AllFusion™ CA-Librarian®

Batch Command Reference Guide

4.3

F00011-2E

Contents

Chapter 1: Introduction

Chapter 2: Control Statements

Chapter 3: Control Commands

Chapter 4: Processing Options

Chapter 5: Compiling a Program (z/OS and OS/390)

Chapter 6: Compiling a Program (VSE/ESA)

Appendix A: Sample Listings

Index

Chapter 1

Introduction

This guide provides the information required to use the AllFusion CA-Librarian batch program.

AllFusion CA-Librarian is a highly sophisticated and flexible storage medium for source programs and other sets of data records, together with the collection of service routines for storing and retrieving them.

AllFusion CA-Librarian helps you keep track of your programs and data. You can store anything as a member on an AllFusion CA-Librarian master file, but AllFusion CA-Librarian is most useful for source programs. It offers many features for auditing and control, and facilities to compile and run these programs.

Master Files

The basic component of AllFusion CA-Librarian is its master file. A *master file* is a preallocated area of space on disk where data is stored in a highly compressed format.

Each master file consists of one or more modules. A module is a collection of data records and the associated control information, identified by a single name. A data record can be any 80-byte record.

Traditional Master Files

An AllFusion CA-Librarian disk master file is a direct-access data set consisting of fixed-length blocks. The block size is set when the master file is initialized. The file includes an index and a data area. Neither area is preallocated. The index expands as needed.

Modules are stored on the master file in a random manner. A free space map, located on the master file, keeps track of which blocks are in use and which are available.

AllFusion CA-Librarian can process an unlimited number of control statements in a disk master execution. Control statements are processed in the order of their appearance in the control deck. There are no work files required to process an AllFusion CA-Librarian disk master.

Wide Record Master Files

The Wide Record Master File has the same external appearance as both DA and VSAM master files except for two major differences. A Wide Record Master File can support data records up to 32,760 bytes, and the archive level limit of 255 is no longer in effect. The archive level limit can be up to 99,999,999 levels.

Data Compression

All data records on an AllFusion CA-Librarian master file are stored in a compressed format. This compression not only increases the capacity of the file, but also significantly shortens execution time by reducing the amount of I/O. The compression level of a master file is established at initialization time as either none, partial, or full compression. The length of a compressed record varies greatly according to its content. The following figure shows the full compression of a few typical COBOL source statements.

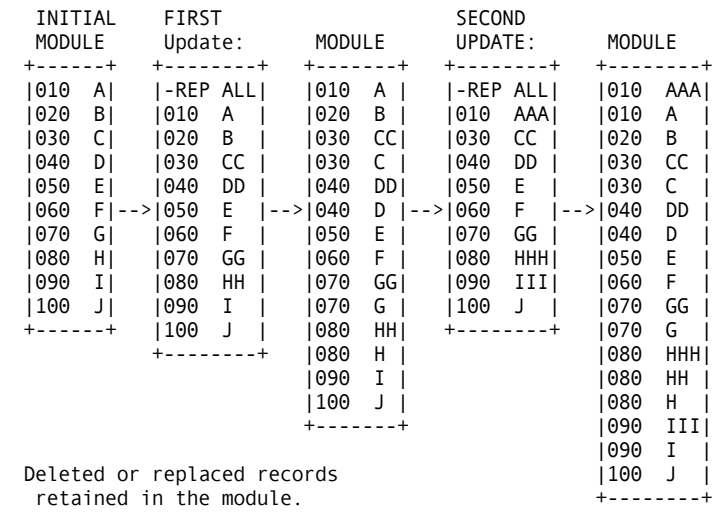
A *logical record* for an AllFusion CA-Librarian master file can contain several compressed data records. AllFusion CA-Librarian thus makes maximum use of space to store the greatest amount of data.

COBOL Statement (80 columns)	Compressed Length (in bytes)
02 EMPL-NUM PICTURE 9(5)	15
MOVE INPUT-ACCOUNT TO MAST-ACCOUNT	25
ADD TWO TO LINE-COUNT	15
PERFORM TAX-COMPUTE	12

Archiving

The Archiving Facility lets you recreate previous versions (or levels) of an updated module quickly and easily. When archiving is activated for a module, AllFusion CA-Librarian retains in the module all records that would otherwise be deleted during an update operation.

As updates progress, the records are marked to indicate the level or levels where each record belongs. If you request an earlier level of a module, archiving reconstructs the level using the records marked as belonging to the specified level. The following figure shows the development of multiple levels in an archived module.



You can retain as many as 255 archiving levels for a single module. The current level number and the total number of levels being maintained for a module are reported on the Management Index listing. Wide Record Master Files have no archiving level limit.

You can access an earlier level of a module by specifying its absolute level number, its relative level number, or the date (or date and time) the level was current.

You must initialize a master file to support archiving, but an archived master file can contain both archived and non-archived modules.

You can examine the differences between two levels of an archived module residing on a disk master file through the COMPARE feature.

Note: Archiving affects other AllFusion CA-Librarian options and facilities. Restrictions are noted in this guide where appropriate.

Chapter 2

Control Statements

Control statements activate AllFusion CA-Librarian operations. An AllFusion CA-Librarian control statement begins with a command. It can refer to a specific module and can include processing options and user comments.

Control Statement Format

The AllFusion CA-Librarian control statement consists of a command beginning in column 1, one or more blanks, any parameters, one or more blanks, and any user comments.

```
command [ module-name][,option(s)] [comments]
```

- Each AllFusion CA-Librarian command begins with a dash (-).
- Each AllFusion CA-Librarian command is followed by one or more spaces.
- Parameters are separated from each other by commas, with no intervening spaces.
- The module name, when required, must be the first parameter.
- The password, when specified, must be the second parameter.
- If you specify any processing options, they must follow the module name and password.
- The last option specified overrides any previous conflicting options.
- User comments must begin before column 70.
- You cannot continue AllFusion CA-Librarian control statements.

Here is a typical AllFusion CA-Librarian control statement:

```
-SEL MODULE1,ZYQR,LIST AN EXAMPLE
```

- -SEL is the command. It begins in column 1, with a dash.
- MODULE1 is the module name. It can be up to eight characters in length.
- ZYQR is the password.
- LIST is a processing option.
- AN EXAMPLE is a user-supplied comment.

Control Stream

A *control stream* is a series of AllFusion CA-Librarian control statements and data records indicating the processing to perform during a single execution of AllFusion CA-Librarian.

There are seven basic kinds of AllFusion CA-Librarian control statements:

- Execution control statements govern the general execution of AllFusion CA-Librarian and can carry options that hold true for the entire control stream.
- The module control statements process a particular module and can carry options that are applied to that module.
- Editing control statements can examine and change strings in existing records in a module.
- Updating control statements can add, replace, or remove entire records in a module.
- Documentary control statements can add special records to a module and other data records that record control information about that module. This documentary data is stored separately and can be retrieved separately from data records.
- Miscellaneous control statements can incorporate modules, other records, and other kinds of data into a module, or generate JCL.
- Utility control statements can transfer data records or whole modules in a master file or search an entire master file for character strings.

The following lists control commands with brief descriptions of their functions. See the "Control Commands" chapter for full descriptions of the command. Processing of a module begins with a module control statement and ends with either the next module control statement, an -EMOD, -END, or -DATA control statement, or the physical end of file.

Execution Control
(-OPT) Establish execution parameters

Module Control
(-ADD) Add a module to the master file

Editing Control
(-SCAN) Search module for character string

Updating Control
(-INS) Insert a record

Documentary Control
(-PGMR) Name the programmer

Miscellaneous Control
(-AUX) Add records from another file

Utility Control
(-INC) Transfer records or modules from the master

Sequence of Control Statements

This is the order in which to place your control statements in a control stream:

1. The -OPT control statement, if you use it.
2. Then, for each module you are working with:
   a. The -ADD, -SEL, or -DLM control statement (module control statement) for that module.
   b. Any -JCL control statements, after the -ADD or -SEL control statement but before any documentary control statements.
   c. Any -JCLT control statements. They must come after -JCL control statements when you are using both for one module during a single execution.
   d. -PGMR, -LANG, -DESC, -HST, and -HSTD control statements (documentary control statements) immediately following -JCL and -JCLT control statements. Do not place -PGMR, -LANG, and -DESC control statements after -HST statements.
   e. -COM control statements, anywhere in the control stream except after -EMOD or -DATA control statements.
   f. -SCAN, -EDIT, and -FILL (editing control statements) after documentary control statements. If you mix them together with updating control statements (-INS, -REP and -DEL), you must arrange both editing and updating statements so that the data records they refer to are processed in ascending order by sequence number.
   g. -INS, -REP, and -DEL (updating control statements), arranged so that the data records they refer to are processed in ascending order by sequence number.
   h. Data records immediately following their associated updating control statements.
   i. -INC and -AUX control statements where you want to copy the data records they represent.
   j. -EMOD control statement at the end of the control statements for each module to terminate processing for that module.
   k. -DATA control statement to terminate processing of a module and to include test data or supplementary job control statements in the compilation file.

3. After that come your control statements for the next module to process, if any. Use as many module control statements (-ADD, -SEL, or -DLM) as you need to perform the operations. Modules are processed in the order in which you refer to them in the control stream. Then AllFusion CA-Librarian processes the modules in ascending order by module name (their sequence on the master file), regardless of their order in the control stream.
4. END control statement at the end of the entire AllFusion CA-Librarian control stream. If you omit the -END control statement, AllFusion CA-Librarian writes a warning message on the Update Record listing. If this warning appears, examine your output for possible misapplied updates.

Sequence of Module Record Update

The order in which you update the records of your module is important.

As AllFusion CA-Librarian goes through your module applying update control statements (-INS, -REP, and -DEL), it reads the records sequentially that it is updating. If you replace (-REP) record number 30 and then try to delete record 20, AllFusion CA-Librarian gives you an error message because record 20 was already read and passed.

Editing control statements (-SCAN, -EDIT, and -FILL) are different in this respect. Unlike updating control statements (-INS, -REP, and -DEL), which must be arranged so that the records they refer to are in ascending alphanumeric order, you can specify editing control statements in any order. But when you are using editing control statements in a control stream with updating control statements, their placement can be critical.

AllFusion CA-Librarian notes the location of edit operations, but does not perform them until the records they refer to are read, often in response to an updating control statement. For this reason, you can put editing control statements before updates to previous records.

Because AllFusion CA-Librarian reads all the records in a module up to the specified record immediately upon detecting any updating control statement, it is not possible to edit previous records once a particular record is updated. You must put the editing control statements first or put them in sequence by record number with the updating control statements. You cannot put them last.

An editing control statement that was read but not acted upon is referred to as an active editing control statement. An editing control statement remains active until all the records it refers to are processed. No more than 30 editing control statements can be active simultaneously.

You can use a single editing control statement to process one record, a range of records, or every record in the module. If you include an editing control statement without identifying a particular record or a range of records, the records that the statement processes are those that AllFusion CA-Librarian reads after encountering the editing control statement. If you place the editing control statement at the beginning, before any updating statement, the editing function is performed on every record in the module. If placed after any updating control statements, the edit operation begins with the record following the one updated by the last processed update control statement and continues to the end of the module.

The results of editing operations are written on the Update Record listing. Each record modified by either -EDIT or -FILL is date stamped on the Module listing.

See the following two figures for an annotated example of the combined use of editing and updating control statements.

```
                                    Number of        Last
                                  Currently Active   Record
                                  Editing Commands Processed
-SEL MIXED,ZXJP
-EDIT   /ALPHA/BETA/ }                   1
-EDIT   /PAY3/PAY4/  } A                 2
-FILL   /NEWPAY/     }                   3
-SCAN   /WTD-/       }                   4
-EDIT   /PIK/PIC/70,300  B               5
-INS  FIRST
        SELECT PAY ASSGN TO SYSUT1.
-REP  80                                                  10
        BLOCK CONTAINS 50 RECORDS
-EDIT   /DAY/PAY/97                      6                80
-EDIT   /WITH/FWTD/          } C         7                80
-EDIT   /MASTER/PAY-MASTER/  }           8                80
-EDIT   /MVE/MOVE/90,110                 9                80
-EDIT   /SAVE//                         10                80
-REP  220
        ADD WK-FWTD TO YTD-FWTD.
-EDIT   /STAY/STAX/300,360               9               220
-INS  400
        ADD WK-FICA TO YTD-FICA.
-EDIT   /MVE/MOVE/450                    8               400
-EDIT   /MVE/MOVE/390  D
-EDIT   /STAY/STAX/460,490               9               400
-EDIT   /STX/STAX/460,490               10               400
-EDIT   /STX/STAX/620,630
-INS  500
        ADD WK-PAY TO YDT-PAY.
-REP  610                                7               500
       .
       .
```

Note the following items:

- A—These operations are performed on the entire module.
- B—This operation is limited to records 70 through 300.
- C—These operations are performed on all records of the module, beginning with record 90.
- D—This statement is rejected since AllFusion CA-Librarian already processed record 390.

```
RUN NO. 25     DATE=mm/dd/yy  TIME=1128 UPDATE RECORD LIBR.DEV.MAST

-SEL MIXED,ZXJP
-EDIT   /ALPHA/BETA/
-EDIT   /PAY3/PAY4/
-FILL   /NEWPAY/
-SCAN   /WTD-/
-EDIT   /PIK/PIC/70,300
-INS  FIRST
    000010  SELECT PAY ASSGN TO SYSUT1.

                              THE NEW CARDS HAVE BEEN INSERTED BEFORE THE FOLLOWING CARDS

                              000010       ACCESS RANDOM

    000020       ACCESS RANDOM                                          NEW FILL PERFORMED
    000030       BLOCK CONTAINS 50 RECORDS                              NEW FILL PERFORMED
    000040       NOMINAL KEY IS PSF-KEY                                 NEW FILL PERFORMED
    000050       RECORD KEY IS PSF-REC-KEY                              NEW FILL PERFORMED
    000060       SELECT PDSQISAM ASSIGN TO DA-I-ISAMDSQ                 NEW FILL PERFORMED
    000070       ACCESS RANDOM                                          NEW FILL PERFORMED
    000080       NOMINAL KEY IS PDSQ-KEY                                NEW FILL PERFORMED
-REP 80
    000090 BLOCK CONTAINS 50 RECORDS

                              FOLLOWING CARDS REMOVED FROM MASTER BY LAST OPERATION

                              000080       NOMINAL KEY IS PDSQ-KEY.

-EDIT   /DAY/PAY/97
-EDIT   /WITH/FWTD/
-EDIT   /MASTER/PAY-MASTER/
-EDIT   /MVE/MOVE/90,110
-EDIT   /SAVE//
    000100       SELECT PAYLOG ASSIGN TO DA-S-XMITQLOG.                 NEW EDIT PERFORMED
    000110       SELECT READER ASSIGN TO DA-S-CARDIN.                   NEW EDIT PERFORMED
    000120       SELECT SCMV ASSIGN TO DA-S-SCMV.                       NEW FILL PERFORMED
    000130       SELECT PAY-MASTER ASSIGN TO UT-S-ERRORS.               NEW EDIT PERFORMED
    000140       SELECT DELPRINT ASSIGN to UT-S-DELETE.                 NEW FILL PERFORMED
    000150 IO-CONTROL                                                   NEW FILL PERFORMED
    000160       APPLY CORE-INDEX ON PSFI PDSQI.                        NEW EDIT PERFORMED
    000170       DATA DIVISION                                          NEW FILL PERFORMED
    000180       FILE SECTION                                           NEW FILL PERFORMED
    000190       FD PAY-MASTER                                          NEW EDIT PERFORMED
    000200       BLOCK CONTAINS 0 RECORDS                               NEW FILL PERFORMED
    000210       RECORD CONTAINS 80 CHARACTERS                          NEW FILL PERFORMED
    000220       RECORDING MODE IS F                                    NEW FILL PERFORMED
-REP 220
    000230  ADD MK-FWTD TO YTD-FWTD.
```

```
                            FOLLOWING CARDS REMOVED FROM MASTER BY LAST OPERATION

                            000220       LABEL RECORDS ARE STANDARD

    000240     DATA RECORD IS PRODUCTS-CONTROL-CARD.                        NEW FILL PERFORMED
    000250 01  PRODUCTS-CONTROL-CARD.                                       NEW FILL PERFORMED
    000260     05 C-C-CTL-CHAR               PIC XXX.                      NEW EDIT PERFORMED
    000270        88 CTL                         VALUE '< >'.              NEW FILL PERFORMED
    000280     05 C-C-C-CODE                  PIC X.                        NEW EDIT PERFORMED
    000290        88 BETA                         VALUE '3'.                NEW EDIT PERFORMED
    000300        88 PROC                         VALUE 'P'.                NEW EDIT PERFORMED
    000310        88 CMNT-STAX                    VALUE '+' THRU 'P'.       NEW EDIT PERFORMED
    000320        88 CHNG                         VALUE 'C'.                NEW FILL PERFORMED
    000330        88 PAY4                         VALUE 'D'.                NEW EDIT PERFORMED
    000340        88 MSG-BETA                     VALUE 'M'.                NEW EDIT PERFORMED
    000350        88 DBD-STAX                     VALUE 'B'.                NEW EDIT PERFORMED
    000360        88 PSB-BETA                     VALUE 'S'.                NEW EDIT PERFORMED
    000370        88 PAT-STAX                     VALUE '2'.                NEW EDIT PERFORMED
    000380 WORKING-STORAGE SECTION.                                          NEW FILL PERFORMED
    000390 PROCEDURE DIVISION USING PARM-1.                                  NEW FILL PERFORMED
    000400 MVPG                                                              NEW FILL PERFORMED
    000410     OPEN INPUT READER OUTPUT DELPRINT ERRPRINT XMITQLOG           NEW FILL PERFORMED
-INS 400
    000420 ADD WK-FICA TO YTD-FICA.


THE NEW CARDS HAVE BEEN INSERTED AFTER THE FOLLOWING CARD
                            000400       OPEN INPUT READER OUTPUT DELPRINT ERROR
-EDIT   /MVE/MOVE/450
-EDIT   /MVE/MOVE/390
 ... ERROR ... UPDATES NOT IN ORDER BY SEQ NO.
-EDIT   /STAY/STAX/460,490
-EDIT   /STX/STAX/460,490
-EDIT   /STX/STAX/620,630
```

Protecting Against Duplicate Updates

AllFusion CA-Librarian automatically protects against consecutive duplicate updates, a condition that occurs when you inadvertently resubmit an update that was already successfully applied. For each update made to a module, AllFusion CA-Librarian computes a hash count, which it stores and then uses to determine if the update was submitted two consecutive times. An update that AllFusion CA-Librarian determines is a duplicate is not applied to the module and an error message is written on the Update Record Listing.

The hash count check is automatically bypassed when you are applying changes on a temporary basis (through the TEMP or TEMPS option), since no permanent updates are made to the module. If you want to bypass the hash check and apply duplicate updates, place an X in column 7 of the -EMOD or -DATA control statement for the module. For additional information on the -EMOD or -DATA control statement, see the “Control Commands” chapter.

Processing Hierarchy

There are three levels where processing options for modules are specified. In general,

each level overrides the specification of the previous (or higher) level.

1. Initialization level—Defaults for processing options are established when the master file is initialized. You can change these defaults by reinitializing the master file. Some of these defaults can be overridden for a single execution; others cannot. Check with your system programmer for the initialization defaults of the master file you are working with.
2. Execution level—For those initialization defaults that can be overridden, you can establish your own defaults by specifying processing options on a -OPT control statement. Defaults established at the execution level are in effect for the processing of each module during the current AllFusion CA-Librarian execution.
3. The Module Control Statement level—You can override many of the processing defaults established at the execution level by specifying an alternate processing option on a module control statement. Options selected at the module control statement level are in effect for the processing of one module only.

Adding JCL to a Module

A slash (/) appearing in column 1 of any data record identifies that record as JCL. AllFusion CA-Librarian automatically positions the sequence numbers of all such records in columns 73-80, unless the sequence number is defined as starting to the right of column 72.

Conversion Characters

For all except the POWER conversion character (/ $$), the conversion is done before adding the data to the master. The POWER Conversion character is stored as is on the master file and the conversion is done when the module is written to the JOBSTR file.

