#SHAREorg

SHARE

DB2 10 for z/OS Migration Planning Workshop - Parts 1 and 2

John Iczkovits
IBM

August 15, 2013
Session Numbers 13985 and 13986

SHARE in Boston

Disclaimer and Trademarks

Information contained in this material has not been submitted to any formal IBM review and is distributed on "as is" basis without any warranty either expressed or implied. Measurements data have been obtained in laboratory environment. Information in this presentation about IBM's future plans reflect current thinking and is subject to change at IBM's business discretion. You should not rely on such information to make business plans. The use of this information is a customer responsibility.

IBM MAY HAVE PATENTS OR PENDING PATENT APPLICATIONS COVERING SUBJECT MATTER IN THIS DOCUMENT. THE FURNISHING OF THIS DOCUMENT DOES NOT IMPLY GIVING LICENSE TO THESE PATENTS.

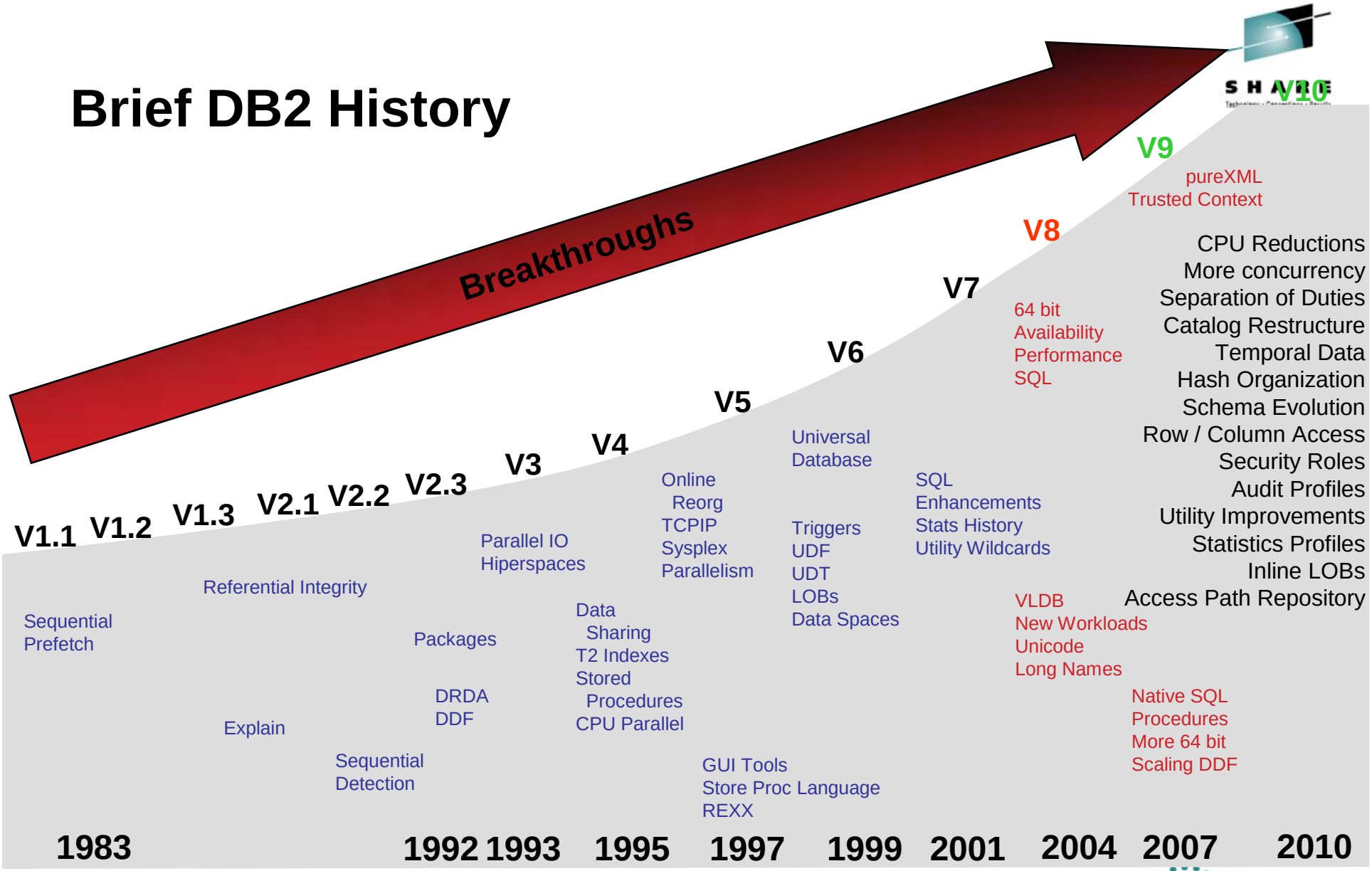
TRADEMARKS: THE FOLLOWING TERMS ARE TRADEMARKS OR ® REGISTERED TRADEMARKS OF THE IBM CORPORATION IN THE UNITED STATES AND/OR OTHER COUNTRIES: AIX, DATABASE 2, DB2, Enterprise Storage Server, FICON, FlashCopy, Netfinity, RISC, RISC SYSTEM/6000, System i, System p, System x, System z, IBM, Lotus, NOTES, WebSphere, z/Architecture, z/OS, zSeries

The FOLLOWING TERMS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF THE MICROSOFT CORPORATION IN THE UNITED STATES AND/OR OTHER COUNTRIES: MICROSOFT, WINDOWS, WINDOWS NT, ODBC, WINDOWS 95

For additional information see ibm.com/legal/copytrade.phtml

DB2 10 for z/OS

Brief DB2 History



DB2 10 Sample Performance Improvements

- DB2 10 CMx with REBIND
  - Run time CPU reductions 5% - 10%
  - 1 MB page size 0% - 4% z10, z196
  - Page fix buffers 0% - 8% since V8
  - Release deallocate 0% - 15% short trans, batch
  - Virtual storage constraints 0% - 5% memory, latches
  - Data sharing fewer members 1% for each 2 members
  - Insert 0% - 40% high volume insert
  - Predicate evaluation 0% - 60% complex predicates
  - Increased use of zIIP 0% - 3% IO, RUNSTATS, parallelism
  - Utilities (from V8) 3% - 20%
- DB2 10 NFM
  - Improved dynamic SQL cache 0% - 20% literals
  - Access: hash, index include 0% - 5% access improved

DB2 10 Performance...

- A beta customer shared some numbers @ IOD 2010

JDBC Distributed application

DB2 Data Sharing

	DB2 10 vs DB2 9
Chargeable Application + DB2 CPU	-40 %
Chargeable DB2 CPU	-40 %
Total Application + DB2 CPU	- 35 %
Total DB2 CPU	- 36 %
Application + DB2 Elapsed Time	-26 %
DB2 Elapsed Time	-34 %
DB2 Suspension Time	-23 %
Lock / Latch Suspension Time	-88 %
Page Latch Suspension Time	-67 %
Log I/O Suspension Time	-77 %

	DB2 10 vs DB2 9
Total Application + DB2 CPU	- 4.64 %
Total DB2 CPU	- 6.45 %
Application + DB2 Elapsed Time	- 14%
DB2 Elapsed Time	- 15 %
DB2 Suspension Time	-24%
Lock / Latch Suspension Time	- 27 %
Log I/O Suspension Time	- 18 %

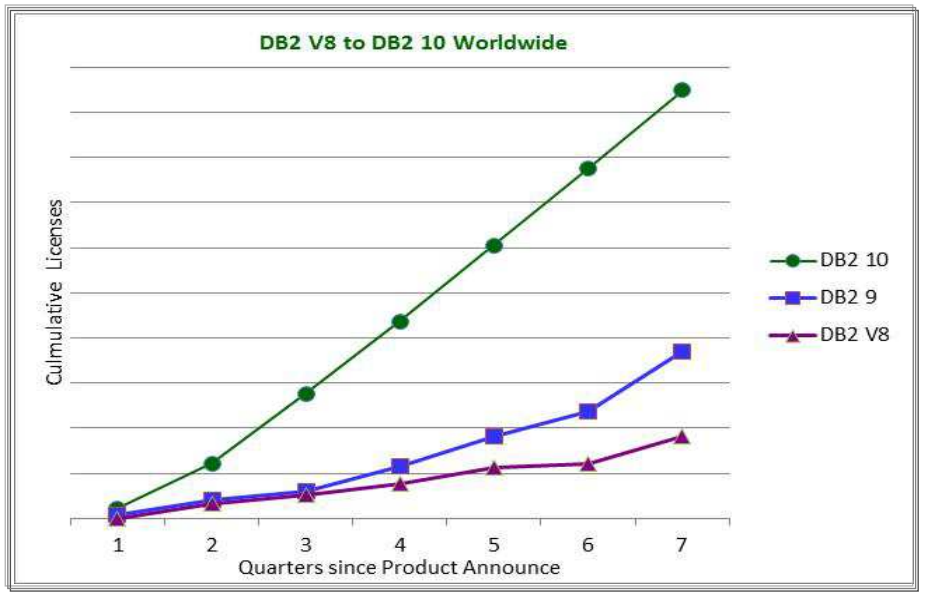
n

20 concurrent LOADs
10M rows per job
SHRLEVEL NONE

DB2 10 for z/OS Snapshot

- **Fastest uptake**
  - **+2x customers vs. V9**
  - **+2.5x licenses vs. V9**
  - **25% coming from DB2 V8**

- **Adoption Driven by:**
  - **Performance improvements without application changes**
  - **Virtual Storage Constraint relief for more threads**
  - **Security, RAS improvements**
  - **Bitemporal data**



DB2 z/OS Availability Summary

Version	PID	General Availability	Marketing Withdrawal	End of Service
5	5655-DB2	June 1997	December 2001	December 2002
6	5645-DB2	June 1999	June 2002	June 2005
7	5675-DB2	March 2001	March 2007	June 2008
8	5625-DB2	March 2004	September 2009	April 2012
9	5635-DB2	March 2007	December 2012	June 2014
10	5605-DB2	October 2010		
11	5615-DB2	ESP March 2013		

DB2 for z/OS Lifecycle

http://www.ibm.com/software/data/support/lifecycle/

IBM Software support lifecycle

DB2 for z/OS 10.1.0

The announcement letter dates below are US only. Information for other country announcements is available on the IBM Offering Information page.

Subscribe via RSS to Product Support Lifecycle site updates.

View lifecycle dates for other products.

