SWG BetaWorks

DB2 9 (for z/OS) Technical Education Series

"Partitioning & Clone Tables"

Important Disclaimer

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, OR SHALL HAVE THE EFFECT OF:

- CREATING ANY WARRANTY OR REPRESENTATION FROM IBM (OR ITS AFFILIATES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS); OR
- ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.

Agenda

- **Universal Table Space**
  - Partition By Range
  - Partition By Growth
- **APPEND**
- **Clone Tables**

Table spaces up to and including V8

- **Simple**
  - Multiple tables per table space, different tables share pages.
  - However, you have to write your own solution to do this making maintenance a real problem.
  - As a result they are not used by most customers.
    - Deprecated as of this release.
- **Segmented**
  - Multiple tables per table space, different tables don't share pages.
  - Handles mass deletes well.
  - Limited to 64Gb (32 x 2Gb datasets).
- **Partitioned**
  - One table per table space.
  - Really big (128Tb) … but needs a partitioning column.
  - But doesn't handle space usage or mass deletes as well as segmented.

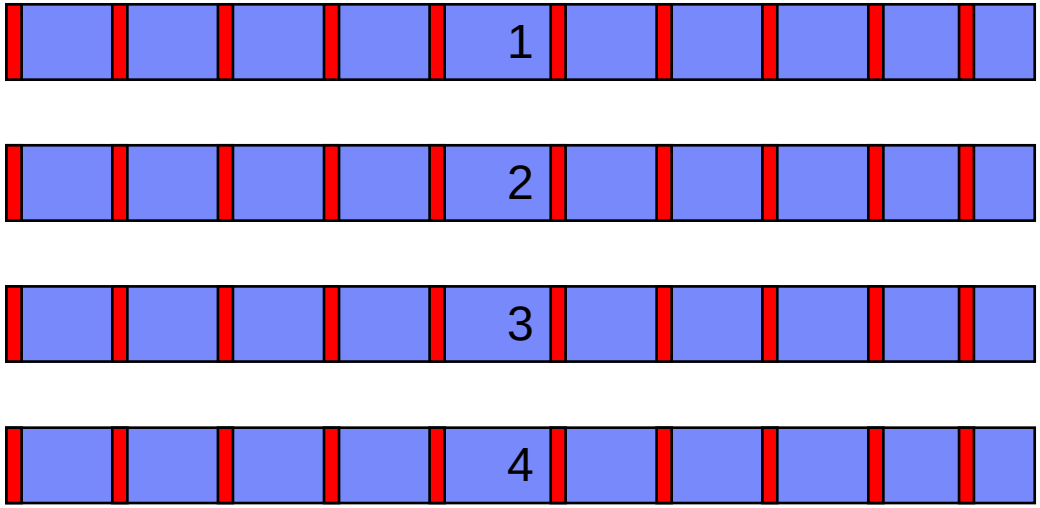
What we need …

- We need a partitioning table space with some of the segmented table space features.
- The advantages of segmented space maps for partitioned table spaces ?
- Universal Table space (Partition By Range)
  - A partitioned segmented table space.
  - Partitioning column required.
  - One table per table space.
- Universal Table space (Partition By Growth)
  - A partitioned segmented table space.
  - No partitioning column required.
  - One table per table space.

SWG BetaWorks

Partition By Range

Partition By Range



- **Just like a partitioned table space today with extra space map pages.**
  - Better space management = Fewer REORGs.
  - Improved mass delete performance.
  - Improved ALTER TABLE ROTATE performance.

Partition By Range - DDL

CREATE TABLESPACE TS1 IN DB1

NUMPARTS 55 SEGSIZE 64;

- Basically creates a partitioned table space but just add the SEGSIZE clause.
- SYSTABLESPACE has TYPE column value “R”.
- DB2 9 NFM.
- Not for the WORKFILE database.
- No MEMBER CLUSTER (not supported for segmented).
- No LOCKSIZE TABLE (uses partitioned table space locking scheme).
- No ALTER SEGSIZE so get SEGSIZE right or it is a DROP/CREATE.
- If the “Partition by Range” table space contains an XML column, the corresponding XML table space will be “Partition by Range” as well.
- All index RIDs will be 5 bytes.

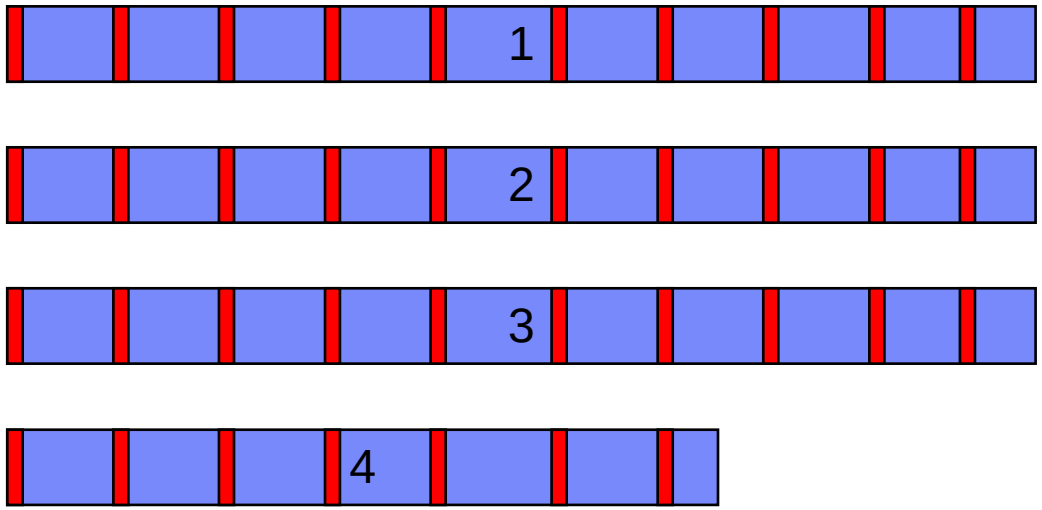
SWG BetaWorks

Partition By Growth

Partition By Growth - Need

- **Instead of partitioning by a RANGE (column values) we partition according to space needs.**
  - Allocated a partition when we need one due to growth.
- **Why?**
  - No obvious column to partition by
  - A segmented table requiring > 64Gb
    - May be a solution implemented using several tables with the application switching between them.
  - Space on Demand

