DB2 Recovery Planning and Modernization Techniques

Baltimore DB2 UG
Thursday, June 11

Hendrik (Hennie) Mynhardt

mynhardt@us.ibm.com
Disclaimer

© Copyright IBM Corporation 2015. All rights reserved.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS AND/OR SOFTWARE.

IBM, the IBM logo, ibm.com, z/OS, DB2, DFSMS, and DFSORT are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml

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

• DB2 Real Time Stats – intro related to Backup and Recovery
• DB2 DSNZPARMs affecting Backup and Recovery
• DFSMS and DASD FlashCopy – high level overview
• Backing up and recovering DB2 data
DB2 Real Time Statistics – basics
Objective of Real-time Statistics – rationale

Older DBA procedures and some home-grown monitor tools had no accurate data to identify objects that need maintenance

Spending time performing maintenance on static and unchanged objects ...
- inefficient use of DBA's time, waste batch window time
- waste of CPU – the cheapest utility is not to run the utility

DB2 system becoming large and complex
- A single DB2 for z/OS may have large amounts of tables/indexes - for ERP-packaged applications, it can be 80K +
- Requires skilled DBAs (and lots of time) to identify unused / static objects

Goal is to self-managed or automate the maintenance process
RTS – Collection in memory

Object RTS collection

• Allocate RTS blocks – pages in memory ... when?
  – Table Spaces - at **first update** - since the page set/partition is opened
  – Index Spaces - at **open time** - DB2 updates SYSINDEXSPACESTATS.LASTUSED

Externalization of in-memory blocks of info

• **STOP/START DATABASE SPACENAM** command
  • Flush in-memory statistics for all target objects

• **DB2 ACCESS DATABASE** command
  • ACCESS DB(dbname) SP(*) MODE(STATS)
  • ACCESS DB(*) SP(*) MODE(STATS)
## RTS Tables – SYSTABLESPACESTATS

<table>
<thead>
<tr>
<th>Global Statistics</th>
<th>Incremental Statistics</th>
<th>Incremental Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>REORG-</td>
<td>COPY-</td>
</tr>
<tr>
<td>DBID</td>
<td>LASTTIME</td>
<td>LASTTIME</td>
</tr>
<tr>
<td>PDID</td>
<td>INSERTS</td>
<td>UPDATEDPAGES</td>
</tr>
<tr>
<td>PARTITION</td>
<td>UPDATES</td>
<td>CHANGES</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>DELETES</td>
<td>UPDATEDLRSN</td>
</tr>
<tr>
<td>DBNAME</td>
<td>DISORGBLOB</td>
<td>UPDATTIME</td>
</tr>
<tr>
<td>NACTIVE</td>
<td>UNCLUSTINS</td>
<td>UPDATTLRSN</td>
</tr>
<tr>
<td>NPAGES</td>
<td>MASSDELETE</td>
<td>UPDATTIME</td>
</tr>
<tr>
<td>EXTENTS</td>
<td>NEARINDREF</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>FARINDREF</td>
<td></td>
</tr>
<tr>
<td>TOTALROWS</td>
<td>SCANACCESS</td>
<td></td>
</tr>
<tr>
<td>DATASIZE</td>
<td>HASHACCESS</td>
<td></td>
</tr>
<tr>
<td>UNCOMPRESSEDDATASIZE</td>
<td>CLUSTERSSENS</td>
<td></td>
</tr>
<tr>
<td>UPDATESTATSTIME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HASHLASTUSED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRIVETYPE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPFACILITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPDATESIZE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LASTDATACHANGE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOAD-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RLASTTIME</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## RTS Tables – SYSINDEXSPACE STATS

<table>
<thead>
<tr>
<th>Global Statistics</th>
<th>Incremental Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>REORG-</td>
</tr>
<tr>
<td>DBID</td>
<td>REBUILDLASTTIME</td>
</tr>
<tr>
<td>ISOBID</td>
<td>LASTTIME</td>
</tr>
<tr>
<td>PSID</td>
<td>UPDATES</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>DELETES</td>
</tr>
<tr>
<td>NACTIVE</td>
<td>PSEUDEODELETE</td>
</tr>
<tr>
<td>NLEVELS</td>
<td>MASSDELETE</td>
</tr>
<tr>
<td>NPAGES</td>
<td>LEAFFAR</td>
</tr>
<tr>
<td>NLEAF</td>
<td>INDEXACCESS</td>
</tr>
<tr>
<td>EXTENTS</td>
<td>LOAD-</td>
</tr>
<tr>
<td>SPACE</td>
<td></td>
</tr>
<tr>
<td>TOTALENTRIES</td>
<td></td>
</tr>
<tr>
<td>LASTUSED</td>
<td></td>
</tr>
<tr>
<td>UPDATESTATSTIME</td>
<td></td>
</tr>
<tr>
<td>DRIVETYPE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOAD-</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Inserts, Deletes, MassDelete
Unused or (in)activity of INDEXES

—LASTUSED column in SYSINDEXSPACESTATS
  • Is a date field
  • Consider using for identifying which IXs to drop
  • The date indicates the index is last used for SELECT, FETCH, searched UPDATE, searched DELETE, or used to enforce referential integrity constraints.
  • The default value is 01/01/0001.

—REORGINDEXACCESS column in SYSINDEXSPACESTATS
  • # of times the IX was accessed (read and updates) since last reorg or since creation
  • NULL denotes never used
There is currently no historical capability in RTS in DB2 itself.

IBM provides RTS history via the no-charge *DB2 Management Console* – see next slide.

