SLIM

Source Library Manager

Version 1.1.0

Issued October 10, 2024

Table of Contents

Table of Figures

Summary of Changes

V1.1.0

- Created new module SLIMDDN to replaced the hardcoded DDN table.
- Updated the CLONE command to create the clone without direct writes to the Archive.
- Added code to insert NOTES into cloned members.
- Modified code to insert NOTES as well as user NOTEs into all “unloads”.

Introduction

SLIM is an easy to use tool for storing projects. It is an “add on” to the ARCHIVER package to simplify the management of software items and for distribution of software. ARCHIVER is included in the TK4 and TK5 distributions. It can be found on CBT File 147 at cbttape.org.

SLIM (***S***ource ***Li***brary ***M***anager) was originally designed as a means of storing production source code and documenting relationships between projects. I view it as a “project dictionary” similar to the way relational databases have a “data dictionary”.

A project consists of one or more modules. A module is a best defined a unit of source code. For example, an assembler macro, a COBOL copy book or an entire source program are each considered modules in SLIM. In addition, source listings and copies of load modules may be added into a SLIM archive as modules.

ARCHIVER uses VSAM to store the items in an encrypted and compressed format. It is not easy to change members except through ARCHIVER or SLIM software.

SLIM Basics

SLIM organizes modules using five factors – module, project, description / subproject, type and version. A note about naming in SLIM vs. ARCHIVER. ARCHIVER refers to the above listed factors as ITEM, GROUP, SUBGROUP, TYPE and VERSION. Through out this document, I will use the SLIM names rather than the ARCHIVER names.

SLIM provides many routine processes to provide simple but effective management of software items. Items are compressed and encrypted to provide a basic level of protection. SLIM has the ability to store different types of data for a project. The source code, which includes macro, copy items, compiler generated source listings as well as load modules.

There are ways other than SLIM to access an Archive. The SLIM software does not provide full maintenance support for a VSAM archive. IDCAMS and ARCHIVER software provide most of the maintenance of the physical copy of the VSAM archive. SLIM provides the maintenance of the logical organization of the archive.

RPF (Rob’s Programming Facility) provides an excelent TSO interface to VSAM archives. RPF (version V1R9M9 or latter) provides the ability to perform a number of functions on an archive in an ISPF like manner. RPF is preinstalled in most Tkx releases. It can be downloaded from https://www.prince-webdesign.nl/rpf

SLIM vs. GIT

If you are familiar with a version control system for PCs called GIT. GIT provides a checkout/checkin function for complete projects. A complete project consists of all code and files to provide a private copy of all pieces of the project. GIT provides tracking of changes so when a project is checked in, it checks if any subprojects are affected. SLIM is provides ***some*** functions similar to GIT. SLIM provides a project based staging/commit facility similar to GIT. SLIM does not provide sophisticated check in/out facilities like GIT does.

Installing SLIM

Download SLIM

Download SLIM-V1-1-0.zip and unzip it. You should find SLIM-V1R1M0.XMI, RESTTK5.TXT, RESTOTHR.TXT and README.txt..

For the balance of the document, the dataset name will start with "SLIM." and volume serial numbers will be your choice.

I suggest you pre-allocate a dataset on MVS with fixed records with LRECL of 80. Transfer SLIM-V1R1M0.XMI as a binary file into the dataset you just allocated on your MVS system.

If you have NJE38 installed (TK5 has it), edit RESTTK5.TXT. The changes you need to make for your installation are documented in the file.

If you do not have NJE38 installed, edit RESTOTHR.TXT. The changes you need to make for your installation are documented in the file..

After the RESTxxxx.JCL job is done, the following 12 datasets are created:

Dataset name	Volume
SLIM.ASM	TSO001
SLIM.COB	TSO001
SLIM.CNTL	TSO001
SLIM.INCLLIB	TSO001
SLIM.LOADLIB	TSO001
SLIM.MACLIB	TSO001
SLIM.NCALIB	TSO001
SLIM.PDSXMI	WORKxx
SLIM.PLI	TSO001
SLIM.PROCLIB	TSO001
SLIM.SAMPLES	TSO001
SLIM-V1R1M0.XMI	WORKxx

Configure SLIM

Before you can use SLIM, it must be configured. Configuration consists of

- Create a SLIM ARCHIVE and SLIMAUTH module
- Updating the SLIM JCL templates

Creating an Archive and SLIMAUTH

An archive is a VSAM Keyed Sequental Data Set (KSDS). An archive must be created using the IDCAMS utility. The JCL for doing this can be found in SLIM.CNTL(NEWARCH). This is the Archiver authors recommended options for an optimal archive. Be sure and make any suggested changes in the JCL.

SLIM dynamicaly allocates ARCHIVER files using a builtin security system. All executions of SLIM require a HLQ (high level qualifier) and a USERID. HLQ is usually a TSOID or catalog alias (i.e. SYS2). USER is a unique name assigned to each user authorized to access SLIM archives. It can be any combination of up to 8 characters. There is a module called SLIMAUTH that verifies the HLQ and USERID to determine the dataset name of the ARCHIVE to be accessed.

The provided SLIMAUTH table is set up for my development environment. I use HERCEL as a development and testing environment and USER1 as a “production” environment. I use HERC01 as a staging area.

Comments may be specified with an “*” in column 1. Column 72 is used to indicate continuations.

Each line begins with “SLIMAUTH TYPE=” followed by a keyword.

TYPE=START and TYPE=END are required.

TYPE=HLQ is defining the list of valid HLQ. In this example, 3 HLQ are defined.

TYPE=USER defines userids, associates the userid with a HLQ and species the dataset name of the ARCHIVE. In the first TYPE=USER, the HLQ HERCEL is related to USER HERCEL. This combination results in the ARCHIVE HERCEL.SLIM.ARCHIVE being accessed. In the second TYPE=USER, the HLQ HERCEL is related to USER HERC01. This combination results in the ARCHIVE HERCEL.SLIM.ARCHIVE being accessed. In the third TYPE=USER, the HLQ USER1 is related to USER HERC01. This combination results in the ARCHIVE USER.SLIM.ARCHIVE being accessed.

In case you have not noticed, this table is an assembler program. It must be assembled and linked to the PDS where the SLIM program is located. The JCL for updating the SLIMAUTH can be found in ‘SLIM.CNTL(MAKEAUTH).

SLIMDDN Table

During a project extract process, the table controls to what datasets the members are extracted from the archive. The provided table covers many common type of files associated with a project. The DDN (data definition name) is used to indicate which dataset receives the member (see page 28).

In SLIM V1.0.0, this table was hard coded in the program. If it was changed, SLIM would have to be recompiled. In V1.1.0, the table is an external module. With the judicious use of STEPLIB or JOBLIB, multiple SLIMDDN tables could exist.

The provided SLIMDDN and the JCL procs (SLIMEXT2 and SLIMNUPR)) must be kept in sync. In other words, each DDN defined in the table must have a corresponding DD name in the proc.

There are some predefined entries in the table. These descriptions and types trigger logic in the SLIM module.

The table is defined using the SLIMDDN macro. The predefined items are always the first items and do not need to be listed in the table definition.

SLIM JCL Templates

SLIM generates JCL to extract an entire project from the ARCHIVE and to compile and link it. There are templates for job cards in the 'SLIM.CNTL' member names TEMPEXTR and TEMPEXT2. You can modify them for your system configuration.

The .CNTL dataset contains the JCL to complete the installation. The members RESTTK5 and RESTOTHR are the same as the .TXT files on the PC. Three members PROCS, PGMS and FINISH contains the JCL to complete the installation of the procs to SYS2.PROCLIB, the SLIM modules to SYS2.LINKLIB and to clean up dataset not longer needed.

Finishing Up

At this point, all the procs in 'SLIM.PROCLIB' can be copied to your system proclib ('SYS2.PROCLIB'). All the executables in SLIM.LOADLIB can be copied to 'SYS2.LINKLIB'. JCL to do this is can be found in SLIM.CNTL(FINISH).

Dependencies

SLIM is dependent on other projects which are freely available. The full source code and executable for each is included

JOBABEND	This is one of my projects. Abends the job.
SETRC	This is one of my projects. Sets the step return code based on EXEC statement parm. Useful for running or skipping subsequent steps in the job.
MAKESTMT	This is one of my projects. Creates a sequential file from data passed in the parms.
MYGENER	This is one of my projects. Like IEBGENER, it can copy and changes record formats F→V, V→F. Records are truncated or padded as appropriate.
CLEANUP	Obtained from CBT183 at CBTTAPE.ORG. Deletes cataloged dataset that will be created in the job. This avoids the dreaded “NOT CATLG 2”.
VSAMIO	Obtained from JAYMOSELEY.COM SYSCPK dasd volume. Provides access to VSAM files for COBOL and PL/I program.
SCOMPARE	This source code compare program was taken from the Stanford Pascal Compiler. Obtained from JAYMOSELEY.COM SYSCPK dasd volume.

I have included the source code and executables as part of the distribution. In addition, ARCHIVER 6.1.8 or higher is required. It is included in TK5. It is also available in the CBTTAPE.ORG file number 147.

Special Note for CLEANUP

CLEANUP accesses MVS internals and will requires an additional library (SYS1.AMODGEN) to assemble clean. If you choose to assemble it, make sure it is not executed from a PDS that CLEANUP will delete.

Using SLIM

General Usage

There are several JCL procedures to implement SLIM. Each PROC has a specific function –

SLIMCL	Proc to Compile and Link without storing results in an archive.
SLIMCLS	Proc to Compile, Link and Store in an archive.
SLIMCLON	Proc to Clone a project.
SLIMCOMP	Proc to Compare two source items.
SLIMDEP	Proc to define a dependencies between a project and sub project
SLIMEXTR SLIMEXT2	Proc SLIMEXTR extracts all the items associated with a given project and generates JCL to compile and link all items using SLIMEXT2.
SLIMNUPR	Proc SLIMNUPR creates new empty projects.
SLIMOTR	Proc used to add items other than source code (i.e. copy items, jcl)
SLIMPAN	Proc used to compile a Panel and store a MAP3270 panel in an archive
SLIMREFR	Proc used to refresh a given set of members for a specific description
SLIMRST	Proc used to reset version numbers.
SLIMSNAP	Proc used to create an “image” of a project.
SLIMXREF	Proc used to print cross reference reports.

