Adding DASD Volumes

May, 2020

In order to create new DASD volumes on your MVS system, you need to complete the steps below. **Note:** Hercules must be executing with MVS before completing these steps. You will need to use an IBM utility to initialize the newly created DASD volume, and you will need to modify a parameter file on MVS to add the new volume.

1. create a DASD image in a file on the host Operating System,
2. attach the DASD image to your Hercules' configuration (attach command at the command prompt on the Hercules console), ensuring that the DASD device type generated for the address where you attach the image matches the type of image you created,
3. initialize the DASD volume for use by MVS (ICKDSF or ICKDSF13, depending upon DASD type),
4. vary the volume online (VARY command on the MVS console),
5. mount the volume with the appropriate Storage Use Class,
6. edit SYS1.PARMLIB(VATLST00) to add the new volume with the proper Storage Use Class, and
7. edit your Hercules configuration file to add the new volume to the device definition statements.

Creating the DASD image (item 1 above)

To create the DASD image, you will need to execute the **dasdinit** program in a terminal window (Linux) or Command Prompt window (Windows). The dasdinit executable is included in the Hercules distribution package. The command format is:

```
dasdinit -a <filename> <devtype> <volser>
```

The **-a** parameter instructs dasdinit to create alternate tracks in the DASD image. Alternate tracks on physical devices are used for error recovery when bad tracks are encountered during I/O operations. Although alternate tracks are not utilized under Hercules' emulation, most MVS utilities and programs will not function properly, if at all, without the presence of the alternate tracks.

The **<filename>** parameter provides the name of the file on the host operating system (Linux, Windows, Mac OS) in which the DASD image will be written. Note that dasdinit will not overwrite an existing file, so the name specified must not refer to an existing file.

The filename is not required to conform to any particular format; the name you choose will be related to the device address that MVS will use the volume by the entry in the Hercules configuration file. However

there are a couple of conventions that are typically followed in the Hercules community:

DASD volume image file names in the form: VVVVVV.aaa

- where VVVVVV is the Volume Serial Number by which MVS identifies the volume, and which is written to the internal file label in the volume image, and
- aaa is the device address (in hexadecimal) at which the image file is accessed by MVS.

or file names in the form VVVVVV.tttt

- where VVVVVV is the Volume Serial Number by which MVS identifies the volume, and which is written to the internal file label in the volume image, and
- tttt is the DASD device type, i.e. 2314, 3330, 3350, etc.

Whatever naming convention you choose to use for the host operating system file is up to you.

The **<devtype>** parameter specifies the type of DASD image: 2314, 3330, 3350, etc. Some of the device types have several different models, which are specified differently for the dasdinit utility and in the UNIT parameter on an MVS DD statement:

Device and Model Description	dasdinit model specification	MVS UNIT specification	Volume Capacity (bytes)
3330 model 1 (404 data cylinders)	3330	3330	100,018,280
3330 model 2 (808 data cylinders)	3330-2	3330-1	200,036,560
3340 model 35 (348 data cylinders)	3340	3340	34,944,768
3340 model 70 (696 data cylinders)	3340-70	3340	68,889,536
3380 (885 data cylinders)	3380	3380	630,243,900
3380-E (1,770 data cylinders)	3380-E	3380	1,260,487,800
3390	3390	3390	946,005,480

There are other DASD types, but these are the most commonly used. I have only included types where the volume capacity will not break MVS because the size is beyond what MVS 3.8j can address.

The **<volser>** parameter specifies the volume serial number written to the DASD image and provides the means for MVS to uniquely identify each volume to differentiate it from other DASD volumes. There must not be multiple DASD volumes visible (online) to MVS with the same volume serial number. Since the initialization process (task three in the list above) will rewrite the volume serial number, I usually specify SCRTCH or 111111 as the volume serial number to dasdinit, and then assign the volume serial number that I plan to use for the volume during the initialization on the MVS system.

An extensive table of DASD geometry information, which includes the number of data and alternate

cylinders, is available @ IBM Mainframe DASD Capacity.pdf. The information in this table was compiled from a post in the (now defunct) Yahoo group hercules-390, message #2833 in June, 2000 and from a file available in the hercules-390 Yahoo group files section. Beginning with version 2.17 of Hercules, it is not really necessary to know the number of cylinders present in each of the DASD types, but these tables are available if you would like to know how much storage is available for each of the types.

