DB2 11 for z/OS Application Functionality (Check out these New Features)

Randy Ebersole
IBM
ebersole@us.ibm.com

Please note

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.

About This Presentation

- The main intent of this session is to highlight some of the new application related features of DB2 11. We want to introduce some of the new application functionality and get you thinking more about DB2 11!

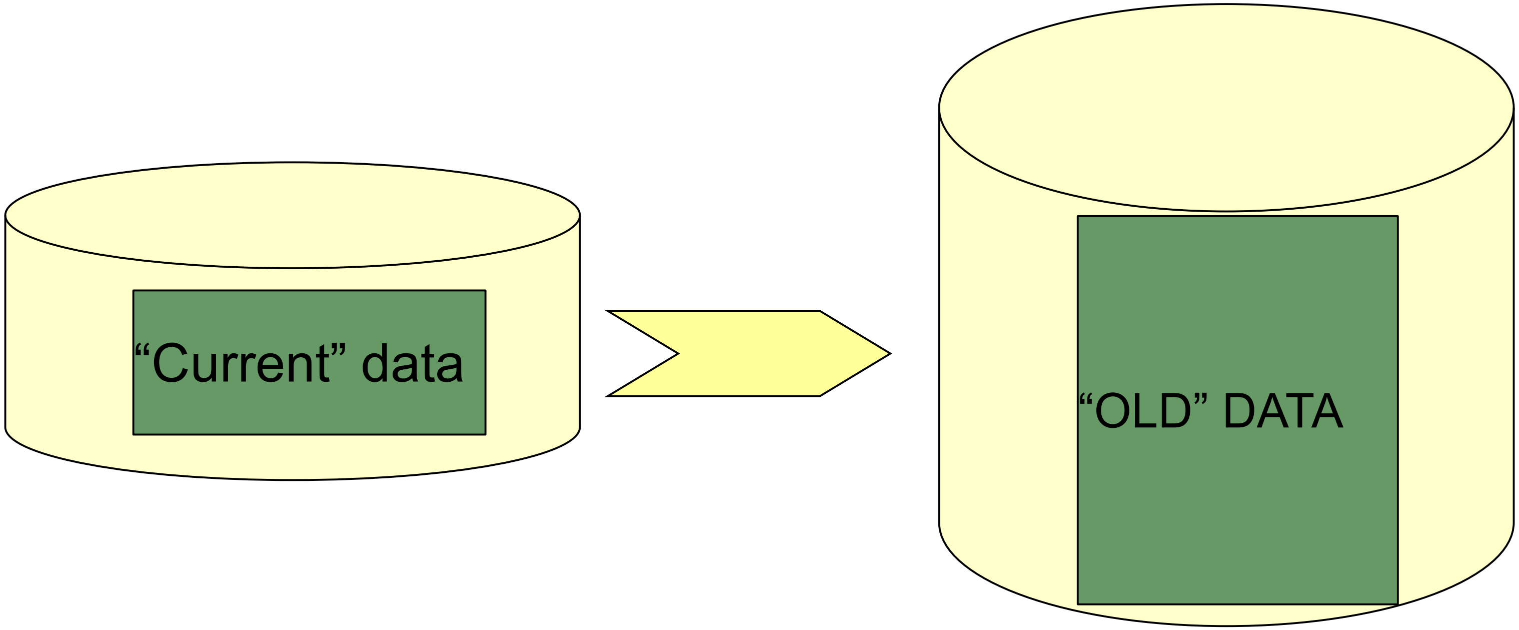
Topics for Review

- Archive Tranparency
- The Optimizer and RUNSTATS
- Autonomous Transaction Management
- Pseudo Deleted Index Entry Cleanup
- Further exploitation of RELEASE(DEALLOCATE)

Managing large amounts of data for a table

- Managing tables that contain a large amount of data is a common problem
  - performance is a challenge
  - Many times, a large portion of the data is commonly accessed
- Many times application design / database design has been done to manage this challenge
  - Move “old” data to another table
  - Keep “current” data separate

A picture of this …..

