

Db2 for z/OS 2025 End-of-Year SWAT Tales Extended Version

Anthony Ciabattoni, Executive IT Specialist - Db2 for z/OS SWAT Team

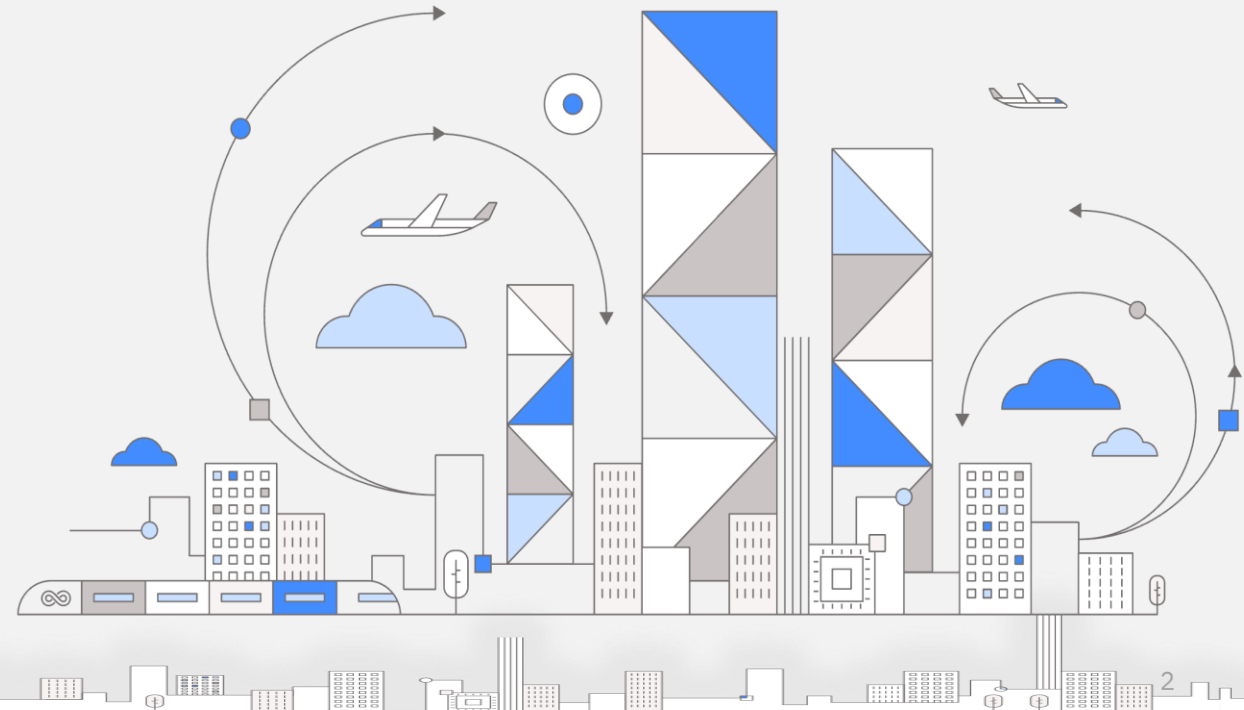
Kirt Dailey, Senior IT Specialist – Db2 for z/OS SWAT Team

Alysia Moreno, IT Specialist – Db2 for z/OS SWAT Team

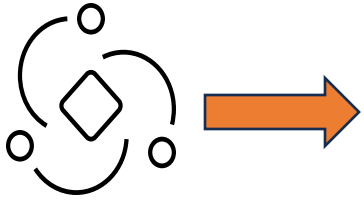


Agenda

- Table Space History
- Universal Table Spaces
- Universal Table Space Conversions
- Db2 REBIND Access Path Stability
- Statement Level Invalidation
- Recovery Boosts
- Db2 Log Scalability
- Online Deletion of an Active Log
- Questions



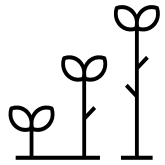
History of Db2 base table types



Simplicity

- **Simplicity**

- Simple table space
- Easy to use
- Capacity limitation
- Data management challenge for multiple tables



Scalability

- **Scalability**

- Partitioned table space - Larger object and be able to manage data in a selective subsets
- More flexible for data volume growth in the future
- Less flexible in data manipulation with larger data set



Better Data Manipulation
structure

- **More flexible with data management**

- Segmented table space
- Higher data availability and data accessing performance
- More flexible with multiple tables
- Capacity limitation of 64G per table space

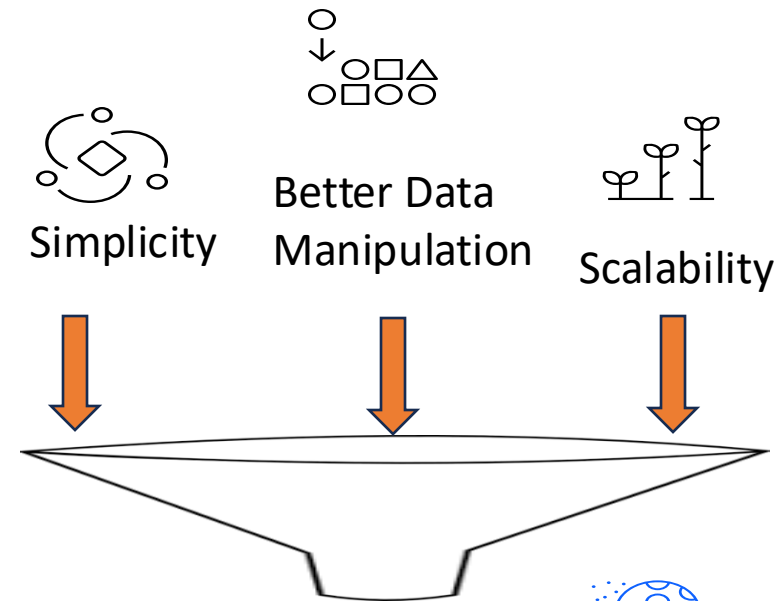


Why Universal Table Space (UTS)

- Why Universal Tablespaces ?
 - Bring together the advantage of partitioned and segmented table space
 - Partition scheme provides higher scalability
 - Segmented structure provides more flexible of data management
 - Single table per table space for simplicity
 - More Flexibility
 - New features/functionality



More Flexibilities



Universal Table Space



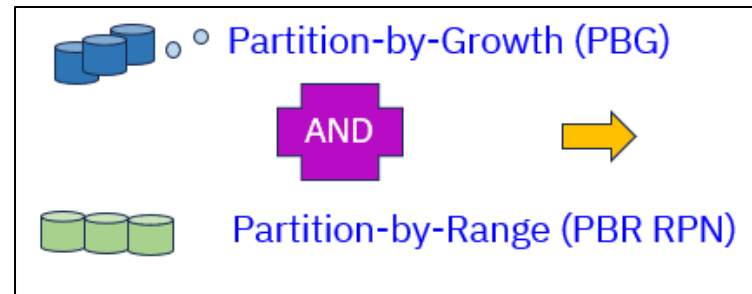
Why Universal Table Space (UTS) ...

- Why Universal Tablespaces ? ...
 - Support of non-UTS table spaces is approaching the end
 - All table spaces must be converted to Universal Table Spaces before migrating to the next release of Db2 (Db2 13+1)
 - Must develop a plan to convert any residual legacy table spaces to Universal Table Space



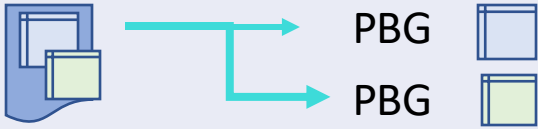
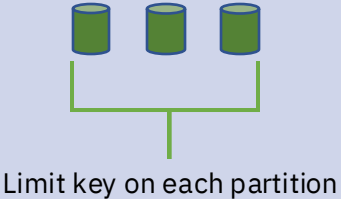
UTS general design rationale

- Partition by Growth (PBG) UTS
 - Should be seen as a replacement for classic table spaces
- Partition by Range (PBR) UTS
 - Should be seen as a replacement for classic partition table space
 - PBR UTS – Partition by Range Absolute page number
 - PBR UTS – Relative Page Number (RPN)
 - Strategic Direction
 - Partition level DSSIZE
 - Supersized UTS PBR
 - Up to 1TB partition size or 4 Petabytes of table space
 - Max number of rows of 4K/35 trillion rows for 32K pages



The strategic table space
types for Db2 tables

Non-UTS to UTS table space conversion

Table space type	Pending ALTER DDL	Target converted table space type
Single table : Segmented / simple table space	ALTER TABLESPACE ... <u>MAXPARTITIONS</u> n	Partitioned-by-growth (PBG UTS) 
Multiple tables: segmented	ALTER TABLESPACE ... <u>MOVE TABLE</u> Tbname TO <u>TABLESPACE</u> TSname	
Classic partitioned table space	ALTER TABLESPACE ... <u>SEGSIZE</u> n	Partitioned-by-Range (PBR UTS) 