SLIM is invoked as a batch program using procs. All information is declared using keyword on the procs.

Each generated JCL job starts with an execution of CLEANUP. Cleanup examines the JCL and will delete any cataloged sequential or PDS that had a DISP of NEW before the balance of the job executes. GDG and VSAM datasets are NOT processed by CLEANUP.

For any of the SLIM procs, a job card is required. If at installation, you choose to NOT copy the load modules to SYS2.LINKLIB, a JOBLIB is required.

```
//jobname JOB

//JOBLIB   DD DSN=SLIM.LOADLIB,DISP=SHR
```

Since SLIM is managed by SLIM, I will use it for examples through out this document.

Proc Keyword Definition

All of the procs have keywords. The same keywords used in each proc have the same meaning. Some procs have unique keywords and will be documented there.

Proc Keywords

CMD is the command – must be ADD or UPDATE

COPYx provides for user supplied COPY or INCLUDE PDS for the compile.

COPYBUFF is set up to default for TK5 3390 volumes. This allows the concatenation of partition datasets (PDS) of different BLKSIZEs.

DESC is the description of the module. See page 28 for a list of valid description or types.

HLQ is the high level qualified of the SLIM archive to be processed.

IN is the DSN including the member to be compiled.

LANG specifies the language of the module. Valid values are ASM, COB, FOR, PAS and PLI.

LOADLIB is the DSN of the PDS where the LOADMOD will be placed.

LOADMOD is the name of the member to be linked into LOADLIB.

LVL is only used for FORTRAN programs to indicate if the G or H Fortran compiler should be used.

MODULE is the name of the item to compiled.

NCAL specified to the module should be linked with the NCAL or CALL option.

NCALIBx sets up for user and system runtime libraries used to resolve link edits.

PLIMAC specifies if the MACRO or NOMACRO option should be used for compiling a PL/I F program.

PLISORT is used only for PL/1 programs that invoke the sort/merge program. Link edits for these program result in return code 4 due to the way the sort interface was link edited. For this case, the RC=4 is considered normal.

PROJECT is the name of the project.

USERID is the SLIM userid. For this version, it is the TSO used id.

VOL is the DASD volume to allocate datasets on.

Adding User Notes

Starting with SLIM V1.1.0, any proc that updates member(s) in an SLIM Archive, also generates NOTES to serve as a form of documentation. ARCHIVER provides the ability to attach notes to each member. Each member updated will have at least one note listing “who done it”. In addition, user notes can be added as follows:

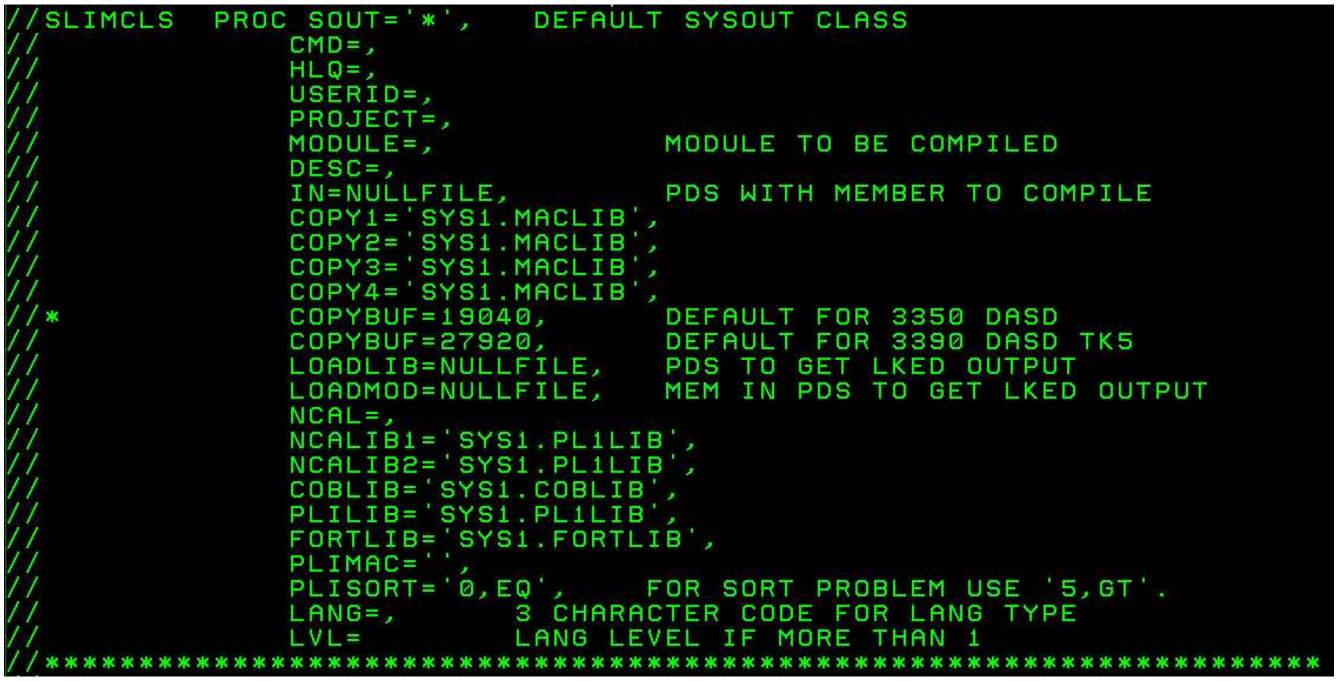
```
//somejob JOB …
//someproc EXEC SLIMxxxx,
//  keywords,
//SLIM.NOTES DD *
THIS IS ADDITIONAL INFO ABOUT THIS OPERATION.
COLUMNS 1-72 ARE INSERTED INTO THE NOTES.
COLUMNS 73-80 ARE IGNORED
/*
```

SLIMCL/SLIMCLS Proc

The SLIMCLS proc will compile and link a program. It is similar to SLIMCLS except the source code, listing and load module are not stored in an archive.

The SLIMCLS proc will compile and link a program. If all steps were successful, the source code, the load module and compiler listing will be stored in the SLIM archive.

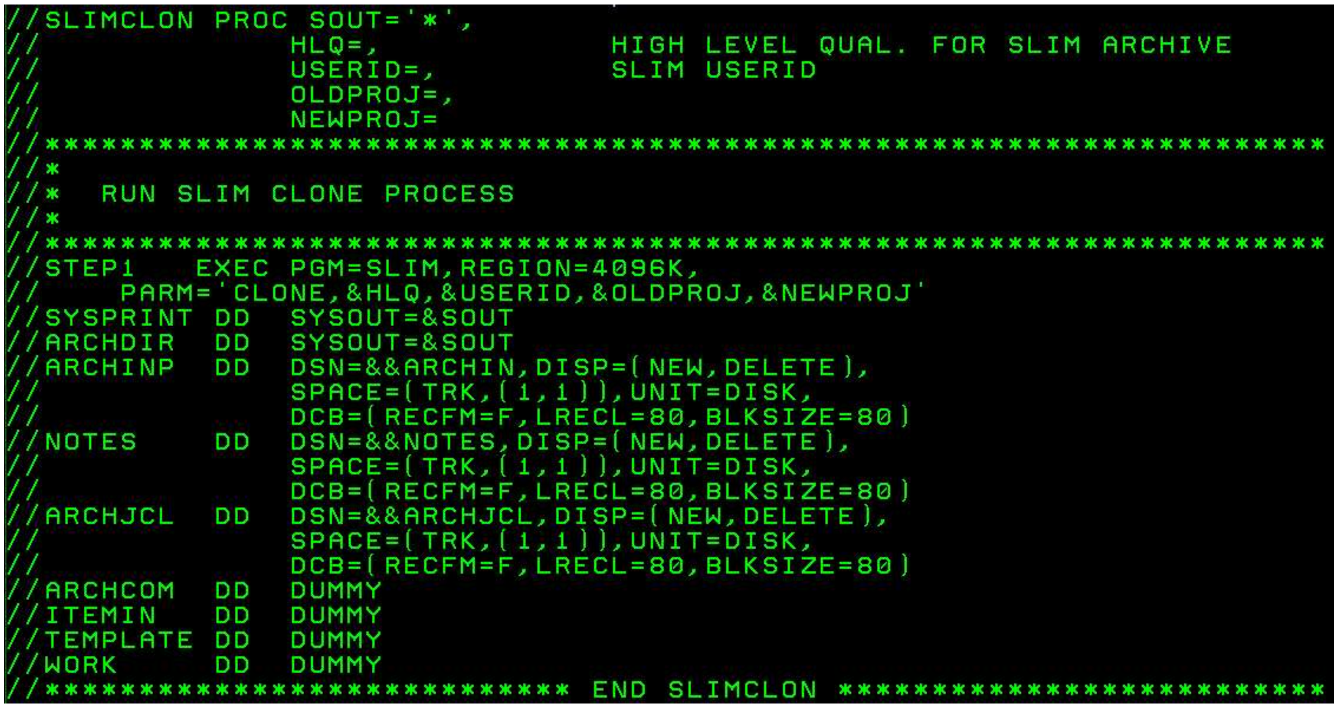
In order to add NOTES using SLIMCLS, there is a minor difference that the other SLIM procs. If you specify NCAL=NCAL, the use the JCL "//SLIMN.NOTES…".   If you specify NCAL=CALL, use the JCL "//SLIMC.NOTES…".



```
//SLIMCLS  PROC SOUT='*',    DEFAULT SYSOUT CLASS
//              CMD=,
//              HLQ=,
//              USERID=,
//              PROJECT=,
//              MODULE=,             MODULE TO BE COMPILED
//              DESC=,
//              IN=NULLFILE,         PDS WITH MEMBER TO COMPILE
//              COPY1='SYS1.MACLIB',
//              COPY2='SYS1.MACLIB',
//              COPY3='SYS1.MACLIB',
//              COPY4='SYS1.MACLIB',
//*             COPYBUF=19040,       DEFAULT FOR 3350 DASD
//              COPYBUF=27920,       DEFAULT FOR 3390 DASD TK5
//              LOADLIB=NULLFILE,    PDS TO GET LKED OUTPUT
//              LOADMOD=NULLFILE,    MEM IN PDS TO GET LKED OUTPUT
//              NCAL=,
//              NCALIB1='SYS1.PL1LIB',
//              NCALIB2='SYS1.PL1LIB',
//              COBLIB='SYS1.COBLIB',
//              PLILIB='SYS1.PL1LIB',
//              FORTLIB='SYS1.FORTLIB',
//              PLIMAC='',
//              PLISORT='0,EQ',     FOR SORT PROBLEM USE '5,GT'.
//              LANG=,       3 CHARACTER CODE FOR LANG TYPE
//              LVL=         LANG LEVEL IF MORE THAN 1
//*******************************************************************
```

