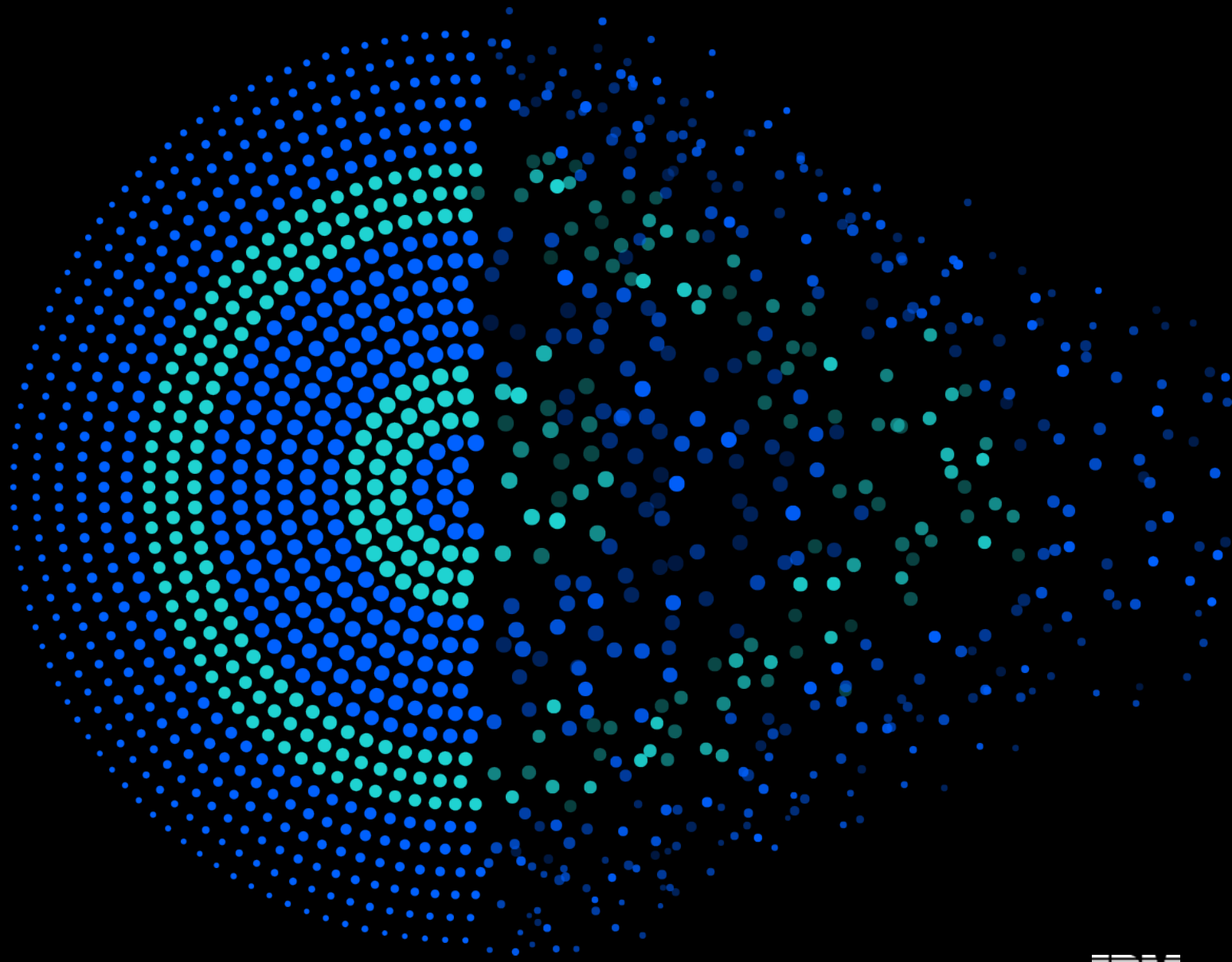
Data and AI

Db2 for z/OS:
Hot Topics and Best Practices with John Campbell – Part 1

John Campbell
Db2 for z/OS Development



IBM

Objectives

- Learn recommended best practice
- Learn from the good, bad and ugly experiences from other installations
- Provide answers to important, frequent or difficult questions