When in-stream JCL is stored online, you are expected to alter the JCL next. This avoids ambiguities between the JCL you are using to execute AllFusion CA-Librarian and the JCL you want to store in the AllFusion CA-Librarian master file. Do **not** alter JCL you enter through the -AUX control statement. Such auxiliary file entry cannot be confused with your in-stream JCL.

Convert the first (two) characters used in your in-stream JCL to AllFusion CA-Librarian *in-stream* characters using the table that follows.

Character Beginning in Column 1	Converted To	Purpose
/: /; /%	/* /& //	To add JCL as data.
/ $$	* $$	To add POWER control statements as data.
=	-	To add AllFusion CA-Librarian control statements as data.
%		To force AllFusion CA-Librarian sequence numbers to columns 73 through 80. A % in column 80 produces the same result. Specifying the COL1% parameter on a -OPT statement inhibits conversion of the % in column 1.

Notes:

- The JCL conversion characters (/:, / , and /%) are **not** required when adding or updating data online through CA-Roscoe EXPORT, ELIPS, CA-Vollie, or LIB/CMS.
- When adding or updating data through the -AUX control statement, data is added **as is** (no conversion is performed). See the "Control Commands" chapter for details on the -AUX control statement.

Forcing Sequence Numbers to Columns 73-80

You can direct AllFusion CA-Librarian to place the sequence number of a specific data record in columns 73-80 by placing a percent sign (%) in either column 1 or column 80 of that record. AllFusion CA-Librarian automatically positions the sequence numbers of the record in columns 73-80 unless the sequence number is defined as starting to the right of column 72.

Default Sequence Numbers

The default for sequence numbers in AllFusion CA-Librarian is columns 73-80 of the record, unless the sequence numbers is defined as starting to the right of column 72, as stated above.

For wide record support, the default is in columns 0,06,10,10 of the record. The zero (0) means that the numbers are external, and not part of the data.

When a module is written to the compilation file (through the EXEC option), AllFusion CA-Librarian replaces each percent sign appearing in column 1 of a data record with a blank. You can override this replacement by using the COL1% option.

Chapter

3

Control Commands

Each AllFusion CA-Librarian command must begin in column one, with a dash (-), and be followed by one or more spaces. The AllFusion CA-Librarian commands use many of the same processing options. Instead of listing and defining the same processing options for each command, this chapter indicates which processing options each command supports and the next chapter, "Processing Options," describes the processing options.

-ADD

Function

The -ADD update control statement adds a module to the master file. Processing options that you include on the -ADD control statement override execution defaults. If the adding of the module fails for any reason, any additional processing requested through the options is not performed.

Syntax

`-ADD` *`module-name`*`[,`*`options`*`]`

module-name

A unique 1- to 8-character name that you assign to the module. The module name can include any character except a comma, blank, or slash.

options

Valid options are:

- ARC
- ARCINC=
- CLEARID
- CCOPY*
- CON
- COPYDD*
- LOCK
- NOCHK
- NOEXEC
- NOLIST
- NOPUNCH
- NORESEQ
- PUNCH
- RESEQ
- SEQ=
- SEQCHK=
- SYNCHK(l,a)*
- UNCON

- EXEC
- LIST
- LISTH
- NOSORT*
- NOVAR
- PSWD=
- UNLOCK
- VAR

For Wide Record Master Files, use the IBM PDS naming convention. The PDS naming convention is as follows:

- The first character must be a national or alpha character ($#@, A-Z)
- The remaining characters must be national or alphanumeric characters ($#@, A-Z, 0-9)

Note: The options with an * are not valid for Wide Record Master files.

-AUX

Note: When running against wide records, the combination of RECFM=V and SUBSYS=LAM can cause unpredictable results. Use fixed or fixed block when using SUBSYS=LAM.

Function

The -AUX miscellaneous control statement adds the records of another file to an AllFusion CA-Librarian master file. Place one or more -AUX control statements in the control stream where you would otherwise place the data records the -AUX control statements represent. For example, you could specify a -AUX control statement immediately following a -INS control statement. Any AllFusion CA-Librarian commands read from the auxiliary input data set are not processed, but are written to the module as data records.

-AUX is not supported in an online environment, except when updating a wide record master file.

The format of a -AUX control statement is different for the OS/390, z/OS and OS/390, and VSE/ESA operating systems.

z/OS and OS/390 Syntax

```
-AUX ddname(memname)
```

ddname

The name of the DD statement defining the file to use as auxiliary input. Any number of -AUX statements can refer to the same DD statement. When using a partitioned data set as input, do not specify a member name in the DD statement, but on the -AUX statement. When using a sequential data set on an unlabeled tape, you must specify RECFM and BLKSIZE as DCB subparameters on the DD statement.

memname

The name of the member of a partitioned data set to use as auxiliary input.

```
//AUX JOB 7__1_,CA.WICK,CLASS=1
//EXEC PGM=ca-librarian
//D1 DD DSN=PAYROLL.SOURCE,DISP=SHR
//D2 DD UNIT=24__,DISP=OLD,LABEL=(,NL),VOL=SER=X123,
// DCB=(DEN=2,RECFM=FB,BLKSIZE=8__)
//MASTER DD DSN=libr.master.file,DISP=SHR
//SYSIN DD P
  -OPT NOSEP,LIST
  -ADD PAYEDIT,CLEARID
  -DESC PAYROLL INPUT EDIT
  -AUX D1(PAYEDIT)
  -EMOD
  -ADD PAYCOM,CLEARID
  -DESC PAYROLL COMPUTE ROUTINE
  -AUX D1(PAYCOMPT)
  -EMOD
  -SEL REPO
  -HST REVISED MODULE FROM HEADQUARTERS
  -REP ALL
  -AUX D2
  -END
/P
```

Tip: Using the -AUX statement increases execution storage requirements by an amount equal to twice the block size of that file having the largest block size of all files referred to by -AUX statements. If you are short of storage, specify BUFNO=1 as a DCB subparameter on each DD statement referred to by a -AUX statement.

For example, assume that you want to add to master file modules PAYEDIT and PAYCOM (currently stored as members PAYEDIT and PAYCOMPT in data set PAYROLL.SOURCE) and replace module REPO (currently stored on unlabeled tape X123). Submit the job shown above.

Important! *You cannot use the -AUX statement between any two master files without using SUBSYS=LAM.*

VSE/ESA SYNTAX

```
-AUX [SYSSLB(s.name)    ]
     [SYSnnn[,mm      ]]
     [      [,filename]]
```

SYSSLB
The auxiliary file is a system or private source statement library. When using a private source statement library, you must supply the appropriate ASSGN, DLBL, and EXTENT information. If you use LIBDEF in place of ASSGN, only the SEARCH= keyword parameter is recognized. For VSE/ESA format libraries, use LIBDEF with the SOURCE,SEARCH= parameter.

s
The library prefix.

name
The book name.

SYS*nnn*
The symbolic unit name of the file to use as auxiliary input. This input file must contain 80- or 81-byte unblocked records.

When specifying the file name value as the second parameter, you must supply either the appropriate DLBL and EXTENT job control statement (if you are using a disk file) or the appropriate TLBL job control statement (if you are using a standard label tape file). Whether you are using a disk file or a tape file, you must supply an ASSGN job control statement for the file.

mm
The file number identifying the input file on a multi-file tape. The default is 01, which corresponds to the file immediately following the first tape mark.

If you are processing an unlabeled tape without a leading tape mark, specify a numeric value one less than the file desired. For example, to access the first file, specify 0, to access the second file, specify 1, and so on.

When specifying a tape file as auxiliary input, the operator is responsible for mounting the correct tapes on the proper drives before starting the job. CA-Librarian does not issue any mounting messages. CA-Librarian positions each tape to the file identified by *mm* when the corresponding -AUX control statement is processed. Tapes can be labeled or unlabeled; no label information is necessary.

filename
The file name you supply on the DLBL or TLBL statement for the file.

If you omit this parameter and the file resides on a disk device, CA-Librarian uses AUXFILE as the file name. However, if you omit this parameter and the file resides on a tape device, CA-Librarian processes the first or only file on an unlabeled tape.

Using the -AUX control statements and the associated job control statements is illustrated in the following figure.

```
A     // ASSGN SYS_14,cuu
      // DLBL AUXFLE1,'disk.auxfile'
B     // EXTENT SYS_15,volser,1,_,reltrk,ntrks
      // ASSGN SYS_15,DISK,VOL=volser,SHR
      // TLBL AUXFLE2,'tape.file'
C     // ASSGN SYS_16,cuu
      // EXEC CA-Librarian,SIZE=AUTO
         -OPT NOLIST
         -ADD PROG5,SEQ=COBOL
         -AUX SYSSLB(C.PAYDF)
         -SEL TRAINING,PUNCH
         -DESC UPDATE REFLECTS NEW REQUIREMENTS
         -INS 7_
         -AUX SYS_14,_5
         -AUX SYS_15,AUXFLE1
         -AUX SYS_16,AUXFLE2
         -END
       /*
```

- A—This ASSGN statement assigns a file to an unlabeled tape.
- B—These statements define a disk file.
- C—These statements define a standard label tape file.

-COM

Function

The -COM update control statement displays comments on the Update record listing. Use as many as you need. You can place a -COM control statement anywhere in the control stream (except immediately following a -EMOD or -DATA control statement). -COM control statements are written on the Update Record listing for the current execution and then dropped.

Syntax

```
-COM narrative
```

narrative

The 1- to 75-character narrative, beginning in column 6. Blanks are valid characters.

-COPY

Function

The -COPY Utility control statement can transfer modules from one master file to another and consolidate modules from several different master files into a single master file. For detailed instructions on how to use the -COPY control statement, see the *Systems Services Guide,* for z/OS and OS/390 or VSE/ESA.

Note: Must follow an -OPT UTILITY control statement.

For wide record master files, you can use -COPY to copy all the archive levels of a member in a PDS or PDS/E master file to a member in a different PDS or PDS/E master file. This is accomplished in a single job step. The destination master file is allocated as DD COPYDEST. A comment is written to the job stream file instead of the special -ADD statement and member data records or LIBRCOPY for the current level.

Syntax

```
-COPY modname[,new-modname][,options]
```

modname
The member name of the member to transfer.

new mod-name
A new member name of the transferred member.

options
The options are LOCK/UNLOCK, PSWD=password, and REP|NOREP.

-DATA

Function

The -DATA control statement includes test data and supplementary job control statements at the end of a module on the compilation file. These statements are passed directly from the control deck to the compilation file. They are not added to the master file.

Use the -DATA control statement during an -ADD or -SEL operation. -DATA is not supported in an online environment. The -DATA control statement must follow all other input associated with the -ADD or -SEL operation but precede the -EMOD control statement, if any. The records to include on the compilation file must immediately follow the -DATA control statement.

Syntax

```
-DATA [X]
```

X

In column 7, AllFusion CA-Librarian bypasses the hash count check for the module being updated. This lets you apply a duplicate set of updates to the module.

For z/OS and OS/390

Whether you are supplying JCL statements for a module or AllFusion CA-Librarian is, you can use the -DATA control statement to write records to the compilation file. The records following the -DATA statement are written to the compilation file after any trailing JCL CA-Librarian generates. If the module is not written to the compilation file because of errors, the -DATA control statement and the data records following it are ignored.

As an example of using the -DATA control statement, consider the module PAYEDIT in the following JCL. AllFusion CA-Librarian generates JCL statements for compiling, link editing, and testing this module when it is written to the job stream.

```
//USEREF   JOB 70010,CA.JAEGER,CLASS=X                    
// EXEC PGM=CA-Librarian                                  
//OSJOB DD SYSOUT=(A,INTRDR),DCB=(LRECL=80,               
//         BLKSIZE=80)                                   
//SYSIN DD DATA,DLM=@@                                    
-OPT NOSEP                                          }    
-SEL PAYEDIT,JMXL,EXEC                              }    
-COM EMPLOYEE LIST UPDATE                           }    
-INS 30                                             } CA-Librarian
       .                                            } Control
       .  (module data records)                     } Stream
       .                                            }    
-DATA                                               }    
// EXEC PGM=PAYEDIT                                 }    
//STEPLIB DD DSN=PAYROLL.TEST,DISP=SHR              }    
//SYSIN DD DSN=PAYROLL.DATA,DISP=OLD                }    
-END                                                }    
@@                                                  }    

//PAYEDIT JOB (70010),                          C   }    
//            GIBSON,                           C   }    
//            MSGLEVEL=1                            }    
//UPDATEXX EXEC COBFCL                      JMXL    }    
//COB.SYSIN DD P                                    }    
       .                                            }    
       .   (module data records)                    } Resulting
       .                                            } Job
/*                                                  } Stream
```

```
//LKED.SYSLMOD DD DISP=OLD,                C   }
//            DSNAME=PAYROLL.TEST(PAYEDIT)     }
// EXEC PGM=PAYEDIT                            }
//STEPLIB DD DSN=PAYROLL.TEST,DISP=SHR         }
//SYSIN DD DSN=PAYROLL.DATA,DISP=OLD           }
```

Note the following:

- When you supply JCL through the -DATA control statement, the AllFusion CA-Librarian control stream (if in the input stream) must be defined by a //SYSIN DD DATA JCL statement rather than a //SYSIN DD * statement.
- To bypass the hash count check for a module, place an X in column 7 of the -DATA control statement.

For VSE/ESA

You can use the -DATA control statement whether you supply job control statements for a module or AllFusion CA-Librarian generates them. The records following the -DATA control statement are written to the compilation file following any trailing job control statements AllFusion CA-Librarian generates, but preceding any end-of-data-file statement (/*) and end-of-job statement (/&) AllFusion CA-Librarian generates.

If you specify TYPE=COMPGO or TYPE=COMPCATGO, any job control statements immediately following the -DATA control statement are placed ahead of the final EXEC statement AllFusion CA-Librarian generates.

For purposes of this processing, any record containing an asterisk (*) in column 1, slashes (//) in columns 1 and 2, or a percent sign (%) in column 80 is treated as a job control statement.

If the module is not written to the compilation file because of errors, the -DATA control statement and the data records following it are ignored.

-DEL

Function

The -DEL updating control statement deletes records from a module.

Syntax

```
-DEL seq1{,seq2 }
        {,LAST }
```

***seq*1**
The sequence number of the first or only record to delete.

Note: If specifying multiple -DEL control statements or -DEL control statements with -REP or -INS control statements, you must arrange the statements in the control stream so that the sequence numbers of the records to which they refer are in ascending numeric order.

***seq*2**
The sequence number of the last record of the range of records to delete. To delete a single record, omit seq2.

LAST
The end of the range of records to delete as the last record of the module.

-DESC

Function

Use the -DESC documentary control statement to describe the module. You can add or change the description at any time. All information supplied on a -DESC statement is permanently stored on the master file, until overlaid by another -DESC statement, and written on the module listing. Do not put a -DESC statement after a -HST statement in the control stream.

Syntax

```
-DESC description
```

description
A 1- to 30-character module description, beginning in column 7. Blanks are valid characters.

-DLM

Function

The -DLM module control statement deletes a module and all information associated with it from the master file.

Syntax

```
-DLM module-name[,password][,options]
```

module-name
The name of the module to delete.

password
The module's password. The password might be required. This is determined by site management at master file initialization or reinitialization time.

options
Valid options are:

- CON
- UNCON
- VERS.

-EDIT

Function

The -EDIT editing control statement directs AllFusion CA-Librarian to examine a group of records for the occurrence of a specified string of characters and, whenever it is found, to replace it with a different string. Every occurrence of the search string is replaced.

Syntax

```
-EDIT *string-1*[string-2]*[seq1[,seq2]]
[,STR={_1}][,END={nn}][,MAX=nnnnn ][,TRUNC]
                                    [ [,LAST]][ {nn}][ { }][ ][ ]
```

*
The delimiter that marks the beginning and end of the search and replacement strings. You can use any special character that does not appear in the search or replacement strings.

string-1
The 1- to 35-character search string. Blanks are valid characters. For wide record masters, the string is from 1 to 123 characters.

string-2
The 0- to 35-character replacement string. Blanks are valid characters. For wide record masters, the string is from 1 to 123 characters.

If the search string is longer than the replacement string, all data in columns to the right of the search string shift left and blanks are inserted at the end of the range of columns being processed.

If the replacement string is longer than the search string:

- All data in columns to the right of the string shift right to accommodate the additional length.
- AllFusion CA-Librarian checks to ensure that nonblank columns are not shifted past the end of the range of columns being processed.

  If shifting of data to the right shifts nonblank characters past the end of the range of columns being processed:

  – Editing of the record being processed is bypassed.

  – A message describing the condition is written on the Update Record listing.

  – All other updates to the module are processed normally. To suppress all updates to the module when this overflow condition occurs, place a Y in column 7 of the -EMOD control statement.

To delete each occurrence of the search string, omit the replacement string. For example:

```
-EDIT %XYZ%%
```

seq1
The sequence number of the first or only record to edit.

seq2
The sequence number of the last record of the range of records to edit. To edit a single record, omit seq2.

LAST
The end of the range of records to edit as the last record of the module.

STR=[01|*nn*]
The number of the first column of the range of columns to edit. Specify a 2-digit number for *nn*. The default is 01.

END=*nn*
The number of the last column of the range of columns to edit. Specify a 2-digit number for *nn*. For AFO master files, the default is 72. For wide record master files, the default is the current record length. You can specify up to a five digit number.

MAX=***nnnnn***
For wide record master files only, specifies the maximum length of the edited record. The default is the length of the record at the time the edit statement is processed for that record. You can specify up to a five digit number.

TRUNC
For wide record master files only, specifies that characters can be truncated to limit the record length to the value specified with MAX or the default for MAX. If TRUNC is not specified, -SEL processing is aborted if non-blank characters must be truncated to limit the record length.

See Chapter 2 for a full description of the importance of sequencing in edit control statements.

Note: You can include a maximum of 30 -EDIT statements in an AllFusion CA-Librarian control stream for a single execution. For GPO processing, the maximum is 20 -EDIT statements.

-EMOD

Function

The -EMOD execution control statement marks the end of one module's control statements and data. Use -END to mark the end of the entire control stream. The -EMOD statement is optional.

Syntax

```
-EMOD [X]
      [Y]
```

X
In column 7, AllFusion CA-Librarian bypasses the hash count check for the module being updated. This lets you apply a duplicate set of updates to the module.

Y
In column 7, AllFusion CA-Librarian suppresses all updating of the module if shifting during a -EDIT operation causes an overflow of data out of the range of columns being edited on any record.

-END

Function

The -END execution control statement marks the end of the entire AllFusion CA-Librarian control stream. It is optional, but if omitted, AllFusion CA-Librarian issues a warning message on the Update Record Listing suggesting that you omitted part of the control stream.

Syntax

```
-END [comments]
```

comments

Any user-supplied comments. Under VSE/ESA, this commentary generates a final record written to the compilation file. It must begin in column 6 and can continue through column 80.

-EXTRACT

Function

The -EXTRACT Utility control statement copies all the data records of a specific level of the named member into the jobstream file (OSJOB for z/OS and OS/390 or JOBSTR for VSE/ESA). -EXTRACT copies the data records to the jobstream file without opening the master file for update. -INC commands embedded among the data records are expanded during the copy process.

The Utility -EXTRACT control statement is a read-only facility that uses File Access Interface Routines (FAIR) to copy the data records of the member level to the jobstream file. Using the jobstream file as input to a subsequent execution of AllFusion CA-Librarian, these data records can be transferred between master files or compiled. If the Utility run is limited to a single execution of the -EXTRACT control statement, a sequential data set is created that contains the copied data records of the named member.

Note: Must follow a -OPT UTILITY control statement.

Syntax

```
-EXTRACT member-name[,options]
```

member-name
The name of the member that AllFusion CA-Librarian copies to the output file.

options
Valid options are: ARC=, ARCINC=, VAR, NOVAR, and NOINC. With ARC= or ARCINC=, use the relative level number or the date (or date and time) form of specifying a level. Absolute level number specification is not supported.

If you do not specify ARC= or ARCINC=, -EXTRACT copies the data records of the most current level for the member.