Attach the DASD image to Hercules (item 2 in the list at the top of this page)

In order to attach an emulated device to Hercules while it is running, you will need to be in the text console display because there is no equivalent command in the graphical display. The command syntax is:

```
attach <address> <devtype> <filename> [cu=3880]
```

The **<address>** parameter specifies the hardware address at which the device is to be attached and consists of the Channel and Unit Address (cuu). The address you use will depend upon the configuration that was specified during System Generation and must be a previously unused address generated for the type of device you are attempting to attach (ie, if the device you are attempting to attach is a 3350 DASD, the address you use must have been generated for a 3350 DASD).

The **<devtype>** parameter specifies the type of DASD image: 2314, 3330, 3350, etc. For the 3330 (also 3340 and 3380, both of which has a sub-model with very different capacities), it doesn't seem to matter to Hercules whether the DASD image specified with <devtype> 3330 is the 3330-1 or 3330-11 (or <devtype> 3340 is the 3340-35 or 3340-70; or <devtype> 3380 is the 3380 or 3380-E); MVS does just fine with a plain old 3330 (or 3340 or 3380) specified here.

The **<filename>** parameter provides the name of the file on the host operating system in which the DASD image is contained (where it was written by dasdinit). Note that if you are using a directory structure to organize your DASD files in a subdirectory beneath the directory where Hercules was started, you will need to include a path designation. Since my DASD image files are contained in the **dasd** subdirectory under the **mvs** directory where I start Hercules, my filenames are in the form of:

dasd/<filename>

The separator characters in the path name may be either a forward slash (/) or a backward slash (\).

The **cu=3880** parameter is required for 3390 type DASD and specifies that the device controller to be emulated for the device should be a model 3880. The default device controller chosen by Hercules will cause some errors from MVS utilities.

Initialize the DASD image for use by MVS (item 3 in the list at the top of this page)

Although the dasdinit program creates the raw DASD image, MVS requires additional information that is not written by dasdinit. Each DASD volume needs a Volume Header label, which contains the Volume Serial Number; this is written by dasdinit. It also needs a group of dataset control blocks (DSCBs), which exist in ten different formats (DSCB0 through DSCB9) and whose number will vary depending upon

several factors. Collectively these DSCBs make up what is generally referred to as the Volume Table of Contents, which functions much like the contents section of a book, pointing to other sections of the VTOC (the DSCBs) and, ultimately to the user files that will be created on the volume. The utility program used under MVS to create the VTOC is ICKDSF, also known as Device Support Facilities. ICKDSF must be used to initialize a new volume before it is placed online for use by MVS.

The version of ICKDSF that installs by default with MVS 3.8 is Release 6. That version of ICKDSF will not initialize DASD types later than 3350 (3375, 3380, or 3390 device types). However, included with the MVS 3.8j Distribution tapes, there is a separate tape which contains Release 13 of ICKDSF. The catch is that Release 13 will not initialize 3340 or 3350 device types without reporting an error; the error doesn't seem to hinder the initialization, but it is disconcerting to see.

If you have followed my instructions for installing MVS 3.8j, you will have both versions of ICKDSF on your system. If your system was not built following my instructions, you can grab the jobstream fdz1d02 from the mvsInstallationResources archive and the product tape from the vs2.DistributionTapes archive and run that job to install ICKDSF Release 13 into your SYS1.LINKLIB under the name ICKDSF13, which will happily reside alongside the Release 6 version which is of course stored under the name ICKDSF. Then it is simply a matter of modifying the EXEC statement in the job you use to initialize DASD volumes to ensure that you are executing the correct version for the type of DASD you intend to initialize.

In order to use either version of ICKDSF, the volume(s) to be initialized must be offline. If you have dynamically added the volume to Hercules while MVS is already running under Hercules, the volume is 'offline' by default.

