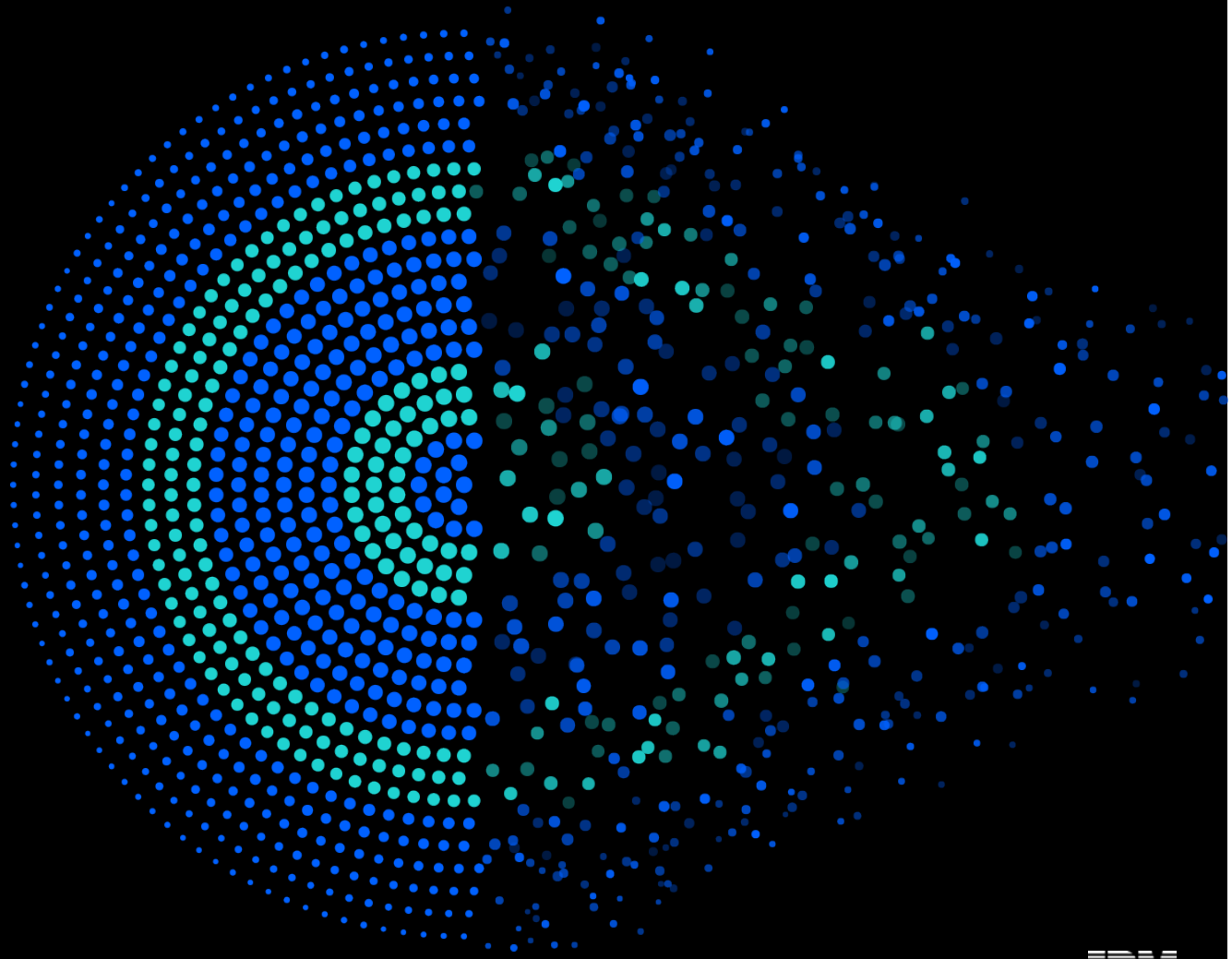
Db2 12 for z/OS Migration Planning and Experiences Part 2

John Campbell
Distinguished Engineer
Db2 for z/OS Development
Email: campbelj@uk.ibm.com



IBM Data and AI

IBM

Objectives

- Share lessons learned, surprises, pitfalls
- Provide hints and tips
- Address some myths
- Provide additional planning information
- Provide usage guidelines and positioning on new enhancements
- Help customers migrate as fast as possible, but safely

Agenda

- Part 1
  - Db2 11 for z/OS prerequisites for migration to Db2 12 for z/OS
  - Db2 12 for z/OS Migration – Quick Hits
  - Maintenance recommendations for early adopters of Db2 12 for z/OS
  - Db2 12 for z/OS Risk Mitigation
  - Understand Continuous Delivery starting with Db2 12 for z/OS
  - Understanding new function levels
  - Db2 12 for z/OS Greatest Hits
  - Fast Un-clustered INSERT
  - RTS enhancements
  - JC Recipe for successful migration
- Summary

Agenda ...

- Part 2
  - Db2 12 for z/OS Greatest Hits
  - Fast Un-clustered INSERT
  - RTS enhancements
  - Fast Index Traversal
  - Data dependent vs. numeric based pagination syntax
  - Increase in log record size after converting BSDS in Db2 11 and entry to Db2 12
  - Dynamic Plan Stability
  - More granular global commit LSN and global read LSN
  - SQLCODE -109 Issue
  - Enhanced SQL MERGE
  - UTS Relative Page Number (RPN)
  - INSERT Partition
  - Asynchronous CF Lock structure duplexing
  - Setting initial Statistics Profile
- Summary

Db2 12 for z/OS Greatest Hits

- Performance
  - REBIND with APREUSE(ERROR|WARN)
  - Dynamic Prefetch Scheduling Avoidance
  - Fast Index Traversal (FTB) ***
  - Dynamic Plan Stability
  - Granular global commit LSN and read LSN
  - LOB compression
  - REORG (and LOAD) use of statistics profiles

- Application Development
  - SQL improvements
    - Pagination syntax LIMIT / OFFSET
    - Enhanced MERGE
    - Piece-wise DELETE
  - Native REST services

Db2 12 for z/OS Greatest Hits …

- Availability
  - Insert Partition
  - Larger size active log datasets
  - Partition by Range RPN
    - Online ALTER to increase DSSIZE
    - Lifting partition size limit (1 TB)
  - Asynch CF lock Duplexing

