Data Set File System (aka DSFS)- Simplified Administration

By Lionel B. Dyck October 20, 2023

IBM introduced the Data Set File System (DSFS) with z/OS 2.5 as an option for shell users of z/OS to easily access z/OS datasets using traditional shell services. By shell I'm referring to the OMVS environment using the Unix System Services (USS) filesystem. Included with DSFS are a number of z/OS operator commands, as well as TSO and shell commands, all intended to help manage the DSFS environment. As with nearly all IBM services, the commands are well documented with the expectation that the admins (aka System Programmers) and users will look them up when they are needed This further assumes that these individuals will even know these exist and where to look for the information.

Enter the z/OS open-source tool **DSFSADM** that was inspired by a presentation given by Kershaw Mehta of IBM at the summer SHARE of 2021. I was however saddened by the fact that it wasn't available for z/OS 2.4 as I worked in an environment with both 2.4 and 2.5 systems and thus was not able to implement it at the time. At the New Orleans SHARE of 2023, I was once again inspired when I attempted to attend a session on DSFS and was unable to get in the door as it was standing-room-only. Once I had a full z/OS 2.5 environment I implemented it and started to experiment with it. I found it worked well but I could never remember all the commands and related parameters.

For those who have not heard about DSFS, or may have heard of it but don't know how it works, once configured it will provide the z/OS shell users (via OMVS or ssh) with the ability to access sequential and partitioned data sets (both PDS and PDSE) using standard shell commands such as cp, grep, vim, awk, etc. without having to specify the z/OS dataset using the arcane // prefix. That is assuming the shell command even supports that nomenclature to access a z/OS dataset. Note that PDSE member generations are ***not*** supported.

For example, to see the contents of a member of my JCL library while in the OMVS shell, I used the `cat` command thus:

```
/u/lbdyck>cat /dsfs/txt/lbdyck/jcl.cntl/tmp
//TESTXX   JOB LBD,'LIONEL',REGION=0M,NOTIFY=&SYSUID,CLASS=A
//OUT      OUTPUT DEFAULT=YES,JESDS=ALL,OUTDISP=(HOLD,HOLD)
/*
//TSO      EXEC PGM=IKJEFT1A
//STEPLIB  DD DISP=SHR,DSN=LBDYCK.LIONEL.LOAD
//SYSXSPRT DD  SYSOUT=*
//SYSTSIN  DD  *
 PDS TEST.PDS HI :
/*
/u/lbdyck>
```

You will notice that the path `/dsfs/txt/lbdyck/jcl.cntl/tmp` is divided thus:

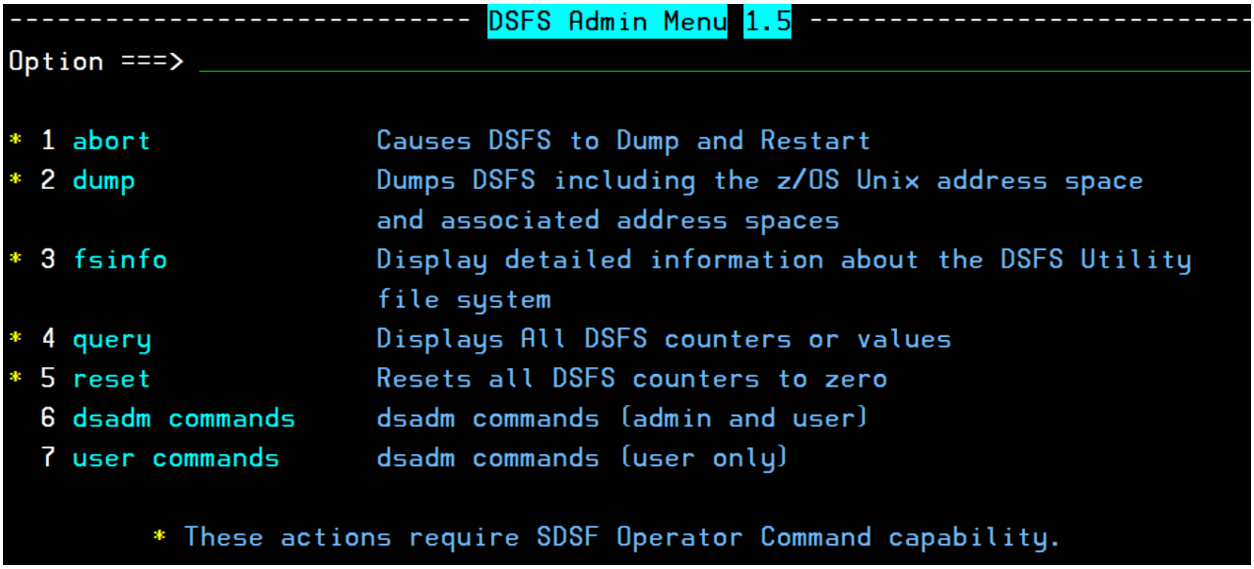
`/dsfs`	The root for the DSFS filesystem
`/txt`	Indicated to DSFS that the files are text files
`/lbdyck`	The userid for the datasets to be accessed (like the users home directory)
`/jcl.cntl`	The dataset name
`/tmp`	The member name as this is a PDS