Partition By Growth – How it works



- Partitions 1, Maxpartitions n, 1 rows on systablepart
- Partitions 2, Maxpartitions n, 2 rows on systablepart
- Partitions 3, Maxpartitions n, 3 rows on systablepart
- Partitions 4, Maxpartitions n, 4 rows on systablepart

- When table space created, one partition created and one row inserted to SYSTABLEPART (assuming created with DEFINE YES)
- When partition fills and not MAXPARTITIONS, new partition created and some catalog updates/inserts
  - Even if unit of work adding rows issues a rollback, new partition remains.
- Compression dictionary will be copied from previous partition to new partitions (also note freespace, caching, define, logging and trackmod attributes are same for each partition)

Partition By Growth – What Happens

- CREATE TABLESPACE …. MAXPARTITIONS n
  - You can set DSSIZE and SEGSIZE.
  - You can't use NUMPART and MAXPARTITIONS together to create say 10 now and but with a maximum of 50.

    MAXPARTITIONS gives you 1 to start with and only 1.

- CREATE TABLE …. PARTITION BY SIZE EVERY mG
  - Only available when you don't specify a table space on the CREATE TABLE.
  - Sets MAXPARTITIONS 256, DSSIZE mGB and SEGSIZE 4.
  - You can alter MAXPARTITIONS.

- Remember DSSIZE and SEGSIZE require a DROP to change, No ALTER option.

Partition By Growth – DB2 Catalog & DDL

- What happens in the DB2 Catalog?
  - SYSTABLEPART gets one entry to start with, an extra row added for each new partition required.
  - SYSTABLEPSPACE has TYPE column value “G” and new MAXPARTITIONS column with a non-zero.
  - SYSTABLESPACE PARTITIONS contains the number of partition (datasets) which currently exist.

- If you specify MAXPARTITIONS you get a partition by growth universal table space which is by definition segmented, whether you specify SEGSIZE or not.

Partition By Growth – Utilities

- REORG TABLESPACE
  - Can mean more or less partitions needed to hold data, if n to start with will be n or more at the end of the REORG i.e. no deleting of existing partitions. Note if MAXPARTITIONS reached REORG fails resource unavailable.
  - No parallelism to ensure data reduced to minimum number of partitions.
  - For rows with XML or LOB objects.
    - Shrinking partitions will not cause an XML or LOB object to relocate i.e. there will be spare space in the partition.
    - Growing partitions will cause REORG to fail because XML and LOB rows cannot change partition because associated objects will not move in the auxiliary table spaces.
  - No REBALANCE option, no key range to support it.

Partition By Growth – Utilities

- COPY
  - If n partitions when COPY utility starts, only first n copied.
- LOAD
  - Utility cannot parallel process on partitions.
- DSN1COPY
  - Specify NUMPARTS parameter with value of MAXPARTITIONS, not current number of partitions.
- If utility processing at partition level, beware number of partitions changes dynamically make sure your utility jobs can handle this.

Partition By Growth - Restrictions

- One table per table space.
- Non-partitioned indexes only.
- All index RIDs will be 5 bytes.
- Table space must be DB2-managed i.e. STOGROUPs.
- No MEMBER CLUSTER or LOCKSIZE TABLE.
- No ALTER TABLE – ADD PARTITION, ROTATE PARTITION, ALTER PARTITION.
- LOB and XML spaces associated with partition by growth are always implicitly defined, regardless of SQLRULES setting.
  - Note:- SQLRULES only applied to first partition. Any additional new added partition will always be created by DB2.
- LOB and XML rows cannot change partition because associated object will not move in auxiliary table space.
  - This can be caused by changing freespace, deletes, compression changes.
- Altering Maxpartitions causes DB2 to close and re-open the page set, a small outage.
  - Set Maxpartitions accordingly

CREATE TABLESPACE

- Note if the user does not specify SEGSIZE, NUMPARTS or MAXPARTITIONS a segmented table space with SEGSIZE of 4, LOCKSIZE of ROW will be created.

- You cannot create a SIMPLE table space any more.

Creating table spaces

SEGSIZE	MAXPARTITIONS	NUMPARTS	Type of table space created
SEGSIZE	MAXPARTITIONS		Partition-by-growth table space (UTS)
SEGSIZE			Segmented table space
SEGSIZE		NUMPARTS	Partition-by-range table space (UTS)
	MAXPARTITIONS		Partition-by-growth table space (UTS) implied SEGSIZE 4
			If PARTITIONED keyword, then partitioned; else simple (V8) or segmented (DB2 9) implied SEGSIZE 4
		NUMPARTS	Partitioned table space

SWG BetaWorks

APPEND

APPEND

- CREATE TABLE … APPEND YES/NO
  - APPEND YES tells DB2 to ignore clustering for SQL (e.g. Insert).
    - APPEND applies to all type of table spaces and can be altered between YES and NO.
  - Not Global Temporary Tables.
  - Can re-cluster table with a REORG.
- Why ?
  - Want to insert data fast and / or insert sequence will be random.
- But access path considerations
  - Want index access to a row or table space scan.
  - Not rid list, sequential, clustering order types of access, joins ?
- We think popular combinations could be:
  - NOT LOGGED and APPEND YES.
  - APPEND YES and partition by growth.

SWG BetaWorks

Cloned Tables

Agenda

- **Rationale and description**
- **DDL statements**
  - ALTER TABLE . . . ADD CLONE
  - EXCHANGE DATA BETWEEN . . .
- **Authorization**
- **Catalog and directory changes**
- **Commands and utilities**

Rationale

- ". . . many of our processes involve completely emptying the table and then reloading it in its entirety. In order to accomplish this the table space must be offline which does impact the availability of the business application . . . "
- ". . . It is increasingly difficult to do LOAD REPLACEs of data which is required for business as their business becomes more and more 24x7x365 with online transactions 24 hrs a day."

