Copy Smarter
Unload/Load, DSN1COPY and beyond
Overview
What are we trying to achieve?

• Create repeatable and reliable copy processes for DB2 data
• Handle deviations in table structures
• Create a scheduler friendly process
• Bonus points for speed

• Spoiler alert: It’s hard.
Building blocks of a copy process

- Copying the data is a small part. The full process looks like this:

<table>
<thead>
<tr>
<th>Structures</th>
<th>Data</th>
<th>Cleanup</th>
</tr>
</thead>
<tbody>
<tr>
<td>• DDL for tablespaces, tables, indexes</td>
<td>• Invoke copy programs to bring data from A to B</td>
<td>• Copy catalog statistics and RTS</td>
</tr>
<tr>
<td>• Also views, triggers, constraints, etc.</td>
<td></td>
<td>• Rebuild indexes if required</td>
</tr>
<tr>
<td>• Optionally rename objects</td>
<td></td>
<td>• Adjust versioning information, row format, RBA format</td>
</tr>
<tr>
<td>• Allocate target objects with sufficient space</td>
<td></td>
<td>• Take care of identity columns, sequences</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Rebind</td>
</tr>
</tbody>
</table>
A quick look at DDL

• Needs to be handled, regardless of data copy mechanism
• Db2 for z/OS does not come with a full-fledged DDL generator, but has ADMIN_INFO_SQL (used by Data Studio)
• Db2 for LUW has db2look, which can work with Db2 for z/OS, but its output is always LUW syntax
• Db2 Admin Tool has ADB2GEN
• Home-grown solutions: REXX and ISPF file tailoring
• Renaming objects is harder than it sounds due to views, triggers
• Many vendor solutions available
Requirement: Reliability

• Process should not break when objects are created, changed, or dropped
• Detect new page sets that were added
• Detect and reset restricted states
• Restart after failure
Copy Smarter - Unload/Load, DSN1COPY and beyond

Requirement: Being scheduler-friendly

• Fixed set of jobs
• Number of jobs does not change
• Contents of jobs do not change
• Can be executed repeatedly
Requirement: Speed

• Programs that copy Db2 data:
  • Unload/Load
  • DSN1COPY
  • ADRDSSU / FlashCopy2
  • Vendor solutions
What are our options to copy data?

- Every Db2 shop has Unload/Load (either from IBM or vendor)
- Every Db2 shop has DSN1COPY
- ADRDSSU always available, can trigger FlashCopy2

<table>
<thead>
<tr>
<th></th>
<th>Ease of use</th>
<th>Automation</th>
<th>Flexibility</th>
<th>Speed</th>
<th>Aware of Db2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload/Load</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>DSN1COPY</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>ADRDSSU / FlashCopy2</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
Copying the data with Unload/Load

• Easy to use, is often the go-to solution
• Db2 manages space for you
• Use LISTDEF and TEMPLATE to process many objects at once
• Changing SYSPUNCH may be tedious
  • Change table names
  • Change RESUME YES to RESUME NO REPLACE
  • Add OVERRIDE (SYSTEMPERIOD, IDENTITY, TRANSID, NONDETERMINISTIC), add ENFORCE NO
• Slow
Copy Smarter - Unload/Load, DSN1COPY and beyond

Copying the data with Unload/Load

• DOs and DON’Ts:
  • Use SPANNED YES for LOB and XML data
  • Use IDXDEFER ALL with partition level LOAD, then rebuild indexes
  • Identify and skip empty partitions (this can save you hours)
  • Use the cross loader if possible
  • Do not use FORMAT INTERNAL – unreliable
  • Do not use partlevel LOAD if number of partitions or limit keys differ
Identity columns and sequences

- Must be adjusted in target
- Use MAXASSIGNEDVAL + INCREMENT as new value
- Sequence objects: Use ALTER SEQUENCE RESTART WITH
- Identity columns: Use ALTER TABLE ALTER COLUMN RESTART WITH
- Implicit XML sequences: Query repeatedly to increase value
  - Cannot be altered directly
  - SQLCODE = -20142, ERROR: SEQUENCE CANNOT BE USED AS SPECIFIED
Why do Unload/Load based copies fail?