Agenda

- COPY SHRLEVEL CHANGE FLASHCOPY CONSISTENT
- Considerations when migrating from segmented to multi-part PBG UTS
- Dos and Don’ts around Continuous Delivery
- Software Maintenance
- Point-In-Time (PIT) Recovery in Db2 12 with SCOPE UPDATED
- Point-In-Time (PIT) Recovery before Function Level (FL) activation and/or CATMAINT
- Fast Index Traversal (FTB memory)
- Dynamic prefetch avoidance

Agenda …

- Local Buffer Pools and Real Memory allocation
- A single local buffer pool can be spread across different size real memory frames
- XXXL Size Local Buffer Pool
- Restrictions with Buffer Pool Simulation
- Using DSNJU999 service aid
- Using Large Frame Size (LFAREA) for Local Buffer Pools
- Growth in CPU time/transaction as CPU busy increases
- Tutorial on HiperDispatch and Efficiency differences between VH, VM, VLs
- REBIND very old plans and packages to avoid Autobind
- REBIND Phase-in and thread reuse with RELEASE(DEALLOCATE)
- Data Sharing - GBP Asynchronous cross-invalidation (XI)
- Data sharing - GBP deallocation
- Data Sharing – Peer recovery
- Database Housekeeping Rules
- Turning off CICS-Db2 thread protection

COPY SHRLEVEL CHANGE FLASHCOPY CONSISTENT

- Creates a FlashCopy Image Copy (FCIC) using FlashCopy/fast replication
  - FCIC is a VSAM Linear Data Set (LDS) – not a sequential data set
  - The end result is that the VSAM FCIC is not fuzzy, because FLASHCOPY CONSISTENT was specified
  - If FLASHCOPY YES was specified, then it would create a “fuzzy” VSAM FCIC
- Object is fully available throughout in UTRW (utility-in-progress R/W access allowed) status
- After the VSAM FCIC is created, control is returned to COPY and the VSAM FCIC will be opened as a Db2 shadow data set for these three log phases to occur
  - LOGAPPLY
    - Apply redo log records to the VSAM FCIC (since it was created from disk and not via Db2 buffer pools) up to the point of consistency
  - LOGCSR
    - Determines if there are any in-flight, in-abort, in-doubt, etc. URs
  - LOGUNDO
    - Will backout any uncommitted work from the VSAM FCIC
    - Only performed if LOGCSR determines there is uncommitted work to backout

COPY SHRLEVEL CHANGE FLASHCOPY CONSISTENT …

- If data set level FlashCopy is not available and FLASHCOPY CONSISTENT is specified, the output is still a VSAM FCIC, *not* a sequential image copy
  - COPY will now invoke DFSMSdss to create the VSAM FCIC using standard I/O
  - Object is still fully available throughout in UTRW status
  - DFSMSdss returns control to COPY and the LOGAPPLY, LOGCSR, and LOGUNDO phases are performed to make the VSAM FCIC consistent
- Remember if you do not have data set level FlashCopy available
  - Must ensure that zparm COPY_FASTREPLICATION is set to PREFERRED

COPY SHRLEVEL CHANGE FLASHCOPY CONSISTENT …

- Please note the you could request both one FCIC and multiple sequential image copies during COPY
  - In this example, a VSAM FCIC with consistency is requested with two sequential image copies, a local primary and a local secondary backup

```
COPY LIST COPY_LIST1
  FULL YES
  COPYDDN(SEQCOPY1,SEQCOPY2)                  -> sequential image copies
  FLASHCOPY CONSISTENT FCCOPYDDN(FCCOPY)      -> 1 FCIC
  SHRLEVEL CHANGE
  PARALLEL
```

  - COPY will create the VSAM FCIC with consistency first, in the COPY phase
  - In the subsequent SEQCOPY phase, the two sequential copies will be created from the VSAM FCIC with consistency
  - So all 3 image copies created will be consistent
- Only one FCIC can be requested for each object in a COPY invocation
- Up to four sequential image copies can be requested for each object in a COPY invocation

