#SHAREorg

SHARE
Technology · Connections · Results

Best Practices for ICF Catalog

Session 12970

Stephen Branch – IBM Corporation
Janet Sun – Rocket Software

SHARE in San Francisco 2013

Agenda

- Catalog Management
- Catalog Backup and Recovery
- Catalog Diagnostics
- Buffering Basics
- Catalog Caching
- Catalog Sharing
- Catalog Tuning
- Final Thoughts

#SHAREorg

SHARE
Technology · Connections · Results

Catalog Management

SHARE in San Francisco 2013

Defining a Catalog

- Use IDCAMS DEFINE USERCATALOG
- Cannot span volumes
- Cannot be defined as a striped data set
- Can only be an Extended Format data set if at z/OS 1.12 or higher, allows extended addressability (>4GB)
- Allocate in cylinders
  - Results in maximum possible CA size of 1 cyl
- Specify secondary space value > 1 cyl
  - To prevent each CA split from requiring another extent

Defining a Catalog (continued)

- Recommend Data CISIZE of 4KB
- Use a minimum Index CISIZE of 3584 if using a 4KB Data CISIZE
  - Don’t take the default
- Start with STRNO(3)
  - Default is 2
- Don’t code BUFFERSPACE
  - Use BUFND and BUFNI instead
- Take the defaults for SHAREOPTIONS and RECORDSIZE

Types of Entries in a Master Catalog

- System related data sets
  - SYS1 and other data sets needed at IPL time
  - Page data sets
  - IODF
- User catalog connector records
  - Created with IDCAMS IMPORT CONNECT
- Alias records
  - Created with IDCAMS DEFINE ALIAS
  - An alias name defined in the master catalog can be used to reference a user catalog
- Keep everything else out!
  - Catalog management must read every record when building the alias in memory search tables

Defining a VVDS

- Always explicitly DEFINE VVDSs!
  - Default is TRACKS(10 10) - usually too small
  - Since z/OS 1.7 a system default can be set
    - Default is not preserved across an IPL
    - F CATALOG,VVDSSPACE(prim,sec) – not in Cylinders
- Plan ahead and review the section in Managing Catalogs “Estimating Space Requirements for the VVDS”
- Allocate in CYLINDERS
  - To provide enough space
  - EAV compatibility – can extend into cylinder managed storage

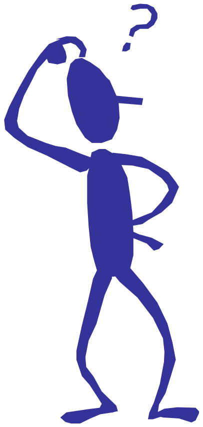
#SHAREorg

Catalog Backup and Recovery

Backing Up the BCS

Rule #1: Back up as often as you can

- How often is enough? It depends…
  - At least once a day for all catalogs
  - More often for volatile catalogs – where you are creating lots of SMF records – indicating heavy data set DEFINEs, DELETEs, and allocation extensions
  - More often for critical catalogs – ones that would present a major problem if recovery isn't fast
  - Less often for non-volatile catalogs



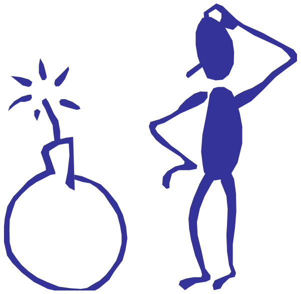
Backing Up the BCS

Rule #2: Verify all BCSs are included

- When was the last time you audited your backup job to see the list of catalogs backed up?
- Obtain a list of connected catalogs in all master catalogs

```
  LISTCAT UCAT
               LISTING FROM CATALOG -- CATALOG.MASTER.CAT
 USERCATALOG --- CAT.ICF.USER1
 USERCATALOG --- CAT.ICF.USER2
 ...
```

- Compare the list to your catalog backup job, and ensure that all are backed up



Backing Up the BCS

Rule #3: Double check the backups

- Establish a regular method to check catalog backup return codes
- Ensure you run EXAMINE INDEXTEST on each catalog – and then check the output!
- Consider duplexing your backups – and create a third copy for your disaster recovery (DR) site

