Part 4. Database and application administration

The database and application administration function in IMS Administration Tool provides a way for you to view, create, and change IMS databases (DBDs) and application views (PSBs).

Topics:

Chapter 12. Database and application administration reference

The database and application administration function in IMS Administration Tool provides a way for you to view, create, and change IMS databases (DBDs) and application views (PSBs).

The function extracts IMS control blocks (DBDs and PSBs) from either the DBDLIB, PSBLIB, ACBLIB, or IMS directory depending on how IMS is configured. Then it decodes the extracted control blocks to readable DBD or PSB source code enabling you to edit the source code through the ISPF interface or the web interface.

After editing the DBD or PSB source code, you can execute the DBD or PSB resource change function or the IMS resource change function to update the IMS environment to reflect changes made to the IMS control blocks, or build JCL to reflect changes at a later time.

When the resource change function is executed, it reads the updated DBD and PSB source code and calls the DBDGEN, PSBGEN, ACBGEN utilities, and, if the IMS catalog is defined in the IMS system, the IMS Catalog Populate utility (DFS3PU00).

IMS Administration Tool automatically determines the DBD library, PSB library, ACB library, IMS directory, and IMS catalog from the parameters and the PROCLIB libraries of the IMS subsystem.

Topics:

DBD and PSB selection

You must select a DBD or a PSB to edit. The DBD and PSB management function extracts the selected control block from either the DBDLIB, PSBLIB, ACBLIB, or IMS directory and decodes the control block to readable DBD or PSB macro source.

Table 16. DBD and PSB selection reference

Option	Description
IMSID	The 1-4 character name of the IMS subsystem.
Resource Type	DBD or PSB object type.
DBD or PSB Filter	Specify a wildcard expression to control the number of DBD or PSB objects that display.
Decoded Source Data Set	The name of the master working data set where DBD- and PSB-related information from a DBDLIB, PSBLIB, ACB library, or the IMS directory is translated into DBD and PSB source code.
Updated Source Data Set	The name of the working data set that contains a duplicate of the decode source data set. Modifications to DBDs or PSBs can be made to the contents of the update source data set.
From Library	Library information and status for the selected IMS subsystem: Discovered indicates that the library is determined from the parameters and the PROCLIB libraries of the IMS subsystem. NA indicates that the library is not registered or could not be determined from the PROCLIB libraries of the IMS subsystem. • DBD or PSB Library Discovered or NA (not available) • ACB Active Library Discovered or NA (not available) • ACB Inactive Library Discovered or NA (not available) • ACB Staging Library Discovered or NA (not available) • Directory Active Data Set Discovered or NA (not available) ACBs managed by IMS catalog or ACB libraries • Directory Staging Data Set Discovered or NA (not available) ACBs managed by IMS catalog or ACB libraries • Specify other DBDLIB data set names or PSBLIB data set names Specify DBD or PSB library data set names to discover different libraries • Specify other ACBLIB data set names Specify ACB library data set names to discover different libraries
Library Information	Data set name information for the libraries enabled on this IMS subsystem.

The following table summarizes the options available after you select a DBD or a PSB. If you select create, alter, or model, you can edit the DBD or the PSB macro source code.

Table 17. DBD and PSB edit option reference

Option	Description
Create	Create a new DBD or PSB.
Alter	Update an existing DBD or PSB. Alter uses the DBD or PSB copy in the update source data set.
Model	Create a new DBD or PSB that is based on (modeled after) the selected DBD or PSB. The new DBD or PSB can then be imported.
Source	View the DBD or PSB code. When working with copybooks, the source view can provide detailed DBD segment information.
Expand Info from IMS	Select an object from an active library (ACB active library or IMS directory active data set) to view detailed (expanded) object information. The detailed information provides a convenient single view of object attributes gathered from multiple sources. For example: • Database level properties • Online status • Data set level properties • Recovery state The expand option is available only through the ISPF interface. If you are using the web interface, detailed (expanded) object information is displayed by default.

To change multiple IMS control blocks, do as follows:

- ISPF interface: After you edit a DBD or a PSB, the DBD Resource Change panel or the PSB Resource Change panel is displayed. With these panels, you can only change a single resource. To change multiple IMS control blocks all at once, after you edit all the DBDs and PSBs that you want to edit, go to the IMS Resource Change panel to process all the edited resources.
- Web interface: The DBD Resource Change panel and the PSB Resource Change panel are not available with the web interface. After you edit DBDs and PSBs, go to the IMS Resource Change panel to process the edited resources.

DBD resource change and PSB resource change

The DBD resource change function and the PSB resource change function call the DBDGEN, PSBGEN, ACBGEN utilities, and if the IMS catalog is defined in the IMS subsystem, populates the IMS catalog by calling the IMS Catalog Populate utility (DFS3PU00).

Review these notes before using the DBD and PSB resource change functions:

- The DBD Resource Change panel and the PSB Resource Change panel are available only through the ISPF interface. If you are using the web interface, the same functionality is available in the IMS Resource Change panel.
- The DBD resource change function and the PSB resource change function can process one DBD or PSB at a time. If you want to change multiple DBDs and PSBs, use the IMS resource change function.

The DBD resource change function calls the DBDGEN utility, the ACBGEN utility, and, if the IMS catalog is defined in the IMS subsystem, the IMS Catalog Populate utility (DFS3PU00) to place resource changes in the IMS subsystem. The copybook import function is also supported to import metadata in COBOL or PL/I copybooks to DBD source.

The PSB resource change function calls the PSBGEN utility, the ACBGEN utility, and, if the IMS catalog is defined in the IMS Catalog Populate utility to place resource changes in the IMS subsystem.

If the IMS catalog is defined in the IMS subsystem, an online IMS system must be active while the DBD or PSB Resource Change function is running.

IMS Administration Tool automatically determines the target DBD library, PSB library, ACB library, IMS directory, and IMS catalog from the parameters and the PROCLIB libraries of the IMS subsystem.

The DBD or PSB resource change updates the ACB staging library or the IMS directory staging data set. After the DBD or PSB resource change completes, you must perform the IMS online change (OLC) or the IMPORT DEFN SOURCE(CATALOG) command to activate DBDs and PSBs in the online IMS system.

Table 18. DBD and PSB change management

Option	Description
Update Data Set	The name of the working data set that contains a duplicate of the decode source data set. Modifications to DBDs or PSBs can be made to the contents of the update source data set.
Member	The name of the changed or newly created DBD or PSB. The DBD or PSB becomes a member of the update data set.
Generation Options:	
Execute (run) or Build JCL	Specify to execute the DBD or PSB resource change or to generate DBD or PSB resource change JCL. For details about the JCL it generates, see Chapter 15, "DBD and PSB update (ATY@OBJU) JCL," on page 111.

Table 18. DBD and PSB change management (continued)

Option	Description
Use COPYBOOK	This option is available only with DBD resource change. Specify Y to import copybook information to the DBD source code. If you specify Y, the function analyzes the copybook and inserts corresponding metadata statements into the DBD source for DBDGEN. **Requirement:** The COBOL or PL/I compiler library DDNAME variable must be registered. **Tip:** To change COBOL compiler options, specify the data set that contains the IGYCDOPT module to DDNAME variable CBLOPT. For details about variables, see "DDname and keyword variables for copybook import" on page 104.
COPYBOOK Cross Reference (XREF) Data Sets	The name of the data set that pairs the DBD with the copybook. You can specify up to 10 data sets. If you are using the ISPF interface, specify Y to view, change, or add data set names. For the format of COPYBOOK XREF files and examples, see "Copybook XREF file" on page 105.
COBOL or PL/I COPYBOOK Data Sets	The names of the data sets where the copybook resides. You can specify up to 120 data sets, maximum of 60 for COBOL copybook data sets and 60 for PL/I copybook data sets. If you are using the ISPF interface, specify Y to view, change, or add data set names. **Requirement:** The compiler library must be specified as a DDNAME variable. DDNAME CBLLIB is for the COBOL compiler library, and DDNAME PLILIB is for the PL/I compiler library. Specify either or both depending on the language of the copybook that you want to import. **Tip:** To change COBOL compiler options, specify the data set that contains the IGYCDOPT module to DDNAME variable CBLOPT. For information about changing DDNAME variables, see Chapter 14, "Copybook import," on page 101.
DBD Source with COPYBOOK	Specify the name of the output data set for storing the updated DBD source.
JCL Output Options:	
JCL Output Data Set	The name of the partitioned data set where the generated JCL is stored. The data set must be pre-allocated before you can generate the JCL
Member	The name of the member in the partitioned data set where the generated JCL is stored.
Job Statements	Specification of the JOB statement of the JCL.
Allocate JCL Output Data Set?	Allocate the data set where the generated JCL is stored.

Tip: If you want to change assembler options used for DBDGEN or PSBGEN, you can do so by describing the options in a sequential data set (PS) and registering the data set to DDNAME variable ASMAOPT.

Chapter 13. IMS resource change

The IMS resource change function calls the DBDGEN, PSBGEN, ACBGEN utilities, and, if the IMS catalog is defined in the IMS subsystem, populates the IMS catalog by calling the IMS Catalog Populate utility (DFS3PU00).

The IMS resource change function can process multiple DBDs and PSBs. Required inputs are DBD and PSB source codes that are decoded and edited. For information about decoding and editing DBD and PSB source, see Chapter 12, "Database and application administration reference," on page 93.

If the IMS catalog is defined in the IMS subsystem, an online IMS system must be active while the IMS resource change function is running.

IMS Administration Tool automatically determines the target DBD library, PSB library, ACB library, IMS directory, and IMS catalog from the parameters and the PROCLIB libraries of the IMS subsystem.