Create a history table manually:

1. **CREATE**
   
   ```
   CREATE
   SYSIBM.(TABLE | INDEX)SPACESTATS_HIST
   LIKE
   SYSIBM.SYS(TABLE | INDEX)SPACESTATS
   ```

   – then add

   ```
   CAPTURE_TIME AS TIMESTAMP NOT NULL WITH DEFAULT column
   ```

Periodically insert into RTS history tables with a sub select from the RTS tables those rows that aren’t already in the history tables; and delete old information.

– Some customers do this weekly, others monthly – depending on needs.
Management Console - Holistic Dashboards of DB2 Objects

...from DB2 Catalog

...from Automation Tool Evaluations

...recorded by RTS_SNAPSHOT

...captured by UET
RTS History in DB2 Management Console

DB2 Stored Procedure built into the MC to perform regular snapshot of RTS

- Autonomics part uses Tables and Stored Procedure
- Ability to execute **RTS_SNAPSHOT** Stored Procedure
  - Via DB2 Administrative Task Scheduler
  - Your own existing job scheduler

RTS history displayed online
Some uses for RTS - Activity

Object activity

• How active are my DB2 objects?
• What activity has taken place for a specific time for TS’ and IX’s
• Use `UPDATESTATSTIME`

```
SELECT DBNAME, NAME, PARTITION, UPDATESTATSTIME
FROM SYSIBM.TABLESPACESTATS
WHERE (JULIAN_DAY(CURRENT DATE) – JULIAN_DAY(UPDATESTATSTIME)) <= 14
AND NAME = xxx;
```

Show me the activity during the last 14 days
Some uses for RTS – Utility executions

- When was the last time a utility was run against my objects?
- When was **COPY**, REORG, LOAD REPLACE, and RUNSTATS last executed against objects ..

```
SELECT DBNAME, NAME, PARTITION, TOTALROWS, NACTIVE, SPACE, EXTENTS, UPDATESTATSTIME, STATSLASTTIME, LOADRLasttime, REORGLASTTIME, COPYLASTTIME FROM SYSIBM.TABLESPACESTATS ORDER BY DBNAME, NAME, PARTITION
```
Some uses for RTS - Backups

Use RTS backup information ...

— For copies based upon changes, consider INCREMENTAL* / FULL

— For sizing – e.g. use DB2 FCIC based upon TEMPLATE switching

— Use RTS history tables for RTO planning:
  • Use SPACE or CHANGES to group objects for backup purposes

— Use RTS / history UPDATETIME to determine when backups are run –
  take back control of backups
  • Determine when and how many times backups run
  • Remove duplicates – where needed
  • Columns of interest

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COPYLASTTIME</td>
<td>last IC timestamp</td>
</tr>
<tr>
<td>COPYUPDATEPAGES</td>
<td># of pages updated since last IC</td>
</tr>
<tr>
<td>COPYCHANGES</td>
<td># of rows changed since last IC</td>
</tr>
</tbody>
</table>
DSNZPARMS affecting Backup and Recovery Utilities
DSNZPARMS affecting utilities

Parms affecting **dataset allocation** operations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/s</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTSORTAL</td>
<td>Yes/No</td>
<td>Enable dynamic allocation of SORTWK datasets. Uses DB2 RTS to estimate the number of rows to sort</td>
</tr>
<tr>
<td>IGNSORTN</td>
<td>Yes/No</td>
<td>Ignore SORTNUM statements</td>
</tr>
</tbody>
</table>

Parms affecting **utility timeout** operations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/s</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTIMOUT</td>
<td>1-254 6</td>
<td>How long, in number of resource values, that a utility or utility command is to wait for a resource. The utility or utility command waits until a lock or all claims on a resource of a particular claim class is released.</td>
</tr>
</tbody>
</table>

When using the default (6), a utility can wait six times longer than an SQL application for a resource. This option allows utilities to wait longer than SQL applications to access a resource.
##Parms affecting **SORT** operations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/s</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOLTDEFT</td>
<td>SYSDA</td>
<td>Device type or unit name that is to be used by DB2 utilities for dynamically allocating temporary data sets. Used for COPY CONCURRENT (DFSMSdss) and CHECK datasets.</td>
</tr>
<tr>
<td>UTIL_TEMP_STORCLAS</td>
<td></td>
<td>SMS storage class that the CHECK INDEX, CHECK DATA, and CHECK LOB utilities are to use when allocating temporary shadow data sets. (These utilities allocate shadow data sets when the SHRLEVEL CHANGE option is used).</td>
</tr>
<tr>
<td>DB2SORT</td>
<td>ENABLE/DISABLE</td>
<td>DB2 utilities are to use DB2 Sort instead of DFSORT for utility sort processing when DB2 Sort is installed.</td>
</tr>
</tbody>
</table>
### DSNZPARMS affecting utilities

#### Parms affecting invocation of **FlashCopy** operations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/s</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLASHCOPY_COPY</td>
<td>YES/NO</td>
<td>Specifies that the COPY utility uses FC technology when the FLASHCOPY option / keyword is not specified in the util control statement</td>
</tr>
<tr>
<td>FLASHCOPY_LOAD</td>
<td>YES/NO</td>
<td>LOAD utility uses FC technology when the FLASHCOPY option is not specified in the control statement – inline copy</td>
</tr>
<tr>
<td>FLASHCOPY_REORG_TS</td>
<td>YES/NO</td>
<td>Inline copy at load phase for REORG – when not coded on LOAD control statement</td>
</tr>
<tr>
<td>FLASHCOPY_REORG_IX</td>
<td>YES/NO</td>
<td>As above</td>
</tr>
<tr>
<td>FLASHCOPY_REBUILD_IX</td>
<td>YES/NO</td>
<td>Can Rebuild IX use FlashCopy by default when not specified on the REBUILD INDEX utility control statement</td>
</tr>
</tbody>
</table>
# DSNZPARMS affecting utilities