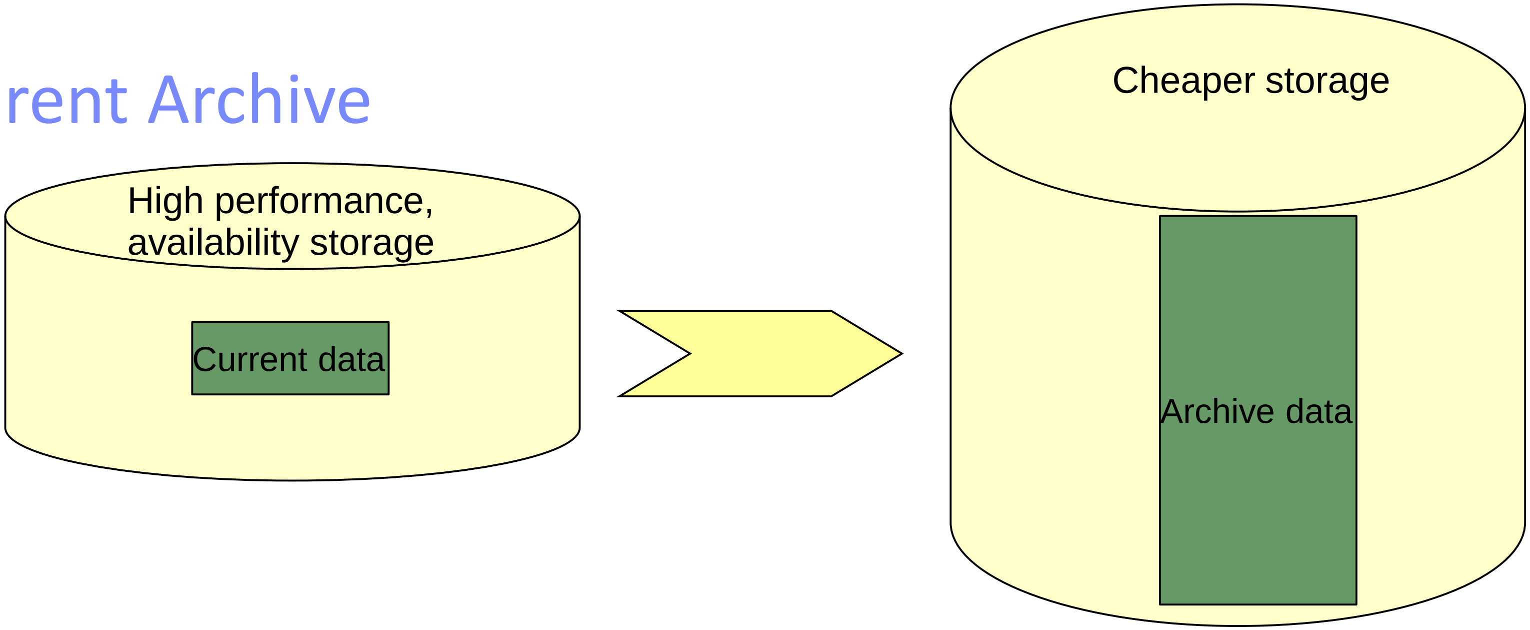


- Applications are designed to handle the access to the data
- Still a challenge to manage and control

What is Archive Transparency in DB2 11 ?

- Data archive management from OLTP current table to “old” data table
- You can include or exclude archive table data access without a need to change the SQL statement
- Control the scope of the query with a global variable
- archive-enabled table and associated archive table