The IMS resource change function updates the ACB staging library or the IMS directory staging data set. After the IMS resource change completes, you must perform the IMS online change (OLC) or the IMPORT DEFN SOURCE(CATALOG) command to activate DBDs and PSBs in the online IMS system.

Table 19. IMS resource change

Option	Description
Object Selection Criteria	Specify the DBDs and PSBs to process.
Object Type	Specify the type of the resources. DBD, PSB, or both.
DBD Source Data Set	Specify the data set that contains the DBD source codes to process.
Select DBDs	Two methods to select DBDs from the DBD update data set: • By filter Specify a wildcard expression to control the number of DBDs that display. • From list View and select DBDs to be updated.
PSB Source Data Set	Specify the data set that contains the PSB source codes to process.
Select PSBs	Two methods to select PSBs from the PSB update data set: • By filter Specify a wildcard expression to control the number of PSBs that display. • From list View and select PSBs to be updated.
IMS Resource Change Options:	
Execute (run) or Build JCL	Specify to execute the IMS resource change or to generate IMS resource change JCL. For details about the JCL it generates, see Chapter 15, "DBD and PSB update (ATY@OBJU) JCL," on page 111.

Table 19. IMS resource change (continued)

Option	Description
Use COPYBOOK	Specify Y to import copybook information to the DBD source code. If you specify Y, the function analyzes the copybook and inserts corresponding metadata statements into the DBD source for DBDGEN. **Requirement:** The COBOL or PL/I compiler library DDNAME variable must be registered. **Tip:** To change COBOL compiler options, specify the data set that contains the IGYCDOPT module to DDNAME variable CBLOPT. For details about variables, see "DDname and keyword variables for copybook import" on page 104.
COPYBOOK Cross Reference (XREF) Data Sets	The name of the data set that pairs the DBD with the copybook. You can specify up to 10 data sets. If you are using the ISPF interface, specify Y to view, change, or add data set names. For the format of COPYBOOK XREF files and examples, see "Copybook XREF file" on page 105.
COBOL or PL/I COPYBOOK Data Sets	The names of the data sets where the copybook resides. You can specify up to 120 data sets, maximum of 60 for COBOL copybook data sets and 60 for PL/I copybook data sets. If you are using the ISPF interface, specify Y to view, change, or add data set names. **Requirement:** The compiler library must be specified as a DDNAME variable. DDNAME CBLLIB is for the COBOL compiler library, and DDNAME PLILIB is for the PL/I compiler library. Specify either or both depending on the language of the copybook that you want to import. **Tip:** To change COBOL compiler options, specify the data set that contains the IGYCDOPT module to DDNAME variable CBLOPT. For information about changing DDNAME variables, see Chapter 14, "Copybook import," on page 101.
DBD Source with COPYBOOK	Specify the name of the output data set for storing the updated DBD source.
JCL Output Options:	
JCL Output Data Set	The name of the partitioned data set where the generated JCL is stored. The data set must be pre-allocated before you can generate the JCL
Member	The name of the member in the partitioned data set where the generated JCL is stored.
Job Statements	Specification of the JOB statement of the JCL.
Allocate JCL Output Data Set?	Allocate the data set where the generated JCL is stored.

Tip: If you want to change assembler options used for DBDGEN or PSBGEN, you can do so by describing the options in a sequential data set (PS) and registering the data set to DDNAME variable ASMAOPT.

Chapter 14. Copybook import

The copybook import function imports metadata in COBOL or PL/I copybooks to DBD source. Copybook import is supported as a part of the database and application administration function and the IMS catalog and ACBLIB management function.

The copybook import function can be called from the following functions:

- IMS resource change function
- DBD resource change function
- Import objects (IMS catalog and ACBLIB management)

After importing metadata from copybooks to the DBD source, the function that called the copybook import function uses the updated DBD source and calls the DBDGEN utility, the PSBGEN utility, the ACBGEN utility, and the IMS Catalog Populate utility (DFS3PU00) to update relevant IMS control blocks in your IMS environment.

The copybook import function uses the following information to import metadata from copybooks to the DBD. You supply the location of the resources through the ISPF interface or the web interface.

DBD The DBD to update.

COBOL or PL/I copybook
One or more copybooks to import.

Copybook cross reference (XREF) file
A file that defines linkage between segments and copybooks.

Output data sets
Output data sets such as for storing updated DBD source and generated DBD resource change JCL.

Subsections:

- "Data attribute mapping"
- "Considerations for importing PL/I copybooks" on page 103

Data attribute mapping

The copybook import function inserts FIELD statements with EXTERNALNAME parameters based on data definitions in the copybook. The copybook import function calculates the start position and the length, and adds START and BYTES parameters. It also adds DATATYPE parameters based on the mapping rules summarized in the following tables:

Table 20. Data attribute mapping from COBOL copybook to DBD DATATYPE

COBOL data type	DBD DATATYPE
PIC S9(4) BINARY	SHORT
PIC S9(9) BINARY	INT
PIC S9(18) BINARY	LONG
PIC 9(4) BINARY	USHORT

Table 20. Data attribute mapping from COBOL copybook to DBD DATATYPE (continued)

COBOL data type	DBD DATATYPE
PIC 9(9) BINARY	UINT
PIC 9(18) BINARY	ULONG
COMP-1	FLOAT
COMP-2	DOUBLE
PIC S9(*n*) COMP-3	DECIMAL(*n*,*p*) INTERNALTYPECONVERTER=PACKEDDECIMAL DFSMARSH statement is added to define data marshaling characteristics.
PIC X(*n*)	CHAR
PIC G(*n*)	BINARY(2*n*)
PIC N(*n*) DISPLAY-1	BINARY(2*n*)
PIC N(*n*) NATIONAL	BINARY(2*n*)
PIC 9(*n*) DISPLAY	DECIMAL(*n*,*p*) INTERNALTYPECONVERTER=ZONEDDECIMAL DFSMARSH statement is added to define data marshaling characteristics.
Field with OCCURS attribute	ARRAY
First element of group items	STRUCT

Table 21. Data attribute mapping from PL/I copybook to DBD DATATYPE

PL/I data type	DBD DATATYPE
REAL FIXED BINARY(15,0)	SHORT
REAL FIXED BINARY(31,0)	INT
REAL FIXED BINARY(63,0)	LONG
REAL FIXED BINARY(16,0) UNSIGNED	USHORT
REAL FIXED BINARY(32,0) UNSIGNED	UINT
REAL FIXED BINARY(64,0) UNSIGNED	ULONG
REAL FLOAT DECIMAL(6)	FLOAT
REAL FLOAT DECIMAL(16)	DOUBLE
FIXED DECIMAL(*n*,*p*)	DECIMAL(*n*,*p*) INTERNALTYPECONVERTER=PACKEDDECIMAL DFSMARSH statement is added to define data marshaling characteristics.
CHAR(*n*)	CHAR
GRAPHIC(*n*)	BINARY(2*n*)
WIDECHAR(*n*)	BINARY(2*n*)
PICTURE '(*n*)9'	CHAR(*n*)
WIDEPIC '(*n*)9'	CHAR(2*n*)
CHAR(*n*) VAR	CHAR(*n*)+2

Table 21. Data attribute mapping from PL/I copybook to DBD DATATYPE (continued)

PL/I data type	DBD DATATYPE
CHAR(*n*) VARYING4	CHAR(*n*)+4
CHAR(*n*) VARYINGZ	CHAR(*n*)+1
Field with array attribute	ARRAY
First element of structure	STRUCT

Considerations for importing PL/I copybooks

The following considerations apply when you import copybooks written in PL/I.

- The length of variable names specified in a PL/I copybook must be equal to or less than 30. Otherwise, the variable names will be truncated.
- When a structure in a PL/I copybook contains an array with the REFER option (variable for declared length), the PL/I compiler does not provide sufficient information about that structure. This may result in having an incorrect length in the DBD source.

  In the following PL/I copybook example, Y is an array with 20 bytes. However, when this structure is imported, the length is changed to 2 bytes in the DBD source.

```
DECLARE 1 STR BASED(P),
            2 X FIXED BINARY(31,0),
            2 Y (10 REFER (X)),
              3 DATA CHAR(2);
```

  To prevent this, review and remove all REFER options in the PL/I copybook before you import or update from the PL/I copybook.

Topics:

DDname and keyword variables for copybook import

Before you can import metadata from copybooks, you must register the following DDname and keyword variables using the ISPF interface or the web interface.

To register DDname and keyword variables:

- ISPF: **0 Setup and Administration** > **1 Update Product Registry** > **3 Variable Management**
- Web interface: **Setup and Admin** > **Variable Management**

COBOL and PL/I compiler library

Required DDname variable. Register the language compiler library for COBOL, PL/I, or both.

Variable name	Description
CBLLIB	Specify the name of the COBOL compiler library data set.
PLILIB	Specify the name of the PL/I compiler library data set.

Copybook XREF format

Optional keyword variable. The copybook XREF file has two formats, type-0 and type-1. Type-0 is supported for both COBOL and PL/I. Type-1 is supported only for COBOL. Type-0 is the default. If you want to use type-1, you must register this keyword variable.

Variable name	Description
XREFFORM	Specify the format of the copybook XREF file, TYPE1 or TYPE0. If omitted, the default is TYPE0.

For more information about the format of copybook XREF files, see "Copybook XREF file" on page 105.

Copybook language

Optional keyword variable. If the copybook XREF file has type-0 format, XREF statements contain the copybook language, either COBOL or PL/I. This keyword variable overrides the language specified on the XREF statements. The default is COBOL. If you mainly use PL/I, you can change the value to PLI.

Variable name	Description
COPYLANG	Specify the language of the copybook, PLI or COBOL. If omitted, the default is COBOL.

