IBM i
7.2

Programming
DDS Concepts

Note

Before using this information and the product it supports, read the information in “Notices” on page 49.

This edition applies to IBM i 7.2 (product number 5770-SS1) and to all subsequent releases and modifications until otherwise indicated in new editions. This version does not run on all reduced instruction set computer (RISC) models nor does it run on CISC models.

This document may contain references to Licensed Internal Code. Licensed Internal Code is Machine Code and is licensed to you under the terms of the IBM License Agreement for Machine Code.

© Copyright International Business Machines Corporation 1999, 2013.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

DDS concepts 1
PDF file for DDS concepts 1
Creating a file using DDS 2
Completing the DDS form 2
Entering the DDS source statements 4
Creating the DDS file 5
DDS coding rules, conventions, and terms 6
Conventions and terminology used in the DDS information 6
Rules for DDS keywords and parameter values 7
DDS naming conventions 8
DDS keywords and parameters 9
General considerations for using DBCS text with DDS files 15
Positional entries for files that use DBCS data 16
Length (positions 30 through 34) 16
Data type (position 35) 16
Keyword entries for files that use DBCS (positions 45 through 80) 16
DBCS character strings 17
Entering bracketed-DBCS character strings 17
Entering DBCS-graphic character strings 18
DDS computer printouts with DBCS output 18
Examples: DDS 19
Examples: DDS syntax 19
DDS syntax for a physical file 19
DDS syntax for a simple logical file 20
DDS syntax for a join logical file 21
DDS syntax for a display file 22
DDS syntax for a printer file 23
DDS syntax for an intersystem communications function file 24
Examples: DDS for each file type 24
Example: A field reference file 25
Example: A physical file with a new record format 26
Example: A logical file specifying multiple formats and new keys 26
Example: A logical file specifying a new record format 27
Example: A join logical file 28
Example: An inquiry display with two record formats in DDS 29
Example: A subfile with SFLPAG value equal to SFLSIZ value 32
Example: A subfile with paging by IBM i and high-level language program 33
Example: A horizontal subfile displayable on two display sizes 35
Example: A message subfile using DDS 36
Example: A printer file using DDS 37
Example: An intersystem communications function file using DDS 38
Example: Program that uses a physical file, display file, and printer file 40
Example: DDS compiler listing 42
DDS debugging template 44
When to specify REF and REFFLD keywords for DDS files 45
Related information for DDS concepts 47

Notices 49
Programming interface information 50
Trademarks 50

Terms and conditions.....51

Index..... 53

DDS concepts

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, data description specifications (DDS) describe data attributes in file descriptions that are external to the application program that processes the data.

DDS is a convenient alternative to describe data attributes on the IBM i operating system.

The file types that use DDS are physical and logical files, display files, printer files, and intersystem communications function (ICF) files.

Note: By using the code examples, you agree to the terms of the "Code license and disclaimer information" on page 48.

Related information

DDS for physical and logical files
DDS for display files
DDS for printer files
DDS for ICF files

PDF file for DDS concepts

You can view and print a PDF file of this information.

To view or download the PDF version of this document, select DDS Concepts (about 1170 KB).

Other information

You can view or download these related topic collections:

- DDS for Physical and Logical files contains reference information for using DDS with physical and logical files.
- DDS for display files contains reference information for using DDS with display files.
- DDS for printer files contains reference information for using DDS with printer files.
- DDS for ICF files contains reference information for using DDS with ICF files.

Saving PDF files

To save a PDF on your workstation for viewing or printing:

1. Right-click the PDF link in your browser.
2. Click the option that saves the PDF locally.
3. Navigate to the directory in which you want to save the PDF.
4. Click **Save**.

Downloading Adobe Reader

You need Adobe Reader installed on your system to view or print these PDFs. You can download a free copy from the Adobe Web site (www.adobe.com/products/acrobat/readstep.html).

Related reference

Related information for DDS concepts

© Copyright IBM Corp. 1999, 2013

Product manuals and other information center topic collections contain information that relates to the DDS concepts topic collection. You can view or print any of the PDF files.

Creating a file using DDS

You can use data description specifications (DDS) to create a file.

To create a file using DDS, follow these steps:

1. Complete the DDS form.
2. Type the source statements into a source file.

   The source file can be part of the IBM i 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 control language (CL) command.

Note: Using the screen design aid (SDA) utility, you can create and test display files without coding DDS directly, using only the functions that apply to display files.

Completing the DDS form

You can use the data description specifications (DDS) form to enter positional and keyword information in the designated columns.

A sample DDS form is printed in reduced size in Figure 1 on page 3.

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 about positional entries for each of the file types, see the following topics:

- Positional entries for physical and logical files
- Positional entries for display files
- Positional entries for printer files
- Positional entries for ICF 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 about keyword entries for each of the file types, see the following topics:

- Keywords for physical and logical files
- Keywords for display files
- Keywords for printer files
- Keywords for ICF files

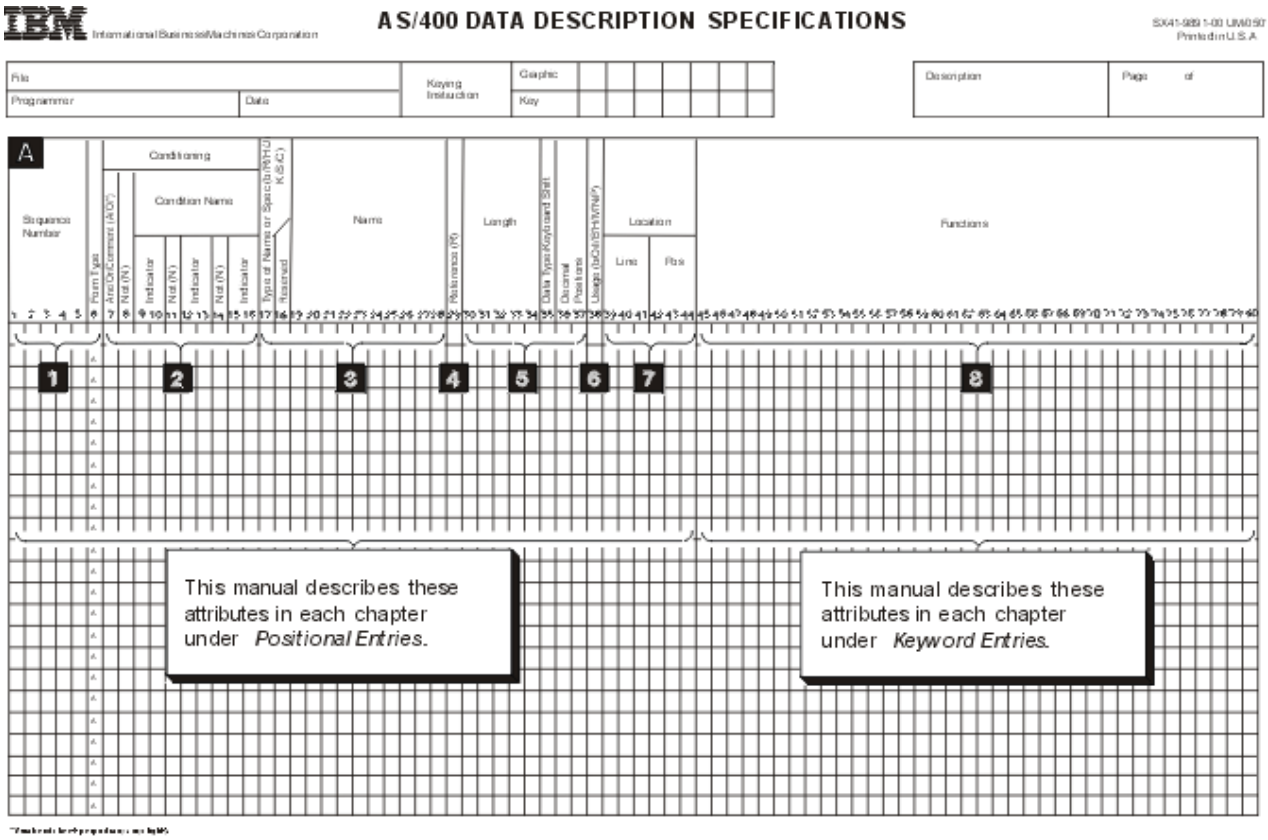


Figure 1. Overview of the positional entries and keywords

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 single quotation marks become the default values for displayed or printed fields.

Related tasks

Entering the DDS source statements
After filling out the forms, enter the source statements into source files.

Creating the DDS file
You can create a data description specifications (DDS) file by running a CL command that corresponds to the type of DDS file.

Entering the DDS 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 WebSphere® Development Studio. Use 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 CL command to create the file. Specify the name of the data file on the SRCFILE parameter and *FILE on the SRCMBR parameter of the command.

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

Related tasks

Completing the DDS form
You can use the data description specifications (DDS) form to enter positional and keyword information in the designated columns.