Using ICKDSF

The JCL to initialize a 3340 or 3350 DASD volume, using ICKDSF Release 6, is:

```
//ICKDSF  JOB (1),ICKDSF,CLASS=A,MSGCLASS=X
//ICKDSF EXEC PGM=ICKDSF,REGION=4096K
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
  INIT UNITADDRESS(<address>) NOVERIFY VOLID(<volser>) OWNER(HERCULES) -
               VTOC(0,1,<extent>)
/*
//
```

The JCL to initialize any type of DASD volume other than 3340 or 3350, using ICKDSF Release 13, is:

```
//ICKDSF  JOB (1),ICKDSF,CLASS=A,MSGCLASS=X
//ICKDSF EXEC PGM=ICKDSF13,REGION=4096K
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
  INIT UNITADDRESS(<address>) NOVERIFY VOLID(<volser>) OWNER(HERCULES) -
               VTOC(0,1,<extent>)
/*
//
```

The <address> parameter in the control card specifies the address at which the DASD image to be initialized is attached to Hercules. The device at that address must be offline to MVS.

Note: I show the syntax for the INIT command using NOVERIFY. This is somewhat risky and a practice I do not actually follow. Instead, when I use the Hercules dasdinit utility to create the raw DASD image, I

specify a value for the Volume Serial Number that is different from any valid volume currently online. Usually I begin with 111111 (digit 1 repeated six times) and increment the digit value for each new volume I am creating at a particular time (i.e. if I am creating two new volumes, they will be created by dasdinit as having the serial numbers 111111 and 222222). Then the syntax I use in the INIT command instead of NOVERIFY is VERIFY(<existing volser>) , and substitute the value I had dasdinit write in the new volume for <existing volser>. This is a final check to ensure that ICKDSF is initializing the volume that I intend to be initialized.

The <volser> parameter specifies the volume serial number to be written to the DASD volume.

The <extent> sub-parameter specifies how many tracks to allocate to the VTOC. I typically allocate an entire cylinder to the VTOC, which is more than adequate and may actually be overkill. There is a formula for calculating the minimum number of tracks to allocate for an intended number of non-VSAM datasets and VSAM data spaces you intend to create on the volume, but the formula is complex and varies depending upon the DASD device type. For each non-VSAM dataset and VSAM dataspace you allocate, there are a certain number of DSCBs that will be created, so you need adequate space to accommodate the number of datasets that will be recorded in the VTOC. If you do not allow enough space for the VTOC, you will eventually run out when you have allocated as many datasets as the VTOC can hold.

The other positional sub-parameters in the VTOC parameter specify the cylinder and head, respectively, where the VTOC is to start. On physical disks, it was common practice to locate the VTOC in the middle of the volume for performance considerations. But under Hercules it is acceptable to place the VTOC at track 1. Note that track 0 is reserved for the volume serial number and so must not be used as the VTOC beginning address.

The JCL should be modified to meet your requirements and submitted to MVS. You will be asked to confirm that you actually want to initialize the volume at the address specified:

```
*06 ICK003D REPLY U TO ALTER VOLUME 253 CONTENTS, ELSE T
```

The reply of **U** allows the initialization to proceed. There will be a number of informational messages printed in the SYSPRINT output during the executing of ICKDSF. The most important thing to verify is that the return code for the job is 0000.

Vary the volume online (VARY command on the MVS console) (item 4 in the list at the top of this page) and Mount the volume with the appropriate Storage Use Class (item 5 in the list at the top of this page)

After the volume is initialized, it must be placed online before MVS will be able to allocate the volume to allow jobs to create datasets on it. On the MVS console issue the command:

```
v <address>,online
```

The **<address>** parameter specifies the address of the DASD device to be placed in online status. Multiple devices may be varied online by specifying more than one address as a list of addresses, separated by commas and enclosed in parentheses: v(131,132,133),online

When the volume is placed online to MVS, it has the default storage use class of **STORAGE**, which is the

least restrictive of the three possible storage use classes. The possible values for storage use class are:

- **PRIVATE** - new datasets will be created on this volume only if the user (via JCL or the TSO ALLOCATE command) specifies the volume serial of this disk volume.
- **PUBLIC** - MVS may place a temporary dataset on this volume, if the user (via JCL or otherwise) did not specify a volume serial for the temporary dataset. Datasets may also be placed on the volume by specifying the volume serial, as with PRIVATE volumes. A temporary dataset is one with a DSNAME beginning with an ampersand and/or with a disposition equivalent to NEW,DELETE.
- **STORAGE** - MVS places permanent datasets on this volume if the user did not supply a volume serial for the datasets. In addition, temporary datasets and datasets placed by volume serial may also be placed on the volume.

