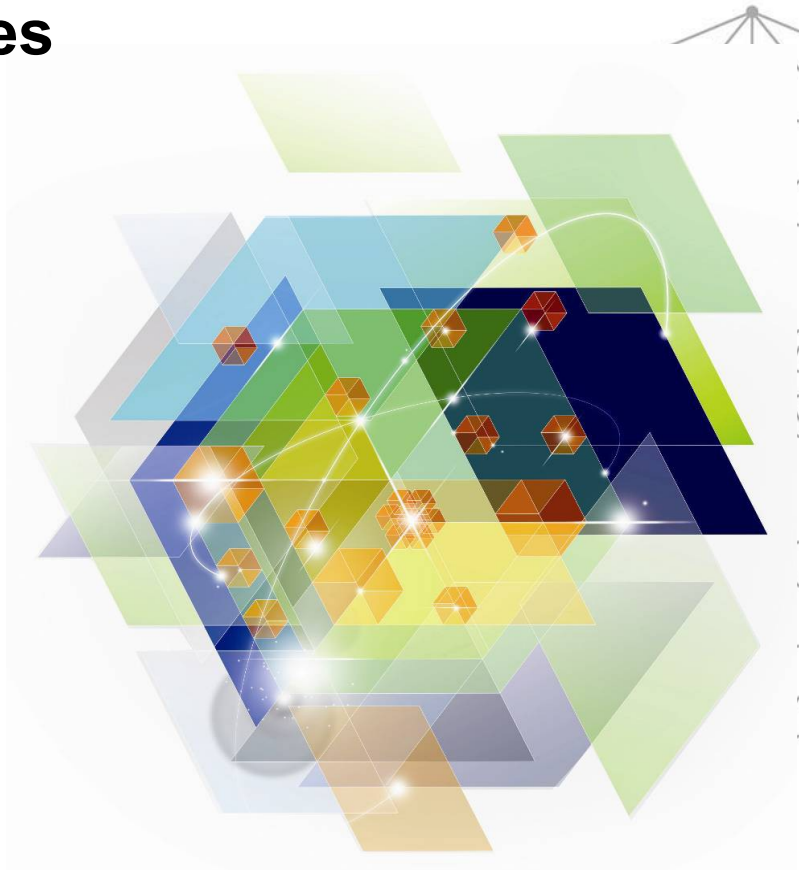
DB2 for z/OS Utilities Best Practices

Andy Lai

DB2 Utilities Development

atlai@us.ibm.com



#SHAREOrg

SHARE is an independent volunteer-run information technology association that provides education, professional networking and industry influence.

SHARE in Pittsburgh 2014

Disclaimer

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

DB2 Utilities Best Practices Agenda

- General recommendations
- Utility SORT processing
- COPY & FlashCopy
- RECOVER/QUIESCE/MODIFY RECOVERY
- LOAD/UNLOAD
- REORG
- RUNSTATS
- CHECK
- DSN1COPY
- Summary

The Increasing Importance of IBM DB2 Utilities

DB2 Utilities Suite is required for core function enablement in DB2

- DB2 Utilities Suite provides data & meta-data conversion capability
- REORG/LOAD row format conversion in DB2 9
- REORG catalog/directory conversion during DB2 10 ENFM
- REORG non-disruptive meta-data changes in DB2 10 and beyond
  - Page set conversion, page size alteration, etc.
- REORG/LOAD inline LOBs in DB2 10
  - Including non-disruptive conversion of existing LOB data
- Utility support for hash page sets in DB2 10
  - Including auto-estimation of hash space in REORG
- Utility support for spatial indexes in DB2 10
  - Retrofitted to DB2 9
- Utility support for pending ALTER LIMIT KEY support in DB2 11
- Utility support for pending ALTER support for DROP COLUMN in V11

… in addition to resizing page sets, restoring clustering, reclaiming physical space etc.

Utilities & CPU

- Focus on total CPU elimination in addition to CPU reduction & zIIP exploitation
- Real CPU cost reduction in V9 and more since
- DB2 utilities have been zIIP-enabled since 2006
- zIIP offload for utility sort with DFSORT
- DB2 Sort product for further reduction in sort elapsed time & CPU consumption
- FlashCopy exploitation in DB2 10 dramatically reduces CPU consumption for COPY & reduces CPU for RECOVER & inline copies
- More zIIP offload in DB2 10 with RUNSTATS and yet even more in V11
- More zIIP offload for REORG in DB2 9 via APAR PM37622
- Continued delivery of performance & CPU-reduction improvements via maintenance & new releases

