z/OS Academy 2022

Data movement in z/OS – why DFSMSdss is so important

—

Redelf Janßen
IBM zSystems Brand Technical Specialist

What are we going to chat about over the next hour?

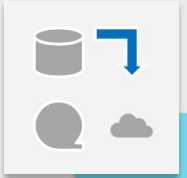
- ⭐ Why use DSS
- ⭐ Capabilities
- ⭐ Logical vs Physical processing
- ⭐ Filtering for data set selection
- ⭐ Space Management
- ⭐ Availability Management
- ⭐ Data movement and Replication

Why use DSS?

Commands

Space Management

- Defrag
- Consolidate
- Compress
- ConvertV
- Release
- Thin Provisioning Space Release
- CloudUtil



Availability Management

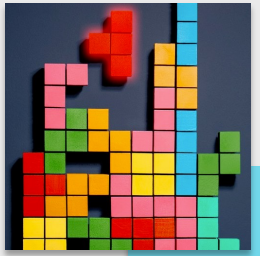
- Dump
- Restore
- Copydump

- Buildsa / Stand Alone Services

Data Movement and Replication

- Copy
  - (migration)

Advantages



Data Selection

- Powerful filtering capabilities and volume selection.

Data Movement

- Lowers application impact:
  - FlashCopy
  - ConcurrentCopy
  - CloneCopy
- Invokes Utilities
  - ICKDSF, Access methods, Catalog, etc.

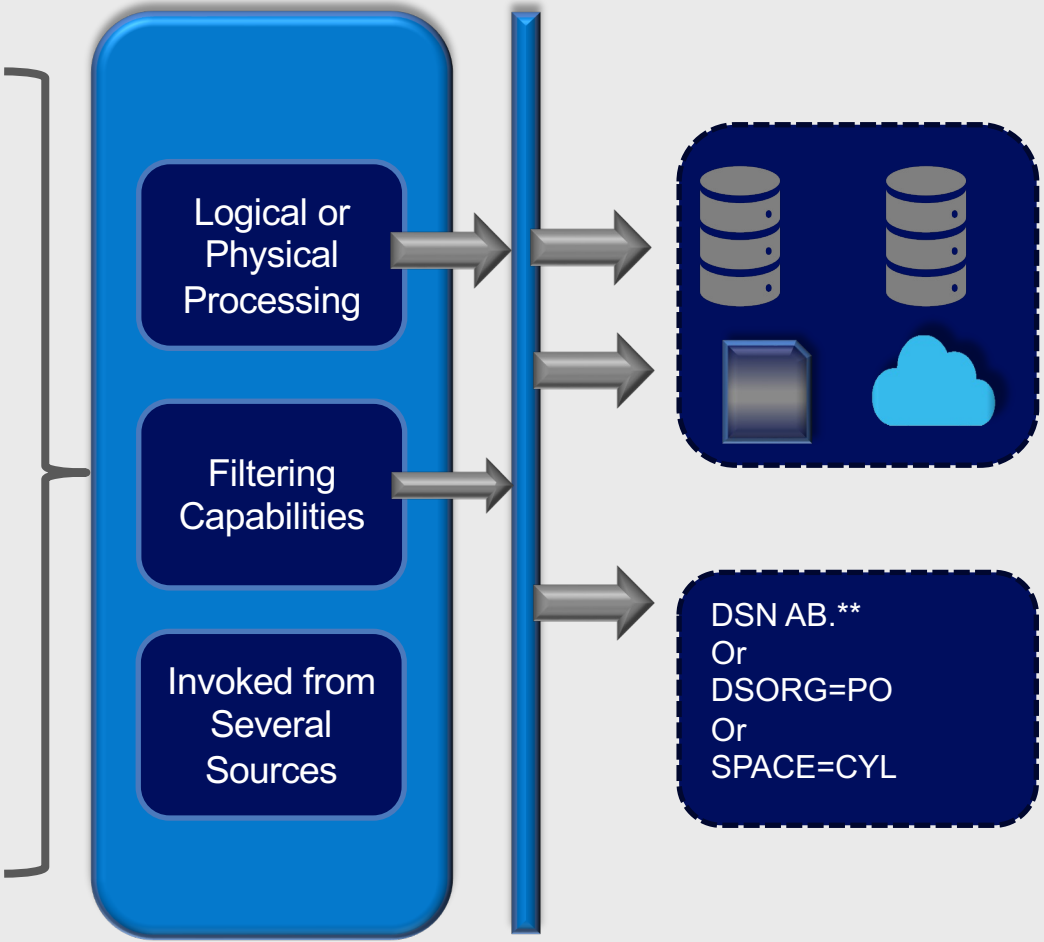
Capabilities.