- Security
  - TRANSFER OWNERSHIP

Fast Un-clustered INSERT

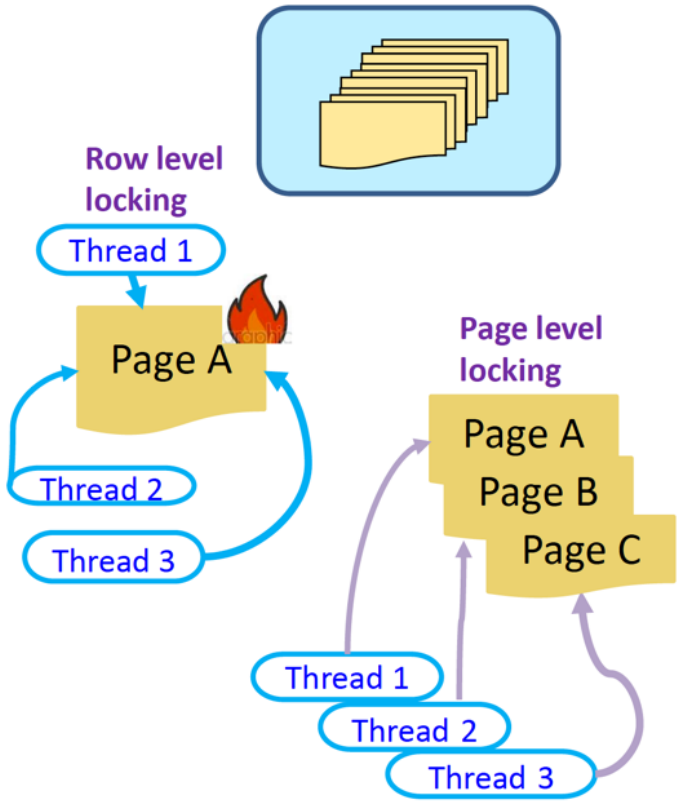
- Insert workloads are amongst the most prevalent and performance critical
- Performance bottleneck will vary across different insert workloads
  - Index maintenance?
  - Log write I/O?
  - Data space search (space map and page contention, false leads)
  - Format write during dataset extend
  - PPRC disk mirroring
  - Network latency
  - etc
- Common that index maintenance and/or log write IO wait may dominate and mask any insert speed bottleneck on table space

Fast Un-clustered INSERT …

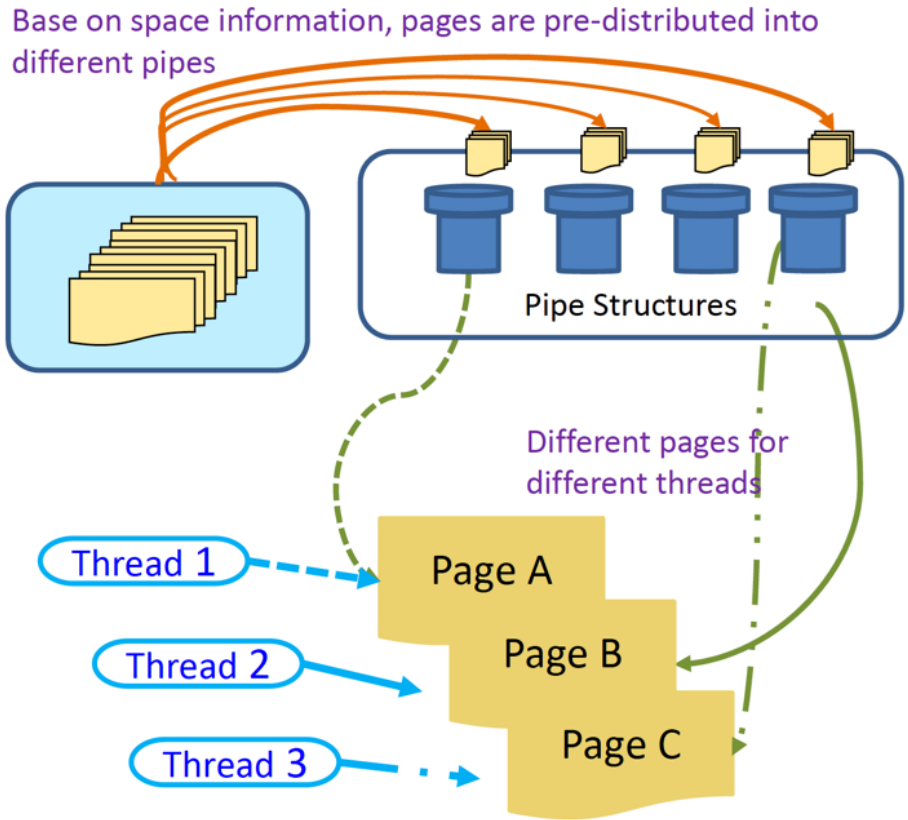
- Officially referred to as “Insert Algorithm 2 (IAG2)”
- Sometimes referred to as “Smart Insert” or even “Fast Insert”
- Potentially delivers significant improvement for un-clustered inserts (e.g., journal table pattern) where both
  - Heavy concurrent insert activity (many concurrent threads)
  - Space search and false leads on data is the constraint on overall insert throughput
- Applies to any UTS table space defined with MEMBER CLUSTER
  - Applies to both tables defined as APPEND YES or NO
- Implemented advanced new insert algorithm to streamline space search and space utilisation
  - Eliminates page contention and false leads
    - Space is preallocated/preassigned in order to fill up pipes which are pulled from
    - Set of pipes per pageset/partition per Db2 member
    - Space allocation occurs at pageset open and real time when there is space shortage in each individual pipe
  - Default is to use the new fast insert algorithm for qualifying table spaces
    - DEFAULT_INSERT_ALGORITHM system parameter can change the default
    - INSERT ALGORITHM table space attribute can override system parameter
- **It is NOT a replacement for the existing insert algorithm (IAG1)!**