Description

Cloned table support provides the ability to generate a table with the exact same attributes as a table that already exists at the current server. This new table that's being created is referred to as the **clone** table of a base table.

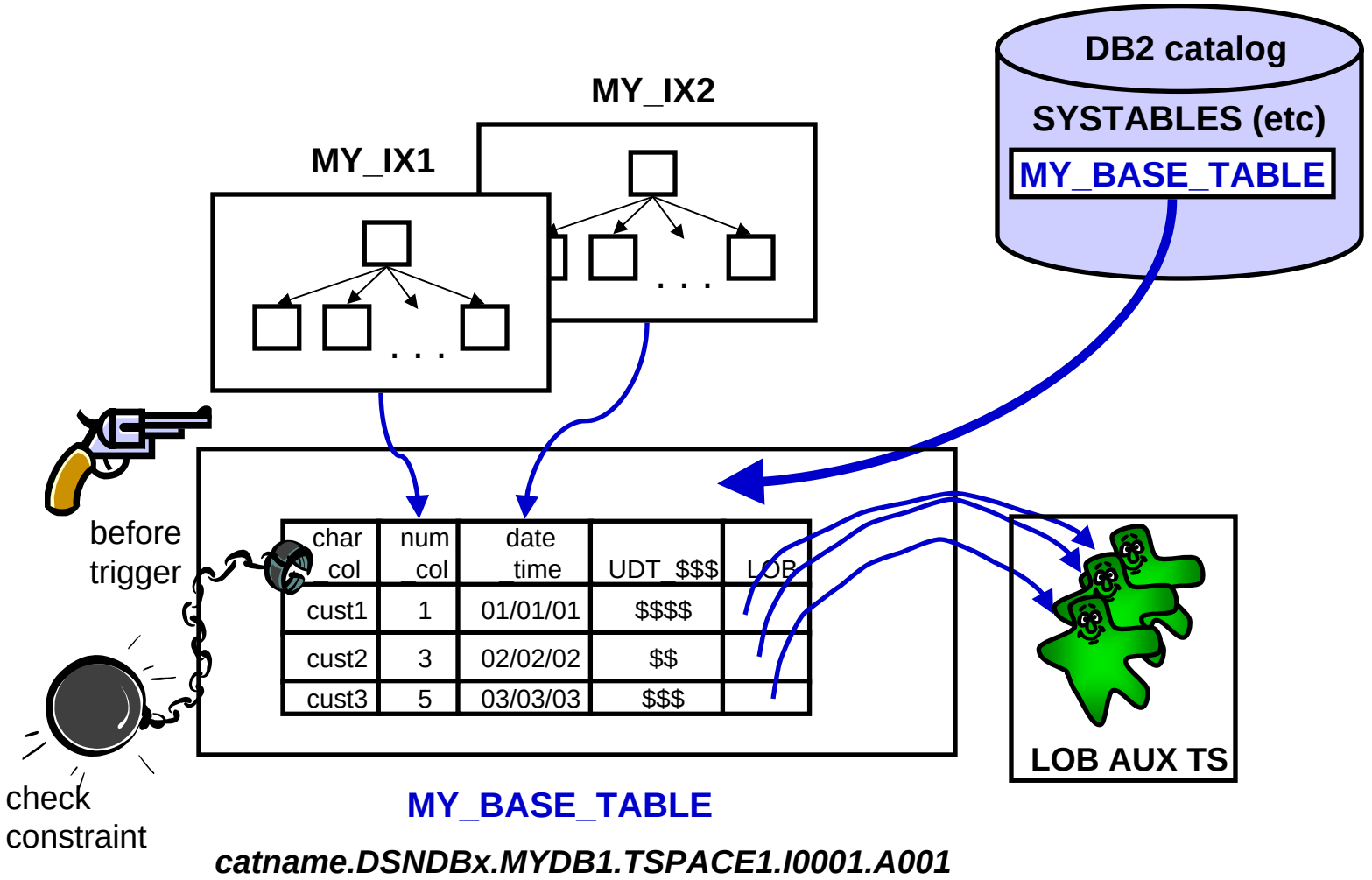
Not only is the clone table that's created identical to the base table in every way:

- same number of columns
- same column names
- same data types
- same check constraints, etc.

