SWG BetaWorks

DB2 9

Technical Education Series

Reordered Row Format and Not Logged Table Spaces

Agenda

- **Reordered Row Format (RRF)**
- **Not Logged Table Spaces**

SWG BetaWorks

Reordered Row Format

DB2 9 for z/OS Technical Education Series

Introduction to RRF

- **Reordered Row Format (RRF): performance improvement for access to data in tables that contain columns of varying length.**
  - RRF optimizes column location for data retrieval and for predicate evaluation.
  - No change to tables only containing fixed-length columns.
  - No new SQL syntax.
  - NFM only.
    - Tolerated in CM* and ENFM*.
    - Table spaces created in NFM automatically use RRF.
  - No conversion for the DB2 catalog and directory.
  - Table spaces that contain tables with varying length columns with EDITPROCs or VALIDPROCs are not converted to RRF.

Basic Row Format and Reordered Row Format

```
CREATE TABLE TB1 (C1 INTEGER NOT NULL, C2 VARCHAR(10) NOT
NULL, C3 CHAR(10) NOT NULL, C4 VARCHAR(20)
```

Basic Row Format

C1	C2	C3	C4
8000000A	0006 WILSON	ANDREW	0008 SAN JOSE

2-byte length

Reordered Row Format

C1	C3	O2	O4	C2	C4
8000000A	ANDREW	12	18	WILSON	SAN JOSE

offset to C2

offset to C4

- **The offset is a hexadecimal value, indicating the offset for the column from the start of the data in the row.**
  - C2 offset: 4 bytes (for two x two-byte offset fields) + 4 bytes for an integer column + 10 bytes for the char field = 18 bytes = x’12’ bytes.
  - C4 offset: 18 bytes for C2 offset + 6 bytes for VARCHAR column C2 = 24 bytes = x’18’ bytes.
- **Note: in BRF, for a varying length field, DB2 logs from the first changed byte to the end of the row. In RRF, the first changed byte may be the offset field for the first varying length field following the one being updated.**

DSN1PRNT for RRF

```
Column Name          Col No Col Type Length Scale  Null
*                         * *             *      *  *
------------------> ------ -------- ------ ------ ----
EYE_CATCHER_1             1 CHAR          3      0 Y
EYE_CATCHER_2             2 VARCHAR       7      0 Y
PRIM_KEY                  3 INTEGER       4      0 Y
EYE_CATCHER_3             4 VARCHAR      11      0 Y
XMLCOL                    5 XML           6      0 Y
DB2_GENERATED_DOCI        6 BIGINT        8      0 Y
```

```
***  BEGINNING OF PAGE NUMBER 00000006 ***
 0000 00000000 00000000 00000600 0A970533   00001A00 01002E00 000100C1 C1C10080
 0020 000001FF 00000000 00000000 0018001A   002200C2 00E5C1D9 D3C5D5E7 80000000
 0040 00000100 30000001 00C2C2C2 00800000   02FF0000 00000000 00000018 001B0024
 0060 00C2C200 E5C1D9D3 C5D5E7E7 80000000   00000100 32000001 00C3C3C3 00800000
```

🟧 Data for column ‘XMLCOL’ (type: XML)

🟪 Offset fields

🟩 Row header

🟦 Page header

🟨 Null indicators

Impact of RRF for ALTER

- **For ALTER TABLE ADD PART of ALTER TABLE ROTATE PART the row format of the new or rotated partition is determined according to the following rules:**
  - If in Compatibility Mode (CM*), having fallen back from New Function Mode (NFM), and there is no EDITPROC or VALIDPROC, the new or rotated partition will be in BRF.
  - If in NFM and there is no EDITPROC or VALIDPROC, the new or rotated partition will be in RRF.
  - If there is an EDITPROC, the new or rotated partition will match the row format of the table space.
  - If there is a VALIDPROC and all partitions are the same format, then the new or rotated partition will match the row format of the table space.
  - If in CM* and there is a VALIDPROC and the row format is mixed, then the new or rotated partition will be in BRF.
  - If NFM and there is a VALIDPROC and the row format is mixed, then the new or rotated partition will be in RRF.
- **ALTER TABLE ADD COLUMN will place the table space in AREO* status.**

Utility Impact of RRF 1

- **LOAD REPLACE & REORG**
  - No syntax changes.
  - When run on table space or partition in BRF, then convert to RRF.
    - Exception: a table in the table space with an EDITPROC or VALIDPROC.
  - Recommendation: when running a utility on a BRF object that will be converted to RRF, do not specify KEEPDICTIONARY so that a new dictionary is built.
  - New TTYPE column of SYSCOPY.
- **DSN1COPY**
  - **WARNING**: when using DSN1COPY of a table space to populate another table space, make sure that the row formats match.

Utility Impact of RRF 2

- **RECOVER**
  - You **cannot** recover a **piece of a non-partitioned** RRF table space to a prior PIT where the table space is BRF. Recover the entire table space.
  - You cannot recover a piece of TS that is in BRF to a PIT when the TS was in RRF.
  - If a table space or partition in RRF is recovered to a point in time when it was in BRF, then the table space or partition will revert to BRF.
  - If TS/part in BRF is recovered to a PIT when it was in RRF, then it will revert to RRF.
  - RECOVER updates the TTYPE column of SYSCOPY to indicate row format.
- **REPORT RECOVERY**
  - The output from REPORT RECOVERY includes an indication of when the table space was converted from BRF to RRF (or from RRF to BRF).

RRF Catalog Changes

- **SYSTABLEPART**
  - FORMAT – new column (CHAR 1)
    - R = RRF
    - Blank = BRF
- **SYSCOPY**
  - TTYPE – new column (CHAR 8)
    - RRF = RRF (!)
    - BRF = BRF (!)
    - This column is also used for information about volume-based utilities and for serviceability enhancements (a broken page indicator).

Not Logged Table Spaces

Pre-DB2 9 Response to Concerns About Logging

- **Previous assumptions about customer requirements to suppress logging:**
  - Customers were concerned about the volume of log data
  - Logging performance is not a bottleneck
    - In the average environment, logging accounted for < 5% of CPU utilisation.