## Parms affecting **FlashCopy** operations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/s</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECK_FASTREPLICATION</td>
<td>PREFERRED, REQUIRED</td>
<td>Specifies the type of replication that DSS COPY uses to copy objects to shadow data sets</td>
</tr>
<tr>
<td>REC_FASTREPLICATION</td>
<td>NONE, PREFERRED, REQUIRED</td>
<td>Should RECOVER utility use FC to recover from a FlashCopy image copy</td>
</tr>
</tbody>
</table>
| FLASHCOPY_PPRC             | NONE, PREFERRED, REQUIRED, blank | • Whether DFSMSdss preserves mirroring while processing a DB2 utilities request  
• Whether the target device pair is allowed to go to duplex pending state                                                                                                                  |
| FCCOPYDDN                  | HLQ.&DB..&SN..N&DS NUM..&UQ.   | Template for the output VSAM dataset name for any FlashCopy image copy.                                                                                                                                    |
## DSNZPARMS affecting utilities

**Parms affecting BACKUP - , RESTORE SYSTEM**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/s</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM-LEVEL_BACKUPS</td>
<td>YES, NO</td>
<td>Enables the RECOVER utility to use SLB as input for object-level recoveries</td>
</tr>
<tr>
<td>RESTORE_RECOVER_FROMDUMP</td>
<td>YES, NO</td>
<td>Specifies if RESTORE SYSTEM and RECOVER can use SLB dump on tape as input for recovery. Yes – use tape. No – use only disk</td>
</tr>
<tr>
<td>UTILS_DUMP_CLASS_NAME</td>
<td></td>
<td>Name of the DFSMSHsm dump class for BACKUP SYSTEM dump to tape</td>
</tr>
<tr>
<td>RESTORE_TAPEUNITS</td>
<td>NOLIMIT or 1 - 255</td>
<td>The number of tapedrives RESTORE SYSTEM allocates for restore from dump</td>
</tr>
</tbody>
</table>
### Parm affecting RECOVER utility

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/s</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>REC_FASTREPLICATION</td>
<td>REQUIRED, PREFERRED, NONE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• REQUIRED - The RECOVER utility forces use of FlashCopy when performing recovery from a FlashCopy image copy, to ensure that recovery occurs as quickly as possible. However, this option will cause RECOVERY to fail if FlashCopy cannot be used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PREFERRED - The RECOVER utility uses FlashCopy only if FlashCopy support is available.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NONE - The RECOVER utility will use standard input/output to restore the FlashCopy image copy. This setting is not permitted when the FLASHCOPY_PPRC parameter is set to PREFERRED or REQUIRED.</td>
</tr>
</tbody>
</table>
DFSMS and Storage for the DBA
DB2 and IBM storage integration

- DB2 Cloning Tool
  - Volume and dataset-level clone

- DB2 Recovery Expert
  - TS and volume-based B/R

- z/OS
  - DFSMSShsm
    - Volume functions
  - DFSMSdss
    - Dataset functions

- IBM DS8000
  - FlashCopy
    - Many flavors
  - Metro Mirror (PPRC)
DFSMSdss, DFSMSHsm and IBM FlashCopy

DFSMS and related components

- Data Facility Storage Management Subsystem – DFSMS has the following components:
  - DFSMSdfp, a base element of z/OS
  - DFSMSdss, an optional feature of z/OS
  - DFSMSHsm, an optional feature of z/OS
  - DFSMSrmm, an optional feature of z/OS
  - DFSMSstvs, an optional feature of z/OS
DFSMS – components 101

- **DFSMSdfp** - Provides storage, data, program, and device management. It is comprised of programs like IDCAMS, SMS, ISMF etc.

- **DFSMSdss** - Provides data movement, copy, backup, and space management functions – known as program ADRDSSU

- **DFSMShsm** - Provides backup, recovery, migration, and space management functions. It does invoke DFSMSdss for certain of its functions.

- **DFSMSrmm** - Provides management functions for tapes

- **DFSMSstvs** - Enables batch jobs and CICS online transactions to update shared VSAM data sets concurrently.

- **Used by DB2 utilities for fast replication functions**
• DFSMSdss

–The following utilities can invoke DFSMSdss’ copy function:

• CHECK DATA with SHRLEVEL CHANGE
• CHECK INDEX with SHRLEVEL CHANGE
• CHECK LOB with SHRLEVEL CHANGE
• COPY with FLASHCOPY YES or FLASHCOPY CONSISTENT
• LOAD with FLASHCOPY YES or FLASHCOPY CONSISTENT
• REBUILD INDEX with FLASHCOPY YES or FLASHCOPY CONSISTENT
• RECOVER with FLASHCOPY YES or FLASHCOPY CONSISTENT
• REORG INDEX with FLASHCOPY YES or FLASHCOPY CONSISTENT
• REORG TABLESPACE with FLASHCOPY YES or FLASHCOPY CONSISTENT
IBM FlashCopy

- Produces instant copy of a volume or dataset
- Source and Target volumes require real disk space AND must be on the same DASD ssid
- Source and target volumes must be same track geometry

```
//STEP1  EXEC PGM=ADRDSSU
//SYSPRINT DD  SYSOUT=*  
//SYSIN   DD *
  COPY DATASET(  -
    INCLUDE(  -
      DB2A.DSNDBC.FLASHDB1.**  -
      DB2A.DSNDBC.MYOWNDB.** )) -
    FASTREPLICATION(PREFERRED)
```

ADR711I (001)-NEWDS(01), DATA SET BEPD.DSNDBC.BEDPRDDB.BAADJUDS.I0001.A001 HAS BEEN ALLOCATED WITH NEWNAME BEPD.FAST.BEDPRDDB.BAADJUDS.I0001.A001 USING STORCLAS STANDARD, DATAACLAS

ADR806I (001)-T0MI (03), DATA SET BEPD.DSNDBC.BEDPRDDB.BAADJUDS.I0001.A001 COPIED USING A FAST REPLICATION FUNCTION
FlashCopy basics ...