SLIMCLON Proc

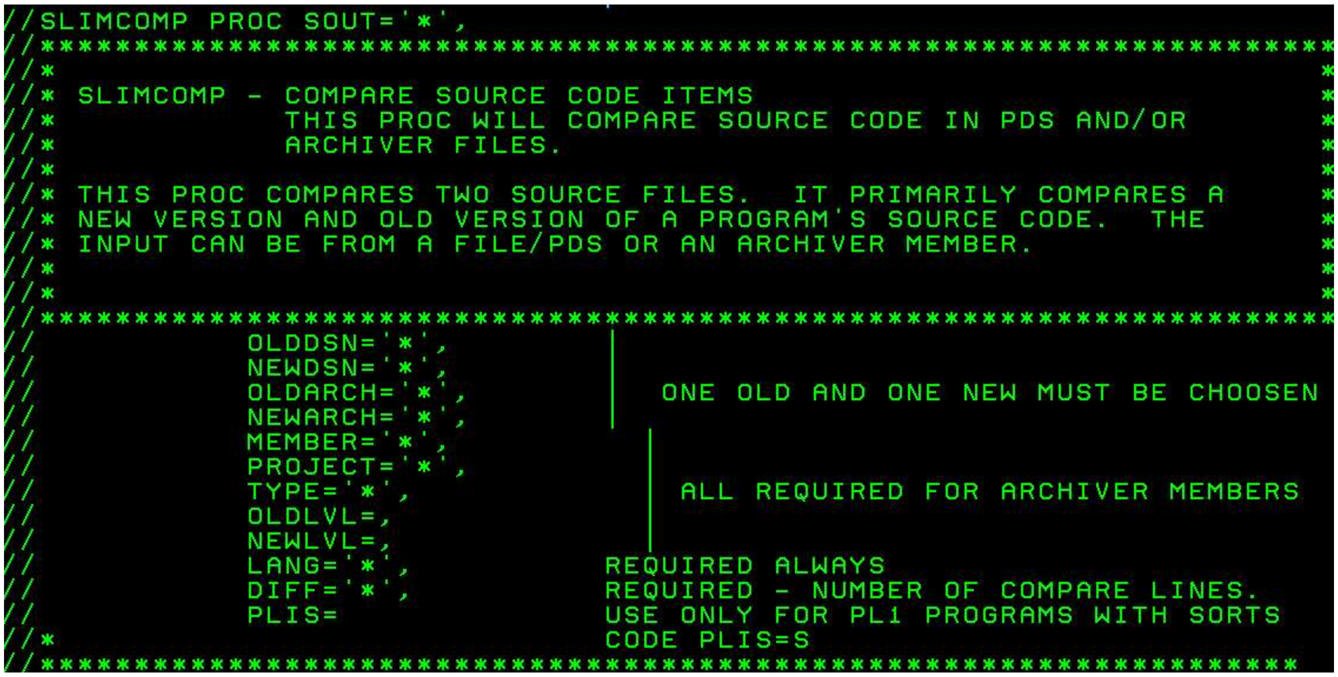
The SLIMCLON proc is used to make a complete copy of a project within the archive.



```
//SLIMCLON PROC SOUT='*',
//             HLQ=,             HIGH LEVEL QUAL. FOR SLIM ARCHIVE
//             USERID=,          SLIM USERID
//             OLDPROJ=,
//             NEWPROJ=
//*******************************************************************
//*
//*  RUN SLIM CLONE PROCESS
//*
//*******************************************************************
//STEP1   EXEC PGM=SLIM,REGION=4096K,
//     PARM='CLONE,&HLQ,&USERID,&OLDPROJ,&NEWPROJ'
//SYSPRINT DD  SYSOUT=&SOUT
//ARCHDIR  DD  SYSOUT=&SOUT
//ARCHINP  DD  DSN=&&ARCHIN,DISP=(NEW,DELETE),
//             SPACE=(TRK,(1,1)),UNIT=DISK,
//             DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//NOTES    DD  DSN=&&NOTES,DISP=(NEW,DELETE),
//             SPACE=(TRK,(1,1)),UNIT=DISK,
//             DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//ARCHJCL  DD  DSN=&&ARCHJCL,DISP=(NEW,DELETE),
//             SPACE=(TRK,(1,1)),UNIT=DISK,
//             DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//ARCHCOM  DD  DUMMY
//ITEMIN   DD  DUMMY
//TEMPLATE DD  DUMMY
//WORK     DD  DUMMY
//**************************** END SLIMCLON *************************
```

SLIMCOMP Proc

The SLIMCOMP proc is used to compare to two members. Input to the compare can be any combination of PDS members and/or ARCHIVE members.



```
//SLIMCOMP PROC SOUT='*',
//*********************************************************************
//*                                                                   *
//*  SLIMCOMP - COMPARE SOURCE CODE ITEMS                             *
//*             THIS PROC WILL COMPARE SOURCE CODE IN PDS AND/OR      *
//*             ARCHIVER FILES.                                       *
//*                                                                   *
//*  THIS PROC COMPARES TWO SOURCE FILES.  IT PRIMARILY COMPARES A    *
//*  NEW VERSION AND OLD VERSION OF A PROGRAM'S SOURCE CODE.  THE     *
//*  INPUT CAN BE FROM A FILE/PDS OR AN ARCHIVER MEMBER.              *
//*                                                                   *
//*                                                                   *
//*********************************************************************
//             OLDDSN='*',        |
//             NEWDSN='*',        |
//             OLDARCH='*',       |  ONE OLD AND ONE NEW MUST BE CHOOSEN
//             NEWARCH='*',       |
//             MEMBER='*',         |
//             PROJECT='*',        |
//             TYPE='*',           |  ALL REQUIRED FOR ARCHIVER MEMBERS
//             OLDLVL=,            |
//             NEWLVL=,            |
//             LANG='*',          REQUIRED ALWAYS
//             DIFF='*',          REQUIRED - NUMBER OF COMPARE LINES.
//             PLIS=              USE ONLY FOR PL1 PROGRAMS WITH SORTS
//*                               CODE PLIS=S
//*******************************************************************
```

OLDDSN/NEWDSN specify the PDS(member) to be compared.

OLDARCH/NEWARCH give the specific ARCHIVE name. If either or both are is used, the MEMBER, PROJECT, TYPE and LANG must be given.

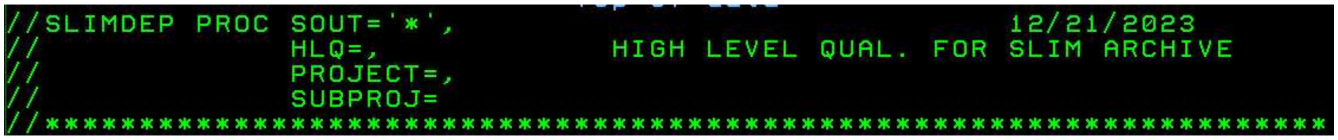
DIFF is used by the compare program to determine number of lines to compare to determine difference. 6 is the recommend value.

The process of compare is fairly straight foreword. However, there are many possible combinations of old and new that can be compared.

See the “SLIM.SAMPLES(COMP)” to all the documented source compares that are possible.

SLIMDEP Proc

The SLIMDEP is used to define dependencies between projects. A dependency exists when module that requires another module for successful use. An example is a main program calls a subroutine. That makes the main program dependent upon the subroutine.

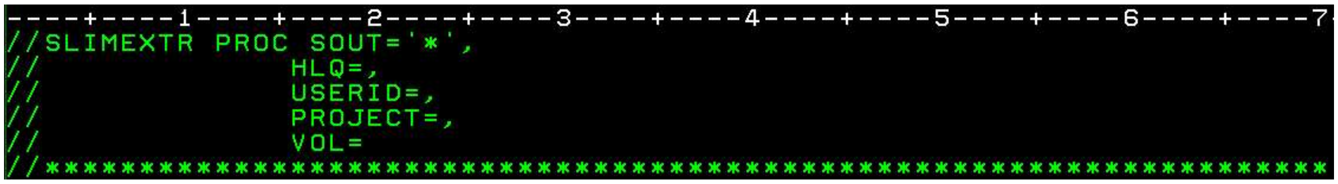


```
//SLIMDEP PROC SOUT='*',                           12/21/2023
//             HLQ=,          HIGH LEVEL QUAL. FOR SLIM ARCHIVE
//             PROJECT=,
//             SUBPROJ=
//*********************************************************************
```

In SLIM, as noted above, a project is a collection of items such as source code, copy/include items, macros, etc that are required for the project to be implemented and compiled if necessary. A full explanation of dependencies is discussed under the SLIMEXTR proc.