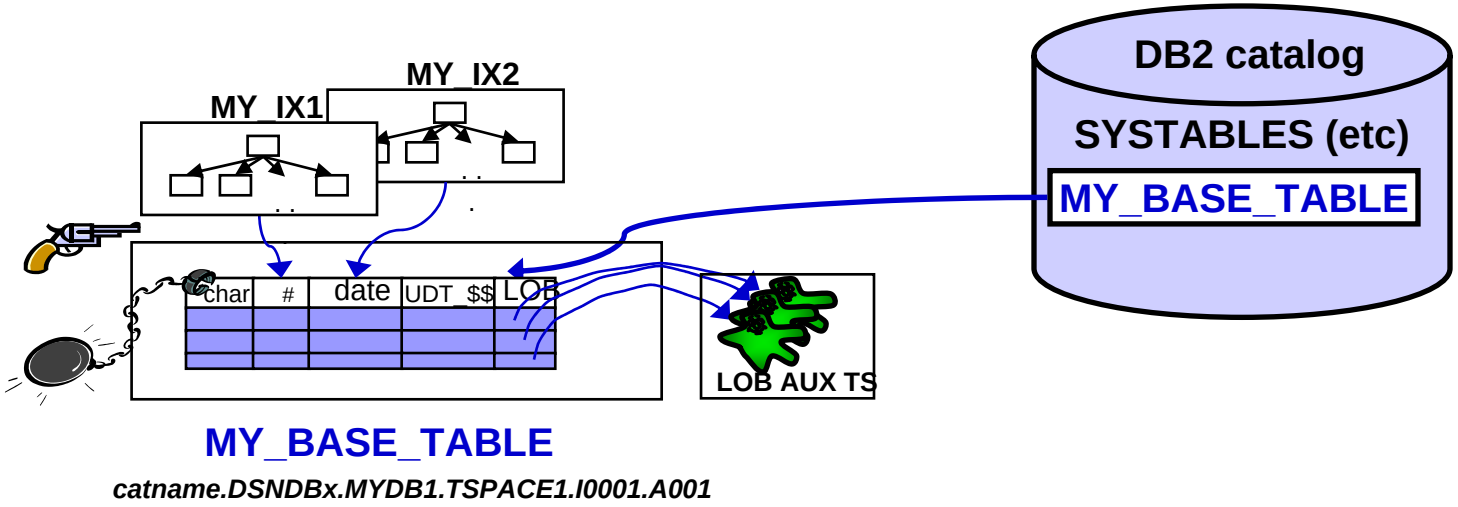
It's also created with the same

- indexes
- before triggers, etc.