Transparent Archive

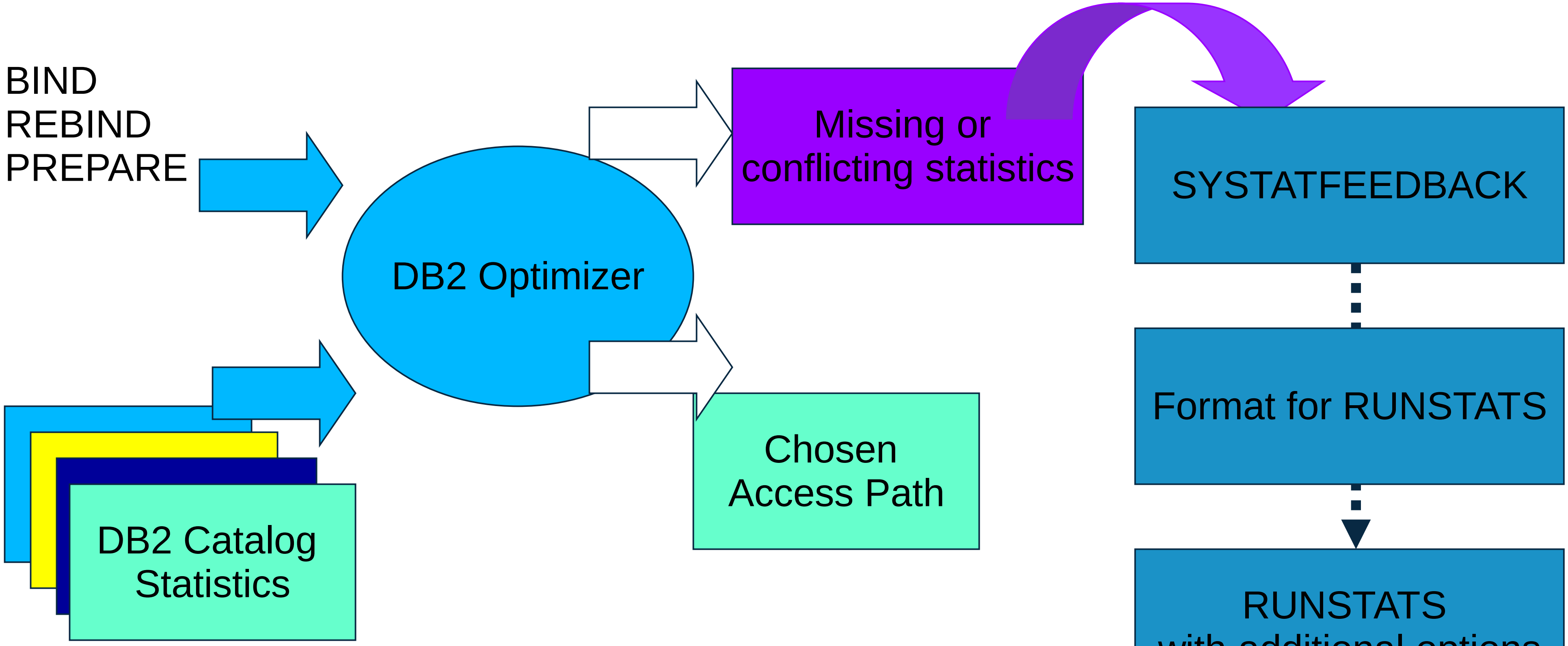


- Applications can query current + archive with no SQL changes
  - By default, data is retrieved from base table only, as usual
  - Set a new global variable when archive data is desired
  - DB2 automatically converts SQL to UNION ALL via dynamic plan switching technique (high performance)
- Archiving process is user-controlled
- Move_To_Archive global variable allows DELETEs to be automatically archived
- Leverages DB2 10 temporal constructs for archiving use cases
- Future potential for more IDAA synergy

Optimizer input for RUNSTATS

- The better the RUNSTATS, the more effective the optimizer
- The standard or default statistics are commonly used
  - RUNSTATS TABLE(ALL) INDEX(ALL) KEYCARD
- Additional statistics will typically help queries perform better
- The challenge is in determining what RUNSTATS to collect

The flow of Optimizer input for RUNSTATS



BIND
REBIND
PREPARE

DB2 Optimizer

Missing or conflicting statistics

SYSTATFEEDBACK

Format for RUNSTATS

RUNSTATS with additional options

Chosen Access Path

DB2 Catalog Statistics

DB2 11 and DB2 Optimizer Feedback

- During access path selection process, the optimizer identifies missing or conflicting statistics
  - Every BIND, REBIND or PREPARE
    - Writes recommendations to SYSIBM.SYSSTATFEEDBACK asynchronously
  - DB2 also provides statistics recommendations on EXPLAIN
    - Populates DSN_STAT_FEEDBACK synchronously
- Information from SYSSTATFEEDBACK or DSN_STAT_FEEDBACK can be used to generate input to RUNSTATS
  - Contents must be interpreted, not directly used by RUNSTATS
  - Requires DBA or tooling to convert to RUNSTATS input