COBOL compiler option

Optional DDname variable. If you want to change the COBOL compiler options, specify the data set that contains the IGYCDOPT module.

The data set is a load library and the data set organization must be RECFM=U, LRECL=0.

Variable name	Description
CBLOPT	Specify the data set that contains the IGYCDOPT module.

Copybook XREF file

A copybook XREF file contains copybook XREF statements that define mapping of each copybook to a segment.

A copybook XREF file is a PDS or PDSE, attributes are RECFM=FB and LRECL=80. The member name must match the name of the DBD to map.

Two formats are supported for copybook XREF files, type-0 and type-1. Type-0 can be used for both COBOL and PL/I, type-1 can be used for COBOL only. Type-0 is assumed unless the copybook XREF format keyword variable (XREFFORM) is set to TYPE1.

Subsections:

- "Type-0 copybook XREF statement syntax "
- "Type-1 copybook XREF statement syntax" on page 106

Type-0 copybook XREF statement syntax

Type-0 copybook XREF file supports both COBOL and PL/I. Each XREF statement specifies the language of the copybook, either COBOL or PL/I.

The following figure shows the syntax for type-0 copybook XREF statements.

```
----+----1----+----2----+----3----+----4----+
SEGM=SEGMENT1 COPYBOOK=SEG1COPY LANG=COBOL
SEGM=SEG2     COPYBOOK=S2COPY   LANG=COBOL
|     |       |         |       |
|     |       |         |       |
|     |       |         |       |-Col33-42 LANG=COBOL or LANG=PLI
|     |       |         |-Col24-31 Copybook name
|     |       |-Col15-23 Keyword
|     |-Col6-13 Segment name
|-Col1-5 Keyword
```

Figure 5. Type-0 copybook XREF statement syntax

Position	Description
Columns 1 - 5	Specify the SEGM= keyword.
Columns 6 - 13	Specify, left-aligned, a segment name.
Column 14	Filler. A blank or any character. The character in this column is ignored.
Columns 15 - 23	Specify the COPYBOOK= keyword.
Columns 24 - 31	Specify, left-aligned, the name of the copybook to map the segment. The name of the copybook must match a member in the copybook data set.
Column 32	Filler. A blank or any character. The character in this column is ignored.

Position	Description
Columns 33 - 42	Optional. Specify LANG=COBOL or LANG=PLI. If omitted, LANG=COBOL is applied. To change the default language, set the copybook language keyword variable (COPYLANG). For more information, see copybook language in "DDname and keyword variables for copybook import" on page 104.

Type-1 copybook XREF statement syntax

Type-1 copybook XREF file supports COBOL only. To use a type-1 copybook XREF file, you must set the copybook XREF format keyword variable (XREFFORM) to TYPE1. For more information, see copybook XREF format in "DDname and keyword variables for copybook import" on page 104.

The following figure shows the syntax for type-1 copybook XREF statements.

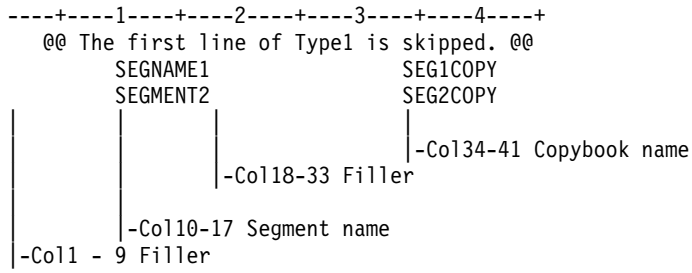

Figure 6. Type-1 copybook XREF statement syntax

The first line is ignored. You can write comments on this line.

Position	Description
Columns 1 - 9	Filler. Blanks or any characters. Characters in these columns are ignored.
Columns 10 - 17	Specify, left-aligned, a segment name.
Columns 18 - 33	Filler. Blanks or any characters. Characters in these columns are ignored.
Columns 34 - 41	Specify, left-aligned, the name of the copybook to map the segment. The name of the copybook must match a member in the copybook data set.

Examples for copybook import

Use the following example to learn how to use the copybook import function.

In this example:

- DBD name is ATYDBD0. The DBD has two segments, ATYSEG1 and ATYSEG2.
- The copybook data set contains two members, ATYCOPY1 and ATYCOPY2. The language used for the copybooks is COBOL.
- The name of the copybook XREF file is ATYDBD0, which is the same as the DBD name. This file exists in the ATY.XREF data set. The format of the copybook XREF file is type-0.
- The copybook XREF file contains the following copybook XREF statements:

  ```
  SEGM=ATYSEG1 COPYBOOK=ATYCOPY1 LANG=COBOL SEGM=ATYSEG2 COPYBOOK=ATYCOPY2 LANG=COBOL
  ```

The decoded DBD (DBD source) contains the following statements.

```
         DBD   NAME=ATYDBD0,ACCESS=(HDAM,OSAM),                        X
               RMNAME=(DFSHDC40,8,360,3000)
*
DS1    DATASET DD1=SAMPL0,SIZE=(4096),SCAN=0
*
 SEGM    NAME=ATYSEG1,BYTES=20,PARENT=0,RULES=(LLL,LAST),             X
               PTR=(TWIN,,,,)
         FIELD NAME=(FLD1,SEQ,U),BYTES=10,START=1,TYPE=C
         FIELD NAME=(FLD2),BYTES=10,START=11,TYPE=C
*
 SEGM    NAME=ATYSEG2,BYTES=40,PARENT=((ATYSEG1,)),                   X
               PTR=(TWIN,,,,),RULES=(LLL,LAST)
         FIELD NAME=(FLD10,SEQ,U),BYTES=30,START=1,TYPE=C
         FIELD NAME=(FLD20),BYTES=5,START=31,TYPE=C
         FIELD NAME=(FLD30),BYTES=5,START=31,TYPE=C
*
 DBDGEN
 FINISH
 END
```

Figure 7. DBD source (decoded)

The following examples show the contents of copybooks ATYCOPY1 and ATYCOPY2. The names of the members in the copybook data set are ATYCOPY1 and ATYCOPY2.

```
000100*************************************************************00010000
000200*   SAMPLE COPYBOOK FOR DBD ATYDBD0                           00020000
000300*   ATYSEG1 SEGMENT                                           00030002
000400*************************************************************00040000
000500*                                                             00050000
000600 01  STRUCT-FIELD0.                                          00060001
000700     10 FIELD1                  PIC X(5).                   00070000
000800     10 FIELD2                  PIC X(10).                  00080000
000900     10 FIELD3                  PIC X(5).                   00090000
```

Figure 8. Content of copybook ATYCOPY1

```
000100*************************************************************************00010000
000200*   SAMPLE COPYBOOK FOR DBD ATYDBD0                                      00020000
000300*   ATYSEG2 SEGMENT                                                      00030001
000400*************************************************************************00040000
000500*                                                                        00050000
000600 01  STRUCT-FIELD10.                                                    00060000
000700     10 FIELD11                 PIC X(2).                              00070000
000800     10 FIELD12                 PIC X(10).                             00080000
```

Figure 9. Content of copybook ATYCOPY2

After the copybook import function imports metadata in copybook ATYCOPY1 to segment ATYSEG1 and metadata in copybook ATYCOPY2 to segment ATYSEG2, the DBD source is updated as follows:

```
         DBD   NAME=ATYDBD0,ACCESS=(HDAM,OSAM),                         X
               RMNAME=(DFSHDC40,8,360,3000)
*
DS1    DATASET DD1=SAMPL0,SIZE=(4096),SCAN=0
*
 SEGM    NAME=ATYSEG1,BYTES=20,PARENT=0,RULES=(LLL,LAST),               X
               PTR=(TWIN,,,,)
         FIELD NAME=(FLD1,SEQ,U),BYTES=10,START=1,TYPE=C
         FIELD NAME=(FLD2),BYTES=10,START=11,TYPE=C
*
         FIELD EXTERNALNAME=STRUCT_FIELD0,                              +
               BYTES=20,                                                +
               START=1,                                                 +
               DATATYPE=STRUCT,                                         +
               REMARKS='Generated from Copybook ATYCOPY0 imported 2019/+
               08/01 04:15:08 by TS6444  '
         FIELD EXTERNALNAME=FIELD1,                                     +
               PARENT=STRUCT_FIELD0,                                    +
               BYTES=5,                                                 +
               START=1,                                                 +
               DATATYPE=CHAR,                                           +
               REMARKS='Generated from Copybook ATYCOPY0 imported 2019/+
               08/01 04:15:08 by TS6444  '
         FIELD EXTERNALNAME=FIELD2,                                     +
               PARENT=STRUCT_FIELD0,                                    +
               BYTES=10,                                                +
               START=6,                                                 +
               DATATYPE=CHAR,                                           +
               REMARKS='Generated from Copybook ATYCOPY0 imported 2019/+
               08/01 04:15:08 by TS6444  '
         FIELD EXTERNALNAME=FIELD3,                                     +
               PARENT=STRUCT_FIELD0,                                    +
               BYTES=5,                                                 +
               START=16,                                                +
               DATATYPE=CHAR,                                           +
               REMARKS='Generated from Copybook ATYCOPY0 imported 2019/+
               08/01 04:15:08 by TS6444  '
```

Figure 10. DBD source updated with copybook (Part 1 of 2)

```
 SEGM    NAME=ATYSEG2,BYTES=40,PARENT=((ATYSEG1,)),                    X
               PTR=(TWIN,,,,),RULES=(LLL,LAST)
         FIELD NAME=(FLD10,SEQ,U),BYTES=30,START=1,TYPE=C
         FIELD NAME=(FLD20),BYTES=5,START=31,TYPE=C
         FIELD NAME=(FLD30),BYTES=5,START=31,TYPE=C
*
         FIELD EXTERNALNAME=STRUCT_FIELD10,                            +
               BYTES=12,                                               +
               START=1,                                                +
               DATATYPE=STRUCT,                                        +
               REMARKS='Generated from Copybook ATYCOPY1 imported 2019/+
               08/01 04:15:08 by TS6444  '
         FIELD EXTERNALNAME=FIELD11,                                   +
               PARENT=STRUCT_FIELD10,                                  +
               BYTES=2,                                                +
               START=1,                                                +
               DATATYPE=CHAR,                                          +
               REMARKS='Generated from Copybook ATYCOPY1 imported 2019/+
               08/01 04:15:08 by TS6444  '
         FIELD EXTERNALNAME=FIELD12,                                   +
               PARENT=STRUCT_FIELD10,                                  +
               BYTES=10,                                               +
               START=3,                                                +
               DATATYPE=CHAR,                                          +
               REMARKS='Generated from Copybook ATYCOPY1 imported 2019/+
               08/01 04:15:08 by TS6444  '
 DBDGEN
 FINISH
 END
```

Figure 11. DBD source updated with copybook (Part 2 of 2)

Chapter 15. DBD and PSB update (ATY@OBJU) JCL

ATY@OBJU JCL, also referred to as DBD and PSB update JCL, updates DBDs and PSBs.