The **DSFSADM** tool is an ISPF dialog that utilizes the SDSF REXX interface for the z/OS operator commands and the `bpxwunix` REXX interface for the shell commands.

It is invoked using the DSFSADM REXX command with one of three options.

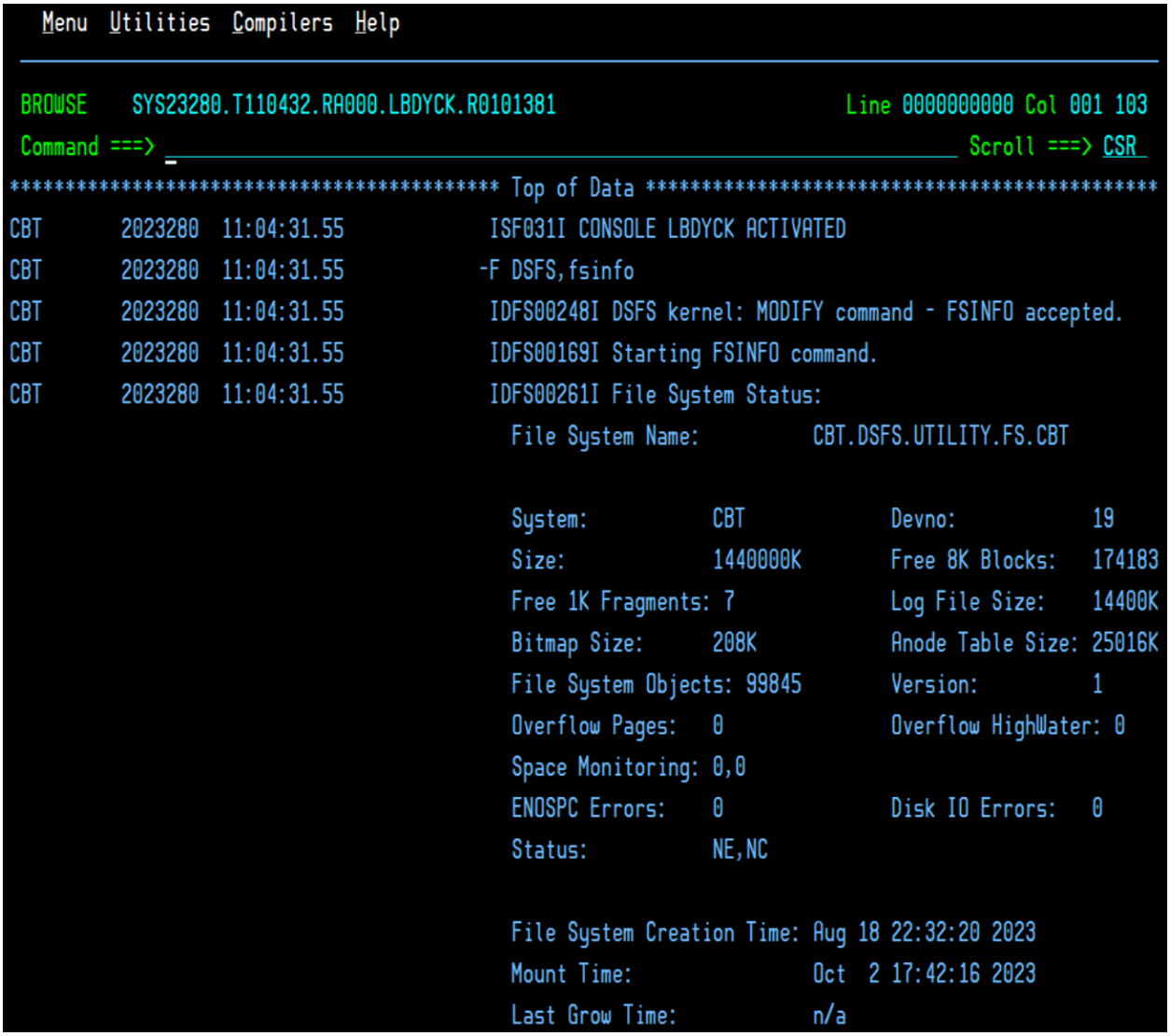
> **A** for admin which is used for the z/OS Operator commands and provides an option for accessing the other two menus.
>
> **D** for the `dsadm` shell commands, both admin and user.
>
> **U** for the `dsadm` shell commands typically used by a user. This is the default.