• During Unload/Load:
  • Missing authorization
  • Missing or incomplete target objects
  • Incompatible target object (e.g., insufficient column length, wrong code page)
  • Insufficient work data sets for sort
  • Objects in use by other utility
Why do Unload/Load based copies fail?

• Post Unload/Load:
  • Incorrect sequences and identity columns
  • Inaccessible due to restricted states
  • Inconsistencies if data was unloaded with SHRLEVEL CHANGE
Transfer Unload files from/to non-z/OS

• Either use FORMAT DELIMITED, then FTP as text
  • Does not work well for binary data, LOB, XML
• Or use standard LOAD format, then FTP as binary
  • Either use: QUOTE SITE RDW
    Each record is prefixed by a 4 byte field, first 2 bytes = length
  • Or use: QUOTE STRU R
    X’FF01’ = end of record, X’FF02’ = end of file, X’FF’ becomes X’FFFF’
• Properly transfer VBS data sets with binary data
  • Use: QUOTE MODE B, then QUOTE TYPE E
  • Use: SITE LRECL=X RECFM=VBS BLOCKSIZE=27998
  • On a PC, you will need a separate program to split/merge record fragments
Transfer Unload files from/to non-z/OS

• Binary transfer with file structure: Record ends are lost

First record ends right here, but there is nothing that would indicate that
Transfer Unload files from/to non-z/OS

- Binary transfer with record structure
Copy Smarter - Unload/Load, DSN1COPY and beyond

What are our options to copy data?

- Every Db2 shop has Unload/Load (either from IBM or vendor)
- Every Db2 shop has DSN1COPY
- ADRDSSU always available, can trigger FlashCopy2

<table>
<thead>
<tr>
<th></th>
<th>Ease of use</th>
<th>Automation</th>
<th>Flexibility</th>
<th>Speed</th>
<th>Aware of Db2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload/Load</td>
<td>Good</td>
<td>Fair</td>
<td>Good</td>
<td>Bad</td>
<td>Yes</td>
</tr>
<tr>
<td>DSN1COPY</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>ADRDSSU / FlashCopy2</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

© UBS Hainer GmbH, the IBM Business Partner
Copy the data with DSN1COPY

- Much faster than Unload/Load
- Works outside of Db2
- Need to allocate target VSAM clusters
- Need to check object compatibility very thoroughly
- Need to write / generate a lot of JCL with correct SYSXLAT members
Find all LOB tablespaces for a given table

SELECT
    STRIP(R.TBOWNER)       AS "BASE_TBCREATOR",
    STRIP(R.TBNAME)        AS "BASE_TBNAME",
    STRIP(R.COLNAME)       AS "BASE_COLNAME",
    R.PARTITION            AS "PARTITION",
    STRIP(S.DBNAME)        AS "LOB_DBNAME",
    STRIP(S.NAME)          AS "LOB_TSNAME",
    S.PGSIZE               AS "LOB_PGSIZE",
    S.DSSIZE               AS "LOB_DSSIZE",
    STRIP(R.AUXTBOWNER)    AS "AUX_TBCREATOR",
    STRIP(R.AUXTBNAME)     AS "AUX_TBNAME",
    STRIP(X.CREATOR)       AS "AUX_IXCREATOR",
    STRIP(X.NAME)          AS "AUX_IXNAME",
    X.PGSIZE               AS "AUX_IXPGSIZE",
    X.PIECESIZE            AS "AUX_IXPIECESIZE"
FROM
    SYSIBM.SYSAUXRELS R
    INNER JOIN
    SYSIBM.SYSTABLES T
    ON
    T.CREATOR = R.AUXTBOWNER AND
    T.NAME = R.AUXTBNAME
    INNER JOIN
    SYSIBM.SYSTABLESPACE S
    ON
    S.DBNAME = T.DBNAME AND
    S.NAME = T.TSNAME
    INNER JOIN
    SYSIBM.SYSTABLEPART P
    ON
    S.DBNAME = P.DBNAME AND
    S.NAME = P.TSNAME AND
    P.PARTITION IN (0, 1)
    INNER JOIN
    SYSIBM.SYSINDEXES X
    ON
    X.TBCREATOR = R.AUXTBOWNER AND
    X.TBNAME = R.AUXTBNAME
    INNER JOIN
    SYSIBM.SYSINDEXPART XP
    ON
    XP.IXCREATOR = X.CREATOR AND
    XP.IXNAME = X.NAME AND
    XP.PARTITION IN (0, 1)
WHERE
    R.TBOWNER = ? AND
    R.TBNAME = ?
FOR READ ONLY WITH UR
Find eligible image copy