COPY SHRLEVEL CHANGE FLASHCOPY CONSISTENT …

- There may be more work to do during recovery if RECOVER uses a “FCIC with consistency” as the base
  - Two additional log phases can occur during RECOVER from FCICs
    - During PRELOGC, RECOVER determines which URs were backed out when the FCIC with consistency was created
    - PRELOGA will redo/apply just those log records for the URs that were backed out up until the point of consistency for the FCIC
  - After that, the usual log apply phases are executed: LOGAPPLY, LOGCSR, LOGUNDO

Considerations when migrating from segmented to multi-part PBG UTS

- PBG UTS has no table level lock as with classic segmented tablespace type
- PBG UTS has a partition lock structure with selective partition locking
  - Row/Page lock -> partition lock -> tablespace lock
- SQL LOCK TABLE on a multi-part PBG UTS
  - Db2 will issue exclusive partition level lock on all parts immediately
  - SQL query with ISO(CS) CD(N) and with effective lock avoidance will not be blocked by the SQL LOCK TABLE and so can return data
- Lock escalation on a multi-part PBG UTS
  - Db2 escalates to partition level lock, but only for partition(s) that the unit of work UR actually touches
  - This is different behaviour relative to classic segmented table space
  - This behaviour for PBG UTS is the same as the existing behaviour for classic partitioned table space and PBR UTS

Considerations when migrating from segmented to PBG UTS …

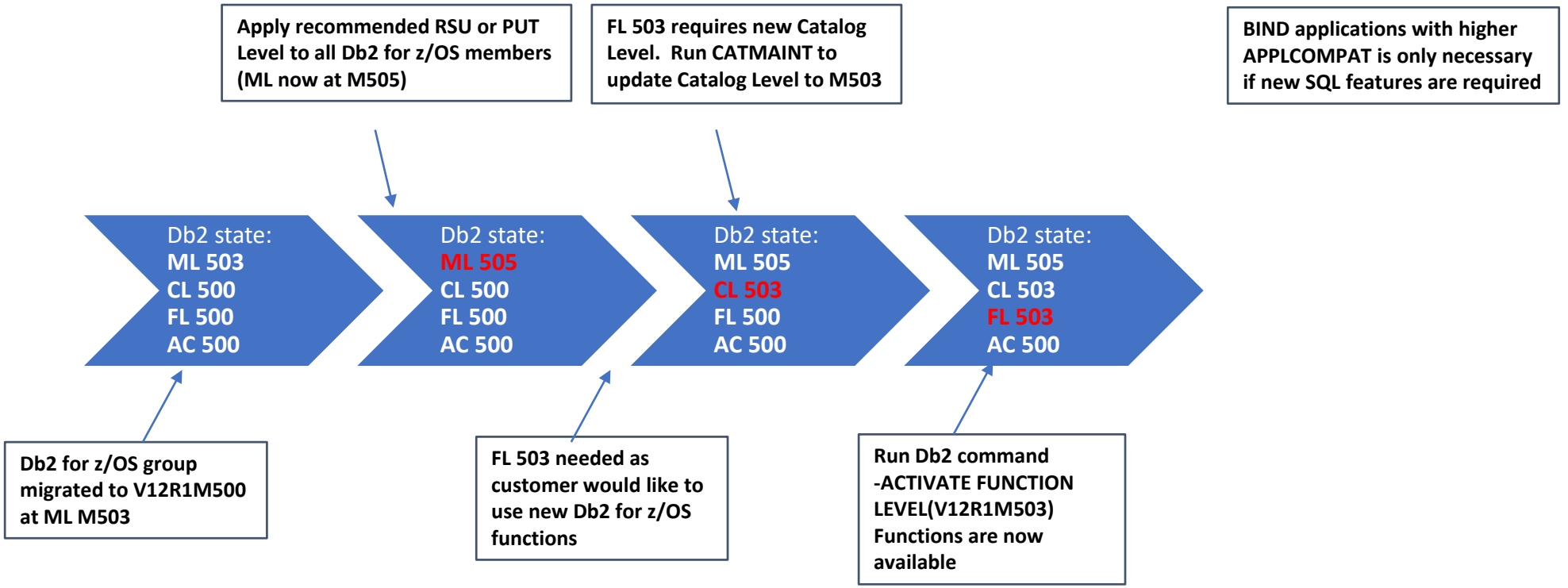
- Timeout or not on insert into PBG UTS when lock escalation occurs
  - Multi-part
    - Db2 will try all possible partitions
    - If lock escalation has occurred on one partition, then Db2 will go to the next partition
    - If Db2 has exhausted all partitions, Db2 will fail the insert immediately with SQL code -904 and reason code 00C9009C
    - If it is PBR UTS, insert will wait behind the lock escalation and eventually timeout with reason code 00C9008E
  - Single part
    - Insert will wait behind the lock escalation
    - No new partition will be added
    - Insert will eventually timeout with reason code 00C9008E