SLIMEXTR Proc

The SLIMEXTR proc executes the extract process to JCL to extract all the current versions of members associated with a project as well as dependencies to compile and link a project. This process generates a dataset (userid.`project.EXTRACT.CNTL`) with JCL statements to extract the items from an archive. No actual extraction takes place in this step. Also, if appropriate, SLIM will generate a dataset (userid.project.`COMPILE.CNTL`) of JCL with a series of jobs to compile and link ALL of the items in the project.



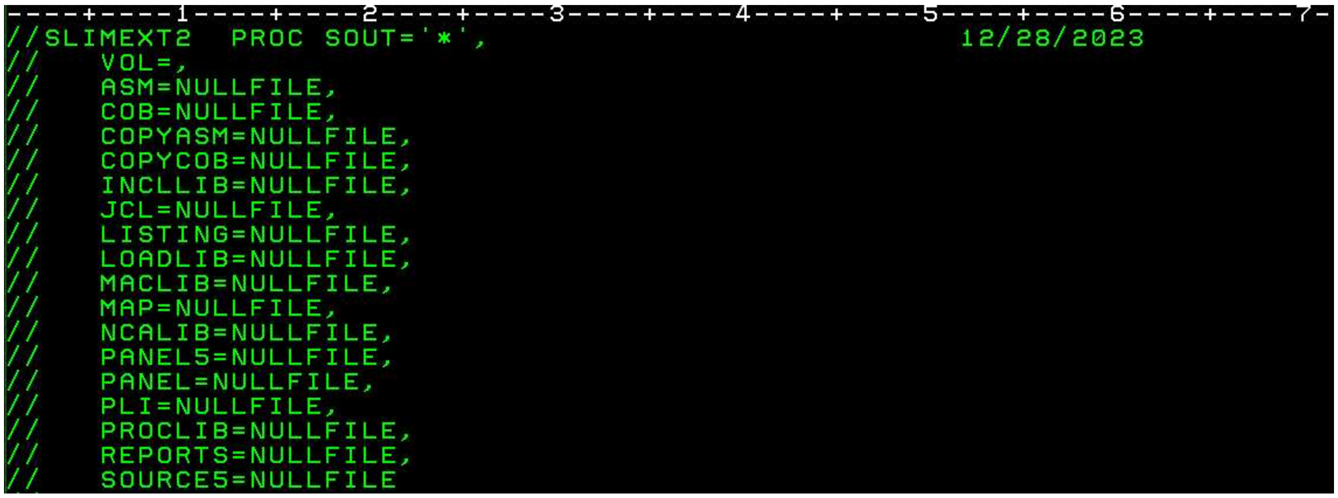
```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7
//SLIMEXTR PROC SOUT='*',
//             HLQ=,
//             USERID=,
//             PROJECT=,
//             VOL=
//*********************************************************************
```

The extract job must be manually submitted. Upon completion, compile job should be submitted. Each job in the compile JCL has the same jobname. This is to ensure the jobs run in sequence of dependency. What this means is subroutines are compiled and linked before any other program calls them. The sequence is determined using the "++DEPEND" items.

For example, Program A calls Sub B. Sub B will be processed first so it is available when Program A is compiled and linked.

SLIMEXT2 PROC

The SLIMEXT2 proc is used in the compile JCL generated by the SLIMEXTR process. Each compile and link is a separate job. The proc is designed to abend if any step is determined to fail based on return codes. The JCL for executing this proc is generated by SLIM and no further explanation will be made.

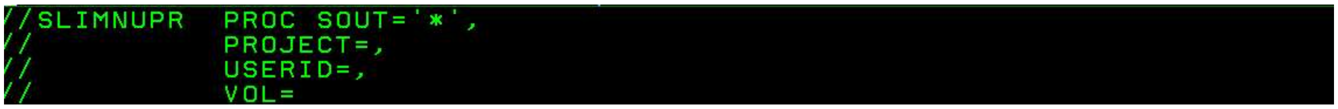


```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7-
//SLIMEXT2  PROC SOUT='*',                              12/28/2023
//    VOL=,
//    ASM=NULLFILE,
//    COB=NULLFILE,
//    COPYASM=NULLFILE,
//    COPYCOB=NULLFILE,
//    INCLLIB=NULLFILE,
//    JCL=NULLFILE,
//    LISTING=NULLFILE,
//    LOADLIB=NULLFILE,
//    MACLIB=NULLFILE,
//    MAP=NULLFILE,
//    NCALIB=NULLFILE,
//    PANELS=NULLFILE,
//    PANEL=NULLFILE,
//    PLI=NULLFILE,
//    PROCLIB=NULLFILE,
//    REPORTS=NULLFILE,
//    SOURCES=NULLFILE
```

There are items in the SLIM distribution that require special handling for compiling. See page 25

SLIMNUPR Proc

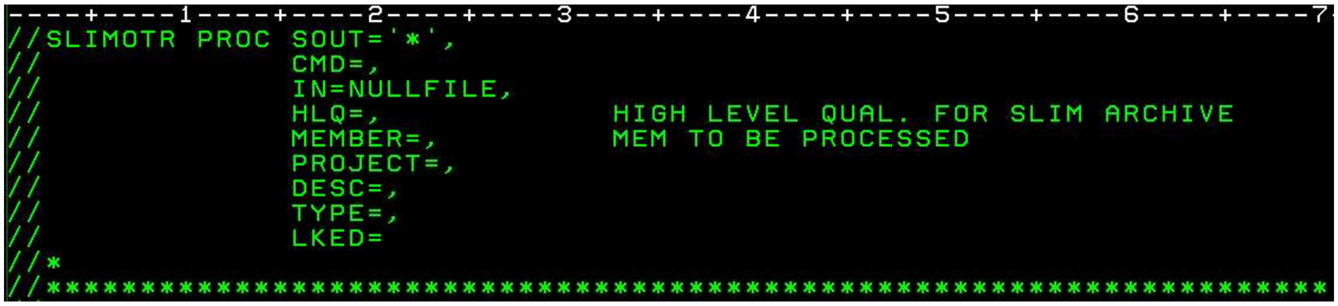
The SLIMNUPR proc is used to create empty SLIM projects.



```
//SLIMNUPR  PROC SOUT='*',
//             PROJECT=,
//             USERID=,
//             VOL=
```

SLIMOTR Proc

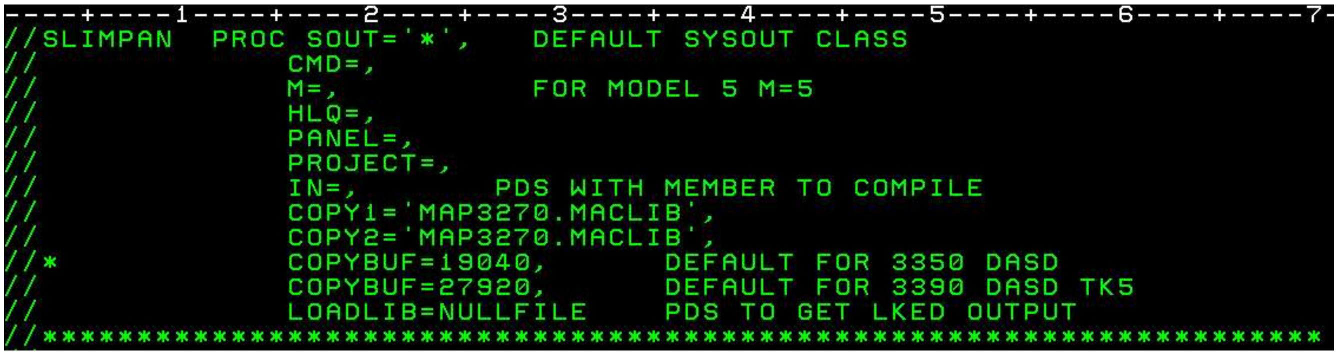
The SLIIMOTR proc is used to process items into an archive that do not need to be compiled such as JCL, PROCS, copy items or any other “card” image.



```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7
//SLIMOTR PROC SOUT='*',
//             CMD=,
//             IN=NULLFILE,
//             HLQ=,            HIGH LEVEL QUAL. FOR SLIM ARCHIVE
//             MEMBER=,         MEM TO BE PROCESSED
//             PROJECT=,
//             DESC=,
//             TYPE=,
//             LKED=
//*
//*********************************************************************
```

SLIMPAN Proc

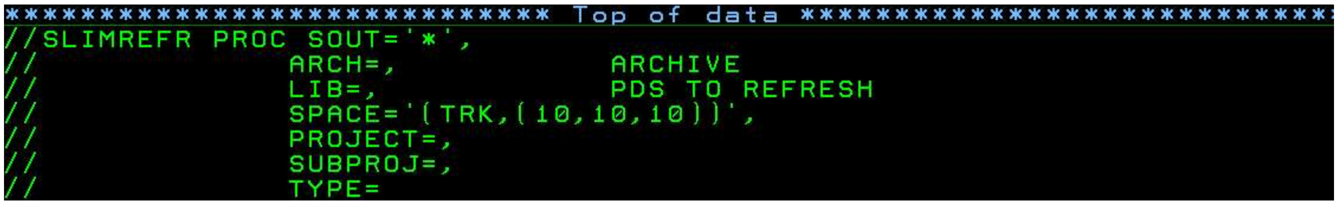
The SLIMPAN proc is used to process a MAP3270 panel into an archive.



```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7
//SLIMPAN  PROC SOUT='*',    DEFAULT SYSOUT CLASS
//              CMD=,
//              M=,          FOR MODEL 5 M=5
//              HLQ=,
//              PANEL=,
//              PROJECT=,
//              IN=,        PDS WITH MEMBER TO COMPILE
//              COPY1='MAP3270.MACLIB',
//              COPY2='MAP3270.MACLIB',
//*             COPYBUF=19040,       DEFAULT FOR 3350 DASD
//              COPYBUF=27920,       DEFAULT FOR 3390 DASD TK5
//              LOADLIB=NULLFILE     PDS TO GET LKED OUTPUT
//*********************************************************************
```

