCNL Keyword

In this example, CA12 is defined as the cancel key for all records in the file. If CA12 is pressed while the pull-down menu, PULLEDIT, is displayed, the pull-down menu is canceled. If CA12 is pressed while the cursor is located on the menu bar (no pull-down menus are displayed), the menu bar is canceled and the cursor is moved back to the application record. If CA12 is pressed while the cursor is not located on the menu bar and no pull-down menu is displayed, response indicator 12 is set on and control is returned to the application program.

MSGALARM (Message Alarm)

Use this file- or record-level keyword to specify that the system is to sound the audible alarm when this record is displayed with an active ERRMSG, ERRMSGID, SFLMSG, or SFLMSGID keyword, or when a validity checking error is detected. The alarm is of short duration.

This keyword has no parameters.

If you specify the MSGALARM and ALARM keywords on the same record format and both are active, the alarm sounds only once.

Option indicators are valid with this keyword.

Figure 3-136 on page 3-163 shows how to specify the MSGALARM keyword at record level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD1
00020A                                        MSGALARM
00030A            FLD01          8A    12 10
00040A  12                                    ERRMSGID(XYZ0123 MSGFILE)
     A
```

Figure 3-136. Specifying the MSGALARM Keyword at Record Level

When record format RCD1 is on the display and RCD1 is put to the display again and indicator 12 is on, the message XYZ0123 from message file MSGFILE is displayed on the message line and the work station alarm sounds.

Figure 3-137 shows how to specify the MSGALARM keyword at file level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  01                                    MSGALARM
00020A          R RCD1
00030A            FLD01          8A    12 10
00040A  12                                    ERRMSGID(XYZ0123 MSGFILE)
00050A
00060A          R RCD2
00070A            FLD02          8A    12 10
00080A  10                                    ERRMSG('Message text')
     A
```

Figure 3-137. Specifying the MSGALARM Keyword at File Level

When record format RCD1 is on the display and RCD1 is put to the display again and indicators 01 and 12 are on, the message XYZ0123 from message file MSGFILE is displayed on the message line, and the work station alarm sounds.

When record format RCD2 is on the display and RCD2 is put to the display again and indicator 10 is on (but 01 is off), the message text is displayed on the message line and the alarm does NOT sound.

MSGCON (Message Constant)

Use this field-level keyword to indicate the text for constant fields is contained in a message description. If the message description does not exist at DDS compile time, the file will not be created. If you change the message description, you will have to create the file again if you want the display file to contain the updated message.

The format of the keyword is:

```
MSGCON(length message-ID [library-name/]message-file-name)
```

The length parameter specifies the maximum length of the message description. The length can be from 1 to 132 bytes. If the message description is less than the length specified, the remaining bytes are padded with blanks (hex 40). If the message description is longer than the length specified, the message description is truncated to the specified length and a warning message appears.

The message-ID parameter specifies the message description that contains the text to use as the value of the constant field.

The message-file-name parameter identifies the message file that contains the message description. The library-name parameter is optional.

The MSGCON keyword must be explicitly specified for the field. The MSGCON keyword cannot be used to initialize a named field.

The DFT and MSGCON keywords are functionally equivalent. If you specify the DFT and MSGCON keywords for the same field, the MSGCON keyword is ignored and the file is not created.

The MSGCON keyword cannot be specified with any of the following keywords:

DATE
DFT
EDTCDE
EDTWRD
TIME

Option indicators are not valid for changing the value of the message line, but they are valid for conditioning the presence or absence of the message on the display.

Figure 3-138 shows how to specify the MSGCON keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A                                    2  1MSGCON(10 MSG0001 MESSAGE/MSGF)
     A
```

Figure 3-138. Specifying the MSGCON Keyword

MSG0001 in message file MSGF in library MESSAGES contains the message text.

MSGID (Message Identifier)

Use this field-level keyword to allow an application program to identify, at program run time, the message description that contains text for a named field. The parameter values on this keyword can specify fields that contain the message identifier, message file, and library. After the program sets the fields to the desired values, an output operation causes the message to be retrieved from the message file and displayed. The retrieved message is truncated if it is longer than the MSGID field. The retrieved message is padded with blanks if it is shorter than the MSGID field.

Refer to Appendix F, "System/36 Environment Considerations," for information on how to specify the MSGID keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
MSGID(message-identifier [library-name/]message-file)
```

The message-identifier parameter describes the message description containing the text for the named field. You can specify this parameter in one of two ways:

- `[msg-prefix] &field-name`

  A message-prefix and a field-name. The field-name must exist in the same record format as the MSGID field. If you specify a prefix, the length of the prefix must be three, and you must define the field-name as a character field of length four (4), and usage H, P, B, or O. If you do not specify a prefix, you must define the field-name as a character field length of seven (7), and usage H, P, B, or O.

- [msgid-prefix] msg-id

  You can also use a value or a combination of values to specify the message identifier. If you specify a prefix, the prefix length must be three (3), and the msg-id length must be four (4). If you do not specify a prefix, the msg-id length must be seven (7).

The message-identifier is a required parameter.

The message-file and library parameters identify the message file containing the message description. You can specify the message-file and library parameters in one of the following forms:

- [library-name/]file-name
- [&field1/]&field2

  where the lengths of field1 and field2 are ten (10).

  The field names must exist in the same record format as the MSGID field, and the fields must be defined as having character lengths of ten (10) and usage H, P, B, or O.

- Combination of field names and constants:
  - library-name/&field1 (length of field1 is 10)
  - &field2/file-name (length of field2 is 10)

The message-file is a required parameter. If you do not specify the library parameter, *LIBL is used to search for the message file at program run time. The library is an optional parameter.

The user or program may override the message file name by using the OVRMSGF command.

You can specify multiple MSGID keywords on a field. When more than one MSGID keyword is specified, option indicators are required on all except the last MSGID keyword on a field. Option indicators are not allowed on the last (or only) MSGID keyword specified on a field. If more than one MSGID keyword is in effect for a field, the first MSGID specified is used.

You can specify multiple MSGID keywords within a record format. You can specify field names used as parameters on more than one MSGID keyword.

You must have use authority to the message file at program run time.

The MSGID field must be output-capable (usage B or O). The length specified for the field is the length of the message text that is to be displayed. It should be the length of the longest message to be displayed.

The MSGID field itself does not appear in the output buffer, but it does appear in the input buffer if it is defined as output/input (usage B). MSGID fields that are defined as output-only cannot be used in high-level language programs.

The MSGID parameter fields (if any) appear in the output buffer, but they appear in the input buffer only if they are defined as hidden or output/input (usage H or B) fields.

A program-to-system field can be specified as the message-id, file, or library name.

You cannot specify MSGID in a subfile record format (SFL keyword).

The following keywords cannot be specified on a field with the MSGID keyword:

DFT
DFTVAL
FLTFIXDEC
FLTPCN
MSGCON

Figure 3-139 shows how to specify the MSGID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1
     A            MSGFIELD1     40A  B 02 10MSGID(CPD0001 QGPL/USRMSG)
     A  01        MSGFIELD2     10A  O 02 60
     A  25                                  MSGID(&MSGIDNUM &MSGFILENM)
     A                                      MSGID(CPD1234 QGPL/USRMSG)
     A            MSGFIELD3     80A  B 02 60
     A                                      MSGID(USR &MSGNBR +
     A                                      QGPL/&MSGGILENM)
     A            MSGIDNUM       7A  P
     A            MSGFILENM     10A  H
     A            MSGNBR         4A  B 07 01
     A
```

Figure 3-139. Specifying the MSGID Keyword

When RECORD1 is displayed:

- MSGFIELD1 contains the first 40 characters of the message CPD0001 from the message file USRMSG in library QGPL. Since the field is output/input (usage B), the value of the field can be changed by the user.
- If option indicator 01 is off, MSGFIELD2 is not displayed. When option indicators 01 and 25 are on, MSGFIELD2 contains the first 10 characters of the message identified by the fields MSGIDNUM and MSGFILENM. Values for MSGIDNUM (the message identifier) and MSGFILENM (the message file) must be set in the program prior to the display of RECORD1.

  When option indicator 01 is on but option indicator 25 is off, MSGFIELD2 contains the first 10 characters of message CPD1234 from the message file USRMSG in library QGPL. Since MSGFIELD2 is an output-only field (usage O), it cannot be used in the program.
- MSGFIELD3 contains the first 80 characters of the message identified by the prefix USR, the message number set in field MSGNBR, and the message file set in field MSGFILENM.

MSGLOC (Message Location)

Use this file-level keyword to move the message line to the specified line number.

The format of the keyword is:

```
MSGLOC(line-number)
```

The parameter value is required and must be in the range 1 through 28. Any of these numbers are always valid, regardless of the display sizes specified on the DSPSIZ keyword. A diagnostic will be issued when the file is opened if a message location is in the 26 to 28 range for a 24 x 80 display size.

If MSGLOC is not specified, the message line is the last line of the display. The message line is the display location for the following messages:

- Validity check errors
- Invalid function key messages
- Messages defined as parameter values for the ERRMSG and SFLMSG keywords
- Messages identified by the ERRMSGID and SFLMSGID keywords (the entire display is used for message help)
- Message fields
- Operator error codes and their associated messages

Display size condition names must be specified if the message line for the secondary display size is different from the default message line.

If you do not specify the MSGLOC keyword, the following default values are to be established:

24 x 80 display size: line 25

27 x 132 display size: line 28

The default of line 25 for the 24 x 80 display size results in the following:

- If the display is sent to a 5250 device or a 5251 model 12, line 24 is used as the message line.
- If the display is sent to a 3180-2 device or a 3197 model D1, D2, W1, or W2 attached to a local 6040 or 6041 controller, or remotely attached to a 5294 or 5394 controller, line 25 is used as the message line.

If the ERRSFL keyword is specified in the file, you may not specify a message location value of 25 for the 24 x 80 display size or 28 for the 27 x 132 display size. When the ERRSFL keyword is in the file, but MSGLOC is not specified, the following default values are to be established:

24 x 80 display size: line 24

27 x 132 display size: line 27

The MSGLOC keyword specification is in effect continuously from file opening to file closing. It can be temporarily overridden if the file you are defining is suspended while another file is opened to the same work station device. The message location in effect for the other file is used until the file you are defining is restored.

Any data on the message line before the message appears is saved and restored after the Reset key is pressed.

Option indicators are not valid for this keyword.

Figure 3-140 on page 3-168 and Figure 3-141 on page 3-168 show how to specify the MSGLOC keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     MSGLOC(1)
     A
```

Figure 3-140. Specifying the MSGLOC Keyword (Example 1)

In Figure 3-140, the message line is moved to line 1 for the primary display size. (Without the DSPSIZ keyword, the primary display size is the 24 x 80 display.)

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                     DSPSIZ(*DS3 *DS4)
     A                                     MSGLOC(1)
     A  *DS4                               MSGLOC(1)
     A
```

Figure 3-141. Specifying the MSGLOC Keyword (Example 2)

The message line is moved to line 1 for both the primary display size 1 and the secondary display size 2.

NOCCSID (No Coded Character Set Identifier)

Use this field-level keyword to specify that CCSID conversion of the field will not be done.

This keyword has no parameters.

The Character Data Representation Architecture (CRDA) specifies the '3F'X character as a replacement character. This character is also a field attribute definition for the 5250 data stream specification. Translation converting '3F'X character to '1F'X for output is done for all fields whether *JOBCCSID translation is active or inactive. Use the NOCCSID keyword to prevent translation at the field level.

If the NOCCSID keyword is not specified, conversion of the field continues normally.

Figure 3-142 shows how to specify the NOCCSID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD
     A            FIELD1         5A  B  2 10NOCCSID
     A
```

Figure 3-142. Specifying the NOCCSID Keyword

OPENPRT (Open Printer File)

Use this file-level keyword to specify that once the printer file is opened (the first time the user presses the Print key), it is to remain open until the associated display file is closed. If you do not specify OPENPRT (and the PRINT keyword is specified), the printer file is opened and closed each time a display image is printed.

The printer file should be spooled if more than one job uses the same printer file and device. While the printer file is open in the nonspooled mode, the associated printer is allocated to the program or process using this function.

This keyword has no parameters.

This keyword is valid only if you have specified a file-level PRINT keyword with a printer file parameter. It is not valid with record-level PRINT keywords.

The OPENPRT keyword has no effect unless the PRINT file is specified on the PRINT keyword.

Option indicators are not valid for this keyword.

Figure 3-143 shows how to specify the OPENPRT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00031A                                  PRINT(PRTFILE)
00032A                                  OPENPRT
     A
```

Figure 3-143. Specifying the OPENPRT Keyword

OVERLAY (Overlay)

Use this record-level keyword to specify that the record format you are defining should appear on the display without the entire display being deleted first.

This keyword has no parameters.

Normally, the entire display is deleted on each output operation. All records on the display with fields that partially or completely overlap fields in this record are deleted before this record is displayed; all others remain on the display and are not changed in any way. A record already on the display is deleted even if fields specified in the record format are not selected for display. For example, assume that the following records are on the display:

REC1 (lines 1 and 2)
REC2 (lines 3 and 4)
REC3 (line 5)
REC4 (line 9)

An output of REC5 (lines 4 and 5) with OVERLAY would leave the display with the following records:

REC1 (lines 1 and 2)
REC5 (lines 4 and 5)
REC4 (line 9)

If the record with the OVERLAY keyword in effect is already on the display and PUTOVR, PUTRETAIN, or CLRL keyword is not specified, it is deleted and rewritten as a new record.

When the beginning attribute character of a record overlaps the ending attribute character of a record already displayed, the attribute characters overlap each other

in position 1 of a line. (The last field of the first record displayed ends in the farthest right display position of the preceding line.)

In the above example, however, if the only portion of REC2 on line 4 is the ending attribute character of the last field of REC2 (which occurs when the last displayed character of the last field of REC2 is in the last position of line 3), REC2 remains displayed following the display of REC5 with OVERLAY. The display would have the following records:

REC1 (lines 1 and 2)
REC2 (line3)
REC5 (lines 4 and 5)
REC9 (line 9)

The display is always deleted on the first output operation after the file is opened, except when both ASSUME and OVERLAY are specified.

OVERLAY is assumed by the OS/400 program for ERRMSG, ERRMSGID, PUTOVR, and CLRL functions.

If OVERLAY is conditioned and not selected, then the ERASE, ERASEINP, MDTOFF, PROTECT, and PUTRETAIN keywords cannot take effect if they are selected, unless the PUTOVR keyword is selected. In such cases, the ERASE, ERASEINP, and MDTOFF keywords can take effect.

If you specify OVERLAY, you should also specify the RSTDSP(*YES) keyword on the CRTDSPF (Create Display File) or CHGDSPF (Change Display File) command. Otherwise, data on the display can be lost if the file is suspended.

To delete any of the records on the display, use the ERASE keyword to specify the names of the record formats to be deleted.

If you also specify the CLRL keyword, processing proceeds according to the CLRL specification, not the OVERLAY specification.

A warning message is sent at file creation time if the OVERLAY keyword is specified on a record with the DSPMOD keyword. At run time, the OVERLAY keyword is ignored when the display mode changes.

Option indicators are valid for this keyword.

Figure 3-144 shows how to specify the OVERLAY keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00021A          R RECL                      OVERLAY
     A
```

Figure 3-144. Specifying the OVERLAY Keyword

OVRATR (Override Attribute)

Use this field- or record-level keyword with the PUTOVR keyword to override the existing display attributes of a field or record already on the display.

The OVRATR keyword can be used with the OVRDTA keyword on the same field or record.

When OVRATR is specified at both the record and field level, the field level specification is used for that field.

See the *Guide to Programming Displays* for information on how to use OVRATR in files that are used in the System/36 environment.

This keyword has no parameters.

The display attributes that can be overridden by the OVRATR keyword are:

CHECK(ER)
CHECK(ME)
DSPATR (all except OID and SP)
DUP

When the OVRDTA keyword is in effect, the display attribute can also be overridden on the same output operation (as if the OVRATR keyword were also in effect).

When the OVRATR keyword is specified at the field level, it is valid only with the following types of fields:

- Input-only
- Output-only
- Output/input
- Constant

When the OVRATR keyword is specified at the record level, it applies to each of the following types of fields:

- Input-only
- Output-only
- Output/input
- Constant

Option indicators are valid for this keyword.

For a discussion and example of how to use the OVRATR keyword, see "PUTOVR (Put with Explicit Override)" on page 3-178.

OVRDTA (Override Data)

Use this field- or record-level keyword with the PUTOVR keyword to override the existing data contents of a field or record already on the display.

The OVRDTA keyword can be used with the OVRATR keyword on the same field or record.

When OVRDTA is specified at both the record and field level, the field-level specification will be used for that field.

This keyword has no parameters.

OVRDTA is required if the DFT keyword is specified for output-only or output/input fields.

When OVRDTA is specified at the field level, it is valid only with the following types of fields:

- Output-only
- Output/input
- Message

When the OVRDTA keyword is specified at the record level, it applies to each of the following types of fields:

- Output-only
- Output/input
- Message

Option indicators are valid for this keyword.

For a discussion and example of how to use the OVRDTA keyword, see "PUTOVR (Put with Explicit Override)" on page 3-178.

PAGEDOWN/PAGEUP (Page down/Page up)

Use these file- or record-level keywords to specify that you want to use your program to handle any situation where the work station user has pressed the pagedown or pageup keys and the OS/400 program cannot page through the display. If this situation occurs and you have not specified this keyword (whichever one is appropriate), the OS/400 program sends an error message indicating that the key is not valid at that time.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the PAGEDOWN/PAGEUP keywords in files that are used in the System/36 environment.

The format for each of these keywords is:

```
PAGEDOWN[(response-indicator ['text'])]
PAGEUP[(response-indicator ['text'])]
```

You can specify a response indicator with these keywords. If you do, and the appropriate Page key is pressed, the OS/400 program sets on the specified response indicator within the input record and returns control to your program after it processes the input data. If you do not specify a response indicator and the specified Page key is pressed, the OS/400 program performs normal input record processing.

The optional text is included on the computer printout created at program compilation to explain the intended use of the indicator. This text functions only as a comment in the file or program. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

These keys cause data to be returned from the display device to your program (similar to command function (CF) and Enter keys).

The ROLLUP keyword cannot be specified with PAGEDOWN. The ROLLDOWN keyword cannot be specified with PAGEUP.

Note: PAGEDOWN is the same as ROLLUP; PAGEUP is the same as ROLLDOWN.

If the OS/400 program is performing the page function for subfiles (SFLSIZ value does not equal SFLPAG value), you do not need to specify these keywords. For a description of what happens when PAGEDOWN and PAGEUP are specified for a subfile, see "SFLROLVAL (Subfile Roll Value)" on page 3-220.

Option indicators are valid for these keywords.

Figure 3-145 shows how to specify the PAGEDOWN and PAGEUP keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A N64                                  PAGEUP(52 'Page Up')
     A                                      PAGEDOWN(61)
     A
```

Figure 3-145. Specifying the PAGEDOWN and PAGEUP Keywords

PASSRCD (Passed Record)

Use this file-level keyword to specify the record format to be used by the OS/400 program when unformatted data is passed to your program by another program. The passed data is processed *only* if your program's first request after file open is an input operation without a record format name. The data must be processed according to this record format.

The format of the keyword is:

```
PASSRCD(record-format-name)
```

The record-format-name is a required parameter value for this keyword and must exist in the file. The following keywords cannot be specified on the record format:

ALWROL
CLRL
SLNO

Option indicators are not valid for this keyword.

Figure 3-146 shows how to specify the PASSRCD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      PASSRCD(RECKEEP)
00020A          R RECORD
     A
```

Figure 3-146. Specifying the PASSRCD Keyword

PRINT (Print)

Use this file- or record-level keyword to specify that the work station user can press the Print key to print the current display.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the PRINT keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
PRINT[(response-indicator ['text']) | (*PGM) |
  ([library-name/]printer-file-name)]
```

The following four examples illustrate the four ways you can specify the PRINT keyword:

`PRINT`	The OS/400 program spools the output to printer file QSYSPRT unless you specify another printer file for the work station on the PRTFILE parameter on the CRTDEVDSP or CHGDEVDSP commands. See "PRINT Keyword without Parameter Values."
`PRINT(01 'User presses Print key')`	Your program is given control and decides what to do (for example, produce formatted printer output). The response indicator is set on. No data is sent from the device.
`PRINT(*PGM)`	Control returns to your program when the Print key is pressed.
`PRINT(LIB1/PRINTFILE1)`	The OS/400 program spools the output to the specified printer file (which can be defined through DDS or on the PRTFILE parameter on the CRTDEVDSP or CHGDEVDSP commands). An Override with Printer File (OVRPRTF) command, if in effect before the printer file is opened (when the Print key is pressed), can change the printer device name.

Further considerations of the ways to specify the PRINT keyword are discussed in the following sections.

If you specify the PRINT keyword in any form, the work station user can print a display containing the message help. In this case, the print operation is performed as if the PRINT keyword were specified with no parameters.

PRINT Keyword without Parameter Values

For the Local Work Station: The OS/400 program spools the output to the specified printer file (which can be defined through DDS or on the PRTFILE parameter on the CRTDEVDSP or CHGDEVDSP commands).

Nondisplay fields appear as blanks. Duplicated characters entered by pressing the Dup key appear as asterisks (*). Display attributes appear as blanks. If the print function cannot be performed successfully, the OS/400 program attempts to complete the print function using the printer file specified on the PRTFILE parameter on the CRTDEVDSP or the CHGDEVDSP command that is used to describe the display device to the system.

For work station printers attached through the work station controller, a message indicating that there is a problem is sent to the work station user requesting the print function. The work station user can make the printer ready or press the Reset key. To cancel a print request before it is complete, the work station user can press and hold the Shift key, then press the Print key.

Note: After the current display is printed, the paper is advanced twice the number of lines as in the current display size (48 lines for a 24 x 80 display, and 54 lines for a 27 x 132 display).

For the Remote Work Station: The OS/400 program attempts to print the display image on the associated work station printer without sending the data through the system. The associated work station printer is the printer device specified on the PRINTER parameter of the CRTDEVDSP or CHGDEVDSP command that is used to describe the local display device to the system.

If the printer is not ready when the Print key is pressed, no specific message is sent to the user. The work station requesting the print function remains inoperable until the printer is made ready, or until the print request is canceled (by using the shifted Print key).

Note: After the current display is printed, the paper is advanced the same number of lines as in the current display size (24 lines for a 24 x 80 display, and 27 lines for a 27 x 132 display).

Option indicators are valid for this keyword.

Figure 3-147 shows how to specify the PRINT keyword with no parameter values.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      PRINT
00020A          R RECORD1
     A
```

Figure 3-147. Specifying the PRINT Keyword

PRINT Keyword with Response Indicator or *PGM Special Value

If you specify the PRINT keyword with a response indicator, the OS/400 program returns control to your program with the specified response indicator set on. No data is received from the device. The keyboard is locked until your program sends another output operation to the display file. There is no difference in the print function between local and remote work stations. If you specify *PGM, the OS/400 program returns control to your program. The only difference between these two forms is the response indicator; all other processing is the same.

The optional text for the response indicator form is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text functions only as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

Figure 3-148 shows how to specify the PRINT keyword with a response indicator.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      PRINT(01 'User presses Print key')
00020A          R RECORD1
     A
```

Figure 3-148. Specifying the PRINT Keyword with Response Indicator

Figure 3-149 on page 3-176 shows how to specify the PRINT keyword with the *PGM special value.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        PRINT(*PGM)
00020A          R RECORD1
     A
```

*Figure 3-149. Specifying the PRINT Keyword with the *PGM Special Value*

PRINT Keyword with a Specified Printer File

The OS/400 program reads the display buffer and prints the display image using the specified printer file. The printer file that you specify as a parameter value for this keyword can be either an externally described or a program-described file. It also can be either spooled or nonspooled. If you specify an externally described printer file, it must contain a record format with the same name as the file.

The printer file must exist and be authorized to the user of the display when the Print key is pressed. This also applies to the library name if it is specified. If the OS/400 program is unable to perform the print function on the specified printer file, it attempts to use the printer file specified on the PRTFILE parameter of the CRTDEVDSP or the CHGDEVDSP command. SPOOL(*YES) should be specified on the CRTPRTF or CHGPRTF command to prevent the keyboard from locking.

If you do not specify the library name, the current library list at program run time is used.

See "OPENPRT (Open Printer File)" on page 3-168 for details of when the printer file is opened and closed.

Figure 3-150 shows how to specify that the display is to be directed to printer file, LIB1/PRINTFILE1.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        PRINT(LIB1/PRINTFILE1)
00020A          R RECORD1
     A
```

Figure 3-150. Specifying the PRINT Keyword with Printer File Specified

---

PROTECT (Protect)

Use this record-level keyword with the OVERLAY keyword to specify that when the record you are defining is displayed, all input-capable fields already on the display are to be changed to output-only fields. This protects them from input typing. This keyword does not affect the record format in which it is specified. The data contents of the affected fields are not changed, but your program cannot read them unless it first displays the record formats again in which the input-capable fields are specified.

To protect a single field from input keying, see the DSPATR(PR) keyword.

This keyword has no parameters.

The OVERLAY keyword must be specified in the record format in which PROTECT is specified. Also, either the OVERLAY keyword or the CLRL keyword must be in effect for PROTECT to be in effect.

You can use PROTECT to protect input-capable fields in other records only on the first output operation for which you have selected PUTOVR. On subsequent output operations, PROTECT is in effect only if the PUTOVR keyword is not in effect.

If the ERASEINP and PROTECT keywords are both in effect for an output operation, the OS/400 program first erases the input-capable fields specified on the ERASEINP parameter value, then protects all input-capable fields on the display from input typing.

A warning message appears at file creation time if the PROTECT keyword is specified on a record with the DSPMOD keyword. At run time, the PROTECT keyword is ignored when the display mode changes.

Option indicators are valid for this keyword.

Figure 3-151 shows how to specify the PROTECT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A            FLD1           5   I  5  3
00030A
00040A          R RECORD2                    OVERLAY
00050A  32                                   PROTECT
00060A            FLDA          10   I  6  3
     A
```

Figure 3-151. Specifying the PROTECT Keyword

In Figure 3-151, RECORD1 has an input-capable field that has been displayed and read and that should be left on the display while RECORD2 is displayed and read. To prevent further entries in FLD1 in RECORD1, send an output operation to RECORD2 with PROTECT in effect. When this is done, FLDA is not protected, but FLD1 is protected.

PULLDOWN (Pull-Down Menu)

Use this record-level keyword to define a record as a pull-down menu for a menu bar. When this record is written, it is saved by the system and it is displayed later as a pull-down menu for a menu-bar choice.

The format of the keyword is:

```
PULLDOWN[(*SLTIND | *NOSLTIND)]
```

The parameter is optional and specifies whether the selection indicators (such as radio buttons) for a selection field in the pull-down menu are displayed. *SLTIND specifies that the selection indicators should be displayed. *NOSLTIND specifies that the selection indicators should not be displayed. The default is *SLTIND.

A record containing the PULLDOWN keyword is considered a window record, although the WINDOW keyword cannot be used. The system calculates the dimensions of the pull-down window and generates the border.

The following keywords cannot be specified on a record with the PULLDOWN keyword:

ALARM	FRCDTA	OVRDTA
ALTNAME	HLPCLR	PUTOVR
ALWGPH	HLPSEQ	PUTRETAIN
ALWROL	INVITE	RTNDTA
ASSUME	INZRCD	SFL
CLEAR	MDTOFF	SLNO
CLRL	MNUBAR	USRDFN
ERASE	OVERLAY	WINDOW
ERASEINP	OVRATR	

Option indicators are not valid for this keyword.

Figure 3-152 shows how to specify the PULLDOWN keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8

     A
     A          R MENUBAR                   MNUBAR
     A            MNUFLD         2Y 0B  1  2
     A                                      MNUBARCHC(1 PULLFILE 'File       ')
     A                                      :
     A                                      :
     A          R PULLFILE                  PULLDOWN
     A            :
     A            :
     A
```

Figure 3-152. Specifying the PULLDOWN Keyword

In this example, record PULLFILE is defined as a pull-down menu for a menu-bar choice. When record PULLFILE is written, the system saves it and displays it when it is selected from the menu bar. When the system displays the PULLFILE record, it calculates the dimensions needed for the pull-down window based on the contents of the PULLFILE record, and generates the pull-down border accordingly.

PUTOVR (Put with Explicit Override)

Use this record-level keyword to permit the override of either display attributes or data contents (or both) of specific fields within a record displayed on a work station device. By using PUTOVR, you can reduce the amount of data sent to the display device.

See the *Guide to Programming Displays* for information on how to use PUTOVR in files that are used in the System/36 environment.

This keyword has no parameters.

If you use the PUTOVR keyword and subfiles, certain restrictions apply. See the *Guide to Programming Displays* for more information about these restrictions.

When selected fields in a record that has already been displayed are to be changed, an output or an output/input operation sent to the record with the PUTOVR, OVRDTA, and OVRATR keywords in effect changes only the fields for which the OVRDTA or OVRATR keyword is in effect. The OVRDTA keyword permits a change in the data contents of the field and the OVRATR keyword permits a change in the display attributes of the field. The way in which fields are to be changed is controlled by setting option indicators.

The following conditions cause the Put-Override keywords to be ignored and no error to occur:

- PUTOVR is not in effect at the time of the output operation.
- Neither the OVRDTA nor OVRATR keyword is in effect at the time of the output operation.
- The record format is not already on the display.

The PUTOVR and OVRDTA keywords must be specified when DFT is specified for a named output-capable field. When the PUTOVR and OVRDTA keywords are both in effect for a field, the default value specified with the DFT keyword is displayed only on the first display of the field. On subsequent displays with the PUTOVR and OVRDTA keywords in effect, the program value is displayed.

If a field is not displayed on the first output operation to a record format, certain restrictions could apply. These restrictions apply when, on a subsequent output operation, the field is selected for display and the put-override keywords are also in effect:

- For output-only fields for which the OVRDTA or OVRATR keyword is selected, the OS/400 program does not send an ending attribute character. Any display attributes (such as reverse image) are continued across the display until the beginning attribute character of the next field on the display. You should display output-only fields on the first output operation (perhaps with the DSPATR(ND) keyword so they cannot be seen) in order to provide an ending attribute character for later overrides.
- For input-capable or message fields for which the OVRDTA or OVRATR keyword is selected, the OS/400 program sends an ending attribute character and no restriction applies.

The PUTRETAIN keyword and the PUTOVR keyword cannot be specified on the same record format.

The OVRDTA keyword is permitted only with output-only, output/input, program-to-system, or message fields (usage O, B, P, or M, respectively).

The OVRATR keyword is permitted only with output-only, input-only, or output/input fields (usage O, I, or B, respectively).

If you specify PUTOVR, you should also specify RSTDSP(*YES) on the Create Display File (CRTDSPF) or Change Display File (CHGDSPF) command. Otherwise, data on the display can be lost if the file is suspended.

A warning message is sent at file creation time if the PUTOVR keyword is specified on a record with the DSPMOD keyword. At run time, the PUTOVR keyword is ignored when the display mode changes.

The OVRATR keyword can be used only to override the following display attributes:

CHECK(ER)
CHECK(ME)
DUP
DSPATR (all except OID and SP)

An output operation with the OVRDTA keyword in effect does not need to have the OVRATR keyword in effect to override display attributes, as well as data contents, of the field or fields being overridden.

Option indicators are valid for the PUTOVR, OVRATR, and OVRDTA keywords.

Figure 3-153 shows how to specify the PUTOVR, OVRATR, and OVRDTA keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R INVRCD                    PUTOVR
00020A            FLD1                   1 26'INVENTORY REMAINING IN WAREHOUSE 1'
00030A*
00040A                                   3  2'Remaining on hand:'
00050A                                      OVRATR
00060A  11                                  DSPATR(HI)
00070A*
00080A            INVBAL          5Y 0     +2
00090A  12                                  OVRDTA
00100A*
00110A                                     +2'Low on stock' OVRATR
00120A N70                                  DSPATR(ND)
00130A  70                                  DSPATR(HI)
00140A*
00150A            SUPPPL        20   B  5  2DFT('INTERNAL')
00160A  13                                  OVRDTA
00170A*
00180A            ACCT          20      6  2OVRDTA
00190A                                      DSPATR(HI)
00200A  14                                  DSPATR(RI)
     A
```

Figure 3-153. Specifying the PUTOVR, OVRATR and OVRDTA Keywords

An initial output operation generates a full display of information; on a second output operation, PUTOVR is in effect and the program can set option indicators to make the following changes to the display:

- If option indicator 11 is set on, the *Remaining on hand:* constant field would be changed to a highlighted field. To reset the display attribute to normal, display the record format again with option indicator 11 off.
- If option indicator 12 is set on, the program can change the displayed value of the field INVBAL.
- If option indicator 70 is set off, the *Low on stock* constant field is a nondisplay field. If option indicator 70 is set on, the field is changed to a highlighted field.
- If option indicator 13 is set on, the program can set the value of the field SUPPL to override the default value (INTERNAL). The first displayed value is always INTERNAL; to display the value INTERNAL again after changing it to something else, the program must set the value of the field to INTERNAL before displaying it again.
- If option indicator 14 is set on, the display attribute of the field ACCT is changed from highlight to highlight and reverse image at the same time that new data is sent to the field. If option indicator 14 is set off, the display attribute is changed back to highlight. New data is sent to the display on each output operation.

PUTRETAIN (Put-Retain)

Use this record- or field-level keyword with the OVERLAY keyword so that the OS/400 program does not delete data already on the display when displaying a record again. The PUTOVR keyword has a function similar to, but more effective than, PUTRETAIN.

This keyword has no parameters.

To understand what effect this keyword has on output operations, consider the following sequence of steps:

1. Your program sends an output operation to RECORD1, displaying RECORD1. PUTRETAIN, if in effect, is ignored. Any data in the record area for RECORD1 is deleted before RECORD1 is displayed.
2. At some later time, with RECORD1 still on the display, your program sends a second output operation to RECORD1. Two conditions can occur:
   - If the PUTRETAIN keyword is not in effect, the OS/400 program first deletes the record area for RECORD1, then displays RECORD1. Fields selected for display at this time are displayed with new data contents and new display attributes, which can be the same as before. The record area includes every line on which a field or part of a field for RECORD1 appears.
   - If PUTRETAIN is in effect, the OS/400 program does not delete the record area for RECORD1. The data contents of selected fields are not changed. However, the display attributes for selected fields are sent to the display and can be changed (by selecting which DSPATR keyword is in effect for this output operation). Fields not selected for display are written over character-by-character by fields selected for display. (For more information, see "When Fields Are Selected by Option Indicators" on page 3-182.)

Note: When using the field-level PUTRETAIN keyword, the entire record area is deleted if none of the fields in the record has PUTRETAIN optioned on. If you specify at least one field with unoptioned field-level PUTRETAIN keyword, this ensures that the record area is not deleted.

If you specify the PUTRETAIN keyword, you should also specify RSTDSP(*YES) on the Create Display File (CRTDSPF) or Change Display File (CHGDSPF) command. Otherwise, data on the display can be lost if the file is suspended.

Option indicators are valid for this keyword.

Conditions Affecting the PUTRETAIN Keyword

PUTRETAIN applies only to the record format for which it is specified, and then only if the record is already displayed. If the record on which PUTRETAIN is specified is not on the display, PUTRETAIN is ignored.

If you specify this keyword at the record level, the keyword applies to all fields in the record format that are selected for display.

This keyword can be specified for more than one field of a record format, but only once per field. This keyword can be specified at the record level and at the field level within the same record format.

PUTRETAIN cannot be specified with the PUTOVR keyword.

A warning message appears at file creation time if the PUTRETAIN keyword is specified on a record with the DSPMOD keyword. At run time, the PUTRETAIN keyword is ignored when the display mode changes.

The OVERLAY keyword must be specified whenever PUTRETAIN is specified.

If the OVERLAY keyword is not in effect, PUTRETAIN is ignored and the entire display deleted before the record is displayed.

When Fields Are Selected by Option Indicators

When PUTRETAIN is in effect on an output operation involving field selection, fields in the record format that are not selected for redisplay are not deleted; they can be partially or completely rewritten by newly selected fields.

If PUTRETAIN is in effect only for a newly selected field (specified at the field level), only the beginning attribute character of the field is sent to the display; the ending attribute character is not sent. For fields without PUTRETAIN in the same record format, the OS/400 program sends the display attribute and the data. If PUTRETAIN is in effect for the whole record (specified at the record level), only the beginning and ending attribute characters are sent to the display. Thus, the display attribute of a field can be reset to normal if the field immediately preceding this field is selected and this field is not selected.

For example, assume that the DSPATR(UL) keyword is in effect for two consecutive fields with overlapping attribute characters. If on an output operation with PUTRETAIN in effect, the first of these fields is selected and the second field is not selected, the display attribute of the second field is reset to the normal display attribute. This is because the OS/400 program sends the first field to the display with beginning and ending attribute characters, and its ending attribute character overrides the beginning attribute character of the second field.

Figure 3-154 shows how to specify the PUTRETAIN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00101A          R CUST
00102A                                      PUTRETAIN OVERLAY
     A
```

Figure 3-154. Specifying the PUTRETAIN Keyword

RANGE (Range)

Use this field-level keyword for input-capable fields to specify that the OS/400 program is to perform validity checking on the data that the work station user types into the field. The data typed in must be greater than or equal to the lower value, and less than or equal to the higher value. Note that the OS/400 program performs this checking only if the field is changed by the work station user or if its changed data tag (MDT) is set on using DSPATR(MDT).

Note: Refer to the CHKMSGID keyword for information on defining user-specified error messages.

The format of the keyword is:

```
RANGE(low-value high-value)
```

When the field is a character field, the parameter values must be enclosed in apostrophes. When the field is numeric, apostrophes must not be specified.

You cannot specify RANGE on a floating-point field (F in position 35).

Option indicators are not valid for this keyword.

Figure 3-155 shows how to specify the RANGE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A* Character fields
00030A            FIELD1         1   I  2  2RANGE('B' 'F')
00040A            FIELD2         1   I  3  2RANGE('2' '5')
00050A* Numeric fields
00070A            FIELD3         1  0I  4  2RANGE(2 5)
00080A            FIELD4         4  0B  5  2RANGE(1 1500)
00090A            FIELD5         7  2B  6  2RANGE(100 99999.99)
00100A            FIELD6         3  0B  7  2RANGE(-100 -50)
00110A            FIELD7         3  2I  8  2RANGE(.50 1.00)
00120A            FIELD8         3  2I  9  2RANGE(.5 1)
00130A            FIELD9         5Y 2I 10  2RANGE(.01 999.99)
     A
```

Figure 3-155. Specifying the RANGE Keyword

FIELD7 and FIELD8 have equivalent RANGE parameter values. The reason is that for numeric fields, decimal alignment is based on the number of decimal positions specified in positions 36 through 37. For FIELD7 and FIELD8, the low value is 0.50 and the high value is 1.00.

Data entered into a numeric field is aligned on the decimal positions specified (in positions 36 through 37), and leading and trailing blanks are filled with zeros. For example, if 1.2 is typed into FIELD9, 00120 is returned to your program. If 100 is typed into FIELD9, 10000 is returned to your program.

REF (Reference)

Use this file-level keyword to specify the name of a file from which field descriptions are to be retrieved or when you want to duplicate descriptive information from several fields in a previously described record format. You can code the file name once here rather than on REFFLD keywords with each of the field descriptions that refer to the file. To refer to more than one file, use the REFFLD keyword. (REF can be specified only once.)

The format of the keyword is:

```
REF([library-name/]database-file-name [record-format-name])
```

If there is more than one record format in the referenced file, specify a record format name as a parameter value for this keyword to tell the OS/400 program which one to use unless the record formats should be searched sequentially.

The database-file-name is a required parameter for this keyword. The library-name and the record-format-name are optional.

If you do not specify the library-name, the current library list (*LIBL) at file creation time is used. If the record-format-name is not specified, each record format is searched in order (as they are specified). The first occurrence of the field name is used. See Appendix A, "When to Specify REF and REFFLD," for the search sequences determined by your choice of REF and REFFLD keywords.

You can specify a Distributed Data Management (DDM) file on this keyword.

When using a DDM file, the database-file-name and library-name are the DDM file and library names on the source system. The record-format-name is the record format name in the remote file on the target system.

Option indicators are not valid for this keyword.

Figure 3-156 and Figure 3-157 show how to specify the REF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      REF(FILE1)
00020A          R RECORD
00030A            FLD1      R           2  2
     A
```

Figure 3-156. Specifying the REF Keyword (Example 1)

In Figure 3-156, FLD1 has the same attributes as the first (or only) FLD1 in FILE1.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      REF(LIB/FILE1 RECORD2)
00020A          R RECORD
00030A            FLD1      R           2  2
     A
```

Figure 3-157. Specifying the REF Keyword (Example 2)

In Figure 3-157, FLD1 has the same attributes as FLD1 in RECORD2 in FILE1 in LIB1.

REFFLD (Referenced Field)

Use this field-level keyword when referring to a field under one of these conditions:

- The name of the referenced field is different from the name in positions 19 through 28.
- The name of the referenced field is the same as the name in positions 19 through 28, but the record format, file, or library of the referenced field is different from that specified with the REF keyword.
- The referenced field occurs in the same DDS source file as the referencing field.

The format of the keyword is:

```
REFFLD([record-format-name/]referenced-field-name
[{*SRC | [library-name/]database-file-name}])
```

The referenced-field-name is required even if it is the same as the referencing field. Use the record format name when the referenced file contains more than one record format. Use *SRC (rather than the database-file-name) when the referenced field name is in the same DDS source file as the referencing field. *SRC is the default value when the database-file-name and library-name are not specified and the REF keyword is not specified at the file level.

Note: When you refer to a field in the same DDS source file, the field you are referring to must precede the field you are defining.

Specify the database-file-name (with its library-name, if necessary) to search a particular database file.

If, in the same DDS source file, you specify the REF keyword at the file level and REFFLD at the field level, the particular search sequence depends on both the REF and REFFLD keywords. For more information, see Appendix A, "When to Specify REF and REFFLD."

You must specify an R in position 29. In some cases, some keywords specified with the field in the database file are not included in the display file. For more information, see "Reference (Position 29)" on page 3-8.

You can specify a Distributed Data Management (DDM) file on this keyword.

When using a DDM file, the database-file-name and library-name are the DDM file and library names on the source system. The referenced-field-name and the record-format-name are the field name and the record format name in the remote file on the target system.

Option indicators are not valid for this keyword.

Figure 3-158 shows how to specify the REFFLD keyword. See the example in Appendix A, "When to Specify REF and REFFLD," for explanations of the various specifications.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R FMAT1
00020A            ITEM           5      3  1
00030A            ITEM1     R           5  2REFFLD(ITEM)
00040A            ITEM2     R           5 12REFFLD(FMAT1/ITEM)
00050A            ITEM3     R           5 22REFFLD(ITEM FILEX)
00060A            ITEM4     R           5 32REFFLD(ITEM LIBY/FILEX)
00070A            ITEM5     R           5 42REFFLD(FMAT1/ITEM LIBY/FILEX)
00080A            ITEM6     R           5 52REFFLD(ITEM *SRC)
     A
```

Figure 3-158. Specifying the REFFLD Keyword

RETKEY (Retain Function Keys) and RETCMDKEY (Retain Command Keys)

Use these record-level keywords to indicate that function keys or command function/attention keys, which were enabled on a display, should be retained when the record you are defining is displayed.

These keywords have no parameters.

Refer to Appendix F, "System/36 Environment Considerations," for information on how to specify the RETKEY and RETCMDKEY keywords.

RETLCKSTS (Retain Lock Status)

Use this record-level keyword to specify that the system unlock of the keyboard on the next input operation should not occur. This keyword prevents the loss of data when an input operation is started and data is already being transmitted from the keyboard.

Note: Normally an input operation explicitly unlocks the keyboard, even if it is already unlocked. Any data being transmitted from the keyboard at the time of the unlock can be lost.

This keyword has no parameters.

Note: Use this keyword only when the keyboard is already unlocked. Use of this keyword when the keyboard is locked can prevent input from the keyboard, because the unlock does not occur on the input operation. The work station remains in an input-inhibited state.

Option indicators are valid for this keyword.

Figure 3-159 shows how to specify the RETLCKSTS keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A
00020A          R REC1                      INVITE
00030A  10                                  RETLCKSTS
     A
```

Figure 3-159. Specifying the RETLCKSTS Keyword

If indicator 10 is on when record REC1 is put to the display, the keyboard is not explicitly unlocked by the system when the display device is invited.

RMVWDW (Remove Window)

Use this record-level keyword to remove all existing windows on the display before this record is displayed.

This keyword has no parameters.

When the RMVWDW keyword is specified, a WINDOW keyword must be specified on the same record format. The RMVWDW keyword functions only when the WINDOW keyword defines a window. The RMVWDW keyword does not function if the WINDOW keyword specified a record format name.

Option indicators are valid for this keyword.

Figure 3-160 shows how to specify the RMVWDW keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R WINDOW1                    WINDOW(6 15 9 30)
     A            FIELD1         5A  B  2  2
     A            FIELD2        20A  B  8  5
     A          R WINDOW2                     WINDOW(&LINE &POS 9 30)
     A  01                                    RMVWDW
     A            FIELD3         5A  B  2  2
     A            FIELD4        20A  B  8  5
     A            LINE           2S  P
     A            POS            3S  P
     A
```

Figure 3-160. Specifying the RMVWDW Keyword

WINDOW1 is already on the display. If indicator 01 is set on and WINDOW2 is written to the display, WINDOW1 is removed before WINDOW2 is displayed. If indicator 01 is off when WINDOW2 is written to the display, WINDOW1 remains on the display when WINDOW2 is displayed.

ROLLUP/ROLLDOWN (Roll up/Roll down)

Use these file- or record-level keywords to specify that you want to use your program to handle any situation where the work station user has pressed the rollup or rolldown keys and the OS/400 program cannot move the text lines on the display. If this situation occurs and you have not specified this keyword (whichever one is appropriate), the OS/400 program sends an error message indicating that the key is not valid at that time.

Refer to Appendix F, "System/36 Environment Considerations," for special considerations when specifying the ROLLUP/ROLLDOWN keywords in files that are used in the System/36 environment.

The format for each of these keywords is:

```
ROLLUP[(response-indicator ['text'])]
ROLLDOWN[(response-indicator ['text'])]
```

You can specify a response indicator with these keywords. If you do, and the appropriate paging key is pressed, the OS/400 program sets on the specified response indicator within the input record and returns control to your program after it processes the input data. If you do not specify a response indicator and the specified paging key is pressed, the OS/400 program performs normal input record processing.

The optional text is included on the computer printout created at program compilation to explain the intended use of the indicator. This text functions only as a comment in the file or program. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

These keys cause data to be returned from the display device to your program (similar to command function (CF) and Enter keys).

The PAGEDOWN keyword cannot be specified with ROLLUP. The PAGEUP keyword cannot be specified with ROLLDOWN.

Note: The ROLLUP keyword is the same as the PAGEDOWN keyword. The ROLLDOWN keyword is the same as the PAGEUP keyword. Roll is the same as page.

If the OS/400 program is performing the paging function for subfiles (SFLSIZ value does not equal SFLPAG value), you do not need to specify these keywords. For a description of what happens when ROLLUP and ROLLDOWN are specified for a subfile, see "SFLROLVAL (Subfile Roll Value)" on page 3-220.

Option indicators are valid for these keywords.

Figure 3-161 shows how to specify the ROLLUP and ROLLDOWN keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A N64                                ROLLDOWN(52 'Roll Down')
     A                                    ROLLUP(61)
     A
```

Figure 3-161. Specifying the ROLLUP and ROLLDOWN Keywords

RTNCSRLOC (Return Cursor Location)

Use this record-level keyword to return the location of the cursor to an application program.

The format of the keyword is:

```
RTNCSRLOC(&cursor-record &cursor-field {&cursor-position})
```

The cursor-record parameter is required. It specifies the name of a hidden field that, on input, will contain the name of the record on which the cursor is located. The field must be defined in the record format as a character (A in position 35) field of length 10, with usage H (hidden). If the cursor is not in a record area on input, the cursor-record field will contain blanks.

The cursor-field parameter is required. It specifies the name of a hidden field that, on input, will contain the name of the field on which the cursor is located. The field must be defined in the record format as a character (A in position 35) field of length 10, with usage H (hidden). If the cursor is not located on a field on input, the cursor-field field will contain blanks.

The optional cursor-position parameter specifies the name of a hidden field that, on input, will contain the relative position of the cursor within the field on which it is located. The field must be defined in the record format as a signed numeric (S in position 35) field of length 4, with 0 decimal positions and usage H (hidden). If the cursor is in the first position of the field, the cursor-position field will contain the value 1. If the cursor is in the *i*th position, the cursor-position field will contain the value *i*. If the cursor is not located on a field, the cursor-position field will contain the value 0.

All three fields specified on the RTNCSRLOC keyword will contain values on input if the cursor is outside the area of the record that contains the RTNCSRLOC keyword. The fields also contain values on input if the cursor is located in a subfile. The cursor-record field will contain a value if the cursor is located any-

where inside the subfile. The cursor-field and cursor-position fields will contain values if the cursor is located on a field within the subfile.

Option indicators are not valid for this keyword.

Figure 3-162 shows how to specify the RTNCSRLOC keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R REC01                       RTNCSRLOC(&RCD &FLD &POS)
     A            FLD           10A  H
     A            RCD           10A  H
     A            POS            4S 0H
     A            FLD1A          2A  I  3  2
     A  10        FLD2A          6A  O  3 18
     A N10        FLD3A         10A  O  3 18
     A*
     A          R REC02                         OVERLAY
     A            FLD1A          2A  I  5  2
     A            FLD2A         10A  O  5  5
     A            FLD3A          6A  O  5 18
     A
```

Figure 3-162. Specifying the RTNCSRLOC Keyword

Both REC01 and REC02 are displayed on the screen and option indicator 10 is off.

The following table shows the values returned when the cursor is at the specified positions.

Row	Column	Cursor Record	Cursor Field	Cursor Position
3	2	REC01	FLD1A	1
3	19	REC01	FLD3A[1]	2
3	25	REC01	FLD3A	8
3	40	REC01	blanks	0
4	40	blanks	blanks	0
5	5	REC02	FLD2A	1
5	40	REC02	blanks	0

[1] If Option indicator 10 were on, FLD2A would be returned when the cursor is at row 3 column 19.

RTNDTA (Return Data)

Use this record-level keyword to specify that, when your program sends an input operation to this record format, the OS/400 program is to return the same data that was returned on the previous input operation sent to this record format. The RTNDTA keyword is ignored if the record format has not already been read. When the RTNDTA keyword is in effect, your program can reread data on the display without requiring the OS/400 program to actually pass data from the display device to your program.

This keyword has no parameters.

The RTNDTA keyword is ignored in the following situations:

- On the input portion of an output/input operation (Put-Get operation)
- On an input operation that is preceded by an output operation to the same record format

The RTNDTA keyword has effect only on an input operation sent to the same record format without an intervening output operation to that record format.

You can use RTNDTA as follows:

- Use RTNDTA to allow a main program to read a record format that is changed by a work station user. The data read tells the main program which subprogram to call. The subprogram sends an input operation to the same record format, with RTNDTA in effect, to read the same data. This procedure can substitute for passing parameters to subprograms.

  Note: SHARE(*YES) must be specified for both display files.

- Use RTNDTA to allow an RPG III program to perform file maintenance with less locking of records in the database. For instance, the program reads a database record and displays the record at the display device. The work station user reviews the record, makes any required changes, and presses the Enter key. While the work station user is making changes, the database record, if locked, is unavailable to other programs. Hence it is recommended to leave the database record unlocked. However, when the program reads the record from the display and updates the database record, the database record overlays the internal representation of the display record in the program. Instead of preventing the overlay by using different field names for the display record and the database record, the program rereads the display file. With RTNDTA specified, the program retrieves the display record again and can then finish updating the database.

If the UNLOCK keyword is specified, the RTNDTA keyword cannot be specified.

Option indicators are not valid for this keyword.

Figure 3-163 shows how to specify the RTNDTA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                     RTNDTA
00020A            FLD1           5   I  2  2
00030A            FLD2           5   B  3  2
     A
```

Figure 3-163. Specifying the RTNDTA Keyword

SETOF (Set Off)

Use this record-level keyword to specify that when an input operation sent to this record format is completed, the specified response indicator is to be set off.

Refer to Appendix F, "System/36 Environment Considerations," for information on how to specify the SETOF keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
SETOF(response-indicator ['text'])
```

The optional text is included on the computer printout created at program compilation to explain the intended use of the indicator. This text functions only as a comment in the file or program. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program listing.

This keyword can be used to cause an option indicator that is on for an output operation to be returned in the off condition when the next input operation to the record is completed. (If no input operation is performed, the response indicator remains unchanged.) Your program does not have to turn off the indicator.

SETOF is equivalent to the SETOFF keyword.

Any indicator is valid for this keyword. It does not have to be previously defined as an option or a response indicator. The indicator becomes a response indicator when you specify SETOF.

If the indicator used with the SETOF keyword is also used with another keyword, such as CHANGE, the on/off status of the indicator is controlled by the other keyword.

Option indicators are not valid for this keyword.

Figure 3-164 shows how to specify the SETOF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CUSMST                    SETOF(63 'On=display MSG2000 +
00020A                                         CONSOLEMSG')
00030A            QRYORD         3  0I  5  3
00040A  63                                  ERRMSGID(MSG2000 CONSOLEMSG)
     A
```

Figure 3-164. Specifying the SETOF Keyword

A MSG2000 message is displayed on the message line when the program sends an output operation to CUSMST with indicator 63 set on. On the next input operation to CUSMST, the SETOF keyword sets off indicator 63. (Indicator 63 is used as both an option and a response indicator.)

SETOFF (Set Off)

The SETOFF keyword is equivalent to the SETOF keyword.

The format of the keyword is:

```
SETOFF(response-indicator ['text'])
```

The SETOF keyword is preferred. See "SETOF (Set Off)" on page 3-190 for a description of this keyword.

SFL (Subfile)

Use this record-level keyword to specify that this record format is to be a subfile record format. This record format (including its related field descriptions) must immediately precede the subfile control record format (identified by the SFLCTL keyword).

This keyword has no parameters.

At least one displayable field must be specified in the subfile record format, unless the subfile is a message subfile (see "SFLMSGRCD (Subfile Message Record)" on page 3-212). The locations specified for fields in this record format are the locations on the display where the first subfile record in any one page of the subfile is displayed. The remaining part of the page of records is displayed below the first record.

The number of records in a page is determined by the parameter value specified for the SFLPAG keyword. For an explanation of how many display lines are occupied by a subfile, see "SFLPAG (Subfile Page)" on page 3-214.

Displayable fields specified on the subfile control record format can be displayed at the same time as subfile records. However, fields specified in the subfile control record format cannot overlap fields specified in the subfile record format, even if they are specified with option indicators.

Overlap errors can occur if the first field of either the subfile record or the subfile control record starts in position 1. A field starting in position 1 has a beginning attribute byte on the previous line. Therefore, the previous line is also part of the record format.

The number of subfiles (each having one SFL and one SFLCTL keyword specified) that can be specified in a display file is limited only by the number of record formats permitted in a display file (1024 record formats, or 512 subfiles, maximum). Twelve subfiles can contain active records or be displayed at one time.

Option indicators are not valid for this keyword.

Besides SFL, the following keywords are also valid on the subfile record format:

- For message subfiles:

  SFLMSGRCD (required at the record level)
  SFLMSGKEY (required at the field level)
  SFLPGMQ

- For all other subfiles (at the record level):

  CHANGE
  CHECK(AB)
  CHECK(RL)
  CHGINPDFT
  INDTXT
  KEEP
  LOGINP
  LOGOUT
  SETOF
  SETOFF
  SFLNXTCHG
  TEXT

  The following otherwise valid keywords are not valid at the field level when specified for the subfile record format:

  DATE
  DFTVAL
  ERRMSG
  ERRMSGID
  MSGID
  TIME

Figure 3-165 on page 3-193 shows how to specify the SFL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
00020A*           (at least one displayable field)
     A*
00030A          R SFLCTLR                   SFLCTL(SFLR)
00040A                                      SFLPAG(17)
00050A                                      SFLSIZ(17)
00060A                                      SFLDSP
00070A                                      SFLDSPCTL
     A
```

Figure 3-165. Specifying the SFL Keyword

SFLCLR (Subfile Clear)

Use this record-level keyword on the subfile control record format so that your program can clear the subfile of all records. This keyword differs from the SFLDLT keyword in that the subfile is not deleted. It differs from the SFLINZ keyword in that after being cleared, the subfile contains no data. Clearing the subfile does not affect the display. However, after being cleared, the subfile contains no active records.

This keyword has no parameters.

When active records already exist in the subfile and all are to be replaced, your program can send an output operation to the subfile control record format after selecting SFLCLR. This clears the subfile and permits your program to write new records to the subfile (by issuing output operations to the subfile record format while incrementing the relative record number). Issuing an output operation to an already active subfile record causes an error message to be returned to your program.

If SFLCLR is in effect on an output operation and no records exist in the subfile, SFLCLR is ignored.

This optional keyword is valid only for the subfile control record format. Display size condition names are not valid for this keyword.

An option indicator is required for this keyword to prevent the OS/400 program from clearing the subfile on every output operation to the subfile control record format.

Figure 3-166 shows how to specify the SFLCLR keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*             (at least one displayable field)
     A*
00020A          R SFLCTLR                   SFLCTL(SFLR)
00030A                                      SFLPAG(17)
00040A                                      SFLSIZ(17)
00050A  01                                  SFLDSP
00060A  01                                  SFLDSPCTL
00070A N01                                  SFLCLR
     A
```

Figure 3-166. Specifying the SFLCLR Keyword

The subfile is displayed when option indicator 01 is set on for an output operation to SFLCTLR, and the subfile is cleared when option indicator 01 is set off for an output operation to SFLCTLR. Normally, the option indicators specified for SFLCLR are the reverse of the option indicators specified for the SFLDSP and SFLDSPCTL keywords.

SFLCSRPRG (Subfile Cursor Progression)

Use this field-level keyword to define when the cursor leaves the field, it goes to the same field in the next subfile record instead of the next field in the same record. The SFLCSRPRG keyword is ignored when the work station is not attached to a controller that supports an enhanced data stream.

This keyword has no parameters.

Option indicators are not valid for this keyword.

The SFLLIN keyword is not allowed in a record that contains the SFLCSRPRG.

Figure 3-167 shows how to specify the SFLCSRPRG keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A    R SFL01                          SFL
     A      S1               10A  B  5  5SFLCSRPRG
     A      S2               10A  B  5 25
     A    R CTL01                          SFLCTL(SFL01)
     A                                     SFLPAG(5) SFLSIZ(20)
     A                                     SFLDSP    SFLDSPCTL
```

Figure 3-167. Specifying the SFLCSRPRG Keyword

In this example, when the cursor leaves field S1, the cursor goes to S1 in the next subfile record.

SFLCSRRRN (Subfile Cursor Relative Record Number)

Use this record-level keyword on the subfile control record format to return the relative record number of the record on which the cursor is located within a subfile. If the subfile records occupy more than one line, use this keyword in conjunction with the SFLMODE keyword to determine the location of the cursor.

The format of the keyword is:

```
SFLCSRRRN(&relative-record)
```

The relative-record parameter is required. It specifies the name of a hidden field that, on input, will contain the relative record number of the subfile record on which the cursor is located. The field must be defined in the subfile control record format as a signed numeric (S in position 35) field of length 5, with 0 decimal positions and usage H (hidden).

The relative-record field will contain the value 0 if the cursor is not located in the subfile associated with this subfile control record, or if the cursor is located within the subfile, but is not in an active record within the subfile. If the SFLMODE keyword is specified, the mode of the subfile will be returned in either case.

This keyword can be used on subfiles with field selection or subfiles with the SFLLIN keyword. If the cursor is located between two horizontal subfile records, the relative record number returned is 0.

For an example of how to specify the SFLCSRRRN keyword, see "SFLMODE (Subfile Mode)" on page 3-207.

SFLCTL (Subfile Control)

Use this record-level keyword to specify that this record format is to be a subfile control record format. This record format must immediately follow the subfile record format.

The format of the keyword is:

```
SFLCTL(subfile-record-format-name)
```

You must specify the name of the subfile record format as the parameter value for this keyword. The subfile control record format can contain field descriptions as well as subfile control keywords.

Your program can display subfile records only by issuing an output operation to the subfile control record format.

The subfile record format (SFL keyword) defines the format of the records in the subfile as opposed to the subfile control record format (SFLCTL keyword), which defines how the subfile can be displayed, cleared, deleted, and initialized. The program sends output operations to the subfile record format to build the subfile. It also sends output operations to the subfile control record format, setting option indicators for various subfile keywords to display, clear, delete, and initialize the subfile. (For information on message subfiles, see "SFLMSGKEY (Subfile Message Key)" on page 3-211, "SFLMSGRCD (Subfile Message Record)" on page 3-212, and "SFLPGMQ (Subfile Program Message Queue)" on page 3-216.)

The following is a summary of subfile keywords used with the SFLCTL. (Field-level keywords are used with fields in the subfile control record format.)

Required
SFLCTL
SFLDSP
SFLPAG
SFLSIZ

Optional
SFLCLR
SFLDLT
SFLDROP
SFLDSPCTL[1]
SFLEND
SFLENTER
SFLFOLD
SFLINZ
SFLLIN
SFLMSG
SFLMSGID
SFLPGMQ
SFLRCDNBR
SFLRNA
SFLROLVAL

[1]The SFLDSPCTL keyword is required if your program sends an input operation to the subfile control record format.

If subfile size equals subfile page, the following keywords are ignored. When several display sizes are used (DSPSIZ keyword specified), these keywords are ignored only for display sizes for which subfile size equals subfile page:

SFLDROP
SFLFOLD
SFLROLVAL

If the subfile record format contains field selection, the following keywords are not valid on the subfile control record format:

SFLDROP
SFLFOLD
SFLINZ
SFLLIN
SFLRCDNBR
SFLRNA (because SFLINZ is not valid)
SFLROLVAL

The USRDFN keyword is not valid for the subfile control record format.

Option indicators are not valid for this keyword.

Figure 3-168 shows how to specify the SFLCTL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*             (at least one displayable field)
     A*
00020A          R SFLCTLR                   SFLCTL(SFLR)
00030A                                      SFLPAG(17)
00040A                                      SFLSIZ(17)
00050A                                      SFLDSP
00060A                                      SFLDSPCTL
     A
```

Figure 3-168. Specifying the SFLCTL Keyword

SFLDLT (Subfile Delete)

Use this record-level keyword with an option indicator on the subfile control record format to enable your program to delete the subfile. When the maximum number of subfiles in a display file are already active (12) and another subfile is to be made active, your program must delete one of the active subfiles before making another active.

This keyword has no parameters.

To make a subfile active, your program sends an output operation to the subfile record format or sends an output operation to the subfile control record format with the SFLINZ keyword in effect. To delete a subfile, your program sends an output operation to the subfile control record format with SFLDLT in effect. (Closing the display file deletes all the active subfiles.)

If your program sends an output operation with SFLDLT in effect to a subfile that is not active, the SFLDLT keyword is ignored.

Option indicators are required for this keyword; display size condition names are not valid.

Figure 3-169 shows how to specify the SFLDLT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*            (at least one displayable field)
     A*
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(17)
00070A  01                                  SFLDSP
00080A  01                                  SFLDSPCTL
00090A  04                                  SFLDLT
     A
```

Figure 3-169. Specifying the SFLDLT Keyword

The subfile is displayed when option indicator 01 is set on for an output operation to SFLCTLR, and the subfile is deleted when option indicator 04 is set on for an output operation to SFLCTLR. Normally, the option indicators specified for SFLDLT are different from the option indicators specified for the SFLDSP and SFLDSPCTL keywords.

SFLDROP (Subfile Drop)

Use this record-level keyword on the subfile control record format to assign a command attention (CA) or a command function (CF) key that the work station user can press to fold or to truncate subfile records that require more than one display line.

The format of the keyword is:

```
SFLDROP(CAnn | CFnn)
```

Without SFLDROP, the OS/400 program displays the entire subfile record and folds it where needed. When SFLDROP is specified, the OS/400 program first displays the subfile in truncated form; subfile records are truncated to fit on one display line. When the work station user presses the specified key, the OS/400 program displays the records again in folded form. Each record continues onto subsequent lines immediately following the line the record starts on.

By pressing the specified key, the form of the displayed subfile changes from one state to the other.

In the truncated form, more records are displayed than are specified on the SFLPAG keyword. In the folded form, as many records are displayed as are specified on the SFLPAG keyword.

The OS/400 program truncates subfile records in the middle of output-only fields. However, if the truncation is in the middle of an input-capable field, the whole field is omitted from the display. If this results in omitting the entire record from the

display, an error message is sent to the display and the record is not truncated. Instead, it is displayed in folded form.

Notes:

1. A warning message is sent at file creation if the entire record fits on a single display line.
2. If subfile size equals subfile page, SFLDROP is ignored. When several display sizes are used (DSPSIZ keyword specified), SFLDROP is ignored only for display sizes for which the subfile size equals subfile page. If the subfile record format contains field selection, SFLDROP is not valid.
3. If the subfile contains input-capable fields, it is recommended that you specify a CF key rather than a CA key. If you specify a CA key in this situation, changed data is lost when the key is pressed.
4. If several subfiles using SFLDROP are displayed at one time, the same function key should be specified on each SFLDROP keyword. If the function keys are different, only the key specified for the most recently displayed subfile is in effect. Pressing the function key affects the subfile containing the cursor. If the cursor is not positioned in a subfile, the function key affects the upper subfile.
5. SFLDROP can be specified on the same subfile control record format as the SFLFOLD keyword. If both keywords are active, the SFLFOLD keyword is used. Indicators are checked at the time the subfile is displayed. Both keywords must use the same key.

Option indicators are valid for this keyword.

Figure 3-170 shows how to specify the SFLDROP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*            (subfile records should not fit on one screen line)
     A*
00040A          R SFLCTLR                     SFLCTL(SFLR)
00050A                                        SFLPAG(17)
00060A                                        SFLSIZ(34)
00070A                                        SFLDSP SFLDSPCTL
00090A                                        SFLDROP(CF03)
     A
```

Figure 3-170. Specifying the SFLDROP Keyword

When the subfile is displayed, the work station user can press the CF03 key to change the subfile from truncated to folded form and from folded to truncated form.

SFLDSP (Subfile Display)

Use this record-level keyword on the subfile control record format so that the OS/400 program displays the subfile when your program sends an output operation to the subfile control record format. If you do not use an option indicator with this keyword, a page of subfile records is displayed on every output operation to the subfile control record format.

This keyword has no parameters.

See "SFLRCDNBR (Subfile Record Number)" on page 3-218 to determine which page of subfile records is displayed when the subfile is displayed.

If your program sends an output operation to the subfile control record format when the SFLDSP keyword is in effect and the subfile is not activated (by adding records to it or by using SFLINZ), an error message is sent to your program.

This keyword is required and is valid only for the subfile control record format. Display size condition names are not valid for this keyword.

Option indicators are valid for this keyword.

Figure 3-171 shows how to specify the SFLDSP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*             (at least one displayable field)
     A*
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(17)
00070A  01                                  SFLDSP
00080A                                      SFLDSPCTL
     A
```

Figure 3-171. Specifying the SFLDSP Keyword

The subfile is displayed when option indicator 01 is set on for an output operation to SFLCTLR.

SFLDSPCTL (Subfile Display Control)

Use this record-level keyword on the subfile control record format so that the OS/400 program displays fields in the subfile control record format when your program sends an output operation to the subfile control record format. If you do not use an option indicator with this keyword, the subfile control record is displayed on every output operation to the subfile control record format.

This keyword has no parameters.

This optional keyword is valid only for the subfile control record format. Display size condition names are not valid for this keyword.

Option indicators are valid for this keyword.

Note: SFLDSPCTL must be in effect when the subfile is displayed for an input operation to the subfile control record to be valid, even if there are no displayable fields in the subfile control record format.

Figure 3-172 shows how to specify the SFLDSPCTL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*            (at least one displayable field)
     A*
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(17)
00070A  01                                  SFLDSP
00080A                                      SFLDSPCTL
     A                                   2 10'NAME'
     A                                   2 34'ADDRESS'
     A
```

Figure 3-172. Specifying the SFLDSPCTL Keyword

Both the subfile and displayable fields in the subfile control record format are displayed when option indicator 01 is set on for an output operation to SFLCTLR.

SFLEND (Subfile End)

Use this record-level keyword on the subfile control record format to permit the display of a plus sign (+) or text (More... or Bottom) in the lower right display location occupied by the subfile. The plus sign or More... text indicates that the work station user can move the text lines on the subfile to display more records by pressing the Page Up key.

The format of the keyword is:

```
SFLEND{(*PLUS | *MORE)}
```

The parameter values *PLUS and *MORE are optional. If no parameter is specified, *PLUS is used.

*PLUS tells the system to use the plus sign to indicate that you can use the Page Up key to see more records.

*MORE tells the system to use the More... text to indicate that you can use the Page Up key to see more records. *MORE also tells the system to use the Bottom text to indicate that the last subfile record is displayed.

When *MORE is specified, the subfile takes up one more line on the screen (SFLPAGE + 1). This line is needed for the text, More... and Bottom. If there is not room for the extra line on the display or in a window, a message is issued at file creation time and the file is not created.

An option indicator must be specified for this keyword.

Paging through Your Program (SFLPAG Equals SFLSIZ)

Your program controls the display of the plus sign or the More... or Bottom text through the use of the indicators on SFLEND. Set the indicators off to display the plus sign or the More text. Set the indicators on to remove the plus sign from the display or to display the Bottom text. When the Page Up key is pressed, your program handles processing. For instance, it reads the subfile, clears it, then rewrites the subfile with new records and displays it again. If your program does this, show the plus sign or the More text. If not, remove the plus sign from the display or show the Bottom text.

Paging through the OS/400 Program (SFLPAG Does Not Equal SFLSIZ)

The OS/400 program displays the plus sign as long as there are more records in the subfile to be displayed, no matter how the option indicator is set. When the last page of the subfile is displayed, the OS/400 program displays the plus sign or More text if the indicator is off. It does not display the plus sign or displays the Bottom text if the indicator is on.

Your program must set the indicator on or off when displaying the subfile. (Your program cannot find out, when the OS/400 program is paging through the subfile, which page of the subfile is displayed.)

If your program sets off the indicator for SFLEND when displaying the subfile, either the plus sign or the More text is displayed with the last page of the subfile. Because the plus sign is displayed but the OS/400 program cannot page the subfile any further, your program must provide for any further paging. Specify the ROLLUP keyword on the subfile control record format so that control is passed to your program when the Page Up key is pressed again. When your program receives control, it can add more records to the end of the subfile and use the SFLRCDNBR keyword to display a new page.

Position of Plus Sign with *PLUS Option

For the 24 x 80 display size, positions 78 through 80 of the last line occupied by the subfile are used for the beginning attribute character, plus sign, and ending attribute character. For the 27 x 132 display size, positions 130 through 132 of the last line occupied by the subfile are used for the beginning attribute character, plus sign, and ending attribute character.

Note: If an input field occupies the location of the plus sign and the field is changed, the plus sign and its attribute characters are returned to the program as data in the field.

Position of More... and Bottom Text with *MORE Option

For the 24 x 80 display size, positions 67 through 80 of the line immediately following the last line occupied by the subfile are used for the beginning attribute character, right justified More... or Bottom text, and ending attribute character. For the 27 x 132 display size, positions 119 through 132 of the line immediately following the last line occupied by the subfile are used for the beginning attribute character, right justified More... or Bottom text, and ending attribute character.

Figure 3-173 on page 3-202 shows how to specify the SFLEND keyword without parameters.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SETSFL1                   SFL
00020A  50                                  SFLNXTCHG
00030A            SETSEL         1Y 0B  6  2VALUES(1 2 9) CHECK(AB)
00040A            SETNAME       10   0  6  4
00050A          R SETCTL1                   SFLCTL(SETSFL1)
00060A                                      SFLSIZ(34)
00070A                                      SFLPAG(17)
00080A  40                                  SFLDSP
00090A  41                                  SFLDSPCTL
00100A  42                                  SFLDLT
00110A  43                                  SFLCLR
00120A  49                                  SFLEND
00130A N49                                  ROLLUP(26)
00140A            SETRRN         4S 0H      SFLRCDNBR(CURSOR)
     A
```

Figure 3-173. Specifying the SFLEND Keyword without Parameters

Paging is provided by the OS/400 program and the plus sign (+) appears in the lower right corner of the display. When the last record is written to the subfile, indicator 49 is set on, which disables the Page Up key and omits the plus sign from the display.

Figure 3-174 shows how to specify the SFLEND keyword with *MORE as a parameter.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SETSFL2                   SFL
00020A  50                                  SFLNXTCHG
00030A            SETSEL         1Y 0B  6  2VALUES(1 2 9) CHECK(AB)
00040A            SETNAM        10   0  6  4
00050A          R SETCTL2                   SFLCTL(SETSFL2)
00060A                                      SFLSIZ(34)
00070A                                      SFLPAG(17)
00080A  40                                  SFLDSP
00090A  41                                  SFLDSPCTL
00100A  42                                  SFLDLT
00110A  43                                  SFLCLR
00120A  49                                  SFLEND(*MORE)
00130A N49                                  ROLLUP(26)
00140A            SETRRN         4S 0H      SFLRCDNBR(CURSOR)
     A
```

*Figure 3-174. Specifying the SFLEND Keyword with the *MORE Parameter*

Paging is provided by the OS/400 program. The More... text appears at the lower right corner of the display on the line immediately following the subfile if there are more records to see in the subfile. When the last record is written to the subfile, indicator 49 is set on, which disables the Page Up key and causes the Bottom text to appear instead of More....

SFLENTER (Subfile Enter)

Use this record-level keyword on the subfile control record format to specify that the Enter key is to be used as a Page Up key. This allows the OS/400 program to page through a subfile of more than one page when the Enter key is pressed.

The format of the keyword is:

```
SFLENTER(CAnn | CFnn)
```

This optional keyword is valid only for the subfile control record format.

If a subfile is not currently displayed or you cannot page through the subfiles displayed (they have one page or less) and the Enter key is pressed, control is returned to your program. The parameter value with this keyword is required. Use it to specify a function key to replace the Enter key while this function is active. This keyword is normally used when the subfile is entirely entered by the work station user or when the user changes some records in the subfile and adds others.

Option indicators are not valid for this keyword.

Note: This keyword is in effect only until the next output operation. At the next output operation, the specifications for that record apply.

If more than one subfile using SFLENTER is displayed at the same time, the only CA or CF key in effect as an Enter key is the CA or CF key specified for SFLENTER on the most recently displayed subfile. The cursor position at the time the Enter key is pressed determines which subfile is affected.

Figure 3-175 shows how to specify the SFLENTER keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
00020A*
00030A*            (at least one displayable field)
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(51)
00070A                                      SFLDSP SFLDSPCTL
00080A                                      SFLENTER(CF01)
     A
```

Figure 3-175. Specifying the SFLENTER Keyword

The Enter key is used as a Page Up key. To enter data, the work station user presses CF01.

SFLFOLD (Subfile Fold)

Use this record-level keyword on the subfile control record format to assign a command attention (CA) or a command function (CF) key that the work station user can press to truncate or to fold subfile records that require more than one display line.

The format of the keyword is:

```
SFLFOLD(CAnn | CFnn)
```

When the SFLFOLD keyword is specified, the subfile is first displayed in folded form. When the work station user presses the specified key, the OS/400 program displays the records again in truncated form. By pressing the specified key, the form of the displayed subfile changes from one state to the other. When truncated, subfile records fit on one display line.

Without SFLFOLD, the OS/400 program displays the entire subfile record folded where needed but the work station user is not given the option to display the subfile record in truncated form.

In the folded form, as many records are displayed as are specified on the SFLPAG keyword. In the truncated form, more records are displayed than are specified on the SFLPAG keyword.

The OS/400 program truncates subfile records in the middle of output-only fields. However, if the truncation is in the middle of an input-capable field, the whole field is omitted from the display. If this results in omitting the entire record from the display, an error message is sent to the display and the record is not truncated. Instead, it is displayed in folded form.

Notes:

1. A warning message (severity 10) is sent at file creation if the entire record fits on a single display line.
2. If subfile size equals subfile page, an error message (severity 20) is issued and SFLFOLD is ignored. When several display sizes are used (DSPSIZ keyword specified), SFLFOLD is ignored only for display sizes for which the subfile size equals subfile page. If the subfile record format contains field selection, SFLFOLD is not valid.
3. If the subfile contains input-capable fields, it is recommended that you specify a CF key rather than a CA key. If you specify a CA key in this situation, changed data is lost when the key is pressed.
4. If several subfiles using SFLFOLD are displayed at one time, the same function key should be specified on each SFLFOLD keyword. If the function keys are different, only the key specified for the most recently displayed subfile is in effect. Pressing the function key affects the subfile containing the cursor. If the cursor is not positioned in a subfile, the function key affects the upper subfile.
5. SFLFOLD can be specified on the same subfile control record format as the SFLDROP keyword. If both keywords are active, SFLFOLD is used. Indicators are checked at the time the subfile is displayed. Both keywords must use the same key.

Option indicators are valid for this keyword.

Figure 3-176 shows how to specify the SFLFOLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
00020A*
00030A*            (subfile records should not fit on one screen line)
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(34)
00070A                                      SFLDSP SFLDSPCTL
00080A                                      SFLFOLD(CF03)
     A
```

Figure 3-176. Specifying the SFLFOLD Keyword

When the subfile is displayed, the work station user can press the CF03 key to change the subfile from folded to truncated form and from truncated to folded form.