Sort processing

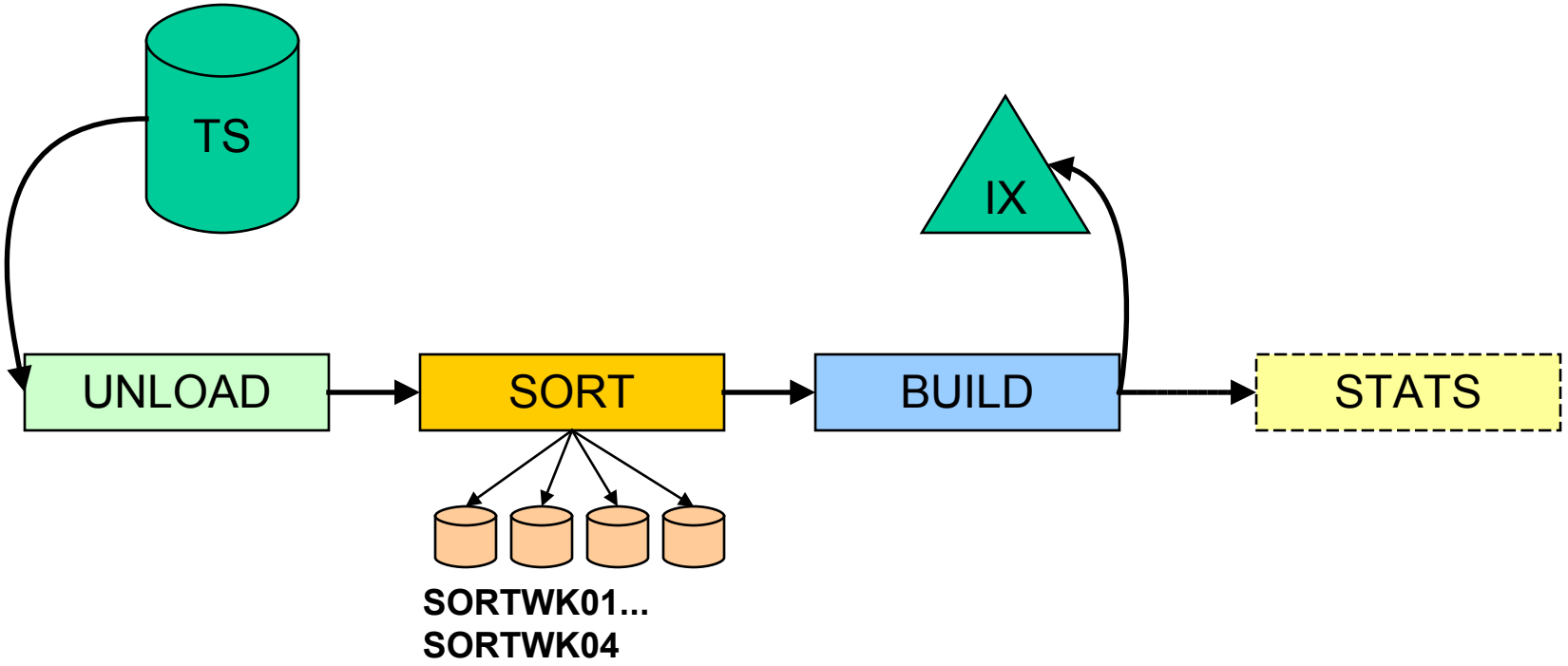
- Improved utility sort processing
  - CHECK INDEX, REBUILD INDEX, REORG, RUNSTATS
  - PK45916 (V8) & PK41899 (V9)
  - Better performance, more robust, simpler
- SORTNUM no longer required
  - Correct value hard to determine, resulting in utility failure if too low or excessive sort work allocation if too high
- New zparms UTSORTAL & IGNSORTN (online changeable)
  - UTSORTAL YES|NO
    - Use RTS data to estimate number of rows to sort
    - DB2 will dynamically allocate sort work datasets
      - If SORTWK DD cards not hard coded
  - IGNSORTN YES|NO
    - Override utility job setting of SORTNUM
- Recommendation
  - Turn on UTSORTAL, test it, then consider turning on IGNSORTN