Fast Un-clustered INSERT …

Insert Algorithm 1 concepts



Insert Algorithm 2 concepts



Fast Un-clustered INSERT …

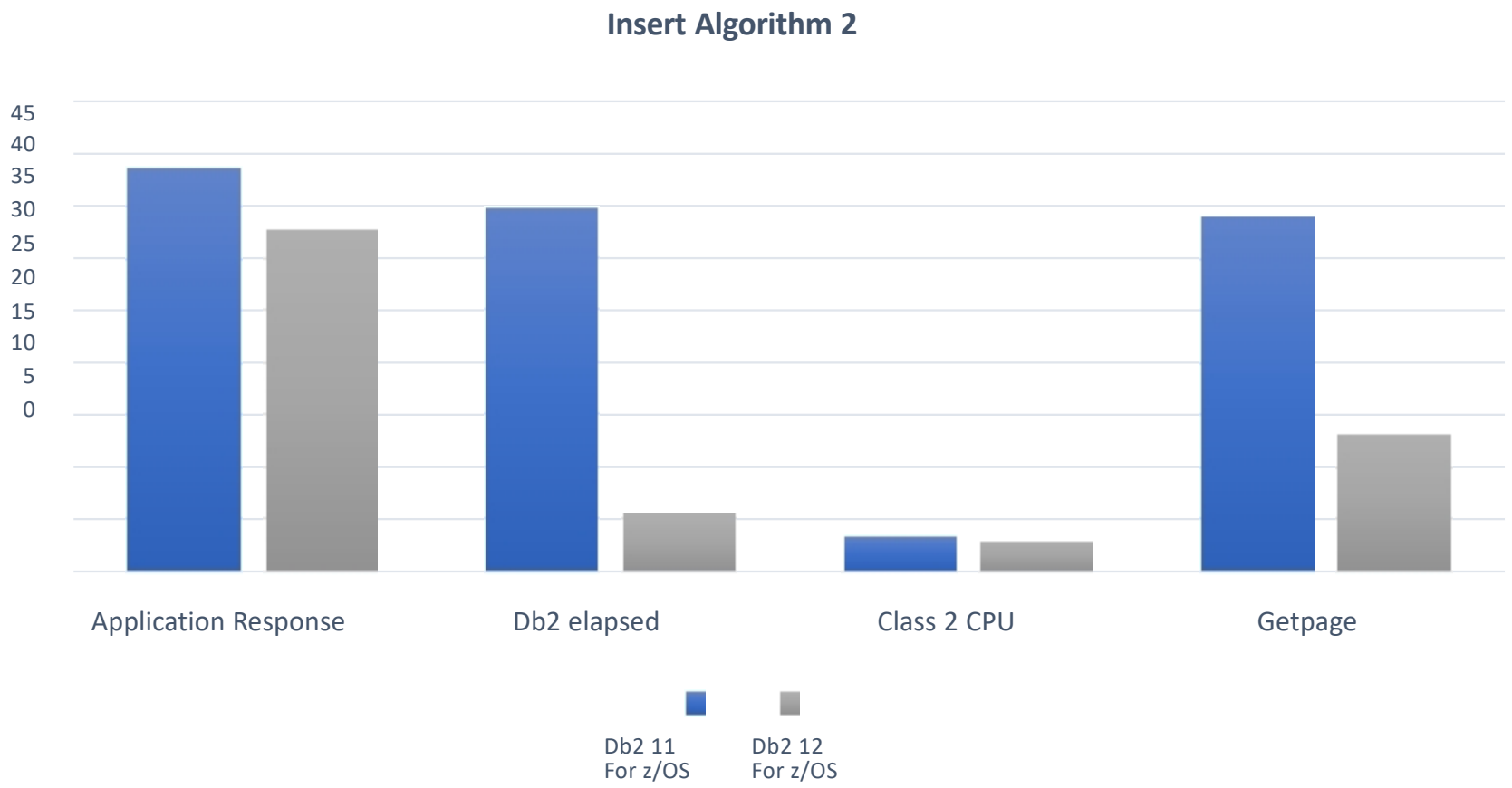
- Your mileage will vary
  - Many insert workloads will see no improvement and is to be expected
  - Will probably not see much difference/improvement when only one insert per commit scope
  - Some specific insert workloads may see significant improvement
  - Less benefit as more indexes are added to the respective table
- Will shift the bottleneck to the next constraining factor
- LOAD SHRLEVEL CHANGE can also use Fast Un-clustered INSERT
- Fast Un-clustered INSERT will be disabled when lock escalation occurs or use of SQL LOCK TABLE
- Available after new function activation (FL=V12R1M500)

Fast Un-clustered INSERT …

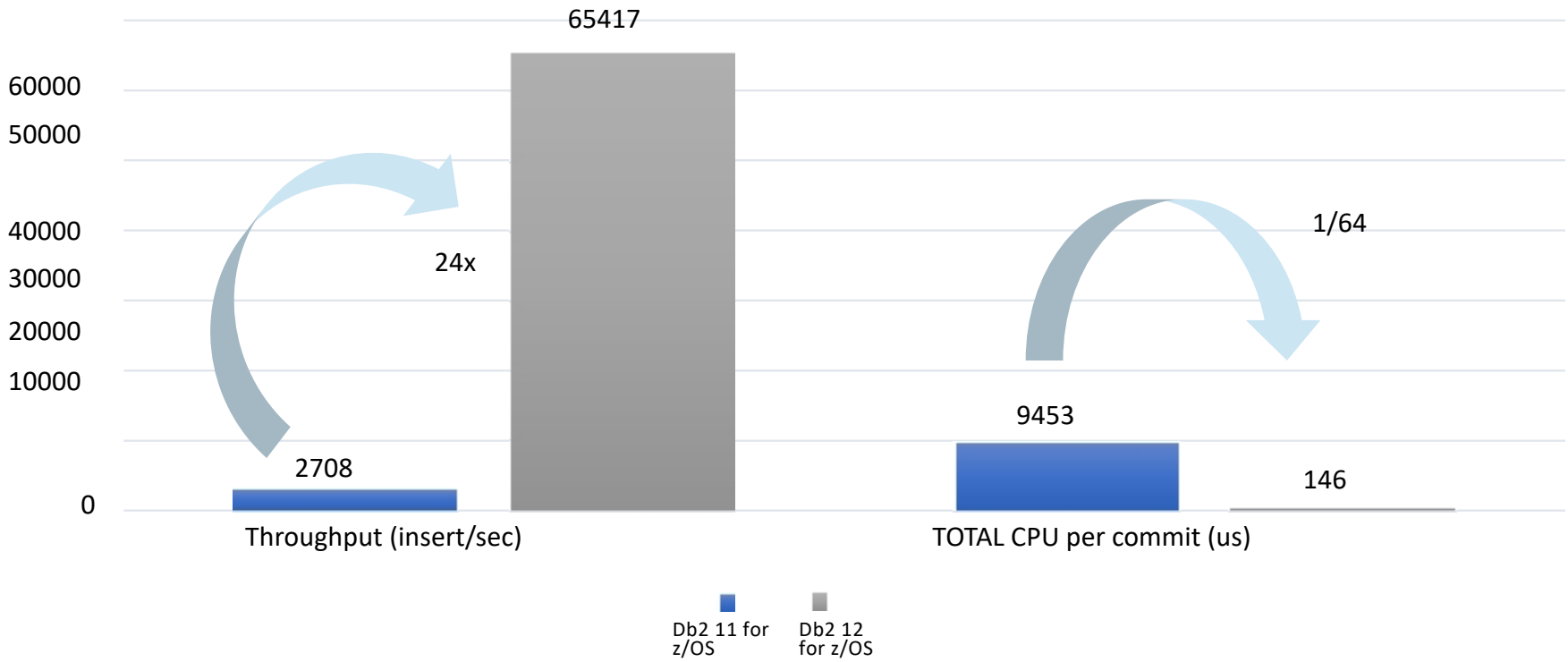
- APAR PH02052 (Closed) implements automatic re-enablement with retry logic

- Current JC point-in-time recommendation
  - One size probably does not fit all tablespaces
  - Change system wide default - set system parameter DEFAULT_INSERT_ALGORITHM = 1 (old basic insert algorithm)
  - Use INSERT ALGORITHM 2 (new fast insert algorithm) selectively at individual table space level to override system wide default
  - Additional benefit with CICS and DRDA thread reuse coupled with packages bound with RELEASE(DEALLOCATE)

Fast Un-clustered INSERT – Shifting The Bottleneck …



Fast Un-clustered INSERT - Db2 11 for z/OS PMR Recreate ...