Backing Up the BCS

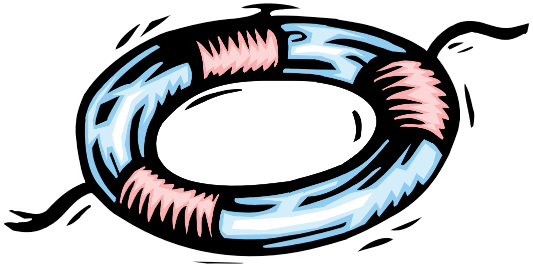
Rule #4: Verify that you can recover

- Can you locate your backups? If duplexed, catalog each in a different user catalog
- Can you locate your SMF data? How many systems are sharing the catalog? What catalog is it cataloged in?
- Test, test, test – if a problem occurs and you can't recover, your goose is …

BCS Forward Recovery

SMF Records Are the Only Way

- For BCS forward recovery, SMF record types required:
  - Type 61 – Data set define
  - Type 65 – Data set delete
  - Type 66 – Data set alter



- These records, written between the time of backup and restore, identify all new data sets created, deleted, and extended
- You must have something, and you must know how to use it!
- You should practice BCS forward recovery

ICFRU

Used for BCS Forward Recovery

- IBM field developed product
- Incorporated into z/OS DFSMS 1.7
- Takes IDCAMS EXPORT copy of the BCS and SMF records from all sharing systems as input
- Creates a new EXPORT format backup which is used as input to IDCAMS IMPORT to rebuild the catalog

ICFRU

Components of ICFRU

- CRURRSV – Record Selection and Validation
  - Processes dumped SMF data sets
  - Extracts appropriate records
- CRURRAP – Record Analysis and Processing
  - Processes the extracted and sorted SMF records, together with an EXPORT copy of the catalog
  - Produces a new EXPORT format data set to be imported to build a new catalog

Catalog Recovery or Maintenance

Locking Catalogs

- When performing certain maintenance or recovering a catalog, it is a good idea to LOCK the catalog to prevent undesired access
  - Use IDCAMS ALTER LOCK command
- Be sure to UNLOCK the catalog as soon as maintenance is complete
- *Be aware:* If you lock a shared master catalog from another system, make sure it is unlocked as soon as possible
  - If the master catalog is locked, the system cannot be IPLed!

#SHAREorg

SHARE

Technology • Connections • Results

Catalog Diagnostics

SHARE in San Francisco 2013

Finding Catalog Problems

Using Access Method Services (IDCAMS)

- EXAMINE INDEXTEST – ensures that sequential and key direct access is accurate
- EXAMINE DATATEST – reads all data CIs to ensure structural integrity
- DIAGNOSE ICFCATALOG (without compare) – checks information integrity within each BCS record (inside-the-BCS only)
- DIAGNOSE VVDS (without compare) – checks information integrity within each VVDS record (inside-the-VVDS only)
- DIAGNOSE is a tool that you use to see synchronization problems between the BCS and VVDS record structure

Fixing Problems

If EXAMINE Identified the Problem with a KSDS

- If it is not a BCS
  - Can delete data set and recover from recent backup
  - Can unload a non-BCS KSDS data set by reading data component directly as an ESDS, sorting data, and then loading into new KSDS
- This does not work for a BCS!
- If a BCS index component is damaged:
  - May be able to use IDCAMS REPRO NOMERGECAT to copy catalog records to a new, empty catalog
    - *FROMKEY and TOKEY can be used to get around bad records in the catalog*
  - Repair can be accomplished with IBM Tivoli Advanced Catalog Management for z/OS
  - Severe damage requires recovery from backup

Fixing Catalog Problems

If DIAGNOSE Identified the Problem

- May indicate an incomplete catalog entry
- If it is an entry in a BCS, delete the catalog record and attempt to recatalog

  DELETE xxx NOSCRATCH

  DEFINE xxx … RECATALOG

- If the truename exists without the associated cluster records:

  DELETE xxx TRUENAME

Fixing Catalog Problems

If DIAGNOSE Identified the Problem (continued)

- If it is an entry in a VVDS:

  DELETE xxx VVR

  DELETE xxx NVR

- It may be possible to recatalog the data set

  DEFINE CLUSTER(NAME(xxx) … RECATALOG)

- If missing some portions from the volume, then it must be deleted
  - IDCAMS cannot recreate the data

#SHAREorg

SHARE
Technology · Connections · Results

Buffering Basics

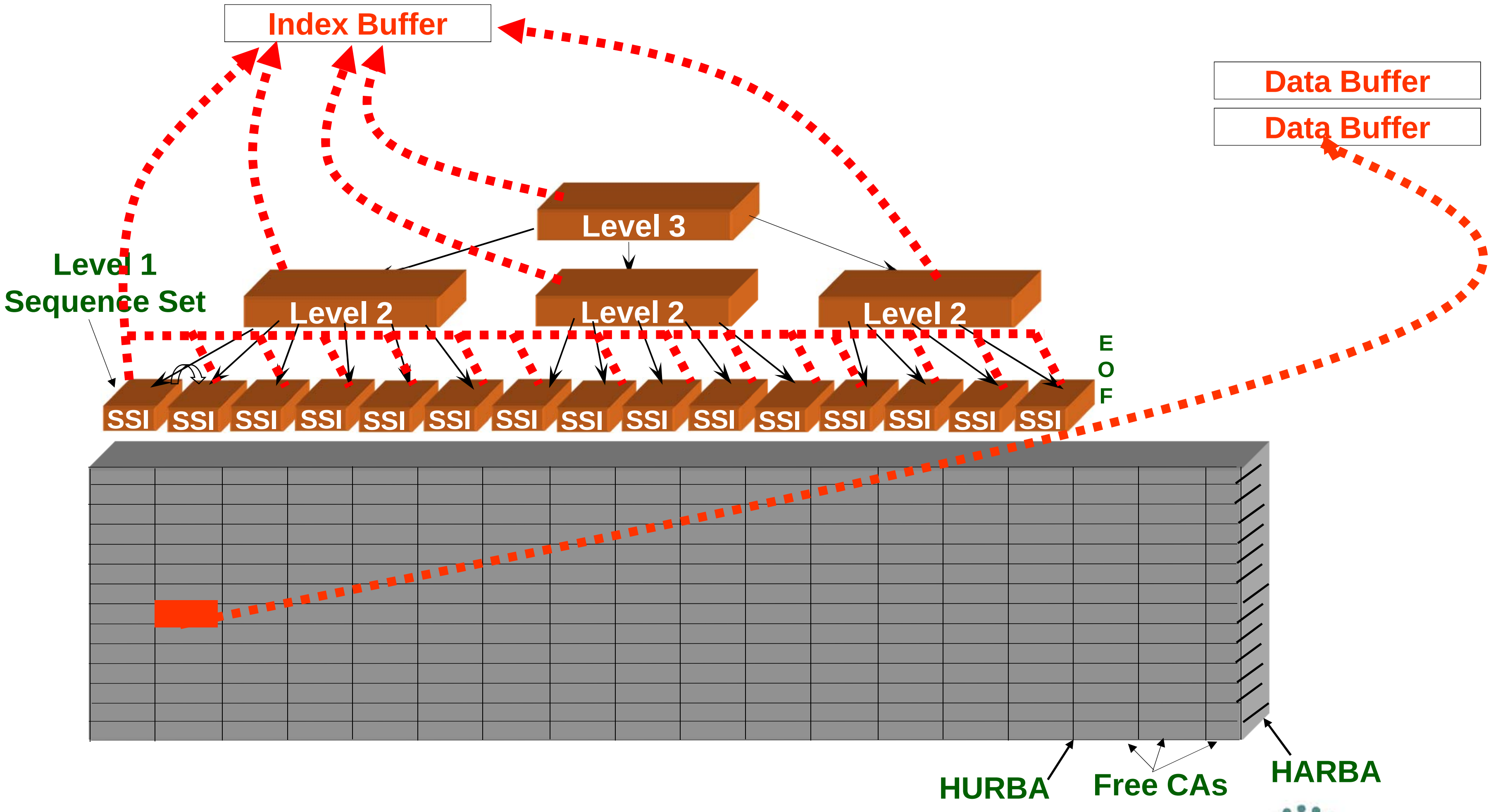
SHARE
in San Francisco
2013

Buffering Basics

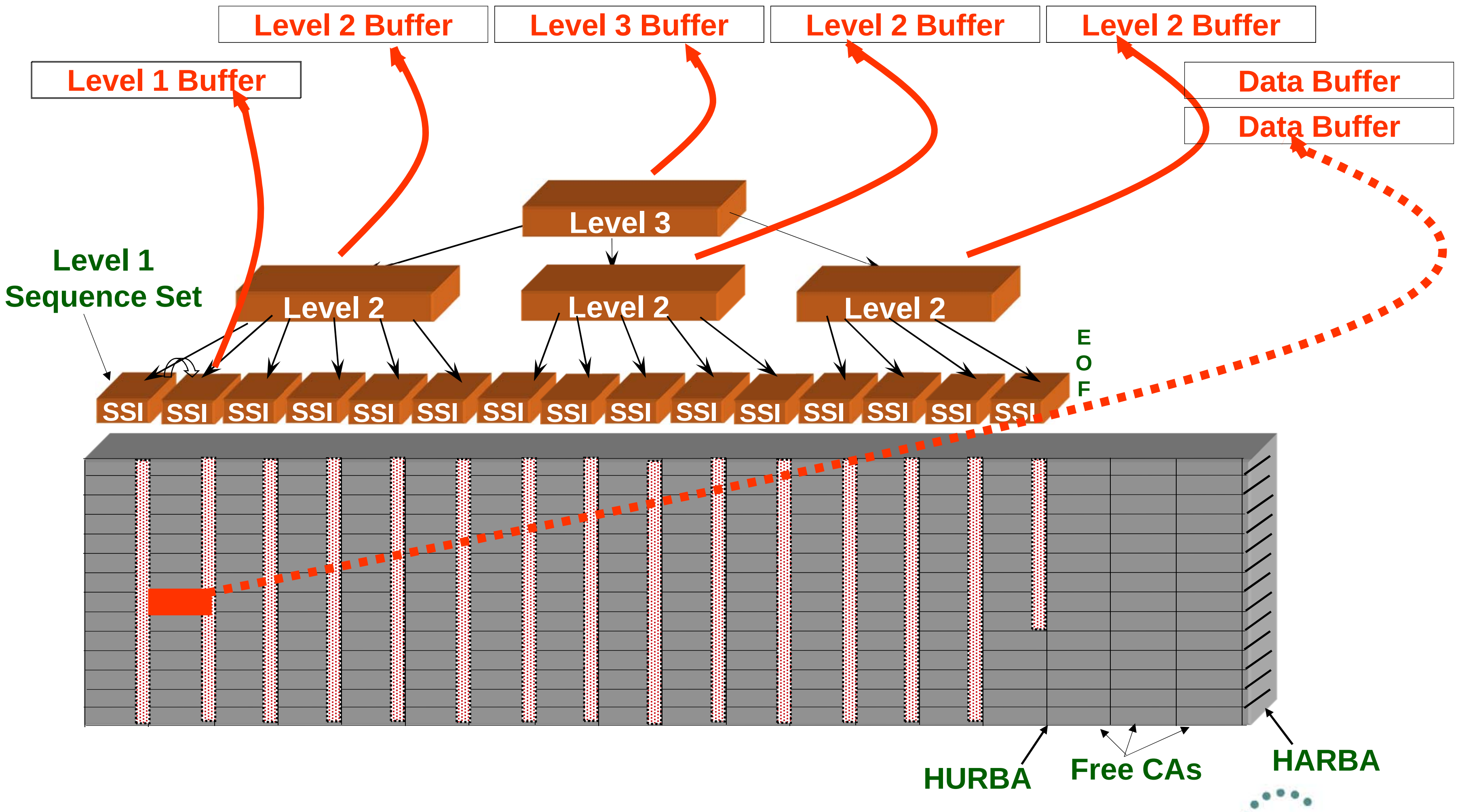
Data and Index Buffers

- The unit of transfer between DASD and storage is a Control Interval (CI)
  - A larger data CI size would favor sequential processing
  - A smaller CI size would favor direct processing
- Extra data buffers improve sequential processing
- Extra index buffers improve direct processing
- Separate data and index buffers
- Most accesses of a BCS are direct
- A data CI size (CISZ) of 4096 provides a compromise between minimizing data transfer time and reducing the occurrence of spanned records