Product information	License type[1] and lifecycle policy[2]	Lifecycle dates[3] and announcement letters	Last updated
DB2 for z/OS Version: 10.1.0 PID: 5605-DB2	License type: ICA Lifecycle policy: Standard	GA: 22-Oct-2010, 210-380	20-Oct-2010

Education / Planning

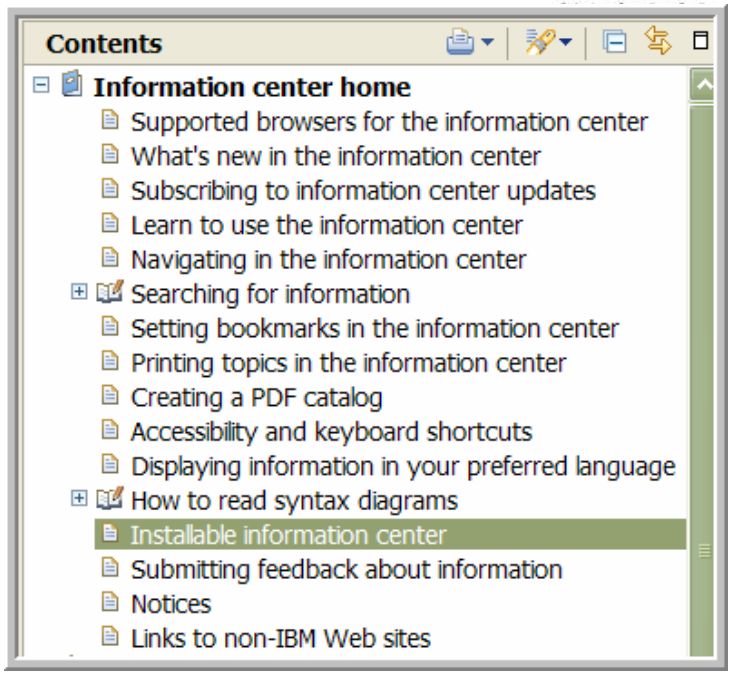
- TechNotes
  - See the link in the DB2 for z/OS landing page
- Planning Page
- DB2 Best Practices
- DB2 10 for z/OS Technical Overview (SG24-7892)
  - See Consolidated Checklist for more Redbook suggestions
- DB2 10 for z/OS Performance Topics
- The Value of DB2 for z/OS Hash access for OLTP transactions
- Security Functions with DB2 10 for z/OS
- DB2 for z/OS and List Prefetch Optimizer
- DB2 10 for z/OS and Query Acceleration
- DB2 11 for z/OS Early Ship Program Announcement

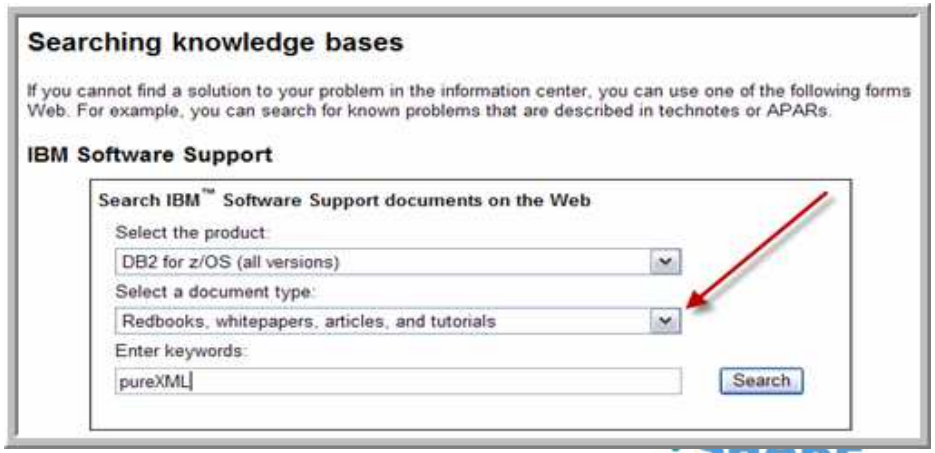
Education / Planning

- IBM Education
  - CV303 Doing skip-level migration? New Functions and Features of DB2 9 and 10 for z/OS (5 days)
  - CV311 DB2 for z/OS New Functions and Features in V10 (4 days)
  - CV312 DB2 for z/OS - New Features in Version 10 Workshop (5 days)
  - CV831 DB2 10 for z/OS Database Administration Part 1 (5 days)
  - CV851 DB2 10 for z/OS System Administration (5 days)
  - Coming Soon:
    - CV871 – DB2 10 Utilities Administration
  - Also available as Instructor Led Online / Virtual Classroom
  - Database Administration and System Administration Certification Crammer courses

Information Center

- Installable IC for local use
- Search capabilities across
  - Redbooks
  - Whitepapers
  - Technotes
  - APARs
- See the Consolidated Checklist (Migration section – Documentation)





DB2 9 for z/OS

Building the e-infrastructure

- ✓Continuous Availability
- ✓Systems Management, Security
- ✓Performance/Capacity
- ✓Connectivity
- ✓Productivity
- ✓Synergy: DB2 family & zSeries

= Total Cost of Ownership

Ready for e business

- –Base Pre-requisites:
  - –zSeries z800, z890, z900, z990, z9, z10 or later
  - –z/OS V1.7 or above
  - –Data Sharing:CFLEVEL 12
  - –Defined shared memory objects
  - –Satisfy the V8 requirements
- –Catalog changes:
  - –Additions for new features
  - –Including Real Time Statistics
- –DB2 Connect
  - –V8 FP13, V8.2 FP 6, V9 FP1
- –Migration Process:
  - –From Version 8 NFM only
  - –More flexibility switching between modes
    - CM, ENFM, NFM
  - –V8 and V9 data sharing coexistence in CM
  - –DSNTIJP9 provided for V8 with PK31841

DB2 9 Highlights

- Storage
  - 64 bit evolution
    - DIST Address Space
    - More above the bar
  - Prefetch Changes
  - Large Buffer Pool Management
  - Distributed Workload Improvements
  - Converged Temp Space
- pureXML
  - XML Publishing / Composition
  - XML Architecture
  - SQL/XML & XPath
  - XML Decomposition
  - XML Indexes
  - XML Schema
  - XML Utilities

DB2 9 Highlights

- SQL
  - MERGE
  - SELECT FROM
    - UPDATE
    - DELETE
    - MERGE
  - FETCH FIRST / ORDER BY in Subselect
  - ORDER OF
  - INTERSECT
  - EXCEPT
  - SKIP LOCKED DATA
  - TRUNCATE
  - Session Variables
  - Native SQL Stored Procedures
- Optimization
  - REOPT(AUTO)
  - Global Query Optimization
    - Subquery optimization
    - Virtual Tables
  - Page Range Screening
  - Histograms
  - Real-Time Statistics in Catalog
  - Access Path Stability
    - PLANMGMT
  - Sort Avoidance
    - Memory replacement for FETCH FIRST n ROWS ORDER BY...
  - REBIND PLAN...COLLID
    - Migrate DBRMs in Plans to Packages
  - Data Warehousing Improvements
    - Star Join improvements

DB2 9 Highlights

- Security
  - Trusted Context
  - Roles
  - Enterprise Identity Mapping
- Advanced Design
  - New Data Types
    - BIGINT, BINARY, VARBIN, DECFLOAT(16|32)
  - Implicitly Hidden Columns
  - Universal Table Space
    - Partitioned By Growth
    - Partitioned By Range
  - Reordered Row Format
- Optimistic Locking
- CLONE Tables
- NOT LOGGED Tables
- Table APPEND Option
- INSTEAD OF Triggers
- RENAME COLUMN, INDEX
- Rename Schema, Owner, VCAT
- SMS DDL Integration
- Implicit Database Support

DB2 9 Highlights

- LOBs
  - Network Flows
    - LOB Indicators / Streaming
    - FETCH CONTINUE
  - File Reference Variables
    - Extended to SQL
  - LOB Utility Improvements
    - REORG LOB SHRLEVEL REFERENCE
  - Locking Changes
- Indexing
  - Larger Page Sizes
  - Randomized Indexes
  - Online Index Rebuild
  - Asymmetric Page Splits
  - Index on Expression
  - Index Compression
  - DPSIs Improvements
  - XML Indexes

DB2 9 Highlights

- Data Sharing
  - Avoid Log Latch Contention
    - LC19
    - LRSN Spin
  - Restart improvements
    - Reordering processes
    - Data available sooner
  - -ACCESS DB command
    - Remove GBP Dependency
    - Prime Open Data Sets
  - GRECP Improvements
    - Automatic START DB
  - Health Monitoring
  - Balance Group Connections
- Utilities
  - pureXML Support
  - REORG BUILD2 elimination
  - Template switching
  - MODIFY improvements
    - RETAIN
  - COPY improvements
    - Always runs CHECKPAGE
  - SYSTEM BACKUP / RECOVERY improvements
    - Object level recovery
  - DFSort Notes
    - zIIP redirectable

DB2 9 Highlights

- Serviceability
  - DISPLAY Improvements
    - DISPLAY THREAD ...SERVICE(WAIT)
    - DISPLAY shows HEALTH %
  - New DSNRLMTxx Table
  - REFRESH DB2,EARLY
- DB2 9 for z/OS Migration Planning Workshop
  - Consider attending a DB2 9 MPW for a complete review of these capabilities & their considerations
  - DB2 9 features, incompatibilities, considerations need to be consolidated for a skip level migration.

DB2 10 for z/OS

- Base Pre-requisites:
  - zSeries z890, z990, z9, z10, z196 or later
  - z/OS V1.10 or above
  - Defined shared memory objects (V9)
- Catalog changes:
  - Additions for new features
  - Hashes and links removed
  - Many tables changed to:
    - Single table, table spaces (UTS, PBG)
    - Row level locking
    - Using Inline LOBs
- DB2 Connect Minimum Levels
  - V9 FP1 / V9.7 FP3a for new features
- PM24292 for Sysplex Workload Balancing
- See CST RSU for current recommendation
- Migration Process:
  - From Version 8 or 9 NFM
  - Data sharing coexistence in CM8 or CM9
  - DSNTIJPA in V8 or V9 / DSNTIJSS for SMS

Major Presentation Sections

SHARE
Technology · Connections · Results

- **SQL** SQL Enhancements
- **XML** pureXML, The Sequel
- **BTD** Temporal Data – Time Travel Query
- **LOB** LOB Enhancements
- **OPT** Optimization Evolution
- **DSN** Advanced Design Options
- **IDX** Indexing Enhancements
- **SEC** Security for Today's Challenges
- **DDF** Distributed Data Performance
- **UTL** Utility Enhancements
- **VST** Virtual Storage Improvements
- **DSH** Data Sharing – Availability Supremacy
- **SVC** Serviceability & Instrumentation

SQL Enhancements

- Access Currently Committed
- Extended SQL Procedure Language (SQL PL)
  - Enhanced Native SQL Procedure
  - SQL Scalar Functions
  - SQL Table Functions
- Extended Indicator Variables
- More OLAP Specifications
- Greater Timestamp Precision
- Timestamp with Time Zone
- Function Modifications
- Stored Procedure Enhancements
- 64-bit ODBC driver for z/OS

SQL Enhancements Business Values

- Possibility of enhanced concurrency using the new Access Currently Committed feature.
- Expanded use of the easy to use SQL Procedure for scalar functions and table functions.
- Simplified coding of INSERT and UPDATE statements by using the new enhanced indicator variables. Eliminates/reduces need to code multiple statements when dealing with multiple values that may be provided or not.
- Several new rolling OLAP functions.
- Greater timestamp precision and inclusion of time zone with timestamp.
- Ability to run 64-bit ODBC applications on z/OS.

Access Currently Committed

- Pre-DB2 10 Problem/Requirements:
  - Reads acquire locks on data
  - Overhead, contention, concurrency issues
  - ISOLATION(UR) avoids contention, but does not return committed data
- DB2 10 Solution:
  - Return currently committed data without waiting for locks.
  - Ability can be specified on:
    - BIND/REBIND PACKAGE or PLAN
    - SQL PREPARE statement
    - CREATE/ALTER PROCEDURE (native SQL) or FUNCTION (SQL Scalar)

Access Currently Committed…

- DB2 10 Solution Restrictions:
  - Support for uncommitted INSERT and DELETE
  - Reader will still wait for uncommitted UPDATE
  - Not applicable when:
    - LOCK TABLE IN EXCLUSIVE used
    - Lock holder is performing mass delete
    - If lock holder escalates
- DB2 10 Solution Requirements:
  - Universal Table Space
  - Row or Page-level locking
    - IRLM will remember first 8 locked rows when using page-level locking.
  - NFM

Access Currently Committed…

- SELECT bound with ISOLATION(CS) or ISOLATION(RS) and CONCURRENTACCESSRESOLUTION(USECURRENTLYCOMMITED)
- Syntax specifications:
  - BIND/REBIND PACKAGE and PLAN:
    - Specify option **CONCURRENTACCESSRESOLUTION (USECURRENTLYCOMMITTED)** or **(WAITFOROUTCOME)**
  - PREPARE attribute string:
    - Specify **USE CURRENTLY COMMITTED** or **WAIT FOR OUTCOME**
  - CREATE/ALTER PROCEDURE or FUNCTION
    - Specify option **CONCURRENTACCESSRESOLUTION (USECURRENTLYCOMMITTED)** or **(WAITFOROUTCOME)**
    - Can be specified for new and existing Native SQL Procedures
  - DRDA supports use of CONCURRENTACCESSRESOLUTION bind option, and can be used for SQLJ applications.