UTS PBG with MEMBER CLUSTER, RLL, with 400 bytes per row, one index,
800 concurrent threads, 10 insert per commit

RTS enhancements

- New messages DSNT535I and DSNT536I e.g.,

  DSNT535I =D2E1 DSNIRTST 2 ATTEMPTS TO EXTERNALIZE IN-MEMORY STATISTICS TO REAL-TIME STATISTICS TABLES FAILED DURING THE PAST 30 MINUTES 'BECAUSE A RESOURCE WAS UNAVAILABLE: TYPE 00000304 NAME DSNDB06 .SYSTSISS.X'0000650D'.X'07'

- New column GETPAGES added to both SYSIBM.SYSTABLESPACESTATS & SYSIBM.SYSINDEXSPACESTATS
  - Very valuable
  - Records number of getpage requests since release migration, last REORG, last LOAD REPLACE or since object creation
  - Do not rely on the value whilst running in mixed release coexistence
- Temporal (system-period data versioning)
  - Requires FL=V12R1M5nn
  - SQL DDL changes performed by CATMAINT
  - Activated by ALTER TABLE … ADD VERSIONING clause -> SYSIBM.SYSTABLESPACESTATS & SYSIBM.SYSINDEXSPACESTATS
  - No indexes provided – must RYO to speed up your SQL queries
  - MAXPART 1 is ‘hard wired’ for history tables
  - Must develop procedures for cleanup of history tables and associated housekeeping

Fast Index Traversal

- In memory index performance optimisation
- One of the most important performance features in Db2 12 for z/OS
- Used for fast index lookup by avoiding expensive index B-tree traversal
- Access must be random (index traversal) pattern to benefit
- SELECT, INSERT, DELETE, UPDATE, … can all benefit
- Separate Fast Traversal Block (FTB) memory area allocated outside of bufferpool
  - Uses a concatenated structure, containing copy of non-leaf pages only, uses relative structure
- Does not use bufferpool
  - Non-leaf pages (except root page) are not fixed in the bufferpool
  - Pages are eligible for stealing and can be LRUed out of the bufferpool when the non-leaf pages are stored in FTB memory
- Improved performance
  - Fast traverse block is L2 cache aware B-Tree like structure
  - Each page is equal to one cache line in size (256 bytes)
- ESP customer example with 9.1% CPU reduction with 3 level index, 22.9% CPU reduction with 4 level index
- Your mileage in terms of CPU reduction will vary
- Be aggressive in applying preventative service e.g., apply PTF for PE resolution APAR PH07545

Fast Index Traversal …

- zparm INDEX_MEMORY_CONTROL = AUTO, DISABLE, x (MB)
  - AUTO = 20% of total allocated bufferpool size (min 10 MB)
    - Subject to maximum limit of 10000 FTBs (one FTB per index partition)
  - Limit with x (MB) is 200,000 MB
- Each Db2 member will determine independently the good candidate indexes (daemon)
  - Index must be unique
  - INCLUDE COLUMNS supported
  - Index entry length (key + additional columns) has maximum size of 64 bytes
  - Re-evaluates every 2 minutes and adjusts priority queue
    - Index traversal (+)
    - Index only access (++)
    - Index leaf page splits (/2)
    - Index lookaside (-)
  - Internal threshold then applied
- Control by SYSIBM.SYSINDEXCONTROL
  - Indicate preference for specific indexes
  - Disable for specific indexes

Fast Index Traversal …

- How does an index partition come into FTB area?
  - Daemon task
    - zIIP eligible
    - Runs every 2 minutes
    - System agent correlation identifier: 014.IFTOMK00

```
DSNV497I  -DB2A SYSTEM THREADS -
DB2 ACTIVE
NAME      ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
...
DB2A      N  *     0 014.RTSTST00 SYSOPR            004C     0
V490-SUSPENDED 17081-10:05:25.83 DSNB1TMR +00000EBF UI38562
DB2A      N  *     0 014.IDAEMK00 SYSOPR            004C     0
V490-SUSPENDED 17081-10:01:16.95 DSNB1TMR +00000EBF UI38562
DB2A      N  *     0 014.IFTOMK00 SYSOPR            004C     0
V490-SUSPENDED 17081-10:05:22.32 DSNB1TMR +00000EBF UI38562
DB2A      N  *     0 010.PM2PCP01 SYSOPR            004C     0
V490-SUSPENDED 17081-10:05:26.51 DSNB1TMR +00000EBF UI38562
...
```

Fast Index Traversal …

- Monitor
  - -DISPLAY STATS(IMU) or –DISPLAY STATS(INDEXMEMORYUSAGE) LIMIT(*) command

```
DSNT783I  -DB2A
DBID PSID DBNAME   CREATOR         INDEXNAME      LEVEL PART  SIZE(KB)
---- ---- -------- -------------- -------------- ----- ----- --------
0256 0005 SZI10D    § § § § § § §   SZI10X          0002  00001 00000025
0261 0005 SZI20D   A2345678901234 SZI20X          0002  00001 00000025
0262 0005 SZI30D   SYSADM          X2345678901234 0002  00001 00000025
0263 0005 SZI40D   SYSADM          SZI40X          0002  00001 00000025
******* DISPLAY OF STATS TERMINATED **********************************
DSN9022I  -DB2A DSNTDSTS 'DISPLAY STATS' NORMAL COMPLETION
```