Default NSR Buffering



Improved Buffering



NSR Key Direct Buffering

Key Direct Processing

- Specify BUFNI= all of the Index Set + 1 for the Sequence Set
- From a LISTCAT:
  - BUFNI = TI – (HURBA / CASZ) + 1

    where: TI is total number of index records

    CASZ is CISZ * CI/CA
- Set BUFND = 2

NSR Key Direct Buffering

CASZ=CISZ x CI/CA

```
DATA---------HSM.MCDS.DATA
  ATTRIBUTES
    KEYLEN----------------44  AVGLRECL-------------200  BUFSPACE----------10,240  CISIZE-----------4,096
    RKP--------------------0  MAXLRECL-----------2,040  CA SIZE---------737,280  CI/CA---------------180
    SHR(3,3)        RECOVERY  NOERASE     NOWRITECHECK  NOIMBED           NOREUSE  NONSPANNED
  STATISTICS
    REC TOTAL-------849,244   CI SPLITS--------145,018  EXCPS----------7,298,906  UPDATE/OUTPUT FLAG---ON
    REC DELETED----3,084,171  CA SPLITS------------776  EXTENTS----------------1
    REC INSERTED---3,170,297  FREESPACE CI%----------0  LAST UPDATED:
    REC UPDATED----8,645,711  FREESPACE CA%----------0        15 OCT 2000--07:39
    REC RETRIEVED-16,245,476  APPROX FREE CI'S-410,487
  ALLOCATION
    SPACE TYPE------CYLINDER  HI ALLO RBA-2,131,476,480
    SPACE PRI----------2,891  HI USED RBA---941,506,560
    SPACE SEC--------------0  APPROX FREE CA'S----1,614
INDEX--------HSM.MCDS.INDEX
  ATTRIBUTES
    KEYLEN----------------44  RECORD SIZE--------2,041  CA SIZE-----------43,008  CISIZE-----------2,048
    RKP--------------------0                                                      CI/CA----------------21
STATISTICS
    REC TOTAL----------1,315  SEQ SET SPLITS-------776  EXCPS----------2,928,653  INDEX:
    REC DELETED----------N/A  IND SET SPLITS--------12  EXTENTS----------------1  LEVELS----------------3
    REC INSERTED---------N/A  APPROX FREE CI'S---1,708  LAST UPDATED:             ENTRIES/SECT---------13
    REC UPDATED----------N/A                                  15 OCT 2000--07:39  SEQ SET RBA-----------0
    REC RETRIEVED--------N/A                                                      HI LEVEL RBA----260,096
```

HURBA

TI

BUFNI = TI – HURBA / CASZ + 1
BUFNI = 1,315 – (941,506,560 / (4096*180) + 1
=1,315 – 1,277 + 1
= 39

NSR Key Direct Buffering

NSR Key Direct Read Buffer Timings

Data Buffers	Index Buffers	EXCPs	CPU time	Elapsed time
2 (default)	1 (default)	199,000	13.62	4
30	1	199,000	13.60	4
2	3	118,000	8.46	2.4
2	39	99,376	7.27	2.0
2	50	99,376	7.42	2.0

Source: VSAM Demystified Redbook, SG24-6105, chapter 2

NSR Key Direct Buffering

STRNO, BUFNI and BUFND

- STRNO – Specifies the number of concurrent read requests for a BCS
  - Default is 2
  - Recommended to define catalogs with STRNO(3)
  - Use RMF to watch for enqueues on the SYSZRPLW.*bcsname* resource and alter STRNO as needed

NSR Key Direct Buffering

STRNO, BUFNI and BUFND (continued)