Access Currently Committed…

- SELECT not blocked by INSERT:
  - Pre-DB2 10 SELECT waits and eventually a row returned or times out.
  - DB2 10 can see row being inserted is not committed, and would immediately skip row.
  - Relation to SKIPUNCI ZParm for uncommitted INSERTs:
    - Bind option or PREPARE attribute value controls outcome when specified
    - SKIPUNCI value controls outcome when not defined
- SELECT not blocked by DELETE:
  - Pre-DB2 10 SELECT waits and eventually no row is returned or times out.
  - Currently committed row, including any LOB or XML data is returned, until DELETE is committed.

Access Currently Committed…

- Consider as second generation lock avoidance.
  - DB2 may revert to unconditional locking.
  - Alternative to DB2 9 “SKIP LOCKED DATA” option, which ignores row.
  - Alternative to ISOLATION(UR)
- QMF global variable DSQEC_CON_ACC_RES can be used to support feature.

Extended SQL Procedure Language (SQL PL)

- Enhanced Native SQL Procedures capabilities:
  - Can define an SQL parameter or SQL variable as a distinct type or XML data type **XML**
  - Makes it easier to port applications from other DBMS
  - CREATE statement for a native SQL procedure can be embedded in an application program
  - ALTER statement for a native SQL procedure can be embedded in an application program, even if the ADD or REPLACE VERSION clause is included. This was not available in DB2 9.
  - DB2 10 NFM

Extended SQL Procedure Language (SQL PL)…

- SQL Scalar Function:
  - DB2 10 NFM introduces two types of SQL Scalar Functions
    - Inline and
    - Non-inline
  - Inline SQL scalar functions are similar in structure to SQL scalar functions prior to DB2 10 with a single RETURN statement, and can use the XML data type in DB2 10 NFM.
  - Non-inline SQL scalar functions have added benefits:
    - Can contain logic using SQL statements and SQL control statements like those used in native SQL procedures. Will create a package.
    - Allow for multiple versions, where one is considered the "active" version.
      - *Initial version defined by CREATE FUNCTION … VERSION id*
      - *Subsequent versions defined by ALTER FUNCTION… ADD VERSION id*
      - *Activate version using ALTER FUNCTION… ACTIVATE VERSION id*
    - Versions can easily be replaced or dropped.

Extended SQL Procedure Language (SQL PL)…

- SQL Scalar Function… additional Non-inline benefits:
  - Packages can be rebound with:
    - **ALTER FUNCTION … REGENERATE** – includes the SQL control statements and SQL statements in the function; mainly used in conjunction with DB2 maintenance.
    - **REBIND PACKAGE** – rebinds only the SQL statements in the function; usually used to generate new access paths.
  - Allow for debugging using the Unified Debugger, like what is used for debugging native SQL procedures.
  - Can replace the use of external scalar functions by:
    - Simplifying development
    - Address performance considerations
    - Providing a simpler change management capability that can be used to maintain multiple versions.
  - Can deploy the function by using the BIND PACKAGE… DEPLOY
  - Makes it easier to port applications from other DBMS.
  - DB2_LINE_NUMBER from GET DIAGNOSTICS, introduced in DB2 9, will return the line number where an error is encountered with the function.

Extended SQL Procedure Language (SQL PL)…

- SQL Scalar Function… Examples:

  Inline: Defines a scalar function that returns a duration in years.

```
create function years(x date)
returns smallint
language sql
contains sql
no external action
not deterministic
return year(current date - x)
;
```

```
select current date from sysibm.sysdummy1
----------
2011-01-06

select empno
      ,lastname
      ,hiredate
      ,years(hiredate) as service_length
from dsn81010.emp
where workdept = 'A00'

EMPNO  LASTNAME        HIREDATE   SERVICE_LENGTH
------ --------------- ---------- --------------
000010 HAAS            1965-01-01             46
000110 LUCCHESI        1958-05-16             52
000120 O'CONNELL       1963-12-05             47
200010 HEMMINGER       1965-01-01             46
200120 ORLANDO         1972-05-05             38
```

Reference Date

Extended SQL Procedure Language (SQL PL)…

- SQL Scalar Function… Examples:

  Non-inline: Defines a scalar function that reverses a string.

```
CREATE FUNCTION REVERSE (REVSTR VARCHAR(100))
RETURNS VARCHAR(100)
DETERMINISTIC
NO EXTERNAL ACTION
CONTAINS SQL
BEGIN
  DECLARE IN_STR
         ,OUT_STR VARCHAR(100) DEFAULT '';
  DECLARE LEN     INT;
  IF IN_STR IS NULL THEN
    RETURN NULL;
  END IF;
  SET (IN_STR, LEN) = (REVSTR,LENGTH(IN_STR));
  WHILE LEN > 0
  DO
    SET OUT_STR = SUBSTR(IN_STR,1,1) || OUT_STR;
    SET IN_STR  = SUBSTR(IN_STR,2);
    SET LEN     = LEN - 1;
  END WHILE;
  RETURN OUT_STR;
END
#
```

```
SELECT LASTNAME
      ,EMPNO
      ,REVERSE(EMPNO) AS ONPME
FROM DSN81010.EMP
WHERE WORKDEPT = 'A00'

LASTNAME        EMPNO  ONPME
--------------- ------ ---------
HAAS            000010 010000
LUCCHESI        000110 011000
O'CONNELL       000120 021000
HEMMINGER       200010 010002
ORLANDO         200120 021002
```

Extended SQL Procedure Language (SQL PL)…

- SQL Table Function:
  - DB2 10 NFM introduces ability to create SQL table functions.
  - Written using SQL statements and includes a single RETURN statement that returns a single result table (a set of rows).
  - Parameters can be defined as:
    - Built-in or distinct data types
    - TABLE LIKE table/view-name AS LOCATOR (transition table)
  - Can be defined to behave as parameterized views
  - CREATE and ALTER statements can be embedded in an application program or issued interactively.

Extended SQL Procedure Language (SQL PL)…

- SQL Table Function… Example:

  Defines a table function that returns a list of employees for a department.

```
CREATE FUNCTION DEPTEMPLOYEES (DEPTNO CHAR(3))
RETURNS TABLE (EMPNO     CHAR(6)
               ,LASTNAME  VARCHAR(15)
               ,FIRSTNAME VARCHAR(12)
               )
LANGUAGE SQL
READS SQL DATA
NO EXTERNAL ACTION
DETERMINISTIC
RETURN
  SELECT EMPNO
         ,LASTNAME
         ,FIRSTNME
  FROM DSN81010.EMP E
  WHERE E.WORKDEPT = DEPTEMPLOYEES.DEPTNO
;
```

```
SELECT *
FROM TABLE (DEPTEMPLOYEES('A00')) AS T

EMPNO  LASTNAME        FIRSTNAME
------ --------------- ------------
000010 HAAS            CHRISTINE
000110 LUCCHESI        VINCENZO
000120 O'CONNELL       SEAN
200010 HEMMINGER       DIAN
200120 ORLANDO         GREG
```

SQL Procedure Language Cross Releases

- PM13525 enables DB2 to be more tolerant of SQL Stored Procedures created across DB2 Versions
- This helps with the following conditions:
  - **A procedure developed on DB2 10 and DEPLOYed on DB2 9**
  - **When a DB2 10 created or regenerated procedure is run on DB2 9 and is rebound or regenerated**

Extended Indicator Variables

- Pre-DB2 10 Problem/Requirements:
  - No way to tell DB2 what value to use in an INSERT, UPDATE or MERGE when values for all the columns are not specified.
  - Need to define all possible SQL combinations… very tedious
  - Very low use of dynamic statement cache.
- DB2 10 Solution:
  - DB2 10 NFM introduces extended indicator variables to specify what value should be used for columns when a value is not provided in an INSERT, UPDATE or MERGE.
    - New values used for existing indicator variables.
  - Simplifies application development… no need to define all possible SQL combinations
  - Improves overall performance

Extended Indicator Variables

- Implementation:
  - EXTENDEDINDICATOR YES/NO option on BIND/REBIND PACKAGE
    - NO is the default for BIND
    - Existing setting is default for REBIND
    - New EXTENDEDINDICATOR column added to SYSPACKAGE
  - WITH EXTENDED INDICATORS or WITHOUT EXTENDED INDICATORS attribute string on PREPARE
    - Default - WITHOUT EXTENDED INDICATORS
  - Various methods available for implementing support with JDBC, SQLJ and ODBC applications.
  - Extended indicator variable values:
    - -5 specifies the DEFAULT value – set column to the default value
    - -7 specifies the UNASSIGNED value - treat as column was not specified

Extended Indicator Variables … Examples

- **Assume the following table definition:**

```
CREATE TABLE T1 (
   C1 INT WITH DEFAULT 100,
   C2 INT,
   C3 VARCHAR(6));
```

- **Request DB2 to use a** DEFAULT **value be used for an INSERT:**

```
IND1 = -5;
EXEC SQL INSERT INTO T1 VALUES(:HV1 :IND1, 999, 'AAAA');
```

  - DB2 will use the default value for C1.

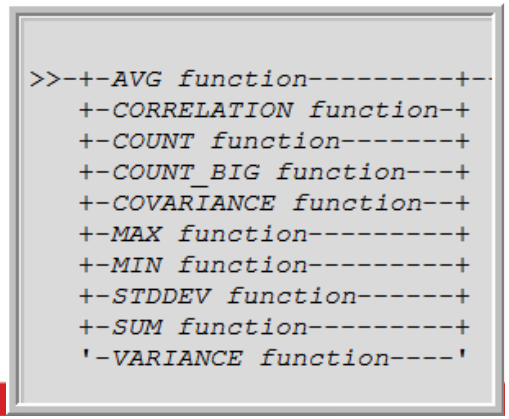
- **Request DB2 to consider a value** UNASSIGNED **for an UPDATE:**

```
IND1 = -7;
CHARHV1 = 'AAAA';
EXEC SQL UPDATE T1 SET C1=:HV1:IND1, C2=888
   WHERE C3=:CHARHV1;
```

  - DB2 will leave the value of C1 untouched and update C2.
  - UPDATE with all SET columns using -7 results in -20434.

More OLAP Specifications

- Moving Sum and Moving Average
  - New DB2 10 NFM specifications (aggregate functions) that compute a single value for the current row based on some or all the rows in a defined group.
  - Supports cumulative sums and moving averages by using a window.
  - A window can specify three optional components:
    - Partitioning of the result table using PARTITION BY clause (in DB2 9).
    - Ordering of rows within a partition using ORDER BY clause (in DB2 9).
    - Aggregation group by using ROW or RANGE clause
    - Two mandatory components need to be specified for an aggregation group.
      - *Keyword ROWS to indicate a physical group … or*
      - *Keyword RANGE to indicate a logical group … and*
      - *The starting and ending row of the aggregation group*
  - Cannot be used within an XMLQUERY, XMLEXISTS, or as an argument of an aggregate function.
  - New -20117 SQL code to indicate an invalid window



More OLAP Specifications… Examples

- Assume a SALES_HISTORY table containing the following.
  - Each TERRITORY has 5 rows.

TERRITORY	MONTH	SALES
EAST	200910	10.00
WEST	200910	8.00
EAST	200911	4.00
WEST	200911	12.00
EAST	200912	10.00
WEST	200912	7.00
EAST	201001	7.00
WEST	201001	11.00
EAST	201002	9.00
WEST	201002	7.00

More OLAP Specifications… Examples

```
SELECT TERRITORY, MONTH, SALES, AVG(SALES)
  OVER(PARTITION BY TERRITORY ORDER BY MONTH
    ROWS 2 PRECEDING) AS MOVING_AVG
  FROM SALES_HISTORY
```

TERRITORY	MONTH	SALES	MOVING_AVG
EAST	200910	10.00	10.00
EAST	200911	4.00	7.00
EAST	200912	10.00	8.00
EAST	201001	7.00	7.00
EAST	201002	9.00	8.66
WEST	200910	8.00	8.00
WEST	200911	12.00	10.00
WEST	200912	7.00	9.00
WEST	201001	11.00	10.00
WEST	201002	7.00	8.33

Results:

1. Grouped by TERRITORY
2. Ordered by MONTH
3. MOVING_AVG values are computed on the average SALES value of the current row with the prior 2 rows within the TERRITORY.