SFLINZ (Subfile Initialize)

Use this record-level keyword on the subfile control record format to specify that the OS/400 program is to initialize all records in the subfile on an output operation to the subfile control record format (identified by the SFLCTL keyword). The fields in each subfile record are initialized to blanks for character type fields, to nulls for floating-point type fields, to zeros for other numeric type fields, or to the constant value specified on input-only fields if the DFT keyword is specified.

When the subfile is displayed (on an output operation to the subfile control record), all records in the subfile are displayed with the same value. Any record previously written is overwritten and no longer has its earlier value.

This keyword has no parameters.

The following is true when SFLINZ is in effect on an output operation to the subfile control record format. If keywords (such as DSPATR(HI)) are specified on fields in the subfile record format, and if option indicators are specified on those keywords, the subfile is displayed as though all option indicators are off (hex F0). Note that a keyword can be selected if N is specified for the option indicator.

After your program sends an output operation to the subfile control record with SFLINZ in effect, all records in the subfile are considered active but not changed. They are considered changed only when the work station user changes them or when your program sends an output operation to the subfile record format with the SFLNXTCHG keyword in effect.

To initialize a subfile with no active records, see "SFLRNA (Subfile Records Not Active)" on page 3-219.

In general, use SFLINZ for the following purposes:

- Specify SFLINZ with the SFLRNA keyword so that your program can initialize a subfile, then add records to that subfile without having the initialized records considered active.
- Specify SFLINZ with the SFLPGMQ keyword so that your program can build a message subfile with a single output operation. See "SFLPGMQ (Subfile Program Message Queue)" on page 3-216 for additional information.

Notes:

1. If field selection is used in the subfile record format, SFLINZ is not valid. Your program can only initialize the subfile by a series of output operations to the subfile record format, selecting fields as needed.
2. SFLINZ cannot be specified on the subfile control record format for a message subfile (see "SFLMSGRCD (Subfile Message Record)" on page 3-212) unless the SFLPGMQ keyword is also specified at the field level in the same subfile control record format.

Option indicators are valid for this keyword. Display size condition names are not valid.

Figure 3-177 on page 3-206 shows how to specify the SFLINZ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                         SFL
00020A*
00030A*            (at least one displayable field)
00040A          R SFLCTLR                      SFLCTL(SFLR)
00050A                                         SFLPAG(17)
00060A                                         SFLSIZ(17)
00070A  01                                     SFLDSP SFLDSPCTL
00080A  02                                     SFLLINZ
00090A                                         UNLOCK(*ERASE *MDTOFF)
     A
```

Figure 3-177. Specifying the SFLINZ Keyword

SFLLIN (Subfile Line)

Use this record-level keyword on the subfile control record format to specify that the subfile is to be displayed as a horizontal subfile (having more than one column of records displayed).

The format of the keyword is:

```
SFLLIN(spaces)
```

The parameter value specifies the number of spaces (including attribute characters) between columns of records.

For example, specifying the SFLLIN keyword causes a subfile of four records to be displayed as:

```
REC1    REC3
REC2    REC4
```

If SFLLIN is not specified, these records appear as:

```
REC1
REC2
REC3
REC4
```

If the subfile record format contains field selection, this keyword is not valid.

To use SFLLIN for secondary display sizes, specify a SFLLIN keyword with a display size condition name for each secondary display size.

Because the SFLPAG keyword specifies the number of subfile records that can be displayed at a single time, you must consider SFLLIN when specifying the SFLPAG value.

SFLLIN is not valid for a message subfile.

Option indicators are not valid for this keyword.

Figure 3-178 on page 3-207 shows how to specify the SFLLIN keyword. Columns of subfile records appear five spaces apart.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                         SFL
     A*
     A*           (at least one displayable field)
     A*
00040A          R SFLCTLR                      SFLCTL(SFLR)
00050A                                         SFLPAG(17)
00060A                                         SFLSIZ(17)
00070A                                         SFLDSP SFLDSPCTL
00080A                                         SFLLIN(5)
     A
```

Figure 3-178. Specifying the SFLLIN Keyword

SFLMODE (Subfile Mode)

Use this record-level keyword on the subfile control record format to return an indication of whether the subfile was in folded or truncated mode on input. You can use this keyword in conjunction with the SFLCSRRRN keyword to determine the location of the cursor within a subfile.

The format of the keyword is:

```
SFLMODE(&mode)
```

The mode parameter is required. It specifies the name of a hidden field that, on input, will contain the subfile mode. The field must be defined in the subfile control record format as a character (A in position 35) field of length 1 with usage H (hidden). The field will contain the value 0 if the subfile is in folded mode; it will contain the value 1 if the subfile is in truncated mode.

IF a SFLDROP or SFLFOLD keyword is not specified on the subfile control record, the mode value returned is 0.

Option indicators are not valid for this keyword.

Figure 3-179 on page 3-208 shows how to specify the SFLMODE and SFLCSRRRN keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R SFL01                        SFL
     A            FLD2A          2A  I  3  2
     A            FLD2B         30A  O  3  5
     A            FLD2C          6A  O  4 18
     A          R CTL01                        SFLCTL(SFL01)
     A                                         SFLSIZ(25)
     A                                         SFLPAG(4)
     A                                         SFLDSP
     A                                         SFLEND
     A                                         SFLCSRRRN(&RELRCD)
     A                                         SFLMODE(&MODE)
     A  10                                     SFLDROP(CF03)
     A  11                                     SFLFOLD(CF03)
     A                                         RTNCSRLOC(&CSRRCD &CSRFLD)
     A            RELRCD         5S 0H
     A            MODE           1A  H
     A            CSRFLD        10A  H
     A            CSRRCD        10A  H
     A
     A          R SFL02                        SFL
     A            FLD2A          2A  I 13  2
     A            FLD2B         30A  O 13  5
     A            FLD2C          6A  O 14 18
     A          R CTL02                        SFLCTL(SFL02)
     A                                         SFLSIZ(25)
     A                                         SFLPAG(4)
     A                                         SFLDSP
     A                                         SFLEND
     A                                         SFLCSRRRN(&RELRCD)
     A                                         SFLMODE(&MODE)
     A  12                                     SFLDROP(CF03)
     A  13                                     SFLFOLD(CF03)
     A                                         RTNCSRLOC(&CSRRCD &CSRFLD)
     A            RELRCD         5S 0H
     A            MODE           1A  H
     A            CSRFLD        10A  H
     A            CSRRCD        10A  H
     A
```

Figure 3-179. Specifying the SFLCSRRRN and SFLMODE Keywords

Indicators 10 and 13 are on. Two records were added to both subfiles. Both subfiles are displayed.

The following table shows the values returned for CTL02 when the cursor is at the specified positions.

Row	Column	RELRCD	MODE	CSRRCD	CSRFLD
13	2	1	0	SFL02	FLD2A
14	18	1	0	SFL02	FLD2C
15	2	2	0	SFL02	FLD2A
15	62	1	0	SFL02	blanks
17	2	0	0	SFL02	blanks
24	2	0	0	blanks	blanks
3	2	0	0	SFL01	FLD2A

The following table shows the values returned for CTL01 when the cursor is at the specified positions.

Row	Column	RELRCD	MODE	CSRRCD	CSRFLD
3	2	1	1	SFL01	FLD2A
4	18	2	1	SFL01	FLD2B
5	18	0	1	SFL01	blanks
13	2	0	1	SFL02	FLD2A

SFLMSG (Subfile Message) and SFLMSGID (Subfile Message Identifier)

Use these record-level keywords on the subfile control record format to identify a message to be displayed on the message line when your program does an output operation to the subfile control record format. Your program has the responsibility to reverse the images of any fields and to position the cursor appropriately in the subfile being displayed.

The formats of the keywords are:

```
SFLMSG('message-text' [response-indicator])

SFLMSGID(msgid [library-name/]msg-file [response-indicator] [&msg-data])
```

SFLMSG Keyword: Specify SFLMSG as you would the ERRMSG keyword. The parameters specify a message text and, optionally, a response indicator. The message text is the message to be displayed.

If you specify a response indicator, it should be the same as the option indicator used to condition SFLMSG. On the input operation that follows the display of the error message, the OS/400 program turns off the indicator. If the response and option indicators are the same, they are both turned off. One exception to this rule is if the response indicator is also specified for another keyword such as CHANGE, CAnn, or CFnn. In that case, the on/off setting of the response indicator is based on the results of the function provided by the CHANGE or CFnn keyword.

When a response indicator is specified, the first 50 characters of the message text are also used as indicator text. Separate response indicator text is not valid for SFLMSG.

SFLMSGID Keyword: Specify SFLMSGID as you would the ERRMSGID keyword.

For SFLMSGID, the parameters specify:

- The message identifier for the message to be displayed
- The message file and, optionally, the library
- Optionally, a response indicator
- Optionally, a msg-data field name

The response indicator, if specified, should be the same as the option indicator used to condition the SFLMSGID keyword. On the subsequent input operation, after the display of the error message, the OS/400 program turns off the indicator. However, if the response indicator is also specified on another keyword such as

CHANGE, CAnn, or CFnn, the on/off setting of the response indicator is based on the results of the function provided by the CHANGE, CAnn, or CFnn keyword.

Note: Indicator text cannot be specified on the SFLMSGID keyword.

The msg-data field, if specified, contains the replacement text for the specified message. The field must exist in the record format, and the field must be defined as a character field (data type A) with usage P. For more information on how replacement text works, see the SNDPGMMSG (Send Program Message) command in the *CL Reference* manual.

Conditions Occurring during Message Display

The display of messages using SFLMSG and SFLMSGID is similar to the display of messages by the OS/400 program when field validation errors are detected. An important difference from ERRMSG and ERRMSGID is that the program, and not the OS/400 program, must position the cursor to the appropriate field within the subfile, reverse the image of that field within the subfile, and also optionally reverse the image of more than one field at a time. On the 5250 work station, blinking cursor and message highlighting are allowed.

Restoration of Reversed Image Fields

See "Restoration of Reversed Image Fields" in the section, "ERRMSG (Error Message) and ERRMSGID (Error Message Identifier)" on page 3-112.

Note: The SFLDSP keyword must be in effect for SFLMSG and SFLMSGID to be processed.

Priority among Selected Keywords

You can specify either SFLMSG or SFLMSGID several times for a single subfile control record format. In your program, set option indicators to select a particular message to be displayed and to select particular fields to be displayed in reverse image. Several fields can be displayed in reverse image in different records of a subfile when the subfile is displayed again. However, only one message can be displayed at one time.

If more than one error message is selected at a time, the OS/400 program displays the first of the following:

1. ERRMSG (If more than one ERRMSG keyword is selected, the first one selected is displayed.)
2. ERRMSGID (If more than one ERRMSGID keyword is selected, the first one selected is displayed.)
3. SFLMSG (If more than one SFLMSG keyword is selected, the first one selected is displayed.)
4. SFLMSGID (If more than one SFLMSGID keyword is selected, the first one selected is displayed.)
5. Message fields (M in position 38) (If more than one message field is selected, the first one selected is displayed.)

Multiple subfile messages (SFLMSG and SFLMSGID) are allowed on error subfiles.

Option indicators are valid for these keywords.

Figure 3-180 on page 3-211 shows how to specify the SFLMSG and SFLMSGID keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
00020A*
00030A*           (at least one displayable field)
00040A*
00050A          R SFLCTLR                   SFLCTL(SFLR)
00060A                                      SFLPAG(17)
00070A                                      SFLSIZ(17)
00080A                                      SFLDSP SFLDSPCTL
00090A  11                                  SFLMSGID(USR0006 PAYROLL/UMSGF1 +
00100A                                      11 &RPLTXT)
00110A  12                                  SFLMSGID(USR0007 PAYROLL/UMSGEF1 +
00120A                                      12 &RPLTXT)
00130A            RPLTXT        78A  P
     A
```

Figure 3-180. Specifying the SFLMSG and SFLMSGID Keywords

SFLMSGKEY (Subfile Message Key)

Use this field-level keyword on the first field in the subfile record format for a message subfile (SFL and SFLMSGRCD keywords specified). It is not valid on the subfile control record format (SFLCTL). To select messages from a program message queue for display, your program places a message reference key in this field. Your program also places the name of the program message queue in the second field in the subfile record format. See "SFLPGMQ (Subfile Program Message Queue)" on page 3-216.

This keyword has no parameters.

This field is predefined as a four-position, character data type, hidden field. The following rules apply:

- This field must always be the first field defined in the subfile record format.
- The field name and SFLMSGKEY are the only DDS you can specify for this field.

Option indicators are not valid for this keyword or with the associated field.

For more information on building and displaying message subfiles, see "SFLPGMQ (Subfile Program Message Queue)" on page 3-216.

Figure 3-181 shows how to specify the SFLMSGKEY keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCDMSG                    SFL SFLMSGRCD(3)
00020A            FLDKEY                    SFLMSGKEY
00030A            FLDPGM                    SFLPGMQ
00040A          R SFLCTL                    SFLCTL(RCDMSG)
00050A  01                                  SFLINZ
00060A                                      SFLPAG(17)
00070A                                      SFLSIZ(17)
00080A                                      SFLDSP SFLDSPCTL
00090A            FLDPGM                    SFLPGMQ
     A
```

Figure 3-181. Specifying the SFLMSGKEY Keyword

SFLMSGRCD (Subfile Message Record)

Use this record-level keyword on the subfile record format to specify that this subfile is to be a message subfile and that the records displayed when the subfile is displayed are to be messages from a program message queue.

The format of the keyword is:

```
SFLMSGRCD(line-number)
```

The parameter value specified with SFLMSGRCD specifies the first line on the display on which messages are displayed. The value specified must not be greater than the maximum line number for the display size being used. The number of messages displayed depends on the SFLPAG value specified for the subfile.

For more information on building and displaying message subfiles, see "SFLPGMQ (Subfile Program Message Queue)" on page 3-216.

The TEXT keyword is valid at the record-level for SFLMSGRCD.

There can be only two predefined fields specified on the subfile record format for a message subfile as follows:

- Message identifier. A 4-position, character data type, hidden field. Your program uses this field to pass a message identifier to the OS/400 program. This field must always be the first field defined in the message subfile. You must specify the SFLMSGKEY keyword with this field.
- Program queue name. A 10-position, character data type, hidden field. Your program passes the name of the program message queue that contains the message(s) in this field. It must be the second field of a subfile message record and must immediately follow the first field. If specified also on the subfile control record, it can be anywhere within the record specification. You must specify SFLPGMQ with this field. See "SFLPGMQ (Subfile Program Message Queue)" on page 3-216 for more details.

Display size condition names can be specified for SFLMSGRCD and are required if the line number for the first message displayed is to change, based on display size.

Data is not returned in your input buffer if your program does an input operation for a message subfile.

The messages are displayed as follows:

- Each message is displayed on a separate line and is truncated if it is longer than the display line length.
- Each message starts in position 2. The maximum message length for the 24 x 80 display size is 76 characters. The maximum message length for the 27 x 132 display size is 128 characters.
- Each message is displayed with the high intensity (HI) field attribute.

When a message subfile is paged through the OS/400 program, the cursor is positioned at the same location as it was when the Page key was pressed.

Message help is supported for these messages. The work station user chooses which message help is to be displayed by placing the cursor on the line containing the message and pressing the Help key.

Notes:

1. When SFLMSGRCD is specified, you cannot specify the SFLINZ keyword without specifying SFLPGMQ.
2. The message subfile, starting on the line specified on SFLMSGRCD, must not overlap any displayable fields in the subfile control record, even if option indicators are specified on the displayable fields.

Option indicators are not valid for this keyword; display size condition names are valid.

Figure 3-182 shows how to specify the SFLMSGRCD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A          R RCDMSG                  SFL SFLMSGRCD(3)
00040A            FLDKEY                  SFLMSGKEY
00050A            FLDPGM                  SFLPGMQ
00060A          R SFLCTL                  SFLCTL(RCDMSG)
00070A                                    SFLPAG(17)
00080A                                    SFLSIZ(17)
00090A                                    SFLDSP SFLDSPCTL
     A
```

Figure 3-182. Specifying the SFLMSGRCD Keyword

The highlighed keywords are required on the subfile record format for a message subfile. The SFLMSGKEY and SFLPGMQ keywords must be specified in the order shown.

SFLNXTCHG (Subfile Next Changed)

Use this record-level keyword on the subfile record format to force the work station user to correct program-detected keying errors in subfile records that have been read by the program. It does this by causing a record to be changed so that a get-next-changed operation must read the record as described in the following section.

This keyword has no parameters.

Subfile Operations with SFLNXTCHG

A typical use of SFLNXTCHG could be as follows:

A work station user changes some records in a displayed subfile (this could be for a data-entry application or a data-update application). After changing some records, the work station user presses the Enter key, and the program reads only the changed records with get-next-changed operations. (For example, READC in RPG III and READ-SUBFILE-NEXT-MODIFIED in COBOL.)

If the program detects keying errors in the changed records, it can send update operations (UPDAT in RPG III, REWRITE SUBFILE in COBOL) to the subfile records in error, setting indicators so that SFLNXTCHG is in effect during the update operations. These update operations are sent to the subfile record format.

After all the records in error have been updated, the program sends an output/input operation to the subfile control record format to display the subfile again.

With the subfile displayed again, the work station user types the data again and presses the Enter key. If the data is correct, the program does not display the subfile again.

The records in error (and any other records changed by the work station user) are returned to the program on the next get-next-changed operation. This is because SFLNXTCHG caused the subfile records to be considered changed even though the work station user did not change them. This allows the program to prohibit the work station user from ignoring program-detected keying errors in subfile records.

Subfile Operations without SFLNXTCHG

If SFLNXTCHG is not specified, or is specified but not selected on the update operations to the subfile records, then the work station user can simply press the Enter key instead of correcting the program-detected errors. The program then reads no records because the get-next-changed operations find no changed records the second time the Enter key is pressed.

Option indicators are valid for this keyword.

You cannot specify SFLNXTCHG with the SFLMSGRCD keyword.

Figure 3-183 shows how to specify the SFLNXTCHG keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
00020A  14                                  SFLNXTCHG
     A*
     A*            (at least one input-capable field should be specified)
     A*
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(17)
00070A                                      SFLDSP SFLDSPCTL
     A
```

Figure 3-183. Specifying the SFLNXTCHG Keyword

SFLPAG (Subfile Page)

Use this record-level keyword on the subfile control record format to specify the number of records in the subfile to be displayed at the same time. (For an exception to this rule, see "Field Selection" on page 3-215.)

The format of the keyword is:

```
SFLPAG(number-of-records-to-be-displayed)
```

The SFLPAG parameter value and the number of lines required by each subfile record determine the number of actual lines required to display the page of records. Not all records within a subfile must be displayed at the same time, and not all lines of the display are required to display a page of subfile records.

This keyword is required for the subfile control record format.

Subfile Page Equals Subfile Size

When you specify the same parameter values for SFLPAG and SFLSIZ, the maximum number of records that can be contained in the subfile equals the maximum number of subfile records that can appear on the display at one time. For this condition, the OS/400 program does not automatically page through the subfile when the Page Up or the Page Down key is pressed. If the ROLLUP and ROLLDOWN keywords are specified and one of the Page keys is pressed, the OS/400 program returns control to your program instead. If ROLLUP and ROLLDOWN are not specified, a message is sent to the work station user, indicating that a key is not supported on the display.

If subfile size equals subfile page, the following keywords are not allowed:

SFLDROP
SFLFOLD
SFLROLVAL

When several display sizes are used (DSPSIZ keyword specified), these keywords are ignored only for display sizes for which subfile size equals subfile page.

Field Selection: When subfile page equals subfile size, you can specify option indicators for fields in the subfile record format. This is called *field selection*. When field selection is used in the subfile record, SFLPAG(value) specifies the number of display lines available to display the records of this subfile. (Without field selection, SFLPAG(value) specifies the number of subfile records that can be displayed at one time.) This specification must be considered when a subfile record occupies more than one display line. The value of SFLPAG must be greater than or equal to the number of display lines occupied by the subfile.

If the subfile record format contains field selection, the following keywords are not valid on the subfile control record format:

SFLDROP
SFLFOLD
SFLINZ
SFLLIN
SFLRCDNBR
SFLRNA (because SFLINZ is not valid)
SFLROLVAL

Subfile Page Does Not Equal Subfile Size

When you specify different parameter values for SFLPAG and SFLSIZ, the OS/400 program recognizes the Page Up and Page Down keys and automatically pages through the subfile according to the value specified in the field for which the SFLROLVAL keyword is specified. If you do not specify the SFLROLVAL keyword, the OS/400 program pages through the subfile by the parameter value specified for the SFLPAG keyword except for subfiles using SFLDROP. If using the SFLDROP keyword, more records are displayed than the SFLPAG value when records are displayed in the truncated format. For truncated records, the OS/400 program pages through the display by the number of records displayed in the truncated format.

Option indicators are not valid for this keyword. Display size condition names are valid. Display size condition names are required if you want the number of records that can be displayed at one time to change, based on the size of the display.

Figure 3-184 shows how to specify the SFLPAG keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*           (at least one displayable field)
     A*
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(17)
00070A                                      SFLDSP SFLDSPCTL
     A
```

Figure 3-184. Specifying the SFLPAG Keyword

Because the value specified for the SFLPAG keyword equals the value specified for SFLSIZ(17), subfile page equals subfile size.

SFLPGMQ (Subfile Program Message Queue)

Use this field-level keyword on the second (and last) field in the subfile record format for a message subfile. This field contains the name of the program message queue used by the OS/400 program to build a message subfile. In addition, SFLPGMQ can be specified on the subfile control record format when the SFLINZ keyword is specified on the subfile control record format.

The format of the keyword is:

```
SFLPGMQ([10] | [276])
```

When 10 is specified, SFLPGMQ generates a 10-byte field. 10 is the default.

When 276 is specified, SFLPGMQ generates a 276-byte field.

This field is predefined as a character data type, hidden field. The following rules apply:

- The field name and the SFLPGMQ keyword and parameters are the only DDS you can specify for this field.
- If the name of the program message queue placed in this field at processing time is less than the field length (10 or 276 bytes), it must be left-adjusted and padded with blanks.

For Integrated Language Environment* (ILE*) programs using the 276-byte parameter value, the format of the field data must be as follows:

- The first 256 bytes contains the ILE call message queue name. The call message queue name is the same as the ILE procedure name. The name must be left-adjusted and padded with blanks.
- Bytes 257 through 266 will optionally contain the ILE module name. The name, when specified, must be left-adjusted and padded with blanks. If no module name is provided, these bytes must be set to blanks.

- Bytes 267 through 276 will optionally contain the name of the ILE bound program name. The name, when specified, must be left-adjusted and padded with blanks. If no bound program name is provided, these bytes must be set to blanks.

  Notes:

  1. If a parameter value of 10 is used on SFLPGMQ and an ILE procedure name longer than 10 bytes is placed into this field at processing time, the procedure name is truncated to 10 bytes. The results will be unpredictable.
  2. If a parameter value of 276 is used on SFLPGMQ and a program message queue name is placed into this field at processing time, bytes 257 through 276 must be set to blanks. If these bytes are not blank, the system assumes that a call message queue name has been given and will not find the specified program message queue.

- If the SFLPGMQ keyword is specified on both the subfile and subfile control record, the SFLPGMQ parameter values must match. However, different subfiles within the same file can use different SFLPGMQ parameter values.

This field is required on the subfile record format (identified by the SFL keyword) to build the subfile one message at a time through multiple output operations to the subfile record format.

You can also specify this field on the subfile control record format (identified by the SFLCTL keyword) to build the subfile all at once through a single output operation to the subfile control record. Specify option indicators with the SFLINZ keyword to control the way the subfile is built.

Multiple Output Operation

If you specify the field name and SFLPGMQ on the subfile record, you build the subfile one message at a time with separate output operations to the subfile record format. For each output operation, the message reference key must be in the first field of the record (SFLMSGKEY keyword), and the name of the program message queue must be in the second field. At the time of the output operation, the OS/400 program retrieves the identified message from the queue and places it in the subfile as a record.

Note: A CL program cannot be used for a multiple output operation. The relative record number required each time a message is built is not supported for CL.

Single Output Operation

If you specify SFLPGMQ (with its named field) and the SFLINZ keyword on the subfile control record format, you build the entire subfile with one output operation directed to the subfile control record format. On the output operation, the OS/400 program initializes the subfile with all messages that are on the program message queue whose name is in the SFLPGMQ field. If necessary, the OS/400 program extends the subfile to contain all messages on the queue. For this function, the SFLMSGRCD, SFLMSGKEY, and SFLPGMQ keywords must be specified with the subfile record format (SFL keyword). The SFLPGMQ and SFLMSGKEY keywords are ignored for this function and your program need not set the values of their fields.

Special Value

The SFLPGMQ field can contain a special value, * (asterisk), instead of a program message queue name. If the program moves an asterisk to the SFLPGMQ field, the OS/400 program uses the message queue of the program issuing the output operation. You cannot use an asterisk if your program is a CL program.

Both Multiple and Single Output Operations

If you specify SFLPGMQ with both the subfile record format and subfile control record format, you can use the single operation function one time and the multiple operation function some other time. Do this by setting indicators before issuing the output operation; however, all operations to a particular subfile must be consistent (multiple or single, but not intermixed) when preparing for a single display of the subfile.

Option indicators and display size condition names are not valid for this keyword.

Figure 3-185 shows how to specify the SFLPGMQ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCDMSGILE               SFL SFLMSGRCD(3)
00020A            FLDKEY                  SFLMSGKEY
00030A            FLDPGM                  SFLPGMQ(276)
00040A          R SFLCTLILE               SFLCTL(RCDMSG)
00050A  01                                SFLINZ
00060A                                    SFLPAG(17)
00070A                                    SFLSIZ(17)
00080A                                    SFLDSP SFLDSPCTL
     A            :
     A            :
00110A            FLDPGM                  SFLPGMQ(276)
     A          R RCDMSGOPM               SFL SFLMSGRCD(3)
     A            FLDKEY                  SFLMSGKEY
     A            FLDPGM                  SFLPGMQ
     A          R SFLCTLOPM               SFLCTL(RCDMSG)
     A  02                                SFLINZ
     A                                    SFLPAG(17)
     A                                    SFLSIZ(17)
     A                                    SFLDSP SFLDSPCTL
     A              :
     A              :
     A            FLDPGM                  SFLPGMQ(10)
     A
```

Figure 3-185. Specifying the SFLPGMQ Keyword

In Figure 3-185, the program can build the subfile with more than one output operation (indicator 01 off) or a single output operation (indicator 01 on) to the subfile control record format.

In the first record, the name of the subfile program queue can be as long as 276 bytes, while the name of the subfile program queue in the third record format can only be 10 bytes long.

SFLRCDNBR (Subfile Record Number)

Use this field-level keyword on the subfile control record format to specify that the page of the subfile to be displayed is the page containing the record whose relative

record number is in this field. If you do not specify this keyword, the OS/400 program displays the first page of the subfile by default.

The format of the keyword is:

```
SFLRCDNBR[(CURSOR)]
```

If CURSOR is specified, the cursor is placed in the subfile record whose relative record number is identified by the contents of this field. The cursor is positioned at the first input-capable field in the subfile record. If there is no input-capable field, the cursor is positioned at the first output-only or constant field. For example, if a page can contain three records, and nine records are contained in the subfile, a SFLRCDNBR field value of 8 causes records 7, 8, and 9 to be displayed. If CURSOR is specified, the cursor appears in record 8.

This field must be a zoned decimal field with zero decimal positions. It must have the keyboard shift attribute of signed numeric (S in position 35), and it can be up to 4 digits in length. It must be defined as an output-only, an output/input, or a hidden field. If a value less than 1 or a value greater than the number of records in the subfile is contained in this field on an output operation to the subfile control record format, an error is returned to your program.

This optional keyword is valid only for the subfile control record format.

You cannot specify both SFLRCDNBR and SFLROLVAL for the same field.

If the subfile record format contains field selection, this keyword is not allowed.

Option indicators are not valid for this keyword.

Figure 3-186 shows how to specify the SFLRCDNBR keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*            (at least one displayable field)
     A*
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(17)
00070A                                      SFLDSP SFLDSPCTL
00080A            DSPREC         4S 0B  4 12SFLRCDNBR(CURSOR)
     A
```

Figure 3-186. Specifying the SFLRCDNBR Keyword

In Figure 3-186, either the program or the work station user can set the value of the field before displaying the subfile.

SFLRNA (Subfile Records Not Active)

Use this record-level keyword with the SFLINZ keyword on the subfile control record format so that your program can initialize a subfile with no active records. To do this, your program sends an output operation to the subfile control record format with the SFLINZ keyword selected. The subfile itself becomes active. Subfile records are not considered active unless one of the following occurs:

- Your program sends an output operation to the subfile record format, placing data in one of the subfile records. The subfile record becomes active but is not considered changed unless the SFLNXTCHG keyword is also in effect.
- After your program displays the subfile, the work station user keys data into subfile records. The records keyed in become active and changed.

This keyword has no parameters.

SFLRNA is normally used so that a program can write some records to the subfile before displaying it, and the work station user can then add records to the subfile.

When your program displays a subfile initialized with the SFLINZ and SFLRNA keywords in effect, fields in inactive records have the following values:

- Character fields are blank.
- Numeric fields are zero-filled.
- An input-only field with a constant value specified has the constant value.

Your program cannot send an input operation to an inactive subfile record. Issuing a get-next-changed operation to one of the subfile records returns the record only when the record has become active and has been changed.

Your program cannot send output operations to active records (SFLRNA not specified). It must send update operations. Also, your program cannot send update operations to inactive records (SFLRNA specified). It must send output operations.

The SFLINZ keyword is required when SFLRNA is specified.

SFLRNA cannot be specified for a message subfile (identified by the SFLMSGRCD keyword on the subfile record format).

If the subfile record format contains field selection, SFLRNA is not valid.

Option indicators are not valid for this keyword.

Figure 3-187 shows how to specify the SFLRNA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                      SFL
     A*
     A*            (at least one displayable field)
     A*
00040A          R SFLCTLR                   SFLCTL(SFLR)
00050A                                      SFLPAG(17)
00060A                                      SFLSIZ(17)
00070A                                      SFLDSP SFLDSPCTL
00080A                                      SFLINZ
00090A                                      SFLRNA
     A
```

Figure 3-187. Specifying the SFLRNA Keyword

SFLROLVAL (Subfile Roll Value)

Use this field-level keyword in the subfile control record format to specify that the work station user can key a value into this field to tell the OS/400 program how many records to page up or down when the appropriate paging key is pressed.

This field must have the keyboard shift attribute of signed numeric with zero decimal positions. It can be up to 4 digits in length, and it must be defined as an output/input or input-only field.

This keyword has no parameters.

The work station user can page through the data being displayed up or down by first keying in the number of records to page through and then pressing the Page Up or the Page Down key. (On subsequent pages, the SFLROLVAL value stays the same unless a new number is keyed in before paging.) If a negative number or zero is keyed into this field and a Page key is pressed, an error message is displayed at the work station.

This keyword is valid only for the subfile control record format. You must specify it if the OS/400 program is to support the page-by-record function.

If this keyword is not specified, the OS/400 program pages through the display by the SFLPAG value except for subfiles using the SFLDROP keyword. If using the SFLDROP keyword, more records are displayed than the SFLPAG value when records are displayed in the truncated format. For truncated records, the OS/400 program pages through the display by the number of records displayed in the truncated format.

If subfile size equals subfile page, SFLROLVAL is ignored. When several display sizes are used (DSPSIZ keyword specified), SFLROLVAL is ignored only for display sizes for which subfile size equals subfile page. If the subfile record format contains field selection, SFLROLVAL is not valid.

This field is returned to your program as part of the input for this subfile control record.

If pressing the Page Down key pages beyond the first page of records of the subfile, one of the following conditions occurs:

- If the first page of records is not currently displayed, paging down will display it.
- If the first page of records is currently displayed, paging down will cause a message to be displayed.

If pressing the Page Up key pages beyond the last active record of the subfile, one of the following conditions occurs:

- If the last full page of records is not already displayed, keying Page Up will display it.
- If the last full page of records is already displayed, keying Page Up will cause a message to be displayed. One exception to this rule is that when the SFLROLVAL value is less than the SFLPAG value, the OS/400 program pages through the subfile again and no message is displayed.

Certain keywords are helpful when specified with SFLROLVAL:

- The SFLEND keyword notifies the work station user when the last subfile record is displayed. For more information on the SFLEND keyword, see "SFLEND (Subfile End)" on page 3-200.
- The ROLLUP or ROLLDOWN keywords cause control to return to the program when pressing a Page Up or a Page Down key would page beyond the end of

the subfile. Without ROLLUP or ROLLDOWN, a message is displayed (as described previously).

Note: The ROLLUP keyword is the same as the PAGEDOWN keyword and the ROLLDOWN keyword is the same as the PAGEUP keyword.

The following examples illustrate the use of SFLROLVAL:

- Paging Down. Assume that the value specified for SFLPAG is 3, and there are 11 active records in the subfile. If records 8 through 10 are currently being displayed, and the user types a page value greater than 7, pressing the Page Down key displays records 1 through 3.

  If records 1 through 3 are currently displayed, and a Page Down key is entered with a SFLROLVAL value greater than 0, either a message is sent to the work station user (ROLLDOWN not specified) or control is returned to the user program (ROLLDOWN specified; the program has responsibility for paging down).

- Paging Up. Assume that the value specified for SFLPAG is 3, and there are 11 active records in the subfile. If records 8 through 10 are currently being displayed, and the user enters a 3 into the SFLROLVAL field, pressing the Page Up key displays record 11 in the uppermost page area of the display. Any lines not occupied by that record are blank. If the Page Up key is pressed again, the last full page of subfile records (records 9 through 11) are displayed. Finally, if the Page Up key is pressed a third time, either a message is sent to the work station user (ROLLUP not specified) or control is returned to the user program (ROLLUP specified; the program has responsibility for paging up).

The following shows the conditions that occur when paging beyond the ends of the subfile (when the SFLROLVAL value is greater than the SFLPAG value).

- On pressing the Page Up key:
  - If the last full page of records is not already displayed, then it is displayed.
  - If the last full page of records is already displayed, then a message is displayed.
- On pressing the Page Up key a second time:
  - If ROLLUP is specified, control returns to your program.
  - If ROLLUP is not specified, a message is displayed.
- On pressing the Page Down key:
  - If the first page of the subfile is not already displayed, then it is displayed.
  - If the first page of the subfile is already displayed, then:
    - If ROLLDOWN is specified, control returns to your program.
    - If ROLLDOWN is not specified, a message is displayed.

You cannot specify both the SFLROLVAL and the SFLRCDNBR keywords for the same field.

Option indicators are not valid for this keyword.

Figure 3-188 on page 3-223 shows how to specify the SFLROLVAL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                         SFL
     A*
     A*           (at least one displayable field)
     A*
00040A          R SFLCTLR                      SFLCTL(SFLR)
00050A                                         SFLPAG(17)
00060A                                         SFLSIZ(17)
00070A                                         SFLDSP SFLDSPCTL
00080A            ROLVAL         4S 0B  1 47SFLROLVAL
     A
```

Figure 3-188. Specifying the SFLROLVAL Keyword

SFLSIZ (Subfile Size)

Use this record-level keyword on the subfile control record format to specify the number of records in the subfile. The maximum number of records allowed is 9999. This keyword is required for the subfile control record format.

The format of the keyword is:

```
SFLSIZ(number-of-records-in-subfile)
```

Subfile Size Equals Subfile Page

When you specify the same parameter values for SFLSIZ and the SFLPAG keyword, you can specify option indicators for fields in the subfile record format. (This is called field selection.)

When the subfile is built, the records can vary in length depending on which fields are selected, and each output operation places records into successive positions within the subfile. When the subfile is displayed, each record can require a different number of display lines. The number of records that actually fit in the subfile depends on the fields selected for each record written to the subfile.

If the last subfile record written to the subfile fits exactly into the subfile, a status message (CPF5003) is returned to your program. If the last subfile record written to the subfile overflows the subfile, a notify message (CPF5043) is returned to your program.

The specified SFLPAG value is increased to equal the maximum number of records that fit on the display if the number of subfile records to be displayed do not occupy a full display.

The SFLSIZ value is increased by the same value. For example, if SFLPAG(13) and SFLSIZ(13) are specified, and the subfile record format and SFLLIN value are specified such that three records can fit on a single display line, SFLPAG and SFLSIZ are increased to 15.

Option indicators are not valid for this keyword. Display size condition names are valid and are required if the number of records within the subfile changes depending on the display size.

Figure 3-189 on page 3-224 shows how to specify the SFLSIZ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SFLR                          SFL
00020A  21        FIELD1        78      2  2
00030A  22        FIELD2        40      3  2
00040A*
00050A          R SFLCTLR                       SFLCTL(SFLR)
00060A                                          SFLPAG(5)
00070A                                          SFLSIZ(5)
00080A                                          SFLDSP
00090A                                          SFLDSPCTL
     A
```

Figure 3-189. Specifying the SFLSIZ Keyword

Your program issues the following output operations:

Output Operation To	Option Indicators Set	Result
SFLR	21 on 22 off	Only FIELD1 written to subfile
SFLR	21 on 22 on	FIELD1 and FIELD2 written to subfile
SFLR	21 off 22 on	Only FIELD2 written to subfile

(The OS/400 program sends status message CPF5003 to your program.)

SFLCTLR	No indicator necessary	Subfile displayed

The resulting display is as follows:

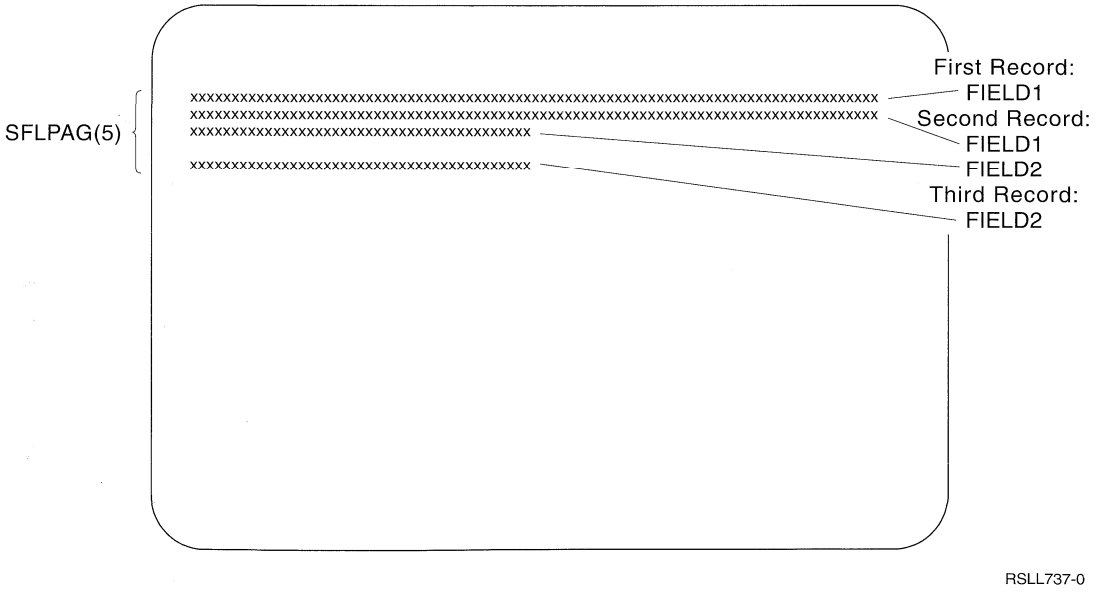


Figure 3-190. Subfile Display

In Figure 3-190, a fourth record cannot be written to the subfile because there is no room on the display for it (SFLPAG(5) has been specified in the DDS).

Subfile Size Does Not Equal Subfile Page

When you specify different parameter values for SFLPAG keyword and SFLSIZ, the SFLSIZ value specifies the number of records that can be placed into the subfile. If your program places a record with a relative record number larger than the SFLSIZ value into the subfile, the subfile is automatically extended to contain it (up to a maximum of 9999 records). The parameter value you specify should be large enough to accommodate the maximum number of records you would normally have in the subfile.

SLNO (Starting Line Number)

Use this record-level keyword to specify a starting line number for the record format you are defining. If specified, SLNO adjusts the actual line numbers for each field in the record format. If you do not specify SLNO, fields in the record format are displayed on the lines you specify for them in positions 39 through 41.

The format of the keyword is:

```
SLNO(n | *VAR)
```

You can specify one of two parameter values for this keyword:

- Specify n, where n is a value between 1 and 27. All fields in the record format are offset n–1 lines down the display from their specified locations. If SLNO(1) is specified, the record format must not contain a field starting at line 1, position 1.
- Specify *VAR to enable your program to set the starting line number at run time before displaying the record format. At file creation time, the OS/400 program sets the starting line number to one. A warning message appears at file creation time if the record contains a field starting at line 1, position 1. If your program does not set the starting line number or sets it to zero, the OS/400 program assumes its value is one. If your program sets the starting line number to a value such that the first field in the record format does not all fit on the display, or sets it to a negative value, the OS/400 program sends a notify message (CPF5002) to your program, and the record is not displayed. If the starting line number is set to one and the record format contains a field starting at line 1, position 1, the OS/400 program sends an error message (CPF5398) to your program. The record is not displayed.

  To calculate the line on which a field is actually displayed, subtract one from the line number specified in positions 39 through 41 and add the starting line number to the result. The record format begins on the line specified with SLNO unless a field is defined at line 1, position 1. In that case, the beginning attribute byte is in the last position of the previous line and the starting line of the format is one less than that specified by SLNO.

  When *VAR is specified, no field in the record can occupy the last position on the display.

If a CLRL(nn) or CLRL(*END) keyword is also in effect for this record when it is to be displayed, lines on the display are cleared beginning with the starting line number for the format.

If you use the SLNO(*VAR) keyword with the OVERLAY keyword but without the CLRL keyword and then write the record several times, each time with a different starting number, the previous record is deleted before the new record displays.

The OS/400 program checks the starting line number to determine whether the previous output operation to the record had the same starting line number if you use the SLNO keyword with the following keywords:

ERRMSG
ERRMSGID
PUTOVR
PUTRETAIN

If the starting numbers are the same, the actions specified by the ERRMSG, ERRMSGID, PUTOVR, or the PUTRETAIN keyword is performed.

If the starting line numbers are not the same, the ERRMSG, ERRMSGID, PUTOVR, or PUTRETAIN keyword is ignored, and the record format displays with the lines adjusted to the new value.

The SLNO keyword is not allowed in a record format that has one of the following keywords specified:

ASSUME
KEEP
SFL
SFLCTL
USRDFN

SLNO cannot be specified for the record format specified by the PASSRCD keyword.

Option indicators are not valid for this keyword.

Figure 3-191 shows how to specify the SLNO keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   SLNO(*VAR)
00020A            FIELD1         5   I  2  2
00030A            FIELD2         5   B  3  2
00040A
00050A          R RECORD2                   SLNO(2)
00060A            FIELD3         5     10  2
00070A            FIELD4         5   B 10 13
     A
```

Figure 3-191. Specifying the SLNO Keyword

In Figure 3-191, when the starting line number is zero or one, FIELD1 is displayed on line 2 and FIELD2 is displayed on line 3, as specified. When the starting line number is set to 2 in your program, FIELD1 is displayed on line 3 (2 – 1 + 2 = 3) and FIELD2 is displayed on line 4 (2 – 1 + 3 = 4).

FIELD3 and FIELD4 are always displayed on line 11 (2 – 1 + 10 = 11).

SNGCHCFLD (Single-Choice Selection Field)

Use this field-level keyword to define a field as a single-choice selection field. A single-choice selection field is a field that contains a fixed number of choices from which a user can select one choice. The field appears as a vertical list of choices, with a single input field at the upper left, or as a group of radio buttons.

This keyword has no parameters.

A field containing the SNGCHCFLD keyword must also contain one or more CHOICE keywords defining the choices for the field.

The field containing the SNGCHCFLD keyword must be defined as an input-capable field with data type Y, length equal to 2, and decimal positions of 0. The position specified for the field is the position of the input field displayed to the left of the first choice, or of the uppermost radio button. On input, the field contains the number of the choice selected, or 0 if no choice was selected. On output, if the field contains a choice number, that choice is displayed as the default selection.

The following keywords can be specified on a field with the SNGCHCFLD keyword:

ALIAS
AUTO(RA)
BLANKS
CHANGE
CHCACCEL
CHCAVAIL
CHCUNAVAIL
CHCSLT[1]
CHCCTL
CHECK(ER, FE)[2]
CHOICE
CHGINPDFT
COLOR[3]
DSPATR(RI UL BL CS HI ND PC)[3]
ENTFLDATR
ERRMSG
ERRMSGID
FLDCSRPRG
INDTXT
OVRATR
OVRDTA
PUTRETAIN
TEXT

Notes:

1. CHCSLT functions only if the single-choice selection field is displayed in a pull-down menu that does not display selection indicators (for example, PULLDOWN(*NOSLTIND) specified).
2. Check(FE) applies only to a display attached to a controller that does not support an enhanced interface.
3. If the COLOR or DSPATR keyword is specified for a field with the SNGCHCFLD keyword, it applies only to the input field portion of the selection field on character-based displays.

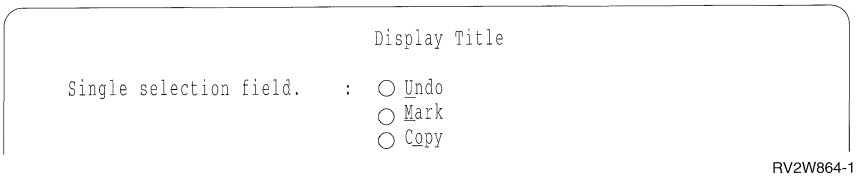
Option indicators are not valid for this keyword.

Figure 3-192 shows how to specify the SNGCHCFLD keyword:

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A
     A          R RECORD
     A            :
     A            :
     A                                    3  3'Single selection field.      :'
     A            F1             2Y 0B  3 35SNGCHCFLD
     A  01                                   CHOICE(1 '>Undo       ')
     A                                       CHOICE(2 &MARKTXT)
     A                                       CHOICE(3 '>Copy       ')
     A
```

Figure 3-192. Specifying the SNGCHCFLD Keyword

In this example, when using a graphical display station attached to a controller that supports an enhanced interface for nonprogrammable work stations, the selection fields look like this:



SYSNAME (System Name)

Use this field-level keyword to display the current system name as a constant (output-only) field that is 8 characters long. You can specify the location of the field, the SYSNAME keyword, and, optionally, the COLOR, DSPATR, and TEXT keywords. Positions 17 through 38 must be blank.

This keyword has no parameters.

Option indicators are not valid for this keyword, although option indicators can be used to condition the field on which it is specified.

Figure 3-193 shows how to specify the SYSNAME keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A                                      2 62'SYSTEM:'
00030A                                      2 72SYSNAME
     A
```

Figure 3-193. Specifying the SYSNAME Keyword

TEXT (Text)

Use this record- or field-level keyword to supply a text description (or comment) for the record format or field that is used for program documentation.

TEXT is valid for any record format or field, except a SFLMSGKEY or SFLPGMQ field.

The format of the keyword is:

```
TEXT('description')
```

The text must be enclosed in apostrophes. If the length of the text is greater than 50 positions, only the first 50 characters are used by the high-level language compiler. Option indicators are not valid for this keyword.

Figure 3-194 on page 3-229 shows how to specify the TEXT keyword at the record and field levels.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CUSMST                    TEXT('Customer Master Record')
00020A            FLD1           3  0       TEXT('ORDER NUMBER FIELD')
     A
```

Figure 3-194. Specifying the TEXT Keyword

TIME (Time)

Use this field-level keyword to display the current system time as a constant (output-only) field.

This keyword has no parameters.

You can specify only the location of the field, TIME, and optionally, the EDTCDE, EDTWRD, COLOR, DSPATR, or TEXT keyword. Positions 17 through 38 must be blank.

The edit word '0_:__:__' (_ represents a blank) is assumed for a TIME field. You can specify another edit word or one of the user-defined edit codes (5 through 9) to change the IBM-supplied editing.

Option indicators are not valid for this keyword, although option indicators can be used to condition the field with which it is specified.

Figure 3-195 shows how to specify the TIME keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  20                            1 56TIME
00020A  21                            1 56TIME
     A                                    EDTWRD('0 &HRS&  &MINS&  &SECS')
     A
```

Figure 3-195. Specifying the TIME Keyword

In Figure 3-195, the system time is 110645.

- If option indicator 20 is on, the time is displayed as:

  ```
  11:06:45
  ```

- If option indicator 21 is on (and option indicator 20 is off), the time is displayed as:

  ```
  11 HRS 06 MINS 45 SECS
  ```

UNLOCK (Unlock)

Use this record-level keyword to specify that the OS/400 program is to unlock the keyboard immediately after issuing an input operation to the record format you are defining. Without the UNLOCK keyword, the OS/400 program leaves the keyboard locked after reading the data on the display. The work station user cannot key data into input-capable fields while the data that has just been read is being processed.

The format of the keyword is:

```
UNLOCK[(*ERASE) | (*MDTOFF)] | [(*ERASE *MDTOFF)] |  [(*MDTOFF *ERASE)]
```

The parameter values *ERASE and *MDTOFF are optional. If you do not specify any parameter value, *ERASE is the default.

When your program sends an input operation, the following sequence of operations usually occurs:

1. The keyboard is unlocked (if it is not already unlocked) to allow the work station user to key into input-capable fields on the display.
2. The work station user presses the Enter key (or a valid function key).
3. Modified data tags (MDTs) for input-capable fields in the record format are set on if they have been keyed into or if they were displayed with the DSPATR(MDT) keyword in effect.
4. When the input operation is completed, the parameter values for UNLOCK affect the input-capable fields with MDTs set on as described in the following sections.

UNLOCK (without GETRETAIN) or UNLOCK(*ERASE): The keyboard remains unlocked, input-capable fields on the display are erased, and their MDTs remain on following the input operation.

UNLOCK(*MDTOFF) or UNLOCK (with GETRETAIN): The keyboard remains unlocked, input-capable fields on the display are not erased, and their MDTs are set off following the input operation.

UNLOCK(*ERASE *MDTOFF) or UNLOCK(*MDTOFF *ERASE): The keyboard remains unlocked, input-capable fields on the display with their MDTs set on are erased, and their MDTs are set off following the input operation.

The GETRETAIN keyword is ignored and an error message results at file creation time if the GETRETAIN keyword is specified with UNLOCK(any parameter).

Note: This keyword does not prevent your program from issuing an output operation immediately after an input operation. However, the keyboard is unlocked and the work station user could be keying input data when the output operation changes the display.

Option indicators are not valid for this keyword.

Figure 3-196 shows how to specify the UNLOCK keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                   UNLOCK(*ERASE)
00020A            FLD1          4   B  2  2
00030A            FLD2          4   B  3  2
     A
00040A          R RECORD2                   UNLOCK(*MDTOFF)
00050A            FLD21         4   B  4  2
00060A            FLD22         4   B  5  2
     A
00070A          R RECORD3                   UNLOCK(*ERASE *MDTOFF)
00080A            FLD31         4   B  6  2
00090A            FLD32         4   B  7  2
     A
```

Figure 3-196. Specifying the UNLOCK Keyword

USER (User)

Use this field-level keyword to display the user profile name for the current job as a constant (output-only) field that is 10 characters long. You can specify the location of the field, the USER keyword, and, optionally, the COLOR, DSPATR, and TEXT keywords. Positions 17 through 38 must be blank.

This keyword has no parameters.

Option indicators are not valid for this keyword, although option indicators can be used to condition the field on which it is specified.

Note: For a System/36 environment Multiple Requester Terminal (MRT) job, the displayed user profile name is the same user profile name of the interactive job for the display station where the display file is shown.

Figure 3-197 shows how to specify the USER keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A  10                              2 12'USER:'
00030A  10                              2 20USER
00040A                                    DSPATR(HI)
00050A N10                             15 18'USER:'
00060A N10                             15 26USER
00070A                                    DSPATR(HI)
     A
```

Figure 3-197. Specifying the USER Keyword

In Figure 3-197, if indicator 10 is on, the user name is displayed on line 2 starting in column 20. If indicator 10 is off, the user name is displayed on line 15 starting in column 26.

USRDFN (User-Defined)

Use this record-level keyword to specify that the data for this record is in the form of a user-defined data stream.

This keyword has no parameters.

No fields are valid for this record because the data stream formats the display. No file- or record-level keywords apply to this record except INVITE, KEEP, PASSRCD, HLPRTN, HELP, HLPCLR, PRINT, OPENPRT, and TEXT. However, the HELP, HLPRTN, and INVITE keywords will only apply if they are specified on this record. They will not apply if they are specified at the file-level. Help specifications are valid for this record. Once you do an output operation for this record, the OS/400 program no longer holds knowledge of the status of the records on the device. You should have in-depth knowledge of the device before using this keyword.

Option indicators are not valid for this keyword.

Figure 3-198 on page 3-232 shows how to specify the USRDFN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00020A          R USRREC                    USRDFN
00030A
     A
```

Figure 3-198. Specifying the USRDFN Keyword

USRDSPMGT (User Display Management)

Use this file-level keyword to specify that all data written to the display is held until overwritten or cleared by using the CLRL keyword.

This keyword has no parameters.

Refer to Appendix F, "System/36 Environment Considerations," for information on how to specify the USRDSPMGT keyword.

USRRSTDSP (User Restore Display)

Use this record-level keyword on a window record to specify that the application will manage the display. Window records are not automatically removed. If this keyword is not specified, the system saves and restores the underlying display when a window record is displayed.

This keyword has no parameters.

The WINDOW keyword must be specified on the same record as the USRRSTDSP keyword. The USRRSTDSP keyword functions only when the window keyword defines a window. The USRRSTDSP keyword does not function if the window keyword specified a record format name. For more information, see "WINDOW (Window)" on page 3-238.

Option indicators are valid for this keyword.

Figure 3-199 shows how to specify the USRRSTDSP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R APPRCD
     A            FIELD1        10A  0  5 40
     A            FIELD2         5S 0B  6 45
     A          R WINDOW1                   WINDOW(6 15 9 30)
     A  25
     A            FIELD1         5A  B  2  2
     A            FIELD2        20S  B  8  5
     A
```

Figure 3-199. Specifying the USRRSTDSP Keyword

In this example, suppose APPRCD is already on the display. If indicator 25 is set on when WINDOW1 is written to the display, the system does not save the underlying display (which contains APPRCD). When the user exits WINDOW1, the application must restore the underlying display, possibly by rewriting APPRCD to the display.

Note: With USRRSTDSP, there is no limit on the number of windows. The limit is 12 without using the USRRSTDSP keyword.

VALUES (Values)

Use this field-level keyword to specify a list of values that are valid for the user to type into the field. The OS/400 program performs an implied equals test on the data typed in against the value(s) you specify here. Note that the OS/400 program performs this checking only if the field is changed by the work station user or if its modified data tag (MDT) is set on using DSPATR(MDT).

Note: Refer to the CHKMSGID keyword for information on defining user-specified error messages.

The format of the keyword is:

```
VALUES(value-1 [value-2... [value-100]])
```

There can be 1 to 100 values; specify them as parameter values with the keyword and separate them by at least one blank.

Note: You cannot specify more than 5000 characters in a single DDS statement. Therefore, you cannot specify character values that cause VALUES to be longer than 5000 characters. If you specify other keywords for the same field, they also count toward the 5000-character limit. For example, specifying DSPATR(HI) for the field reduces the number of characters left for VALUES.

A value can be a numeric or a character value, corresponding in length to the field that is to be tested. A character value must be enclosed in apostrophes. A numeric value is restricted to the digits 0 through 9 and can be preceded by a minus sign (-) for negative values. Alignment is on the low-order character position.

Defining a Numeric Field

When a work station user types in data, the OS/400 program aligns the entered characters according to the number of decimal positions in the field. Leading and trailing blanks are filled with zeros when the field is passed to your program. If no decimal character is typed, the OS/400 program places a decimal character to the right of the farthest right character typed in. For example, for a numeric field with a length of 5 (specified in position 34) and 2 decimal positions (specified in position 37), 1.2 is interpreted as 001.20, and 100 is interpreted as 100.00.

The compare is based on the value as it is passed to your program (for example, right-justified and padded or left-adjusted and padded).

You cannot specify VALUES on a floating-point field (F in position 35).

Option indicators are not valid for this keyword.

Figure 3-200 shows how to specify the VALUES keyword for a character and numeric values test.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A           RESPC         11   I  8  2VALUES('A' 'B' 'C' 'D')
00020A           RESPN         31  0I  9  2VALUES(33 -42 01)
00030A           DECFLD         1  2I 11  2VALUES(1.2 100)
     A
```

Figure 3-200. Specifying the VALUES Keyword

VLDCMDKEY (Valid Command Key)

Use this file- or record-level keyword to specify that the OS/400 program is to set on the specified response indicator when any valid command key other than the Enter key is pressed by the work station user. One use of this function is to perform a simple test to determine if the work station user has requested a function you want to watch for in your program.

Refer to Appendix F, "System/36 Environment Considerations," for information on how to specify the VLDCMDKEY keyword in files that are used in the System/36 environment.

The format of the keyword is:

```
VLDCMDKEY(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation to explain the intended use of the indicator. This text's only function in the file or the program is a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

For a command key to be considered valid, you must have activated the key by specifying it with one of the following keywords:

Keyword	Comments
ALTHELP(CAnn)	With or without response indicator on the HELP keyword. Causes the command attention key specified to be considered a valid command key.
ALTPAGEUP(CFnn)	With or without response indicator on the PAGEUP keyword. Causes the command function key specified to be considered a valid command key if PAGEUP is also specified.
ALTPAGEDWN(CFnn)	With or without response indicator on the PAGEDOWN keyword. Causes the command function key specified to be considered a valid command key if PAGEDOWN is also specified.
CAnn	With or without response indicator.
CFnn	With or without response indicator.
CLEAR	With or without response indicator.
HELP	Valid only when HELP key is passed back to application as follows: • HELP and HLPRTN (with or without a response indicator). • HELP (with or without a response indicator) and no help areas are defined for any records currently being displayed.
HOME	With or without response indicator.
PAGEDOWN	With or without response indicator.
PAGEUP	With or without response indicator.

PRINT	Valid only when PRINT key is passed back to application as follows: • PRINT (with a response indicator). • PRINT(*PGM).
ROLLUP	With or without response indicator.
ROLLDOWN	With or without response indicator.

Option indicators are not valid for this keyword.

Figure 3-201 shows how to specify the VLDCMDKEY keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R REC1                      VLDCMDKEY(90 'Any valid key')
00020A                                      CA01(91)
00030A                                      CA02(92)
00040A                                      CA03(93)
00050A                                      CLEAR(94)
     A
```

Figure 3-201. Specifying the VLDCMDKEY Keyword

In Figure 3-201, Indicator 90 is set on if any of four keys (CA01, CA02, CA03, or Clear) is pressed.

WDWBORDER (Window Border)

Use this file- or record-level keyword to specify the color, display attributes, and characters used to form the border of a window.

The format of the keyword is:

```
WDWBORDER([color] [display-attribute] [characters])
```

At least one parameter must be specified.

The color parameter specifies the color of the border characters on a color display station (3179 Models C1 and C2, 5292 Color display stations only, or 5555 Models C01 and F01). The parameter is specified as an expression of the form (*COLOR value).

The valid values for the color parameter are:

Value	Meaning
BLU	Blue
GRN	Green
WHT	White
RED	Red
TRQ	Turquoise
YLW	Yellow
PNK	Pink

If the color parameter is not specified, the default is BLU. This parameter is ignored if it is specified for a window on a monochrome display.

The display-attribute parameter specifies the display attributes of the border characters. The parameter is specified as an expression of the form (*DSPATR [value1 [value2 [value3...]]]). If more than one DSPATR value is used, they are combined to form one DSPATR that is used for the entire border.

The valid values for the display-attributes values are:

Value	Meaning
BL	Blink
CS	Column separator
HI	High intensity
ND	Nondisplay
RI	Reverse image
UL	Underline

There is no default for display-attributes.

Note: Display-attributes CS, HI, and BL can cause fields on 5292, 3179, and 3197 Model C1 and C2 display stations to appear as color fields. Display-attributes HI, RI, and UL cause a border not to be displayed. See "COLOR (Color)" on page 3-72 for more information.

The characters parameter specifies the characters that make up the border. The parameter is specified as an expression of the form (*CHAR 'border-characters'). The border character value is an 8-character string that defines the border characters in the following order:

```
top-left-corner
top-border
top-right-corner
left-border
right-border
bottom-left-corner
bottom-border
bottom-right-corner
```

If this parameter is not specified, the default border characters are period (.) for the upper-left and right corners and the top and bottom borders, colon (:) for the left and right borders and lower-left and right corners. Although any displayable character can be specified as a border character, it is recommended that you use invariant characters. The following table shows the invariant characters:

Figure 3-202. Character Set for System Data

Hexadecimal	Character	Description
40		Blank
4B	.	Period
4C	<	Less than sign
4D	(	Left parenthesis
4E	+	Plus sign
50	&	Ampersand
5C	*	Asterisk
5D	)	Right parenthesis
5E	;	Semicolon
60	-	Minus sign
61	/	Slash
6B	,	Comma
6C	%	Percent sign
6D	_	Underline
6E	>	Greater than sign
6F	?	Question mark
7A	:	Colon
7D	'	Single quotation mark
7E	=	Equal sign

Note:

In addition, you can use any of the following:

- Uppercase English alphabetic characters: A through Z
- Numeric characters: 0 through 9

If the WDWBORDER keyword is specified at the record level, the WINDOW definition keyword must also be specified on the same record. If a WINDOW keyword that references another window is also specified, a warning message is issued.

Option indicators are valid for this keyword.

You can specify more than one WDWBORDER on a record. If you specify the WDWBORDER keyword more than once at the file level or at a record level, the parameters for the keywords that are in effect are combined on the same level. If different values are specified for the same parameter, the parameter value of the first keyword is used.

If the WDWBORDER keyword is specified both at the file level and on a window or pull-down definition record, the parameter values defined at both levels are combined. If different values are specified for the same parameter, the parameter value at the record level is used.

Figure 3-203 on page 3-238 shows how to specify the WDWBORDER keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A  01                                  WDWBORDER((*COLOR PNK) +
     A                                      (*DSPATR BL))
     A          R RECORD1                   WINDOW(6 15 9 30)
     A N01                                  WDWBORDER((*COLOR GRN))
     A  01                                  WDWBORDER((*COLOR RED))
     A            FIELD1         5A  B  2  2
     A            FIELD2        20A  B  8  5
     A          R RECORD2                   WINDOW(8 20 9 30)
     A                                      WDWBORDER((*COLOR YLW) +
     A                                       (*DSPATR RI))
     A            FIELD3         5A  B  2  2
     A            FIELD4        20A  B  8  5
     A            LINE           2S 0P
     A            POS            2S 0P
     A          R RECORD3                   WINDOW(&LINE &POS 9 30)
     A                                      WDWBORDER((*CHAR +
     A  02                                   '+-+||+-+'))
     A            FIELD3         5A  B  2  2
     A            FIELD4        20A  B  8  5
     A            LINE           2S 0P
     A            POS            3S 0P
     A
```

Figure 3-203. Specifying the WDWBORDER Keyword

If the window defined by RECORD1 is written to the display with indicator 01 set off, it has a green border constructed of colons for the vertical borders and periods for the horizontal borders. If indicator 01 is set on, the window has a blinking red border.

When the window defined by RECORD2 is written to the display, it has a yellow border displayed in reverse image constructed of the default border characters.

When the window defined by RECORD3 is written to the display, the following appears:

- If indicator 02 is set on, and indicator 01 is off, the window has a blue border constructed of dashes for the top and bottom borders, vertical bars for the left and right borders, and plus signs for the corners.
- If indicator 02 is set off, and indicator 01 is set on, the window has a pink border.
- If indicators 01 and 02 are on, the window has a pink border constructed of dashes for the top and bottom borders, vertical bars for the left and right borders, and plus signs for the corners.

WINDOW (Window)

Use this record-level keyword to specify that the record format you are defining will be displayed using a window. A window is information that overlays part of the display. A window is usually smaller than the actual work station display, and can be positioned anywhere on the display.

The WINDOW keyword has two formats that can be used. These formats do the following:

- Define a window by specifying the location and size of a window; this is known as a window definition record.

- Refer to a record format name where the window location and size have been defined; this is known as a window reference record.

The format for the keyword is one of the following:

```
WINDOW(start-line | &start-line-field start-position
       | &start-position-field window-lines window-positions
       [*MSGLIN | *NOMSGLIN])
                  or
WINDOW(*DFT window-lines window-position [*MSGLIN | *NOMSGLIN])
```

Specify this format of the WINDOW keyword to define a window. The record format you are defining is displayed in this window. Up to 12 windows can be shown on the display at one time. You can define more than 12 windows in DDS, but only 12 can be displayed at the same time. However, if USRRSTDSP is specified, the number of windows is unlimited. All fields defined in this record must fit within the window.

The parameters specify:

- The number or the name of a field containing the number of the line that is to contain the upper-left corner of the window border. If a field name is specified, the field must exist in the record format and the field must be defined as a signed numeric (data type S) and program-to-system (usage P) field with length no greater than 3.
- The number or the name of a field containing the number of the position that is to contain the upper-left corner of the window border. If a field name is specified, the field must exist in the record format and the field must be defined as a signed numeric (data type S) and program-to-system (usage P) field with length no greater than 3.
- The number of window-lines within the window. The window-lines can be no more than the available lines for the display size minus 2. This is because the upper and lower window borders each occupy one line.

  The last window-line in a window is used as the message line and cannot contain any fields. For example, if a WINDOW keyword is coded that specifies 10 window-lines for the window, only 9 of those lines can contain fields; the 10th line is the message line.
- The number of window-positions within the window. The window-positions can be no more than the available positions for the display size minus 4. This is because both right and left borders need an attribute byte inside the window. An attribute byte exists between the border character and the available window positions. For DBCS-capable windows, the system may need an additional 2 bytes on each side of the window for a shift-out character and shift-in character for any underlying DBCS fields.
- The MSGLIN parameter specifies if a window contains a message line. If this parameter is not specified, the default is *MSGLIN. *NOMSGLIN moves the message out of the window and places it at the bottom of the display or where the MSGLOC keyword defines the location. The last usable line in the window is reserved for error messages; no records are displayed there. If the error message is longer than the line, it is truncated to fit.

The special value, *DFT, specified in place of the start-line and start-position parameters, indicates that the system will determine the start line and start position

of the window. The window is positioned relative to the cursor location, similar to application help windows with variable starting locations. More information on the rules the system uses to position the window can be found in the *Guide to Programming Displays*.

The second format for the WINDOW keyword is:

```
WINDOW(record-format-name)
```

Specify this format of the WINDOW keyword to display the record format you are defining in a window that is defined on another record format.

The parameter specifies the record format name that has the window attributes specified. The record format that uses this parameter is displayed in the window defined on the referenced record.

The field locations specified within a record format with the WINDOW keyword are relative to the first usable window location in the upper-left corner of the window. The first usable window location is on the first line below the upper border and two positions to the right of the left border (an ending attribute byte occupies the first byte to the right of the border).

When a window is displayed, any records currently on the display are suspended if USRRSTDSP is not specified. The suspended records may be visible around the sides of the window. Input is allowed only within the active window. To remove the window from the display, a record can be written to an underlying window or a non-window record must be overlaid on the display.

The WINDOW keyword is not allowed on a record format that has any one of the following keywords specified:

ALWROL
ASSUME
MNUBAR
PULLDOWN
SFL
USRDFN

Note: The WINDOW keyword is allowed on a record with the SFLCTL keyword. This allows subfiles to be displayed within a window.

WINDOW cannot be specified for the record format specified by the PASSRCD keyword.

The ERRSFL keyword is ignored for records that have the WINDOW keyword specified.

The MSGLOC keyword is ignored for records that have the WINDOW keyword specified, unless NOMSGLIN is specified.

If a record format has both a WINDOW and WDWBORDER keyword specified, specify the start-line, start-position, window-lines, and window-positions parameters on the WINDOW keyword. The WINDOW keyword should not specify the record-format-name parameter.

Option indicators are not valid for this keyword. However, display size condition names can be used.

Figure 3-204 on page 3-241 shows how to specify the WINDOW keyword to define a window.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A         R WINDOW1                   WINDOW(4 20 9 30)
     A           FIELD1         8A  B  5 10
     A           FIELD2        10A  B  6 10
     A         R WINDOW2                   WINDOW(*DFT 9 30 *NOMSGLIN)
     A
```

Figure 3-204. Specifying the WINDOW Keyword (Example 1)

When the WINDOW1 record is displayed, the upper-left corner of the window border is on line 4 position 20 of the display. The lower-right corner of the border is located 10 lines lower than the upper border and 33 positions to the right of the left border.

- Lower border line = upper border line + window-lines + 1
- Right border position = left border position + window-positions + 3

The FIELD1 field starts 2 lines lower than the upper border and 11 positions (the ending attribute byte for the border character has been taken into account) to the right of the left border character (line 6, position 31 on the display).

- Actual field line = upper border line + line number of field
- Actual field position = left border position + position of field + 1

The FIELD2 field starts 6 lines lower than the upper border and 11 positions to the right of the left border (line 10, position 31 on the display).

When the WINDOW 2 record is displayed, the upper-left corner of the window border is at the cursor position during run time. The message line does not appear inside the window, it appears at the bottom of the display.

Figure 3-205 shows how to use the WINDOW keyword to display multiple records in the same window.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8

     A         R WINDOW1                   WINDOW(&LINE &POS 9 30)
     A           USERI  D       8A  0  2 10
     A           LINE           2S 0P
     A           POS            3S 0P
     A
     A         R RECORD1                   WINDOW(WINDOW1)
     A                                     OVERLAY
     A           FIELD1         5A  B  7  2
     A           FIELD2        20A  B  8  5
     A
     A         R RECORD2                   WINDOW(WINDOW1)
     A           FIELD3        10A  B  2  2
     A           FIELD4         8A  B  8  5
     A           FIELD4         8A  B  8  5
     A
```

Figure 3-205. Specifying the WINDOW Keyword (Example 2)

When the WINDOW1 record is displayed, the upper-left corner of the border will be at the line and position numbers specified by the LINE and POS fields, respectively.

The lower-right corner of the border is located 10 lines lower than the upper border and 33 positions to the right of the left border.

The USERID field starts 2 lines lower than the upper border and 11 positions to the right of the left border character.

If RECORD1 (from the previous example) is displayed, it is placed within WINDOW1. Its fields are positioned with respect to the upper-left corner of the window. The fields from record WINDOW1 which were on the display remain because the OVERLAY keyword was used on RECORD1 and the two records do not overlap. For more information on the OVERLAY keyword, see the "OVERLAY (Overlay)" on page 3-169.

If RECORD2 (from the previous example) is displayed, it is also placed within WINDOW1. Its fields are positioned with respect to the upper-left corner of the window. Because the OVERLAY keyword was not used, the fields from records WINDOW1 and RECORD1 are removed from the window.

Figure 3-206 shows how to use the WINDOW keyword with a subfile.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R SFLDATA                    SFL
     A            NAME          20A  B  4  5
     A            RANK          10A  B  4 27
     A            SERIAL         8A  B  4 38
     A
     A          R WINDOW1                    SFLCTL(SFLDATA)
     A                                       WINDOW(8 25 10 50)
     A                                       SFLPAG(4)
     A                                       SFLSIZ(17)
     A                                       SFLDSP
     A                                       SFLDSPCTL
     A                                    2  5'Full Name'
     A                                    2 27'Rank'
     A                                    2 38'Serial Nbr'
     A
```

Figure 3-206. Specifying the WINDOW Keyword (Example 3)

When the WINDOW1 subfile control record is displayed, it and the subfile are displayed in a window. The upper-left corner of the window border is at line 8, position 25 on the display. The lower-right corner of the border is located on line 19, position 78.

The fields from both the subfile record and subfile control record are located with respect to the first usable window position in the upper-left corner of the window. For example, the NAME field in the SFLDATA record starts on the 4th window line and the 5th window position, which is the same as the 12th line on the display and the 31st position on the display.

Chapter 4. Printer Files

This chapter provides the following information regarding printer files:

- Definition
- Positional entries
- Keyword entries

"Positional Entries (Positions 1 through 44)" on page 4-2 gives rules and examples for filling in positions 1 through 44 of the data description specifications form for printer files. "Keyword Entries (Positions 45 through 80)" on page 4-11 gives the rules for specifying the DDS keywords for printer files. The keywords are presented in alphabetical order.

For more information about which DDS keywords are valid for any particular printer type, see the *Guide to Programming for Printing*.

Defining a Printer File

Specify the entries in the following order to define a printer file:

1. File-level entries (optional)
2. Record-level entries
3. Field-level entries

Specify at least one record format in each file. The maximum number of record formats in a printer file is 1024. The maximum number of fields in any one record format is 32 767. The maximum combined length of all named fields and indicators in a record format is 32 767 bytes.

Note: Specify the file name through the Create Printer File (CRTPRTF) command, not through DDS.

You can find an explanation of file level, record level, and field level in Chapter 1, "Introduction." You can find a complete printer file example in Appendix B, "Examples."

You can find syntax rules for specifying DDS keywords in "Syntax Rules" on page 1-4.

Figure 4-1 on page 4-2 shows a printer file coding example.

© Copyright IBM Corp. 1991, 1993

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* PRINTER FILE CODING EXAMPLE
00020A*
00030A          R TITLER                  SKIPB(3)
00040A            FLD1          40        47SPACEA(2) UNDERLINE
00050A  30        FLD2          40        47SPACEA(2) UNDERLINE
00060A*
00070A          R AUTHORR
00080A                                    66'by'
00090A            FIELD1        40        47SPACEB(1)
00100A                                    50DFT('Task Force I')
00110A  31                                  SPACEA(1)
00120A  31                                65'and'
00130A  31        FIELD2        40        47
00140A*
00150A          R PUBR                      SKIPB(58)
00160A                                    47'Published by Department'
00170A            DEPT           3  0     +1
00180A                                    47DATE  EDTCDE(Y)
00190A                                      SPACEB(1)
00200A N15
00210AO 32 33 34                          47TIME
00220A                                      SPACEB(1)
     A
```

Figure 4-1. Printer File Coding Example

Positional Entries (Positions 1 through 44)

This section describes how to specify the first 44 positions of the data description specifications form for printer files. To code the remaining part of the form, see "Keyword Entries (Positions 45 through 80)" on page 4-11.

Figure 4-1 shows some positional entries for printer files.

Sequence Number (Positions 1 through 5)

Use these positions to specify a sequence number for each line on the form. The sequence number is optional and is for documentation purposes only.

Form Type (Position 6)

Enter an A in this position to designate this as a DDS form. The form type is optional and is for documentation purposes only.

Comment (Position 7)

Enter an asterisk (*) in this position to identify this line as a comment. Use positions 8 through 80 for comment text. A blank line (no characters specified in positions 7 through 80) is also treated as a comment. Comment lines can appear anywhere in DDS and are kept only in the source file. Comments appear on the source computer printout but not on the expanded source computer printout.

Conditioning (Positions 7 through 16)

Use these positions to specify option indicators (2-digit numbers from 01 to 99). Then have your program set option indicators on (hex F1) or off (hex F0) to select a field or keyword.

By specifying two to nine indicators that must all be in effect (set off if N is specified; set on if N is not specified) before the field or the keyword is selected, you create an AND condition.

Note: You must specify the field or keyword on the same line as the last (or only) set of indicators you specified.

Specify up to nine indicators for each condition and nine conditions for each field or keyword. A maximum of 81 indicators can be specified for each keyword when nine indicators are used with nine conditions.

By specifying several conditions for a field or keyword, you create an OR relationship. If any one of the conditions is satisfied, the field or the keyword is selected.

Note: Conditions within the OR relationship can consist of one indicator or of several indicators ANDed together. Indicators can be ANDed to form a condition; conditions can be ORed to give your program several ways to select the field or the keyword.

Specify the conditions by entering the following values:

Position 7 (AND) If you need more than three indicators to form an ANDed condition, specify them in the same positions on the next line or lines of the DDS form. Specify A in position 7 on the following lines to designate the continuation of the AND condition, or leave it blank (A is the default).