Converting legacy table spaces

Simple table space or
Segmented table space

A single table in the table space

```
ALTER TABLESPACE ...MAXPARTITIONS n
```

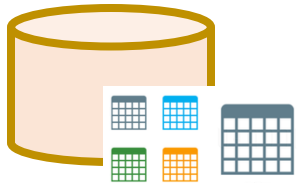
REORG table
space

Partition by
Growth



Segmented table space

Multiple Tables



```
ALTER TABLESPACE ...MOVE TABLE Tbname  
TO TABLESPACE newtsname
```

REORG table
space

Partition by
Growth



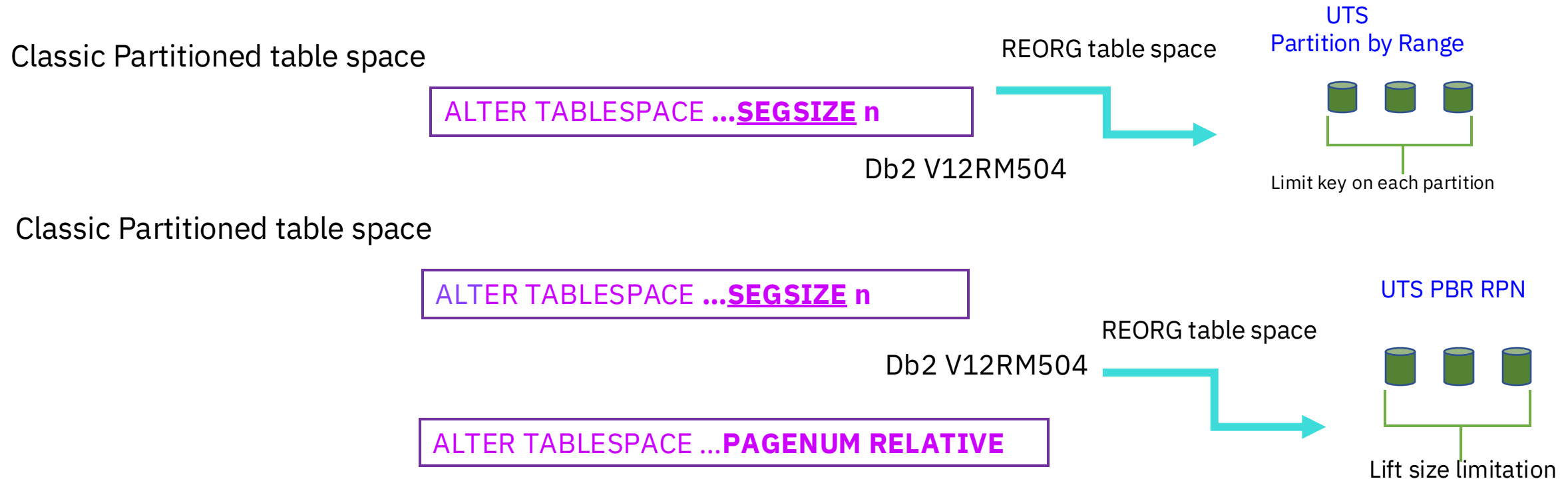
Partition by
Growth



Db2 V12R1M508

- Non-UTS table space has been deprecated through out the releases
- Non-Universal table spaces are still to be supported in Db2 13
 - Simple table space is not supported since release 9
 - All other objects are initially deprecated @V12R1M504
- Converting multiple tables in segmented table space to PBG
 - Using MOVE TABLE DDL
 - Available in V12RM508

Converting classic partition table space



- Converting classic partition table space to UTS PBR or UTS PBR RPN
 - Stacking two alters with one REORG materialize all the changes



When to consider converting UTS PBG to UTS PBR RPN

- Under certain circumstances PBG UTS may no longer be the optimal UTS choice
 - Table usage and data volume usage has changed over time
 - Incorrect historical decision based on the point-in-time knowledge
 - *“Set it and forget it”*
- Consequences
 - Insert or query performance degradation due to size of table space
 - Performance degradation can occur for large tables in PBG table spaces, the size of the table space is often a major cause.
 - Db2 13 also introduces SQL INSERT enhancements for crossed partition search that can help with certain large PBG table space situation
 - Size of LOB table space associated with base table
 - ✓ PBR table space is more flexible to control the size of the LOB table space
 - Problems associated with very large non-partitioned indexes (NPI)
 - Difficulty completing REORG to maintain data clustering
 - Lack of parallelism features support for utilities
 - Limited support for partition-level utility operations
 - Minimal utility parallelism capabilities are fully supported for PBG table spaces
- Blog in IBM Data Management Community

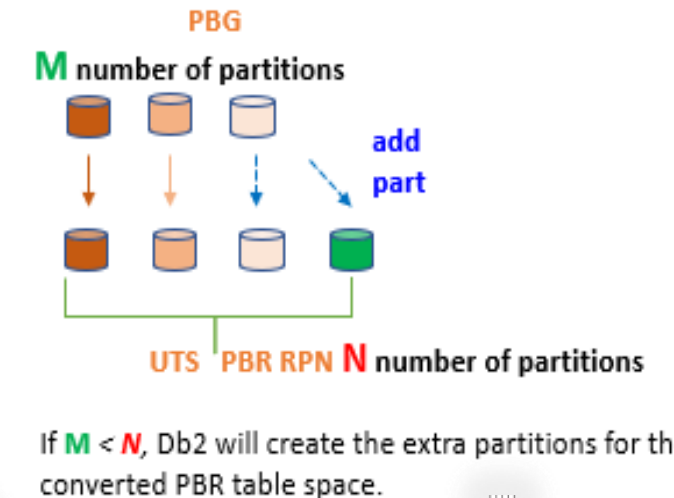
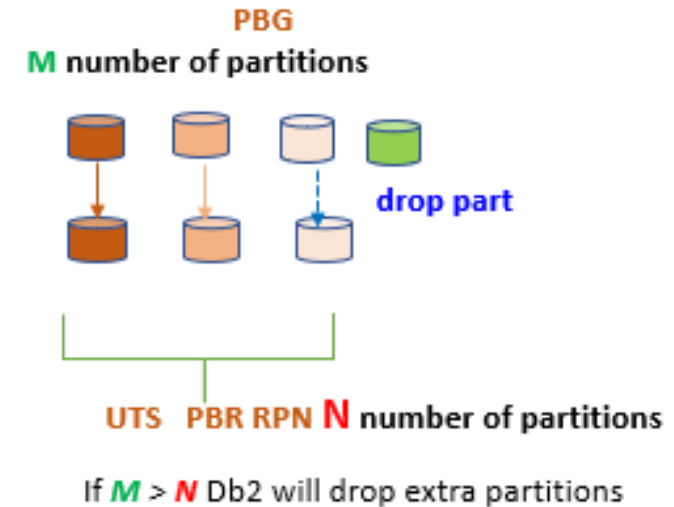


<https://community.ibm.com/community/user/blogs/frances-villafuerte/2022/08/15/convert-tables-in-pbg-table-spaces-to-pbr-db2-13?hlmlt=BL>



Basic concept of converting PBG to PBR RPN

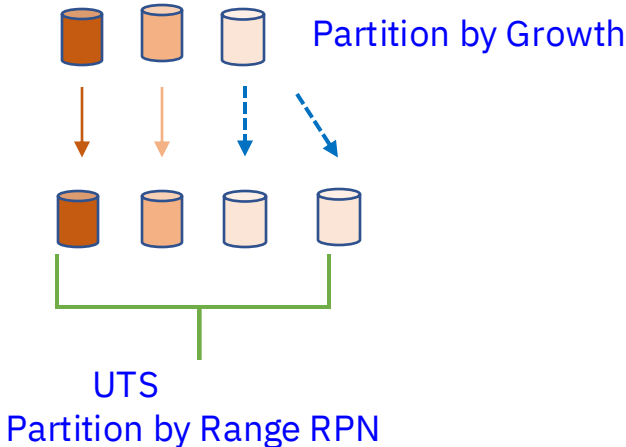
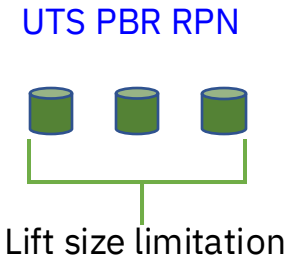
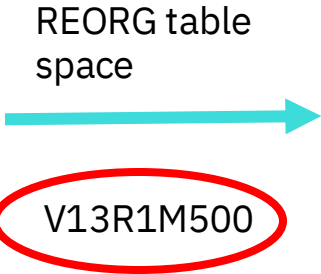
- There are no changes to the table space OBIDs and table OBID
- The converted table space is UTS PBR RPN
- The number of partitions between two table spaces can be different
 - **For Replication** – the target converted table space must contain equal or more partitions
- The existing NPI indexes on the PBG table is still NPI
 - Db2 will not change any aspects or attributes of these indexes through conversion
- No Partitioned indexes (PI) will be created
 - If desired, user can create after the conversion has been materialized
- Conversion from UTS PBG to UTS PBR needs to meet the following conditions
 - High limit key for last partition requires MAXVALUE or MINVALUE
 - Cannot be an accelerated table
 - Cannot have unmaterialized pending changes