This is the Admin Menu:



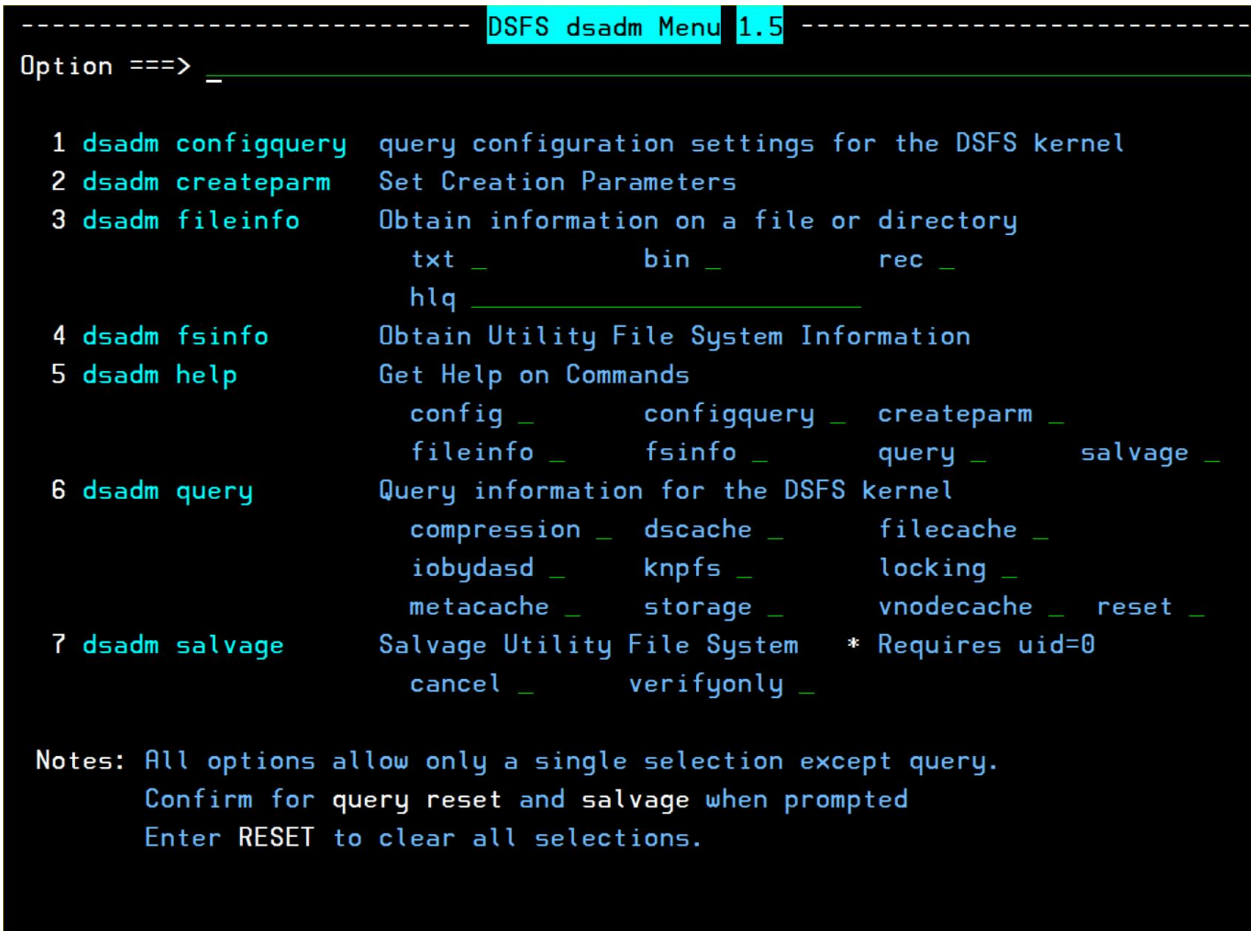
Be careful of the **abort** (option 1) and **dump** (option 2) options as they will do what they say. Option 5, **reset**, should be used sparingly (imho).