Supports data movement of any type of data on z/OS

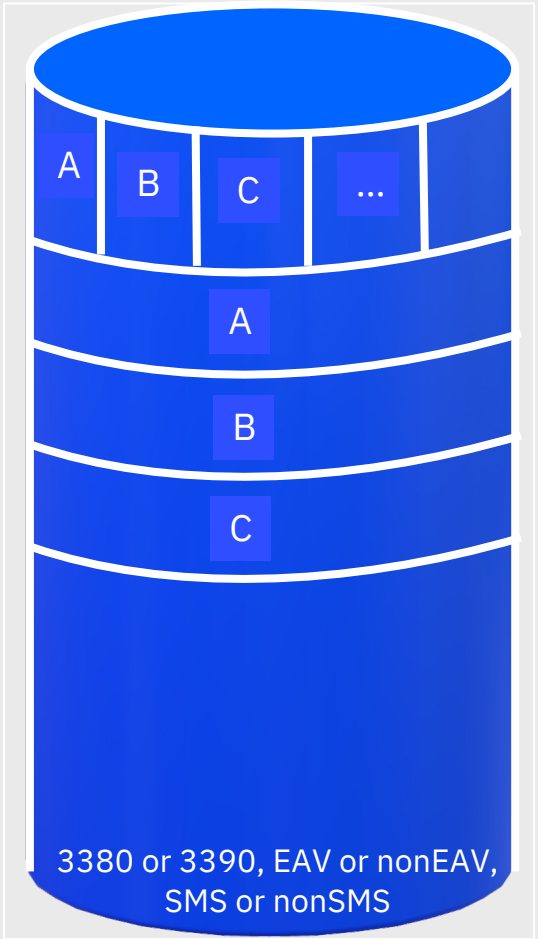
Data Set Selection by Name or Data Set Characteristics

UNIX file selection by name

Invoked from JCL, ISMF, or by Application Program



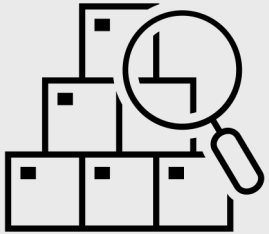
Input types



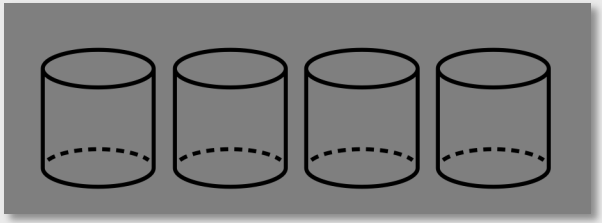
VTOC

Data Sets
- Non-VSAM
- VSAM
  - ZFS (UNIX files)

Free Space

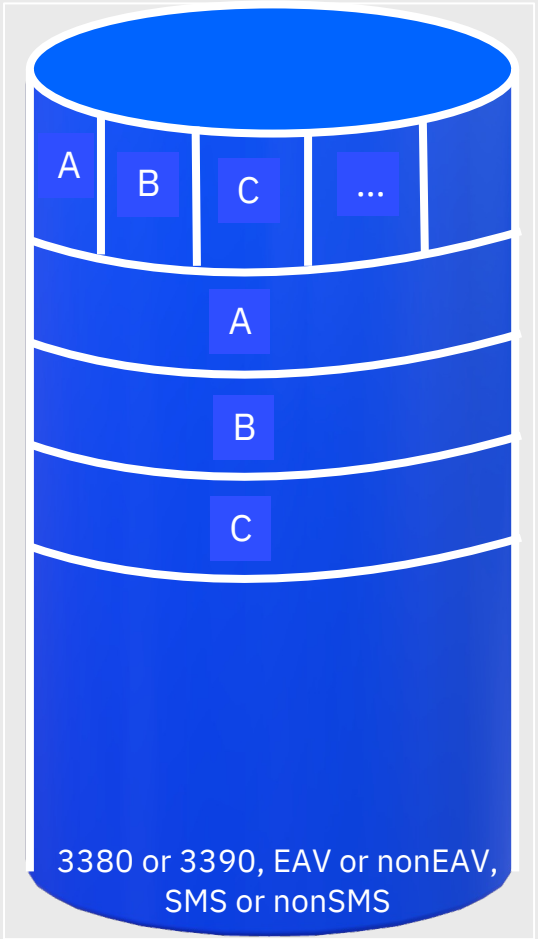


Catalog



Storage Group

Output types



VTOC

Data Sets
- Non-VSAM
- VSAM
  - ZFS (UNIX files)