- **Initial response to requirement:**
  - Global Temporary Tables
  - LOB data optionally not logged

The Need for Not Logged Table Spaces

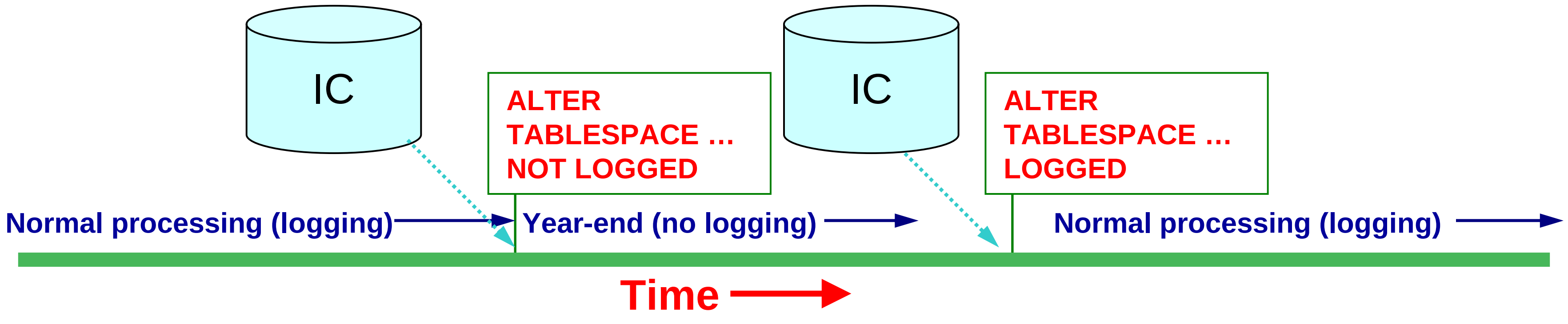
1. **Now increasingly apparent that logging *can* be a severe bottleneck (log latch contention) when:**
   – Large volumes of data are being inserted
   – Insert processes are running in parallel
   – Example: creation of summary tables in a Data Warehouse environment
   – Problems will increase in severity and frequency when more and faster CPUs are available to a single z/OS image running a single DB2
2. **The volume of archive logs**

■ Solution: extend existing logging volume reduction

Before You Implement Not Logged Table Spaces

- **Consider:**
  - Faster devices (DS8000) and MIDAW
  - DB2 9 logging improvements:
    - Data sharing – LRSN increment wait reduction
    - Archive logs BDAM => BSAM => EF data sets, striping and/or compression for archive logs
    - Dual buffering for archive log reads
    - Larger DASD archive logs (up to 4GB)
  - LOAD … NOT LOGGED
  - No logging for work files or global temporary tables
  - Limited logging for declared temporary tables
  - Data Sharing (each member has its own log)
  - Log striping (DB2 V7)
  - More archive, more active logs (DB2 V8)

NOT LOGGED Scenario 1


IC
ALTER TABLESPACE … NOT LOGGED
IC
ALTER TABLESPACE … LOGGED
Normal processing (logging)
Year-end (no logging)
Normal processing (logging)
Time

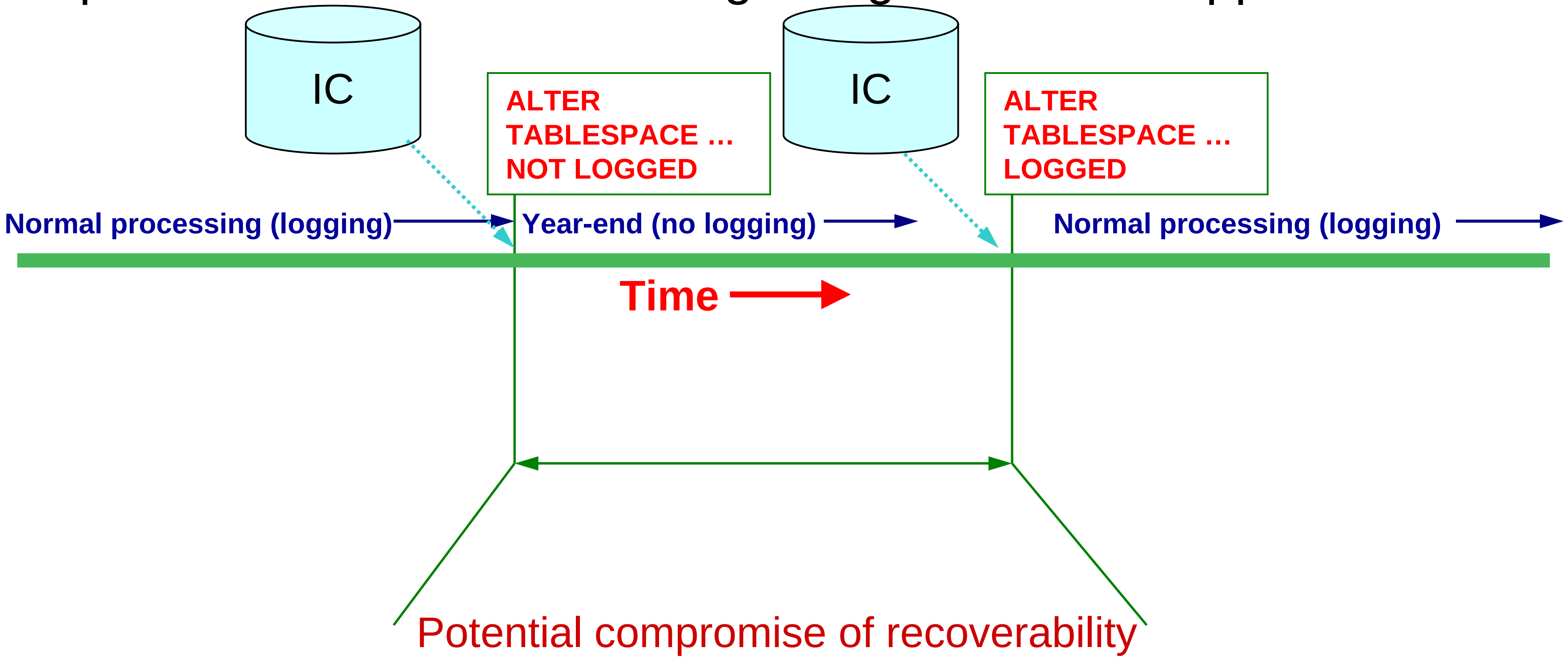
- **Most of the year, a table is modified infrequently and few rows are changed.**
- **During year end, extensive changes are made in a short period of time.**
- **At year end: take an image copy; turn off logging; make massive changes; turn on logging, take another image copy**
- **Avoid other changes to the data (lost if anything goes wrong!).**
- **If anything goes wrong during the NOT LOGGED period, recover the original data from the prior image copy, and rerun the bulk change processes (restore and rerun).**

NOT LOGGED Scenarios 2 & 3

- **Data loaded into a particular table must be modified to conform to requirements for that table**
  - It might be advantageous to modify the data after it is loaded into the table, using SQL because of its power and flexibility.
  - Before DB2 9, impossible to avoid the logging of the changes to the loaded data
- Scenario 2
- **With DB2 9 you can specify NOT LOGGED for both the loading operation and for the subsequent modifications to the data**
- Scenario 3
- **Materialized Query Tables are a summarization of data that exists elsewhere and are candidates for the NOT LOGGED attribute**
  - Since an MQT can be rebuilt from data that exists elsewhere, there is no need for changes to the MQT to be logged
- **This also applies to the kind of summary tables built prior to the availability of MQTs (DB2 V8).**