And a sample report after entering **3** for the **fsinfo** command:

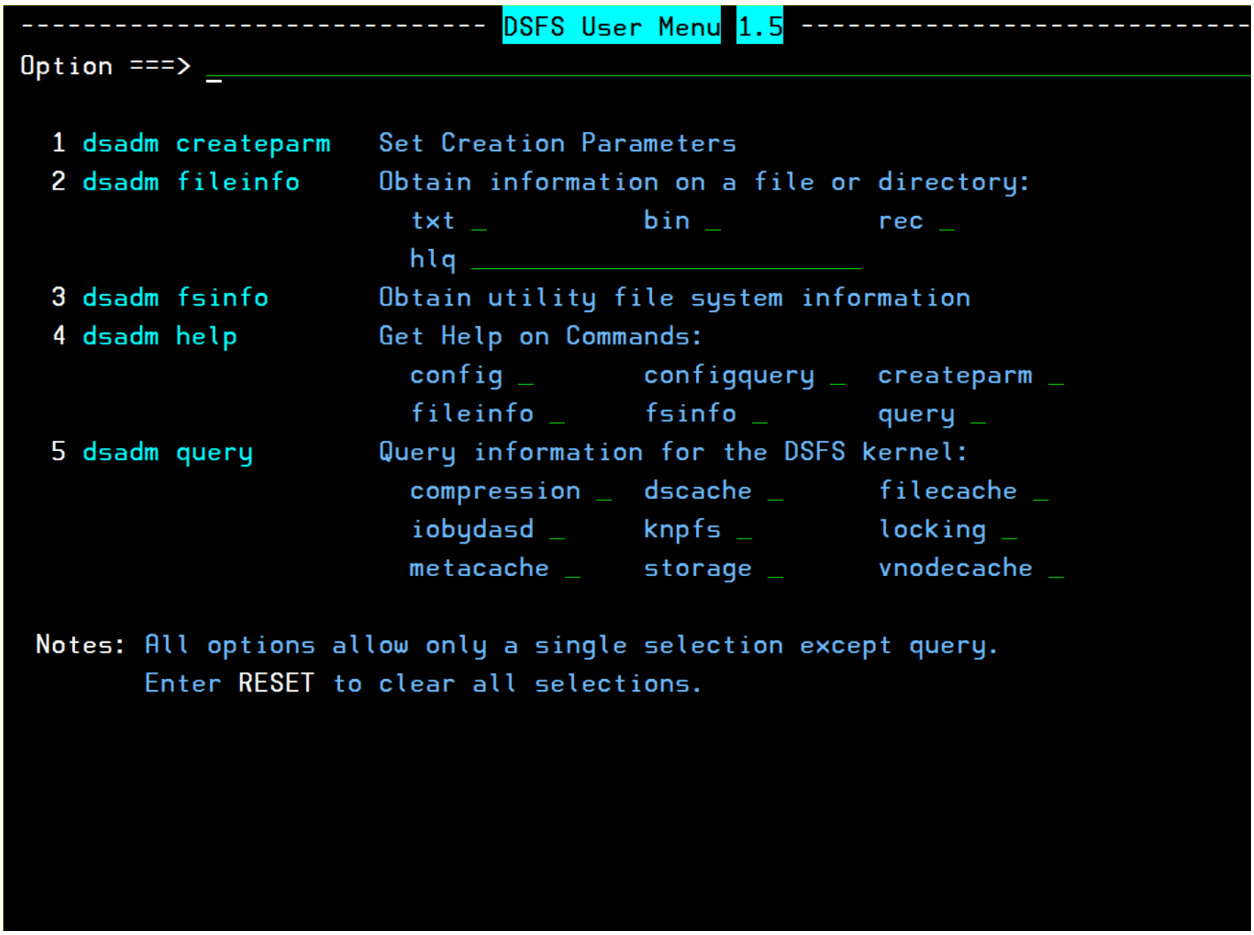


As you can see, the report is presented using ISPF Browse and as this is an operator command presents the system log as returned by the SDSF REXX interface. On the second line of the display data you can see the command that was generated F **DSFDS,fsinfo** followed by the commands response.

The **dsadm** menu, option **6** on the admin menu or option **D** with the **DSFSADM** command, presents the set of supported **dsadm** commands, some of which are not included for various reasons but which you'll probably not miss.