Creating the DDS file
You can create a data description specifications (DDS) file by running a CL command that corresponds to the type of DDS file.

Creating the DDS file

You can create a data description specifications (DDS) file by running a CL command that corresponds to the type of DDS file.

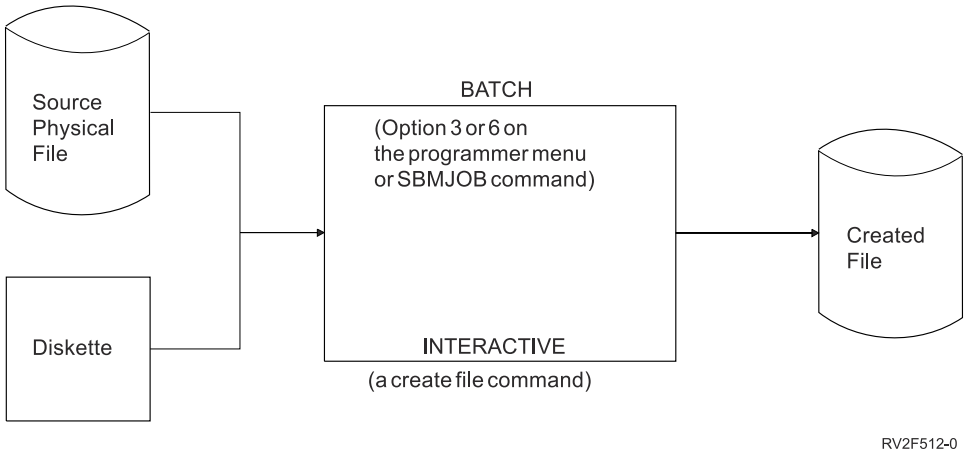
The file types and their corresponding commands are listed in the following table.

File type	Command
Physical file	Create Physical File (CRTPF)
Logical file	Create Logical File (CRTLF)
Display file	Create Display File (CRTDSPF)
Printer file	Create Printer File (CRTPRTF)
Intersystem communications function (ICF) file	Create ICF File (CRTICFF)

When you issue a CL command to create a file, 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 CL command that creates the file. 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 can 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.

The following figure shows a source file that can be processed in batch or interactively to create a file on diskette.



Related concepts

Example: DDS compiler listing
This is an example of a data description specifications (DDS) compiler computer printout.

DDS debugging template

A special template is available to help you in interpreting the fields on the data description specifications (DDS) compiler computer printout.

Related tasks

Completing the DDS form
You can use the data description specifications (DDS) form to enter positional and keyword information in the designated columns.

Entering the DDS source statements
After filling out the forms, enter the source statements into source files.

DDS coding rules, conventions, and terms

When you describe data attributes with data description specifications (DDS), you need to be aware of specific coding rules, naming conventions, and terminology.

Conventions and terminology used in the DDS information

The DDS information uses specific phrases and terms to describe key concepts.

When you read the DDS information in the information center, be aware of 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 that 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 IBM i operating system.
- *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.
- 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.

  |
  Specify the value either to the left or to the right (can refer to optional values).

Rules for DDS keywords and parameter values

When you code data description specifications (DDS) entries, you must follow these conventions for entering keywords and their parameters.

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 single quotation marks 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)
  ```
- A parameter expression consists of a set of values surrounded by left and right parentheses. Generally, the first value within the expression is a special value. The special value begins with an asterisk and must immediately follow the left parenthesis. One or more parameter values follow the special value. Separate the special value and the parameter value(s) by at least one blank. The last parameter value must immediately precede the right parenthesis. 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 single quotation marks to enclose character values. Numeric values appear without single quotation marks. 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 a single quotation mark within a character string, specify two single quotation marks so that one single quotation mark 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 that the continuation begins in position 45 of the next line (the first position in the functions field).
  - A **plus (+) sign** means that 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 continues a DDS statement until you specify one of the following fields:
  - 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.

DDS naming conventions

Data description specifications (DDS) require that files, records, fields, and other labels and identifiers that are described in DDS keywords follow specific rules.

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 control language (CL), which often allows *USRLIBL.
  - Specify a maximum of 10 characters for object names. If you enclose the name in quotation marks, you can 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. See the Control language topic collection 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, a single quotation mark, 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 _.
  - 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 single quotation marks. The enclosing single quotation marks are required when the name contains an opening or closing parenthesis or a single quotation mark character. When the name is enclosed in single quotation marks, specify two single quotation marks for each single quotation mark character within the name. If a folder name is concatenated, the enclosing single quotation marks, if specified, must be around the entire concatenated name.

DDS keywords and parameters

Data description specifications (DDS) keywords and parameters typically consist of combinations of standard abbreviations.

All the keywords and parameters used in DDS are listed in alphabetical order as follows:

Abbreviation	Meaning
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

Abbreviation	Meaning
B	before
BDY	boundary
BL	blinking field
BLANKS	blanks
BLINK	blinking cursor
BLK	blank
BOX	box
BTN	button
CA	command attention key
CCS	coded character 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
CMT	commit
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

Abbreviation	Meaning
DFT	default
DLT	delete
DOC	document
DSP	display
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
GRD	grid
GRP	group
GT	greater than
HDG	heading
HI	high intensity
HLP	help
ID	identifier

Abbreviation	Meaning
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
MAP	map
MDT	modified data tag
ME	mandatory enter
MF	mandatory fill
MGT	management
MLT	multiple
MNU	menu
MOU	mouse
MSG	message
MSK	mask
M10	IBM Modulus 10
M10F	IBM Modulus 10
M11	IBM Modulus 11
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

Abbreviation	Meaning
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
PRP	prepare
PRT	print or printer
PSH	push
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
REL	relative
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

Abbreviation	Meaning
RQS	request
RRN	relative record number
RSP	response
RST	restore
RTN	return
RTT	rotate
RZ	right-justify with zero fill
SCH	search
SCROLL	scroll
SEL	select
SEG	segment
SEP	separator
SEQ	sequence
SFL	subfile
SHELF	shelf
SIZ	size
SKIPA	skip after
SKIPB	skip before
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
TNS	transaction
TRNRND	turn around
TRNTBL	translation table
TXT	text
TYP	type
UL	underline

Abbreviation	Meaning
UNAVAIL	unavailable
USR	user
VAL	values
VAR	variable
VLD	valid
VN	validate name
VNE	validate name extended
WDW	window
WRD	word
WRT	write

General considerations for using DBCS text with DDS files

Be aware of these general considerations for positional entries, keyword entries, double-byte character set (DBCS) text literals, and data description specifications (DDS) computer printouts that contain DBCS characters.

DBCS text can be encoded as either Unicode or Extended Binary Coded Decimal Interchange Code (EBCDIC). If you are working on a new application, enabling an existing application for DBCS text, or working on an application involving Java™, Open Database Connectivity (ODBC), Java Database Connectivity (JDBC), or other Web methods, then IBM i Unicode support provides the easiest way to support not only DBCS text but also other text types. If you are working on an existing application already supporting DBCS text stored as EBCDIC, then IBM i EBCDIC support for DBCS text is useful.

DDS uses the following terms to describe the different types of DBCS data:

DBCS data
A general term to describe any form of EBCDIC-encoded DBCS data.

DBCS field
A general term to describe any field that can contain EBCDIC-encoded DBCS data.

Bracketed DBCS data
EBCDIC-encoded DBCS data that begins with a shift-out character and ends with a shift-in character.

DBCS graphic data
EBCDIC-encoded DBCS data that contains only DBCS data and does not contain shift-out and shift-in characters.

Unicode data
A general term to describe any form of Unicode-encoded DBCS data.

Unicode field
When the graphic data type is used with CCSID 1200 specified, then Unicode (UTF-16) is stored in the field instead of EBCDIC. When character data type is used with CCSID 1208 specified, then Unicode (UTF-8) is stored in the field instead of EBCDIC.

Related information

DBCS considerations for database files
DBCS considerations for display files
DBCS considerations for printer files
DBCS considerations for ICF files

Positional entries for files that use DBCS data

Positional entries are adapted so that you can define data fields that contain double-byte character set (DBCS) data. The entries are adapted for the length and data type positional entries.

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 or shift-in characters. The term DBCS refers to both bracketed and graphic DBCS data.

Length (positions 30 through 34)

Consider these effects when determining the length of a double-byte character set (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)

You can use these double-byte character set (DBCS) data types to identify DBCS fields.

For bracketed DBCS fields:

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.

For graphic DBCS fields:

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 for files that use DBCS (positions 45 through 80)

When you work with double-byte character set (DBCS) data, you can specify DBCS-specific attributes with certain DDS keywords.

Use the following keywords to perform these functions:

- 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

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.

Considerations for using DBCS character strings

Consider the following information when you use double-byte character set (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 single quotation mark, 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 can 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 that 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 that it is a valid DBCS character string.

Entering bracketed-DBCS character strings

You must enter bracketed-DBCS character strings in this way.

1. Begin the character string with a single quotation mark (').
2. Type a shift-out character.
3. Type the DBCS text.
4. Type a shift-in character.
5. End the character string with a single quotation mark (').

For example, to type the DBCS literal ABC, enter the following text, where 0_E represents the shift-out character and 0_F represents the shift-in character:

```
'0EABC0F'
```

Entering DBCS-graphic character strings

You must enter DBCS-graphic character strings in this way.

1. Type a G to indicate that the string contains DBCS-graphic data.
2. Begin the character string with a single quotation mark (').
3. Type a shift-out character.
4. Type the DBCS text.
5. Type a shift-in character.
6. End the character string with a single quotation mark (').

For example, to type the DBCS literal ABC, enter the following text, where 0_E represents the shift-out character and 0_F represents the shift-in character:

```
G'0EABC0F'
```

DDS computer printouts with DBCS output

DDS computer printouts are printed as DBCS output in these instances.

- The source file is DBCS.
- DBCS character strings were added to the source file as a result of a reference operation.

Examples: DDS

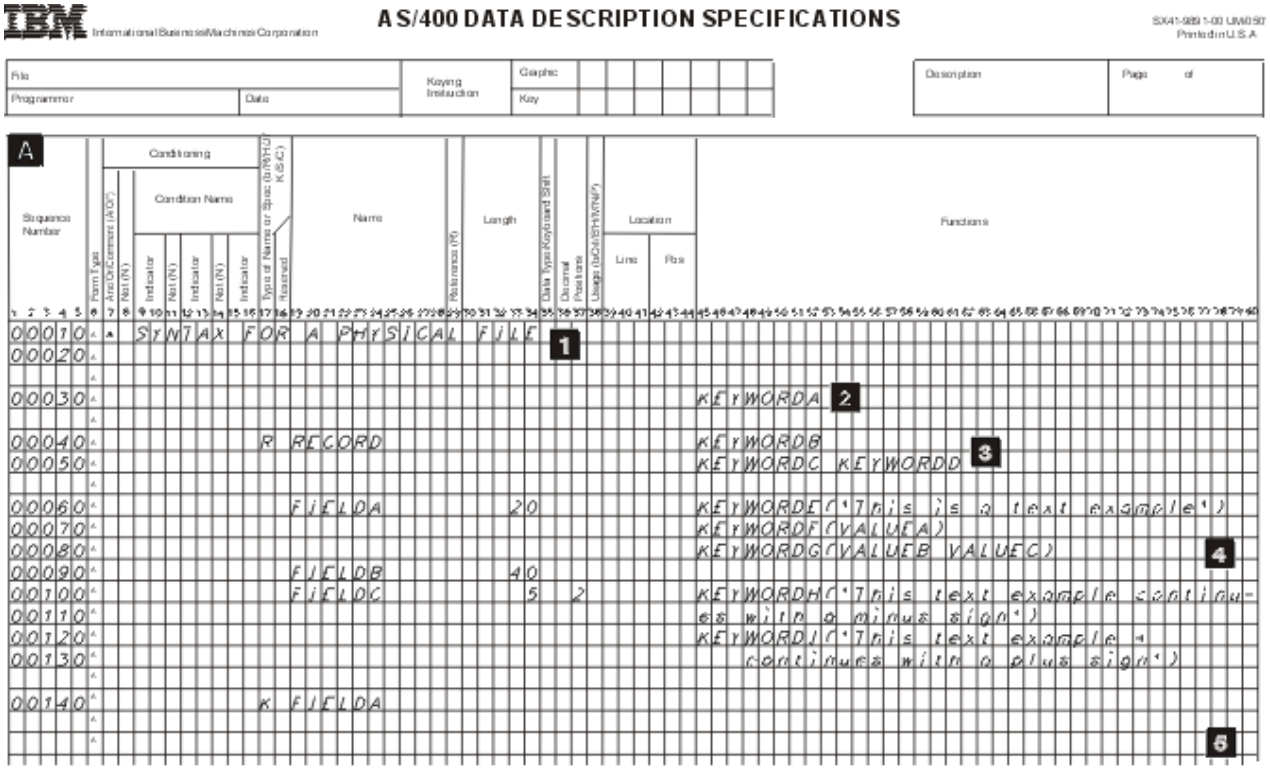
These examples show how to use data description specifications (DDS).

Examples: DDS syntax

Here are some data description specifications (DDS) syntax examples. Except for HLPARA, JFILE, JFLD, and PFILE, the keywords shown in these examples are not actual keywords. They indicate where you can specify the keywords.

DDS syntax for a physical file

This is the 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.

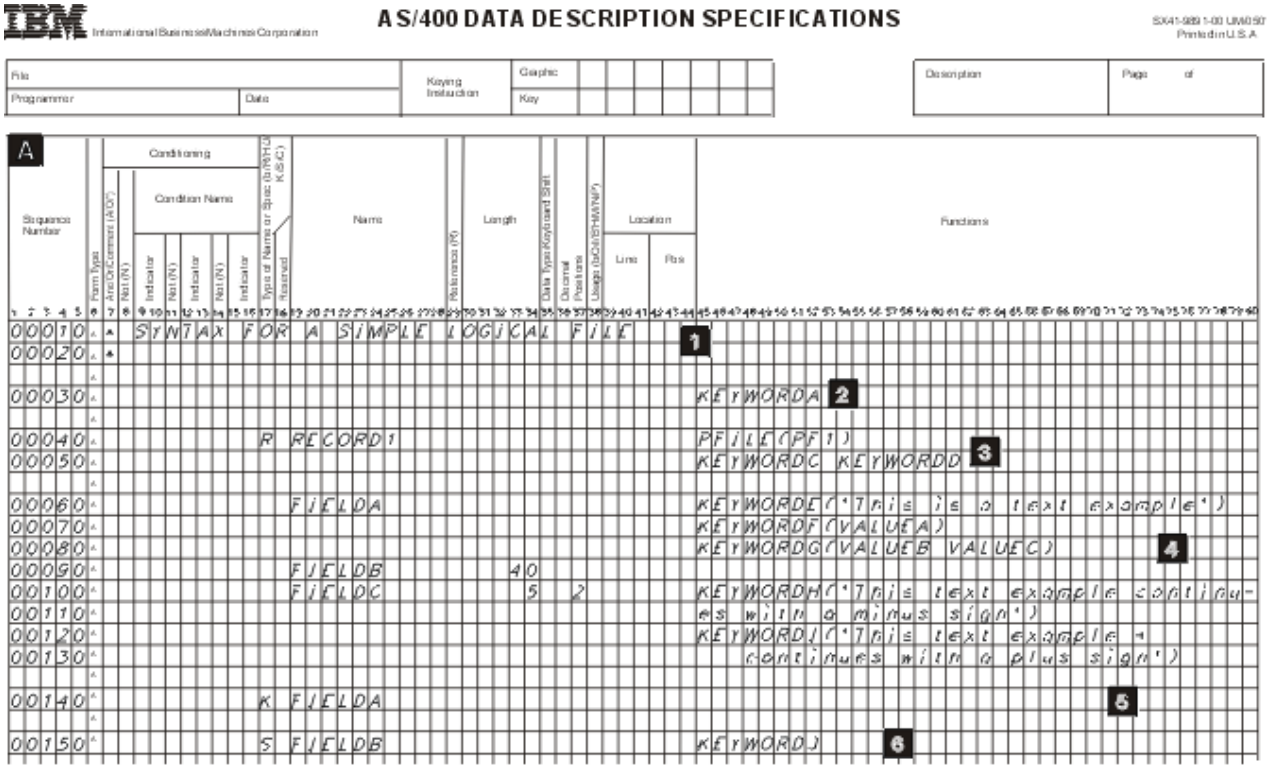
5

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.

Note: See Positional entries for physical and logical files for a description of each of the columns shown in the figure.

DDS syntax for a simple logical file

This is the 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.

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

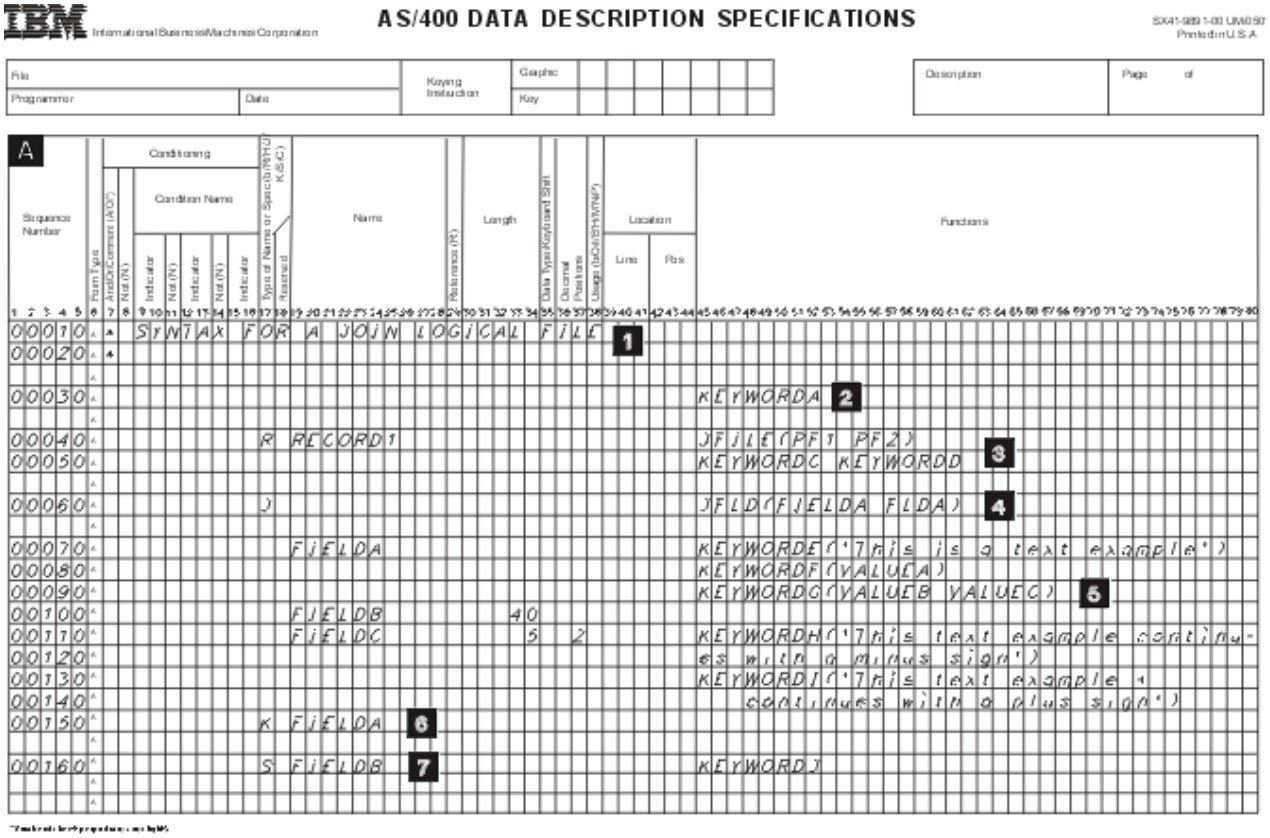
Select and omit levels (optional): S in position 17 identifies FIELDB as a select field. (O in position 17 identifies a field as an omit field.) The select and omit levels follow the key field level.

Notes:

1. 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.
2. See Positional entries for physical and logical files for a description of each of the columns shown in the figure.

DDS syntax for a join logical file

This is the 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.

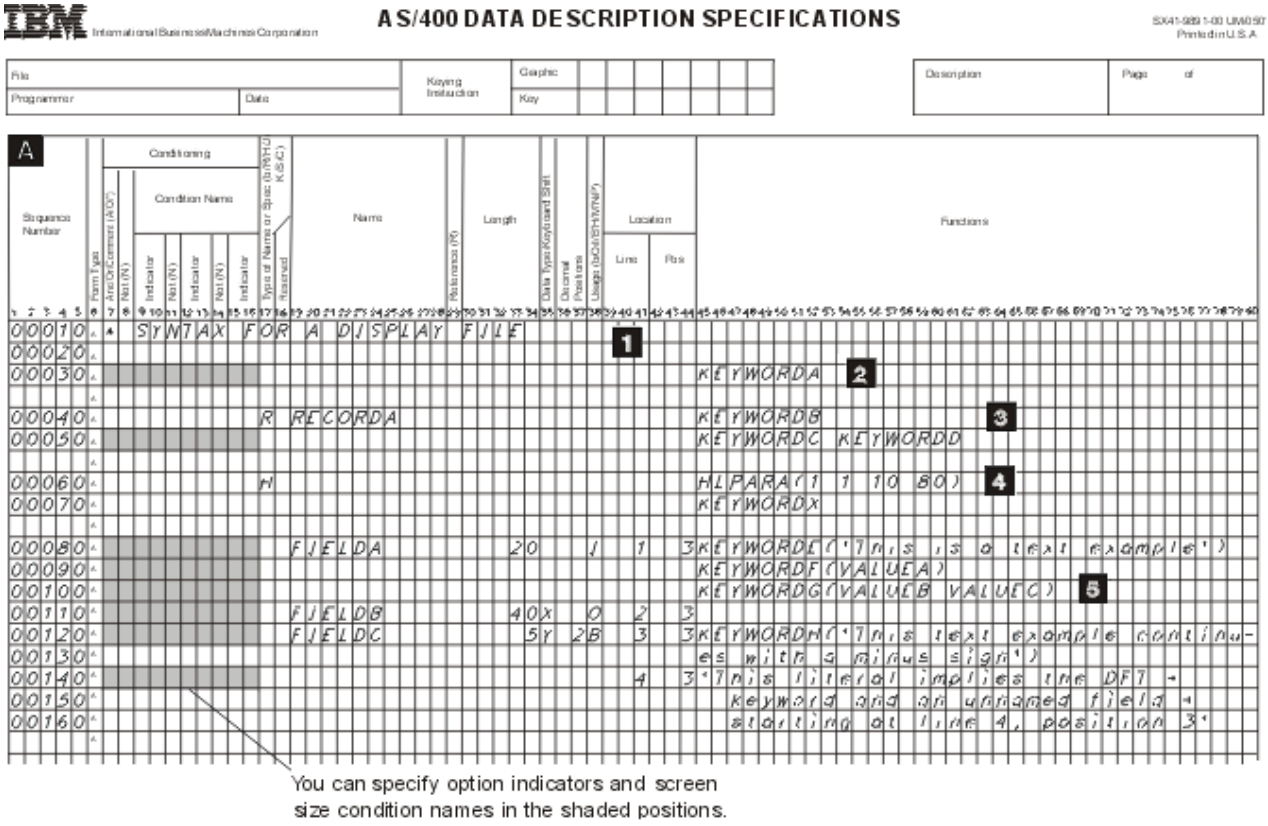
7

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.

Note: See Positional entries for physical and logical files for a description of each of the columns shown in the figure.

DDS syntax for a display file

This is the 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

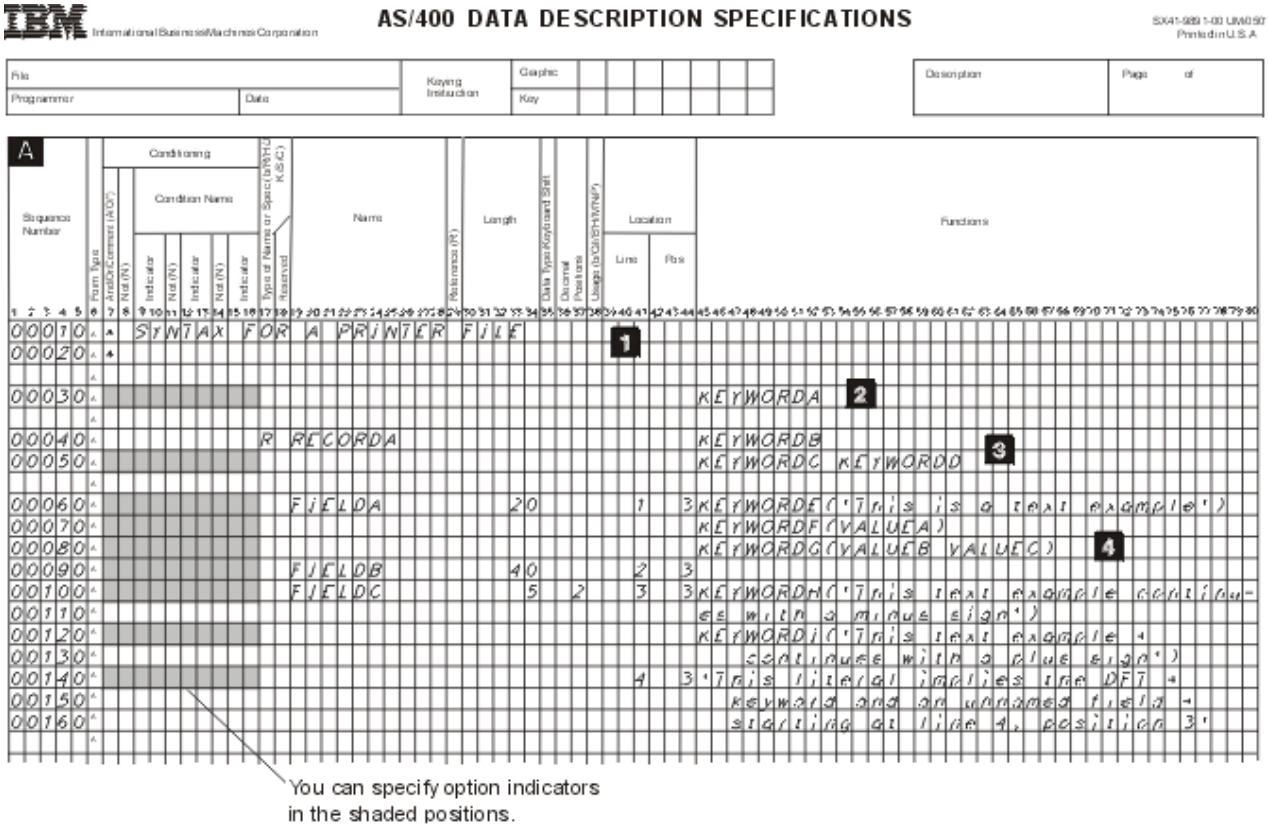
Field level (optional): Display file fields that are passed between the display device and the program must be named fields and must have a length specified. Other attributes can be specified explicitly or by default. Constant (unnamed) fields require only a location and a keyword, as described in DATE, DFT, TIME, and MSGCON keyword descriptions in DDS keyword entries for display files (position 45 through 80). Positions 17 through 38 do not apply to constant fields.

Notes:

1. Items 3 through 5 can be repeated to specify new record formats within the display file.
2. See Positional entries for display files for a description of each of the columns shown in the figure.

DDS syntax for a printer file

This is the 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

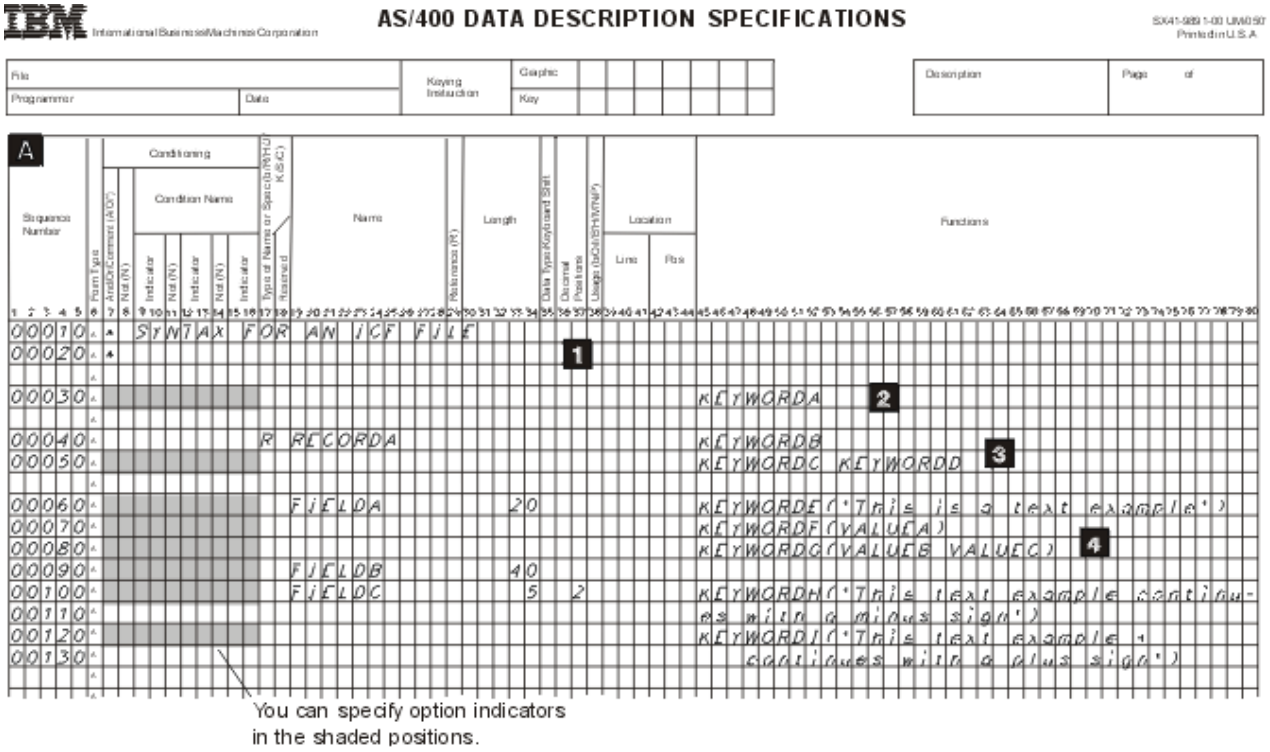
Field level (at least one field, whether named or unnamed, is required in each record format in the file): Printer file fields that are passed from the program to the printer must be named fields and must have a length specified. Other attributes can be specified explicitly or by default. Constant (unnamed) fields require only a location and a keyword, as described in the DATE, DFT, PAGNBR, TIME, and MSGCON keyword descriptions in Keyword entries for printer files (positions 45 through 80).

Notes:

1. Items 3 and 4 can be repeated to specify new record formats within the printer file.
2. See Positional entries for printer files for a description of each of the columns shown in the figure.

DDS syntax for an intersystem communications function file

This is the syntax for an intersystem communications function (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.

Notes:

1. Items 3 and 4 can be repeated to specify new record formats within the ICF file.
2. See Positional entries for ICF files for a description of each of the columns shown in the figure.

Examples: DDS for each file type

You can use these examples with appropriate high-level language programs.

- Examples of database files include the following types:
  - Field reference file (a physical file used for reference, not data storage)
  - Physical file
  - Logical file
- Examples of device files include the following types:
  - Display file with help specifications
  - Subfile examples

- Printer file
- ICF file

Example: A field reference file

This example defines all of the fields used in an application and refers to fields only within the field reference file itself.

The following keywords are important in the example:

COLHDG
EDTCDE(Z)
REFFLD
REFSHIFT
TEXT

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
```

```
|...+....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 2. DDS for a field reference file

Legend:

(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 will be 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.

Example: A physical file with a new record format

This example uses fields in a reference file (REF keyword) and uses a keyed-sequence access path.

The REF keyword is important in this example. This file has one record format. The names of all fields in the record format are specified.

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 3. DDS for a physical file

Legend:

(1)
At the file level, the REF keyword refers the IBM i operating system 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.

Example: A logical file specifying multiple formats and new keys

This example 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.

The PFILE keyword is important in this example.