- BUFNI - Specifies the number of index buffers
  - The default is STRNO+2
  - Improve performance by specifying enough index buffers to contain the entire index set plus a sequence set control interval for each string
- BUFND - Specifies the number of data buffers
  - The default is BUFND = STRNO + 1
    - *This value is usually adequate*

Catalog Caching

Caching Basics

Types of Caching for Catalogs

- Two kinds of cache exclusively for catalogs
  - In-storage catalog (ISC) cache
  - Catalog data space cache (CDSC)
- A single catalog can be cached in either ISC or CDSC – not both at the same time
- Records cached in ISC or CDSC
  - Master catalog – all records accessed sequentially or by key (except for alias records)
  - User catalog – only records accessed by key
- Breakeven point is hit rate of 20%
  - Where the overhead of maintaining the cache becomes less than the cost of doing the I/O to the catalog

Catalog Data Space Cache

CDSC Caching – Recommended!

- Records reside in a VLF managed data space
- Catalogs are defined in the COFVLF*xx* PARMLIB member
- Catalogs are not limited to a set amount of storage
- When the data space is full, the least recently used record is removed
- When a change is detected, only the changed records are released
- Entire CDSC space for a BCS is invalidated only if excessive changes occur
- Reverts to ISC when turned off

Catalog Data Space Cache

Specifying CDSC

- In SYS1.PARMLIB(COFVLF*xx*)

```
CLASS NAME(IGGCAS)
    EMAJ(BCS1)
    EMAJ(BCS2)
    …
    MAXVIRT(4096|nnnn)
```

- Where '*nnnn*' is the number of 4K blocks of virtual storage to be used for caching catalogs
  - The minimum value is 256 (1 MB)
  - The default value is 4096 (16 MB)
- Allow 15 blocks (60 KB) per catalog

#SHAREorg

SHARE
Technology · Connections · Results

Catalog Sharing

SHARE in San Francisco 2013

Catalog Sharing

Sharing Catalogs

- A shared catalog is one that is eligible to be used by more than one system
- A catalog is shared if:

  SHAREOPTIONS (3 4) (this is the default)

  and

  It resides on a shared DASD volume
- Catalog address control structures are refreshed when updates are made to the catalog from any system

Catalog Sharing

Sharing Catalogs (continued)

- Sharing requires checking for changes before using cached records
  - To ensure that the ISC or CDSC contains current information
  - To ensure that the control blocks for the catalog are updated in the event the catalog has been extended or otherwise altered from another system
  - This checking maintains data integrity
  - This checking affects performance because the VVR for a shared catalog must be read before using the cached version of the record

Catalog Sharing

Key Points

- Convert the resource SYSIGGV2 to a SYSTEMS enqueue
  - Failing to do so could break catalogs
- Convert the resource SYSZVVDS to a SYSTEMS enqueue

Catalog Sharing Protocols

VVDS Mode

- Information necessary to communicate changes to other systems sharing the catalog is stored in a special ‘integrity VVR’ in the VVDS of the volume the catalog is defined on
- I/O to the VVDS is required to store and retrieve this information

ECS Mode

- Information that describes changes to a shared catalog is stored in the Coupling Facility (CF)
- The I/O to the VVDS that is required in VVDS mode is eliminated
- The SYSZVVDS RESERVE is avoided

Enhanced Catalog Sharing

Using Enhanced Catalog Sharing (ECS) Mode

- Substantial performance benefit for catalogs shared between systems in a sysplex
- Must include the ECS structure in the Coupling Facility Resource Manager (CFRM) policy
  - Policy name is SYSIGGCAS_ECS
- Must define one or more catalogs with the ECSHARING attribute
  - Use IDCAMS DEFINE or ALTER commands to set this attribute
  - Makes a catalog eligible for sharing with the ECS protocol

Enhanced Catalog Sharing

Using Enhanced Catalog Sharing (ECS) Mode (continued)

- ECS protocol will only be used:
  - If there is an active connection to the ECS cache structure
  - If ECS mode has been activated by the MODIFY CATALOG, ECSHR(AUTOADD) command
- Control catalogs using ECS with
  - MODIFY CATALOG ECSHR(REMOVE,*catname*)
    - *Does not remove the ECSHARING attribute from the catalog, but removes the catalog from ECS mode*
  - MODIFY CATALOG ECSHR(ENABLE,*catname*)