More OLAP Specifications… Examples

```
SELECT TERRITORY, MONTH, SALES, SUM(SALES)
 OVER(PARTITION BY TERRITORY ORDER BY MONTH
   ROWS UNBOUNDED PRECEDING) AS CUMULATIVE_SUM
 FROM SALES_HISTORY
```

"ROWS UNBOUNDED PRECEDING" is optional. Results are the same without it.

TERRITORY	MONTH	SALES	CUMULATIVE_SUM
EAST	200910	10.00	10.00
EAST	200911	4.00	14.00
EAST	200912	10.00	24.00
EAST	201001	7.00	31.00
EAST	201002	9.00	40.00
WEST	200910	8.00	8.00
WEST	200911	12.00	20.00
WEST	200912	7.00	27.00
WEST	201001	11.00	38.00
WEST	201002	7.00	45.00

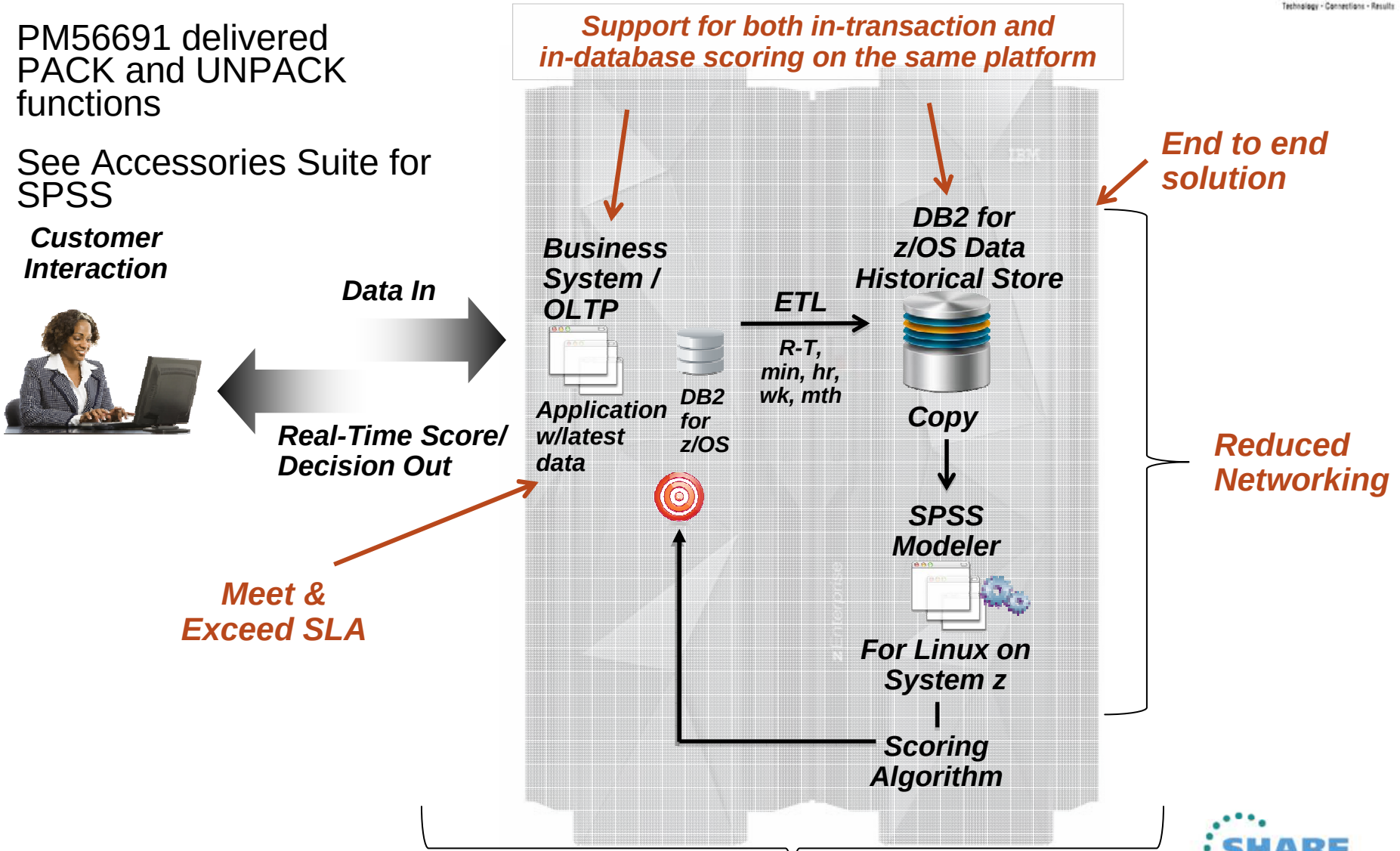
Results:

1. Grouped by TERRITORY
2. Ordered by MONTH
3. CUMULATIVE_SUM values are computed on the sum of the SALES value of the current row with the prior row(s) within each TERRITORY.

Modeler 15 Real-time Scoring with DB2 for z/OS

PM56691 delivered PACK and UNPACK functions

See Accessories Suite for SPSS



Greater Timestamp Precision

- Number of digits for the fractional second in a timestamp extended
  - The DB2 9 default of 6 digits remains
    - TIMESTAMP is the same as TIMESTAMP(6)
    - String representation: yyyy-mm-dd-hh.mm.ss.nnnnnn
  - Range supported in DB2 10 NFM is 0 to 12 digits
    - E.g. TIMESTAMP(12) is the maximum
    - String representation: yyyy-mm-dd-hh.mm.ss.nnnnnnnnnnnn
  - Other capabilities like timestamp duration and CURRENT TIMESTAMP extended to support new precision capability
  - Can be altered to higher precision. Example: TS(6) to TS(10)
    - Will fail with SQLCODE -190 if lowered. Example: TS(10) to TS(6)

Timestamp with Time Zone

- New data type TIMESTAMP WITH TIME ZONE
  - SET SESSION TIME ZONE or rely on IMPLICIT_TIMEZONE (DECP)
  - Year.month.day.hour.minutes.seconds +-timezone

```
CREATE TABLE TABLE1  (C1 TIMESTAMP WITH TIME ZONE, C2 INTEGER);
INSERT INTO TABLE1 VALUES ( '2009-11-05-08.00.00-08:00', 1);
INSERT INTO TABLE1 VALUES ( '2009-11-05-11.00.00-05:00', 2);
INSERT INTO TABLE1 VALUES ( '2009-11-05-10.00.00', 3);
```

- Comparison takes place after converting internally to UTC
  - '2009-11-05-08.00.00 -08:00‘ is converted to ‘2009-11-05-16.00.000000’
  - ‘2009-11-05-11.00.00 -05.00’ is converted to ‘2009-11-05-16.00.000000’
    - SELECT C2 FROM TABLE1 WHERE C1 = '2009-11-05-10.00.00 -06:00';
    - would retrieve all 3 of the above
      - *‘-6:00’ is the timezone, US Central which is UTC -6, except during daylight savings time*

Timestamp with Time Zone…

- New DSNHDECP parameter IMPLICIT_TIMEZONE.
  - Determines implicit time zone to be associated with DB2 table columns and routine parameters declared with TIMESTAMP WITH TIME ZONE, when a time zone is not provided.
  - Defined in new IMPLICT TIME ZONE field on installation panel DSNTIP4 (Application programming defaults panel 2)
  - Values:
    - CURRENT (default) = uses setting of the CURRENT TIME ZONE special register
    - SESSION = uses setting of the SESSION TIME ZONE special register
    - -12:59 to +14:00 = uses the value of the string. Format of value is *th:tm*, where *th* is the time zone hour (-12 to +14), and *tm* is the time zone minute (00 to 59). Value is an offset of hours and minutes from UTC.

Timestamp with Time Zone…

- Definitions:
  - CURRENT TIME ZONE special register: (data type = DECIMAL(6,0))
    - *Contains the difference between UTC and local time at the current server. The difference is represented by a decimal number in the format ttmmss.*
      - *tt = number of hours between -24 and 24 exclusive*
      - *mm = number of minutes*
      - *ss = number of seconds*
    - *Example:*
      - *CURRENT TIME = 08.30.00 US Central time*
      - *CURRENT TIME ZONE = -60000.*
        (Except during daylight savings time, then -50000.)
  - SESSION TIME ZONE special register: (data type = VARCHAR(128))
    - *Identifies the time zone of the application process in the format tt:tm*
    - *tt = time zone hour offset between -12 and +14*
    - *tm = time zone minutes offset between 0 and 59*
    - *Example (using above current time) = -06:00 (Follows same DST exception.)*

Functions Modified for Competitive Compatibility

- See PM40724
  - Enabling APAR PM38326
- Functions added / modified
  - Modified
    - LTRIM
    - RTRIM
    - REPLACE
    - ROUND
    - TRUNCATE or TRUNC
  - Added
    - TRIM

Stored Procedure Enhancements

- PM37668
  - Stored Procedure RETURN TO CLIENT cursor support
    - Returns result set to client from a nested stored procedure
    - Result set is invisible to intermediate stored procedures
  - For NSPs RETURN TO CLIENT cursors
    - Duplicate cursor instance support (see APAR for details)
    - Becomes a result set cursor
- PM53243
  - Stored procedure monitoring improvements
  - Begin / End for each SP or UDF invocation (IFCID 233)
  - Execution details (IFCID 380 & 381)
  - Statement level details (IFCID 497, 498, 499)

64-bit ODBC driver for z/OS

- **Provides the ability for 64-bit ODBC applications to run on z/OS, and take advantage of the new addressability.**
  - In DB2 9 with PK83072.
- **New driver benefits:**
  - runs in AMODE(64)
  - reduces virtual storage constraint
  - can accept 64-bit user data pointers
  - access user data above the 2GB bar in the application address space
  - XPLINK only
  - Shipped in addition to the 31-bit ODBC driver.
- **31-bit ODBC driver is XPLINK and non-XPLINK. The APIs for existing 31-bit applications have not been changed, and continue to work using the 31-bit ODBC driver.**
- **Consider migrating 31-bit applications to 64-bit, if the application can take advantage of more than 2GB of memory.**
  - **This is a migration**
    Migrating an ODBC 31-bit application to a 64-bit application
- **The XPLINK driver is recommended to enhance performance, but only if your ODBC application uses XPLINK code exclusively.**
- **For more information:** DB2 ODBC application requirements

SQL Enhancements – IBM DB2 Tools Support

- DB2 Admin Tool & Object Compare
  - Access currently committed
  - Extended SQL Procedure Language (SQL PL)
    - Native SQL Procedures
  - Extended indicator variables
  - Timestamp precision
  - Timestamp with time zone
- DB2 Table Editor
  - Extended indicator variables
  - Greater timestamp precision
  - Timestamp with time zone
- Data Studio (Query Tuner)
  - See Migration presentation for enablement instructions
- Optim Query Workload Tuner

pureXML, The Sequel

- DB2 9 pureXML Recap
- Multi-Versioning Support
- Subdocument Update
- Validation Improvements
- CHECK & REPAIR
- Native SQL Procedures & UDFs
- XML Date and Time
- Binary XML

pureXML Business Value

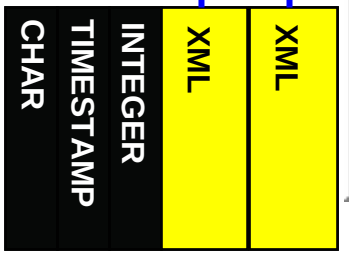
- Enable / support modern application requirements
- Support for the W3C standard for information exchange
- Store native XML or publish XML from relational
- Performance based architecture to minimize parsing
- Cost effective exploitation of specialty processors
- Familiar SQL and Utilities for use and administration
- XML Schema support built in

DB2 9 pureXML Recap...