Note the following:

- -EXTRACT does not copy the control information of the selected member, only the data records.
- You can code multiple -EXTRACT statements for a single Utility run. The extracted data records from each member are stacked in the jobstream file.
- You must specify all Utility control statements that the utility is to process before the first -EXTRACT statement. An exception to this rule is the -INC statement or any additional -EXTRACT statement. These two statements are always processed regardless of where they are specified in the Utility. Utility control statements, other than -INC or -EXTRACT, that are specified after the first -EXTRACT statement are written unaltered to the jobstream file. Non-Utility commands specified in a Utility run are always written unaltered to the jobstream file.
- -EXTRACT expands -INC commands embedded in the data records of the selected member. SLAT variables of included members are resolved using the calling member's control information, rather than the called member's control information.
- -EXTRACT requires File Access Interface Routines (FAIR) to execute. Therefore, make certain that FAIR is available before using -EXTRACT. FAIR is described in the *File Access Interface Routines Guide*.

For additional instruction in the use of this command and of the Utility generally, see the *User Guide.*

-FILL

Function

The -FILL control statement fills a specified range of columns with a string of characters. The string overlays any data in those columns. -FILL is an editing control statement. See Chapter 2 for a description of the importance of sequencing in editing control statements.

Syntax

```
-FILL *string*[seq1[,seq2]][,COL={73}] [,MAX={nnnnn}]
              [    [,LAST]][     {nn}] [     {     }]
```

*

The delimiter that marks the beginning and end of the string. It can be any special character that does not appear in the string.

string

The 1- to 35-character string. Blanks are valid characters. For wide record masters, the string is from 1 to 123 characters.

seq1

The sequence number of the first or only record affected by the -FILL control statement.

seq2

The sequence number of the last record of the range of records affected. To alter a single record, omit seq2.

LAST

The end of the range of records to process as the last record of the module.

COL=[73|*nn*]

The number of the column where the operation is to begin. Specify a 2-digit number for *nn* For members with records larger than 80 bytes, you must specify COL. It can be up to five digits. For members with records no longer than 80 bytes, including all members of AFO master files, the default is 73. The number of columns affected is equal to the length of the string.

MAX=*nnnnn*

For wide record master files only, specifies the maximum length of the edited record. The default is the length of the record at the time the edit statement is processed for that record. You can specify up to a five digit number.

-HST

Function

The -HST documentary control statement records the reasons for updating a module. You can use as many -HST control statements as necessary to fully document the updates. For each -HST control statement, AllFusion CA-Librarian creates a history record that becomes a permanent part of the module. You can add to the set of history records at any time. Each history record is stamped with the date it was added to the module, unless more than one was added on the same day. In that case, the first record is date stamped.

By examining the date stamps of history records and the corresponding updated data records, you can determine the reasons for the updates. All information you supply on a -HST statement is permanently stored on the master file, unless deleted by a -HSTD control statement, and written to the Module listing.

Syntax

```
-HST narrative
```

narrative

The 1- to 75-character narrative stored with the module, beginning in column 6. Blanks are valid characters.

-HSTD

Function

Use the -HSTD documentary control statement to delete all history records from the current archive level of a module. You cannot selectively delete history records. New -HST control statements can follow, but not precede, the -HSTD control statement.

Syntax

```
-HSTD
```

-INC (Non-Utility -INC)

Function

Use the -INC control statement to include all or some of the data records of one module in another module at compilation time. Specify the module to include on the -INC control statement. The module where the data records are included is the calling module.

The data records of the included module are not duplicated in the calling module; instead, the -INC control statement itself is maintained in the calling module and acts as a pointer to the included module. Each time the calling module is written to the compilation file (through the EXEC option), the data records from the included module are written in place of the -INC control statement.

An included module can be a calling module. Both the calling and included modules must reside on the same disk master file. You can use the AUXINC User Contributed Routine for z/OS and OS/390, and AUXDINC for VSE/ESA, to search up to eight additional AllFusion CA-Librarian master files to expand members referenced in unexpanded -INC statements. See the *Systems Services Guide* for details on AUXINC and AUXDINC.

Because the -INC control statement is expanded only in the compilation file, it is useful for saving disk space when a DSECT or file description or any set of records must be made available to more than one module.

Syntax

```
-INC module-name[,seq1[,seq2][,ARC={date}]]
                [     [,LAST][     {Lx } ]]
                [     [     ][     {-y } ]]
```

module-name
The name of the included module.

seq1
The sequence number of the first or only record to be included by the -INC control statement.

seq2
The sequence number of the last record of the range of records to include. To include a single record, omit seq2.

LAST
The end of the range of records to include as the last record of the module.

ARC=[*date*|L*x*|*-y*] date

A date (or date and time) when the level was current. Its full form is:

yymmddhhmmss

You can omit an even number of digits from the right. If you do, AllFusion CA-Librarian assumes the highest possible values for those digits. For example, if you specify ARC=96, AllFusion CA-Librarian selects the level that was current on the last second of the last minute of the last day of December 1996.

L*x*

The absolute level number as reported on the Update Record listing. You must precede the absolute level number with the letter L.

-y

The relative level number. You must precede the relative number with a minus sign (-). The current level number is -0; -1 is one level older than the current level; -2 is two levels older than the current level, and so on.

Note the following:

- The calling and all included modules must reside on the same disk master file.
- AllFusion CA-Librarian treats the -INC control statement as a data record. You can edit and update it.
- Columns 73 through 80 of the -INC control statement itself must be blank unless the calling module is added with the SEQCHK= option. In this case, a valid sequence number must appear in those columns, regardless of the location you specified in the SEQCHK= option.
- You can use any number of -INC control statements in a module. Any number of -INC control statements can refer to the same module.
- An included module can also be a calling module, to a maximum of 64 levels of nesting.
- A module must not include itself or any module that directly or indirectly called it.
- ARC= specified on the -INC control statement always overrides ARCINC= on the -SEL control statement. You can specify default levels by supplying the ARCINC= parameter on an -ADD or -SEL control statement.

- AllFusion CA-Librarian validates the -INC control statement when the module containing it is written to the compilation file. If the -INC control statement cannot be expanded (that is, replaced by the data records of another module), the control statement itself is written. If an -INC control statement appears on the compiler listing, this is what could be wrong:
  - The included module does not exist on the master file.
  - The format of the -INC control statement is invalid.
  - The -INC control statement exceeds the maximum 64 levels of nesting.
  - The -INC control statement refers to the module it is part of or to a module that has called that module.
  - You specified ARC= on the -INC control statement and it is a non-archived module, or you specified a level that does not exist.

The following figures show a module containing an -INC statement as it appears on the module listing, and then as it appears on the compilation file.

```
RUN NO. 49     DATE=mm/dd/yy  TIME=0933                    LISTING OF MODULE COBOLTST                    PAGE 1
DESCRIPTION        COBOL TEST PROGRAM

MASTER FILE        LIBR.DEV.MAST
ADDED TO MASTER    07/03/97
LAST DATE COPIED     NONE
LAST DATE UPDATED  03/14/97 092609
ARCHIVING STATUS     ACTIVATED
NUMBER OF LEVELS        4
CURRENT LEVEL NO.       3
NUMBER OF RECORDS      24
NUMBER OF UPDATES       3
NUMBER OF ACCESSES      7
SEQUENCE PARAMS    01/6/0010/0010 - RESEQ
COMPRESS STATUS      FULL
COPYDD STATUS        NOT ACTIVATED
COBOL COPY STATUS    NOT ACTIVATED
MODULE STATUS        TEST

PASSWORD           GCFG
PROGRAMMER         SMITH
LANGUAGE           COB
PROC PARAMETER     $NOJCL

HISTORY RECORD(S) FOR THIS MODULE:
  07/03/97         MODULE FOR TEST COMPILES
000010 ID DIVISION.
000020 PROGRAM-ID. COBOLTST.                                                              03/14/97
000030 ENVIRONMENT DIVISION.
000040 INPUT-OUTPUT SECTION.
000050 FILE CONTROL.
000060     SELECT INFILE PASSING TO UT-S-IN.
000070 DATA DIVISION.
000080 FILE SECTION.
000090 FD INFILE
000100 DATA RECORD IS INREC.
-INC INREC                                                                 00000110
000120 WORKING-STORAGE SECTION.
000130 77 IN-COUNT     PIC 999.
000140 77 IN-INDEX     PIC 999.
000150 77 IN-ID        PIC XXXXX.
000160 PROCEDURE DIVISION.
000170     ACCEPT IN-COUNT FROM SYSIN.
000180     OPEN INPUT INFILE. _                                                          03/14/97
000190     PERFORM IN-COPY VARYING IN-INDEX FROM 1 BY 1 UNTIL                            03/14/97
000200        IN-INDEX EQUAL IN-COUNT. STOP RUN.                                         03/14/97
000210 IN-COPY.
000220     READ INFILE.                                                                  03/14/97
000230     MOVE INFILE-ID TO IN-ID.                                                      03/14/97
000240     DISPLAY IN-ID UPON SYSOUT.
```

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. COBOLTST.
000030 ENVIRONMENT DIVISION.
000040 INPUT-OUTPUT SECTION.
000050 FILE CONTROL.
000060     SELECT INFILE ASSIGN TO UT-S-IN.
000070 DATA DIVISION.
000080 FILE SECTION.
000090 FD  INFILE
000100     DATA RECORD IS INREC.
000010 01  INREC.                          }
000020     02 INFILE-ID    PIC XXXXX.      } *
000030     02 FILLER       PIC X(75).      }
000120 WORKING-STORAGE SECTION.
000130 77  IN-COUNT        PIC 999.
000140 77  IN-INDEX        PIC 999.
000150 77  IN-ID           PIX XXXXX.
000160 PROCEDURE DIVISION.
000170     ACCEPT IN-COUNT FROM SYSIN.
000180     OPEN INPUT INFILE.
000190     PERFORM IN-COPY VARYING IN-INDEX FROM 1 BY 1 UNTIL
000200        IN-INDEX EQUAL IN-COUNT. STOP RUN.
000210 IN-COPY.
000220     READ INFILE.
000230     MOVE INFILE-ID TO IN-ID.
000240     DISPLAY IN-ID UPON SYSOUT.
```

* The -INC statement that appears here in the calling module is replaced in the compilation file by the module to which it refers.

-INC (Utility -INC)

Function

The -INC Utility control statement directs AllFusion CA-Librarian to place the named module or portion of a module into an output file.

Note: Must follow a -OPT UTILITY control statement.

You can use the Utility -INC statement to create a general-purpose card-image file. Using this file as input to a subsequent execution of AllFusion CA-Librarian, you can:

- Transfer groups of records in a module or between modules.
- Transfer modules between master files.
- Create a new module from portions of one or more existing modules.
- Split an existing module.

For detailed instructions on using this command and the Utility generally, see the *User Guide* and the *Systems Services Guide.*

Syntax

```
-INC module-name[,seq1[,seq2]]
```

module-name
The name of the module that AllFusion CA-Librarian copies to the output file.

seq1
The sequence number of the first or only record to copy to the output file. You can omit leading zeros. To copy an entire module, omit seq1.

seq2
The sequence number of the last record of the range of records to copy to the output file. You can omit leading zeros. To copy a single record, omit seq2.

-INDEX

Function

The -INDEX UTILITY control statement generates a Master File Index Listing, providing a one-line summary of information stored about each module on the master file. The listing includes:

- Module name
- Description
- Password
- Programmer
- Date added
- Date and time of last update

This differs from the -OPT INDEX command in that it is a read-only facility; UPDATE authority to the master file is not required.

The Utility -INDEX control statement is not valid for wide record master files.

Note: Must follow a -OPT UTILITY control statement.

Syntax

```
-INDEX
```

See the appendix for an illustration of a Master File Utility Index Listing (which is the same as the non-utility version).

-INDEX(M)

Function

The -INDEX(M) UTILITY control statement generates a Management Index Listing, providing a two-line summary of information stored about each module on the master file. The listing includes:

- Module name
- Description
- Password
- Programmer
- Date added
- Date and time of last update
- Archiving status of each module
- Programmer name
- Number of records
- Number of accesses through the batch AllFusion CA-Librarian program and the Immediate Online Update
- Number of updates
- Number of blocks
- Three-character language code
- Cataloged procedure
- Date of last access
- Sequence parameters
- Number of archive levels

This differs from the -OPT INDEX(M) command in that it is a read-only facility; UPDATE authority to the master file is not required.

The Utility -INDEX(M) control statement is not valid for wide record master files.

Note: Must follow a -OPT UTILITY control statement.

Syntax

```
-INDEX(M)
```

See the *Systems Services Guide* for a sample Manager Utility Index Listing.

-INDEX(pgmr)

Function

The -INDEX(pgmr) UTILITY control statement generates a Programmer Index Listing, providing a one-line summary of information stored about each module on the master file for which the specified programmer is recorded as responsible. The listing includes:

- Module name
- Description
- Password
- Programmer
- Date added
- Date and time of last update

This differs from the -OPT INDEX(pgmr) command in that it is a read-only facility; UPDATE authority to the master file is not required.

The Utility -INDEX(pgmr) control statement is not valid for wide record master files.

Note: Must follow a -OPT UTILITY control statement.

Syntax

```
-INDEX(SMITH)
```

See the appendix for a sample Programmer Utility Index Listing (which is the same as the non-utility version).

-INDEX(S)

Function

The -INDEX(S) UTILITY control statement generates a Statistical Index Listing of the master file. The listing includes:

- Total number of records
- Members
- Blocks
- Free blocks
- Used blocks
- Dead blocks
- Index blocks
- Master index entries
- Percentage of free space
- Percentage of dead space
- Member occupying the largest number of blocks

This differs from the -OPT INDEX(S) command in that it is a read-only facility; UPDATE authority to the master file is not required.

The Utility -INDEX(S) control statement is not valid for wide record master files.

Note: Must follow a -OPT UTILITY control statement.

Syntax

```
-INDEX(S)
```

See the *Systems Services Guide* for a sample Statistical Utility Index Listing.

-INS

Function

The -INS updating control statement inserts one or more new records into an existing module. New data records must immediately follow this statement.

Syntax

```
-INS {seq1 }
     {FIRST}
     {LAST }
```

seq1
The sequence number of the record where new data records are inserted. (You can omit leading zeros when specifying sequence numbers.)

Note: If specifying multiple -INS control statements or -INS control statements with -REP or -DEL control statements, you must arrange the statements in the control stream so that the sequence numbers of the records they refer to are in ascending numeric order. See Chapter 2 for more information.

FIRST
New records are inserted before the first record of the module.

LAST
New records are inserted after the last record of the module.

-JCL

Function

Parameters on the -JCL miscellaneous control statement control the automatic generation of JCL for a module. You can find detailed instructions on the use of this control statement in Chapter 5 for z/OS and OS/390 and Chapter 6 for VSE/ESA. Using this statement is both operating-system specific and site-dependent.

You can use as many -JCL control statements as are necessary to supply the parameters you need. Place -JCL control statements in the control deck following an -ADD or -SEL control statement and preceding any documentary control statements.

To delete a previously defined parameter, specify the keyword and its equal sign, but do not specify a value.

Syntax

```
-JCL parameter[,parameter...]
```

parameter
Consists of a keyword, an equal sign (=), and a value. If the value includes an embedded comma or blank, you must enclose the entire value in slashes. You can find a list of the parameters for z/OS and OS/390 in Chapter 5 and in Chapter 6 for VSE/ESA.

-JCLT

Function

The -JCLT miscellaneous control statement modifies one or more of the parameters that can be specified on the -JCL control statement, during the current execution only. The parameters stored on the master file are not changed. You can use both -JCL and -JCLT control statements for a single module. -JCL control statements must precede -JCLT control statements.

You can find detailed instructions on using this control statement in Chapter 5 for z/OS and OS/390 and Chapter 6 for VSE/ESA. Using this control statement is both operating-system specific and site-dependent.

Syntax

```
-JCLT parameter[,parameter...]
```

parameter
Consists of a keyword, an equal sign (=), and a value. For more information on these parameters, see Chapter 5 for z/OS and OS/390 and Chapter 6 for VSE/ESA.

-LANG

Function

Use the -LANG control statement to identify the language in which the module is written. You can add or change this specification at any time.

Syntax

```
-LANG lan
```

lan
The 1- to 3-character language code identifying the language (that is, COB for COBOL, ASM for Assembler, FOR for FORTRAN, or DAT for data). No imbedded spaces are allowed.

-OPT

Function

The -OPT control statement governs the processing of a single AllFusion CA-Librarian execution. The -OPT statement can carry processing options to override initialization defaults.

There are many special uses in AllFusion CA-Librarian for the -OPT control statement, such as:

- Invoking the Utility (-OPT UTILITY).
- Producing an index (-OPT INDEX or -OPT INDEX(*pgmr*)).
- Comparing two archiving levels of a module (-OPT COMPARE).
- Listing occurrences of -INC, COPYDD, or COBOL COPY statements in a master file (-OPT XREF).

For many operations of CA-Librarian, you do not need to specify the -OPT control statement. For example, for processing only one module or for processing several modules requiring different options, you need not specify the -OPT control statement.

When you use the -OPT control statement, it must be the first statement in the control stream. You can use as many -OPT control statements as necessary to specify the options you want since these statements cannot be continued.

Syntax

```
-OPT options[,options]
```

option
Valid options are:

COL1%	NOINDEX	PUNCH
COMPARE	NOLIST	REINIT
CON	NOPR	RESEQ
EXEC	NOPUNCH	SEP
INDEX	NORESEQ	TEMP
INDEX(pgmr)	NOSEP	TEMPS
LIST	NOSORT	UNCON
LISTH	NOVAR	VAR
NOEXEC	PR	XREF

Note the following:

- Be aware that you must specify the following to have NORESEQ affect multiple existing members:

  ```
  OPT REINIT, -SEL NAME=,/1,6,10,10/, SEQ=NORESEQ
  ```

- The following commands affect all future members until you specify another option:
  - -OPT REINIT
  - -OPT RESEQ

-PGMR

Function

Use the -PGMR documentary control statement to identify the programmer responsible for the module. You can add or change this specification at any time.

A -PGMR statement can follow an -ADD, -SEL, -LANG, or -DESC statement. Do not put a -PGMR statement after a -HIST statement.

Syntax

```
-PGMR programmer-name
```

programmer-name
The 1- to 15-character programmer name. Blanks and commas are not valid characters.

If you include blanks or commas in the programmer name, they are not picked up, and whatever follows them in the -PGMR control statement is lost.

-PRINT

Function

The -PRINT Utility control statement writes, on the Update Record listing, all or some of the data records of a module. -PRINT control statements must immediately follow the -OPT UTILITY control statement. (If the -PRINT statements do not immediately follow the -OPT UTILITY statement, they are treated as data records.) -PRINT statements can appear in any order in the control deck and can refer to modules in any sequence.

When a -PRINT control statement refers to an archived module, only the current level of the module prints.

Each -PRINT control statement is written at the top of a new page of the Update Record listing, followed by the data records to which it refers.

Note: Must follow a -OPT UTILITY control statement.

Syntax

```
-PRINT module-name[,seq1[,seq2]]
```

module-name
The name of the module to process.

seq1
The sequence number of the first or only record to print or punch. You can omit leading zeros. To print an entire module, omit seq1.

seq2
The sequence number of the last record of the range of records to print. You can omit leading zeros. To print a single record, omit seq2.

The following example is a typical Utility PRINT/PUNCH jobstream in z/OS and OS/390.

```
//UTIPUNCH JOB 70020,CA.SOLOMON,CLASS=X
// EXEC PGM=CA-Librarian
//SYSPUNCH DD SYSOUT=P
//MASTER   DD DSN=libr.master.file,DISP=SHR
//SYSIN    DD *
  -OPT UTILITY
  -PRINT GAMMA,1120,1690
  -PUNCH ALPHA,420,635
  -PRINT DELTA,1800,3450
  -PRINT DELTA,200,670
/*
//
```

The following example is a typical Utility PRINT/PUNCH jobstream in VSE/ESA.

