ABSTRACT
A Multiple Virtual Storage (MVS) data set must be allocated before it can be used during a SAS session. The MVS operating system provides two command languages, Job Control Language (JCL) and Time Sharing Option (TSO), which can be used to allocate data sets. JCL allocates data sets during batch mode processing, while TSO allocates data sets during interactive processing. The SAS system also provides tools capable of allocating data sets within the MVS environment. In fact, the SAS tools LIBNAME and FILENAME are better suited than their JCL and TSO counterparts to allocate data sets that will be used during a SAS session. This article provides examples of allocating new and existing data sets in the MVS environment, and shows how allocating a data set using a SAS tool is more efficient than allocating using either a JCL statement or a TSO command.

INTRODUCTION
The process of requesting access to a data set in the MVS environment is called allocation. During allocation, an association is formed between an assigned logical name and the data set. Once a data set is allocated, access to its resources is granted. If an existing data set is allocated, the system allows access to that data set. During creation of a new data set, allocation sets aside space for the newly created data set and grants access to that space.

Before an MVS data set may be used, or created, by a SAS program, it must be allocated. Allocation of the data set can be done in two ways: (1) outside of the SAS program using either a JCL statement or a TSO command, or (2) inside of the SAS program using the SAS tools LIBNAME and FILENAME. Both methods achieve the same result: an MVS data set is made available for use within a SAS program.

External allocation of a data set is done with either a JCL statement or a TSO command. JCL data definition (DD) statements are used to externally allocate data sets when running programs in the MVS background (batch mode), while TSO's ALLOCATE command is used to externally allocate data sets when running programs in the MVS foreground (interactive mode).

Internal allocation of a data set is dependent upon the type of data set being allocated. A SAS data set is allocated with a LIBNAME statement, while a non-SAS data set is allocated with a FILENAME statement. The non-SAS data sets created with BMDP, OSIRIS, and SPSSX are an exception. These data set formats are internally allocated with a LIBNAME statement.

When a data set's resources are no longer needed, the data set should be deallocated. Deallocating a data set discontinues the association between the assigned logical name and the data set name, and either deletes or stores the data set, depending on what is specified during allocation.

The MVS system does not distinguish between an externally allocated data set and an internally allocated data set. The SAS system does, however, and offers several advantages for choosing internal over external allocation. Chief among these is the ability to conditionally allocate data sets.

ALLOCATING A NEWLY CREATED DATA SET
In order to allocate a new data set in the MVS environment, the following data set attributes must be specified:

- data set name (DSNAME)
- data definition name (DDNAME)
- type of access desired
- storage space
- organization and record specifications
- data set disposition

The data set name, or DSNAME, is used by the system to identify the data set. An unqualified DSNAME may be one to eight characters in length and consist of alphanumeric characters, national characters (@, #, $), a hyphen, or a hexadecimal character. A qualified DSNAME is made up of unqualified names, or qualifiers, joined by periods. The maximum length for a fully qualified DSNAME is 44 characters, with two exceptions:

- 35 characters for a generation data set.
- 17 characters for an output tape data set.

The data definition name, or DDNAME, is defined during allocation and acts as a link between the data set and the program which will use the data set as input. A data set may have several different DDNAMEs assigned to it, however, the system only allocates it once. Before the data set may be deallocated, all DDNAMEs assigned to the data set must be cleared.

The type of access desired for the data set is determined by the data processing tasks to be performed. Since a new data set is being created, the type of access desired is NEW.
The amount of disk space required must be specified when creating a new data set. Primary and secondary space specifications may be made in terms of blocks, cylinders, or tracks. The primary space allocation value specifies how many units (blocks, cylinders, or tracks) to set aside for the new data set. The secondary space allocation value specifies how many units of space to acquire if the primary space allocation is too small.

The organization and record specifications of the new data set must also be specified during allocation. These attributes are:

- **DSORG** - Describes the organization of the data set. For example, physical sequential or partitioned.
- **RECFM** - Describes the characteristics of the records in the data set. For example, variable-length or fixed-length.
- **LRECL** - Describes the length in bytes of each record in the data set.
- **BLKSIZE** - Describes the number of bytes the system must read or write at one time.

The disposition of the new data set may be either **KEEP**, **DELETE**, or **CATLG**.

- **KEEP** - The data set will be retained by the system.
- **DELETE** - The data set will be deleted by the system.
- **CATLG** - The data set will be retained by the system and retrievability information will be stored.

Table 1 illustrates allocating a new data set in the MVS environment.