Position 7 (OR) If you specify several conditions that are to be ORed together, each condition must start on a new line and each condition except the first must have an O in position 7. An O specified for the first condition produces a warning message (that position is assumed to be blank).

Positions 8, 11, 14 (NOT) If you want an indicator to be off instead of on to satisfy a condition, specify N in the position just preceding the indicator (position 8, 11, or 14).

Conditioning More than One Field or Keyword

If you condition a field, the field name (or the constant) and the last (or only) indicator must be on the same line. If you specify one or more keywords for the field, only the field is conditioned, not the keyword(s). If the field is not selected for an output operation, no keywords specified for that field are in effect, regardless of how the keywords are conditioned. For example, in Figure 4-1 on page 4-2 if indicator 30 is off, SPACEA and UNDERLINE are not in effect.

If you want to condition one or more keywords, specify the keywords and the last (or only) indicator on the same line. If the conditioning applies to keywords on multiple lines, keyword continuation must be used for the indicators to apply to all the keywords.

Type of Name or Specification (Position 17)

Enter a value in this position to identify the type of name in positions 19 through 28. The valid entries for printer files are:

Entry	Meaning
R	Record format name
Blank	Field name

Figure 4-1 on page 4-2 shows how to code the name type.

For more information on types of names, see "Name (Positions 19 through 28)."

Reserved (Position 18)

This position does not apply to any file type. Leave this position blank unless you use it for comment text.

Name (Positions 19 through 28)

Use these positions to specify record format names and field names.

Refer to "Syntax Rules" on page 1-4 for rules to use when specifying record or field names in DDS.

Name must begin in position 19.

Figure 4-1 on page 4-2 shows how to specify record format names and field names.

Record Format Name

When you specify an R in position 17, the name specified in positions 19 through 28 is a record format name. You can specify more than one record format for a printer file, but each name must be unique within that file. You must also specify field names or constant fields to complete a record format in a printer file.

Field Name

When position 17 is left blank, the name specified in positions 19 through 28 is a field name. Field names must be unique within the record format.

Constant Fields

Constant fields are unnamed fields (positions 19 through 28 must be blank). The following rules apply when specifying a constant field:

- Positions 17 through 38 must be blank.
- The location of the field is required (positions 39 through 44).
- The field can be conditioned using option indicators (positions 7 through 16).
- A constant field cannot be specified in a record format that has a BOX, ENDPAGE, GDF, LINE, OVERLAY, or PAGSEG keyword specified on the record, or a POSITION keyword specified on any of the fields specified within the record format.
- The constant itself is defined in positions 45 through 80 using one of the following entries:

  - Explicit DFT keyword (specify the value within apostrophes with the DFT keyword)
  - Implicit DFT keyword (specify the value within apostrophes without the DFT keyword)
  - DATE keyword (specify no value; see the DATE keyword description)
  - TIME keyword (specify no value; see the TIME keyword description)
  - PAGNBR keyword (specify no value; see the PAGNBR keyword description)
  - MSGCON keyword (specify the message description, the message file, the library name, and the length of the message description; see the MSGCON keyword description)
- The EDTCDE or the EDTWRD keyword can be specified for constant fields only when DATE, TIME, or PAGNBR keywords are also specified.

If you use line numbers you can specify the fields in any order. If you do not use line numbers, you must specify the fields in the order in which they are to appear on the printed page.

Reference (Position 29)

Specify R in position 29 to copy the attributes of a previously defined named field (the *referenced field*) to the field you are defining. (If you do not specify R, you must specify the field attributes.) For example, you might want to reference fields for an externally defined file to print a report from a database file.

When using the reference function, specify the referenced field name, even if it is the same as the referencing field. (The referenced field name can be in a previously created database file specified on the REF or REFFLD keywords.) The field attributes referenced are the length, data type, and decimal positions of the field, as well as the ALIAS, FLTPCN, TEXT, and editing keywords.

If the referenced field name is the same as the field you are defining, specify R in position 29 and the name of the field you are defining in positions 19 through 28. If the referenced field is different from the field you are defining, specify the name of the referenced field with the REFFLD keyword.

Position 29 must be blank at the file and record levels.

You can specify the name of the file defining the referenced field as a parameter value with the REF or the REFFLD keyword. See the REF and REFFLD keyword descriptions later in this chapter and in Appendix A, "When to Specify REF and REFFLD," for explanations of how the OS/400 program finds the referenced field.

If you do not want to copy all the attributes from the previously defined field, specify those attributes for the field you are defining, as follows:

- To override editing keywords (EDTCDE or EDTWRD), specify EDTCDE or EDTWRD for the field you are defining. To delete these keywords, specify DLTEDT for the field you are defining.
- Validity checking keywords (CHECK, COMP, RANGE, VALUES), if specified for the referenced field, are ignored in the printer file.

When you override some specifications, others are affected:

- If you specify a value for data type, field length, or decimal positions for the field you are defining, editing keywords are not copied from the referenced field.
- Packed decimal and binary fields are not supported for printer files. Therefore, when you reference fields of these types, the data type is converted to zoned decimal (S in position 35) in the printer file.

Note: Once the printer file is created, the referenced file can be deleted or changed without affecting the field definitions in the printer file. To incorporate changes made in the referenced file, delete and re-create the printer file.

Length (Positions 30 through 34)

Specify the field length for each named field (unless you copy it from a referenced field). Your entry represents the number of bytes of data to be passed from your program when an output operation is done for this field. (If the field is to be edited, the associated edit code or edit word is used to determine the printed length of the field.) Figure 4-1 on page 4-2 shows how to code the field length.

The maximum length of a zoned decimal field is 31. Data description specifications allow a maximum field length of 32 767 characters. If the field length causes the field to extend beyond the page size, a warning diagnostic appears. The maximum length of a single precision floating-point field is 9 digits; the maximum length of a double precision floating-point field is 17 digits.

If you use a referenced field, override the referenced length by specifying a new value or by specifying the increase or decrease in length. To increase the length, specify +n, where n is the increase. To decrease the length, specify –n, where n is the decrease. For example, an entry of +4 indicates that the field is to be 4 digits longer than the referenced field. The field length can be overridden without overriding the decimals.

If you specify length, it must be right-justified; leading zeros are optional.

Figure 4-2 shows correct and incorrect field-length specifications.

```
|...+....1....+....2....+....3....+....4....+....5
00010A            FIELD1       7
     A
00020A            FIELD2         7
     A
00030A            FIELD3    R   +7
```

Note: FIELD1 shows the field length specified incorrectly. FIELD2 and FIELD3 show the field length specified correctly.

Figure 4-2. Correct and Incorrect Field-Length Specifications for Printer Files

For floating-point fields, 7 positions will be added to the length you specify in positions 30 through 34. The 7 extra positions are for the significand sign, the decimal point or comma, the exponent character, the exponent sign, and the exponent.

In some cases, if you specify a value for length, some keywords specified with the field in the database file are not included in the printer file. For more information, see "Reference (Position 29)" on page 4-5.

Data Type (Position 35)

Use this position to specify the data type associated with the field. The valid entries for this field for a printer file are:

Entry	Meaning
S	Zoned decimal
A	Character
F	Floating point

Figure 4-1 on page 4-2 shows how to code the data type.

If you do not specify a data type, and do not duplicate one from a referenced field, the OS/400 program assigns a default value as follows:

- A (character) if the decimal positions (36 and 37) are blank
- S (zoned decimal) if the decimal positions (36 and 37) contain a number in the range 0 through 31

Notes:

1. Specify 0 in position 37 to indicate an integer numeric field.
2. Specify F in position 35 for a single precision floating-point field. Use the FLTPCN keyword to specify double precision or to change the precision of an already specified floating-point field.
3. A floating-point value consists of five parts: (a) the significand sign, (b) the significand, (c) the exponent character (d) the exponent sign, and (e) the exponent.

   The significand sign is not printed for a positive value. The number of digits in the significand is determined by the length specified (positions 30 through 34). The location of the decimal point or the comma is determined by the decimal positions specified (positions 36 and 37). The exponent character and the exponent sign are always printed. The exponent is always 3 digits.

   The printed length for a floating-point field is 7 positions greater than the length specified in positions 30 through 34. The 7 extra positions are for (a) the significand sign, (b) the decimal point or comma, (c) the exponent character, (d) the exponent sign, and (e) the 3 exponent digits.

Decimal Positions (Positions 36 and 37)

Use these positions to specify the decimal placement within a zoned decimal field and the data type of the field as it appears in your program.

If you leave these positions blank, the OS/400 program assumes a data type of character for the field.

If you enter a number in these positions, the OS/400 program assumes a data type of zoned decimal for the field. The number specified is the number of positions to the right of the decimal point; it must be less than or equal to the field length, with a maximum of 31. Figure 4-1 on page 4-2 shows how to code decimal positions.

If you use a referenced field, you do not need to specify decimal positions because the information is retrieved from the referenced file. You can override or change the decimal positions retrieved if you choose.

To override the decimal positions, specify the new value. To change the decimal positions, specify the amount you want to increase or decrease the field by and precede the amount with either a + or –. For example, an entry of +4 indicates there are 4 more digits to the right of the decimal point than in the referenced field.

Note: High-level languages can impose specific length and value restrictions on the decimal positions. Observe these restrictions for files used by those languages.

Usage (Position 38)

Use this position to specify that a named field is an output-only or program-to-system field. Do not make an entry in this position for a constant (unnamed) field.

The valid entries for printer files are:

Entry	Meaning
blank or O	Output only
P	Program-to-system (special output field)

Output-only fields pass data from a program to the printer when the program prints a record.

A program-to-system field is a named, numeric, or alphanumeric output-only field used to pass data between the program and the system. It is not printed. A program can send data to the field with an output operation, but the data is not printed.

The following rules apply to program-to-system fields:

- The field is always named.
- Locations are not valid.
- Length, data type, and decimal positions are specified as for other named fields.
- The program-to-system field must be specified as a parameter on a PAGSEG, OVERLAY, GDF, LINE, or POSITION keyword within the same record format. The program-to-system field is not a valid parameter on any other keyword. A severe error message is issued if the field is not specified on at least one of these keywords.
- Program-to-system fields can appear anywhere in the buffer.
- The only valid keywords for a program-to-system field are ALIAS, INDTXT, REFFLD, and TEXT.

Location (Positions 39 through 44)

Use these positions to specify where the beginning of the field you are defining appears on the page. You specify the line (positions 39 through 41) and the position (positions 42 through 44). The following conditions apply:

- When line numbers are specified, fields can appear in any order. They will be sequenced again into line-position order when placed in the printer file.
- When line numbers are not specified, the field order within the printer file is the same as that specified in the DDS.

- When fields or space/skip keywords are conditioned, the data description processor treats them as if they were selected when diagnosing overlapping fields.
- For a record format with several fields, when a field that has skip/space keywords is conditioned or a field-level skip/space keyword is conditioned, a warning message appears indicating that overlapping fields could occur and not be diagnosed.
- When the page size is exceeded because of field length, location, or associated skip/space keywords, or because of a combination of these, a warning message appears.
- The maximum that can be entered for line number is 255. The actual maximum for the page may be less depending on the page-length value of the PAGESIZE parameter on the Create Printer file (CRTPRTF) command and on the lines per inch specified.
- The maximum value that can be entered for position number is 255. The actual maximum for the page depends on the page-width value of the PAGESIZE parameter on the Create Printer File (CRTPRTF) command and on the characters per inch, either specified or implicit in the font being used.
- The overflow line (the last printed line on a page) depends on the values of OVRFLW and PAGESIZE parameters on the Create Printer File (CRTPRTF), Change Printer File (CHGPRTF), and Override with Printer File (OVRPRTF) commands. For externally defined files, RPG cannot control page overflow.
- When externally defined files are used by high-level language compilers, the fields are sequenced in the output record area according to the DDS. Refer to the appropriate high-level language manual for specific information. If fields overlap, the printer overprints. See the expanded source in the compiler printout generated by the Create Printer File command for field lengths and output buffer positions.
- These positions must be blank if the POSITION keyword is specified.

Line (Positions 39 through 41): These positions specify the line on the page in which the field begins. The entry must be right-justified; leading zeros are optional. Line numbers can be specified for either named or unnamed fields within a record. If you specify a line number for one field within a record, you must specify either a line number in positions 39 through 41 or a plus value (+n) in positions 42 through 44 for all fields within that record.

Line numbers are not valid if one of the skip or space keywords has been specified at either the record level or field level. Line numbers are valid, however, if one of the skip keywords has been specified at the file level. (Space keywords are not valid at the file level.)

Position (Positions 42 through 44): Specify the starting position of the field. The position you specify is based on the value of the characters per inch value for the printer file, which is either specified or implicit in the font being used. If the printer file uses a font character set or coded font, the implied value 10 CPI is used. The entry must be right-justified; leading zeros are optional.

If you specify a location of a field in a record and the field is not ignored, you can specify the location of subsequent fields within that record by leaving the line number blank and specifying a plus value (+n) for 42 through 44 (position entry).

The plus value indicates the number of spaces to be left between the end of the previous field and the beginning of the field you are defining. The plus value must be in the range of 0 through 99. If you specify a plus value, the line number entry must be blank. If the plus value causes an implicit space operation, and line numbers are not being used for the record format, then skip/space keywords must be used to cause spacing to occur. Figure 4-3 illustrates this problem and two possible solutions (for a page width of 132 characters).

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A* POSITION PLUS VALUE CAUSES PRFLD1 TO OVERLAP PRFLD2
     A*
     A          R PRTOUT                    SKIPB(1)
     A            PRFLD1       130         1TEXT('START LOC 1,1 END LOC 1,130')
     A            PRFLD2       130        +2TEXT('OVERLAPS PRFLD1')
     A*
     A* SOLUTION 1 TO PREVENT OVERLAP IS TO SPECIFY SPACEA OR SKIPA WITH PRFLD1
     A*   OR TO SPECIFY SPACEB OR SKIPB WITH PRFLD2
     A*
     A          R PRTOUT2                   SKIPB(1)
     A            PRFLD1A      130         1
     A            PRFLD2A      130        +2SPACEB(1)
     A*
     A* SOLUTION 2 PROVIDES A FUNCTIONAL EQUIVALENT NOT USING SKIP/SPACE
     A*
     A          R PRTOUT3
     A            PRFLD1B      130      1  1
     A            PRFLD2B      130        +2
     A
```

Figure 4-3. Specifying the Line and Position Location

If FOLD(*YES) is specified for the CRTPRTF, CHGPRTF, or OVRPRTF command, any field that extends beyond the end of a line is continued on the next line. The break occurs at the end of the line but you can cause it to be folded at a blank by specifying the BLKFOLD keyword. If FOLD(*NO) is in effect, a field that extends beyond the end of a line is truncated.

The data description specifications determine which fields appear on the same print line. The data description processor diagnoses overlap at file creation time. Keywords or fields containing optional keywords are assumed to be selected. Therefore, no overlap check is made for the cases where the keywords or fields are not selected. In Figure 4-4 on page 4-11, no fields would overlap unless indicator 01 is on, in which case F1, F3, and F4 would overlap. A diagnostic is sent for the format indicating that field selection or conditioning of space/skip keywords can cause fields to overlap at run time.

On some printers, printer throughput speed is better when fields on the same line are specified in the DDS in right-to-left order.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A* OVERLAPPING FIELDS ONLY IF IND 01 IS ON
     A*
     A          R REC1                      SKIPB(1)
     A            F1            1         1
     A NO1        F2            1         1SPACEB(1) SPACEA(1)
     A            F3            1         1
     A            F4            1         1
     A NO1                                  SPACEB(1)
     A
```

Figure 4-4. Overlapping Fields

Keyword Entries (Positions 45 through 80)

This section contains the valid keyword entries for defining printer files. The keywords are entered in positions 45 through 80 (functions). See "Syntax Rules" on page 1-4 for a discussion of the general rules for specifying keywords.

The following keywords are valid for printer files:

ALIAS	EDTWRD	PAGSEG
BARCODE	ENDPAGE	POSITION
BLKFOLD	FLTFIXDEC	PRTQLTY
BOX	FLTPCN	REF
CDEFNT	FNTCHRSET	REFFLD
CHRID	FONT	SKIPA
CHRSIZ	GDF	SKIPB
COLOR	HIGHLIGHT	SPACEA
CPI	INDARA	SPACEB
CVTDTA	INDTXT	TEXT
DATE	LINE	TIME
DFNCHR	LPI	TRNSPY
DFT	MSGCON	TXTRTT
DLTEDT	OVERLAY	UNDERLINE
DRAWER	PAGNBR	
EDTCDE	PAGRTT	

ALIAS (Alternative Name)

Use this field-level keyword to specify an alternative name for a field. When the program is compiled, the alternative name is brought into the program instead of the DDS field name. The high-level language compiler in use determines if the alternative name is used. Refer to the appropriate high-level language reference manual for information about ALIAS support for that language.

The format of the keyword is:

```
ALIAS(alternative-name)
```

Refer to the "Syntax Rules" on page 1-4 for ALIAS naming conventions.

The alternative-name parameter must be different from all other alternative names and from all DDS field names in the record format. If a duplicate name is found, an error appears on the field name or alternative name.

An alternative name cannot be used within DDS or any other OS/400 function (for example, as a key field name, as the field name specified for the REFFLD keyword, or as a field name used in the Copy File (CPYF) command).

When you reference a field that has the ALIAS keyword, the ALIAS keyword is copied in unless the ALIAS keyword is explicitly specified on the referencing field.

Option indicators are not valid for this keyword.

Figure 4-5 shows how to specify the ALIAS keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00070A            FIELDA        25A     1  2ALIAS(CUSTOMERNAME)
     A
```

Figure 4-5. Specifying the ALIAS Keyword

In Figure 4-5, the alternative name for FIELDA is CUSTOMERNAME.

BARCODE (Bar Code)

Use this field-level keyword to print a field as a user-specified bar code. BARCODE is valid only for Intelligent Printer Data Stream* (IPDS*) printers and only for printer files with device type *IPDS or *AFPDS specified.

The format of the keyword is:

```
BARCODE(bar-code-ID [height] [[*HRZ | *VRT]
[*HRI | *HRITOP | *NOHRI] [*AST | *NOAST]
[check-digit] [unit-width]
[wide/narrow-ratio]])
```

The bar-code-ID parameter is required. Valid values for the bar code ID are listed in Figure 4-6 on page 4-13.

The height parameter is optional, but if it is specified, it must be the second parameter following the keyword. Valid values for the bar code height are 1 through 9 lines. The value you specify for the bar code height does not include the human readable interpretation below the bar code. If you do not specify the height parameter, the printer uses a default height.

You can specify the last 6 parameters (all optional) in any order. Using these parameters, you can specify that BARCODE:

- Print the bar code horizontally or vertically. The default is horizontal printing (*HRZ).
- Include or exclude the human readable interpretation of the bar code. The default is to include the human readable interpretation printed at the bottom of the bar code (*HRI).
- Indicate that the human readable interpretation should be printed at the top (*HRITOP) of the bar code. (Check individual printer manuals for different bar code support of *HRITOP.)
- Include or exclude the asterisk on CODE3OF9 bar codes. The default is to exclude the asterisk (*NOAST).

- Select the check digit type. This is a 1-character hex value that cannot be hex FF.
- Specify the width (in inches) of the narrow bar/space. It is specified as an expression of the form (*WIDTH value). For more information on how to specify expressions, see "Syntax Rules" on page 1-4. The valid values for the parameter are 0.007 through 0.208.

  Note: This parameter is ignored for the 4234 and 4224 printers.
- Specify the ratio of the wide bar/space to the narrow bar/space. It is specified as an expression of the form: (*RATIO value). The valid values for the parameter are 2.00 through 3.00.

  Note: This parameter is ignored for the 4234 and 4224 printers.

If you specify an optional parameter that does not apply to the bar code ID you have specified, the printer ignores the optional parameter.

If you attempt to print a bar code on a printer that does not support bar codes, the digits in the code are treated as text, and a diagnostic message results stating that the bar code could not print.

The line and position you specify for the field is used as the upper left corner of the bar code. Because the line specified in the DDS is the base line (the imaginary line on which characters are printed) and this is used as the upper edge of the bar code, the bar code appears to extend down from the bottom of the line you specify.

Figure 4-6 describes valid data types and field lengths for the BARCODE field.

Figure 4-6. Valid Bar Code Definitions

Bar Code ID	Data Type	Field Length
MSI	S	1 through 31[1]
UPCA	S	11
UPCE	S	10
UPC2	S	2
UPC5	S	5
EAN8	S	7
EAN13	S	12
EAN2	S	2
EAN5	S	5
CODEABAR	A	1 through 50
CODE128	A	1 through 50
CODE3OF9	A	1 through 50
INTERL2OF5	S	1 through 31
INDUST2OF5	S	1 through 31
MATRIX2OF5	S	1 through 31
POSTNET	S	1 through 31

Note:

1. The 4234 Printer only supports 14 digits.

Figure 4-7 describes the supported bar codes.

Figure 4-7. Bar Codes Supported by DDS

BARCODE	Digits per Code	Range of Characters Allowed	Default Check Digit(s) Generated	Default Check Digit(s) Printed	Valid Check Digit(s)
MSI (changed Plessey)	31[1]	0 through 9	2 Modulus 10	No	01 through 09
UPC-A	11	0 through 9	1	No	00
UPC-E	10	0 through 9	1	No	00
UPC-2 digit add on (must follow a UPC A or E bar code)	2	0 through 9	No	No	00
UPC-5 digit add on (must follow a UPC A or E bar code)	5	0 through 9	No	No	00
EAN-8	7	0 through 9	1	Yes	00
EAN-13	12	0 through 9	1	Yes	00
EAN-2 digit add on (must follow an EAN 8 or 13 bar code)	2	0 through 9	No	No	00
EAN-5 digit add on (must follow an EAN 8 or 13 bar code)	5	0 through 9	No	No	00
INDUST2OF5 or industrial 2 of 5	31	0 through 9	1	Yes	01 02
MATRIX2OF5 or matrix 2 of 5	31	0 through 9	1	Yes	01 02
INTERL2OF5 or interleaved 2 of 5	31	0 through 9	1	Yes	01 02
CODEABAR	Up to 50 characters	0 through 9, A through D (begin/end only), -, ., $, /, +, and :	1	Yes	01 02
CODE128	Up to 50 characters	Refer to Appendix G, "CODE128 Character Set"	1	No	01 02
CODE3OF9 or code 3 of 9	Up to 50 characters	0 through 9, A through Z (upper case only), -, ., $, /, +, %, and a blank	No	No	01 02
POSTNET	Up to 31 characters	0 through 9	1	Yes	(Ignored)

Note:

1. The 4234 Printer only supports 14 digits.

CODEABAR field data must begin with an A, B, C, or D and must end with an A, B, C, or D. For example, A11224455C or D33447799D.

Do not specify BARCODE in the same field with the CHRSIZ, CHRID, CVTDTA, DATE, EDTCDE, EDTWRD, FONT, HIGHLIGHT, PAGNBR, TIME, or UNDERLINE keywords.

See the CVTDTA keyword for information on coding IPDS bar code commands yourself.

If you specify CHRSIZ at the record level, it applies to all fields in that record. If you specify BARCODE in one of those fields, the BARCODE keyword is not allowed.

You cannot specify BARCODE on the same record format with BLKFOLD, CPI, or DFNCHR.

When you specify BARCODE on a numeric field, the number of decimal positions must be zero.

When you specify BARCODE on a constant field, the only valid bar code IDs are CODEABAR, CODE128, and CODE3OF9, and you must also specify the DFT keyword either implicitly or explicitly.

You should specify DEVTYPE (*IPDS) or DEVTYPE(*AFPDS) on the CRTPRTF command when BARCODE is specified in the file.

BARCODE is allowed only on data types S and A (see Figure 4-7 on page 4-14 for restrictions).

Option indicators are not valid for this keyword.

User specified check digits are not validity checked and could cause bar code errors if they are not valid. The *Intelligent Printer Data Stream Reference* manual contains more information on bar codes and valid check digits.

Figure 4-8 shows how to specify the BARCODE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A           FIELD1        11S  0  2  4BARCODE(UPCA 6)
     A           FIELD2         3A    10  6BARCODE(CODE30F9 4 *NOHRI-
     A                                     *AST X'02')
     A           FIELD3        10S  0  4  5BARCODE(UPCE 6 (*RATIO 2.75) *HRZ +
     A                                     X'00' (*WIDTH .02))
     A           FIELD4        10A  0  5  5BARCODE(CODEABAR 1 (*RATIO 2.1) +
     A                                     *HRITOP)
     A                                  6  5'01234567'
     A                                     BARCODE(CODE128 2 *HRITOP *HRZ +
     A                                     (*WIDTH 0.1) (*RATIO 2) X'01')
     A
```

Figure 4-8. Specifying the BARCODE Keyword

BLKFOLD (Blank Fold)

Use this field-level keyword for named fields that overflow onto subsequent print lines, to cause folding to occur at a blank rather than at the end of a line. If the blank fold keyword is not specified, the line folds at the end of the physical print line.

This keyword has no parameters.

BLKFOLD has effect only if you specify FOLD(*YES) on the CRTPRTF, CHGPRTF, or OVRPRTF command. If you specify FOLD(*NO), BLKFOLD has no effect until a CHGPRTF or OVRPRTF command with FOLD(*YES) is issued.

When you use BLKFOLD, the field length is not increased; therefore, a portion of the output data might be truncated.

You cannot specify BLKFOLD on a floating-point field (F in position 35).

Use BLKFOLD only with SCS printers. A warning message results at create time if BLKFOLD is specified in a file created with DEVTYPE(*IPDS) or DEVTYPE(*AFPDS). You cannot specify BLKFOLD on the same record format with IPDS printer keywords or keywords that support Advanced Function Printing* (AFP*). If any format in the file contains a combination of SCS and IPDS printer keywords or AFP-support keywords, the file is not created.

Option indicators are not valid for this keyword. However, option indicators can be used to condition the field (whether named or constant) associated with this keyword.

Figure 4-9 shows how to specify the BLKFOLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A           FIELD1       180A    10 20BLKFOLD
     A
```

Figure 4-9. Specifying the BLKFOLD Keyword

BOX (Box)

Use this record-level keyword to print a rectangle.

The format of the keyword is:

```
BOX(first-corner-down first-corner-across diagonal-corner-down
diagonal-corner-across line-width)
```

The first-corner-down, first-corner-across, diagonal-corner-down, and diagonal-corner-across parameters define the diagonal corners of the box. All are required parameters.

The first-corner-down parameter defines the vertical starting point of the BOX relative to the margins specified on the FRONTMGN or BACKMGN parameter of the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The first-corner-across parameter defines the horizontal starting point of the BOX relative to the margins specified on the FRONTMGN or BACKMGN parameter of the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The diagonal-corner-down parameter defines the vertical end point of the BOX relative to the margins specified on the FRONTMGN or BACKMGN parameter of the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The diagonal-corner-across parameter defines the horizontal end point of the BOX relative to the margins specified on the FRONTMGN or BACKMGN parameter of the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The line-width parameter is required and defines the width of the lines. Valid values are 0.001 to 57.790 cm (0.001 to 22.750 in.). The following special values can also be specified:

Value	Line Width
*NARROW	12/1440 in. (0.008 in., 0.022 cm)
*MEDIUM	24/1440 in. (0.017 in., 0.042 cm)
*WIDE	36/1440 in. (0.025 in., 0.064 cm)

Notes:

1. The UOM parameter on the CRTPRTF command determines the units of measure for the first-corner-down, first-corner-across, diagonal-corner-down, diagonal-corner-across, and line-width parameter values. If the value specified for a parameter is outside the valid range, it is flagged when the spooled file is created.
2. Depending on printer hardware, lines smaller than approximately 0.004 in. (0.010 cm) might not print because of printer resolution. No message is issued when this occurs.

The line width is drawn on the inside of the box.

When the BOX keyword is specified on a record format, all fields within the record format must be positioned using the POSITION keyword. See "POSITION (Position)" on page 4-74 for more information.

An error message is issued if a constant field is specified in a record format where the BOX keyword is also specified.

An error message is issued at application run time if the box extends beyond the page boundaries.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when BOX is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

You can specify this keyword multiple times on a record.

You cannot specify BOX with the SPACEA, SPACEB, SKIPA, or SKIPB keywords.

Option indicators are valid for this keyword.

Figure 4-10 shows how to specify the BOX keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R BOX1                      BOX(1.2 0.5 5.1 6.3 0.2)
     A*
     A          R BOX2                      BOX(2 5 5.0 3.33 *WIDE)
     A                                      BOX(0.5 0.1 2.1 2.0 0.09)
     A*
     A          R BOX3
     A  01                                  BOX(0 0 8.5 11.0 0.5)
     A*
     A
     A
```

Figure 4-10. Specifying the BOX Keyword

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the parameter values.

BOX1 prints a box with one corner located 1.2 units down and 0.5 units across from the location specified on the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The diagonal corner of this box is located 5.1 units down and 6.3 units across from the margins specified on the CRTPRTF command. The edges of the box are 0.2 units wide.

BOX2 prints two boxes. The first box starts 2 units down and 5 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The diagonal corner of this box is located 5.0 units down and 3.33 units across from the margins specified on the CRTPRTF command. The edges of the box are determined by the special value *WIDE.

The second box starts 0.5 units down and 0.1 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The diagonal corner of this box is located 2.1 units down and 2.0 units across from the margins specified on the CRTPRTF command. The edges of the box are 0.09 units wide.

BOX3 prints only if indicator 01 is on.

CDEFNT (Coded Font Name)

Use this field- or record-level keyword to specify the coded font for printing a named or constant field or fields within a record.

The format of the keyword is:

```
CDEFNT([library-name/]coded-font-name)
```

The coded-font-name parameter is required and can be up to 8 characters in length.

Use the optional library-name parameter to further qualify the coded font name. If you do not specify a library name, *LIBL is used to search for the coded font name at print time. If *LIBL is used, the system-supplied font libraries are added to the library list when searching for the requested font. Using the library-name parameter allows the coded font name to be located more rapidly. However, the library list is still used to locate the character set and code page defined by the coded font name.

Note: If an application uses private resources (for example, fonts, page segments, overlays, or GDF files not distributed with the system), be aware of the following. When referencing these resources, if you specify *LIBL or you do not specify a library name, the resources must be available through the library list used by the application creating the spooled file.

The coded font value is validated at print time. An error message is issued if it is not valid or when the resource cannot be located.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when CDEFNT is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

CDEFNT cannot be specified at the same level as the FONT or FNTCHRSET keywords.

Option indicators are valid for this keyword.

Figure 4-11 shows how to specify the CDEFNT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1
     A            FLD1           8A    10 13CDEFNT(QFNTCPL/X0BRTR)
     A*
     A
```

Figure 4-11. Specifying the CDEFNT Keyword

FLD1 specifies coded font X0BRTR found in library QFNTCPL.

CHRID (Character Identifier)

Use this field-level keyword to specify that a graphic character set and code page other than the device default can be used for this field. This can be important when extended alphabetics (characters such as u with an umlaut or c with a cedilla) are to be printed.

This keyword has no parameters.

If you do not specify CHRID for a field, data printed in that field is printed in the character set of the printer device. How the data is printed depends on how code points used in the original code page correspond to the code page used on the device. For example, a slash (/) can be printed as a blank, because hex 51 is an empty code point in the basic U.S. character set.

The CHRID keyword is not valid on constant fields or numeric fields (fields with decimal positions specified in positions 36 through 37).

The CHRID keyword is ignored for fields in printer files which specify *JOBCCSID for the CHRID parameter on the Create Printer File (CRTPRTF), Change Printer File (CHGPRTF), or Override Printer File (OVRPRTF) commands. All printer data is assumed to be in the CCSID of the current job for these printer files. For more information about CCSID support, see *National Language Support Planning Guide*.

In SCS printer files (DEVTYPE(*SCS) on the CRTPRTF command), you cannot specify CHRID on the same field as the TRNSPY keyword. In IPDS printer files (DEVTYPE(*IPDS) on the CRTPRTF command), you can specify CHRID on the same field as the TRNSPY keyword. However, a warning message results stating that the DEVTYPE should not be changed to *SCS.

For printer files created with DEVTYPE(*AFPDS), CHRID applies only to files that use registered font IDs. If the file uses downloaded coded fonts or character set/code pages, the keyword is ignored and a message is issued.

If the DFNCHR keyword is specified for a record format, the CHRID keyword cannot be specified in that record format. If the DFNCHR keyword is specified at the file level, CHRID cannot be specified in that file.

Option indicators are not valid for this keyword. However, option indicators can be used to condition the field associated with this keyword.

Figure 4-12 shows how to specify the CHRID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A            TITLE         40      1 20CHRID
     A
```

Figure 4-12. Specifying the CHRID Keyword

The field TITLE is a named field. With CHRID specified, the printer attempts to print the appropriate characters. See the *Guide to Programming for Printing* for more information on the printing conditions with the CHRID keyword.

CHRSIZ (Character Size)

Use this record- or field-level keyword to expand the width and height of a record or field. CHRSIZ is valid only for IPDS printers and only for printer files with device type *IPDS or *AFPDS specified.

The format of the keyword is:

```
CHRSIZ(width height)
```

The valid values for the width and height parameters are 1.0 through 20.0.

Any formatting you choose, such as using a specific font or editing via the EDTCDE and EDTWRD keywords, is done prior to the expansion.

If you specify CHRSIZ on a record, it applies to all fields in that record for which you do not specify CHRSIZ at the field level. When you specify a numeric font (for example: 011–Courier 10 pitch) with CHRSIZ, the printer scales the hardware fonts (scaling with integer values only).

Graphics fonts may also be specified with CHRSIZ. When you specify a GDDM graphic font (for example: ADMMVSS) with CHRSIZ, the system scales the graphic font. You can use decimal values to scale graphic fonts.

It is recommended that you do not use FONT (*DEVD) (on the CRTPRTF, CHGPRTF, or OVRPRTF command) when using CHRSIZ. If you do use FONT (*DEVD), then fields specified with CHRSIZ are positioned on the page assuming the font in the device description is a 10-pitch font.

In expanding a field, CHRSIZ uses the current font and lines-per-inch value. For example, if you specify FONT(011), a 10-pitch font, and lpi(6) for the printer file, specifying CHRSIZ (3 3) for a 10-character field expands the field to 3 inches wide (30 characters/10 characters per inch) and 1/2 inch high (3 lines/6 lines per inch).

Note: If the current font is a coded font or font character set/code page, 10-pitch is assumed.

If, however, you specified FONT(222), a 15-pitch font, and lpi(4) on a record format, the 10-character field mentioned above expands to 2 inches wide (30 characters/15 characters per inch) and 3/4 inch high (3 lines/4 lines per inch).

You cannot specify a hardware font on the FONT keyword when a decimal value is specified on the CHRSIZ keyword. If both of these keywords apply to the same field (specified either at the record or field level), the file is not created.

Note: When you specify FONT(*VECTOR) with the CHRSIZ keyword, the 4234 printer uses a default code page.

The CHRSIZ keyword does not work if the specified font is a typographic font and the printer is either a 3812 or 3816 printer. Typographic fonts are not scalable on these printers. This is a limitation of the printer. The typographic fonts are the following:

751	1351
1051	1653
1053	2103
1056	

When you create a file, exceeding or overlapping the page length is not diagnosed for expanded height. It is diagnosed, however, for expanded width. The field length used is the DDS field length multiplied by the expansion width you specify on CHRSIZ and rounded up to the integer value.

Valid data types for this keyword are A, F, and S.

Option indicators are not valid for this keyword.

Figure 4-13 shows how to specify the CHRSIZ keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                    CHRSIZ(3  3)
     A  02 03                                FONT(222)
     A            FIELD1          3A     6 01
     A            FIELD2          6A    16 01CHRSIZ(2.5 2)
     A                                        FONT(ADMMVSS)
     A            FIELD3          6S 0  20 01CHRISIZ(1 1)
     A
```

Figure 4-13. Specifying the CHRSIZ Keyword

In Figure 4-13, FIELD1 is printed using CHRSIZ(3 3). FIELD2 is printed using CHRSIZ(2.5 2). Note that the decimal CHRSIZ is valid because the FONT specified for the field is not numeric. FIELD3 prints using CHRSIZ(1 1).

COLOR (Color)

Use this field-level keyword to specify the color for a field, if it is supported by the printer device. The COLOR keyword is used only by the 4224 printer. If you do not specify COLOR, or if the keyword is not valid for a printer device, black (the default value) is used.

The format of the keyword is:

```
COLOR(BLK | BLU |BRN | GRN | PNK | RED | TRQ | YLW)
```

You can specify one, and only one, parameter value for COLOR. The valid parameter values are:

Parameter	Meaning
BLK	Black
BLU	Blue
BRN	Brown
GRN	Green
PNK	Pink
RED	Red
TRQ	Turquoise
YLW	Yellow

If you use COLOR on the same record format with the BLKFOLD, CPI, or DFNCHR keyword, the file is not created.

COLOR is valid on IPDS and IPDS AFP(*YES) printers. If you specify DEVTYPE(*SCS) on the CRTPRTF command, a warning message results but the file is created successfully.

Valid data types for this keyword are A, S, and F.

When you specify COLOR more than once for a field, you must specify option indicators each time you specify COLOR. If more than one COLOR is in effect when printing the field, the first one in effect is used. You cannot specify the same color more than once on the same field. Figure 4-14 shows the effects of specifying COLOR for a field.

Option indicators are valid for this keyword.

Figure 4-14 shows how to specify the COLOR keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A  99                              1  3'PRINT RED TEXT'
     A                                     COLOR(RED)
     A            FIELD1          3A    12 01TEXT('PINK IF 02, +
     A                                     YELLOW IF 07, +
     A                                     BLACK IF NEITHER')
     A  02                                 COLOR(PNK)
     A  07                                 COLOR(YLW)
     A
```

Figure 4-14. Specifying the COLOR Keyword

In Figure 4-14, if indicator 99 is ON, the constant field 'PRINT RED TEXT' is printed in red. FIELD1 is printed in pink, yellow, or black, depending on the indicators 02 and 07.

CPI (Characters Per Inch)

This record- or field-level keyword specifies the horizontal printing density for the record format or field you are defining. Use CPI to:

- Darken logos and other printed graphics that you create using the DFNCHR keyword
- Place more data in less space on printed forms
- Fit the appearance of a form to your needs

The format of the keyword is:

CPI (10 | 15)

10 or 15 specifies the number of characters per inch.

This keyword is valid only for the 5224 and 5225 SCS printers. If you do not specify CPI, the density is set by the CPI parameter on the Create Printer File (CRTPRTF), Change Printer File (CHGPRTF), or Override with Printer File (OVRPRTF) command.

If you specify CPI at the record level, all fields in the record format are at the same density except those for which you specify CPI at the field level.

If you specify CPI at the field level, you can specify different densities for fields printed on the same line. The position you specify for each field (in positions 42 through 44) is based on the value of the CPI parameter on the CRTPRTF, CHGPRTF, or OVRPRTF command (see Figure 4-15 and Figure 4-16 on page 4-24).

When you specify CPI at the field level, overlapping fields are not diagnosed.

A warning message results at creation time if you specify CPI in a file created with DEVTYPE(*IPDS) or DEVTYPE(*AFPDS). To change the CPI, you must specify the FONT keyword (see "FONT (Font)" on page 4-56).

You cannot specify CPI on the same record format as the DRAWER keyword.

Option indicators are valid for this keyword.

Figure 4-15 shows how to specify the CPI keyword for a record format.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A  02                                      CPI(15)
00030A            FLD1          20       3  1
00040A            FLD2           5  0      +2
00050A          R RECORD2                      SPACEB(1)
00060A            FLD3                       1
     A
```

Figure 4-15. Specifying the CPI Keyword (Example 1)

In Figure 4-15, if option indicator 02 is set to on, both FLD1 and FLD2 in RECORD1 are printed at 15 characters per inch. If option indicator 02 is set to off, FLD1 and FLD2 are printed at the density specified for the CPI parameter on the CRTPRTF, CHGPRTF, or OVRPRTF command.

The printer spaces one line before printing RECORD2. FLD3 in RECORD2 is printed at the density specified for the CPI parameter on the CRTPRTF, CHGPRTF, or OVRPRTF command.

Figure 4-16 shows what happens when a field at 15 CPI is printed between fields printed at 10 CPI.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCDA                      SPACEA(1)
00020A            FLD1          10          1
00030A            FLD2          10         11CPI(15) 1
00040A            FLD3          10         21
     A
```

Figure 4-16. Specifying the CPI Keyword (Example 2)

In Figure 4-16, all positions entries 1 refer to columns measured at 10 CPI (as specified on the CRTPRTF, OVRPRTF, or CHGPRTF command). Therefore, RCDA is printed as follows:

```
1111111112222222222      3333333333
```

FLD2, being compressed at 15 CPI, uses less room than FLD1 or FLD3. To avoid the gap, specify FLD3 more to the left. To calculate the position of FLD3, add the length of FLD2 to the specified position of FLD2. To calculate the length of FLD2, use the following formula:

$$\frac{\text{length specified X file density}}{\text{density for the field}} = \text{printed length}$$

or, for FLD2:

$$\frac{10 \times 10}{15} = \frac{10 \times 2}{3} = 6.67 \text{ (rounded up to 7)}$$

Add 7 to 11, the specified position of FLD2, as follows:

```
7 + 11 = 18
```

The resulting corrected DDS for Example 2 becomes:

```
R RCDA                      SPACEA(1)
  FLD1           10         1
  FLD2           10        11CPI(15)
  FLD3           10        18
```

The record format then prints as follows:

```
1111111112222222222 3333333333
```

Figure 4-17 shows what happens when a field at 10 CPI is printed between fields printed at 15 CPI.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCDB                      SPACEA(1)
00020A            FLD4          10          1
00030A            FLD5          10         11CPI(10)  1
00040A            FLD6          10         21
     A
```

Figure 4-17. Specifying the CPI Keyword (Example 3)

In Figure 4-17, the positions entries 1 refer to positions measured at 15 CPI (as specified on the CRTPRTF, OVRPRTF, or CHGPRTF command). The system uses the following formula to calculate the beginning position of fields printed at 10 CPI within files printed at 15 CPI:

$$\frac{2(\text{specified position} - 1)}{3} + 1 = \text{printed position (truncated if fractional)}$$

or, for FLD5:

```
2(11-1)  + 1 = 7.67 (truncated to 7)
  3
```

The truncation can cause overprinting of FLD4 by FLD5, as shown by the following:

```
444444444455555566666666666666
```

To avoid the overprinting, specify FLD5 one more position to the right (position 12).

To calculate the position of FLD6, add the length of FLD5 to the position of FLD5. To calculate the length of FLD5, use the following formula:

```
length specified X density for the file = printed length
        density for the field
```

or, for FLD5:

```
10 X 15 = 15 (rounded up if necessary)
  10
```

Add 15 to the (adjusted) position of FLD5:

```
15 + 12 = 27
```

The resulting corrected DDS for Example 3 becomes:

```
R RCDB                SPACEA(1)
  FLD4        10       1
  FLD5        10      12CPI(10)
  FLD6        10      27
```

The record format then prints as follows:

```
4444444444 5555555555 6666666666
```

Figure 4-18 shows the effect of the CPI keyword on how the system truncates or folds fields at the right side of the printer form. This depends on the values of the FOLD and PAGESIZE parameters on the CRTPRTF, CHGPRTF, or OVRPRTF commands.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCDC                    SPACEA(1)
00020A            FLD7          10       140
00030A            FLD8          10       150CPI(10)
     A
```

Figure 4-18. Specifying the CPI Keyword (Example 4)

In Figure 4-18, if the file is being printed at 15 CPI with a forms width of 160, FLD7 and FLD8 are printed as follows:

- FLD7 starts at position 140 for a print length of 10 at 15 CPI (16.9 mm or 0.667 inch).
- FLD8 starts at position 150 for a print length of 10 at 10 CPI (25.4 mm or one inch).

Printing FLD8 at position 150 calculated at 15 CPI causes FLD8 to extend beyond the right margin. Therefore, FLD8 is either truncated or folded onto the next line (depending on the FOLD parameter on the CRTPRTF, CHGPRTF, or OVRPRTF command). To calculate the length of FLD8, use the following formula:

$$\frac{\text{length specified X density for the file}}{\text{density for the field}} = \text{printed length}$$

or, for FLD8:

$$\frac{10 \text{ X } 15}{10} = 15 \text{ (truncated to next lower integer if necessary)}$$

Note: When a file printed at 15 CPI contains fields printed at 10 CPI, the right margin of the form is adjusted for all fields according to the following formula:

$$\frac{2(\text{specified length of the field} - 1)}{3} + 1 = \text{adjustment (truncated if fractional)}$$

CVTDTA (Convert Data)

This field-level keyword converts character data to hexadecimal data when the field is passed to the printer. You can use the CVTDTA keyword to define:

- Logos or emblems for a letterhead on your forms
- Alternative character sets or symbols (such as a copyright symbol)
- The appearance of a physical form (by adding vertical and horizontal lines that act as boundaries on the form or between positions on an invoice)
- IPDS bar code commands

This keyword has no parameters.

In an SCS printer file (DEVTYPE(*SCS) on the CRTPRTF command), specify CVTDTA only when you use the DFNCHR keyword. Furthermore, use CVTDTA when you define characters for unassigned code points. A **code point** is one of the 256 values that you can assign a character in a character set. An **unassigned code point** is a code point to which no character is assigned. On the AS/400 system, a code point is identified by a 2-digit hexadecimal number. For example, in the EBCDIC character set, code point hex C1 is assigned the character A; hex 51 is an unassigned code point.

CVTDTA is valid for the 5224, 5225, and IPDS printers. For IPDS printers, CVTDTA allows you to specify code points to be included in the data stream. These code points print as preassigned on the printer. Do not use the CVTDTA keyword with the TRNSPY and DFNCHR keywords for IPDS printers.

If you define characters for unassigned code points, do one of the following:

- Specify CVTDTA
- Work with hexadecimal data in your program

Specify CVTDTA only for named fields. For user-defined characters in constant fields, use the DFT and DFNCHR keywords.

In an SCS printer file (DEVTYPE(*SCS) on the CRTPRTF command), if you specify CVTDTA, you must also specify the TRNSPY keyword. In printer files created with DEVTYPE(*IPDS) or DEVTYPE(*AFPDS) on the CRTPRTF command, if you specify CVTDTA, you do not need to specify the TRNSPY keyword. However, a warning message appears stating that the DEVTYPE should not be changed to *SCS.

If you specify CVTDTA on a field, the length of the field must be an even number. The printed length of the field is the length you specify, divided by two.

If you specify CVTDTA for a field, the character data your program passes in the field must contain only valid hexadecimal characters (0 through 9 and A through F). Blanks, whether embedded or trailing, are not valid hexadecimal characters. If characters that are not valid are specified in the field at program run time, the OS/400 program sends escape message CPF5234 to your program.

Option indicators are not valid for this keyword.

For an example of how to use the CVTDTA keyword, see "TRNSPY (Transparency)" on page 4-83.

The following rules apply to using DDS CVTDTA for bar code commands:

- The support is only for printers with device type *IPDS.
- The support allows the following commands:
  - WBCC (Write Bar Code Control)
  - WBC (Write Bar Code)
  - END

  All three commands must be in the same field. No other commands can be in that field.
- The length of the field must be exact.
- The length within each command must be exact.
- The file should contain a DDS BARCODE keyword on another record in the file. This record does not have to be used. It indicates to the OS/400 program that bar codes should be expected when the file is used.
- Correlation IDs are not required on the IPDS commands.
- No validity checking is done on the user's bar code data. Data that is not valid will cause the printer to report that the command is not valid.
- Examples of the commands are shown in Figure 4-19 on page 4-28. Adding the lengths of these commands in the example totals 69 (45 + 17 + 7 = 69). This will be multiplied by two to indicate the number of characters included in the CVTDTA field. This means the field with CVTDTA for this example would require a length of 138 (69 x 2 = 138).
- See the *Intelligent Printer Data Stream Reference* manual for more information on bar code commands.

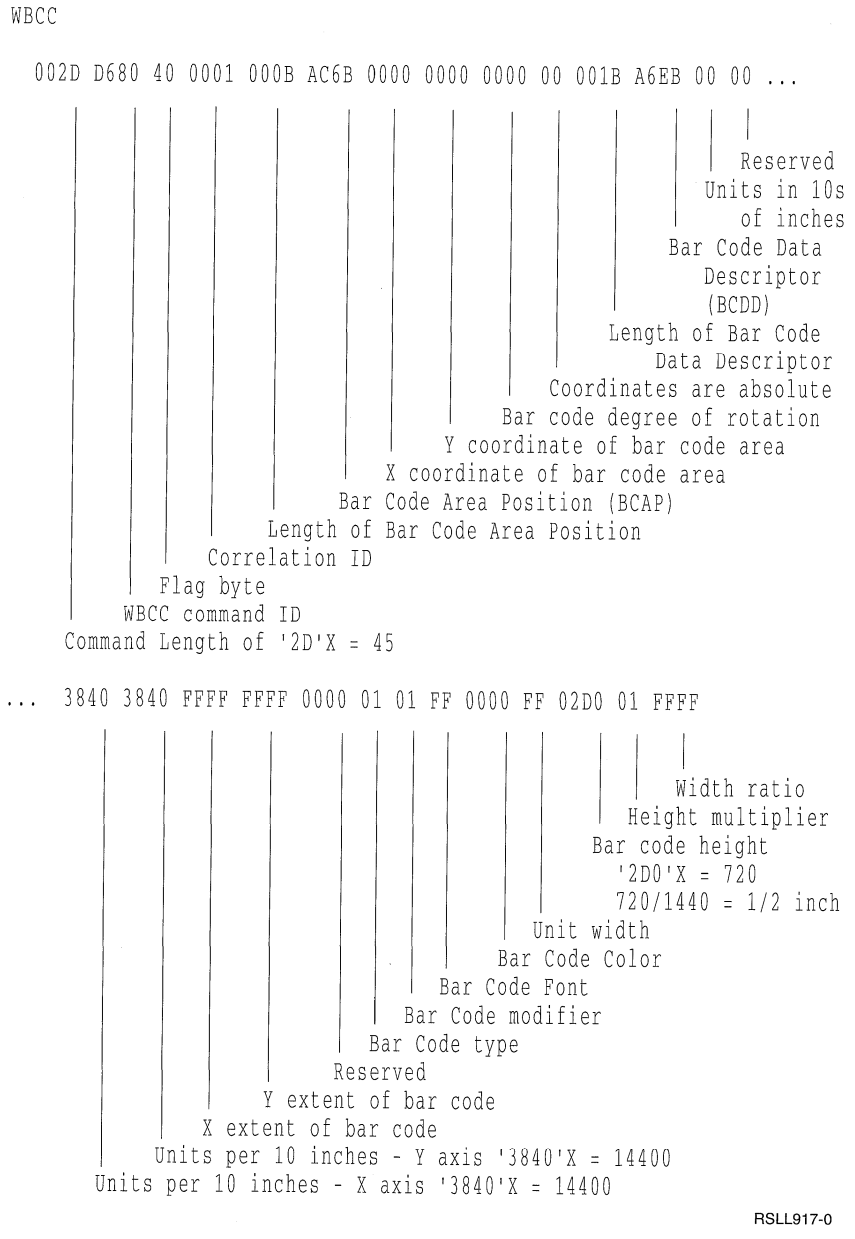


Figure 4-19 (Part 1 of 2). Command Format for Bar Code Commands Using CVTDTA

```
WBC

  0011 D681 40 0002 00 10E0 0B40 F1F2F3F4F5

      |    |  |    |  |    |    |      |
      |    |  |    |  |    |    |      Data to be encripted ('12345')
      |    |  |    |  |    |    Y coordinate: '0B40'X = 2880
      |    |  |    |  |    |       2880/1440 = 2 inches
      |    |  |    |  |    X coordinate: '10E0'X = 4320
      |    |  |    |  |       4320/1440 = 3 inches
      |    |  |    |  Option byte
      |    |  |    Correlation ID
      |    |  Flag byte
      |    WBC command ID
      Command Length of '11'X = 17

END

  0007 D65D 40 0003

      |    |  |  |
      |    |  |  Correlation ID
      |    |  Flag byte
      |    END command ID
      Command Length of '7'X = 7
```

RSLL918-0

Figure 4-19 (Part 2 of 2). Command Format for Bar Code Commands Using CVTDTA

DATE (Date)

Use this field-level keyword to display the current job date or the current system date as a constant field 6 bytes long. You can specify the location of the field, the DATE keyword, and optionally, the CDEFNT, CHRSIZ, COLOR, EDTCDE, EDTWRD, FNTCHRSET, FONT, HIGHLIGHT, UNDERLINE, or TEXT keyword. Positions 17 through 38 must be blank.

The format of the keyword is:

```
DATE{(*JOB | *SYS)}
```

The *JOB value specifies that the current job date is printed. If no parameter is specified, *JOB is used. The *SYS parameter specifies that the current system date is printed.

If you specify EDTCDE(Y) for a DATE field, the print length is 8 positions. Specifying EDTCDE(Y) changes the date from

```
mmddyy
```

to

```
mm/dd/yy
```

The slashes (/) represent the job attribute DATSEP at run time and the order of month, day, and year is determined by the job attribute DATFMT. (DATFMT can be *SYSVAL, indicating that your program is to retrieve the date from the system value QDATFMT, or MDY, DMY, YMD, or JUL, where M=month, D=day, Y=year, and JUL=Julian.)

If the DATFMT specified for the job is *JUL (Julian), you cannot use the EDTWRD keyword to edit the result.

Option indicators are not valid for this keyword. However, option indicators can be used to condition the field associated with this keyword.

Figure 4-20 shows how to specify the DATE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R REC01
     A                                     1 56
     A                                        DATE
     A  21                                 2 56
     A                                        DATE(*JOB)
     A  22                                 2 56
     A                                        DATE EDTCDE(Y)
     A  23                                 2 56
     A                                        DATE(*JOB) EDTCDE(Y)
     A  24                                 2 56
     A                                        DATE(*SYS)
     A  25                                 2 56
     A                                        DATE(SYS) EDTCDE(Y)
     A
```

Figure 4-20. Specifying the DATE Keyword

The job date is printed without editing on line position 56.

The job date is also printed without editing if option indicator 21 is on. The job date is printed with editing if either option indicator 22 or 23 is on. The system date is printed without editing if option indicator 24 is on. The system date is printed with editing if option indicator 25 is on.

DFNCHR (Define Character)

The DFNCHR keyword allows you to define characters of your own design at the file or record level for the 5224 Printer and 5225 Printer. With this keyword you can specify DFNCHR more than once at the file or record level, or as many as 50 characters each time you specify DFNCHR.

The format of the keyword is:

```
DFNCHR(X'code-point-1' X'dot-matrix-pattern-1'
[X'code-point-2' X'dot-matrix-pattern-2'...
[X'code-point-50' X'dot-matrix-pattern-50']])
```

Note: You cannot specify more than 5000 characters in a single DDS statement. If in specifying DFNCHR several times together, you need to specify more than 5000 characters, start a new DDS statement by specifying an option indicator for the new DFNCHR keywords. To avoid having to set the indicators on, specify an N (for example, N50). This causes the conditioning to be on for the keyword with no program action.

User-defined characters can take up one print position (as in Figure 4-21 on page 4-31) or more than one print position (as in Figure 4-23 on page 4-34 and Figure 4-27 on page 4-36). For each print position, specify a code point and a dot matrix pattern. In the EBCDIC character set, hex C1 is assigned the character A; hex 51 is an unassigned code point (see "Selecting Which Code Points to Redefine" on page 4-32).

You define a dot matrix pattern in DDS by specifying nine 2-digit pairs of hex digits. You can specify only the characters 0 through 9 and A through F (see "Specifying Dots to be Printed in the Dot Matrix" on page 4-33).

When your program sends an output operation to a record format for which DFNCHR defines code points different from those defined for the previous output operation, the OS/400 program loads the new definitions, thereby changing the defined characters. This process can slow printing.

If, however, the same DFNCHR keywords are in effect for two output operations in a row, the OS/400 program does not reload code points for the second output operation.

You can use DFNCHR only with SCS printers. It cannot be specified on the same record format with IPDS printer keywords such as COLOR, LPI, and BARCODE. If any format in the file contains a combination of SCS and IPDS printer keywords, the file is not created.

If you specify DFNCHR in a file created with DEVTYPE(*IPDS) or DEVTYPE(*AFPDS), a warning message appears at create time.

You cannot specify DFNCHR on the same record format as the DRAWER keyword. If any format in the file contains DFNCHR at the record-level and a DRAWER keyword, the file is not created.

Option indicators are valid for this keyword.

Figure 4-21, Figure 4-27 on page 4-36, and Figure 4-32 on page 4-39 show how to specify DFNCHR.

Example 1:

Figure 4-22 on page 4-32 uses a single dot matrix to show how to specify DFNCHR at the record level so that hex 7C prints © instead of @.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
                                                  1     2
00010A          R RECORD                    DFNCHR(X'7C' X'007E813CC324817E00')
00020A                                  58  4DFT(X'7C')   TRNSPY
00030A                                     +2DFT('1982')
     A
```

Figure 4-21. Specifying the DFNCHR Keyword (Example 1)

This example redefines code point hex 7C **1**, normally @ in the EBCDIC character set, as a copyright mark. The copyright mark is printed as follows (on line 58 of a printer form):

© 1982

The hex digits **2** define the following dot matrix pattern:

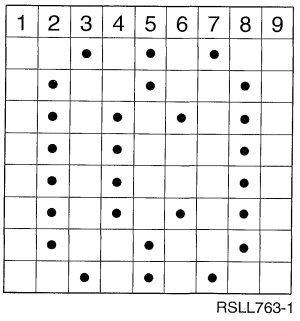


Figure 4-22. Dot Matrix Pattern for Example 1

See "Specifying Dots to be Printed in the Dot Matrix" on page 4-33 for more information.

Selecting Which Code Points to Redefine

You can define any code point except hex 00. For a list of commonly used character sets, see the section on character codes in the *Programming Reference Summary* manual. When you define a code point, you can do one of the following:

- Redefine an existing character. If you then attempt to print that character, the alternate character is printed. Example 1 (Figure 4-21 on page 4-31) uses such a code point.
- Define a character for an unassigned code point (for which there is no existing character in your system character set). Example 2 (Figure 4-23 on page 4-34) uses such code points.

If you redefine an existing character, select which character to print (the existing character or the redefined character) by specifying option indicators for the DFNCHR keyword. If you select DFNCHR, the user-defined character is printed. If you do not select DFNCHR, the existing character is printed.

Note: Output operations run faster when you define an unassigned code point. An unassigned code point avoids the reloading of code points when your program selects different DFNCHR keywords.

The Dot Matrix

The dot matrix for the 5224 Printer and 5225 Printer is an 8-row-by-9-column matrix. All 8 rows and 9 columns are printed regardless of CPI or LPI settings.

The vertical distance between dots is always 0.352 mm (0.014 inch) regardless of the LPI setting (LPI parameter on the CRTPRTF command). The LPI setting determines the space between lines, not the height of characters. At a setting of 9 lpi (2.82 mm or 0.111 inch for each line), there is no vertical space between lines if all rows of dots are used. However, the normal character set does not use the bottom row of dots (line 8) in the matrix, so that even at 9 lpi there is some space between lines. If you choose, use row 8 to define your own characters.

The horizontal distance between dots depends on the CPI setting. At 10 CPI, each column is spaced 0.262 mm (0.0111 inches) apart, giving each character 2.54 mm (0.1 inches). At 15 CPI, each column is spaced 0.188 mm (0.0074 inches) apart, giving each character 1.69 mm (0.0667 inches). The standard character set does not use columns 1 and 9 (to allow spacing between characters).

You can use columns 1 and 9 to define your own characters with one restriction: the 5224 Printer and 5225 Printer cannot print two adjacent horizontal dots. To print two adjacent horizontal dots (such as in a solid underline), the line must be printed twice. This can be done using a different set of code points on each pass, one to define the odd dots, and the other to define the even dots. Both passes occur during one output operation. If your program attempts to print two adjacent horizontal dots, no error message appears, but one of the dots is not printed. (The last position of dots in one character and the first position in the character to its right are considered adjacent dots.) There is no restriction on adjacent vertical dots.

On any one output operation, each code point represents a single eight-by-nine matrix. To print characters larger than this requires more than one eight-by-nine matrix, each one normally defined by a different code point. Overprinting may also be required.

For example, to print a double-wide character, specify a code point for the left half of the character and another code point for the right half. Double-high characters require a code point for the top half of the character and another for the bottom half. On the first line, the top half of all characters are printed, and on the next line, the bottom half of all characters. You must specify lpi(9) on the CRTPRTF, CHGPRTF, or OVRPRTF command to avoid a space between the top and bottom halves. Using DDS, you can define two fields in one record format, one for the upper half and one for the lower half. Example 2 (Figure 4-23 on page 4-34) shows a character two wide by two high.

Specifying Dots to be Printed in the Dot Matrix

When you define a dot matrix pattern for a user-defined character, specify nine 2-digit pairs of hexadecimal digits. Each 2-digit pair corresponds with a column in the matrix, the first pair with the first column, the second pair with the second column, and so forth. Specify the left character of each pair to control which dots are printed in the upper half of the column. Specify the right character to control the lower half.

Use the approach shown in Figure 4-23 on page 4-34 to specify the dot matrix pattern for a copyright mark, which prints as ©.

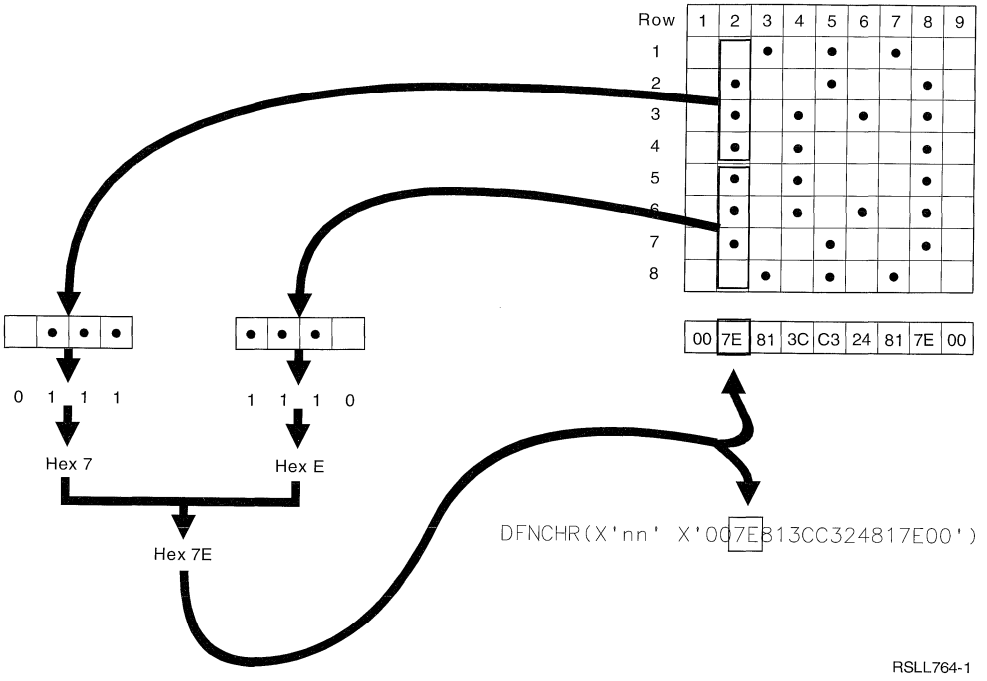


Figure 4-23. Specifying the Dot Matrix for a Copyright Mark

Use the form in Figure 4-24 on page 4-35 to plan your dot patterns and specify the required hex digits for characters as large as three columns wide by three lines high. In this grid pattern, mark the dot patterns for as many as nine print positions (three across and three down).

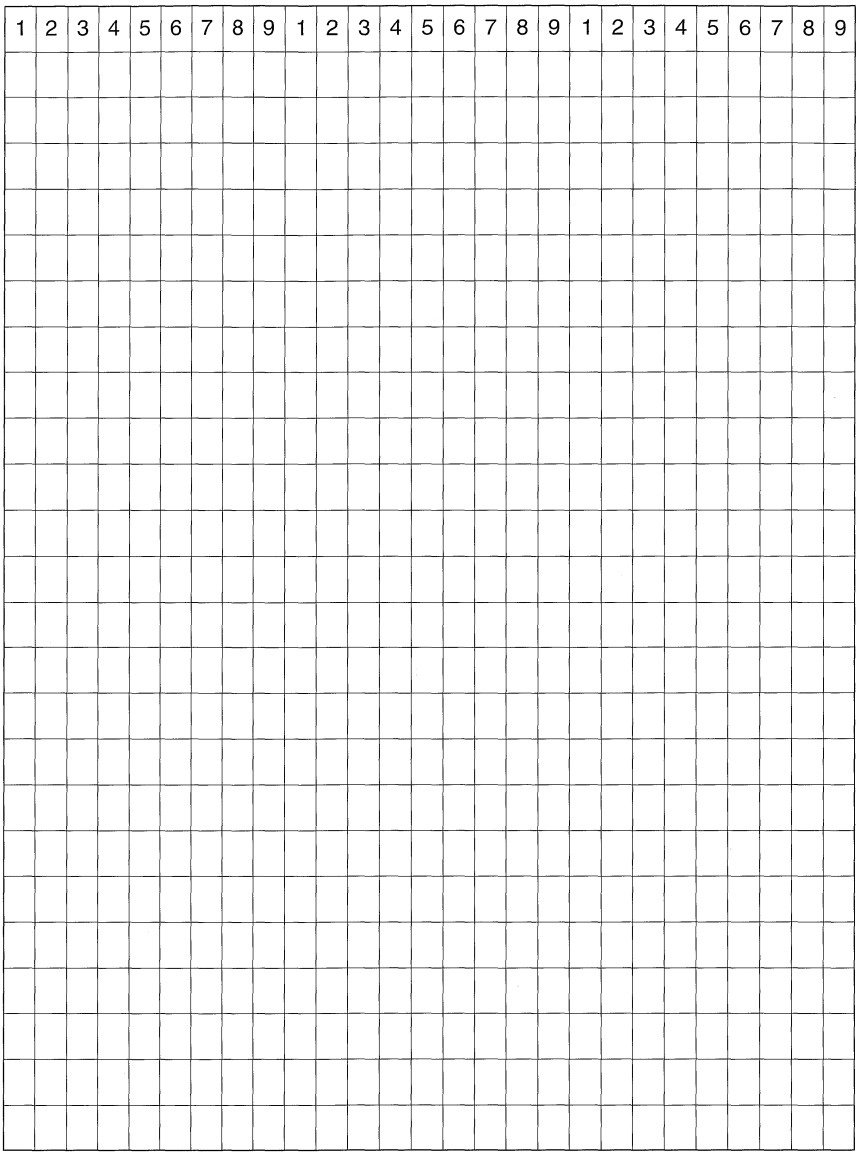


Figure 4-24. Grid Pattern for Specifying Dot Matrix Characters

Use Figure 4-25 to determine which hex digit to specify for each half-column in the grid pattern.

Figure 4-25. Hex Digits for the Bit Patterns

Bit Patterns	Hex Digits	Bit Patterns	Hex Digits
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

For each print position, complete one row of the grid pattern, shown in Figure 4-26. There should be one pair of hex digits to each box.

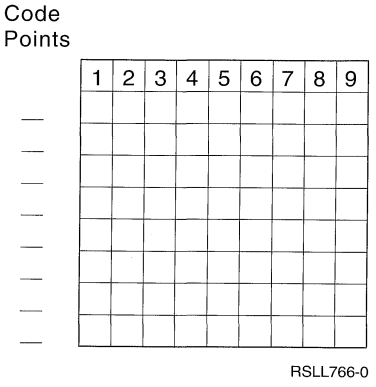


Figure 4-26. Completing the Grid Pattern

Example 2:

Figure 4-27 uses a dot matrix for a large character. This example shows how to specify DFNCHR at the file level for a character two positions wide by two lines high.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
                                                    1     2
00010A                                      DFNCHR(X'51' X'000000FF00FF00E700' +
00020A                                             X'52' X'E700E700E700E10000' +
00030A                                             X'53' X'0000009C001F000700' +
00040A                                             X'54' X'07000700FF00FC0000' +
00050A                                             X'55' X'00000000FF00E700E7' +
00060A                                             X'56' X'00E700E700E3000000' +
00070A                                             X'57' X'000000001E00070007' +
00080A                                             X'58' X'0007000700FE000000')
00090A          R RECORD1
00100A                        3        58  4DFT(X'5152')   TRNSPY
00110A                                 58  4DFT(X'5556')   TRNSPY
00120A                                 59  4DFT(X'5354')   TRNSPY
00130A                                 59  4DFT(X'5758')   TRNSPY
     A
```

Figure 4-27. Specifying the DFNCHR Keyword (Example 2)

Figure 4-27 redefines eight code points (hex 51 through hex 58) **1**. Each position of the two-by-two character is printed twice so that adjacent horizontal dots can print. The hex codes **2** define the dot matrix pattern.

The information marked **3** shows how the large character 5 looks when printed (using four print positions, two on line 58 and two on line 59 of a printer form):

5

Note: The file should be at 9 lpi to avoid a horizontal gap in the large character (LPI parameter on the CRTPRTF, CHGPRTF, or OVRPRTF command).

In the grid pattern, mark the dot patterns for as many as nine print columns (three across and three down), as shown in Figure 4-28 on page 4-37.

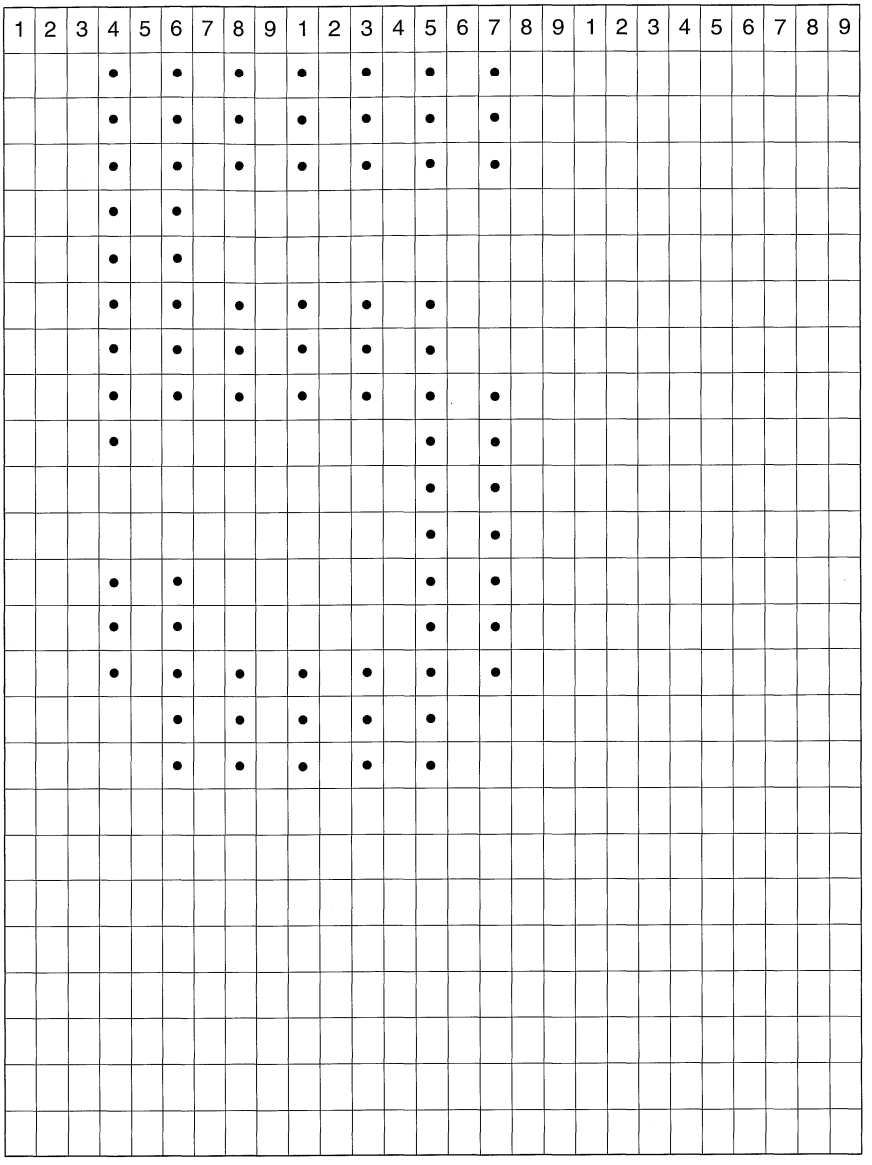


Figure 4-28. Specifying the Grid Pattern for Example 2 (Points 51 through 54)

Use Figure 4-25 on page 4-35 to determine which hex digit to specify for each half-column in the grid pattern.

For each print position, complete one row of the grid pattern (one pair of hex digits to each box) as shown in Figure 4-29 on page 4-38 for code points 51 through 54.

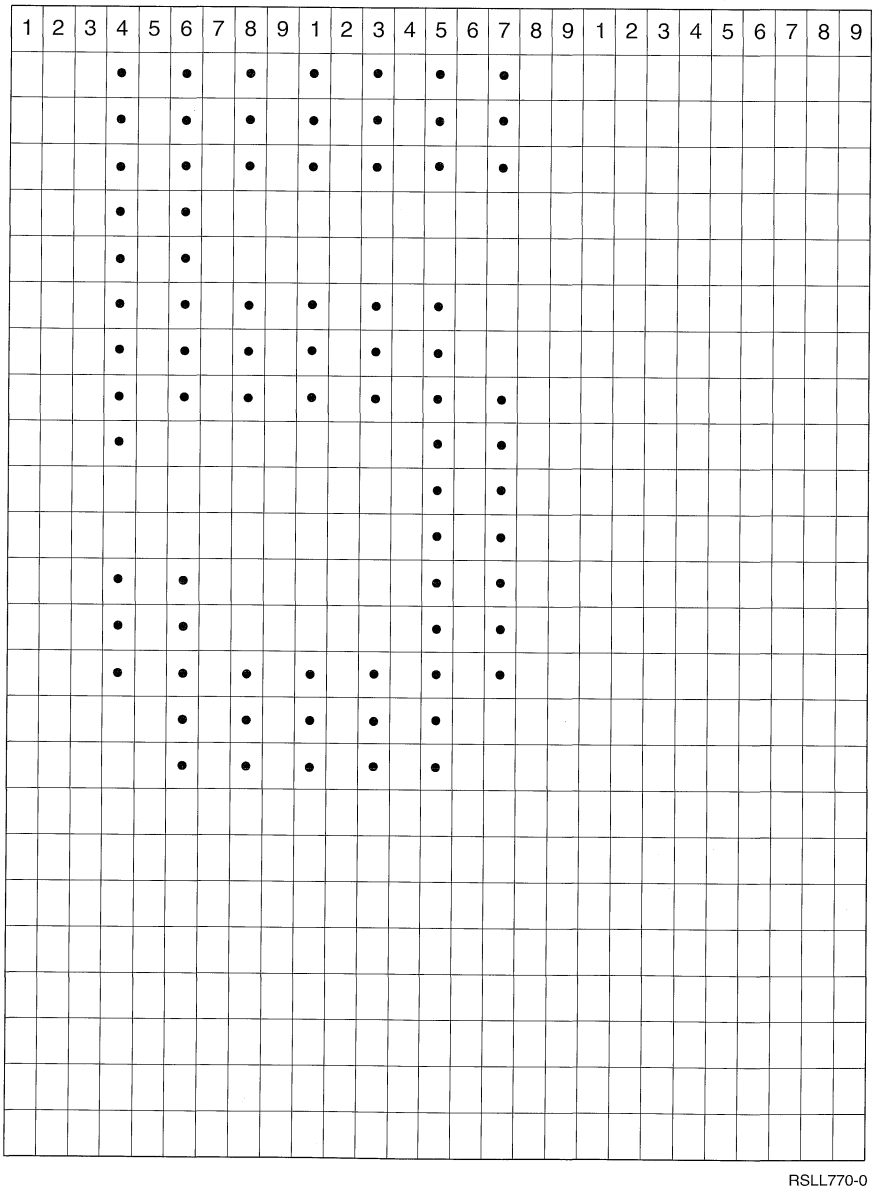
Code Points

	1	2	3	4	5	6	7	8	9
51	00	00	00	FF	00	FF	00	E7	00
52	E7	00	E7	00	E7	00	E1	00	00
53	00	00	00	9C	00	1F	00	07	00
54	07	00	07	00	FF	00	FC	00	00
—									
—									
—									
—									

RSLL769-0

Figure 4-29. Completing Code Points for Example 2 (Points 51 through 54)

In the grid pattern, mark the dot patterns for as many as nine print columns (three across and three down), as shown in Figure 4-30.



RSLL770-0

Figure 4-30. Specifying the Grid Pattern for Example 2 (Points 55 through 58)

Use Figure 4-25 on page 4-35 to determine which hex digit to specify for each half-column in the grid pattern.

For each print position, complete one row of the grid pattern (one pair of hex digits to each box) as shown in Figure 4-31 for code points 55 through 58.

Code Points

	1	2	3	4	5	6	7	8	9
55	00	00	00	00	FF	00	E7	00	E7
56	00	E7	00	E7	00	E3	00	00	00
57	00	00	00	00	1E	00	07	00	07
58	00	07	00	07	00	FE	00	00	00
—									
—									
—									
—									

RSLL772-0

Figure 4-31. Completing the Code Points for Example 2 (Points 55 through 58)

Example 3:

Figure 4-32 uses a dot matrix for a large graphic to show how to specify DFNCHR at the file level for a large graphic three columns wide by two lines high.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                                 1    2
00020A                                       DFNCHR(X'B1' X'FF00E000D000C800C4' +
00030A                                              X'B2' X'00C200C100C100C200' +
00040A                                              X'B3' X'C400C800D000E000FF' +
00050A                                              X'B4' X'FF0007000800130023' +
00060A                                              X'B5' X'004300830083004300' +
00070A                                              X'B6' X'230013000B000700FF' +
00080A                                              X'B7' X'00FF00F000D800CC00' +
00090A                                              X'B8' X'C600C100C100C100C6' +
00100A                                              X'B9' X'00CC00D800F000FF00' +
00110A                                              X'BA' X'00FF000F001B003300' +
00120A                                              X'BB' X'630083008300830063' +
00130A                                              X'BC' X'0033001B000F00FF00')
00140A          R RECORD1                            CPI(15)
00150A                                  58  4DFT(X'B1B2B3')   TRNSPY
00160A                                  58  4DFT(X'B7B8B9')   TRNSPY
00170A                                  59  4DFT(X'B4B5B6')   TRNSPY
00180A                                  59  4DFT(X'BABBBC')   TRNSPY
     A
```

Figure 4-32. Specifying the DFNCHR Keyword (Example 3)

Figure 4-32 redefines 12 code points (hex B1 through hex BC) **1**. Each column of the three-by-two character prints twice so that adjacent horizontal dots can print. The hex codes **2** define the dot matrix pattern.