SLIMREFR Proc

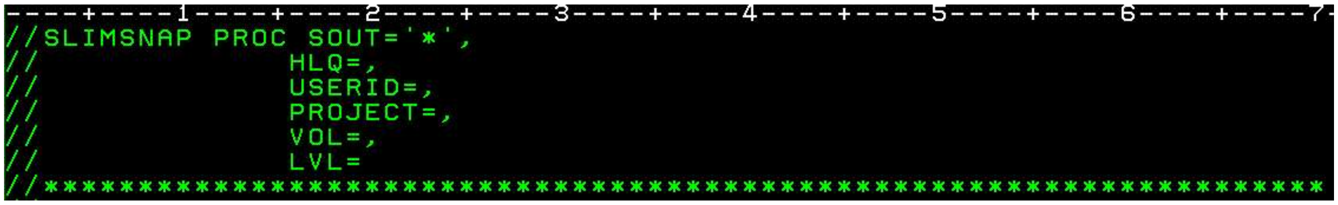
The SLIMREFR proc is used to refresh a PDS from a SLIM archive. For a given project, a PDS is created only for current SLIM members of a given description/subproject.


```
****************************** Top of data ******************************
//SLIMREFR PROC SOUT='*',
//              ARCH=,             ARCHIVE
//              LIB=,              PDS TO REFRESH
//              SPACE='(TRK,(10,10,10))',
//              PROJECT=,
//              SUBPROJ=,
//              TYPE=
```

SLIMSNAP Proc

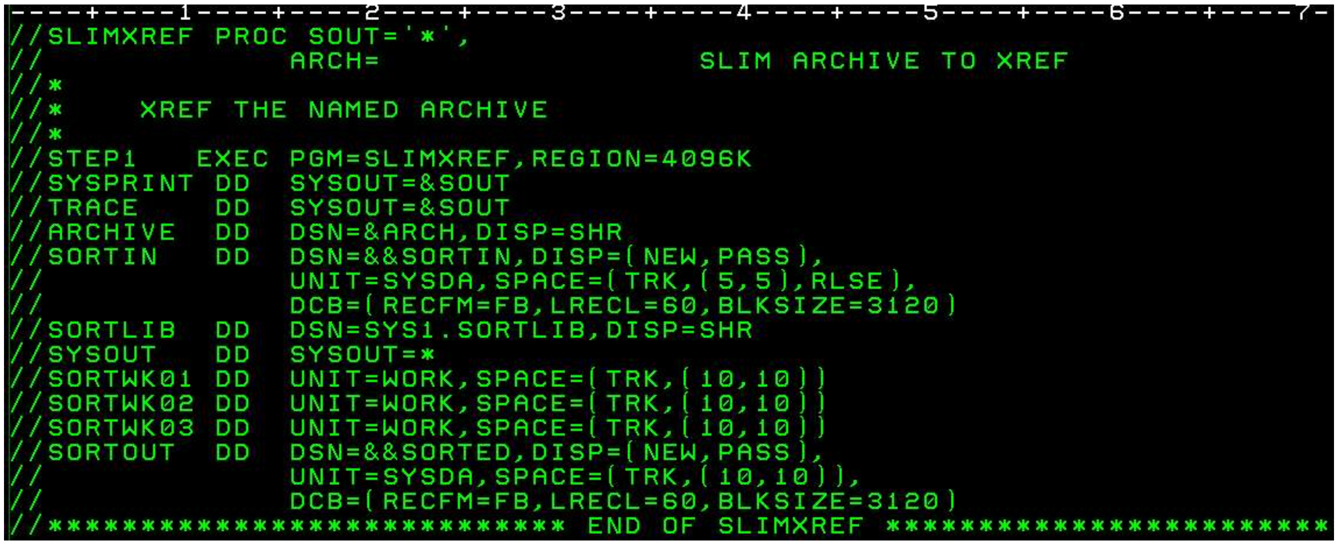
The SLIMSNAP proc produces a snap shot of a project. A snap shot is all the JCL to extract and compile a project in its current state. It works like SLIMEXTR except the generated JCL is stored in an archive. This enables the state of a project be captured.

Items in an archive are identified by MEMBER, PROJECT, SUBPROJECT, TYPE and version. Each time an item is updated, the version becomes 1 greater. This allows history of changes to be kept.

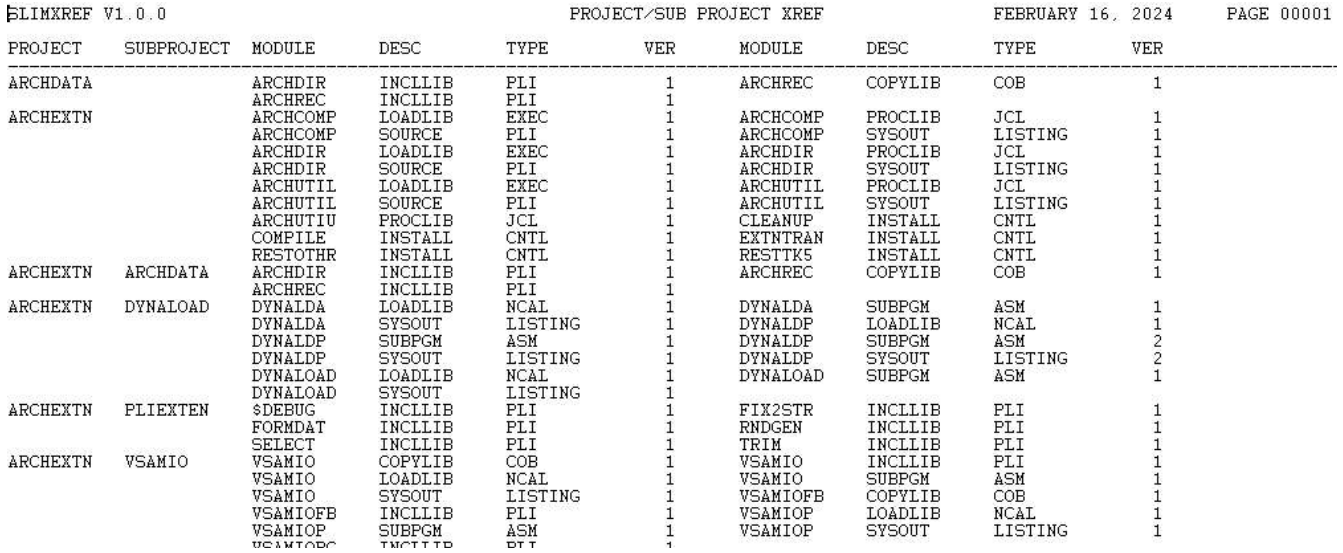

```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7
//SLIMSNAP PROC SOUT='*',
//              HLQ=,
//              USERID=,
//              PROJECT=,
//              VOL=,
//              LVL=
//*********************************************************************
```

SLIMXREF Proc

The SLIMXREF proc print two reports – Project/Subproject which lists all the items in a project. The other Subproject/Project which lists all the projects where the project is reference.