Enhanced Catalog Sharing

Restrictions on Usage

- A catalog cannot be shared using both the ECS mode and VVDS mode protocols at the same time
- Maximum of 1024 catalogs can be shared using ECS from a single system
- All systems sharing a catalog in ECS mode must be using the same Coupling Facility and be in the same GRS complex
- Attempting to use a catalog which is ECS active from a system outside the sysplex could break the catalog

#SHAREorg

SHARE
Technology · Connections · Results

Catalog Tuning

SHARE in San Francisco 2013

Creating a Balanced Catalog Environment

Most z/OS Systems:

- Have hundreds of thousands to millions of data sets cataloged
- Typically have 25 to 100 catalogs on a system
- Do the math:
  - Assume you have 1 million cataloged data sets and 25 catalogs (a fairly common ratio)
  - If the spread of data sets across catalogs is even, that works out to 40,000 data sets in each catalog
  - Therefore, if any one of the 25 catalogs suffers an outage, access to 40,000 data sets is lost until the catalog is recovered

Creating a Balanced Catalog Environment

The Problem

- Data sets are rarely distributed evenly across catalogs
- Most frequently, just a handful of catalogs contain a high percentage of the system’s application data sets
- If one of those catalogs suffers an outage, access to a far higher number of data sets will be lost

The Solution

- Analyze your z/OS environment to determine if your data sets are concentrated in a few user catalogs, and assess the risk to your critical business functions
- If they are concentrated, initiate a project to spread your cataloged data sets across more of your user catalogs

Using Space Efficiently

Recommendations:

- Specify FREESPACE(0 0)
  - Most BCSs will have unevenly distributed record insertion activity
  - Evenly distributed free space is of very little value
- Don’t worry about splits
  - Splits, both CI and CA, are the best technique available to handle this uneven distribution
  - To allow splits to obtain free space where it’s needed, the best approach is to leave them alone once they’ve occurred
- Reorganize catalogs as infrequently as you can
  - Reorganization removes all of the free space that has been imbedded within the CAs, requiring the splits to occur again

Reorganizing Catalogs

When Should a Catalog Be Reorganized?

- Recommended only when:
  - The catalog is approaching maximum extents
  - There is not enough room on the volume where the catalog resides to allow more extents to be taken
  - An attribute of the catalog needs to be changed
- Options for reorganizing a catalog:
  - IDCAMS EXPORT followed by IMPORT can be used to reorganize a catalog
  - Tivoli Advanced Catalog Management for z/OS can be used to reorganize a catalog without taking applications out of service

#SHAREorg

SHARE
Technology · Connections · Results

Final Thoughts

SHARE in San Francisco 2013

Summary Best Practices

Recommendations for ICF Catalogs

- Always apply current maintenance
  - Just applying HIPER is not good enough, but RSU is!
  - Failure to install fixes can result in broken catalogs/VSAM data sets or incorrect behavior
- Remove IMBED and REPLICATE when they are found
  - These keywords can impair performance and use more DASD space
  - z/OS 1.11 Health Check – opens all Catalogs in the system!
- Look into the use of CA Reclaim when at z/OS 1.12 or higher
  - *Be aware*: Disabled by default on a system level, but is enabled by default for all KSDSs without having to redefine the data set
  - Test before implementing
  - Consider use for VSAM data sets with many empty CAs

For More Information

- *z/OS DFSMS Access Method Services for Catalogs* – SC26-7394
- *z/OS DFSMS: Managing Catalogs* – SC26-7409
- *z/OS DFSMS Using Data Sets* – SC26-7410
- *VSAM Demystified* – IBM Redbook SG24-6105
- *Enhanced Catalog Sharing and Management* – IBM Redbook SG24-5594
- *ICF Catalog Backup and Recovery: A Practical Guide* – IBM Redbook SG24-5644

Q & A

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

IBM®, Tivoli®, z/OS®, DFSMSdfp™

The following terms are trademarks of Rocket Software, Incorporated in the United States and/or other countries:

Rocket®

Other company, product, and service names may be trademarks or service marks of others.