- IQDSNUM = 0: Tablespace level
- IQDSNUM > 0: Partition level
- Considers FlashCopy consistent image copies

```sql
SELECT
    DSNUM     AS IQDSNUM,
FROM
    SYSIBM.SYSCOPY
WHERE
    DBNAME = ? AND
    TSNAME = ? AND
    DSNUM IN (0, 1) AND
    ICTYPE = 'F' AND
    (STYPE IN (' ', 'R', 'S', 'W', 'X') OR
    (STYPE = 'T' AND DSNUM <> 0) OR
    (STYPE = 'T' AND DSNUM = 0
    AND LOWDSNUM = 1 AND HIGHDSNUM = 1))
AND
    SHRLEVEL IN ('R', 'C')
ORDER BY
    TIMESTAMP DESC
FOR READ ONLY WITH UR
```
Copy Smarter - Unload/Load, DSN1COPY and beyond

Find eligible image copy

SELECT
  STRIP(C1.DSNAME) AS ICDSN,
  C1.TIMESTAMP    AS ICTS,
  HEX(C1.START_RBA) AS ICRBA,
  C1.DEVTYP       AS ICDEV,
  C1.DSVOLSER     AS ICVOL,
  C1.ICUNIT       AS ICUNIT,
  C1.FILESEQNO    AS ICSEQNO,
  C1.NPAGESF      AS ICNPAGES,
  C1.ICBACKUP     AS ICBACKUP,
  C1.ICTYPE       AS ICTYPE,
  C1.STYPE        AS ICSTYP
FROM
  SYSIBM.SYSCOPY C1
WHERE
  C1.DBNAME = ? AND
  C1.TSNAME = ? AND
  C1.ICTYPE  = 'F' AND
  C1.DSNUM  = ? AND
  C1.STYPE IN (' ', 'R', 'S', 'T', 'W', 'X') AND
  C1.SHRLEVEL IN ('R', ?)
AND (   
  C1.DSNUM = 0
)
OR EXISTS (   
  SELECT * FROM SYSIBM.SYSCOPY C2
  WHERE C2.START_RBA = C1.START_RBA AND C2.DSNUM = 1
)

AND (   
  C1.DSNUM = 0
)
OR EXISTS (   
  SELECT * FROM SYSIBM.SYSCOPY C3
  WHERE C3.START_RBA = C1.START_RBA AND C3.DSNUM = ?
)
)
ORDER BY
  C1.TIMESTAMP DESC
FOR READ ONLY

• Makes sure that partition level copies are only picked up if a copy was made for all partitions
Allocate target objects

- PBG tablespaces often problematic since Db2 can add partitions
  - Might create additional LOBs (and indexes), XML tablespaces (and indexes)
- Target PBG has fewer parts: Use ALTER TABLE ADD PARTITION
- Target PBG has too many parts:
  - Either: Drop and recreate
  - Or in V12: REORG with DROP_PART YES
  - Or in V11: REORG_DROP_PBG_PARTS = ENABLE
  - Neat trick: Empty all target partitions using LOAD REPLACE with empty SYSREC before the copy, then ignore extra partitions
Allocate target objects

• Non-partitioned tablespaces: Up to 32 VSAMs, use IDCAMS

• # of pieces based on TYPE, DSSIZE, PGSIZE, NUMPAGESF:

```c
/* Get piece size (non-LOBS) or DSSIZE (LOBs) */
IF TYPE = "O" THEN DSSIZE_IN_KB = DSSIZE * 1024 * 1024
ELSE DSSIZE_IN_KB = 2 * 1024 * 1024
/* Correction for LOBs with DSSIZE 4 G */
IF DSSIZE_IN_KB = 4096 * 1024 THEN DSSIZE_IN_KB = 4095 * 1024
/* Calculate number of pieces */
SIZE_IN_KB = (NPAGESF * PGSIZE)
NUMPIECES = SIZE_IN_KB % DSSIZE_IN_KB
REMAINDER_IN_KB = SIZE_IN_KB // DSSIZE_IN_KB
IF REMAINDER_IN_KB > 0 THEN NUMPIECES = NUMPIECES + 1
```
Allocate target objects