The DDS example in Figure 4-32 prints an X inside a □ using three print columns (on lines 58 and 59 on a printer form).

Note: The file should be at 9 lpi to avoid a horizontal gap in the large character (LPI parameter on the CRTPRTF, CHGPRTF, or OVRPRTF command).

Mark the dot patterns for as many as nine print columns (three across and three down) in the grid, as shown in Figure 4-33 on page 4-40.

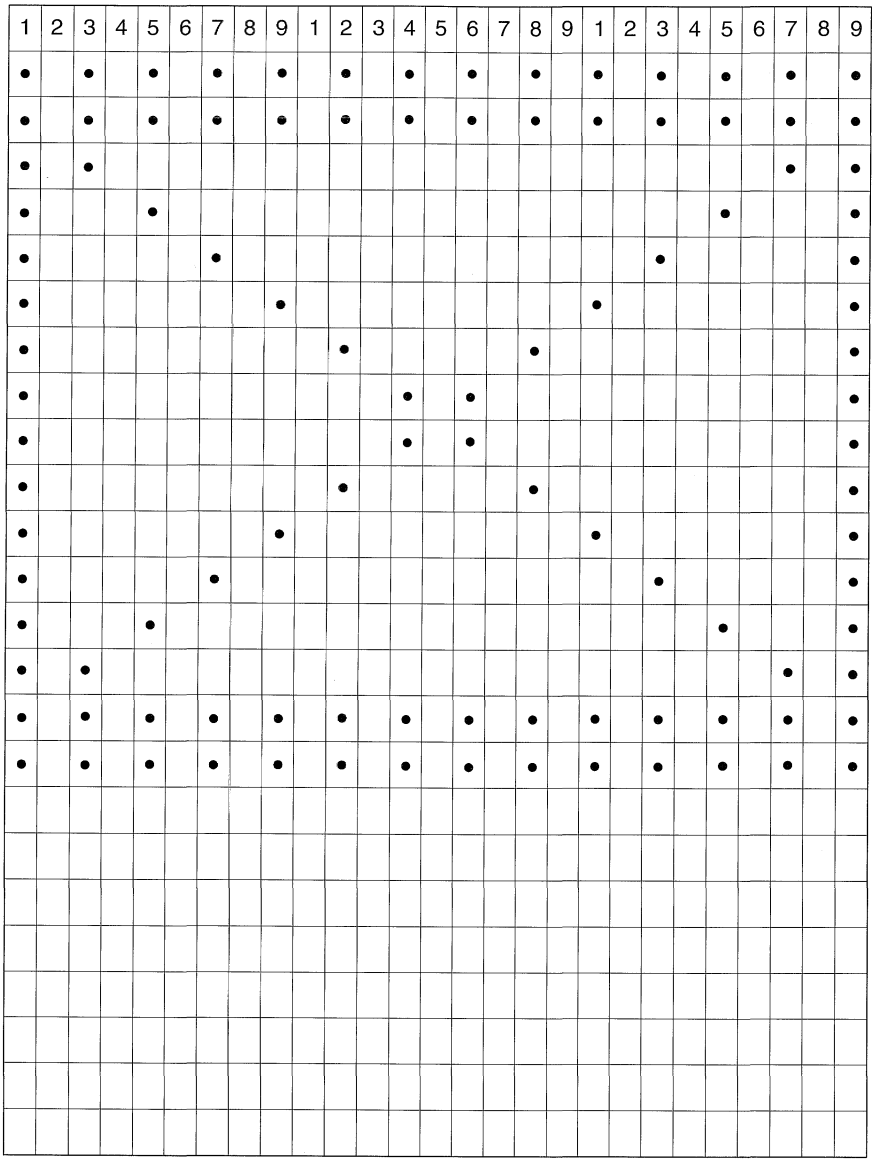


Figure 4-33. Specifying the Grid Pattern for Example 3 (Points B1 through B6)

Use Figure 4-25 on page 4-35 to determine which hex digit to specify for each half-column in the grid. For each print position, complete one row of the grid pattern (one pair of hex digits to each box) as shown in Figure 4-34 on page 4-41.

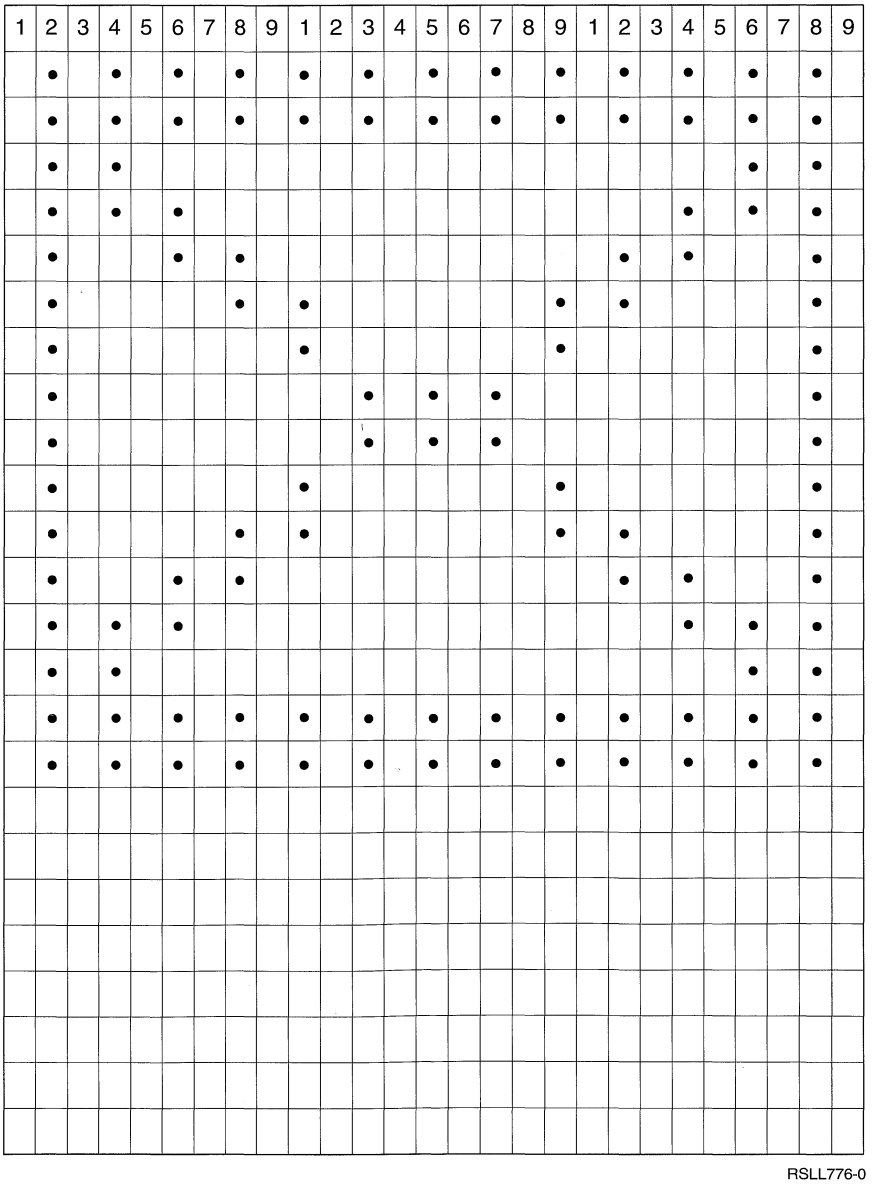
Code Points

	1	2	3	4	5	6	7	8	9
B1	FF	00	E0	00	D0	00	C8	00	C4
B2	00	C2	00	C1	00	C1	00	C2	00
B3	C4	00	C8	00	D0	00	E0	00	FF
B4	FF	00	07	00	0B	00	13	00	23
B5	00	43	00	83	00	83	00	43	00
B6	23	00	13	00	0B	00	07	00	FF
—									
—									

RSLL775-0

Figure 4-34. Completing the Code Points for Example 3 (Points B1 through B6)

As shown in Figure 4-35, mark the dot patterns in the grid for as many as nine print columns (three across and three down).



RSLL776-0

Figure 4-35. Specifying the Grid Pattern for Example 3 (Points B7 through BC)

Use Figure 4-25 on page 4-35 to determine which hex digit to specify for each half-column in the grid. For each print position, complete one row of the grid pattern (one pair of hex digits to each box) as shown in Figure 4-36 on page 4-42.

Code Points

	1	2	3	4	5	6	7	8	9
B7	00	FF	00	F0	00	D8	00	CC	00
B8	C6	00	C1	00	C1	00	C1	00	C5
B9	00	CC	00	D8	00	F0	00	F0	00
BA	00	FF	00	0F	00	1B	00	33	00
BB	63	00	83	00	83	00	83	00	63
BC	00	33	00	1B	00	0F	00	FF	00
—									
—									

RSLL777-0

Figure 4-36. Completing the Code Points for Example 3 (Points B7 through BC)

DFT (Default)

Use the DFT keyword to specify a constant value for constant (unnamed) fields.

The format of the keyword is:

```
DFT('value')
'value'
DFT(X'hexadecimal-value')
X'hexadecimal-value'
```

Constant values can be:

- A character value, in which each character prints as you specify it. The number of characters is equal to the printed length of the field. (Within the value, two adjacent apostrophes are printed as one.) See Figure 4-37 on page 4-43.
- A hexadecimal value, in which two characters identify a code point in the character set. These characters print in this field (define alternate characters using DFNCHR). The printed length is half the number of characters you specify between apostrophes. You must specify the TRNSPY keyword if you specify a hexadecimal value for DFT. See Figure 4-38 on page 4-43.

You can specify DFT implicitly by omitting DFT and the parentheses (this is true for both character and hexadecimal values). Simply specify the value within apostrophes. For hexadecimal values, you must also precede the value with an X.

The EDTCDE and EDTWRD keywords cannot be specified with the DFT keyword.

Option indicators are not valid for this keyword. However, they can be used to condition the constant field with which this keyword is specified, specifying the last option indicator on the same line as the field location.

Figure 4-37 on page 4-43 shows how to specify DFT using character values.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SUPPLIES
00020A            PENS          20      2  1
00030A            INK           20      3  1
00040A            PAPER         20      4  1
00050A                                    7  9DFT('ON')
00060A                                    8  9'ON'
00070A
00080A  01                                12  1'Hotel name: ''Terrace Inn'''
00100A
00110A  02                                12  1'Hotel name: ''Riverview Inn'''
     A
```

Figure 4-37. Specifying the DFT Keyword (Character Values)

The specifications DFT('ON') and 'ON' are equivalent and show the difference between specifying DFT explicitly and implicitly.

If indicator 01 is on, this prints:

```
Hotel name: 'Terrace Inn'
```

If indicator 02 is on and indicator 01 is off, this prints:

```
Hotel name: 'Riverview Inn'
```

Figure 4-38 shows how to specify DFT for a constant field containing an alternate character.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
                                          1        2
00010A          R RECORD                          DFNCHR(X'7C' X'007E813CC324817E00')
00020A                                  3 58  4DFT(X'7C')   TRNSPY   4
00030A                                       +2DFT('1982')
     A
```

Figure 4-38. Specifying the DFT Keyword (Hexadecimal Values)

The constant field for which DFT is specified **1** appears on line 58, position 4. The character defined for hex 7C prints in this field. DFNCHR **2**, specified at the record level for this example, defines hex 7C as a copyright mark.

The following are equivalent ways to specify the value as defined in this example **3**:

```
DFT(X'7C')
X'7C'
DFT('©')
'©'
```

The TRNSPY keyword **4** is required when hexadecimal values are specified for DFT.

DLTEDT (Delete Edit)

Use this field-level keyword to specify that the OS/400 program is to ignore any edit code or edit word keywords specified for the referenced field. If a field description is referred to from a database file, DLTEDT prevents certain information from being referenced.

This keyword has no parameters.

This keyword is valid only when you specify R in position 29 for this field, and also specify either the REF or the REFFLD keyword.

If replacement edit information is needed, specify the EDTCDE or the EDTWRD for the field you define. The new keyword overrides the editing from the referenced field. In this case, you do not need to specify DLTEDT.

Option indicators are not valid for this keyword.

Figure 4-39 shows how to specify the DLTEDT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            AMT       R           5  20DLTEDT
     A
```

Figure 4-39. Specifying the DLTEDT Keyword

DRAWER (Drawer)

Use this record-level keyword to specify the drawer from which noncontinuous forms will be selected.

The format of the keyword is:

```
DRAWER(drawer-number)
```

Drawer-number specifies the drawer from which the paper or the envelope is to be fed. Valid values are 1, 2, 3, and *E1 as follows:

1 The paper is fed from the first drawer on the sheet-feed paper handler.

2 The paper is fed from the second drawer on the sheet-feed paper handler.

3 The paper is fed from the third drawer on the sheet-feed paper handler.

*E1 The envelope is fed from the envelope drawer on the sheet-feed paper handler.

If you do not specify the DRAWER keyword, the value specified on the DRAWER parameter of the CRTPRTF, CHGPRTF or OVRPRTF command determines the paper source drawer.

DRAWER is ignored at run time if it is not specified on a page boundary. The printer is on a page boundary when no named or constant fields are processed for a page. Once a named or constant field is processed, the printer is no longer on a page boundary. The printer is on a page boundary again when a SKIP or SPACE keyword is processed that causes the printer to move to a new page.

DRAWER, SKIP, and SPACE keywords are processed in the following order:

SKIPB
SPACEB
DRAWER
SPACEA
SKIPA

DRAWER is in effect only for the record format specified. Once records with the specified record format are processed, the paper-source drawer for the next record

format (if the DRAWER keyword is not specified) is the drawer specified at the file level (CRTPRTF, CHGPRTF, or OVRPRTF command).

For files created with DEVTYPE(*SCS), if the DRAWER keyword is specified on a record format that spans several pages, it remains in effect only for the page on which it is specified.

You cannot specify DRAWER on the same record format with the CPI keyword or a record level DFNCHR keyword. If any format in the file contains both DRAWER and either CPI or a record-level DFNCHR keyword, the file is not created.

Option indicators are valid for this keyword.

Figure 4-40 shows how to specify the DRAWER keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1                  SKIPB(3)
00020A            FIELD1        10         1SPACEA(1)
00030A            FIELD2         5         1SPACEA(1)
00040A
00050A          R RECORD2                  DRAWER(2)
00060A            FIELD3         5         1
00070A            FIELD4         5         6SKIPA(1)
00080A
00090A          R RECORD3                  DRAWER(2)
00100A            FIELD5        10         1SPACEA(1)
00110A            FIELD6        10         1SKIPA(1)
00120A            FIELD7        10         1SPACEA(1)
00130A            FIELD8        10         1SPACEA(1)
00140A
00150A          R RECORD4
00160A            FIELD9        10         1SKIPB(30)
00170A            FIELD10       10        21
00180A          R RECORD5                    SKIPB(3)
00190A            FIELD11       10         1SPACEA(1)
00200A            FIELD12       10         1SPACEA(1)
     A
```

Figure 4-40. Specifying the DRAWER Keyword

The printer is not on a page boundary after record format RECORD1 is processed. When record format RECORD2 is processed, DRAWER is ignored and paper continues to come from the source drawer previously specified (file level). Because SKIPA(1) is specified for FIELD4 of RECORD2, the printer is on a page boundary after RECORD2 is processed. The paper for both pages of RECORD3 comes from drawer 2. The paper source for record formats RECORD4 and RECORD5 is the drawer specified at the file level (drawer 1 in this example). But because RECORD4 starts in the middle of a page, it prints on the same page as RECORD3 (drawer 2). Record format RECORD5 prints on a different page (SKIPB(3)) and prints on paper from drawer 1.

EDTCDE (Edit Code)

Use this keyword to edit output-capable numeric fields.

The format of the keyword is:

```
EDTCDE(edit-code [* | floating-currency-symbol])
```

Depending on which edit code you specify, you can change the appearance of the printed fields as follows:

- Leading zeros are suppressed.
- The field can be punctuated with commas and periods to show decimal position and to group digits by threes.
- Negative values can be printed with a minus sign or CR to the right.
- Zero values can be printed as zeros or blanks.
- Asterisks can be printed to the left of significant digits to provide asterisk protection.
- A currency symbol (corresponding to the system value QCURSYM) can be printed immediately to the left of the farthest right significant digit (called a **floating-currency symbol**). For fixed-currency symbols, use the EDTWRD keyword.
- The field can be further edited using a user-defined edit code. See "User-Defined Edit Codes" on page 4-49 for more information.

EDTCDE covers most editing requirements. Use the EDTWRD keyword when EDTCDE is not sufficient.

EDTCDE is valid only for fields with S or a blank in position 35 (Data Type).

You cannot specify EDTCDE and EDTWRD for the same field. If you specify EDTCDE for a field previously defined in a database file, you need not specify EDTCDE for the field you are defining. Instead, specify R in column 29 to refer to the previously defined field. The editing specified for that field is included in the printer file. If you specify length, data type, or decimal columns for a printer file field, editing specified for the referenced field is not included in the printer file and you must specify editing again in the printer file.

The DFT keyword cannot be specified with the EDTCDE keyword.

Option indicators are not valid for this keyword.

For more information on the EDTCDE keyword, see "EDTCDE (Edit Code)" on page 3-99.

You can specify two kinds of edit codes: OS/400 edit codes and user-defined edit codes.

OS/400 Edit Codes

The OS/400 edit codes are:

1 through 4
A through D
J through Q
X through Z

Note: The AS/400 system hardware operates with a preferred sign of F, which is equivalent to using edit code X. If the DATE or TIME keyword is specified with edit code X, the separator character is not displayed.

Asterisk Fill or Floating Currency Symbol: You can optionally specify asterisk fill or floating currency symbol with edit codes 1 through 4, A through D, and J through Q.

When you specify asterisk fill, an asterisk (*) is written for each zero suppressed. A complete field of asterisks is printed for a zero-balance field.

When you specify floating-currency symbol, the symbol appears to the left of the first significant digit. It does not print on a zero balance when an edit code is used that suppresses the zero balance. The symbol you specify must match the system value for the currency symbol (QCURSYM). (The symbol must match when the file is created. It does not have to match when the file is used.)

Note: If an edit code is changed after a file is created, the new edit code is not used unless the file is re-created. Instead, the editing specified at the time the file was created continues to be used.

The following table summarizes the functions provided by OS/400 edit codes.

Figure 4-41. Summary Chart for OS/400 Edit Codes

Edit Codes	Commas[1] Printed	Decimal Points[1] Printed	Signs Printed When Negative Number	Blank Value of QDECFMT System Value	I Value of QDECFMT System Value	J Value of QDECFMT System Value	Leading Zero Suppressed
1	Yes	Yes	No sign	.00 or 0	,00 or 0	0,00 or 0	Yes
2	Yes	Yes	No sign	Blanks	Blanks	Blanks	Yes
3		Yes	No sign	.00 or 0	,00 or 0	0,00 or 0	Yes
4		Yes	No sign	Blanks	Blanks	Blanks	Yes
A	Yes	Yes	CR	.00 or 0	,00 or 0	0,00 or 0	Yes
B	Yes	Yes	CR	Blanks	Blanks	Blanks	Yes
C		Yes	CR	.00 or 0	,00 or 0	0,00 or 0	Yes
D		Yes	CR	Blanks	Blanks	Blanks	Yes
J	Yes	Yes	–(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
K	Yes	Yes	–(Minus)	Blanks	Blanks	Blanks	Yes
L		Yes	–(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
M		Yes	–(Minus)	Blanks	Blanks	Blanks	Yes
N	Yes	Yes	–(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
O	Yes	Yes	–(Minus)	Blanks	Blanks	Blanks	Yes
P		Yes	–(Minus)	.00 or 0	,00 or 0	0,00 or 0	Yes
Q		Yes	–(Minus)	Blanks	Blanks	Blanks	Yes
Y[2]							Yes
Z[3]							Yes

Notes:

1. The QDECFMT system value determines the decimal point character (period as used in the U.S.), the character used to separate groups of three digits (comma as used in the U.S.), and the type of zero suppression (depending on comma and period placement). For more information on the QDECFMT system value, see the *Work Management Guide*.
2. The Y edit code suppresses the farthest left zero of a date field that is three to six digits long, and it suppresses the two farthest left zeros of a field that is seven positions long. The Y edit code also inserts slashes (/) between the month, day, and year according to the following pattern:
   nn/n
   nn/nn
   nn/nn/n
   nn/nn/nn
   nnn/nn/nn

   If the DATE keyword is specified with EDTCDE(Y), the separator character used is the job attribute, DATSEP at run time. If a separator character is not specified on the DATSEP job attribute, the system value, QDATSEP, is used (where slash (/) is the default value). If, at file creation time, DATFMT is JUL (Julian), the date is formatted as nnnnn. If EDTCDE(Y) is specified, the date is formatted as nn/nnn, where the slash (/) represents the job date separator.
3. The Z edit code removes the sign (plus and minus) from a numeric field. The sign of the units column is changed to a hexadecimal F before the field is written.

User-Defined Edit Codes

Edit codes 5 through 9 are user-defined edit codes. A user-defined edit code can do more editing than an OS/400 edit code. For example, you may need to edit numbers that include hyphens (such as telephone numbers) or more than one decimal point. You can use user-defined edit codes for these functions. These edit codes are named QEDIT5, QEDIT6, QEDIT7, QEDIT8, and QEDIT9, and can be referred to in DDS or a high-level language program by number (5, 6, 7, 8, or 9).

A user-defined edit code is an OS/400 object and must exist before printer file creation. It is created using the Create Edit Description (CRTEDTD) command. When you create a printer file in which a user-defined edit code is specified, editing information is extracted from the previously created edit description. Changing a user-defined edit code after printer file creation does not affect the printer file unless the printer file is re-created.

The following table shows valid edit codes with examples of unedited source data and edited output. Zero suppression and decimal characters are determined by the system value QDECFMT. The date separator character is determined by the job attribute DATSEP. In this figure, QDECFMT is assumed to equal x (blank), and DATSEP is assumed to equal / (slash).

Figure 4-42 (Page 1 of 2). Valid Edit Codes, Source Data, and Edited Output

Edit Codes	Positive Number — Two Decimal Positions	Positive Number — No Decimal Positions	Negative Number — Three Decimal Positions[1]	Negative Number — No Decimal Positions	Zero Balance— Two Decimal Positions[1]	Zero Balance — No Decimal Positions[1]
Unedited	1234567	1234567	xxxx125–	xxxx125–	xxxxxx	xxxxxx
1	12,345.67	1,234,567	.125	125	.00	0
2	12,345.67	1,234,567	.125	125		
3	12345.67	1234567	.125	125	.00	0
4	12345.67	1234567	.125	125		
A	12,345.67	1,234,567	.125CR	125CR	.00	0
B	12,345.67	1,234,567	.125CR	125CR		
C	12345.67	1234567	.125CR	125CR	.00	0
D	12345.67	1234567	.125CR	125CR		
J	12,345.67	1,234,567	.125–	125–	.00	0
K	12,345.67	1,234,567	.125–	125–		
L	12345.67	1234567	.125–	125–	.00	0
M	12345.67	1234567	.125–	125–		
N	12,345.67	1,234,567	-.125	-125	.00	0
O	12,345.67	1,234,567	-.125	-125		

Figure 4-42 (Page 2 of 2). Valid Edit Codes, Source Data, and Edited Output

Edit Codes	Positive Number — Two Decimal Positions	Positive Number — No Decimal Positions	Negative Number — Three Decimal Positions[1]	Negative Number — No Decimal Positions	Zero Balance— Two Decimal Positions[1]	Zero Balance — No Decimal Positions[1]
P	12345.67	1234567	-.125	-125	.00	0
Q	12345.67	1234567	-.125	-125		
Y[2]	123/45/67	123/45/67	0/01/25	0/01/25	0/00/00	0/00/00
Z[3]	1234567	1234567	125	125		

Notes:

1. The x represents a blank.
2. The Y edit code suppresses the farthest left zero of a date field that is three to six digits long, and it suppresses the two farthest left zeros of a field that is seven positions long. For more information, see the second footnote in Figure 3-78 on page 3-100.
3. The Z edit code removes the sign (plus or minus) and suppresses leading zeros.

Figure 4-43 shows how to specify the EDTCDE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00050A           PRICE          5  2   5  2EDTCDE(J *)
     A
```

Figure 4-43. Specifying the EDTCDE Keyword

EDTWRD (Edit Word)

If you cannot accomplish the desired editing by using the EDTCDE keyword, specify an edit word instead. An edit word specifies the form in which the field values are to print and clarifies the data by inserting characters, such as decimal points, commas, floating- and fixed-currency symbols, and credit balance indicators. Also use it to suppress leading zeros and to provide asterisk fill protection.

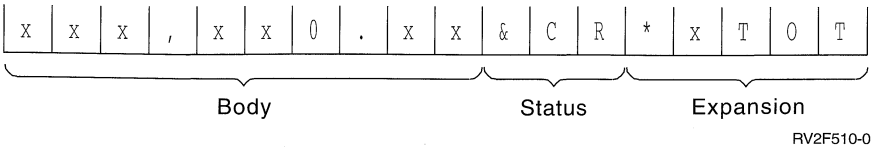
The format of the keyword is:

```
EDTWRD('edit-word')
```

If you specify EDTWRD in a field previously defined in a database file, you need not specify EDTWRD for the field you are defining. Instead, specify R in column 29 to refer to the previously defined field. The editing specified for the referenced field is then included in the printer file. If, however, you specify length, data type, or decimal positions for a printer file field, editing specified for the referenced field is not included in the printer file, and you must specify editing in the printer file.

Parts of an Edit Word

An edit word consists of: the body, the status, and the expansion. The following illustration shows the three parts.



RV2F510-0

The body contains the digits transferred from the data field to the output record. It begins at the farthest left column of the edit word and ends with the farthest right character that can be replaced by a digit. The number of blanks (plus one zero or an asterisk) it contains is equal to the number of digits of the data field to be edited.

The status positions display the sign (+ or –) of the data field. It continues to the right of the body and has either a CR (credit) or – (minus) symbol, which print only when the field is negative. Edit words without the CR or – symbol have no status columns.

The expansion positions are not changed by the edit operation. The expansion starts at the first position to the right of the status (or body, if status is not specified) and ends with the farthest right character of the edit word.

Forming the Body of an Edit Word

The following characters have special meanings when used in the body of an edit word:

Blank: A blank is replaced with the character from the corresponding position of the data field. A blank position is referred to as a digit position.

Ampersand: An ampersand causes a blank in the edited field. The ampersand is not printed.

Zero: To stop zero suppression, place a zero in the farthest right position where it is to stop. The zero is then replaced with the character from the corresponding position of the data field, unless that character is a zero. Any zeros in the data that appear to the right of the stop-zero-suppression character are printed. The stop-zero-suppression character is considered a digit position. At least one leading zero is suppressed. Each zero that is suppressed is replaced by a blank.

Asterisk: Placing an asterisk in the farthest right position where zero suppression is to stop, stops zero suppression and replaces the zeros with asterisks (asterisk protection). An asterisk preceding a zero is interpreted as representing asterisk protection. In this case, the zero prints as a constant. Any asterisks or zeros to the right of the stop-zero-suppression character are constants.

Currency Symbol: A currency symbol coded immediately to the left of the zero suppression code causes the insertion of a currency symbol in the position to the left of the first significant digit. It is called the *floating-currency symbol* when used in this manner.

A currency symbol coded in the farthest left column of the edit word is fixed and prints in the same location each time. When used in this manner, it is called the *fixed-currency symbol.*

The currency symbol is not considered a digit-replace position. This symbol must correspond to the system value QCURSYM.

Decimals and Commas: Decimals and commas are printed in the same relative positions in which they are coded in the edit word unless they are to the left of the first significant digit. In that case, they are blanked out or replaced by an asterisk.

All other characters are printed if they are to the right of significant digits in the data field. If they are to the left of the high-order significant digits in the data, they are blanked out or replaced by asterisks if asterisk protection is being used.

Forming the Status of an Edit Word

The following characters have special meanings when used in the status of an edit word.

Ampersand: Causes a blank in the edited output field. An ampersand cannot be placed in the edited output field.

CR or Minus Symbol: If the sign in the edited output field is plus (+), these positions are blanked out. If the sign in the edited output field is minus (–), these positions remain undisturbed.

Formatting the Expansion of an Edit Word

The characters in the expansion portion of an edit word are always written. The expansion cannot contain blanks. If a blank is required in the edited output field, specify an ampersand in the body of the edit word.

Follow these guidelines when specifying a valid edit word:

- You cannot specify both EDTWRD and EDTCDE for the same field.
- You must enclose the edit word in apostrophes.
- The EDTWRD keyword is valid for numeric-only fields (S specified in position 35).
- The sum of the blanks and stop-zero-suppression characters (digit positions) in the edit word must equal the length of the field.
- If you use the floating-currency symbol, it is not counted as a digit position. For example, if you specify the floating-currency symbol for a field length of 7 and 2 decimal positions, the edit word is:

  ```
  EDTWRD('___$0.__')
  ```

  where _ represents a blank.
- If you want to show a negative sign with a negative number, include a sign in the edit word. Use either the minus sign (–) or the letters CR (credit) to the right of the last digit replacement character. These print only if the number is negative.

Option indicators are not valid for this keyword.

The DFT keyword cannot be specified with the EDTWRD keyword.

For more information on the EDTWRD keyword, see "EDTWRD (Edit Word)" on page 3-105.

Figure 4-44 shows how to specify the EDTWRD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A           CRYCST         7  2   5  2EDTWRD('    $0.  ')
     A
```

Figure 4-44. Specifying the EDTWRD Keyword

Figure 4-45 on page 4-53 shows sample edit words with the program value of the field and the printed value of the field (as edited).

Edit Word	Program Value	Printed As
' , , 0. &CR*'	0000000005-	.05 CR*
' , , $0. CR**'	0000000005+	$0.05 *
' , , $0. CR**'	0034567890-	$345,678.90CR**
'$, , 0. '	0000000000	$ .00
'$& , , 0. &-&GROSS'	1234567890-	$ 12,345,678.90 - GROSS
' , , * . &-'	0000135792	***1,357.92
	0000135792	0000135792
	0000135792-	0000135792
' '	0000000000	
' '	0000135678+	135678
' '	0000135678-	135678
' 0'	0000135678-	135678
'0 '	0000135678+	0001356 78
'$ &-&NET'	0000135678+	$ 135678 NET
'$ &-&NET'	0000135678-	$ 135678 - NET
'$0 -&NET'	0000135678	$ 0001356 78 NET
' $0 &CR*'	0000135678-	$135678 CR*
' $0 &CR*'	1234567809-	$12345678 09 CR*
' *&CR'	0000000000-	********** CR
' * &CR'	0000000000-	********00 CR
'* '	0000135678-	*0001356 78
' , , . &CR*&NET'	0000135678-	1,356.78 CR* NET
' , , . &CR*&NET'	0000135678	1,356.78 * NET
' , , $0. '	0000000005	$.05
' , , $0. CR'	0001356789-	$13,567.89CR
' , , *. *CR**'	0000135678+	***1,356.78* **
' , , DOLLARS CENTS'	0000135678	1,356DOLLARS78CENTS
'0 - - '	095140036	95-14-0036
'& ,* 0, '	0130579	**130,579
' - - &LATER'	093076	9-30-76 LATER
' & & &LATER'	093076	9 30 76 LATER
' / / '	100176	10/01/76

RV2F514-0

Figure 4-45. Sample Edit Words

ENDPAGE (End Page)

Use this record-level keyword to eject the current page after the record is printed.

This keyword has no parameters.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when ENDPAGE is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

An error message is issued if a constant field is specified in a record format where the ENDPAGE keyword is also specified.

You cannot specify ENDPAGE with the following keywords:

SPACEA
SPACEB
SKIPA
SKIPB

Option indicators are valid for this keyword.

Figure 4-46 shows how to specify the ENDPAGE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1                       ENDPAGE
     A            FLD1           5A    66 10
     A            FLD1
     A*
     A          R REC2
     A  01                                   ENDPAGE
     A            FLD1           5A         POSITION(8.5 10.2)
     A
```

Figure 4-46. Specifying the ENDPAGE Keyword

A page eject always occurs after REC1 prints.

If indicator 01 is on when the application writes REC2, a page eject occurs after REC2 prints. If indicator 01 is off when the application writes REC2, no page eject occurs.

FLTFIXDEC (Floating-Point to Fixed Decimal)

Use this field-level keyword to print a number in a floating-point field in fixed decimal notation.

This keyword has no parameters.

When you use FLTFIXDEC, the floating-point number is first converted to the equivalent number with an exponent of zero. If the resulting number (digits and exponent) fits in the field defined by the length and decimal positions values, the number is printed with the exponent suppressed and aligned at the decimal point. If the number does not fit in the field, the number prints in standard floating-point form, n.nnnnnnE+nnn. When the FLTFIXDEC keyword is specified, the length of the field is the DDS length plus two (the sign and the decimal point). The minimum length of the field is six.

When the number is too large or small for the fixed-point form, specified by the FLTFIXDEC keyword with the total digits and fractional digits specified for the field, a floating-point form prints that presents the significand as follows (the significand is the string of digits with the decimal point to the left of the exponent sign E):

- Total significand decimal digits: DDS total digits minus 5
- Fractional significand digits: DDS total digits minus 6

Option indicators are not valid for this keyword.

Figure 4-47 on page 4-55 shows how to specify the FLTFIXDEC keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECFMT1
     A            FIELD1        10F 3   1  2FLTFIXDEC
     A                                       FLTPCN(*DOUBLE)
     A
```

Figure 4-47. Specifying the FLTFIXDEC Keyword

These output numbers would be converted as follows:

Output Number	Printed As
-4.99994321000000E-004	'-4.0000E-004'
-5.00010000000000E-004	' -0.001'
-2.69123400000000E-002	' -0.027'
-0.00000000000000E+000	' 0.000'
0.00000000000000E+000	' 0.000'
2.71828182845900E+003	' 2718.282'
3.14159000000000E-052	' 3.14163-052'
9.87654321012345E+006	' 9876543.210'
9.99999999960000E+006	' 1.0000E+007'

FLTPCN (Floating-Point Precision)

Use this keyword to specify the precision of a floating-point field.

The format of the keyword is:

```
FLTPCN(*SINGLE | *DOUBLE)
```

The *SINGLE parameter specifies single precision and the *DOUBLE parameter specifies double precision. This keyword is valid for floating-point fields only (data type F).

A single-precision field can be up to 9 digits; a double-precision field can be up to 17 digits. If you specify a field length greater than 9 (single precision) or 17 (double precision), an error message is issued and the file is not created.

Option indicators are not valid for this keyword.

Figure 4-48 shows how to specify the FLTPCN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00090A            FIELDA        17F 4   2  3FLTPCN(*DOUBLE)
     A
```

Figure 4-48. Specifying the FLTPCN Keyword

FIELDA is a floating-point field with double precision.

FNTCHRSET (Font Character Set)

Use this field- or record-level keyword to specify the font for printing a named or constant field within a record.

The format of the keyword is:

```
FNTCHRSET([library-name/]font-character-set [library-name/]code-page)
```

The font-character-set and code-page parameters are required. Both can be up to 8 characters long.

Use the optional library-name parameter to further qualify the font character set or code page. If library-name is not specified, *LIBL is used to search for the font character set and code page. If *LIBL is used, the system-supplied font libraries are added to the library list when searching for the requested font.

Note: If an application uses private resources (for example, fonts, page segments, overlays, or GDF files not distributed with the system), be aware of the following. When referencing these resources, if you specify *LIBL or you do not specify a library name, the resources must be available through the library list used by the application creating the spooled file.

The font character set and code page values are validated at print time. An error message is issued if they are not valid.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when FNTCHRSET is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

FNTCHRSET cannot be specified at the same level as the FONT and CDEFNT keywords.

Option indicators are valid for this keyword.

Figure 4-49 shows how to specify the FNTCHRSET keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1
     A            FLD1A         14A     3  8FNTCHRSET(C0S0CE12 T1L0PCHN)
     A*
     A
```

Figure 4-49. Specifying the FNTCHRSET Keyword

FLD1A specifies font character set C0S0CE12 and code page T1L0PCHN. *LIBL is used to search for the font character set and code page.

FONT (Font)

Use this record- or field-level keyword to specify the font ID for printing a named or constant field or fields within a record.

The format of the keyword is:

```
FONT(font-identifier [point-size])
```

The font-identifier is a required parameter and must be the first parameter following the keyword. Specify either a numeric font identifier or a graphic font name, or *VECTOR.

Use the optional point-size parameter to further define a numeric font that specifies a point size. Specify the point-size parameter as an expression of the form (*POINTSIZE *value*). The valid values for this parameter are 0.1 through 999.9.

A warning message is issued at create time if you specify the point-size parameter on the FONT keyword with a graphic font name, or *VECTOR. In that case, the point-size parameter is ignored.

If you do not specify this keyword, the font ID and point size are set by the font parameter on the CRTPRTF, CHGPRTF, or OVRPRTF command. If you specify this keyword at the record level, all fields in the record format use the same font ID and point size except those for which you specify the FONT keyword at the field level.

You may specify graphic fonts (alphanumeric characters) or hardware fonts (numeric font identifiers). For graphic fonts, use graphic symbol sets (GSS) available with an AS/400 system, GDDM, PGR, and BGU. Only vector symbols (where each character is built with a set of straight or curved lines) are supported; however, most of the vector symbol sets supplied by an AS/400 system and GDDM are supported. Image symbols are not supported. In searching for the graphic symbol set, *LIBL will be used for the qualified library name.

The name of a graphic font can consist of up to 10 alphanumeric characters.

The hardware font can consist of up to 10 digits and must be a registered font number. See the FONT parameter on the CRTPRTF command in the *CL Reference* manual for a complete list of numeric fonts.

You may specify *VECTOR on the FONT keyword to take advantage of vector fonts on the 4234 IPDS printer. Vector fonts print expanded characters faster than they can be printed using the PRTQLTY(*DRAFT) keyword. Use the CHRSIZ keyword to specify expanded characters.

Note: When you specify FONT(*VECTOR) with the CHRSIZ keyword, the 4234 printer uses a default code page.

Vector fonts are valid only for the following characters:

A through Z
0 through 9
Special characters (. + $ * - / % and a blank)

If the data to be printed contains any characters other than these, all characters are printed using a default font on the printer.

FONT(*VECTOR) has no effect on characters that have not been expanded. If FONT(*VECTOR) is specified on a record or field for which CHRSIZ (1 1) applies or to which no CHRSIZ keyword applies, a warning message is issued.

Note: If you use FONT(*VECTOR) on a 4224 or 3812 printer, the printer uses a default font and code page.

The font name or number and the point size values are not checked during file creation. If the specified font-id and point size values are not valid, a diagnostic is issued while the record prints and the keyword is not used.

When FONT is specified at the field level, overlapping fields are not diagnosed.

When you use a graphics font on the CRTPRTF, CHGPRTF, or OVRPRTF command, the font ID has an implied page code associated with it. To get the desired code, you must use the proper font ID; the code page specified on the CHRID parameter is not used.

If you specify OCR-A and OCR-B fonts with the CHRID keyword, the fonts require code pages 892 and 893, respectively.

A warning message is issued at create-time if a FONT DDS keyword is specified in a file created with DEVTYPE(*IPDS) and FONT(*DEVD). For SCS printer files, the FONT keyword is ignored when the record or field is printed. For IPDS printers, the FONT keyword can be changed at the record or field level.

When printing a file that uses the FONT keyword to an IPDS AFP(*YES) printer that does not support registered fonts, a font substitution is performed.

When you specify FONT(*CPI) with either the CRTPRTF, CHGPRTF, or OVRPRTF command to a device that uses font support, the host system selects a font with the pitch of the CPI for the current printer file.

FONT(graphic-font-name) and CHRID cannot apply to the same field. The CHRID keyword is ignored if:

- You specify FONT(graphic-font-name) and CHRID on the same field.
- You specify FONT(graphic-font-name) at the record level and a field in the record specifies CHRID but not a numeric FONT.

You cannot specify FONT at the same level as the CDEFNT and FNTCHRSET keywords.

You can specify this keyword only once for each record and once per field.

This keyword is valid for data types A, S, and F.

Option indicators are valid for this keyword.

Figure 4-50 shows how to specify the FONT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A  02 03                                  FONT(222)
00030A            FLD1          6A    16 01
00040A  01                                     FONT(ADMMVSS)
00050A            FLD2          6S    20 01
00060A          R RECORD2
00070A  04                                     FONT(16951 (*POINTSIZE 10))
00080A            FLD3          6A    16 01
00090A  05                                     FONT(16951 (*POINTSIZE 12))
00100A            FLD4          6S    20 01FONT(4919)
     A
```

Figure 4-50. Specifying the FONT Keyword

FLD1 uses the multinational vector symbol set (FONT(ADMMVSS)) if indicator 01 is on, or Gothic 15 (FONT(222)) if indicator 01 is off and indicators 02 and 03 are on. Otherwise, the font specified on the CRTPRTF command is used.

FLD2 uses Gothic 15 if indicators 02 and 03 are on. Otherwise, the font specified on the CRTPRTF command is used.

FLD3 uses Century Schoolbook** with a point size of 12 (FONT(16951 (*POINTSIZE 12))) if indicator 05 is on, or Century Schoolbook with a point size of 10 if indicator 05 is off and indicator 04 is on. Otherwise, the font specified on the CRTPRTF command is used.

FLD4 uses Goudy old style (FONT(4919)).

GDF (Graphic Data File)

Use this record-level keyword to print a graphic data file.

The format of the keyword is:

```
GDF([library-name/]graph-file graph-member position-down position-across
graph-depth graph-width graph-rotation)
```

The graph-file and graph-member parameters identify the chart to be printed. Both are required parameters.

Use the optional library-name parameter to further qualify the graphic data file and member. If you do not specify the library-name parameter, *LIBL is used to search for the graphic data file at print time.

You can specify the library-name, graph-file, and graph-member parameters as constants, program-to-system fields, or a combination of both, as shown in the following:

- [library-name/]graph-file graph-member
- [library-name/]&field1 graph-member
- [&field2/]graph-file &field3
- [&field4/]&field5 &field6

Field1, field2, field3, field4, field5, and field6 are the names of program-to-system fields. The fields must exist in the same record format as the GDF keyword and be defined as length 10, data type A (character), and usage P (program-to-system).

The position-down parameter is required and defines the vertical starting point of the chart relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The position-across parameter is required and defines the horizontal starting point of the chart relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The graph-depth parameter is required and defines the depth of the chart. The chart is scaled to fit within the area specified by the graph-depth parameter. Valid values are 0.001 to 57.790 cm (0.001 to 22.750 in.).

The graph-width parameter is required and defines the width of the chart. The chart is scaled to fit within the area specified by the graph-width parameter. Valid values are 0.001 to 57.790 cm (0.001 to 22.750 in.).

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the position-down, position-across, graph-depth, and graph-width parameter values. If the value specified for a parameter is outside the valid range, it is flagged when the spooled file is created.

The graph-rotation parameter is required and defines the orientation of the chart with respect to the text on the page. Valid values are 0, 90, 180, and 270.

An error message is issued at print time if the chart is not positioned on the page.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when GDF is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

When GDF is specified on a record format, all fields within the record format must be positioned using the POSITION keyword. See "POSITION (Position)" on page 4-74 for more information.

An error message is issued if a constant field is specified in a record format where the GDF keyword is also specified.

You can specify this keyword multiple times on a record.

You cannot specify GDF with the following keywords:

SPACEA
SPACEB
SKIPA
SKIPB

Option indicators are valid for this keyword.

Figure 4-51 shows how to specify the GDF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1                    GDF(GRAPHLIB/GFILE MYGRAPH 1.557 +
     A                                    2.831 7.0 4.5 90)
     A*
     A          R REC2                    GDF(&GLIB/&GFILE &GRAF 1.3 5.1 +
     A                                    5.25 6.75 180)
     A            GLIB          10A  P
     A            GFILE         10A  P
     A            GRAF          10A  P
     A*
     A          R REC3                    GDF(GFILE MYGRAF 2.0 7.0 4.5 11.25 +
     A                                    180)
     A*
     A                                    GDF(GFILE YOURGRAF 0.1 0.5 3.67 +
     A                                    6.5 90)
     A*
     A          R REC4
     A  01                                GDF(YOURFILE THATGRAF 2.5 7.3 3.0 +
     A                                    5.25 0)
     A*
```

Figure 4-51. Specifying the GDF Keyword

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the parameter values.

REC1 prints member MYGRAPH from file GFILE in library GRAPHLIB. The chart prints 1.557 units down and 2.831 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The chart is 7.0 units deep, 4.5 units wide, and is rotated 90 degrees.

REC2 allows the application program to specify the library, file, and graph names by setting the fields GLIB, GFILE, and GRAF, respectively. The chart prints 1.3 units down and 5.1 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The chart is 5.25 units deep, 6.75 units wide, and is rotated 180 degrees.

REC3 prints two charts. MYGRAF prints 2.0 units down and 7.0 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The chart is 4.5 units deep, 11.25 units wide, and is rotated 180 degrees. YOURGRAF prints 0.1 units down and 0.5 units across from the margins specified on the CRTPRTF command. The chart is 3.67 units deep, 6.5 units wide, and is rotated 90 degrees. Both charts are located using *LIBL and file GFILE.

REC4 prints THATGRAF only if indicator 01 is on.

HIGHLIGHT (Highlight)

Use this record- or field-level keyword to indicate that a field should be printed in bold letters.

This keyword has no parameters.

This keyword is valid for both IPDS and SCS printers.

For files created with DEVTYPE(*AFPDS), this keyword applies only to registered font IDs. If HIGHLIGHT is used with a coded font or character set and code page, a message is issued.

If you specify HIGHLIGHT at the record level, the keyword applies to all fields in that record. Thus, if both the record- and field-level HIGHLIGHT keywords are optioned and either indicator condition is met, the HIGHLIGHT keyword is used.

The HIGHLIGHT keyword may not apply during printing because of the font being used. Do not use HIGHLIGHT if a numeric font is specified that does not support the highlight font or if a graphics font is specified.

The HIGHLIGHT keyword is valid on either named or constant fields.

This keyword is valid for data types A, S, and F. You can specify HIGHLIGHT only once for each record and once for each field.

Option indicators are valid for this keyword.

Figure 4-52 on page 4-62 shows how to specify the HIGHLIGHT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1
     A  01                                    HIGHLIGHT
     A                                     4 01'HIGHLIGHT IF 01'
     A            FLD1           3A    11 01TEXT('HIGHTLIGHT IF 02N90  +
     A                                        OR 01')
     A  02N90                                 HIGHLIGHT
     A
```

Figure 4-52. Specifying the HIGHLIGHT Keyword

INDARA (Indicator Area)

Use this file-level keyword to remove option indicators from the buffer (also called the record area) and place them in a 99-byte separate indicator area.

This keyword has no parameters.

If you specify the INDARA keyword, some high-level languages require that you specify in your program that a separate indicator area is to be used. See the appropriate high-level language manual.

If you originally specified the INDARA keyword on a file, you can add, change, or delete option indicators in the DDS and re-create the file without having to re-create the high-level language program. You can do this because the field locations in the buffer have not changed and, therefore, the level check data has not changed. If the program is to take advantage of new indicators, however, change and re-create the program.

Option indicators are not valid for this keyword.

Figure 4-53 shows how to specify the INDARA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                   INDARA
00020A          R RCD
00030A  41                               SPACEB(1)
00040A            ACTNBR        10       2
     A
```

Figure 4-53. Specifying the INDARA Keyword

If you specify the INDARA keyword, option indicator 41 is removed from the buffer for record format RCD and placed in the separate indicator area. Only ACTNBR, a named field, remains in the buffer for RCD.

INDTXT (Indicator Text)

Use this file-, record-, or field-level keyword to associate descriptive text (indicating intent or use) with a specific indicator. You can specify INDTXT once for each indicator.

The format of the keyword is:

```
INDTXT(indicator 'indicator-text')
```

If you specify the INDTXT keyword, indicator-text is a required parameter value. Indicator use text must be a character constant and must be enclosed in apostro-

phes. If the length of the text is greater than 50 characters, the high-level language compiler only uses the first 50 characters.

Option indicators are not valid for this keyword.

Note: This specification by itself does not cause the specified indicator to appear in the output record area. The specification merely provides text to be associated with the indicator. If you do not specify the indicator elsewhere, the text is lost without a diagnostic. Also, once an indicator is given a textual assignment (either by this keyword or the response indicator text), no other textual assignment is made.

Figure 4-54 shows how to specify the INDTXT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                        INDTXT(02 'Alternate month')
00020A          R MASTER
00030A            MTH                   2 10
00040A  02        ALTMTH                2 10
00050A
     A
```

Figure 4-54. Specifying the INDTXT Keyword

The INDTXT keyword describes the use of option indicator 02. In a compiler listing for a high-level language, 'Alternate month' is printed as a comment with the description of indicator 02.

LINE (Line)

Use this record-level keyword to print a horizontal or vertical line.

The format of the keyword is:

```
LINE(position-down position-across line-length
line-direction line-width [line-pad])
```

The position-down parameter is required and defines the vertical starting point of the line relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The position-across parameter is required and defines the horizontal starting point of the line relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

You can specify the position-down and position-across parameters as constants, program-to-system fields, or a combination of both, as shown in the following:

- LINE(0.5 7.1 ...)
- LINE(&field1 1.3 ...)
- LINE(2.75 &field2 ...)
- LINE(&field3 &field4 ...)

Field1, field2, field3, and field4 are the names of program-to-system fields. The fields must exist in the same record format as the LINE keyword and be defined as having length 5 with 3 decimal positions, data type S (zoned decimal), and usage P (program-to-system).

The line-length parameter is required and defines the length of the line. Valid values are 0.001 to 57.790 cm (0.001 to 22.750 in.).

The line-width parameter is required and defines the width of the line. Valid values are 0.001 to 57.790 cm (0.001 to 22.750 in.). The following special values can also be specified:

Value	Line Width
*NARROW	12/1440 in. (0.008 in., 0.022 cm)
*MEDIUM	24/1440 in. (0.017 in., 0.042 cm)
*WIDE	36/1440 in. (0.025 in., 0.064 cm)

Notes:

1. The UOM parameter on the CRTPRTF command determines the units of measure for the position-down, position-across, line-length, and line-width parameter values. If the value specified for a parameter is outside the valid range, it is flagged when the spooled file is created.
2. Depending on printer hardware, lines smaller than approximately 0.004 in. (0.010 cm) might not print because of printer resolution. No message is issued when this occurs.

The line-direction parameter is required and can have a value of horizontal (*HRZ) or vertical (*VRT).

The line-pad parameter is optional. It specifies where the line-width value is placed in relationship to the actual line coordinates. For example, if line-width is 5 and line-pad is *TOP, the line extends above the point identified by the position-across and position-down parameters. Valid values are *TOP and *BOT for *HRZ lines, and *LEFT and *RIGHT for *VRT lines. The defaults are *BOT for horizontal lines and *RIGHT for vertical lines.

When the LINE keyword is specified on a record format, all fields within the record format must be positioned using the POSITION keyword. See "POSITION (Position)" on page 4-74 for more information.

An error message is issued if a constant field is specified in a record format where the LINE keyword is also specified.

An error message is issued at print time if the line does not fit on the page.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when LINE is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

You can specify this keyword multiple times on a record.

You cannot specify LINE with the following keywords:

SPACEA
SPACEB
SKIPA
SKIPB

Option indicators are valid for this keyword.

Figure 4-55 on page 4-65 shows how to specify the LINE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1                      LINE(1.5 3.0 4.25 *HRZ 0.2 *TOP)
     A*
     A          R REC2                      LINE(2.1 1.5 7.5 *HRZ 0.05 *BOT)
     A                                      LINE(&FLD1 &FLD2 4.25 *VRT 0.01 +
     A                                      *LEFT)
     A            FLD1          5S 3P
     A            FLD2          5S 3P
     A*
     A          R REC3
     A  02                                  LINE(1.0 1.1 6 *HRZ 1)
     A*
     A
     A
```

Figure 4-55. Specifying the LINE Keyword

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the parameter values.

REC1 prints a horizontal line 4.25 units long. The line starts 1.5 units down and 3.0 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The line is 0.2 units wide. The extra width is added at the top of the line.

REC2 prints two lines. The first line, printed horizontally, starts 2.1 units down and 1.5 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. The line is 7.5 units long and 0.05 units wide. The extra width is added below the line.

The position of the second line is determined by the value assigned to program-to-system fields FLD1 and FLD2. The line, printed vertically, is 4.25 units long and 0.01 units wide. The extra width is added on the left side of the line.

REC3 prints a line only if indicator 02 is on. The extra width is added on the bottom of the line.

LPI (Lines Per Inch)

Use this record-level keyword to change lines per inch within a file. If you do not specify LPI for a record, the LPI value is set from the LPI value on the CRTPRTF, CHGPRTF, or OVRPRTF command.

The format of the keyword is:

```
LPI( 4 | 6 | 8 | 9 | 12)
```

4, 6, 8, 9, and 12 are the valid parameter values.

When you use multiple LPI per page, all skip-to line numbers (on SKIPB, SKIPA) become absolute positions (fixed locations on the paper). For example, if the page length is 66 lines and the file LPI value is 6, then the forms are 11.0 inches long. If you indicate that it should skip to line number 48 it skips down 8 inches on the page and prints. If, in this example, you print 24 lines at 6 LPI (4 inches) and then print 24 lines at 8 LPI (3 inches), the 48th line is 7 inches down on the page.

In both of these examples, 48 lines are processed. If a SKIPB(55) keyword is used, the first example skips to line 55, based on 6 LPI (55/6 inch down the page). In the second example, a page eject occurs and printing starts on line 55, based on 8 LPI (55/8 inch down the page). A page eject occurs in the second example because we printed down 7 inches on the page. A skip to line 55, based on 8 LPI, is less than 7 inches. Therefore, to print on line 55, the current page must be ejected.

Data is processed in a sequential fashion based on location, not line number. If you use a skip-to to go to a line number that is a location above the current location (even though the line number is greater than the current line number), a page eject occurs. Printing continues on the next page.

When using multiple LPI per page, all spacing (SPACEA and SPACEB) is done relative to the current position. For example, if you print 24 lines at 6 LPI (4 inches), and then print 24 lines at 8 LPI (3 inches), the 48th line is 7 inches down on the page. If you then do a SPACEA(4) (LPI is still 8 LPI), you will space down 1/2 inch from the last line and be a total of 7.5 inches down on the page. It is recommended that you use SPACEA and SPACEB keywords when you use the LPI keyword.

The LPI, SKIP, and SPACE keywords are processed in the following order:

LPI
SKIPB
SPACEB
SPACEA
SKIPA

Thus, the SPACE and SKIP keywords use the new LPI value. The LPI for the next and the following lines print at the LPI value specified on the LPI parameter. This parameter value remains in effect until the next record format processes. At the end of the record format, the LPI value changes back to the file level.

The LPI takes effect only on a line boundary. If you change the LPI within a line, the new value takes effect at the end of the line and no diagnostic appears.

You cannot specify this keyword on the same record format with BLKFOLD, CPI, or DFNCHR. If you use any of these keywords with LPI, the file is not created.

This keyword is valid for IPDS printers and printers capable of Advanced Function Printing support. A warning message is issued when you specify the LPI keyword in a file created with DEVTYPE(*SCS).

The overflow line number (OVRFLW parameter on the CRTPRTF command) is not converted to an absolute position (in inches) based on the file level LPI value. The overflow condition signals when the overflow position (in inches) is reached.

Example: Page length = 66, LPI = 6, OVRFLW = 60 (10 inches).

- Print 36 lines at 6 LPI (6 inches).
- Print 16 lines at 4 LPI (4 inches).

After line 52 processes, the overflow condition is signaled.

Option indicators are not valid for this keyword.

Figure 4-56 shows how to specify the LPI keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                  LPI(6)
     A                                     SPACEB(6)
     A
```

Figure 4-56. Specifying the LPI Keyword

Regardless of the LPI you specify on the CRTPRTF, CHGPRTF, or OVRPRTF command, the printer device spaces down one inch before it prints the next line.

MSGCON (Message Constant)

Use this field-level keyword to indicate that the text for a constant field is contained in a message description. If the message description does not exist at DDS compile time, the file is not created. If you change the message description, you must create the file again.

The format of the keyword is:

```
MSGCON(length message-ID [library-name/]message-file-name)
```

The length parameter specifies the maximum length of the message description. The length can be from 1 to 132 bytes. If the message description is less than the length specified, the remaining bytes are padded with blanks (hex 40). If the message description is longer than the length specified, the message description is truncated to the specified length and a warning message appears.

The message-ID parameter specifies the message description that contains the text to use as the value of the constant field.

The message-file-name parameter identifies the message file that contains the message description. The library-name parameter is optional.

You must explicitly specify the MSGCON keyword for the field. You cannot use the MSGCON keyword to initialize a named field.

The DFT and MSGCON keywords are functionally equivalent. If you specify the DFT and MSGCON keywords for the same field, the MSGCON keyword is ignored and the file is not created.

You cannot specify the DATE, DFT, EDTCDE, EDTWRD, and TIME keywords with the MSGCON keyword.

Option indicators are not valid for this keyword. However, they can be used to condition the field with which this keyword is specified.

Figure 4-57 shows how to specify the MSGCON keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD1
00020A                                2  1MSGCON(10 MSG0001 MESSAGES/MSGF)
     A
```

Figure 4-57. Specifying the MSGCON Keyword

MSG0001 in message file MSGF in library MESSAGES contains the message text.

OVERLAY (Overlay)

Use this record-level keyword to print an overlay.

The format of the keyword is:

```
OVERLAY([library-name/]overlay-name position-down position-across)
```

The overlay-name, position-down, and position-across parameters are required.

Use the optional library-name parameter to further qualify the overlay. If you do not specify the library-name parameter, *LIBL is used to search for the overlay at print time.

Note: If an application uses private resources (for example, fonts, page segments, overlays, or GDF files not distributed with the system), be aware of the following. When referencing these resources, if you specify *LIBL or you do not specify a library name, the resources must be available through the library list used by the application creating the spooled file.

You can specify the library-name and overlay-name parameters as constants, program-to-system fields, or a combination of both, as shown in the following:

- [library-name/]overlay-name
- [library-name/]&field1
- [&field2/]overlay-name
- [&field3/]&field4

Field1, field2, field3, and field4 are the names of program-to-system fields. The fields must exist in the same record format as the OVERLAY keyword and be defined as data type A (character) and usage P (program-to-system). Field1 and field4 must have length 8; field2 and field3 have length 10.

The position-down parameter defines the vertical starting point of the overlay relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The position-across parameter defines the horizontal starting point of the overlay relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the position-down and position-across parameter values. If the value specified for a parameter is outside the valid range, it is flagged when the spooled file is created.

An error message is issued at print time if the overlay does not fit on the page.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when OVERLAY is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

When the OVERLAY keyword is specified on a record format, all fields within the record format must be positioned using the POSITION keyword. See "POSITION (Position)" on page 4-74 for more information.

An error message is issued if a constant field is specified in a record format where the OVERLAY keyword is also specified.

You can specify this keyword multiple times on a record.

A maximum of 10 overlays can be used on a single page.

Overlays are not automatically rotated when using the PAGRTT keyword or the PAGRTT parameter on the printer file. See the *Guide to Programming for Printing* for information on overlays and rotation.

You cannot specify OVERLAY with the following keywords:

SPACEA
SPACEB
SKIPA
SKIPB

Option indicators are valid for this keyword.

Figure 4-58 shows how to specify the OVERLAY keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1                        OVERLAY(MYLIB/OVL04 1.234 14.62)
     A*
     A          R REC2                        OVERLAY(&LIB/&OVLS 0.15 1.92)
     A            LIB           10A  P
     A            OVLS           8A  P
     A*
     A          R REC3                        OVERLAY(MYOVL 11.219 0.2)
     A                                        OVERLAY(YOUROVL 7.3 9.27)
     A*
     A          R REC4
     A  01                                    OVERLAY(MYLOGO 0.0 3.01)
     A
```

Figure 4-58. Specifying the OVERLAY Keyword

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the parameter values.

REC1 prints overlay OVL04 found in library MYLIB. The overlay prints 1.234 units down and 14.62 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command.

REC2 allows the application program to specify the library and overlay name by setting program variables LIB and OVLS, respectively. The overlay prints 0.15 units down and 1.92 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command.

REC3 prints two overlays. MYOVL prints 11.219 units down and 0.2 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. YOUROVL prints 7.3 units down and 9.27 units across from the margins specified on the CRTPRTF command. Both overlays are located using *LIBL.

REC4 prints MYLOGO only if indicator 01 is on.

PAGNBR (Page Number)

Use this field-level keyword to specify the location of an unnamed, 4-digit, zoned decimal field to contain the page number. Specify only the PAGNBR keyword, the location of the field (the location of the field can be either position only, or line number and position), and, optionally, the CHRSIZ, COLOR, FONT, HIGHLIGHT, UNDERLINE, or TEXT keyword.

This keyword has no parameters.

When the printer file is opened, the OS/400 program sets the page count to zero and increases it by one before it prints each new page. This is done even if you do not specify the PAGNBR keyword. The page number is printed each time any field for which the PAGNBR keyword is specified is printed. The page number does not increase beyond 9999; it stays 9999 until it is reset. To reset the page count, condition PAGNBR with option indicators. The OS/400 program resets the page number when your program selects PAGNBR (see "Specifying on One Line with No Indicators").

You can also specify EDTCDE or EDTWRD with the PAGNBR keyword.

Option indicators are valid for this keyword.

You can specify the field (location only) and the keyword on one line with no indicators or on separate lines with separate indicators. The following sections explain the differences.

Specifying on One Line with No Indicators: The page number is always printed, and you cannot reset the page number to one, as shown in Figure 4-59.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A          R RECORD
00110A                                    1 60'PAGE:'
00120A                                      +1PAGNBR
     A
```

Note: On lines 110 and 120 two constant fields are specified:
'PAGE:' and the page number itself (location specified as +1).

Figure 4-59. Specifying PAGNBR on One Line with No Indicators

Specifying on Separate Lines with Separate Indicators: If the field indicator (01 in the following example) is off, the field is not printed, even if the keyword indicator (02 in the following example) is set on. If the field indicator is on, the field is printed. The page count is increased when the keyword indicator is off. The page count is reset to one when the keyword indicator is on. The page number prints whether the keyword indicator is off or on.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A          R RECORD
00110A  01                                1 60'PAGE'
00120A  01                                  +1
00130A  02                                    PAGNBR
     A
```

Figure 4-60. Specifying PAGNBR on Separate Lines with Separate Indicators

PAGRTT (Page Rotation)

Use this record-level keyword to specify the degree of rotation of the text with respect to the way the page is loaded into the printer. The PAGRTT keyword is valid only for the 3812, 3816, 3820, 3825, 3827, 3835, and 4028 printers. If you do not specify a PAGRTT keyword for a record, the page rotation is set from the value specified on the Create Printer File (CRTPRTF), Change Printer File (CHGPRTF), or Override Printer File (OVRPRTF) commands.

The format of the keyword is:

```
PAGRTT(0 | 90 | 180 | 270)
```

The valid parameter values for the keyword are 0, 90, 180, and 270. Zero indicates no rotation. The other values specify the number of degrees rotation clockwise from the 0 degree rotation column.

Note: For the 3835 Printer using landscape paper, a counterclockwise rotation is used for a DDS file.

The PAGRTT keyword does not cause an implicit page eject. If the paper is not on a page boundary, the keyword is not used, and a diagnostic message is issued.

The PAGRTT, SKIP, and SPACE keywords are processed in the following order:

SKIPB
SPACEB
PAGRTT
SPACEA
SKIPA

The PAGRTT keyword remains in effect for the duration of the record format. If a PAGRTT keyword is not used on the next record format, it reverts back to the PAGRTT value specified at the command level.

Notes:

1. The PAGRTT keyword remains in effect for the entire page. At the end of a record format that specifies PAGRTT, the file is not changed back to the file level rotation until the next page is processed. For example, the file does not return to rotation 0 until record E is written if the file level parameter specifies PAGRTT(0) and you write one of the following:
   - Record format A (PAGRTT (90))
   - Record format B (same page as record A)
   - Record format C (same page as record A)
   - Record format D (same page as record A)
   - Record format E (next page)
2. For files created with DEVTYPE(*SCS), if the PAGRTT keyword is specified on a record format that spans several pages, it remains in effect only for the page on which it is specified.

When a page rotates, the page length and page width are exchanged so that the length becomes the width and the width becomes the length. This exchange cannot be done under certain conditions, including:

- PAGRTT on the CRTPRTF, CHGPRTF, or OVRPRTF command is *DEVD or *COR.
- The font on the CRT/CHG/OVRPRTF command is *DEVD.

When the length and width cannot be exchanged, a diagnostic message is issued.

Folding and truncation support is not performed when pages rotate. That is, the BLKFOLD DDS keyword and the FOLD parameter on the CRTPRTF, CHGPRTF, or OVRPRTF commands are not used.

Option indicators are valid for this keyword.

Figure 4-61 shows how to specify the PAGRTT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1
     A  02                                  SKIPB(1)
     A  02                                  PAGRTT(270)
     A            FIELD1         3A     2 01
     A          R RECORD2
     A            FIELD1         3A     6 01
     A          R RECORD3
     A            FIELD1         3A     5 01
     A
```

Figure 4-61. Specifying the PAGRTT Keyword

If indicator 02 is on, RECORD1 is rotated 270 degrees. SKIPB(1) is specified so that the record starts on a new page.

When RECORD2 is printed, it will also be rotated 270 degrees since it is on the same page as RECORD1.

RECORD3 uses the PAGRTT value specified on the CRTPRTF, CHGPRTF, or OVRPRTF commands, since it is on a new page.

PAGSEG (Page Segment)

Use this record-level keyword to print a page segment.

The format of the keyword is:

```
PAGSEG([library-name/]page-segment-name position-down position-across)
```

The page-segment-name, position-down, and position-across parameters are required.

Use the optional library-name parameter to further qualify the page segment. If you do not specify the library name, *LIBL is used to search for the page segment at print time.

Note: If an application uses private resources (for example, fonts, page segments, overlays, or GDF files not distributed with the system), be aware of the following. When referencing these resources, if you specify *LIBL or you do not specify a library name, the resources must be available through the library list used by the application creating the spooled file.

You can specify the library-name and page-segment-name parameters as constants, program-to-system fields, or a combination of both, as shown in the following:

- [library-name/]page-segment-name
- [library-name/]&field1
- [&field2/]page-segment-name
- [&field3/]&field4

Field1, field2, field3, and field4 are names of program-to-system fields. The fields must exist in the same record format as the PAGSEG keyword and be defined as data type A (character) and usage P (program-to-system). Field1 and field4 must have length 8; field2 and field3 have length 10.

The position-down parameter defines the vertical starting point of the page segment relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The position-across parameter defines the horizontal starting point of the page segment relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the position-down and position-across parameter values. If the value specified for a parameter is outside the valid range, it is flagged when the spooled file is created.

An error message is issued at print time if the page segment does not fit on the page.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when PAGSEG is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

When PAGSEG is specified on a record format, all fields within the record format must be positioned using the POSITION keyword. See "POSITION (Position)" on page 4-74 for more information.

An error message is issued if a constant field is specified in a record format where the PAGSEG keyword is also specified.

You can specify the PAGSEG keyword multiple times on a record.

A maximum of 10 page segments can be used per page.

Page segments are not automatically rotated when using the PAGRTT keyword or the PAGRTT parameter on the printer file. See the *Guide to Programming for Printing* for information on page segments and rotation.

You cannot specify PAGSEG at the same level as the following keywords:

SPACEA
SPACEB
SKIPA
SKIPB

Option indicators are valid for this keyword.

Figure 4-62 shows how to specify the PAGSEG keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1                      PAGSEG(MYLIB/PAGSEG5 3.527 4.162)
     A*
     A          R REC2                      PAGSEG(&LIB/&PSEG 8.5 6.72)
     A            LIB           10A  P
     A            PSEG           8A  P
     A*
     A          R REC3                      PAGSEG(MYSEG 0.0 3.759)
     A                                      PAGSEG(YOURSEG 0.0 5.233)
     A*
     A          R REC4
     A  01                                  PAGSEG(MYSEG 0.0 3.01)
     A*
     A
```

Figure 4-62. Specifying the PAGSEG Keyword

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the parameter values.

REC1 prints page segment PAGSEG5 found in library MYLIB. The page segment prints 3.527 units down and 4.162 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command.

REC2 allows the application program to specify the library and page segment name by setting fields LIB and PSEG, respectively. The page segment prints 8.5 units down and 6.72 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command.

REC3 prints two page segments. MYSEG prints 0 units down and 3.759 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. YOURSEG prints 0 units down and 5.233 units across from the margins specified on the CRTPRTF command. Both page segments are located using *LIBL.

REC4 prints MYSEG only if indicator 01 is on.

POSITION (Position)

Use this field-level keyword to define the location of a named field on the page.

The format of the keyword is:

```
POSITION(position-down position-across)
```

The position-down parameter is required and defines the vertical starting point of the field relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

The position-across parameter is required and defines the horizontal starting point of the field relative to the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command. Valid values are 0 to 57.790 cm (0 to 22.750 in.).

You can specify the position-down and position-across parameters as constants, program-to-system fields, or a combination of both, as shown in the following:

- POSITION(3.56 6.24)
- POSITION(&field1 9.625)
- POSITION(0.5 &field2)
- POSITION(&field3 &field4)

Field1, field2, field3, and field4 are the names of program-to-system fields. The fields must exist in the same record format as the POSITION keyword and be defined as having length 5 with 3 decimal positions, data type S (zoned decimal), and usage P (program-to-system).

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the position-down and position-across parameter values. If the value specified for a parameter is outside the valid range, it is flagged when the spooled file is created.

An error message is issued at print time if the field does not fit on the page.

An error message is issued at create time if line and position values, columns 39 through 44, are also specified.

If the POSITION keyword is specified for a field, all fields in the record format must also have the POSITION keyword specified. Location entries in positions 39 through 44 are not allowed.

An error message is issued if a constant field is specified in a record format where the POSITION keyword is also specified.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when POSITION is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

You cannot specify POSITION with the following keywords:

SPACEA
SPACEB
SKIPA
SKIPB

Option indicators are valid for this keyword.

Figure 4-63 shows how to specify the POSITION keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1
     A            FLD1           6S 2       POSITION(2.0 1.983)
     A*
     A            FLD2          42A         POSITION(&FLD2A &FLD2B)
     A            FLD2A          5S 3P
     A            FLD2B          5S 3P
     A*
```

Figure 4-63. Specifying the POSITION Keyword

Note: The UOM parameter on the CRTPRTF command determines the units of measure for the parameter values.

In REC1, FLD1 prints 2.0 units down and 1.983 units across from the margins specified on the FRONTMGN or BACKMGN parameter on the CRTPRTF command.

The application program determines the position of FLD2 by assigning values to program-to-system variables FLD2A and FLD2B.

PRTQLTY (Print Quality)

Use this record- or field-level keyword to vary the print quality within the file.

The format of the keyword is:

```
PRTQLTY(print-quality)
```

The parameter is required and must be one of the following special values:

*STD (Standard quality)

*DRAFT (Draft quality)

*NLQ (Near letter quality)

*FASTDRAFT (Fast draft quality)

The PRTQLTY keyword is allowed only on records or fields for which a CHRSIZ or BARCODE keyword applies.

If you do not specify this keyword, the print quality is set by the PRTQLTY parameter on the CRTPRTF, CHGPRTF, and OVRPRTF commands.

If you specify PRTQLTY at the record level, it applies to all fields in that record that do not have PRTQLTY specified at the field level.

PRTQLTY is valid for IPDS printers only. If you specify this keyword in a file created with DEVTYPE(*SCS), a warning message is issued at file creation time.

If you specify this keyword in a file created with DEVTYPE(*AFPDS), the print quality can only change on a page boundary. If PRTQLTY is received before any data is placed on the page, the quality of the page changes. Otherwise, the keyword is ignored and a diagnostic message is sent to the application program.

If you use PRTQLTY in the same record format with BLKFOLD, CPI, or DFNCHR keyword, the file is not created.

Option indicators are allowed for this keyword.

Figure 4-64 shows how to specify the PRTQLTY keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD
     A            FIELD3        10S  0  4 65BARCODE(UPCE 6)
     A                                       PRTQLTY(*DRAFT)
     A
```

Figure 4-64. Specifying the PRTQLTY Keyword

The UPCE bar code in FIELD3 will be printed with draft quality.

REF (Reference)

Use this file-level keyword to specify the name of a file from which field descriptions are to be retrieved.

The format of the keyword is:

```
REF([library-name/]data-base-file-name [record-format-name])
```

Use REF when you want to duplicate descriptive information from one or more fields in a previously defined record format. You can code the file name once on the REF keyword rather than on the REFFLD keyword with each of the field descriptions that reference the file.

If there is more than one record format in the referenced file, specify a record format name as a parameter value for this keyword to tell the OS/400 program which to use, unless the formats should be searched sequentially.

The database-file-name is required for this keyword. The record-format-name and the library-name are optional.

If you do not specify the library-name, the current library list at file creation time is used. If you do not specify the record-format-name, each format is searched in order (as they are specified). The first occurrence of the field name is used. For more information, see Appendix A, "When to Specify REF and REFFLD."

You can specify a Distributed Data Management (DDM) file on this keyword. When using a DDM file, the data-base-file-name and library-name are the DDM file and library names on the source system. The record-format-name is the record format name in the remote file on the target system.

Option indicators are not valid for this keyword.

Figure 4-65 and Figure 4-66 show how to specify the REF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     REF(FILE1)
00020A        R RECORD
00030A          FLD1        R           2  2
     A
```

Figure 4-65. Specifying the REF Keyword (Example 1)

FLD1 has the same attributes as the first (or only) FLD1 in FILE1.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     REF(LIB1/FILE1 RECORD2)
00020A        R RECORD
00030A          FLD1        R           2  2
     A
```

Figure 4-66. Specifying the REF Keyword (Example 2)

FLD1 has the same attributes as FLD1 in RECORD2 in FILE1 in LIB1.

REFFLD (Referenced Field)

Use this field-level keyword when referring to a field under one of these three conditions:

- The name of the referenced field is different from the name in positions 19 through 28.
- The name of the referenced field is the same as the name in positions 19 through 28, but the record format, file, or library of the referenced field is different from that specified with the REF keyword.
- The referenced field occurs in the same DDS source file as the referencing field.

The format of the keyword is:

```
REFFLD([record-format-name/]referenced-field-name
[{*SRC | [library-name/]data-base-file-name}])
```

The referenced-field-name is required even if it is the same as the referencing field. Use the record-format-name when the referenced file contains more than one record format. Use *SRC (rather than the database-file-name) when the referenced field name is in the same DDS source file as the referencing field. *SRC is the default value when the database-file-name and the library-name are not specified.

Note: When you refer to a field in the same DDS source file, the field being referred to must precede the field being defined.

Specify the database-file-name (qualified by its library-name, if necessary) when you want to search a particular database file.

If, in the same DDS source file, you specify REF at the file level and REFFLD at the field level, the particular search sequence depends on both the REF and REFFLD keywords. For more information, see Appendix A, "When to Specify REF and REFFLD."

You must specify an R in position 29. In some cases, some keywords specified with the field in the database file are not included in the printer file. For more information see, "Reference (Position 29)" in this chapter.

You can specify a Distributed Data Management (DDM) file on this keyword.

When using a DDM file, the data-base-file-name and library-name are the DDM file and library names on the source system. The referenced-field-name and the record-format-name are the field name and the record format name in the remote file on the target system.

Option indicators are not valid for this keyword.

Figure 4-67 on page 4-79 shows how to specify the REFFLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R FMAT1
00020A            ITEM           5         1
00030A            ITEM1      R              2REFFLD(ITEM)
00040A            ITEM2      R             12REFFLD(FMAT1/ITEM)
00050A            ITEM3      R             22REFFLD(ITEM FILEX)
00060A            ITEM4      R             32REFFLD(ITEM LIBY/FILEX)
00070A            ITEM5      R             42REFFLD(FMAT1/ITEM LIBY/FILEX)
00080A            ITEM6      R             52REFFLD(ITEM *SRC)
     A
```

Figure 4-67. Specifying the REFFLD Keyword

Because the REF keyword is not specified, the default for lines 00030 and 00040 is to search the DDS source file in which they are specified. In line 00080, the parameter value *SRC explicitly specifies the source file. See the example in Appendix A, "When to Specify REF and REFFLD," for explanations of the specifications.