```
* $$ JOB JNM=PRIJOB,CLASS=A
// JOB PRIJOB
// DLBL MASTER,'CAI.LIBR.LIBTESTB',,DA
// EXTENT SYS004,SYSWK2,1,0,8816,40
// ASSGN SYS004,DISK,VOL=SYSWK2,SHR
// DLBL JOBSTR,'LIBR.JOBSTR1',,SD
// EXTENT SYS008,SYSWK2,1,0,9100,10
// ASSGN SYS008,DISK,VOL=SYSWK2,SHR
// EXEC LIBRPROG,SIZE=AUTO
  -OPT UTILITY
  -PRINT TESTMOD,1120,1690
  -PUNCH MODULE1,420,635
  -PRINT MODULE3,1800,3450
  -PRINT MODULE3,200,670
  -END
/*
/&
* $$ EOJ
```

You can combine the creation of card-image output file with print and punch requests in a single execution of the Utility.

-PUNCH

Function

The -PUNCH Utility control statement copies all or some of the data records of a module to the punch file.

-PUNCH control statements must immediately follow the -OPT UTILITY control statement. (If the -PUNCH statements do not immediately follow the -OPT UTILITY statement, they are treated as data records.) -PUNCH statements can appear in any order in the control deck and can refer to modules in any sequence.

When a -PUNCH control statement refers to an archived module, only the current level of the module is punched.

Each -PUNCH control statement is written on the Update Record listing (at the top of a page if preceded by a -PRINT control statement) and to the punch file, preceded by two blank records and followed by the data records to which it refers.

Note: Must follow a -OPT UTILITY control statement.

Syntax

```
-PUNCH module-name[,seq1[,seq2]]
```

module-name
The name of the module to process.

seq1
The sequence number of the first or only record to punch. You can omit leading zeros. To punch an entire module, omit seq1.

seq2
The sequence number of the last record of the range of records to punch. You can omit leading zeros. To punch a single record, omit seq2.

The previous example is a typical Utility PRINT/PUNCH jobstream. You can combine the creation of card-image output file with print and punch requests in a single execution of the Utility.

-REP

Function

The -REP updating control statement replaces one or more existing records with new ones. New data records must immediately follow this statement.

The number of new data records supplied need not be equal to the number of records replaced, but must not be zero.

Syntax

```
-REP {seq1[,seq2   ] }
     {    [,LAST   ] }
     {ALL [,NOAUDIT] }
```

seq1
The sequence number of the first or only record to replace.

Note: If specifying multiple -REP control statements or -REP control statements with -INS or -DEL control statements, you must arrange the statements in the control stream so that the sequence numbers of the records they refer to are in ascending numeric order.

seq2
The sequence number of the last record of the range of records to replace. To replace a single record, omit seq2.

LAST
The end of the range of records to replace as the last record of the module.

ALL
Replaces all records of the module. You cannot perform any other updating operation when you specify -REP ALL. Resequenced updating (RESEQ) is invoked when you use -REP ALL, except when you also specify the SEQCHK= option. SEQCHK= is ignored on a -REP ALL when LIBAUDIT is activated.

Note: Specifying -REP ALL automatically invokes LIBAUDIT processing. Thus, only the records actually modified are date stamped in the case of a non-archived module, or are stored as a separate level in the case of an archived module.

NOAUDIT
NOAUDIT suppresses LIBAUDIT processing. When you specify NOAUDIT, no date stamps are generated and, in the case of an archived module, an entire copy of the module is retained as a separate level.

-SCAN (Non-Utility -SCAN)

Function

The -SCAN editing control statement examines a group of records for the occurrence of a specified string of characters and writes all records containing the string on the Update Record listing. This operation does not modify any data records.

-SCAN is an editing control statement. See Chapter 2 for a description of the importance of sequencing in editing control statements.

Syntax

```
-SCAN *string-1*[seq1[,seq2]][,STR={01}][,END={72}]
                [    [,LAST]][     {nn}][     {nn}]
```

*

The delimiter that marks the beginning and end of the search string. You can use any special character that does not appear in the search string except for the backslash and close bracket characters.

string

The 1- to 35-character search string. Blanks are valid characters. For wide record masters, the string is from 1 to 123 characters.

seq1

The sequence number of the first or only record to scan.

seq2

The sequence number of the last record of the range of records to scan. To scan a single record, omit seq2.

LAST

The end of the range of records to scan as the last record of the module.

STR=[01|*nn*]

The number of the first column of the range of columns to scan. Specify a 2-digit number for nn; the default is 01. For wide record masters, the member can be up to five digits.

END=[72|*nn*]

The number of the last column of the range of columns to scan. Specify a two-digit number for *nn*. The default is 72. For wide record masters, the member can be up to five digits.

-SCAN (Utility -SCAN)

Function

The -SCAN Utility control statement scans all data records stored on the entire master file for occurrences of a character string. AllFusion CA-Librarian writes each record containing the string and the name of the module where the record appears on the Update Record listing. You can include a maximum of 30 -SCAN statements in the control deck for a single execution. You cannot combine the Utility -SCAN function with any other Utility function (for example, creating a card-image output file). The -SCAN statement must follow the -OPT UTILITY control statement.

If the master file to scan contains archived modules, only the current level of each is scanned.

This facility does not scan printer modules (see the PRMOD processing option).

Syntax

```
-SCAN *string*[STR={01}][,END={72}][,CTX={00}]
              [    {nn}][     {nn}][     {nn}]
```

*

The delimiter you choose to mark the beginning and end of the search string. It can be any special character that does not appear in the search string. The operand immediately following the end delimiter of the search string is not preceded by a comma.

string

The 1- to 72-character search string. Blanks are valid characters. For wide record masters, the string is from 1 to 123 characters.

STR=[01|*nn*]

The number of the first column of the range of columns to scan. Specify a two-digit number for *nn*. The default is 01. For wide record masters, you can specify up to a five digit number.

END=[72|*nn*]

The number of the last column of the range of columns to scan. Specify a two-digit number for *nn*. The default is 72. For wide record masters, you can specify up to a five digit number.

CTX=[00]

Specifies how many data records are written from the same module preceding and following each occurrence of the search string. Specify a two-digit number for *nn*. The maximum value of *nn* is 10. The default is 00. For example, to write the record preceding and following each record containing the search string, specify CTX=01. CTX= shows the context of each string.

Each -SCAN statement read is assigned a number between 01 and 30. When a record containing a specified search string is written on the Update Record listing, the number of the -SCAN statement specifying that string is written alongside it (for example, [02]). If a record contains more than one specified search string, only the lowest applicable number is written.

Following is an example of a global scan.

```
//UTISCAN JOB 70017,CA.JAEGER,CLASS=0
// EXEC PGM=CA-Librarian
  -OPT UTILITY
  -SCAN &LA    R9,AWORK&,CTX=05
  -SCAN *BWORK*
  -END
/*
//
```

-SCANR

Function

The -SCANR Utility control statement limits a global scan operation to a specific range of modules on the master file. You can use this statement only as part of a global scan operation. It must directly follow the -OPT UTILITY statement.

Syntax

```
-SCANR {module-1[,module-2]}
       {,module-2          }
```

module-1

The name of the first module of a range of modules to scan. To designate the first module on the master file as the beginning of the range of modules, specify a comma instead of mod1, followed by mod2.

module-2

The name of the last module of a range of modules to scan. If you specified a module name for mod1, you can omit mod2. This designates the last module on the master file as the end of the range of modules.

The following example illustrates the use of the -SCANR Utility control statement.

```
//UTISCANR JOB 70017,CA.PRINCE,CLASS=0
// EXEC CA-Librarian
     .
     .     (Standard JCL statements)
     .
  -OPT UTILITY
  -SCANR ,LVS201B
  -SCAN *BWORK*,CTX=04
  -SCAN %LA     R9,AWORK%
  -END
/*
//
```

AllFusion CA-Librarian scans the master file from the first module residing on it through the module named LVS201B.

-SEL

Function

The -SEL module control statement selects a module from the master file for processing. Processing options that you include on the -SEL control statement override any execution defaults that you specify on a -OPT statement.

Syntax

-SEL *module-name*[,*password*][,*options*]

module-name
The name of the module.

password
The password, which might be required. Site management determines this at master file initialization.

AllFusion CA-Librarian assigns a password, consisting of four random alphabetic characters, to each module that you add to the master file. You can assign your own password with the PSWD= option.

By including the password on the -SEL statement, you protect yourself from inadvertently updating a module with a similar name.

If you forget a module's password, refer to an index listing. When you supply an incorrect password (whether it is required or not), CA-Librarian does not process the module.

Note: AllFusion CA-Librarian passwords are not intended for use as a security mechanism since they appear in various batch AllFusion CA-Librarian listings and in many online displays. However, you can use them as a check to verify that you keyed a member name correctly.

options
Valid options are:

- ARC
- ARC=
- ARCINC=
- ARCLR=
- ARCOFF
- CLEARID
- CCOPY*
- CCOPYOFF*
- CON
- COPYDD*
- COPYDDOFF*
- COPY=
- EXEC
- EXEC(R)
- LIST
- LISTH
- LOCK
- NOARC
- NOCHK*
- NOEXEC
- NOLIST
- NOPR
- NOPUNCH
- NORESEQ
- NOSORT
- NOVAR
- PERM
- PR
- PSWD=
- PUNCH
- RENAME=
- RESEQ
- SEQ=
- SEQCHK=
- SEQUPD
- SYNCHK(D)*
- SYNCHK(l,a)*
- TEMP
- TEMPS
- UNCON
- UNLOCK
- VAR
- VERS=

Note: The options with an * are not valid for Wide Record Master Files.

If the processing of the module fails for any reason, any additional processing requested is not performed.

The -SEL control statement has a special syntax with the -OPT COMPARE control statement. See the COMPARE option for details.

When you use the RESEQ parameter, you must include positional parameters in the -SEL command:

```
-SEL TESTMOD,ZDZW,SEQ=/1,6,1_,1_/,RESEQ
```

-XREF

Function

The -XREF control statement produces a cross-reference report of modules containing -INC control statements, COPYDD commands, and COBOL COPY verbs.

Syntax

```
-XREF [keyword] [keyword] ...
-XREF [INC][,COPYDD][,COPY]
```

keyword

Must be one of the following:

- INC—Members containing -INC control statements appear in the report.
- COPYDD—Members containing COPYDD commands appear in the report.
- COPY—Members containing COBOL COPY verbs appear in the report.

To include a module containing COBOL COPY verbs in the Module Cross Reference Report, either the syntax checker or the CCOPY option must be active for that module.

Notice the following about AFO master files:

- You can specify any of the three keywords or any combination.
- If you do not use the -XREF command and only specify the -OPT XREF, all occurrences of -INC statements, COPYDD commands, and COBOL COPY verbs appear in the report.
- If you specify -XREF and omit all three operands, all three types are included in the report.
- If you use the -XREF command, it must directly follow the -OPT XREF statement.
- Additional work files are required. See the *CA-Librarian User Guide* for -OPT XREF JCL examples.

Notice the following about wide record master files:

- You can specify any keyword. Members containing the keyword appear in the report.
- Each keyword can be up to eight characters and cannot have imbedded blanks.
- You can specify up to 64 keywords.

- You can use multiple -XREF control statements to specify additional keywords.
- The -XREF statements must follow the -OPT XREF statement.
- If you omit all operands, only -INC statements are included in the report.
- Workfiles are not needed.

-XREFR

Function

The -XREFR control statement produces a cross-reference report of modules containing -INC control statements, COPYDD commands, and COBOL COPY verbs for a range of modules.

Syntax

```
-XREFR [module-1[,module-2]]
       [,module-2          ]
```

module-1
The name of the first module of the range of modules to select for the cross reference. If omitted, selection of modules begins with the first module in the master file.

module-2
The name of the last module of a range of modules to cross reference. If omitted, selection of the modules ends with the last module in the master file.

Note the following:

- You must specify at least one of the parameters.
- If you omit the -XREFR command, all of the members in the master file are cross referenced.
- Use the -XREFR command with -OPT XREF alone to generate a report listing all occurrences of the default keywords in a specific range of members. For AFO master files, the default keywords are -INC, COPYADD, and COPY. For wide-record master files, the default keyword is -INC.

See the *User Guide* for -OPT XREF JCL examples as additional work files are required for AFO master files.

Chapter

4

Processing Options

Processing options come after commands on the control statement. If your control statement specifies a module name, the processing options must follow the module name and the password (if one is needed). You can specify processing options in any order on a control statement. Separate them with a comma (,).

You can use some AllFusion CA-Librarian processing options in the online environments. Other do not work. Where an option is supported online, it is mentioned in the description of that option.

ARC

ARC invokes the Archiving Facility for a module (that is, the module becomes an archived module). AllFusion CA-Librarian does not archive modules until you code ARC for the module. Once coded, archiving begins with the current execution and continues until you specify the NOARC or ARCOFF option.

Archiving is useful for reconstructing a module as it existed at some time in the past. This is a key requirement of any auditing operation.

Another use for archiving is during program development. As specifications for a given project change or should a gross error occur (for example, the deletion of a large group of data records), you can go back to a previous level of the module.

Use ARC on an -ADD or -SEL control statement. It works online.

Use the NOARC option to suspend archiving. Use the ARCOFF option to change an archived module back to a non-archived module (terminate archiving). ARCOFF and NOARC are described later in this section.

Note the following:

- You must have initialized the master file containing the module you want to archive before archiving. Check with your system programmer to see if archiving is available for your use.
- The maximum number of levels that you can store at one time is established at master file initialization. AllFusion CA-Librarian issues a warning message when you approach this maximum. When the maximum is reached, the oldest level of the module is **permanently** deleted.
- You cannot change the RESEQ/NORESEQ status of an archived module.
- The Utility -INC, -PRINT, -PUNCH, and -SCAN statements process only the current level of an archived module. The Utility -COPY statement copies all levels of an archived module.
- If you use the RENAME= option for an archived module, all levels are renamed.

For further information, see the ARC=, ARCLR=, ARCOFF, NOARC, NORESEQ, RENAME=, and RESEQ processing options and the -INC control statement.

ARC=[date | Lx | -y]

Selects a specific level of an archived module for processing. When you use the ARC= option to update a module, AllFusion CA-Librarian deletes all levels of the module created more recently than the one specified. The new level of the module that you create then becomes the current level. For example, if a module has five levels (levels 0, 1, 2, 3, and 4) and you want to select level 2 for updating, specify ARC=L2. If the module is successfully updated, levels 3 and 4 are permanently deleted and the newly created level becomes level 3.

Use this option when you want to access or update a level other than the current level.

Use ARC= on a -SEL or -INC control statement. It is supported online with -SEL.

Specify one of the following to select the level of the archived module:

date
A date (or date and time) when the level was current. Its full form in the short date format is:

```
yymmddhhmmss
```

Its full form in the long date format is:

```
yyyymmddhhmmss
```

The long date format is available only for wide record master files. You can omit an even number of digits from the right. If you do, AllFusion CA-Librarian assumes the highest possible values for those digits. For example, if you specify ARC=96 (short date format), AllFusion CA-Librarian selects the level that was current on 12/31/96 at 23:59:59, not on 01/01/96 at 00:00:00.

Lx
The absolute level number as reported on the Update Record listing. You must precede the absolute level number with the letter L.

-y
The relative level number. You must precede the relative level number with a minus sign (-). The current level number is -0, -1 is one level older than the current level, -2 is two levels older than the current level, and so on.

Note: You cannot access a level of an archived module (ARC= option) and delete the same level (ARCLR= option) at the same time.

For further information, see the ARC, ARCLR=, ARCOFF, COPY=, and NOARC options.

ARCINC=[date | Lx | -y]

Provides a comprehensive archiving level selection for all of the included modules that are referred to in your module.

Use the ARCINC= option to select a level, other than the current level, for included archived modules appearing in your module.

Use ARCINC= on a -ADD or -SEL control statement.

Specify **one** of the following to select the level of the archived module to include:

date
A date (or date and time) when the level was current. Its full form in the short date format is:

```
yymmddhhmmss
```

Its full form in the long date format is:

```
yyyymmddhhmmss
```

The long date format is available only for wide record master files.

You can omit an even number of digits from the right. If you do, AllFusion CA-Librarian assumes the highest possible values for those digits. For example, if you specify ARC=96, AllFusion CA-Librarian selects the level that was current on 12/31/96 at 23:59:59.

Lx
The absolute level number as reported on the Update Record listing. You must precede the absolute level number with the letter L.

-y
The relative level number. You must precede the relative level number with a minus sign (-). The current level number is -0, -1 is one level older than the current level, -2 is two levels older than the current level, and so on.

Note: -INC statements specifying the ARC= option override the ARCINC= option specified on a -ADD or -SEL statement. For example, you select a module and specify ARCINC=-1 and, in that module are -INC statements specifying the option ARC=-2. The data records for that particular included module are taken from the -2 level.

For further information, see the ARC= option for the -INC control statement.

ARCLR=date

Deletes every level of an archived module up to, but not including, the level that was current on the date that you specify. Archiving continues for updates made after an ARCLR= operation.

The ARCLR= option lets you delete old levels.

Use ARCLR= on a -SEL control statement. It is supported online.

Select the oldest level that you want to retain by specifying:

date

Date (or date and time) when the level was current. Its full form in the short date format is:

```
yymmddhhmmss
```

Its full form in the long date format is:

```
yyyymmddhhmmss
```

The long date format is available only for wide record master files.

You can omit an even number of digits from the right. If you do, AllFusion CA-Librarian assumes the highest possible values for those digits.

For example, assume that today's date is 03/01/96 and that the module to process was updated on the following dates only:

08/15/94, 09/23/94, 02/11/96, and 02/20/96.

If you specify ARCLR=94, the level created on 08/15/94 is deleted. If you specify ARCLR=95, the level created on 08/15/94 is still the only level deleted, since the level created on 09/23/94 was still current at the end of 1994. If you specify ARCLR=96, the levels created on 08/15/94 and 09/23/94 are deleted.

Note: You cannot access a level of an archived module (ARC= option) and delete the same level (ARCLR= option) at the same time. For further information, see the ARC, ARC=, ARCOFF, COPY=, and NOARC options.

ARCOFF

Terminates archiving of a module and deletes all levels except the current level. The module becomes a non-archived AllFusion CA-Librarian module. Once specified, ARCOFF remains in effect until you specify the ARC option.

Use this option when it is no longer necessary to retain more than one level of a module. It is more efficient in both storage and processing time to maintain a 1-level module as a non-archived AllFusion CA-Librarian module.

Use ARCOFF on a -SEL control statement. It is supported online. ARCOFF is ignored if specified on a -ADD control statement.

AUTOARC

Overrides the wide record master file default of NOAUTOARC so that new members are automatically archived without specifying the ARC option in the -ADD statement.

CCOPY

Invokes the expansion of the AllFusion CA-Librarian COBOL COPY verb for a module. The verb expansion begins with the current execution. AllFusion CA-Librarian searches for the named module first in the master file that contains the copying module. If the named module is not found on that master file, AllFusion CA-Librarian searches for it in a master file defined by a DD statement with a name of COPYMAST (z/OS and OS/390) or in file SYS006 (VSE/ESA). If the module to copy does not exist in either of those files, the COPY verb is not expanded.

CCOPY lets you expand the COBOL COPY verb when either your site has not installed the COBOL Syntax Checker or you do not want to activate the COBOL Syntax Checker for the module.

Use the CCOPY option on a -ADD or -SEL control statement.

The COPY verb can exist anywhere between column 12 and 72 of a record. The name of the module to copy can appear in quotation marks. The format of the AllFusion CA-Librarian COBOL COPY statement is as follows:

```
[0x dataname] COPY copy-module [SUPPRESS]
              [REPLACING operand-a(1) BY operand-b(1)    ]
              [         [operand-a(n) BY operand-b(n)]...]
```

0x *dataname*
Replaces the corresponding dataname in the copy-module if the level number of the copy-module matches 0x.

copy-module
The name of the AllFusion CA-Librarian module to introduce into the jobstream file.

SUPPRESS
AllFusion CA-Librarian ignores this parameter.

REPLACING
Changes some or all of the words in the copy-module to words the program references. All occurrences of operand-a(n) in the copy-module are replaced by operand-b(n) as the copy-module is introduced into the jobstream. The REPLACING clause is limited to 600 bytes of information.

operand-a
A word composed of a combination of no more than 30 characters from the set of characters 0 through 9, A through Z, and hyphen (-) but not beginning with a hyphen. If operand-a terminates with a hyphen, any word in the copy-module having operand-a as a prefix has the prefix replaced by operand-b (which need not have a hyphen).

BY
Required if REPLACING was specified.

operand-b
A word (see above) or a literal. A literal is a combination of no more than 58 characters from the set of characters 0 through 9, A through Z, hyphen (-), plus sign (+), minus sign (-), and decimal point (.), enclosed in quotation marks (") or single quotes (').

Example 1

Assume the following two modules:

Module MODA