Source Volumes      Target Volumes

Flashcopy command sent to HSM
- Relationship established
- Bitmap created
- Source vol avail for updates
LOGICAL COPY complete

Tracks are copied from source to target volumes
- Bitmap updated
PHYSICAL COPY
FlashCopy Basics ...

- **FlashCopy – Volume and dataset level**
  - *Volume level* used by BACKUP / RESTORE SYSTEM utilities
  - *Dataset level* used by DB2 10+ FCIC and other utilities

- **Incremental** – only for volume-based copies
  - Supported by BACKUP / RESTORE SYSTEM

- **FlashCopy SE – Space efficient**
  - Volume level only
  - Uses ‘virtual’ Space Efficient volumes as target volumes and a repository
  - Only uses the space needed for updates to the source volume

- ...

...
All I/O and CPU costs are host based and charged
Fast Replication – Storage based – fast replication

An instant copy of a volume/data set at a specific point in time

Data movement (CPU and I/O) offloaded to storage processor

Frees up resources on host processor

Volume and dataset level FlashCopy on IBM, EMC and HDS DASD

No host or I/O costs for the data movement
Fast Replication uses by DB2 – not new

**DB2 V8**
- BACKUP SYSTEM
- RESTORE SYSTEM
- Dataset FC support for CHECK INDEX SHRLEVEL CHANGE

**DB2 9**
- Incremental FC support for BACKUP SYSTEM
- Dataset FC for RECOVER with system-level backup (SLB) as input
- Dataset FC for CHECK DATA SHRLEVEL CHANGE and CHECK LOB SHRLEVEL CHANGE

**DB2 10**
- Dataset FC for COPY
- Dataset FC for inline copy in REORG TABLESPACE, REORG INDEX, REBUILD INDEX and LOAD
- FC image copies with consistency and no application outage (SHRLEVEL CHANGE)
- FCIC accepted as input to RECOVER, COPYTOCOPY, DSN1COPY, DSN1COMP and DSN1PRNT
COPYing your data
DB2 Backup utilities – two flavors

**COPY**

- **Sequential COPY**
- **CONCURRENT COPY**
- **FLASHCOPY IC**

**by OBJECT**

- DB1.TS1.COPY
- DB1.TS2.COPY

**by VOLUME**

- All Volumes as a group
- By DFSMSShsm COPYPOOL *

**BACKUP SYSTEM**

- DSN$DB2$DB *
- DSN$DB2$LG
Activated through utility statement **FLASHCOPY YES** or DSNZPARMs

- **FLASHCOPY_COPY**
- **FLASHCOPY_LOAD**
- **FLASHCOPY_REORG_TS** & **FLASHCOPY_REORG_IX**
- **FLASHCOPY_REBUILD_IX**

```plaintext
//UTIL EXEC
DSNUPROC,SYSTEM=DB0B,UID=',A',UTPROC=""
//DSNUPROC.SYSCOPY DD
DSN=DB0BI.DB1.TS1.FC01,
  // DISP=(MOD,CATLG), UNIT=SYSDA,
  // SPACE=(16384,(20,20),,,ROUND)
//DSNUPROC.SYSIN DD *
COPY TABLESPACE DB1.TS1

**FLASHCOPY_COPY=NO**

Sequential file: 
**DB0BI.DB1.TS1.FC01**
```

```plaintext
//UTIL EXEC
DSNUPROC,SYSTEM=DB0B,UID=',A',UTPROC=""
//DSNUPROC.SYSCOPY DD
DSN=DB0BI.DB1.TS1.FC01,
  // DISP=(MOD,CATLG), UNIT=SYSDA,
  // SPACE=(16384,(20,20),,,ROUND)
//DSNUPROC.SYSIN DD *
COPY TABLESPACE DB1.TS1

**FLASHCOPY_COPY=YES**

VSAM cluster: 
**DB0BI.DB1.TS1.LOCAL.COPYFC**
(name generated using FCCOPYDDN)**
```
COPY TABLESPACE DB1.TS1
FLASHCOPY YES
COPYDDN(SYSCOPY)

SYSCOPY rows

<table>
<thead>
<tr>
<th>DSNUM</th>
<th>ICTYPE</th>
<th>ICDATE</th>
<th>START_RBA</th>
<th>PIT_RBA</th>
<th>STYPE</th>
<th>ICBACKUP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>F</td>
<td>100827</td>
<td>00002DC978EF</td>
<td>00002DC97923</td>
<td>Q</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>F</td>
<td>100827</td>
<td>00002DC978EF</td>
<td>00002DC97923</td>
<td>T</td>
<td>FC</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>100827</td>
<td>00002DC978EF</td>
<td>00002DC97923</td>
<td>T</td>
<td>FC</td>
</tr>
<tr>
<td>2</td>
<td>F</td>
<td>100827</td>
<td>00002DC978EF</td>
<td>00002DC97923</td>
<td>T</td>
<td>FC</td>
</tr>
<tr>
<td>1</td>
<td>F</td>
<td>100827</td>
<td>00002DC978EF</td>
<td>00002DC97923</td>
<td>T</td>
<td>FC</td>
</tr>
</tbody>
</table>
SYSCOPY Records introduced in DB2 10