- Extends XML Publishing (introduced in DB2 V8)
  - Create XML documents from relational information
- DB2 for z/OS became a hybrid relational and XML engine.
- New XML storage architecture using auxiliary XML spaces.
- Native XML data type
  - XML stored in parsed node format
  - DB2 ensures XML is well formed
  - UTS, RRF, 16KB page
  - Stored in Unicode
- XPath introduced into SQL to administer and manipulate XML
  - Use familiar CREATE, ALTER, DROP DDL for administration
    - XPath used in CREATE INDEX to identify the Element or Attribute
  - Use familiar SELECT, INSERT, DELETE, UPDATE DML
- XML Schema Repository (DSNXSR)

DB2 9 pureXML Recap...

- Key Functions
  - XMLQUERY to SELECT specific elements or attributes from a document
  - XMLEXISTS to apply a WHERE predicate against elements or attributes
  - XMLSERIALIZE to construct the document in a external string format
  - XMLPARSE to parse an XML document into the internal node format
  - XMLTABLE to externalize an XML document in relational row / column format with SQL data types
    - Used for XML Decomposition



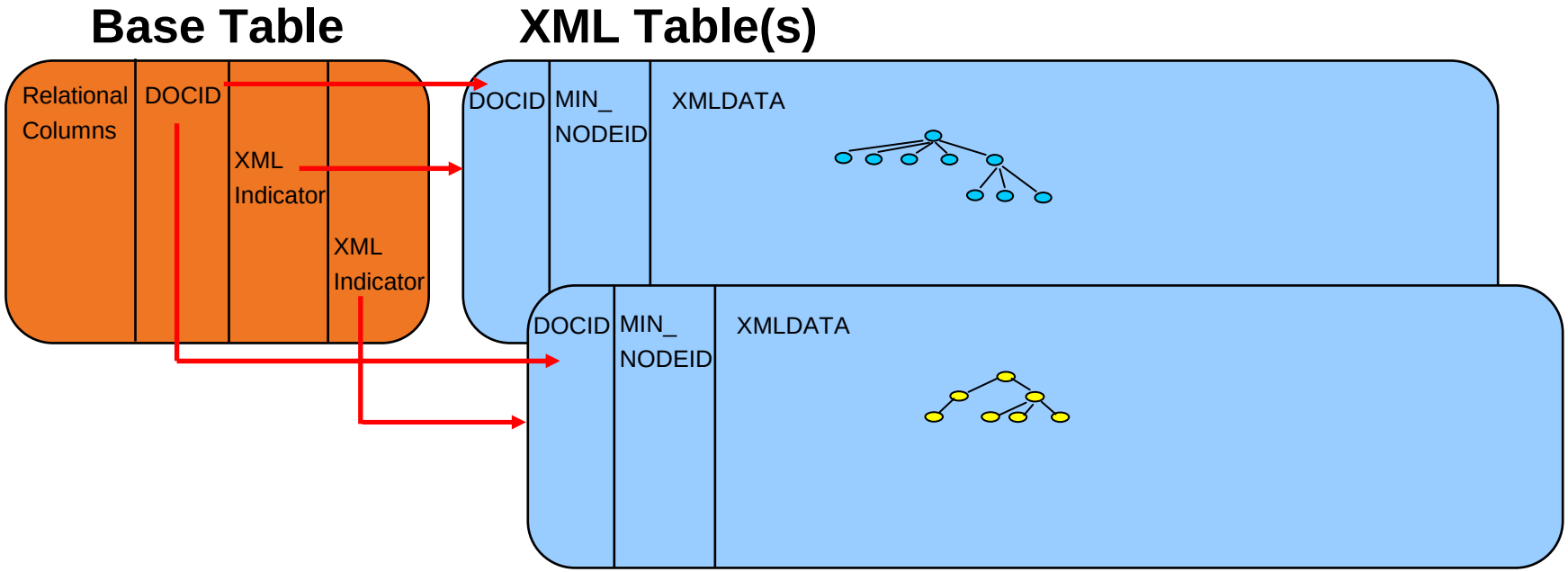

```
select deptno
      ,row_timestamp
      ,xmlpot_sequence
      ,xmlserialize(xmlquery('/DEPT/PROJ[@PROJNO="AD3110"]'
       passing xml_time_account) as clob) as xml_time_account_proj
      ,xmlserialize(xml_audit as clob) as xml_audit
from time_accounting_part
where xmlexists('/DEPT[@DEPTNO="D21"]' passing xml_time_account)
  AND XMLEXISTS('/DEPT/PROJ/EMP[STARTDATE>="1983-01-01"]'
                PASSING XML_TIME_ACCOUNT)
;
```

Multi-Versioning (MV) Support...

- Requires the base table to be a Universal Tablespace (UTS)
- Existing tables with XML columns must be dropped and recreated after DB2 10 NFM for MV format.
- MV format required for several V10 features:
  - XMLMODIFY
  - Temporal data ("**AS OF**") **BTD**
  - Currently Committed
  - SELECT FROM OLD TABLE (V9 feature) **SQL**
- The MV format reduces locking
  - V9
    - XML data is not kept with base row data in the work files
    - Therefore a lock is maintained on the XML row until it is processed after work file
    - MV eliminates the need for this lock except for UR Readers (DOCID lock). Avoids a UR Reader from getting an incomplete document.

Multi-Versioning (MV) Support...

- DB2 9 pureXML Table Formats



DOCID:
Colname: DB2_GENERATED_DOCID_FOR_XML
Data Type: BIGINT
One per base table that has 1 or more XML column(s)

XML:
Colname: DDL XML column name
Data Type: VARBIN(6)
One per XML column associated with the base table

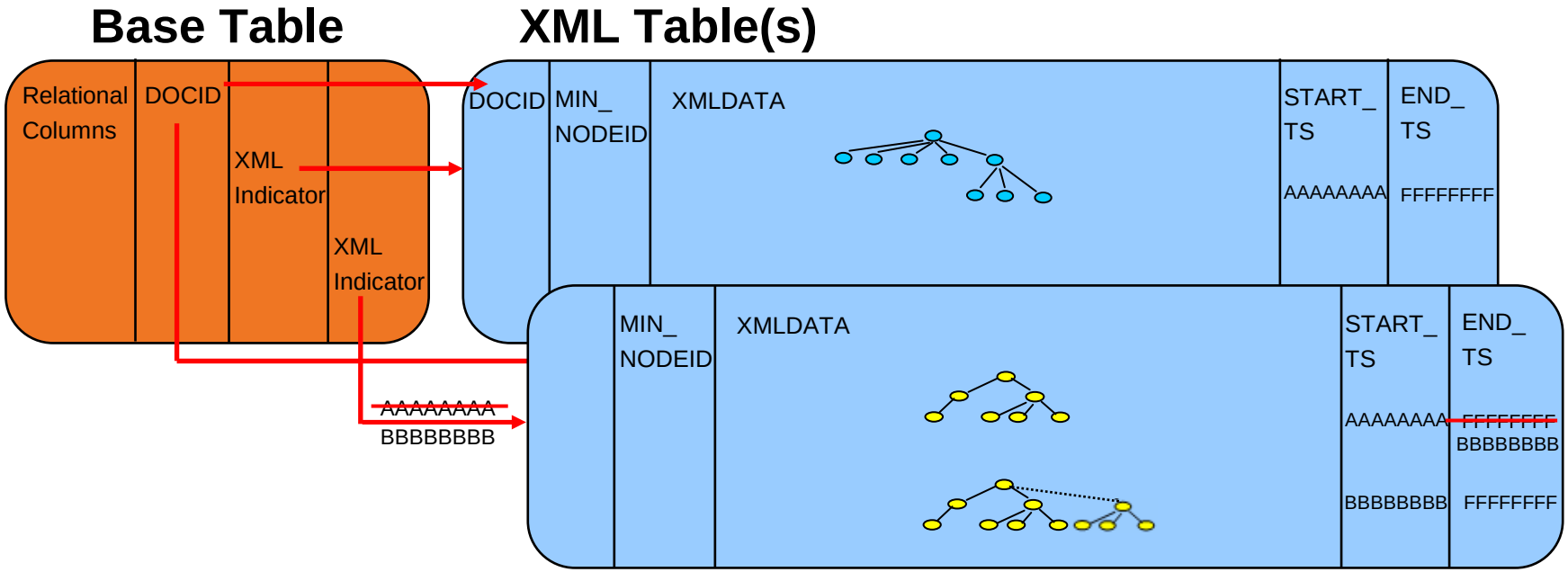
DOCID:
Colname: DOCID
Data Type: BIGINT

MIN_NODEID:
Colname: MIN_NODEID
Data Type: VARBIN(128)

XMLDATA:
Colname: XMLDATA
Data Type: VARBIN(15850)

Multi-Versioning (MV) Support...

- DB2 10 pureXML Table Formats



DOCID:
Colname: DB2_GENERATED_DOCID_FOR_XML
Data Type: BIGINT
One per base table that has 1 or more XML column(s)

XML:
Colname: DDL XML column name
Data Type: VARBIN(14). Now includes 8 byte TS.
One per XML column associated with the base table

DOCID:
Colname: DOCID
Data Type: BIGINT

MIN_NODEID:
Colname: MIN_NODEID
Data Type: VARBIN(128)

XMLDATA:
Colname: XMLDATA
Data Type: VARBIN(15850)

START_TS:
Colname: START_TS
Data Type: BINARY(8)

END_TS:
Colname: END_TS
Data Type: BINARY(8)

RRF makes physical column order:
DOCID, START_TS, END_TS, MIN_NODEID, XMLDATA

Subdocument Update with XMLMODIFY...

- DB2 9 pureXML supported the updating of an XML column
  - But the entire column needed to be replaced
- DB2 10 delivers the XMLMODIFY function to:
  - Insert nodes within an existing document
  - Replace existing nodes of a document
  - Delete nodes from an existing document
- Invoked via UPDATE....SET SQL
- Requires MV format
- One updater at a time for a document
  - Concurrency control by the base table
  - Document level lock to prevent UR reader

```
DSNT408I SQLCODE = -4730, ERROR:  INVALID SPECIFICATION OF XML COLUMN
         XML_MV_TEST_FROMV10.XMLCOL IS NOT DEFINED IN THE XML VERSIONING
         FORMAT, REASON 1
```

Subdocument Update with XMLMODIFY...

- Only changed rows in XML table are updated
- UPDATE...SET...XMLMODIFY with “insert node”
  - insert node (must be in lower case)
    - Adds a node to an existing document
    - *Example*: Insert a new PROJ element node (case matters)
      - *With a PROJNO attribute of “z02” and NAME attribute of “pureXML V10 Updates”*
      - ***After** an existing PROJ element with a PROJNO attribute of “z01”*
      - *A -16085 will be returned if a referenced after / before node cannot be located or multiple nodes qualify the XPath expression*

```
update time_accounting_part
set xml_time_account =
      xmlmodify ('insert node $m/PROJ after /DEPT/PROJ[@PROJNO="z01"]',
                  xmlparse(document
                           '<PROJ PROJNO="z02" NAME="pureXML V10 Updates"/>')
                           as "m")
WHERE .....
```

Subdocument Update with XMLMODIFY...

- UPDATE...SET...XMLMODIFY “insert node” (continued)
  - Positioning options:
    - **after** <*XPath identifying an existing node*>
    - **before** <*XPath identifying an existing node*>
    - **as first into**
    - **as last into (default when “into” is coded)**
- UPDATE...SET...XMLMODIFY “replace value of node”
  - Replaces the value of an existing element (text node) or attribute
  - *Example*: The value of an existing DEPTNO attribute of a DEPT element is updated to a value of “R32”
  - A -16085 will be returned if multiple nodes qualify the XPath expression

```
update time_accounting_part
 set xml_time_account =
       xmlmodify ('replace value of node /DEPT/@DEPTNO with "R32"')
 WHERE ...
```

Subdocument Update with XMLMODIFY

- UPDATE...SET...XMLMODIFY “delete node”
  - Removes an existing node from a document
  - *Example*: The PROJ element with a PROJNO attribute of “z01” is removed from the document
  - The returned SQLCODE may not be a good indication of what has taken place
    - If the WHERE clause qualifies a row
    - But the delete expression does not match any nodes of a document
    - A SQLCODE of +0 is returned and there is no indication of the number of nodes removed
    - Also, there is no reporting of the number of nodes impacted
    - If multiple nodes qualify, they will be deleted

```
set xml_time_account =
      xmlmodify ('delete node /DEPT/PROJ[@PROJNO="z01"]')
WHERE ...
```

Validation Improvements...