```
01   MODA COPY MODB.
```

Module MODB

```
01   MODB.
02   A PIC XX
03   B PIC XX.
```

When MODA is selected for execution, the following is written to the compilation file:

```
01 MODA.
   02 A PIC XX.
   02 B PIC XX.
```

Example 2

Assume the following two modules:

Module MODA

```
02   MODA COPY MODB.
REPLACING PRE- BY PAYROLL.
```

Module MODB

```
02   MODB.
     03   PRE-TAX [OC XX.
     03   PRE-RATE PIC XX.
```

When MODA is selected for execution, the following is written to the compilation file:

```
02   MODA.
     03   PAYROLLTAX PIC XX.
     03   PAYROLLRATE PIC XX.
```

Use the CCOPYOFF option to terminate expansion of the COBOL COPY verb. When a module has syntax checking invoked, you can also terminate verb expansion by using the SYNCHK option.

Note the following:

- You should not use the literal COPY in the REMARKS or NOTES section of a program. Its appearance can invoke expansion.
- Any AllFusion CA-Librarian -INC control statements in the named module are expanded.
- Pseudo-text is not supported.
- You cannot continue nonnumeric literals in the COPY clause.
- Errors in the assignment of SYS006/COPYMAST suppress COBOL COPY expansion.
- You must format the AllFusion CA-Librarian COBOL COPY verb as described here, regardless of the particular compiler in use. Syntax errors or use beyond the examples given here can cause unpredictable results, including incomplete copying to the compilation file of the module.
- REPLACING clauses containing more than 600 bytes of information can cause a table overflow condition resulting in a failure to expand the COPY verb.

For further information, see the SYNCHK and CCOPYOFF options. These options are for traditional master files only.

CCOPYOFF

Permanently deactivates expansion of the COBOL COPY verb for a module.

Use the CCOPYOFF option when you do not want to expand the COBOL COPY verb. You can leave the COPY verb in the module and reactivate it later by specifying the CCOPY option.

Use CCOPYOFF on a -SEL control statement.

Use the CCOPY option to activate COBOL COPY expansion for a module permanently. To deactivate COBOL COPY expansion temporarily, use CCOPYOFF with the TEMP option.

Note: You cannot use the CCOPYOFF option for a module in the same execution that sets the second part of the SYNCHK(l,a) option for the module to either SE or CE.

For further information, see the CCOPY, SYNCHK(l,a), and TEMP options. These options are for traditional master files only.

CLEARID

Clears, to blanks, columns 73-80 of data records **added** to a module. You can use this option to clear old sequence numbers from records you are adding to a module. Use this option to reduce the amount of space required to store the module.

Use CLEARID on a -ADD or -SEL control statement. CLEARID is supported online.

Note: CLEARID does not affect any record beginning with a slash (/). The CLEARID option does not clear columns 73-80 of existing data records.

COL1%

Passes a percent sign (%) in column 1 of any data record to the compilation file as a percent sign, rather than as a blank. Use COL1% on a -OPT control statement.

COMPARE

For disk master files only. Examines two different levels of one archived module and reports the differences between those two levels. A read-only function of AllFusion CA-Librarian, COMPARE does not produce an updated master file, but it does produce an output file on SYS008 (VSE/ESA) or OSJOB (z/OS and OS/390) that can be used as input to a subsequent execution of AllFusion CA-Librarian to update a master file. COMPARE produces:

- A Cross-level report of the differences between two levels of a module that documents the changes that were made to the module.
- A AllFusion CA-Librarian update control stream on SYS008 or OSJOB. The output control stream, if applied to the older level of the module, produces the newer level.
  - Incorporate the changes made by several update runs into a single control stream.
  - Produce an update deck that can be sent to other sites if a module is maintained at more than one.
  - Replace multiple levels of a module with a single level.

To invoke COMPARE, place the COMPARE option on a -OPT control statement:

```
-OPT COMPARE[,options]
```

The following options are recognized on the -OPT COMPARE statement: NOSEP, NOPR, PR, and SEP.

Any additional -OPT statements required for the subsequent execution should directly follow the -OPT COMPARE statement. These additional -OPT statements are written to the output file and print on the Cross-level report.

Follow the -OPT statement with a -SEL control statement specifying the NEW= and OLD= options.

The format is:

```
-SEL module-name[,password],NEW={yymmddhhmmss            },
                               {yyyymmddhhmmss           }
                               {Lx                       }
                               {-y                       }
                          OLD={yymmddhhmmss }[,options]
                               {yyyymmddhhmmss           }
                               {Lx }
                               {-y }
```

module-name
The name of the archived module to process.

password
The password of the module.

NEW=
The most recently created of the two levels being compared.

OLD=
The older of the two levels being compared.

Specify one of the following to select the level of the archived module:

yymmddhhmmss
The date and time (in the short date format) that the level was created.

yyyymmddhhmmss
The date and time (in the long date format) that the level was created. The long date format is available only for wide record master files.

Lx
The absolute level number as reported on the Update Record listing. You must precede the absolute level number with the letter L.

-y
The relative level number. You must precede the relative level number with a minus sign (-). The current level number is -0, -1 is one level older than the current level, -2 is two levels older than the current level, and so on.

options
Provides a means of requesting additional processing for the module. COMPARE processes the PR and NOPR options and writes to the output file. All other options are written to the output file for subsequent processing.

You do not have to express new and old levels in the same way. For example, you can specify the NEW= option by date and time and the OLD= option by absolute level number. You can specify NEW= and OLD= in any sequence on the -SEL statement. They can be intermixed with other processing options.

See the *User Guide* for -OPT COMPARE JCL examples.

COMPRESS=FULL | PART | NONE

Changes the compression level of your module from the default compression level that was established for your master file at initialization time. When you change the compression level of a module, only those records that are subsequently added or changed are compressed to the new level.

FULL
The default. Removes all spaces from each record and each set of four bytes to compress into three. For modules with sequence numbers in columns 1 through 6, FULL compresses all COBOL reserved words into a single byte.

PART
Removes all spaces from each record before it is stored on the master file.

NONE
Stores 80 byte records in 80 bytes with no compression.

Specify COMPRESS= on a -OPT control statement. It is supported in online environments.

Note: Space reduction depends on the contents of the file.

CON

Processing each module for which CON is specified is conditional. Such a module is not processed unless every preceding attempt to process a module during the same execution was successful. When processing of a module is suppressed in this manner, updates to the module are not applied and a diagnostic message is written on the Update Record listing. In addition, any other processing options specified for the module are ignored.

When updating two modules, A and B, it is possible to bypass updating of module B if the update of module A fails for any reason:

```
-OPT NOSEP
-SEL A
   .
   . (updates to A)
   .
-EMOD
-SEL B,CON
   .
   . (updates to B)
   .
-END
```

The CON option on the -SEL statement for module B instructs AllFusion CA-Librarian not to apply the specified changes to module B, regardless of their validity, unless the changes to module A were completely successful.

You can use the CON option to make the generation of a job stream conditional.

Assume that you want to update module A. Leading JCL for compilation is in module LEADA. Trailing JCL for link editing and testing is in module TRAILA.

```
-SEL A
   .
   . (updates to A)
   .
-SEL LEADA,EXEC,CON
-SEL A,EXEC,CON
-SEL TRAILA,EXEC,CON
```

The CON option instructs AllFusion CA-Librarian not to generate the job stream for module A unless all updates to A were successful.

Use CON on a -OPT, -ADD, -SEL, or -DLM control statement. CON is supported in an online environment.

To override this option, use the UNCON option.

Note: This option is available only for disk master files. The initialization default is CON.

For further information, see the UNCON option.

COPY=newname

Creates a copy of the selected module and gives it the new name that you specify.

AllFusion CA-Librarian automatically assigns a new password to the new module. COPY= generally runs tests on a duplicate version of a module while retaining the unaltered original on the master file as a backup. Once the duplicate is fully tested, you can delete the original module and rename the duplicate so that is has the name of the original module.

Use COPY= on a -SEL control statement. It is supported online.

Note the following:

- Copy processing is related to COPYMAST. COPYMAST must be a LIBR master file.
- The name that you assign to the new module must not already exist on the master file.
- If you combine COPY= with the ARC= or the ARCLR= option, the new module consists of all the archived levels that the original module would include after the update if you did not specify COPY=.
- If you combine COPY= with ARCOFF, the new module becomes a non-archived AllFusion CA-Librarian module.
- If you combine COPY= with ARCOFF and ARC= options, the new module becomes a non-archived AllFusion CA-Librarian module consisting of the level specified through the ARC= parameter.
- Processing options of the -SEL statement and any updates following the -SEL statement are applied to the duplicate, not to the original module.

For further information, see the ARC=, ARCLR=, ARCOFF, PSWD=, and RENAME= options.

COPYDD

Permanently activates the AllFusion CA-Librarian/DataDictionary interface (LIB/DD) for a module. LIB/DD scans a AllFusion CA-Librarian module for the COPYDD statement whenever the module is written to the compilation file (through the EXEC option).

Use COPYDD on a -ADD or -SEL control statement when you want to expand COPYDD statements that you coded in a module. COPYDD is not supported online.

For further information on the use of the COPYDD statement, see the *User Guide*.

Use the COPYDDOFF option to permanently deactivate LIB/DD for a module.

Note: The interface checks the format of the COPYDD statement, but does not check the logic or validity of the statements in the program. If there is a format error, AllFusion CA-Librarian writes a message to the Update Record listing and the COPYDD statement is not expanded. Detailed error messages appear in either the compilation file listing or the compiler or assembler listing, depending on the kind of COPYDD error made. AllFusion CA-Librarian does not issue an abnormal return code.

See the *Systems Services Guide* (VSE/ESA or z/OS and OS/390) for information about the return codes.

For further information, see the COPYDDOFF option. This option is for traditional master files only.

COPYDDOFF

Permanently deactivates the AllFusion CA-Librarian/DataDictionary interface (LIB/DD) for a module.

Use COPYDDOFF on a -SEL control statement.

Use the COPYDD option to permanently activate LIB/DD for a module. To temporarily deactivate LIB/DD, use COPYDDOFF with the TEMP option.

For further information, see the COPYDD and TEMP options. These options are for traditional master files only.

EXEC

Writes a copy of each module for which it is specified to the compilation file. JCL is generated automatically, if requested, and -INC statements are expanded. Any statements that follow a -DATA statement are also written to the compilation file.

Use EXEC on a -OPT, -ADD, or -SEL control statement. It is not supported in an online environment.

To override this option, use the NOEXEC option.

For further information, see Chapter 5 for z/OS and OS/390, Chapter 6 for VSE/ESA, and the NOEXEC, NOVAR, and VAR options.

EXEC(R)

For z/OS and OS/390 systems. When directing the output of a compile to a partitioned data set, if the member name AllFusion CA-Librarian assigned already exists in the PDS, AllFusion CA-Librarian does not normally stow the member, but issues an error message.

To direct AllFusion CA-Librarian to replace a like-named member, use the EXEC(R) option on the -OPT, -ADD, or -SEL statement.

EXEC(R) is not supported in an online environment.

See Chapter 5 for a full description of EXEC and EXEC(R) as they apply to z/OS and OS/390 execution of source programs stored as AllFusion CA-Librarian modules.

GPO

Used on a -OPT control statement to invoke the Group Processing Option, GPO can identify modules in a master file on the basis of a variety of criteria and generate a control stream to process these identified modules in a number of different ways.

The format is:

```
-OPT GPO
```

Follow the -OPT GPO statement with a GPO control statement. You can find detailed information on the use of this option and GPO statements and their functions in the *Systems Services Guides* for both z/OS and OS/390 and VSE/ESA.

Note: GPO processing has a limit of 20 -EDIT commands.

INDEX

Generates a Master File Index listing, providing a one-line summary of information stored about each module on the master file. The listing includes: module name, description, password, programmer, date added, and date and time of last update.

For example, use INDEX on a -OPT control statement:

```
-OPT INDEX
```

Note: You can generate only one index listing during a single execution of AllFusion CA-Librarian.

See the INDEX(pgmr) option. See the Appendix for an illustration of a Master File Index listing.

INDEX(M)

Generates a Management Index listing, providing a two line summary of information stored about each module on the master file. The listing includes:

- The module name
- Description
- Password
- Programmer
- Date added
- Date and time of last update
- Date and time member copied via batch Librarian Utility Processing (Wide master files only)
- Archiving status of each module
- Programmer name
- Number of records
- Number of accesses through the batch AllFusion CA-Librarian program and the Immediate Online Update
- Number of updates
- Number of blocks
- Three-character language code
- Cataloged procedure
- Date of last access
- Sequence parameters
- Number of archive levels.

See the *System Services Guide* for a sample INDEX(M) listing.

For example, use INDEX(M) on a -OPT control statement:

```
-OPT INDEX(M)
```

Note: You can generate only one index listing during a single execution of AllFusion CA-Librarian.

INDEX(pgmr)

Generates a Programmer Index listing, providing a one line summary of information stored about each module on the master file for which the specified programmer is recorded as responsible. The listing includes: module name, description, password, date added, and the date and time of last update.

For example, use INDEX(pgmr) on a -OPT control statement:

```
-OPT INDEX(SMITH)
```

Note the following:

- You can generate only one index listing during a single execution of AllFusion CA-Librarian.
- You must store the programmer name with the module control information through a -PGMR statement, the PGMR= parameter of a -JCL statement, or a PGMR command in an on-line environment.

See the Appendix for a sample Programmer Index.

INDEX(S)

Generates a Statistical Index listing of master file information including:

- Total number of records
- Members
- Blocks
- Free blocks
- Used blocks
- Dead blocks
- Index blocks
- Master index entries
- Percentage of free space
- Percentage of dead space
- Member occupying the largest number of blocks.

See the *System Services Guide* for a sample INDEX(S) listing.

For example, use INDEX(S) on a -OPT control statement:

```
-OPT INDEX(S)
```

A Statistical Index listing of a wide record master file includes the total number of records, history records, members, the widest record and the members with the most total records, most history records and most data records. See the *Systems Services Guide* for a sample wide records INDEX(S) listing.

Note: You can generate only one index listing during a single execution of AllFusion CA-Librarian.

LIST

Generates a Module listing of each module it is specified for, consisting of module control information, history records, and data records with their associated date stamps.

Use the LIST option to generate a Module listing whenever you need to view a module exactly as it appears on the master file or to determine what changes were made to a module on a given date.

Use LIST on a -OPT, -ADD, and -SEL control statement. It is not supported in an online environment.

To override this option, use the NOLIST option.

See the LISTH and NOLIST options.

LISTH

Generates an abbreviated Module listing of each module it is specified for, consisting of module control information and history records. Data records are not listed.

Use LISTH on a -OPT, -ADD, or -SEL control statement. LISTH is not supported in an online environment.

LOCK

Prevents anyone other than the LOCK owner from updating a member. Users with unlimited access authority for the member or the lock owner can unlock a member.

A lock owner is the user ID who originally locked the member. Use LOCK on -ADD and -SEL control statements. LOCK and UNLOCK does not create a new archive level. The last update date and time remains unchanged.

This option is valid for wide record master files only.

See the *Security Administration Guide* for more information about ALL member access authority.

LONGDATE

Overrides the master file default of SHORTDATE so that you can specify archive dates with four-digit years. You must use LONGDATE for years before 1960 or after 2059 (for example, 19960819 instead of 960819).

The LONGDATE format is available only for wide record master files.

NOARC

Suspends archiving of a module. When the module is updated while NOARC is in effect, the level selected for updating is replaced, more recently created levels (if any) are deleted, and older levels (if any) are retained. Once specified, NOARC remains in effect for the module until you specify either the ARC or the ARCOFF option.

Use this option to replace an unneeded level of a module.

Use NOARC on a -SEL control statement. NOARC is ignored if specified on an -ADD control statement. NOARC is supported online.

To resume archiving of a module, use the ARC option. To terminate archiving of a module, use the ARCOFF option.

For further information, see the ARC, ARC=, ARCLR=, and ARCOFF options.

NOAUTOARC

Overrides the wide record master file default of AUTOARC so that new members are **not** archived unless the ARC option is specified in the -ADD statement.

NOCHK

Deactivates the COBOL Syntax Checker option for a module for the current execution only. The stored values of the SYNCHK(l,a) option are not altered.

Deactivating the Syntax Checker reduces execution time.

Use NOCHK on a -ADD or -SEL control statement.

Note: To permanently deactivate the COBOL Syntax Checker option for a module, use the SYNCHK(D) option.

For further information, see the SYNCHK(D) and SYNCHK(l,a) options.

NOEXEC

Prevents AllFusion CA-Librarian from writing a copy of any module it is specified for to the compilation file.

When EXEC is specified as the default during master file initialization, you can use NOEXEC to override the default for a single execution.

Use NOEXEC on a -OPT, -ADD, or -SEL control statement. In an online environment, NOEXEC is ignored.

To override this option, use the EXEC option.

NOINDEX

Inhibits the writing of a Master File Index listing for a tape backup file. Use NOINDEX on a -OPT control statement.

For further information, see the INDEX option.

NOLIST

Inhibits the writing of a Module listing of any module for which it is specified.

When LIST is specified as the default during master file initialization, you can use NOLIST to override the default for a single execution.

Use NOLIST on a -OPT, -ADD, or -SEL control statement. In an online environment, NOLIST is ignored.

To override this option, use the LIST option.

For further information, see the LIST and LISTH options.

NOPR

Abbreviates the Update Record listing by suppressing the writing of all but the first and last record of each group of records deleted.

AllFusion CA-Librarian usually writes all records replaced during a -REP operation and all records deleted during a -DEL operation on the Update Record listing. When you are replacing or deleting a large number of records, use this option to reduce paper consumption.

When PR is specified as the default during master file initialization, you can use NOPR to override the default for a single execution.

Use NOPR on a -OPT or -SEL control statement. It is supported in an online environment.

To override this option, use the PR option.

NOPUNCH

Prevents AllFusion CA-Librarian from copying the data records of any module it is specified for to the punch file.

When PUNCH is specified as the default during master file initialization, you can use NOPUNCH to override the default for a single execution.

Use NOPUNCH on a -OPT, -ADD, or -SEL control statement. It is not supported online.

To override this option, use the PUNCH option.

NORESEQ

Invokes nonresequenced updating of each module for which it is specified.

AllFusion CA-Librarian assigns sequence numbers to new data records, beginning with 1 greater than the sequence number of the record after which the new records are inserted, and incrementing by 1. If necessary, the sequence numbers of existing records are also incremented to prevent duplicate sequence numbers.

Use NORESEQ on a -OPT, -ADD, or -SEL control statement. It is supported online.

To override this option, use the RESEQ option.

Note the following:

- NORESEQ is ignored when you use -REP ALL unless you have LIBAUDIT or archived modules.
- You cannot specify NORESEQ for an archived module that has RESEQ status.

The only way to reset the RESEQ/NORESEQ status of an archived module is to change the module back to a non-archived AllFusion CA-Librarian module (using ARCOFF) and respecify the status.

For a comparison of the NORESEQ and RESEQ options, see the RESEQ option. See also the ARC, ARCOFF, and SEQUPD options.

NOSEP

Compresses the front piece and eliminates the page ejects that normally appear on the Update Record listing preceding each module control statement.

Use this option to reduce paper consumption in the Update Record listing. Use NOSEP on a -OPT control statement. To override this option, use the SEP option.

For further information, see the SEP option.

NOSORT

For backup tapes only. You can specify the control statements used as input to a AllFusion CA-Librarian execution to process a tape backup file in any order. To minimize tape movement and positioning, AllFusion CA-Librarian sorts the input stream, placing the control statements in proper alphanumeric order before processing them. This sorting adds processing overhead and imposes a limit of 500 modules per execution.

You can bypass the sorting and the limit on the number of modules that can be processed. Place the control statements in proper order, and then execute AllFusion CA-Librarian with the NOSORT option specified.

NOSORT indicates that the input module control statements are in ascending order and that the sorting of the control stream can be bypassed.

Use NOSORT on a -OPT control statement.

Note: You can specify this option only for Utility executions.

For further information, see the appropriate *Systems Services Guide* (z/OS and OS/390 or VSE/ESA).

NOVAR

Deactivates the VAR option for the current execution only. It is necessary to specify NOVAR only if the AllFusion CA-Librarian master file default at your site is VAR and you want to override the default.

Use NOVAR on a-OPT,-ADD, or -SEL control statement.

To override this option, use the VAR option.

For further information, see the EXEC and VAR options.

PERM

Designates that updates to a module are permanently applied.

Use PERM to override an execution default of TEMP or TEMPS. Use PERM on a -SEL control statement.

For further information, see the TEMP and TEMPS options.

PR

Writes a copy of every record deleted from each module it is specified for on the Update Record listing.

Use PR to override an initialization or execution default of NOPR.

Use PR on a -OPT or -SEL control statement. Its use is supported online.

For further information, see the NOPR option.

PRMOD

For z/OS and OS/390 only. A printer listing is any data set containing 121- or 133-byte, fixed length records, with the first byte of each record being a valid ASA or machine carriage control character. Each printer listing is stored on the master file as a separate printer module.

For traditional master files, you can store printer listings as 80-byte records on a AllFusion CA-Librarian disk master file under z/OS and OS/390 systems, although not all options are supported for these printer modules. You can add, replace, delete, or print an entire printer module, but you cannot modify individual lines in it. You can specify these processing options for a printer module: COPY=, EXEC, LIST, and RENAME=.

These options are not supported for printer modules: ARC, ARCOFF, COBOL COPY, COPYDD, NOARC, PUNCH, the Utility COPY and PUNCH options, and VAR.

For wide record masters, all processing and updating options are supported for PRMODs.

To add a printer module, specify the PRMOD option on a -ADD control statement followed by an -AUX control statement identifying the data set where the printer listing is copied.