- Records for FlashCopy copies have an ICTYPE=F and ICBACKUP=FC

- START_RBA is the point at which the pages for the object was externalized to disk

- PIT_RBA is the point of consistency, i.e. more or less the point when the FLASHCOPY relationship was established

- SYSCOPY records for each piece/part.
  - STYPE = T - FlashCopy copy is consistent
  - STYPE = N - FlashCopy copy is not consistent
  - STYPE = Q - Sequential copy is consistent
  - STYPE = U - Sequential copy is not consistent
  - TTYPE - one character indicating type of utility which made the copy

- The SYSCOPY record for a RECOVER with the BACKOUT keyword specified has a ICTYPE= P and a STYPE=B.
COPY TABLESPACE DSN8D81A.DDS1011B

14:14:46.55 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = DNET963.DNET963D
14:14:46.56 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
14:14:46.57 DSNUGUTC - COPY TABLESPACE DSN8D81A.DDS1011B DSNUM ALL FULL YES SH
14:14:46.72 DNSUBBID - COPY PROCESSED FOR TABLESPACE DSN8D81A.DDS1011B

   NUMBER OF PAGES=3
   AVERAGE PERCENT FREE SPACE PER PAGE = 32.66
   PERCENT OF CHANGED PAGES = 0.00
   ELAPSED TIME=00:00:00
16 14:14:46.79 DNSUBAFL - DB2 IMAGE COPY SUCCESSFUL FOR TABLESPACE DSN8D81A.DDS

You can use the output as input for:
- DSN1COMP
- DSN1COPY
- DSN1PRNT

Output format

<table>
<thead>
<tr>
<th>Header pages</th>
<th>Data Pages</th>
</tr>
</thead>
</table>

You can use the output as input for:
- DSN1COMP
- DSN1COPY
- DSN1PRNT
COPY TABLESPACE DSN8D81A.DDS1011B CONCURRENT

<table>
<thead>
<tr>
<th>PAGE 0001 5695-DF175</th>
<th>DFSMSDSS V1R12.0 DATA SET SERVICES</th>
<th>2012.016 14:19:19 PARALLEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PARALLEL'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DUM OPT(3) DATAS(INCL(DSNSCAT.DSNDBC.DSN8D81A.DDS1011B.J0001.A%%%)) - CAN CONC SHA TOL(ENQF) WAIT(0,0) - OUTDD(SYSCOPY)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'DUM'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADR109I (R/I)-RI01 (01), 2012.016 14:19:16 INITIAL SCAN OF USER CONTROL STATEME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADR014I (SCH)-DSSU (02), 2012.016 14:19:16 ALL PREVIOUSLY SCHEDULED TASKS COMPL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You CANNOT use the output as input for:

- DSN1COMP
- DSN1COPY
- DSN1PRNT

Output format

| Tracks | | | | | |
|--------|---|---|---|---|
COPY TABLESPACE DSN8D81A.DDS1011B FLASHCOPY YES

FLASHCOPY YES
DSNU421I 016 15:20:51.67 DSNUGFUM - START OF DFSMS MESSAGES
PAGE 0001 5695-DF175 DFSMSDSS V1R12.0 DATA SET SERVICES
ADR030I (SCH)-PRIME( 0), DCB VALUES HAVE BEEN MODIFIED FOR SYSPRINT
COPY DATASET(INCLUDE( -
      DSNBCAT.DSNDBC.DSN8D51A.DSN8S51E.I0001.A001 , -
      ...
RENAMEU( -
      (DSNBCAT.DSNDBC.DSN8D51A.DSN8S51E.I0001.A001 , -
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'COPY '
ADR109I (R/I)-RI01 (01), 2012.016 15:20:51 INITIAL SCAN OF USER CONTROL STATE
ADR050I (001)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (001)-STEND(01), 2012.016 15:20:51 EXECUTION BEGINS
ADR711I (001)-NEWDS(01), DATA SET DSNBCAT.DSNDBC.DSN8D51A.DSN8S51E.I0001.A001
      HLQ.DSN8D51A.DSN8S51E.N00001.DNDJZ8MW USING STORCLAS
      USRMGMT
ADR755W (001)-PROTD(01), SOURCE DATA SET DSNBCAT.DSNDBC.DSN8D51A.DSN8S51E.I00
      TARGET DATA SET HLQ.DSN8D51A.DSN8S51E.N00001.DNDJZ8M

Use COPYTOCOPY to make sequential copy when needed to DASD or TAPE

Output format VSAM Cluster
Significant CPU and elapsed time savings:

CPU time per object (z10)