• Popular choice PRIQTY -1 SECQTY -1 causes problems with copy programs that work outside of Db2
• Inspect actual HI-U-RBA or use SYSIBM.SYSCOPY.NPAGESF
  • Non-partitioned TS or partition level copy of partitioned TS:
    \[ \text{NPAGESF} \times \text{PGSIZE} \]
  • TS-level copy of range-partitioned TS (average size per partition):
    \[
    \frac{(\text{NPAGESF} \times \text{PGSIZE}) + \text{NUMPARTS} - 1}{\text{NUMPARTS}}
    \]
  • Non-partitioned TS with \( n \) pieces or TS-level copy of PBG with \( n \) partitions:
    Piece 1 to \((n-1)\): \( \text{DSSIZE} \)
    Piece \( n \):
    \[ \text{MOD} (\text{NPAGESF} \times \text{PGSIZE}, \text{DSSIZE}) \]
Allocate target objects

• LOBs with DSSIZE 4 G only use 4095 MB
  • When non-EA: Final HI-A-RBA must be between 4095 and 4096 MB
  • Use MEGABYTES(94, 200) in your IDCAMS statement (YMMV)

• Pitfall: Partitioned objects with DSSIZE 4 G use 4096 MB

• Best way: Total size as PRIQTY, let SMS handle the details
  • This minimizes the number of extents (good performance)

```
DEFINE CLUSTER(CISZ(32768) REUSE LINEAR SHR(3 3) -
  NAME(DSNC10.DSNDBC.BIGLOBDB.L1.I0001.A001)) -
  DATA(NAME(DSNC10.DSNDBD.BIGLOBDB.L1.I0001.A001) -
  KILOBYTES(4193280 419328))
```
Copy Smarter - Unload/Load, DSN1COPY and beyond

SMS settings that make your life easier

Avoids unnecessary candidate volume entries in catalog

Simplifies space calculation

© UBS Hainer GmbH, the IBM Business Partner
SMS settings that make your life easier

- **CDS Name**: ACTIVE
- **Data Class Name**: MULVXDC
- **Data Set Name Type**: EXTENDED
  - **If Extended**: REQUIRED
  - **Extended Addressability**: YES
  - **RMODE31**:...
- **Space Constraint Relief**: YES
  - **Reduce Space Up To (%)**: 0
  - **Guaranteed Space Reduction**: NO
  - **Dynamic Volume Count**: 10
- **Compaction**...
- **Spanned / Nonspanned**...

Increases chance of successful allocation
Avoids under-allocation
Allows SMS to add more volumes from the data set’s storage group automatically

© UBS Hainer GmbH, the IBM Business Partner
SMS settings that make your life easier

must be specified if Add’l Volume Amount is set
SMS settings that make your life easier

- Allows up to 7,257 extents per data set
SMS settings that make your life easier

Avoids multi stripe data sets (multi stripe data sets cannot use space constraint relief)
### SMS settings that make your life easier

Avoids over-allocation (YES means primary space is allocated on all volumes)

---

**Panel Utilities Scroll Help**

---

<table>
<thead>
<tr>
<th>Command</th>
<th>STORAGE CLASS DISPLAY</th>
<th>Page 2 of 2</th>
</tr>
</thead>
</table>

**CDS Name** | ACTIVE

**Storage Class Name** | DEFAULT

**Guaranteed Space** | NO

**Guaranteed Synchronous Write** | NO

**Multi-Tiered SGs**

**Parallel Access Volume Capability** | NOPREF

**Cache Set Name**

**CF Direct Write**

**CF Sequential Weight**

**Lock Set Name**

**Disconnect Sphere at CLOSE** | NO

---

© UBS Hainer GmbH, the IBM Business Partner
Copying the data with DSN1COPY

• You need a script to generate required jobs (could be several thousand job steps)