```
-ADD USERLIST,PRMOD,LIST
-AUX D1(USERMEM)
```

To generate a listing of a printer module, use the LIST option.

Note: AllFusion CA-Librarian assumes that the first byte of each record of the printer module contains an appropriate carriage control character, the same type used to write to the Module listing. AllFusion CA-Librarian normally uses machine control characters to write the Module listing. To list printer modules containing ASA control characters, you must specify RECFM=FA or RECFM=FBA as a DCB= subparameter on the SYSPRINT DD statement.

All listings printed in a single execution must have the same type of control character. To list a printer module with ASA characters and a printer module with machine control characters, a separate execution of AllFusion CA-Librarian is required for each.

For traditional master files, if you specify the EXEC option, only the first 80 bytes of each record are written to the job stream file.

The only updating operation permitted for a printer module is replacement of the entire module. To accomplish this, select the module through a -SEL control statement and follow that -SEL control statement with a -AUX control statement identifying the data set where the new printer listing is copied.

```
-SEL USERLIST,PSWD,LIST
-AUX D1(USERMEM)
-END
```

For wide record master files, EXEC output is not limited except by LRECL of OSJOB DCB.

PSWD=pswd

Initially assigns or changes the password of a module.

AllFusion CA-Librarian passwords are not intended for use as a security mechanism since they appear in various batch AllFusion CA-Librarian listings and in many online displays. However, you can use them as a check to verify that you correctly keyed a member name.

Passwords can be any four-character, alphanumeric string.

You can assign the same password to more than one module.

If you do not specify one, AllFusion CA-Librarian assigns a password to any module you add to a master file.

Note the following:

- You cannot assign the following AllFusion CA-Librarian reserved words as passwords:

  BYPP, EXEC, FULL, LIST, NONE, NOPC, NOPR, PERM, TEMP, and TEST.

- For wide record master files, the list of AllFusion CA-Librarian reserved words that you cannot assign as passwords varies slightly from regular AllFusion CA-Librarian. They are EXEC, LIST, NOPR, PERM, TEMP, and LOCK.

- AllFusion CA-Librarian reserved words NOPC and TEST are valid passwords for wide record master files.

PUNCH

Copies the data records of each module it is specified for to the punch file.

To override this option, use the NOPUNCH option.

RENAME=newname

Changes the name of a module to the new name that you specify.

Use RENAME= on a -SEL control statement. It is supported online.

Note the following:

- The new name that you specify must not already exist on the master file.
- If you use the RENAME= option for an archived module, all levels are renamed.

For further information, see the COPY= option.

RESEQ

Invokes resequenced updating of each module for which it is specified. AllFusion CA-Librarian assigns a sequence number to each record of the module, based on the sequence number attributes of the module.

```
Existing Module    Control Deck 1    Updated Module

Module  ABC        -OPT RESEQ        000020 record A
000020  record A   -SEL ABC          000025 record 1
000025  record B   -INS 20           000030 record 2
000030  record C    record 1         000035 record 3
00035   record D    record 2         000040 record 4
                    record 3         000045 record 5
                    record 4         000050 record B
                    record 5         000055 record C
                   -END              000060 record D

                   Control Deck 2    Updated Module
                   -OPT NORESEQ      000020 record A
                   -SEL ABC          000021 record 1
                   -INS 20           000022 record 2
                    record 1         000023 record 3
                    record 2         000024 record 4
                    record 3         000025 record 5
                    record 4        *000026 record B
                    record 5         000030 record C
                   -END              000035 record D
```

* Represents a record of the original module: Its sequence number was incremented to accommodate the new records.

The figure above shows one control deck with the RESEQ option specified and one with the NORESEQ option specified. The content and arrangement of records in both control decks are identical. Only the options differ. The values that were specified for the SEQ= option are /1,6,5,20/.

Use RESEQ on a -OPT, -ADD, or -SEL control statement. RESEQ is supported online.

To override this option, use either the NORESEQ option or the SEQCHK= option.

Note: When processing an archived module, you cannot invoke RESEQ if the module was previously processed using NORESEQ.

For further information, see the NORESEQ, SEQ=, and SEQCHK= options.

SEP

Forces a page eject on the Update Record listing before each module control statement.

Use SEP to override an initialization default of NOSEP.

Use SEP on a -OPT control statement.

For further information, see the NOSEP option.

SEQ=[s,l,i[,v] | COBOL]

Assigns sequence number attributes to a module. AllFusion CA-Librarian uses sequence number attributes to assign a sequence number to each data record of the module. Use SEQ= on a -ADD or -SEL control statement.

The operands are:

s
The column where the sequence number starts. For traditional master files, acceptable values are 1-81. If you specify 81, the sequence number associated with each record is stored outside the record. All 80 bytes of the record are then available for data.

For wide record master files, values 0 - 99999. The value zero (0) or a number greater than the members widest record length indicates that the sequence number is external.

l
The length of the sequence number field. Acceptable values are 1-9 for traditional master files. If you specify a starting column number of 81, you must specify a length of 6. If you do not specify a starting column number of 81, the sum of the starting column number and the length must not exceed 81.

i
The increment. Acceptable values are 1-9999. The default value is 10.

v
The sequence number assigned to the first record of the module during addition of the module or during resequenced updating. Acceptable values are 1 through 9999. The default value is the increment value (i).

COBOL
Sequence numbers are in columns 1 through 6, with a starting value of 10 and an increment of 10.

AllFusion CA-Librarian places the sequence number of any record containing a slash or percent sign in column 1 or a percent sign in column 80, in columns 73 through 80, regardless of the location defined in the SEQ= option, unless the specified starting column is 73 or more.

Use SEQ= on a -ADD or -SEL control statement.

To alter one or more of the sequence number attributes, respecify the SEQ= option on a -SEL control statement.

Note: The maximum value of a sequence number is 16,777,215. If you exceed this maximum, unpredictable results occur.

SEQCHK=[/s,l,i,v/|COBOL]

AllFusion CA-Librarian does **not** generate sequence numbers but checks instead that sequence numbers on records added to the master file are present in the specified columns and are in ascending numeric order. If an error is found in the sequencing of the records, the module is not added to the master file (or replaced) and an error message is written on the Update Record listing.

Use SEQCHK= on a -ADD control statement or on a -SEL control statement when performing a -REP ALL operation. It is supported online.

The operands are:

s
The column in which the sequence number starts.

l
The length of the sequence number field. Sequence numbers can be 1 to 9 digits in length.

i
The increment used when the module is resequenced. Acceptable values are 1 through 9999. The default value is 10.

v
The sequence number to assign to the first record of the module during resequenced updating. Acceptable values are 1 through 9999. The default value is the increment value (i).

COBOL
Defines sequence numbers as in columns 1-6, beginning with 10 and incrementing by 10.

SEQCHK= Assigns sequence number attributes to the module in the same way that the SEQ= operand does. The NORESEQ option is automatically set. You do not have to include leading zeros or right-adjust the sequence numbers. The increment and starting value are not used while the module is added, but are retained for resequencing when the module is subsequently updated.

Note the following:

- This option is valid **only** when a permanent update is made to the module for which it is specified. The SEQCHK option is ignored on a -SEL control statement when performing a -REP ALL operation using LIBAUDIT.
- You **must** place the sequence number of a record containing a slash in column 1 and of any -INC control statement in the last 8 columns of the record, regardless of the sequence number attributes of the module.
- The maximum value of a sequence number is 16,777,215. If you exceed this maximum, unpredictable results occur.

For further information, see the SEQ= option.

SEQUPD

Allows implicit updating of a module by sequence number when the sequence number is defined in the record. There are two methods:

- Explicit updating. AllFusion CA-Librarian control statements identify the types of updates to perform (for example, -INS for inserting records, -REP for replacing records, -DEL for deleting records). You identify the specific data records to update by including their sequence numbers on the control statement.
- Implicit updating. Disk master files only. The sequence numbers that you specify on new data records identify the types of updates to perform. If the sequence number matches an existing record number, the existing record is replaced. If the sequence number does not match an existing record number, the new record is inserted. To delete a record, you still must use the -DEL control statement.

The sequence numbers that you include on new data records identify the types of updates to perform. You must position the sequence numbers on the new data records in the columns designated by the sequence number attributes. You do not have to include leading zeros or right-adjust the sequence numbers.

Place new data records after the -SEL statement in ascending order by sequence number. New data records either replace existing records or are inserted into the module. If the sequence number of a new record matches one on the master file, the new record replaces the corresponding master file record. If the sequence number of a new record does not match one in the master file, the record is inserted into the module to maintain ascending sequence.

Input

```
-SEL PAYROLL,SEQUPD,NORESEQ
200     MOVE A-TAX TO B-TAX.
201     ADD B-TAX TO C-TAX.
215     GO TO TAX-C-COMP.
```

Existing Module

```
          .
          .
          .
000180 PERFORM A-TAX-CALC.
000190 MOVE ZEROS TO TAX-IN.
000200 MOVE ATAX TO BTAX.
000210 PERFORM D-TAX-CALC.
          .
          .
          .
          .
```

Resulting Module

```
          .
          .
          .
000180 PERFORM A-TAX-CALC.
000190 MOVE ZEROS TO TAX-IN.
000200 MOVE A-TAX TO B-TAX.
000201 ADD B-TAX TO C-TAX.
000210 PERFORM D-TAX-CALC.
000215 GO TO TAX-C-COMP.
          .
          .
          .
```

To insert records before the first record of the module, assign a sequence number less than the sequence number of the first record. To insert records after the last record of the module, assign a sequence number greater than the sequence number of the last record.

You can combine implicit and explicit updating in a single execution.

Use SEQUPD on a -SEL control statement. It is supported online.

Note the following:

- The master file you are working with might have been initialized to prohibit implicit updating.
- The SEQUPD option lets you perform insert and replace operations only. To delete records from a module, you must use the -DEL control statement.
- You cannot use implicit updating for modules whose sequence numbers are defined as outside the record.
- If you specify the SEQUPD and NORESEQ options for a module simultaneously, the records that you insert into the module retain their sequence numbers.

SYNCHK(D) permanently deactivates the COBOL Syntax Checker facility for a module.

Use SYNCHK(D) on a -SEL control statement. It is not supported online.

For further information, see the NOCHK and SYNCHK(l,a) options.

SHORTDATE

Overrides the master file default of LONGDATE so that you can enter archive dates with two-digit years (for example, 960819) instead of four-digit years (for example, 19960819). Year 60 and higher is assumed to be the 20th century. Years less than 60 are assumed to be the 21st century (for example, 2059).

This option is for wide record master files only.

SYNCHK(l,a)

Verifies that the records of a module consist of syntactically valid COBOL statements for every execution.

Using the second operand of this option, you can replace COBOL COPY statements with the data records of other modules when the compilation file is written. This works much the same as expansion of -INC statements.

When you use this option, the COBOL Syntax Checker scans each record of the module and verifies that sentence structure, punctuation, use of reserved words, and so on, conform to COBOL rules. The Syntax Checker does not check the validity of data or procedure references.

Records containing errors are written on the Update Record listing, followed by an error message. The sequence number of a record in error can be different from its original sequence number (number on the original listing) if a new sequence number was assigned during updating. Since the records in error are reported on the Update Record listing in the order of their appearance in the module, you should be able to locate them without exact sequence numbers.

Use SYNCHK(l,a) on a -ADD or -SEL control statement. SYNCHK is not supported in an on-line environment, where it is flagged as an error.

The operands are as follows:

l

The level of COBOL whose rules of syntax are applied. Specify one of the following:

- CA—ANS COBOL.
- CD—COBOL D.
- CE—COBOL E.
- CF—COBOL F.
- CW—z/OS and OS/390 COBOL (ANS COBOL 74).

a

The action to take. Specify one of the following:

- S—Suppresses writing of the module to the compilation file in the event of a syntax error and permanently deactivates the expansion of the COBOL COPY verb. The LIST option is automatically invoked for the module in such a case, thereby providing you with a complete listing of the module that, together with the messages on the Update Record listing, let you readily locate the errors.
- C—Permanently deactivates the expansion of the COBOL COPY verb. In the event of a syntax error, writing of the module to the compilation file is not suppressed, nor is the LIST option automatically invoked.
- SE—Same as S, except that it permanently activates the expansion of the COBOL COPY verb for disk master files (see notes on the expansion of the COBOL COPY verb).
- CE—Same as C, except that it permanently activates the expansion of the COBOL COPY verb for disk master files.

To alter one or more of the values stored with this option, place a SYNCHK(l,a) option with the new values on a -SEL control statement. You must specify both values, even if only one value is altered. The respecified SYNCHK(l,a) option is stored with the module.

To permanently deactivate the SYNCHK(l,a) option, use the SYNCHK(D) option on a -SEL control statement. To suppress it during a single execution, specify NOCHK on a -SEL statement.

Note the following

- Certain versions of the compiler accept statements that are direct violations of language specifications. Corrections made to a later release of the compiler can cause these previously accepted statements to cause compilation errors at a future date. The Syntax Checker adheres strictly to the language specifications established for the level; therefore, you might find that it occasionally rejects statements that are acceptable to the compiler.

  You should correct syntax violations of this type immediately to avoid problems at a later date. If it is not possible to correct these errors at once, you can suppress the use of the Syntax Checker for a module or specify that the detection of syntax errors shall not suppress copying of the module to the compilation file.

- In a few situations, the Syntax Checker can detect errors that, although they would not cause compilation errors, create problems during program execution. As an example, assume that you want to place the following three statements into a COBOL module:

  ```
  IF condition
  MOVE INPUT-ACCOUNT TO MASTER-ACCOUNT
  GO TO EDIT.
  PERFORM COMPUTE.
  ```

  If you inadvertently put a period at the end of the first statement, an illogical condition is created that causes serious consequences during execution. The Syntax Checker recognizes that the third statement cannot execute if that period is there and treats the condition as an error.

For further information, see the CCOPY, CCOPYOFF, NOCHK, and SYNCHK(D) options.

Note: This option is for traditional master files only.

TEMP

You can make temporary updates to a module through the TEMP or TEMPS option. These options let you test modifications before they are actually applied to the module. Updates made when either TEMP or TEMPS is in effect are written to the output files (if requested), but are not applied to the master file. Once you test your updates, resubmit them without the TEMP or TEMPS option to apply the changes permanently.

You might find it necessary to apply such a large number of updates to a module that you do not want to have to resubmit them. In this case, you can create a test version of the module. To do this, use the COPY= option to create a duplicate of the module. When you test the updates, you can delete the original module and use the RENAME= option to rename the duplicate so that it has the name of the original module.

TEMP designates that the updates to each module it is specified for are temporary. Sequence numbers of existing records are not affected by updates made while TEMP is in effect. Records that are altered or inserted while TEMP is in effect contain the character string TEMP**** in their sequence number fields. (If the length of the sequence field is less than eight characters long, TEMP**** is adjusted accordingly.)

All updating operations are reflected in the listings and in the copy of the module written to the compilation or punch file. The module on the master file remains unaltered.

Use TEMP on a -OPT or -SEL control statement. This option is not supported in an online environment, where it is flagged as an error.

To override this option, use the PERM or TEMPS option.

Note: The hash count check is bypassed when TEMP is specified.

See also the PERM and TEMPS options.

TEMPS

Updates to each module it is specified for are temporary. Records that are inserted while TEMPS is in effect are assigned sequence numbers with an increment of 1. Existing records are reassigned sequence numbers using the NORESEQ mode of updating.

All updating operations are reflected in the listings and in the copy of the module written to the compilation or punch file. The module on the master file remains unaltered.

Use TEMPS on a -OPT or -SEL control statement. It is not supported online.

To override this option, use the PERM or TEMP option.

Note the following

- The hash count check is bypassed when TEMPS is specified.
- When applying temporary updates, use TEMPS rather than TEMP if the compiler requires sequence numbers.

See also the NORESEQ, PERM, and TEMP options.

UNCON

Processing of each module it is specified for is unconditional (that is, the module is processed regardless of the success or failure of preceding processing).

Use UNCON on a -OPT, -ADD, -SEL, or -DLM control statement. UNCON is supported in an online environment.

To override this option, use the CON option.

For further information, see the CON option.

UNLOCK

Unlocks a locked member. Users with unlimited access authority or the lock owner can unlock a locked member.

A lock owner is the user ID who originally locked the member. Use UNLOC on -ADD and -SEL control statements. LOCK and UNLOCK do not create a new archive level. The last update date and time remains unchanged.

See the *Security Administration Guide* for more information about access authority.

VAR

Invokes the Source-Load Audit Trail facility for the current execution only. During the writing of the compilation file, AllFusion CA-Librarian scans the module source for one or more of the Audit Trail variables listed in the following table and replaces the variables with appropriate data.

If you are accessing a past archiving level of a AllFusion CA-Librarian module, you receive the replacement information that was current as of that level of the module.

You can use a variable more than once in a module.

Use the Source-Load Audit Trail facility to establish a link between your source module library and your load library. When analyzing a dump containing a load module, you can see at once which source program listing to use.

To save coding time, your site can establish a standard Audit Trail variable module to include (through the -INC control statement) in every module.

Note the following:

- Do **not** use the COBOL COPY verb when copying the source load audit trail variables. If you do, the module name of the module you are copying (the copy book name) is put into the ¢MODNAME field instead of the name of the calling module (the primary module name).
- When using the AllFusion CA-Librarian -INC control statement to include the data records of one module in another at compilation time, any SLAT variables contained in the included module reflect information from the primary module (that is, the module that contains the -INC control statement).
- For COBOL II programs and programs compiled using IBM Cobol for z/OS and OS/390 and VM/ESA, insert the SLAT variables after the WORKING-STORAGE SECTION as a single 01 level as shown below.