Continuous Delivery - Recap

- With Continuous Delivery, there is a single delivery mechanism for defect fixes and enhancements
  - PTFs (and collections of PTFs like PUTLEVEL and RSU) → same as today
- With Continuous Delivery, there are four Db2 for z/OS levels to consider
  - Maintenance level (ML) = increased by applying maintenance
    - Also known as code level - contains defect and new enhancement fixes
    - Most new functions are shipped disabled until the appropriate new function level is activated
  - Catalog level (CL) – vehicle to enable new FL - accumulative (skip level possible)
    - Db2 Catalog changes that are needed for some FLs
  - Function level (FL) – needs to be activated - accumulative (skip level possible)
    - Introduces new Db2 for z/OS features and functionality
    - No impact or change in existing application behavior which are protected by APPLCOMPAT level on BIND/REBIND
  - APPLCOMPAT level (AC) – set by application - provides an “island of stability” for a given application
    - Determines SQL function level of applications – can increase FL of the application (and fallback)
    - AC on BIND must be advanced to exploit new SQL function
    - AC level in BIND/REBIND of package must be <= FL and rules over FL
    - Freezes new SQL syntax even if FL is later moved back to earlier level

Continuous Delivery – Recap …

- Example of how to get to a new function level



Dos and Don’ts around Continuous Delivery

- Major misconception
  - When progressively applying preventative service, which raises Maintenance Level (ML/code level) of Db2, then it must be complimented by activating the corresponding Function Level (FL) to match the ML i.e. to activate the new maintenance
  - In reality, new maintenance is active once applied
- Risks:
  - If the FL is increased after introducing a new preventative service level and the maintenance proves to be unstable, you cannot backout to a previous maintenance service level which has a ML that is less than the respective FL
  - Must fix forward!
- Recommendations:
  - Run on a new preventative service level in production for a minimum of 3 months before raising the FL or running a CATMAINT in support of a new FL
  - Stabilize the zParm APPLCOMPAT setting at V12R1M500
  - Stabilize the APPLCOMPAT setting of the Db2 Connect packages in the NULLID collection at V12R1M500

Dos and Don’ts around Continuous Delivery …

- More recommendations
  - APPLCOMPAT should be explicitly specified on every BIND ADD and BIND REPLACE
    - No BIND should rely on picking up the default value from zParm APPLCOMPAT
    - APPLCOMPAT is a “sticky” option on REBIND
    - zparm APPLCOMPAT should be set conservatively and down-level at V12R1M500
  - SQLLEVEL setting in DSNHDECP should match the current Db2 Function Level
  - Db2 Connect packages in the NULLID collection should remain with APPLCOMPAT=V12R1M500
  - For distributed applications looking to exploit new SQL features, the Db2 Connect packages should be bound into a new collection specific to the Function Level and with the corresponding APPLCOMPAT equal to the Function Level
    - Applications must be directed to the proper collection using the currentPackagePath property in the driver or by setting this special register using a System Monitor profile