DSNU3340I 168 08:13:52.66 DSNUGLSR - UTILITY PERFORMS DYNAMIC ALLOCATION OF SORT DISK SPACE

Invoking Sort

Sample invocation of sort for REBUILD INDEX

Non partitioned table space, 1 index

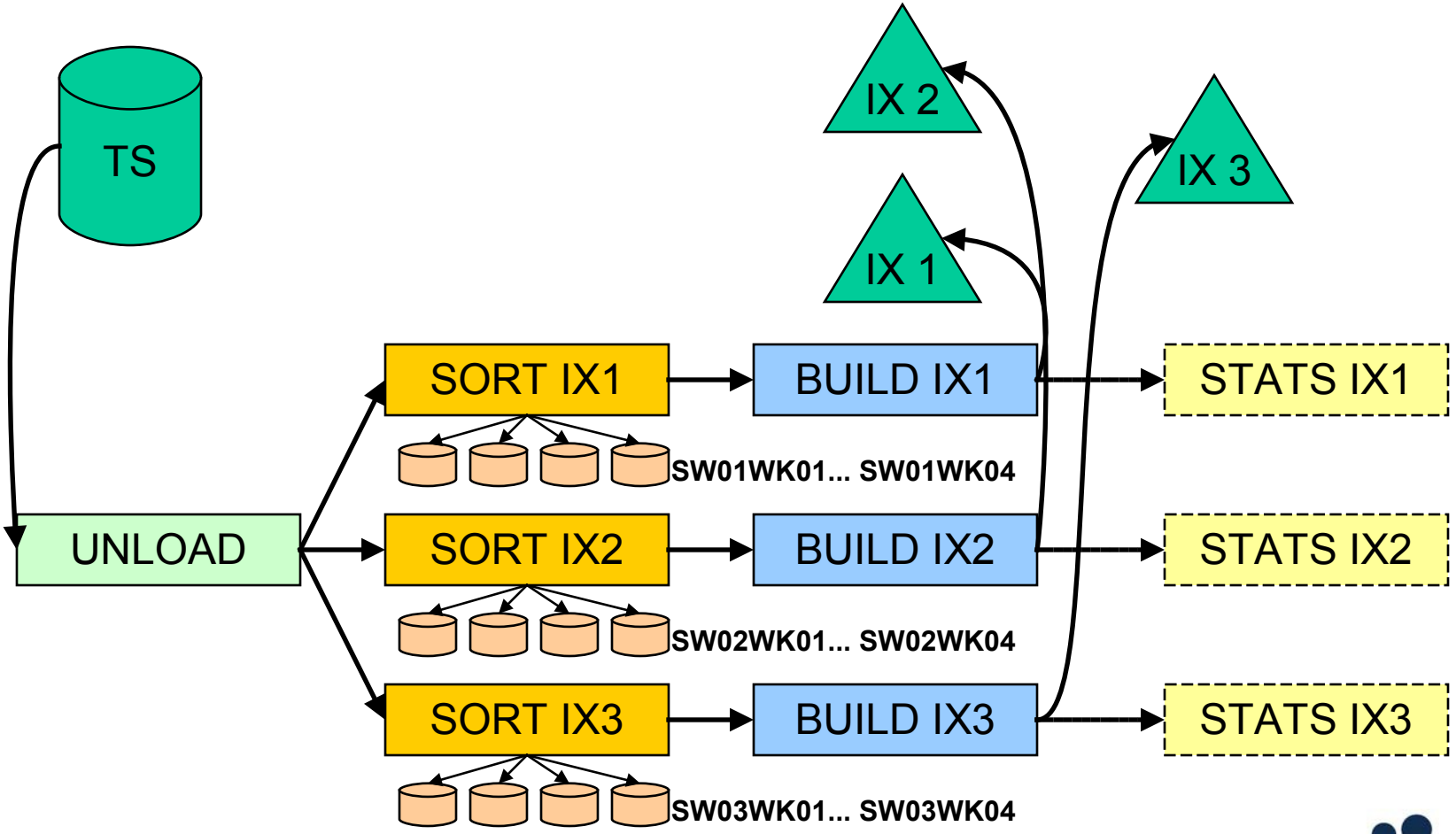


Improving Elapsed Time With Parallel Sorts

- Most utilities invoke multiple sorts in parallel to reduce elapsed time
- Parallelization can be achieved over table space partitions or indexes, depends on utility
- Needs separate sort work data sets and sort output data sets for each subtask
  - Use dynamic allocation for those data sets
  - Can use hard coded data sets, but need to follow naming scheme correctly
- Increases load on system
  - More CPU needed in parallel
  - More concurrent I/O
  - More memory required
- Can actually reduce amount of sort work space needed

Parallel Sorts Example

REBUILD INDEX, non partitioned table space, 3 indexes



Degree of Parallelism

- Utilities determine the degree of parallelism based on
  - Specification of SORTDEVT to allow dynamic allocation, or naming of hard coded sort work data set DD cards
  - Number of CPUs available (some utilities)
  - Available memory below and above the line
  - Estimated number of sort work data sets required, depending on size of data
- Recommendations for optimal degree of parallelism
  - Specify SORTDEVT option
  - Use DB2 allocation of sort work data sets (UTSORTAL=YES)
  - Provide sufficient memory size in REGION
- If degree of parallelism needs to be limited
  - Use PARALLEL n option in DB2 11
  - Or, use //UTPRIN01.. //UTPRIN03 DD cards to limit the number of tasks

Enable Good Estimates for Sorting

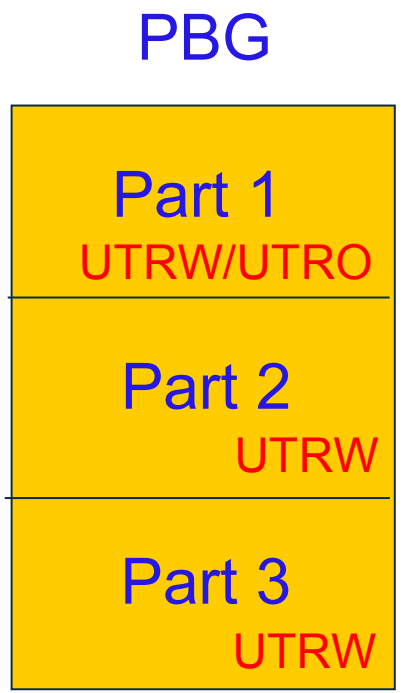
- Make sure Real-Time Statistics exist and are not misleading, e.g. after DSN1COPY
  - Check SYSTABLESPACESTATS.TOTALROWS and DATASIZE, and SYSINDEXSPACESTATS.TOTALENTRIES
  - If incorrect after DSN1COPY, either set these values to NULL or to estimated numbers, they don‘t have to be exact
- Check last run of RUNSTATS and consider running RUNSTATS
  - When average row length changed significantly
  - Or compression ratio changed significantly
- If MAXROWS set to a small number (> 1)
  - Is it really needed, or can it be removed?
  - If it cannot be changed, it can have a negative impact on sort estimates

DB2 Sort for z/OS

- Provides high speed utility sort processing for DB2 for z/OS
- More effective sort algorithms to save CPU and elapsed time
  - Specifically tailored to the way how utilities operate
- Much higher zIIP offload capability for further CPU savings
- More robust against incorrect estimates
  - Can allocate additional sort work data sets later on
- Based on Syncsort technology with additional enhancements
  - Special sort algorithm only found in DB2 Sort
  - API to optimally distribute available memory resources among parallel sorts
  - Tight integration with DB2 code for further CPU improvements
- Easy installation and customization

PBGs

- No LOAD or REORG parallelism
- No pruning of partitions until V11
- No LOAD at partition level
- Rows can flow from one part to another within part range
- LOAD / REORG at table space level will grow new parts as needed
  - V9 restrictions lifted in DB2 10:
    - REORG cannot grow new parts if LOB column exists
    - REORG cannot move rows between parts if LOB column exists
- REORG of single part or subset of parts will not grow new parts
  - Rows must fit back into part, but may not!
    - PCTFREE/FREEPAGE may cause REORG to fail
- What to do if REORG fails because rows won't fit?
  - View as single table and REORG whole table space
    - If LOB columns exist then may need to UNLOAD/RELOAD if pre-V10
  - Use zparm REORG_IGNORE_FREESPACE to ignore PCTFREE/FREEPAGE for part-level PBG REORG
    - PM53254 – forward fit zparm to DB2 10



RRF

- RRF row format introduced in V9 NFM to remove overhead of processing rows with variable length columns
- Some concern about conversion to RRF from BRF on REORG or LOAD in V9 NFM
  - Primarily concern is worse compression ratio for rows with many var length columns
- PK87348 & PK85881 provide complete control
  - ZPARM SPRMRRF can enable or disable conversion to RRF
    - Default is ENABLE
    - DISABLE will not convert back to BRF
  - ROWFORMAT option on LOAD & REORG to permit conversion to/from RRF regardless of SPRMRRF setting
- Apply PM40646 to avoid BRF/RRF mismatch problems in utilities if using DSN1COPY to move data
- General recommendation: Keep SPRMRRF default
  - DB2 will handle tables with mixed partitions
  - DB2 will handle PIT recovery to prior to BRF/RRF conversion