```
 01  SLAT-VARIABLE-AREA.       PIC X(122) VALUE 'SLAT VARS START:'
-              '¢DATEUPD¢TIMEUPD¢PROGRAMMERNAME¢MODNAME¢LVNO¢UPNO'
-    '¢DATA-SET-NAME-FOR-LIBRARIAN-MASTER-FILESLAT VARS END'.
```

Note: Under both COBOL II and IBM Cobol for z/OS and OS/390 and VM/ESA, there is no guarantee that the separate pieces of an eyecatcher will be in the proper order or even stay together, unless declared in a single VALUE clause. That is why the variables must be declared as a single 01 level when using these compilers.

Use VAR on a -OPT, -ADD, or -SEL control statement.

Audit Trail Variable	Min Length	Will Return	Meaning
¢DATEUPD	8	yy/mm/dd NONEbbbb	Date the module was last permanently updated.
¢TIMEUPD	8	hh:mm:ss hh:mmbb	Time the module was last permanently updated.
¢PROGRAMMERNAME	15	name, left justified, blank filled	Programmer name.
¢MODNAME	8	name, left justified, blank-filled	Module name.
¢LVNO	5	nnnnn NONEb	Current Archiving level number.
¢UPNO	5	nnnnn NONEb	Current number of permanent updates.
¢DATEACS	8	yy/mm/dd NONEbbbb	Date of last access by AllFusion CA-Librarian.

Audit Trail Variable	Min Length	Will Return	Meaning
¢TEMPDAT	8	yy/mm/dd NONEbbbb	The execution date. This appears only if the module is temporarily updated. Otherwise, the value is NONE.
¢DATA-SET-NAME-FOR-THE-LIBRARIAN-MASTER-FILE	44	name, left justified, blank-filled	Disk master file name.

Use VAR on a -OPT, -ADD, or -SEL control statement.

For example, assume a programmer coded the following module statements in a COBOL module named PAROLL70 that resides on a AllFusion CA-Librarian disk master file:

```
01  AUDIT-STAMP              .
    02 FILLER              PIC X(15) VALUE 'AUDIT-STAMP****'.
    02 MODULE-NAME         PIC X(15) VALUE '¢MODNAME '.
    02 DATE-UPDATED        PIC X(15) VALUE '¢DATEUPD '.
    02 PROGRAMMER          PIC X(15) VALUE '¢PROGRAMMERNAME'.
    02 UPDATE-NUMBER       PIC X(15) VALUE '¢UPNO '.
    02 FILLER              PIC X(15) VALUE '****AUDIT-STAMP'.
                             .
                             .
```

If a programmer named Parmley updates the module on November 1, 1996, and requests a compilation, AllFusion CA-Librarian writes the following information to the compilation file:

```
01  AUDIT-STAMP              .
    02 FILLER              PIC X(15) VALUE 'AUDIT-STAMP****'.
    02 MODULE-NAME         PIC X(15) VALUE 'PAROLL70 '.
    02 DATE-UPDATED        PIC X(15) VALUE '96/11/01 '.
    02 PROGRAMMER          PIC X(15) VALUE 'PARMLEY '.
    02 UPDATE-NUMBER       PIC X(15) VALUE '00073 '.
    02 FILLER              PIC X(15) VALUE '****AUDIT-STAMP'.
                             .
                             .
```

To override this option, use the NOVAR option.

For further information, see ARC=, EXEC, and NOVAR options.

VERS=date

Verifies that the specified date (or date and time) is the date (or date and time) of the update most recently applied to the module processed. If so, processing continues normally. If not, an error message is written on the Update Record listing and further processing of the module is bypassed.

Use VERS= when you are not certain whether the listing you are working from reflects the latest version of the module.

Use VERS= on a -SEL or -DLM control statement. With a -SEL control statement, VERS= is supported in an online environment.

date

The date (date and time) of the last update of the module as reported on the Module listing.

Its form is:

- mmdd[hhmm]

  or

- ddmm[hhmm] where mm is the month and dd is the day.

hhmm

The hour and minute (using the 24-hour clock). You must specify the month and day in the order in which they appear on the Module listing. For example, if the date and time are reported on the Module listing as:

```
08/14/94 1805
```

specify:

```
VERS=08141805
```

If you are processing a non-archived module and your listing reports the date (or date and time) of the last update as NONE, specify either VERS=0000 or the date the module was added to the master file. If you are processing an archived module and your listing reports the date (or date and time) as NONE, you must specify the date the module was added to the master file.

XREF

Used on a -OPT control statement to produce a Module Cross-Reference report. Specify the -OPT XREF control statement without specifying the optional -XREF and -XREFR commands to generate a Module Cross-Reference Report listing all the modules of a master file containing all the occurrences of the three types. -OPT XREF

- For wide record, -OPT XREF produces a listing of -INC occurrences only.
- For details on this option, see the -XREF and -XREFR commands in Chapter 3.

Chapter 5

Compiling a Program (z/OS and OS/390)

The facilities of AllFusion CA-Librarian let you compile, link edit, and execute any program stored as an AllFusion CA-Librarian module.

Note: In addition to using the batch AllFusion CA-Librarian program to extract a member from a master file to compile it, you can use the AllFusion CA-Librarian Access Method (LIB/AM) to compile programs *directly* from a master file. See the *User Guide* for more information on LIB/AM.

To compile a module, use the EXEC option on a -OPT, -ADD, or -SEL statement. AllFusion CA-Librarian places each module you specified the EXEC option for into the compilation file in 80-byte card-image format. At the conclusion of execution, you can process the compilation file as a data set for the appropriate compiler to read in a subsequent step or job, or as a complete job stream, containing all necessary job control statements and source modules for an operating system reader to read.

Your site management determines which one of these methods you use. Check with your system programmer for conventions established at your site.

This chapter contains examples of control stream organization and JCL requirements for illustrative purposes. They might not conform to the conventions established at your site. Your system programmer can supply exact JCL specifications and descriptions of z/OS and OS/390 cataloged procedures related to the use of AllFusion CA-Librarian.

Compilation File as Input to a Compiler

If your site conventions specify that the compilation file is passed to a compiler, you must supply all JCL necessary for updating and compiling and, optionally, for link editing and testing for each execution. AllFusion CA-Librarian returns condition codes that you can test to determine whether errors occurred during updating. For additional information on AllFusion CA-Librarian condition codes, see the appropriate *System Services Guide*.

The JCL for executing AllFusion CA-Librarian is usually incorporated into a z/OS and OS/390 *cataloged procedure*. This procedure defines the compilation file as either a sequential or partitioned data set that is passed to a subsequent job step. Your system programmer can provide JCL specifications applicable to your site.

If the compilation file is a sequential data set and you request that several modules be written to it during a single execution of AllFusion CA-Librarian, the modules are written to it one after another. In such a case, you must be certain that the compiler can accept batched input. If not, you must run a separate job (or at least a separate sequence of steps) for each module.

If the compilation file is a partitioned data set, AllFusion CA-Librarian writes each module that you select for compilation as a member of that PDS. Each of those members can be read during a separate execution of a compiler. AllFusion CA-Librarian uses the module name as the member name, unless the module name begins with a numeric character, in which case AllFusion CA-Librarian prefixes the letter Z to the first seven characters of the module name to create the member name (Z*xxxxxxx*).

You can also specify a member name through the MEMNAME= parameter of the -JCL or -JCLT statement. If the member name already exists in the PDS, AllFusion CA-Librarian does not normally stow the member, but issues an error message. To direct AllFusion CA-Librarian to replace a like-named member, use the EXEC(R) option on the -OPT, -ADD, or -SEL statement.

The compilation file is always a partitioned data set if you allocate directory blocks in the SPACE parameter of the DD statement creating it. You need never code the DSORG DCB subparameter.

The following examples illustrate JCL and control statements to execute AllFusion CA-Librarian and subsequent processors.

Example 1

To update, compile, link edit, and execute a module using a sequential data set as a compilation file, your input would be as follows:

```
//LIBUCLG JOB 70010,CA.JAEGER,CLASS=0
//STEP1 EXEC PGM=librarian,PARM='NRJS,NJTA'
//OSJOB DD  DSN=&&TEMP,UNIT=DISK,SPACE=(TRK,(5,1)),
//          DISP=(PASS)
//SYSIN DD *
-SEL  program,rvjl,EXEC
      .
      .     (AllFusion CA-Librarian updating statements)
      .
-END
//STEP EXEC ASMFCLG
//ASM.SYSIN  DD DSN=&&TEMP,DISP=(OLD,DELETE)
//
```

Example 2

To update and compile two modules using a non-temporary partitioned data set as a compilation file, your input would be as follows:

```
//LIBUPDTE JOB 70010,CA.STROUSE,CLASS=0
//STEP1 EXEC PGM=librarian,PARM='NRJS,NJTA'
//OSJOB DD DSN=your.pds.name,UNIT=DISK,
//         SPACE=(TRK,(50,20,3)),DISP=(,PASS)
//SYSIN DD *
-SEL modula,wkrt,EXEC
     .
     .     (AllFusion CA-Librarian updating statements)
     .
-SEL modulb,wjdw,EXEC
     .
     .     (AllFusion CA-Librarian Updating statements)
     .
-END
//STEP2 EXEC ASMFC
//ASM.SYSIN DD DSN=&&TEMP(modula),DISP=OLD
//STEP3 EXEC ASMFC
//ASM.SYSIN DD DSN=&&TEMP(modulb),DISP=OLD
//
```

Compilation File as a Job Stream

AllFusion CA-Librarian can create a job stream on the compilation file, normally consisting of a separate job for each module to compile. At the conclusion of AllFusion CA-Librarian execution, an operating system reader can read the compilation file to enter the jobs into the system work queues.

You can supply JCL (for compilation, and so on) as an integral part of a module, or you can direct AllFusion CA-Librarian to generate JCL for you.

User-supplied JCL

To supply JCL statements for a module directly, insert them into the module itself. Since these JCL statements are part of the module, they automatically become part of the job stream when the module is written to the compilation file. The first few records of the module are a JOB statement and other JCL needed for compilation. You can put JCL for link editing and testing at the end of the module. You can include any data needed for testing at the appropriate point in the module or following a -DATA control statement.

Generated JCL

AllFusion CA-Librarian provides a set of parameters that, when specified on a -JCL or -JCLT control statement, direct AllFusion CA-Librarian to generate JCL. Some of these parameters might have been established as initialization defaults at your site. You can add to or respecify these parameters (through the -JCL control statement) when adding or updating a module. They are stored with the module. AllFusion CA-Librarian accesses them whenever you use the EXEC option. Parameters respecified for an archived module are archived in the same way as are data records.

To alter parameters during a single execution only, use the -JCLT control statement.

The following example shows the basic structure of a job AllFusion CA-Librarian creates.

```
/*PRIORITY    hprty
//jobname JOB acct,programmer-name,CLASS=class,
//            REGION=region,PRTY=prty,TIME=time,
//            MSGLEVEL=1,MSGCLASS=msgclass
//UPDATEXX EXEC proc
//step ddname DD DSN=dsn(memname),DISP=disp
//step sysin  DD P
    .
    .
    .
/*
//LKED.SYSLMOD DD DSN=dsn(memname),DISP=disp
```

Use the -JCL (or -JCLT) control statement to assign values to the terms that appear in lower case in the example above.

The executed procedure determines the main function of the job. (The first step of the procedure usually executes the appropriate compiler. Subsequent steps in the procedure can provide for link editing and testing the program.) You supply the name of the procedure in the PROC= parameter.

-JCL Control Statement

Parameters on the -JCL control statement control the automatic generation of JCL for a module.

The format is:

```
-JCL parameter[,parameter...]
```

parameter

Consists of a keyword, an equal sign (=), and a value. If the value includes an embedded comma or blank, you must enclose the entire value in slashes. A list of the parameters follows.

You can use as many -JCL control statements as are necessary to supply the parameters you need. Place -JCL control statements in the control deck following a -ADD or -SEL control statement and preceding any documentary control statements.

To delete a previously defined parameter, specify the keyword and its equal sign, but do not specify a value.

-JCLT Control Statement

The -JCLT control statement modifies one or more of the parameters that you can specify on the -JCL control statement during the current execution only. The parameters stored on the master file are not changed. You can use both -JCL and -JCLT control statements for a single module. -JCL control statements must precede -JCLT control statements.

Except for the command itself, the -JCLT control statement has the same format as the -JCL control statement.

-JCL and -JCLT Parameters

You can specify the following parameters on the -JCL and -JCLT control statements.

ACCT=*acct*
Generates the account number on the JOB statement. A maximum of 55 characters is permitted. If you include commas or blanks in the account number, you must enclose the entire value in slashes. No more than two consecutive blanks can appear in the account number. For example:

```
ACCT=/(456,23,55)/
```

CLASS=*class*
Generates the CLASS= parameter on the JOB statement.

DDNAME=*ddname*
The name of the DD statement defining the library where the output of the compilation or link-edit step is stowed. If you specify DDNAME=SYSLMOD, a stepname of LKED is automatically generated and the DD statement follows the module's data records. If you specify any other value for ddname, the DD statement precedes the module's data records. When you specify this parameter, you must also specify the DSN=parameter.

DESC=*/desc/*
A 1- to 30-character module description. The function of this parameter is identical to the -DESC control statement. This parameter does not generate JCL.

DISP=[OLD | SHR | NONE]
The DISP= parameter used on the DD statement the DDNAME= parameter generates. This parameter is generated only if you specified the DDNAME= parameter. If you specify DISP=NONE, the DISP= parameter is omitted from the DD statement. If you omit this parameter, DISP=OLD is used.

DSN=*name*
The fully qualified name of the data set where the object module or load module is stowed. The data set must already exist and must be cataloged. For example:

```
DSN=PAYROLL.TEST.RELOC
```

Note: This parameter is generated only if you specified the DDNAME= parameter.

HPRTY=*hprty*
Generates a HASP /*PRIORITY statement preceding the JOB statement and specifying the indicated priority.

JOBNAME=[*name*|$]
Generates the job name of the JOB statement. You can assign any name that is acceptable to the operating system, except $. If you omit this parameter, the module name is used as the job name. To suppress generation of the JOB statement, specify JOBNAME='$' (all other JCL, except the HASP /*PRIORITY statement, are generated).

LANG=*lang*
The 1- to 30-character language code identifying the language in which the module is written. The function of this parameter is identical to the -LANG control statement. This parameter does not generate JCL.

MEMNAME=*memname*
The member name the object module or load module is stowed under (for example, in the data set the DSN= parameter specifies). If you omit this parameter, the module name is used as the member name. You can also use this parameter if a partitioned data set is used as the compilation file and no JCL is generated. In this case, it specifies the member name the copy of the module written to the compilation file is stowed under.

MSG=*msgclass*
Generates the MSGCLASS= parameter on the JOB statement.

PGMR=*programmer-name*
Generates the 1- to 15-character programmer name on the JOB statement. Blanks and commas are not valid characters. The function of this parameter is identical to the -PGMR control statement.

PROC=[$NOJCL|*procname*]
The operand portion of the first or only record of the EXEC statement. For example:

```
PROC=COBFCLG
```

To suppress all JCL generation for the module, specify PROC=$NOJCL.

In addition to (or instead of) the procedure name, you can specify any other values that are permitted on the EXEC statement. Specify all values exactly as they should appear on the EXEC statement. If you include commas or blanks, you must enclose the entire value field in slashes. For example:

```
PROC=/ASMFC,PARM='LOAD,DECK'/
```

If you must specify more than 55 characters of operands to include on the EXEC statement, use the PROC2= parameter to generate a single continuation record containing up to 45 additional characters. End the PROC statement with a comma and code the PROC2 statement as you would a PROC statement. If the ending comma is omitted from the PROC statement, the following PROC2 statement is ignored.

PRTY=*priority*
Generates the PRTY= parameter on the JOB statement.

REGION=*region*
Generates the REGION= parameter on the JOB statement. The maximum length of region is 11 characters. For example:

```
REGION=80K
REGION=/(64K,768K)/
```

STEP=*step*
The name of the compilation step in the procedure executed (for example, ASM, COB, FORT, and so on).

SYSIN=*sysin*
The name of a DD statement the compiler uses to read the module to process. The name can be no more than six characters in length. If this parameter is omitted, a ddname of SYSIN is generated.

TIME=*time*

Generates the TIME= parameter on the JOB statement. The maximum length of time is seven characters. For example:

```
   TIME=10
   TIME=/(5,30,)/
//TEST JOB 70010,CA.SOLOMON,CLASS=x              }
// EXEC  PGM=CA-Librarian                        }
//OSJOB  DD SYSOUT=(A,INTRDR),DCB=(LRECL=80,     }
//          BLKSIZE=80)                         }
//SYSIN DD *                                    }
-OPT NOSEP                                      } AllFusion
-ADD PAYEDIT,SEQ=COBOL,EXEC                     } CA-Librarian
-JCL PROC=COBFCL,STEP=COB,PGMR=MILLER           } Stream
-JCL ACCT=/(70010)/,DDNAME=SYSLMOD              }
-JCL DSN=PAYROLL.TEST                           }
          .                                     }
          .    (module data records)            }
          .                                     }
-DESC PAYROLL EDIT ROUTINE                      } Control
-END                                            }
/*                                              }


//PAYEDIT JOB (70010),                        C }
//            MILLER,                         C }
//            MSGLEVEL=1                        }
//UPDATEXX EXEC COBFCL                          }
//COB.SYSIN DD *                                } Resulting
         .                                      } Job
         . (module data records)                } Stream
         .                                      }
/*                                              }
//LKED.SYSLMOD DD DISP=OLD,                   C }
//             DSN=PAYROLL.TEST(PAYEDIT)          }
```

Example of a -JCL Control Statement

To add a COBOL module to a master file and compile and link edit it using the IBM-supplied procedure COBFCL and place the resulting load module in the data set PAYROLL.TEST, your input would be as shown in the previous example.

Chapter

6

Compiling a Program (VSE/ESA)

To compile a module, use the EXEC option on a -OPT, -ADD, or -SEL statement. AllFusion CA-Librarian places each module you specify the EXEC option for into the compilation file in 80-byte card-image format. At the conclusion of AllFusion CA-Librarian execution, you can process the compilation file as a SYSIPT file for the appropriate compiler in a subsequent step or job to read or, as a complete job stream (for example, a SYSIN file), containing all necessary job control statements and source modules for the operating system to read.

Your site management determines which of these methods you use. Check with your system programmer for conventions established at your site.

Note: Examples of control stream organization and job control statement requirements are provided for illustrative purposes. They might not conform to conventions established at your site. Your system programmer can supply exact specifications.

Compilation File as Input to a Compiler

If your site conventions specify that the compilation file is read as a SYSIPT file by a compiler, you must supply all job control statements necessary for updating and compiling and, optionally, for link editing and testing for each execution. The final record of the module must be an end-of-data-file statement /*.

If the compilation file is a sequential file and you request that several modules be written to it during a single execution of AllFusion CA-Librarian, the modules are written to it one after another. In such a case, you must be certain that the compiler can accept batched input. If not, you must run a separate job or a separate sequence of steps for each module.

The following example illustrates how to update and compile a module using a SYSIPT file for compilation.

```
// JOB ASSEMBLY USING SYSIPT FILE
// DLBL MASTER,'librarian.master.file-id',,DA
// EXTENT SYS__4,volser,1,_,reltrk,ntrks
// ASSGN SYS__4,DISK,VOL=volser,SHR
// DLBL JOBSTR,'jobstr.one.file-id',,SD
// EXTENT SYS__8,volser,1,_,reltrk,ntrks
// ASSGN SYS__8,DISK,VOL=volser,SHR
// EXEC CA-Librarian,SIZE=AUTO
-SEL ALPHA,XWRK,EXEC
-END
/*
// DLBL IJSYSIN,'jobstr.one.file-id',,SD
// EXTENT SYSIPT,volser
ASSGN SYSIPT,DISK,VOL=volser,SHR
// EXEC ASSEMBLY
/*
CLOSE SYSIPT,SYSRDR
/&
```

Compilation File as a Job Stream

AllFusion CA-Librarian can create a job stream on the compilation file, consisting of a separate job for each module to compile. At the conclusion of AllFusion CA-Librarian execution, the operating system can read the SYSIN file.

You can supply job control statements (for compilation, and so on) as an integral part of a module or you can direct AllFusion CA-Librarian to generate job control statements for you.

User-supplied JCL

Job control statements for a module are supplied directly by inserting them into the module itself. Because these job control statements are part of the module, they automatically become a part of the job stream when the module is written to the compilation file.

You can include any data needed for testing the module at the appropriate point in the module or after a -DATA control statement (see Chapter 3 for information about the -DATA control statement).

The following example adds and compiles a module with user-supplied job control statements in it.