• Not very scheduler friendly
  • Can be invoked from REXX in one single job step via
    ADDRESS ATTCHMVS "DSN1COPY <parms>"
  • Requires dynamic allocation and error handling in REXX

• Read my rant at:
Adjust version numbers, RBA / row format

• Extremely important after DSN1COPY based copy
• Failure to do so can lead to INCORROUT, ABEND S04E, S04F
• Db2 V10: Use REPAIR VERSIONS
• Db2 V11, V12: Use REPAIR CATALOG
• Problem:
  • Adjusting version numbers requires system pages
  • No system pages if tables have never been altered
  • This is changing (PI86880, UI51746)
Copy Smarter - Unload/Load, DSN1COPY and beyond

Copying catalog statistics and RTS

• Catalog statistics are important for the Db2 optimizer
  • Dynamic SQL: Copy statistics
  • Static SQL: Copy statistics, rebind plan

• Rebind after updating catalog statistics

• Do not forget to rebind implicit trigger packages
  • Basic triggers: REBIND TRIGGER PACKAGE (creator.name)
  • Advanced triggers: Basic triggers: REBIND PACKAGE (creator.name.(*))

• RTS are important when UTSORTAL = YES
Rebuild indexes

• Use dynamic allocation of sort work data sets
  • Specify SORTDEVT, do not specify SORTNUM (or set IGNSORTN=YES)
  • Remove DFSORT related DDs from utility jobs

• Make sure to copy RTS for index first

• If RTS for index is unavailable:
  • Make sure you have good RTS for associated tablespace
  • REPAIR OBJECT SET INDEXSPACE (dbname.spacenam) RBDPEND
  • Then rebuild index
Identity columns and sequences

- Must be adjusted in target
- Use MAXASSIGNEDVAL + INCREMENT as new value
- Sequence objects: Use ALTER SEQUENCE RESTART WITH
- Identity columns: Use ALTER TABLE ALTER COLUMN RESTART WITH
- Implicit XML sequences: Query repeatedly to increase value
  - Cannot be altered directly
  - SQLCODE = -20142, ERROR: SEQUENCE CANNOT BE USED AS SPECIFIED
Copying the data with DSN1COPY

• DOs and DON’Ts:
  • Pre-allocate all target VSAMs with the correct size
  • Check for restricted states in the source
  • Don’t copy XML tables into another Db2 subsystem
  • Don’t copy from an object that has not been reorganized after the most recent ALTER TABLE or DROP TABLE
  • Don’t copy partitioned tablespaces if partitions have been rotated, or if partitions have been inserted at any position other than the end
    • You’d think that „Relative Page Numbering“ helps, but it does not
Why do DSN1COPY based copies fail?

• During file system level copy:
  • Missing target page set
  • Cannot extend target page set or grow beyond 4 GB if non-EA
  • Remains of dropped tables in source causes OBID translation errors
Why do DSN1COPY based copies fail?

- Post file system level copy:
  - Did not include all source data sets
  - Incorrect sequences and identity columns
  - Did not take care of restricted states
  - Did not rebuild all target indexes
  - Incorrect OBID translation, log RBA, level ID
  - Did not do REORG before, REPAIR CATALOG after copy
  - Versioning problems / did not run REPAIR CATALOG in target
  - Copy was made despite structural incompatibilities
Fun with versions after APAR PI57004

• REPAIR CATALOG can have unexpected results:
  • Source tablespace and table are both version x, freshly reorged
  • Tables and tables are 100% compatible
  • You run DSN1COPY, REPAIR CATALOG
  • You check the target catalog and it says version x+1

• Reason: TIMESTAMP or DECIMAL columns in the source table
  • TIMESTAMP columns problematic if source was created in V10 or older
  • DECIMAL columns problematic if source was created in V7 or older
My biggest problem with DSN1COPY

• There are situations where the copy process itself succeeds, and the target objects *look* OK, and *seem* to be accessible, but on occasion accessing the target tables will produce abend S04E with reason 00C90101 or similar reason codes.

• It’s usually caused by missing REORGs / wrong SYSXLAT

• It can still be very hard to detect
What are our options to copy data?