NOT LOGGED Caution

- WARNING: **Do not sacrifice recoverability in an attempt to gain a performance benefit**
  - In most cases you will not be able to notice a difference in performance once the writing of log records is suppressed

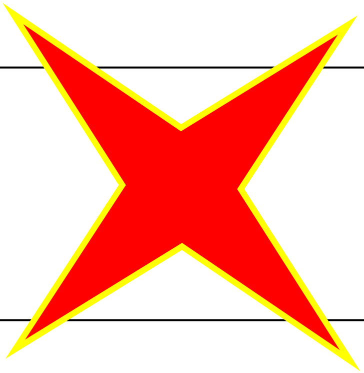

IC
ALTER TABLESPACE ... NOT LOGGED
IC
ALTER TABLESPACE ... LOGGED
Normal processing (logging)
Year-end (no logging)
Normal processing (logging)
Time
Potential compromise of recoverability

The Table Space Logging Attribute

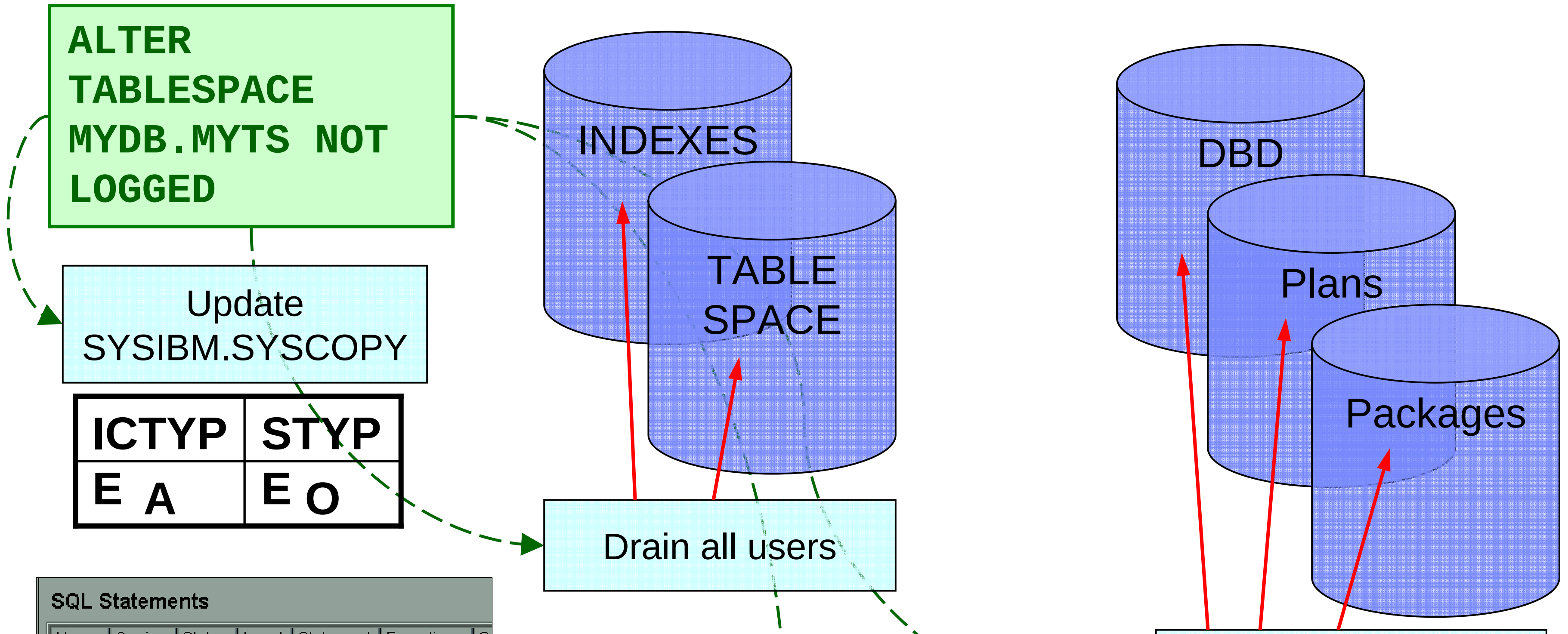
- The table space logging attribute – LOGGED or NOT LOGGED
  - Default setting: LOGGED
  - Synonyms LOG YES and LOG NO
  - New Function Mode Only
  - Applies to all partitions.
- NOT LOGGED
  - Only suppresses the writing of UNDO and REDO information - control records are still written.
  - Indexes (including auxiliary indexes) inherit the logging attribute from the associated base table space.

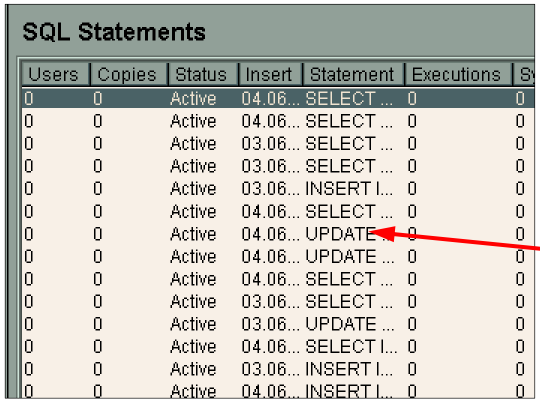
```
CREATE TABLESPACE ... NOT LOGGED


CREATE TABLE ... DATA CAPTURE CHANGES
```