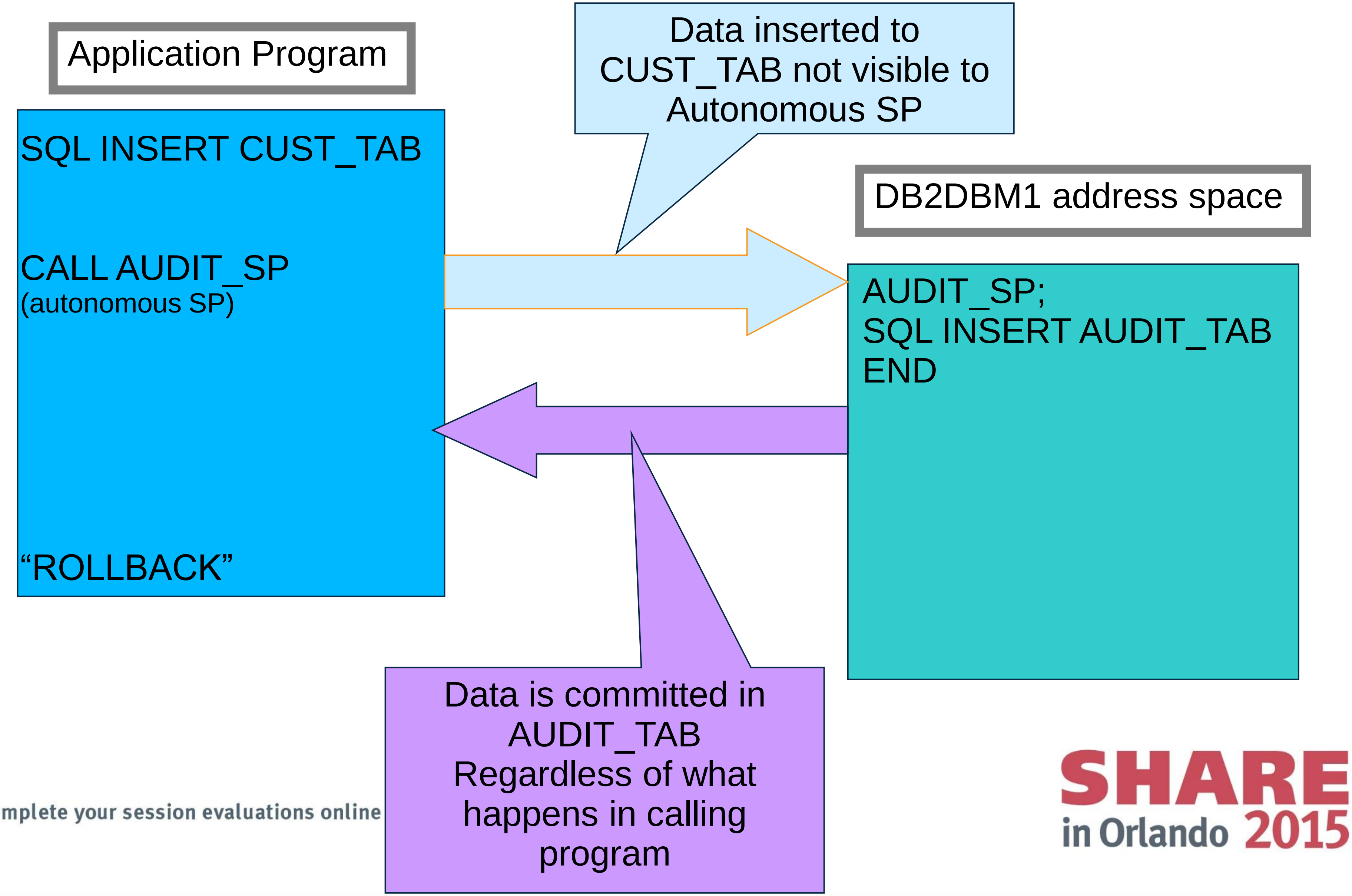
Autonomous Transactions

- Useful for event logging, Audit data, special data management
- DB2 executes native SQL procedure in a unit of work that is independent of the calling program
    - May perform SQL, COMMITs, and ROLLBACK
    - No uncommitted changes from it’s caller are available
    - Autonomous SP and caller of SP do not share locks
- COMMIT is done when autonomous SP completes successfully, (SQLCODE >=0)

Definition of Autonomous Transactions

- Specify AUTONOMOUS keyword on ALTER or CREATE PROCEDURE:
  - DYNAMIC RESULT SETS 0 must be in effect.
  - Stored procedure parameters must not be defined as:
    - A LOB type
    - The XML data type
    - A distinct data type that is based on a LOB or XML value
    - An array type that is defined with array elements that are a LOB type
    - A value must not be assigned to a global variable when an autonomous procedure is executing.