- Every Db2 shop has Unload/Load (either from IBM or vendor)
- Every Db2 shop has DSN1COPY
- ADRDSSU always available, can trigger FlashCopy2

<table>
<thead>
<tr>
<th></th>
<th>Ease of use</th>
<th>Automation</th>
<th>Flexibility</th>
<th>Speed</th>
<th>Aware of Db2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload/Load</td>
<td>Good</td>
<td>Fair</td>
<td>Good</td>
<td>Bad</td>
<td>Yes</td>
</tr>
<tr>
<td>DSN1COPY</td>
<td>Bad</td>
<td>Bad</td>
<td>Bad</td>
<td>Good</td>
<td>Yes</td>
</tr>
<tr>
<td>ADRDSSU / FlashCopy2</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
What about ADRDSSSU / FlashCopy2?

- NOT a good tool to copy one tablespace to another
- Does not translate DBID, PSID, OBIDs
  - Some people think they can use the REPAIR utility to fix DBID and PSID, but this is not always true.
- Does not reset log RBAs
- Does not set the PG1COPY flag bit, which is used by REPAIR CATALOG to trigger schema checking
- Read my other rant at:

© UBS Hainer GmbH, the IBM Business Partner
After ADRDSSU, use REPAIR to fix Level ID, DBID, PSID, versions.

Table OBID is identical. We change the DBID from 012B to 012C, we change the PSID from 0004 to 0002.