Converting from UTS PBG to UTS PBR RPN

UTS Partition by Growth (PBG)

```
ALTER TABLE ... ALTER PARTITIONING TO  
PARTITION BY RANGE
```



```
ALTER TABLE SCR001.TB01 ALTER PARTITIONING TO  
PARTITION BY RANGE (ACCT_NUM)  
( PARTITION 1 ENDING AT (199),  
PARTITION 2 ENDING AT (299),  
PARTITION 3 ENDING AT (399),  
PARTITION 4 ENDING AT (MAXVALUE) );
```



Partitioning key considerations PBG to RBR RPN

- Options of selecting partition key column
 - Exiting column from the table
 - No application changes required
 - No natural partitioning key for data distribution
 - Adding a new column to the table
 - Applications required
 - Utilizing Db2 managed value
 - Benefit
 - ✓ Db2 automatically manages a unique value, either in a monotonically ascending order within the table or randomized value such as **ROWID** column
 - ✓ Application does not need to manage the value of the key
 - **ROWID**
 - Data type can uniquely identify rows in a Db2 data sharing group
 - Generated value for the column when a row is inserted/REORG
 - Can be created as a **hidden column**
 - Supported on CREATE TABLE starting Db2 12
 - Can be ALTER added to existing table
 - Enabled in APPLCOMPAT **V13R1M506** (UTS online)



Using Db2 managed value – ROWID column

- Characteristic of column with ROWID data type (aka ROWID column)
 - A data type can uniquely identify rows in a Db2 data sharing group
- For table has LOB column, the ROWID column is required
 - Column with ROWID data type can be created without LOB column
- ROWID value
 - The value is generated for the column when a row is inserted/REORG
 - If alter added,
 - Can specify Db2 to generated default value for existing rows during query
- It can be implicitly or explicitly created
 - Can be created as *hidden* column
 - It is supported on CREATE TABLE starting V12 NFM
 - Can be ALTER added to the existing table
 - Enabled in APPLCOMPAT *V13R1506*
 - Only supports on UTS table space
 - ✓ **SQLCODE -270** for Non-UTS



Using Db2 managed value – ROWID column ...

- Benefit
 - Creates as *hidden*, no application changes
 - Unique value for each data row within the object
 - Value is generated from the scramble of LOG SEQUENCE number
 - Using for partition key for ease control of data distribution among table space
 - Once the value is generated it will not be changed at all
 - Easier to manage
- Alter added a ROWID column
 - No immediate REORG required to materialize the value of the column



Using Db2 managed value – ROWID column ...

- Example of ROWID value
 - Scramble 17 bytes of LOG sequence number
 - Internally format

A3E414F191F7B108609C014C6B20010000

A54AC14F191F7B609609C014C6B2001000

- If ALTER add to the table,
 - Db2 generated the internal value base on the position of the row within the data base

6440000080000117000302010000000000



Example of adding hidden ROWID column

```
CREATE DATABASE TestDB .....  
commit ;  
CREATE TABLESPACE TestTS IN TestDB ....  
    MAXPARTITIONS 5;  
commit ;
```

```
CREATE TABLE TestTB  
    (COL1 BIGINT DEFAULT 1,  
    ...  
    COL18 CHAR(160) DEFAULT 'COL18')  
IN TESTTS.TESTDB;
```

```
ALTER TABLE TestTS.TestDB  
    ADD COLUMN HIDROWID ROWID  
    NOT NULL GENERATED ALWAYS IMPLICITLY HIDDEN;  
COMMIT;
```



Example of using ROWID column as partitioning key

```
ALTER TABLE TestTS.TestTB;
```

```
Commit;
```

```
ALTER PARTITIONING TO PARTITION BY RANGE  
(HIDROWID)
```

```
(PARTITION 1 ENDING AT (x'3000'),  
PARTITION 2 ENDING AT (x'6000'),  
PARTITION 3 ENDING AT (x'9000'),  
PARTITION 4 ENDING AT (x'C000'),  
PARTITION 5 ENDING AT (x'D000'),  
PARTITION 6 ENDING AT (x'F000'),  
PARTITION 7 ENDING AT (MAXVALUE)  
)  
;
```

Converting from UTS PBR to UTS PBG

UTS Partition by Range
(Absolute page numbering
or
Relative page numbering)

```
ALTER TABLE ... ALTER PARTITIONING TO  
PARTITION BY GROWTH
```

REORG table
space

V13R1M507

Partition by
Growth



```
ALTER TABLE E8071.TB01  
ALTER PARTITIONING TO PARTITION BY GROWTH  
DSSIZE 8G MAXPARTITIONS 10
```

Note: The syntax is enforced at application compatibility level V13R1M507



Stacking Support

- Stacking support of the following pending alter options with one REORG materializing all the changes
 - Table space level
 - MAXPARTITION
 - BUFFERPOOL
 - DSSIZE
 - SEGSIZE (exclude using SEGSIZE for conversion to UTS)
 - MEMBER CLUSTER
 - Table level
 - DROP COLUMN, ALTER COLUMN
 - ✓ Only apply to ALTER COLUMN is pending alter
 - ✓ Set ZPARM DDL_MATERIALIZATION = ALWAYS_PENDING
 - ALTER PARTITIONS
 - ✓ Only support syntax for **PBR-->PBG, PBG-->PBR or PBR-->PBG-->PBR** with one REORG materialization process
 - ✓ No support of the stacked **PBG-->PBR-->PBG**
 - Index level
 - BUFFERPOOL
 - COMPRESS attribute

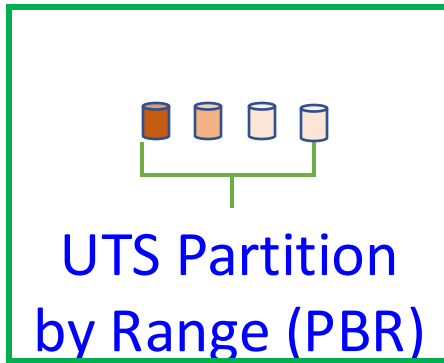


Innovating partitioning changes

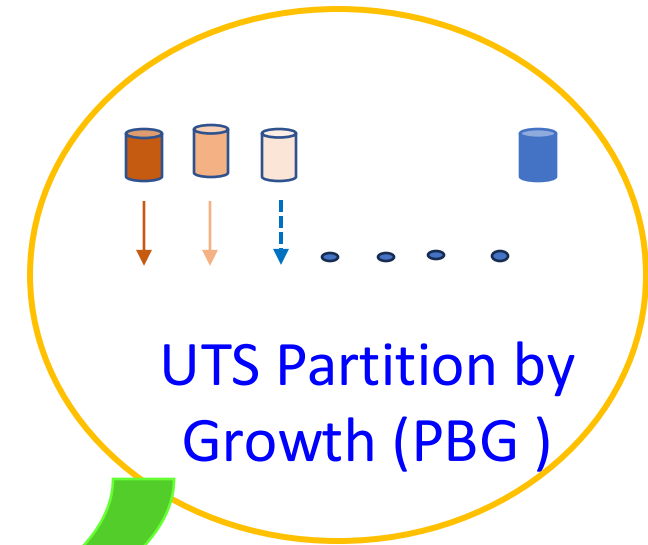
- Partition schema changes utilizing table space conversions within UTS
 - Db2 support PBR -- > PBG -- > PBR with one REORG materialization process
 - Achieve high availability partition schema changes
 - Partition limit key modification
 - ✓ Add column in the partition key
 - ✓ Single Alter statement requires rolling out the affected partition one at a time
 - ✓ Conversion enables ALTER limit key values of multiple partitions via a single ALTER statement
 - Drop partitions
 - ✓ Current DDL DROP PARTITION statement only supports the drop of the trailing partition
 - ✓ Conversion enables the dropping the middle of the partition with new partition scheme definitions
 - Removes the legacy 40-byte truncated limit key values
 - Benefits of new feature
 - Allows high availability with minimum impact to applications
 - Simplifies the process
 - Saves on valuable people resources



Partition schema changes using pending ALTER



- Drops all the partitioning key definition on the PBR UTS
 - Number of partitions can be changed
- NPI index remains the same
- Most table space attributes carries over with exception on the PBG only attributes (such as MAXPARTITIONS)
- Convert existing partitioned (PI, DPSI) indexes to non-partitioned indexes with default PIECESIZE 4GB
 - Other aspects are unchanged



- Redefines the partition limit key for each partition
 - Can change all the limit key value in one ALTER statement
- NPI index remains the same
- Most table space attributes carry over
- Note: If PI or DPSI is needed, it needs to be recreated
- One REORG materialized all the pending changes

Function level : **V13R1M507** and application compatibility level: **V13R1M507**