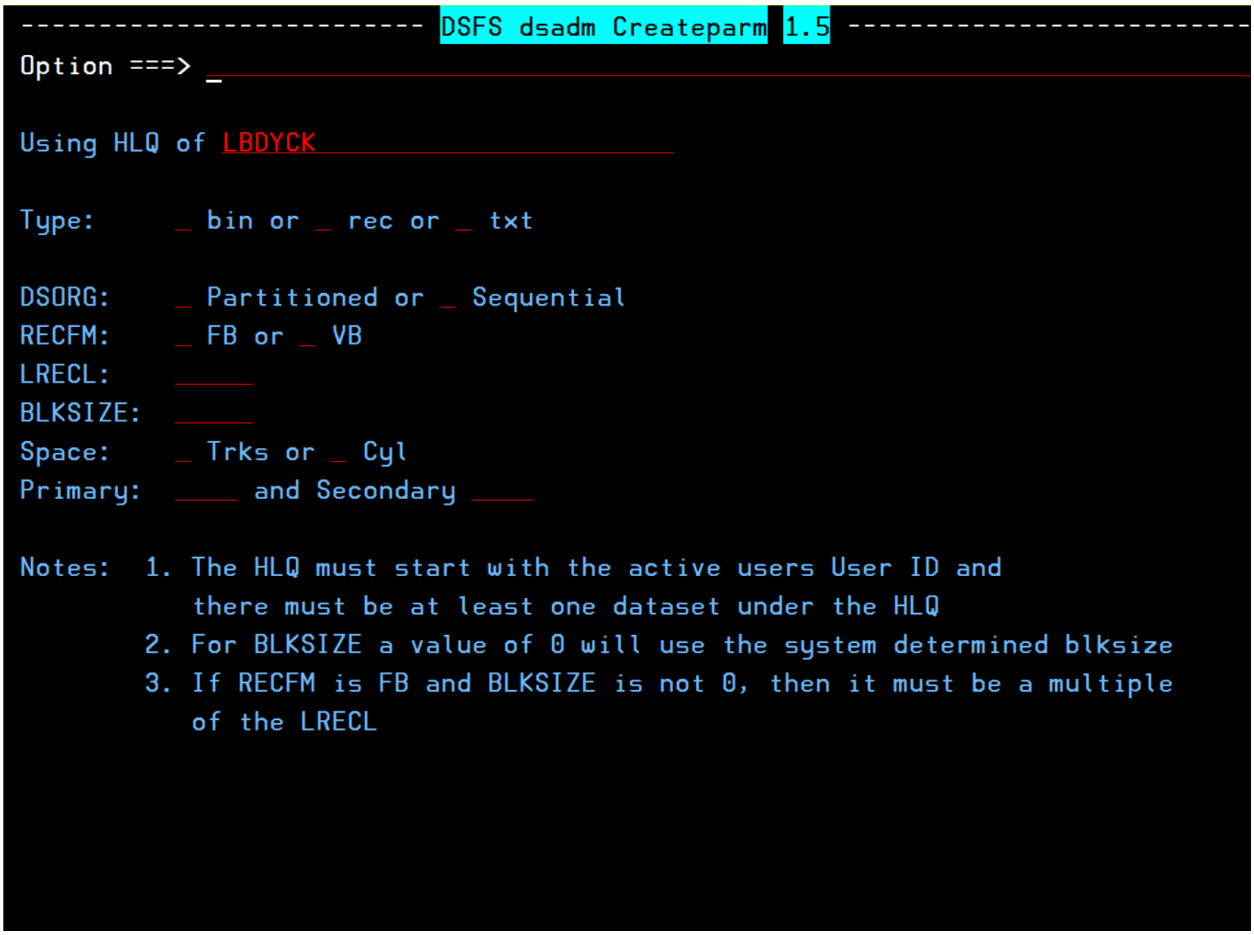


The **user** menu, option **7** on the admin menu or option **U** with the **DSFSADM** command, includes those commands that I, as the author of this tool, felt are most useful for users (and safest).



Note that the ***dsadm fileinfo*** will only report on the ***createparm*** path models that match the provided **hlq**. Thus, if the **hlq** is *userid* and there is a model under *userid.test,* that model will not be reported on.

The ***dsadm createparm*** option presents a panel to specify those options:



The HLQ, or high-level-qualifier, is pre-filled in with the active users **userid** as is required by DSFS at this time. And another requirement is that at least one dataset already be allocated under the provided HLQ for this to work.

Where to get DSFSADM

To get this open-source tool there are two options:

1. https://github.com/lbdyck/dsfsadm
2. File 312 on https://cbttape.org (check https://cbttape.org/updates.htm first)

IBM Ideas

The IBM Ideas portal is where you can help to influence the direction of any IBM product. In this case, as of the time this article was written, there are 11 Ideas for DSFS.

The link that I use to access the Ideas portal is: https://ibm-z-hardware-and-operating-systems.ideas.ibm.com/ideas. Then in the 'Search all ideas' field enter **dsfs** and then enter to do the search.

You can vote for any idea as well as add your own comments if you are so inclined. Both of these will require that you register a userid with IBM.

Among the ideas are:

Enhance DSFS CreateParm to support masking for the path

Add DSFS DeleteParm to remove obsolete HLQ's

DSFS CreateParm should NOT require a previous allocation under secondary qualifiers

DSFS Fileinfo should report All models under HLQ

DSFS CreateParm limitations

DSFS support for GDGs