Free Space

TAPE

Cloud Object Storage

Logical VS Physical

Logical processing

Generally considered to be functions that operate against a data set.

- Copy, dump, restore

Physical processing:

Generally considered to be functions that operate against a volume.

- Copy, dump, restore, defrag, consolidate, etc.

Logical processing

Generally considered to be functions that c
data set.

There are always exceptions:

- Physical data set processing
- Logical data set processing with input or output volumes
- Etc.

Physical processing:

Generally cons
functions that operate against a volume

Filtering.

Pattern matching

- ABC.*.DSET
  - Selects all data sets that have high level qualifier ‘ABC’ and a third qualifier matching ‘DSET’ with any second qualifier:
  - ABC.DEF.DSET would be selected
  - ABC.DSET and ABC.DEF.GHI.DSET would not be selected.
- ABC.**
  - Selects all data sets that have high level qualifier ‘ABC’ and have any number of other qualifiers
  - ABC.DEF.DSET would be selected
  - ABC.DSET and ABC.DEF.GHI.DSET would also be selected
- **.DSET
  - Selects all data sets that have their last qualifier matching ‘DSET’
  - Make sure to specify volumes!!!
- ABC*.**
  - Selects all data sets that start with ABC in the first qualifier and have any additional qualifiers.
    - ABC1.DSET, ABCDEFG.DSET, etc.

Pattern matching

- % - Single character substitution
- ABC.D%%G.DSET
  - Would match ABC.DEFG.DSET or ABC.D12G
  - Would not match ABC.DOG.DSET

Filtering

Include

Selection of data sets by name or pattern

Exclude

De-selection of data sets by name or pattern

By

Selection of data sets by data set characteristics

Filtering

Include

Selection of data sets by name or pattern

Exclude

De-selection of dat

- **255 entries each !!!**
- **Not enough?**
  - **Use FILTERDD to a sequential or partitioned data set member that has your filter criteria.**

By

Selection of data sets by data set characteristics

Building a list of data set to process:

Make a backup of all partitioned data sets with the HLQ of DSSDATA that have changed since the last backup cycle.

```
DUMP DATASET(                  -
     INCLUDE(DSSDATA.**)       -
     BY( (DSORG,EQ,PDS)        -
         (DSCHA,EQ,YES) )-
     ) -
 ...
```

- **BY filters**:
  - ALLOC, CATLG, CREDT, DATACLAS, DSCHA, DSORG, EXPDT, EXTNT, FSIZE, MGMTCLAS, MUTLI, REFDT, STORCLAS
- Supports inequalities
  - EQ or =
  - LT or <
  - LE or <=
  - GT or >
  - GE or >=
  - NE or !=

Volume vs Catalog filtering

- **VTOC filtering**:
  - Filters are applied only to the data sets found on the volumes specified.

```
DATASET(INCLUDE(DSSDATA.**)) -
    LOGINDY(VOL001,VOL002,…,VOL00n)

DATASET(INCLUDE(DSSDATA.**)) -
    STORGRP(STORGRPX)
```

- **Catalog filtering:**
  - Filters are used against all cataloged data sets on the system.

```
DATASET(INCLUDE(DSSDATA.**))
```

Invoking DSS.

DFSMSdss can be invoked using:

- **ISMF**
  - Lots of information via ISMF panels on how to setup profiles. Refer to: *z/OS DFSMS Using the Interactive Storage Management Facility*.

- **Application Interface**
  - Hello HSM!
  - You might be an application programmer - Get to know our Application Exits!
    - Topic is beyond the scope of this presentation.