ATY@OBJU JCL is generated by the following functions, regardless of whether the ISPF interface or the web interface is used:

- The Build JCL option of the DBD resource change function, the PSB resource change function, and the IMS resource change function (database and application administration)
- Import objects (IMS catalog and ACBLIB management)

The ATY@OBJU job performs DBDGEN, PSBGEN, ACBGEN, and, if IMS catalog is defined in the IMS system, the IMS catalog populate utility (DFS3PU00). If the Use COPYBOOK option is selected, the ATY@OBJU job also performs copybook import before DBDGEN.

The target DBDLIB, PSBLIB, ACBLIB, IMS directory, and IMS catalog are automatically determined by IMS Administration Tool from the parameters and the PROCLIB libraries of the IMS system.

Requirements:

- The IMS Tools Base Knowledge Base server and the Distributed Access Infrastructure (DAI) TCP server, TAS, and SOT address spaces must be active.
- The IMS system must be either active or inactive under certain circumstances. See “Requirement: Status of the online IMS system” on page 114.

Topics:

ATY@OBJU JCL statements

The data sets specified in the JCL are automatically determined from the parameters and PROCLIB of the IMS system.

You can change the data sets if you want to use other DBDLIB, PSBLIB, and ACBLIB data sets. IMS directory and IMS catalog cannot be specified in the JCL. If you do not want to update the IMS catalog and the IMS directory, you can suppress the IMS catalog populate step. For more information, see "ATYMSGI DD" statement.

STEPLIB DD
The product and customized loadlib data sets of IMS Administration Tool and IMS Tools Base.

Input DD statements

ATYMSGI DD
Pre-coded internal control statements of ATY@OBJU.

- FUNCTION=UPDATE is set if the JCL was generated by DBD, PSB, or IMS resource change.
- FUNCTION=IMPORT is set if the JCL was generated by Import Objects.

Generally, you do not need to change the ATYMSGI control statements. However, if you do not want to update the IMS catalog and the IMS directory, modify as follows so that only the IMS catalog populate (DFS3PU00) step is suppressed.

- CATALOG=N
- PENDCAT=N
- INITILOAD=N, if exists.

ATYDBD DD
The data set that contains DBD source codes. The data set organization is PDS or PDSE, RECFM=FB,LRECL=80. This DD statement is required when ATYMSGI FUNCTION=UPDATE.

ATYPSB DD
The data set that contains PSB source codes. The data set organization is PDS or PDSE, RECFM=FB,LRECL=80. This DD statement is required when ATYMSGI FUNCTION=UPDATE.

ATYXREF DD
The data sets that contain cross reference (XREF) files for copybook import. Up to 10 data sets can be specified. The data set organization is RECFM=FB,LRECL=80.

This DD statement is present if you selected the Use COPYBOOK option when generating the ATY@OBJU JCL. For details about copybook import, see Chapter 14, "Copybook import," on page 101.

ATYPLI DD
The data sets that contain PL/I copybooks. Up to 60 data sets can be specified. The data set organization is RECFM=FB,LRECL=80.

This DD statement is present if you selected the Use COPYBOOK option when generating the ATY@OBJU JCL.

ATYCOPY DD
The data sets that contain COBOL copybooks. Up to 60 data sets can be specified. The data set organization is RECFM=FB,LRECL=80.

This DD statement is present if you selected the Use COPYBOOK option when generating the ATY@OBJU JCL.

Output DD statements

ATYPUTDB DD
The data sets where IMS Administration Tool stores DBD source codes that are updated with copybooks. The data set organization is RECFM=FB,LRECL=80.

This DD statement is used if you selected the Use COPYBOOK option when generating the ATY@OBJU JCL.

DBDLIB DD
The IMS DBD library. This DBD library will be updated by DBDGEN, and will be referred to during ACBGEN.

PSBLIB DD
The IMS PSB library. This PSB library will be updated by PSBGEN, and will be referred to during ACBGEN.

IMSACBA DD
IMSACBB DD
IMSACB DD
IMS active, inactive, and staging ACB libraries. The IMSACB (staging ACB library) will be updated by ACBGEN, and will be referred to by the IMS catalog populate utility.

DBDPRINT DD
PSBPRINT DD
LNKPRINT DD
SYSPRINT DD
ATYMSGS DD
ATYERROR DD
FABXAMSG DD
Output destination for reports, messages, and assemble listing.

Requirement: Status of the online IMS system

The ATY@OBJU job updates or initializes the IMS catalog and the IMS directory by performing the IMS catalog populate utility (DFS3PU00). The online IMS system must be either active or inactive depending on the INITLOAD parameter value of the ATYMSGI control statement.

The INITLOAD parameter value in the ATYMSGI control statement is inherited from the Initial Load value that is specified through the web interface or the ISPF interface.

If INITLOAD=N, or INITLOAD is not present in the ATYMSGI control statement

The IMS system must be active while the ATY@OBJU job is running because the job updates the existing IMS catalog and the IMS directory with BMP.

The ACB members of DBDs and PSBs are stored in the ACB staging library or the IMS directory staging data set. After the ATY@OBJU job completes, you must perform the IMS online change (OLC) or the IMPORT DEFN SOURCE(CATALOG) command to activate DBDs and PSBs in the IMS system.

If INITLOAD=Y is present in the ATYMSGI control statement

The ATY@OBJU initializes (create, or delete and define) an IMS catalog and IMS directory.

- If the IMS management of ACBs is enabled, the online IMS system must be inactive. The ACB members of DBDs and PSBs are stored in the IMS directory active data set. When the IMS online system starts, the ACB members in the IMS directory active data set are loaded to IMS.
- If the IMS management of ACBs is not enabled, the online IMS system must be inactive, or the online IMS system must be active with /DBR commands issued against the IMS catalog database and index (/DBR'd). The ACB members of DBDs and PSBs are stored in the ACB library staging data set. You must perform the IMS online change (OLC) to activate DBDs and PSBs in the IMS system.

Scenarios for "Initial Load" and "Overwrite Existing Objects"

DBD and PSB objects to update and the destination data sets (ACB active or staging library, IMS directory active or staging data sets) are determined based on the Overwrite Existing Objects option and the Initial Load option.

The following scenarios explore the possible object management combinations and describe the effect of these options on each environment.

1) IMS management of ACBs is enabled

The following conditions apply to this scenario:

- IMS catalog is defined to IMS.
- IMS environment uses IMS directory to manage ACBs. (ACBMGMT=CATALOG is present in the DFSDF*xxx* member)
- IMS environment does not use ACB library to manage ACBs.

  Note: The import objects function requires an ACB library to run ACBGEN and the IMS catalog populate utility (DFS3PU00). The function obtains the ACB library data set information from the IMSACB DD statement in the IMS control region JCL.

How the import objects function performs in this scenario:

Overwrite Existing Objects is Yes and Initial Load is No

- Performs DBDGEN, PSBGEN, and ACBGEN against all the selected DBDs and PSBs. If any of the existing objects in the ACB library has the same name, they are overwritten.
- Performs DFS3PU00 to populate the IMS directory staging data set and the IMS catalog with the new and updated DBDs and PSBs from the ACB library.

Overwrite Existing Objects is Yes and Initial Load is Yes

- Performs DBDGEN, PSBGEN, and ACBGEN against all the selected DBDs and PSBs. If any of the existing objects in the ACB library has the same name, they are overwritten.
- Performs DFS3PU00. After initializing (delete and define) the IMS catalog and the IMS directory, it populates the IMS directory active data sets and the IMS catalog with all DBDs and PSBs (new, updated, and existing) from the ACB library.

Overwrite Existing Objects is No and Initial Load is No

- Checks the ACB library to determine if members with the same names exist in the ACB library. Performs DBDGEN, PSBGEN, and ACBGEN only against new DBDs and PSBs so that no existing objects are overwritten.
- Performs DFS3PU00 to populate the IMS directory staging data set and the IMS catalog with the new objects from the ACB library.

Overwrite Existing Objects is No and Initial Load is Yes

- Checks the ACB library to determine if members with the same names exist in the ACB library. Performs DBDGEN, PSBGEN, and ACBGEN only against new DBDs and PSBs so that no existing objects are overwritten.

- Performs DFS3PU00. After initializing (delete and define) the IMS catalog and the IMS directory, it populates the IMS directory active data sets and the IMS catalog with all DBDs and PSBs (new and existing) from the ACB library.