Dos and Don’ts around Continuous Delivery …

- More recommendations …
  - The advancement of the Function Level of the data sharing group will be driven by one of the following
    - User requirement for new functionality tied to a specific Function Level
    - Maintaining currency aka “evergreening” e.g. every 2-3 years
  - Reminder: the Function Level of the data sharing group should not be increased until
    - Latest preventative service level has been proven to be running stable in production for at least 3 months
    - IFCID 376 has been used to analyze application incompatibilities and have them addressed
    - Investigation with vendors is complete, and the implementation is complete as to
      - What levels/patches are required to support/exploit the new Function Level
      - What the recommended APPLCOMPAT level should be for the BIND of the respective packages
  - Prior to exploiting any new Function Level beyond V12R1M500, the Db2 Connect infrastructure and drivers must be updated to the following minimum levels:
    - Driver level V3.72 or 4.22
    - Db2 Connect Server level V11.1 M1 FP1

Dos and Don’ts around Continuous Delivery …

- More recommendations …
  - Function Level V12R1M504
    - After advancing the Function Level to V12R1M504 or later and running SQL DDL under a package having an APPLCOMPAT of V12R1M504 or later, then certain objects types are deprecated and any SQL DDL against these existing objects will fail e.g.:
      - Classic simple and segmented table spaces
      - Classic partitioned table spaces
      - Index controlled partitioned table spaces
      - Hash access table spaces
      - etc.
    - In order to run SQL DDL against these objects you must either downgrade the APPLCOMPAT of the associated package or preferably downgrade the APPLCOMPAT level by setting the special register (SET CURRENT APPLICATION COMPATIBLITY)
    - Reach out to all your tool vendors to identify which releases/patches are required to support Function Level V12R1M504 and will toggle the special register automatically

Dos and Don’ts around Continuous Delivery …

- Function Levels and Quiesced Db2 members of Data Sharing Group
  - Quiesced member = inactive member
  - A migrating member, or one that is changing the Catalog Level (CL) or Function Level (FL) will only check the maintenance levels of the active members in a Db2 data sharing group when determining if something can be done or not
    - If a quiesced member has a down level Maintenance Level (ML) it will not be checked
  - Potential issues will then arise when restarting the quiesced member
    - ML of quiesced member does not support the current CL or FL
    - Subject Db2 member will not be allowed to start
  - Db2 member is essentially orphaned until it is restarted with a library that has a ML that supports both the current CL and FL

Software Maintenance - Preventative Service Planning

- Problem statement
  - Many installations where preventative service level is only upgraded once or twice per year
  - Production level can be missing up to 10-16 months of HIPER maintenance and up to 16-21 months of non-HIPER maintenance
  - Otherwise apply corrective service and associated prerequisites after experiencing a failure
  - Most serious problems are only uncovered in production
- Risks:
  - Experience high impact problems with Db2 12 and the new functions (not all of them are FL dependent)
  - High risk of dragging in a long prerequisite chain of PTFs which adds to the risk of applying a corrective PTF
  - Significant amount of or critical blocked maintained that cannot be applied because of prevailing PEs

Software Maintenance - Preventative Service Planning …

- Recommendations:
  - Need to evaluate your maintenance strategy for Db2 12 and the new Continuous Delivery concept
  - Upgrading preventative service level once or twice a year is most definitely not enough!
  - It is essential to stay very current on preventative maintenance with Db2 12 and when exploiting the new functions (not all of them are FL dependent)
  - Reduce the gap on missing HIPER maintenance by applying preventative service 3-4 times a year based on the latest quarterly RSU or latest quarterly RSU +1 month
  - This should be a continuous program, pre-planned, and scheduled on the calendar

Software Maintenance – Enhanced HOLDDATA

- Problem statement
  - Many installations making very limited use of Enhanced HOLDDATA to identify new HIPERs which have not been applied, and PTFs which have been applied into the package and which are now in error
    - After building a new maintenance package before reaching production
    - After post-production cut-over of a new maintenance package
- Risks:
  - Not aware of operational exposures incurred due to missing HIPERs and PTFs applied which are now in error – the duration of an exposure could be up to 10-21 months