```
|...+....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 4. DDS for a logical file specifying new keys

Legend:

(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 because this is not a join logical file.

Example: A logical file specifying a new record format

This example specifies a record format different from the associated physical file. The UNIQUE keyword is important in this example.

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 can name all fields, name fields in other physical files, concatenate fields, change the order of fields, rename fields, or choose different key fields. In this logical file, the programmer merely omits 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 5. DDS for a logical file

Legend:

(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 do in physical files or in any device file.

(4)
As in CUSMSTP, the field ACTNUM is treated as a key field.

Example: A join logical file

This example specifies that the join logical file join three physical files (PF1, PF2, and PF3). This way, 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.

The following keywords are important in the example:

- JFILE
- JFLD
- JOIN
- JREF

```
|...+....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 6. DDS for a join logical file

Legend:

(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 either to the primary file or to another secondary file. In this example, PF2 in the second JOIN keyword can be PF1. There is 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.

Example: An inquiry display with two record formats in DDS

This example defines a display in data description specifications (DDS). The display is shown 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      XXXXXXXXXXXXXXXXXXXXXX
Address   XXXXXXXXXXXXXXXXXXXXXX
City      XXXXXXXXXXXXXXXXXXXXXX
State     XX       Zip code NNNNN


Credit limit        $NNN,NNN.NN









F3 - End Program & Print Report  F6 - Return to prompt
```

Name XXXXXXXXXXXXXXXXXXXXXX
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.

The following keywords are important in the example:

CAnn	HELP
CHECK	HLPARA
DSPATR(HI BL)	HLPRCD
DSPATR(UL)	HLPBDY
EDTCDE(Y)	HLPDOC
EDTCDE(2 $)	OVERLAY
ERRMSG	PRINT

The example uses +n to specify position.

```
|...+....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(ADDRHELP)
00210A                                    HLPARA(7 10 9 33)
00220A                                (12) HLPBDY
00230A          H                         HLPRCD(HELPRCD1 HELPFILE)
00240A                               (13) HLPARA(12 18 12 40)
     A
     A
     A
```

```
|...+....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 7. Display with two record formats

Legend:

(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 displays 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 required 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 is 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 line 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 and position specifications. The default of the NAME, ADDRES, CITY, and STATE fields is set 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 workstation user are generally located at the bottom of the display, just above the message line.

Example: A subfile with SFLPAG value equal to SFLSIZ value

This example defines a display in data description specifications (DDS). When an output operation to the subfile-control record format SFLCTL1 is performed, the display appears.