Elapsed time per object (z10)
FLASHCOPY CONSISTENT

COPY TABLESPACE DB1.TS1

SHRLEVEL CHANGE

FLASHCOPY CONSISTENT

Point of consistency, i.e. PIT_RBA

RBA of last bufferpool externalization

START_RBA

FCIC

UR3

UR2

UR1

LOG
FlashCopy – in-line FCIC

//DSNUPROC.SYSIN DD *
COPY TABLESPACE HASHDB.HASHTS FLASHCOPY CONSISTENT SHRLEVEL CHANGE

1PAGE 0001 5695-DF175 DFSMSDSS V1R11.0 DATA SET SERVICES 2010.239 18:58
-ADR030I (SCH)-PRIME(01), DCB VALUES HAVE BEEN MODIFIED FOR SYSPRINT
COPY DATASET(INCLUDE(-
DB0BD.DSNDDBC.HASHDB.HASHTS.J0001.A001 )) -
RENAMEU(-
(DB0BD.DSNDDBC.HASHDB.HASHTS.J0001.A001, -
DB0BI.HASHDB.HASHTS.N00001.CY25YIE5 )) -
REPUnc ALLDATA(*) ALLEXCP CANCELERROR SHARE -
WRITECHECK TOLERATE(ENQF)
ADR101I (R/I)-RI01(01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'COPY'
ADR109I (R/I)-RI01(01), 2010.239 18:58:45 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR050I (001)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
0ADR006I (001)-STEND(01), 2010.239 18:58:45 EXECUTION BEGINS
0ADR711I (001)-NEWDS(01), DATA SET DB0BD.DSNDDBC.HASHDB.HASHTS.J0001.A001 HAS BEEN ALLOCATED WITH NEWNAME
   DB0BI.HASHDB.HASHTS.N00001.CY25YIE5 USING STORCLAS DB0BDATA, DATACLAS DB0B,
   AND MGMTCLAS MCDB22
0ADR806I (001)-T0MI(03), DATA SET DB0BD.DSNDDBC.HASHDB.HASHTS.J0001.A001 COPIED USING A FAST REPLICATION FUNCTION
0ADR801I (001)-DDDS(01), DATA SET FILTERING IS COMPLETE. 1 OF 1 DATA SETS WERE SELECTED: 0 FAILED 0ADR454I (001)-DDDS
(01), THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED
FlashCopy – Recover using FCIC – DB2 10

RECOVER TABLESPACE DSN00020.FLASHTES TOCOPY

`DB0AU.DSN00020.FLASHTES.N00001.CV11XBMY`

`DSNU421I  173 17:40:46.58 DSNUGFUM - START OF DFSMS MESSAGES`

`1PAGE 0001  5695-DF175 DFSMSDSS V1R11.0 DATA SET SERVICES  2010.173 17:40`

`-ADR030I (SCH)-PRIME(01), DCB VALUES HAVE BEEN MODIFIED FOR SYSPRINT`

COPY DATASET(INCLUDE(`

DB0AU.DSN00020.FLASHTES.N00001.CV11XBMY `)) -

RENAMEU(`

(DB0AU.DSN00020.FLASHTES.N00001.CV11XBMY , -

DB0AU.DSNDBC.DSN00020.FLASHTES.I0001.A001 )) -

ALLDATA(*) ALLEXCP CANCELERROR SHARE -

REPUNC TOLERATE(ENQF) DEBUG(FRMSG(DTL))`

`ADR101I (R/I)-R101 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'COPY '

ADR109I (R/I)-R101 (01), 2010.173 17:40:46 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED`

`ADR050I (001)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE`

`ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK`

`ADR006I (001)-STEND(01), 2010.173 17:40:46 EXECUTION BEGINS`

`ADR442I (001)-PREVS(01), DATA SET DB0AU.DSN00020.FLASHTES.N00001.CV11XBMY PREALLOCATED WITH NEW NAME`

`DB0AU.DSNDBC.DSN00020.FLASHTES.I0001.A001, FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED`

`DB0AU.DSN00020.FLASHTES.N00001.CV11XBMY`
FLASHCOPY CONSISTENT

COPY TABLESPACE DB1.TS1
SHRLEVEL CHANGE
FLASHCOPY CONSISTENT

Point of consistency, i.e. PIT_RBA
RBA of last bufferpool externalization

START_RBA
UR1
UR2
UR3

FCIC
LOG
BACKUP SYSTEM (using DFSMShsm)

DB2 BACKUP / RESTORE SYSTEM

DFSMShsm commands
FRBACKUP, FRECOV, FRDELETE

DFSMsdss Fast Replication

DB2 Recovery Structures

ICF User Catalogs
Active Logs
Archive Logs

SMS Group, DB2 System Backup Volume Pool, Target Unit Range

DB9ACPB
Example of a **BACKUP SYSTEM** job – JCL and output

```plaintext
//BACKUP EXEC DSNUPROC,SYSTEM=D9CG,
//    UID='SYSBACK'
//STEPLIB DD DISP=SHR,DSN=DB9C9.SDSNEXIT
//    DD DISP=SHR,DSN=DB9C9.SDSNLOAD
//SYSIN  DD *
  BACKUP SYSTEM FULL

ARC1801I  FAST REPLICATION BACKUP IS STARTING FOR COPY POOL DSN$DB9C$DB, AT 17:07:44 ON 2008/02/12,
  TOKEN=X'C4F9C3F1C1F0D91FD1AE7903C1FD058C8ED3'
ARC0640I  ARCFRTM - PAGE 0001  5695-DF175  DFSMSDSS V1R09.0 DATA SET SERVICES  2008.043 17:07
ARC0640I  ARCFRTM - ADR035I (SCH)-PRIME(06), INSTALLATION EXIT ALTERED BYPASS FAC CLASS CHK DEFAULT TO YES
ARC0640I  ARCFRTM - PARALLEL
ARC0640I  ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PARALLEL'
ARC0640I  ARCFRTM - COPY IDY(SBOX5S) ODY(SBOX6A) DUMPCOND FR(REQ) PUR ALLX ALLD(*)
ARC0640I  ARCFRTM - ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'COPY
...
ARC0640I  ARCFRTM - ADR012I (SCH)-DSSU (01), 2008.043 17:07:45 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000
ARC1805I  THE FOLLOWING 00004 VOLUME(S) WERE SUCCESSFULLY PROCESSED BY FAST REPLICATION BACKUP OF COPY POOL DSN$DB9C$DB
ARC1805I (CONT.) SBOX5S
ARC1805I (CONT.) SBOX5T
ARC1805I (CONT.) SBOX5U
ARC1805I (CONT.) SBOX5V
ARC1802I  FAST REPLICATION BACKUP HAS COMPLETED FOR COPY POOL DSN$DB9C$DB, AT 17:07:45 ON 2008/02/12, FUNCTION RC=0000,
    MAXIMUM VOLUME RC=0000
```
Incremental FlashCopy (BACKUP SYSTEM not DB2 COPY)

```
//BACKUP EXEC PGM=DSNUTILB,PARM=(DB9A,BKSYS),REGION=OM
//STEPLIB DD DISP=DHR,DSN=DB9A9.SDSNEXIT
// DD DISP=SHR,DSN=DB9A9.SDSNLOAD
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
BACKUP SYSTEM ESTABLISH FCINCREMENTAL

DSNUGUTC - BACKUP SYSTEM FULL ESTABLISH FCINCREMENTAL
DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA STARTING,
COPYPOOL = DSN$DB9A$DB
TOKEN = X'E2D5F6F1C29ED4388D5B2A46C29ED30F6510'.
DSNUVBBD - THE SYSTEM LEVEL BACKUP TAKEN IS AN INCREMENTAL FLASHCOPY OF
THE DATABASE
COPYPOOL = DSN$DB9A$DB
TOKEN = X'E2D5F6F1C29ED4388D5B2A46C29ED30F6510'
ELAPSED TIME = 00:00:05.
DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS STARTING,
COPYPOOL = DSN$DB9A$LG
TOKEN = X'E2D5F6F1C29ED4388D5B2A46C29ED30F6510'.
DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS COMPLETED SUCCESSFULLY,
COPYPOOL = DSN$DB9A$LG
TOKEN = X'E2D5F6F1C29ED4388D5B2A46C29ED30F6510'
ELAPSED TIME = 00:00:00.
```
RESTORE SYSTEM uses the RBLP (Recovery Based Log Point)
which is stored in DBD01 Page 0