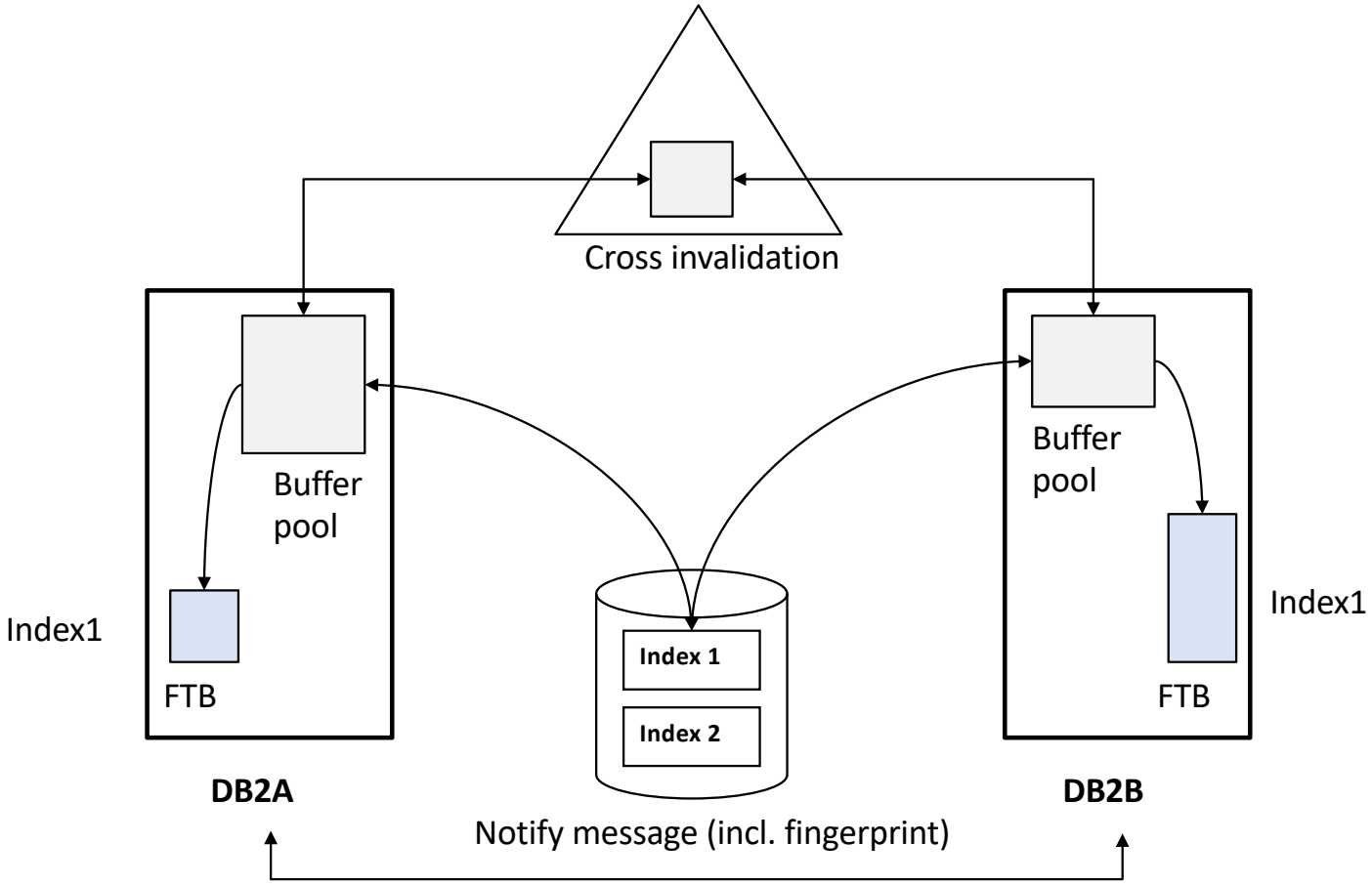
  - Trace
    - -START TRACE (PERFM) DEST(SMF) IFCID(477)
    - -START TRACE (STAT) DEST(SMF) CLASS(8) IFCID(389)

Fast Index Traversal …

- Free FTB area for an index
  - Pageset close
  - SQL mass delete
  - ALTER INDEX, RECOVER INDEX, REBUILD INDEX
  - Trick: ALTER INDEX from COPY YES to COPY NO (and the other way around)

Fast Index Traversal …

- Data Sharing considerations – high level picture



Fast Index Traversal …

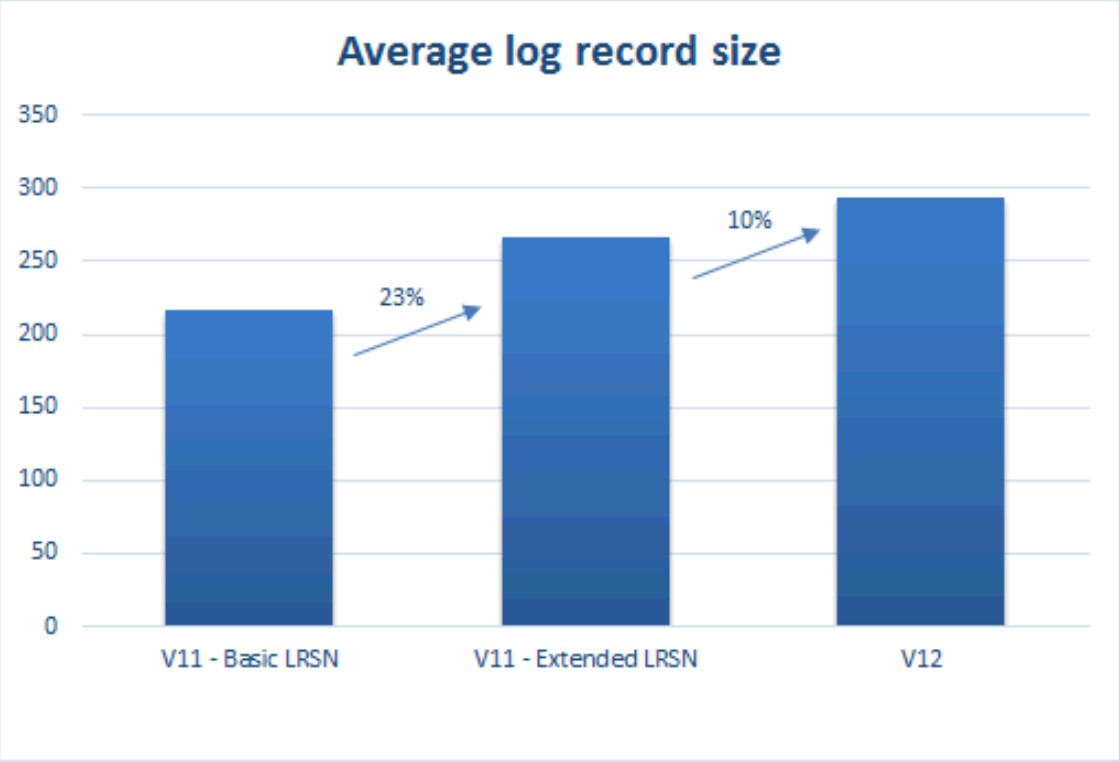
- Migration
  - Available in mixed release coexistence (Db2 11 and Db2 12 for z/OS) or Db2 12 for z/OS before new function activation (V12R1M100)
    - FTB only used while index object is not GBP-dependent
    - If index object becomes GBP-dependent, the FTB content will be deleted/bypassed
  - After new function activation (V12R1M500)
    - FTB can now also be used when index object is GBP-dependent

Data dependent vs. numeric based pagination syntax

- Data dependent pagination syntax e.g., SELECT … FROM … WHERE (LASTNAME, FIRSTNAME) >= (:lname, :fname)
  - Performance advantage predicated on correct index design and ORDER BY
    - Can go directly to the needed rows
    - Exploits range-list index scan (ACCESSTYPE='NR')
- Numeric based pagination syntax e.g., SELECT … FROM … OFFSET 10 ROWS FETCH FIRST :hv ROWS ONLY
  - Will have to skip through the unneeded rows
  - If rows are deleted/inserted from other applications in between
    - May see the same rows twice or not see the rows at all
- Many static scrollable cursors can be replaced by SQL pagination
  - Result set is no longer materialized
  - Read-only applications will not create long running unit of recoveries
  - Performance can be improved
- Requires APPLCOMPAT(V12R1M500)
- Works very well as advertised