```
        First Field     Second Field

                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
                        XXXXXXXXXXXXXXXXXXX
```

The following keywords are important in the example:

ROLLDOWN	SFLDSPCTL
ROLLUP	SFLPAG
SFLCLR	SFLSIZ
SFLDSP	

The file in Figure 8 on page 32 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.

```
|...+....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 8. Subfile with subfile size equal to subfile page

Legend:

(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 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) 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 workstation 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 workstation user will 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.

Example: A subfile with paging by IBM i and high-level language program

This example describes a combined method of paging a subfile that uses system resources efficiently.

In some applications, the number of records in the subfile might be quite large. However, the application user might want to view only the first page or two of these records. In this case, it might be faster and more efficient for the application program to build the subfile a page at a time as requested by the user. The IBM i operating system 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 IBM i operating system and one handled by the high-level language program.

```
NAME                                DEPARTMENT     PHONE

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX      XXXXXXXXXX     NNNN
```

The following keywords are important in the example:

SFLPAG
SFLSIZ

The SFLSIZ value is larger than the SFLPAG value. The subfile is paged by the SFLPAG value.

```
|...+....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 9. Subfile with paging by IBM i operating system and high-level language program in DDS

Legend:

(1)
The SFLSIZ value must be greater than the SFLPAG value so that the IBM i operating system will handle paging within the subfile. A maximum of 9999 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.

Example: A horizontal subfile displayable on two display sizes

This example shows how a subfile defined in data description specifications (DDS) appears on the 24 x 80 and 27 x 132 display sizes.

```
    COLUMN 1                    COLUMN 2
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
```

```
    COLUMN 1                    COLUMN 2
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
_____ XXXXXXXXXXXXXXXX      _____ XXXXXXXXXXXXXXXX
```

The following keywords are important in the example:

DSPSIZ
SFLLIN

Subfile records appear in two columns (SFLLIN keyword). The subfile can be displayed on two display sizes (DSPSIZ keyword).

```
|...+....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 10. Horizontal subfile on two display sizes

Legend:

(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.

Example: A message subfile using DDS

This example shows how to define a message subfile. The display shows the message subfile.

```
                                MESSAGE SUBFILE

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

The following keywords are important in the example:

SFLMSGKEY
SFLPGMQ
SFLMSGRCD

Records in the subfile are messages from a message file.

```
|...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
00010A*
00020A* MESSAGE SUBFILE
00030A*
00040A          R SFLR                      SFL
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 11. Message subfile

Legend:

(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 IBM i operating system when the display station user presses a Page Up or a Page Down key. The SFLEND keyword allows the IBM i operating system to display a plus sign whenever the subfile can be paged up.

Example: A printer file using DDS

This example contains data description specifications (DDS) for printing a customer master list.

The following keywords are important in the example:

EDTCDE(Y)
EDTCDE(Z)
PAGNBR
SKIPB
SPACEA
UNDERLINE
BARCODE
CHRSIZ
COLOR

This printer file uses space and skip keywords instead of line numbers.

```
|...+....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
```

```
|...+....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 12. Printer file

Legend:

(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.

Example: An intersystem communications function file using DDS

This example contains data description specifications (DDS) for transmitting data between the IBM i platform and a remote system or device.

The following keywords are important in the example:

CONFIRM
DETACH
EVOKE
RCVDETACH
RCVFAIL
RCVCONFIRM
RCVTRNRND

RECID
SYNLVL
EOS
RSPCONFIRM
ALWWRT
FAIL
RQSWRT

```
|...+....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
```

```
|...+....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 13. ICF file

Legend:

(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 performs any of the following actions:

- 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 system sends and receives data in the form of header records (record format HEADER) and detail records (record format DETAIL). If your program is sending, option indicator 09 can be set on, which enables the CONFIRM keyword to request the remote system to confirm receiving the data.

When receiving data, 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 an 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 that uses a physical file, display file, and printer file

This example program shows the use of externally described data in a program.

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) command, 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 Report Program Generator (RPG). You can enter the RPG specifications shown in the example into a source file on your system and create the program using the Create RPG Program (CRTRPGPGM) command.

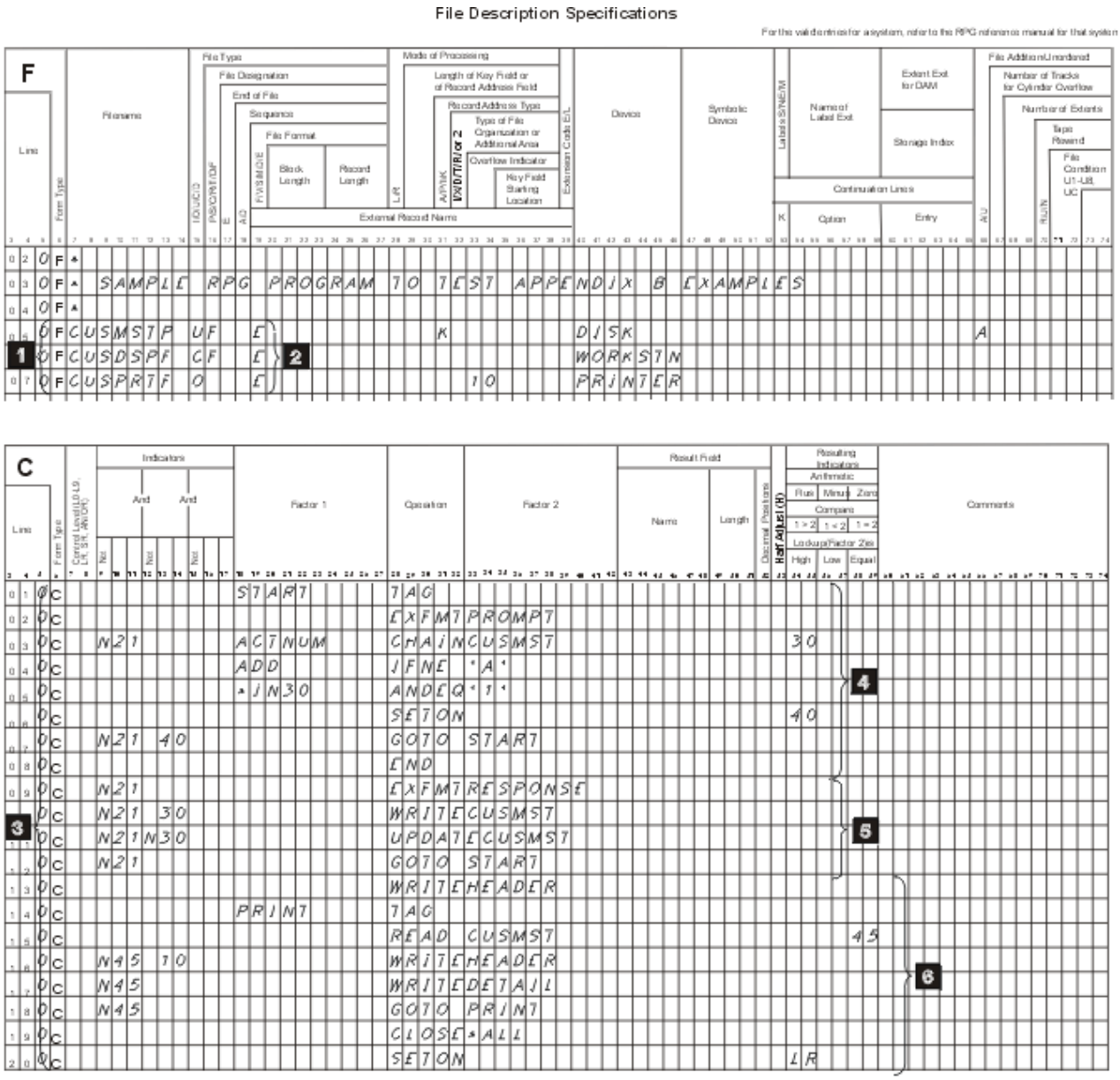


Figure 14. Example RPG program using a physical, display, and printer file

Legend:

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 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 operation code indicates a label in the program. The EXFMT operation code writes the record PROMPT to the display, and then reads the record when the user presses the Enter key. The CHAIN

operation 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 until 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 operation code adds a new record to the physical file. Otherwise, the UPDAT operation 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