- **Batch JCL**
  - Most users

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *

.........1.........2.........3.........4.........5.........6.........7 .........8
/*
```

DFSMSdss Batch JCL:

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *

.........1.........2.........3.........4.........5.........6.........7 .........8
/*
```

JOB CARD

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *

.........1.........2.........3.........4.........5.........6.........7 .........8
/*
```

EXEC Statement

DFSMSdss Batch JCL:

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU,PARM='UTILMSG=YES'
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *

.........1.........2.........3.........4.........5.........6.........7 .........8
/*
```

PARM Statement

- UTILMSG

DSS may call Utilities to perform several operations. This setting will place utility output into our SYSPRINT

DFSMSdss Batch JCL:

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *

.........1.........2.........3.........4.........5.........6.........7 .........8
/*
```

SYSPRINT DD

- A sequential message data set residing on:
  - System output device
  - TAPE volume
  - DASD volume
- 84 <= LRECL <= 137

DFSMSdss Batch JCL:

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *

.........1.........2.........3.........4.........5.........6.........7 .........8
/*
```

Command data set

- Typically an input stream
  - Can be sequential or portioned data set member with LRECL=80, Fixed.

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
```


```
.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
```


```
.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Characters in column 1 or 73-80 are ignored
** May result in unpredictable results ***

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//OUTPUT   DD DISP=(,CATLG),DSN=TDS.DUMP1,
//            DCB=(RECFM=U,LRECL=32760,BLKSIZE=0),
//            UNIT=SYSDA,SPACE=(CYL,(10,50))
//SYSIN    DD *
 DUMP DATASET(INCLUDE(DSSDATA.**)) INDD(INPUT) OUTDD(OUTPUT)



.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//OUTPUT   DD DISP=(,CATLG),DSN=TDS.DUMP1,
//            DCB=(RECFM=U,LRECL=32760,BLKSIZE=0),
//            UNIT=SYSDA,SPACE=(CYL,(10,50))
//SYSIN    DD *
 /* THIS IS A COMMENT:  NOT REQUIRED TO USE ALL 72 COLUMNS */

            DUMP DATASET(INCLUDE(DSSDATA.**)) -
                                                 INDD(INPUT) OUT+
DD(OUTPUT)



.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//OUTPUT   DD DISP=(,CATLG),DSN=TDS.DUMP1,
//            DCB=(RECFM=U,LRECL=32760,BLKSIZE=0),
//            UNIT=SYSDA,SPACE=(CYL,(10,50))
//SYSIN    DD *
 /* THIS IS A COMMENT:  NOT REQUIRED TO USE ALL 72 COLUMNS */

           DUMP DATASET(INCLUDE(DSSDATA.**)) -
                                                 INDD(INPUT) OUT+
DD(OUTPUT)