Utilities on demand with tools & stored procedures

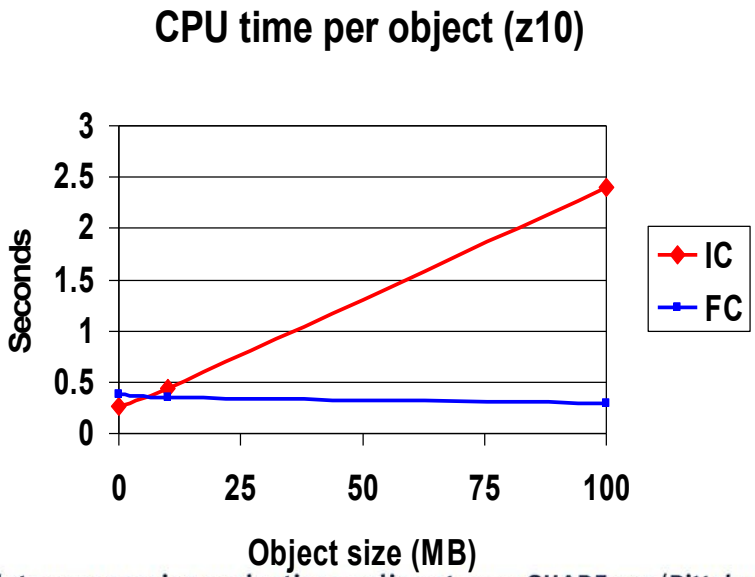
- Run utilities only when necessary and not on fixed schedules
- Information on the current status of all objects is contained in Real-Time Statistics (RTS) tables
- DSNACCOR/DSNACCOX apply our suggested thresholds and formulas against a list of objects and recommend utility actions
  - DSNACCOX in V9 NFM has improved RTS exploitation and recommendations
  - Use RESTRICT option to get restricted/advisory states also
- Leverage the ability to invoke utilities programmatically via stored procedures
  - DSNUTILU for UNICODE parameters
  - DSNUTILS for EBCDIC parameters (deprecated in DB2 10)
- Rich application logic can control what is run and when
- Refer to the DB2 Utility Guide and Reference Appendix B and samples
- Use LISTDEFs and TEMPLATEs for further simplification
- Consider automation tools for simplified automation, improved efficiency & automatic exploitation of new features

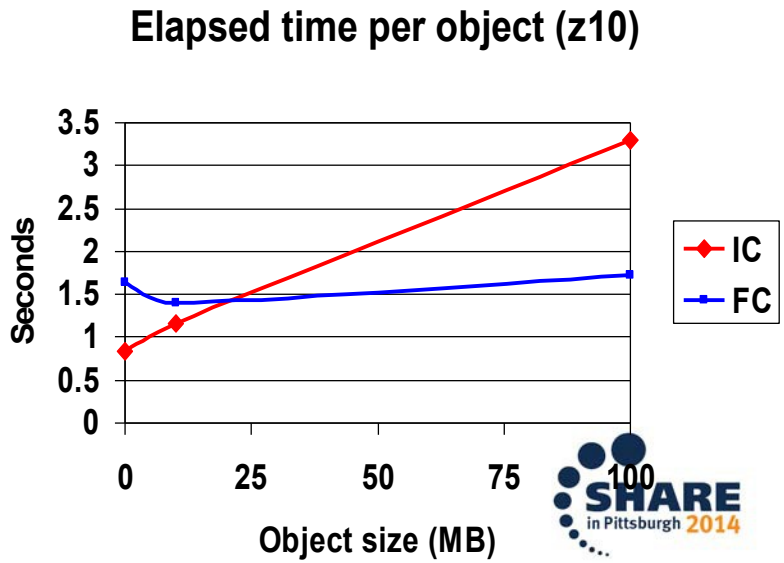
COPY

- SHRLEVEL CHANGE unless consistent copies are essential, in DB2 10 FLASHCOPY CONSISTENT
- Use PARALLEL keyword to exploit parallelism and STACK when writing to tape
- Consider OPTIONS EVENT(ITEMERROR,SKIP)
  - Sets UTRW state only for duration of copy of individual page set
  - But increases COPY overhead
    - Serialization required for each page set on the fly
- Consider taking incremental copies and using MERGECOPY
  - MERGECOPY marks relevant page set UTRW
- Copy indexes on large, critical tables
  - Particularly if rarely or never updated
  - Only drawback – increase in SYSLGRNX & SYSCOPY recording
  - Automatically included in MODIFY RECOVERY
- Consider CONCURRENT COPY
  - Can reduce CPU & elapsed time
  - Uses DFDSS backup/restore
  - Prohibits use of DSN1COPY & UNLOAD from copy

FlashCopy image copies

- Dataset-level Flashcopy support delivered in DB2 10
  - COPY, RECOVER, REORG, LOAD, REBUILD INDEX, REORG INDEX
  - New zparms & utility parms to govern
  - Virtually eliminate CPU & elapsed time for large page sets
  - Create transaction-consistent image copies from COPY SHRLEVEL CHANGE
  - Create partition-level inline image copies from REORG





FlashCopy considerations

- Can provide significant elapsed & CPU savings
  - Slightly higher setup cost so may see increased elapsed time for very small datasets
- RECOVER will work even though background copy not complete
  - Relationship will be broken & then slow restore of copied tracks
- No incremental copy permitted after FlashCopy
  - Incremental copy requests will be converted to full
- Ensure that data resides on FlashCopy-enabled DASD to avoid slow DFDSS copy
  - REORG with only FlashCopy will leave object in copy-pending if FlashCopy cannot be taken
- FlashCopies cannot have GDGs as a target
- HSM migrated data sets not supported by RECOVER
  - If FlashCopy dataset is migrated then it will be skipped
- Do not create transaction-consistent image copies in DB2 10 unnecessarily
  - DB2 10 allows creation of consistent image copies from COPY SHRLEVEL CHANGE
  - Can result in longer recovery time