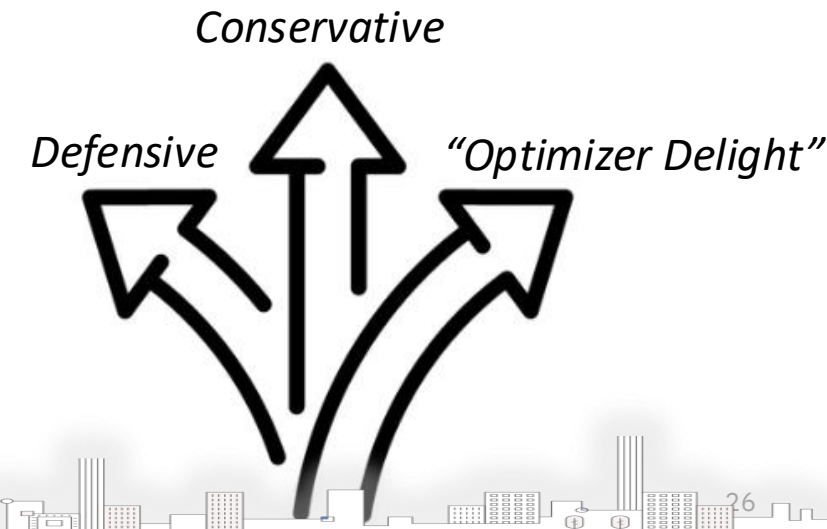
Summary of table space conversion within UTS

Table space type	Pending ALTER DDL	Target converted table space type
Partitioned by Range (absolute page numbering)	<code>ALTER TABLESPACE ...<u>PAGENUM RELATIVE</u></code>	Partitioned-by-Range relative page numbering (PBR RPN)
Partitioned by growth (PBG)	<code>ALTER TABLE ... <u>ALTER PARTITIONING TO PARTITION BY RANGE</u></code>	PBR RPN (V13R1M500)
Partitioned by Range (PBR RPN or PBR APN)	<code>ALTER TABLE ... <u>ALTER PARTITIONING TO PARTITION BY GROWTH</u></code>	Partitioned-by-growth (PBG)



Db2 REBIND

- When are Db2 REBINDS are necessary ?
 - After successfully migrating to a new Db2 release and running smoothly, progressively REBIND high used packages
 - Re-enable fast column processing
 - Avoid performance overhead of “puffing” code
 - Pickup latest runtime performance enhancements
 - Pickup latest maintenance to address issues previously seeded
 - Package invalidation following an online REORG to materialize an online schema change
 - NON-UTS to UTS conversions
 - UTS PBG to UTS PBR conversions
 - Exploit RELEASE(DEALLOCATE) optimizations
 - CICS-Db2 Protected threads
 - HP-DBATs
 - Elevate APPLCOMPAT level
 - After applying a Db2 preventative maintenance package
- *What is your appetite for Db2 access path change?*
 - Defensive
 - Adverse to change
 - Conservative
 - “Optimizer Delight”
 - Allow optimizer to choose at each REBIND



REBIND Options

- REBIND options
 - APREUSE (ERROR|WARN|NONE)
 - ERROR
 - ✓ Db2 tries to reuse the previous access paths for SQL statements in the package
 - ✓ Will guarantee the same access path or REBIND will fail
 - ✓ Db2 indicates the number of statements that cannot be reused in any package in a message
 - WARN
 - ✓ Db2 tries to reuse the previous access paths for SQL statements in the package
 - ✓ Successful with no warnings if same access path is available
 - ✓ If same access path is not available, optimizer will choose new access path (evaluated in previous step) and will be successful with warnings
 - NONE
 - ✓ Db2 does not try to reuse previous access paths for statements in the package



REBIND Options ...



- REBIND options ...
 - APCOMPARE (ERROR|WARN|NONE)
 - ERROR
 - ✓ Optimal access path will be selected (no guarantee the same access path will be selected)
 - ✓ If access path is structurally dissimilar when compare previous access path to current
 - REBIND will fail
 - WARN
 - ✓ Optimal access path will be selected (no guarantee the same access path will be selected)
 - ✓ If access path is structurally dissimilar compare previous access path to current
 - ✓ REBIND will be successful with warnings
 - Dissimilar SQL statements will be reported
 - NONE
 - ✓ Db2 does not try to reuse previous access paths for statements in the package
 - APREUSE = APCOMPARE = NONE
 - “Optimizer Delight”
 - Allow the optimizer to choose appropriate access path at each REBIND
 - Strongly recommend using Extended Plan Management
 - Revert a package to use previously saved access paths



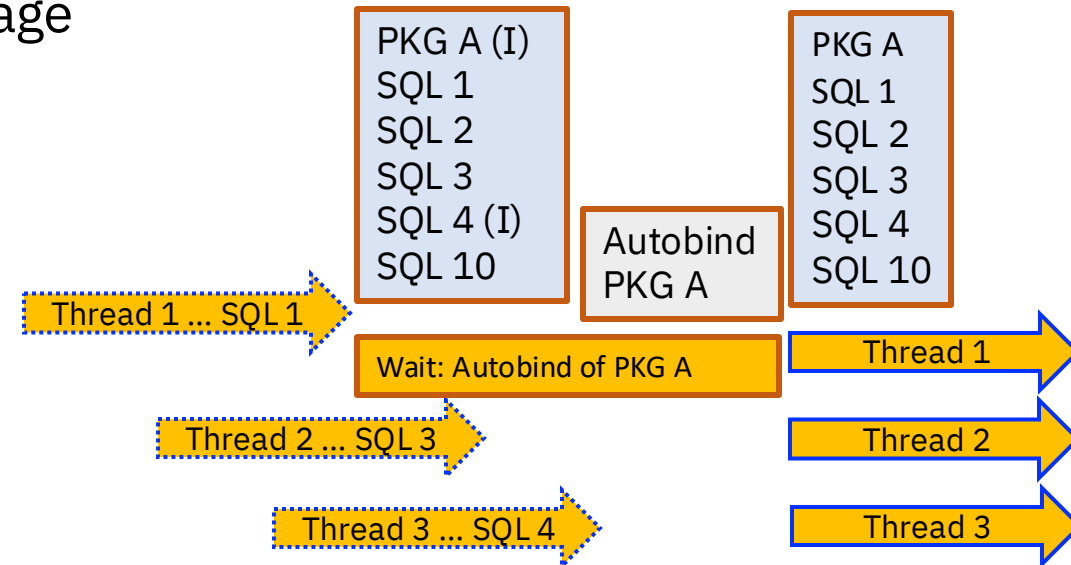
REBIND Options ...

- To minimize risk when performing a REBIND needed for non-access path operational improvements or when a Db2 package is invalidated
 - APREUSE(ERROR) should be the installation default
 - Defensive option to re-establish previous access path
 - Must apply APAR PH63063 (Db2 13)
 - APREUSE fails when attempting to reuse a Db2 12 access path that involves a view or table expression
 - Addressing failed APREUSE REBINDS
 - ✓ REBIND with APCOMARE(ERROR) EXPLAIN ONLY
 - Db2 13 APAR PH61970 supplies “Phase-In” support for REBIND package EXPLAIN ONLY
 - Will identify access path change
 - Evaluate access path differences
 - ✓ REBIND APREUSE(WARN)
 - Successful with no warnings if same access path is available
 - If same access path is not available, optimizer will choose new access path (evaluated in previous step) and will be successful with warnings
 - If successful REBIND using APREUSE (ERROR or WARN)
 - Develop a process to evaluate potential access path improvements
 - ✓ REBIND APCOMPARE(WARN) EXPLAIN ONLY
 - ✓ Perform analysis on any access path changes identified by the Db2 Optimizer
 - ✓ REBIND with APCOMPARE(WARN)
 - Optimal access path will be selected (no guarantee the same access path will be selected)



Package-Level Invalidation

- Package Invalidation
 - One or more SQL invalidations will result in the package being invalidated
 - Example
 - Package contains 10 SQL statements
 - One SQL invalidation will result in package (all 10 SQL statements) being marked invalid
 - Next request to execute will result in triggering a package autobind
 - Prior to V13R1M503
 - Package invalidation and subsequent autobind can significantly impact workload (application unavailability)
 - Execution must wait until autobind completes
 - Other executions must wait until initial autobind completes
 - Autobind failure (infrequent)
 - Package is marked inoperative
 - Explicit rebind is necessary



Statement-Level Invalidation

- Reducing the impact of an AUTOBIND operation ...
 - Previous behavior
 - Db2 tracks application dependencies at package level
 - An operation on any object requiring invalidation results in the entire package marked as invalid; even when only a subset of SQL statements in that package needs to be invalidated
 - This is broad and limits Db2 flexibility to enhance and improve invalidation processing
 - Db2 13 behavior
 - Provide more granular dependency & validity tracking infrastructure, laying foundation for enhancements such as reduced impact of invalidated packages and improved DDL & static DML concurrency
 - New **DEPLEVEL** BIND/REBIND option determines recording of statement level dependencies in addition to package level dependencies.
 - New system parameter **PACKAGE_DEPENDENCY_LEVEL (SPRMPKGDEPLVL)** sets DEPLEVEL default
 - New catalog tables **SYSPACKSTMTCOPY** and **SYSPACKSTMTDEP**