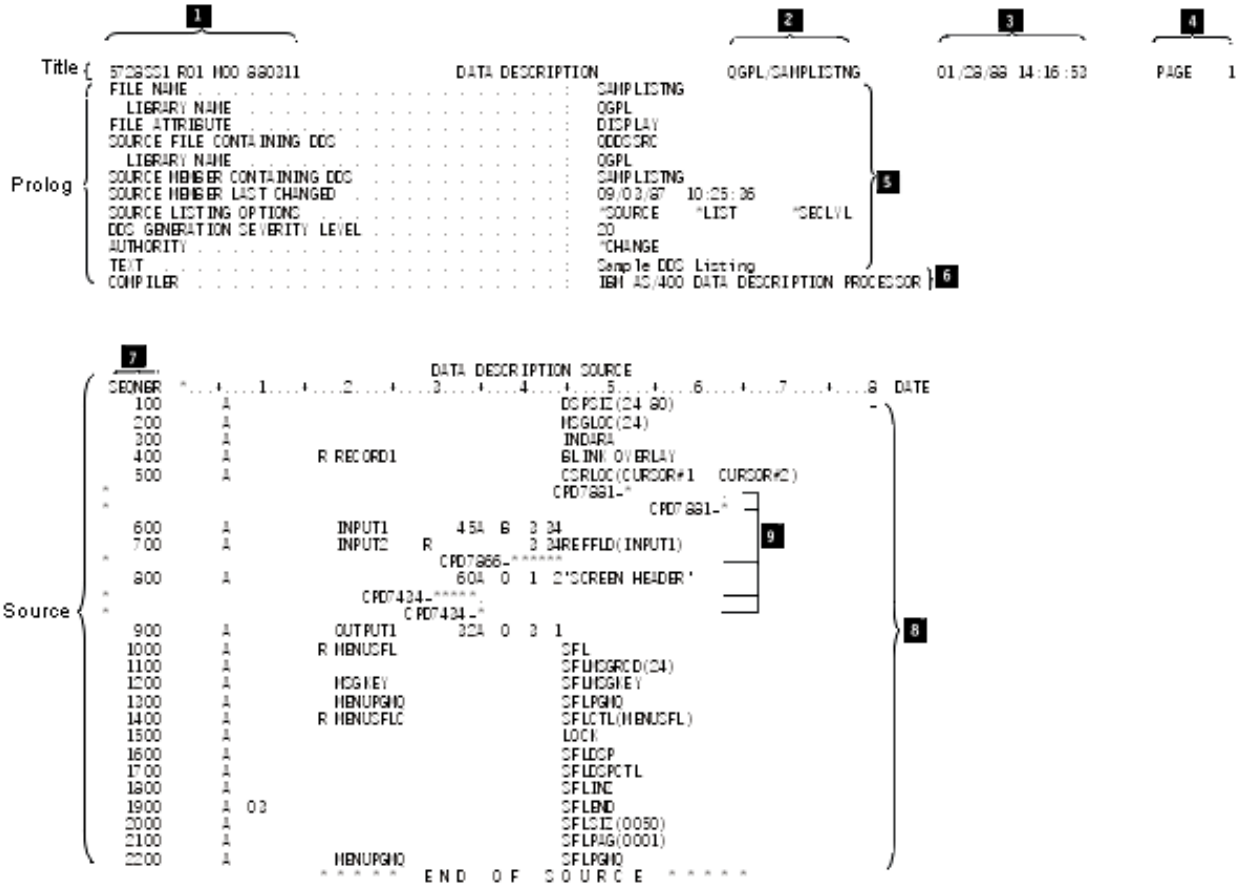
This section prints the report.

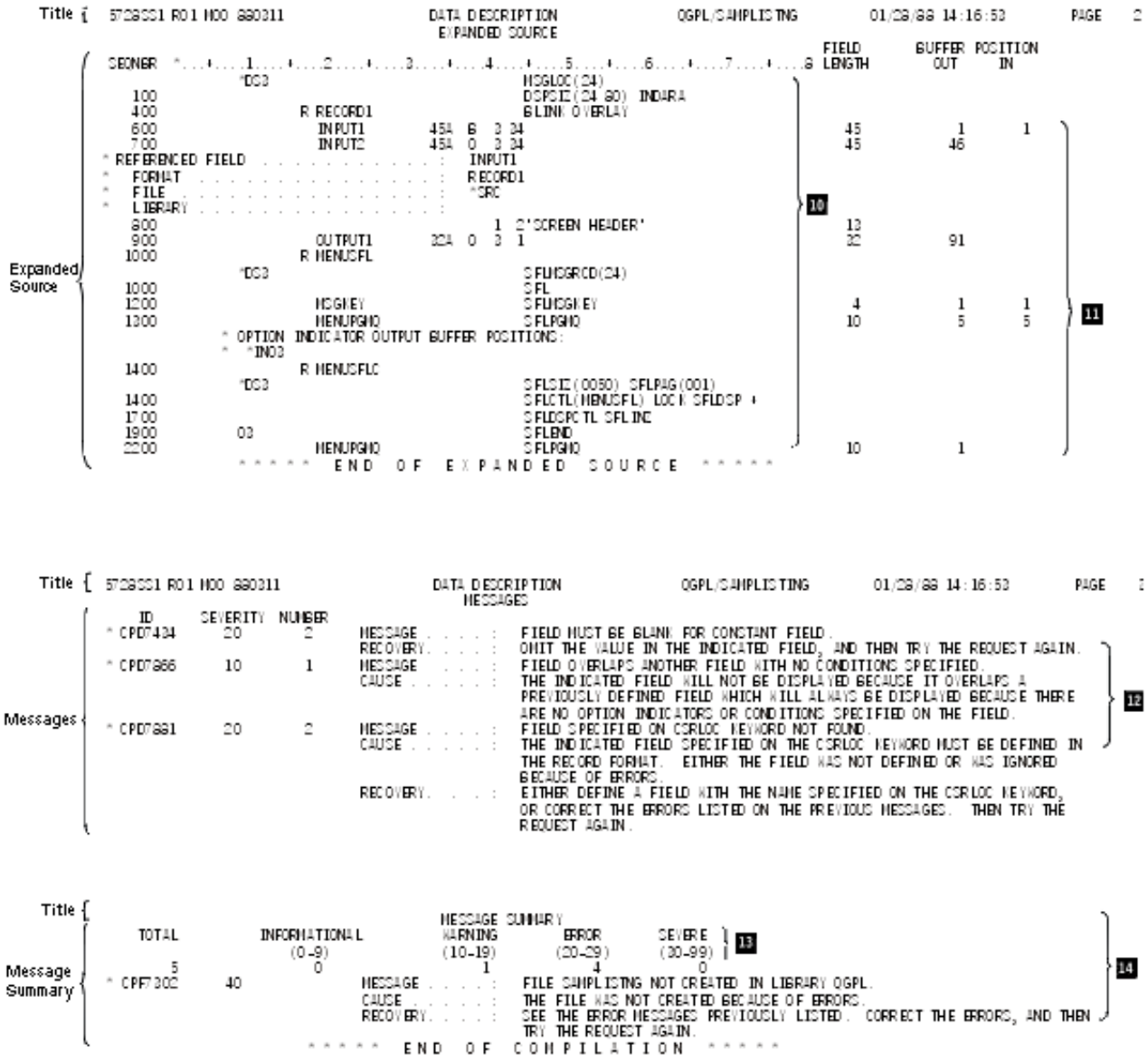
A record is read from the physical file and the DETAIL record is written to the printer file until the end of the physical file is reached (indicator 45 is set on). The HEADER record is written on the first page and then written again on each new page (indicator 10 is on). When all the records have been written, the CLOSE operation code closes all the files and SETON LR ends the program.

Example: DDS compiler listing

This is an example of a data description specifications (DDS) compiler computer printout.

After 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. You can enter the CL command interactively or in a batch job. The data description processor retrieves the data description specifications from the source file designated on the CL command that creates the file, validates the specifications, and creates a computer printout with any errors and any referenced specifications, as this example illustrates.





Compiler listing title (appears at top of each output page):

1

The program number, release modification level, and date of the IBM i operating system.

2

The qualified name.

3

The date and time of this run.

4

The page number in the computer printout.

Compiler listing prolog:

5

The type of file and the parameter values specified (or defaults if not specified) on the CL command that creates the file.

6

The name of the DDS processor.

Compiler listing source:

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.

Compiler listing expanded source:

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.

Compiler listing messages:

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.

Compiler listing message summary:

13
The number of messages at each severity level.

14
The final completion message.

Related tasks

Creating the DDS file
You can create a data description specifications (DDS) file by running a CL command that corresponds to the type of DDS file.

DDS debugging template

A special template is available to help you in interpreting the fields on the data description specifications (DDS) compiler computer printout.

The following figure shows a reduced debugging template.