Base table objects



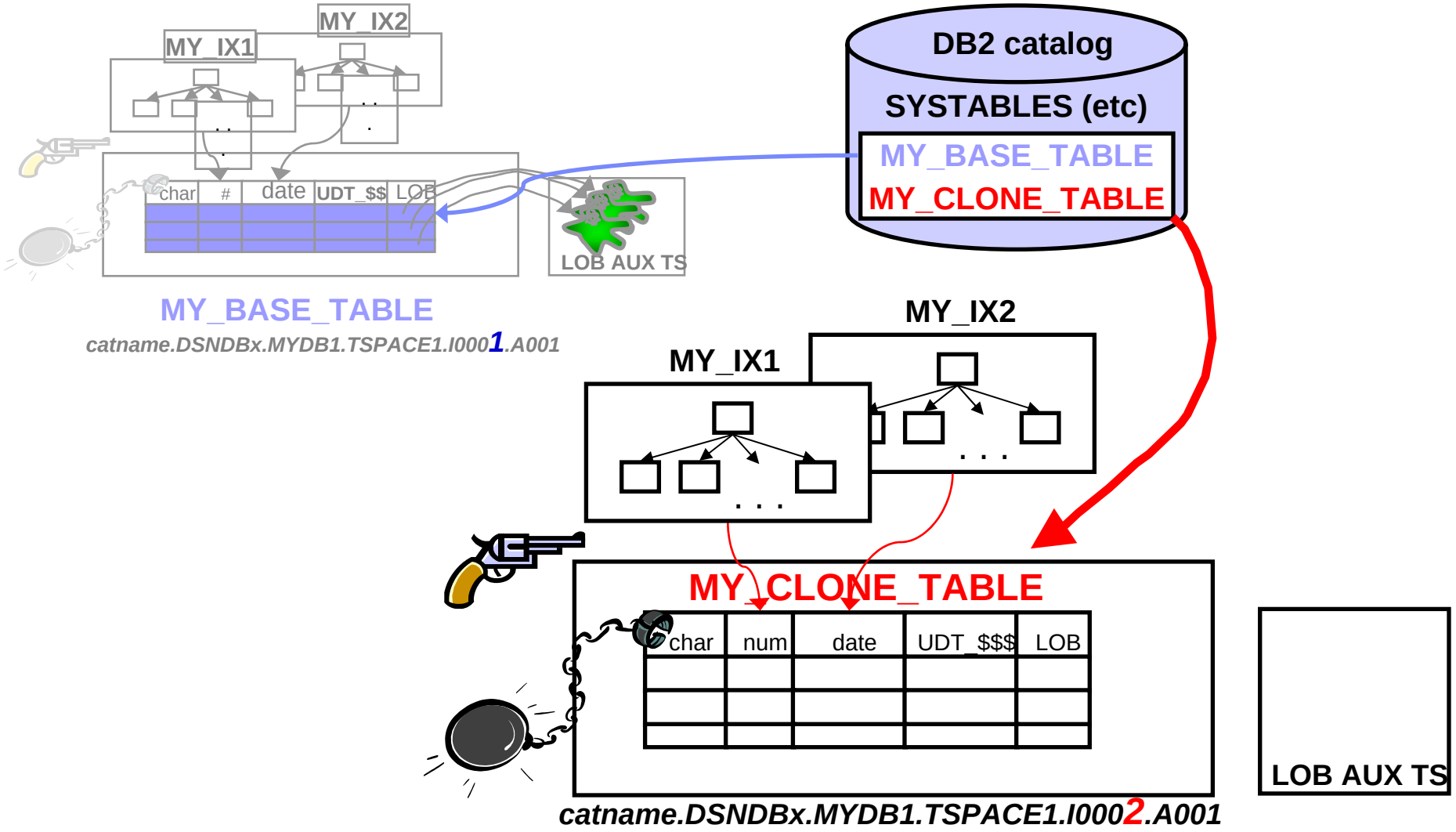
ALTER TABLE . . . ADD CLONE



ALTER TABLE **MY_BASE_TABLE**

ADD CLONE MY_CLONE_TABLE

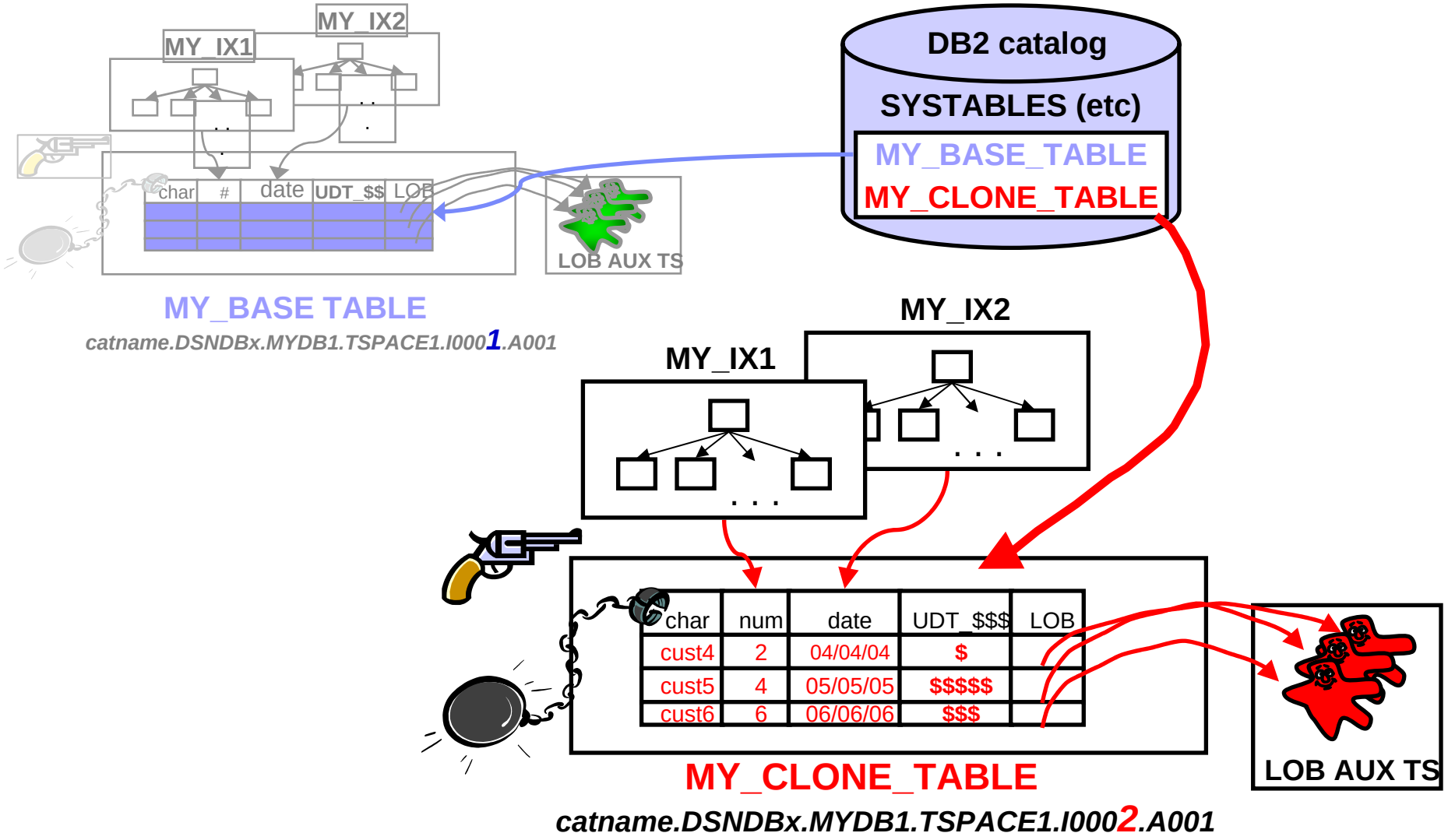
Base table and clone



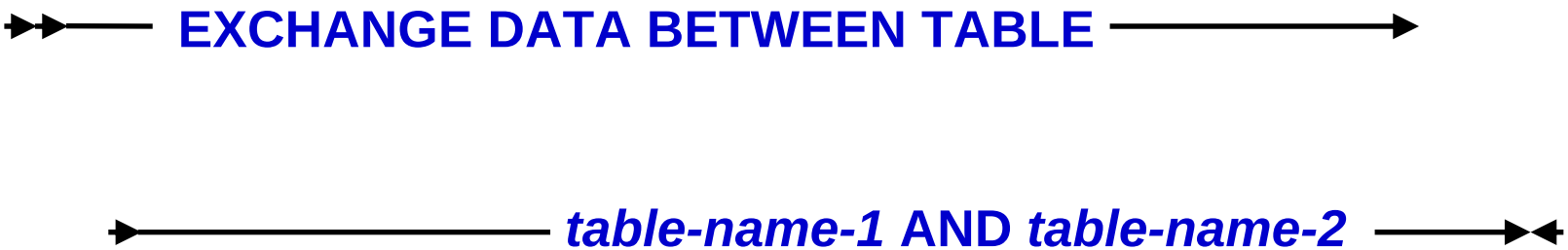
Notes about the clone

- Catalog tables cannot be cloned
- Could be created with a different schema (owner) than the base table
- Once created, access must be granted to users -- the clone functions like a “normal” DB2 table
- Like a “normal” DB2 table, after creation, the clone objects (table, indexes, aux objects, etc.) are empty. They can be populated by:
  - INSERT
  - LOAD
  - etc.