- XML Column Type Modifier
  - Allows a “constraint” type of definition on an XML column via CREATE or ALTER TABLE
    - Several schemas may be specified in the type modifier
      - *This supports the evolution of a schema*
      - *See the XML Schema Versioning topic*
  - All documents INSERTed, UPDATEd, or LOADed in the column are validated to conform to the specified schema(s)
  - Adding an XML Schema into XML Column Type Modifier via an ALTER may place the XML Table Space into CHKP status
    - When ALTERing the Type Modifier, include all Schemas to be used
      - *Including those already in use*
      - *Let DB2 figure out the delta*
    - For CHKP status
      - *Run CHECK DATA to validate schema conformance*
        - *Validates that existing document conform to specified schema(s)*
      - *Note: Removing an XML Column Type Modifier will not remove a CHKP status*

Validation Improvements...

- XML Column Type Modifier (continued)
  - Add

```
alter table time_accounting_part
      alter xml_time_account
              set data type xml(xmlschema ID SYSXSR.XMLPOT element DEPT)
      ;
```

  - Remove

```
alter table time_accounting_part
      alter xml_time_account
              set data type xml
      ;
```

Change from Type Modifier	To Type Modifier	XML Table Space Impact
No modifier	XMLPOT1	Check pending
XMLPOT1	XMLPOT1, XMLPOT2	None
XMLPOT1	XMLPOT2	Check pending
XMLPOT1, XMLPOT2	XMLPOT1	Check pending
XMLPOT1	No modifier	None

Validation Improvements...

- XML Schema Versioning
  - Multiple versions of a schema can be stored in the XML Schema Repository
  - An XML Schema is identified by:
    - Target Namespace
    - Schema Location (also known as Schema Location Hint)
    - Schema Name
  - When there is no Schema Location and multiple schemas are found then the latest registered schema is chosen
  - Administration
    - Use the same stored procedures provided in DB2 9 to administer the XSR
    - Use Data Studio
    - CLP
  - If setting up the XML Schema Repository under DB2 10, see installation job DSNTIJNX & DSNTIJRT if migrating from V8. Only DSNTIJRT from V9.

See “How DB2 chooses an XML schema from an XML type modifier” in the XML Guide

SYSIBM.XSROBJECTS	**One row per registered XML Schema**
SYSIBM.XSROBJECTCOMPONENTS	**One row per document in an XML Schema**
SYSIBM.XSROBJECTHIERARCHIES	**Records the XML Schema document relationships**
SYSIBM.SYSXMLSTRINGS (DSNDB06)	**Not part of the XSR, but contains the decode information for schema namespace & location**

Validation Improvements...

- XML Schema Versioning (continued)
  - V9 XML Schemas still work for V10
  - Validation process changes
    - **SYSFUN.DSN_XMLVALIDATE** V9 UDF is replaced by V10 BIF **SYSIBM.DSN_XMLVALIDATE** in CM9 or NFM
      - *Increases validation document size from 50MB to 2GB – 1.*
      - *No application changes are needed if **DSN_XMLVALIDATE** is not qualified.*
      - ***SYSFUN.DSN_XMLVALIDATE** is ALTERed in CM9, invalidating packages*
      - *Available in DB2 9 with PK90032 & PK90040 (recommended for migration from DB2 9)*
    - If **DSN_XMLVALIDATE** is used for a column with a XML type modifier
      - *If the BIF specified schema is also defined in the type modifier, then this directs DB2 to use the BIF specified schema*
      - *If the BIF specified schema is not defined in the type modifier, then the document must conform to the BIF schema AND the selected schema from the type modifier*
    - 100% zIIP / zAAP redirectable, if available
  - z/OS XML System Services is used for validation
    - This produces different error messages from those in DB2 9.
    - "LOCATION x", points to the byte offset location in the document

Validation Improvements...

- What XML Schema was used to validate a document?
  - The XMLXSROBJECTID function will return the internal identifier for the XML Schema from the XML column
    - **XMLXSROBJECTID**(*xml_column*)
  - The text of the schema name can be retrieved from XSR
  - The IDs of the XML Namespace and Schema Location Hint can also be retrieved from the XSR (**SYSIBM.XSROBJECTS**)
    - The text of the XML Namespace and Schema Location Hint is available in the catalog (**SYSIBM.SYSXMLSTRINGS**)

```
select c.xsrobjectname as "XML Schema Name"
      ,d.string        as "XML Schema Namespace"
      ,e.string        as "XML Schema Location"
from (
      select distinct
             b.xsrobjectname
            ,b.targetnamespace
            ,b.schemalocation
      from  time_accounting_part a
           ,sysibm.xsrobjects b
       where b.xsrobjectid = XMLXSROBJECTID(a.xml_time_account)
      ) as c
      left outer join
      sysibm.sysxmlstrings d
      on d.stringID = c.targetnamespace
      left outer join
      sysibm.sysxmlstrings e
      on e.stringid = c.schemalocation
```

The XMLXSROBJECTID function retrieves the XML Schema ID from the documents in the XML_TIME_ACCOUNT column.

These IDs are used to find the XML Schema Names (XSROBJECTNAME) from the SYSIBM.XSROBJECTS table.

The Namespace and Location IDs are retrieved from the XSROBJECTS table and the description (STRING) retrieved from the SYSIBM.SYSXMLSTRINGS table.

(LOJ is used because Namespace and Location are optional)

CHECK DATA & REPAIR...

- CHECK DATA
  - **DB2 9**
    - CHECK DATA utility does not cover inconsistency inside XML data
    - NODEID index consistency with XML TS (CHECK INDEX)
    - Base table consistency with NODEID index (CHECK DATA)
  - **DB2 10 CHECK DATA** does:
    - Base table consistency with NODEID index (CHECK DATA)
    - NODEID index consistency with XML TS (V9 CHECK INDEX)
    - Check document structure consistency for each document
    - Schema validation if column(s) have a type modifier
      - *SHRLEVEL REFERENCE moves invalid documents to an automatically created exception table*
        - *With XMLERROR INVALIDATE or AUXERROR INVALIDATE*
      - *SHRLEVEL CHANGE will generate REPAIR statements*

CHECK DATA & REPAIR...

```
CHECK DATA TABLESPACE IOD05SDB.IOD05STS
  SHRLEVEL REFERENCE
  INCLUDE XML TABLESPACES ALL
  SCOPE XMLSCHEMAONLY
  SORTDEVT SYSDA SORTNUM 4
  WORKDDN (TSYSUT1,TSORTOUT)
  ERRDDN TSYSERR
```

- CHECK DATA (continued)
  - **DB2 10 CHECK DATA** options:
    - **INCLUDE XML TABLESPACES** (**ALL** | *specify XML table space or column*)
      - ***XMLSCHEMA*** *as part of the* XML table space or column spec *does schema checking*
      - ***XMLERROR INVALIDATE*** *(**SHRLEVEL REFERENCE**) removes nonconforming XML documents to an exception table*
        - *The base table space is places in AUXW status*
      - ***SHRLEVEL CHANGE*** *generates **REPAIR LOCATE..DOCID...DELETE** statements*
        - *The base table space status is not changed*
    - **SCOPE XMLSCHEMAONLY** for schema validation only
    - **SCOPE PENDING** (default)
      - *Will verify the NODEID index*
      - *If **INCLUDE XML TABLESPACES** is specified, all XML Schema validation is also done*
    - **EXCEPTIONS** <count> does not apply to XML
  - For MV format, only the latest version of a document is checked
  - Use **CHECK INDEX** to verify XML Value Indexes (user created XML indexes)
  - When table version is involved, the REPAIR statements from SHRLEVEL CHANGE must be used
    - This avoids versioning

Native SQL Procedure & UDF Enhancements

- XML is now valid as a passed variable
- XML can be a data type of a declared variable
- With these enhancements parsed XML can be processed within the routine without repetitively reparsing the serialized XML string

```
-- Declare variables
       DECLARE Return_SQLCODE_TMP  VARCHAR(500) DEFAULT ' ';
       DECLARE Return_Message_TMP  VARCHAR(500) DEFAULT ' ';
       DECLARE SQLCODE             INTEGER;
       DECLARE num_rows            INTEGER DEFAULT 0;
       DECLARE Parsed_XMLDOC           XML;
DECLARE EXIT HANDLER FOR SQLSTATE '23502'
  BEGIN
    SET Return_Message_TMP = Return_Message_TMP ||
    ' NULL INSERT Exit Handler Terminated routine!!';
    SET Return_SQLCODE = Return_SQLCODE_TMP;
    SET Return_Message = Return_Message_TMP;
  END;
  SET Parsed_XMLDOC = XMLPARSE(DOCUMENT XMLDOC);
```

This snippet of code shows an example of a DECLARE statement for the XML data type for a case when XML is received as a CLOB (*DB2 10 could pass the document in to the procedure as type XML*)

The body of the Native SQL Procedure then uses the XMLPARSE function to store a document in the variable.

Subsequent use of the Parsed_XMLDOC variable does not require parsing.

XML Date & Time Support

- DATE and TIME are not supported types in DB2 9 XPath
- DB2 10 provides
  - xs:dateTime
  - xs:time
  - xs:date
  - xs:duration
  - xs:yearMonthDuration
  - xs:dayTimeDuration
  - Several functions (fn) for comparison, value extraction, arithmetic operations
- Implicit time zone will be UTC if not specified
- These types can be used in the creation of XML Value Indexes

```
,xmlquery('fn:max(/DEPT/PROJ/EMP/xs:date(ENDDATE))
          - fn:min(/DEPT/PROJ/EMP/xs:date(STARTDATE))'
 passing xml_time_account) as "xs:date Duration"
```

```
CREATE INDEX XMLPOTX3
  ON TIME_ACCOUNTING_PART
  (XML_TIME_ACCOUNT)
  GENERATE KEYS USING XMLPATTERN
  '/DEPT/PROJ/EMP/STARTDATE' AS SQL DATE
```

```
SELECT DEPTNO
      ,ROW_TIMESTAMP
      ,XMLPOT_SEQUENCE
      ,XMLSERIALIZE(XML_TIME_ACCOUNT AS CLOB) AS XML_TIME_ACCOUNT
      ,XMLSERIALIZE(XML_AUDIT AS CLOB) AS XML_AUDIT
FROM TIME_ACCOUNTING_PART
WHERE XMLEXISTS('/DEPT[@DEPTNO="D21"]' PASSING XML_TIME_ACCOUNT)
  AND XMLEXISTS('/DEPT/PROJ/EMP[STARTDATE >= xs:date("1983-01-01")]'
                PASSING XML_TIME_ACCOUNT)
```

Binary XML & XQuery

- DB2 9 supported textual XML as input / output
  - This may cause the application to perform some additional parsing
- DB2 10 also supports binary XML as input / output
  - Smaller in size (up to 46%)
  - More efficient INSERT
    - CPU (up to 30%)
    - Elapsed (up to 50%)
- Binary XML used by default in the V10 JDBC Driver
- UNLOAD / LOAD support
- Validation supported with PM53282 /
  - See UA63422 / UA65591 for performance enhancement
- PM47618 added Embedded XQuery support

Additional pureXML Topics

- DB2 for z/OS pureXML Proof of Technology
  - One day learning opportunity
  - Brief lecture / followed by detailed labs
  - DB2 10 ready
    - XML Publishing
    - Creating / Altering XML columns
    - Unload and Load
    - XPath and function basics
    - Updating and Deleting XML
    - Tuning (XML Value Indexes & Explain)
    - Working with XML Schema validation
    - Decomposing XML via NSPs with the XMLTABLE function

pureXML, The Sequel – IBM DB2 Tools Support

- DB2 Administration Tool and DB2 Object Compare
  - Migrate, Alter, Compare
- DB2 Utilities Suite
  - Check
  - Repair
  - Unload, Load
  - Reorg, Copy……..
- DB2 Table Editor
  - Updates

Temporal Data – Time Travel Query

- What is temporal data?
- Business Time & System time
- What are the benefits of using the database in temporal data
- Example of a table with bi-temporal data

What is temporal data?