BACKUP/RESTORE SYSTEM

- After initial setup of copy pools or if volume topography has changed, execute HSM FRBACKUP CP with the PREPARE option
  - Performs validation
  - Maps source and target volume pairs
- For SLBs dumped to tape
  - Use BACKUP SYSTEM to create the SLB on disk then monitor background copy completion with HSM QUERY CP.
  - When background copy is complete, then use BACKUP SYSTEM DUMPONLY to dump the SLB to tape
  - Avoids on demand background copy from the source volumes to the target volumes which can occur during the dump to tape if the dump is requested before the background copy is complete
- BACKUP SYSTEM batch job
  - Add a jobstep to issue HSM QUERY ACTIVE - indicates that HSM is responding before the BACKUP SYSTEM jobstep is executed.
  - Add jobsteps to Issue HSM LIST CP before and after the BACKUP SYSTEM has completed
- Consider adding automation to monitor for HSM ARC1802I (FAST REPLICATION BACKUP HAS COMPLETED) message during BACKUP SYSTEM execution to create SLB on disk. Issue an alert for action if not received within x minutes, where x is determined by the customer based on their production environment.

Backup solutions

Multiple options:

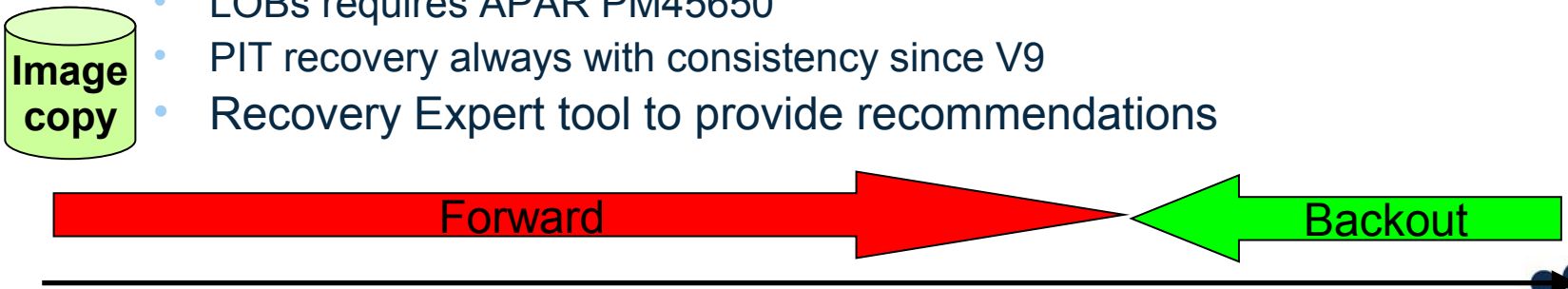
- BACKUP SYSTEM
  - Volume-level FlashCopy
  - Significant DASD investment required
  - Can be complex to set up & administer, but invocation simple
  - Must understand any limitations that currently exist
    - E.g. DASD mirroring issues, dataset movement issues, etc.
- Sequential image copies
  - Tried and trusted solution since V1.1
- Other external backups, such as volume-level backups, DSN1COPY
  - Outside of DB2's control
  - Requires careful management and co-ordination
- FlashCopy in DB2 10
  - Massive CPU, ET, resource reduction on z
- Choice is dependent on environment and requirements, all options will continue to be supported

RECOVER

- For recovery of many objects, prioritize based on critical apps
- Maximize exploitation of parallel restore and Fast Log Apply
  - Recover multiple objects in a list in parallel but ideally <100
  - Avoid running more than 10 RECOVER jobs per member prior to V10
  - Limit increased to 51 RECOVER jobs per member in DB2 10 with PM31641
  - New APAR PI07694 gives FLA support for index log processing
- Copy indexes and include in recovery list, particularly for PIT recovery
- Split off page sets that are not updated and recover separately
- For PIT recovery, include whole RI set in same RECOVER statement.
- For PIT recovery, include base and aux objects in same RECOVER statement
  - V10 enforces this via new keyword VERIFYSET

RECOVER options in DB2 10

- VERIFYSET option to fail PIT recovery if entire set not included
  - Base, LOB, XML & history objects
- ENFORCE NO option to avoid CHKP/ACHKP on PIT recovery of subset of set
  - Improved performance due to avoidance of set checking (RI, aux)
- Fast recovery to point in time through new BACKOUT option
  - Include indexes in RECOVER list to avoid the need to rebuild them
  - Indexes must be COPY YES
    - No image copy required though
  - LOBs requires APAR PM45650
  - PIT recovery always with consistency since V9
  - Recovery Expert tool to provide recommendations



QUIESCE & MODIFY RECOVERY

- QUIESCE
  - Do you still need it in V9 with PIT recovery with consistency?
    - If you want an LRSN or RBA marked in SYSCOPY, run QUIESCE on DSNDB06.SYSEBCDC
  - Use WRITE NO unless you absolutely must have pages written out
- MODIFY RECOVERY
  - Base your MODIFY strategy on your backup strategy and not vice versa
  - Consider running every time a backup is taken or at least weekly
  - REORG SYSLGRNX regularly for optimal performance and minimal MODIFY impact on system
  - DB2 9 has RETAIN LAST n, GDGLIMIT and BSDS options
    - Careful with GDGLIMIT if you use multiple GDGs for a single object
  - Will not clean up “orphan” entries, but no more orphans are created in DB2 10
  - Run MODIFY to delete recovery information from prior to a REORG that materializes row alterations
    - Makes subsequent REORGs more efficient

LOAD/UNLOAD

- Use TEMPLATE with &PA or &PART to drive partition parallelism for UNLOAD
- UNLOAD from image copy should use SYSTEMPAGES YES image copy
- Run LOAD with LOG NO, REUSE, KEEPDICTIONARY if possible
- Use SORTDEVT to drive parallel index build
- Allocate inline copy data sets to DASD
- Split up input data set and drive LOAD partition parallelism in a single job
- Use SORTNUM elimination
- Specify NUMRECS if input is on tape or variable length
- If loading partitioned table with single input data set, presort data in clustering (partitioning) key order
  - PRESORT option in Utility Enhancement Tool
- For LOAD REPLACE, consider loading into a “clone” then renaming tables Consider using USS named pipes
- Use FORMAT INTERNAL, PRESORTED or INDEXDEFER if possible
- DB2 10: Do not expect LOAD into hashed tables to perform as well as non-hashed
  - UET provides PRESORT option to significantly improve LOAD-to-hash performance