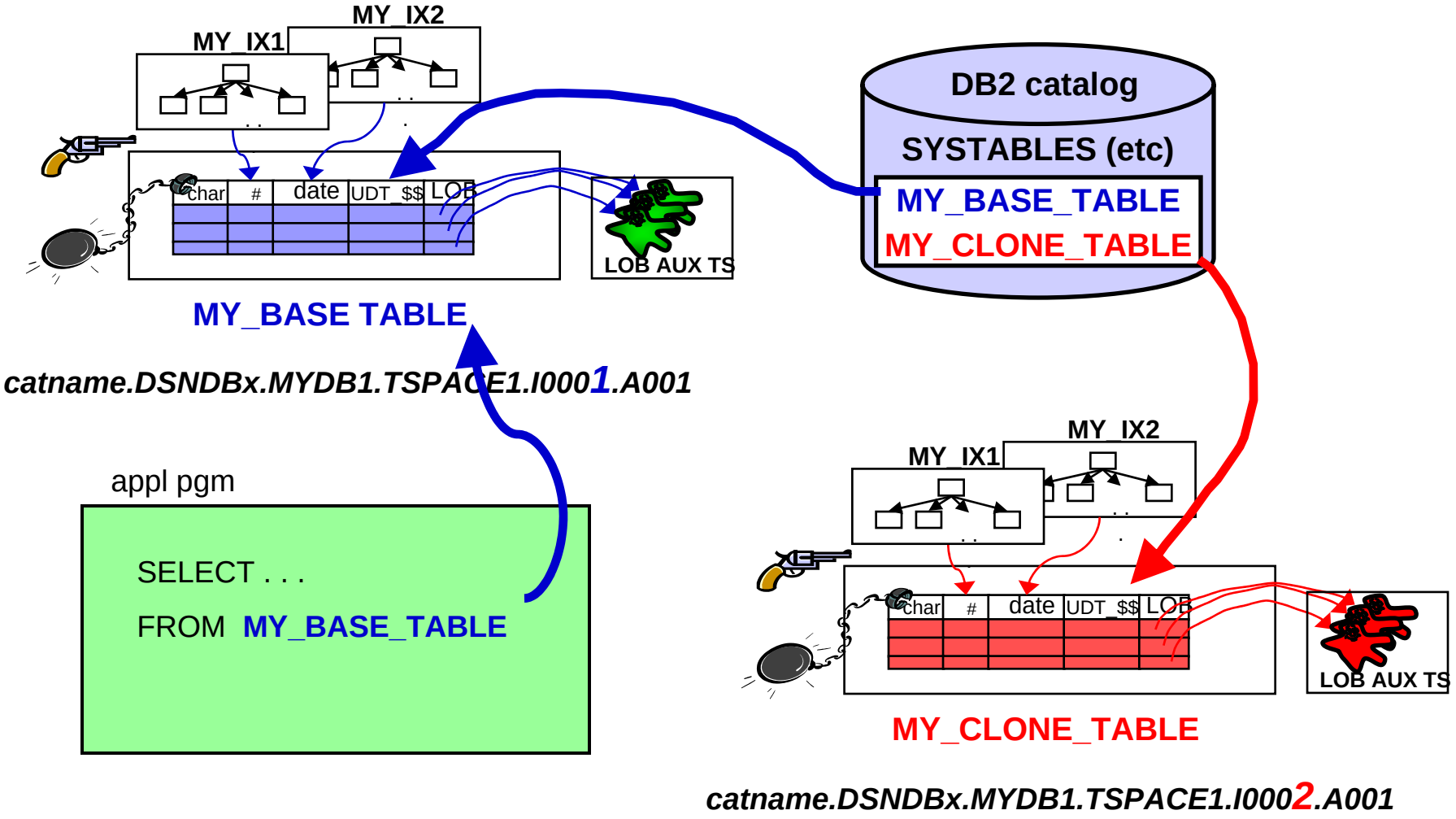
Load data into the clone



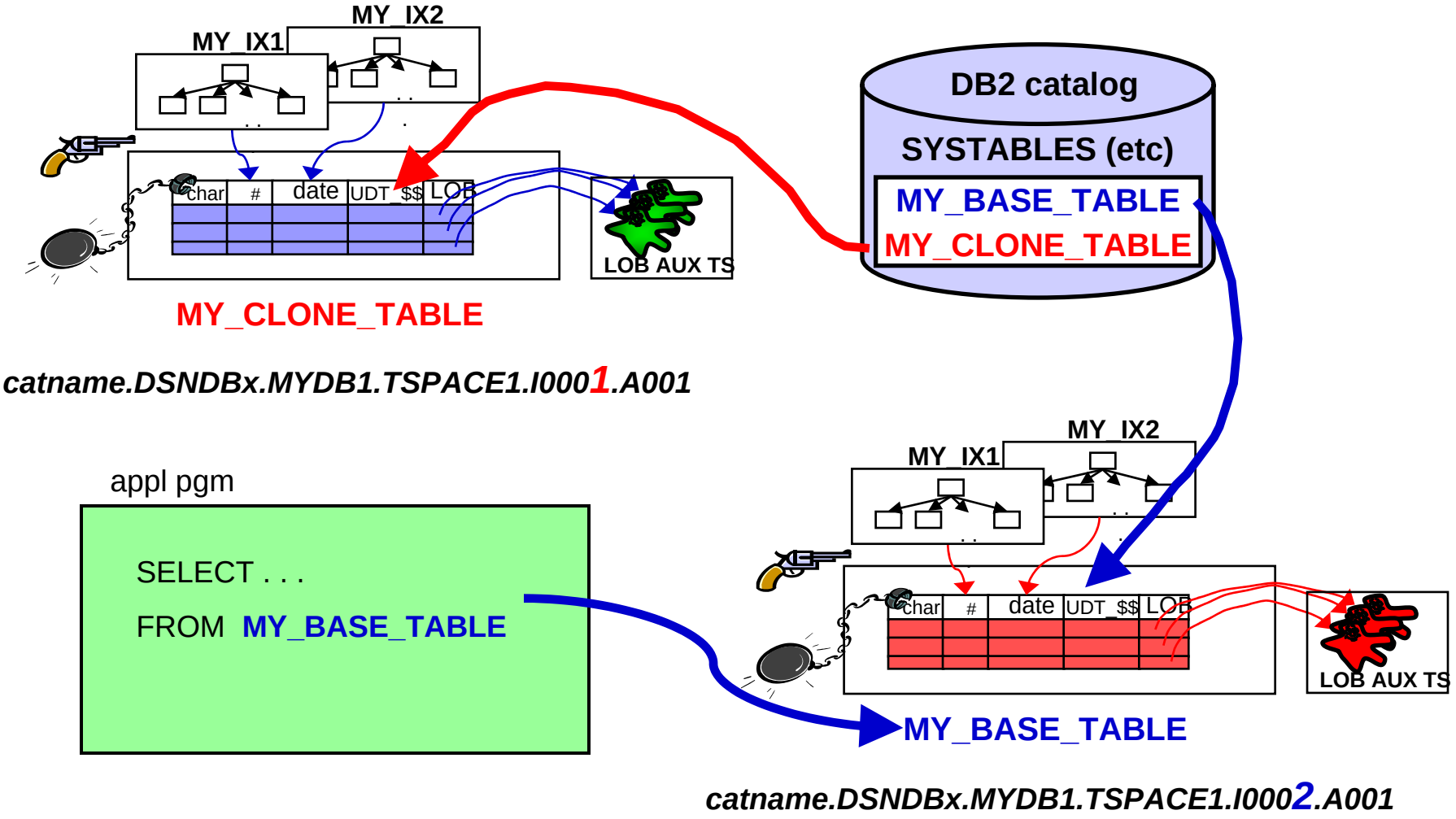
EXCHANGE



EXCHANGE (before)