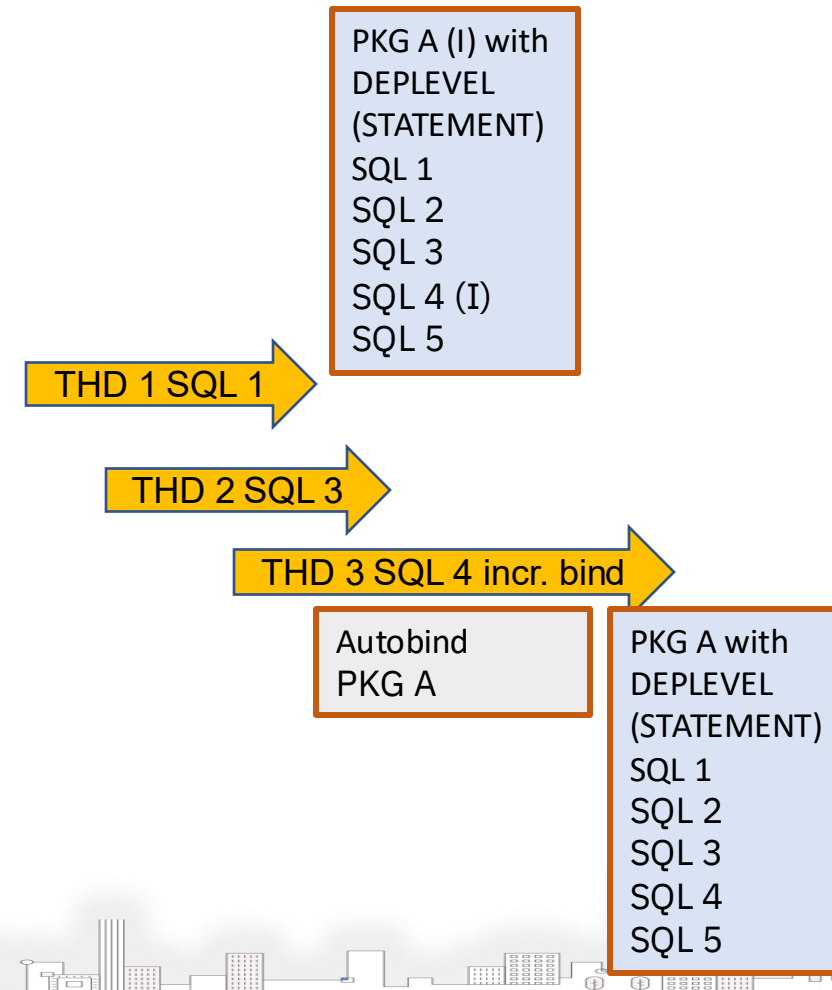
Statement-Level Invalidation ...

- Reducing the impact of an AUTOBIND operation
 - FL500: CATMAINT can be executed to take catalog level to V13R1M501 level
 - One of the new tables in V13R1M501 catalog is SYSPACKSTMTDEP
 - FL502: packages can bound/rebound with new DEPLEVEL(STATEMENT) option
 - That causes statement-level dependencies to be recorded in SYSPACKSTMTDEP
 - FL504: if ALTER causes invalidation of package bound with DEPLEVEL(STATEMENT), next request to execute package still triggers autobind, BUT...
 - Autobind done in background, package can still be executed even before autobind completes, non-invalidated statements execute as usual, invalidated statements incrementally bound when executed
 - When autobind completes, newly-regenerated package phased in (similar to rebind phase-in functionality of Db2 12 FL505)
 - If autobind fails, package gets “advisory rebind” status and can still be executed (non-invalidated statements execute as usual, invalidated statements incrementally bound when executed) – explicit rebind will put package back in valid state

FL 500

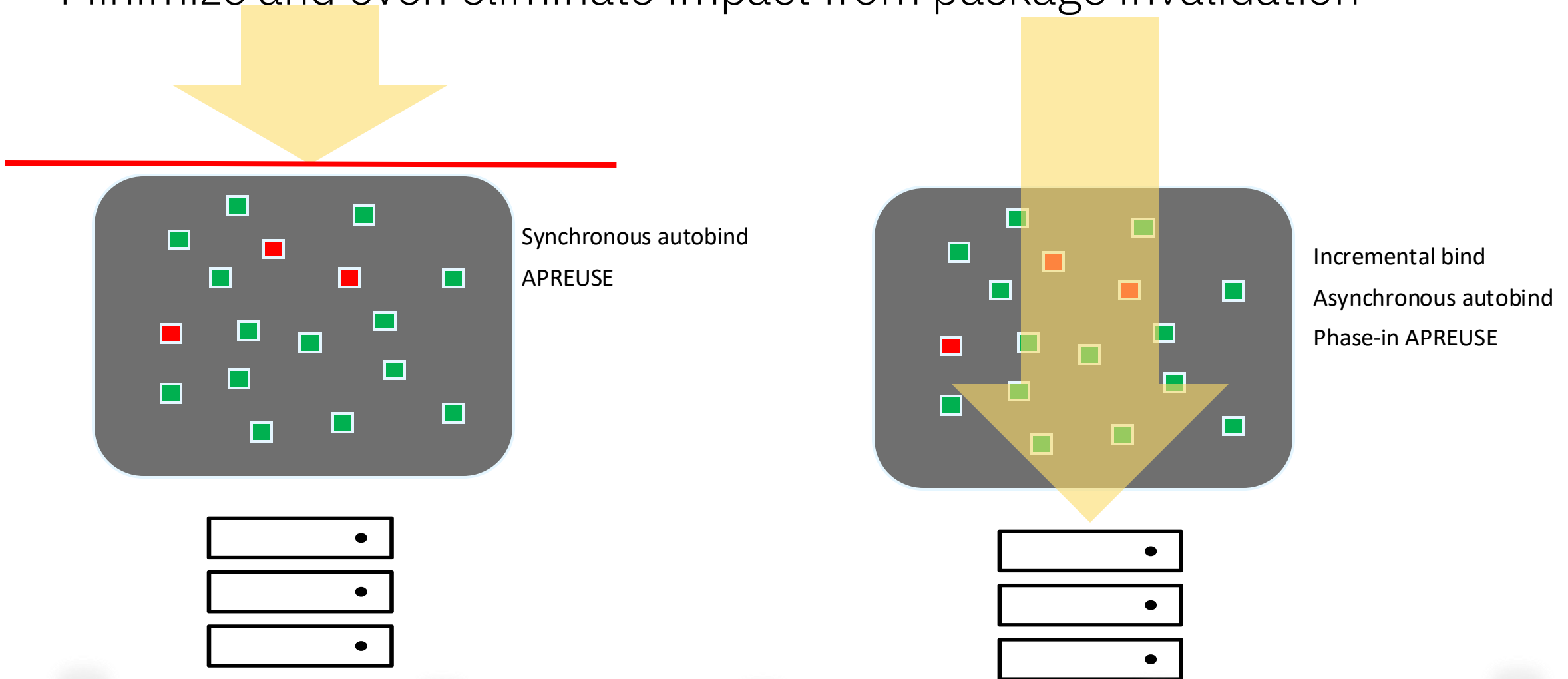
FL 502

FL 504



Statement-Level Invalidation ...

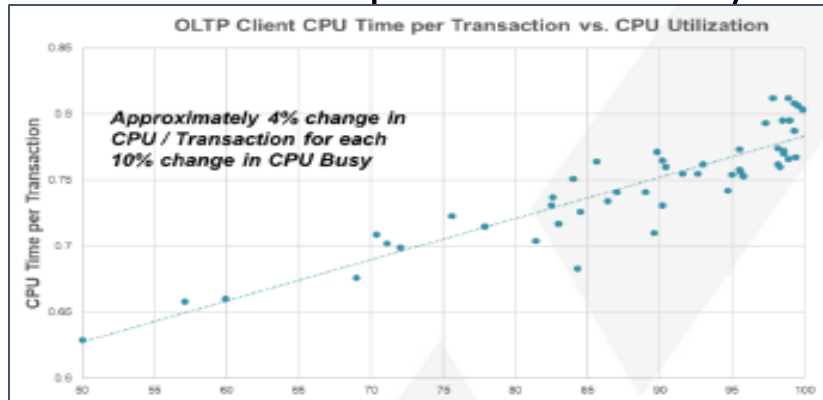
- Minimize and even eliminate impact from package invalidation



Efficiency Considerations

- Understanding available capacity

- MSU Consumption Sensitivity to Utilization – 4-10 Rule of Thumb



- Machines run more efficiently at lower utilizations

- More HW cache per SW work unit
 - Cost per transaction drops

- Magnitude of effects varies by workload and N-way

- Lower N-way have smaller effect
 - Lower RNI workload have smaller effect

- The 4-10 ROT: on average, a 10% change in processor utilization results in a 4% change in cost (CPU time, MIPS, MSUs) per transaction. (3-10 for LOW RNI, and 5-10 for HIGH RNI)