Example LOAD job

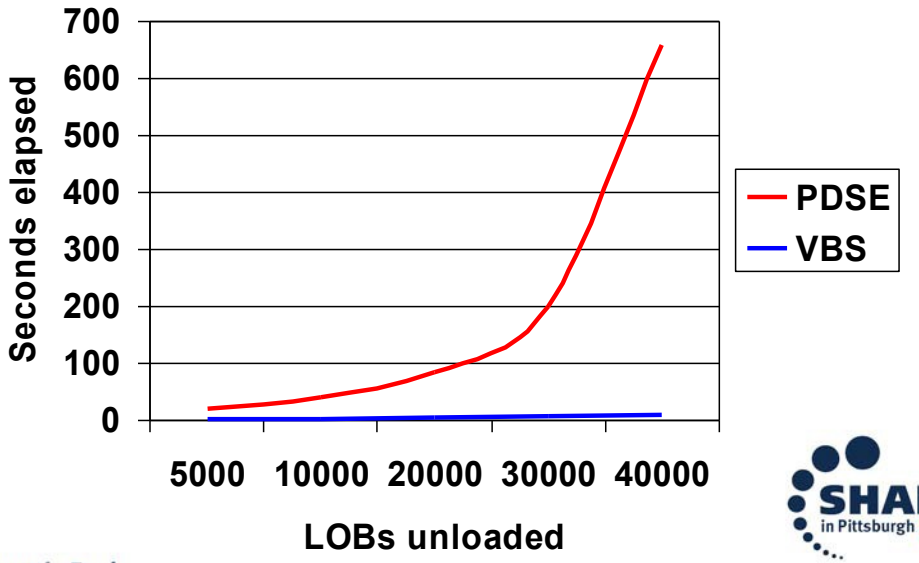
```
LOAD REPLACE REUSE LOG NO FORMAT INTERNAL
     KEEPDICTIONARY SORTDEVT SYSDA
  INTO TABLE tb PART 1 INDDN sysrec1 NUMRECS n
           ….
  INTO TABLE tb PART 2 INDDN sysrec2 NUMRECS n
           ….
  INTO TABLE tb PART 3 INDDN sysrec3 NUMRECS n
```

Look for in job output:

- DSNU395I message for parallel index build active
- DSNU364I message indicating partition parallelism

LOAD/UNLOAD

- Consider whether you want UNLOAD utility or HPU
  - High Performance Unload is a separately chargeable tool
    - Comparable or less elapsed time
    - Less CPU
    - Permits SQL interface
    - Permits unload from page set on DASD
- LOB/XML processing for LOAD/UNLOAD
  - HFS still performs better in elapsed time
  - Note limit of 522,239 members in a PDS/E
  - DB2 10 use VBS format SPANNED YES option for improved performance.



REORG

- SHRLEVEL CHANGE for availability
- If an NPI exists then concurrent REORGs of parts in the same table space is not permitted in V9 until PK87762
  - Support of REORG PART 1,3,7
- REORG of a small set of parts with an NPI may take longer in V9 than V8
  - Entire NPI is shadowed
  - Particularly if NPI is disorganized since keys unloaded in order
  - Performance improved in V10 with index list prefetch
  - Apply PM55051 for SORTNPSI option to improve performance
- Parallelism in V9 on unload/reload/logapply means multi-part REORG is more efficient, faster and the log phase is much better at keeping up with logging rates
  - So REORG multiple parts in the same REORG statement
- Create mapping table on PBG table space to support large table spaces (PM58177 V9)

REORG

- Move to DB2 10 to reduce application outage with inline statistics
- If using DISCARD, NOPAD performs better than the default setting of PAD
- SHRLEVEL CHANGE main recommendations
  - Use DRAIN ALL to minimize application impact
  - Use TIMEOUT TERM to free up objects on timeouts
  - (DRAIN_WAIT+MAXRO)<(IRLMRWT -5 or 10 seconds)
    - Avoid application timeouts
    - But don’t set MAXRO too low
  - RETRY=6
  - RETRY_WAIT=DRAIN_WAIT*RETRY
  - Consider MAXRO DEFER & -ALTER UTILITY if REORG needs to complete in short window

REORG

- REORG of LOB page sets
  - Get to V9 and use SHRLEVEL REFERENCE, available in V9 CM
  - SHRLEVEL CHANGE provided in V10
  - Move away from SHRLEVEL NONE before getting to V10 NFM
    - Noop in DB2 10, error in DB2 11
- LOB table space may be left in copy-pending
  - If REORG of base pulls in LOBs and no template exists for image copy
  - If REORG of multiple parts of a PBG and PBG grows during log phase
  - Both scenarios can be avoided by using inline FlashCopy and specifying template either in zparm or on REORG statement
- DB2 10 - May have difficulty determining size of inline copy for LOB table spaces, e.g. for REBALANCE

REORG INDEX vs. REBUILD INDEX

- REORG is not a substitute for REBUILD, but REBUILD could be a substitute for REORG
- Use REBUILD INDEX SHRLEVEL CHANGE provided in V9
  - Excellent for create of new non-unique indexes and for indexes that are broken or already in RBDP
  - Does not operate against a shadow, so will set RBDP if not already set
  - Note SHRLEVEL CHANGE means data is RW, not index – index is RBDP
- REORG INDEX operates against a shadow – better availability than REBUILD INDEX
- REBUILD can be faster in V9, particularly if index is disorganized
- REORG INDEX performance improvement in V10 due to prefetch
- Summary:
  - If you need to REBUILD then REBUILD
  - If you need to REORG on V9, and you can tolerate the index being RBDP, then REBUILD may be quicker
  - If you need to REORG on V10 then REORG