SKIPA (Skip After)

Use this file-, record-, or field-level keyword to specify that the printer device is to skip to a specific line number after it prints one or more lines.

The format of the keyword is:

```
SKIPA(skip-after-line-number)
```

The parameter value is required and must be in the range 1 through 255.

If you specify the keyword at the file level, you must option it with one or more indicators. If you specify the keyword at the record or field level, option indicators are optional. The specified skip is performed after each record in the file prints.

If you specify the keyword at the record level, skipping is performed after all the lines associated with the record print and before any file-level SKIPA keywords are applied.

If you specify the keyword at the field level, skipping is performed after the field prints.

Note: If you do not use line numbers and do not specify skip or space keywords, overprinting can result.

You can specify this keyword once at the file level, once at the record level, and once for each field.

This keyword is valid at the file level for all records, but not at the record level or the field level for records that have line numbers specified (positions 39 through 41). (The line number entries are flagged as errors.)

The SKIPA keyword is not valid at either the field level or record level if the record format also has the BOX, ENDPAGE, GDF, LINE, OVERLAY, PAGSEG, or POSITION keywords specified.

This keyword is not allowed at the file level for files defined as DEVTYPE(*AFPDS) on the CRTPRTF command.

Option indicators are valid for this keyword.

Figure 4-68 shows how to specify the SKIPA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00011A           FIELDA       132A        1SKIPA(12)
     A
```

Figure 4-68. Specifying the SKIPA Keyword

SKIPB (Skip Before)

Use this file-, record-, or field-level keyword to specify that the printer device is to skip to a specific line number before it prints the next line(s).

The format of the keyword is:

```
SKIPB(skip-before-line-number)
```

The parameter value is required and must be in the range 1 through 255.

If you specify this keyword at the file level, you must option it with one or more indicators; otherwise, option indicators are optional. The specified skip is performed before each record in the file prints and after any file-level SKIPB operations are applied.

If you specify this keyword at the record level, skipping is performed before any of the lines associated with that record print.

If you specify this keyword at the field level, skipping is performed before the field prints.

You can specify this keyword once at the file level, once at the record level, and once for each field.

This keyword is valid at the file level for all records, but not at the record or field level for records that have line numbers specified (positions 39 through 41). (The line numbers are flagged as errors.)

Note: If you do not use line numbers and do not specify skip or space keywords, overprinting can result.

The SKIPB keyword is not valid at either the field level or record level if the record format also has the BOX, ENDPAGE, GDF, LINE, OVERLAY, PAGSEG, or POSITION keywords specified.

This keyword is not allowed at the file level for files defined as DEVTYPE(*AFPDS) on the CRTPRTF command.

Option indicators are valid for this keyword.

Figure 4-69 shows how to specify the SKIPB keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00022A         R RFMTPR                  SKIPB(5)
     A
```

Figure 4-69. Specifying the SKIPB Keyword

SPACEA (Space After)

Use this record- or field-level keyword to specify that the printer device is to space some number of lines after it prints one or more lines.

The format of the keyword is:

```
SPACEA(space-after-value)
```

The parameter value is required and must be in the range 0 through 255.

If you specify this keyword at the record level, spacing occurs after all lines associated with that record are printed. You can specify this keyword only once at the record level and once for each field.

If you specify SPACEA at the field level, spacing is performed after the field is printed.

This keyword is not valid for records with specified line numbers (positions 39 through 41). (The line numbers are flagged as errors.)

Note: If you do not use line numbers and do not specify space or skip keywords, overprinting can result.

The SPACEA keyword is not valid at either the field level or record level if the record format also has the BOX, ENDPAGE, GDF, LINE, OVERLAY, PAGSEG, or POSITION keywords specified.

Option indicators are valid for this keyword.

Figure 4-70 shows how to specify the SPACEA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A          R RFMTPR                    SPACEA(1)
00101A            FIELDA       132         1
     A
```

Figure 4-70. Specifying the SPACEA Keyword

SPACEB (Space Before)

Use this record- or field-level keyword to specify that the printer device is to space some number of lines before it prints the next line or lines.

The format of the keyword is:

```
SPACEB(space-before-value)
```

The parameter value is required and must be in the range 0 through 255.

If you specify this keyword at the record level, spacing occurs before any lines associated with that record are printed. You can specify this keyword only once at the record level or once for each field.

If you specify SPACEB at the field level, spacing is performed before the line containing that field prints.

This keyword is not valid for records with specified line numbers (positions 39 through 41). (The line numbers are flagged as errors.)

Note: If you do not use line numbers and do not specify space or skip keywords, overprinting can result.

The SPACEB keyword is not valid at either the field level or record level if the record format also has the BOX, ENDPAGE, GDF, LINE, OVERLAY, PAGSEG, or POSITION keywords specified.

Option indicators are valid for this keyword.

Figure 4-71 shows how to specify the SPACEB keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A            FIELDA        25A       55SPACEB(3)
00011A            FIELDB        30       100
     A
```

Figure 4-71. Specifying the SPACEB Keyword

TEXT (Text)

Use this record- or field-level keyword to supply a text description (or comment) for the record format or field that is used for program documentation.

The format of the keyword is:

```
TEXT('description')
```

The text must be enclosed in apostrophes. If the length of the text is greater than 50 positions, only the first 50 characters are used by the high-level language compiler.

Option indicators are not valid for this keyword.

Figure 4-72 shows how to specify the TEXT keyword at the record and field levels.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CUSMST                    TEXT('Customer Master Record'
00020A            FLD1           3  0       TEXT('ORDER NUMBER FIELD')
     A
```

Figure 4-72. Specifying the TEXT Keyword

TIME (Time)

This field-level keyword prints the current system time as a constant field 6 bytes long. You can specify the location of the field, the TIME keyword, and, optionally, the EDTCDE, EDTWRD, COLOR, HIGHLIGHT, CHRSIZ, FONT, UNDERLINE, or TEXT keyword. Positions 17 through 38 must be blank.

This keyword has no parameters.

The edit word '0_:__:__' (_ represents a blank) is assumed for a TIME field. You can specify another edit word or one of the user-defined edit codes (5 through 9) to change the IBM-supplied editing.

Option indicators are not valid for this keyword, although you can use option indicators to condition the field specified.

Figure 4-73 shows how to specify the TIME keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  20                            1 56TIME
00020A  21                            1 56TIME
     A                                   EDTWRD('0 &HRS&  &MINS&  &SECS')
     A
```

Figure 4-73. Specifying the TIME Keyword

If the system time is 110645, the time prints as follows:

- If option indicator 20 is on, the time prints as

  `11:06:45`

- If option indicator 21 is on (and option indicator 20 is off), the time prints as

  `11 HRS 06 MINS 45 SECS`

TRNSPY (Transparency)

This field-level keyword prevents code points you have redefined (using the DFNCHR keyword) from being interpreted as SCS printer control commands when your program sends an output operation that prints the field you are defining.

This keyword has no parameters.

If you do not specify the TRNSPY keyword for a field in which your program passes hexadecimal data to an SCS printer, code points can be interpreted as SCS commands that affect printer operation. A **code point** is one of the 256 values you can assign a character in a character set. On the AS/400 system, a code point is identified by a 2-digit hexadecimal number.

You must specify the TRNSPY keyword when you specify:

- The CVTDTA keyword in a printer file created with DEVTYPE(*SCS)
- A hexadecimal value (with or without the DFT keyword explicitly specified)

In a file created with DEVTYPE(*IPDS), you need not specify the TRNSPY keyword with the CVTDTA keyword. However, a warning message appears stating that the DEVTYPE should not be changed to *SCS.

If you specify TRNSPY in a file created with DEVTYPE(*AFPDS), a warning message appears at create time.

The TRNSPY keyword is valid only when the data type is character.

When you specify the TRNSPY keyword with the CVTDTA keyword, your program can place character data in the field and the OS/400 program converts it to hexadecimal data when the field is passed to the printer. Each pair of hexadecimal digits corresponds to a code point in the character set of your AS/400 system. Using the DFNCHR keyword, you can define characters of your own design for the 5224 Printer or 5225 Printer. See the DFNCHR keyword description. Also, the printed length of the field is one half the length you specify. Therefore, the length of the field must be an even number.

If you specify the TRNSPY keyword without the CVTDTA keyword, the field length you specify is the printed length.

This keyword is supported only for the 5224 Printer and 5225 Printer.

Option indicators are not valid for this keyword.

The following examples show how to specify the TRNSPY keyword.

Figure 4-74 shows how to specify the TRNSPY keyword with the CVTDTA keyword. When your program passes character data in a field, the OS/400 program converts it to hexadecimal data.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD                    SPACEB(1)
00020A            FLD1          10         1TRNSPY CVTDTA
00030A            FLD2          20         6TRNSPY CVTDTA
     A
```

Figure 4-74. Specifying the TRNSPY Keyword (Example 1)

The program can pass character data in FLD1 and FLD2. The OS/400 program converts it to hexadecimal data for the printer. Only the characters 0 through 9 and A through F are valid. Blanks are not valid.

The printed length of FLD1 and FLD2 is one half the specified length (FLD1 is 5 positions long; FLD2 is 10 positions long).

You must also specify the DFNCHR keyword with this DDS in order to print user-defined characters.

The following is how RECORD prints when the contents of FLD1 are 'C1C1C1C1C1' and the contents of FLD2 are 'C2C2C2C2C2C2C2C2C2C2':

```
AAAAABBBBBBBBBB
```

Figure 4-75 shows how to specify the TRNSPY keyword without the CVTDTA keyword. In this example, your program must pass hexadecimal data in the field.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD2                   SPACEB(1)
00020A            FLD3           5         1TRNSPY
00030A            FLD4          10         6TRNSPY
     A
```

Figure 4-75. Specifying the TRNSPY Keyword (Example 2)

The program must pass hexadecimal data in FLD3 and FLD4. Only hexadecimal characters 0 through 9 and A through F are valid. Blanks are not valid. Without the CVTDTA keyword, the printed length of both fields is the specified length.

TXTRTT (Text Rotation)

Use this field-level keyword to rotate any text contained in the field.

The format of the keyword is:

```
TXTRTT(field-rotation)
```

The field-rotation parameter is required and controls the rotation of the field. Valid values are 0, 90, 180, and 270 degrees.

Specify DEVTYPE(*AFPDS) on the CRTPRTF command when TXTRTT is specified in the file. If DEVTYPE is changed to anything other than *AFPDS, the keyword is ignored and a warning message is issued at print time.

Option indicators are valid for this keyword.

Figure 4-76 shows how to specify the TXTRTT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC1
     A            FLD05         10     35 15TXTRTT(90)
     A*
     A          R REC2
     A            FLD06          7S 2       TXTRTT(270)
     A                                      POSITION(6.5 13.8)
     A
```

Figure 4-76. Specifying the TXTRTT Keyword

FLD05 in REC1 is rotated 90 degrees.

FLD06 in REC2 is rotated 270 degrees.

UNDERLINE (Underline)

Use this field-level keyword to specify that the OS/400 program is to underline the field when it is printed. Specify UNDERLINE only if the printer supports underlining.

This keyword has no parameters.

Option indicators are valid for this keyword.

Figure 4-77 shows how to specify the UNDERLINE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A           ALLOC     R          17 11
00101A  03 04                              UNDERLINE
     A
```

Figure 4-77. Specifying the UNDERLINE Keyword

Chapter 5. Intersystem Communications Function Files

This chapter provides the following information regarding intersystem communications function (ICF) files:

- Definition
- Positional entries
- Keyword entries

"Positional Entries (Positions 1 through 44)" gives rules and examples for filling in positions 1 through 44 of the data description specifications (DDS) form. "Keyword Entries (Positions 45 through 80)" gives rules and examples for specifying DDS keywords. The keywords are described in alphabetical order.

For more information about keywords for ICF files, see the *ICF Programmer's Guide*.

Defining an ICF File

Specify the entries in the following order to define an ICF file:

1. File-level entries (optional)
2. Record-level entries
3. Field-level entries (optional)

Repeat the record-level entries and field-level entries for each record format in the file.

Specify at least one record format in the file.

The maximum number of record formats in an ICF file is 1024. The maximum number of fields in any one record format is 32 767.

Note: Specify the file name through the Create Intersystem Communications Function File (CRTICFF) command, not through DDS.

You can find an explanation of file level, record level, and field level as well as the syntax rules for specifying DDS keywords in Chapter 1, "Introduction." You can find a complete ICF example in Appendix B, "Examples."

Figure 5-1 on page 5-2 shows an ICF file coding example.

© Copyright IBM Corp. 1991, 1993

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* ICF FILE CODING EXAMPLE
00020A*
00030A          R RCD1                        RCVENDGRP(14)
00040A            FLDA           5
00050A            FLDB           5  0
00060A            FLDC          10  2
00070A
00080A          R RCD2
00090A  72 73
00110AON74                                    FAIL
00120A            FLDD          12
     A            FLDC      R                REFFLD(A LIB1/FILEA)
     A
```

Figure 5-1. ICF File Coding Example

Positional Entries (Positions 1 through 44)

This section describes how to specify the first 44 positions of the data description specifications (DDS) form for ICF files. To code the remaining part of the form, see "Keyword Entries (Positions 45 through 80)."

Figure 5-1 shows some positional entries for ICF files.

Sequence Number (Positions 1 through 5)

Use these positions to specify a sequence number for each line on the form. The sequence number is optional and is for documentation purposes only.

Form Type (Position 6)

Specify an A in this position to designate this as a DDS form. The form type is optional and is for documentation purposes only.

Comment (Position 7)

Specify an asterisk (*) in this position to identify this line as a comment. Comment lines can appear anywhere in DDS and are kept only in the source file. They appear on the source computer printout but do not appear on the expanded source computer printout.

Use positions 8 through 80 for comment text. A blank line (no characters specified in positions 7 through 80) is treated as a comment.

Conditioning (Positions 7 through 16)

Positions 7 through 16 are a multiple field area in which you can specify option indicators. Option indicators are 2-digit numbers from 01 to 99. Your program can set option indicators on (hex F1) or off (hex F0) to select a keyword for output operations. In ICF files, option indicators are valid only for record- and file-level keywords.

A condition is an ANDed grouping of two through nine indicators that must all be in effect (set off if the letter N is specified; set on if N is not specified) before the keyword is selected. You can specify a maximum of nine indicators for each condition and nine conditions for each keyword. Therefore, a maximum of 81 indicators

can be specified for each keyword, when nine indicators are used with nine conditions. An AND condition occurs when you specify a condition requiring that more than one indicator must be on or off before the condition is satisfied. The first indicator you specify, AND the second, AND the third, and so on, must all be in effect before the condition is satisfied and the keyword is selected. You must specify the keyword on the same line as the last (or only) set of indicators specified.

You can also specify several conditions for a keyword so that if any one of them is satisfied, the keyword is selected. This is called an OR relationship. In an OR relationship, if the first condition is satisfied, OR the second condition, OR the third condition, and so on, the keyword is selected. Note that conditions within the OR relationship can consist of just one indicator or can consist of several indicators ANDed together. Indicators can be ANDed to form a condition. Conditions can be ORed to give your program several ways to select the keyword.

Specify the conditions by entering the following values:

Position 7 (AND) If you need more than three indicators to form an ANDed condition, specify the indicators on the next line or lines. You can specify an A in position 7 on the second or following lines to continue the ANDed condition, or you can leave it blank because A is the default.

Position 7 (OR) If you specify several conditions that are to be ORed together, each condition must start on a new line and each condition, except the first, must have an O in position 7. An O specified for the first condition produces a warning message, and that position is assumed to be blank.

Positions 8, 11, 14 (NOT) If you want an indicator to be off instead of on to satisfy a condition, specify an N in the position just preceding the indicator (position 8, 11, or 14).

Conditioning More Than One Keyword

If you want to condition one or more keywords, the last (or only) indicator must appear on the same line as the keywords. If the conditioning applies to keywords on more than one line, you must use keyword continuation for the indicators to apply to all keywords. See Chapter 1, "Introduction," for more information about DDS syntax rules.

Type of Name or Specification (Position 17)

Enter a value in this position to identify the type of name specified in positions 19 through 28. The valid entries for ICF files are:

Entry	Meaning
R	Record format name
Blank	Field name

Figure 5-1 on page 5-2 shows how to code the name type.

For more information on types of names, see "Name (Positions 19 through 28)" on page 5-4.

Reserved (Position 18)

This position does not apply to any file type. Leave this position blank unless you use it for comment text.

Name (Positions 19 through 28)

Use these positions to specify record format names and field names.

Refer to "Syntax Rules" on page 1-4 for rules to use when specifying record or field names in DDS.

Names must begin in position 19.

Record Format Name

When you specify an R in position 17, the name specified in positions 19 through 28 is a record format name. You can specify more than one record format for an ICF file, but each record format name must be unique within that file.

Field Name

When you specify a blank in position 17, the name specified in positions 19 through 28 is a field name. Field names must be unique within the record format. For ICF files, the order in which field names are specified in the DDS is the order the fields take in the input and output buffers.

The keywords CANCEL, EOS, FAIL, and RQSWRT must have option indicators when they apply to a record with fields. Fields are ignored at run time (not sent across the line) when any of these keywords are in effect. At creation time, if any of these keywords have no option indicator and apply to a record with fields, a severe error is issued and the file will not be created.

Reference (Position 29)

Specify R in this position to use the reference function of the OS/400 program to copy the attributes of a previously defined named field (called the referenced field) to the field you are defining. The referenced field can be previously defined in the ICF file you are defining. The referenced field can also be defined in a previously created database file (the database file to be referenced is specified with the REF or REFFLD keyword). The field attributes referenced are the length, data type, and decimal positions of the field, as well as the ALIAS, FLTPCN, and TEXT keywords.

If you do not specify R, you cannot use the reference function for this field and you must specify field attributes for this field.

Position 29 must be blank at the file and record levels.

The name of the referenced field can be either the same as the field you are defining or different from the field you are defining. If the name of the referenced field is the same as the field you are defining, you need only specify the letter R in position 29 (in addition to specifying the name of the field you are defining in positions 19 through 28). If the name of the field you are defining is different, you must specify the name of the referenced field with the REFFLD (Referenced Field) keyword.

You can specify the name of the file defining the referenced field as a parameter value with the REF or REFFLD keyword. See REF and REFFLD keyword

descriptions later in this chapter and in Appendix A, "When to Specify REF and REFFLD," for explanations of how the OS/400 program finds the referenced field.

You do not need to copy all attributes from the previously described field to the field you are defining. To override specific attributes of the referenced field, specify those attributes for the field you are defining. For example, if you specify a length for the field you are defining, the length is not copied from the referenced field.

When you override the data type to character (by specifying A in position 35), the decimal positions value is not copied from the referenced field.

Note: Once the ICF file is created, you can delete or change the referenced file without affecting the field descriptions in the ICF file. Delete and re-create the ICF file to incorporate changes made in the referenced file.

Length (Positions 30 through 34)

Specify the field length for each field (unless you copy the field's attributes from a referenced field). Specify the number of digits for a numeric field, or the number of characters for a character field. The length specification must be right-justified; leading zeros are optional. Valid length specifications for ICF files are as follows:

Data Type	Valid Length
Character	1 through 32 767
Binary	1 through 9
Zoned decimal	1 through 31
Packed decimal	1 through 31
Floating-point single precision	1 through 9
Floating-point double precision	1 through 17

You can specify a maximum of 9 digits for single precision and 17 digits for double precision. However, the OS/400 program supports a floating-point accuracy of 7 digits for single precision and 15 digits for double precision.

The sum of the number of bytes occupied by all fields in a record must not exceed 32 767 for ICF files. The system determines the number of bytes actually occupied as follows:

Data Type	Bytes Occupied in Storage
Character	Number of characters
Binary	
1-4 digits	2 bytes
5-9 digits	4 bytes
Zoned decimal	Number of digits
Packed decimal	(Number of digits/2) + 1 (truncated if fractional)
Floating-point (single precision)	4 bytes
Floating-point (double precision)	8 bytes

If you are using a referenced field, you can override the length of the field by specifying a new value or by specifying the increase or decrease in length. To increase the length, specify +n where n is the increase. To decrease the length, specify –n, where n is the decrease. For example, an entry of +4 for a numeric field indicates that it is to be 4 digits longer than the referenced field.

Note: High-level languages can impose specific length and value restrictions on the field length. Observe these restrictions for files used by those languages.

Data Type (Position 35)

Use this position to specify the data type of the field within the file. The valid data type entries for ICF files are as follows:

Entry	Meaning
P	Packed decimal
S	Zoned decimal
B	Binary
F	Floating-point
A	Character

If a data type is not specified for the field and is not duplicated from a referenced field, DDS assigns a default depending on the value in the decimal positions (positions 36 and 37). If the decimal positions are blank, a default of character (A) is assigned. If the decimal positions contain a number in the range 0 through 31, a default of zoned decimal (S) is assigned.

Note: Specifying F in position 35 results in a single precision floating-point field. Use the FLTPCN keyword to specify double precision or to change the precision of an already specified floating-point field.

Decimal Positions (Positions 36 and 37)

Use these positions to specify the decimal placement within a packed decimal, a zoned decimal, a floating point, or a binary field. Specify a decimal number from 0 through 31 to indicate the number of decimal positions to the right of the decimal point. (This number must not be greater than the number of digits specified in the field length.)

You can override or change these positions if you are using a referenced field. To override the positions, specify the new value. To change the positions, specify the amount by which you want the field increased or decreased and precede it with either a + or –, respectively. For example, an entry of +4 indicates there are to be four more digits to the right of the decimal point than were in the referenced field. If the resulting number of decimal positions is greater than the maximum allowed, you will receive an error message.

Note: High-level languages can impose specific length and value restrictions on the decimal positions. Observe these restrictions for files used by those languages.

Usage (Position 38)

The valid entries for this position are:

Entry	Meaning
B or blank	Both input/output field
P	Program-to-system field

A program-to-system field is used to communicate between an application program and the sending system (which is local to the application program). It is not sent as part of the data record across the line to the receiving system.

The following rules apply to program-to-system fields:

- The field must be a named, numeric, or alphanumeric output-only field.

- In the record format, the program-to-system fields must be defined after all the data fields (those with a use of B or blank).
- A field cannot be defined as both a data field and a program-to-system field; the field names must be unique.
- A program-to-system field can be named on an EVOKE, SECURITY, TIMER, or VARLEN keyword.
- The only valid keywords for a program-to-system field are ALIAS, FLTPCN, REFFLD and TEXT.

Location (Positions 39 through 44)

These positions do not apply to ICF files. Leave these positions blank unless you use them for comment text.

Keyword Entries (Positions 45 through 80)

This section contains keyword entries valid for defining ICF files. They are typed in positions 45 through 80 (functions). See "Syntax Rules" on page 1-4 for a discussion of the general rules for specifying keywords.

The following keywords are valid for ICF files:

ALIAS	FMTNAME	RCVTRNRND
ALWWRT	FRCDTA	RECID
CANCEL	INDARA	REF
CNLINVITE	INDTXT	REFFLD
CONFIRM	INVITE	RQSWRT
CTLDTA	NEGRSP	RSPCONFIRM
DETACH	RCVCANCEL	SECURITY
DFREVOKE	RCVCONFIRM	SUBDEV
ENDGRP	RCVCTLDTA	SYNLVL
EOS	RCVDETACH	TEXT
EVOKE	RCVENDGRP	TIMER
FAIL	RCVFAIL	VARBUFMGT
FLTPCN	RCVFMH	VARLEN
FMH	RCVNEGRSP	

ALIAS (Alternative Name)

Use this field-level keyword to specify an alternative name for a field. When the program is compiled, the alternative name is brought into the program instead of the DDS field name. The high-level language compiler in use determines whether the ALIAS name is used. Refer to the appropriate high-level language reference manual for information about ALIAS support for that language.

The format of the keyword is:

```
ALIAS(alternative-name)
```

Refer to "Syntax Rules" on page 1-4 for ALIAS naming conventions.

The alternative-name must be different from all other alternative names and from all DDS field names in the record format. If a duplicate name is found, an error is issued on the field name or alternative name.

An alternative name cannot be used within DDS or any other OS/400 function (for example, as a key field name, as the field name specified for the REFFLD keyword, or as a field name used in the Copy File (CPYF) command).

When you refer to a field that has the ALIAS keyword, the ALIAS keyword is copied in unless the ALIAS keyword is explicitly specified on the referencing field.

Option indicators are not valid for this keyword.

Figure 5-2 shows how to specify the ALIAS keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00070A            FIELDA        25A         ALIAS(CUSTOMERNAME)
     A
```

Figure 5-2. Specifying the ALIAS Keyword

ALWWRT (Allow Write)

Use this file- or record-level keyword so that your program can indicate when it has finished sending.

This keyword has no parameters.

ALWWRT is ignored at run time when DETACH, EOS, RSPCONFIRM, or RQSWRT are in effect. These keywords must have option indicators if they apply to a record for which ALWWRT applies. If a DETACH, EOS, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which ALWWRT applies, an error message is issued and the ALWWRT keyword is ignored at creation time.

You cannot specify ALWWRT with the TIMER keyword.

The ALWWRT keyword can be specified once at the file-level or once for each record format.

Option indicators are valid with this keyword. When you specify this keyword at the file-level, you should specify an option indicator.

Figure 5-3 shows how to specify the ALWWRT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
01000A  21                                  ALWWRT
02000A          R CUSMST
     A
```

Figure 5-3. Specifying the ALWWRT Keyword

CANCEL (Cancel)

Use this file- or record-level keyword to cancel the current chain of data (group of records) that is being sent to the remote program.

This keyword has no parameters.

The CANCEL keyword must have an option indicator when it applies to a record for which any of the following keywords apply:

CNLINVITE
EVOKE
RQSWRT
RSPCONFIRM
VARBUFMGT
VARLEN

Data fields and these keywords are ignored at run time when the CANCEL keyword is in effect. If a CANCEL keyword with no option indicator applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time. If a CANCEL keyword with no option indicator applies to a record with data fields, a severe error is issued and the file is not created.

The CANCEL keyword is ignored at run time when EOS, FAIL, or NEGRSP is in effect. These keywords must have option indicators if they apply to a record for which CANCEL applies. If a EOS, FAIL, or NEGRSP keyword with no option indicator applies to a record for which CANCEL applies, an error message is issued and the CANCEL keyword is ignored at creation time.

You cannot specify CANCEL with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-4 shows how to specify the CANCEL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A  02                                  CANCEL
     A           R RCD1
     A
```

Figure 5-4. Specifying the CANCEL Keyword

CNLINVITE (Cancel Invite)

Use this file- or record-level keyword to cancel any valid invite operation for which no input has yet been received.

This keyword has no parameters.

The CNLINVITE keyword must have an option indicator when it applies to a record for which a RQSWRT, RSPCONFIRM, or EVOKE keyword applies. At run time, these keywords are ignored when CNLINVITE is in effect. If a CNLINVITE keyword with no option indicator applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time.

The CNLINVITE keyword is ignored at run time when CANCEL, EOS, FAIL, or NEGRSP is in effect. These keywords must have option indicators when they apply to a record for which the CNLINVITE keyword applies. If a CANCEL, EOS, FAIL, or NEGRSP keyword with no option indicator applies to a record for which CNLINVITE applies, an error message is issued and the CNLINVITE keyword is ignored at creation time.

You cannot specify CNLINVITE with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-5 shows how to specify the CNLINVITE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD1                      CNLINVITE
     A
```

Figure 5-5. Specifying the CNLINVITE Keyword

CONFIRM (Confirm)

Use this file- or record-level keyword to request that the remote program confirms receiving the data.

This keyword has no parameters.

The CONFIRM keyword is valid only if the transaction was established with a synchronization level of confirm (SYNLVL(*CONFIRM) keyword). If the transaction was established with a synchronization level of none (SYNLVL(*NONE) keyword), the CONFIRM keyword is rejected with an OS/400 error message.

The CONFIRM keyword is ignored at run time when EOS, RSPCONFIRM, or RQSWRT is in effect. These keywords must have option indicators when they apply to a record for which the CONFIRM keyword applies. If an EOS, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which CONFIRM applies, an error message is issued and the CONFIRM keyword is ignored at creation time.

You cannot specify CONFIRM with the TIMER keyword.

The CONFIRM keyword can be specified once at the file-level or once for every record format.

Option indicators are valid with this keyword.

Figure 5-6 shows how to specify the CONFIRM keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD
00020A  01                                  CONFIRM
     A
```

Figure 5-6. Specifying the CONFIRM Keyword

If option indicator 01 is on, the remote program will confirm receiving the data by sending either a positive or negative response.

CTLDTA (Control Data)

Use this file- or record-level keyword to inform the remote program that control data is being sent.

This keyword has no parameters.

The CTLDTA keyword is ignored at run time when the EOS, RSPCONFIRM, or RQSWRT keywords are in effect. These keywords must have option indicators if they apply to a record for which CTLDTA applies. If an EOS, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which CTLDTA applies, an error message is issued and the CTLDTA keyword is ignored at creation time.

You cannot specify CTLDTA with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-7 shows how to specify the CTLDTA keyword at the record level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R SNDCTLD
     A                                        CTLDTA
     A            USRSCTLD     100A
     A
```

Figure 5-7. Specifying the CTLDTA Keyword

DETACH (Detach)

Use this file- or record-level keyword to explicitly inform the remote program that your program is done sending data and wants to end the transaction.

This keyword has no parameters.

The DETACH keyword must have an option indicator when it applies to a record for which any of the following keywords apply:

ALWWRT
ENDGRP
FMH
FRCDTA
INVITE
SUBDEV

At run time, these keywords will be ignored when the DETACH keyword is in effect. If a DETACH keyword with no option indicator applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time.

The DETACH keyword is ignored at run time when EOS, RSPCONFIRM, or RQSWRT is in effect. These keywords must have option indicators when they apply to a record for which the DETACH keyword applies. If an EOS, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which DETACH applies, an error message is issued and the DETACH keyword is ignored at creation time.

You cannot specify DETACH with the TIMER keyword.

At most, the DETACH keyword can be specified once at the file-level or once per record format.

Option indicators are valid for this keyword.

Figure 5-8 shows how to specify the DETACH keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD
00020A  01                                    DETACH
     A
```

Figure 5-8. Specifying the DETACH Keyword

If option indicator 01 is on, the transaction between your program and the remote program will be ended.

DFREVOKE (Defer Evoke)

Use this file- or record-level keyword with the EVOKE keyword to delay an evoke request until either the send buffer is full of data or a FRCDTA keyword is received. The DFREVOKE keyword is useful only for specialized applications that must have the data sent at the same time as the EVOKE.

This keyword has no parameters.

You cannot specify DFREVOKE with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-9 shows how to specify the DFREVOKE keyword at the record level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R PGMSTART
     A                                        EVOKE(&LIB/&PGMID)
     A                                        DFREVOKE
     A
     A
```

Figure 5-9. Specifying the DFREVOKE Keyword

ENDGRP (End of Group)

Use this file- or record-level keyword so that your program can indicate the end of a user-defined group of records.

This keyword has no parameters.

The ENDGRP keyword is ignored at run time when DETACH, EOS, RSPCONFIRM, or RQSWRT is in effect. These keywords must have option indicators when they apply to a record for which the ENDGRP keyword applies. If a DETACH, EOS, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which ENDGRP applies, an error message is issued and the ENDGRP keyword is ignored at creation time.

You cannot specify ENDGRP with the TIMER keyword.

Option indicators are valid for this keyword. (When you specify this keyword at the file-level, you should specify an option indicator.)

Figure 5-10 on page 5-13 shows how to specify the ENDGRP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00030A          R RECORD1                   ENDGRP
     A
```

Figure 5-10. Specifying the ENDGRP Keyword

EOS (End of Session)

Use this file- or record-level keyword to specify an end of session function. To end a session, your program issues a write operation with the EOS keyword in effect.

This keyword has no parameters.

The EOS keyword must have an option indicator when it applies to a record for which any of the following keywords apply:

ALWWRT	EVOKE	NEGRSP
CANCEL	FAIL	RQSWRT
CNLINVITE	FMH	RSPCONFIRM
CONFIRM	FMTNAME	SUBDEV
DETACH	FRCDTA	VARBUFMGT
ENDGRP	INVITE	VARLEN

At run time, data fields and these keywords are ignored when the EOS keyword is in effect. If an EOS keyword with no option indicator applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time. If an EOS keyword with no option indicator applies to a record with data fields, a severe error is issued and the file is not created.

You cannot specify EOS with the TIMER keyword.

Option indicators are valid for this keyword. When you specify this keyword at the file level, you should specify an option indicator.

Figure 5-11 shows how to specify the EOS keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A  01                              EOS
     A          R RCD
     A
```

Figure 5-11. Specifying the EOS Keyword

If indicator 01 is on and the program does an output operation, the session will be ended.

EVOKE (Evoke)

Use this file- or record-level keyword to start a program on the remote system.

The format of the keyword is:

```
EVOKE([library-/ele/]program-name[parameter-1...[parameter-255]])
```

The program-name can be any one of the following:

program-name
This is the name of the program to be started on the remote system. The name is syntax-checked at creation time for a valid AS/400 object name.

'character-string-1'
This is the name of the program to be started on the remote system. The name you specify must be in a format acceptable to the remote system because the character string will not be syntax-checked.

&field-name-1
The specified field contains the name of the program to be started on the remote system. The field name must be a valid field you have specified in the record format and must be a character field (data type of A). The name you specify must be in a format acceptable to the remote system.

The optional library-name can be any one of the following:

library-name/
This is the name of the library that contains the program to be started on the remote system. The name is syntax-checked at creation time for a valid AS/400 object name. For this keyword, *CURLIB and *LIBL are not valid names. If either one needs to be specified, a quoted character string should be used.

'character-string-2'/
This is the name of the library that contains the program to be started on the remote system. The name you specify must be in a format acceptable to the remote system because the character string will not be syntax-checked.

&field-name-2/
The specified field contains the name of the library that contains the program to be started on the remote system. The field name must be a valid field you have specified in the record format and must be a character field (data type of A). The name you specify must be in a format acceptable to the remote system.

Note: If the remote system is an AS/400 system and no library was specified, the library list will be used to search for the program.

Parameter-1...parameter-255 can be any of the following:

'character-string-3'
This is a character string that is passed to the program on the remote system. The character string must be in a format acceptable to the remote system because it will not be syntax-checked.

[&]field-name-3
This is the name of the field that contains the data you want passed to the program on the remote system. The field name must be a valid field you have specified in the record format.

numeric-value-3
This is a numeric value that is passed to the program on the remote system. The numeric value can be a negative or positive value (signed or unsigned). A decimal point of , or . is optional. No decimal alignment will be performed. Leading zeros will not be suppressed. The data is sent as a zoned decimal value. The following are all valid numeric values:

999.6
-999,6
01587

Special Considerations

The following are special considerations when using the EVOKE keyword.

- When the EVOKE keyword is specified at the file-level, you cannot specify a field name as a parameter value.
- The maximum length allowed for the combined program name and library name is 64. The slash between the program name and the library name is counted as part of the 64 bytes. APPC does not send the slash unless it is specified within a literal (for example, LIBRARY/PROGRAM).
- The total length of parameter-1 through parameter-255 cannot be more than 32 767 bytes.

  Note: In calculating the maximum length of PIP data for APPC, the following must be considered:

  Four bytes must be added to the length of each of these parameters. An additional 4 bytes must be added if any parameters are specified. These bytes are required by the system.

  Use the following formula to determine the total length of the parameters:

  4 + (length of 1st parameter + 4) + (length of 2nd parameter + 4) + ... (length of nth parameter + 4)

  Following is an example using this formula:

  EVOKE(LIBRARY1/PROGRAM1 'THIS IS AN EXAMPLE OF A CHARACTER STRING' &FIELD1 35)

  Assume that &FIELD1 has a length of 10.
  4 + (40 + 4) + (10 + 4) + (2 + 4) = 68
- The length of each parameter (parameter-1 through parameter-255) should be the same as the length of the corresponding parameter in the remote program.
- If a field name with a usage of P is specified as a parameter of the EVOKE keyword, this field is not sent as part of the data record.
- A program evoked on an AS/400 system will receive any parameters sent by the remote program just as if they had been passed by the CL CALL command.

  Note: If the job on the AS/400 system is a prestart job, the program must use the RTVDTAARA command to receive the parameters.

This keyword is required when either the SECURITY or SYNLVL keywords are specified. At run time, the SECURITY and SYNLVL keywords are used only when EVOKE is also in effect.

The EVOKE keyword is ignored at run time when CANCEL, CNLINVITE, EOS, FAIL, NEGRSP, RSPCONFIRM, or RQSWRT is in effect. These keywords must have option indicators when they apply to a record for which the EVOKE keyword applies. If a CANCEL, CNLINVITE, EOS, FAIL, NEGRSP, or RQSWRT keyword

with no option indicator applies to a record for which EVOKE applies, an error message is issued and the EVOKE keyword is ignored at creation time.

You cannot specify EVOKE with the TIMER keyword.

Option indicators are valid for this keyword and are required if this keyword is specified more than once for each record format or file.

Figure 5-12 shows how to specify the EVOKE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD
00020A  01        :                       EVOKE(LIBRARY1/PROGRAM1)    1
00030A  02        :                       EVOKE(LIBRARY2/PROGRAM2)    1
     A            :                        :
     A            :                        :
00090A          R RCD2                    EVOKE(&FIELD2/&FIELD1 'ABC' 10.1 +
00100A                                    FIELD3)             2
00110A            FIELD1        10A  P
00120A            FIELD2        10A  P
00130A            FIELD3         5B  P
     A
```

Figure 5-12. Specifying the EVOKE Keyword

1 If indicator 01 is on, PROGRAM1 in LIBRARY1 will be started. If indicator 02 is on, PROGRAM2 in LIBRARY2 will be started.

2 &FIELD1 contains the name of the program to be started. &FIELD2 contains the name of the library. The character string ABC, numeric value 10.1, and the value in FIELD3 will be passed to the program on the remote system.

FAIL (Fail)

Use this file- or record-level keyword to inform the remote program that the data sent or received was invalid.

This keyword has no parameters.

The FAIL keyword must have an option indicator when it applies to a record that has data fields (use of B or blank) or for which any of the following keywords apply:

CANCEL
CNLINVITE
EVOKE
NEGRSP
RQSWRT
RSPCONFIRM
VARBUFMGT
VARLEN

At run time, data fields and these keywords will be ignored when the FAIL keyword is in effect. If a FAIL keyword with no option indicator applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time. If a FAIL keyword with no option indicator applies to a record with data fields, a severe error is issued and the file will not be created.

The FAIL keyword is ignored at run time when the EOS keyword is in effect. EOS must have an option indicator when it applies to a record for which the FAIL keyword applies. If an EOS keyword with no option indicator applies to a record for which FAIL applies, an error message is issued and the FAIL keyword is ignored at creation time.

FAIL cannot be specified with the TIMER keyword.

Option indicators are valid for this keyword. When you specify this keyword at the file level, you should specify an option indicator.

Figure 5-13 shows how to specify the FAIL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R INQ
00020A  99                                  FAIL
     A
```

Figure 5-13. Specifying the FAIL Keyword

FLTPCN (Floating-Point Precision)

Use this field-level keyword to specify the precision of a floating-point field.

The format of the keyword is:

```
FLTPCN(*SINGLE | *DOUBLE)
```

where the *SINGLE parameter is single precision and the *DOUBLE parameter is double precision.

This keyword is valid for floating-point fields only (data type F).

A single-precision field can be up to 9 digits. A double-precision field can be up to 17 digits. If you specify a field length greater than 9 (single precision) or 17 (double precision), an error message is issued and the file is not created. ICF supports a floating-point accuracy of 7 digits for single precision and 15 digits for double precision.

Option indicators are not valid for this keyword.

Figure 5-14 shows how to specify the FLTPCN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00090A            FIELDA        17F 4       FLTPCN(*DOUBLE)
     A
```

Figure 5-14. Specifying the FLTPCN Keyword

FMH (Function Management Header)

Use this file- or record-level keyword to inform the remote program that a function management header (FMH) is being sent.

This keyword has no parameters.

The FMH keyword is ignored at run time when EOS, DETACH, RSPCONFIRM, or RQSWRT is in effect. These keywords must have option indicators when they apply to a record for which the FMH keyword applies. If an EOS, DETACH, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which FMH applies, an error message is issued and the FMH keyword is ignored at creation time.

You cannot specify FMH with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-15 shows how to specify the FMH keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD                         FMH
     A            FLD1          10A  B
     A
```

Figure 5-15. Specifying the FMH Keyword

FMTNAME (Format Name)

Use this file- or record-level keyword to specify that the record format name is to be sent to the remote program when your program issues an output operation.

This keyword has no parameters.

The FMTNAME keyword is ignored at run time when EOS, RSPCONFIRM, or RQSWRT is in effect. These keywords must have option indicators when they apply to a record for which the FMTNAME keyword applies. If an EOS, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which FMTNAME applies, an error message is issued and the FMTNAME keyword is ignored at creation time.

You cannot specify FMTNAME with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-16 shows how to specify the FMTNAME keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD1
     A  01                                  FMTNAME
     A             FIELD1         10A  B
     A
```

Figure 5-16. Specifying the FMTNAME Keyword

If indicator 01 is on and the program does a write operation, the record format name will be sent as an APPC map name to the remote system.

FRCDTA (Force Data)

Use this record-level keyword to clear the buffer when there is no more data to send, without waiting for the buffer to become full.

Note: If the keyword is specified after each write statement, performance problems may occur.

There is no wait for confirmation. (The CONFIRM keyword provides similar function but additionally provides confirmation of data sent. Your program must wait for the response from the other end before continuing to the next program statement.)

This keyword has no parameters.

The FRCDTA keyword is ignored at run time when any of the following keywords is in effect:

DETACH
EOS
RQSWRT
RSPCONFIRM

These keywords must have option indicators when they apply to a record specifying FRCDTA. If a keyword from this list has no option indicator and applies to a record with FRCDTA, an error message is issued and the FRCDTA keyword is ignored at creation time.

You cannot specify FRCDTA with the TIMER keyword.

The FRCDTA keyword can be specified at most once per record format.

Option indicators are valid for this keyword.

Figure 5-17 shows how to specify the FRCDTA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A        R REC1
00020A  10                                    FRCDTA
00030A          FLD1            10
00040A          FLD2             5
     A
```

Figure 5-17. Specifying the FRCDTA Keyword

When option indicator 10 is on and the program does a write operation, the FRCDTA keyword sends communications data currently held in the buffer.

INDARA (Indicator Area)

Use this file-level keyword to remove option and response indicators from the buffer, or record area, and place them in a 99-byte separate indicator area. Specifying the INDARA keyword provides the following advantages:

- Simplifies COBOL/400* programming when both option and response indicators are used. If the same indicator is used as a response indicator and as an option indicator, both indicators always have the same value, regardless of the order in which they are specified in the DDS.
- Assists the RPG/400* programmer using program-described WORKSTN files.

This keyword has no parameters.

If you specify the INDARA keyword, some high-level languages require that you specify in your program that a separate indicator area is to be used. See the appropriate high-level language manual.

If you specify the INDARA keyword, you can add, change, or delete option and response indicators in the DDS and recompile the file without recompiling the high-level language program. This is allowed because the field locations in the buffer have not changed and, therefore, the level check data has not changed. However, if the program is to take advantage of the new indicators, the program would still need to be changed and recompiled.

Option indicators are not valid for this keyword.

Figure 5-18 shows how to specify the INDARA keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     INDARA
00020A  41                                 FAIL
00030A                                     RCVTRNRND(14 'Turn around')
00040A           R RCD
00050A             ACTNBR        10
     A
```

Figure 5-18. Specifying the INDARA Keyword

With the INDARA keyword specified, option indicator 41 and response indicator 14 are removed from the buffer for RCD and placed in the separate indicator area. Only ACTNBR, a named field, remains in the buffer for record format RCD.

INDTXT (Indicator Text)

Use this file- or record-level keyword to associate descriptive text (indicating intent or use) with a specific response or option indicator.

The format of the keyword is:

```
INDTXT(response-or-option-indicator 'indicator-text')
```

You can specify this keyword once for each response and option indicator.

Indicator-text is a required parameter value, and must be a character string enclosed in apostrophes. If the length of the string is greater than 50 positions, only the first 50 characters are used by the high-level language compiler. The text is used during compilation to help program documentation.

The INDTXT keyword does not cause the specified indicator to appear in either the input or output record area. It provides text to be associated with the indicator. Once an indicator has been given a textual assignment (either by this keyword or by the response indicator text), no other textual assignment is made. A message is issued and the keyword is ignored. This differs from other keywords that can have indicators specified as parameter values; for other keywords, only the text is ignored.

Option indicators are not valid for this keyword.

Figure 5-19 shows how to specify the INDTXT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     INDTXT(02 'Alternate month')
00020A           R MASTER
00030A             MTH                   2 10
00040A  02         ALTMTH                2 10
     A
```

Figure 5-19. Specifying the INDTXT Keyword

The INDTXT keyword describes the use of option indicator 02. In a compiler computer printout for a high-level language, 'Alternate month' would be printed as a comment with the description of indicator 02.

INVITE (Invite)

Use this file- or record-level keyword to invite the program device for a later read. To send an invite to the program device, your program issues a write operation to that program device with the INVITE keyword in effect.

The INVITE keyword provides some performance improvement if your application program is doing interactive processing with the program device. Normally, a read request is sent to a device when your program issues an input operation. However, the INVITE keyword allows you to request the read when you issue the output operation. After the output operation is complete, your program can do other processing while the invited program device is sending data and the OS/400 program processes the received data. This may improve the performance of your program. When your application program is ready to process the data, it issues an input operation.

This keyword has no parameters.

The INVITE keyword is ignored at run time when EOS, RSPCONFIRM, or DETACH is in effect. These keywords must have option indicators when they apply to a record for which the INVITE keyword applies. If an EOS, RSPCONFIRM, or DETACH keyword with no option indicator applies to a record for which INVITE applies, an error message is issued and the INVITE keyword is ignored at creation time.

You cannot specify INVITE with the TIMER keyword.

Option indicators are valid for this keyword.

The INVITE keyword cannot be specified at both the file- and record-level.

Figure 5-20 shows how to specify the INVITE keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  01                                  INVITE
00020A          R RCD1
00030A            FLD1          10
00040A            FLD2           5
     A
```

Figure 5-20. Specifying the INVITE Keyword

The INVITE keyword is in effect only when option indicator 01 is set on.

NEGRSP (Negative Response)

This file- or record-level keyword sends a negative response to the remote program to indicate that your program detected an error in the data received.

The format of the keyword is:

```
NEGRSP[(&field-name)]
```

The optional parameter, &field-name, specifies the name of a field that contains sense data to be sent to the remote program with the negative response. The

specified field name must exist in the record format, and the field must be a character field with length at least 8, data type A, and usage B or blank.

The NEGRSP keyword must have an option indicator when it applies to a record for which any of the following keywords apply:

CANCEL
CNLINVITE
EVOKE
RQSWRT
RSPCONFIRM
VARBUFMGT
VARLEN

At run time, these keywords are ignored when the NEGRSP keyword is in effect. If a NEGRSP keyword with no option indicator applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time.

NEGRSP is ignored at run time when EOS or FAIL is in effect. These keywords must have option indicators if they apply to a record for which NEGRSP applies. If an EOS or FAIL keyword with no option indicator applies to a record for which NEGRSP applies, an error message is issued and the NEGRSP keyword is ignored at creation time.

When you specify NEGRSP at the file-level, you cannot specify the field name parameter.

You cannot specify NEGRSP with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-21 shows how to specify the NEGRSP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD1
     A  01                                  NEGRSP(&FIELDB)
     A            FIELDA        25A  B
     A            FIELDB        80A  B
     A
```

Figure 5-21. Specifying the NEGRSP Keyword

If indicator 01 is on, a write operation to RCD1 will send a negative response and send the first 8 bytes of FIELDB to the remote program. Note that no data from RCD1 other than the sense data will be sent with the negative response.

RCVCANCEL (Receive Cancel)

Use this file- or record-level keyword to set on a response indicator to inform your program that the remote program has sent a CANCEL.

The format of the keyword is:

```
RCVCANCEL(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. The text has no function in the file or the program other than as a comment. The apostrophes are required. If

you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

You cannot specify RCVCANCEL with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-22 shows how to specify the RCVCANCEL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD1                      RCVCANCEL(34 'Received - +
     A                                      Cancel')
     A
```

Figure 5-22. Specifying the RCVCANCEL Keyword

Indicator 34 is set on when a CANCEL is received on an input operation from RCD1.

RCVCONFIRM (Receive Confirm)

Use this file- or record-level keyword to set on a response indicator if the data received by your program contains a confirmation request from the remote program.

The format of the keyword is:

```
RCVCONFIRM(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. The text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

You cannot specify RCVCONFIRM with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-23 shows how to specify the RCVCONFIRM keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                  RCVCONFIRM(44 'Waiting for a +
00020A                                  response')
00030A          R RCD
     A
```

Figure 5-23. Specifying the RCVCONFIRM Keyword

Response indicator 44 is set on to indicate receipt of the confirmation request from the remote program.

RCVCTLDTA (Receive Control Data)

Use this file- or record-level keyword to set on a response indicator to inform your program that control data has been received.

The format of this keyword is:

```
RCVCTLDTA(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. The text has no function in the file or the program except as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, only the first 50 characters are printed.

You cannot specify RCVCTLDTA with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-24 shows how to specify the RCVCTLDTA keyword at the record level.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCVCTLD
     A                                    RCVCTLDTA(66 'received control +
     A                                    data')
     A            USRRCTLD     100A
     A
```

Figure 5-24. Specifying the RCVCTLDTA Keyword

RCVDETACH (Receive Detach)

Use this file- or record-level keyword to set on a response indicator if the remote program is ending the transaction.

The format of the keyword is:

```
RCVDETACH(response-indicator [‘text’])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. The text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

You cannot specify RCVDETACH with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-25 shows how to specify the RCVDETACH keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                               RCVDETACH(44 'Transaction is  +
00020A                               finished')
00030A          R RCD
     A
```

Figure 5-25. Specifying the RCVDETACH Keyword

Response indicator 44 is set on when the remote program ends the transaction.

RCVENDGRP (Receive End of Group)

Use this file- or record-level keyword to set on a response indicator to inform your program of the end of a user-defined group of records.

The format of the keyword is:

```
RCVENDGRP(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

You cannot specify RCVENDGRP with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-26 shows how to specify the RCVENDGRP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A          R CUSMST
00200A                                      RCVENDGRP(68 'End of group received-
     A                                      ')
     A
```

Figure 5-26. Specifying the RCVENDGRP Keyword

Response indicator 66 is set on when the remote program indicates that this is the end of a user-defined group of records.

RCVFAIL (Receive Fail)

Use this file- or record-level keyword to set on a response indicator when the local program determines that the remote program has sent a FAIL. If this condition occurs when the RCVFAIL keyword was not specified, an OS/400 message notifies the local program that the remote program has sent a FAIL.

The format of the keyword is:

```
RCVFAIL(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. The text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

Option indicators are not valid for this keyword.

Figure 5-27 shows how to specify the RCVFAIL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                      RCVFAIL(10 'Fail received')
00020A
00030A          R RCD
     A
```

Figure 5-27. Specifying the RCVFAIL Keyword

Indicator 10 is set on when the remote program sends a fail indication.

RCVFMH (Receive Function Management Header)

Use this file- or record-level keyword to set on a response indicator to inform your program that a function management header has been received.

The format of the keyword is:

```
RCVFMH(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

You cannot specify RCVFMH with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-28 shows how to specify the RCVFMH keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      RCVFMH(24 'Received FMH')
     A          R RCD1
     A
```

Figure 5-28. Specifying the RCVFMH Keyword

Indicator 24 is set on when a function management header is received.

RCVNEGRSP (Receive Negative Response)

Use this file- or record-level keyword to set on a response indicator to inform your program that the remote program has sent a negative response.

The format of the keyword is:

```
RCVNEGRSP(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

You cannot specify RCVNEGRSP with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-29 shows how to specify the RCVNEGRSP keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      RCVNEGRSP(67 'Negative re-
     A                                      sponse')
     A          R RCD1
     A
```

Figure 5-29. Specifying the RCVNEGRSP Keyword

Indicator 67 is set on when a negative response is received.

RCVTRNRND (Receive Turnaround)

Use this file- or record-level keyword to set on a response indicator to inform your program that the sending program has stopped sending and has given the local program the right to send.

The format of the keyword is:

```
RCVTRNRND(response-indicator ['text'])
```

The response-indicator parameter is required.

The optional text is included on the computer printout created at program compilation time to explain the intended use of the indicator. This text has no function in the file or the program other than as a comment. The apostrophes are required. If you specify more than 50 characters between the apostrophes, the text is truncated to 50 characters on the program computer printout.

You cannot specify RCVTRNRND with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-30 shows how to specify the RCVTRNRND keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                  RCVTRNRND(44 'Host has stopped +
00020A                                  sending')
00030A          R CUSMST
     A
```

Figure 5-30. Specifying the RCVTRNRND Keyword

RECID (Record Identification)

Use this record-level keyword to allow your program to identify a record-by-record format when it issues a read-from-invited-devices operation using the name of the file. When you use an input operation, the OS/400 program compares data in the record received with the selection value specified in the parameter values. The selection value is that data beginning at the specified starting position must equal the specified compare value. Your program can then determine the record format of the data just read.

The format of the keyword is:

```
RECID(starting-position compare-value)
```

The starting-position parameter specifies a position relative to the start of the data in the buffer (disregarding indicators) to test for the record's ID. If the INDARA keyword is used, the start of data and buffer positions are the same. For a description of the buffer, see Figure 5-33 on page 5-30. The position parameter can be either

```
nnnnn
```

or

```
*POSnnnnn
```

where nnnnn is a number that is one to five digits long. For example, the following are equivalent pairs:

```
1 and *POS1
34 and *POS34
12025 and *POS12025
```

The compare-value parameter can be one of the following:

Value	Meaning
*ZERO	The value to be tested for is zero (hex F0). Equivalent to 0.
*BLANK	The value to be tested for is blank (hex 40). Equivalent to blank.
'character-string'	The value to be tested for is the specified character string. The length of the string is limited to the length from the RECID position parameter specified to the end of the shortest nonzero record format in the file (not including that record format's indicators or program fields).

A record format specifying the RECID keyword must contain at least one data field (usage of B).

You can specify the RECID keyword more than once in a record format. If you do so, data in the record is compared with each RECID keyword in the order specified until a match is found. The first record format whose selection value is satisfied by the data is the record format selected. If no match is found or no user data is received, the RECID default record format is used. The RECID default record format will be the first record format in the file that does not have the RECID keyword specified for it. However, if every record format in the file has the RECID keyword specified for it, the default record format will be the first record format in the file.

A message is issued to your program when data is received and no match is found and the RECID default record format has the RECID keyword specified for it.

When comparing the data received with the RECID keyword, if the position to be compared is beyond the last byte of data received, the data will be assumed to be blanks (hex 40).

This keyword is ignored at program run time unless the FMTSLT(*RECID) parameter is specified on the ADDICFDEVE, CHGICFDEVE, or OVRICFDEVE command.

You cannot specify RECID on the same record format as the VARBUFMGT keyword.

Option indicators are not valid for this keyword.

Figure 5-31 through Figure 5-34 show how to specify the RECID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R DFTFMT
00020A            ID              3A
00030A            FLD1           20A
00040A            FLD2            5B 0
     A
00050A          R RCD1                        RECID(1 'ABC')
00060A            ID              3A
00070A            FLD1           10S 0
00080A            FLD2            5B 0
     A
00090A          R RCD2                        RECID(1 'DEF')
00100A            ID              3A
00110A            FLD1           10S 0
00120A            FLD2            5A
00130A            FLD3            2B 0
     A
```

Figure 5-31. Specifying the RECID Keyword (Default Record Format)

Record format DFTFMT will be the RECID default record format.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD1                        RECID(1 'H')
00020A            ID              1A
00030A            FLD1           10A
00040A            FLD2           10A
00050A            FLD3            6S 2
     A
00060A          R RCD2                        RECID(1 'D')
00070A            ID              1A
00080A            FLD1            8S 2
00090A            FLD2           10A
00100A            FLD3            5B 0
     A
00110A          R RCD3                        RECID(1 'L')
00120A            ID              1A
00130A            FLD1           50A
     A
```

Figure 5-32. Specifying the RECID Keyword (Default Record Format)

Record format RCD1 will be the RECID default record format. If no match is found, an escape message will be issued to your program because the RECID default record format has the RECID keyword specified for it. If no data is received, record format RCD1 will be used.

An application program reads header and detail records from an ICF file. The program issues input operations to the file name (not to individual record names) and receives the records (headers and details) in the order the sending application sends them. In this example, the sending and receiving applications provide for an explicit code (an H for header records and a D for detail records) to identify which type of record is being sent and received. The RECID keyword identifies where in

the input buffer (disregarding indicators) the H or D appears and specifies the value (starting in the position specified) that identifies the type of record.

In Figure 5-33, three record formats are defined in ICF file. The application program issues input operations using the file name, for instance, RPTFILE.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R HEADER                    RECID(1 'H')
00020A                                      RCVTRNRND(10 'Host stopped sending')
00030A            CODE           1
00040A            TITLE         30
00050A            ACTNBR         6  0
00060A          R DETAIL                    RECID(1 'D')
00070A            CODE           1
00080A            ITMNBR         8  0
00090A            DESCRP        20
00100A          R CATCH
00110A            FIELD         37
     A
```

Figure 5-33. Specifying the RECID Keyword (Purpose of the RECID Keyword)

Assume that the records received on nine successive input operations are one header, then three details, then one header, then four details. The sending application must identify the headers by placing an H in field CODE and the details by placing a D in field CODE. For each input operation, the OS/400 program compares the value in position 1 in the buffer with the value specified on the RECID keyword. (Position 1 is the location of field CODE in the buffer.) If the value in a record is H, the OS/400 program selects record format name HEADER; if the value in a record is D, the OS/400 program selects record format name DETAIL.

Record format CATCH (the RECID default record format) is the record format name selected if a record is received that does not contain either H or D in the first position of the data portion of the buffer.

Following is the buffer for record format HEADER:

Response indicator 10 (1 byte)

CODE (1 byte)

TITLE (30 bytes)

ACTNBR (6 bytes)

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD1                       RECID(1 'ABC')
     A            FLD1          10
     A          R RCD2                       RECID(1 'AB')
     A            FLD1          10
     A          R RCD3                       RECID(1 'A')
     A            FLD1          10
     A          R CATCH
     A            FIELD         10
     A
```

Figure 5-34. Specifying the RECID Keyword (Relation between the RECID Keywords when the Value Parameters Have Similar Values)

Three record formats need to be distinguished from one another; the first character in the value parameter is the same. Specifying the most specific (longest) value parameter first in the DDS enables the OS/400 program to distinguish the first record format from the others. The reason is that if the first 10 positions of the buffer contain ABCDEFGHIJ, and RCD3 were specified first, RCD3 would be identified even though RCD1 is desired. RCD1 and RCD2 could not be identified because the OS/400 program does not test after one successful match.

REF (Reference)

Use this file-level keyword to specify the name of a file from which field descriptions are to be retrieved. Use it when you want to duplicate descriptive information from several fields in a previously defined record format. This keyword allows you to code the file name once rather than on each field that refers to the file as is required by the REFFLD keyword. To refer to more than one file, use the REFFLD keyword. The REF keyword can be specified only once.

The format of the keyword is:

```
REF([library-name/]database-file-name [record-format-name])
```

If there is more than one record format in the reference file, specify a record format name as a parameter value for this keyword to tell the OS/400 program which one to use unless the record formats should be searched sequentially.

The database-file-name is a required parameter value for this keyword. The library-name and the record-format-name are optional.

If you do not specify the library-name, the current library list (*LIBL) at file creation time is used. If the record-format-name is not specified, each format is searched in order (as they are specified). The first occurrence of the field is used. For more information, see Appendix A, "When to Specify REF and REFFLD."

You can specify a Distributed Data Management (DDM) file on this keyword. When using a DDM file, the database-file-name and the library-name are the DDM file name and library name on the source system. The record-format-name is the record format name in the remote file on the target system.

Option indicators are not valid for this keyword.

Figure 5-35 and Figure 5-36 show how to specify the REF keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      REF(FILE1)
     A          R RECORD
     A            FLD1       R
     A
```

Figure 5-35. Specifying the REF Keyword (Example 1)

FLD1 has the same attributes as the first (or only) FLD1 in the file, FILE1.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      REF(LIB1/FILE1 RECORD2)
     A          R RECORD
     A            FLD1       R
     A
```

Figure 5-36. Specifying the REF Keyword (Example 2)

FLD1 has the same attributes as FLD1 in RECORD2 in FILE1 in LIB1.

REFFLD (Referenced Field)

Use this field-level keyword when referring to a field under one of these three conditions:

- The name of the referenced field is different from the name in positions 19 through 28.
- The name of the referenced field is the same as the name in positions 19 through 28, but the record format, file, or library of the referenced field is different from that specified with the REF keyword.
- The referenced field occurs in the same DDS source file as the referencing field.

The format of the keyword is:

```
REFFLD([record-format-name/]referenced-field-name [  {*SRC |
[library-name/]database-file-name}])
```

The referenced-field-name is required even if it is the same as the referencing field. Use the record-format-name when the referenced file contains more than one record format. Use *SRC (rather than the database-file-name) when the referenced-field-name is in the same DDS source file as the referencing field. *SRC is the default value when the database-file-name, the library-name, and the REF keyword are not specified.

Note: When you refer to a field in the same DDS source file, the field you are referring to must precede the field you are defining.

Specify the database-file-name (qualified by its library-name if necessary) when you want to search a particular database file.

If, in the same DDS source file, you specify REF at the file-level and REFFLD at the field level, the particular search sequence depends on both the REF and REFFLD keywords. For more information, see Appendix A, "When to Specify REF and REFFLD."

The letter R must be specified in position 29. In some cases, if you specify a value for length, some keywords specified with the field in the database file are not included in the ICF file. For more information, see "Reference (Position 29)" on page 5-4.

You can specify a Distributed Data Management (DDM) file on this keyword.

When using a DDM file, the database-file-name and the library-name are the DDM file and library name on the source system. The referenced-field-name and the record-format-name are the field name and the record format name in the remote file on the target system.

Option indicators are not valid for this keyword.

Figure 5-37 shows how to specify the REFFLD keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R FMAT1
00020A            ITEM           5
00030A            ITEM1     R               REFFLD(ITEM)
00040A            ITEM2     R               REFFLD(FMAT1/ITEM)
00050A            ITEM3     R               REFFLD(ITEM FILEX)
00060A            ITEM4     R               REFFLD(ITEM LIBY/FILEX)
00070A            ITEM5     R               REFFLD(FMAT1/ITEM LIBY/FILEX)
00080A            ITEM6     R               REFFLD(ITEM *SCR)
     A
```

Figure 5-37. Specifying the REFFLD Keyword

Because the REF keyword is not specified, the default for lines 00030 and 00040 is to search the DDS source file in which they are specified. In line 00080, the parameter value *SRC explicitly specifies the source file.

RQSWRT (Request Write)

Use this file- or record-level keyword to request permission for your program to send data.

This keyword has no parameters.

The RQSWRT keyword must have an option indicator when it applies to a record for which any of the following keywords apply:

ALWWRT
CONFIRM
DETACH
ENDGRP
EVOKE
FMH
FMTNAME
FRCDTA
SUBDEV
VARBUFMGT
VARLEN

At run time, these keywords are ignored when the RQSWRT keyword is in effect. If a RQSWRT keyword with no option indicator applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time.

The RQSWRT keyword is ignored at run time when CANCEL, CNLINVITE, EOS, FAIL, RSPCONFIRM, or NEGRSP is in effect. These keywords must have option indicators when they apply to a record for which the RQSWRT keyword applies. If a CANCEL, CNLINVITE, EOS, FAIL, or NEGRSP keyword with no option indicator applies to a record for which RQSWRT applies, an error message is issued and the RQSWRT keyword is ignored at creation time.

You cannot specify RQSWRT with the TIMER keyword.

Option indicators are valid for this keyword. When you specify this keyword at the file level, you should specify an option indicator.

Figure 5-38 shows how to specify the RQSWRT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CUSMST
00020A  14                                      RQSWRT
     A
```

Figure 5-38. Specifying the RQSWRT Keyword

RSPCONFIRM (Respond Confirm)

Use this file- or record-level keyword to send a positive response to a received confirm request.

This keyword has no parameters.

The RSPCONFIRM keyword must have an option indicator when it applies to a record for which any of the following keywords apply:

ALWWRT	FMH	RQSWRT
CONFIRM	FMTNAME	SUBDEV
DETACH	FRCDTA	VARBUFMGT
ENDGRP	INVITE	VARLEN
EVOKE		

At run time, data fields and these keywords are ignored when the RSPCONFIRM keyword is in effect. If an unoptioned RSPCONFIRM keyword applies to a record for which any of these keywords apply, error messages are issued and these keywords are ignored at creation time. If an unoptioned RSPCONFIRM applies to a record with data fields, a severe error is issued and the file is not created.

The RSPCONFIRM keyword is ignored at run time when EOS, FAIL, NEGRSP, CANCEL, or CNLINVITE is in effect. These keywords must have option indicators when they apply to a record for which RSPCONFIRM applies. If an unoptioned EOS, FAIL, NEGRSP, CANCEL, or CNLINVITE applies to a record for which RSPCONFIRM applies, an error message is issued and the RSPCONFIRM keyword is ignored at creation time.

Option indicators are valid for this keyword. When you specify this keyword at the file level, you should specify an option indicator.

You cannot specify RSPCONFIRM with the TIMER keyword.

Figure 5-39 shows how to specify the RSPCONFIRM keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD
     A
     A  20                                  RSPCONFIRM
     A
```

Figure 5-39. Specifying the RSPCONFIRM Keyword

If option indicator 20 is on, an output operation to RCD will send a positive response to the confirm request received from the remote program.

SECURITY (Security)

Use this file- and record-level keyword to include security information when your program starts a program on a remote system (see the EVOKE keyword). Any record format that has the SECURITY keyword specified for it or implied for it by being specified at the file level must have the EVOKE keyword specified on that record format or implied for that record format by being specified at the file level. If you do not specify the EVOKE keyword, a severe error occurs and the file is not created.

The format of the keyword is:

```
SECURITY(security-subfield subfield-definition[.3.])
```

The security-subfield parameter identifies the subfield being defined. This parameter is required. The value specified must be one of the following:

Value	Meaning
1	(Profile ID)
2	(Password)
3	(User ID)

The subfield-definition parameter must be one of the following:

***USER**
Indicates that the user profile name of the AS/400 user should be used as the value of the security subfield. For example, if *USER is specified for the password subfield, the user profile name is used as the password.

***NONE**
Indicates that a null security value should be used.

'character-string'
You may specify up to 10 characters of security information.

field-name
The specified field contains the security information. The length of the field cannot exceed 10 characters. This parameter is not valid if the SECURITY keyword is specified at the file level.

&field-name
The specified field contains the security information. The length of the field cannot exceed 10 characters. This parameter is not valid if the SECURITY keyword is specified at the file level.

You cannot specify SECURITY with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-40 shows how to specify the SECURITY keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  01                                 SECURITY(2 'JONES' 3 'WHITE')
00020A
00030A
00040A
00050A
00060A          R RCD1
00070A  03                                 SECURITY(2 'JONES' 3 *USER)
00080A                                     EVOKE(LIB2/PGM2)
00090A
00100A          R RCD2
00110A                                     EVOKE(LIB3/PGM3)
00120A
00130A            FIELD1         5A
00140A
00150A          R RCD3
00160A  60                                 SECURITY(2 &CLVAR1 3 &CLVAR2)
00170A                                     EVOKE(LIB4/PGM4)
00180A            CLVAR1        10A
00190A            CLVAR2        10A
     A
```

Figure 5-40. Specifying the SECURITY Keyword

SECURITY specified at the file-level applies to all formats and if selected (indicator 01 is on), the password of JONES and user ID of WHITE are sent to the remote system.

For RCD1, if indicator 03 is set on, the user profile name of the current user is used as the user ID and is sent with the password JONES as security information to the remote system.

For RCD2, no security information is sent to the remote system.

For RCD3, if indicator 60 is set on, the value contained in CLVAR1 is used as the password, the value in CLVAR2 is used as the user ID, and both are sent as security information to the remote system.

SUBDEV (Subdevice)

Use this file- or record-level keyword to allow your program to request a specific subdevice (for example, a printer) to which transmitted data should be directed.

The format of the keyword is:

```
SUBDEV(*DC1 | *DC2 | *DC3 | *DC4)
```

The SUBDEV keyword is ignored at run time when EOS, DETACH, RSPCONFIRM, or RQSWRT is in effect. These keywords must have option indicators when they apply to a record for which the SUBDEV keyword applies. If an EOS, DETACH, RSPCONFIRM, or RQSWRT keyword with no option indicator applies to a record for which SUBDEV applies, an error message is issued and the SUBDEV keyword is ignored at creation time.

You can specify only one parameter value for each SUBDEV keyword.

You can specify this keyword more than once in the file; however, you cannot specify the same parameter value at the file-level and again at the record-level. This is true even if you specify option indicators each time. For example, if you specify SUBDEV(*DC1) at the file-level, you cannot specify SUBDEV(*DC1) anywhere else in the file.

If you specify the SUBDEV keyword at both the file-level and the record level, and your program selects the one at the file-level, the record-level keyword(s) have no effect even if also selected.

You can specify the SUBDEV keyword a maximum of four times for each record format. If you specify the SUBDEV keyword more than once, you must specify option indicators each time, and you can specify each keyword value only once.

The OS/400 program sends a device selection character as follows. The meaning of the device selection character is set by the remote system or device.

Parameter Value	Character Sent
*DC1	Hex 11
*DC2	Hex 12
*DC3	Hex 13
*DC4	Hex 5D

You cannot specify SUBDEV with the TIMER keyword.

Option indicators are valid for this keyword.

Figure 5-41 shows how to specify the SUBDEV keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A  01                                  SUBDEV(*DC1)
00020A  02                                  SUBDEV(*DC4)
     A          R RECORD
     A
```

Figure 5-41. Specifying the SUBDEV Keyword

If indicator 01 is on, the OS/400 program sends the component selection character hex 11 on an output operation (no matter how indicator 02 is set).

If indicator 02 is on and indicator 01 is off, the OS/400 program sends component selection character hex 5D.

SYNLVL (Synchronization Level)

This file- and record-level keyword indicates the level of synchronization your program requires. The SYNLVL keyword is valid only when the EVOKE keyword is in effect.

The format of the keyword is:

```
SYNLVL[(*NONE | *CONFIRM)]
```

Specify *NONE when neither your program nor the remote program will use the CONFIRM keyword. Specify *CONFIRM if either your program or the remote program will use the CONFIRM keyword.

If the SYNLVL(*NONE) keyword is specified when the program is evoked, the CONFIRM keyword cannot be specified.

An EVOKE keyword must apply for any record for which this keyword applies.

You cannot specify SYNLVL with the TIMER keyword.

Option indicators are valid for this keyword and are required if this keyword is specified on more than one record format in the file.

Figure 5-42 shows how to specify the SYNLVL keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RCD
00020A                                      EVOKE(LIBRARY1/PROGRAM1)
00030A                                      SYNLVL(*CONFIRM)
     A
```

Figure 5-42. Specifying the SYNLVL Keyword

The EVOKE keyword will start PROGRAM1 in LIBRARY1 on the remote system. The SYNLVL keyword will set up a synchronization level that will confirm whether the data was received. When you request a confirmation (specify the CONFIRM keyword), the remote program must acknowledge whether it received the data by sending a positive or negative response.

TEXT (Text)

Use this record- or field-level keyword to supply a text description (or comment) for the record format or field that is used for program documentation.

The format of the keyword is:

```
TEXT('description')
```

The text must be enclosed in apostrophes. If the length of the text is greater than 50 positions, only the first 50 characters are used by the high-level language compiler.

Option indicators are not valid for this keyword.

Figure 5-43 shows how to specify the TEXT keyword at the record and field levels.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R CUSMST                    TEXT('Customer Master Record')
00020A            FLD1           3  0       TEXT('ORDER NUMBER FIELD')
     A
```

Figure 5-43. Specifying the TEXT Keyword

TIMER (Timer)

Use this record-level keyword to specify an interval of time for your program to wait before performing some specified function. To set the timer, your program issues an output operation with the TIMER keyword in effect.

The format of the keyword is:

```
TIMER(HHMMSS | &field-name)
```

HHMMSS
The time interval is a 6-digit value where HH is the number of hours (00 through 99), MM is the number of minutes (00 through 59), and SS is the number of seconds (00 through 59).

&field-name
The time interval parameter is the name of a field that contains the timer value in the form HHMMSS described above. The specified field name must exist in the record format, and the field must be a zoned field with length 6, data type S, usage P, and zero decimal positions.

The following keywords cannot be specified with TIMER:

ALWWRT	ENDGRP	RCVCONFIRM	RCVTRNRND
CANCEL	EVOKE	RCVCANCEL	RECID
CNLINVITE	FAIL	RCVCTLDTA	RQSWRT
CONFIRM	FMH	RCVDETACH	SECURITY
CTLDTA	FMTNAME	RCVENDGRP	SUBDEV
DETACH	FRCDTA	RCVFAIL	SYNLVL
DFREVOKE	INVITE	RCVFMH	VARBUFMGT
EOS	NEGRSP	RCVNEGRSP	VARLEN

TIMER overrides the WAITRCD parameter on the Create ICF File (CRTICFF), Change ICF File (CHGICFF), and Override ICF File (OVRICFF) commands. The WAITRCD parameter value is ignored during the interval that the timer function is in effect.

Option indicators are not valid for this keyword.

Figure 5-44 shows how to specify the TIMER keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD1                      TIMER(002512)
     A          R RCD2                      TIMER(&FIELD1)
     A            FIELD1         6S  P
     A
     A
```

Figure 5-44. Specifying the TIMER Keyword

On an output operation to RCD1, the timer will be set to 0 hours, 25 minutes, and 12 seconds. On an output operation to RCD2, the timer will be set to the value that has been set in FIELD1.

VARBUFMGT (Variable Buffer Management)

Use this record-level keyword to send or receive multiple or partial records using one record format per output operation. On a send operation, you must specify the length of data to be sent using the VARLEN keyword. Otherwise, the length of the record format is used. On a receive operation, the length of data received is the length of the record format.

This keyword has no parameters.

VARBUFMGT is ignored at run time when a CANCEL, EOS, FAIL, NEGRSP, RSPCONFIRM, or RQSWRT keyword is in effect. These keywords must have option indicators when they apply to a record that has the VARBUFMGT keyword specified. If a CANCEL, EOS, FAIL, NEGRSP, or RQSWRT keyword with no option indicator applies to a record for which VARBUFMGT applies, an error message results and the VARBUFMGT keyword is ignored at create time.

Specify at least one data field (usage B or blank) for the user data in the record format.

You cannot specify the VARBUFMGT keyword:

- When the TIMER keyword is used
- On the same record format as the RECID keyword
- On the RECID or INVITE default record formats

Option indicators are not valid for this keyword.