- Processor utilization can be lowered by:

- Running less workload on a constant HW configuration
 - Running a constant workload on a larger HW configuration

- Thus, it is possible for clients to realize a 4% MSU savings (with some variation) by adding enough HW to lower absolute processor utilization by 10%

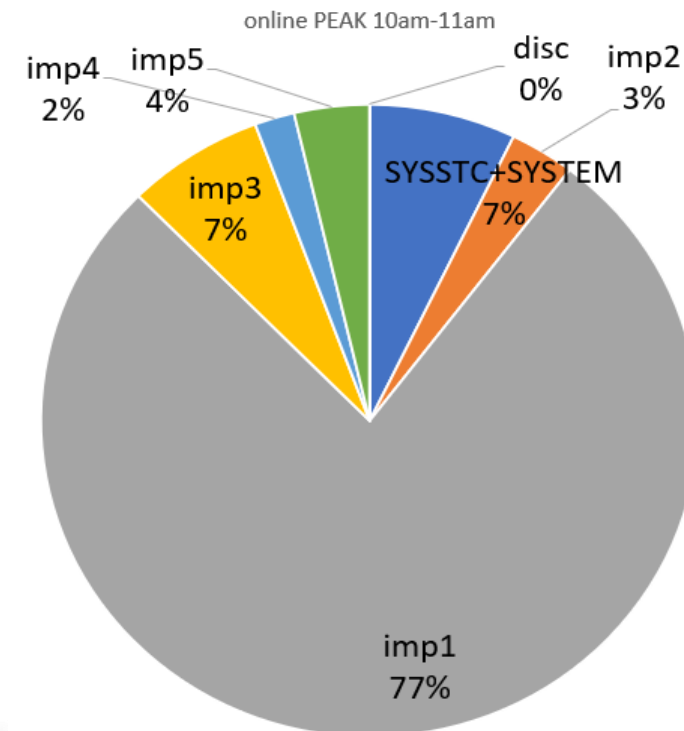
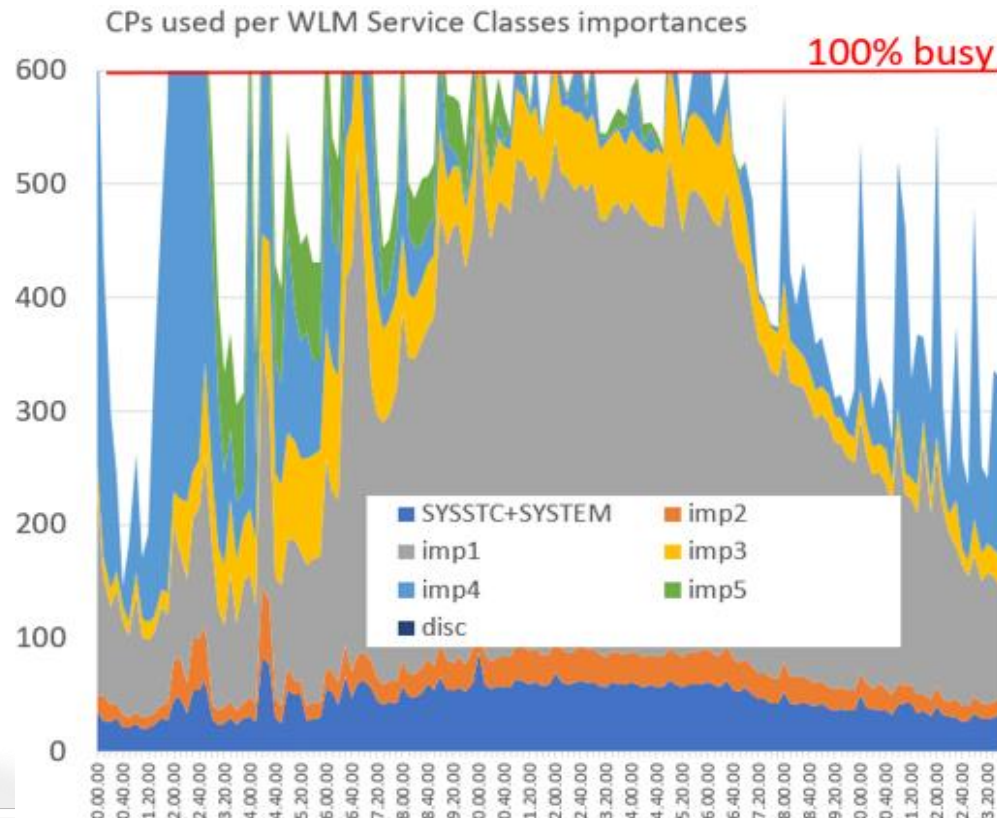
- How much HW would need to be added to lower utilization by 10%?

- If utilization was 100%, would need to add $100/90 = 1.11$ or +11% capacity

- If utilization was 50% would need to add $50/40 = 1.25$ or +25% capacity

Efficiency Considerations ...

- Understanding available capacity ...
 - Additional CPU or designated sacrificial workload is needed for the surviving infrastructure to absorb the workload from that was lost with the failing component(s)
 - Systems are run at very high CPU utilization for elongated periods of time
 - Little or no non-Db2 work to pre-empt when the system becomes 100% busy i.e., no non-Db2 work that can be sacrificed



Recovery Boosts

- Db2 restart after failure ...
 - Exploit Middleware Recovery Boost on z/16+ for faster Db2
 - 6 available boosts a day, 5 minute in duration
 - Set WLM boost attribute to YES for the Db2 “MSTR” address spaces

Summary: z/OS System Recovery Boost										
Boosts are: 1. Speed: subcap can run fullcap 2. zIIP: allowing workload onto zIIPs On by default in IEASYSxx BOOST=SYSTEM	SMP/E FIXCAT IBM.Function.SystemRecoveryBoost									
	z16									
	z15					z16				
	System Recovery Boost			Recovery Process Boost* at MCL P46602.005 for IBM z15 Driver 41C (Bundle S29)				Recovery Process Boost*		
	IPL Startup	Standalone Dump (no zIIP boost)	Shutdown	Sysplex Partitioning - planned or unplanned removal	CF Structure Recovery - rebuild or duplex	Recovery - disconnect or failure from locking resources	CF Data sharing Member Planned/unplanned HyperSwap	SVC Dump	STC start/restart	HyperSwap configuration load
Intended Duration	60 min	60 min	30 min	2 min	1 min	1 min	2 min	2 min	5 min	2 min
Basis of use	Auto	Auto	S IEASDBS	Auto	Auto	Auto	Auto	CHNGDUMP SET, SDUMP, RPBMINSZ threshold	WLM service definition BOOST attribute	Auto
z/OS 2.3	PTF	PTF	PTF	PTF	PTF	PTF	PTF			
z/OS 2.4	PTF	PTF	PTF	PTF	PTF	PTF	PTF	PTF	PTF	PTF
z/OS 2.5	PTF	PTF	PTF	PTF	PTF	PTF	PTF	PTF	PTF	PTF
z/OS 3.1	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES
*Recovery Process Boost limited to 30 min per day per LPAR in aggregate. In V2.4 and higher, can be enabled or disabled with S IEASRB,CLASS=RP,REQ=DISABLE ENABLE										
GDPS provides configuration and orchestration parallelization in V4R2, V4R3, and V4R4.										

Attributes:			
#	Qualifier type	Qualifier name	Boost
1	TN	*MASTER*	NO
1	TN	MSTJCL00	NO
1	TN	CATALOG	NO
1	TN	ALLOCAS	NO
1	TN	CONSOLE	NO
1	TN	ANTMAIN	NO
1	TN	JES2MON	NO
1	TN	JESXCF	NO
1	TN	XCF*	NO
1	TN	WLM	NO
1	TN	SMSPDSE*	NO
1	TN	SMSVSAM	NO
1	TN	RASP	NO
1	TN	TRACE	NO
1	TN	DUMPSRV	NO
1	TN	GRS	NO
1	TN	DB%PDBM1	NO
1	TN	DB%PDIST	NO
1	TN	DB%PMSTR	YES

- Enable “Stage 3” SVC Dump Recovery Process Boost
 - Set RPBMINSZ = 0 (all dumps), xxxxG (by size), large value such as 99999G or NA (disable)
 - Enable at IPL
 - In SYS1.PARMLIB(IEADMCxx)
 - or
 - Dynamically
 - CHNGDUMP SET, SDUMP, RPBMINSZ = 0 or specify size i.e., 20G

– Middleware/”Stage 3” Recovery Boosts can be used at a maximum of 30 minutes per day per LPAR