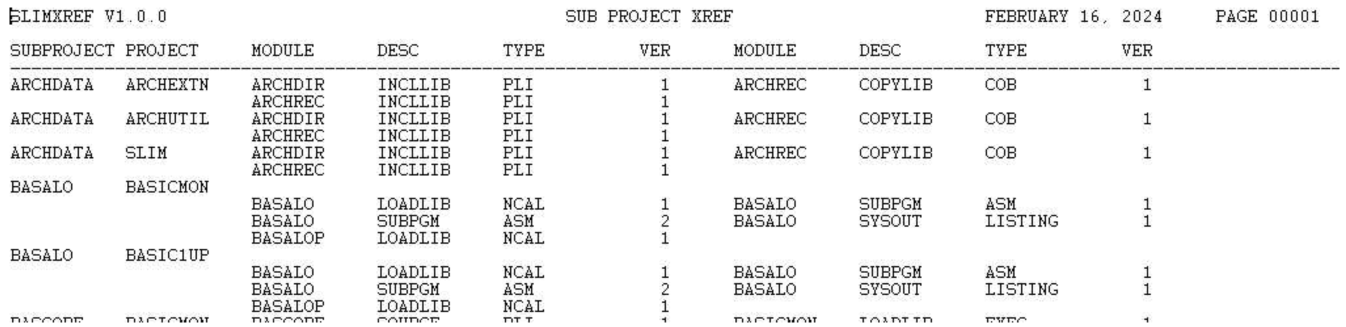
```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7-
//SLIMXREF PROC SOUT='*',
//             ARCH=                     SLIM ARCHIVE TO XREF
//*
//*    XREF THE NAMED ARCHIVE
//*
//STEP1   EXEC PGM=SLIMXREF,REGION=4096K
//SYSPRINT DD  SYSOUT=&SOUT
//TRACE    DD  SYSOUT=&SOUT
//ARCHIVE  DD  DSN=&ARCH,DISP=SHR
//SORTIN   DD  DSN=&&SORTIN,DISP=(NEW,PASS),
//             UNIT=SYSDA,SPACE=(TRK,(5,5),RLSE),
//             DCB=(RECFM=FB,LRECL=60,BLKSIZE=3120)
//SORTLIB  DD  DSN=SYS1.SORTLIB,DISP=SHR
//SYSOUT   DD  SYSOUT=*
//SORTWK01 DD  UNIT=WORK,SPACE=(TRK,(10,10))
//SORTWK02 DD  UNIT=WORK,SPACE=(TRK,(10,10))
//SORTWK03 DD  UNIT=WORK,SPACE=(TRK,(10,10))
//SORTOUT  DD  DSN=&&SORTED,DISP=(NEW,PASS),
//             UNIT=SYSDA,SPACE=(TRK,(10,10)),
//             DCB=(RECFM=FB,LRECL=60,BLKSIZE=3120)
//**************************** END OF SLIMXREF ************************
```



```
SLIMXREF V1.0.0                                      PROJECT/SUB PROJECT XREF                 FEBRUARY 16, 2024     PAGE 00001

PROJECT    SUBPROJECT  MODULE      DESC        TYPE         VER      MODULE      DESC        TYPE         VER
-----------------------------------------------------------------------------------------------------------------------------
ARCHDATA               ARCHDIR     INCLLIB     PLI            1      ARCHREC     COPYLIB     COB            1
                       ARCHREC     INCLLIB     PLI            1
ARCHEXTN               ARCHCOMP    LOADLIB     EXEC           1      ARCHCOMP    PROCLIB     JCL            1
                       ARCHCOMP    SOURCE      PLI            1      ARCHCOMP    SYSOUT      LISTING        1
                       ARCHDIR     LOADLIB     EXEC           1      ARCHDIR     PROCLIB     JCL            1
                       ARCHDIR     SOURCE      PLI            1      ARCHDIR     SYSOUT      LISTING        1
                       ARCHUTIL    LOADLIB     EXEC           1      ARCHUTIL    PROCLIB     JCL            1
                       ARCHUTIL    SOURCE      PLI            1      ARCHUTIL    SYSOUT      LISTING        1
                       ARCHUTIU    PROCLIB     JCL            1      CLEANUP     INSTALL     CNTL           1
                       COMPILE     INSTALL     CNTL           1      EXTNTRAN    INSTALL     CNTL           1
                       RESTOTHR    INSTALL     CNTL           1      RESTTK5     INSTALL     CNTL           1
ARCHEXTN   ARCHDATA    ARCHDIR     INCLLIB     PLI            1      ARCHREC     COPYLIB     COB            1
                       ARCHREC     INCLLIB     PLI            1
ARCHEXTN   DYNALOAD    DYNALDA     LOADLIB     NCAL           1      DYNALDA     SUBPGM      ASM            1
                       DYNALDA     SYSOUT      LISTING        1      DYNALDP     LOADLIB     NCAL           1
                       DYNALDP     SUBPGM      ASM            1      DYNALDP     SUBPGM      ASM            2
                       DYNALDP     SYSOUT      LISTING        1      DYNALDP     SYSOUT      LISTING        2
                       DYNALOAD    LOADLIB     NCAL           1      DYNALOAD    SUBPGM      ASM            1
                       DYNALOAD    SYSOUT      LISTING        1
ARCHEXTN   PLIEXTEN    $DEBUG      INCLLIB     PLI            1      FIX2STR     INCLLIB     PLI            1
                       FORMDAT     INCLLIB     PLI            1      RNDGEN      INCLLIB     PLI            1
                       SELECT      INCLLIB     PLI            1      TRIM        INCLLIB     PLI            1
ARCHEXTN   VSAMIO      VSAMIO      COPYLIB     COB            1      VSAMIO      INCLLIB     PLI            1
                       VSAMIO      LOADLIB     NCAL           1      VSAMIO      SUBPGM      ASM            1
                       VSAMIO      SYSOUT      LISTING        1      VSAMIOFB    COPYLIB     COB            1
                       VSAMIOFB    INCLLIB     PLI            1      VSAMIOP     LOADLIB     NCAL           1
                       VSAMIOP     SUBPGM      ASM            1      VSAMIOP     SYSOUT      LISTING        1
                       VSAMIOPC    INCLLIB     PLI            1
```

Figure 1: Sample Project/Subproject Report



```
SLIMXREF V1.0.0                                       SUB PROJECT XREF                                FEBRUARY 16, 2024     PAGE 00001
SUBPROJECT PROJECT     MODULE      DESC        TYPE         VER      MODULE      DESC        TYPE         VER
------------------------------------------------------------------------------------------------------------------------------------
ARCHDATA   ARCHEXTN    ARCHDIR     INCLLIB     PLI            1      ARCHREC     COPYLIB     COB            1
                       ARCHREC     INCLLIB     PLI            1
ARCHDATA   ARCHUTIL    ARCHDIR     INCLLIB     PLI            1      ARCHREC     COPYLIB     COB            1
                       ARCHREC     INCLLIB     PLI            1
ARCHDATA   SLIM        ARCHDIR     INCLLIB     PLI            1      ARCHREC     COPYLIB     COB            1
                       ARCHREC     INCLLIB     PLI            1
BASALO     BASICMON
                       BASALO      LOADLIB     NCAL           1      BASALO      SUBPGM      ASM            1
                       BASALO      SUBPGM      ASM            2      BASALO      SYSOUT      LISTING        1
                       BASALOP     LOADLIB     NCAL           1
BASALO     BASIC1UP
                       BASALO      LOADLIB     NCAL           1      BASALO      SUBPGM      ASM            1
                       BASALO      SUBPGM      ASM            2      BASALO      SYSOUT      LISTING        1
                       BASALOP     LOADLIB     NCAL           1
BASCORE    BASICMON    BASCORE     SOURCE      PLI            1      BASICMON    LOADLIB     EXEC           1
```

Figure 2: Sample Subprojec/Project Report

Creating a SLIM Project

As stated back in SLIM Basics (page 5), a SLIM project consists of one or more components and sub components. A component is one or more items that generally are used together. Each item in a component is a piece of that project or a dependency on another component, called a sub project.

The key to setting up a project is to breaking it up into sub projects. For the balance of this section, I will refer to “copy code” to mean any items that are copied into COBOL programs using the COPY statement, Assembler COPY statements, user macros and PL/I INCLUDE statements.

In order to explain how a project is defined in SLIM, I will be using as an example the SLIM project as defined in SLIM.

The following figure (on page 24 Figure 3: SLIM Project as of this writing.) is a screen print of the SLIM project as viewed using RPF option 3.9. Note that RPF uses the column names as defined by Archiver. I will go step by step on how to create a SLIM project.

Initialize The Project

To create an empty project for SLIM, execute proc SLIMNUPR. This will create all the possible datasets to hold the components of the project. The proc has 3 parms:

- USERID is the high level qualifier for the datasets, most likely will be TSO userid.
- PROJECT is the mid level qualifier for the datasets.
- VOL is the volume where the datasets should be created.

For example, to create an empty project for user HERCEL called SLIM on volume TSO001:

```
//STEP   EXEC  SLIMNUPR,USERID=HERCEL,PROJECT=SLIM,VOL=TSO001
```

All the of possible empty datasets for the project called SLIM will be created. **Please note – any existing datasets starting with HERCEL.SLIM will be DELETED.**

Populate The Project

At this point, you can start creating the members into the project PDSs as you normally would.

- Macros, if any, in the MACLIB,
- copy code, if any, in an appropriate COPY/INCLUDE PDS,
- source code in appropriate PDS.

You can compile and test using your regular procs.

When you are ready, you can now store your project in SLIM. To do this, each support piece should be added to SLIM. The proc SLIMOTR can be used to add or update any item that does not need to be compiled. This include macro definition, copy items, etc. Proc SLIMCLS is used to compile, link and store a module in SLIM. More on SLIMCLS later.

An example of proc SLIMOTR is shown below. This example add a proc called SLIMEXTR to project SLIM.

```
//S1  EXEC SLIMOTR,CMD=ADD,
//             HLQ=HERCEL,
//             USERID=HERCEL,
//             PROJECT=SLIM,
//             MEMBER=SLIMEXTR,
//             DESC=PROCLIB,
//             TYPE=JCL,
//             IN='HERCEL.SLIM.PROCLIB(SLIMEXTR)'
//SLIM.NOTES   DD *
SLIM V1.1.0 RELEASE
```

The last two lines are optional and allow you to add notes/comments to the member. Note – only positions 1-72 are added. You can have upto 9,999 note lines. If omitted, a default note is generated indicating who and when the member was added to SLIM.

As mentioned earlier, proc SLIMCLS is used to compile, link and then *store the source code, the source and link edit listing as well as the load module into SLIM*. Proc SLIMCL is a version of SLIMCLS that *does not store anything into SLIM*.