<table>
<thead>
<tr>
<th>Message</th>
<th>Time</th>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNU050I</td>
<td>219 22:49:34.20</td>
<td>DSNUGUTC</td>
<td>REPAIR</td>
</tr>
<tr>
<td>DSNU650I</td>
<td>-DBBG 219 22:49:34.21</td>
<td>DSNUCBLI</td>
<td>LEVELID TABLESPACE TVERSIDX.T1</td>
</tr>
<tr>
<td>DSNU683I</td>
<td>-DBBG 219 22:49:34.57</td>
<td>DSNUCBRP</td>
<td>REPAIR LEVELID OPERATION SUCCESSFUL</td>
</tr>
<tr>
<td>DSNU050I</td>
<td>219 22:49:34.57</td>
<td>DSNUGUTC</td>
<td>REPAIR OBJECT</td>
</tr>
<tr>
<td>DSNU650I</td>
<td>-DBBG 219 22:49:34.58</td>
<td>DSNUCBRL</td>
<td>LOCATE TABLESPACE TVERSIDX.T1 PAGE X'00'</td>
</tr>
<tr>
<td>DSNU650I</td>
<td>-DBBG 219 22:49:34.86</td>
<td>DSNUCBRP</td>
<td>VERIFY OFFSET X'000C' DATA X'012B0004'</td>
</tr>
<tr>
<td>DSNU652I</td>
<td>-DBBG 219 22:49:34.86</td>
<td>DSNUCBRR</td>
<td>VERIFY OPERATION SUCCESSFUL</td>
</tr>
<tr>
<td>DSNU650I</td>
<td>-DBBG 219 22:49:34.86</td>
<td>DSNUCBRP</td>
<td>REPLACE OFFSET X'000C' DATA X'012C0002'</td>
</tr>
<tr>
<td>DSNU656I</td>
<td>-DBBG 219 22:49:34.91</td>
<td>DSNUCBRR</td>
<td>REPLACE OPERATION SUCCESSFUL, DATA WAS X'012B0004'</td>
</tr>
<tr>
<td>DSNU050I</td>
<td>219 22:49:34.94</td>
<td>DSNUGUTC</td>
<td>REPAIR</td>
</tr>
<tr>
<td>DSNU650I</td>
<td>-DBBG 219 22:49:34.95</td>
<td>DSNUCBVR</td>
<td>CATALOG TABLESPACE TVERSIDX.T1</td>
</tr>
<tr>
<td>DSNU675I</td>
<td>-DBBG 219 22:49:34.27</td>
<td>DSNUCBVR</td>
<td>HIGH VERSION FOR DBID=X'012C' PSID=X'0002' IN THE Db2 CATALOG IS 0, BUT IN THE PAGE SET IS 1.</td>
</tr>
<tr>
<td>DSNU675I</td>
<td>-DBBG 219 22:49:34.27</td>
<td>DSNUCBVR</td>
<td>LOW VERSION FOR DBID=X'012C' PSID=X'0002' IN THE Db2 CATALOG IS 0, BUT IN THE PAGE SET IS 1.</td>
</tr>
<tr>
<td>DSNU671I</td>
<td>-DBBG 219 22:49:34.27</td>
<td>DSNUCBVR</td>
<td>DBID=X'012C' PSID=X'0002' OBID=X'07D0' TABLE VERSION IN THE CATALOG DOES NOT MATCH THE PAGE SET</td>
</tr>
<tr>
<td>DSNU695I</td>
<td>-DBBG 219 22:49:34.29</td>
<td>DSNUCBVR</td>
<td>INFORMATION IN THE CATALOG WAS UPDATED TO MATCH THE PAGE SET</td>
</tr>
</tbody>
</table>
Then run REBUILD INDEX. It works, so the tablespace is OK, right?
Verifying the data is OK using `SELECT *`, all is good.

```
**INPUT STATEMENT:**
SELECT * FROM KAI.SVERSIDX_TB1;
```

```
+-----------------------------------------------------------------------------------------------------+
|     COL01      |     COL02      |     COL03      |     COL04      |     COL05      |     COL06      |
+-----------------------------------------------------------------------------------------------------+
|  1_ |              1 |         114921 |         624590 |         550815 |         939021 | ?              |
|  2_ |              2 |         724429 |         839504 |         684609 |         635107 |         181827 |
|  3_ |              3 |         552245 |         750931 |         245715 |         751530 |         724362 |
|  4_ |              4 |         612184 |         519980 |         182482 |         817834 |            464 |
|  5_ |              5 |         988725 |         654720 |         336550 |         924940 |         426401 |
|  6_ |              6 |         341380 | ?              |         118057 | ?              |          35152 |
|  7_ |              7 |         203656 |         182620 |         505645 |         698866 |         45088 |
|  8_ |              8 |         88569  |         214801 |         809647 |         318800 |         931594 |
|  9_ |              9 |         308779 |         77896  |         786255 |         919866 |         639709 |
| 10_ |             10 |         176410 |         847833 |           9513 |         749993 |         675077 |
| 11_ |             11 |         515483 |         139905 |         555669 |         869535 |         869535 |
| 99997_|          99997 |         857691 |         335066 |         729772 |          35849 |          22198 |
| 99998_|          99998 |          46105 |         953504 |         617066 |         275336 |         766092 |
| 99999_|          99999 |         361769 |         913569 | ?              |         819001 |         892942 |
| 100000_ |         100000 |          35631 |         989423 |         536785 |         986317 |         251987 |
+-----------------------------------------------------------------------------------------------------+
SUCCESSFUL RETRIEVAL OF 100000 ROW(S)
```
Wrap things up by running RUNSTATS. Wait a second…

```
LISTDEF L1 INCLUDE TABLESPACES DATABASE TVERSIDX BASE
LISTDEF STATEMENT PROCESSED SUCCESSFULLY
LISTDEF L2 INCLUDE INDEXSPACES DATABASE TVERSIDX
LISTDEF STATEMENT PROCESSED SUCCESSFULLY
RUNSTATS TABLESPACE LIST L1 SHRLEVEL REFERENCE REPORT NO UPDATE ALL HISTORY ALL
TABLE(ALL)
PROCESSING LIST ITEM: TABLESPACE TVERSIDX.T1
UTILITY DATA BASE SERVICES MEMORY EXECUTION ABENDED, REASON=X'00C9021C'
```

**00C9021C**

While running a utility, the data manager detected an inconsistent data condition. A row was encountered that is not represented by a record OBD in the database descriptor (DBD). This abend may indicate an internal Db2® error, but most likely occurs due to a user error. Possible user errors may include:

- Data from a Db2 subsystem was copied to another Db2 subsystem incorrectly. This is the most common error.
- DSNDB01.DBD01 was regressed to a time prior to a table being created.
What happened?

DBID X'012B' PSID X'0004'
Table OBID X'07D0'
What happened?