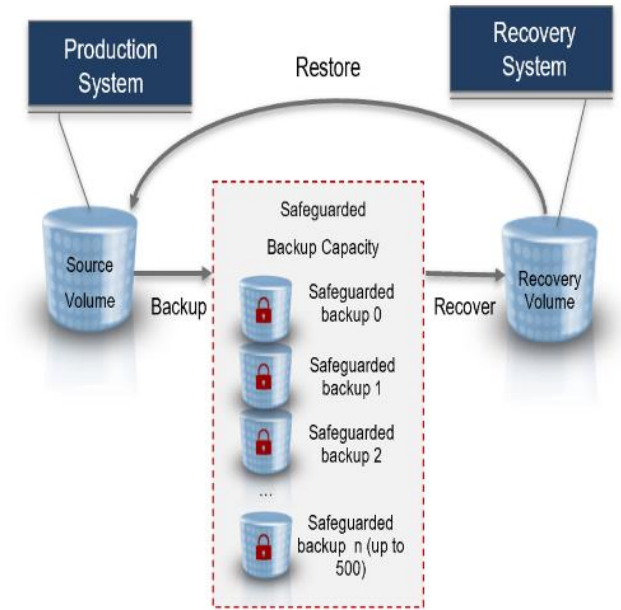
Db2 Log Scalability

- Active log datasets size are customer specific and depends on the amount of log data that is required to be maintained in the Db2 active log databases
 - Dependent on the number of logs, size of each log , logging volume and operational considerations
 - Db2 Supports up to 93 Active Log Datasets
 - Db2 8 increased from 31 to 93 active logs
 - Maximum size of a Db2 active log dataset increased in Db2 12 to 768GB
 - Typical large logs are defined at 48-64gb
 - Larger active log datasets presents additional considerations
 - Elapsed time to archive
 - Reading the archive log
 - Typical requirements
 - Resiliency keep a minimum of 6-hours for Db2 active log wrap around time
 - Data replication is more efficient reading from active log datasets
 - IDAA, QREP, other vendor related log reader products
 - Scheduled or ad-hoc log analysis tools for reporting and diagnostic
 - Recovery performance especially if archive logs reside on virtual tape
- Db2 Archive Log
 - Maximum number of archive logs (MAXARCH) has a maximum limit of 10,000 archive log entries in the Db2 Bootstrap dataset (BSDS)
 - Recommended requirement is for 30-day archive log (recovery asset) retention
 - Root cause analyses, problem diagnoses, advanced data recovery and surgical recovery



Db2 Log Scalability ...

- Reducing the Recovery Point Objective (RPO) in a Cyber Vault solutions introduces a new variables
 - SafeGuarded Copies are executed on some frequency typically every 2-6 hours
 - Frequency is driven by RPO, number of SafeGuarded copies (1,024) and retention period
 - In order to reduce the RPO additional recovery assets must be transferred/copied to a Cyber Vault environment on a more granular schedule
 - Recovery assets
 - Archive Logs
 - Log NO image copies
 - ICF catalog or ICF metadata
 - Can be as frequent as ever 3-4 minutes
- Db2 Active log datasets
 - Resiliency keep a **minimum** of 6-hours for Db2 active log wrap around time
 - 93 active log switching @3:53 minutes
- Db2 Archive Log
 - Archiving every 4 minutes will result in reaching 10,000 archive logs in 27.7 days

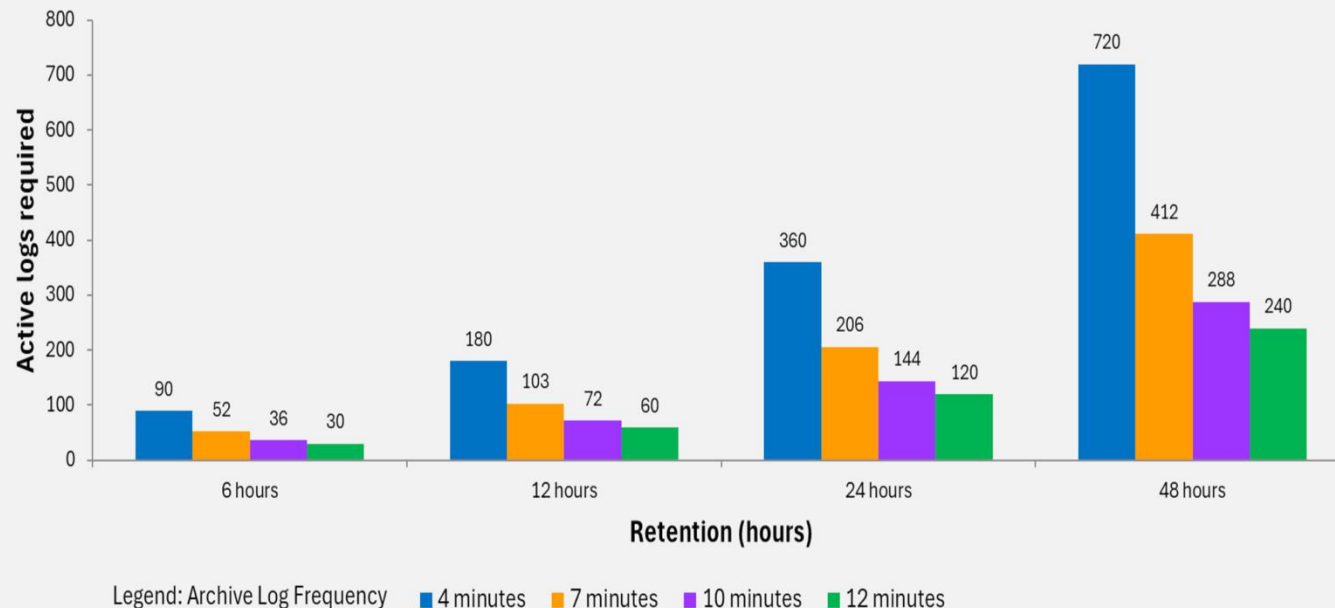


Archive Logs	ICF's
ICF Catalog/Metadata	

Db2 Log Scalability ...

- V13R1M508 (APAR PH67380) increases the available number of active log datasets and the number of archive log dataset in the BSDS

Cyber Resiliency Active Log Requirements



- Active log datasets increased from 93 -- > 1,000
 - Increase number of active logs and potentially decrease the size
 - Will be able to support 24-48 hours of log data in the active log dataset
- Maximum number of archive log datasets has increased from 10,000 -- > 14,500
 - MAXARCH is now hidden ZPARM & deprecated in Db2 13
 - Archive log frequency can go as low as three minutes (2:58) to support 30 days of archive logs in the BSDS

- APAR **PH67380** -- > “DB2 SUPPORT FOR A FUTURE ENHANCEMENT TO DB2 ACTIVE & ARCHIVE LOGS”
 - DSNJ111E OUT OF SPACE IN ACTIVE LOG DATA SETS
 - DSNJ116I ERROR ATTEMPTING TO ADD ARCHIVE ENTRY TO BSDS - REPEATEDLY
- PE fixing APAR **PH68326**

Db2 Log Scalability ...

- V13R1M508 Db2 Boost Strap (BSDS) dataset requirements
 - A BSDS created prior to Db2 11 for z/OS and migrated to Db2 12 or 13 for z/OS contains enough formatted BSDS records to support 1,000 active and 14,500 archive log data sets per copy
 - BSDS created older than Db2 11 and created before V13R1M508 do not contain enough formatted records and will need to be converted using a stand-alone utility
 - DSNJCNVU will increase the number of formatted BSDS records
 - New installs of Db2 13 for z/OS @V13R1M508 supports the new maximum values
 - V13R1M507* or lower, will be support the new limits as long as the BSDS supports the new values
 - BSDS can be converted prior to activating V13R1508 after APARs PH67380, PH68326 have been applied
 - MAXARCH = 14,500 will not increase until activation
- Additional recommended BSDS SMS data class attributes
 - Extended addressability
 - Extended format
 - Extent Constraint Relief (ECR)
 - CA reclaim



Db2 Log Scalability ...