- The is the most recent system checkpoint prior to BACKUP SYSTEM
COPY BEST PRACTICES

• Use SHRLEVEL CHANGE unless consistent copies are essential

• Use PARALLEL keyword to exploit parallelism (without the n)

• Consider OPTIONS EVENT(ITEMERROR, SKIP)
  – Sets UTRW state only for duration of copy of individual page set
  – But increases COPY overhead
    • Serialization required for each pageset on the fly

• Consider taking incremental copies and using MERGECOPY
  – MERGECOPY marks relevant page set UTRW
COPY BEST PRACTICES

- Copy indexes on large, critical tables
  - Particularly if rarely or never updated
  - Only drawback – increase in SYSLGRNX & SYSCOPY recording
  - Automatically included in MODIFY RECOVERY

- Consider CONCURRENT COPY
  - Can reduce CPU & elapsed time
  - Uses DFSMSdss backup/restore
  - Prohibits use of DSN1COPY & UNLOAD from copy
All FCICs are registered in SYSCOPY

— New columns in SYSCOPY

- **ICTYPE** – *L* – SQL
- **DSVOLSER** – Used for FLASHCOPY CONSISTENT
  - Contains VOLSER, checkpoint RBA of DS member etc
- **ICBACKUP** – **FC** – FlashCopy Copy
- **STYPE**
  - For FULL COPIES – **STYPE F**:
    - **N** – FCIC not consistent
    - **Q** – Sequential copy IS consistent
    - **T** – FCIC is consistent
COPY Utility – FCIC considerations

FCIC output datasets are VSAM files
- To make it a sequential copy run COPYTOCOPY
- Challenges:
  - DASD space is required for the output copy
  - When a FCIC is archived, the RECOVER utility will bypass it and use the previous entry
  - FCIC to tape not possible
  - GDGs not feasible
  - In data mirror environment, there maybe challenges depending on implementation – Metro-mirror, Global mirror (XRC.PPRC)
  - DB2 Recovery Expert solves some of these challenges
Backup solutions - summary

- Multiple options

- BACKUP SYSTEM
  - Volume-level FlashCopy
  - DASD investment required
  - For DBAs can be complex to set up & administer without tools, but invocation simple
  - Must understand any limitations that currently exist
    - E.g. DASD mirroring issues, dataset movement issues, etc.

- Sequential image copies
  - Tried and trusted solution since V1.1

- Other external backups, such as volume-level backups, DSN1COPY
  - Outside of DB2’s control
  - Requires careful management and co-ordination

- In-line FlashCopy in DB2 10
DB2 RECOVER and RESTORE SYSTEM

DSNUTILB

RECOVER

RESTORE SYSTEM

by OBJECT

SYSCOPY

DB2 SLB

- Sequential COPY
- CONCURRENT COPY
- FLASHCOPY IC

* Enabled via ZPARM

DB1.TS1.COPY

DB1.TS2.COPY

by VOLUME

B SDS

HSM

- All Volumes as a group
- By DFSMSHsm COPYPOOL *

DSN$DB2$DB

* Enabled via ZPARM

DSN$DB2$LG
RECOVER to point in time with consistency

_Avoid_ need for QUIESCEs
• Recover data to one of the following:
  – Full recovery – restores objects to the current state
  – Partial recovery or Point in Time (PIT)
• Can recover a single object or a list of objects, including LISTDEFs
• Recover using Image Copy’s, Flashcopy image copy, System Level Backup, or the DB2 Log
DB2 RECOVER utility – some recommendations

- Maximize exploitation of parallel restore and Fast Log Apply
- Recover an index to the last full image copy without deleting and redefining data sets:
  - `RECOVER INDEX qualifier.ixname REUSE`
  - `TOLASTFULLCOPY`
- Copy indexes and include in recovery list, particularly for PIT recovery – avoid sort costs
- Split off page sets that are not updated and recover separately – utilize RTS
- For PIT recovery, include whole RI set in same RECOVER statement
- For PIT recovery, include base and aux objects in same RECOVER statement – enforced via ZPARM
- Consider copying Indexes now – with LOGAPPLY for IXs
RESTORE SYSTEM – object level