Software Maintenance – Enhanced HOLDDATA …

- Recommendations:
  - Pull Enhanced HOLDDATA on a weekly basis and analyze the new operational exposures
    - Some problems may not be applicable to your use of Db2
    - Some problems may have a practical operational bypass
  - Expedite the fixing PTF and prerequisites where there is a significant exposure and no practical operational bypass is available for the problem
    - Prove the stability of the new maintenance in pre-production for 1-2 weeks
    - Perform targeted testing to make sure the maintenance addresses the problem
  - Need a decision-making process around how to handle PEs which are blocking the build of a new maintenance package and open APARs on the pre-requisite chain
    - Perform an assessment as to the volume of blocked maintenance and/or potential impact of individual vicious problems marked HIPER
    - If the operational exposure is significant
      - Look for and apply practical operational bypasses should they exist
      - Be prepared to wait for and apply the official PTF fix, or take a tested APAR fix earlier to provide relief and make progress
      - Forward fit the maintenance that was previously blocked

Point-In-Time (PIT) Recovery in Db2 12 with SCOPE UPDATED

- New Db2 12 parameter SCOPE for RECOVER utility
  - Goal is to speed up the elapsed time for PIT recovery (TORBA/TOLOGPOINT)
  - SCOPE UPDATED is the default
    - Objects are excluded from recovery that have not changed since the given recovery point
      - Avoids wasting time restoring the mage copy for a given object
    - Potentially a great performance optimization
    - Might be known to a few customers by running RECOVER under DIAGNOSE(607) for Db2 11
  - As before SCOPE ALL enforces the recovery of any object even if it has not been changed
  - How does Db2 know if the objects have been changed or not after the recovery point?
    - By reading the entries in SYSLGRNX

Point-In-Time (PIT) Recovery in Db2 12 with SCOPE UPDATED …

- Problem statement
  - If a PIT recovery of SYSLGRNX is performed then can run into a big issue when performing a PIT recovery for application objects after the PIT recovery of SYSLGRNX
    - Scenario
      - No data changes have been done to the application object between the time of COPY and the recovery point of SYSLGRNX
      - But data changes are done later after the the recovery point of SYSLGRNX
    - Db2 will not detect that situation because the entries of SYSLGRNX have been eliminated by the PIT recovery of SYSLGRNX
      - Db2 will incorrectly exclude the recovery of the application object which will lead to data inconsistencies
      - REPORT RECOVERY utility also reads SYSLGRNX and will not report the changes
      - The RECOVER job ends with RC04 and DSNU1322I message for each excluded object
        - DSNU1322I =D2LC 318 12:10:14.52 DSNUCALX - PROCESSING SKIPPED FOR TABLESPACE DSNDB06.SYSTSLVH BECAUSE THE OBJECT DOES NOT NEED TO BE RECOVERED

Point-In-Time (PIT) Recovery in Db2 12 with SCOPE UPDATED …

- Recommendations
  - Apply PTF for APAR PH20056, the plan is not to change the default from SCOPE UPDATED to SCOPE ALL.
    - RECOVER will internally change SCOPE UPDATED to ALL for PIT recovery of Catalog/Directory objects
    - RECOVER will internally change SCOPE UPDATED to ALL for any object after SYSLGRNX has been recovered to a prior point in time
      - DSNU124I message will be issued to indicate the override
      - RC will still be 0
    - No plan to change the default from SCOPE UPDATED to SCOPE ALL
  - Until PTF applied for APAR PH20056 applied, explicitly specify SCOPE ALL on PIT recoveries
    - Any pre-existing “canned” job to recover the Catalog/Directory should be modified