2) IMS catalog is enabled and IMS Management of ACBs is not enabled

The following conditions apply to this scenario:

- IMS catalog is defined to IMS.
- IMS environment uses ACB library to manage ACBs.

How the import objects function performs in this scenario:

Overwrite Existing Objects is Yes and Initial Load is No:

- Performs DBDGEN, PSBGEN, and ACBGEN against all the selected DBDs and PSBs. If any of the existing objects in the ACB staging library has the same name, they are overwritten.
- Performs DFS3PU00 to populate the IMS catalog with the new and updated DBDs and PSBs from the ACB staging library.

Overwrite Existing Objects is Yes and Initial Load is Yes

- Performs DBDGEN, PSBGEN, and ACBGEN against all the selected DBDs and PSBs. If any of the existing objects in the ACB staging library has the same name, they are overwritten.
- Performs DFS3PU00. After initializing (delete and define) the IMS catalog and the IMS directory (see Note), it populates the IMS catalog with all DBDs and PSBs (new, updated, and existing) from the ACB staging library.

Overwrite Existing Objects is No and Initial Load is No

- Checks the ACB staging library to determine if members with the same names exist in the ACB staging library. Performs DBDGEN, PSBGEN, and ACBGEN only against new DBDs and PSBs so that no existing objects are overwritten.
- Performs DFS3PU00 to populate the IMS catalog with the new DBDs and PSBs from the ACB staging library.

Overwrite Existing Objects is No and Initial Load is Yes

- Checks the ACB staging library to determine if members with the same names exist in the ACB staging library. Performs DBDGEN, PSBGEN, and ACBGEN only against new DBDs and PSBs so that no existing objects are overwritten.
- Performs DFS3PU00. After initializing (delete and define) the IMS catalog and the IMS directory (see Note), it populates the IMS catalog with all DBDs and PSBs (new and existing) from the ACB staging library.

Note: Although IMS directory is not used in this environment, IMS directory data sets, if they exist, are deleted and defined.

3) IMS catalog is not enabled

The following conditions apply to this scenario:

- IMS catalog is not defined to IMS.

- IMS environment uses ACB library to manage ACBs.

In this scenario, the Initial Load option is not available.

How the import operation performs in this scenario:

Overwrite Existing Objects is Yes
Performs DBDGEN, PSBGEN, and ACBGEN against all the selected DBDs and PSBs. If any of the existing objects in the ACB staging library has the same name, they are overwritten.

Overwrite Existing Objects is No
Checks the ACB staging library to determine if members with the same names exist in the ACB staging library. Performs DBDGEN, PSBGEN, and ACBGEN only against new DBDs and PSBs so that no existing objects are overwritten.

Part 5. IMS catalog management

The IMS catalog is a system database that, when enabled, stores the definitions of your databases and program specification blocks (PSBs), as well as other metadata about your databases and application programs.

Topics:

Chapter 16. IMS catalog overview

The IMS catalog is an optional system database that, when enabled, stores trusted metadata and definitions about your databases (DBDs) and application program specification blocks (PSBs) that are defined to IMS.

The IMS catalog is itself a HALDB PHIDAM database. Each database and application program view that is defined to IMS is stored in a separate record in the IMS catalog. In each record, the root header segment identifies the type of resource that it contains: either a database definition (DBD) or a program view (PSB).

Depending on whether you enable the IMS management of application control blocks (ACBs), you have different options for how you define databases and program views, add them to the IMS catalog, and activate them in the IMS system.

When IMS manages the ACBs, you can define databases and program views either by using SQL data definition language (DDL) statements or by using the input macros of the DBD Generation utility and PSB Generation utility.

When you use DDL statements, IMS can add the database and program view definitions to the IMS catalog, build the required runtime control blocks, and, in some cases, load them into the online IMS system automatically.

When you use the DBD and PSB Generation utilities to define databases and program views in an IMS system that manages ACBs, after you run the utilities, you must also run the ACB Generation and Populate utility (DFS3UACB) or equivalent utilities to build the ACBs, update the IMS catalog, and load the ACBs into the IMS system.

In an IMS system that manages ACBs, the IMS catalog completely replaces DBD, PSB, and ACB libraries as the component that determines which database and program view definitions are used by the online IMS system and by batch application programs.

When the IMS management of ACBs is disabled, you cannot use DDL to define databases and program views. Instead, you must define them by using the DBD and PSB Generation utilities, you must generate members into an ACB library, and you must use the online change process to activate the ACB library. You must also make sure that the IMS catalog remains in sync with the active ACB libraries.

The IMS catalog serves to make IMS data more widely and easily accessible outside of the mainframe. The catalog's trusted and comprehensive view of IMS database metadata, fully managed by IMS, allows IMS to participate in solutions that require the exchange of metadata. An example of a solution that requires such an exchange is business impact analysis.

IMS directory data sets

When the IMS management of application control blocks (ACBs) is enabled, IMS stores the active ACBs in the IMS directory, a collection of system-managed data sets that are an extension of the IMS catalog. The IMS directory data sets include:

- Data sets for the ACBs that are active in the IMS system.

- A staging data set for ACBs that are pending activation.
- A bootstrap data set that IMS uses to manage the IMS directory.

The IMS directory data sets that store the active and pending ACBs are functionally similar to the ACB library (ACBLIB) data sets that you would use to manage ACBs when the IMS management of ACBs is not enabled.

Unlike an active ACBLIB data set, the active ACB data sets of the IMS directory are system data sets that IMS creates, updates, and manages automatically. IMS automatically allocates the data sets for the IMS directory and keeps the IMS directory in sync with the IMS catalog. When an active ACB data set becomes full, IMS automatically allocates another data set.

When IMS ACB management is enabled:

- IMS references the directory data sets to get the runtime application control blocks
- IMS uses the directory to indicate which members are active in the IMS catalog

Catalog and non-catalog IMS environments

The IMS catalog contains trusted metadata and definitions of the IMS databases and application program views that are defined to IMS.

If IMS management of ACBs is enabled, the IMS catalog also determines the active databases and program views (PSBs) in the IMS system, because ACB libraries are not used.

When IMS uses ACB libraries, the ACB library determines which databases and program views are active, and you must ensure that the IMS catalog is always in synch with the ACB library.

When the IMS catalog is enabled, the following scenarios are possible for ACB management:

- IMS catalog enabled, no ACBLIB present, ACBs managed by IMS.
- IMS catalog enabled, ACBLIB present, ACBs managed by IMS.
- IMS catalog enabled, ACBLIB present, ACBs managed by ACBLIB.
- IMS catalog not enabled, ACBLIB present, ACBs managed by ACBLIB.

1) Catalog enabled - No ACBLIB - IMS management of ACBs

The following conditions apply to this scenario:

- IMS environment is catalog-enabled
- IMS environment does not use ACB library
- IMS management of ACBs

Environment characteristics:

- IMS stores ACBs in the IMS directory.

  The IMS directory is a collection of system-managed data sets that are an extension of the IMS catalog.
- IMS stores and refers to active ACBs in IMS directory active data sets.
- IMS stores and refers to pending ACBs in the IMS directory staging data set.

  The pending ACBs are new or changed objects that are imported with more recent timestamps than active ACBs.
- DDL is the only mechanism available to update objects in the IMS directory.

2) Catalog enabled - ACBLIB used - IMS management of ACBs

The following conditions apply to this scenario:

- IMS environment is catalog-enabled
- IMS environment uses ACB library
- IMS management of ACBs

Environment characteristics:

- ACBLIB can be present even for IMS systems that have catalog managed ACBs. Reasons for this configuration include:
  1. IMS environment is being converted to IMS management of ACBs (catalog) and ACBLIB is kept present for fallback purposes.
  2. The IMS instance is part of an IMSPLEX, and not all members of the plex have been converted to use IMS management of ACBs.

3. Administrators do not want to be limited to using DDL to control database and program definitions.

- Both ACBLIB and IMS directory (catalog) are used to update DBD and PSB definitions.

  Synchronization of objects between ACBLIB and IMS directory (catalog) is the responsibility of the administrator.
- To update objects in the IMS directory (catalog), use DDL, or alternatively use ACBGEN and the IMS Catalog Populate utility.
- DBDLIBs are still required for certain types of DBDs (GSAM and logical DBDs).

3) Catalog enabled - ACBLIB used - ACBLIB management of ACBs

The following conditions apply to this scenario:

- IMS environment is catalog-enabled
- IMS environment uses ACB library
- ACBLIB management of ACBs

Environment characteristics:

- DBDs are managed in DBDLIBs.
- PSBs are managed in PSBLIBs.
- ACBs are managed in ACBLIB.

  ACBLIBs contain pre-processed DBDs and PSBs.

  Pre-processing meaning that an IMS utility has already performed some validation and outputs the DBDs and PSBs into a format where the IMS online system only needs to load them in order to use them.
- Each IMS environment using ACBLIB typically has:
  - A staging ACBLIB
  - An inactive ACBLIB
  - An active ACBLIB
- Typically there is just one staging ACBLIB, one inactive ACBLIB, and one active ACBLIB per IMS environment.

  However, some environments have more than one of each type of ACBLIB.
- The catalog is not automatically updated with the new and updated objects.

  Synchronization of objects between ACBLIB and IMS directory (catalog) is the responsibility of the administrator.
- To update objects in the catalog, use ACBGEN and the IMS Catalog Populate utility.
- Synchronization of new and updated objects between ACBLIB and catalog is recommended because extended information in DBDs is required by applications using SQL.

  This extended information comes from the catalog even in an environment with ACBLIB management of ACBs.