Figure 5-45 shows how to specify the VARBUFMGT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R MULTFMT1
00020A                                      VARBUFMGT
00030A            DATAFLD       32A
00040A          R MULTFMT2
00050A                                      VARLEN(&LENFLD)
00060A                                      VARBUFMGT
00070A            DATAFLD       32A
00080A            LENFLD         5S P
```

Figure 5-45. Specifying the VARBUFMGT Keyword

Suppose `42THIS RECORD WILL NOT FIT INTO ONE BUFFER` was the data to be sent or received. The VARBUFMGT keyword on the first record format sends or receives the first 32 bytes of data. The second record format sends 10 bytes of data. The data length of 10 is set in LENFLD.

VARLEN (Variable-Length User Data)

Use this record-level keyword to indicate that the length of the record sent across the line is variable. The length is specified at run time in the field parameter.

The format of the keyword is:

VARLEN(&field-name)

The &field-name parameter is required and specifies the name of the field that contains the length of the user data to be sent. The field name must exist in the record format and the field must be defined as a zoned field of length 5, data type S, usage P, and zero decimal positions.

The length value set in the parameter field is the length of the user data and does not include indicators. The length value is specified in decimal and is checked at run time. The value should not be greater than the length of the DDS record format. The maximum value depends on the communication type you are using.

VARLEN is valid only on output operations.

VARLEN is ignored at run time when an CANCEL, EOS, FAIL, NEGRSP, RSPCONFIRM, or RQSWRT keyword is in effect. These keywords must have option indicators when they apply to a record that has the VARLEN keyword specified. If a CANCEL, EOS, FAIL, NEGRSP, or RQSWRT keyword with no option indicator applies to a record for which VARLEN applies, an error message is issued and the VARLEN keyword is ignored at creation time.

At least one data field (usage B or blank) for the user data must be specified in the record format.

You cannot specify VARLEN with the TIMER keyword.

Option indicators are not valid for this keyword.

Figure 5-46 shows how to specify the VARLEN keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RCD                        VARLEN(&LENFLD)
     A            DATAFLD    32760A
     A            LENFLD         5S  P
     A
```

Figure 5-46. Specifying the VARLEN Keyword

On an output operation to RCD, the length of the data in DATAFLD that is sent across the line will be the length set in LENFLD.

Appendix A. When to Specify REF and REFFLD

This appendix provides information to help you decide if you want to specify the REF (reference) or REFFLD (referenced field) keyword or both. It also tells you how to specify parameter values for each REF or REFFLD keyword you specify.

Note: You must specify R in position 29 for each field that refers to another field that was previously defined.

Answer the following questions to determine which keyword to use:

- REF or REFFLD or both?

  If all or most of the fields you refer to are defined REF at the file level.

  Specify REFFLD for every field you reference:

  - That is not in the file you specify on the REF keyword.

    or

  - Whose name differs from the name of the field it references. This includes fields that reference fields in the file you are defining.

- Do you need a database file name for each REFFLD keyword you specify?

  The database file name specified on a REFFLD keyword overrides the database file name specified on the REF keyword.

  On the REFFLD keyword, you can specify:

  - *SRC so that the OS/400 program searches the file you are defining for the referenced field. The referenced field must be defined before you define the field that references it.
  - The name of the database file that the OS/400 program is to search through to find the referenced field.

  If you do not specify *SRC or a database file name on the REFFLD keyword, the default is *SRC if the REF keyword is not specified. If the REF keyword is specified, the default is the database file name specified on the REF keyword.

- Is a library name necessary for each database file you specify?

  If the job that will create the file you are defining (perhaps your interactive job) has a library list, and the database file you specified is on the library list, enter only the file name (FILE1). Otherwise, specify the file name qualified by the library name (LIB1/FILE1).

- Do you need a record format name for each REF or REFFLD keyword you specify?

  If the file you reference has only one record format, do not specify a record format name.

  If it has more than one record format, specify a record format name.

See Figure A-1 on page A-2 for an example of reference function specifications.

© Copyright IBM Corp. 1991, 1993

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                     REF(FILE1)     1
00020A          R RECORD1
00030A            FIELD1    R                           1
00040A            FIELD2    R                           1
00050A            FIELD3    R              REFFLD(FLD3)   2
00060A            FIELD4    R              REFFLD(FLD4 FILE2)   3
00070A            FIELD5    R              REFFLD(FLD5 LIB1/FILE3)   4
00080A            FIELD6    R              REFFLD(RECORDB/FLD6 LIB1/FILE4)   5
00090A            FIELD7    R              REFFLD(FIELD6 *SRC)   6
00100A            FIELD8    R              REFFLD(FLD6)   7
00110A          R RECORD2
00120A            FIELD1        20   8
00130A
00140A          R RECORD3
00150A            FIELD1    R              REFFLD(RECORD2/FIELD1 *SRC)   9
00160A
00170A          R RECORD4
00180A            FIELD1    R              REFFLD(FIELD1 *SRC)   10
     A
```

Note: For line 00010, you can also specify library name and record format name. See the REF keyword example.

Figure A-1. Sample Reference Function Specifications

The following refer to Figure A-1:

1 FIELD1 and FIELD2 have the same attributes as FIELD1 and FIELD2 in FILE1.

2 FIELD3 has the same attributes as FLD3 in FILE1.

3 FIELD4 has the same attributes as FLD4 in FILE2.

4 FIELD5 has the same attributes as FLD5 in FILE3 in LIB1.

5 FIELD6 has the same attributes as FLD6 in record format RECORDB in FILE4 in LIB1.

6 FIELD7 has the same attributes as FIELD6 (on the preceding line in this file).

7 FIELD8 has the same attributes as FLD6 in FILE1.

8 FIELD1 in RECORD2 has unique field attributes. (It does not use the reference function; notice that R is not specified in position 29.)

9 FIELD1 in RECORD3 has the same attributes as FIELD1 in RECORD2.

10 FIELD1 in RECORD4 has the same attributes as FIELD1 in RECORD1.

Note: Figure A-1 is not a valid example of any file except an ICF File. Display and printer files must have a location specified for each field. Physical files can have only one record format. The REF and REFFLD keywords are not allowed in logical files.

Appendix B. Examples

This appendix provides examples of data description specifications (DDS) for each type of file discussed in this manual. If you choose you can use the examples in this section with appropriate high-level language programs. An example program, using some of the file examples and illustrating the use of externally described data, is also provided.

The following sections appear in this appendix:

- Database files
  - Field reference file (a physical file used for reference, not data storage)
  - Physical file
  - Logical file
- Device files
  - Display file with help specifications
  - Subfile examples
  - Printer file
  - ICF file
- Example program using the physical, display, and printer file
- Example compiler listing (output from a create-file command)
- IBM Data Description Specifications Debugging Template (reduced)

Database Files

This section contains examples of a field reference file, a physical file, and three logical files.

Field Reference File

The following keywords are important in Figure B-1 on page B-2:

COLHDG
EDTCDE(Z)
REFFLD
REFSHIFT
TEXT

© Copyright IBM Corp. 1991, 1993

The file in Figure B-1 has one record format. It defines all of the fields used in an application and refers to fields only within the field reference file itself.

The following field reference file (MLGREFP) describes all fields used by any program in the application. The other files use the fields in this file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A** FLDREF   MLGREFP    MAILING LIST FIELD REFERENCE FILE
00020A        1 R MLGREFR                    TEXT('Mailing List Field Reference')
00030A            ACTNUM         5  0       COLHDG('Account' 'Number')
00040A                                       EDTCDE(Z)
00050A            ACTTYP         1  0       COLHDG('Acct' 'Type')
00060A                                       TEXT('Acct Type 1=Bus 2=Gvt +
00070A                                       3=Org 4=Sch 5=Pvt 9=Oth')
00080A            NAME          18          COLHDG('Name')
00090A                                       REFSHIFT(X) 4
00100A            ADDR       R  2         2 REFFLD(NAME)
00110A                                       COLHDG('Address') 3
00120A            CITY       R  2         2 REFFLD(NAME)
00130A                                       COLHDG('City')    3
00140A            STATE          2          COLHDG('State')
00150A            ZIP            5  0       COLHDG('ZIP' 'Code')
00160A                                       EDTCDE(X)
00170A            BATNUM         6  0       COLHDG('Batch' 'Number')
00180A                                       EDTCDE(Z)
00190A            TRNTYP         1          COLHDG('Trans' 'Type')
     A
```

Figure B-1 (Part 1 of 2). DDS for a Field Reference File

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00200A                                       TEXT('Trans Type A=Add +
00210A                                       C=Change D=Delete')
00220A            XACTNM     R               REFFLD(ACTNUM)
00230A            XACTTTP    R               REFFLD(ACTTYP)
00240A            XNAME      R               REFFLD(NAME)
00250A            XADDR      R               REFFLD(ADDR)
00260A            XCITY      R               REFFLD(CITY)
00270A            XSTATE     R               REFFLD(STATE)
00280A            XZIP       R               REFFLD(ZIP)
00290A            TRNNUM         5  0       COLHDG('Transaction' 'Number')
00300A                                       EDTCDE(Z)
00310A            MLGLK1         3  0       COLHDG('Lock' 'Control')
00320A                                       TEXT('Control Number Used for +
00330A                                       record locking')
     A
```

Figure B-1 (Part 2 of 2). DDS for a Field Reference File

The following items refer to Figure B-1:

1 Like all physical files, a field reference file has only one record format. The R in position 17 specifies that MLGREFR is the record format name.

2 The Rs in position 29 and REFFLD in positions 45 through 80 specify that the fields ADDR and CITY are to have the same attributes as NAME.

3 Specifying COLHDG for ADDR and CITY overrides the COLHDG attribute for NAME, which otherwise would have been in effect.

4 Specifying REFSHIFT for NAME will cause the keyboard shift specified (X) to be used when this field (NAME) is referred to in a display file.

Physical File with a New Record Format

The REF keyword is important in Figure B-2. This file has one record format. The names of all fields in the record format are specified. Figure B-2 uses fields in a reference file (REF keyword) and uses a keyed-sequence access path.

The following physical file (called CUSMSTP for customer master physical file) describes the fields physically present in the database.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A* SAMPLE PHYSICAL FILE(CUSMSTP)
00030A*
00040A                                        1 REF(MLGREFP)
00050A      2   R CUSMST                        TEXT('Customer Master Record')
00060A            ACTNUM     R   3
00070A            NAME       R   3
00080A            ADDR       R   3
00090A            CITY       R   3
00100A            STATE      R   3
00110A            ZIP        R   3
00120A      4     SEARCH          10  0
00130A      4     CRDLMT           8  2
00140A      5   K ACTNUM
     A
```

Figure B-2. DDS for a Physical File

The following items refer to Figure B-2:

1 At the file level, the REF keyword refers the OS/400 program to the physical file MLGREFP, which is a field reference file for this database.

2 At the record level, R in position 17 specifies that CUSMST is the record format name of the record in this file. (There can only be one record format in a physical file.)

3 At the field level, Rs in position 29 specify that the attributes of fields of the same name in the REF file are to be used as attributes of these fields.

4 The fields SEARCH and CRDLMT are not defined in MLGREFP; therefore, their field attributes are specified here.

5 At the key field level, K in position 17 specifies that ACTNUM is the key field for the file.

Logical File Specifying Multiple Formats and New Keys

The PFILE keyword is important in Figure B-3. Figure B-3 uses new field specifications and provides two record formats. Each record format provides a different view of the associated physical file and uses a key different from the associated physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A* SAMPLE LOGICAL FILE
00030A
00040A          R CUSMST1                  1 PFILE(CUSMSTP)
00050A            ACTNUM
00060A            NAME
00070A            STATE
00080A            LASTNAME           I    3 SST(NAME 8 10)
00090A     2      K ACTNUM
00100A*
00110A          R CUSMST2                  1 PFILE(CUSMSTP)
00120A            ACTNUM
00130A            NAME
00140A            ZIP
00150A          K *NONE
00160A     2      K NAME
     A
```

Figure B-3. DDS for a Logical File Specifying New Keys

The following items refer to Figure B-3:

1 The two record formats (CUSMST1 and CUSMST2) in this logical file are based on the same physical file (CUSMSTP).

2 Record format CUSMST1 has a key different from record format CUSMST2, providing the application program with a different sequence of the same records.

3 The LASTNAME field is a substring of the field NAME. The usage I in position 38 must be specified since this is not a join logical file.

Logical File Specifying a New Record Format

The UNIQUE keyword is important in Figure B-4. Figure B-4 specifies a record format different from the associated physical file.

The following logical file (called CUSMSTL2 for customer master logical file two) uses some of the fields in the physical file CUSMSTP. Another logical file could name all fields, name fields in other physical files as well, concatenate fields, change the order of fields, rename fields, or choose different key fields. In this logical file, the programmer merely omitted some fields from the physical file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A* SAMPLE LOGICAL FILE (CUSMSTL2)
00030A*
00040A                                      1 UNIQUE
00050A          R CUSREC                    2 PFILE(CUSMSTP)
00060A                                        TEXT('Logical File Master Record')
00070A            ACTNUM   3
00080A            NAME     3
00090A            ADDR     3
00100A        4 K ACTNUM
     A
```

Figure B-4. DDS for a Logical File

The following items refer to Figure B-4:

1 The UNIQUE keyword specifies that records with duplicate keys are not allowed within a member of this logical file.

2 The keyword PFILE (required for logical files) specifies CUSMSTP.

3 The field names do not have R specified in position 29 as they would in physical files or in any device file.

4 As in CUSMSTP, the field ACTNUM is treated as a key field.

Join Logical File

The following keywords are important in Figure B-5:

JFILE	JFLD
JOIN	JREF

The following join logical file joins three physical files (PF1, PF2, and PF3) so that an application program can get name, address, and salary information in one input operation, even though the information is stored in three different physical files.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A         1 R JOINREC                2 JFILE(PF1 PF2 PF3)
00020A           J                          JOIN(PF1 PF2)
00030A         3                            JFLD(NAME NAME)
00040A           J                          JOIN(PF2 PF3)
00050A                                      JFLD(NAME NAME)
00060A         4   NAME                   5 JREF(1)
00070A         4   ADDR
00080A         4   PHONE
00090A         4   SALARY
     A
```

Figure B-5. DDS for a Join Logical File

The following items refer to Figure B-5:

1 R identifies the record format. There can be only one record format in a join logical file.

2 The JFILE keyword specifies that PF1, PF2, and PF3 are the physical files on which this join logical file is based. Because it is specified first, PF1 is the primary file. There are two secondary files: PF2 and PF3.

3 J identifies the join specifications. With two secondary files in this join logical file, there must be two join specifications. Each join specification defines how a pair of files is to be joined, as follows:

- JOIN is required when more than two physical files are being joined, and it identifies which two files are being joined in this join specification. In the first join specification, PF1 and PF2 are joined. In the second join specification, PF2 and PF3 are joined.

  Note: Secondary files can be joined to either the primary file or to another secondary file. In this example, PF2 in the second JOIN keyword could be PF1; there would be no difference in the order of records supplied to the program or in performance.

- JFLD identifies which fields are used to link together records from the physical files being joined. In the first join specification, NAME from PF1 links with NAME from PF2. In the second join specification, NAME from PF2 links with NAME from PF3.

4 The field names show which fields are presented to the program. At least one field name is required.

5 The JREF keyword identifies which physical file to search for the field name; in this example, NAME from PF1 is used. Note the use of the direct file number: JREF(1) indicates to use the first file on the JFILE keyword, which is PF1.

Device Files

This section contains examples of a display file, printer file, and an ICF file.

Inquiry Display with Two Record Formats

The following keywords are important in Figure B-6 on page B-8:

CAnn	HELP
CHECK	HLPARA
DSPATR(HI BL)	HLPRCD
DSPATR(UL)	HLPBDY
EDTCDE(Y)	HLPDOC
EDTCDE(2 $)	OVERLAY
ERRMSG	PRINT

Figure B-6 uses +n to specify position.

The following display is defined by the DDS in Figure B-6 on page B-8. It is displayed by output operations to the record formats PROMPT and RESPONSE.

```
                          CUSTOMER FILE ADD/UPDATE

Enter new or existing customer number
Enter A to ADD new Customer

Name      XXXXXXXXXXXXXXXXXXXXX  1
Address   XXXXXXXXXXXXXXXXXXXXX
City      XXXXXXXXXXXXXXXXXXXXX
State     XX       Zip code NNNNN


Credit limit        $NNN,NNN.NN









F3 - End Program & Print Report  F6 - Return to prompt
```

1 If the cursor is positioned in the area with Xs on the NAME field and the Help key is pressed, online help information will appear.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A                                    1  PRINT
00020A                                       CA03(21 'End & Print')
00030A                                       CA06(22 'Display PROMPT')
00040A                                    7  HELP
00050A                                    8  HLPDOC(START GENERAL HELP)
00060A       2  R PROMPT
00070A          H                         9  HLPDOC(LBL1 HELP#1 HELP)
00080A                                    9  HLPARA(2 2 2 50)
00090A                                   1 30'CUSTOMER FILE ADD/UPDATE'
00100A                                   3  2'Enter new or existing customer +
00110A                                       number'
00120A  3         ACTNUM         5  0B    +1CHECK(MF)
00130A 40                                 4  ERRMSG('Customer number not +
00140A                                    4  found' 40)
00150A                                   4  2'Enter A to ADD new Customer'
00160A            ADD            1   I    +1
00170A       5  R RESPONSE                6  OVERLAY
00180A          H                        10  HLPRCD(NAMEHELP)
00190A                                       HLPARA(6 10 6 28)
00200A          H                        11  HLPRCD(ADDHELP)
00210A                                       HLPARA(7 10 9 33)
00220A                                   12  HLPBDY
00230A          H                            HLPRCD(HELPRCD1 HELPFILE)
00240A                                   13  HLPARA(12 18 12 40)
     A
     A
     A
```

Figure B-6 (Part 1 of 2). Display with Two Record Formats

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00250A*
00260A                                    6  2'Name'       14
00270A            NAME          18   B  6 10             14
00280A                                    7  2'Address'    14
00290A            ADDR          18   B  7 10             14
00300A                                    8  2'City'       14
00310A            CITY          18   B  8 10             14
00320A                                    9  2'State'      14
00330A            STATE          2   B  9 10             14
00340A                                    9 19'Zip code'   14
00350A            ZIP            5Y 0B   +1               14
00360A                                  12  2'Credit Limit'
00370A       15  CRDLMT         8Y 2B 12 21EDTCDE(2 $)   DSPATR(HI) 16
00380A                          17   23  2'F3 - End Program & Print Report   +
00390A                                       F6 - Return to prompt'
00400A*
00410A*    HELP RECORDS
00420A*
00430A          R NAMEHELP
00440A                                   2  2'HELP TEXT FOR NAME FIELD'
00450A                                   4  2'ENTER THE CUSTOMER NAME'
00460A          R ADDRHELP
00470A                                   4  2'HELP FOR ADDRESS,CITY,STATE,ZIP'
00480A                                   6  2'ENTER ADDRESS,CITY,STATE & ZIP'
     A
     A
     A
```

Figure B-6 (Part 2 of 2). Display with Two Record Formats

1 The PRINT keyword allows the display station user to print the display at any time by pressing the Print key.

2 An application program would display a prompt by issuing an output operation to the record PROMPT, displaying the constant fields 'CUSTOMER FILE ADD/UPDATE', 'ENTER EXISTING CUSTOMER NUMBER', 'ENTER A TO ADD NEW CUSTOMER', and the named fields ACTNUM and ADD.

3 The CHECK(MF) keyword specifies that when the user types into one position of the field ACTNUM, he must type into all five positions before pressing the Enter key, or an error message is displayed and the keyboard is locked. The user must press the Reset key and reenter through the input field.

4 If record format PROMPT is displayed and your program sets on indicator 40 when an output operation is sent to record format PROMPT, the error message 'Customer number not found' is displayed on the message line (line 24 of the 24-line display unless the MSGLOC keyword is specified). The message is highlighted, field ACTNUM is displayed with its image reversed, and the keyboard is locked until the user presses the Reset key.

5 After the user presses the Enter key, the application program retrieves the desired information from the database and sends an output operation to record format RESPONSE, displaying the fields described in the next paragraphs.

6 The OVERLAY keyword specifies that an output operation to this record format (RESPONSE) does not cause the entire display to be cleared, as it would be by default.

7 The HELP keyword enables the Help key for this display.

8 The HLPDOC keyword specified at the file level identifies the document to be displayed when no help area for the active records contains the current cursor location.

9 The HLPDOC and HLPARA keywords on this H specification specify that the HELP#1 document in the HELP folder will be displayed starting at the LBL1 help label if the Help key is pressed while the cursor is in positions 2 through 50 of line 2.

10 The HLPRCD and HLPARA keywords on this H specification cause the record NAMEHELP to be displayed if the Help key is pressed while the cursor is in positions 10 through 28 of line 6. The record NAMEHELP is defined in this display file; therefore, a file name does not have to be specified on the HLPRCD keyword.

Note: When using application help keywords, the screen is automatically cleared.

11 The HLPRCD and HLPARA keywords on this H specification cause the ADDRHELP record to be displayed if the Help key is pressed while the cursor is in positions 10 through 28 of lines 7, 8 or 9.

12 The HLPBDY keyword limits the help records displayed when the Page key is pressed. If either NAMEHELP or ADDRHELP is displayed when the Help key is pressed, the NAMEHELP and ADDRHELP records are accessible using the Page key. If HELPRCD1 is displayed when the Help key is pressed, the help records from other H specifications are not accessible using the Page key.

13 The HLPRCD and HLPARA keywords on this H specification cause the record HELPRCD1 in the file HELPFILE to be displayed if the Help key is pressed while the cursor is in positions 18 through 40 of lines 12 through 40. This record is in a separate display file called HELPFILE.

14 Five constant fields ('Name', 'Address', 'City', 'State', and 'Zip Code') and five named fields (NAME, ADDRES, CITY, STATE, ZIP) are grouped together on the display by the line/position specifications. NAME, ADDRES, CITY, and STATE default to character-type fields (A in position 35) because no decimal positions are specified. ZIP is a numeric-only, integer field (Y in position 35; 0 in position 37), so its display length equals its specified length.

15 The field CRDLMT is specified with EDTCDE (2 $). EDTCDE(2) is used for monetary amounts and the $ specifies the floating currency symbol.

16 The DSPATR(H1) keyword highlights the field CRDLMT.

17 Instructions to the work station user are generally located at the bottom of the display, just above the message line.

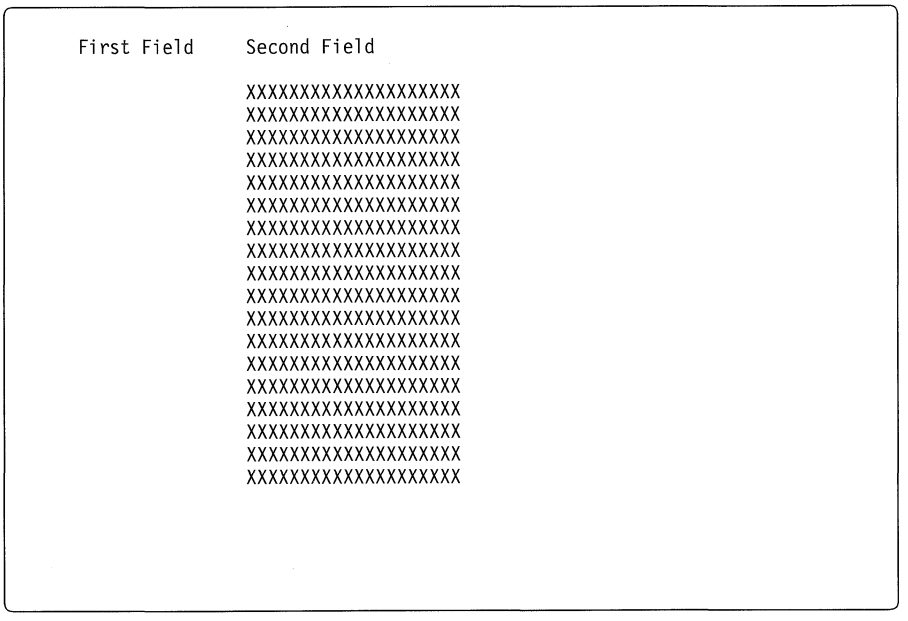
Subfile with SFLPAG Value Equal to SFLSIZ Value

The following keywords are important in Figure B-7:

ROLLDOWN	SFLDSPCTL
ROLLUP	SFLPAG
SFLCLR	SFLSIZ
SFLDSP	

The file in Figure B-7 has one column of subfile records. Constant fields in the subfile control-record format are used as headings for columns of fields in the subfile records.

The following display is defined by the DDS in Figure B-7. It is displayed by an output operation to the subfile control-record format SFLCTL1.




```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* USE OF SUBFILE KEYWORDS
00020A    1     R SFL1                          SFL
00030A    2       FLD1          10  0I  3 11
00040A    2       FLD2          16   0  3 26
00050A
00060A    1     R SFLCTL1                       SFLCTL(SFL1)
00070A                                        SFLSIZ(18)      3
00080A                                        SFLPAG(18)      3
00090A  05                                    SFLDSP          4
00100A  05                                    SFLDSPCTL       4
00110A N05                                    SFLCLR          5
00120A                                        ROLLUP(01) ROLLDOWN(02) 6
00130A                                    1 11'First Field'    7
00140A                                    1 26'Second Field'   7
     A
```

Figure B-7. Subfile with Subfile Size Equal to Subfile Page

The following items refer to Figure B-7 on page B-11:

1. The subfile record format SFL1 and the subfile control-record format SFLCTL1 together define one subfile. The parameter value for the SFLCTL keyword is the name of the subfile record format.

2. Each subfile record is made up of two fields: FLD1 and FLD2. FLD1 is 10 bytes long (11 bytes display length because it defaults to signed numeric); FLD2 is 16 bytes long. FLD1 is an input-only field; FLD2 is an output-only field. Eighteen subfile records would appear on the display, with the first one on line 3 and the last one on line 20. For each subfile record on the display, two fields (FLD1 and FLD2) would appear, with four spaces between FLD1 and FLD2.

3. SFLSIZ and SFLPAG (required keywords) have equal values (18). Therefore one page equals the whole subfile. For all subfiles, the value of the SFLPAG keyword is the number of subfile records displayed at any one time (unless the SFLDROP keyword or variable-length records are used).

4. SFLDSP (a required keyword) and SFLDSPCTL (an optional keyword) are specified with indicator 05. Therefore, when indicator 05 is set on, the subfile and subfile control records can be displayed by an output operation to the subfile control-record format SFLCTL1.

5. SFLCLR (an optional keyword) is specified with option indicator 05 preceded by an N. When indicator 05 is set off, the subfile can be cleared by an output operation to SFLCTL1.

6. ROLLUP (an optional keyword) is specified with response indicator 01, and ROLLDOWN (an optional keyword) is specified with response indicator 02. Note also that the entire subfile equals one page, which means that the whole subfile is displayed at one time. Therefore, when the display station user presses the Page Up key, control passes to the program with indicator 01 on, and when the work station user presses the Page Down key, control passes to the program with indicator 02 on. The program must handle paging, by reading, clearing, rewriting, and redisplaying the subfile. Without ROLLUP and ROLLDOWN specified, the work station user would receive an error message when pressing the Page Up or Page Down key.

7. Two constants ('First Field' and 'Second Field') are displayed when the subfile control-record is displayed (SFLDSPCTL in effect). As specified in this subfile, they act as column headings to the subfile records.

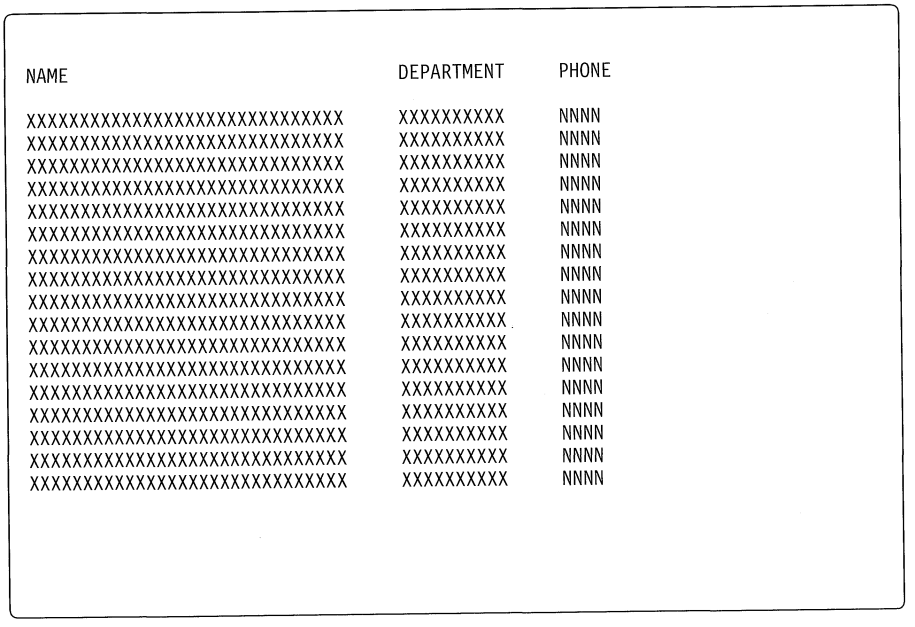
Subfile with Paging by OS/400 Program and High-Level Language Program

The following keywords are important in Figure B-8 on page B-14:

SFLPAG
SFLSIZ

The SFLSIZ value is larger than the SFLPAG value. The subfile is paged by the SFLPAG value.

The following example describes a combined method of paging a subfile that uses system resources efficiently. In some applications, the number of records in the subfile could be quite large. However, the application user may want to view only the first page or two of these records. In this case, it may be faster and more efficient for the application program to build the subfile a page at a time as requested by the user. The OS/400 program handles the paging of records in the subfile. It also returns a Page Up key indicator to the high-level language program when another page should be added to the end of the subfile. The application user can detect no difference between a page up request handled by the OS/400 program and one handled by the high-level language program.



```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R SETSFL                        SFL
00020A  50                                      SFLNXTCHG
00030A            NAME          30   0  5  2
00040A            DEPT          10      5 40
00050A            PHONE          4   0  5 58
00060A          R SETCTL                        SFLCTL(SETSFL)
00070A                                          SFLSIZ(0034)      1
00080A                                          SFLPAG(0017)
00090A  40                                      SFLDSP
00100A  41                                      SFLDSPCTL
00110A  42                                      SFLDLT
00120A  43                                      SFLCLR
00130A  49                                      SFLEND            2
00140A N49                                      ROLLUP(26)
00150A                                          LOCK
00160A                                          OVERLAY
00170A            SETRRN         4S 0H          SFLRCDNBR(CURSOR) 3
00180A                                        3  2'NAME'
00190A                                        3 40'DEPARTMENT'
00200A                                        3 58'PHONE'
     A
```

Figure B-8. Subfile with Paging by OS/400 Program and High-Level Language Program

The following items refer to Figure B-8:

1. The SFLSIZ value must be greater than the SFLPAG value so that the OS/400 program will handle paging within the subfile. A maximum of 9 999 records can be stored in the subfile.

2. The SFLEND keyword can be specified with the ROLLUP keyword. One indicator can be used to option both keywords. The application program turns on the indicator to disable the Page Up key and omit the plus sign (+) on the last subfile page when the last page of the subfile is displayed.

3. The SFLRCDNBR keyword should be specified so the last subfile page can be displayed after it is built by the high-level language program.

   Records in the subfile with changed input fields (modified data tags) will be changed after a new page is added to the subfile by the high-level language program.

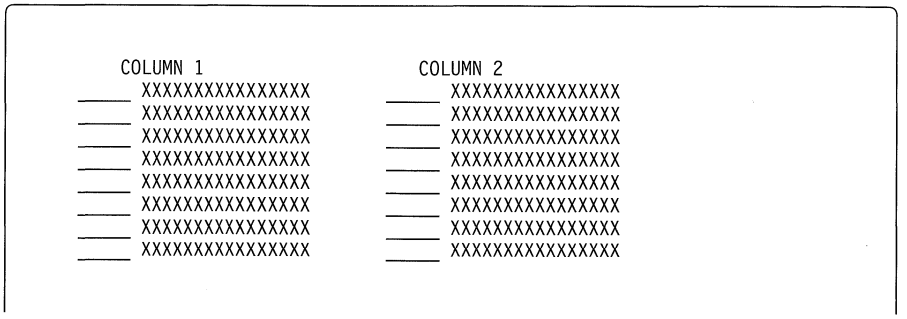
Horizontal Subfile Displayable on Two Display Sizes

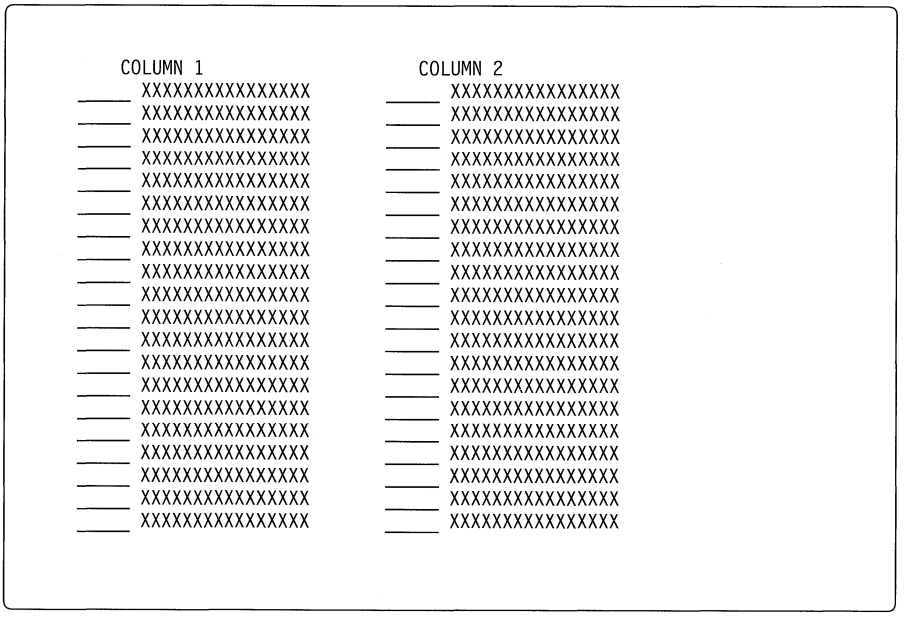
The following keywords are important in Figure B-9 on page B-16:

DSPSIZ
SFLLIN

Subfile records appear in two columns (SFLLIN keyword). The subfile can be displayed on two display sizes (DSPSIZ keyword).

The following two displays show the subfile defined in Figure B-9 on page B-16 as it appears on the 24 x 80 and 27 x 132 display sizes, respectively:





```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A* HORIZONTAL SUBFILE ON TWO DISPLAY SIZES
00020A*
00030A                                    1  DSPSIZ(*DS3 *DS4)
00040A          R SFL1                       SFL
00050A            FLDA          10Y 0I  3 11
00060A            FLDB          16   0  3 23
     A
00070A          R SFLCTL1                    SFLCTL(SFL1)
00080A                                       SFLSIZ(50)
00090A                                    2  SFLPAG(16)
00100A  *DS4                              2  SFLPAG(40)
00110A                                    3  SFLLIN(5)
00120A  *DS4                              3  SFLLIN(5)
     A
00130A  01                                   SFLEND
00140A  02                                   SFLDSP
00150A  03                                   SFLDSPCTL
00160A  04                                   SFLCLR
     A
00170A                                    1 21'COLUMN 1'
     A
00180A                                    1 55'COLUMN 2'
     A
```

Figure B-9. Horizontal Subfile on Two Display Sizes

The following items refer to Figure B-9:

1 There is one keyword at the file level, the keyword DSPSIZ (optional). This keyword has two values, *DS3 and *DS4, which indicate that the primary display size is 24 x 80, and the secondary display size is 27 x 132.

2 The SFLPAG keyword (required), is specified once with a value of 16 and again with a value of 40. The first time it applies to a device with the primary display size (default of *DS3, or 24 x 80); the second time, coded with a condition name of *DS4, it applies to a device with the secondary display size (27 x 132).

3 The SFLLIN keyword causes a subfile to be displayed horizontally. The parameter value specifies the number of spaces between columns of records. In this example, five spaces separate columns of records on both the 24 x 80 display size (*DS3) and the 27 x 132 display size (*DS4). Because *DS3 is the primary display size, it does not need to be specified in positions 9 through 12.

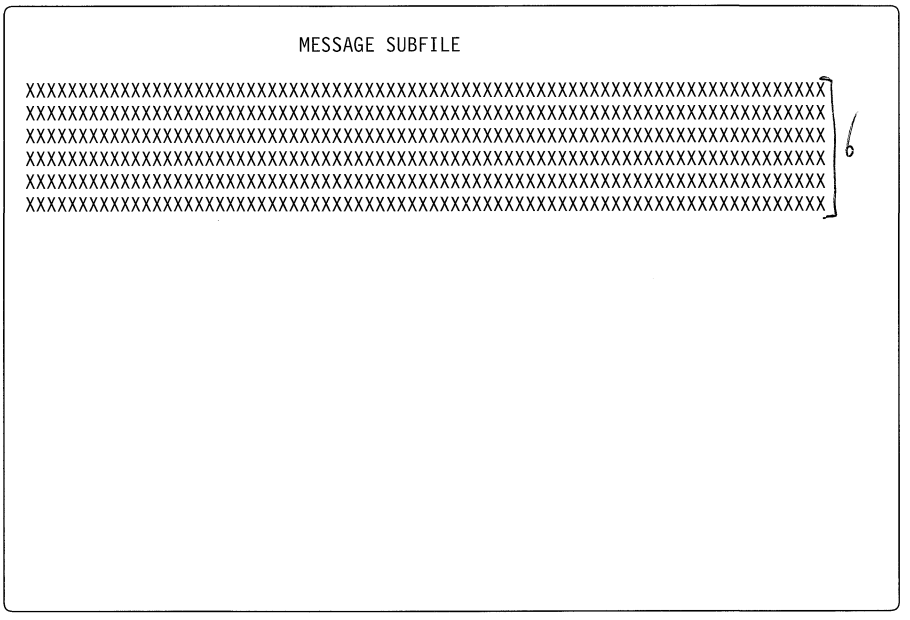
Message Subfile

The following keywords are important in Figure B-10 on page B-17:

SFLMSGKEY
SFLPGMQ
SFLMSGRCD

Records in the subfile are messages from a message file.

The following display shows the message subfile defined in Figure B-10:




```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A* MESSAGE SUBFILE
00030A*
00040A          R SFLR                         SFL                24
00050A                                         SFLMSGRCD(3)       1
00060A            MSGKEY    2                  SFLMSGKEY          2
00070A            PGMQ      2                  SFLPGMQ            2
     A
00080A          R STLCTLR                      SFLCTL(SFLR)
00090A                                         SFLSIZ(12)         3
00100A                                         SFLPAG(6)          3
00110A  01                                     SFLDSP
00120A  02                                     SFLDSPCTL
00130A  03                                     SFLCLR
00140A  04                                     SFLEND             3
     A
     A
00150A                                  1 32'MESSAGE SUBFILE'
     A
```

Figure B-10. Message Subfile

The following items refer to Figure B-10:

1 Specifying the SFLMSGRCD keyword on the subfile record format identifies this subfile as a *message subfile*. The parameter value specified causes the subfile to appear on line 3 of the display.

2 The fields MSGKEY and PGMQ are user-defined names given to the two fields required for the subfile record format for a message subfile. The only specifications allowed for them are their names and the SFLMSGKEY and SFLPGMQ keywords, in the order shown.

This subfile is built by a series of output operations to SFLR that place messages in the subfile as subfile records. Messages are truncated to fit single lines (76 characters or 128 characters, depending on display size), and

second-level help is available. This subfile is displayed by an output operation to SFLCTLR with option indicator 01 set on.

3 This subfile is paged by the OS/400 program when the display station user presses a Page Up or a Page Down key. The SFLEND keyword allows the OS/400 program to display a plus sign whenever the subfile can be paged up.

Printer File

The following keywords are important in Figure B-11:

EDTCDE(Y)	UNDERLINE
EDTCDE(Z)	BARCODE
PAGNBR	CHRSIZ
SKIPB	COLOR
SPACEA	

This printer file uses space and skip keywords instead of line numbers.

The following printer file contains DDS for printing a customer master list.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A* SAMPLE PRINTER FILE
00030A*
00040A                                    1  REF(MLGREFP)
00050A          R HEADER                     TEXT('TWO-LINE HEADING, UNDERLINED')
00060A                                    2  SKIPB(2)
00070A                                2     29'CUSTOMER MASTER FILE'
00080A                                2     75DATE EDTCDE(Y)
00090A                                2     +1TIME
00100A                                2    122'Page'
00110A                                2     +1PAGNBR  EDTCDE(Z)  SPACEA(2)
00120A                                2      2'ACCOUNT  CUSTOMER'
00130A                                2        SPACEA(1)
00140A                                2      2'NUMBER   NAME        +
00150A                                         ADDRESS             +
00160A                                         CITY                +
00170A                                         STATE   ZIP  '
00180A                                    3  UNDERLINE
00190A                                       SPACEA(2)
     A
```

Figure B-11 (Part 1 of 2). Printer File

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00200A          R DETAIL                     TEXT('ONE-LINE RECORD')
00210A                                    4  SPACEA(5)
00220A            ACTNUM    R             4  2CHRSIZ(2 2)
00230A            NAME      R             5 +4COLOR(BLU)
00240A            ADDR      R               +3
00250A            CITY      R               +3
00260A            STATE     R             6 +3BARCODE(CODE30F9 4 *NOHRI *AST)
00270A            ZIP       R               +5
     A
```

Figure B-11 (Part 2 of 2). Printer File

The following items apply to Figure B-11:

1 This printer file refers to the field reference file MLGREFP.

2 When SKIPB(2) is specified at the record level, the printer skips to line 2 before printing the record format (HEADER). Also, line numbers in positions 39 through 41 are not allowed.

3 UNDERLINE is a field-level keyword that causes the constant field preceding it to be underlined on the printout.

4 The CHRSIZ keyword specified here causes the ACTNBR field to print with its height and width expanded by 2.

5 The COLOR keyword causes the NAME field to print in blue if you use a printer that supports color (the 4224 Printer).

6 The BARCODE keyword specified for the STATE field causes the CODE30F9 bar code to print for the STATE field if you use an IPDS printer.

ICF File

The following keywords are important for Figure B-12:

CONFIRM	RECID
DETACH	SYNLVL
EVOKE	EOS
RCVDETACH	RSPCONFIRM
RCVFAIL	ALWWRT
RCVCONFIRM	FAIL
RCVTRNRND	RQSWRT

Figure B-12 contains DDS for transmitting data between the AS/400 system and a remote system or device.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*  SAMPLE ICF FILE
00030A*
00040A  01                              3  EOS
00050A                                  2  RCVTRNRND(01 'TRNRND INDICATION')
00060A                                  2  RCVDETACH(02 'DETACH RECEIVED')
00070A                                  2  RCVCONFIRM(03 'CONFIRM REQUEST')
00080A                                  2  RCVFAIL(04 'FAIL RECEIVED')
00090A
00100A          R CATCHALL
00110A            FLD1         132A
00120A*
00130A          R SNDEVOKE                 EVOKE(&LIBNME/&PGMNME)
00140A     1                               SYNLVL(*CONFIRM) SECURITY(2 PASSWRD)
00150A     1                               CONFIRM
00160A            PGMNME        10A  P
00170A            LIBNME        10A  P
00180A            PASSWRD        8A  P
00190A*
00200A          R HEADER                   RECID(1 'H')
      A
```

Figure B-12 (Part 1 of 2). ICF File

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00210A  09                                  CONFIRM
00220A            ID             1A
00230A         4  PART#          12A
00240A            UNTCST          6S 2
00250A            QTYONORDR       9B 0
00260A            TOTAL           9B 0
00270A*
00280A          R DETAIL                    RECID(1 'D') RECID(1 'E')
00290A  09                                  CONFIRM
00300A            ID             1A
00310A         4  LOC            6A
00320A            QTY            9B 0
00330A*
00340A          R COMMANDS
00350A  05                              5   FALL
00360A  06                              5   ALWWRT
00370A  07                              5   DETACH
00380A  08                              5   RQSWRT
00390A  09                              5   CONFIRM
00400A  10                              5   RSPCONFIRM
      A
```

Figure B-12 (Part 2 of 2). ICF File

The following items apply to Figure B-12:

1 The record format SNDEVOKE causes the program specified in the field PGMNME and the library specified in the field LIBNME to be started on the remote system. It also establishes a synchronization level of *CONFIRM for the transaction and passes the data in the field PASSWRD as security information. The CONFIRM keyword requests that the remote system confirm the start of the program.

2 If the remote program does any of the following:

- Requests to end sending data
- Requests to end the transaction
- Requests to confirm receiving the data
- Sends a FAIL

this sets on one of the following response indicators:

- 01 (the RCVTRNRND keyword)
- 02 (the RCVDETACH keyword)
- 03 (the RCVCONFIRM keyword)
- 04 (the RCVFAIL keyword)

3 The EOS keyword causes the session to end if indicator 01 is on and the program issues an output operation.

4 The AS/400 system sends and receives data in the form of header (record format HEADER) and detail (record format DETAIL) records. If your program is sending, option indicator 09 can be set on (enabling the CONFIRM keyword) to request that the remote system confirms receiving the data.

When receiving, the record selection processing (RECID keyword) determines which record is received. If an H is in position 1, record format HEADER is selected. If a D or E is in position 1, record format DETAIL is selected. If anything else is in position 1, (unexpected record format, application error, or indicators with no data, RCVDETACH, RCVFAIL, and so on) record format CATCHALL is selected.

5 Record format, COMMANDS, is used to request the following communications functions:

- If indicator 05 is on, the FAIL function is performed.
- If indicator 06 is on, the ALWWRT function is performed.
- If indicator 07 is on, the DETACH function is performed.
- If indicator 08 is on, the RQSWRT function is performed.
- If indicator 09 is on, the CONFIRM function is performed.
- If indicator 10 is on, the RSPCONFIRM function is performed.

Example Program Using a Physical, Display, and Printer File

The sample program shown in Figure B-13 on page B-22 illustrates the use of externally described data in a program. This program uses the sample physical file, display file, and printer file given in this appendix.

If you enter the DDS for these files on your system and create them using the Create Physical File (CRTPF) command, the Create Display File (CRTDSPF) command, and the Create Printer File (CRTPRTF) commands, this program allows you to add records to the physical file, display and update the records, and print a report.

The program is written in RPG III. You can enter the RPG III specifications shown in Figure B-13 into a source file on your system and create the program using the Create RPG Program (CRTRPGPGM) command. For more information on RPG III, refer to the *RPG/400* Reference* manual.

File Description Specifications

For the valid entries for a system, refer to the RPG reference manual for that system

Line	Form Type	Filename	I/O/U/C/D	P/S/C/R/T/D/F	E	A/D	F/V/S/N/D/E	Block Length	Record Length	L/R	A/P/I/K	I/X/D/ /R/ or 2	Type of File Organization or Additional Area	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels S/N/E/M	Name of Label Exit	Extent Exit for DAM	Storage Index	File Addition/Unordered	Number of Tracks for Cylinder Overflow	Number of Extents	Tape Rewind	File Condition U1-U8, UC	K	Option	Entry	A/U	R/U/N
020	F	*																													
030	F	* SAMPLE RPG PROGRAM TO TEST APPENDIX B EXAMPLES																													
040	F	*																													
050	F	CUSMSTP	UF		E							K					DISK					A									
1 060	F	CUSDSPF	CF		E		2										WORKSTN														
070	F	CUSPRTF	O		E									10			PRINTER														

Line	Form Type	Control Level (LO-L9, LR, SR, AN/OR)	Indicators And Not	Indicators And Not	Indicators And Not	Factor 1	Operation	Factor 2	Result Field Name	Result Field Length	Result Field Decimal Positions	Result Field Half Adjust (H)	Resulting Indicators Arithmetic Plus	Minus	Zero	Compare 1>2	1<2	1=2	Lookup(Factor 2)is High	Low	Equa	Comments
010	C					START	TAG															
020	C						EXFMT PROMPT															
030	C		N21			ACTNUM	CHAIN CUSMST												30			
040	C					ADD	IFNE 'A'														4	
050	C					*IN30	ANDEQ '1'															
060	C						SETON												40			
070	C		N21	40			GOTO START															
080	C						END															
090	C		N21				EXFMT RESPONSE															
3 100	C		N21	30			WRITE CUSMST														5	
110	C		N21	N30			UPDATE CUSMST															
120	C		N21				GOTO START															
130	C						WRITE HEADER															
140	C					PRINT	TAG															
150	C						READ CUSMST														45	
160	C		N45	10			WRITE HEADER															6
170	C		N45				WRITE DETAIL															
180	C		N45				GOTO PRINT															
190	C						CLOSE *ALL															
200	C						SETON												LR			

RV2F513-0

Figure B-13. Example RPG Program Using a Physical, Display, and Printer File

The following items refer to the RPG coding specification example shown in Figure B-13:

1 These are the file description specifications (F-specs) for the example program. Positions 7 through 14 specify the file names. These file names should be the same file names used when you created the files from the DDS specifications earlier in this appendix.

2 The Es in position 19 indicate that these files are externally described (described in DDS specifications outside of the program rather than inside the program).

3 These are the calculation specifications (C-specs).

4 The first section of the C-specs displays a prompt and retrieves a record from the database.

The TAG op code indicates a label in the program. The EXFMT op code writes the record PROMPT to the display, then reads it when the user presses the Enter key. The CHAIN op code retrieves a record from CUSMST based on the key field ACTNUM. If no record with that key value is found, indicator 30 is turned on. The program continues to prompt until the key value of an existing record is entered in field ACTNUM (indicator 30 off) or an A is entered in field ADD.

5 This section adds a new record or updates an existing record in the database file.

If a new customer is being added (indicator 30 is on), the WRITE op code adds a new record to the physical file. Otherwise, the UPDAT op code updates an existing record. The program continues to prompt for, retrieve, add, and update records in the physical file until F3 is pressed, setting on indicator 21.

6 This section prints the report.

A record is read from the physical file and the DETAIL record is written to the printer file until the end of the physical file is reached (indicator 45 is set on). The HEADER record is written on the first page and then written again on each new page (indicator 10 is on). When all the records have been written, the CLOSE op code closes all the files and SETON LR ends the program.

Compiler Listing Example

Once data description specifications are written, they must be put into a source file. Then, database or device files are created by entering the CL command that starts the data description processor. The CL command can be entered interactively or in a batch job. The data description processor retrieves the data description specifications from the source file designated on the create-file command, validates the specifications, and creates a computer printout with any errors and any referenced specifications.

An example of a data description specifications compiler computer printout follows:

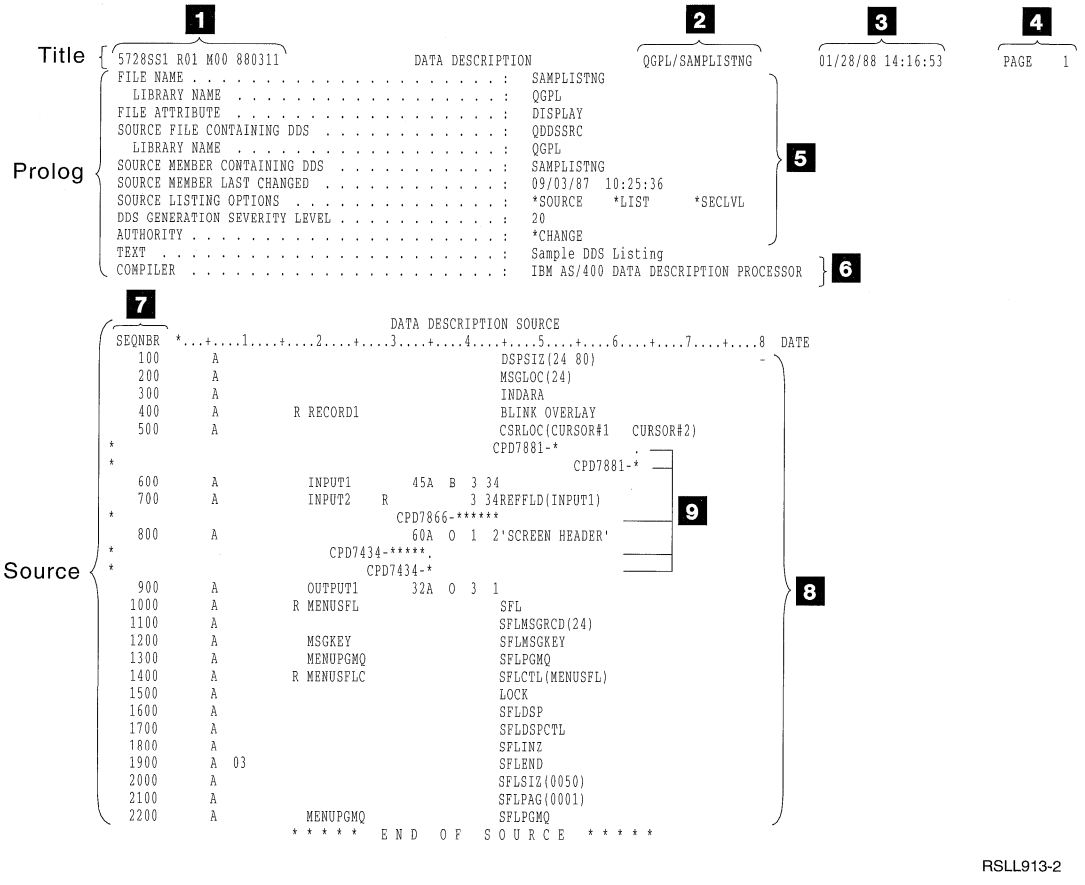
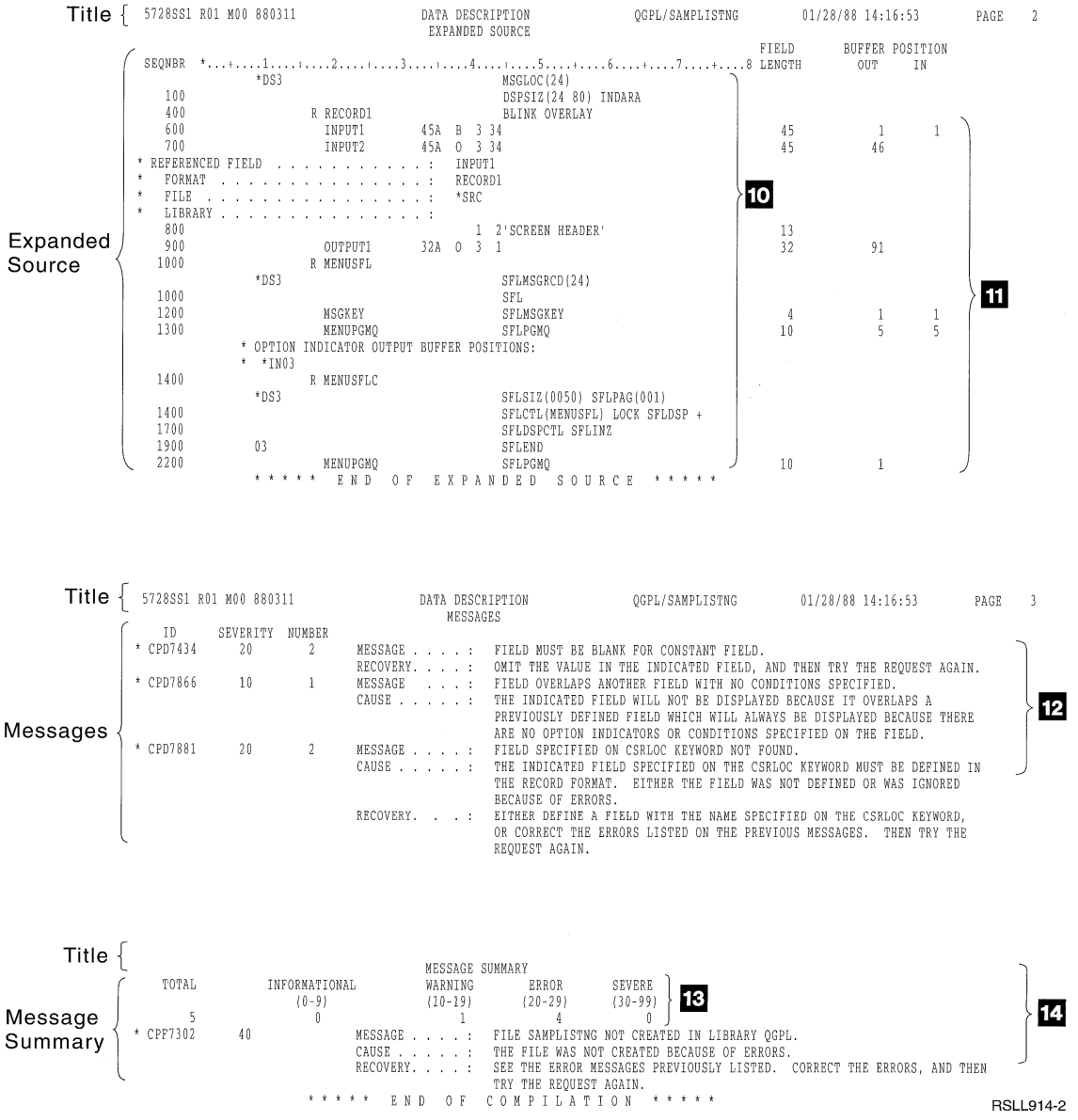


Figure B-14 (Part 1 of 2). DDS Compiler Computer Printout



RSLL914-2

Figure B-14 (Part 2 of 2). DDS Compiler Computer Printout

Title (appears at top of each output page):

The following items apply to Figure B-14.

1 The program number, release modification level, and date of the OS/400 program.

2 The qualified name.

3 The date and time of this run.

4 The page number in the computer printout.

Prolog:

The following items apply to Figure B-14 on page B-24.

5 The type of file and the parameter values specified (or defaults if not specified) on the create-file command.

6 The name of the DDS processor.

Source:

The following items apply to Figure B-14 on page B-24.

7 The sequence numbers of lines (records) in the source. Comments are treated like any other specification line and are given sequence numbers.

8 The source specifications.

9 If an error is found during processing of the DDS and can be traced specifically to a source specification, the error message identifier and an asterisk indicating where the error is are printed immediately following the source specification line. An asterisk is also printed under the sequence number to indicate that the line contains an error message.

Expanded Source:

The following items apply to Figure B-14 on page B-24.

10 Only the valid DDS. This list is what is actually in the file description. No comments or messages are printed. Default values and referenced values are printed for the valid DDS.

11 The length and the buffer (input or output) position of each field.

Messages:

The following items apply to Figure B-14 on page B-24.

12 This section contains a list of all messages (general messages and those already indicated in the source section) encountered during processing of the DDS. For each message, the message identifier, the severity, the number of times the message occurred, and the message text are listed.

Message Summary:

The following items apply to Figure B-14 on page B-24.

13 The number of messages at each severity level.

14 The final completion message.

Debugging Template

A special template is available to help you in interpreting the fields on the DDS compiler computer printout. Figure B-15 shows a reduced Debugging Template.

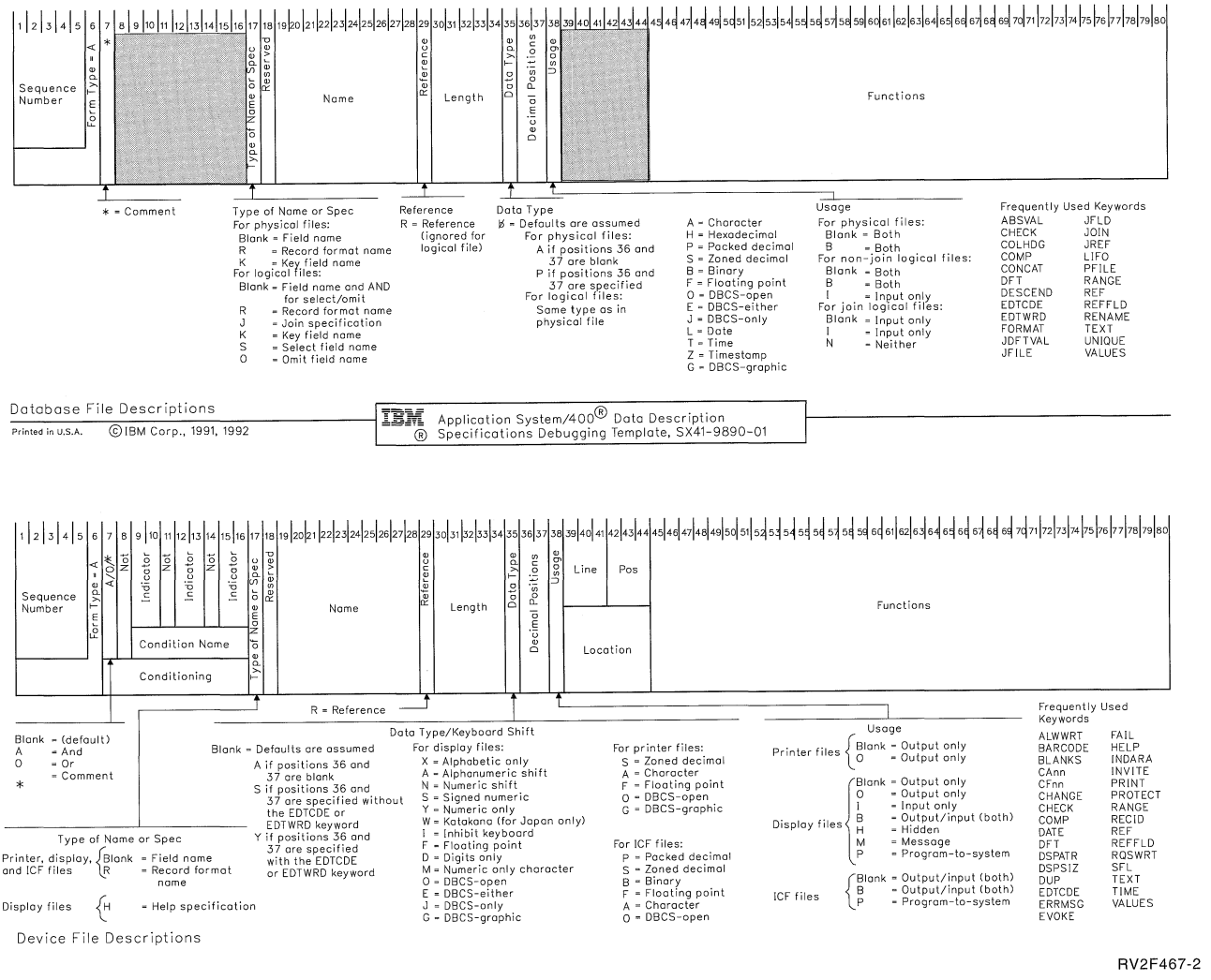


Figure B-15. IBM Data Description Specifications Debugging Template (Reduced)

Appendix C. DDS Keyword and Value Abbreviations

Following are all of the keyword and value abbreviations, listed in alphabetical order, used in data description specifications (DDS):

Figure C-1 (Page 1 of 4). DDS Keywords and Values

A	after
AB	allow blanks
ABS	absolute
ACC	access
ACCEL	accelerator
ALARM	alarm
ALT	alternative
ALW	allow
ARA	area
ATR	attributes
AUTO	automatic
AVAIL	available
B	before
BDY	boundary
BL	blinking field
BLANKS	blanks
BLINK	blinking cursor
BLK	blank
BOX	box
CA	command attention key
CCS	character coded set
CDE	code
CF	command function key
CHC	choice
CHG	change
CHK	check
CHR	character
CLR	clear
CLRL	clear line
CLS	class
CMD	command
CMP	comparison
CNL	cancel
CNT	continued
CNV	conversion
COL	column
COMP	comparison
CON	constant
CONCAT	concatenate
CS	column separator
CSR	cursor
CTL	control
DAT	date
DEC	decimal
DEV	device
DFN	defined
DFR	defer
DFT	default
DLT	delete
DOC	document
DSP	display

© Copyright IBM Corp. 1991, 1993

Figure C-1 (Page 2 of 4). DDS Keywords and Values

DTA	data
DUP	duplicate
DWN	down
DYN	dynamic
EDT	edit
END	end
ENT	entry
EOS	end of session
EQ	equal
ER	end of record (same as end of field)
ERR	error
EXCLD	excluded
FAIL	fail
FCFO	first-changed first-out
FE	field exit
FIFO	first-in first-out
FIX	fixed
FLD	field
FLT	floating point
FMT	format
FNT	font
FRC	force
FULL	full
GDF	graphic data file
GE	greater than or equal to
GPH	graphics
GRP	group
GT	greater than
HDG	heading
HI	high intensity
HLP	help
ID	identifier
IDX	index
IGC	double-byte character set (DBCS)
IND	indicator
INP	input
INZ	initialize
J	join
LC	lowercase
LCK	lock
LE	less than or equal to
LEN	length
LIFO	last in first out
LIN	line
LOC	location
LT	less than
LVL	level
MDT	modified data tag
ME	mandatory enter
MF	mandatory fill
MGT	management
MLT	multiple
MNU	menu
MSG	message
MSK	mask
M10	IBM Modulus 10
M10F	IBM Modulus 10
M11	IBM Modulus 11

Figure C-1 (Page 3 of 4). DDS Keywords and Values

M11F	IBM Modulus 11
NBR	number
ND	nondisplay
NE	not equal to
NEG	negative
NG	not greater than
NL	not less than
NULL	null
NXT	next
OF	off
OID	operator identification
OUT	output
OVR	override
P	physical
PAG	page
PC	position cursor
PCN	precision
PFILE	physical file
PGM	program
PNL	panel
POS	position
PR	protect
PRG	progression
PRT	print or printer
PTH	path
Q	queue
QLTY	quality
RA	record advance
RAB	right-justify with blank fill
RAZ	right-justify with zero fill
RB	right-justify with blank fill
RCD	record
RCV	receive
RECID	record identification
REF	reference
RET	retain
REV	reverse
RI	reverse image
RL	right to left
RLTB	right to left, top to bottom
RMV	remove
RNA	record not active
ROL	roll
RQS	request
RRN	relative record number
RSP	response
RST	restore
RTN	return
RTT	rotate
RZ	right-justify with zero fill
SCH	search
SEG	segment
SEP	separator
SEQ	sequence
SFL	subfile
SIZ	size
SKIPA	skip after
SKIPB	skip before

Figure C-1 (Page 4 of 4). DDS Keywords and Values

SLNO	starting line number
SLT	select
SNG	single
SP	select by light pen
SPACEA	space after
SPACEB	space before
SST	substring
STS	status
SW	switch
SYN	sync
SYS	system
TIM	time
TITLE	title
TRNRND	turn around
TRNTBL	translation table
TXT	text
TYP	type
UL	underline
UNAVAIL	unavailable
USR	user
VAL	values
VAR	variable
VLD	valid
VN	validate name
VNE	validate name extended
WDW	window
WRD	word
WRT	write

Appendix D. DDS for 3270 Remote Attachment

The 3270 Remote Attachment feature allows a 3270 SNA controller or a 3274 emulating device to be attached to an AS/400 system.

Some applications for the 3270 that use data description specifications (DDS) may require programming changes on an AS/400 system. An operator at a 3277, 3278, or 3279 data entry keyboard can use most features and functions as a similarly configured and authorized 5250 work station. The attached unit has the same capabilities and limitations as any remote 5251, with the following exceptions:

- No display attributes are seen on either a 3278 or a 3279 when the field is defined as a nondisplay field.
- Numeric-only fields that are used for negative numbers act differently on the 3270 than on the 5250. On the 3270, if the operator enters a negative number in the field, the sign occupies the first position of the field followed by the number. This causes the maximum size of the field to decrease by 1 and should be considered when designing the displays and fields.
- Results that cannot be predicted can occur when data is entered in an input field that is involved in a page command, if any lines involved in the page are not of the same type field attributes and location.
- The following DDS keywords are ignored by 3270 support:
  - AUTO (RA) and CHECK (ER)
  - BLINK (cursor blink is controlled by keyboard for 3270)
  - CHRID
  - CHANGE
  - LOWER or CHECK (LC) (lowercase is controlled by switch on display for 3270)
  - CHECK (RL and RLTB)
  - DSPSIZ (other than 24 x 80)
  - LOCK
  - MSGLOC (set to row 24)

    Note: For display devices configured as a 3278 Model 4, the MSGLOC keyword is set to display messages on row 43.
- The following DDS display attributes are the only ones valid for 3270 Remote Attachment (except for 3277):

  DSPATR RI used for 3278 and 3279
  CS used for 3278 and 3279 (changed to UL)
  UL used for 3278 and 3279
  BL used for 3278 (determines color on 3279)
- During a write operation to the error line, the Enter key is defined as a Reset key and cannot be mapped to any other function.

  In addition, if a write operation is requested by the user application to display an error message, a read operation should immediately follow the write operation for the remote 3270 display to allow resetting of the error message. A

© Copyright IBM Corp. 1991, 1993

read operation following a write operation of an error message should be used by call applications regardless of the type of target display. If a read operation does not immediately follow the write error message requested by the user application, remote 3270 displays may overlay the error message before it can be read by the user.

Appendix E. Double-Byte Character Set Considerations

This appendix describes double-byte character set (DBCS) considerations for each of the following DDS file types:

- Database (physical and logical) files
- Display files
- Printer files
- ICF files

The functions described in this appendix are supported on both DBCS and non-DBCS systems.

General Considerations

The following sections describe the general considerations for positional entries, keyword entries, DBCS literals, and the DDS computer printouts.

Positional Entries

Positional entries are adapted so that you can define data fields that contain DBCS data. The entries are adapted as described in the following sections.

DBCS data is either **bracketed** or **graphic**. Bracketed-DBCS data begins with a shift-out character and ends with a shift-in character. DBCS-graphic data contains only DBCS data and does not contain shift-out and shift-in characters. The term DBCS refers to both bracketed and graphic DBCS data.

Length (Positions 30 through 34)

Consider the effects of the following when determining the length of a DBCS field.

For bracketed-DBCS fields:

- Positions occupied by double-byte characters as compared to the positions occupied by alphanumeric characters
- Shift-control characters
- Keyboard shift (if any)

For graphic-DBCS fields:

- Length is specified as the number of DBCS characters with no shift-control characters

Data Type (Position 35)

Use the following DBCS data types to identify DBCS fields.

Bracketed DBCS:

J (Only)	Fields can contain only DBCS data.
E (Either)	Fields can contain DBCS or alphanumeric data.
O (Open)	Fields can contain both DBCS and alphanumeric data.

Graphic DBCS:

© Copyright IBM Corp. 1991, 1993

G (Graphic) Fields can contain only DBCS data with no shift-control characters.

Data type O is allowed in all types of files. Data types J and E are allowed only in database and display files. Data type G is allowed in database, display, and printer files.

Keyword Entries (Positions 45 through 80)

When you work with DBCS data, and specify certain DDS keywords, you can:

- Specify alternative ways to enter data through display files
- Change input- and output-capable alphanumeric data fields to DBCS data fields
- Specify the special features of the DBCS printer

Use the following keywords to perform these functions:

CHRSIZ (Character Size)
Specify this keyword for printer files to be printed on the 5553 printers. CHRSIZ can expand printed characters to twice their normal size (width and height).

DFNLIN (Define Line)
Specify this keyword for printer files. DFNLIN draws horizontal or vertical lines.

IGCALTTYP (DBCS Alternative Data Type)
Specify this keyword for display and printer files. IGCALTTYP changes the data type of character input- and output-capable fields from A to O (open) if IGCDTA(*YES) is specified on the command used to create the file.

IGCANKCNV (Alphanumeric-to-DBCS Conversion)
Specify this keyword for printer files (for Japanese only). IGCANKCNV converts alphanumeric characters to equivalent DBCS characters so printed alphanumeric characters have the same appearance as printed DBCS characters.

IGCCDEFNT (DBCS Coded Font)
Specify this keyword for printer files. IGCCDEFNT allows you to specify the DBCS coded font for printing a named or constant field.

IGCCNV (DBCS Conversion)
Specify this keyword for display files (for Japanese use only). IGCCNV allows you to use DBCS conversion, which is an alternative to directly typing in DBCS characters from a keyboard.

IGCCHRRTT (DBCS Character Rotation)
Specify this keyword for printer files to be printed on the 5553 printers. IGCCHRRTT rotates each DBCS character 90 degrees counterclockwise before printing. By rotating characters, the printouts can be read vertically.

DBCS Character Strings

You can use bracketed-DBCS character strings in DDS files for text-related keywords, such as TEXT and COLHDG, and both bracketed and DBCS-graphic character strings as parameters on the COMP, DFT, RANGE, and VALUES keywords.

Entering Bracketed-DBCS Character Strings

Enter bracketed-DBCS character strings as follows:

1. Begin the character string with an apostrophe (').
2. Type a shift-out character.
3. Type the DBCS text.
4. Type a shift-in character.
5. End the character string with an apostrophe (').

For example, to type the DBCS literal ABC, enter the following, where 0_E represents the shift-out character and 0_F represents the shift-in character:

'0_EABC0_F'

Entering DBCS-Graphic Character Strings

Enter DBCS-graphic character strings as follows:

1. Type a G to indicate that the string contains DBCS-graphic data.
2. Begin the character string with an apostrophe (').
3. Type a shift-out character.
4. Type the DBCS text.
5. Type a shift-in character.
6. End the character string with an apostrophe (').

For example, to type the DBCS literal ABC, enter the following, where 0_E represents the shift-out character and 0_F represents the shift-in character:

G'0_EABC0_F'

Considerations for Using DBCS Character Strings

Consider the following information when you use DBCS character strings:

- Do not specify DBCS character strings for those DDS keywords that are dependent on data type, and for which you did not specify a DBCS data type (data type O, J, E, or G).
- When the source file is defined as DBCS, DDS scans all character strings as DBCS character strings and considers all data between the shift-control characters to be part of the character string.

  Remember to include both shift-control characters. If the shift-in character indicating the end of the DBCS string is missing, the system considers the remainder of the record, including the ending apostrophe, to be part of the character string.
- When the source file is alphanumeric, DDS does not check the character string to make sure that you have included only DBCS characters between the shift-control characters.
- When the source file is alphanumeric, DDS identifies DBCS character strings as alphanumeric literals.

- You may refer to a field that contains a DBCS character string from another file, using the reference function. DDS copies the attributes of the field containing the DBCS character string (the referenced field) to the field you are defining. If the file containing the referenced field is DBCS and the file you are defining is alphanumeric, DDS does not check the character string to make sure it is a valid DBCS character string. If the file containing the referenced field is alphanumeric and the file you are defining is DBCS, DDS checks the character string to make sure it is a valid DBCS character string.

DDS Computer Printouts

DDS computer printouts are printed as DBCS output in each of these instances:

- The source file is DBCS.
- DBCS character strings were added to the source file as a result of a reference operation.

Database Files

This section describes the DBCS considerations for the positional entries, and keyword entries for physical and logical files.

Positional Entry Considerations

The following section describes, by position, DDS for describing database files. Positions not mentioned have no special considerations for DBCS.

Length (Positions 30 through 34)

The length of a field containing bracketed-DBCS data can range from 4 through 32 766 bytes (4 through 32 740 bytes if the field is variable length). The length of a DBCS-graphic field can range from 1 through 16 383 characters (1 through 16 370 characters if the field is variable length).

When determining the length of a field containing DBCS data, consider the following:

- Each DBCS character is 2 bytes long.
- For DBCS-graphic fields, the length of the field is specified in number of DBCS characters.
- Include both shift-control characters in the length of the field for fields with a data type of J, E, or O. Together, these characters are 2 bytes long.
- Fields specified with the J or E data types must have an even length.

For example, a bracketed-DBCS field that contains up to 3 DBCS characters, 1 shift-in character, and 1 shift-out character, has 8 bytes of data:

```
(3 characters x 2 bytes) + (shift-out + shift-in) = 8
```

A DBCS-graphic field that contains up to 3 DBCS characters has 6 bytes of data:

```
(3 characters x 2 bytes) = 6
```

Data Type (Position 35)

You can use one of the following DBCS data types:

J (Only)	Fields can contain only DBCS data.
E (Either)	Fields can contain either DBCS or alphanumeric data.
O (Open)	Fields can contain both DBCS and alphanumeric data. Distinguish DBCS data from alphanumeric data with shift-control characters.
G (Graphic)	Fields can contain only DBCS data with no shift-control characters.

Decimal (Positions 36 and 37)

Leave these positions blank when using DBCS data.

Keyword Considerations

Do not specify DDS keywords intended for use with numeric data for fields containing DBCS data. The system treats DBCS data the same as character data, and, therefore, cannot perform arithmetic operations on it.

For additional information on the keywords for database files, refer to the keyword descriptions in Chapter 2, "Physical and Logical Files."

Do not use the following DDS keywords with DBCS data fields (the data type specified in position 35 is O, J, E, or G):

ABSVAL	CHECK(VNE)	REFSHIFT
ALTSEQ	DATFMT	SIGNED
CHECK(M10)	DATSEP	SST
CHECK(M10F)	DIGIT	TIMFMT
CHECK(M11)	EDTCDE	TIMSEP
CHECK(M11F)	EDTWRD	TRNTBL
CHECK(VN)	FLTPCN	ZONE

Notes:

1. The SST keyword is allowed on fields with a data type of G.
2. The REFSHIFT keyword is allowed on fields with a data type of O, J, or E.

The CONCAT keyword can be used as described below.

CONCAT (Concatenate): Use this field-level keyword when you want to combine two or more fields from the physical file record format into one field in the logical file record format you are describing. The name of this concatenated field must appear in positions 19 through 28. Specify the physical file field names in the order in which you want them to be concatenated, and separate them by blanks.

The following rules and restrictions apply:

- The OS/400 program assigns the length of the concatenated field as the sum of the lengths (digits and characters) of the fields that are included in the concatenation.

  Note: For fields with data type J, the shift-out and shift-in pairs between the concatenated fields are removed from the resulting field. If the resulting data type is hexadecimal, the shift-out and shift-in pairs are eliminated for DBCS fields that precede the first hexadecimal fields.

- A DBCS-graphic field can be concatenated only with another graphic data type field.
- The OS/400 program assigns the data type based on the data types of the fields that are being concatenated. When bracketed-DBCS fields are included in a concatenation, the general rules are:
  - If the concatenation contains one or more hexadecimal (H) fields, the resulting data type is hexadecimal (H).
  - If all fields in the concatenation are DBCS-only (J), the resulting data type is DBCS-only (J).
  - If the concatenation contains one or more DBCS (O, E, J) fields, but no hexadecimal fields, the resulting data type is DBCS-open (O).
- The OS/400 program assigns the field to be fixed length or variable length based on the fields that are being concatenated. The general rules are:
  - Concatenation of a variable length field to either a fixed length field or another variable length field results in a variable length field.
  - Concatenation of a fixed length field to a fixed length field results in a fixed length field unless the VARLEN keyword is also specified on the same field as the CONCAT keyword.
- The maximum length of a concatenated field varies depending on the data type of the concatenated field and the length of the fields being concatenated. If the concatenated field is zoned decimal (S), its total length cannot exceed 31 bytes; if it is character (A) or DBCS(O, J), its total length cannot exceed 32 766 bytes. If the concatenated field is variable length, its total length cannot exceed 32 740 bytes (32 739 if the field also allows the null value).

  If the concatenated field is a DBCS-graphic (G) field, its total length cannot exceed 16 383 characters. If the concatenated field is variable length, its total length cannot exceed 16 370 characters.
- In join logical files, the fields to be concatenated must be from the same physical file. The first field specified on the CONCAT keyword identifies which physical file is used. The first field must, therefore, be unique among the physical files on which the logical file is based, or you must also specify the JREF keyword to specify which physical file to use.
- When one or more of the fields being concatenated are DBCS fields, none of the fields on the CONCAT keyword can be specified as a key, select, or omit field unless the field name is also specified in positions 19 through 28 or on a RENAME or CONCAT keyword specified before the DBCS concatenation.
- The usage of a concatenated field must be I (input only).
- REFSHIFT cannot be specified on a concatenated field that has been assigned a data type of O or J.

  Figure E-1 shows how to specify the CONCAT keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD 1                    PFILE(PF1)
     A            FLD1                 I      CONCAT(PFLD1 PFLD2)
     A            FLD2                 I      CONCAT(PFLD1 PFLD2 PFLD3)
     A            FLD3                 I      CONCAT(PFLD4 PFLD5)
     A
```

Figure E-1. Specifying the CONCAT Keyword for Database Files

In Figure E-1, if the fields from PF1 are:

- PFLD1 with data type J
- PFLD2 with data type J
- PFLD3 with data type E
- PFLD4 and PFLD5 with data type G

Then the resulting fields are:

- FLD1 with data type J
- FLD2 with data type O
- FLD3 with data type G

Additional Considerations for Describing Database Files

Consider the following when describing a database file that contains DBCS fields:

- If you describe DBCS fields in the DDS, the system treats the file as a DBCS file. You do not have to specify IGCDTA(*YES) on the file creation command to identify the file as DBCS.
- The data type of a field in a physical file may be changed as follows when you refer to that field in a logical file:

Physical File Data Type	Logical File Data Type
J	J, O, E, H, G
O	O, H
E	O, E, H
A	A, O, E, H
H	J, O, E, A, H
G	G, O, J, E

  Note: When the physical file data type is character (A) or hexadecimal (H), and the logical file data type is DBCS-only (J) or DBCS-either (E), the physical file field length (columns 30 through 34) must be an even number greater than or equal to 4.