Increase on log record size after converting BSDS in Db2 11 & entry to Db2 12

- About 50 byte increase after converting BSDS under Db2 11 for z/OS NFM
- Further increase in log record size in Db2 12 for z/OS because of larger 7-byte RID values
  - Increase is about 20 bytes for table space and about 28 bytes for index



Dynamic Plan Stability

- Welcome new feature that will bring some relief in the area of performance management of dynamic SQL
  - Goal is to provide consistent, more reliable performance
  - Sweet spot is short running SQL that is executed 1000s of times
  - Helps with high "turnover" periods in dynamic statement cache
- In Db2 11 for z/OS a miss in dynamic statement cache requires a new full prepare e.g.,
  - Db2 subsystem recycle
  - Release migration
  - RUNSTATS
- In Db2 12 for z/OS can stabilize a query statement from the dynamic statement cache
  - No new full prepare needed
  - Statement is loaded into the dynamic statement cache from the Catalog
  - Statement is invalidated by SQL DDL like a static SQL package
- Can stabilize
  - Specific dynamic query statement
  - Dynamic query statements with more than a certain amount of executions

Dynamic Plan Stability ...

- Change of APPLCOMPAT and/or special registers (DEGREE, OPTHINT, etc) will cause cache miss
- No REBIND capability to “repair” after invalidations
  - Need to wait for new stabilization
- Restrictions
  - Display command has only local scope
  - No support for concentrated statements
  - No support for query statements against temporal and transparent archive
- FREE stabilized dynamic query STBLGRP(x)
  - Will also invalidate the statements in the dynamic statement cache
  - May result in a “storm” of full prepares
- Stabilized dynamic query statements do consume more CPU than the equivalent static query statement

More granular global commit LSN and global read LSN

- Db2 for z/OS does not actually track "more current" value for each individual object
- Each member maintains two global lists of the 500 objects that have the oldest CLSN and read-LSN values
- Global lists built by a system task that wakes every 2 seconds (subject to change)
- Rebuilds its own list
- Merges it with every other member's list to create the global list
- When it comes time to pick up an object's CLSN or read-LSN value
  - Check the appropriate global list for the object
    - If it is on there, then we know what its LSN is
    - If not, then use as an "alternate" LSN for the newest object (as object's LSN cannot be worse than this value)
  - Either way Db2 will compare the LSN picked up with the old global value (from SCA), and use that if it is better
- Very nice enhancement that has great potential to improve lock avoidance and/or space reuse on LOB insert when the inevitable long running reader-UR is in play

LOB compression

- Requires zEDC hardware feature
  - Will decompress existing compressed LOB if zEDC not available
  - Will not compress a LOB if zEDC not available
- Inline LOB is completely separate from LOB compression
  - LOB compression only applies to the the out-of-line portion
  - Split and compressed independently
- Aimed at textual
  - Not video and audio as these are already heavily compressed outside of Db2 for z/OS e.g., MP3 or MP4

SQLCODE -109 Issue

- Problem:
  - Non-documented and illegal use of SELECT ... INTO ... UNION ALL syntax
    - Customer complaints, can produce wrong results, defect
- Solution:
  - Loophole closed in Db2 12 for z/OS
  - Retrofitted back to Db2 11 for z/OS with APAR PI67611
  - New zparm: DISALLOW_SEL_INTO_UNION
    - NO (Db2 11 for z/OS default)
      - Allows usage of this illegal SQL syntax when such usage is encountered during execution of a BIND or REBIND command
      - Db2 will write an incompatibility trace record to IFCID 376
      - Use these trace records to identify and correct applications that are using the illegal SQL syntax
    - **YES** (Db2 12 for z/OS default)
      - Disallow usage of this illegal SQL syntax
      - Statements that include syntax will fail with SQLCODE -109
      - Running IFCID 376 under Db2 11 for z/OS will help identify problem applications
  - Need to deal with this potential issue before migration to Db2 12 for z/OS or change the Db2 12 for z/OS default

Enhanced SQL MERGE

- Db2 12 for z/OS delivers ANSI compliant MERGE capability
- SQL MERGE is now very powerful
  - Source can now include TABLE, VIEW and full Select
  - Additional predicates on MATCHED/NOT MATCHED
  - Can do DELETE
  - Can do multiple UPDATE, INSERT and DELETE phrases
    - But not on same row
    - Can accept SIGNAL and IGNORE
- Benefits
  - Development productivity
  - Improved performance
  - Application porting to Db2 for z/OS
- Requires APPLCOMPAT(V12R1M500)

Enhanced SQL MERGE ...

- But SQL MERGE is now so powerful ...
  - Input can be a SELECT (JOIN) returning many rows (millions, billions)
  - # UPDATEs, INSERTs and DELETEs could explode
  - Considerations
    - No intermediate commit points
    - Long rollback time
    - Lock escalation and impact on concurrency
    - No SQL pagination support

UTS PBR Relative Page Number (RPN)

- Motivation
  - Tremendous improvement in terms of availability and usability
    - DSSIZE can vary for different partitions
    - DSSIZE can now be increased for an individual partition with zero application impact
      - Immediate alter and no REORG required to increase DSSIZE
    - Note: A decrease in DSSIZE is still a pending alter and requires a full table space level REORG
  - Scalability
    - Maximum partition size increases to 1 TB
    - Maximum table size increases to 4 PB
    - Maximum number of rows in a table increases from 1.1 Tn to 280 Tn

UTS PBR Relative Page Number (RPN) …

- Migration possible from either classic partitioned and UTS Partition By Range (PBR) table spaces
  - Steps for conversion
    1. ALTER TABLESPACE … SEGSIZE n
       - If starting from classic partitioned
    2. ALTER TABLESPACE … PAGENUM RELATIVE
       - Table space put into AREOR state
    3. REORG TABLESPACE …
  - Base and XML table spaces can be migrated separately
    - Can “coexist” running with mixed RELATIVE/ABSOLUTE attributes
  - One-way ticket – no fallback to absolute page numbering (PAGENUM ABSOLUTE)
  - Extended Addressability (EA) must be used for UTS PBR RPN datasets
  - DASD space for large datasets can lead to problems (e.g. running out of volumes)
    - Datasets can only be spread across 59 volumes
    - For example, a 1 TB dataset will require 3390 Model 27 or above
- Requires APPLCOMPAT(V12R1M500)

UTS PBR Relative Page Number (RPN) …

- Migration issues
  - Cannot convert to RPN or even create new RPN tablespace because cannot REORG them when inline part-level image copies (forced) go to tape
    - New TAPEUNITS option will be made available with APAR PI75518 (open)
  - Pre-V6 range partitioned tablespaces with limit key values truncated at 40 bytes cannot be converted over
    - Should only affect a small number of customers
    - Problem is fenced and the conversion will not succeed
      - ALTER TABLESPACE PAGENUM RELATIVE fails with SQLCODE -650 RC 39
    - No target date at present time for providing relief