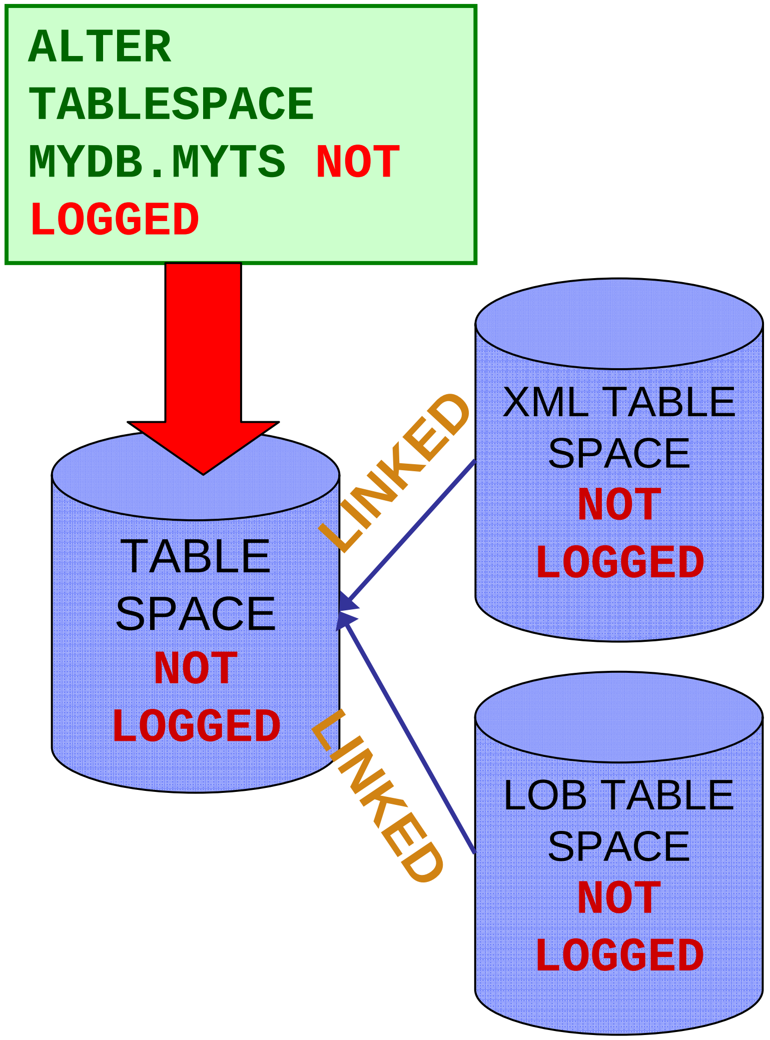
ALTER TABLESPACE Mechanics


ALTER TABLESPACE MYDB.MYTS NOT LOGGED
Update SYSIBM.SYSCOPY
ICTYP
STYP
E A
E O
INDEXES
TABLE SPACE
Drain all users
DBD
Plans
Packages
Exclusive lock on DBD and dependent plans and packages
Purge dependent statements from DSC


SQL Statements

Users	Copies	Status	Insert	Statement	Executions	Sy
0	0	Active	04.06...	SELECT ...	0	0
0	0	Active	04.06...	SELECT ...	0	0
0	0	Active	03.06...	SELECT ...	0	0
0	0	Active	03.06...	SELECT ...	0	0
0	0	Active	03.06...	INSERT I...	0	0
0	0	Active	04.06...	SELECT ...	0	0
0	0	Active	04.06...	UPDATE	0	0
0	0	Active	04.06...	UPDATE ...	0	0
0	0	Active	04.06...	SELECT ...	0	0
0	0	Active	03.06...	SELECT ...	0	0
0	0	Active	03.06...	UPDATE ...	0	0
0	0	Active	04.06...	SELECT I...	0	0
0	0	Active	03.06...	INSERT I...	0	0
0	0	Active	04.06...	INSERT I...	0	0

Auxiliary Tables Spaces and the Logging Attribute


ALTER TABLESPACE MYDB.MYTS NOT LOGGED
TABLE SPACE NOT LOGGED
LINKED
XML TABLE SPACE NOT LOGGED
LINKED
LOB TABLE SPACE NOT LOGGED

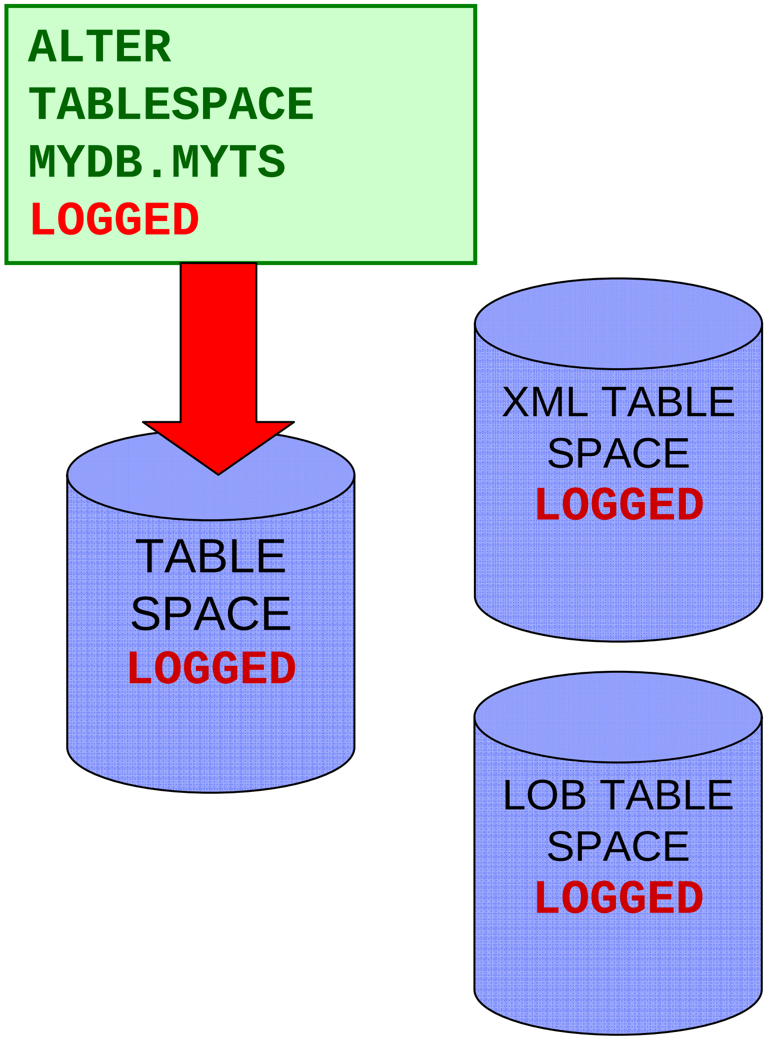
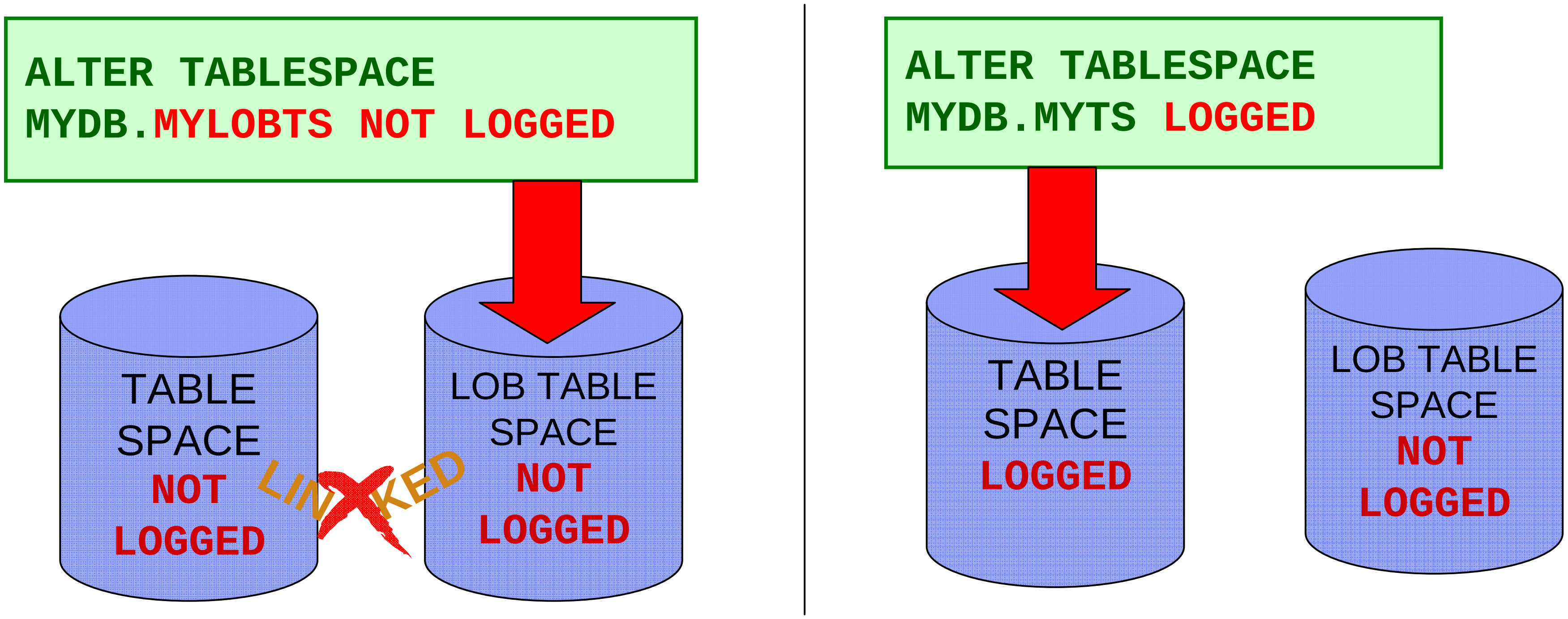