- DDS treats DBCS key fields as character fields (the data type specified in position 35 is O).
- DDS uses the EBCDIC collating sequence to sort DBCS data.
- Any key field sequencing keywords that can be used with character fields can be used with DBCS fields, except the following keywords:

  ALTSEQ
  DIGIT
  ZONE

- Use bracketed-DBCS data anywhere that comments and character strings are allowed. See "DBCS Character Strings" on page E-3 for instructions on using DBCS character strings.

- Any bracketed-DBCS field except a field with data type J can be compared with a character field (data type A).
- A DBCS-graphic field can be compared only with another graphic field.
- The following validity checking keywords may be specified on DBCS fields:

  COMP
  RANGE
  VALUES
- When specifying the VARLEN keyword in a physical file, the minimum allowed length for the allocated length is 4 for a bracketed-DBCS field. The minimum allowed length for the allocated length is 1 for a DBCS-graphic field.

Display Files

This section describes the DBCS considerations for the positional entries and keyword entries for display files.

Positional Entry Considerations

The following section describes, by position, DDS for describing display files. Positions not mentioned have no special considerations for DBCS.

Length (Positions 30 through 34)

Specify the length of the field in these positions. The length of a field containing bracketed-DBCS data can range from 4 through 32 763 bytes. The length of a DBCS-graphic field can range from 1 through 16 381 characters.

When determining the length of a DBCS field, consider the following:

- Each DBCS character is 2 bytes long.
- For DBCS-graphic fields, the length of the field is specified in number of DBCS characters.
- Include both shift-control characters in the length of the field for fields with a data type of J, E, or O. Together, these characters are 2 bytes long.
- Fields specified with the J or E data type or keyboard shift must have an even length.

For example, a bracketed-DBCS field that contains up to 3 DBCS characters, 1 shift-in character, and 1 shift-out character, has 8 bytes of data:

```
(3 characters x 2 bytes) + (shift-out + shift-in) = 8
```

A DBCS-graphic field that contains up to 3 DBCS characters has 6 bytes of data:

```
(3 characters x 2 bytes) = 6
```

Data Type (Position 35)

Specify the data type in this position by typing one of the following:

J (Only) Type J to specify this field as a DBCS-only field.

The display station automatically inserts shift-control characters in fields specified with this data type.

If you specify J, you must specify an even number for the field length (positions 30 through 34).

E (Either) Type E to specify this field as a DBCS-either field.

You can type either DBCS or alphanumeric characters in the field. The type of data typed in the first position of the field determines the type of data that can be typed in the remainder of the field. If the field is empty, the system assumes that alphanumeric data will be typed in. To change the field so DBCS data can be typed, position the cursor in the field and put the keyboard in DBCS mode.

If the field contains DBCS data, the display station automatically inserts shift-control characters.

If you specify E, you must specify an even number for the field length (positions 30 through 34).

O (Open) Type O to specify this field as a DBCS-open field. You can type both DBCS and alphanumeric characters in this field. Use shift-control characters to distinguish DBCS data from alphanumeric data.

If the field contains DBCS data, the system does not ensure that the data is enclosed between shift-control characters.

If you specify O, you can specify either an even or an odd number for the field length (positions 30 through 34).

G (Graphic) Type G to specify this field as a DBCS-graphic field. Data typed in this field does not contain shift-control characters.

If you specify G, you must specify the number of DBCS characters for the field length (positions 30 through 34).

Decimal Positions (Positions 36 and 37)

Leave these positions blank when using DBCS data.

Keyword Considerations

Do not use the following DDS keywords with DBCS data fields (the data type specified in position 35 is J, E, O, or G):

AUTO(RAZ)	CHECK(VN)	FLTFIXDEC
BLKFOLD	CHECK(VNE)	FLTPCN
CHECK(M10)	CHRID	MSGCON
CHECK(M10F)	DATE	REFSHIFT
CHECK(M11)	DLTEDT	SFLMSGKEY
CHECK(M11F)	DSPATR(OID)	SFLPGMQ
CHECK(RL)	DSPATR(SP)	SFLRCDNBR
CHECK(RLTB)	EDTCDE	SFLROLVAL
CHECK(RZ)	EDTWRD	TIME

Do not use the CHECK(LC) and LOWER keywords on DBCS-only fields (J specified in position 35).

Do not use the IGCALTTYP, IGCANKCNV, CHECK(LC), and LOWER keywords on DBCS-graphic fields (G specified in position 35).

For additional information on the keywords for display files, refer to the keyword descriptions in Chapter 3, "Display Files."

The following DDS keywords may be used in files containing DBCS data only when the function indicated by the keyword is available on the display device or with the type of data used. However, DDS does not apply record- and file-level keywords to DBCS fields.

CHECK(RL)	DSPSIZ(*DS4)	ERASEINP(*ALL)
CHECK(RLTB)	DSPATR(SP)	MDTOFF(*ALL)
COLOR	DSPMOD	

The IGCALTTYP and IGCCNV keywords also can be used as described in "IGCALTTYP (Alternative Data Type)" and "IGCCNV (DBCS Conversion)" on page E-11.

IGCALTTYP (Alternative Data Type)

Specify this field-level keyword to change input- and output-capable alphanumeric character fields to DBCS fields with data type O.

This keyword has no parameters.

Put the keyword into effect by specifying IGCDTA(*YES) on the CRTDSPF, CHGDSPF, and OVRDSPF commands. Fields specified with this keyword are DBCS fields when you specify IGCDTA(*YES) and are alphanumeric character fields when you specify IGCDTA(*NO). For example, you could create the file by specifying IGCDTA(*NO) on the CRTDSPF command. When using the file to display DBCS data, override the file with the OVRDSPF command, specifying IGCDTA(*YES). To override the display file IGCDSPF, type:

```
OVRDSPF FILE(IGCLIB/IGCDSPF) IGCDTA(*YES)
```

Consider the following, when using the IGCALTTYP keyword:

- Specify this keyword only for input- and output-capable fields whose keyboard shift type is A, N, X, W, or I. Do not specify this keyword for DBCS fields.

  Note: Fields specified with IGCDTA(*YES) are recognized as alphanumeric OPEN data type fields (O) when keyboard shift type is defined as N, X, W, or I.

- The following keywords are not allowed with the IGCALTTYP keyword:

  AUTO(RAZ)
  BLKFOLD
  CHECK(M10 M11 M10F M11F RL RZ VN VNE)
  CMP(EQ GE GT LE LT NE NG NL)
  COMP(EQ GE GT LE LT NE NG NL)
  DUP
  RANGE
  VALUES

Option indicators are not allowed with IGCALTTYP.

Figure E-2 on page E-11 shows how to specify the IGCALTTYP keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A          R RECORD
00020A            FLDA          79A  I 23  2IGCALTTYP
     A
```

Figure E-2. Specifying the IGCALTTYP Keyword for Display Files

When the IGCALTTYP keyword is put into effect, FLDA can contain DBCS data.

IGCCNV (DBCS Conversion)

This file-level keyword lets you use DBCS conversion, an alternative to directly typing in DBCS characters from a keyboard, in display files.

The format for the keyword is:

```
IGCCNV(CFnn line-number)
```

The first parameter, CFnn, identifies which command function key, when pressed, begins and ends the conversion function. Specify any CF key (CF01 through CF24) as the parameter value. Do not specify a CF key that has already been assigned a function.

The second parameter, line-number, identifies where (on the display) the system should place the conversion prompt line. When you type the alphanumeric word to be converted on the conversion prompt line, the system displays related DBCS words. The prompt line requires an entire display line and looks like this:

```
_ _____  _ XXXXXXXXXXXXXXXXXXXXX
```

The underlined fields (_) are input fields, in which you type the word to be converted and specify the type of conversion to be performed.

The field indicated with XXXX is an output field, in which the system displays DBCS words related to the alphanumeric entry to be converted.

The prompt line can be placed anywhere on the display, as long as it does not overlap other displayed records that contain input fields.

Consider the following when using the IGCCNV keyword:

- Use this keyword only with files displayed on DBCS display stations.
- At least one field in the file must be an input-capable DBCS field, or an input-capable field specified with the IGCALTTYP keyword.
- Avoid using the IGCCNV keyword with the CHECK(ME) keyword. Use of DBCS conversion with a mandatory entry field causes operational problems.
- Avoid using the IGCCNV keyword with field validation keywords (CHECK, CMP, RANGE, and VALUES). Using this keyword causes DBCS conversion to work improperly.
- You must define the file for a 24 x 80 display.
- Do not display the DBCS conversion format over a format that uses the USRDFN (user-defined) keyword.
- Option indicators are not allowed with this keyword.

Figure E-3 on page E-12 shows how to specify the IGCCNV keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00100A*
00101A*
00102A                                        IGCCNV(CF24 24)
00103A          R MENU
     A
```

Figure E-3. Specifying the IGCCNV Keyword for Display Files

A user can press the F24 key to begin and end DBCS conversion. Conversion may be used on all input-capable DBCS fields. The conversion format is displayed on line 24.

Additional Considerations for Describing Display Files

Consider the following when describing a display file that contains DBCS data:

- Specify IGCDTA(*YES) on the CRTDSPF command when DBCS data is present in the file, but not indicated in DDS. For example, specify IGCDTA(*YES) if the file sends messages that are DBCS (DDS keyword MSGCON).
- Prevent users from using display files to insert alphanumeric data into DBCS database files by specifying the keyboard shift for a field in a field reference file rather than in a display file. Users cannot type alphanumeric data in input-capable fields of DBCS display files and, therefore, cannot type alphanumeric data into the database file.

  Specify data type J or G in a database field reference file and R in position 29 of the associated display file.

  Use data type J or G for all fields in a field reference file to reduce the possibility of incorrectly setting the default keyboard shift to O (open).
- Describe fields in the file as DBCS fields to cause the system to consider the file to be DBCS even if you do not specify IGCDTA(*YES) on the CRTDSPF command.
- The system displays the DBCS data that does not fit on one display line onto the next display line with the following effects:
  - DDS sends a warning message stating that it split DBCS characters for constant and initialized fields containing DBCS data.
  - DDS sends a warning message stating that it split DBCS characters if you specified the J, E, or G data type.
  - DDS does not send a warning message stating that it split DBCS characters if you specified the O data type. DDS warns you of the potential for this problem when the file is created.
  - The second display line of a continued field might not make sense if the system must split a DBCS character in order to continue the line.
- Text with bracketed-DBCS characters can be used anywhere that comments and character strings are allowed.
- Consider the following when specifying subfiles:

- Use the SFLMSG keyword to create DBCS messages by typing DBCS data for the character string in the message.

  Check the length of the message. The space available to display it must be long enough to contain the message. DDS warns you when a display field might be truncated. However, the field might be truncated in the middle of a DBCS character, and the data displayed following the truncated character will not make sense.

- The system ignores the SFLEND keyword when displaying a plus sign (+) to indicate that more records exist in the subfile. When displaying the plus sign, the system writes over a DBCS character.

- Consider the following when you specify the MSGID keyword:
  - If the message text contains DBCS characters, and the message length exceeds the MSGID field length, the message text is truncated so it ends with an alphanumeric character. If the truncation occurs in the middle of a DBCS character, the text truncates after the previous DBCS character and a shift in character is added to the end of the text.
  - If the message text contains DBCS characters, either define the MSGID field so that it does not wrap to the next line, or make sure the message text does not wrap in the middle of a DBCS character.

Printer Files

This section describes the DBCS considerations for the positional entries and keyword entries for printer files.

Positional Entry Considerations

The following section describes, by position, DDS for describing printer files. Positions not mentioned have no special considerations for DBCS.

Length (Positions 30 through 34)

Specify the length of a field in these positions. The length of a field containing bracketed-DBCS data can range from 4 through 32 767 bytes. The length of a DBCS-graphic field can range from 1 through 16 383 characters. However, because a field cannot span a printed page, the maximum length of a field might not reach the maximum size.

When determining the length of a DBCS field, consider the following:

- Each DBCS character is 2 bytes long.
- For DBCS-graphic fields, the length of the field is specified in number of DBCS characters.
- Include both shift-control characters in the length of the field for fields with a data type of O. Together, these characters are 2 bytes long.

For example, a bracketed-DBCS field that contains up to 3 DBCS characters, 1 shift-in character and 1 shift-out character, has 8 bytes of data:

```
(3 characters x 2 bytes) + (shift-out + shift-in) = 8
```

A DBCS-graphic field that contains up to 3 DBCS characters has 6 bytes of data:

```
(3 characters x 2 bytes) = 6
```

Data Type/Keyboard Shift (Position 35)

Type an O in this position to make a field a DBCS-open field. You can use DBCS and alphanumeric data in the same field. Use shift-control characters to distinguish DBCS from alphanumeric data.

Type a G in this position to make a field a DBCS-graphic field.

Decimal Positions (Positions 36 and 37)

Leave these positions blank when using DBCS data.

Keyword Considerations

Do not use the following keywords with DBCS data fields (the data type specified in position 35 is O or G):

BARCODE	DLTEDT	HIGHLIGHT
BLKFOLD	EDTCDE	MSGCON
CDEFNT	EDTWRD	PAGNBR
COLOR	FLTFIXDEC	TIME
CPI	FLTPCN	TRNSPY
CVTDTA	FNTCHRSET	
DATE	FONT	

Do not use the IGCALTTYP and IGCANKCNV keywords on DBCS-graphic data fields (G specified in position 35).

For additional information on the keywords for printer files, refer to the keyword descriptions in Chapter 4, "Printer Files."

You can also use the CHRSIZ keyword as described below.

CHRSIZ (Character size): Use this record- or field-level keyword to expand the printed characters to twice their normal width, their normal height, or their normal size (width and height). Before expanding the characters, the system performs any formatting operations specified, such as those specified with the DDS keywords EDTCDE and EDTWRD.

The format of this keyword is:

```
CHRSIZ(width [height])
```

The valid values for the width and height parameters are 1 and 2. The width parameter is required; the height parameter is optional. If height is not specified, it defaults to 1.

This format is valid only on the 5553 printer.

Consider the following when using CHRSIZ(width [height]):

- It can be used for both DBCS and alphanumeric data.
- It can be specified with the IGCCHRRTT keyword. The characters are first rotated, then expanded.
- It cannot be specified on a record or on a field in a record if that record also contains COLOR, BARCODE, or LPI.

- It expands characters in alphanumeric fields specified with the keyword IGCANKCNV. Characters in other alphanumeric fields are not expanded.
- If you specify CHRSIZ(1) at the field level, data will be printed in its normal width even when CHRSIZ(2) is specified for the record.
- Option indicators are not allowed with this keyword.

Figure E-4 shows how to specify the CHRSIZ keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*
00030A          R RECORD1                   CHRSIZ(2 1)
00040A            FIELD1       230        20SPACEA(2)
00050A            FIELD2       80A        20SPACEA(2) CHRSIZ(1 1)
00060A            FIELD3       410        20SPACEA(2)
00070A            FIELD4       450        20SPACEA(2) CHRSIZ(1 2)
00080A            FIELD5       40A        20SPACEA(2) CHRSIZ(1)
00090A            FIELD6       25A        20SPACEA(2) CHRSIZ(2 2)
     A
```

Figure E-4. Specifying the CHRSIZ Keyword for Printer Files

In Figure E-4, the DBCS characters in FIELD1 and FIELD3 expand to twice their normal width when printed. The DBCS characters in FIELD2 and FIELD5 are in their normal size when printed. The DBCS characters in FIELD4 expand to twice their normal height when printed. The DBCS characters in FIELD6 expand to twice their normal size (width and height) when printed.

DFNLIN (Define Line)

Use this record-level keyword to draw a horizontal or a vertical line. A horizontal line is drawn at the bottom of the character spaces from left to right. A vertical line is drawn on the right edge of the character spaces from top to bottom.

The format of this keyword is:

```
DFNLIN(direction  start_line  start_position  length)
```

The direction parameter specifies whether the defined line is horizontal or vertical. The value specified must be one of the following:

*VRT

*HRZ

The start line parameter specifies the line number, from the top of the page, where the defined line starts. The possible values are 1 through 255, but the value specified must not exceed the page length value specified on the PAGESIZE parameter of the Create Printer File (CRTPRTF) command.

The start position parameter specifies the position number, from the left margin of the page, where the defined line starts. The possible values are 1 through 378, but the value specified should not exceed the page width value specified on the PAGESIZE parameter of the CRTPRTF command.

The length parameter specifies the length in number of lines when the defined line is vertical or in number of characters when the defined line is horizontal.

The length specified must be greater than zero. For a vertical line, the sum of the length and the value of the start line parameter cannot exceed 255. Although 255 is the maximum value of this sum, valid values must not exceed the page length specified on the PAGESIZE parameter of the CRTPRTF command.

For a horizontal line, the sum of the length and the value of the start position parameter cannot exceed 378. Although 378 is the maximum value of this sum, valid values must not exceed the page width specified on the PAGESIZE parameter of the CRTPRTF command.

All parameters are required.

A warning message will be issued at create time if:

- The start line value specified is larger than the page length value specified on the PAGESIZE parameter of the CRTPRTF command.
- The start position value specified is larger than the page width value specified on the PAGESIZE parameter of the CRTPRTF command.
- The sum of the length and the start line value for a vertical line is larger than the page length specified on the PAGESIZE parameter of the CRTPRTF command.
- The sum of the length and the start position value for a horizontal line is larger than the page width specified on the PAGESIZE parameter of the CRTPRTF command.

The DFNLIN keyword can be specified more than once at the record level.

Option indicators are allowed for this keyword.

The DFNLIN keyword cannot be specified on a record that also contains keywords that are valid only on IPDS printers (such as COLOR, BARCODE, and LPI). If DFNLIN is specified with any of those keywords, a severe error (severity 30) message will be issued.

If the DFNLIN keyword is specified when a printer file is created with DEVTYPE(*IPDS), a warning (severity 10) message is issued but the keyword is not ignored at creation time. However, the keyword is ignored and a message is issued when the printer file is used.

If the DFNLIN keyword is specified with a printer file created with DEVTYPE(*AFPDS), the keyword is ignored and a warning message is issued.

Figure E-5 shows how to specify the DFNLIN keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*
00030A          R RECORD1                   DFNLIN(*HRZ  4  12  20)
00040A                                      DFNLIN(*VRT  5  12  6)
00050A                                      DFNLIN(*HRZ  10 12  20)
00060A                                      DFNLIN(*VRT  5  32  6)
     A
```

Figure E-5. Specifying the DFNLIN Keyword for Printer Files

The output of RECORD1 format is a box.

The output of the first DFNLIN keyword is a horizontal line. The line is drawn to the right from the twelfth character position on the fourth line for a length of 20 characters.

The output of the second DFNLIN keyword is a vertical line. The line is drawn down from the twelfth character position on the fifth line for a length of 6 lines.

The output of the third DFNLIN keyword is a horizontal line. The line is drawn to the right from the ending point of the second line (10 = 4 + 6) for a length of 20 characters.

The output of the fourth DFNLIN keyword is a vertical line. The line is drawn down from the ending point of the first line (32 = 12 + 20) for a length of 6 lines.

IGCALTTYP (Alternative Data Type)

Use this field-level keyword to change alphanumeric character fields to the DBCS fields of data type O.

This keyword has no parameters.

Put the keyword function into effect by changing the IGCDTA parameter value in the file description, using the CRTPRTF, CHGPRTF, or OVRPRTF command. Fields specified with this keyword are alphanumeric character fields when you specify IGCDTA(*NO) and are DBCS fields of data type O when you specify IGCDTA(*YES). For example, create the file by specifying IGCDTA(*NO) on the CRTPRTF command. When using the file to print DBCS data, override the file with the OVRPRTF command, specifying IGCDTA(*YES). To override the printer file IGCPRTF, type:

```
OVRPRTF FILE(IGCLIB/IGCPRTF) IGCDTA(*YES)
```

Consider the following when using the IGCALTTYP keyword.

- Specify IGCALTTYP only for character fields. Do not specify it for DBCS fields.
- Do not specify IGCALTTYP when other keywords defined for the field depend on the data type, because the function of this keyword is to change the data type.
- Option indicators are not allowed with IGCALTTYP.
- This keyword is ignored for files created with DEVTYPE(*AFPDS).
- The following keywords are not allowed with the IGCALTTYP keyword:

  BLKFOLD
  CPI
  CVTDTA
  IGCANKCHV
  TRNSCY

Figure E-6 on page E-18 shows how to specify the IGCALTTYP keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*
00030A          R TITLER                    SKIPB(3)
00040A            FLD1          40        47SPACEA(2) UNDERLINE
00050A  30        FLD2          40A       47SPACEA(2) UNDERLINE IGCALTTYP
     A
```

Figure E-6. Specifying the IGCALTTYP Keyword for Printer Files

When the IGCALTTYP keyword is put into effect, FLD2 can contain DBCS data.

IGCANKCNV (Alphanumeric-to-DBCS Conversion)

Use this field-level keyword to convert alphanumeric characters to equivalent DBCS characters (Japanese only). Each DBCS character is printed twice as wide as a printed alphanumeric character.

This keyword has no parameters.

In addition to converting alphanumeric characters, the system adds shift-control characters at the beginning and end of converted character strings.

For example, the string

```
ABCDE
```

appears as:

0_EA B C D E0_F

after it is converted. Notice that shift-control characters were added to the string (0_E=shift-out, 0_F=shift-in).

You may specify IGCANKCNV for any named field.

Consider the following when using the IGCANKCNV keyword:

- The converted characters are printed according to the instructions specified for printing DBCS data, such as expanded characters. For example, if you specify the CHRSIZ(2) keyword, the characters converted by this keyword are doubled in width.
- This conversion does not affect other attributes of a file. For example, if you specify this DDS keyword for a field that contains floating-point data, the system leaves the data in the floating-point format. Only the printed appearance of the field changes. Also, any other attributes defined for the field are still applicable, even those that are not valid for DBCS fields.
- The following DDS keywords are ignored when you specify the keyword IGCANKCNV:

  BLKFOLD
  CPI
  DFT
  IGCALTTYP

- The length of the printed string of characters expanded by the IGCANKCNV function is at least two times the length of the original string plus 2 positions for

the shift-control characters. For example, after a string of 4 Katakana characters is converted, its length is:

```
10 ((4 characters by 2) + 2 shift-control characters)
```

If you specified additional characters to be included in the string, such as with the EDTWRD function, those characters also are expanded and the length of the string changes accordingly. For example, suppose you specified a 4-position field that also includes a dollar sign and a decimal point, such as:

```
$12.34
```

After the field is converted, the field length is 14. The four numbers in the field are expanded (8 positions), the dollar sign and the decimal point are expanded (4 positions) and shift-control characters are added (2 positions).

- The field for which IGCANKCNV is specified should not contain any DBCS data. The system does not support conversion of fields with both alphanumeric and DBCS data. If a field with IGCANKCNV contains DBCS characters, the results of the conversion cannot be predicted.
- The field for which IGCANKCNV is specified cannot be a DBCS-graphic field (a field with a data type of G).
- The system replaces unprintable alphanumeric characters before it converts them to equivalent DBCS characters as specified by the RPLUNPRT value on the Create Printer File (CRTPRTF) command.
- The output must be printed on a DBCS printer.
- A warning message appears if IGCANKCNV is specified in a file created with DEVTYPE(*IPDS).
- For files created with DEVTYPE(*AFPDS), characters in the field specified with IGCANKCNV are printed using the font identified by the IGCCDEFNT keyword. See "IGCCDEFNT (DBCS Coded Font)" on page E-20 for more information.
- IGCANKCNV cannot be specified on a record or on a field in a record if that record also contains COLOR, BARCODE, or LPI.
- Option indicators are not allowed with IGCANKCNV.

Figure E-7 shows how to specify the IGCANKCNV keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*
00030A          R RECORD                    CHRSIZ(2) SKIPB(3)
00040A            FLDA          400       20SPACEA(2)
00050A            FLDB          80A       20SPACEA(2) CHRISIZ(1)
00060A            FLDC          20A       20SPACEA(2) IGCANKCNV
     A
```

Figure E-7. Specifying the IGCANKCNV Keyword for Printer Files

The alphanumeric characters printed from FLDC are converted to equivalent DBCS characters. These converted characters are then expanded because the record was specified with the DDS keyword CHRSIZ(2).

IGCCDEFNT (DBCS Coded Font)

Use this record- or field-level keyword to specify the DBCS coded font for printing a named or constant field or fields.

The format of this keyword is:

```
IGCCDEFNT([library-name/]coded-font)
```

The coded-font parameter is required and must be the name of an AS/400 DBCS coded font. It can be up to 8 characters long.

Use the optional library-name parameter to further qualify the coded-font. If the library name is not specified, *LIBL is used to search for the coded font at print time.

Note: If an application uses private resources (for example, fonts, page segments, overlays, or GDF files not distributed with the system), be aware of the following. When referencing these resources, if you specify *LIBL or you do not specify a library name, the resources must be available through the library list used by the application creating the spooled file.

The coded font value is validated at print time. An error message is issued if the value is not valid.

Option indicators are valid for this keyword.

Figure E-8 shows how to specify the IGCCDEFNT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A*
     A          R REC
     A            FLD1          240     2 14IGCCDEFNT(XOG16B)
     A
```

Figure E-8. Specifying the IGCCDEFNT Keyword

FLD1 in REC specifies coded font X0G16G. *LIBL is used to locate the DBCS resource.

IGCCHRRTT (DBCS Character Rotation)

This field- or record-level keyword rotates each DBCS character 90 degrees counterclockwise before printing. Rotation allows the system to print the characters so the printout can be read vertically. Use this keyword only for printer files to be printed with 5553 printers or IPDS AFP(*YES) printers.

This keyword has no parameters.

Consider the following when using this keyword:

- Use IGCCHRRTT only with DBCS fields (the data type specified in position 35 is O or G) or with DBCS in constant fields.
- IGCCHRRTT rotates characters in alphanumeric fields specified with the DDS keyword IGCANKCNV, but does not rotate other alphanumeric characters.

- The system ignores IGCCHRRTT if IGCCHRRTT(*YES) was specified on the CRTPRTF, CHGPRTF, or OVRPRTF command.
- If IGCCHRRTT is specified in a file created with DEVTYPE(*IPDS), a warning message appears.
- Characters in a field specified with the IGCCHRRTT keyword are rotated 270 degrees with respect to the page for files created with DEVTYPE(*AFPDS). Only DBCS characters are rotated.
- Option indicators are not allowed with this keyword.

Figure E-9 shows how to specify the IGCCHRRTT keyword on the DDS coding form.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A*
00030A          R TITLER                    SKIPB(3)
00035A                                      IGCCHRRTT
00040A            FLD1          400       47SPACEA(2) UNDERLINE
     A
```

Figure E-9. Specifying the IGCCHRRTT Keyword for Printer Files

The printer prints DBCS characters from this format 90 degrees counterclockwise. The DBCS output can be read vertically.

Additional Considerations for Describing Printer Files

Consider the following when describing a printer file that contains DBCS data:

- If you describe fields in the file as DBCS fields, the system considers the file to be DBCS even if you do not specify IGCDTA(*YES) on the file creation command.
- Specify IGCDTA(*YES) on the CRTPRTF command when DBCS data is present in the file, but is not indicated in DDS. For example, specify IGCDTA(*YES) if the file sends messages that are DBCS (DDS keyword, MSGCON).
- Each printed DBCS character is twice as wide as a printed alphanumeric character. The location of a character on a printed page, as specified in DDS, is affected by the value specified for the CPI and IGCCPI parameters in the file description. Although the system does not use the CPI or IGCCPI values to determine the printed length of a field, this value does affect the physical space used on a printed form.

  The physical space occupied on a printed form is also affected by the method used to print the shift-control characters. Specify shift-control character printing in the file description (IGCSOSI parameter on the CRTPRTF, CHGPRTF, and OVRPRTF commands). The IGCSOSI value specified for the printer file is ignored for DBCS-graphic fields. These fields are printed as if IGCSOSI(*NO) was specified.

  Note: DDS does not consider the values specified for the CPI, IGCCPI, and IGCSOSI parameter values when calculating the printed length. Therefore, overlapping may occur when the field is actually printed, although the problem is not indicated when the DDS file is compiled.

  This information applies to constant fields with DBCS data and to named fields.

- When using the reference function in a printer file, if you refer to a field in a database file that has data type J, O, or E, DDS assigns data type O for the field in the printer file. If you refer to a field that has data type G, DDS assigns data type G for the field in the printer file.

ICF Files

This section describes the bracketed-DBCS considerations for positional entries and keyword entries for ICF files.

Positional Entry Considerations

The following section describes, by position, DDS for describing ICF files. Those positions not mentioned have no special considerations for DBCS.

Length (Positions 30 through 34)

The length of a field containing bracketed-DBCS data can range from 4 through 32 767 bytes.

Consider the following when determining the length of a DBCS field:

- Each DBCS character is 2 bytes long.
- Include both shift-control characters in the length of the field. Together, these characters are 2 bytes long.

  For example, a field that contains up to 3 DBCS characters, 1 shift-in character, and 1 shift-out character, has 8 bytes of data:

  ```
  (3 characters x 2 bytes) + (shift-out + shift-in) = 8
  ```

Data Type (Position 35)

The data type O (DBCS capable) makes a field DBCS. Both bracketed-DBCS and alphanumeric data can be used in a DBCS-capable field. Use shift-control characters to distinguish DBCS data from alphanumeric data.

Additional Considerations for Describing ICF Files

Consider the following when describing an ICF file that contains DBCS data:

- Use ICF files only to send data from the AS/400 system to another system or to a remote work station.
- Send shift-control characters with DBCS data. The system is not aware of DBCS data in an ICF file, and treats DBCS data the same as alphanumeric data.
- When using the reference function in an ICF file, if you refer to a field in a database file that has data type J, O, or E, DDS will assign data type O for the field in the ICF file.
- DBCS-graphic fields cannot be referenced from a database file because fields with data type G are not allowed in an ICF file. If a field with a data type of G is referenced from a database file, an error message is issued.

Appendix F. System/36 Environment Considerations

This appendix describes DDS keyword differences and considerations for display files used in the System/36 environment. For a description of operational differences, see the *Guide to Programming Displays*.

General Considerations

The User Display Management (USRDSPMGT) keyword causes the record formats in the display file to function similarly to System/36 SFGR display formats. The SFGR-to-DDS conversion utility always generates the USRDSPMGT keyword. If you are defining a DDS display file to be used in the System/36 environment, you should specify the USRDSPMGT keyword.

Keyword Considerations

You cannot specify any of the following keywords in a display file containing USRDSPMGT:

ASSUME	MNUBAR
ERASE	PULLDOWN
ERRSFL	PUTRETAIN
HLPCMDKEY	SFL
IGCCNV	SFLCTL
KEEP	SNGCHCFLD

In a file containing the USRDSPMGT keyword, the OVERLAY keyword is ignored.

For additional information on the keywords for display files, refer to the keyword descriptions in Chapter 3, "Display Files."

The following keywords have a response indicator parameter. Since System/36 environment applications do not support response indicators, you should not specify a response indicator on any of these keywords in a file to be used in the System/36 environment. If you specify a response indicator for any of these keywords in a file containing the USRDSPMGT keyword, a warning message is issued.

BLANKS	ERRMSGID	PRINT
CAnn	HELP	ROLLDOWN
CFnn	HLPRTN	ROLLUP
CHANGE	HOME	SETOF
CLEAR	PAGEDOWN	SETOFF
DUP	PAGEUP	VLDCMDKEY
ERRMSG		

If you specify a response indicator on the HELP keyword in a file containing the USRDSPMGT keyword, an error message is issued.

Other keywords with special considerations can be used as described below.

© Copyright IBM Corp. 1991, 1993

CHANGE

Use this record-level keyword to indicate that, on an input operation, the record is to be returned to the application program only if the user has changed it. If the user enters data into any input-capable field, all of the input-capable fields in the record will be returned. If the user does not enter data into any field, the input-capable fields will be returned initialized by the compiler.

This keyword has no parameters.

This format of the CHANGE keyword is allowed only in files containing the USRDSPMGT keyword.

Option indicators are not allowed with this keyword.

For files that are used by the OS/400 program applications (with or without the USRDSPMGT keyword), use the format of the CHANGE keyword described under "CHANGE (Change)" on page 3-43.

HELP and HLPRTN

In a file with the USRDSPMGT keyword, the HELP keyword alone will not return control to the application program. HLPRTN must be specified to return control to the application program.

If you specify a response indicator on the HELP keyword in a file containing the USRDSPMGT keyword, an error message is issued.

MSGID

You can specify the MSGID keyword in either of the following formats:

```
MSGID(message-identifier [library-name/]message-file)
MSGID(*NONE)
```

You can specify the message-file parameter in one of the following forms:

- `&field3`

  where the field3 length is two (2).

  The field name must exist in the same record as the MSGID field, and the field must be defined as a character field with usage H, P, B, or O.

  You should, for this form only, specify only special values for the file parameter. You cannot specify a library.

  The special values are: U1, U2, P1, P2, M1, and M2. If the specified value is not one of these special values, U1 is used. See Figure F-1 on page F-3 for more information on these values.

- Special values for message-file:
  - *USR1
  - *USR2
  - *PGM1
  - *PGM2
  - *SYS1
  - *SYS2

When you code a special value for the message-file, the library parameter is not allowed and the library defaults to *LIBL. Refer to Figure F-1 for more information on these special values.

The following table describes the meaning of the special file values for the MSGID keyword.

Figure F-1. Special Values on MSGID Keyword

DDS Special Value	Length 2 Field Value	Message Text Retrieved
*USR1	U1	First level text from message file USR1
*USR2	U2	Second level text from message file USR2
*PGM1	P1	First level text from message file PGM1
*PGM2	P2	Second level text from message file PGM2
*SYS1	M1	First level text from message file SYS1
*SYS2	M2	Second level text from message file SYS2

For information on using message retrieval, see the *Guide to Programming Displays*.

The *NONE parameter indicates that no message text is displayed.

For more information on how to specify the MSGID keyword, refer to "MSGID (Message Identifier)" on page 3-164.

Figure F-2 shows how to specify the MSGID keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1
     A            MSGFIELD1     40A  B 02 10MSGID(CPD0001 *USR1)
     A            MSGFIELD2     10A  O 02 60MSGID(&MSGIDNUM &MSGFILENM)
     A            MSGFIELD3     80A  B 02 60
     A  99                                 MSGID(USR &MSGNBR +
     A                                     &MSGFILENM)
     A                                     MSGID(*NONE)
     A            MSGIDNUM       7A  P      TEXT('Message id')
     A            MSGFILENM      2A  P      TEXT('Message file name')
     A            MSGNBR         4A  P 07 01TEXT('Message number')
     A
```

Figure F-2. Specifying the MSGID Keyword

When RECORD1 is displayed:

- MSGFIELD1 contains the first 40 characters of the message CPD0001 from the message file USR1. Since the field is output/input (usage B), the value of the field can be changed by the user.
- MSGFIELD2 contains the first 10 characters of the message identified by the fields MSGIDNUM and MSGFILENM. Values for MSGIDNUM (the message identifier) and MSGFILENM (the message file) must be set in the program prior to the display of RECORD1. Since MSGFIELD2 is an output-only field (usage O), it cannot be used in the program.
- If option indicator 99 is on, MSGFIELD3 contains the first 80 characters of the message identified by the prefix USR, the message number set in field

MSGNBR, and the message file set in field MSGFILENM. If option indicator 99 is off, MSGFIELD3 does not contain any message text.

PRINT(*PGM)

How the PRINT key is handled for a display file with the PRINT(*PGM) and USRDSPMGT keywords specified is determined by how the program that is reading data from the display file is compiled and coded.

If the program is compiled with a System/36-compatible compiler (RPGII or COBOL) and the program is coded to handle the Print key exception, the program is given control when the Print key is pressed. If the program is coded not to handle the Print key exception, the screen image is printed.

If the program is compiled with an AS/400 compiler (RPGIII or COBOL), the program is always given control when the Print key is pressed. If the program does not handle the Print key exception, it acts as if the Enter key was pressed.

ALTNAME (Alternative Record Name)

Use this record-level keyword to specify an alternative name for a record. The alternative name can be specified for I/O operations when using program-described files. The syntax of the alternative record name must be valid for the high-level language compiler in use.

The format of the keyword is:

```
ALTNAME('alternative-name')
```

The length of the alternative name is 1 to 8 characters. The first character of the name must not be an asterisk.

The alternative name must be different from all other names and from all DDS record names (positions 19 through 28) in the file. When a duplicate name is found, an error is issued on the record name or alternative record name.

ALTNAME is not allowed on subfile records (SFL keyword).

Option indicators are not valid for this keyword.

Figure F-3 shows how to specify the ALTNAME keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A          R RECORD1                  ALTNAME('R( 2).a')
     A
```

Figure F-3. Specifying the ALTNAME Keyword

The alternative name for RECORD1 is 'R(2).a'.

RETKEY (Retain Function Keys) and RETCMDKEY (Retain Command Keys)

Use these record-level keywords to indicate that function keys or command function/attention keys, which were enabled on a display, should be retained when the record you are defining is displayed.

In most cases, the keys enabled on the display are those which were specified on the last output operation. Additionally, the OS/400 program automatically retains valid keys when a record format sends no data to the display. See "CAnn (Command Attention)" on page 3-39 and "CFnn (Command Function)" on page 3-41 for more information.

Note that if the record previously displayed is defined in another display file, the keys enabled for that record will not be retained when the record you are defining is displayed.

These keywords have no parameters.

RETKEY

RETKEY indicates to retain any CLEAR, HELP, HLPRTN, HOME, PAGEDOWN, PAGEUP, PRINT, ROLLDOWN, and ROLLUP keywords when the record is displayed. You cannot specify RETKEY with a CLEAR, HELP, HOME, PAGEUP, PAGEDOWN, ROLLDOWN, or ROLLUP keyword on the file level or on this record format. PRINT is not allowed on the same record format with RETKEY.

The HLPRTN and PRINT keywords are allowed at the file level with RETKEY. If option indicators are specified on either HLPRTN or PRINT, the state of the indicators when the current record is displayed determines whether the keyword is active.

If you specify HLPRTN on the same record format with RETKEY, the HLPRTN function will not be retained from the previous record.

RETCMDKEY

RETCMDKEY indicates whether to retain CAnn or CFnn keys when the record is displayed. You cannot specify the CAnn or CFnn keywords with the RETCMDKEY on the file level or on this record format. You cannot specify any CAnn, CFnn, SFLDROP, SFLENTER, or SFLFOLD keywords on the record being defined.

Specifying RETKEY and RETCMDKEY

The following must be considered when specifying the RETKEY and RETCMDKEY keywords:

- The file must specify a separate indicator area (INDARA keyword).
- RETKEY and RETCMDKEY are ignored on the first output operation after a file is opened. The retain function is valid only between record formats in the same display file.
- The response indicator of the VLDCMDKEY keyword is set by the OS/400 program according to the current valid command keys, including the keys retained when you specify the RETKEY and RETCMDKEY keywords.
- Neither keyword is allowed on a subfile format (SFL keyword) or on a user-defined record (USRDFN keyword).

- You cannot specify either RETKEY or RETCMDKEY in a file that contains the ALTHELP, ALTPAGEUP, or ALTPAGEDWN keyword.

Option indicators are not valid for these keywords.

Figure F-4 shows how to specify the RETKEY and RETCMDKEY keywords.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      INDARA
     A          R RECORD1                   CF01(01 'EXIT SCREEN')
     A                                      CF02(02 'SET ON IN02')
     A                                      ROLLUP(03)
     A  08                                  CLEAR(03 'CLEAR KEY')
     A                                    1  3'COMPANY NAME'
     A                                    1 25'CF01 TO EXIT'
     A          R RECORD2                   RETKEY
     A                                      RETCMDKEY
     A                                      OVERLAY
     A            FIELD1         4A  B  5  5
     A          R RECORD3                   RETKEY
     A                                      CF01(90 'ALTERNATE CF01')
     A            FIELD1        10A  B  7  5
     A
```

Figure F-4. Specifying the RETKEY and RETCMDKEY Keywords

The records are displayed in the following order: RECORD1, RECORD2, RECORD3.

When RECORD1 is displayed, CF01, CF02, Clear, and Page Up keys are activated. The same keys are valid for RECORD2, since RETKEY and RETCMDKEY are specified. Since RECORD3 specifies RETKEY, the Clear and Page Up keys are valid. CF01 has been redefined for this record. CF02 is not valid for this record.

Note: The retain function does not require the record format to be displayed. (RECORD3 uses function keys defined in RECORD1, but since no OVERLAY keyword is specified in RECORD3, the display is erased prior to displaying RECORD3.)

USRDSPMGT (User Display Management)

Use this file-level keyword to indicate that this display file should be processed with System/36 environment functions.

This keyword has no parameters.

You cannot use USRDSPMGT in display files containing any of the following keywords:

ASSUME
ERASE
HLPCMDKEY
IGCCNV
KEEP
PUTRETAIN
SFL
SFLCTL

In a file containing the USRDSPMGT keyword, the OVERLAY keyword is ignored.

Option indicators are not valid for this keyword.

Figure F-5 shows how to specify the USRDSPMGT keyword.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
     A                                      USRDSPMGT
     A          R RECORD
     A            FIELD1
     A
```

Figure F-5. Specifying the USRDSPMGT Keyword

Appendix G. CODE128 Character Set

Figure G-1 lists the range of characters available in the CODE128 bar code character set.

Figure G-1. Code 128 Character Set (EBCDIC)

Character	Hex	Character	Hex	Character	Hex	Character	Hex
NUL	00	.	4B	i	89	I	C9
SOH	01	<	4C	FNC 1	8F	}	D0
STX	02	(	4D	j	91	J	D1
ETX	03	+	4E	k	92	K	D2
HT	05	¦	4F	l	93	L	D3
VT	0B	&	50	m	94	M	D4
FF	0C	!	5A	n	95	N	D5
CR	0D	$	5B	o	96	O	D6
SO	0E	*	5C	p	97	P	D7
SI	0F	)	5D	q	98	Q	D8
DLE	10	;	5E	r	99	R	D9
DC1	11	-	60	~	A1	\	E0
DC2	12	/	61	s	A2	S	E2
DC3	13	,	6B	t	A3	T	E3
BS	16	%	6C	u	A4	U	E4
CAN	18	_	6D	v	A5	V	E5
EM	19	>	6E	w	A6	W	E6
GS	1D	?	6F	x	A7	X	E7
RS	1E	`	79	y	A8	Y	E8
US	1F	:	7A	z	A9	Z	E9
FS	22	#	7B	¬	B0	FNC 2	EA
LF	25	@	7C	¢	BA	0	F0
ETB	26	'	7D	|	BB	1	F1
ESC	27	=	7E	FNC 4	BE	2	F2
ENQ	2D	"	7F		C0	3	F3
ACK	2E	a	81	A	C1	4	F4
BEL	2F	b	82	B	C2	5	F5
SYN	32	c	83	C	C3	6	F6
EOT	37	d	84	D	C4	7	F7
DC4	3C	e	85	E	C5	8	F8
NAK	3D	f	86	F	C6	9	F9
SUB	3F	g	87	G	C7	FNC 3	FA
SP	40	h	88	H	C8	DEL	FF

© Copyright IBM Corp. 1991, 1993

Glossary

This glossary includes terms and definitions from:

- The *American National Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 1430 Broadway, New York, New York 10018. Definitions are identified by the symbol (A) after the definition.
- The *Information Technology Vocabulary*, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Committee (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.

A/N/K. Pertaining to alphabetic, numeric, or Katakana characters.

absolute value. The magnitude of a number.

access path. The order in which records in a database file are organized for processing by a program. See *arrival sequence access path* and *keyed sequence access path*.

active record. An active subfile record or any record format that is currently shown on a display. See also *active subfile record*. Contrast with *inactive record*.

active subfile. A subfile in which a write operation was issued to the subfile record format or to the subfile control record format with the DDS keyword SFLINZ in effect.

active subfile record. A record that was added to the subfile by a write operation, or a record that was initialized by the DDS keyword SFLINZ. Contrast with *inactive subfile record*.

Advanced Function Printing (AFP). Pertaining to the ability of programs to use the all-points-addressable concept to print text and images on a printer.

Advanced Function Printing data stream (AFPDS). In AFP support, the printer data stream used for printing Advanced Function Printing data. The AFPDS includes composed text, page segments, electronic overlays, form definitions, and fonts that are downloaded from the AS/400 system to the printer.

AFP. See *Advanced Function Printing (AFP)*.

AFP resources. The form definitions, page definitions, fonts, overlays (electronic forms), and page segments (graphic images). With the PrintManager program, resources can either exist in a system library, or be placed inline with a print job as the job is written to the spool.

AFPDS. See *Advanced Function Printing data stream (AFPDS)*.

alphabetic character. In DDS and IDDU, any one of the uppercase letters A through Z or one of the characters #, $, or @.

alphanumeric. Pertaining to the letters A through Z or a through z; numbers 0-9; and special symbols $, #, @, ., or _.

alternative collating sequence. A user-defined collating sequence that replaces the standard EBCDIC collating sequence. See *collating sequence*.

arrival sequence access path. An access path to a database file that is arranged according to the order in which records are stored in the physical file. See also *keyed sequence access path* and *access path*.

ascending sequence. The arrangement of data in order from the lowest value to the highest value, according to the rules for comparing data. Contrast with *descending sequence*.

asterisk fill. A type of numeric editing that puts asterisks to the left of a number to fill unused positions. Example: *****476.12

attribute. A characteristic or trait of one or more items.

attribute character. A character associated with a field in a display file record format that defines how the field is displayed.

bar code. A pattern of bars of various widths containing data to be interpreted by a scanning device.

beginning attribute character. For a display file, the character that precedes the first position in a field and that defines how the data in the field is displayed.

both field. A field that can be used for either input data or output data.

© Copyright IBM Corp. 1991, 1993

bracketed DBCS. A character string in which each character is represented by 2 bytes. The character string starts with a shift-out (SO) character and ends with a shift-in (SI) character. Contrast with *DBCS-graphic*.

byte. A group of 8 adjacent bits. In the EBCDIC coding system, 1 byte can represent a character. In the double-byte coding system, 2 bytes represent a character.

changed subfile record. A subfile record into which the work station user has entered data, or a subfile record for which a write or change operation was issued with the DDS keyword SFLNXTCHG or DSPATR(MDT) in effect.

character. Any letter, number, or other symbol in the data character set that is part of the organization, control, or representation of data.

character constant. The actual character value (a symbol, quantity, or constant) in a source program that is itself data, instead of reference to a field that contains the data. Contrast with *numeric constant*.

character field. An area that is reserved for information that can contain any of the characters in the character set. Contrast with *numeric field*.

character set. A group of characters used for a specific reason; for example, the set of characters the display station can display, the set of characters a printer can print, or a particular set of graphic characters in a code page; for example, the 256 EBCDIC characters.

character string. A sequence of consecutive characters that are used as a value.

characters per inch (cpi). The number of characters printed horizontally within an inch across a page.

check digit. The far right number of a self-check field used to verify the accuracy of the field.

close. The function that ends the connection between a file and a program, and ends the processing. Contrast with *open*.

code page. (1) A particular assignment of hexadecimal identifiers to graphic characters. (2) In AFP support, a font file that associates code points and graphic character identifiers.

code point. (1) One of the bit patterns assigned to a character in a character set. On the AS/400 system, a code point is represented by a hexadecimal number. For example, in code page 256 (EBCDIC), the letter "e" is assigned a code point of hex 85. (2) In AFP support, an 8-bit binary number representing one of 256 potential characters.

code-page ID. A 5-digit registered identifier used to specify a particular assignment of code points to graphic characters. The code-page ID is the second part of the QCHRID system value or the CHRID parameter value. See also *graphic character-set ID*.

coded character set identifier (CCSID). A 16-bit number identifying a specific set of encoding scheme identifiers, character set identifiers, code page identifiers, and other relevant information that uniquely identifies the coded graphic character representation used.

coded font. In AFP support, a font file that associates a code page and a font character set. For double-byte fonts, a coded font associates multiple pairs of code pages and font character sets.

coded graphic character-set ID. A 10-digit identifier (two 5-digit identifiers separated by a space) that is the combination of a graphic character-set ID and a code-page ID. See also *graphic character-set ID* and *code-page ID*.

collating sequence. The order in which characters are arranged within the computer for sorting, combining, or comparing.

column separator. A symbol on each side of a position of a field on a display. This symbol does not occupy a position on the display.

command. A statement used to request a function of the system. A command consists of the command name abbreviation, which identifies the requested function, and its parameters.

command attention (CA) key. In DDS, a keyboard key that can be specified with the CA keyword to request the function specified by the keyword. Data is not returned to the system. Contrast with *command function (CF) key*.

command function (CF) key. In DDS, a keyboard key that can be specified with the CF keyword to request the function specified by the keyword. Data is returned to the system. Contrast with *command attention (CA) key*.

compile. To translate a program written in a high-level programming language into a machine-language program.

compile time. The time during which a source program is translated by a compiler into a machine-language program.

compiler listing. A printout that is produced by compiling a program or creating a file and that optionally

includes, for example, a line-by-line list of the high-level language source, a cross-reference list, diagnostic information; and for programs, the description of the externally described files. See also *source listing*.

composite key. A key for a file or record format that is composed of more than one key field.

concatenated field. Two or more fields that are combined to make one field in a logical file.

condition name. For display files, a name used to control the selection of DDS keywords and display locations based on the model of the display station.

conditioning. A feature of the SAA AD/Cycle CoOperative Development Environment/400 licensed program that helps users to develop and create display files, printer files, and database files.

constant field. In an externally described display or printer file, an unnamed field that contains actual data that is passed to the display or printer but is unknown to the program passing it.

continuation line. An additional line (or lines) required to continue the coding of a CL command or a DDS keyword and its value.

country ID. See *country identifier (country ID)*.

country identifier (country ID). The 2-character representation for the country associated with an object. For example, documents and user profiles can have a country associated with them.

creation date. The system date when an object is created. See also *job date*, and *system date*.

currency symbol. A character such as the dollar sign ($) used to identify monetary values.

cursor. A movable symbol, often a blinking or solid block of light, that tells the display station user where to type, or identifies a choice to select.

data description specifications (DDS). A description of the user's database or device files that is entered into the system in a fixed form. The description is then used to create files.

data file. A group of related data records organized in a specific order. A data file can be created by the specification of FILETYPE(*DATA) on the create commands. Contrast with *source file*.

data file utility (DFU). The part of the AS/400 Application Development Tools licensed program that is used to enter, maintain, and display records in a database file.

data stream. All information (data and control commands) sent over a data link usually in a single read or write operation.

data type. A characteristic used for defining data as numeric or character.

database. All the data files stored in the system.

database file. One of several types of the system object type *FILE kept in the system that contains descriptions of how input data is to be presented to a program from internal storage and how output data is to be presented to internal storage from a program. See also *physical file* and *logical file*.

DBCS. See *double-byte character set (DBCS)*.

DBCS conversion. A function of the operating system that allows a display station user to enter alphanumeric data and request that the alphanumeric data be converted to double-byte data.

DBCS-either. Pertaining to a character string that is either SBCS or bracketed DBCS, but not both. Contrast with *DBCS-graphic*, *DBCS-only*, and *DBCS-open*.

DBCS-graphic. Pertaining to a character string in which each character is represented by 2 bytes. The character string does not contain shift-out (SO) and shift-in (SI) characters. Contrast with *DBCS-either*, *DBCS-only*, and *DBCS-open*.

DBCS-only. Pertaining to a character string that is only bracketed DBCS. Contrast with *DBCS-either*, *DBCS-graphic*, and *DBCS-open*.

DBCS-open. Pertaining to a character string that can be a mixture of SBCS and bracketed DBCS. Contrast with *DBCS-either*, *DBCS-graphic*, and *DBCS-only*.

DDS. See *data description specifications (DDS)*.

decimal position. (1) The location of the decimal point in a series of numbers. (2) Numbers to the right of the decimal point. For example, 4.009 has three decimal positions.

default. (1) A value that is automatically supplied or assumed by the system or program when no value is specified by the user. (2) In DDS, the value specified by the user with the DFT or DFTVAL keyword in DDS.

descending sequence. The arrangement of data in order from the highest value to the lowest value, according to the rules for comparing data. Contrast with *ascending sequence*.

DEVD. See *device description*.

device description. An object that contains information describing a particular device or logical unit (LU) that is attached to the system. A device description is a description of the logical connection between two LUs (local and remote locations). The system-recognized identifier for the object type is *DEVD.

device file. One of several types of the system object type *FILE. A device file contains a description of how data is to be presented to a program from a device or how data is to be presented to the device from the program. Devices can be display stations, printers, diskette units, tape units, or remote systems.

DFU. See *data file utility (DFU)*.

digit. Any of the numerals from 0 through 9.

display file. A device file to support a display station.

display screen. The part of the display device, which is similar to a television (TV) picture tube, used to display information entered or received at a display station.

display station. A device that includes a keyboard from which an operator can send information to the system and a display screen on which an operator can see the information sent to or the information received from the system.

double precision. The specification that causes a floating-point value to be stored (internally) in the long format (two computer words). Double precision is known as *long precision* in BASIC. Contrast with *single precision*.

double-byte character. An entity that requires two character bytes.

double-byte character set (DBCS). A set of characters in which each character is represented by 2 bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets. Because each character requires 2 bytes, the typing, displaying, and printing of DBCS characters requires hardware and programs that support DBCS. Four double-byte character sets are supported by the system: Japanese, Korean, Simplified Chinese, and Traditional Chinese. Contrast with *single-byte character set (SBCS)*.

double-byte coded font. In AFP support, a font in which the characters are defined by 2 bytes: the first defining a coded font section, the second defining a code point. Synonymous with double-byte font. Contrast with *single-byte coded font*.

draft. A printed copy of a document that is not yet completed.

duplicate key value. The occurrence of the same value in a key field or in a composite key in more than one record in a file.

dynamic select/omit. Selection and omission of logical file records performed during processing, instead of when the access path (if any) is maintained. Dynamic select/omit may also be used when no keyed access path exists.

edit. To change a numeric field for output by suppressing zeros and inserting commas, periods, currency symbols, the sign status, or other constant information.

edit code. A letter or number indicating that editing should be done according to a defined pattern before a field is displayed or printed. Contrast with *edit word*.

edit description. A description of a user-defined edit code. The system-recognized identifier is *EDTD.

edit word. A user-defined word with a specific format that indicates how editing should be done. Contrast with *edit code*.

electronic overlay. An AFP resource object that is a collection of predefined data, such as lines, shading, text, boxes, or logos, that can be merged with variable data on a page while printing. The system-recognized identifier for the object type is *OVL.

embedded blank. A space between characters within a unit of data.

ending attribute character. For a display file, the character following the last position in a field.

exponent. (1) A number, indicating to which power another number (the base) is to be raised. (2) In floating-point format, an integer constant specifying the power of ten by which the base of the decimal floating-point number is to be multiplied.

expression. In DDS, a pair of values that represents a single parameter value.

externally described data. Data contained in a file for which the fields and the records are described outside of the program (such as with files created by DDS, IDDU, or the SAA Structured Query Language/400 licensed program) that processes the file. Contrast with *program-described data*.

externally described file. A file in which the records and fields are described to the system when the file is created, and used by the program when the file is processed. Contrast with *program-described file*.

field. A group of related bytes (such as name or amount) that is treated as a unit in a record.

field level specifications. In DDS, specifications coded on the same line as a field name or on lines immediately following a field name. See also *file level specifications*, *record level specifications*, *help level specifications*, *join level specifications*, *key field level specifications*, and *select/omit level specification*.

field reference file. A physical file that contains no data, only descriptions of fields.

field selection. A function that uses the state of the option indicators to display or print data when a record format is written.

file. A generic term for the object type that refers to a database file, a device file, or a save file. The system-recognized identifier for the object type is *FILE.

file level specifications. In DDS, specifications coded on the lines before the first record format name. See also *field level specifications*, *key field level specifications*, *record level specifications*, *join level specifications*, *select/omit level specifications*, and *help level specifications*.

fixed currency symbol. A currency symbol that appears in the far left position of an edited field. Contrast with *floating currency symbol*.

fixed-length. Pertaining to a characteristic of a file in which all of the records are the same length, or pertaining to a characteristic of a field on a display that is of a defined length.

floating currency symbol. A currency symbol that appears immediately to the left of the far left position in an edited field. Contrast with *fixed currency symbol*.

floating-point. A method of encoding real numbers within the limits of finite precision available on computers.

fold. To continue data on the next line. Contrast with *truncate*.

folder. A directory for documents. A folder is used to group related documents and to find documents by name. The system-recognized identifier for the object type is *FLR. See also *document library object*. Compare with *library*.

font. An assortment of characters of a given size and type style.

font character set. In AFP support, a font file that contains the raster patterns, identifiers, and descriptions of characters.

font ID. A number that identifies the character style and size for certain printers.

function key. A keyboard key that allows the user to select keyboard functions or programmer functions. Contrast with *character key*.

GDF file. See *graphics data format (GDF) file*.

graphic character set. A set of graphic characters in a code page.

graphic character-set ID. A 5-digit registered identifier used to specify a graphic character set. The graphic character-set ID is the first part of the QCHRID system value or the CHRID parameter value. See also *code-page ID*.

graphics data format (GDF) file. A picture definition in a coded order format used internally by the GDDM function and, optionally, providing the user with a lower-level programming interface than the GDDM application programming interface.

help level specifications. In a display file, data description specifications coded between the record and field level that define areas on the screen and associate help information with those areas. See also *file level specifications*, *field level specifications*, *join level specifications*, *key field level specifications*, *record level specifications*, and *select/omit level specifications*.

hexadecimal. Pertaining to a numbering system with a base of 16.

hidden field. A field in a display file that is passed to and from the program but is not sent to the display.

high-level language (HLL). A programming language, such as RPG, BASIC, PL/I, Pascal, COBOL, and C used to write computer programs.

Hiragana. A graphic character set that is used to write Japanese words phonetically. This set of characters is used as word endings when writing in Kanji. Contrast with *Katakana*.

human readable interpretation (HRI). In Advanced Function Printing Utilities/400, the characters printed above or below a bar code. These characters are read by people, not by scanners.

IBM Operating System/400 Version 2 (OS/400). Pertaining to the IBM licensed program that can be used as the operating system for the AS/400 system.

ICF. See *intersystem communications function (ICF)*.

ICF file. A device file that allows a program on one system to communicate with a program on another system. There can be one or more sessions with the same or different communications devices at the same time.

IGC. Abbreviation used in commands and keywords to represent double-byte character set functions.

ILE. See *Integrated Language Environment (ILE)*.

implicit. Capable of being understood from something else, though unexpressed.

inactive record. An inactive subfile record or any record format that is not currently shown on a display. See also *inactive subfile record*. Contrast with *active record*.

inactive subfile record. A subfile record that either was not added to a subfile by a write operation or was described as inactive by the data description specification (DDS) keywords SFLINZ and SFLRNA. Contrast with *active subfile record*.

indicator. (1) A 1-character or 2-character code that is used by a program to test a field or record or to tell when certain operations are to be performed. (2) An internal switch used by a program to remember when a certain event occurs and what to do when that event occurs.

input field. A field specified in a display file or database file that is reserved for information supplied by a user. Contrast with *output field*.

integer. A positive or negative whole number.

Integrated Language Environment (ILE). Pertaining to a set of constructs and interfaces that provides a common run-time environment and run-time bindable application program interfaces (APIs) for all ILE-conforming high-level languages.

Intelligent Printer Data Stream (IPDS). Pertaining to an all-points-addressable data stream that allows users to position text, images, and graphics at any defined point on a printed page.

intersystem communications function (ICF). A function of the operating system that allows a program to communicate interactively with another program or system.

IPDS. See *Intelligent Printer Data Stream (IPDS)*.

job date. The date associated with a job. The job date usually assumes the system date, but it can be changed by the user. See also *creation date* and *system date*.

join. An operation that combines data from two or more files using specified fields.

join field. A comparison field that identifies records from two files to be combined into one record.

join level specifications. For a join logical file, data description specifications coded between the record and field level that define how to join two physical files. See also *file level specifications*, *field level specifications*, *key field level specifications*, *help level specifications*, *record level specifications*, and *select/omit level specifications*.

join logical file. A logical file that combines (in one record format) fields from two or more physical files. See also *logical file*.

justify. To adjust text so that line endings are even. See *left-justify* and *right-justify*.

Kanji. Characters originating from the Chinese characters used in the Japanese written language.

Katakana. A graphic Japanese character set that is used to write non-Japanese words phonetically in Japanese. Contrast with *Hiragana*.

key field. A field used to arrange the records of a particular type within a file member.

key field level specifications. Data description specifications coded on the lines following the last field specification. Key field level specifications are permitted only for physical files or logical files. See also *field level specifications*, *file level specifications*, *record level specifications*, *help level specifications*, *join level specifications*, and *select/omit level specifications*.

keyboard shift. In DDS, a characteristic that can be specified for a field in a display file that automatically shifts the display station keyboard to control what the display station user can enter into the field. In IDDU and DDS, the keyboard shift can also be specified in database files, but only applies when these fields are referred to in a display file.

keyed sequence access path. An access path to a database file that is arranged according to the contents of key fields contained in the individual records. See also *arrival sequence access path* and *access path*.

keyword. In DDS, a name that identifies a function.

keyword functions. The result of processing DDS keywords in a record format specified on an operation. See also *operation*.

library. A system object that serves as a directory to other objects. A library groups related objects, and allows the user to find objects by name. The system-recognized identifier for the object type is *LIB. Compare with *folder* and *document library*.

library list. A list that indicates which libraries are to be searched and the order in which they are to be searched. The system-recognized identifier is *LIBL.

library name. A user-defined word that names a library.

lines per inch (lpi). The number of characters that can be printed vertically within an inch.

local. Pertaining to a device or system that is connected directly to your system or a file that is read directly from your system, without the use of a communications line. Contrast with *remote*.

local work station. A work station that is connected directly to the system without a need for data transmission functions. Contrast with *remote work station*.

locked keyboard. A keyboard condition where the display station accepts no input.

logical file. A description of how data is to be presented to or received from a program. This type of database file contains no data, but it defines record formats for one or more physical files. See also *join logical file* and *database file*. Contrast with *physical file*.

logical file member. A named logical grouping of data records from one or more physical file members. See also *member*.

lpi. See *lines per inch (lpi)*.

magnetic stripe reader. A device, attached to a display station, that reads data from a magnetic stripe on a badge before allowing an operator to sign on.

mandatory entry field. A field in which an operator must enter at least one character.

mandatory fill field. A field that an operator must leave blank, or must fill in completely.

MDT. See *modified data tag (MDT)*.

member. Different sets of data, each with the same format, within one database file. See also *source member*.

message file. An object that contains message descriptions. The system-recognized identifier for the object type is *MSGF.

message identifier. A seven-character code that identifies a predefined message, and is used to get the message description from a message file. See *predefined message*.

message line. An area on the display where messages are displayed.

message reference key. A key assigned to every message on a message waiting line. This key is used to remove a message from a message waiting line, to receive a message, and to reply to a message.

message subfile. A subfile where the records are messages from a program message queue.

modified data tag (MDT). An indicator, associated with each input or output field in a displayed record, that is automatically set on when data is typed into the field. The modified data tag is maintained by the display file and can be used by the program using the file.

modulus. In BASIC and communications, a number, such as a positive integer, in a relationship that divides the difference between two related numbers without leaving a remainder. For example, 9 and 4 have a modulus of 5 (9 - 4 = 5; 4 - 9 = -5; and 5 divides both 5 and -5 without leaving a remainder).

modulus 10 checking/modulus 11 checking. (1) A method for verifying data. (2) Formulas used to calculate the check digit for a self-check field.

negative response. In data communications, a reply indicating that data was not received correctly or that a command was incorrect or unacceptable. Contrast with *positive response*. See also *exception response*.

numeric character. Any one of the digits 0 through 9.

numeric constant. The actual numeric value to be used in processing, instead of the name of a field containing the data. A numeric constant can contain any of the numeric digits 0 through 9, a sign (plus or minus), and a decimal point. See also *floating-point constant*. Contrast with *character constant*.

numeric field. An area that is reserved for a particular unit of information and that can contain only the digits 0 through 9. Contrast with *character field*.

object. A named storage space that consists of a set of characteristics that describe itself and, in some cases, data. An object is anything that exists in and occupies space in storage and on which operations can be performed. Some examples of objects are programs, files, libraries, and folders.

object name. The name of an object. Contrast with *qualified name*.

online information. Information on the display screen that explains displays, messages, and programs. See also *textual data*.

open. The function that connects an object of type *FILE to a program for processing. Contrast with *close*.

operation. The result of processing statements in a high-level language. See also *keyword functions*.

option indicator. A 1-character field that is passed with an output data record from a program to the system that is used to control the output function, such as controlling which fields in the record are displayed.

OS/400. See *IBM Operating System/400 Version 2 (OS/400)*.

output. Information or data received from a computer that is shown on a display, printed on the printer, or stored on disk, diskette, or tape.

output field. A field specified in a display file, database file, printer file, or ICF file that is reserved for the information processed by a program. Contrast with *input field*.

output/input field. A field specified in a database, display, or ICF file that can be used for both the information supplied to the program and the information received from the program during processing. See also *input field* and *output field*.

overflow. The condition that occurs when the last line specified as the overflow line to be printed on a page has been passed.

overlapping fields. Fields in the same display or printer record that occupy the same positions on the display or page. Option indicators can be used to select which of the overlapping fields is to be displayed or printed.

overlay. For AFP support, see *electronic overlay*.

page. (1) Each group of records in a subfile that are displayed at the same time. (2) To move information up or down on the display.

page down. To move to the information below the information currently shown on the display. Contrast with *page up*.

page up. To move to the information above the information currently shown on the display. Contrast with *page down*.

parameter. A value supplied to a command or program that is used either as input or to control the actions of the command or program.

physical file. A description of how data is to be presented to or received from a program and how data is actually stored in the database. A physical file contains one record format and one or more members. See also *database file*. Contrast with *logical file*.

physical file member. A named subset of the data records in a physical file. See also *member*.

PIP. See *problem isolation procedure (PIP)* or *program initialization parameters (PIP)*.

positive response. In SNA, a response indicating that a request arrived and was successfully received and processed. Contrast with *negative response*. See also *definite response*.

primary file. In the DDS for a join logical file, the first physical file specified on the JFILE keyword. Contrast with *secondary file*.

printer file. A device file that determines what attributes printed output will have. A particular printer may or may not support all of the attributes specified in a printer file.

program initialization parameters (PIP). The initial parameter value(s) passed to a target program as input or used to set up the process environment.

program message queue. An object used to hold messages that are sent between program calls of a routing step. The program message queue is part of the job message queue.

program-described data. Data contained in a file for which the fields in the records are described in the program that processes the file. Contrast with *externally described data*.

program-described file. A file for which the fields in the records are described only in the programs that process the file. To the operating system, the record appears as a character string. Contrast with *externally described file*.

protected field. A field on a display in which a user cannot add, change, or delete data.

qualified name. The name of the library containing the object and the name of the object. Contrast with *object name*.

read operation. An input operation that obtains data from a file or device and passes it to a program.

read-from-invited-program-devices operation. An input operation that waits for input from any one of the invited program devices for a user-specified time. Contrast with *read-from-one-program-device operation*.

read-from-one-program-device operation. An input operation that will not complete until the specified device has responded with input. Contrast with *read-from-invited-program-devices operation*.

record. A group of related data, words, or fields treated as a unit, such as one name, address, and telephone number.

record format. A named part of a file that identifies records of a specified record format description.

record level specifications. Data description specifications coded on the same line as a record format name or on lines immediately following a record format name (until the first field is specified). See also *field level specifications*, *file level specifications*, *key field level specifications*, *help level specifications*, *join level specifications*, and *select/omit level specifications*.

relational operator. The reserved words or symbols used to express a relational condition or a relational expression.

relative file number. In the DDS for a join logical file, a sequential number assigned to a physical file based on the position of that file on the JFILE keyword specification.

relative record number. A number that specifies the relationship between the location of a record and the beginning of a database file, member, or subfile. For example, the first record in a database file, member, or subfile has a relative record number of 1.

remote. Pertaining to a device, system, or file that is connected to another device, system, or file through a communications line. Contrast with *local*.

remote work station. A work station that is connected to the system by data communications. Contrast with *local work station*.

response indicator. A 1-character field passed with an input record from the system to a program to provide information about the data record or actions taken by the work station user.

reverse image. Text that appears on the display in the opposite color (for example, black on green instead of green on black).

screen design aid (SDA). A function of the AS/400 Application Development Tools licensed program that helps the user design, create, and maintain displays and menus.

SCS. See *SNA character string (SCS)*.

SDA. See *screen design aid (SDA)*.

secondary file. In the DDS for a join logical file, any physical file, other than the first physical file, that is specified on the JFILE keyword. Contrast with *primary file*.

select/omit field. A field in a logical file record format whose value is tested by the system to determine if records including that field are to be used. The test is a comparison with a constant, the contents of another field, a range of values, or a list of values; and the record is either selected or omitted as a result of the test. See also *dynamic select/omit*.

select/omit level specifications. Data description specifications coded on the lines following the last key-field specification. These specifications are permitted only in a logical file. See also *field level specifications*, *file level specifications*, *key field level specifications*, *record level specifications*, *help level specifications*, and *join level specifications*.

self-check digit. The far right digit of a self-check field.

self-check field. A field, such as an account number, consisting of a base number and a self-check digit. For data entry applications, the operator-entered self-check number is compared with the self-check number calculated by the system.

sequence number. The number of a record that identifies the record within the source member.

session. (1) The length of time that starts when a user signs on at a display station and ends when the user signs off. (2) In communications, the logical connection by which a program or device can communicate with a program or device at a remote location. See also *conversation* and *transaction*.

SEU. See *source entry utility (SEU)*.

shared file. A file whose open data path can be shared between two or more programs processing in the same job. See *open data path (ODP)*.

shift control character. See *shift-in character* and *shift-out character*.

shift-in character. A control character (hex 0F) that indicates the end of a string of double-byte characters. Contrast with *shift-out character*.

shift-out character. A control character (hex 0E) that indicates the start of a string of double-byte characters. Contrast with *shift-in character*.

significand. In binary floating-point format, the part of a number that contains the whole number and fraction.

significant digit. Any number of a series of numbers that follows the farthest left number, that is not a zero, and that is within the accuracy allowed.

single precision. The specification that causes the floating-point value to be stored (internally) in the short format. Contrast with *double precision*.

single-byte character set (SBCS). A character set in which each character is represented by a one-byte code. Contrast with *double-byte character set (DBCS)*.

single-byte coded font. In AFP support, a font in which the characters are defined by a 1-byte code point. A single-byte coded font has only one coded font section. Synonymous with single-byte font. Contrast with *double-byte coded font*.

SNA. See *Systems Network Architecture (SNA)*.

SNA character string (SCS). In SNA, a data stream composed of EBCDIC controls, optionally intermixed with end-user data, that is carried within a request/response unit.

source entry utility (SEU). A function of the AS/400 Application Development Tools licensed program that is used to create and change source members.

source file. A file of programming code that is not compiled into machine language. A source file can be created by the specification of FILETYPE(*SRC) on the Create command. A source file can contain source statements for such items as high-level language programs and data description specifications. Contrast with *data file*.

source member. A member of a database source file that contains source statements, such as AS/400 BASIC, AS/400 PL/I, SAA C/400, SAA COBOL/400, SAA FORTRAN/400, SAA RPG/400, or DDS statements. See also *member*.

source statement. A statement written in symbols of a programming language. For example, AS/400 BASIC, AS/400 PL/I, SAA COBOL/400, SAA RPG/400, and DDS statements are source statements.

special character. A character other than a digit, a letter, or $, #, @, ., or _. For example, *, +, and % are special characters.

subfile. A group of records of the same record format that can be displayed at the same time at a display station. The system sends the entire group of records to the display in a single operation and receives the group from the display in another operation.

subfile control record format. One of two record formats required to define a subfile in DDS. The subfile control record format describes the size of the subfile and the size of the subfile page, and is used by the program to write the subfile to and read the subfile from the display. See also *subfile record format*.

subfile record format. One of two record formats required to define a subfile in DDS. The subfile record format defines the fields in a subfile record and is used by the program to perform input, output, and update operations to the subfile. See also *subfile control record format*.

substring. A part of a character string.

syntax checking. A function of the system, a compiler, the BASIC interpreter, the SAA AD/Cycle CoOperative Development Environment/400 syntax checker, or SEU that checks individual statements for errors in their structure.

system date. The date assigned in the system values when the system is started. See also *creation date* and *job date*.

system value. Control information for the operation of certain parts of the system. A user can change the system value to define his working environment. System date and library list are examples of system values. Contrast with *network attribute*.

Systems Application Architecture (SAA). Pertaining to an architecture defining a set of rules for designing a common user interface, programming interface, application programs, and communications support for strategic operating systems such as the OS/2, OS/400, VM, and MVS operating systems.