.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Command continuation

DFSMSdss Batch JCL:


```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//OUTPUT   DD DISP=(,CATLG),DSN=TDS.DUMP1,
//            DCB=(RECFM=U,LRECL=32760,BLKSIZE=0),
//            UNIT=SYSDA,SPACE=(CYL,(10,50))
//SYSIN    DD *
 /* THIS IS A COMMENT:  NOT REQUIRED TO USE ALL 72 COLUMNS */

            DUMP DATASET(INCLUDE(DSSDATA.**)) -
                                                 INDD(INPUT) OUT+
DD(OUTPUT)



.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Word continuation

Space Management.

DEFRAG:

Consolidates the free space on a volume to help prevent out-of-space abends on new allocations.

Track managed and Cylinder managed regions treated separately

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//SYSIN    DD *

 DEFRAG DDNAME(INPUT) FASTREPLICATION(PREF)

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Additional keywords available to tailor the request.

CONSOLIDATE:

Performs data set extent consolidation or reduction.

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//SYSIN    DD *

 CONSOLIDATE DATASET(INCLUDE(DSSDATA.**)) PHYSINDDNAME(INPUT) FR(PREF)

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Additional keywords available to tailor the request.

COMPRESS:

De-gasses partitioned data sets (PDS) using IEBCOPY.

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//SYSIN    DD *

 COMPRESS DATASET(INCLUDE(DSSDATA.**)) DDNAME(INPUT)

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Additional keywords available to tailor the request.

CONVERTV:

Convert a volume from non-SMS to SMS

- Can also convert SMS to non-SMS, not used as often because of data set conversion limitations.

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//SYSIN    DD *

 CONVERTV SMS DDNAM(INPUT) TEST

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

- TEST keyword: Trial run, no actual conversion
- PREPARE keyword: Puts volume in state ready for conversion.

RELEASE:

Releases allocated but unused space from sequential, partitioned, and extended-format VSAM data sets.

Can be performed using logical or physical processing.

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//SYSIN    DD *

 RELEASE INCLUDE(DSSDATA.**) SPHERE

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

SPACE RELEASE:

For volumes that are Thin Provisioned.

Free space extents are released on the volume(s) so that space on chunk boundaries can be released back to the storage extent pool.

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//SYSIN    DD *

 SPACEREL DDNAME(INPUT) MINCYLS(42)

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

CLOUDUTILS:

LIST or DELETE dump data sets residing in object-storage (CLOUD)

Operates on DSS dump data only

Any cloud type is supported; however browsers (like CYBERDUCK) cannot browse object store on TAPE servers.

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//INPUT    DD DISP=SHR,UNIT=SYSDA,VOL=SER=VOL001
//SYSIN    DD *

 CLOUDUTILS LIST CLOUD(MYCLOUD) CONTAINER(DSSDUMPS) -
 OBJPFX(MYPREFIX) CDACREDSTORE

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Availability Management.

DUMP/RESTORE:

Create backups of your data at varying levels:

- FULL (volume)
  - z/OS and zLinux volumes
- TRACKS (Specific tracks of a volume)
  - z/OS, zLinux, and zVM volumes
- DATASET
- PATH (UNIX files)

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//OUTPUT   DD DISP=(,CATLG),DSN=TDS.DUMP1,
//            DCB=(RECFM=U,LRECL=32760,BLKSIZE=0),
//            UNIT=SYSDA,SPACE=(CYL,(10,50))
//SYSIN    DD *
 DUMP DATASET(INCLUDE(DSSDATA.**)) OUTDD(OUTPUT)


 RESTORE DATASET(INCLUDE(**)) INDD(OUTPUT) RENAMEUNCONDITIONAL(NEWDATA)
.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

DUMP:

Delete data sets
- Take advantage of our filtering capability
- Data sets do not need to be cataloged

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//OUTPUT   DD DUMMY
//SYSIN    DD *
 DUMP DATASET(INCLUDE(DSSDATA.**) BY(REFDT,LE,*,-366)) OUTDD(OUTPUT) -
      DELETE PURGE

.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

RESTORE:

What is in the backup?
- Use PARM='TYPRUN=NORUN'

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU,PARM='TYPRUN=NORUN'
//SYSPRINT DD SYSOUT=A
//INPUT    DD DSN=TDS.DUMP1,DISP=SHR
//SYSIN    DD *
 RESTORE DATASET(INCLUDE(**)) INDD(INPUT)
.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

Stand-Alone services :

Need to restore Full volume images so that you can IPL z/OS?

Step 1: Keep a BUILDSA IPLable image on DASD or TAPE.

Step 2: From HMC, IPL from device that has BUILDSA image.

Step 3: Using DFSMSdss Stand alone services, Restore full volume images from TAPE

Data movement and Replication.

COPY DATASET:

Data movement and Replication

Make copies of your data sets:
- Give new copies a new name (RENAMEUNCONDITIONAL)
- Replace old copies with the same name (REPLACEUNCONDITONAL)
- Change SMS management detail (STORCLAS, MGMTCLAS,NULLSTORCLAS, NULLMGMTCLAS,BYPASSACS)
- Migrate to faster/different DASD (DELETE)

DSS will:
- Consolidate extents
- Maintain the source data set attributes (Compression, Encryption, etc).

COPY FULL:

Migrate to Larger DASD at Volume-level

- Larger volume image is reflected as free space after Copy.

Make a temporary copy of your volume so that your backups are performed from the copy and does not affect production data access

- FlashCopy capability using DUMPCONDITIONED keyword.
  - FlashCopy discussion would need another session to cover!

COPY:

Data movement and Replication

```
//MYJOB    JOB  accounting info,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSSU,PARM='TYPRUN=NORUN'
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
 COPY FULL INDYNAM(VOL001) OUTDYNAM(VOL002) DUMPCONDITIONING FR(PREF)