Table Space Linkage


ALTER TABLESPACE MYDB.MYLOBTS NOT LOGGED
TABLE SPACE NOT LOGGED
LINKED
LOB TABLE SPACE NOT LOGGED
ALTER TABLESPACE MYDB.MYTS LOGGED
TABLE SPACE LOGGED
LOB TABLE SPACE NOT LOGGED

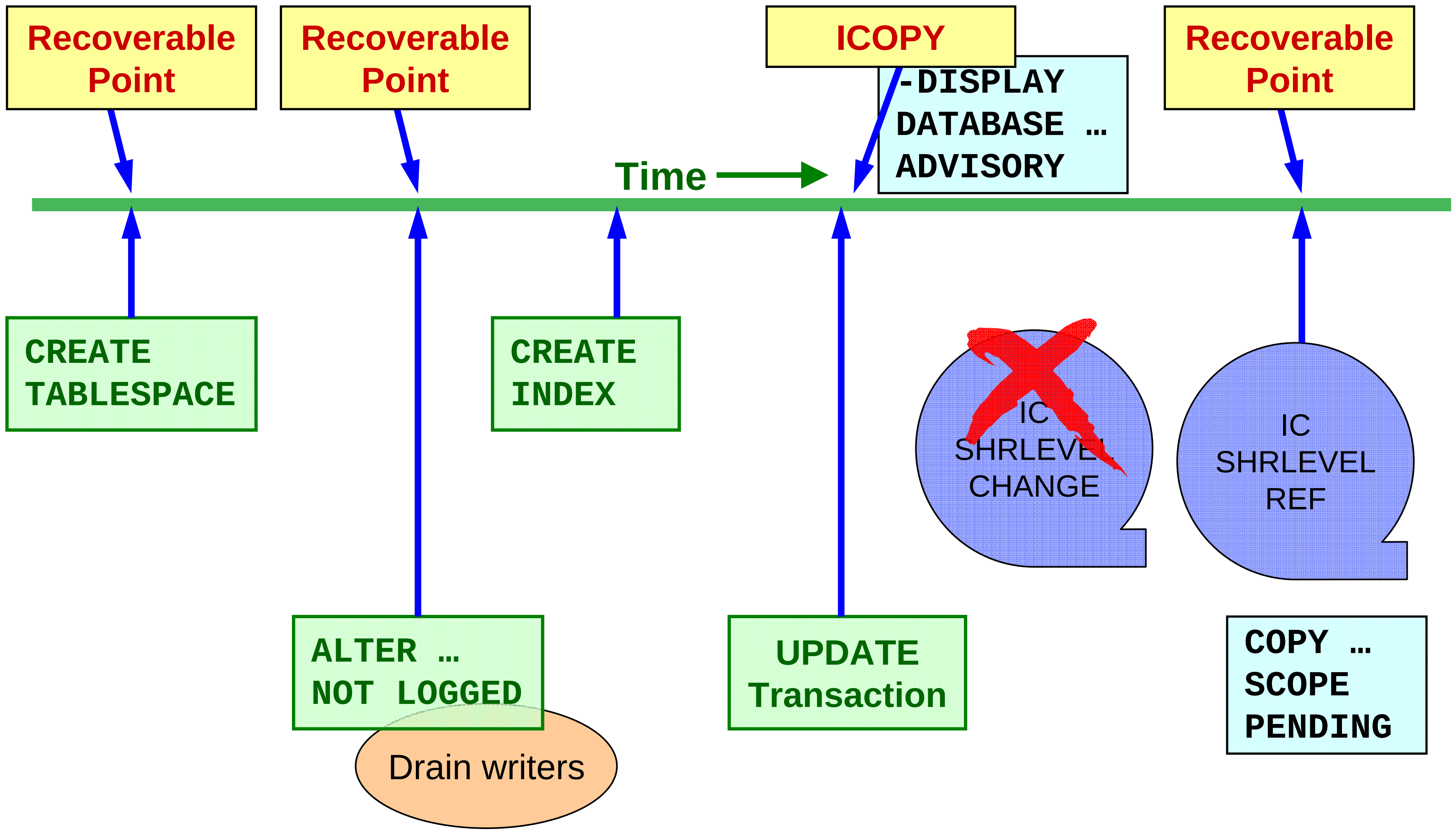
- **The LOG column of SYSIBM.SYSTABLESPACE is used for all table spaces.**
  - 'Y', logged
  - 'N', not logged
  - 'X', not logged, linked to base
- **Valid for a base table space: 'Y' or 'N'**
- **Valid for a LOB table space: 'Y', 'N', or 'X'**
- **Valid for an XML table space: 'Y' or 'X'**