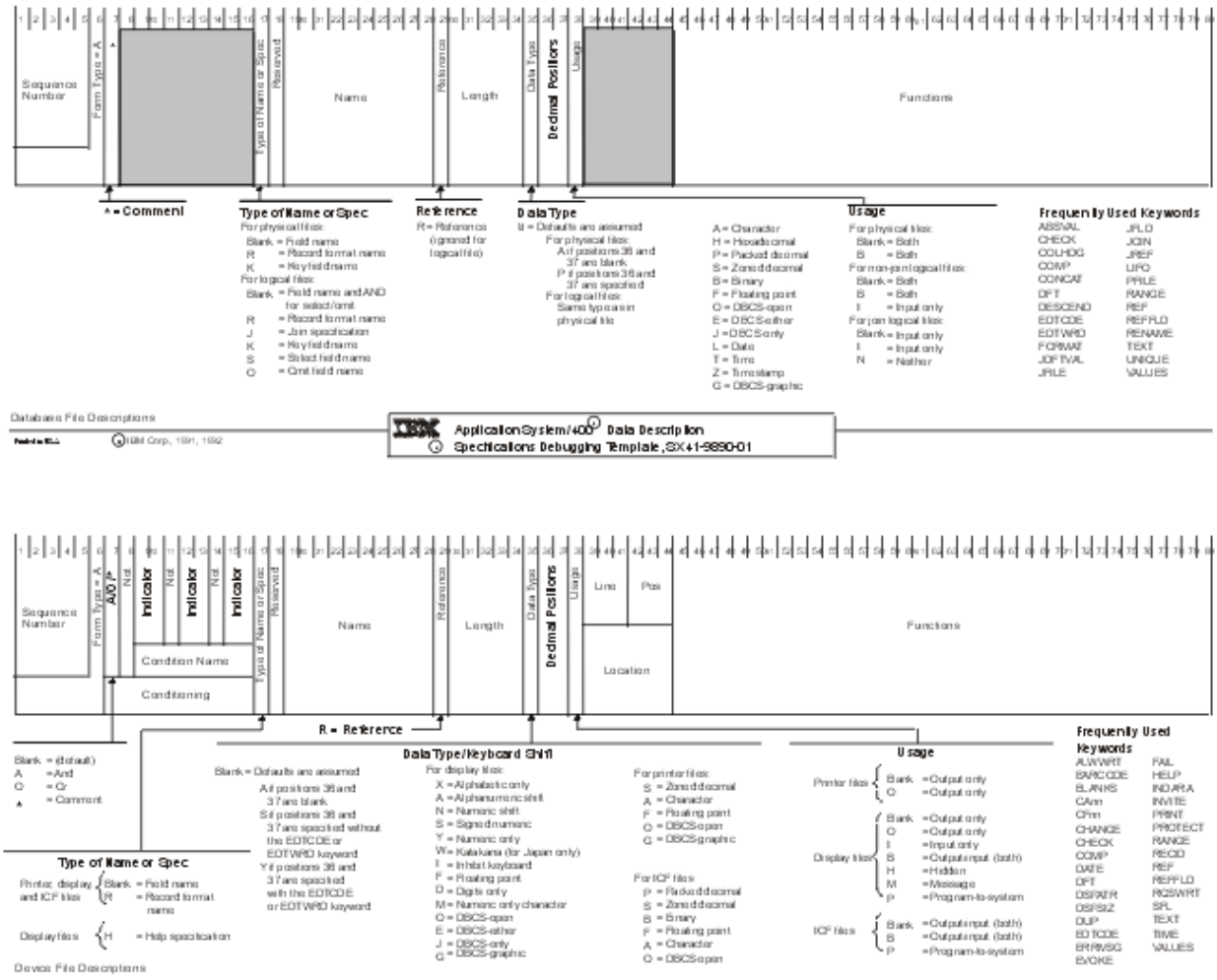


Figure 15. IBM data description specifications debugging template (Reduced)

Related tasks

Creating the DDS file

You can create a data description specifications (DDS) file by running a CL command that corresponds to the type of DDS file.

When to specify REF and REFFLD keywords for DDS files

When you decide whether to specify the REF (reference) keyword, the REFFLD (referenced field) keyword, or both, you need to consider some questions. You also need to know how to specify the parameter values for each REF or REFFLD keyword that you specify.

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 IBM i operating system 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 IBM i operating system 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.

The following example illustrates reference function specifications. It 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.

```
|...+....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 16. Sample Reference Function Specifications

Legend:

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.

Related information for DDS concepts

Product manuals and other information center topic collections contain information that relates to the DDS concepts topic collection. You can view or print any of the PDF files.

Manuals

- Application Display Programming, SC41-5715

  This manual describes the use of display devices by application programs.

The following manuals are not included in the IBM i Information Center. However, these manuals might be a useful reference to you. Each of the manuals is available from the IBM Publications Center as a printed hardcopy that you can order, in an online format that you can download at no charge, or both.

- *Advanced Function Printing: Data Stream Reference*, S544-3202

  This manual provides information about the Advanced Function Printing data stream.
- *ADTS for AS/400: Screen Design Aid*, SC09-2604

  This manual provides the application programmer, system programmer, or data processing manager with information about using the WebSphere Development Studio screen design aid (SDA) to design, create, and maintain display formats and menus.
- *ADTS for AS/400: Source Entry Utility*, SC09-2605

  This manual provides the application programmer or system programmer with information about using the WebSphere Development Studio source entry utility (SEU) to create and edit source members.
- *IBM Personal Computer Enhanced 5250 Emulation Program Version 2.12 Technical Reference*, GS57-0222

  This manual provides the information about the 5250 Emulation Program.
- *Intelligent Printer Data Stream Reference*, S544-3417

  This manual provides information about bar codes and valid check digits.
- *RPG/400® Reference*, SC09-1349

  This manual describes how to design, code, enter, compile, test, and run RPG III programs.

Other information

- Control language describes control language syntax, commands, and command parameters.
- CL programming provides a wide-range discussion of programming topics.

- Work management provides information about how to create and change a work management environment.
- Basic system operations describes how to operate workstations and how to use the function keys to enter commands.
- Backup and recovery contains information about how to plan a backup and recovery strategy and how to back up your system. It also includes information about the Backup, Recovery, and Media Services plug-in to System i® Navigator, information about recovering your system, and answers to some frequently asked questions about backup and recovery.

Related reference

PDF file for DDS concepts
You can view and print a PDF file of this information.

Code license and disclaimer information

IBM grants you a nonexclusive copyright license to use all programming code examples from which you can generate similar function tailored to your own specific needs.

SUBJECT TO ANY STATUTORY WARRANTIES WHICH CANNOT BE EXCLUDED, IBM, ITS PROGRAM DEVELOPERS AND SUPPLIERS MAKE NO WARRANTIES OR CONDITIONS EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, REGARDING THE PROGRAM OR TECHNICAL SUPPORT, IF ANY.

UNDER NO CIRCUMSTANCES IS IBM, ITS PROGRAM DEVELOPERS OR SUPPLIERS LIABLE FOR ANY OF THE FOLLOWING, EVEN IF INFORMED OF THEIR POSSIBILITY:

1. LOSS OF, OR DAMAGE TO, DATA;
2. DIRECT, SPECIAL, INCIDENTAL, OR INDIRECT DAMAGES, OR FOR ANY ECONOMIC CONSEQUENTIAL DAMAGES; OR
3. LOST PROFITS, BUSINESS, REVENUE, GOODWILL, OR ANTICIPATED SAVINGS.

SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF DIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, SO SOME OR ALL OF THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. 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 IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department YBWA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

© Copyright IBM Corp. 1999, 2013

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information 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.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_.

Programming interface information

This SQL call level interface publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of IBM i.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux® is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of IBM or other companies.

Terms and conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

Personal Use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Index

A

abbreviations of values and keywords 9

B

batch source statements 4

C

commands used to create files 5
compiler example 42
completing the DDS form 2
considerations 15
creating files 5
creating files with DDS 2

D

DBCS
    considerations 15
DBCS considerations
    character strings 17
    entering bracketed character strings 17
    printouts 18
DDS form
    completing 2
DDS naming conventions 8
debugging template 44
describing data attributes 1
display files
    creating 2
    syntax coding examples 22
double-byte character set
    considerations 15

E

entering source statements 4
example of compiler listing 42
examples of DDS
    field reference file 25
    horizontal subfile 35
    ICF file 38
    inquiry display 29
    join logical file 28
    logical file with a new record format 27
    logical file with multiple formats and new keys 26
    message subfile 36
    physical file with new record format 26
    printer file 37
    program using physical, display, and printer files 40
    subfile 32, 33
examples of syntax coding 19

F

FLAG parameter 5

G

GENLVL (Severity Level) parameter 5

I

ICF files
    creating 2
    syntax coding examples 24
interactive source statements 4

K

keyword
    abbreviations 9

L

library list 46
logical files
    creating 2
    syntax coding examples 20

N

naming conventions 8

O

OPTION parameter 5
overview of DDS 1

P

PFILE keyword 26
physical files
    creating 2
    syntax coding examples 19
printer files
    creating 2
    syntax coding examples 23

R

REF keyword 26, 45
REFFLD keyword 45
rules for keywords and parameter values 7

S

Severity Level (GENLVL) parameter 5

source statements 4
specifying REF and REFFLD keywords 45
syntax
    keywords and parameter values 7
    rules 7
syntax coding examples
    display files 22
    ICF files 24
    join logical files 21
    logical files 20
    physical files 19
    printer files 23

U

UNIQUE keyword 27

V

value
    abbreviations 9

W

when to specify REF and REFFLD keywords 45

Product Number: 5770-SS1