The PSID appears on the space map page to identify the segment that contains the compression dictionary.
Anything else?

- Space map pages and system pages contain PSID and OBIDs.
- Number and locations of these pages varies (and so does the offset of the PSID/OBID fields on these pages).
- In the future, there will be system pages even if objects have never been altered.
- Remember PI86880, UI51746?
What about ADRDSSU / FlashCopy2?

• DOs and DON’Ts:
  • Don’t use ADRDSSU / FlashCopy2 to make a copy from one Db2 tablespace to another
  • No really, don’t.
Copy Smarter - Unload/Load, DSN1COPY and beyond

What are our options to copy data?

- Every Db2 shop has Unload/Load (either from IBM or vendor)
- Every Db2 shop has DSN1COPY
- ADRDSSU always available, can trigger FlashCopy2

<table>
<thead>
<tr>
<th></th>
<th>Ease of use</th>
<th>Automation</th>
<th>Flexibility</th>
<th>Speed</th>
<th>Aware of Db2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload/Load</td>
<td>Good</td>
<td>Fair</td>
<td>Good</td>
<td>Bad</td>
<td>Yes</td>
</tr>
<tr>
<td>DSN1COPY</td>
<td>Bad</td>
<td>Bad</td>
<td>Bad</td>
<td>Good</td>
<td>Yes</td>
</tr>
<tr>
<td>ADRDSSU / FlashCopy2</td>
<td>Bad</td>
<td>Bad</td>
<td>Bad</td>
<td>Good</td>
<td>No</td>
</tr>
</tbody>
</table>

© UBS Hainer GmbH, the IBM Business Partner
Conclusion
Copy Smarter - Unload/Load, DSN1COPY and beyond

Conclusion

- Db2 itself does not provide a good mechanism to copy objects
- Problems mainly stem from:
  1. Missing tools for DDL generation
  2. Dependencies between Db2 catalog and contents of page sets
  3. Concept of version numbers after online schema changes
  4. Quirks of the native z/OS file system
- Unload/Load solves problems 2, 3, 4, but is too slow
- DSN1COPY lacks automation, is error prone
Conclusion

• Many Db2 shops simply use Unload/Load
• Some Db2 shops try to automate DSN1COPY
  • Works reasonably well for simple environments
  • Problems arise when newer Db2 features are exploited (table versioning, universal PBG tablespaces, partition rotation, clone tables, adding partitions in the middle of a tablespace, XML, etc.)
  • DSN1COPY may end with return code 0 even if the target is broken
Is there a better way?

• Vendor tools provide a degree of automation that is very hard to achieve manually
• UBS Hainer offers BCV5, which can do everything that was discussed today and more
• It combines unmatched flexibility with a very high copy speed
• BCV5 is easy to use, setting up a copy process takes mere minutes
• BCV5 is very scheduler friendly (fixed number of jobs, static JCL)
• BCV5 can also make consistent copies without stopping the source
Copy Smarter - Unload/Load, DSN1COPY and beyond

What are our options to copy data?

- Every Db2 shop has Unload/Load (either from IBM or vendor)
- Every Db2 shop has DSN1COPY
- ADRDSSU always available, can trigger FlashCopy2

<table>
<thead>
<tr>
<th></th>
<th>Ease of use</th>
<th>Automation</th>
<th>Flexibility</th>
<th>Speed</th>
<th>Aware of Db2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload/Load</td>
<td>Good</td>
<td>Fair</td>
<td>Good</td>
<td>Bad</td>
<td>Yes</td>
</tr>
<tr>
<td>DSN1COPY</td>
<td>Bad</td>
<td>Bad</td>
<td>Bad</td>
<td>Good</td>
<td>Yes</td>
</tr>
<tr>
<td>ADRDSSU / FlashCopy2</td>
<td>Bad</td>
<td>Bad</td>
<td>Bad</td>
<td>Good</td>
<td>No</td>
</tr>
<tr>
<td>Vendor solutions</td>
<td>We certainly think so.</td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
</tbody>
</table>
Questions or comments?
Thank you for your attention!

For more information visit www.ubs-hainer.com or send an email to info@ubs-hainersoftware.com