SYSCOPY and SYSLGRNX

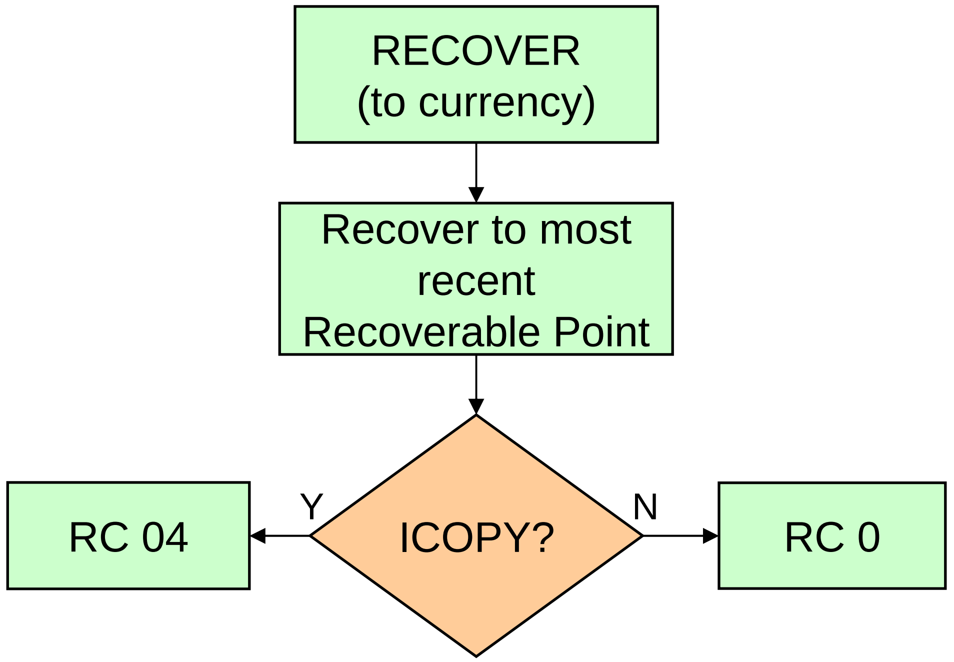
	DB NAME	TS NAME	ICTYPE	START_RBA	STYPE	LOGGED
LOAD … LOG NO	MYDB	TS01	S	xxxxxxxxxx		
CREATE … LOGGED	MYDB	TS02	C	xxxxxxxxxx	L	Y
CREATE … NOT LOGGED	MYDB	TS03	C	xxxxxxxxxx	O	N
ALTER … LOGGED	MYDB	TS04	A	xxxxxxxxxx	L	Y
ALTER … NOT LOGGED	MYDB	TS05	A	xxxxxxxxxx	O	N
REORG … LOG NO	MYDB	TS04	W	xxxxxxxxxx	A	Y
COPY	MYDB	TSO5	F	xxxxxxxxxx		N

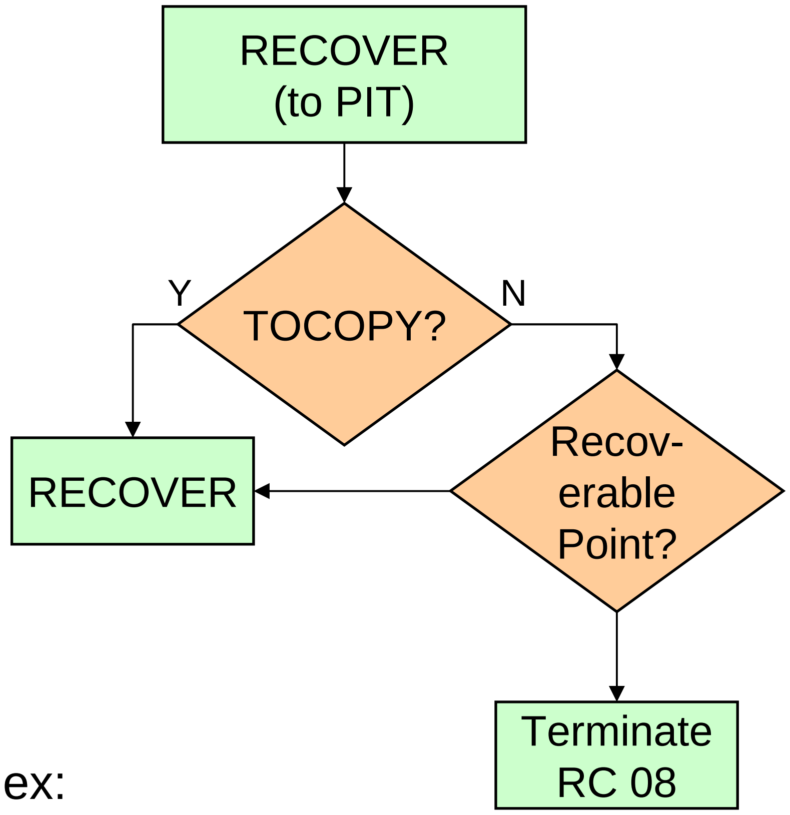
SYSLGRNX records are not maintained for NOT LOGGED table spaces