- One of the major improvements in DB2 10 will be the ability for the database to reduce the complexity and amount of coding needed to implement “versioned” data, data that has different values at different points in time.
- Data that you need to keep a record of for any given point in time
- Data that you may need to look at for the past, current or future situation
- The ability to support history or auditing queries
- Supporting Business Time and System Time

Benefits of using temporal tables …

- Move the logic from the application layer to the database layer
  - Consistent handling of temporal data
- Reduce Application development time by up to 10x
  - Application development can focus on business functions
- Run current applications with no code change
  - For System Time working with the current version of data
- Preserve execution time for current queries going after current data (System Time)
- You probably have these types of applications running in your shop

Benefits of using temporal tables …

- Business Problems you can solve with temporal tables
  - Ensure that a customer only has one financial position at a given time
  - Was an insured covered for a procedure on a specific date?
    - Was that information correct at the time the claim was processed?
  - Establish prices for a catalog ahead of time, so that they are completed before the change needs to be made
  - Answer a customer complaint about an old bill
  - … and many, many more

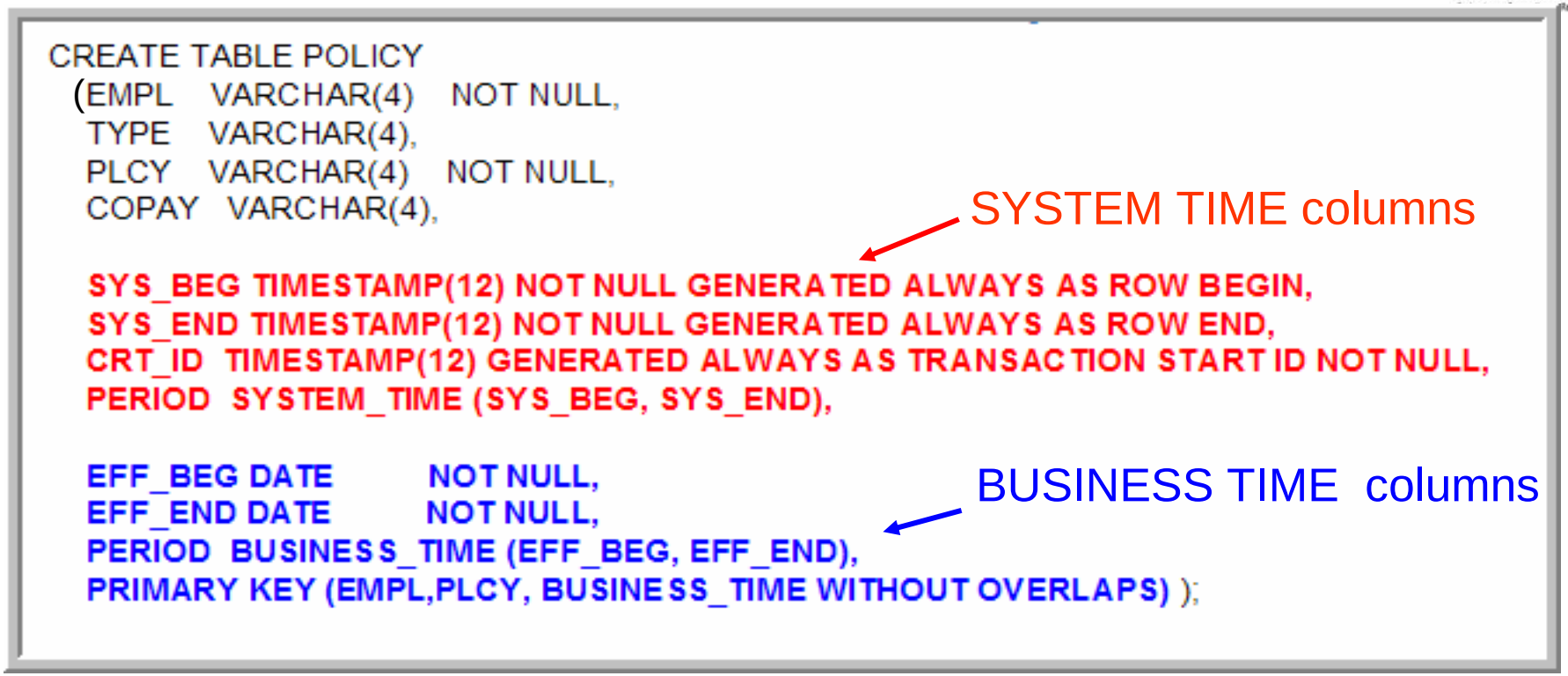
Basic Temporal Concepts

- Business Time (Effective Dates, Valid Time, From/To-dates)
  - Every row has a pair of TIMESTAMP(6) or DATE columns **set by Application**
    - Begin time : when the business deems the row valid
    - End Time : when the business deems row validity ends
  - Constraint created to ensure Begin time < End time
  - Query at current, any prior, or future point/period in business time
- System Time (Assertion Dates, Knowledge Dates, Transaction Time, Audit Time, In/Out-dates)
  - Every row has a pair of TIMESTAMP(12) columns **set by DBMS**
    - Begin time : when the row was inserted in the DBMS
    - End Time : when the row was modified/deleted
  - Every base row has a Transaction Start ID timestamp
  - PM31314 (9/2011) allows the use of TIMESTAMP WITH TIMEZONE
  - Query at current or any prior point/period in system time
- Times are inclusive for start time and exclusive for end times

Basic Temporal Concepts

- Bi-temporal
  - Inclusion of both System Time and Business Time in row
- Temporal Uniqueness
  - PK or Unique Key with BUSINESS_TIME WITHOUT OVERLAPS
  - Support for a unique constraint for a point in time
  - This is optional, however without it:
    - Unique constraints will likely return errors due to multiple rows per key
- History Table
  - Table to save “old” rows when using System Time

Table Defined with Business and System time



Adding the PERIOD BUSINESS_TIME clause enables business time.

Adding BUSINESS_TIME WITHOUT OVERLAPS guarantees there can only be one row for a given business time.

It is possible to define the TRANSACTION START ID (required for System Time) as NULLABLE.

Any System Time columns may also define as Implicitly Hidden.

ALTER TABLE ADD PERIOD... can be used to add Business / System Time periods to existing tables.

History table for SYSTEM TIME

```
CREATE TABLE POLICYHISTORY
 (EMPL     VARCHAR(4)     NOT NULL,
  TYPE     VARCHAR(4),
  PLCY     VARCHAR(4)     NOT NULL,
  COPAY    VARCHAR(4),
  EFF_BEG  DATE           NOT NULL,
  EFF_END  DATE           NOT NULL,
  SYS_BEG  TIMESTAMP(12)  NOT NULL,
  SYS_END  TIMESTAMP(12)  NOT NULL,
  CRT_ID   TIMESTAMP(12)  NOT NULL );
```

OR

```
CREATE TABLE POLICYHISTORY LIKE POLICY;
```

To enable SYSTEM TIME you then alter the table:

```
ALTER TABLE POLICY
  ADD VERSIONING USE HISTORY TABLE POLICYHISTORY;
```

- **The Table structures must be the same**
- **The Table must be in single-table, Table Spaces**
- **The Table Spaces do not have to have the same attributes**

Row Maintenance with System Time

- No temporal syntax for System Time maintenance
  - Use regular Update, Delete, Insert statements
- If the modification impacts existing base table rows

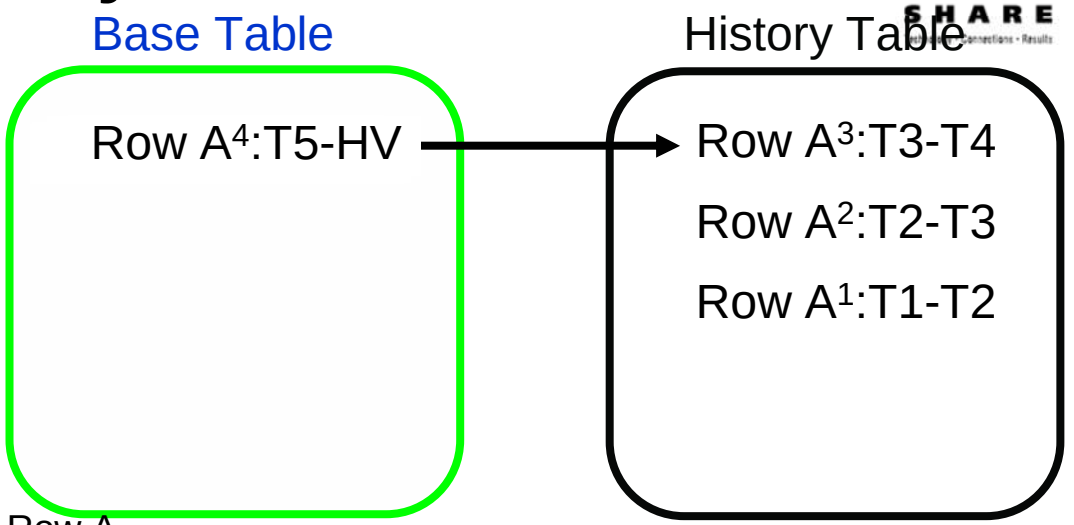
–Insert or Update	–The base table row(s) are created / updated with a UOW start time as System Start Time and a high value System End Time.
–Delete	–Remove the base table row.
–Update or Delete	–Create a “before-image” copy of all qualified base table rows in the History Table. –The newly created History row(s) are added with a System End Time equal to the UOW start time (System Start Time of the associated base table row for an update)

Update & Delete do the bulk of the System Time work.

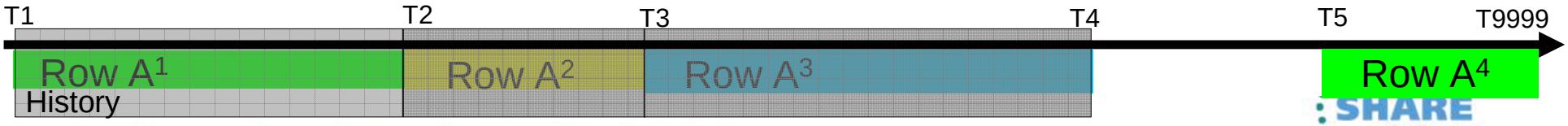
History table rows are a inserted copy of the Base table EXCEPT for the ST End.