UTS PBR Relative Page Number (RPN) …

- Other considerations
  - Indexes will increase in size because of larger 7-byte RID values
  - Recommend the index COPY/RECOVER for XXXL size NPIs
  - Note: can no longer identify the partition number from the page number

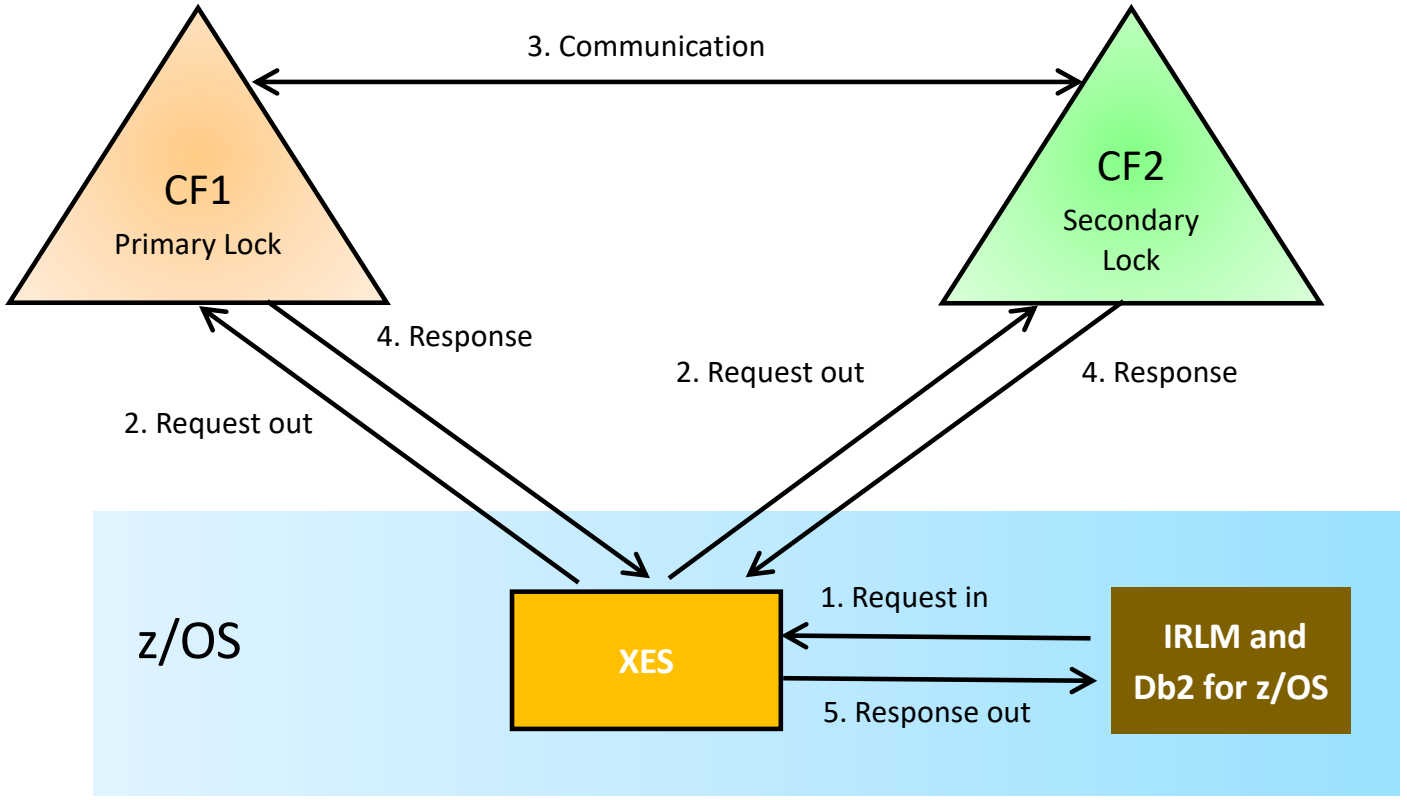
Insert Partition

- Insert Partition “in the middle” where it is required
- UTS PBR only, BUT no requirement for RPN
- Restriction: no LOB or XML
- ALTER … ADD PART ENDING AT (…) is a pending alter
- Necessary REORG can be limited to a minimum subset of partitions (only affected partitions)
- Be aware that logical partition numbers have to be translated to physical partition numbers
  - New physical partition is added at the end i.e., A00n+1
  - New logical partition is added in the middle and logical partitions are appropriately renumbered
  - Awkward consideration with utilities - range of parts - as it is based on physical partition numbers
- Do not have to take care of adjacent partitions which possibly reach their space limit
- Once you determine the limit key for the new inserted partition, the procedure for handling “partition full” conditions is very easy to automate
  - Add new partition
  - Run REORG against the new and adjacent partitions
- Requires APPLCOMPAT(V12R1M500)

System Managed Duplexing (SMD) of CF Lock Structure – Challenges

- Required for highest availability in Db2 for z/OS data sharing environments
  - Single and Multi-site z/OS Parallel Sysplex environments with no failure isolated CFs or external CFs
  - Without SMD, the failure of the 'wrong CF' may result in a group-wide outage
    - LOCK1 or SCA can only be dynamically rebuilt into an alternate CF if all the Db2 for z/OS members survive the failure
- Existing synchronous SMD of LOCK1 structure can be expensive in terms of increased host CPU resource consumption, degraded application elapsed time performance, and aggravated global lock contention
  - All types of requests are duplexed
  - Duplexed request can consume 3x-4x host CPU cost vs. simplex structure
  - Synchronous lock requests are converted to asynchronous requests to limit host CPU penalty
  - CF service times will increase which will elongate transaction response times and batch processing elapsed time, and possibly aggravate global lock contention
  - Performance impact will vary
    - Dependent on locking intensity of respective application workload
    - Stretched distance for Multi-site data sharing group