Recoverable Points



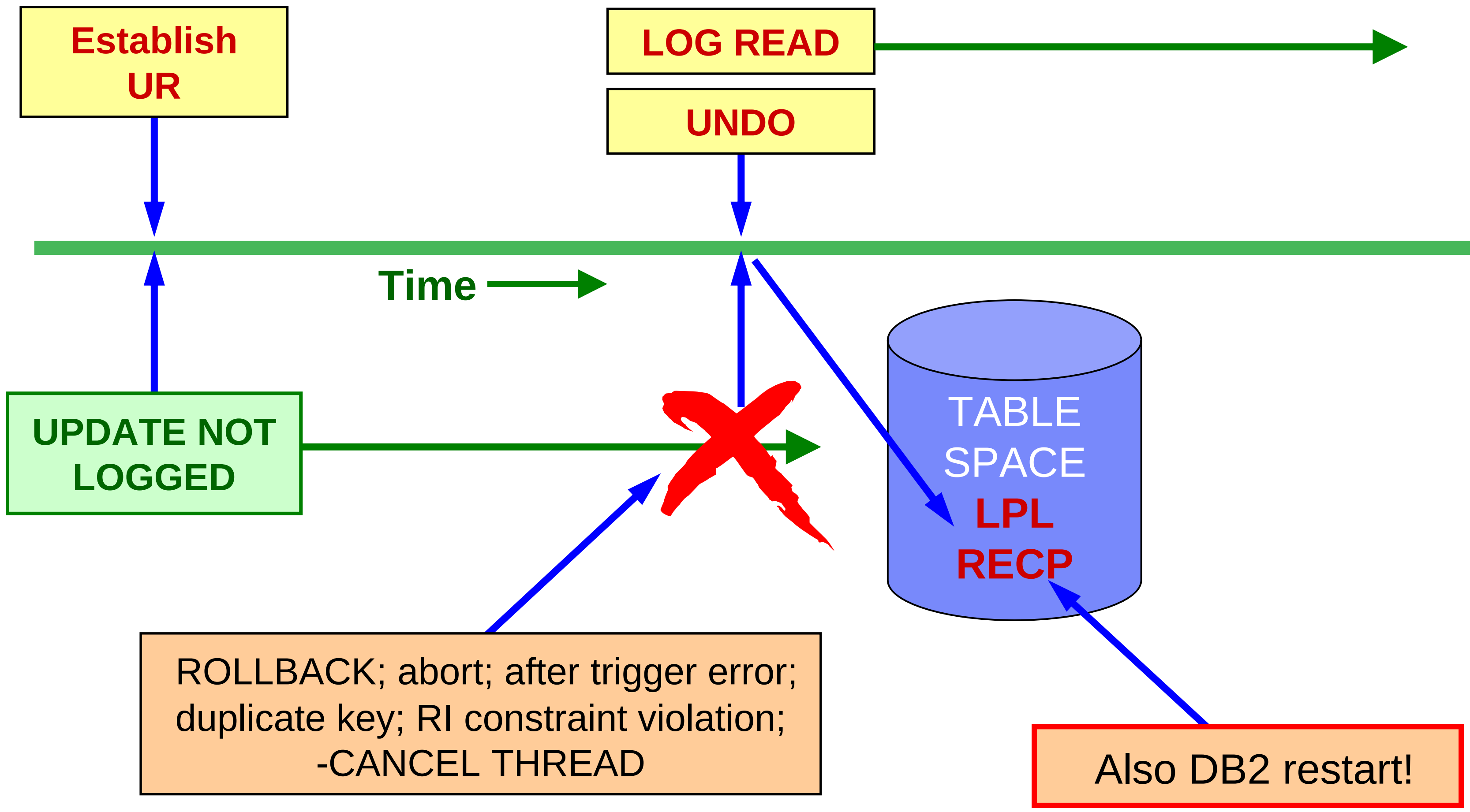
RECOVER and Not Logged Table Spaces





- **Watch out for:**
  - Restrictions for recovery of a COPY YES Index:
    - ICOPY status inhibits recover to currency.
    - Recoverable points must also be table space recoverable points.
  - RESTORE SYSTEM, if an open PSCR for a not logged table space: table space => RECP.
  - Image copy after ALTER … LOGGED needed for RECOVER to current.

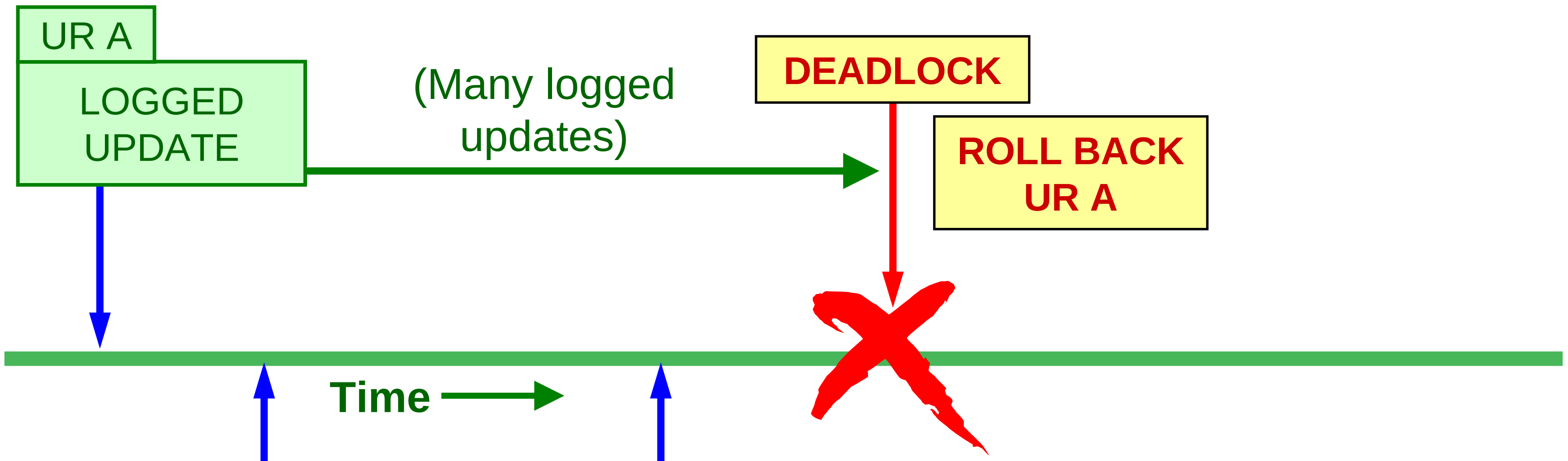
COMMIT and ROLL BACK Considerations


Establish UR
LOG READ
UNDO
Time
UPDATE NOT LOGGED
TABLE SPACE
LPL
RECP
ROLLBACK; abort; after trigger error; duplicate key; RI constraint violation; -CANCEL THREAD
Also DB2 restart!

LPL Considerations

- **Automatic LPL recovery is** not **initiated for not logged table spaces.**
- **-START DATABASE has no effect on the LPL status of the table space.**
- **Possible actions:**
  - REFRESH TABLE for an MQT
  - RECOVER (to currency or to PIT)
  - LOAD REPLACE
  - Drop, recreate, repopulate.
  - Unqualified DELETE or TRUNCATE TABLE.
- **All updated indexes are placed in LPL.**
  - COPY YES indexes => RECP
  - Other indexes => RBDP
- **XML table spaces => LPL and RECP**
- **LOB table spaces => AUXW and LOB marked invalid**

Deadlock and Timeout



NOT LOGGED UR timeout multiplier:
Maximum of 3 or the thread's current timeout multiplier

Online Load Resume LOG NO

"Now I can get that online LOAD RESUME to work without having to log!"