**ALLOCATING AN EXISTING DATA SET**

In order to allocate an existing data set in the MVS environment, the following data set attributes must be specified:

- data set name (DSNAME)
- data definition name (DDNAME)
- type of access desired
- data set disposition

Usage of the DSNAME and DDNAME data set attributes is the same whether allocating a new data set or an existing data set. Refer to the section "Allocating a Newly Created Data Set" for details.

The type of access desired for the data set is determined by the data processing tasks to be performed. The three types of access allowed for existing data sets are:

- **OLD** - Provides exclusive access to the data set, preventing others from accessing the data set until it is released. This disposition is used when modifying a data set.
- **MOD** - Provides exclusive access to the data set, preventing others from accessing the data set until it is released. This disposition is used to append data to the data set.
- **SHR** - Provides non-exclusive access to the data set, allowing others to access the data set simultaneously. This disposition is used when no modifications are made to the data set.

The disposition of a data set determines what will happen after it is deallocated. The disposition of an existing data set can be either **KEEP** or **DELETE**. KEEP is the default for existing data sets. KEEP and DELETE are described in detail under the section "Allocating a Newly Created Data Set".

Table 2 illustrates how to allocate an existing data set in the MVS environment.

**TABLE 2. Allocating an Existing Data Set**

<table>
<thead>
<tr>
<th><strong>External Allocation (JCL DD statement)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>IN DD DSN=EXAMPLE.DAT,DISP=(OLD,KEEP)</code></td>
</tr>
</tbody>
</table>

**Internal Allocation (SAS data set)**

| `LIBNAME IN 'EXAMPLE.DAT' DISP=(OLD,KEEP);` |

**Internal Allocation (non-SAS data set)**

| `FILENAME IN 'EXAMPLE.DAT' DISP=(OLD,KEEP);` |

**NOTE:** The data set name, or DSNAME, in the table is EXAMPLE.DAT; the data definition name, or DDNAME, is IN; the type of access desired is OLD; and the disposition of the data set is KEEP.

**TABLE 1. Allocating a Newly Created Data Set**

<table>
<thead>
<tr>
<th><strong>External Allocation (JCL DD statement)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>IN DD DSN=EXAMPLE.DAT,DISP=(NEW,CATLG),SPACE=(TRK,(20,10)),DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3200</code></td>
</tr>
</tbody>
</table>

**External Allocation (TSO command)**

| `ALLOCATE DATASET (EXAMPLE.DAT FILE(IN) OLD KEEP)` |

**Internal Allocation (SAS data set)**

| `LIBNAME IN 'EXAMPLE.DAT' DISP=(NEW,CATLG) SPACE=(TRK,(20,10)) BLKSIZE=3200;` |

**Internal Allocation (non-SAS data set)**

| `FILENAME IN 'EXAMPLE.DAT' DISP=(NEW,CATLG) SPACE=(TRK,(20,10)) DSORG=PS RECFM=FB LRECL=80 BLKSIZE=3200;` |

**NOTE:** The data set name, or DSNAME, in the table is EXAMPLE.DAT; the data definition name, or DDNAME, is IN; the type of access desired is NEW; the amount of disk space is 20 tracks; the amount of additional space to be allocated if more space is needed is 10 tracks; the organization is physical sequential; the record format is fixed block; the record length is 80; the number of bytes in a block of records is 3200; and the disposition of the data set is CATLG.
DEALLOCATING A DATA SET

When a data set's resources are no longer needed, the data set should be deallocated. Releasing a data set from a previous allocation depends on how it was initially allocated. A data set externally allocated with a JCL DD statement is automatically deallocated at the completion of the JOB step. A data set externally allocated with a TSO ALLOCATE command is de-allocated with a TSO FREE command. A data set internally allocated with either a LIBNAME or FILENAME statement is automatically deallocated at termination of the SAS session, or may be deallocated with either a LIBNAME or FILENAME statement with the CLEAR argument.

External Deallocate (TSO command)
FREE DATASET('EXAMPLE.OAT')

Internal Deallocate (SAS data set)
LIBNAME IN CLEAR;
FILENAME IN CLEAR;

ADVANTAGES OF INTERNAL ALLOCATION OF DATA SETS

A data set can be allocated either externally or internally. Both methods instruct the MVS system to allocate a data set's resources. The MVS system does not distinguish between external and internal allocation. The SAS system, however, does differentiate, and offers several advantages for choosing the latter.

SAS macro statements provide a means of conditionally allocating a data set. A LIBNAME or FILENAME statement used to internally allocate a data set may be placed within conditionally executed macro statements. Thus, only those data sets deemed necessary by the SAS program will be allocated. In contrast, during external allocation, all data sets that potentially could be used during the SAS session must be allocated.