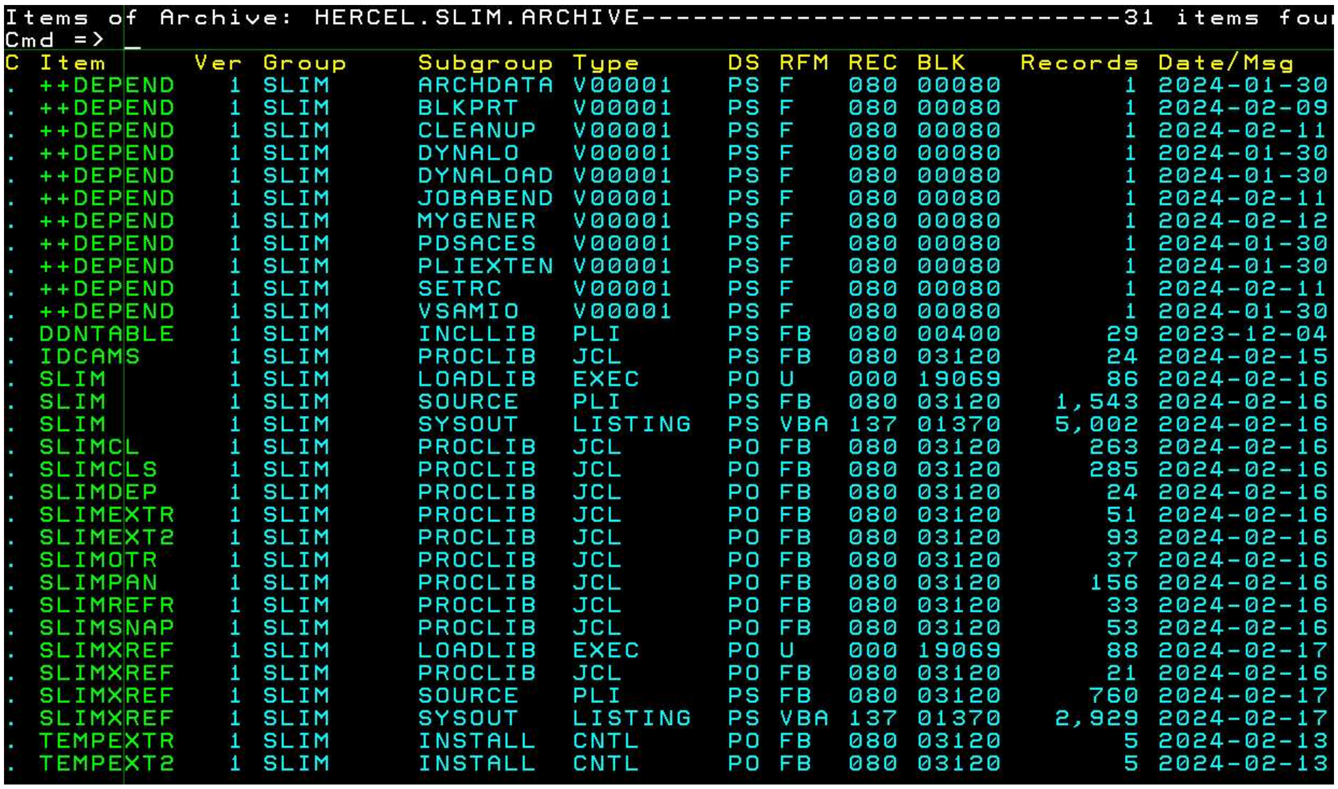
Creating Dependencies

The items with the name “++DEPEND” define the relationship between a project (group) and sub-project (subgroup). The type currently is just a filler. This group of “++DEPEND” declares that the SLIM project has a dependency on all those projects listed.

The balance of the items list the names of the items that are members of the project SLIM.

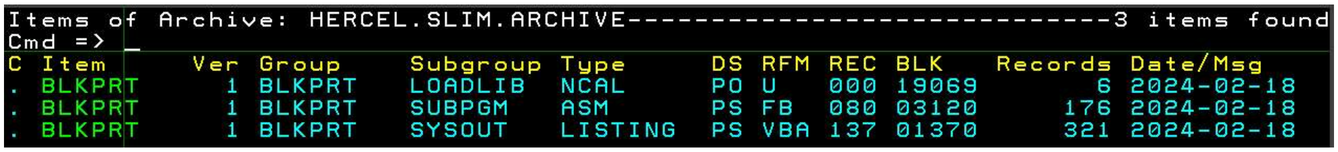
The next figure (on page 24 Figure 4: Example of a sub project) shows one of the projects that SLIM is dependent on.

When an SLIM extract process (SLIMEXTR Proc on page 16) for a project is executed, the Archive is scanned and a list of all the items the a project is created. All items in the subprojects specified in the "++DEPEND" are included in this list. The list is then used to produce JCL to extract all the items from the archive and JCL to compile the entire project is generated. It is possible that sub-projects can have dependencies. These will be processed as well.



```
Items of Archive: HERCEL.SLIM.ARCHIVE----------------------------31 items fou
Cmd =>  _
C Item      Ver Group      Subgroup Type      DS RFM REC BLK      Records Date/Msg
. ++DEPEND    1 SLIM       ARCHDATA V00001    PS F   080 00080          1 2024-01-30
. ++DEPEND    1 SLIM       BLKPRT   V00001    PS F   080 00080          1 2024-02-09
. ++DEPEND    1 SLIM       CLEANUP  V00001    PS F   080 00080          1 2024-02-11
. ++DEPEND    1 SLIM       DYNALO   V00001    PS F   080 00080          1 2024-01-30
. ++DEPEND    1 SLIM       DYNALOAD V00001    PS F   080 00080          1 2024-01-30
. ++DEPEND    1 SLIM       JOBABEND V00001    PS F   080 00080          1 2024-02-11
. ++DEPEND    1 SLIM       MYGENER  V00001    PS F   080 00080          1 2024-02-12
. ++DEPEND    1 SLIM       PDSACES  V00001    PS F   080 00080          1 2024-01-30
. ++DEPEND    1 SLIM       PLIEXTEN V00001    PS F   080 00080          1 2024-01-30
. ++DEPEND    1 SLIM       SETRC    V00001    PS F   080 00080          1 2024-02-11
. ++DEPEND    1 SLIM       VSAMIO   V00001    PS F   080 00080          1 2024-01-30
. DDNTABLE    1 SLIM       INCLLIB  PLI       PS FB  080 00400         29 2023-12-04
. IDCAMS      1 SLIM       PROCLIB  JCL       PS FB  080 03120         24 2024-02-15
. SLIM        1 SLIM       LOADLIB  EXEC      PO U   000 19069         86 2024-02-16
. SLIM        1 SLIM       SOURCE   PLI       PS FB  080 03120      1,543 2024-02-16
. SLIM        1 SLIM       SYSOUT   LISTING   PS VBA 137 01370      5,002 2024-02-16
. SLIMCL      1 SLIM       PROCLIB  JCL       PO FB  080 03120        263 2024-02-16
. SLIMCLS     1 SLIM       PROCLIB  JCL       PO FB  080 03120        285 2024-02-16
. SLIMDEP     1 SLIM       PROCLIB  JCL       PO FB  080 03120         24 2024-02-16
. SLIMEXTR    1 SLIM       PROCLIB  JCL       PO FB  080 03120         51 2024-02-16
. SLIMEXT2    1 SLIM       PROCLIB  JCL       PO FB  080 03120         93 2024-02-16
. SLIMOTR     1 SLIM       PROCLIB  JCL       PO FB  080 03120         37 2024-02-16
. SLIMPAN     1 SLIM       PROCLIB  JCL       PO FB  080 03120        156 2024-02-16
. SLIMREFR    1 SLIM       PROCLIB  JCL       PO FB  080 03120         33 2024-02-16
. SLIMSNAP    1 SLIM       PROCLIB  JCL       PO FB  080 03120         53 2024-02-16
. SLIMXREF    1 SLIM       LOADLIB  EXEC      PO U   000 19069         88 2024-02-17
. SLIMXREF    1 SLIM       PROCLIB  JCL       PO FB  080 03120         21 2024-02-16
. SLIMXREF    1 SLIM       SOURCE   PLI       PS FB  080 03120        760 2024-02-17
. SLIMXREF    1 SLIM       SYSOUT   LISTING   PS VBA 137 01370      2,929 2024-02-17
. TEMPEXTR    1 SLIM       INSTALL  CNTL      PO FB  080 03120          5 2024-02-13
. TEMPEXT2    1 SLIM       INSTALL  CNTL      PO FB  080 03120          5 2024-02-13
```

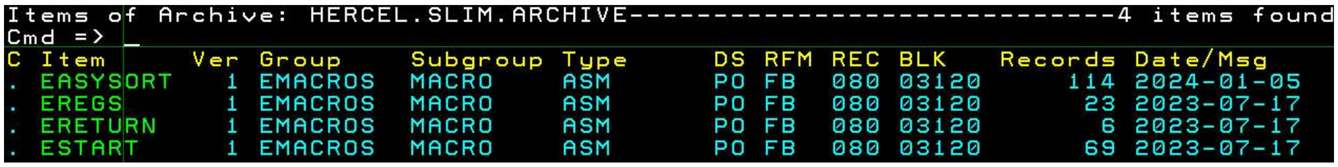
Figure 3: SLIM Project as of this writing.



```
Items of Archive: HERCEL.SLIM.ARCHIVE-----------------------------3 items found
Cmd =>  _
C Item      Ver Group      Subgroup Type      DS RFM REC BLK      Records Date/Msg
. BLKPRT      1 BLKPRT     LOADLIB  NCAL      PO U   000 19069          6 2024-02-18
. BLKPRT      1 BLKPRT     SUBPGM   ASM       PS FB  080 03120        176 2024-02-18
. BLKPRT      1 BLKPRT     SYSOUT   LISTING   PS VBA 137 01370        321 2024-02-18
```

Figure 4: Example of a sub project

Creation of a project is a multi step process and need not be completed all at once. The process can be dynamic and spread over the life of the development cycle. Also, consider the possible reuse of items by other projects. Items like copy or include members, macros etc. For example, I wrote some macros that provide for standard sub routine linkages. I defined the macros as a project EMACROS (see Figure 5: EMACROS Definition on page 25).



```
Items of Archive: HERCEL.SLIM.ARCHIVE----------------------------4 items found
Cmd => _
C Item       Ver Group    Subgroup Type     DS RFM REC BLK     Records Date/Msg
. EASYSORT     1 EMACROS  MACRO    ASM      PO FB  080 03120       114 2024-01-05
. EREGS        1 EMACROS  MACRO    ASM      PO FB  080 03120        23 2023-07-17
. ERETURN      1 EMACROS  MACRO    ASM      PO FB  080 03120         6 2023-07-17
. ESTART       1 EMACROS  MACRO    ASM      PO FB  080 03120        69 2023-07-17
```

Figure 5: EMACROS Definition

First, the components must be added to the archive. This can be done using the SLIMxxxx procs (see Using SLIM on page 10). Next the dependencies must be defined using the SLIMDEP proc (see page 16).

First, look through all the code for any copy code. These would be candidates for sub projects. For example, I have some user macros I wrote and use in almost all the assembler programs. These macros would be candidates for a sub project.