It is always best to follow the VARY command with a MOUNT command, to explicitly set the storage use class for the new DASD volume just placed online. On the MVS console issue the command:

m <address>,vol=(sl,<volser>),use=<useclass>

The **<address>** parameter specifies the address of the DASD device to be placed in online status.

The **<volser>** parameter specifies the volume serial number of the volume.

The **<useclass>** parameter should specify STORAGE, PUBLIC, or PRIVATE.

Edit SYS1.PARMLIB(VATLST00) to add the new volume with the proper Storage Use Class (item 6 in the list at the top of this page)

If this is to be a permanent addition to your configuration, you should add the volume to the VATLST member in SYS1.PARMLIB.

The VATLST00 member of SYS1.PARMLIB controls the mounting of DASD volumes at IPL. Each line of this member contains the information pertaining to a single DASD volume. You will need to add an entry (a line containing five parameters) for each DASD volume you add to your system.

If you have TSO installed on your system, you can easily edit this member to make the addition. If you do not have TSO installed, you will have to use the MVS utility IEBUPDTE to make the changes. I will describe the process to use if TSO is not installed as that is the more difficult process. First you need to find out the current contents of VATLST00 and for that you can use the IEBPTPCH utility:

```
//IEBPTPCH JOB CLASS=A,MSGLEVEL=(1,1)
//IEBPTPCH EXEC PGM=IEBPTPCH
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  DSN=SYS1.PARMLIB(VATLST00),DISP=SHR
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  *
  PRINT MAXFLDS=1
  RECORD FIELD=(80)
/*
//
```

For my system, this produces the following output:

```
MVSRES,0,2,3350    ,Y        SYSTEM RESIDENCE (PRIVATE)                00010000
MVS000,0,2,3350    ,Y        SYSTEM DATASETS (PRIVATE)                 00020000
MVS370,0,0,3375    ,N        STORAGE                                   00030000
MVS380,0,0,3380    ,N        STORAGE                                   00040000
MVSSRC,0,2,3380    ,N        MVS SOURCE (OPTIONAL MATERIALS)           00050000
PAGE00,0,2,3350    ,Y        PAGE DATASETS (PRIVATE)                   00060000
PUB000,1,2,3380    ,N        PUBLIC DATASETS (PRIVATE)                 00070000
PUB001,1,2,3390    ,N        PUBLIC DATASETS (PRIVATE)                 00080000
SMP000,1,2,3350    ,N        DISTRIBUTION LIBRARIES (PRIVATE)          00090000
SORTW1,1,1,2314    ,N        SORT WORK (PUBLIC)                        00100000
SORTW2,1,1,2314    ,N        SORT WORK (PUBLIC)                        00110000
SORTW3,1,1,2314    ,N        SORT WORK (PUBLIC)                        00120000
SORTW4,1,1,2314    ,N        SORT WORK (PUBLIC)                        00130000
SORTW5,1,1,2314    ,N        SORT WORK (PUBLIC)                        00140000
SORTW6,1,1,2314    ,N        SORT WORK (PUBLIC)                        00150000
SPOOL1,0,2,3350    ,Y        JES2 QUEUES (PRIVATE)                     00160000
SYSCPK,1,2,3350    ,N        COMPILER/TOOLS (PRIVATE)                  00170000
WORK00,1,0,3350    ,N        WORK PACK (STORAGE)                       00180000
WORK01,1,0,3350    ,N        WORK PACK (STORAGE)                       00190000
```

Note that the numbers in columns 73 through 80 are sequence numbers.

The parameters specified for each volume are:

- Columns 1 through 6 (six characters) contain the volume serial number.
- Column 8 (one character) is 0 for permanently resident volumes, and 1 for reserved volumes. These states are no longer very relevant, and you can use 0 in most cases.
- Column 10 (one character) is the storage use class attribute. STORAGE is denoted by 0, PUBLIC by 1, and PRIVATE by 2.
    - PRIVATE - new datasets will be created on this volume only if the user (via JCL or the TSO ALLOCATE command) specifies the volume serial of this disk volume.
    - PUBLIC - MVS may place a temporary dataset on this volume, if the user (via JCL or otherwise) did not specify a volume serial for the temporary dataset. Datasets may also be placed on the volume by specifying the volume serial, as with PRIVATE volumes. A temporary dataset is one with a DSNAME beginning with an ampersand and/or with a disposition equivalent to NEW,DELETE.
    - STORAGE - MVS places permanent datasets on this volume if the user did not supply a volume serial for the datasets. In addition, temporary datasets and datasets placed by volume serial may also be placed on the volume.
- Columns 12 through 19 (eight characters, left justified) is the device type. MVS will probe the device at IPL time and if the device it finds with this volume serial number is not the device type specified here, MVS will dismount the volume. If you have the next parameter as Y and this happens, MVS will stop until you mount the correct volume type with this volume serial number; an entry of N will allow MVS proceed after dismounting the incorrect volume.
- Column 21 (one character) indicates whether the operator should be requested to mount the volume if it is not found during IPL. In most cases you should specify N (for NO) in this column.
- Columns 23 through 71 may be used for comments. There are some CBT tape programs that expect

to find the comments beginning in column 30, so that is the format I follow.

A detailed discussion of the VATLST member may be read at https://www.tommysprinkle.com/mvs/vatlst00.htm.

In order to add a new 3350 work volume to the VATLST00 entries shown above using IEBUPDTE, the following job would be used:

```
//IEBUPDTE JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//IEBUPDTE EXEC PGM=IEBUPDTE,REGION=1024K
//*
//SYSUT1   DD  DSN=SYS1.PARMLIB,DISP=SHR
//SYSUT2   DD  DSN=SYS1.PARMLIB,DISP=MOD
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
./ CHANGE NAME=VATLST00,LIST=ALL
WORK03,1,0,3350    ,N        WORK PACK (STORAGE)                        00200000
./ ENDUP
//
```

The sequence number of the card to be added (columns 73 through 80) is 00200000, which directs IEBUPDTE to place this new line following the lines already present in the VATLST00 member.

At the next IPL after submission of this job, MVS will look for a 3350 volume with the volume serial number WORK03 and the volume will be made available as a STORAGE volume.

Edit your Hercules configuration file to add the new volume to the device definition statements (item 7 in the list at the top of this page)

Use a text editor to edit the Hercules configuration file (located in the **conf** directory under the **mvs** directory if you have followed by file structure) to add the line in the device entries list to ensure that the next time Hercules is started with the configuration file to IPL MVS the new DASD volume will be available to MVS.

Here (provided only as an example) is the section of my Hercules configuration file for running MVS that contains DASD volume definitions:

```
# ------------------------------------------------ 3350 on Channel 1
0150    3350    dasd/mvsres.3350                 # PRIVATE
0151    3350    dasd/mvs000.3350                 # PRIVATE
0152    3350    dasd/page00.3350                 # PRIVATE
0153    3350    dasd/spool1.3350                 # PRIVATE
#------------------------------------------------ 3380 on Channel 1
0180    3380    dasd/pub000.3380                 # PRIVATE
#------------------------------------------------ 3390 on Channel 1
0190    3390    dasd/pub001.3390 cu=3880         # PRIVATE
#------------------------------------------------ 2314 on Channel 2
0220    2314    dasd/sortw1.2314                 # PUBLIC
0221    2314    dasd/sortw2.2314                 # PUBLIC
0222    2314    dasd/sortw3.2314                 # PUBLIC
0223    2314    dasd/sortw4.2314                 # PUBLIC
0224    2314    dasd/sortw5.2314                 # PUBLIC
0225    2314    dasd/sortw6.2314                 # PUBLIC
#------------------------------------------------ 3350 on Channel 2
```

```
0250    3350    dasd/smp000.3350                # PRIVATE
0251    3350    dasd/work00.3350                # STORAGE
0252    3350    dasd/work01.3350                # STORAGE
0253    3350    dasd/syscpk.3350                # PRIVATE
#------------------------------------------- 3375 on Channel 3
0370    3375    dasd/mvs370.3375                # STORAGE
#------------------------------------------- 3380 on Channel 3
0380    3380    dasd/mvs380.3380                # STORAGE
0381    3380    dasd/mvssrc.3380                # PRIVATE
```

The syntax for the device definition is:

<address> <devtype> <filename> [cu=3880]

Note that I liberally use spaces and comments to make the Hercules configuration file more easily read and to remind me of what storage use class DASD volumes are assigned in the VATLST00 parameter.