Point-In-Time (PIT) Recovery before FL activation and/or CATMAINT

- Series of steps to provide a solution
  1. Determine the point you want to recover to: Call it LRSN x
  2. Reset internal DSNDB08 DBET information using DIAGNOSE TYPE(0) REPAIR SET DBID(8) PSID(0000) RESET
  3. Shut down the Db2 data sharing group and delete all CF structures related to Db2
  4. If x is prior to a CATMAINT or FL migration, then restore the BSDS pair from the most recent copy prior to x, update archive/active log information in BSDS pair as necessary to include logs up to x using DSNJU003
  5. Create CRCR with ENDLRSN=
  6. Restart one Db2 member in MAINT mode with DEFER ALL
  7. Recover Catalog/Directory to current, and rebuild associated indexes
  8. Recover application objects to current, and rebuild associated indexes
  9. Recycle Db2 member and bring up other Db2 members of data sharing group
- Note: apply PTF for APAR PH19361 (open) where underlying VSAM datasets of Catalog and Directory objects are deleted as planned
  - PIT recovery for SYSUTILX failed because RC0C90094 received on DSNDB01X, and subsequent recovery of other Catalog and Directory objects could not continue

Fast Index Traversal (FTB memory)

- Recap
  - One of the most important performance features in Db2 12
  - In memory index performance optimization for random (index traversal) pattern
  - Used for fast index lookup by avoiding expensive index B-tree traversal
  - Improved performance as Fast Traverse Blocks are L2 cache aware B-Tree like structure
    - Each page is equal to one cache line in size (256 bytes)
  - Customers have seen very good CPU performance improvements

Fast Index Traversal (FTB memory) …

- A number of customers have encountered problems related to the use of FTBs
  - Robustness issues: INCORROUT, concurrency, latch contention, memory leak, serviceability, etc.
  - Many customers had to disable this feature pending improvements in this area
    - Set zparm INDEX_MEMORY_CONTROL = DISABLE
  - Db2 Development has undertaken extensive analysis to improve the quality of FTBs
    - Improved serviceability
    - Improved internal test coverage
    - Resolution of all known external and internal software defects
- When using FTBs or before re-enabling use of FTBs must apply the following maintenance and remain vigilant for new corrective maintenance
  - APAR PH16429 correction in the cancel window during index structure modification
  - APAR PH19484 correction in handling include column
  - APARs PH21916, PH24667, PH25240 correction in FTB p-lock handling at various index operations
  - APAR PH25801 correction in the timing window between mass-delete and FTB creation
  - APAR PH26109 correction in the error handling of SYSIBM.SYSINDEXCONTROL entries

Dynamic Prefetch Avoidance

- Goal in Db2 12 to reduce CPU resource consumption by avoiding scheduling a dynamic prefetch engine when sequential detection triggers dynamic prefetch and where all pages are in the buffer pool
- Benefits
  - Reduces the risk of running out of prefetch engines
  - Reduces DBM1 SRB CPU time
  - Reduces LC24 latch contention
- Solution
  - Dynamic prefetch now uses buffer pool statistics - # dynamic prefetch I/Os and # sequential sync I/Os
  - When about to schedule dynamic prefetch requests
    - Db2 maintains a counter about # of times dynamic prefetch I/O actually occurred
    - Db2 checks whether the dynamic prefetch I/O count has not changed
    - If this happens 3 times, dynamic prefetch scheduling is disabled and starts recording # sequential sync I/Os for the buffer pool
    - Once dynamic prefetch scheduling is disabled, but the # of sequential sync I/Os has changed, then scheduling of dynamic prefetch is re-enabled
  - Tracked “per object” within the scope of a single SQL statement
    - If a table is referenced multiple times in the same statement (e.g., correlated subquery and outer select), each is tracked individually

Summary

- Learn recommended best practice
- Learn from the good, bad and ugly experiences from other installations
- Provide answers to important, frequent or difficult questions

Top DB2z Social Media Channels

#DB2z

- Join the World of DB2 www.worldofdb2.com
- Follow @IBMDB2 on Twitter https://twitter.com/IBMDB2
- Join DB2z LinkedIn Group
- https://www.youtube.com/user/IBMDB2forzOS
- DB2z on Facebook
  - https://www.facebook.com/IBMDB2forzOS/

Now … Live Q&A with John Campbell