ITEMS That Require Special Handling

The CLEANUP assembler program requires macros that are not in SYS1.MACLIB, an additional PDS SYS1.AMODGEN is included in the SLIM procs for a clean assembly.

Any PL/I (F) program such as SLIMXREF that uses the internal sort feature will have a link edit return code of 4. Normally the return code (rc) other than 0 indicates a problem with the link edit and proc SLIMCLS will abend. The RC of 4 is is caused by the way a PL/I runtime module was link edited. However, the program will still execute correctly. A SLIM type of ‘PLIS’ is included to handle this situation. With an item with type PLIS is compiled, SLIMCLS will not abend with this condition.

Customizing SLIM

The full source code for SLIM and all modules it depends on are supplied. SLIM is designed for easy customization. In order to customize, you must understand how SLIM works.

In this section, it is assumed that all dataset names start with an * refer to HERC01.SLIM.

Each member stored has a module (member) name, a project name, a description and a type. Member names and project names are not validated. The description and type are validated using a table contained in a module call SLIMDDN. See appendix A for the table.

Customizing SLIMAUTH Table

SLIM has the ability to dynamical allocate ARCHIVER files, it has a builtin "security" system. All executions of SLIM require a HLQ (high level qualifier) and a USERID. HLQ is usually a TSOID or catalog alias. USER is a unique name assigned to each user authorized to access SLIM archives. It can be any combination of up to 8 characters. There is a module called SLIMAUTH that verifies the HLQ and USERID to determine the dataset name of the ARCHIVE to be accessed.

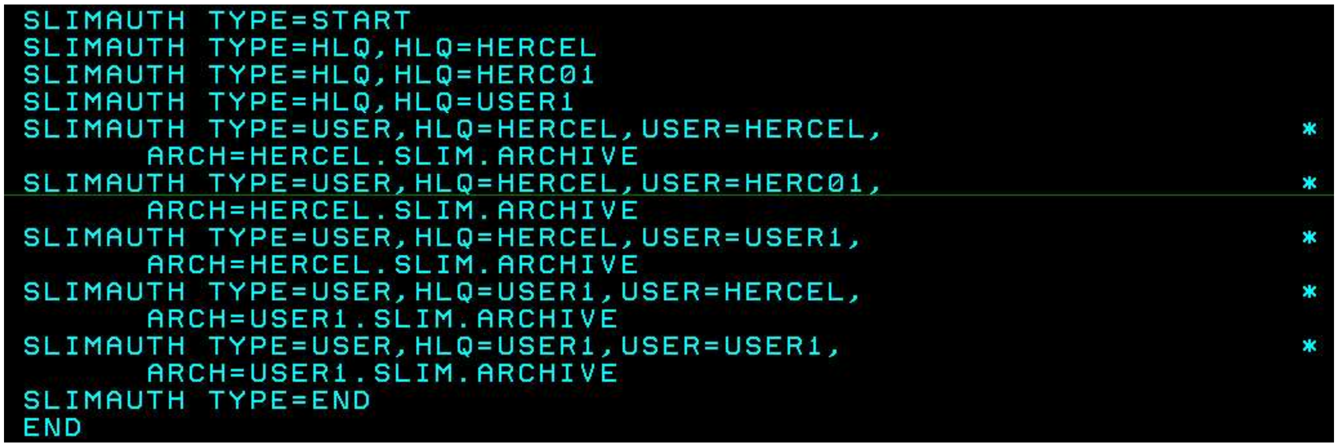
Below is a screen print of the SLIMAUTH module contained in the distribution.

Comments may be specified with an "*" in column 1. Column 72 is used to indicate continuations.

Each line begins with "SLIMAUTH TYPE=" followed by a keyword.

TYPE=HLQ is defining the list of valid HLQ. In this example, 3 HLQ are defined.

TYPE=USER defines userids, associates the userid with a HLQ and species the dataset name of the ARCHIVE. In the first TYPE=USER, the HLQ HERCEL is related to USER HERCEL. This combination results in the ARCHIVE HERCEL.SLIM.ARCHIVE being accessed. In the second TYPE=USER, the HLQ HERCEL is related to USER HERC01. This combination results in the ARCHIVE HERCEL.SLIM.ARCHIVE being accessed.



```
SLIMAUTH TYPE=START
SLIMAUTH TYPE=HLQ,HLQ=HERCEL
SLIMAUTH TYPE=HLQ,HLQ=HERC01
SLIMAUTH TYPE=HLQ,HLQ=USER1
SLIMAUTH TYPE=USER,HLQ=HERCEL,USER=HERCEL,                          *
      ARCH=HERCEL.SLIM.ARCHIVE
SLIMAUTH TYPE=USER,HLQ=HERCEL,USER=HERC01,                          *
      ARCH=HERCEL.SLIM.ARCHIVE
SLIMAUTH TYPE=USER,HLQ=HERCEL,USER=USER1,                           *
      ARCH=HERCEL.SLIM.ARCHIVE
SLIMAUTH TYPE=USER,HLQ=USER1,USER=HERCEL,                           *
      ARCH=USER1.SLIM.ARCHIVE
SLIMAUTH TYPE=USER,HLQ=USER1,USER=USER1,                            *
      ARCH=USER1.SLIM.ARCHIVE
SLIMAUTH TYPE=END
END
```

Customize the SLIMDDN Table

Listed below is the SLIMDDN table as distributed. Note there is a group of entries that are defaulted and should not be changed. There is special logic in SLIM that is triggered by these entries. If you do need to change them, the extract module in SLIM may need to be modified.

```
         TITLE '*** SLIMDDN V1.0.0 ***'
***********************************************************************
*                                                                     *
*    SLIMDDN - SLIM DDNAME TABLE                                      *
*                                                                     *
*    THIS MODULE ACCEPTS THE SLIM DESCRIPTION AND TYPE TO DETERMINE   *
*    THE IF THE COMBINATION IS VALID AND IF SO THE DDNAME FOR JCL.    *
*                                                                     *
*    ====================> NOTICE <=====================              *
*    THERE IS A GROUP OF SPECIAL DESC/TYPE COMBINATION THAT ARE       *
*    REQUIRED BY SLIM.  THESE ARE GENERATED BY DEFAULT.  IF ANY       *
*    OF THESE ARE CHANGED, THE SLIM MODULE MAY NEED TO BE MODIFIED.   *
*    THESE CANNOT BE OVER RIDEN.                                      *
*                                                                     *
*      THE GROUP IS:                                                  *
*          COPYLIB  COB   COPYCOB                                     *
*          COPYLIB  ASM   COPYASM                                     *
*          INCLLIB  PLIS  INCLLIB                                     *
*          MACRO    ASM   MACLIB                                      *
*          SOURCE   PLIS  PLI                                         *
*          SUBPGM   PLIS  PLI                                         *
*                                                                     *
***********************************************************************
*
         SLIMDDN TYPE=START
*
         SLIMDDN DESC=INSTALL,DTYPE=CNTL,DDN=JCL
         SLIMDDN DESC=LOADLIB,DTYPE=EXEC,DDN=LOADLIB
         SLIMDDN DESC=LOADLIB,DTYPE=NCAL,DDN=NCALIB
         SLIMDDN DESC=PROCLIB,DTYPE=JCL,DDN=PROCLIB
*
         SLIMDDN DESC=SOURCE,DTYPE=ASM,DDN=ASM
         SLIMDDN DESC=SOURCE,DTYPE=COB,DDN=COB
         SLIMDDN DESC=SOURCE,DTYPE=FOR,DDN=FOR
         SLIMDDN DESC=SOURCE,DTYPE=PAS,DDN=PAS
         SLIMDDN DESC=SOURCE,DTYPE=PLI,DDN=PLI
*
         SLIMDDN DESC=MAP,DTYPE=ASM,DDN=MAP
         SLIMDDN DESC=PANEL,DTYPE=DATA,DDN=PANEL
         SLIMDDN DESC=PANEL5,DTYPE=DATA,DDN=PANEL5
         SLIMDDN DESC=SOURCE5,DTYPE=ASM,DDN=SOURCE5
*
         SLIMDDN DESC=SUBPGM,DTYPE=ASM,DDN=ASM
         SLIMDDN DESC=SUBPGM,DTYPE=COB,DDN=COB
         SLIMDDN DESC=SUBPGM,DTYPE=FOR,DDN=FOR
         SLIMDDN DESC=SUBPGM,DTYPE=PAS,DDN=PAS
         SLIMDDN DESC=SUBPGM,DTYPE=PLI,DDN=PLI
*
         SLIMDDN DESC=SYSOUT,DTYPE=LISTING,DDN=LISTING
         SLIMDDN DESC=SYSOUT,DTYPE=REPORT,DDN=REPORTS
         SLIMDDN DESC=DOCUMENT,DTYPE=TEXT,DDN=DOC
         SLIMDDN DESC=TEXT,DTYPE=TEXT,DDN=TEXT
*
         SLIMDDN TYPE=END
         END
```

The DDN items must have corresponding DD statements in the SLIMEXT2 proc.

Except for the TYPE= entries, the order of the entries is not important. “*” in column 1 is a comment.

Words for the DESC= and DTYPE= should be limited to 10 or less characters. DDN should be 7 or less characters.

Each DESC describes the TYPE. For example, DESC=SOURCE,TYPE=ASM is used for assembler programs that are executable. DESC=SUBPGM, TYPE=ASM is used for assembler sub programs that are called by other programs.

Keep in mind if you decide to change this table. This table is used to validate combinations of DESC and TYPE code. If also control the order in which the JCL for SLIM project compiles are generated.

1. all copy code, include code, and macros are extracted first
2. all sub programs are compiled and link edited
3. all main programs are compiled and link edited
4. all other code

The source code can be found in “SLIM.ASM(SLIMDDN)”. JCL to assemble can be found in “SLIM.SAMPLE(MAKEDDN)”.