The tasks above are the steps required to add a new DASD volume to MVS running under Hercules. The tasks I continue with below are often steps that will make the DASD volumes more useful to you as you create/access datasets on DASD volumes under MVS.

Set Up User Catalogs

Generally, when you are adding new storage space to the system, it is also a good time to think about how that storage space will integrate with the catalog structure you have in place for the system. This is an overview of the User Catalogs I have set up on my MVS 3.8j system.

During the System Generation, a VSAM Master Catalog was created. It resides on MVSRES and the dataset name of the catalog itself is SYS1.VSAM.MASTER.CATALOG. Although you can continue to catalog any new datasets you create in the Master Catalog, it is not considered a good practice and would undoubtedly not be allowed in any *real world* shop.

In my experience, shops set up user catalogs that group datasets either by project - payroll, accounts

receivable, budget, etc - or by location, i.e. the DASD volume on which the dataset resides determines the user catalog into which the dataset is catalogued. The structure of catalogs you set up for your system is up to you, but I will describe here the way I have set mine up as a guide. Keep in mind that what I am documenting here is relevant to the status of my system as I am currently rebuilding it (and updating this documentation, as well) and may not exactly match the configuration referred to and/or documented in other pages on this site. Nor will it stay static for me ... it will change and grow as I add volumes. This is only to show you how to go about setting your own system up.

First, I should say that "behind the scenes" I have created the following DASD volumes in addition to those that were created as I went through the processes described in Installing MVS 3.8j:

MVS370 - 3375 for system storage
MVS380 - 3380 for system storage
MVSSRC - 3380 containing the MVS 3.8j optional materials
SYSP01 - 3380-E for all the datasets related to setting up my system, which I preserve across system rebuilds

During the last couple of revisions of my installation instructions/tutorial, I decided to create two new User Catalogs - UCPUB000 into which all datasets created on PUB000 will be catalogued, and UCPUB001 into which all datasets created on PUB001 will be cataloged. The 'Systems Programming' volume - SYSP01 - has its own User Catalog (UCSYSP01 into which all datasets residing on SYSP01 are cataloged; and the MVSSRC volume, which contains the optional materials for MVS 3.8j also has its own User Catalog. As you can see, this leaves some volumes not specifically planned for, but a JOBCAT or STEPCAT card can be utilized to select the catalog explicitly, overriding automatic catalog selection based upon Alias information. The important thing that this process accomplishs is to set up a means to control unchecked use of the Master Catalog for all user datasets.

The names I have chosen for the user catalogs reflect mnemonically that they are user catalogs - UC as the first two characters of the name - and indicate the connection to the datasets which will be cataloged within them with the remainder of the name, which refers to the Volume Serial of the DASD device on which the datasets reside that are catalogued in that particular User Catalog. The user catalogs themselves will be cataloged in the Master Catalog.

This is the jobstream that I used to create the User Catalogs and set up Aliases for them:

```
//DEFUCATS JOB 'JAY MOSELEY',CLASS=A,MSGLEVEL=(1,1),MSGCLASS=A
//IDCAMS EXEC PGM=IDCAMS,REGION=4096K
//SYSPRINT DD SYSOUT=*
//PUB000 DD UNIT=3380,VOL=SER=PUB000,DISP=OLD
//PUB001 DD UNIT=3390,VOL=SER=PUB001,DISP=OLD
//SYSIN DD *

  /* This User Catalog will contain NON-VSAM datasets that     */
  /* reside on volume PUB000.                                  */

  DEFINE USERCATALOG ( -
         NAME (UCPUB000) -
         VOLUME (PUB000) -
         CYLINDERS (20) -
         FOR (9999) -
         BUFFERSPACE (8192) )

  /* An Alias is defined so that all datasets with a high-     */
  /* level qualifier of PUB000 will be catalogued in the       */
  /* User Catalog UCPUB000.                                    */

  DEFINE ALIAS ( -
         NAME (PUB000) -
```

```
        RELATE (UCPUB000) )

  /* This User Catalog will contain NON-VSAM and VSAM objects  */
  /* that reside on volume PUB001.  Half of the volume will be */
  /* allocated at the same time for use as a VSAM Dataspace.   */
  /* The User Catalog is sub-allocated from that Dataspace.    */

  DEFINE USERCATALOG ( -
         NAME (UCPUB001) -
         VOLUME (PUB001) -
         CYLINDERS (556) -
         FOR (9999) -
         BUFFERSPACE (8192) ) -
           DATA (CYLINDERS (30) ) -
           INDEX (CYLINDERS (15) )

  /* An Alias is defined so that all datasets and VSAM objects */
  /* with the high-level qualifier of PUB000 will be           */
  /* catalogued in the User Catalog UCPUB001.                  */

  DEFINE ALIAS ( -
         NAME (PUB001) -
         RELATE (UCPUB001) )

/*
//
```