 DUMP FULL INDYNAM(VOL002) FCWITHDRAW
.........1.........2.........3.........4.........5.........6.........7.........8
/*
```

RESULT:

- VOL01 and all its data is accessible for production use while DUMP is performed.
- DUMP of VOL01 data is performed via dump-conditioned VOL002

You have been initiated.

References:

z/OS DFSMSdss Storage Administration Guide
z/OS DFSMSdss Storage Administration Reference
z/OS DFSMS Advanced Copy Services (Fast Replication, ConcurrentCopy, etc)

Latest Development in DSS:

If time permits…

- UNIX file-level backup and restore
  - Previously can only backup and restore File system level (zFS)
  - Now can use DSS/HSM for file-level

- Cloud support for backup and restore – Transparent Cloud Tiering
  - MIPs are used for bringing data into host and writing out to backup media
  - IBM Disk and Tape storage solution to move user data transparently (no MIPs) to the cloud.
    - Traditional object storage or Tape-Object storage.

UNIX File-level backup:

If time permits…

```
DUMP PATH(INCLUDE('usr/lib/nls/locale/Bg_BG.lp64')) -
WORKINGDIRECTORY('/') OUTDDNAME(OUTPUTDD) CLONE(PREFERRED)
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'DUMP '
ADR109I (R/I)-RI01 (01), 2020.311 11:30:28 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR050I (001)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR035I (001)-PRIME(50), INSTALLATION EXIT ALTERED TAPE BLOCK SIZE DEFAULT TO 32 K-BYTES
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (001)-STEND(01), 2020.311 11:30:28 EXECUTION BEGINS
ADR650I (001)-UDFLT(01), ALL PATHS ARE RELATIVE TO WORKING DIRECTORY
/
ADR664I (001)-DTUNX(01), 2020.311 11:30:28 PATH FILTERING IS COMPLETE. 5 OF 5 FILES WERE SELECTED
ADR454I (001)-UPRTT(01), THE FOLLOWING FILES WERE SUCCESSFULLY PROCESSED
                         d ./usr
                         d ./usr/lib
                         d ./usr/lib/nls
                         d ./usr/lib/nls/locale
                         - ./usr/lib/nls/locale/Bg_BG.lp64
ADR006I (001)-STEND(02), 2020.311 11:30:28 EXECUTION ENDS
ADR013I (001)-CLTSK(01), 2020.311 11:30:28 TASK COMPLETED WITH RETURN CODE 0000
ADR012I (SCH)-DSSU (01), 2020.311 11:30:28 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000
```

TCT:

If time permits...

```
DUMP FULL INDD(INPUTV) -
    CLOUD(RAYQUAZA) -
    CONTAINER(GT123EXTENTS) -
    OBJPFX(TEST)  DEBUG(CLMSG(S)) -
    CDACREDSTORE ALLDATA(*)  ALLEXCP
ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'DUMP '
ADR109I (R/I)-RI01 (01), 2020.296 20:46:08 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR016I (002)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (002)-STEND(01), 2020.296 20:46:08 EXECUTION BEGINS
ADR603I (002)-CLDDR(01), OBJECT 'TEST/HDR ' WAS STORED TO THE CLOUD
ADR603I (002)-CLDDR(01), OBJECT 'TEST/C9SS01/META/DTPTRK0 ' WAS STORED TO THE CLOUD
ADR603I (002)-CLDDR(01), OBJECT 'TEST/C9SS01/META/DTPVTOC ' WAS STORED TO THE CLOUD
ADR603I (002)-CLDDR(01), OBJECT 'TEST/C9SS01/META/DTPVVDS ' WAS STORED TO THE CLOUD
ADR603I (002)-CLDDR(01), OBJECT 'TEST/C9SS01/META/DTPBTMT ' WAS STORED TO THE CLOUD
ADR603I (002)-CLDDR(02), OBJECT 'TEST/C9SS01/DATA/EXTENTS00000001 ' WAS STORED TO THE CLOUD
ADR603I (002)-CLDDR(02), OBJECT 'TEST/C9SS01/DATA/EXTENTS00000002 ' WAS STORED TO THE CLOUD
ADR006I (002)-STEND(02), 2020.296 20:50:25 EXECUTION ENDS
ADR013I (002)-CLTSK(01), 2020.296 20:50:25 TASK COMPLETED WITH RETURN CODE 0000
ADR012I (SCH)-DSSU (01), 2020.296 20:50:25 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000
```

Thank you

Redelf Janßen
IBM zSystems Brand Technical Specialist

Contact

E-Mail	redelf.janssen@de.ibm.com
Phone	+49-171-5538587
LinkedIn	Redelf Janßen
Xing	Redelf Janßen