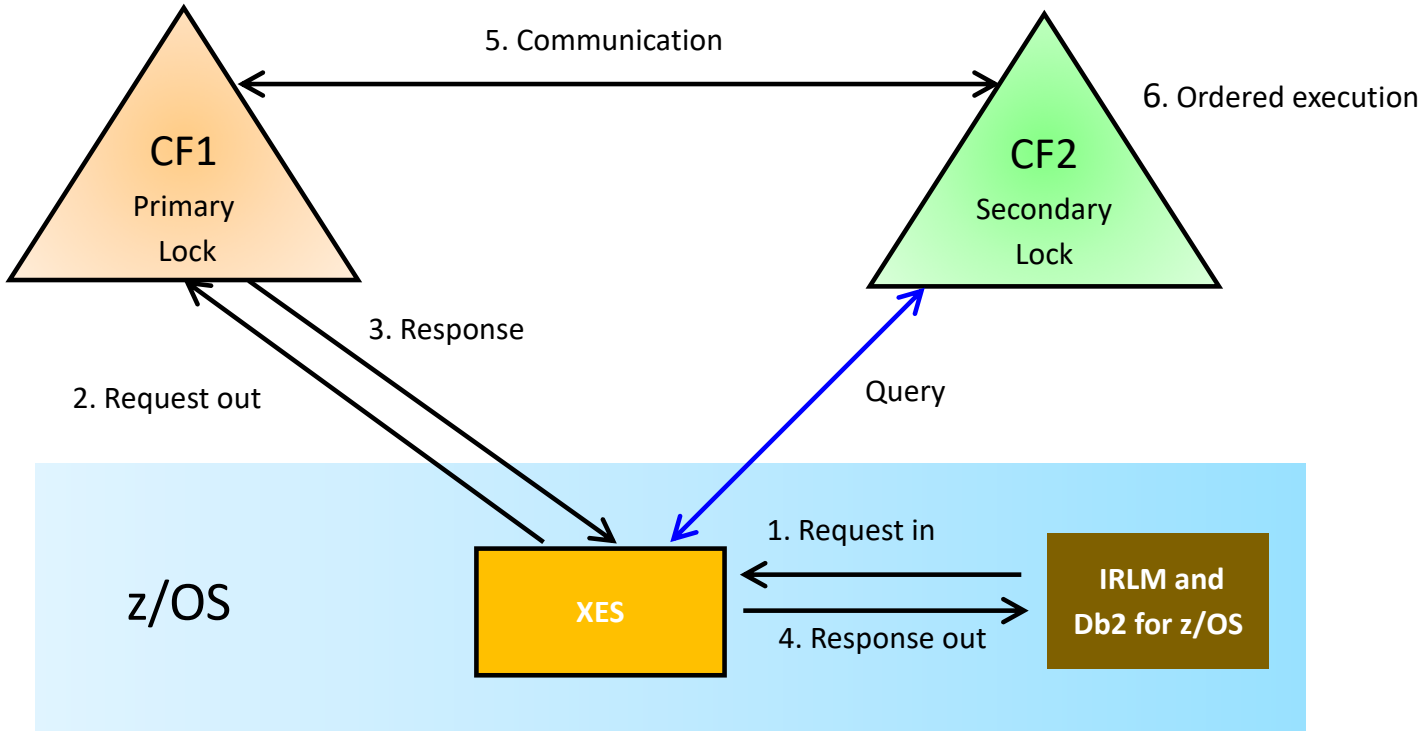
Synchronous CF lock structure duplexing – how it works today



Asynchronous CF Lock structure duplexing new in Db2 12 for z/OS

- Reduces overhead for system managed duplexing of CF LOCK1
- Secondary structure updates are performed asynchronously with respect to primary updates
- Db2 for z/OS will sync up with z/OS to ensure data integrity i.e., all modify locks have been “hardened” in the secondary lock structure before the corresponding undo/redo record for the update is written to the Db2 for z/OS active log on DASD
- The physical log writer performs the "synch" call to query the secondary, and it happens whenever log records get physically written to DASD which can be earlier than commit
- Increases the practical distance for multi-site sysplex operations while duplexing of CF LOCK1 structure
- Requirements:
  - IRLM V2R3 Function Level 40 with PTFs
    - Db2 12 for z/OS FL=V12R1M100 with PTF for APAR PI66689
    - IRLM V2R3 with PTF for APAR PI68378
  - CFCC firmware support for CFLEVEL 21 Service Level 02.16 (z13)
  - z/OS V2R2 SPE with PTFs for APARs OA47796 and OA49148
  - CF to CF connectivity via coupling links

Asynchronous CF lock structure duplexing – how it works



* Requires CF on z13 GA2

Asynchronous CF Lock structure duplexing new in Db2 12 for z/OS …

- Benefits
  - Cost of lock structure duplexing is significantly lower than before
    - Host CPU for lock requests decreases
    - IRLMs receive responses sooner
  - Existing sites using synchronous SMD should see lower host CPU cost and better elapsed times
  - More environments can now achieve higher availability in all-ICF configurations with SMD
    - Reduce risk with asynchronous SMD and lower cost all round
      - Hardware maintenance
      - Capital cost for extra frames
  - Processor technology refresh applies to both host GCP and ICF engines
- But it is not free
  - Will have to acquire ICF engines and coupling links for CF to CF connectivity
  - CF Utilisation is significantly higher for async SMD relative to simplex case, but it is much less than sync SMD
    - Expected to be higher than simplex because there is simply more work for the CF to do

Asynchronous CF Lock structure duplexing new in Db2 12 for z/OS …

- Performance Summary comparing async SMD relative to simplex
  - ITR degraded by 7.5%
  - Response time and ETR are comparable
  - z/OS host CPU resource consumption is higher
  - CF CPU resource consumption is significantly higher

Setting initial STATISTICS PROFILE

- It is important to clean up any (SYSCOLDIST) statistics that you do not intend to regularly collect before first BIND/REBIND, PREPARE or EXPLAIN after entry to Db2 12 for z/OS
- These statistics could be stale or inconsistent today because they are not being regularly collected
- Statistics profile is created on first BIND/REBIND/PREPARE/EXPLAIN after entry to Db2 12 for z/OS
- After the initial create, cannot tell from the subject statistics profile what statistics are the ones that were the older/inconsistent statistics

Setting initial STATISTICS PROFILE ...

- Use the following sample query to identify the inconsistent statistics

```
SELECT TYPE, NUMCOLUMNS, TBOWNER, TBNAME, NAME
, MIN(STATSTIME), COUNT(*)
FROM SYSIBM.SYSCOLDIST CD
WHERE STATSTIME < CURRENT TIMESTAMP - 1 MONTH
AND (TYPE IN ('C', 'H') OR NUMCOLUMNS > 1
  OR STATSTIME < CURRENT TIMESTAMP - 1 YEAR)
AND NOT EXISTS
(SELECT 1
FROM SYSIBM.SYSINDEXES I
WHERE I.TBCREATOR = CD.TBOWNER
AND   I.TBNAME = CD.TBNAME
AND   CD.STATSTIME BETWEEN I.STATSTIME - 8 DAYS
                       AND I.STATSTIME + 8 DAYS)
AND NOT EXISTS
(SELECT 1
FROM SYSIBM.SYSTABLES T
WHERE T.CREATOR = CD.TBOWNER
AND   T.NAME = CD.TBNAME
AND   CD.STATSTIME BETWEEN T.STATSTIME - 8 DAYS
                       AND T.STATSTIME + 8 DAYS)

GROUP BY TYPE, NUMCOLUMNS, TBOWNER, TBNAME, NAME
ORDER BY TYPE, NUMCOLUMNS, TBOWNER, TBNAME, NAME
WITH UR;
```

Summary

- Share lessons learned, surprises, pitfalls
- Provide hints and tips
- Address some myths
- Provide additional planning information
- Provide usage guidelines and positioning on new enhancements
- Help customers migrate as fast as possible, but safely

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