Systems Network Architecture (SNA). In IBM networks, the description of the layered logical structure, formats, protocols, and operational sequences that are used for transmitting information units through networks, as well as controlling the configuration and operation of networks.

time. A three-part value or data type that designates a time of day in hours, minutes, and seconds.

time stamp. (1) To apply the current system time. (2) The value of an object that indicates the system time at some critical point in the object's history.

timestamp. In AS/400 database support, a seven-part value or data type that consists of a date and time, expressed in years, months, days, hours, minutes, seconds, and microseconds.

transaction. In communications, an exchange between a program on a local system and a program on a remote system that accomplishes a particular action or result. See also *conversation* and *session*.

translation table. An object that contains a set of hexadecimal characters used to translate one or more characters of data. The table can be used for translation of data being moved between the system and a device. For example, data stored in one national language character set may need to be displayed or entered on display devices that support a different national language character set. The table can also be used to specify an alternative collating sequence or field

translation functions. The system-recognized identifier for the object type is *TBL. See also *table*.

truncate. (1) To cut off data that cannot be printed or displayed in the line width specified or available. Contrast with *fold*. (2) To cut off data that does not fit in the specified field length in a field definition.

unprotected field. A displayed field for which operators can enter, change, or delete data.

update operation. An I/O process that changes the data in a record.

user profile. An object with a unique name that contains the user's password, the list of special authorities assigned to a user, and the objects the user owns. The system-recognized identifier for the object type is *USRPRF.

user-defined edit code. A number (5 through 9) indicating that editing should be done on a numeric output field according to a pattern predefined to the system program. User-defined edit codes can take the place of edit words, so that repetitive coding of the same edit word is not necessary.

validity checking. To verify the contents of a field.

variable. A name used to represent data whose value can be changed while the program is running by referring to the name of the variable.

variable-length. Pertaining to a characteristic of a file in which the records can be of varying length, or pertaining to a characteristic of a field on a display that can be of varying length. Contrast with *fixed-length*.

work station. A device used to transmit information to or receive information from a computer, for example, a display station or printer.

write operation. An output operation that sends a processed record to an output device or output file.

zero suppression. The substitution of blanks for leading zeros in a number. For example, 00057 becomes 57 with zero suppression.

Bibliography

The following manuals contain information you may need. The manuals are listed with their full title and base order number. When these manuals are referred to in this manual, The short title is used.

The related printed information includes the following:

- *Advanced Backup and Recovery Guide*, SC41-8079, describes how to make copies of the system and database files for safe keeping.

  Short Title: *Advanced Backup and Recovery Guide*.

- *Advanced Function Printing: Data Stream Reference*, S544-3202, provides information on the Advanced Function Printing data stream.

  Short Title: *Advanced Function Printing: Data Stream Reference*.

- *Application Development Tools: Screen Design Aid User's Guide and Reference*, SC09-1340, provides the application programmer, system programmer, or data processing manager with information about using the Application Development Tools screen design aid (SDA) to design, create, and maintain display formats and menus.

  Short Title: *SDA User's Guide and Reference*.

- *Application Development Tools: Source Entry Utility User's Guide and Reference*, SC09-1338, provides the application programmer or system programmer with information about using the Application Development Tools source entry utility (SEU) to create and edit source members.

  Short Title: *SEU User's Guide and Reference*.

- *Communications: Intersystem Communications Function Programmer's Guide*, SC41-9590, provides the application programmer with the information needed to write application programs that use AS/400 communications and the ICF file.

  Short Title: *ICF Programmer's Guide*.

- *Data Management Guide*, SC41-9658, provides information to help the application programmer manage key aspects of the system.

  Short Title: *Data Management Guide*.

- *Database Guide*, SC41-9659, describes database concepts used to create, describe, and manipulate database files on the system. This manual also describes how to define files to the system using DDS keywords.

  Short Title: *Database Guide*

- *Distributed Data Management Guide*, SC41-9600, contains information about remote file processing.

  Short Title: *DDM Guide*.

- *Guide to Programming Application and Help Displays*, SC41-0011, describes the use of display devices by application programs.

  Short Title: *Guide to Programming Displays*.

- *Guide to Programming for Printing*, SC41-8194, provides information to help the programmer manage key aspects of the system. For example, it describes how to use printer files.

  Short Title: *Guide to Programming for Printing*.

- *IBM Personal Computer Enhanced 5250 Emulation Program Version 2.12 Technical Reference*, GS570-222-02, provides the information on the 5250 Emulation Program.

  Short Title: *IBM Personal Computer Enhanced 5250 Emulation Program Technical Reference*.

- *Intelligent Printer Data Stream Reference*, 5544-3417, provides information on bar codes and valid check digits.

  Short Title: *Intelligent Printer Data Stream Reference*.

- *Languages: Systems Application Architecture* AD/Cycle* RPG/400* Reference*, SC09-1349, describes how to design, code, enter, compile, test, and run RPG/400 programs.

  Short Title: *RPG/400* Reference*.

- *National Language Support Planning Guide*, GC41-9877, provides the data processing manager, system operator and manager, application programmer, end user, IBM marketing representative, and system engineer with information on how to use and understand the national language support function on the AS/400 system.

  Short Title: *National Language Support Planning Guide*.

- *Programming: Control Language Programmer's Guide*, SC41-8077, provides a wide-range discussion of programming topics.

  Short Title: *CL Programmer's Guide*.

- *Programming: Control Language Reference*, SC41-0030, describes control language syntax, commands, and command parameters.

  Short Title: *CL Reference*.

- *Programming: Reference Summary*, SX41-0028, provides the system programmer with quick reference information.

© Copyright IBM Corp. 1991, 1993

Short Title: *Programming Reference Summary.*

- *Programming: Work Management Guide*, SC41-8078, provides programmers with information about how to create and change a work management environment.

  Short Title: *Work Management Guide.*

- *System Operator's Guide*, SC41-8082, describes how to operate work stations and how to use the function keys to enter commands.

  Short Title: *Operator's Guide.*

- *Systems Application Architecture* OfficeVision/400*: *Using OfficeVision/400*, SC41-9616, provides the office user with procedural and reference information.

  Short Title: *Using OfficeVision/400**.

- *Systems Application Architecture* OfficeVision/400*: *Using OfficeVision/400 Word Processing*, SC41-9618, provides the office user with detailed information on how to use the word processing functions of OfficeVision/400.

  Short Title: *Using OfficeVision/400* Word Processing.*

Index

Special Characters

***NONE key field 2-9, 2-13**

Numerics

3270 remote attachment D-1

A

Absolute Value (ABSVAL) keyword
physical and logical files
description 2-31
ABSVAL (Absolute Value) keyword
physical and logical files
description 2-31
ALARM (Audible Alarm) keyword
display files
description 3-28
ALIAS (Alternative Name) keyword
display files
description 3-28
ICF files
description 5-7
physical and logical files
description 2-32
printer files
description 4-11
ALL (All) keyword
logical files
description 2-33
Allow Graphics (ALWGPH) keyword
display files
description 3-32
Allow Null (ALWNULL) keyword
physical files
description 2-34
Allow Roll (ALWROL) keyword
display files
description 3-33
Allow Write (ALWWRT) keyword
ICF files
description 5-8
Alphanumeric-to-DBCS Conversion (IGCANKCNV) keyword E-2, E-18
Alternative Collating Sequence (ALTSEQ) keyword
physical and logical files
description 2-33
Alternative Help Key (ALTHELP) keyword
display files
description 3-29
Alternative Name (ALIAS) keyword
display files
description 3-28
Alternative Name (ALIAS) keyword *(continued)*
ICF files
description 5-7
physical and logical files
description 2-32
printer files
description 4-11
Alternative Page Up/Down (ALTPAGEUP/ALTPAGEDWN) keyword
display files
description 3-30
Alternative Record Name (ALTNAME) keyword
display files
description 3-30, F-4
ALTHELP (Alternative Help Key) keyword
display files
description 3-29
ALTNAME (Alternative Record Name) keyword
display files
description 3-30, F-4
ALTPAGEUP/ALTPAGEDWN (Alternative Page Up/Down) keyword
display files
description 3-30
ALTSEQ (Alternative Collating Sequence) keyword
physical and logical files
description 2-33
ALWGPH (Allow Graphics) keyword
display files
description 3-32
ALWNULL (Allow Null) keyword
physical files
description 2-34
ALWROL (Allow Roll) keyword
display files
description 3-33
ALWWRT (Allow Write) keyword
ICF files
description 5-8
assigning
menu-bar switch key 3-160
menu-cancel key 3-161
ASSUME (Assume) keyword
display files
description 3-35
Audible Alarm (ALARM) keyword
display files
description 3-28
AUTO (Auto) keyword
display files
description 3-36

© Copyright IBM Corp. 1991, 1993

B

Bar Code (BARCODE) keyword
printer files
description 4-12
BARCODE (Bar Code) keyword
printer files
description 4-12
Blank Fold (BLKFOLD) keyword
display files
description 3-39
printer files
description 4-15
BLANKS (Blanks) keyword
display files
description 3-36
BLINK (Blink) keyword
display files
description 3-38
example B-7
BLKFOLD (Blank Fold) keyword
display files
description 3-39
printer files
description 4-15
BOX (Box) keyword
printer files
description 4-16

C

CANCEL (Cancel) keyword
ICF files
description 5-8
Cancel Invite (CNLINVITE) keyword
ICF files
description 5-9
CAnn (Command Attention) keyword
display files
description 3-39
inquiry display example B-7
CCSID (Character Code Set Identifier) keyword
physical files
description 2-35
CDEFNT (Coded Font Name) keyword
printer files
description 4-18
CFnn (Command Function) keyword
display files
description 3-41
CHANGE (Change) keyword
display files
description 3-43, F-2
Change Display File (CHGDSPF) command 3-115, 3-119
Change Input Default (CHGINPDFT) keyword
display files
description 3-62
Change Logical File (CHGLF) command 2-48
Change Printer File (CHGPRTF) command 4-23
changing
display file 3-115, 3-119
logical file 2-48
printer file 4-23
Character Code Set Identifier (CCSID) keyword
physical files
description 2-35
Character Identifier (CHRID) keyword
display files
description 3-67
printer files
description 4-19
character set CODE128 G-1
Character Size (CHRSIZ) keyword
double-byte character set considerations E-2, E-14
printer files
description 4-20
character string
DBCS characters E-3
Characters Per Inch (CPI) keyword
printer files
description 4-22
CHCACCEL (Choice Accelerator Text) keyword
display files
description 3-44
CHCAVAIL (Choice Color/Display Attribute when Available)
display files
description 3-45, 3-50
CHCAVAIL (Choice Color/Display Attributes when Available) keyword 3-45
CHCCTL (Choice Control) keyword 3-47
display files
description 3-47
CHCSLT (Choice Color/Display Attribute when Selected) 3-49
display files
description 3-49
CHCUNAVAIL (Choice Color/Display Attribute when Unavailable)
CHCUNAVAIL (Choice Color/Display Attribute When Unavailable) keyword 3-50
CHECK (Check) keyword
display files
AB (allow blanks) 3-53
cursor control 3-60
description 3-52
ER (end of record) 3-58
FE (field exit check) 3-58
keyboard control 3-57
LC (lower case) 3-58
M10/M10F (modulus 10 or 10F) 3-54

CHECK (Check) keyword *(continued)*
display files *(continued)*
M11/M11F (modulus 11 or 11F) 3-54
ME (mandatory enter) 3-53
MF (mandatory fill) 3-54
RB (right adjust with blank fill) 3-59
RL (right to left) 3-61
RLTB (right to left, top to bottom) 3-62
RZ (right adjust with zero fill) 3-59
validity checking 3-52
VN (validate name) 3-56
VNE (validate name extended) 3-56
inquiry display example B-7
physical and logical files
description 2-35
Check Message Identifier (CHKMSGID) keyword
display files
description 3-64
physical and logical files
description 2-36
CHGDSPF (Change Display File) command 3-115, 3-119
CHGINPDFT (Change Input Default) keyword
display files
description 3-62
CHGLF (Change Logical File) command 2-48
CHGPRTF (Change Printer File) command 4-23
CHKMSGID (Check Message Identifier) keyword
display files
description 3-64
physical and logical files
description 2-36
CHOICE (Selection Field Choice) keyword 3-65
display files
description 3-65
Choice Accelerator Text (CHCACCEL) keyword
display files
description 3-44
Choice Color/Display Attribute when Available (CHCAVAIL) keyword
display files
description 3-45, 3-50
Choice Color/Display Attribute when Selected (CHCSLT) keyword 3-49
display files
description 3-49
Choice Color/Display Attribute when Unavailable (CHCUNAVAIL) keyword 3-50
Choice Color/Display Attributes when Available (CHCAVAIL) keyword 3-45
Choice Control (CHCCTL) keyword 3-47
display files
description 3-47
choice field
controlling 3-47
CHRID (Character Identifier) keyword
display files
description 3-67
printer files
description 4-19
CHRSIZ (Character Size) keyword
double-byte character set considerations E-2, E-14
printer files
description 4-20
CLEAR (Clear) keyword
display files
description 3-67
Clear Line (CLRL) keyword
display files
description 3-68
CLRL (Clear Line) keyword
display files
description 3-68
CMP (Comparison) keyword
display files
description 3-70
physical and logical files
description 2-37
CNLINVITE (Cancel Invite) keyword
ICF files
description 5-9
CNTFLD (Continued-Entry Field) keyword 3-71
display files
description 3-71
code point 4-32
CODE128 character set G-1
Coded Font Name (CDEFNT) keyword
printer files
description 4-18
COLHDG (Column Heading) keyword
field reference file example B-1
physical and logical files
description 2-37
COLOR (Color) keyword
display files
description 3-72
on color displays 3-74
with DSPATR 3-73
printer files
description 4-21
color/display attribute
available 3-45
selected 3-49
unavailable 3-50
Column Heading (COLHDG) keyword
field reference file example B-1
physical and logical files
description 2-37
Command Attention (CAnn) keyword
display files
description 3-39

Command Attention (CAnn) keyword *(continued)*
inquiry display example B-7
Command Function (CFnn) keyword
display files
description 3-41
command, CL
Change Display File (CHGDSPF) command 3-115, 3-119
Change Logical File (CHGLF) command 2-48
Change Printer File (CHGPRTF) command 4-23
CHGDSPF (Change Display File) command 3-115, 3-119
CHGLF (Change Logical File) command 2-48
CHGPRTF (Change Printer File) command 4-23
Copy File (CPYF) command 2-46
CPYF (Copy File) command 2-46
Create Display File (CRTDSPF) command 3-115, 3-119
Create Edit Description (CRTEDTD) command 3-102, 4-49
Create Logical File (CRTLF) command 2-48
Create Physical File (CRTPF) command 2-1
Create Printer File (CRTPRTF) command 4-23
CRTDSPF (Create Display File) command 3-115, 3-119
CRTEDTD (Create Edit Description) command 3-102, 4-49
CRTLF (Create Logical File) command 2-48
CRTPF (Create Physical File) command 2-1
CRTPRTF (Create Printer File) command 4-23
Initialize Physical File Member (INZPFM) command 2-46
INZPFM (Initialize Physical File Member) command 2-46
Override with Printer File (OVRPRTF) command 4-23
OVRPRTF (Override with Printer File) command 4-23
comment
display files 3-2
ICF files 5-2
logical files 2-5
physical files 2-5
printer files 4-2
COMP (Comparison) keyword
display files
description 3-76
physical and logical files
description 2-38
Comparison (CMP) keyword
display files
description 3-70
physical and logical files
description 2-37
Comparison (COMP) keyword
display files
description 3-76
Comparison (COMP) keyword *(continued)*
physical and logical files
description 2-38
compiler example B-24
CONCAT (Concatenate) keyword
data files E-5
logical files
description 2-40
Concatenate (CONCAT) keyword
data files E-5
logical files
description 2-40
conditioning
display files 3-2
ICF files 5-2
logical files 2-5
physical files 2-5
printer files 4-3
CONFIRM (Confirm) keyword
ICF files
description 5-10
example B-19
constant field
example B-7
printer files 4-4
continued-entry field
creating 3-71
menu-bar separator 3-158
Continued-Entry Field (CNTFLD) keyword 3-71
display files
description 3-71
Control Data (CTLDTA) keyword
ICF files
description 5-10
controlling
choice field 3-47
Convert Data (CVTDTA) keyword
printer files
description 4-26
Copy File (CPYF) command 2-46
copying
file 2-46
CPI (Characters Per Inch) keyword
printer files
description 4-22
CPYF (Copy File) command 2-46
Create Display File (CRTDSPF) command 3-115, 3-119
Create Edit Description (CRTEDTD) command 3-102, 4-49
Create Logical File (CRTLF) command 2-48
Create Physical File (CRTPF) command 2-1
Create Printer File (CRTPRTF) command 4-23
creating
continued-entry field 3-71
cursor-progression field 3-116

creating *(continued)*
display file 3-115, 3-119
edit description 3-102, 4-49
edit mask 3-104
entry field attribute 3-108
files using DDS
completing the form 1-1
creating files 1-3
entering source statements 1-3
logical file 2-48
menu bar 3-151
menu-bar choice 3-152
menu-bar separator 3-158
multiple-choice selection field 3-149
physical file 2-1
printer file 4-23
pull-down menu 3-177
subfile cursor progression 3-194
window 3-238
CRTDSPF (Create Display File) command 3-115, 3-119
CRTEDTD (Create Edit Description) command 3-102, 4-49
CRTLF (Create Logical File) command 2-48
CRTPF (Create Physical File) command 2-1
CRTPRTF (Create Printer File) command 4-23
CSRLOC (Cursor Location) keyword
display files
description 3-77
CTLDTA (Control Data) keyword
ICF files
description 5-10
cursor control 3-60
Cursor Location (CSRLOC) keyword
display files
description 3-77
cursor progression
field 3-116
Cursor Progression Field (FLDCSRPRG) keyword 3-116
display files
description 3-116
CVTDTA (Convert Data) keyword
printer files
description 4-26

D

data description specifications (DDS)
CHCAVAIL (Choice Color/Display Attributes when Available) keyword 3-45
CHCCTL (Choice Control) keyword 3-47
CHCSLT (Choice Color/Display Attribute when Selected) 3-49
CHCUNAVAIL (Choice Color/Display Attribute When Unavailable) keyword 3-50
data description specifications (DDS) *(continued)*
CHOICE (Selection Field Choice) keyword 3-65
CNTFLD (Continued-Entry Field) keyword 3-71
EDTMSK (Edit mask) keyword 3-104
ENTFLDATR (Entry Field Attribute) keyword 3-108
FLDCSRPRG (Cursor Progression Field) keyword 3-116
HLPARA (Help Area) keyword 3-121
HLPID (Help Identifier) keyword 3-132
MLTCHCFLD (Multiple-Choice Selection Field) keyword 3-149
MNUBAR (Menu Bar) keyword 3-151
MNUBARCHC (Menu-Bar Choice) keyword 3-152
MNUBARDSP (Menu-Bar Display) keyword 3-156
MNUBARSEP (Menu-Bar Separator) keyword 3-158
MNUBARSW (Menu-Bar Switch Key) keyword 3-160
MNUCNL (Menu-Cancel Key) keyword 3-161
naming conventions 1-6
printouts E-4
PULLDOWN (Pull-Down Menu) keyword 3-177
SFLCSRPRG (Subfile Cursor Progression) keyword 3-194
SNGCHCFLD (Single-Choice Selection Field) keyword 3-226
WINDOW (Window) keyword 3-238
data type
converting
between zoned decimal and character 2-26
between zoned decimal and hexadecimal 2-26
floating point to binary 2-27
floating point to packed decimal 2-27
floating point to zoned decimal 2-27
numeric types 2-26
when concatenating fields 2-27
when substringing fields 2-27
display files 3-10
ICF files 5-6
logical files 2-25
physical files 2-25
printer files 4-7
DATE (Date) keyword
display files
description 3-78
printer files
description 4-29
Date Format (DATFMT) keyword
physical and logical files
description 2-43
Date Separator (DATSEP) keyword
physical and logical files
description 2-44
DATFMT (Date Format) keyword
physical and logical files
description 2-43

DATSEP (Date Separator) keyword
physical and logical files
description 2-44
DBCS (double-byte character set)
character strings
entering E-3
considerations E-1
database files
description E-4
display files
description E-8
IGCALTTYP (DBCS Alternative Data Type) E-2, E-10
IGCCNV (DBCS Conversion) E-2, E-11
ICF files
description E-23
printer files
description E-13
DFNLIN (Define Line) E-2, E-15
IGCALTTYP (DBCS Alternative Data Type) E-2, E-17
IGCANKCNV (Alphanumeric-to-DBCS Conversion) E-2, E-18
IGCCDEFNT (DBCS Coded Font) E-2, E-20
IGCCHRRTT (DBCS Character Rotation) E-2, E-20
DBCS Alternative Data Type (IGCALTTYP) keyword
description E-2
display files E-10
printer files E-17
DBCS Character Rotation (IGCCHRRTT) keyword E-2, E-20
DBCS Coded Font (IGCCDEFNT) keyword E-2, E-20
DBCS Conversion (IGCCNV) keyword E-2, E-11
DDS (data description specifications)
CHCAVAIL (Choice Color/Display Attributes when Available) keyword 3-45
CHCCTL (Choice Control) keyword 3-47
CHCSLT (Choice Color/Display Attribute when Selected) 3-49
CHCUNAVAIL (Choice Color/Display Attribute When Unavailable) keyword 3-50
CHOICE (Selection Field Choice) keyword 3-65
CNTFLD (Continued-Entry Field) keyword 3-71
EDTMSK (Edit mask) keyword 3-104
ENTFLDATR (Entry Field Attribute) keyword 3-108
FLDCSRPRG (Cursor Progression Field) keyword 3-116
HLPARA (Help Area) keyword 3-121
HLPID (Help Identifier) keyword 3-132
MLTCHCFLD (Multiple-Choice Selection Field) keyword 3-149
MNUBAR (Menu Bar) keyword 3-151
MNUBARCHC (Menu-Bar Choice) keyword 3-152
MNUBARDSP (Menu-Bar Display) keyword 3-156
MNUBARSEP (Menu-Bar Separator) keyword 3-158
DDS (data description specifications) *(continued)*
MNUBARSW (Menu-Bar Switch Key) keyword 3-160
MNUCNL (Menu-Cancel Key) keyword 3-161
naming conventions 1-6
printouts E-4
PULLDOWN (Pull-Down Menu) keyword 3-177
SFLCSRPRG (Subfile Cursor Progression) keyword 3-194
SNGCHCFLD (Single-Choice Selection Field) keyword 3-226
WINDOW (Window) keyword 3-238
debugging template B-27
decimal positions
display files 3-21
ICF files 5-6
logical files 2-27
physical files 2-27
printer files 4-7
Default (DFT) keyword
display files
description 3-78
physical files
description 2-45
printer files
description 4-42
Default Value (DFTVAL) keyword
display files
description 3-80
Defer Evoke (DFREVOKE) keyword
ICF files
description 5-12
Define Character (DFNCHR) keyword
coding examples 4-36
printer files
code points 4-32
description 4-30
dot matrix 4-32
specifying dots to be printed 4-33
Define Line (DFNLIN) keyword E-2, E-15
defining
help area 3-121
selection field choice 3-65
single-choice selection field 3-226
Delete Check (DLTCHK) keyword
display files
description 3-81
Delete Edit (DLTEDT) keyword
display files
description 3-81
printer files
description 4-43
DESCEND (Descend) keyword
physical and logical files
description 2-45

DETACH (Detach) keyword
ICF files
description 5-11
example 5-12, B-19
device file B-7
DFNCHR (Define Character) keyword
coding examples 4-36
printer files
code points 4-32
description 4-30
dot matrix 4-32
specifying dots to be printed 4-33
DFNLIN (Define Line) keyword E-2, E-15
DFREVOKE (Defer Evoke) keyword
ICF files
description 5-12
DFT (Default) keyword
display files
description 3-78
physical files
description 2-45
printer files
description 4-42
DFTVAL (Default Value) keyword
display files
description 3-80
DIGIT (Digit) keyword
physical and logical files
description 2-47
Display Attribute (DSPATR) keyword
display files
description 3-82
display attributes for all fields 3-84
display attributes for input-capable fields 3-85
inquiry display B-7
display file
attribute character
beginning 3-25
ending 3-25
creating 1-1
definition 3-1
field name position 3-7
field specification, incorrect 3-26
introduction 3-1
keyword entries 3-26
message field 3-23
name specification 3-1
positional entries
comment 3-2
conditioning 3-2
data type 3-10
decimal positions 3-21
field name 3-7
form type 3-2
keyboard shift 3-10
length 3-2, 3-9
location 3-24
display file *(continued)*
positional entries *(continued)*
name 3-7
name type 3-6
record format name 3-7
reference 3-8
reserved 3-7
sequence number 3-2
specification type 3-6
usage 3-22
record specification 3-1
syntax examples 1-10
valid entries 3-10
Display Mode (DSPMOD) keyword
display files
description 3-90
Display Right to Left (DSPRL) keyword
display files
description 3-91
Display Size (DSPSIZ) keyword
display files
description 3-92
primary and secondary display sizes 3-92
special cases 3-96
horizontal subfile example B-14
displaying
menu bar 3-156
DLTCHK (Delete Check) keyword
display files
description 3-81
DLTEDT (Delete Edit) keyword
display files
description 3-81
printer files
description 4-43
dot matrix
copyright example 4-33
for 5224 and 5225 printers 4-32
grid pattern example 4-34, 4-36
hexadecimal digits for bit patterns 4-35
specifying dots 4-33
double-byte character set (DBCS)
character strings
entering E-3
considerations E-1
database files
description E-4
display files
description E-8
IGCALTTYP (DBCS Alternative Data Type) E-2, E-10
IGCCNV (DBCS Conversion) E-2, E-11
ICF files
description E-23
printer files
description E-13
DFNLIN (Define Line) E-2, E-15

double-byte character set (DBCS) *(continued)*
printer files *(continued)*
IGCALTTYP (DBCS Alternative Data Type) E-2, E-17
IGCANKCNV (Alphanumeric-to-DBCS Conversion) E-2, E-18
IGCCDEFNT (DBCS Coded Font) E-2, E-20
IGCCHRRTT (DBCS Character Rotation) E-2, E-20
DRAWER (Drawer) keyword
printer files
description 4-44
DSPATR (Display Attribute) keyword
display files
description 3-82
display attributes for all fields 3-84
display attributes for input-capable fields 3-85
inquiry display B-7
DSPMOD (Display Mode) keyword
display files
description 3-90
DSPRL (Display Right to Left) keyword
display files
description 3-91
DSPSIZ (Display Size) keyword
display files
description 3-92
primary and secondary display sizes 3-92
special cases 3-96
horizontal subfile example B-14
DUP (Duplication) keyword
display files
description 3-97
programming for the Dup key 3-97
restrictions on validity checking 3-98
Duplication (DUP) keyword
display files
description 3-97
programming for the Dup key 3-97
restrictions on validity checking 3-98
Dynamic Select (DYNSLT) keyword
logical files
description 2-48
DYNSLT (Dynamic Select) keyword
logical files
description 2-48

E

Edit Code (EDTCDE) keyword
display files
asterisk fill 3-100
description 3-99
example 3-103, B-18
floating-currency symbol 3-100
user-defined edit codes 3-102
Edit Code (EDTCDE) keyword *(continued)*
examples
field reference file B-1
inquiry display B-7
inquiry display example B-7
physical and logical files
description 2-51
printer files
asterisk fill 4-47
description 4-45
edit codes 4-46
floating currency symbol 4-47
user-defined edit codes 4-49
edit codes for OS/400 program
See OS/400 edit code
edit description
creating 3-102, 4-49
edit mask
creating 3-104
Edit Mask (EDTMSK) keyword 3-104
display files
description 3-104
edit word
body
ampersand 4-51
asterisk 4-51
blank 4-51
decimals and commas 4-51
description 4-51
floating currency symbol 4-51
zero suppression 4-51
example 4-50
expansion
description 4-51
formatting 4-52
status
ampersand 4-52
credit or minus symbol 4-52
description 4-51
Edit Word (EDTWRD) keyword
display files
description 3-105
forming the body of 3-106
forming the expansion of 3-107
forming the status of 3-107
parts of edit word 3-105
specifying a valid 3-107
physical and logical files
description 2-51
printer files
description 4-50
forming the body of 4-51
forming the expansion of 4-52
forming the status of 4-52
parts of edit word 4-50

EDTCDE (Edit Code) keyword
display files
asterisk fill 3-100
description 3-99
example 3-103, B-18
floating-currency symbol 3-100
user-defined edit codes 3-102
examples
field reference file B-1
inquiry display B-7
inquiry display example B-7
physical and logical files
description 2-51
printer files
asterisk fill 4-47
description 4-45
edit codes 4-46
floating currency symbol 4-47
user-defined edit codes 4-49
EDTMSK (Edit Mask) keyword 3-104
display files
description 3-104
EDTWRD (Edit Word) keyword
display files
description 3-105
forming the body of 3-106
forming the expansion of 3-107
forming the status of 3-107
parts of edit word 3-105
specifying a valid 3-107
physical and logical files
description 2-51
printer files
description 4-50
forming the body of 4-51
forming the expansion of 4-52
forming the status of 4-52
parts of edit word 4-50
End of Group (ENDGRP) keyword
ICF files
description 5-12
End of Session (EOS) keyword
ICF files
description 5-13
End Page (ENDPAGE) keyword
printer files
description 4-53
ENDGRP (End of Group) keyword
ICF files
description 5-12
ENDPAGE (End Page) keyword
printer files
description 4-53
ENTFLDATR (Entry Field Attribute) keyword 3-108
display files
description 3-108
entry field attribute
creating 3-108
Entry Field Attribute (ENTFLDATR) keyword 3-108
display files
description 3-108
EOS (End of Session) keyword
ICF files
description 5-13
ERASE (Erase) keyword
display files
description 3-110
Erase Input (ERASEINP) keyword
display files
description 3-111
ERASEINP (Erase Input) keyword
display files
description 3-111
ERRMSG (Error Message) keyword
display files
description 3-112
priority among keywords 3-113
restoring reversed-image fields 3-114
restrictions and notes 3-115
inquiry display example B-7
ERRMSGID (Error Message Identifier) keyword
display files
description 3-112
priority among keywords 3-113
restoring reversed-image fields 3-114
restrictions and notes 3-115
Error Message (ERRMSG) keyword
display files
description 3-112
priority among keywords 3-113
restoring reversed-image fields 3-114
restrictions and notes 3-115
inquiry display example B-7
Error Message Identifier (ERRMSGID) keyword
display files
description 3-112
priority among keywords 3-113
restoring reversed-image fields 3-114
restrictions and notes 3-115
Error Message Subfile (ERRSFL) keyword
display files
description 3-115
ERRSFL (Error Message Subfile) keyword
display files
description 3-115
EVOKE (Evoke) keyword
ICF files
description 5-13
example 5-16, B-19
special considerations 5-15

F

FAIL (Fail) keyword
ICF files
description 5-16
FCFO (First-Changed First-Out) keyword
physical and logical files
description 2-52
field name
display files 3-7
logical files 2-7
physical files 2-7
printer files 4-4
FIFO (First-In First-Out) keyword
physical and logical files
description 2-52
file
copying 2-46
creating
batch source statements 1-3
commands 1-3
creation 1-1
form filling 1-1
interactive source statements 1-3
procedure 1-3
source statements 1-3
description 1-1
device
inquiry display example B-7
message subfile example B-16
simple menu example B-7
display
description 3-1
example B-1
ICF
description 5-1
introduction 1-1
join example B-6
logical
defining 2-2
new record format example B-3
physical
defining 2-1
printer
description 4-1
specifying
new keys B-4
subfile example B-11
subfiles E-12
syntax examples
display files 1-10
ICF files 1-12
logical files 1-8, 1-9
physical files 1-8
printer files 1-11
syntax rules
description 1-4
First-Changed First-Out (FCFO) keyword
physical and logical files
description 2-52
First-In First-Out (FIFO) keyword
physical and logical files
description 2-52
FLAG parameter 1-4
FLDCSRPRG (Cursor Progression Field) keyword 3-116
display files
description 3-116
Floating-Point Precision (FLTPCN) keyword
display files
description 3-118
ICF files
description 5-17
physical and logical files
description 2-53
printer files
description 4-55
Floating-Point to Fixed Decimal (FLTFIXDEC) keyword
display files
description 3-117
printer files
description 4-54
FLTFIXDEC (Floating-Point to Fixed Decimal) keyword
display files
description 3-117
printer files
description 4-54
FLTPCN (Floating-Point Precision) keyword
display files
description 3-118
ICF files
description 5-17
physical and logical files
description 2-53
printer files
description 4-55
FMH (Function Management Header) keyword
ICF files
description 5-17
FMTNAME (Format Name) keyword
ICF files
description 5-18
FNTCHRID (Font Character Set ID) keyword
printer files
description 4-55
FONT (Font) keyword
printer files
description 4-56
Font Character Set ID (FNTCHRID) keyword
printer files
description 4-55

font type
description 4-57
graphic
description 4-57
HIGHLIGHT keyword 4-61
implied code page 4-58
hardware
description 4-57
HIGHLIGHT keyword 4-61
Force Data (FRCDTA) keyword
display files
description 3-118
ICF files
description 5-18
form type
display files 3-2
ICF files 5-2
logical files 2-5
physical files 2-5
printer files 4-2
FORMAT (Format) keyword
physical and logical files
description 2-53
Format Name (FMTNAME) keyword
ICF files
description 5-18
FRCDTA (Force Data) keyword
display files
description 3-118
ICF files
description 5-18
Function Management Header (FMH) keyword
ICF files
description 5-17

G

GDF (Graphic Data File) keyword
printer files
description 4-59
GENLVL (Severity Level) parameter 1-4
Get Retain (GETRETAIN) keyword
display files
description 3-120
GETRETAIN (Get Retain) keyword
display files
description 3-120
Graphic Data File (GDF) keyword
printer files
description 4-59

H

HELP (Help) keyword
display files
description 3-120, F-2
help area
defining 3-121
Help Area (HLPARA) keyword 3-121
display files
Help Boundary (HLPBDY) keyword
display files
description 3-125
Help Cleared (HLPCLR) keyword
display files
description 3-126
Help Command Key (HLPCMDKEY) keyword
display files
description 3-127
Help Document (HLPDOC) keyword
display files
description 3-129
Help Excluded (HLPEXCLD) keyword
display files
description 3-130
Help Full (HLPFULL) keyword
display files
description 3-131
help identifier
specifying 3-132
Help Identifier (HLPID) keyword 3-132
display files
description 3-132
Help Panel Group (HLPPNLGRP) keyword
display files
description 3-132
Help Record (HLPRCD) keyword
display files
description 3-133
Help Return (HLPRTN) keyword
display files
description 3-134, F-2
Help Search Index (HLPSCHIDX) keyword
display files
description 3-135
Help Sequencing (HLPSEQ) keyword
display files
description 3-136
Help Title (HLPTITLE) keyword
display files
description 3-137
HIGHLIGHT (Highlight) keyword
printer files
description 4-61
HLPARA (Help Area) keyword 3-121
display files
HLPBDY (Help Boundary) keyword
display files
description 3-125
HLPCLR (Help Cleared) keyword
display files
description 3-126

HLPCMDKEY (Help Command Key) keyword
display files
description 3-127
HLPDOC (Help Document) keyword
display files
description 3-129
HLPEXCLD (Help Excluded) keyword
display files
description 3-130
HLPFULL (Help Full) keyword
display files
description 3-131
HLPID (Help Identifier) keyword 3-132
display files
description 3-132
HLPPNLGRP (Help Panel Group) keyword
display files
description 3-132
HLPRCD (Help Record) keyword
display files
description 3-133
HLPRTN (Help Return) keyword
display files
description 3-134, F-2
HLPSCHIDX (Help Search Index) keyword
display files
description 3-135
HLPSEQ (Help Sequencing) keyword
display files
description 3-136
HLPTITLE (Help Title) keyword
display files
description 3-137
HOME (Home) keyword
display files
description 3-138

I

ICF file
creating 1-1
defining 5-1
example B-19
file coding example 5-1
keyword
entries 5-7
option indicators 5-2
positional entries
comment 5-2
conditioning 5-2
data type 5-6
decimal positions 5-6
form type 5-2
length 5-5
location 5-7
name 5-4
name type 5-3
ICF file *(continued)*
positional entries *(continued)*
reference 5-4
reserved 5-4
sequence number 5-2
usage 5-6
referenced field A-1
syntax example 1-12
IGCALTTYP (DBCS Alternative Data Type) keyword
description E-2
display files E-10
printer files E-17
IGCANKCNV (Alphanumeric-to-DBCS Conversion) keyword E-2, E-18
IGCCDEFNT (DBCS Coded Font) keyword E-2, E-20
IGCCHRRTT (DBCS Character Rotation) keyword E-2, E-20
IGCCNV (DBCS Conversion) keyword E-2, E-11
INDARA (Indicator Area) keyword
display files
description 3-139
ICF files
description 5-19
printer files
description 4-62
Indicator Area (INDARA) keyword
display files
description 3-139
ICF files
description 5-19
printer files
description 4-62
Indicator Text (INDTXT) keyword
display files
description 3-139
ICF files
description 5-20
printer files
description 4-62
INDTXT (Indicator Text) keyword
display files
description 3-139
ICF files
description 5-20
printer files
description 4-62
Initialize Input (INZINP) keyword
display files
description 3-142
Initialize Physical File Member (INZPFM) command 2-46
Initialize Record (INZRCD) keyword
display files
description 3-145
initializing
physical file member 2-46

INVITE (Invite) keyword
display files
description 3-140
ICF files
description 5-21
INZINP (Initialize Input) keyword
display files
description 3-142
INZPFM (Initialize Physical File Member) command 2-46
INZRCD (Initialize Record) keyword
display files
description 3-145

J

JDFTVAL (Join Default Values) keyword
logical files
description 2-54
JDUPSEQ (Join Duplicate Sequence) keyword
logical files
description 2-55
JFILE (Joined Files) keyword
logical files
description 2-57
example 2-58, B-6
JFLD (Joined Fields) keyword
logical files
description 2-58
example 2-59, B-6
JOIN (Join) keyword
logical files
description 2-60
example 2-61, B-6
Join Default Values (JDFTVAL) keyword
logical files
description 2-54
Join Duplicate Sequence (JDUPSEQ) keyword
logical files
description 2-55
join logical file
description 2-3
logical files 2-7
Join Reference (JREF) keyword
logical files
description 2-62
example 2-63, B-6
Joined Fields (JFLD) keyword
logical files
description 2-58
example 2-59, B-6
Joined Files (JFILE) keyword
logical files
description 2-57
example 2-58, B-6
JREF (Join Reference) keyword
logical files
description 2-62
example 2-63, B-6

K

KEEP (Keep) keyword
display files
description 3-146
key field name
example 2-9
logical files 2-8
physical files 2-8
keyboard shift
display files 3-10
keyword entry
display files 3-26
ICF files 5-7
logical files 2-30
physical files 2-30
printer files 4-11
keyword, DDS
abbreviations C-1
ABSVAL (Absolute Value) 2-31
ALARM (Audible Alarm) 3-28
ALIAS (Alternative Name) 2-32, 3-28, 4-11, 5-7
ALL (All) 2-33
ALTHELP (Alternative Help) 3-29
ALTNAME (Alternative Record Name) 3-30, F-4
ALTPAGEDWN/ALTPAGEUP (Alternative Page Down/Alternative Page Up) 3-30
ALTSEQ (Alternative Collating Sequence) 2-33
ALWGPH (Allow Graphics) 3-32
ALWNULL (Allow Null) 2-34
ALWROL (Allow Roll) 3-33
ALWWRT (Allow Write) 5-8
ASSUME (Assume) 3-35
AUTO (Auto) 3-36
BARCODE (Bar Code) 4-12
BLANKS (Blanks) 3-36
BLINK (Blink) 3-38
BLKFOLD (Blank Fold) 3-39, 4-15
BOX (Box) 4-16
CANCEL (Cancel) 5-8
CAnn (Command Attention) 3-39
CCSID (Character Code Set Identifier) 2-35
CDEFNT (Coded Font Name) 4-18
CFnn (Command Function) 3-41
CHANGE (Change) 3-43, F-2
CHCACCEL (Choice Accelerator Text) 3-44
CHCAVAIL (Choice Color/Display Attribute when Available 3-45
CHCCTL (Choice Control) 3-47
CHCSLT (Choice Color/Display Attribute when Selected) 3-49

keyword, DDS *(continued)*
CHCUNAVAIL (Choice Color/Display Attribute when Unavailable) 3-50
CHECK (Check) 2-35, 3-52
CHGINPDFT (Change Input Default) 3-62
CHKMSGID (Check Message Identifier) 2-36, 3-64
CHOICE (Selection Field Choice) 3-65
CHRID (Character Identifier) 3-67, 4-19
CHRSIZ (Character Size) 4-20
CLEAR (Clear) 3-67
CLRL (Clear Line) 3-68
CMP (Comparison) 2-37, 3-70
CNLINVITE (Cancel Invite) 5-9
CNTFLD (Continued-Entry Field) 3-71
COLHDG (Column Heading) 2-37
COLOR (Color) 3-72, 4-21
COMP (Comparison) 2-38, 3-76
CONCAT (Concatenate) 2-40
CONFIRM (Confirm) 5-10
CPI (Characters Per Inch) 4-22
CSRLOC (Cursor Location) 3-77
CTLDTA (Control Data) 5-10
CVTDTA (Convert Data) 4-26
DATE (Date) 3-78, 4-29
DATFMT (Date Format) 2-43
DATSEP (Date Separator) 2-44
DESCEND (Descend) 2-45
DETACH (Detach) 5-11
DFNCHR (Define Character) 4-30
DFNLIN (Define Line) E-2, E-15
DFREVOKE (Defer Evoke) 5-12
DFT (Default) 2-45, 3-78, 4-42
DFTVAL (Default Value) 3-80
DIGIT (Digit) 2-47
DLTCHK (Delete Check) 3-81
DLTEDT (Delete Edit) 3-81, 4-43
DRAWER (Drawer) 4-44
DSPATR (Display Attribute) 3-82
DSPMOD (Display Mode) 3-90
DSPRL (Display Right to Left) 3-91
DSPSIZ (Display Size) 3-92
DUP (Duplication) 3-97
DYNSLT (Dynamic Select) 2-48
EDTCDE (Edit Code) 2-51, 3-99, 4-45
EDTMSK (Edit Mask) 3-104
EDTWRD (Edit Word) 2-51, 3-105, 4-50
ENDGRP (End of Group) 5-12
ENDPAGE (End Page) 4-53
ENTFLDATR (Entry Field Attribute) 3-108
EOS (End of Session) 5-13
ERASE (Erase) 3-110
ERASEINP (Erase Input) 3-111
ERRMSG (Error Message) 3-112
ERRMSGID (Error Message Identifier) 3-112
ERRSFL (Error Subfile) 3-115
EVOKE (Evoke) 5-13

keyword, DDS *(continued)*
FAIL (Fail) 5-16
FCFO (First-Changed First-Out) 2-52
FIFO (First-In First-Out) 2-52
FLDCSRPRG (Cursor Progression Field) 3-116
FLTFIXDEC (Floating-Point to Fixed Decimal) 3-117, 4-54
FLTPCN (Floating-Point Precision) 2-53, 3-118, 4-55, 5-17
FMH (Function Management Header) 5-17
FMTNAME (Format Name) 5-18
FNTCHRSET (Font Character Set) 4-55
FONT (Font) 4-56
FORMAT (Format) 2-53
FRCDTA (Force Data) 3-118, 5-18
GDF (Graphic Data File) 4-59
GETRETAIN (Get Retain) 3-120
HELP (Help) 3-120, F-2
HIGHLIGHT (Highlight) 4-61
HLPBDY (Help Boundary) 3-125
HLPCLR (Help Cleared) 3-126
HLPCMDKEY (Help Command Key) 3-127
HLPDOC (Help Document) 3-129
HLPEXCLD (Help Excluded) 3-130
HLPFULL (Help Full) 3-131
HLPID (Help Identifier) 3-132
HLPPNLGRP (Help Panel Group) 3-132
HLPRCD (Help Record) 3-133
HLPRTN (Help Return) 3-134, F-2
HLPSCHIDX (Help Search Index) 3-135
HLPSEQ (Help Sequencing) 3-136
HLPTITLE (Help Title) 3-137
HOME (Home) 3-138
IGCALTTYP (DBCS Alternative Data Type) E-2, E-10, E-17
IGCANKCNV (Alphanumeric-to-DBCS Conversion) E-2, E-18
IGCCDEFNT (DBCS Coded Font) E-2, E-20
IGCCHRRTT (DBCS Character Rotation) E-2, E-20
IGCCNV (DBCS Conversion) E-2, E-11
INDARA (Indicator Area) 3-139, 4-62, 5-19
INDTXT (Indicator Text) 3-139, 4-62, 5-20
INVITE (Invite) 3-140, 5-21
INZINP (Initialize Input) 3-142
INZRCD (Initialize Record) 3-145
JDFTVAL (Join Default Values) 2-54
JDUPSEQ (Join Duplicate Sequence) 2-55
JFILE (Joined Files) 2-57
JFLD (Joined Fields) 2-58
JOIN (Join) 2-60
JREF (Join Reference) 2-62
KEEP (Keep) 3-146
LIFO (Last-In First-Out) 2-64
LINE (Line) 4-63
LOCK (Lock) 3-147
LOGINP (Log Input) 3-147

keyword, DDS *(continued)*
LOGOUT (Log Output) 3-148
LOWER (Lower) 3-148
LPI (Lines Per Inch) 4-65
MDTOFF (Modified Data Tag Off) 3-148
MLTCHCFLD (Multiple-Choice Selection Field) 3-149
MNUBAR (Menu Bar) 3-151
MNUBARCHC (Menu-Bar Choice) 3-152
MNUBARDSP (Menu-Bar Display) 3-156
MNUBARSEP (Menu-Bar Separator) 3-158
MNUBARSW (Menu-Bar Switch Key) 3-160
MNUCNL (Menu-Cancel Key) 3-161
MSGALARM (Message Alarm) 3-162
MSGCON (Message Constant) 3-163, 4-67
MSGID (Message Identifier) 3-164, F-2
MSGLOC (Message Location) 3-166
NEGRSP (Negative Response) 5-21
NOALTSEQ (No Alternative Collating Sequence) 2-64
NOCCSID (No Coded Character Set Identifier) 3-168
OPENPRT (Open Printer File) 3-168
OVERLAY (Overlay) 3-169, 4-68
OVRATR (Override Attribute) 3-170
OVRDTA (Override Data) 3-171
PAGEDOWN/PAGEUP (Page down/Page up) 3-172
PAGNBR (Page Number) 4-70
PAGRTT (Page Rotation) 4-71
PAGSEG (Page Segment) 4-72
PASSRCD (Passed Record) 3-173
PFILE (Physical File) 2-65, B-4
POSITION (Position) 4-74
PRINT (Print) 3-173
PROTECT (Protect) 3-176
PRTQLTY (Print Quality) 4-76
PULLDOWN (Pull-Down Menu) 3-177
PUTOVR (Put with Explicit Override) 3-178
PUTRETAIN (Put-Retain) 3-181
RANGE (Range) 2-66, 3-182
RCVCANCEL (Receive Cancel) 5-22
RCVCONFIRM (Receive Confirm) 5-23
RCVCTLDTA (Receive Control Data) 5-23
RCVDETACH (Receive Detach) 5-24
RCVENDGRP (Receive End of Group) 5-25
RCVFAIL (Receive Fail) 5-25
RCVFMH (Receive Function Management Header) 5-26
RCVNEGRSP (Receive Negative Response) 5-26
RCVTRNRND (Receive Turnaround) 5-27
RECID (Record Identification) 5-27
REF (Reference) 2-68, 3-183, 4-77, 5-31, B-3
REFACCPTH (Reference Access Path Definition) 2-69
REFFLD (Referenced Field) 2-70, 3-184, 4-78, 5-32, A-1

keyword, DDS *(continued)*
REFSHIFT (Reference Shift) 2-71
RENAME (Rename) 2-72
RETCMDKEY (Retain Command Keys) 3-186, F-5
RETKEY (Retain Function Keys) 3-186, F-5
RETLCKSTS (Retain Lock Status) 3-186
RMVWDW (Remove Window) 3-186
ROLLUP/ROLLDOWN (Roll up/Roll down) 3-187
RQSWRT (Request Write) 5-33
RSPCONFIRM (Respond Confirm) 5-34
RTNCSRLOC (Return Cursor Location) 3-188
RTNDTA (Return Data) 3-189
SECURITY (Security) 5-35
SETOF (Set Off) 3-190
SETOFF (Set Off) 3-191
SFL (Subfile) 3-191
SFLCLR (Subfile Clear) 3-193
SFLCSRPRG (Subfile Cursor Progression) 3-194
SFLCSRRRN (Subfile Cursor Relative Record Number) 3-194
SFLCTL (Subfile Control) 3-195
SFLDLT (Subfile Delete) 3-196
SFLDROP (Subfile Drop) 3-197
SFLDSP (Subfile Display) 3-199
SFLDSPCTL (Subfile Display Control) 3-199
SFLEND (Subfile End) 3-200
SFLENTER (Subfile Enter) 3-202
SFLFOLD (Subfile Fold) 3-203
SFLINZ (Subfile Initialize) 3-205
SFLLIN (Subfile Line) 3-206
SFLMODE (Subfile Mode) 3-207
SFLMSG (Subfile Message) 3-209
SFLMSGID (Subfile Message Identifier) 3-209
SFLMSGKEY (Subfile Message Key) 3-211
SFLMSGRCD (Subfile Message Record) 3-212
SFLNXTCHG (Subfile Next Changed) 3-213
SFLPAG (Subfile Page) 3-214
SFLPGMQ (Subfile Program Message Queue) 3-216
SFLRCDNBR (Subfile Record Number) 3-218
SFLRNA (Subfile Records Not Active) 3-219
SFLROLVAL (Subfile Roll Value) 3-220
SFLSIZ (Subfile Size) 3-223
SIGNED (Signed) 2-73
SKIPA (Skip After) 4-79
SKIPB (Skip Before) 4-80
SLNO (Starting Line Number) 3-225
SNGCHCFLD (Single-Choice Selection Field) 3-226
SPACEA (Space After) 4-81
SPACEB (Space Before) 4-81
SST (Substring) 2-74
SUBDEV (Subdevice) 5-36
SYNLVL (Synchronization Level) 5-38
SYSNAME (System Name) 3-228
TEXT (Text) 2-76, 3-228, 4-82, 5-38
TIME (Time) 3-229, 4-82

keyword, DDS *(continued)*
TIMER (Timer) 5-39
TIMFMT (Time Format) 2-76
TIMSEP (Time Separator) 2-77
TRNSPY (Transparency) 4-83
TRNTBL (Translation Table) 2-78
TXTRTT (Text Rotation) 4-84
UNDERLINE (Underline) 4-85
UNIQUE (Unique) 2-79, B-5
UNLOCK (Unlock) 3-229
UNSIGNED (Unsigned) 2-81
USER (User) 3-231
USRDFN (User-Defined) 3-231
USRDSPMGT (User Display Management) 3-232, F-6
USRRSTDSP (User Restore Display) 3-232
VALUES (Values) 2-81, 3-233
VARBUFMGT (Variable Buffer Management) 5-40
VARLEN (Variable-Length Field) 2-83
VARLEN (Variable-Length User Data) 5-40
VLDCMDKEY (Valid Command Key) 3-234
WDWBORDER (Window Border) 3-235
ZONE (Zone) 2-84

L

Last-In First-Out (LIFO) keyword
physical and logical files
description 2-64
length
display files 3-9, E-8
ICF files 5-5, E-23
logical files 2-22
physical files 2-22
printer files 4-6
library list A-1
LIFO (Last-In First-Out) keyword
physical and logical files
description 2-64
LINE (Line) keyword
printer files
description 4-63
Lines Per Inch (LPI) keyword
printer files
description 4-65
listing
DDS E-4
location
display files 3-24
ICF files 5-7
logical files 2-30
physical files 2-30
printer files 4-8
LOCK (Lock) keyword
display files
description 3-147
Log Input (LOGINP) keyword
display files
description 3-147
Log Output (LOGOUT) keyword
display files
description 3-148
logical file
*NONE key field 2-9, 2-13
changing 2-48
creating 1-1, 2-48
definition 2-2
JOIN example B-6
join files
example 2-3
multiple and simple format files
description 2-2
example 2-2
new keys example B-4
new record format example B-5
positional entries
comment 2-5
conditioning 2-5
data type 2-25
decimal positions 2-27
field name 2-7
form type 2-5
join logical files 2-7
key field name 2-8
length 2-22
location 2-30
name 2-6
name type 2-5
record format name 2-6
reference 2-21
reserved 2-5
select/omit field name 2-18
sequence number 2-4
simple and multiple format logical files 2-6
specification type 2-5
usage 2-29
record format
more than one 2-11
specifying 2-3
select/omit field name
description 2-18
example 2-20
simple and multiple format files
description 2-2
example 2-2
syntax example 1-8, 1-9
LOGINP (Log Input) keyword
display files
description 3-147
LOGOUT (Log Output) keyword
display files
description 3-148

LOWER (Lower) keyword
display files
description 3-148
LPI (Lines Per Inch) keyword
printer files
description 4-65

M

MDTOFF (Modified Data Tag Off) keyword
display files
description 3-148
menu bar
creating 3-151
displaying 3-156
Menu Bar (MNUBAR) keyword 3-151
display files
description 3-151
menu-bar choice
creating 3-152
Menu-Bar Choice (MNUBARCHC) keyword 3-152
display files
description 3-152
Menu-Bar Display (MNUBARDSP) keyword
display files
description 3-156
menu-bar separator
keyword 3-158
Menu-Bar Separator (MNUBARSEP) keyword 3-158
display files
description 3-158
menu-bar switch key
assigning 3-160
Menu-Bar Switch Key (MNUBARSW) keyword 3-160
display files
description 3-160
menu-cancel key
assigning 3-161
Menu-Cancel Key (MNUCNL) keyword 3-161
display files
description 3-161
Message Alarm (MSGALARM) keyword
display files
description 3-162
Message Constant (MSGCON) keyword
display files
description 3-163
printer files
description 4-67
Message ID (MSGID) keyword
display files
description 3-164, F-2
Message Location (MSGLOC) keyword
display files
description 3-166
MLTCHCFLD (Multiple-Choice Selection Field) keyword 3-149
display files
description 3-149
MNUBAR (Menu Bar) keyword 3-151
display files
description 3-151
MNUBARCHC (Menu-Bar Choice) keyword 3-152
display files
description 3-152
MNUBARDSP (Menu-Bar Display) keyword
display files
description 3-156
MNUBARSEP (Menu-Bar Separator) keyword 3-158
display files
description 3-158
MNUBARSW (Menu-Bar Switch Key) keyword 3-160
display files
description 3-160
MNUCNL (Menu-Cancel Key) keyword 3-161
display files
description 3-161
Modified Data Tag Off (MDTOFF) keyword
display files
description 3-148
MSGALARM (Message Alarm) keyword
display files
description 3-162
MSGCON (Message Constant) keyword
display files
description 3-163
printer files
description 4-67
MSGID (Message ID) keyword
display files
description 3-164, F-2
MSGLOC (Message Location) keyword
display files
description 3-166
multiple format file
See logical file
multiple-choice selection field
creating 3-149
keyword 3-149
Multiple-Choice Selection Field (MLTCHCFLD) keyword 3-149
display files
description 3-149

N

name
display files 3-7
ICF files 5-4
logical files 2-6

name *(continued)*
physical files 2-6
printer files 4-4
name type
display files 3-6
ICF files 5-3
logical files 2-5
physical files 2-5
printer files 4-4
naming conventions
DDS 1-6
Negative Response (NEGRSP) keyword
ICF files
description 5-21
NEGRSP (Negative Response) keyword
ICF files
description 5-21
No Alternating Sequence (NOALTSEQ) keyword
physical and logical files
description 2-64
No Coded Character Set Identifier (NOCCSID) keyword
display files
description 3-168
NOALTSEQ (No Alternating Sequence) keyword
physical and logical files
description 2-64
NOCCSID (No Coded Character Set Identifier) keyword
display files
description 3-168
***NONE key field 2-9, 2-13**
nonjoin logical file
logical files 2-6
numeric-only input-only field B-7

O

Open Printer File (OPENPRT) keyword
display files
description 3-168
OPENPRT (Open Printer File) keyword
display files
description 3-168
option indicator
AND condition 5-3
description 5-2
NOT condition 5-3
OR condition 5-3
OPTION parameter 1-4
OS/400 edit code
asterisk fill 4-47
description 4-46
floating currency symbol 4-47
function summary 4-47
user-defined 4-49
OS/400 edit code *(continued)*
valid 4-49
OVERLAY (Overlay) keyword
display files
description 3-169
printer files
description 4-68
Override Attribute (OVRATR) keyword
display files
description 3-170
Override Data (OVRDTA) keyword
display files
description 3-171
Override with Printer File (OVRPRTF) command 4-23
overriding with
printer file 4-23
OVRATR (Override Attribute) keyword
display files
description 3-170
OVRDTA (Override Data) keyword
display files
description 3-171
OVRPRTF (Override with Printer File) command 4-23

P

Page down/Page up (PAGEDOWN/PAGEUP) keyword
display files
description 3-172
Page Number (PAGNBR) keyword
printer files
description 4-70
example 4-70, B-18
Page Rotation (PAGRTT) keyword
printer files
description 4-71
Page Segment (PAGSEG) keyword
printer files
description 4-72
PAGEDOWN/PAGEUP (Page down/Page up) keyword
display files
description 3-172
PAGNBR (Page Number) keyword
printer files
description 4-70
example 4-70, B-18
PAGRTT (Page Rotation) keyword
printer files
description 4-71
PAGSEG (Page Segment) keyword
printer files
description 4-72

Passed Record (PASSRCD) keyword
display files
description 3-173
PASSRCD (Passed Record) keyword
display files
description 3-173
PFILE (Physical File) keyword
logical files
description 2-65
specifying new keys example B-4
physical file
creating 1-1
decimal positions 2-27
description 2-1
examples
field reference file B-1
new record format B-3
keyword
keyword entries 2-30
positional entries
comment 2-5
conditioning 2-5
data type 2-25
field name 2-7
form type 2-5
identified positions 2-4
key field name 2-8
length 2-22
location 2-30
name 2-6
name type 2-5
record format name 2-6
reference 2-21
reserved 2-5
sequence number 2-4
usage 2-29
procedure 2-1
Physical File (PFILE) keyword
logical files
description 2-65
specifying new keys example B-4
position
display files 3-24
POSITION (Position) keyword
printer files
description 4-74
PRINT (Print) keyword
display files
description 3-173
with *PGM special value 3-175
with a specified printer file 3-176
with response indicator 3-175
without parameter values 3-174
Print Quality (PRTQLTY) keyword
printer files
description 4-76
printer
3812 4-71
3816 4-71
3820 4-71
3825 4-71
3827 4-71
3835 4-71
4028 4-71
5224 4-26, 4-30, 4-84
5225 4-26, 4-30, 4-84
dot matrix 4-32
printer file
coding example 4-1
creating 1-1
DEVTYPE(*AFPDS) files
BOX keyword 4-16
CDEFNT keyword 4-18
ENDPAGE keyword 4-53
FNTCHRSET keyword 4-55
GDF keyword 4-59
IGCCDEFNT keyword E-20
LINE keyword 4-63
OVERLAY 4-68
PAGSEG keyword 4-72
POSITION keyword 4-74
TXTRTT keyword 4-84
example
coding 4-1
specify line and position 4-10
syntax coding 1-8
IPDS files
BLKFOLD keyword 4-16
CHRID keyword 4-19
COLOR keyword 4-22
CVTDTA keyword 4-26
DFNCHR keyword 4-31
HIGHLIGHT keyword 4-61
LPI keyword 4-66
keyword
entries 4-11
OS/400 edit codes 4-46
positional entries
comment 4-2
conditioning 4-3
constant fields 4-4
data type 4-7
decimal position 4-7
field name 4-4
form type 4-2
keyword entries 4-11
length 4-6
length example 4-6
location 4-8
name 4-4
name type 4-4
record format name 4-4
reference 4-5

printer file *(continued)*
positional entries *(continued)*
reserved 4-4
sequence number 4-2
specify the line 4-9
specify the line and position example 4-10
specify the position 4-9
usage 4-8
SCS files
BLKFOLD keyword 4-16
CHRID keyword 4-19
CHRSIZ keyword 4-20
COLOR keyword 4-22
CVTDTA keyword 4-26
DFNCHR keyword 4-31
FONT keyword 4-58
HIGHLIGHT keyword 4-61
LPI keyword 4-66
syntax example 1-11
user-defined edit codes 4-49
valid keywords list 4-11
PROTECT (Protect) keyword
display files
description 3-176
PRTQLTY (Print Quality) keyword
printer files
description 4-76
pull-down menu
creating 3-177
Pull-Down Menu (PULLDOWN) keyword 3-177
display files
description 3-177
PULLDOWN (Pull-Down Menu) keyword 3-177
display files
description 3-177
Put with Explicit Override (PUTOVR) keyword
display files
description 3-178
Put-Retain (PUTRETAIN) keyword
display files
description 3-181
PUTOVR (Put with Explicit Override) keyword
display files
description 3-178
PUTRETAIN (Put-Retain) keyword
display files
description 3-181

R

RANGE (Range) keyword
display files
description 3-182
physical and logical files
description 2-66
RCVCANCEL (Receive Cancel) keyword
ICF files
description 5-22
RCVCONFIRM (Receive Confirm) keyword
ICF files
description 5-23
example 5-23, B-19
RCVCTLDTA (Receive Control Data) keyword
ICF files
description 5-23
RCVDETACH (Receive Detach) keyword
ICF files
description 5-24
example 5-24, B-19
RCVENDGRP (Receive End of Group) keyword
ICF files
description 5-25
RCVFAIL (Receive Fail) keyword
ICF files
description 5-25
example 5-25, B-19
RCVFMH (Receive Function Management Header) keyword
ICF files
description 5-26
RCVNEGRSP (Receive Negative Response) keyword
ICF files
description 5-26
RCVTRNRND (Receive Turnaround) keyword
ICF files
description 5-27
example 5-27, B-19
Receive Cancel (RCVCANCEl) keyword
ICF files
description 5-22
Receive Confirm (RCVCONFIRM) keyword
ICF files
description 5-23
example 5-23, B-19
Receive Control Data (RCVCTLDTA) keyword
ICF files
description 5-23
Receive Detach (RCVDETACH) keyword
ICF files
description 5-24
example 5-24, B-19
Receive End of Group (RCVENDGRP) keyword
ICF files
description 5-25
Receive Fail (RCVFAIL) keyword
ICF files
description 5-25
example 5-25, B-19
Receive Function Management Header (RCVFMH) keyword
ICF files
description 5-26

Receive Negative Response (RCVNEGRSP) keyword
ICF files
description 5-26
Receive Turnaround (RCVTRNRND) keyword
ICF files
description 5-27
example 5-27, B-19
RECID (Record Identification) keyword
ICF files
description 5-27
example 5-29—5-31, B-19
record format
logical files 2-3
new B-5
physical files 2-6
record format name
display files 3-7
logical files 2-6
physical files 2-6
printer files 4-4
Record Identification (RECID) keyword
ICF files
description 5-27
example 5-29—5-31, B-19
REF (Reference) keyword
display files
description 3-183
ICF files
description 5-31
parameter values A-1
physical files
description 2-68
example 2-69, B-3
when to specify A-1
printer files
description 4-77
REFACCPTH (Reference Access Path Definition) keyword
logical files
description 2-69
reference
display files 3-8
ICF files 5-4
logical files 2-21
physical files 2-21
printer files 4-5
Reference (REF) keyword
display files
description 3-183
ICF files
description 5-31
parameter values A-1
physical files
description 2-68
example 2-69, B-3
when to specify A-1
Reference (REF) keyword *(continued)*
printer files
description 4-77
Reference Access Path Definition (REFACCPTH) keyword
logical files
description 2-69
Reference Shift (REFSHIFT) keyword
physical and logical files
description 2-71
referenced field A-1
Referenced Field (REFFLD) keyword
display files
description 3-184
ICF files
description 5-32
parameter values A-1
physical files
description 2-70
when to specify A-1
printer files
description 4-78
REFFLD (Referenced Field) keyword
display files
description 3-184
ICF files
description 5-32
parameter values A-1
physical files
description 2-70
when to specify A-1
printer files
description 4-78
REFSHIFT (Reference Shift) keyword
physical and logical files
description 2-71
Remove Window (RMVWDW) keyword
display files
description 3-186
RENAME (Rename) keyword
logical files
description 2-72
Request Write (RQSWRT) keyword
ICF files
description 5-33
Respond Confirm (RSPCONFIRM) keyword
ICF files
description 5-34
restoring reversed-image fields 3-210
Retain Command Keys (RETCMDKEY) keyword
display files
description 3-186, F-5
Retain Function Keys (RETKEY) keyword
display files
description 3-186, F-5

Retain Lock Status (RETLCKSTS) keyword
display files
description 3-186
RETCMDKEY (Retain Command Keys) keyword
display files
description 3-186, F-5
RETKEY (Retain Function Keys) keyword
display files
description 3-186, F-5
RETLCKSTS (Retain Lock Status) keyword
display files
description 3-186
Return Cursor Location (RTNCSRLOC) keyword
display files
description 3-188
Return Data (RTNDTA) keyword
display files
description 3-189
RMVWDW (Remove Window) keyword
display files
description 3-186
Roll up/Roll down (ROLLUP/ROLLDOWN) keyword
display files
description 3-187
subfile example B-11
ROLLUP/ROLLDOWN (Roll up/Roll down) keyword
display files
description 3-187
subfile example B-11
RQSWRT (Request Write) keyword
ICF files
description 5-33
RSPCONFIRM (Respond Confirm) keyword
ICF files
description 5-34
RTNCSRLOC (Return Cursor Location) keyword
display files
description 3-188
RTNDTA (Return Data) keyword
display files
description 3-189

S

SECURITY (Security) keyword
ICF files
description 5-35
select/omit field name
logical files 2-18
selection field choice
defining 3-65
Selection Field Choice (CHOICE) keyword 3-65
display files
description 3-65
sequence number
display files 3-2
sequence number *(continued)*
ICF files 5-2
logical files 2-4
physical files 2-4
printer files 4-2
Set Off (SETOF) keyword
display files
description 3-190
Set Off (SETOFF) keyword
display files
description 3-191
SETOF (Set Off) keyword
display files
description 3-190
SETOFF (Set Off) keyword
display files
description 3-191
Severity Level (GENLVL) parameter 1-4
SFL (Subfile) keyword
display files
description 3-191
SFLCLR (Subfile Clear) keyword
display files
description 3-193
SFLCSRPRG (Subfile Cursor Progression) keyword 3-194
display files
description 3-194
SFLCSRRRN (Subfile Cursor Relative Record Number) keyword
display files
description 3-194
SFLCTL (Subfile Control) keyword
display files
description 3-195
SFLDLT (Subfile Delete) keyword
display files
description 3-196
SFLDROP (Subfile Drop) keyword
display files
description 3-197
SFLDSP (Subfile Display) keyword
display files
description 3-199
SFLDSPCTL (Subfile Display Control) keyword
display files
description 3-199
SFLEND (Subfile End) keyword
display files
description 3-200
SFLENTER (Subfile Enter) keyword
display files
description 3-202
SFLFOLD (Subfile Fold) keyword
display files
description 3-203

SFLINZ (Subfile Initialize) keyword
display files
description 3-205
SFLLIN (Subfile Line) keyword
display files
description 3-206
horizontal subfile example B-14
SFLMODE (Subfile Mode) keyword
display files
description 3-207
SFLMSG (Subfile Message) keyword
display files
description 3-209
message subfile example B-16
SFLMSGID (Subfile Message ID) keyword
display files
description 3-209
SFLMSGKEY (Subfile Message Key) keyword
display files
description 3-211
message subfile example B-16
SFLMSGRCD (Subfile Message Record) keyword
display files
description 3-212
SFLNXTCHG (Subfile Next Changed) keyword
display files
description 3-213
SFLPAG (Subfile Page) keyword
display files
description 3-214
example B-11, B-13
SFLPGMQ (Subfile Program Message Queue) keyword
display files
description 3-216
message subfile example B-16
SFLRCDNBR (Subfile Record Number) keyword
display files
description 3-218
SFLRNA (Subfile Records Not Active) keyword
display files
description 3-219
SFLROLVAL (Subfile Roll Value) keyword
display files
description 3-220
SFLSIZ (Subfile Size) keyword
display files
description 3-223
example B-11, B-13
SIGNED (Signed) keyword
physical and logical files
description 2-73
simple file
See logical file
single-choice selection field
defining 3-226
Single-Choice Selection Field (SNGCHCFLD) keyword 3-226
display files
description 3-226
Skip After (SKIPA) keyword
printer files
description 4-79
Skip Before (SKIPB) keyword
printer files
description 4-80
example 4-80, B-18
SKIPA (Skip After) keyword
printer files
description 4-79
SKIPB (Skip Before) keyword
printer files
description 4-80
example 4-80, B-18
SLNO (Starting Line Number) keyword
display files
description 3-225
SNGCHCFLD (Single-Choice Selection Field) keyword 3-226
display files
description 3-226
source statements 1-3
Space After (SPACEA) keyword
printer files
description 4-81
Space Before (SPACEB) keyword
printer files
description 4-81
example 4-82, B-18
SPACEA (Space After) keyword
printer files
description 4-81
SPACEB (Space Before) keyword
printer files
description 4-81
example 4-82, B-18
specifying
color/display attribute unavailable 3-50
color/display attributes available 3-45
color/display selected 3-49
help identifier 3-132
line
display files 3-6
logical files 2-5
printer files 4-9
position
printer files 4-9
SST (Substring) keyword
logical files
description 2-74
Starting Line Number (SLNO) keyword
display files
description 3-225

SUBDEV (Subdevice) keyword
ICF files
description 5-36
Subdevice (SUBDEV) keyword
ICF files
description 5-36
Subfile (SFL) keyword
display files
description 3-191
Subfile Clear (SFLCLR) keyword
display files
description 3-193
Subfile Control (SFLCTL) keyword
display files
description 3-195
subfile cursor progression
creating 3-194
Subfile Cursor Progression (SFLCSRPRG) keyword 3-194
display files
description 3-194
Subfile Cursor Relative Record Number (SFLCSRRRN) keyword
display files
description 3-194
Subfile Delete (SFLDLT) keyword
display files
description 3-196
Subfile Display (SFLDSP) keyword
display files
description 3-199
Subfile Display Control (SFLDSPCTL) keyword
display files
description 3-199
Subfile Drop (SFLDROP) keyword
display files
description 3-197
Subfile End (SFLEND) keyword
display files
description 3-200
Subfile Enter (SFLENTER) keyword
display files
description 3-202
Subfile Fold (SFLFOLD) keyword
display files
description 3-203
Subfile Initialize (SFLINZ) keyword
display files
description 3-205
Subfile Line (SFLLIN) keyword
display files
description 3-206
horizontal subfile example B-14
Subfile Message (SFLMSG) keyword
display files
description 3-209
message subfile example B-16
Subfile Message ID (SFLMSGID) keyword
display files
description 3-209
Subfile Message Key (SFLMSGKEY) keyword
display files
description 3-211
message subfile example B-16
Subfile Message Record (SFLMSGRCD) keyword
display files
description 3-212
Subfile Mode (SFLMODE) keyword
display files
description 3-207
Subfile Next Changed (SFLNXTCHG) keyword
display files
description 3-213
Subfile Page (SFLPAG) keyword
display files
description 3-214
example B-11, B-13
Subfile Program Message Queue (SFLPGMQ) keyword
display files
description 3-216
message subfile example B-16
Subfile Record Number (SFLRCDNBR) keyword
display files
description 3-218
Subfile Records Not Active (SFLRNA) keyword
display files
description 3-219
Subfile Roll Value (SFLROLVAL) keyword
display files
description 3-220
Subfile Size (SFLSIZ) keyword
display files
description 3-223
example B-11, B-13
Substring (SST) keyword
logical files
description 2-74
Synchronization Level (SYNLVL) keyword
ICF files
description 5-38
example B-19
SYNLVL (Synchronization Level) keyword
ICF files
description 5-38
example B-19
syntax coding example
display files 1-11
ICF files 1-13
join logical files 1-10
physical files 1-8
printer files 1-12
simple logical files 1-9

syntax rules
DDS naming conventions 1-6
keywords and parameter values 1-4
SYSNAME (System Name) keyword
display files
description 3-228
System Name (SYSNAME) keyword
display files
description 3-228
System/36 environment
ALTNAME (Alternative Record Name) keyword F-4
considerations F-1
keyword considerations
CHANGE F-2
HELP F-2
HLPRTN F-2
MSGID F-2
RETCMDKEY (Retain Command Keys) keyword F-5
RETKEY (Retain Function Keys) keyword F-5
USRDSPMGT (User Display Management) keyword F-6

T

TEXT (Text) keyword
display files
description 3-228
field reference file example B-1
ICF files
description 5-38
physical and logical files
description 2-76
printer files
description 4-82
Text Rotation (TXTRTT) keyword
printer files
description 4-84
TIME (Time) keyword
display files
description 3-229
printer files
description 4-82
Time Format (TIMFMT) keyword
physical and logical files
description 2-76
Time Separator (TIMSEP) keyword
physical and logical files
description 2-77
TIMER (Timer) keyword
ICF files
description 5-39
TIMFMT (Time Format) keyword
physical and logical files
description 2-76
TIMSEP (Time Separator) keyword
physical and logical files
description 2-77
Translation Table (TRNTBL) keyword
logical files
description 2-78
Transparency (TRNSPY) keyword
printer files
description 4-83
TRNSPY (Transparency) keyword
printer files
description 4-83
TRNTBL (Translation Table) keyword
logical files
description 2-78
TXTRTT (Text Rotation) keyword
printer files
description 4-84

U

UNDERLINE (Underline) keyword
printer files
description 4-85
example 4-85, B-18
UNIQUE (Unique) keyword
physical and logical files
description 2-79
example 2-80, B-5
UNLOCK (Unlock) keyword
display files
description 3-229
UNSIGNED (Unsigned) keyword
physical and logical files
description 2-81
usage
display files 3-22
ICF files 5-6
logical files 2-29
physical files 2-29
printer files 4-8
USER (User) keyword
display files
description 3-231
User Display Management (USRDSPMGT) keyword
display files
description 3-232, F-6
User Restore Display (USRRSTDSP) keyword
display files
description 3-232
User-Defined (USRDFN) keyword
display files
description 3-231
USRDFN (User-Defined) keyword
display files
description 3-231

USRDSPMGT (User Display Management) keyword
display files
description 3-232, F-6
USRRSTDSP (User Restore Display) keyword
display files
description 3-232

V

Valid Command Key (VLDCMDKEY) keyword
display files
description 3-234
value
abbreviations C-1
VALUES (Values) keyword
display files
description 3-233
example B-7
physical and logical files
description 2-81
VARBUFMGT (Variable Buffer Management) keyword
ICF files
description 5-40
Variable Buffer Management (VARBUFMGT) keyword
ICF files
description 5-40
Variable Length Field (VARLEN) keyword
physical and logical files
description 2-83
Variable Length User Data (VARLEN) keyword
ICF files
description 5-40
VARLEN (Variable Length Field) keyword
physical and logical files
description 2-83
VARLEN (Variable Length User Data) keyword
ICF files
description 5-40
VLDCMDKEY (Valid Command Key) keyword
display files
description 3-234

W

WDWBORDER (Window Border) keyword
display files
description 3-235
window
creating 3-238
performance 3-232
RMVWDW (Remove Window) keyword 3-186
WINDOW (Window) keyword 3-238
display files
Window Border (WDWBORDER) keyword
display files
description 3-235

Z

ZONE (Zone) keyword
physical and logical files
description 2-84

Customer Satisfaction Feedback

Application System/400
Data Description Specifications
Reference
Version 2

Publication No. SC41-9620-02

Overall, how would you rate this manual?

	Very Satisfied	Satisfied	Dissatis-fied	Very Dissatis-fied
Overall satisfaction				

How satisfied are you that the information in this manual is:

Accurate				
Complete				
Easy to find				
Easy to understand				
Well organized				
Applicable to your tasks				
THANK YOU!				

Please tell us how we can improve this manual:

_____

_____

_____

_____

_____

May we contact you to discuss your responses? __ Yes __ No

Phone: (____) _____ Fax: (____) _____

To return this form:

- Mail it
- Fax it
  United States and Canada: **800+937-3430**
  Other countries: **(+1)+507+253-5192**
- Hand it to your IBM representative.

Note that IBM may use or distribute the responses to this form without obligation.

_____ _____
Name Address

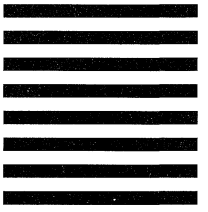
_____ _____
Company or Organization

_____ _____
Phone No.

Cu
Alc

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN DEPT 245
IBM CORPORATION
3605 HWY 52 N
ROCHESTER MN 55901-9986

Fold and Tape **Please do not staple** Fold and Tape

Cut
Alo

Program Number: 5738-SS1

Printed in Denmark by Bonde's

SC41-9620-02