EXCHANGE (after)



EXCHANGE notes

- EXCHANGE DATA BETWEEN . . .
  - Will switch the data set instance numbers. This *effectively* switches the data between the base and clone table objects. *No data is actually moved.*
  - Only applicable if cloning in effect. If cloning is in effect then no other ALTER TABLE clauses may be specified.
  - Must be done for all partitions of a partitioned table.
  - Must be a COMMIT between subsequent EXCHANGE statements

Notes about cloning

- Can only create a clone in a DB2 managed table space
- Can only create a clone for a table if it's the only table in a table space
- Can only create a clone if the table space is a universal table space
- No FASTSWITCH if there's a clone
  - Can create a clone if base table has a mix of I/J datasets. However, cannot fix/change this as long as the clone exists.
- No ALTERs (online schema changes for prod/cloned table)
- Cannot clone a table that has RI or add RI to a table if it is involved in cloning
- After triggers NOT allowed on either table. Only before triggers allowed.
- No clones allowed on MQTs

Statistics and the clone

- No RUNSTATS against clone table
- No real time stats for clone table
- On EXCHANGE, the prior clone assumes the base objects' statistics
  - This allows bound static SQL to function without a rebind
  - EXCHANGE invalidates the real time statistics for the base table

DDL . . .

- ALTER TABLESPACE changes apply to both base and clone objects.
- ALTER TABLE *base-table-name* DROP CLONE
  - Drops only the clone table objects and their underlying data sets.
- DROP TABLE will drop base table and its clone.

Authorization

- DML authorization on base and clone can be different

- EXCHANGE DATA BETWEEN . . . operation requires one of these:
  - Ownership of both tables
  - INSERT and DELETE privileges for both tables
  - DBADM for the database
  - SYSADM auth

- ALTER TABLE table-name ADD CLONE clone-table-name
  - Same privilege set as for CREATE TABLE for table being cloned:
    - CREATETAB for the database
    - DBADM, DBCTRL, DBMAINT
    - SYSADM, or SYSCTRL

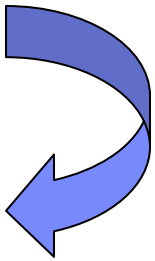
Catalog changes

- **SYSTABLESPACE**
  - INSTANCE SMALLINT NOT NULL WITH DEFAULT 1
  - CLONE CHAR(1) NOT NULL WITH DEFAULT 'N'

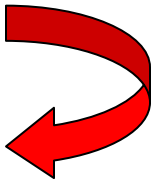
- **SYSCOPY**
  - INSTANCE SMALLINT NOT NULL WITH DEFAULT 1
  - The timestamp of a data exchange will be tracked in SYSCOPY in ICTYPE = 'A' and STYPE = 'E' records.

DISPLAY DATABASE

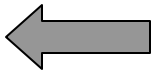
```
-DISPLAY DATABASE(MYDB*) SPACENAM(*) LIMIT(*)
DSNT360I ( ************************************
DSNT361I ( * DISPLAY DATABASE SUMMARY
. . .
NAME     TYPE     PART     STATUS                    . . .
-------- ----     ----     ------------------        . . .
CHKOUT2  TSB1     001      RW
CHKOUT2  TSB1     002      RW
. . .
XDEPT1   IXB1     001      RECP
XDEPT1   IXB1     002      RW
. . .
CHKOUT2  TSC2     001      RO
. . .
XDEPT1   IXC2     001      ICOPY
. . .
CHKOUT3  TS       001      RO
. . .
******* DISPLAY OF DATABASE MYDBX ENDED **********************
```



base objects



clone objects



objects not involved in cloning

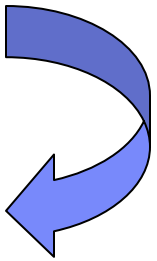
EXCHANGE affect on DISPLAY DATABASE

EXCHANGE DATA BETWEEN TABLE

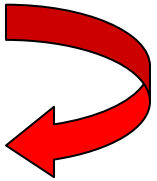
table-name-1* AND *table-name-2

DISPLAY DATABASE after EXCHANGE

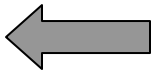
```
-DISPLAY DATABASE(MYDB*) SPACENAM(*) LIMIT(*)
DSNT360I ( ************************************
DSNT361I ( * DISPLAY DATABASE SUMMARY
. . .
NAME     TYPE     PART     STATUS                    . . .
-------- ----     ----     ------------------        . . .
CHKOUT2  TSB2     001      RO
CHKOUT2  TSB2     002      RW
. . .
XDEPT1   IXB2     001      ICOPY
XDEPT1   IXB2     002      RO
. . .
CHKOUT2  TSC1     001      RW
CHKOUT2  TSC1     002      RW
. . .
XDEPT1   IXC1     001      RECP
XDEPT1   IXC1     002      RW
. . .
CHKOUT3  TS       001      RO
. . .
******* DISPLAY OF DATABASE MYDBX ENDED **********************
```



base objects



clone objects



objects not involved in cloning

Commands & Stand alone utilities

- **STOP/START & DISPLAY DATABASE gain a CLONE keyword**
  - But display by default displays all objects…
  - Stop/Start by default processes only base objects, unless CLONE specified then just CLONEd objects.

- **DSN1LOGP**
  - DSN1LOGP can print the log records for both base and clone table objects. The base and clone objects are differentiated by differences in the PSID value.
- **DSN1COPY**
  - DSN1COPY can operate on both base and clone table objects. The base and clone objects are differentiated by differences in the PSID value.