The following is an example of conditionally allocating a data set using SAS macro statements. The data set EXAMPLE.OAT is internally allocated by a LIBNAME statement when the macro variable SYSDATE is equal to 03OCT94.

%IF "&SYSDATE"="03OCT94" %THEN
  %DO;
  LIBNAME IN 'EXAMPLE.OAT';
  %END;

Under MVS, DSNNAME must be a fully qualified data set name. The SAS system option, SYSPREF=, provides a prefix for constructing a fully qualified name from a partially qualified name. The SYSPREF= option inserts the prefix provided in front of any partially qualified name beginning with a period located on the LIBNAME or FILENAME statement. The MVS system provides the user profile prefix as a means to allocate a partially qualified DSNNAME. However, this method limits the prefix to one qualifier, and may not be conditionally set.

The following is an example of constructing a fully qualified data set name from a partially qualified name using the SAS system option SYSPREF=.

The LIBNAME and FILENAME statements used during internal allocation are portable to other operating systems with minor changes to the data set name and options parameters. The JCL DD statement and the TSO ALLOCATE command are specific to the MVS environment, thus, they must be replaced entirely when moving to a new environment.

The LIBNAME statement is used to assign a SAS engine to a data set. Internally allocated data sets can be allocated and assigned an engine in a single step with the LIBNAME statement.

The following is an example of internally allocating a data set and assigning it an engine.

LIBNAME IN 'V605 EXAMPLE.DAT DISP=SHR';

Externally allocated files require one step to allocate a data set, and a second step to assign an engine. The data set is allocated with either a JCL DD statement or a TSO ALLOCATE command. Then, in a second step, the data set is assigned an engine with a LIBNAME statement.

The following is an example of assigning an engine to an externally allocated data set.

%IN DSN='EXAMPLE.DAT',DISP=SHR
LIBNAME IN 'V605';

An internally allocated data set may be deallocated in a single step. The LIBNAME and FILENAME statement with the CLEAR argument deallocate data sets which were internally allocated. If multiple DDNAMEs are associated with a data set, all DDNAMEs must be disassociated from the data set before it may be deallocated.

The following is an example of deallocating a data set which was initially allocated internally.

LIBNAME IN CLEAR;
Externally allocated data sets may also be deallocated with a LIBNAME or FILENAME statement. However, the LIBNAME or FILENAME statement will only discontinue the association between DDNAME and DSNAME; the data set will not be freed. In a second step, an operating system command must be issued to free the data set.

The following is an example of internally deallocating a data set which was initially allocated externally.

```
LIBNAME IN CLEAR;
X TSO FREE DATASET('EXAMPLE.DAT');
```

Internally allocated data sets are listed immediately in the LIBNAME and FILENAME windows, and by the LIST argument provided by the LIBNAME and FILENAME statements. Externally allocated data sets are not listed by either method until after they have been used as a LIBREF or FILEREF during a SAS session.

When an error occurs during internal allocation, the error message is printed directly after the LIBNAME or FILENAME statement. This simplifies error identification and resolution. When an error occurs during external allocation with a JCL DD statement, the error message is printed at the end of the JCL stream and may not contain the necessary information needed to identify and resolve the error. For example, when a data set name is mistyped in the JCL DD statement, a JCL error occurs. Following the JCL stream, an error message is printed giving only the DDNAME and an error message indicating that the data set could not be found. The user must refer to the JCL DD statement printed earlier in the JCL stream to identify the corresponding data set.

Although the LIBNAME and FILENAME statements have rules governing their syntax, these rules are not nearly as stringent as those placed on JCL DD statements. For example, a few of the rules governing JCL DD statements are:

- The entire JCL DD statement must be in uppercase.
- The JCL DD statement must begin in column 1 and end before column 72.
- The JCL DD statement DDNAME must begin in column 3.
- Complete parameters of the JCL DD statement must be ended with a comma before continuation to the next line.

None of these syntax rules apply when allocating a data set internally with either a LIBNAME or FILENAME statement.

Internal and external methods of allocation are equally capable of allocating a data set in the MVS environment. However, internal allocation clearly becomes the better method when the SAS system will use or create the data set.

**SUMMARY**

A data set in the MVS environment must be allocated before it can be used during a SAS session. Allocation can be either external, using a JCL DD statement or a TSO ALLOCATE command, or internal, using a LIBNAME or FILENAME statement. Whether a data set is allocated externally or internally, the outcome is the same: a data set's resources are made available. However, when a data set's resources are required during a SAS session, internal allocation methods offer many advantages over their external counterparts.

**REFERENCES**