RUNSTATS

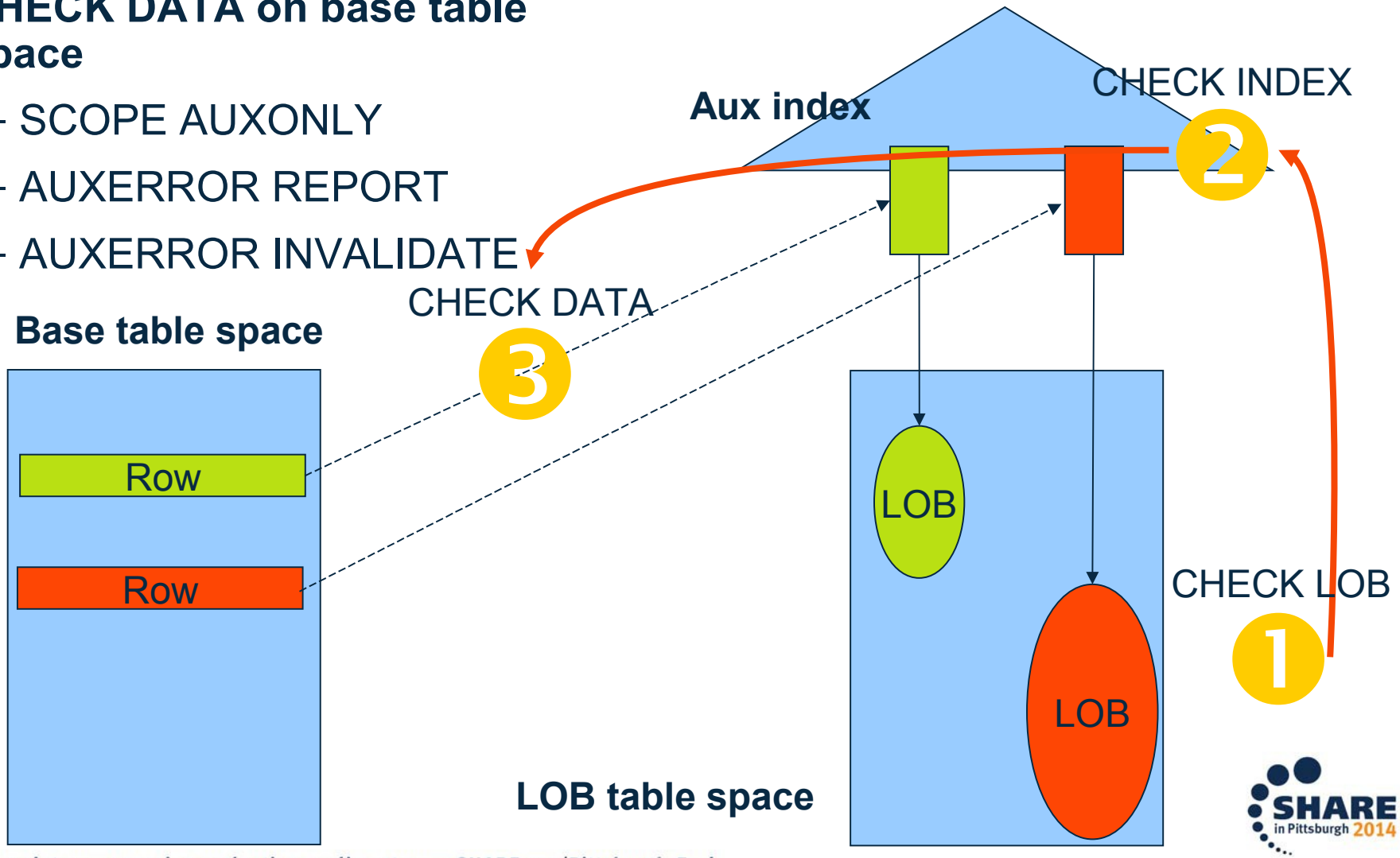
- Do not use RUNSTATS to gather space statistics
  - rely on RTS
- Do not gather unnecessary stats
- Use inline stats where possible rather than RUNSTATS.
  - zIIP offload for inline stats in DB2 11
- Specify KEYCARD prior to DB2 10
  - Index cardinality stats are cheap to collect and heavily relied upon by optimizer
- Don’t bother running RUNSTATS on LOB table spaces
  - RTS contains all the information you need
- Use RUNSTATS PROFILE support for simplified usage in DB2 10.
- Use TABLESAMPLE SYSTEM AUTO for improved CPU & elapsed time
- DB2 11 allows support for colgroups with inline statistics

CHECK Utilities

- If no applications are impacted and you just want to check consistency, then:
  - Either run CHECK… SHRLEVEL CHANGE
  - Or run standard CHECK utility but be ready with REPAIR to reset CHKP/ACHKP states
  - Or could consider running SQL ISO(UR) instead
- If running SHRLEVEL CHANGE:
  - It will not reset CHKP or ACHKP states, nor will it set them
  - Make sure the page set is on FlashCopy-enabled DASD
    - If not then you'll get a slow copy with the data in UTRO
  - Set CHECK_FASTREPLICATION zparm to REQUIRED
  - If using DASD mirroring or BACKUP SYSTEM then use ZPARM UTIL_TEMP_STORCLAS to prevent impacting either
    - Refer to PK41711