Utilities with changes

- **CHECK DATA**
- **CHECK INDEX**
- **CHECK LOB**
- **COPY**
- **COPYTOCOPY**
- **DIAGNOSE**
- **LISTDEF**
- **MERGECOPY**
- **MODIFY RECOVERY**
- **MODIFY STATISTICS**
- **QUIESCE**
- **REBUILD INDEX**
- **RECOVER**
- **REORG INDEX**
- **REORG TABLESPACE**
- **REPORT**
- **RUNSTATS ***
- **UNLOAD**

Note: FASTSWITCH cannot be specified for utilities on any object involved with cloning (base table, clone table, indexes)!

Utilities 1

- **CHECK DATA *table-space-spec . . .* CLONE . . .**
  - Indicates that CHECK DATA is to check the clone table in the specified table space. Because clone tables cannot have referential constraints, the utility checks only table check constraints and for consistencies between the clone table data and the corresponding LOB data. If you do not specify CLONE, CHECK DATA operates on the base table.
- **CHECK INDEX . . . CLONE**
  - Indicates that CHECK INDEX is to check only the specified indexes that are on table clones.
- **CHECK LOB TABLESPACE *lob-table-space-spec* . . . CLONE**
  - Indicates that CHECK LOB is to check the LOB table space data for only the table clone, note the LOB data for the base table.

Utilities 2

- **COPY . . . CLONE . . .**
  - Indicates that COPY is to copy only table clone data in the specified table spaces or indexes on table clones in the specified index spaces. If you use the LIST keyword to specify a list of objects, COPY processes only those objects in the list that contain table clones or indexes on table clones. The other objects in the list are ignored.
- **COPYTOCOPY . . . CLONE . . .**
  - Indicates that COPYTOCOPY is to process only image copy data sets that were taken against table clones or indexes on table clones.
- **DIAGNOSE . . . CLONE . . .**
  - Indicates that DIAGNOSE is to display information for only the specified objects that are table clones, table spaces that contain table clones, indexes on table clones, or index spaces that contain indexes on table clones.

Utilities 3 - LISTDEF

- **CLONE**
  - Indicates that the INCLUDE or EXCLUDE expression is to return only clone tables, table spaces that contain clone tables, indexes on table clones, or index spaces that contain indexes on clone tables. If you specify ALL, LOB objects are also included. The CLONE keyword is ignored if you also specify a table name.
- **ALL**
  - Specifies that both BASE and LOB objects are to be included in the list. Auxiliary relationships are to be followed from all objects that result from the initial object lookup, and both BASE and LOB objects are to remain in the final enumerated list. Clone objects not included.
- **TABLE *creator-id.table-name***
  - Specifies the table that is to be used for the initial search for the object. . . . If you specify a table name with CLONE, the CLONE keyword is ignored.

Utilities 4

- **MERGECOPY . . . CLONE . . .**
  - Indicates that MERGECOPY is to process only image copy data sets that were taken against clone objects.
- **MODIFY RECOVERY . . . CLONE . . .**
  - Indicates that MODIFY RECOVERY is to delete SYSCOPY records and any related SYSLGRNX records for only clone objects. DBD entries for clone tables are not deleted.
- **QUIESCE . . . CLONE . . .**
  - Indicates that QUIESCE is to create a quiesce point for only the specified table spaces that contain clone tables.

Utilities 5

- **MODIFY STATISTICS . . . CLONE . . .**
  - The online MODIFY STATISTICS utility deletes unwanted statistics history records from the corresponding catalog tables. You can remove statistics history records that were written before a specific date, or you can remove records of a specific age. You can delete records for an entire table space, index space, or index.

    Run MODIFY STATISTICS regularly to clear outdated information from the statistics history catalog tables. By deleting outdated information from those tables, you can improve performance for processes that access data from those tables.

Restriction: MODIFY STATISTICS does not process table clones, because statistics are not collected for these tables.

Utilities 6

- **REBUILD INDEX . . . CLONE . . .**
  - ALL
    - All indexes in the table space referred to by the TABLESPACE keyword are to be rebuilt. Indexes on table clones are not included; only indexes on the base table are included.
  - CLONE
    - Indicates that both REBUILD INDEX is to reconstruct only the specified indexes that are on table clones. If you specify CLONE, you cannot specify STATISTICS. Statistics are not collected for table clones.

- **RECOVER . . . CLONE . . .**

  No special recovery considerations for a cloned table space or cloned index.
  - CLONE
    - Indicates that RECOVER is to recover only table clone data in the specified table spaces or the specified index spaces that contain index on table clones.

Utilities 7

- **REORG INDEX . . . CLONE . . .**
  - REORG INDEX is to reorganize only the specified index spaces that contain indexes on table clones.
- **REORG TABLESPACE . . . CLONE . . .**
  - REORG TABLESPACE is to reorganize only table clones from the specified table spaces. Base tables in those table spaces are not reorganized. If you specify CLONE, you cannot specify STATISTICS. Statistics are not collected for table clones.
- **REPAIR . . . CLONE**
  - Indicates that REPAIR is to process only the specified objects that are table spaces that contain table clones, indexes on table clones, or index spaces that contain indexes on table clones. If you specify CLONE, you cannot specify VERSIONS, because table clones to not have versions.
    Clones cannot be created for tables with active versions.

    If you specify SET with CLONE, the status is changed for only the specified clone tables and their indexes. The CLONE keyword applies to all SET statements and LOCATE statements within the same REPAIR utility control statement.

Utilities 8

- **REPORT**
  - The report provides information for both base and clone objects. If you specify TABLESPACESET, REPORT also processes related LOBs.

- **UNLOAD . . . CLONE . . .**
  - Indicates that UNLOAD is to unload data from only clone tables in the specified table spaces. If you specify the name of the clone table in the FROM TABLE clause, you do not need to specify the CLONE keyword.

"Thank You for listening"

If you have any questions on this DB2 9 for z/OS session, then please send them to the BetaWorks team at:

Ian_Cook@uk.ibm.com
FLETCHPL@uk.ibm.com