- **To achieve Online LOAD RESUME with LOG NO:**
  - ALTER … NOT LOGGED
  - Online LOAD RESUME
  - ALTER … LOGGED
  - COPY
- **Online LOAD against a NOT LOGGED table space is not restartable.**
- **An abend places the table space in the LPL/RECP status**
- **Recovery:**
  - TERMINATE the utility
  - RECOVER to a prior image copy
  - RERUN
  - Caution – what about concurrent updates?

REORG and QUIESCE

- **REORG SHRLEVEL(REFERENCE) or SHRLEVEL (NONE)** ✓
- **For SHRLEVEL NONE, LOG YES is changed to LOG NO** ✓
  - If no inline copy is requested the table space is marked ICOPY
- **For SHRLEVEL REFERENCE, the LOG parameter is not allowed and an inline copy is always produced**
- **REORG TABLESPACE PART SHRLEVEL (REFERENCE), with NPSIs** ✗
- **REORG SHRLEVEL (CHANGE)** ✗

- **QUIESCE does not create a recoverable point** ✗
- **If QUIESCE with WRITE(NO) is requested, the object is skipped** ✗
- **If QUIESCE with WRITE(YES) is requested, the table space and indexes are drained of writers, and their pages are written from the buffer pool to DASD** ✓

NOT LOGGED Operational Considerations

- **Because the data in a NOT LOGGED table space is not protected by the DB2 log, there is a need to externalize data changes quickly, without elongating commit processing or elapsed time.**
- **RO SWITCH CHKPTS (PCLOSEN) and RO SWITCH TIME (PCLOSET) are effectively set to 1 for NOT LOGGED table spaces**
- **In data sharing, NOT LOGGED table spaces may tend to change their GBP-dependency more often than logged table spaces**
- **DSN1LOGP SUMMARY(YES) is enhanced to show whether an object is logged or not logged, when possible.**
- **DSN1LOGP formatting of PSCRs is enhanced to indicate whether an object is logged or not.**

In the Reference Material

- **Converting tables with EDITPROCs or VALIDPROCs to RRF**
- **NOT LOGGED table space considerations:**
  - Detailed LOAD and REORG considerations
  - REBUILD INDEX and CHECK DATA considerations
  - Application programming considerations

Reference Material

Converting Table Spaces to RRF 1

- **Normally, a table space is converted to RRF by running the LOAD REPLACE or REORG utility in NFM.**

- **If a table space cannot be converted to RRF because it contains one or more tables with a VALIDPROC, for each such table:**
  - Alter the VALIDPROC to NULL (ALTER TABLE) You'll want to do this step for all tables in the TS that have a VALIPROC before running the REORG or LOAD REPLACE
  - Run REORG or LOAD REPLACE to convert the table space to RRF
  - Make any necessary changes to the VALIDPROC so that it can be used with rows in RRF
  - Add the VALIDPROC back to the table (ALTER TABLE)

Converting Table Spaces to RRF 2

- **If a table space cannot be converted to RRF because it contains one or more tables with an EDITPROC, for each such table:**
  - UNLOAD data from all tables with EDITPROCs
  - DROP all tables with EDITPROCs
  - Make any necessary changes to the EDITPROCs so that they can be used with rows in RRF
  - Run REORG to convert the table space to RRF
  - Recreate the tables with new EDITPROCs. Also recreate any indexes, check constraints, etc.
  - LOAD RESUME the data into the tables with EDITPROCs

- **NOTE: the parameter list used by EDITPROCs and VALIDPROCs (generated by DSNDROW) is modified so that RMFTTYPE may have a value of x'08' to indicate that the row is in RRF.**

LOAD and REORG Interaction – 1

LOAD REORG LOG Keyword	Table Space Logging Attribute	Table Space Type	What is logged	Table space status after Utility completes
LOG YES	**LOGGED**	**Non-LOB**	**Control records and data**	**No pending status**
LOG YES	**LOGGED**	**LOB**	**Control records and LOB data Redo information. LOB data Undo information is never logged for LOB table spaces.**	**No pending status**
LOG YES	**NOT LOGGED**	**Non-LOB**	**(LOG YES changed to LOG NO)**	**(See LOG NO)**
LOG YES	**NOT LOGGED**	**LOB**	**Control information**	**No pending status**
LOG NO	**LOGGED**	**Non-LOB**	**Nothing**	**COPY pending**
LOG NO	**LOGGED**	**LOB**	**Nothing**	**COPY Pending**
LOG NO	**NOT LOGGED**	**Non-LOB**	**Nothing**	**(See next slide)**
LOG NO	**NOT LOGGED**	**LOB**	**Nothing**	**No pending status**

This table illustrates how the LOAD and REORG ‘LOG’ keyword interacts with LOGGED and NOT LOGGED table spaces

LOAD and REORG Interaction – 2

Inline Copy	Records Discarded	Table space Status
YES	**NO**	**No pending status**
YES	**YES**	**ICOPY**
NO	**n/a**	**ICOPY**

- **This table is an expansion of the table on the previous slide for the case of a non-LOB NOT LOGGED table space with LOG NO specified. If LOG YES was specified, it is changed to LOG NO.**
- **LOAD LOG NO has an optional NOCOPYPEND option that indicates that the load should not leave the table space in Copy Pending.**
- **Because LOAD on a table in a NOT LOGGED table space never leaves the table space in Copy Pending, NOCOPYPEND is ignored.**
- **Therefore, if discard processing is necessary, the table space will be flagged as being in the ICOPY state, even if NOCOPYPEND is specified.**

REBUILD INDEX and CHECK DATA Considerations

- REBUILD INDEX:
  – When the NOT LOGGED attribute is specified at a table space level, the attribute property is applied to indexes defined on tables within that table space. Therefore, REBUILD INDEX SHRLEVEL(CHANGE), which requires log records to be applied, is not allowed on a NOT LOGGED table space.
- CHECK DATA:
  – For NOT LOGGED table spaces, if LOG YES is in effect, CHECK DATA issues new message DSNU1153I to indicate that the LOG YES option will be ignored. If rows are deleted in a NOT LOGGED table space, CHECK DATA will set the ICOPY state for the table space regardless of the LOG option.

Application Programming Considerations

- **Issuing frequent commits is still important when updating not logged objects**
- **Take care to avoid issuing a ROLLBACK after modifying a table in a NOT LOGGED table space, this including ROLLBACK TO SAVEPOINT.**
  - Doing this will cause new SQL CODE +20187, ‘`ROLLBACK TO SAVEPOINT CAUSED A NOT LOGGED TABLE SPACE TO BE PLACED IN THE LPL`’ to be issued.
- **Avoid issuing CANCEL THREAD for applications which update NOT LOGGED table spaces**
- **Avoid causing duplicate key or RI violations when modifying a table in a NOT LOGGED table space**
- **All of the above cause the table space or partition to be placed in the LPL and marked recover pending**