Autonomous Transaction Basic Flow



Application Program

SQL INSERT CUST_TAB

CALL AUDIT_SP
(autonomous SP)

"ROLLBACK"

Data inserted to CUST_TAB not visible to Autonomous SP

DB2DBM1 address space

AUDIT_SP;
SQL INSERT AUDIT_TAB
END

Data is committed in AUDIT_TAB Regardless of what happens in calling program

Monitoring Autonomous Transactions

-DISPLAY THREAD(*) command

```
12.34.56 DB2P DISPLAY THREAD(*)
12.34.56 STC00090 DSNV401I DB2P DISPLAY THREAD REPORT
  FOLLOWS -
12.34.56 STC00090 DSNV402I DB2P ACTIVE THREADS -
NAME     ST A   REQ  ID     AUTHID   PLAN     ASID  TOKEN
BATCH    SP *     1  APPL   USERA    PLANA    0022     31
         AT *   641  APPL   USERA    PLANA    002A     13
```

Pseudo deleted index entries

- When table rows are deleted, a pseudo delete operation is done for the index RIDs
  - Index RIDS are “flagged” as deleted
  - unless the delete process has locked the entire table
- Impact of pseudo deleted index entries
  - Index size grows with increasing number of pseudo-deleted index entries
  - More getpages and lock requests required
  - Increased CPU cost and possibly longer elapsed times for access via index search

Pseudo deleted index entry clean up

- Prior to DB2 11, clean up of pseudo deleted index entries were part of main line processing and does impact application with increased costs
  - REORG INDEX takes care of pseudo-empty index pages and pseudo-deleted entries
- DB2 11 introduces additional action in the clean up of the pseudo deleted index entries
- Clean up pseudo-empty index pages and pseudo deleted index entries
  - Could reduce the size of indexes
  - Could improve SQL performance
  - Could reduce the need to run the REORG INDEX utility

DB2 11 Pseudo deleted index entry Process

- In Conversion Mode
- Automated cleanup of pseudo-deleted index entries in index leaf pages and of pseudo-empty index pages
- Mainline processing continues to do cleanup
- System tasks, running as enclave SRBs and are zIIP eligible are utilized
- Utilizes RTS information to identify indexes for cleanup
- Clean up work is only down for an index that is already open for “update”
- Page consistency is maintained

DB2 11 Pseudo deleted index management

- Cleanup threads are controlled
  - System parameter INDEX_CLEANUP_THREADS (0-128)
  - 0 disables index cleanup
  - Default is 10
  - Child clean up threads CORRELATION ID = 014.IDAEMKxx
- Specific index clean is done via SYSIBM.SYSINDEXCLEANUP table
  - indexes can be enabled or disabled for cleanup
  - Can specify:
    - Name of databases and indexes
    - Cleanup enabled or disabled
    - Day of week or day of month
    - Start time and end time

DB2 11 Pseudo deleted index monitoring

- IFCID 377 - tracks cleanup at the index page level
  - DBID, PSID, partition number, page number
  - pseudo empty page OR pseudo-deleted entries
  - Includes count of pseudo-deleted entries cleaned up
- **Not included in any trace class**
- RECORD TRACE used for reporting

Thread Management – Breaking in

- DB2 10 provided a performance alternative with the use of RELEASE(DEALLOCATE)
  - This creates a challenge in executing DDL, REORG, BIND)
- DB2 11 delivers a break-in mechanism for persistent RELEASE(DEALLOCATE) threads
  - Persistent thread automatically detects operations that would like to break in
  - If detected, then RELEASE(DEALLOCATE) will behave like RELEASE(COMMIT)
  - zPARM PKGREL_COMMIT=YES must be set (parameter is online changeable)
    - Default is YES
- The release of the resources after COMMIT/ROLLBACK occurs only if other DB2 operations (e.g. Bind, DDL, online REORG) are waiting for exclusive control to this package.
- Packages resume normal RELEASE(DEALLOCATE) behavior after the break-in operation completes

Summary

- DB2 11 once again provides a wealth of new application functionality features
- Many features provide performance benefits like CPU reduction
- Many features provide options to handle application design

Randy Ebersole – *IBM - ebersole@us.ibm.com*

Typical Utilization for Servers

Windows: 5-10%   Unix: 10-20%   **System z: 85-100%**

System z can help **reduce** your floor space up to **75%-85%** in the data center

Questions

System z can lower your total cost of ownership, requiring **as little as 30%** of the power of a distributed server farm running equivalent workloads

The cost of storage is typically **three times more** in distributed environments