4) Catalog not enabled - ACBLIB used - ACBLIB management of ACBs

The following conditions apply to this scenario:

- IMS environment is not catalog-enabled
- IMS environment uses ACB library

- ACBLIB management of ACBs

Environment characteristics:

- DBDs are managed in DBDLIBs.
- PSBs are managed in PSBLIBs.
- ACBs are managed in ACBLIB.

  ACBLIBs contain pre-processed DBDs and PSBs.

  Pre-processing meaning that an IMS utility has already performed some validation and outputs the DBDs and PSBs into a format where the IMS online system only needs to load them in order to use them.
- Each IMS environment using ACBLIB typically has:
  - A staging ACBLIB
  - An inactive ACBLIB
  - An active ACBLIB
- Typically there is just one staging ACBLIB, one inactive ACBLIB, and one active ACBLIB per IMS environment.

  However, some environments have more than one of each type of ACBLIB.

IMS catalog management business scenarios

IMS catalog analysis and validation functions allow you to:

- Copy objects between the IMS catalog on one IMS system to the IMS catalog on another IMS system.
- Compare versions of DBD and PSB resources between the IMS catalog and the IMS ACB library.
- Generate reports to help analyze the databases and applications defined in the IMS catalog.
- Perform space utilization analysis and view the number of objects and instances in the IMS catalog.
- Perform impact analysis when either 1) planning for the IMS catalog or 2) adding a large number of objects to the IMS catalog.
- Include and update individual (or bulk) IMS database definitions (DBD) with schema from COBOL or PL/I copybooks during the import process to the IMS catalog.

  Adding or updating schema to individual databases or in bulk can be accomplished either interactively or schedule through a batch process.

Chapter 17. IMS catalog space analysis and summary reports

IMS catalog database analysis and validation functions allow you to view the number of objects and instances in the IMS catalog, determine IMS catalog database space utilization status, and perform impact analysis for both initial IMS catalog planning and the addition of large number of objects to the existing IMS catalog.

IMS catalog analysis and validation provides three report views:

1. IMS catalog database space analysis
   - IMS catalog environment
   - IMS catalog database space usage
   - Program and database instances in IMS catalog database
2. DBD and PSB summary reports
3. DBD and PSB detail reports

Note: The IMSID selection list only shows IMS subsystems that have the IMS catalog enabled and populated.

Note: When the IMS control region is active in a z/OS LPAR where the IMS Tools Base Distributed Access Infrastructure (DAI) server is not running, the IMSID must be in a data sharing group.

IMS catalog analysis issues DL/I calls to the IMS catalog database. Therefore, data sharing must be configured for the IMS systems so that they can communicate with the LPAR where the DAI server is running.

Use the IRLM to configure data sharing for the IMS systems. Then create an IMS data sharing group for IMS Administration Tool and register the IMS systems to the group. The IRLM of one of the IMS systems in the group must be defined to the LPAR where the DAI server is running.

Analysis and report terminology

For DBD and PSB analysis and report details, DBDs and PSBs are known as **objects**.

Objects can be further distinguished as resources and instances:

- **Resource** refers to a DBD object that is identified by a DBD name, or a PSB object that identified by a PSB name.
- **Instance** refers to a specific time/date occurrence of a resource.

  For example, a PSB resource can have multiple instances with different time-stamps.

Space analysis: IMS catalog environment

The IMS catalog environment report displays the following information:

- IMS ID
- IMS version
- Managed ACBs

ACBLIB
ACBs are managed by ACB libraries

IMS catalog
ACBs are managed by IMS catalog (directory)

- DFSDF member

  DFSDF*xxx* member name in IMS PROCLIB
- IMS catalog PHIDAM database name
- Number of PHIDAM partitions
- Data set organization (PHIDAM partitions)

Space analysis: IMS catalog database space usage

The IMS catalog database space usage report displays the following information:

- IMS catalog PHIDAM database name
- PHIDAM partition name
- Data set group
- Data set name
- Allocated extents

  The number of allocated extents of the database data set.
- IMS size limit

  Maximum data set size that is limited by IMS.
- Allocated space (Bytes)

  Allocated space size of the database data set.
- Used space (Bytes)

  Used space size that is high used RBA (Relative Bytes Address) of the database data set. It is the place of end-of-file.
- IMS limit used (%)

  Ratio of used space to IMS space limit.
- Allocated space used (%)

  Ratio of used space to allocated space.

Space analysis: Program and database instances in IMS catalog database - Estimated sizes

The Program and database instances in IMS catalog database report displays the following information:

- Program (PSB) instances

  The number of PSB instances in the IMS catalog database.
- Database (DBD) instances

  The number of DBD instances in the IMS catalog database
- Total

  The number of PSB and DBD instances in the IMS catalog database.
- Estimated average size

  Estimated average size of PSB and DBD instances.

  This estimation does not take the extra time to read the IMS catalog database directly. Therefore, IMS segment data and free space information is not analyzed.

  As the result, the estimated size value can be larger than the average size value because the estimated average size value includes the IMS free space.

Space analysis: Program and database instances in IMS catalog database - Calculated sizes

The average sizes of DBD and PSB instances are calculated by directly reading the IMS catalog database. IMS free spaces are excluded from the average size values. Therefore, the average size values are more accurate than the estimated average size values.

- Number of PSB instances
- Calculated average size of PSB instances
- Number of DBD instances
- Calculated average size of DBD instances
- Total number of PSB and DBD instances
- Calculated average size of PSB and DBD instances

PSB summary report

The PSB summary report displays the following information:

- All PSB instances
  - Number of PSB instances
  - Average size of PSB instances
- Obsolete PSB instances
  - Number of obsolete PSB instances

    An obsolete instance is not used by IMS.

    For the details of obsolete instances, refer an explanation of status in PSB List.
  - Average size of obsolete PSB instances
- Number of PSB resources having multiple instances
  - Number of PSB resources
  - Number of PSB resources having multiple instances
  - Average number of instances per PSB resource
  - Highest number of instances within one PSB resource

Show full PSB list (detail report)

The Show full PSB list (detail) report displays the following information:

- PSB resource name
- Generation date and time
- Size of PSB instance in IMS catalog database
- Status - IMS catalog managed ACBs (application control blocks)

  Active The PSB instance is active.

  The time-stamp is equivalent to the active object in an IMS directory active data set.

  Staging
  The PSB instance is pending.

  The time-stamp is equivalent to the pending object in an IMS directory staging data set.

  "Blank"
  The PSB instance is obsolete and it is not used by IMS. The following conditions can apply:

- The instance has an old time-stamp.
- The instance has a newer time-stamp than Active, but it is not in Staging.
- If a PSB resource is not in an IMS directory active or staging data set, "blank" is set for every instance of the PSB resource.

- Status - ACBLIB managed ACBs (application control blocks)

  Active The PSB instance is active.

  When Online Change (OLC) is enabled, the time-stamp is equivalent to the PSB member in the active ACB libraries.

  When OLC is not enabled, the time-stamp is equivalent to the PSB member in the ACB libraries.

  Inactive
  The PSB instance is inactive.

  The time-stamp is equivalent to the PSB member in the inactive ACB libraries.

  Inactive is displayed only when OLC is enabled.

  Staging
  The PSB instance is in the staging ACB library.

  The time-stamp is equivalent to the PSB member in the staging ACB libraries.

  Staging is displayed only when OLC is enabled.

  "Blank"
  The PSB instance is obsolete and it is not used by IMS. The following conditions can apply:
  - The instance has an old time-stamp.
  - The instance has a newer time-stamp than Active, but it is not Inactive or Staging.
  - If a PSB resource is not in the ACB libraries, "blank" is set for every instance of the PSB resource.

DBD summary report

The DBD summary report displays the following information:

- All DBD instances
  - Number of DBD instances
  - Average size of DBD instances
- Obsolete DBD instances
  - Number of obsolete DBD instances

    An obsolete instance is not used by IMS.

    For the details of obsolete instances, refer an explanation of status in DBD List.
  - Average size of obsolete DBD instances
- Number of DBD resources having multiple instances
  - Number of DBD resources
  - Number of DBD resources having multiple instances
  - Average number of instances per DBD resource

  - Highest number of instances within one DBD resource
- DBD instances not pointed to by PSBs
  - Number of DBD instances not pointed to by PSBs
  - Average Size (Bytes)

Show full DBD list (detail report)

The Show full DBD list (detail) report displays the following information:

- DBD resource name
- Database (DB) version
- Generation date and time
- Size of DBD instance in IMS catalog database
- Status - IMS catalog managed ACBs (application control blocks)

  Active The DBD instance is active.

  The time-stamp is equivalent to the active object in an IMS directory active data set.

  Staging
  The DBD instance is pending.

  The time-stamp is equivalent to the pending object in an IMS directory staging data set.

  Usable
  The most recent time-stamp DBD instance within the old DB Version.

  The DBD instance can be used by IMS if the DB Version is specified by a PSB or an application program.

  (Logical)
  This is a logical DBD and the latest time-stamp instance.

  IMS does not store the logical DBD block in an IMS directory active or staging data set.

  For this reason, (Logical) is set instead of Active or Staging.

  "Blank"
  The DBD instance is obsolete and it is not used by IMS. The following conditions can apply:
  - The instance has an old time-stamp.
  - The instance has a newer time-stamp than Active, but it is not Staging.
  - If a DBD resource is not in an IMS directory active or staging data set, "blank" is set for every instance of the DBD resource (except for the logical DBD).
- Status - ACBLIB managed ACBs (application control blocks)

  Active The DBD instance is active.

  When Online Change (OLC) is enabled, the time-stamp is equivalent to the DBD member in the active ACB libraries.

  When OLC is not enabled, the time-stamp is equivalent to the DBD member in the ACB libraries.