**FIC**

**Updates**

**BACKUP SYSTEM**

**Updates**

**Incr. FC Backup**

**Recover Object**

**DB2R7.D9C1.IC1.DB2R7.D2R7**

**CATALOGED**

13:39:35.20 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = DB2R7.DB2R7001
13:39:35.24 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
13:39:35.30 DSNUGUTC - COPY TABLESPACE DB2R7.DB2R7 DSNUM ALL FULL YES
13:39:35.44 DSNUBBID - COPY PROCESSED FOR TABLESPACE DB2R7.DB2R7

NUMBER OF PAGES=3
AVERAGE PERCENT FREE SPACE PER PAGE = 25.33
PERCENT OF CHANGED PAGES = 33.33
ELAPSED TIME=00:00:00

279 13:39:35.63 DSNUBAFI - DB2 IMAGE COPY SUCCESSFUL FOR TABLESPACE DB2R7.DB2R7
13:39:35.64 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
RESTORE SYSTEM – object level

FIC  Updates  BACKUP SYSTEM  Updates  Incr. FC Backup  Recover Object

13:40:36.61 DSNUGUTC - BACKUP SYSTEM
13:40:37.17 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA STARTING,
COPYPOOL = DSN$DB9C$DB
TOKEN = X'C4F9C3F1C4E58049024A8A0BC4E3145B2232'.
13:40:37.41 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA COMPLETED SUCCESSFULLY,
COPYPOOL = DSN$DB9C$DB
TOKEN = X'C4F9C3F1C4E58049024A8A0BC4E3145B2232'
ELAPSED TIME = 00:00:00.
RESTORE SYSTEM – object level

FIC Updates BACKUP SYSTEM Updates Incr. FC Backup Recover Object

13:41:37.70 DSNUGUTC - BACKUP SYSTEM ESTABLISH FCINCREMENTAL
13:41:38.26 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA STARTING,
COPYPOOL = DSN$DB9C$DB
TOKEN = X'C4F9C3F1C4E5808344FF8846C4E3145B2232'.
13:41:38.97 DSNUVBBD - THE SYSTEM LEVEL BACKUP TAKEN IS AN INCREMENTAL FLASHCOPY
13:41:38.97 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA COMPLETED SUCCESSFULLY,
COPYPOOL = DSN$DB9C$DB
TOKEN = X'C4F9C3F1C4E5808344FF8846C4E3145B2232'
ELAPSED TIME = 00:00:00.
DFSMShsm and RESTORE SYSTEM

RESTORE SYSTEM – object level

FIC Updates BACKUP SYSTEM Updates Incr. FC Backup Recover Object

DSNU0501 279 13:43:19.69 DSNUGUTC - RECOVER TABLESPACE DB2R7.DB2R7 DSNUM ALL
DSNU1527I 279 13:43:20.15 DSNCBMT - TABLESPACE DB2R7.DB2R7 WAS SUCCESSFULLY RESTORED FROM A FLASHCOPY TIME=00:00:00
DSNU578I -D9C1 279 13:43:20.18 DSNUCALA - SYSLGRNX INFORMATION FOR MEMBER D9C1
DSNU513I -D9C1 279 13:43:20.18 DSNUCALA - RECOVER UTILITY LOG APPLY RANGE IS RBA 000134DA9882 LRSN RBA 000134DAC464 LRSN C4E56F761AB7
DSNU513I -D9C1 279 13:43:20.18 DSNUCALA - RECOVER UTILITY LOG APPLY RANGE IS RBA 000134DB1F08 LRSN RBA 000134DBFO00 LRSN C4E570E7C54B
DSNU513I -D9C1 279 13:43:20.18 DSNUCALA - RECOVER UTILITY LOG APPLY RANGE IS RBA 000134DBF658
New VCAT translation for RESTORE SYSTEM
—Some customers want to clone using BACKUP / RESTORE
  • Could not clone within same sysplex due to VCAT names, HLQ and ICF restrictions
  • DB2 11 solves these restrictions with a new option for SYSTEM RETORE
    o Think of it as a ‘OBID/DBID’ translation for RESTORE SYSTEM
Other backup and recovery items

- Faster index recovery with FLA support for index log records
- Lifted many restrictions on point-in-time recovery prior to materializing REORG
  - PIT recovery restrictions lifted for
  - LOB table spaces
  - XML table spaces
  - PBR table spaces
  - Including when immediate alters have occurred since materializing REORG
- PIT recovery restrictions still in place
  - Table space conversion
  - PBG table spaces
  - PBG partition pruning
  - Online DROP COLUMN
Recent RECOVER enhancements

Fast Log Apply (FLA) now implemented for RECOVER INDEX
- Previously DB2 would wait until a log record was to be applied before reading the associated index page into the local bufferpool where it would then be cached
- Now DB2 will use list prefetch to read all the index pages that are needed to apply log records for, before applying any log record
- Potential for significant savings in elapsed time
- Should now reconsider decision: run RECOVER INDEX in parallel with RECOVER TABLESPACE [PART] vs. wait for RECOVER TABLESPACE [PARTs] to complete and then run REBUILD INDEX
- Enhancement taken back to V9 and V10 via APAR PI07694

Optimization to point-in-time RECOVER list of objects
- Recover objects only when necessary when performing PIT recovery when TOLOGPOINT or TORBA are specified
- It does not apply to log only recoveries, RECOVER BACKOUT, and recovers to current
- DIAGNOSE TYPE(607) is required to activate this behavior
Questions?
Thank You