```
// JOB COMPILE USING SYS008 AS INPUT
// DLBL JOBSTR,'JOBSTREAM FILE',0,SD
// EXTENT SYS008,222222,1,0,1220,600
// ASSGN SYS008,X'131'
// DLBL MASTER,'LIBRARIAN MASTER FILE',99/365,DA
// EXTENT SYS004,333333,1,0,2000,100
// ASSGN SYS004,X'132'
// EXEC librarian,SIZE=AUTO
-OPT EXEC
-ADD module-name,SEQ=COBOL
/% JOB COMPILE                                  A
/% OPTION CATAL
% PHASE MODULE,*                                B
/% EXEC FCOBOL
  .
  . (SOURCE MODULE RECORDS)
  .
/% LBLTYP TAPE
/% EXEC LNKED
/:
/;
-END ASSGN SYSIN,X'00C'                         C
/*
// DLBL IJSYSIN,222222,1,0,1220,600
ASSGN SYSIN,X'131'
/*
/&
```

A See the *User Guide* for information on using AllFusion CA-Librarian conversion characters (/:, /;, / $$, and =).

B The percent sign (%) in column 1 is required to force the sequence number to columns 73 through 80 (see the *User Guide*). Placing a percent sign in column 80 accomplishes the same.

C See Chapter 3 for information on using the -END control statement to write a final record to the compilation file.

Generated JCL

AllFusion CA-Librarian provides a set of parameters that, when specified on a -JCL or -JCLT control statement, direct AllFusion CA-Librarian to generate job control statements. Some of these parameters might have been established as initialization defaults. You can add to or respecify these parameters (through the -JCL control statement) when adding or updating a module. They are stored with the module and AllFusion CA-Librarian accesses them whenever you use the EXEC option. Parameters respecified for an archived module are archived in the same way as data records. To alter parameters during a single execution only, use the -JCLT control statement.

-JCL Control Statement

Parameters on the -JCL control statement control the automatic generation of job control statements for a module.

The format is:

```
-JCL parameter[,parameter...]
```

parameter
Consists of a keyword, an equal sign (=), and a value. If the value includes an embedded comma or blank, you must enclose the entire value in slashes.

You can use as many -JCL control statements as are necessary to supply the parameters you need. Place -JCL control statements in the control deck following a -ADD or -SEL control statement and preceding any documentary control statements.

To delete a previously defined parameter, specify the keyword and its equal sign, but do not specify a value.

-JCLT Control Statement

The -JCLT control statement modifies one or more of the parameters that you can specify on the -JCL control statement during the current execution only. The parameters stored on the master file are not changed. You can use both -JCL and -JCLT control statements for a single module. -JCL control statements must precede -JCLT control statements.

Except for the command itself, the -JCLT control statement has the same format as the -JCL control statement.

-JCL and -JCLT Parameters

You can specify the following parameters on the -JCL and -JCLT control statements.

TYPE=[NONE|COMP|COMPGO|COMPCAT|COMPCATGO|PARAM]
Defines the basic structure of the job to create. The allowable values and the types of jobs each creates are shown below. Terms that appear in lower case represent user-supplied values.

NONE
Suppresses generation of job control statements for the module. It is the default for this parameter. Specify TYPE=NONE to process the module as a SYSIPT file.

COMP

Generates job control statements for compilation. The structure of the job created is:

```
// JOB jobname,acct
// OPTION options
   CATALR name
// EXEC language
   .
   . (module data records)
   .
/*
/&
```

COMPGO

Generates job control statements for compilation, link editing, and execution. The structure of the job created is:

```
// JOB jobname,acct
// OPTION options
   ACTION action
   CATALR name
   PHASE phasename
// EXEC language
   .
   . (module data records)
   .
/*
   ENTRY entrypoint
// LBLTYP lbltyp
// EXEC LNKEDT
// EXEC
/&
```

COMPCAT

Generates job control statements for compilation, link editing, and cataloging the program in a core image library. The structure of the job created is:

```
// JOB jobname,acct
// OPTION options
CATALR name
PHASE phasename
// EXEC language
.
. (Module data records)
.
/P
ENTRY entrypoint
// LBLTYP lbltyp
// EXEC LNKEDT
/&
```

COMPCATGO

Generates job control statements for compilation, link editing, cataloging, and execution. The structure of the job created is:

```
// JOB jobname,acct
// OPTION options
   CATALR name
   ACTION action
   PHASE phasename
// EXEC language
   .
   . (module data records)
   .
/*
   ENTRY entrypoint
// LBLTYP lbltyp
// EXEC LNKEDT
// EXEC phasename
/&
```

PARAM

Generates JOB, OPTION, CATALR, ACTION, PHASE, EXEC, ENTRY, and LBLTYP statements (if you specified corresponding parameters on a -JCL or -JCLT control statement) and generates an end-of-data-file statement (/*) following the module's data records. An end-of-job statement (/&) is also generated at the end of the job, unless you specify SLAMP=NO. Specify TYPE=PARAM if you included some of the job control statements needed to process the module in the module itself.

ACCT=*acct*

Generates up to 16 characters of accounting information to include on the JOB statement, immediately following the job name. Blanks are not valid characters.

ACTION=*action*

Generates a linkage editor ACTION statement for each of the actions you specify. Blanks are not valid characters. When specifying more than one action, separate them from each other with commas and enclose the entire value in slashes. The maximum length of the value is 30 characters. For example:

```
ACTION=/F1,CLEAR/
```

CATALR=*name*

Generates a CATALR statement containing the one- to eight-character name that you specify.

COMPOPT=*option*

Specifies the options to place on the EXEC statement that invokes the compiler (up to 15 characters in length). For example:

```
COMPOPT=SIZE=50K
```

If you include commas or blanks, you must enclose the entire value in slashes.

/DESC=/desc/
Specifies a 1- to 30-character module description. The function of this parameter is identical with that of the -DESC control statement. This parameter does not generate a job control statement.

ENTRY=entrypt
Generates a linkage editor ENTRY statement containing the entry point name that you specify.

EXEC=*language*
Generates an EXEC statement to execute the named compiler (up to eight characters in length). You cannot specify this parameter if you specified the PROC= parameter.

GOOPT=*option*
Specifies the options to place on the EXEC statement that invokes the compiled program (up to 15 characters in length). If you include commas or blanks, you must enclose the entire value in slashes. For example:

```
GOOPT=/SIZE=50K,REAL/
```

JOBNAME=[*jobname*|NONE]
Specifies the job name to place on the JOB statement. If you omit this parameter, the module name is used. To suppress generation of a JOB statement, specify JOBNAME=NONE.

LANG=*lan*
The one- to three-character code identifying the language in which the module is written. The function of this parameter is identical to the -LANG control statement. This parameter does not generate a job control statement.

LBLTYP=*lbltyp*
Generates a LBLTYP statement containing the value that you specify. The maximum length of the value is nine characters. For example:

```
LBLTYP=NSD(5)
```

LINKOPT=*option*
Specifies the options to place on the EXEC statement that invokes the linkage editor. If you include commas or blanks, you must enclose the entire value in slashes. The maximum length of the value is 15 characters. For example:

```
LINKOPT=REAL
```

OPTION=*option*
Generates an OPTION statement containing the options that you specify (up to 60 characters in length). If you include commas or blanks, you must enclose the entire value in slashes. For example:

```
OPTION=/LINK,DUMP/
```

PGMR=*programmer-name*
The 1- to 15-character name identifying the programmer who is responsible for the module. The function of this parameter is identical to the -PGMR control statement. This parameter does not generate a job control statement.

PHASE=*phasename*
Generates a linkage editor PHASE statement containing the phase name and options that you specify (to a maximum length of 44 characters). If you include commas or blanks, you must enclose the entire value in slashes. For example:

```
PHASE=/EDITCOMP,P/
```

PROC=*procname*
Generates an EXEC statement that invokes the procedure whose name you specify. You cannot specify this parameter if you specified the EXEC= parameter. If you use the PROC= parameter, you must specify the TYPE= parameter as TYPE=PARAM.

SLAMP=NO
Prevents an end-of-job statement (/&) from generating.

Examples of -JCL Parameter Use

A module named MATINV is added to the master file as follows.

```
// JOB ADDMAT
// EXEC librarian,SIZE=AUTO
-ADD MATINV
-DESC INVENTORY MAINTENANCE
-JCL TYPE=COMP,EXEC=COBOL,PGMR=SMITH
     .
     .      (module data records)
     .
-END
/*
/&
Whenever the EXEC option is specified for this module, the following job is
written to the compilation file:
// JOB MATINV
// OPTION DECK
// EXEC COBOL
     .
     .      (module data records)
     .
/*
/&
```

Compile, Link, Go With Test Data, ASSGN, and UPSI Statements

A module named PAYEDIT is added to the master file as shown below.

```
// JOB ADDPAY
// EXEC librarian,SIZE=AUTO
-ADD PAYEDIT
-DESC EDIT DAILY TIME SHEETS
-JCL PGMR=BERG,TYPE=COMPGO,EXEC=COBOL
-JCL PHASE=/PEDTPROG,*/,OPTION=LINK,DUMP/
     .
     .      (module data records for PAYEDIT)
     .
-END
/*
/&
```

To update the module, compile it, and test it (including ASSGN and UPSI statements and appropriate test data), submit the job shown below.

```
// JOB UPDPAY
// EXEC librarian,SIZE=AUTO
-SEL PAYEDIT,WXLT,EXEC
-INS 70
          .
     .      (new module data records)
     .
-DATA
/% ASSGN SYS004,X'180'
/% ASSGN SYS005,X'181'
/% UPSI 00000001
      .
      .     (test data for module PAYEDIT)
      .
-EMOD
-END
/*
/&
```

Whenever you specify the EXEC option for this module, the job shown below is written to the compilation file:

```
// JOB PAYEDIT
// OPTION LINK,DUMP
PHASE PEDTPROG,P
// EXEC COBOL
     .
     .      (module data records for PAYEDIT)
     .
/*
// EXEC LNKEDT
// ASSGN SYS004,X'180'
// ASSGN SYS005,X'181'
// UPSI 00000001
// EXEC PEDTPROG
     .
     .      (module data records for PAYEDIT)
     .
/*
/&
```

Note: In the example above, the ASSGN and UPSI statements that follow the -DATA control statement are automatically placed in the job in their proper location. AllFusion CA-Librarian generated the /* and /& at the end of the test data.

Inserting JCL Before the First EXEC

To include job control statements (for example, // ASSGN, // DLBL, and // EXTENT) before the EXEC statement that invokes the compiler, place them at the beginning of a module. This lets you specify, for example, the library the COBOL compiler uses to expand COPY statements. AllFusion CA-Librarian places job control statements found at the beginning of the module ahead of the EXEC statement that invokes the compiler when writing the module to the compilation file. The sequence numbers of these job control statements are stored in columns 73-80.

Generating a Final Record

You can use the -END control statement to generate a final record to write to the compilation file. Place the data to copy to columns 1-67 of this final record in columns 6-72 of the -END control statement.

The following example illustrates the use of the -END statement:

```
// JOB FINALJS
// EXEC librarian,SIZE=AUTO
-OPT EXEC
-SEL PAYEDIT,NXLT,EXEC
-SEL MATINV,WXRJ,EXEC
-END CLOSE SYSIN,X'00C'
/*
/&
This input results in the following job stream:
// JOB PAYEDIT
// OPTION LINK,DUMP
        .
        .         (module data records)
        .
// JOB MATINV
// OPTION DECK
        .
        .         (module data records)
        .
/&
CLOSE SYSIN,X'00C'
```

Appendix

A

Sample Listings

AllFusion CA-Librarian produces listings that document all operations performed.

Listings of Operations Performed

The listings are:

Update Record Listing

A complete record of the updating operations performed during an AllFusion CA-Librarian execution. This listing includes all AllFusion CA-Librarian control statements, all inserted or deleted data records, and all documentary statements processed.

The Update Record listing is produced automatically.

You can reduce the quantity of paper used in the Update Record listing by specifying the NOSEP or NOPR options.

Summary of Activity Listing

A one-line summary of the results of the processing of each of the first 100 modules processed during an AllFusion CA-Librarian execution.

The Summary of Activity listing is, of course, a summary. For complete information on the processing of all modules, look at the Update Record listing.

The Summary of Activity listing is also produced automatically.

Module Listing

Produced only when you specify the LIST option. This listing shows:

- The module control information (that is, module description, date last updated, and so on)
- History records.
- All the data records in the module.
- Date stamps of all date stamped records.

To request that AllFusion CA-Librarian abbreviate the Module listing by reporting only the module control information and history records, specify the LISTH option.

Master File Index Listing
Produced when you specify the INDEX option. This listing provides a one-line summary of information about each module on the master file. Only one index listing can be provided during a single execution of AllFusion CA-Librarian.

Programmer Index Listing
Produced when you specify the INDEX(pgmr) option. This listing provides a one-line summary of information about each module for which the specified programmer is responsible.

Master File Utility Index Listing
Produced when you specify -INDEX following an -OPT utility control statement. This listing provides a one-line summary of information about each module on the master file. Only one index listing can be provided during a single execution of AllFusion CA-Librarian.

Programmer Utility Index Listing
Produced when you specify -INDEX(pgmr) following an -OPT utility control statement. This listing provides a one-line summary of information about each module for which the specified programmer is responsible.

You can produce only one index listing during a single AllFusion CA-Librarian execution.

All AllFusion CA-Librarian listings have the date and time of the update printed on them. To determine whether the listing you are working from documents the current version of the module, use the VERS= option.

Update Record Listing

```
RUN NO. 55     DATE=mm/dd/yy  TIME=1001              UPDATE RECORD             LIBR.DEV.MAST             PAGE 2

-SEL COBOLTST,EXEC,LIST
-HST UPDATE TO CORRECT TYPOS.
-HST AND TO DEFINE IN-COUNT
    -REP 30
         000030 ENVIRONMENT DIVISION.

                              THE FOLLOWING RECORD(S) HAVE BEEN REPLACED BY THE ABOVE RECORD(S):
                              000030 ENVIRN DIVISION.
    -INS 120
         000130 77 IN-COUNT      PIC 999.

                              THE ABOVE RECORD(S) HAVE BEEN INSERTED AFTER THE FOLLOWING RECORD:
                              000120 WORKING-STORAGE SECTION.

MODULE COBOLTST SUCCESSFULLY PROCESSED AT 10.01.36. UPDATE NO.-    7.  LEVEL NO.-  7
```

Summary of Activity Listing

```
RUN NO. 55     DATE=mm/dd/yy  TIME=1001              SUMMARY OF ACTIVITY       LIBR.DEV.MAST             PAGE 1

-SEL FOR MODULE 'COBOLTST' WAS SUCCESSFUL.     MODULE IS TO BE COMPILED.
```

Listing of Module 1

```
RUN NO. 55     DATE=mm/dd/yy  TIME=1009              LISTING OF MODULE         COBOLTST                  PAGE 1
DESCRIPTION           COBOL TEST PROGRAM

MASTER FILE           LIBR.DEV.MAST
ADDED TO MASTER       07/03/96
LAST DATE COPIED      NONE
LAST DATE UPDATED     mm/dd/yy 100136
ARCHIVING STATUS      ACTIVATED
NUMBER OF LEVELS      8
CURRENT LEVEL NO.     7
NUMBER OF RECORDS    24
NUMBER OF UPDATES     7
NUMBER OF ACCESSES   16
SEQUENCE PARAMS       01/6/0010/0010 - RESEQ
COMPRESS STATUS       FULL
COPYDD STATUS         NOT ACTIVATED
COBOL COPY STATUS     NOT ACTIVATED
MODULE STATUS         TEST

PASSWORD              GCFG
PROGRAMMER            SMITH
LANGUAGE              COB
PROC PARAMETER        $NOJCL

HISTORY RECORD(S) FOR THIS MODULE:
  mm/dd/yy          UPDATE TO CORRECT TYPOS
                    AND TO DEFINE IN-COUNT

000010 ID DIVISION.
000020 PROGRAM-ID. COBOLTST.
000030 ENVIRONMENT DIVISION.                                          03/14/97
000040 INPUT-OUTPUT SECTION.
000050 FILE CONTROL.
000060     SELECT INFILE PASSING TO UT-S-IN.
000070 DATA DIVISION.
000080 FILE SECTION.
000090 FD INFILE
000100    DATA RECORD IS INREC.
-INC INREC                                              00000110
000120 WORKING-STORAGE SECTION.
000130 77 IN-COUNT      PIC 999.                                     03/14/97
000140 77 IN-INDEX      PIC 999.
000150 77 IN-ID         PIC XXXXX.
000160 PROCEDURE DIVISION.
000170     ACCEPT IN-COUNT FROM SYSIN.
000180     OPEN INPUT INFILE.
000190     PERFORM IN-COPY VARYING IN-INDEX FROM 1 BY 1 UNTIL
000200        IN-INDEX EQUAL IN-COUNT. STOP RUN.
000210 IN-COPY.
000220     READ INFILE.
000230     MOVE INFILE-ID TO IN-ID.
000240     DISPLAY IN-ID UPON SYSOUT.
```

Listing of Module 2

```
RUN NO. 55     DATE=mm/dd/yy  TIME=1009             LISTING OF MODULE         COBOLTST                 PAGE 1
DESCRIPTION         COBOL TEST PROGRAM

MASTER FILE         LIBR.DEV.MAST
ADDED TO MASTER     07/03/96
LAST DATE COPIED    NONE
LAST DATE UPDATED   mm/dd/yy 100136
ARCHIVING STATUS    ACTIVATED
NUMBER OF LEVELS     8
CURRENT LEVEL NO.    7
NUMBER OF RECORDS   24
NUMBER OF UPDATES    7
NUMBER OF ACCESSES  16
SEQUENCE PARAMS     01/6/0010/0010 - RESEQ
COMPRESS STATUS     FULL
COPYDD STATUS       NOT ACTIVATED
COBOL COPY STATUS   NOT ACTIVATED
MODULE STATUS       TEST

PASSWORD            GCFG
PROGRAMMER          SMITH
LANGUAGE            COB
PROC PARAMETER      $NOJCL

HISTORY RECORD(S) FOR THIS MODULE:
  06/14/96          UPDATE TO CORRECT TYPOS
                    AND TO DEFINE IN-COUNT
```

Master File Index Listing 1

```
RUN NO. 109    DATE=mm/dd/yy  TIME=1022             MASTER FILE INDEX         LIBR.DEV.MAST            PAGE 1

MODULE   PSWD   MODULE DESCRIPTION               ADDED        UPDATED   RECORDS     LANG     PROC   PROGRAMMER

BSYSJ030 RLGX   SYSTEM B ROUTINE               01/12/97    01/12/97 115115   60     ASM     $NOJCL  SMITH

CSYSJ100 TDSL   SYSTEM C ROUTINE               02/22/97    NOT UPDATED        7     ASM     $NOJCL  BROWN

TRACMOD1 TDSV   ACCOUNT TRACS SYSTEM           03/30/97    NOT UPDATED        6     COB     $NOJCL  JONES

TRACMOD2 BZTS   ACCOUNT TRACS SYSTEM           09/22/96    02/19/97 140651 1080     COB     $NOJCL  SMITH

VARIABLE JRPK   SLAT VARIABLE MODULE           10/06/96    01/15/97 132927   21     ASM     $NOJCL  SMITH

THE TOTAL NUMBER OF RECORDS ON THE MASTER FILE.....1174
THE TOTAL NUMBER OF MODULES ON THE MASTER FILE.....5
THE TOTAL NUMBER OF ALLOCATED TRACKS.....35
THE TOTAL NUMBER OF UNUSED TRACKS.....32
THE TOTAL NUMBER OF USED TRACKS.....3
```

Note: The compression level of a module is reported on the records field of the Master File Index. A -U appears after the number of records for an uncompressed module. A -P appears for a partially compressed module. If there is no -P or -U, it is a fully compressed module.

Master File Index Listing 2

```
RUN NO. 110    DATE=mm/dd/yy  TIME=1022              MASTER FILE INDEX         LIBR.DEV.MAST             PAGE 1

MODULE   PSWD   MODULE DESCRIPTION                ADDED        UPDATED   RECORDS     LANG     PROC   SEQUENCE PARAM

BSYSJ030 RLGX   SYSTEM B ROUTINE               01/12/97  01/12/97 115115      60      ASM    $NOJCL  72/8/0010/0010

TRACMOD2 BZTS   ACCOUNT TRACS SYSTEM           09/22/96  02/19/97 140651    1080      COB    $NOJCL  01/8/0010/0010

VARIABLE JRPK   SLAT VARIABLE MODULE           10/06/96  01/15/97 132927      21      ASM    $NOJCL  72/8/0010/0010

THE TOTAL NUMBER OF MODULES ON THE MASTER FILE.............3
```

Note: The output of the utility versions of these index listings is the same as the ouput of the non-utility versions.

Index

*

/

¢

0

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Y