  Inactive
  The DBD instance is inactive.

The time-stamp is equivalent to the DBD member in the inactive ACB libraries.

Inactive is displayed only when OLC is enabled.

Staging
The DBD instance is in the staging ACB library.

The time-stamp is equivalent to the DBD member in the staging ACB libraries.

Staging is displayed only when OLC is enabled.

Usable
The most recent time-stamp DBD instance within the old DB Version.

The DBD instance can be used by IMS if the DB Version is specified by a PSB or an application program.

(Logical)
This is a logical DBD.

IMS does not store the logical DBD block in IMS ACB libraries.

For this reason, (Logical) is set instead of Active, Inactive, or Staging.

(Logical) is set on the most recent time-stamp instance of the logical DBD.

(GSAM)
This is a GSAM DBD.

IMS does not store the GSAM DBD block in IMS ACB libraries.

For this reason, (GSAM) is set instead of Active, Inactive, or Staging.

(GSAM) is set on the most recent time-stamp instance of the GSAM DBD.

"Blank"
The DBD instance is obsolete and it is not used by IMS. The following conditions can apply:

- The instance has an old time-stamp.
- The instance has a newer time-stamp than Active, but it is not Inactive or Staging.
- If a DBD resource is not in the ACB libraries, "blank" is set for every instance of the DBD resource.

- Number of PSB Resources Referring this DBD

  The number of PSBs that reference this DBD.

  - For the PSB resources, only active instances are calculated.
  - For the DBD resources, active or usable DBD instances are calculated.

    For GSAM and Logical DBDs, instances flagged with (GSAM) or (Logical) are calculated.
  - Obsolete, inactive, and staging DBD or PSB instances are out of scope for this calculation.

    These instances are not used by IMS at this point.
  - When DB Versioning is enabled, the following IMS definitions are evaluated for this calculation:
    - DBLEVEL=BASE or CURRENT in the DFSDF*xxx* member of the IMS PROCLIB

- DBLEVEL=BASE or CURRENT in the PSB
- DBVER=*n* in the PSB

Note: The INIT VERSION call in an IMS application program is not evaluated.

Chapter 18. DBD/PSB compare

The compare function of IMS Administration Tool allows you to compare versions of DBD and PSB resources in the IMS directory data sets and the IMS ACB library.

Compare business scenarios

You can use the compare function to:

- Confirm consistency of resources in the IMS directory to resources in the ACB library. The IMS directory and the ACB library to compare can be for different IMS subsystems. For example, in a data sharing environment consisting of two IMS subsystems, you can compare the IMS directory for an IMS subsystem to the ACB library used by another IMS subsystem.
- Identify and review differences in resources between the IMS directory active data sets and the IMS directory staging data set.

Here are some common business scenarios:

- After migrating from ACBLIB-managed ACBs to IMS catalog-managed ACBs (IMS management of ACBs), use the compare function to verify that the IMS directory is successfully populated from the ACB library.
- When migrating from ACBLIB-managed ACBs to IMS catalog-managed ACBs in a data sharing environment where one IMS subsystem uses ACBLIB-managed ACBs and the other IMS subsystem uses IMS catalog-managed ACBs, the resources in the ACB library and the IMS directory must be in sync. Use the compare function to ensure that the consistency is maintained during migration.
- If IMS catalog-managed ACBs are used, use the compare function before activating changes to resources. The compare function reports differences between the resources in the IMS directory active data sets and the IMS directory staging data set and you can ensure that the changes that will be activated are what you intended.

DBD/PSB compare criteria selection reference

The compare function of IMS Administration Tool allows you to compare versions of DBD and PSB resources between the IMS catalog (directory) and the IMS ACB library.

For comparison selection, DBDs and PSBs are known as **objects**.

Objects can be further distinguished as resources and instances:

- **Resource** refers to a DBD object that is identified by a DBD name, or a PSB object that identified by a PSB name.
- **Instance** refers to a specific time/date occurrence of a resource.

  For example, a PSB resource can have multiple instances with different time-stamps.

Table 22. Compare criteria selection

Option	Description
Comparison Scope	Options for comparison: • Compare a single resource (Compare) • Compare multiple resources (Compare All) You can choose one more DBDs or PSBs to compare. Resources can be selected using filters.
Resource Type	Resource types include: • DBD • PSB You can choose one more DBDs or PSBs to compare. Resources can be selected using filters.
IMS directory resource criteria	
IMSID (of IMS Directory)	The IMSIDs in the selection list are catalog-managed ACBs. Catalog-managed ACBs means the IMS catalog is enabled, ACBs are managed with the IMS catalog, and resources are stored in the IMS directory.
Resource Name	You can choose one more DBDs or PSBs to compare. Select a single resource from the IMS directory for single resource comparison. Select multiple resources from the IMS directory for multiple resource comparisons.
Resource (Instance) Status (Data and Time Instance)	The selected DBD or PSB resource instance status can be: • Active Active instances are stored in the IMS directory active data sets. • Staging Pending instances are stored in the IMS directory staging data set.
ACB library resource criteria	

Table 22. Compare criteria selection (continued)

Option	Description
IMSID (of ACB Library)	The IMSIDs in the selection list satisfies the one of the following conditions: • IMS catalog is not enabled. • IMS catalog is enabled and ACBs are managed with ACB libraries. You can alternatively specify another ACBLIB library data set.
Resource Name	You can choose one more DBDs or PSBs to compare. Select a single resource from the ACB library for single resource comparison. Select multiple resources from the ACB library for multiple resource comparisons.
Resource (Instance) Status (Data and Time Instance)	When OLC is enabled, the selected DBD or PSB resource instance status can be: • Active Active instances are in the active ACB libraries. • Inactive Inactive instances are in the inactive ACB libraries. • Staging Staging instances are in the staging ACB libraries.

Table 22. Compare criteria selection (continued)

Option	Description
Comparison Options	Compare options to ignore certain comparison differences: **Ignore VERSION= in DBD** Ignore the differences of VERSION= *parameter* in the DBD statement. **Note:** VERSION= *parameter* is different from DBVER= *parameter*. DBVER is the version number of the database versioning and is always compared. **Ignore METADATA in DBD and PSB** Ignore the metadata differences in DBD and PSB. The metadata is as follows: **DBD** • DFSMARSH, DFSMAP, DFSCASE statements Includes the statements and any parameters on the statements. • FIELD statements CASENAME=, DATATYPE=, DEPENDSON=, EXTERNALNAME=, MINOCCURS=, MAXOCCURS=, MAXBYTES=, PARENT=, REDEFINES=, RELSTART=, REMARKS=, STARTAFTER= • Other statements ENCODING=, EXTERNALNAME=, REMARKS= **PSB** EXTERNALNAME=, REMARKS= **Ignore PCB Name** Ignore the differences for the NAME= *parameter* or the label in the PSBGEN statement of the PSB. **Ignore RMNAME= in DBD** Ignore the differences for the RMNAME= *parameter* in the DBD statement. **Ignore Segment/Edit Compression Exit Routine Name** Ignore the differences for the COMPRTN= *parameter* in the SEGM statement of the DBD. **Ignore KEYLEN of PCB** Ignore KEYLEN= in the PCB statement of the PSB **Ignore DEDB AREA Statement** Ignore AREA statements in the DBD and any parameters on the AREA statements.

DBD/PSB compare results reference

A compare report contains results from the comparison of two instances.

The source of DBD or PSB in the IMS directory is taken as the basis for the comparisons.

An initial comparison results report provides a summary analysis.

You can also access a detailed results report with side-by-side comparison.

Table 23. Compare results

Option	Description
Compare Results	The initial comparison results report indicates one of the following analysis categories: **Identical** The resource instances in the IMS directory and in the ACB library are identical. **Different** The resource instances in the IMS directory and the ACB library are different. **Unmatched** The resource instance exists in the IMS directory or the ACB library, but not both.
Comparison results detail	

Number of Different Statements	The top header section of the comparison report contains the summary information about statements which were inserted, deleted, or changed.
	INSERTED
	The number of statements which were found only in the DBDs or the PSBs in the ACB library.
	DELETED
	The number of statements which were found only in the DBDs or the PSBs in IMS directory.
	CHANGED
	The number of statements which were found in both the DBDs or the PSBs in the IMS directory and the ACB library, but were detected to be different.
	Example:
	`NUMBER OF DIFFERENT STATEMENTS`
	`INSERTED  :     44`
	`DELETED   :      8`
	`CHANGED   :     10`

Table 23. Compare results (continued)

Option	Description

IMS Environment and DBD or PSB Profile	The second header section of the comparison report contains the summary of the IMS environment and the compared instances of DBD or PSB resources:
	• IMSID
	• IMS directory high level qualifier and ACB library data set name
	• Status of the selected resource instance
	• DBD or PSB resource name
	• Time stamp when the resource instance was generated
	• IMS Version when the resource instance was generated
	
	Example:
	`IMSID : IFB8`
	`CATALOG HLQ : IMS.IFB8.DFSCD000`
	`STATUS : ACTIVE`
	`RESOURCE : DBFSAMD4`
	`GENERATED : 06/12/2018 19.41`
	`GENERATED IMS : 1510`

Line-by-line Comparison Result	The detail section of the comparison report shows a side-by-side and line-by-line display of the similarities and differences between the DBD or PSB sources. The following characters in the CHK column of the report indicate the type of difference found in the DBD or PSB source between the IMS directory and the ACB library: **I** A statement is inserted into the DBD or PSB in the ACB library. **D** A statement is deleted from the DBD or PSB in the IMS directory. **C** A statement in the DBD or PSB in the IMS directory is different from that in the ACB library. An asterisk (*) is shown on the row of each data that is determined to be different. The SOURCE LINES column shows the IMS DBDGEN or PSBGEN utility control statements that were decoded from the DBD or PSB instance in the IMS directory or the ACB library.