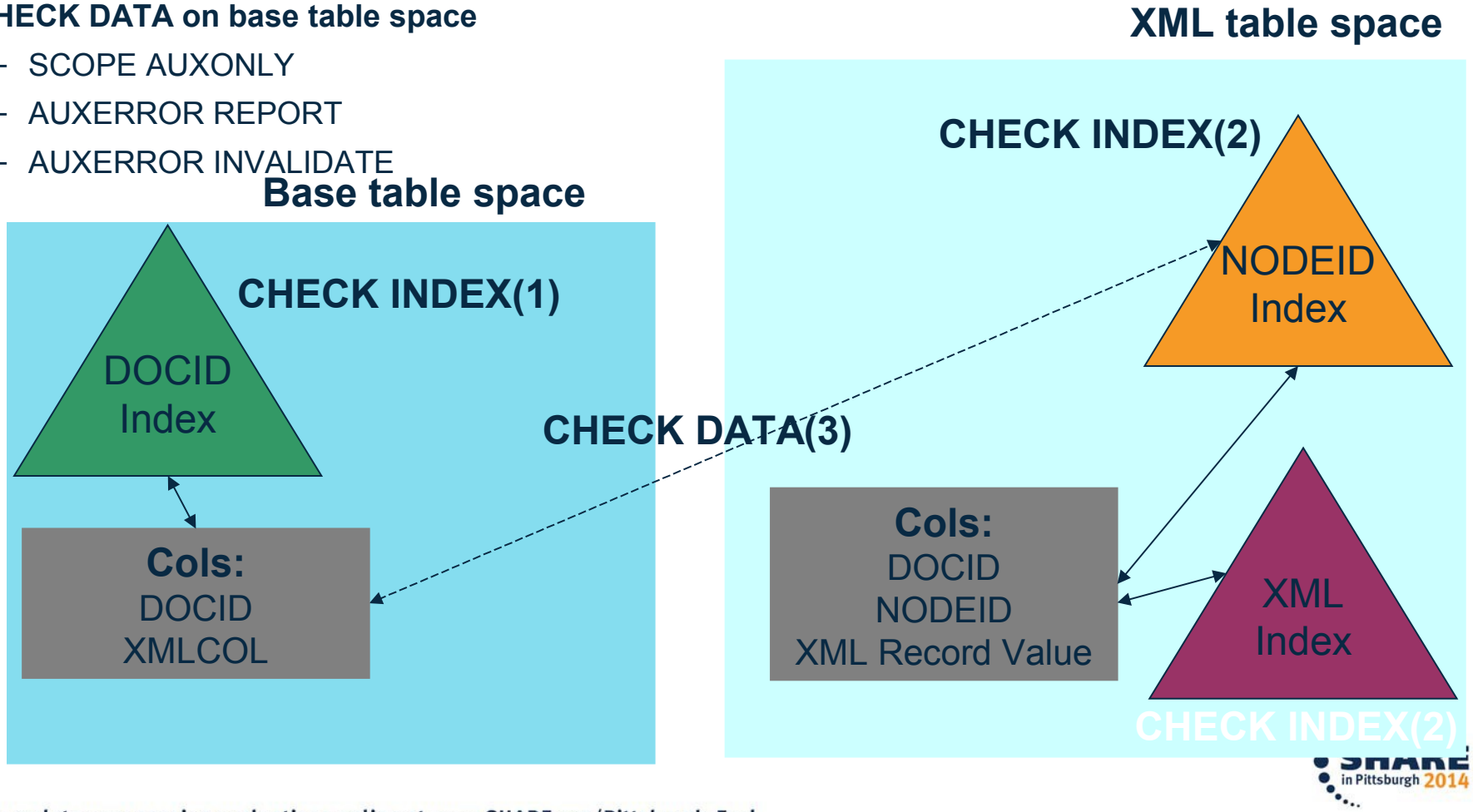
CHECK Utilities

- **CHECK DATA on base table space**
  - SCOPE AUXONLY
  - AUXERROR REPORT
  - AUXERROR INVALIDATE



CHECK Utilities

- **CHECK INDEX on DOCID, NODEID, XML indexes**
- **CHECK DATA on base table space**
  - SCOPE AUXONLY
  - AUXERROR REPORT
  - AUXERROR INVALIDATE



DSN1COPY

- DSN1COPY is an essential part of the utilities portfolio
- DSN1COPY runs standalone and cannot ensure that data matches definition at target
- All target data sets must be pre-allocated for multi-piece table spaces
- Areas to watch out for:
  - **BRF-RRF mismatch**
    - Tolerated by SQL, and by utilities with PM40646
  - **Data definition changes, e.g. columns added**
    - Use REPAIR VERSIONS at target site
    - REORG at source before DSN1COPY, particularly if no updates since first ALTER
      - Problem if first ALTER of table creates new version but no data inserted so no system page information and then DSN1COPY to target
  - **Different table space types or different segsizes**
    - Not policed, abends will occur, expect to police in future and fail page set open

DSN1COPY

- Areas to watch out for:
  - **XML**
    - Data-dependent information kept in catalog table XMLSTRINGS
      - Cannot DSN1COPY XML table space from one subsystem/group to another
      - DSN1COPY within a subsystem/group is fine
      - Solution is UNLOAD/LOAD
    - DOCID is a sequence generated by DB2 – DSN1COPY to a new target where the DOCID is lower will result in -803 on insert because DB2 generates a value of n but n already exists in the table. ALTER of the sequence isn't allowed for DB2-generated sequences.
      - Use dummy inserts to increase the sequence value

Recent News

- Part-level REORG NPSI insert performance improvement
  - PM87403 (V9)
  - 100m row table, 6 indexes
  - LOAD RESUME – 66% CPU reduction, 30% ET reduction
  - REORG PART 9 – 45% CPU reduction, 26% ET reduction
- Fast log apply for faster index recovery
  - PI07694 (V9)
- Retrofit REORG SWITCH phase performance to V10
  - PI09303 (V10)
  - Cut SWITCH phase duration by 90%
- Retrofit REORG REBALANCE SHRLEVEL CHANGE to V10
  - PI11839 (V10)
- LOAD REPLACE SHRLEVEL REFERENCE, LOAD RESUME SHRLEVEL REFERENCE, LOAD prevalidation
  - Delivered in Utilities Enhancement Tool / Utilities Solution Pack
  - PI04864

Summary

- Stay reasonably current on versions and maintenance
- Understand what this gives you in terms of utility capability
- Revisit your existing utility jobs to benefit from new options
- Use automation to only run those utilities that need to be run & take advantage of new features


Thank
You