The above job is include in the mvsInstallationResources archive as job MVS01. If you have followed my installation instructions, these catalogs and aliases are already defined on your MVS system.

When I bring a volume into the system from a prior functioning system that has its own User Catalog on the volume, I use the following jobstream to import that catalogs and set up Aliases to it:

```
//UCSYSP01 JOB (SYS),'IMPORT UCAT SYSP01',CLASS=S,MSGCLASS=X
//IDCAMS01 EXEC PGM=IDCAMS,REGION=4096K
//SYSPRINT DD  SYSOUT=*
//SYSP01   DD  UNIT=3380,DISP=OLD,VOL=SER=SYSP01
//SYSIN    DD  *

  /* THERE IS A USER CATALOG IN EXISTENCE ON SYSP01 THAT       */
  /* CONTAINS CATALOG ENTRIES FOR THE DATASETS ON THAT VOLUME. */
  /* IT IS CONNECTED TO THE MASTER CATALOG AND AN ALIAS IS     */
  /* DEFINED TO RELATE FUTURE DATASETS REFERENCES TO IT.       */

  IMPORT CONNECT OBJECTS((UCSYSP01 VOLUME(SYSP01) DEVT(3380)))

  DEFINE ALIAS(NAME(SYSP) RELATE(UCSYSP01))

//
```

The 3380 volume SYSP01 has been in use on my system for a long time, although I have recently expanded it by moving the entire contents over to a 3380-E volume, which is the largest DASD volume that MVS 3.8j can handle because of size limitations in the control blocks built into MVS 3.8j.

There is a similar User Catalog set up on the MVSSRC volume, and an alias defined to relate the non-VSAM datasets catalogued on that volume so that they are allocated by MVS without the use of JOBCAT or STEPCAT DD statements.

Again, I emphasize that I am providing this as a guide to setting up your own User Catalogs. You will have to decide what User Catalogs you need for your system and prepare an appropriate jobstream using this example as a guide . If you are unfamiliar with VSAM Catalogs, I have written more about them in

my VSAM Tutorial.

Once you have created a plan for the structure of the catalogs on your system, and set up User Catalogs to control all of your non-system datasets, you should modify the Master Catalog so that any changes to the catalog, such as adding new entries or modifying existing entries, will require a password. The jobstream I used for that is:

```
//MASTERPW JOB 'JAY MOSELEY',CLASS=A,MSGLEVEL=(1,1),MSGCLASS=A
//IDCAMS EXEC PGM=IDCAMS,REGION=4096K
//SYSPRINT DD SYSOUT=A
//SYSIN DD *

 /* ALTER THE MASTER CATALOG TO REQUIRE A PASSWORD IN ORDER TO
    COMPLETE ANY UPDATE OPERATIONS                              */

   ALTER SYS1.AMASTCAT UPDATEPW(SYSPROG)

/*
//
```

The step accomplished by the above jobstream is also a part of job MVS01, so if you built your system following my installation instructions, your Master Catalog is already protected by a password. That password is **SYSPROG**.

The password, shown above in blue, may be from 1 to 8 alphanumeric characters of your choice. I want to repeat that adding password protection to the Master Catalog does not prevent you from adding (or deleting) datasets entries from it, but the password must be supplied by the operator (or in the jobstream) in order for the Master Catalog to be updated.

I hope that you have found my instructions useful. If you have questions that I can answer to help expand upon my explanations and examples shown here, please don't hesitate to send them to me:

Back to Previous Page

Return to Site Home Page

Frequently Asked Questions

This page was last updated on May 10, 2020 .