Chapter 19. Export objects and import objects

The export objects function, in combination with the import objects function, allows you to easily bulk copy DBD and PSB resource definitions from one IMS system to another IMS system, regardless of whether both systems are using the IMS catalog or not.

The export objects function extracts IMS ACB control blocks of DBDs and PSBs from ACB libraries or IMS directory, decodes the control blocks into readable DBD and PSB sources codes, and stores them in the *export data set*. An export data set is an intermediate data set generated by the export objects function and used by the import objects function.

The import objects function reads the export data set and calls the DBDGEN, PSBGEN, ACBGEN utilities, and, if IMS catalog is defined in the IMS system, the IMS catalog populate utility (DFS3PU00). If the Use COPYBOOK option is selected, the import objects function also performs copybook import before DBDGEN.

By using the export objects function and the import objects function, you can transfer DBDs and PSBs from one IMS subsystem to another IMS subsystem.

Export/import business scenarios

Because these two functions use DBD and PSB source codes as intermediate data, and DBDGEN, PSBGEN, and ACBGEN are done in the target system environment, the functions allow to maintain different IMS environments for the source system and the target system. For example, you can transfer DBDs and PSBs to a target system that uses a different release of IMS, or transfer DBDs and PSBs between two systems that manage ACBs differently (one with ACBLIB and another by IMS). You can also use these functions to manage objects in a single IMS system. For example, you can export and import objects when you migrate to a new release of IMS, or when you want to import copybooks to DBDs and PSBs and replace DBDs and PSBs in the IMS system.

Typical business scenarios can include:

- Build a test IMS subsystem.
- Synchronize two IMS environments.
- Create a mirror-image IMS subsystem from an existing IMS subsystem.
- Move the IMS subsystem to a different ACB management environment, for example from ACBLIB-managed to IMS-managed (ACBs managed with IMS directory).
- Restore the IMS subsystem from IMS managed ACBs (ACBs managed with IMS directory) to ACBLIB.
- Import COBOL or PL/I copybooks in bulk to the IMS catalog to accommodate a change of application programs and make the information available to Java applications.

Topics:

Export objects reference

The export objects function extracts IMS control blocks (DBDs and PSBs) from either the ACB library or IMS directory depending on how IMS is configured. Then it decodes the extracted control blocks to readable DBD or PSB source code enabling you to import DBDs and PSBs with the import objects function.

The ACB library and IMS directory are automatically determined by IMS Administration Tool from the parameters and the PROCLIB libraries of the IMS system.

The following options allow you to set up the process of exporting selected resource objects to export data sets. The export objects function generates a JCL job based on the options you select. You submit the JCL job to export objects to the export data set.

Table 24. Exporting objects

Option	Description
Object Selection Criteria:	
IMSID	The 1-4 character name of the IMS subsystem to export from.
Export Objects	Specification of resource types to export (and import): • DBD • PSB • Both (DBD and PSB)
DBD and PSB Filters	Specify a wildcard expression to control the number of resource objects that display.
Export Object Options:	
Export from and Object Status	Specify the location and the status of the objects to export from. • If the IMS management of ACBs is not enabled, select ACB library. • If the IMS management of ACBs is enabled, you can select from ACB library or IMS directory. To select ACB library, the IMSACB DD statement must be present in the IMS control region JCL or procedure. **ACB library** • Active Active ACB library. • Inactive Inactive ACB library. • Staging Staging ACB library. **IMS directory** • Active IMS directory data sets. • Staging IMS directory staging data set.
Prefix of Export Data Sets	The high-level qualifier prefix of the output data sets that are used for the export process. (35 character maximum)

Import objects reference

The import objects function calls the DBDGEN, PSBGEN, ACBGEN utilities, and, if IMS catalog is defined in the IMS system, the IMS catalog populate utility (DFS3PU00). If the Use COPYBOOK option is selected, the import objects function also performs copybook import before DBDGEN.

The import objects function requires an export data set as input. The export data set must contain DBD and PSB source codes generated by the export objects function.

The DBDLIB, PSBLIB, ACBLIB, IMS directory, and IMS catalog are automatically determined by IMS Administration Tool from the parameters and the PROCLIB libraries of the IMS system.

Requirement: When the IMS management of ACBs is enabled, the IMS system does not require ACB libraries. However, you must create an ACB library and either allocate it to the IMS control region with DD name IMSACB or create a DFSMDA member for the IMSACB.

The following options allow you to set up the process of importing selected resource objects from the export data set to a new target destination. The import objects function generates a JCL job, ATY@OBJU, based on the options you select. For more information about ATY@OBJU, see Chapter 15, "DBD and PSB update (ATY@OBJU) JCL," on page 111.

Table 25. Importing objects

Option	Description
Object Selection Criteria:	
IMSID	The 1-4 character name of the IMS subsystem to import to.
Import Objects	Specification of resource types to import from the export data set: • DBD • PSB • Both (DBD and PSB)
DBD and PSB Filters	Specify a wildcard expression to control the number of resource objects that display.
Import Object Options:	

Table 25. Importing objects (continued)

Option	Description
Initial Load	This option is available if the IMS catalog is defined to the IMS subsystem. Specify whether to initialize or update the IMS catalog and IMS directory. **Y** Initialize the IMS catalog and IMS directory. If an IMS catalog and IMS directory exist, the data sets are deleted and defined. The ACB members of DBDs and PSBs are stored in the IMS directory active data sets. **N** Update the existing IMS catalog and IMS directory. The DBDs and PSBs are stored in IMS directory active data sets and IMS catalog. The ACB members of DBDs and PSBs are stored in the IMS directory staging data set. **Requirement:** While the import objects job is running, the IMS system must be either active or inactive depending on the INITLOAD parameter value set in the generated JCL (ATY@OBJU JCL). For details, see "Requirement: Status of the online IMS system" on page 114. See also "Scenarios for "Initial Load" and "Overwrite Existing Objects"" on page 115.
Overwrite Existing Objects	Specify whether to overwrite existing DBD and PSB objects. **Y** Overwrite existing objects. **No** Do not overwrite existing objects. IMS Administration Tool checks if a member with the same name exists in the ACB library: • If the IMS management of ACBs is not enabled, IMS Administration Tool checks the staging ACB library. • If the IMS management of ACBs is enabled, IMS Administration Tool checks the data set that is pointed to from the IMSACB DD statement in the IMS control region JCL. For more information, see "Scenarios for "Initial Load" and "Overwrite Existing Objects"" on page 115.
Prefix of Export Data Sets	The high-level qualifier prefix of the export data set created by the export objects function.

Table 25. Importing objects (continued)

Option	Description
Backup Existing Objects	To provide rollback capability, backup existing objects in the backup data sets before importing. This option creates backup copies of the library or the data sets that the import objects function might update. The backup copies contain DBDs and PSBs in the form of source codes. **Yes** Back up existing objects. The import objects function decodes ACB members in the ACB library or IMS directory into DBD and PSB source codes and stores them in the backup data sets. • If the IMS management of ACBs is not enabled, creates a backup copy of the ACB staging library. • If the IMS management of ACBs is enabled, creates backup copies of the following data sets: – IMS directory active and staging data sets – The ACB library that is pointed to from the IMSACB DD in the IMS control region JCL. **No** Do not create backup data sets.
Prefix of Backup Data Sets	The high-level qualifier prefix of the backup data sets. (35 character maximum)
Use COPYBOOK	This option is available only for DBD objects. Specify Y to import copybook information to the DBD source code. If you specify Y, the function analyzes the copybook and inserts corresponding metadata statements into the DBD source for DBDGEN. **Requirement:** The COBOL or PL/I compiler library DDNAME variable must be registered. **Tip:** To change COBOL compiler options, specify the data set that contains the IGYCDOPT module to DDNAME variable CBLOPT. For details about variables, see "DDname and keyword variables for copybook import" on page 104.
COPYBOOK Data Sets:	
COPYBOOK Cross Reference (XREF) Data Sets	The name of the data set that pairs the DBD with the copybook. You can specify up to 10 data sets. If you are using the ISPF interface, specify Y to view, change, or add data set names. For the format of COPYBOOK XREF files and examples, see "Copybook XREF file" on page 105.

Table 25. Importing objects (continued)

Option	Description
COBOL or PL/I COPYBOOK Data Sets	The names of the data sets where the copybook resides. You can specify up to 120 data sets, maximum of 60 for COBOL copybook data sets and 60 for PL/I copybook data sets. If you are using the ISPF interface, specify Y to view, change, or add data set names. **Requirement:** The compiler library must be specified as a DDNAME variable. DDNAME CBLLIB is for the COBOL compiler library, and DDNAME PLILIB is for the PL/I compiler library. Specify either or both depending on the language of the copybook that you want to import. **Tip:** To change COBOL compiler options, specify the data set that contains the IGYCDOPT module to DDNAME variable CBLOPT. For information about changing DDNAME variables, see Chapter 14, "Copybook import," on page 101.
DBD Source with COPYBOOK	Specify the name of the output data set for storing the updated DBD source.
JCL Output Options:	
JCL Output Data Set	The name of the partitioned data set where the generated import JCL is stored. The data set must be pre-allocated before you can generate the JCL
Member	The name of the member in the partitioned data set where the generated import JCL is stored.
Job Statements	Specification of the JOB statement of the import JCL.
Allocate Data Set	Allocate the data set where the generated import JCL is stored.

Tip: If you want to change assembler options used for DBDGEN or PSBGEN, you can do so by describing the options in a sequential data set (PS) and registering the data set to DDNAME variable ASMAOPT.