- Pre V13R1M508 validation via DSNJU004

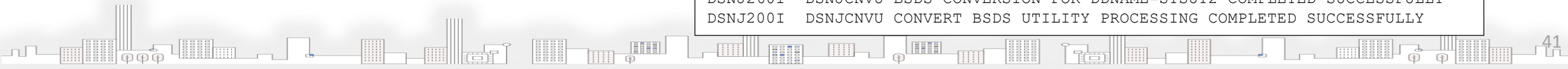
```
*****
*
*          LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'DB2A      ' OF GROUP 'DSNCAT  '.
*
*****
DSNJCNVU CONVERSION PROGRAM HAS RUN   DDNAME=GROUP
LOG MAP OF BSDS DATA SET COPY 1, DSN=DSNC000.DB2A.BSDS01
MAXIMUM NUMBER OF ACTIVE LOGS PER COPY = 93
MAXIMUM NUMBER OF ARCHIVE LOGS PER COPY = 10000
```

```
*****
*
*          LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'DB2A      ' OF GROUP 'DSNCAT  '.
*
*****
DSNJCNVU CONVERSION PROGRAM HAS RUN   DDNAME=GROUP
LOG MAP OF BSDS DATA SET COPY 1, DSN=DSNC000.DB2A.BSDS01
MAXIMUM NUMBER OF ACTIVE LOGS PER COPY = 1000
MAXIMUM NUMBER OF ARCHIVE LOGS PER COPY = 14500
```

- BSDS conversion - DSNJCONVU

```
//DSNJCNVU EXEC PGM=DSNJCNVU
//STEPLIB DD DISP=SHR,DSN=DSND10.SDSNEXIT
//          DD DISP=SHR,DSN=DSND10.SDSNLOAD
//SYSUT1 DD DISP=OLD,DSN=DSNC000.BSDS01
//SYSUT2 DD DISP=OLD,DSN=DSNC000.BSDS02
//SYSPRINT DD SYSOUT=*
```

```
CONVERSION OF BSDS DATA SET - COPY 1, DSN=DSNC000.DB2A.BSDS01.OLD
SYSTEM TIMESTAMP - DATE=2025.205 LTIME= 7:34:50.98
UTILITY TIMESTAMP - DATE=2025.205 LTIME= 7:28:34.90
PREVIOUS HIKEY - 040001E7
NEW HIKEY - 040002F0
RECORDS ADDED - 265
CONVERSION OF BSDS DATA SET - COPY 2, DSN=DSNC000.DB2A.BSDS02.OLD
SYSTEM TIMESTAMP - DATE=2025.205 LTIME= 7:34:50.98
UTILITY TIMESTAMP - DATE=2025.205 LTIME= 7:28:34.90
PREVIOUS HIKEY - 040001E7
NEW HIKEY - 040002F0
RECORDS ADDED - 265
DSNJ260I DSNJCNVU BSDS CONVERSION FOR DDNAME=SYSUT1 COMPLETED SUCCESSFULLY
DSNJ260I DSNJCNVU BSDS CONVERSION FOR DDNAME=SYSUT2 COMPLETED SUCCESSFULLY
DSNJ200I DSNJCNVU CONVERT BSDS UTILITY PROCESSING COMPLETED SUCCESSFULLY
```



Online delete of an active log

- Adding/deleting/removal of a Db2 active log data sets is needed for a variety of reasons
 - Number of active log data sets reaches the maximum limit of 93
 - Existing active log data sets must be deleted before replacing them with larger log data sets
 - Take advantage of Db2 12 support for active log data sets greater than 4 GB
 - Existing active log data sets can be replaced with active log data sets that are greater than 4 GB
 - Leverage z/OS DFSMS data set encryption for log data sets
 - introduced in Db2 11
 - Unencrypted log data sets must be replaced with new encrypted log data sets
 - Encrypted log data sets must be redefined periodically to replace encryption keys with new keys
 - Storage management maintenance
 - DASD subsystem refresh
 - Moving active log data sets to a different storage system



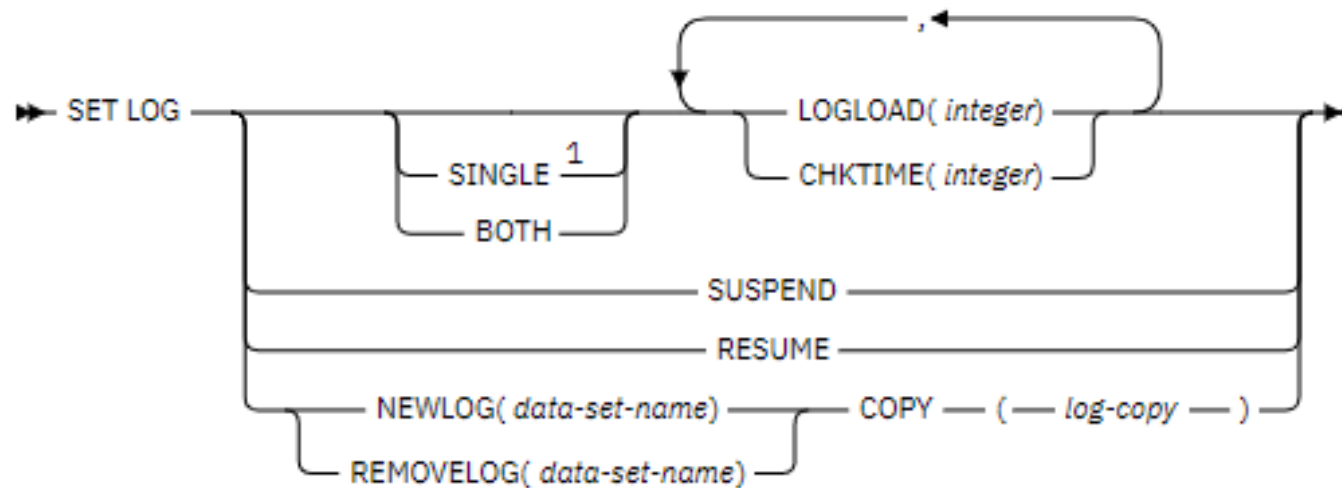
Online delete of an active log ...

- Deletion of active logs from the bootstrap data set (BSDS) prior to Db2 13
 - Db2 12 process/behavior
 - Db2 subsystem needs to be stopped
 - DSNJU003 utility executed
 - Start the Db2 subsystem
 - Disruption impacts the Db2 ability to support continuously available application-processing
- Starting with function level V13R1M500 deletion of an active log data set that is not in use can be an online event
- One of the most customer tested features in Db2 13
 - Requirements to remove active log data set
 - Must exist in specified COPY
 - Must be marked REUSABLE
 - Must not be current active log
 - Must not be next current active log
 - There must be at least 3 REUSABLE log data sets in the specified COPY
 - Subsystem parameter OFFLOAD = YES
 - FL 500 or higher; not supported for FL 100*
 - Data sharing: no peer Db2s have it allocated



Online delete of an active log ...

- **REMOVELOG** option added to –SET LOG command
 - COPY is required
 - Double check for intended data set
- Examples:
 - -DBP1 SET LOG REMOVELOG(DB2P.DBP1.LOGCOPY1.DS03) COPY(1)
 - -DBP1 SET LOG REMOVELOG(DB2P.DBP1.LOGCOPY2.DS03) COPY(2)



Online delete of an active log ...

- -SET LOG REMOVELOG new messages
 - DSNJ391I ... REMOVELOG OPERATION FAILED...
 - Includes data set name, reason
 - DSNJ392I ... *data-set* REMOVED FROM THE ACTIVE LOG INVENTORY
 - DSNJ393I ... *data-set* MARKED AS REMOVAL PENDING IN THE ACTIVE LOG INVENTORY
 - DSNJ394I ... INVALID KEYWORD COMBINATION...
- Other messages modified
 - DSNJU004 adds support for REMOVAL PENDING
- DISPLAY LOG DETAIL
 - Includes new message DSNJ384I

FL 500

FL 100*

```

DSNJ384I  -DB2A
COPY1 LOG DATA SETS: TOTAL=6 NOTREUSABLE=1 REUSABLE=3  STOPPED=0
                        REMOVAL PENDING=2

LOG DATA SET NAME                                REMOVAL PENDING READERS
DSNC000.DB2A.LOGCOPY1.DS03                        21.122 10.00.42         2
DSNC000.DB2A.LOGCOPY1.DS01                        21.123 10.32.56         1

COPY2 LOG DATA SETS: TOTAL=6 NOTREUSABLE=1 REUSABLE=3  STOPPED=0
                        REMOVAL PENDING=2

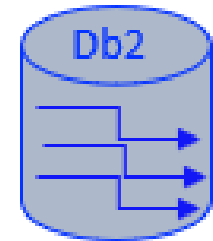
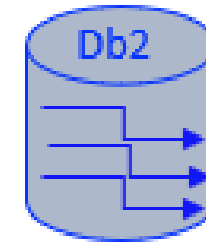
LOG DATA SET NAME                                REMOVAL PENDING READERS
DSNC000.DB2A.LOGCOPY2.DS02                        21.119 09.33.02         0
DSNC000.DB2A.LOGCOPY2.DS03                        21.123 12.32.27         0

```

- Data sharing - D GRS can be used to determine if peer members have the active log data set allocated

Online delete of an active log ...

- “-SET LOG REMOVELOG” three possible outcomes
 1. Log is not in use and is successfully deleted from the BSDS
 2. Log in use and gets flagged with REMOVAL PENDING
 3. If error, failed command, for example:
 - Log is current active log
 - Log status is NOTREUSABLE
- Physical deletion of data set controlled by user
- Active log copy data flagged for REMOVAL PENDING
 - Cannot be unflagged or reversed
 - No longer can be used for new log read requests
 - Db2 will use another active log copy, if it exists
 - Archive log copy data set is used
 - Db2 will not switch to it for log write
 - Monitor status; reissue -SET LOG REMOVELOG
 - REMOVAL PENDING status remains on restart of Db2
 - Can still be deleted using DSNJU003



Questions



Thank You