Row Maintenance with System Time

- **T1: INSERT Row A**
- **T2: UPDATE Row A**
- **T3: UPDATE Row A**
- **T4: DELETE Row A**
- **T5: INSERT Row A**



- Notes:
  - INSERT has no History Table impact
  - The first UPDATE begins a lineage for Row A.
    - History Table ST End = Base Table ST Begin (No gap)
    - The Base Table ST End is always High Values (HV)
  - The second UPDATE deepens the lineage
    - No gaps exist across all generations of Row A.
  - The DELETE adds to the lineage in the History Table.
    - There is no current row (Base Table) after the DELETE
  - The second INSERT begins a new row lineage
    - There is a gap between the History Table rows and the Base Table
  - If all of the above statements happen in the same UOW, there would be no History Table rows



Row Maintenance with Business Time

- UPDATE/DELETE temporal syntax is used for Business Time maintenance
  - FOR PORTION OF BUSINESS_TIME FROM x TO y
- If the modification impacts existing base table rows

–Insert

–If a PK includes Business Time check for overlaps for the same PK of different base table rows

-803 returned if overlaps are found

–Insert the base row with the specified Begin & End Business Times

–Update / Delete

–Check the specified Business Time against existing qualified rows

–Rows **contained within** the specified Business Time range are updated / deleted

- row Business Time remains unchanged for the update

–Rows that **span the specified From OR To** Business Time are

- Updates: split into two rows, and updates applied to the portion of Business Time within the From and To

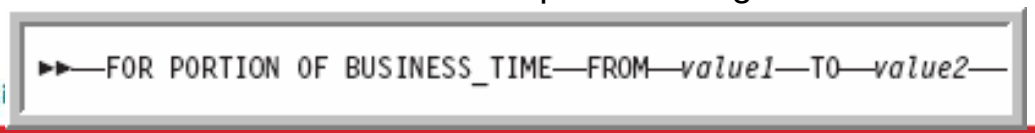
- Deletes: The Begin or End Business Time is updated so no portion of the specified range remains

–Row that **span the specified From AND To** are split into:

- Updates: three rows, and updates applied to the portion of Business Time within the From and To

- Deletes: two rows representing the remaining Business Time on either end of the specified range

In a bi-temporal implementation any changes to existing rows would also go through the System Time steps on the prior slide



Row Maintenance with Business Time

For each row that qualifies:

Rows Contained

FOR PORTION OF Business Time Row Result

Span FROM or TO

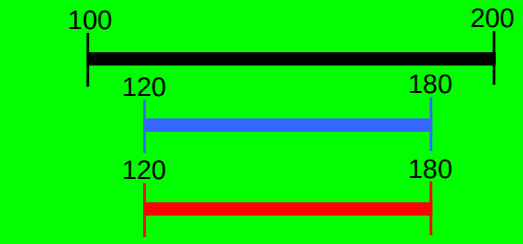
FOR PORTION OF Business Time Row Result

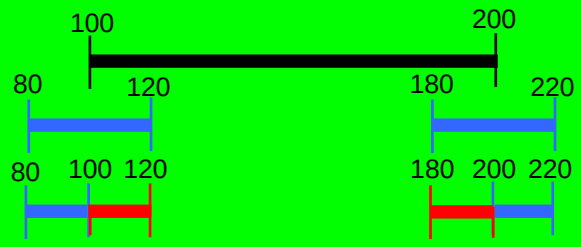
Span FROM and TO

FOR PORTION OF Business Time Row Result

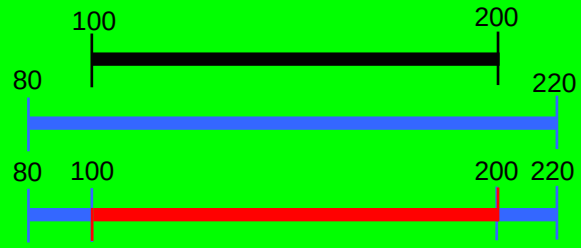
UPDATE

UPDATE <table...>
FOR PORTION OF BUSINESS_TIME
FROM <100> TO <200>
SET....
WHERE ....





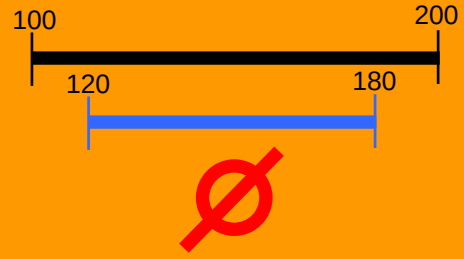
Executes UPDATE/INSERT, 1 row becomes 2



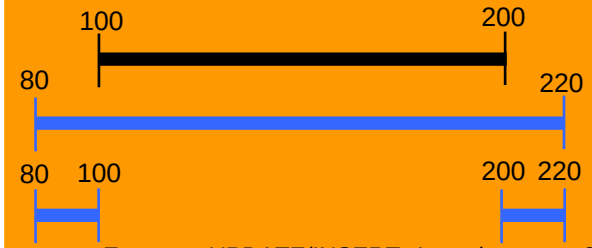
Executes UPDATE/INSERT/INSERT, 1 row becomes 3

DELETE

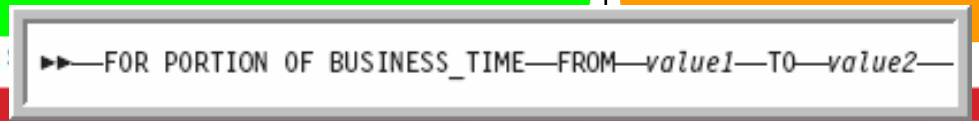
DELETE FROM <table...>
FOR PORTION OF BUSINESS_TIME
FROM <100> TO <200>
WHERE ....



Executes UPDATE



Executes UPDATE/INSERT, 1 row becomes 2



Bi-temporal example …

Step 1 – 9/21/2010 Employee C054 chooses HMO policy with $10 copay effective 1/1/2004

```
INSERT INTO POLICY
 (EMPL,TYPE,PLCY,COPAY,EFF_BEG,EFF_END)
  VALUES ('C054','HMO', 'P667', '$10', '1/1/2004', '12/31/9999');
```

Policy Table

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14.745082721000	9999-12-30-00.00.00.000000000000
				Business Time start	Business Time end	SYSTEM Time start	SYSTEM Time end

POLICYHISTORY table is empty at this point

Bi-temporal example

Step 1 – 09/21/2010 Employee C054 chooses HMO policy with $10 copay effective 1/1/2004
Step 2 – 09/24/2010 Update all P667 policies to a copay of $15 beginning 01/01/2011

Original Row

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14.745082721000	9999-12-30-00.00.00.000000000000

```
UPDATE POLICY FOR PORTION OF BUSINESS_TIME
FROM '01/01/2011' TO '12/31/9999'
SET COPAY='$15'
WHERE PLCY='P667';
```

Policy Table

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22.50672497000	9999-12-30-00.00.00.000000000000
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22.50672497000	9999-12-30-20000.00.000000000000

Policy History table

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14.745082721000	2010-09-24-17.33.22.50672497000

System Time / Point In Time...

```
SELECT * FROM POLICY FOR SYSTEM_TIME AS OF
    '2010-09-22-00.00.00.000000000000';
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

System Time / Point In Time...

```
SELECT * FROM POLICY FOR SYSTEM_TIME AS OF
   '2010-09-22-00.00.00.000000000000';
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

As of 09-22-2010 , the only row that qualifies is the row from the history table, because on 09-24-2010 we updated the rows, and both rows in the current table begin on 09-24-2010.

As of 09-24-2010-17.33 and after, rows from the current table would be returned

Only the POLICY appears in the SELECT statement. POLICYHISTORY is automatically accessed.

Results only come from the history table

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY

System Time / Point In Time …

```
SELECT * FROM POLICY FOR SYSTEM_TIME AS OF
    '2010-09-25-00.00.00.000000000000';
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

System Time / Point In Time

```
SELECT * FROM POLICY FOR SYSTEM_TIME AS OF
    '2010-09-25-00.00.00.000000000000';
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

As of 09-25-2010 , the only rows that qualify are the rows from the current table, because on 09-24-2010 we updated the rows, and both rows in the current table begin as of 09-24-2010.

The Base results are differentiated from the History results by the SYS_END column. The Base will reflect 9999-12-30-00.00.00...

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00

System Time / Range …

```
SELECT * FROM POLICY FOR SYSTEM_TIME
  FROM '2010-09-22-00.00.00.000000000000'
    TO '2010-09-25-00.00.00.000000000000';
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

QBLOCKNO	PLANNO	METHOD	CREATOR	TBNAME	TABNO	ACCESSTYPE	PREFETCH	QBLOCK_TYPE
1	1	0	DBA015	POLICYHISTORY	2	R	S	NCOSUB
2	1	0			0			UNIONA
5	1	0	DBA015	POLICY	1	R	S	NCOSUB

System Time / Range

```
SELECT * FROM POLICY FOR SYSTEM_TIME
  FROM '2010-09-22-00.00.00.000000000000'
    TO '2010-09-25-00.00.00.000000000000';
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

For this query, we look for rows where:

System Start Time (SYS_BEG) < 9/25 AND

System End Time (SYS_END) > 9/22

Result of query

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

Bi-temporal example …

Step 1 - 09/21/2010 Employee C054 chooses HMO policy with $10 copay effective 1/1/2011
Step 2 - 09/24/2010 Update all policies P667 to a copay of $15 beginning 01/01/2011
Step 3 - 09/24/2010 Later on the same day(19.44) of 09/24, the customer cancelled the policy

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

```
DELETE FROM POLICY FOR PORTION OF BUSINESS_TIME
FROM CURRENT DATE TO '12/31/9999'
WHERE EMPL='C054' AND PLCY='P667';
```

Bi-temporal example …

Step 1 - 09/21/2010 Employee C054 chooses HMO policy with $10 copay effective 1/1/2011
Step 2 - 09/24/2010 Update all policies P667 to a copay of $15 beginning 01/01/2011
Step 3 - 09/24/2010 Later on the same day(19.44) of 09/24, the customer cancelled the policy

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	9999-12-30-00.00.00	BASE

```
DELETE FROM POLICY FOR PORTION OF BUSINESS_TIME
FROM CURRENT DATE TO '12/31/9999'
WHERE EMPL='C054' AND PLCY='P667';
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	2010-09-24-19.44.47	HISTORY
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	2010-09-24-19.44.47	HISTORY
CO54	HMO	P667	$10	2004-01-01	2010-09-24	2010-09-24-19.44.47	9999-12-30-00.00.00	BASE

Business time example

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	9999-12-31	2010-09-21-21.50.14	2010-09-24-17.33.22	HISTORY
CO54	HMO	P667	$10	2004-01-01	2011-01-01	2010-09-24-17.33.22	2010-09-24-19.44.47	HISTORY
C054	HMO	P667	$15	2011-01-01	9999-12-31	2010-09-24-17.33.22	2010-09-24-19.44.47	HISTORY
CO54	HMO	P667	$10	2004-01-01	2010-09-24	2010-09-24-19.44.47	9999-12-30-00.00.00	BASE

```
SELECT * FROM POLICY  FOR BUSINESS_TIME AS OF
    '2010-09-23' ORDER BY EFF_BEG;
```

EMPL	TYPE	PLCY	COPAY	EFF_BEG	EFF_END	SYS_BEG	SYS_END	Which Table
CO54	HMO	P667	$10	2004-01-01	2010-09-24	2010-09-24-19.44.47	9999-12-30-00.00.00	BASE

```
SELECT * FROM POLICY  FOR BUSINESS_TIME AS OF
    '2011-09-25' ORDER BY EFF_BEG;
```

No rows returned.

Business time only looks at base table, not the history table

System Period Versioning Information...

- Base and History tables must be RECOVERed as a set
  - VERIFYSET NO can override the need to RECOVER together
- No utility operations that deletes data from base table
  - LOAD REPLACE
  - REORG DISCARD
  - CHECK DATA DELETE YES
- No CHECK utilities that invalidate AUX/LOB/XML
- PM31313 allows for ALTER TABLE...ADD COLUMN
  - The same column is added to the History Table
- PM31314 changed the high value to '9999-12-30-00.00.00.000000000000'
- The Base and History table, table spaces must be single table
- No temporal SELECT, UPDATE, or DELETE against the History
- Cannot be an MQT
- Cannot have a Clone Table, Column Mask, Row Permission, or Security Label column

System Period Versioning Information...

- Cannot TRUNCATE
  - INSERT, UPDATE, DELETE, and MERGE are accepted
- To find the Base / History Tables

- System Time can be altered on an existing table
  - See Information Center topic: http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.db2z10.doc.admin/src/tpc/db2z_addingsystime.htm
- QUIESCE of the Base or History
  - Will cause a quiesce against all tables in the versioning relationship, including auxiliary spaces

```
SELECT VERSIONING_SCHEMA,
       VERSIONING_TABLE
FROM SYSIBM.SYSTABLES
WHERE NAME    = 'table-name'
  AND CREATOR = 'creator-name'
```

System Period Versioning Information

- REPORT TABLESPACESET identifies versioning relationships in the system-maintained temporal table space or history table space
- CURSORs & VIEWs referencing system temporal table with a period specification will be READ ONLY

Business Period Versioning Information

- Business Time can be altered on an existing table
  - See Information Center topic: http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.db2z10.doc.admin/src/tpc/db2z_alteringtableaddbusitime.htm
- Consider the implications of non-temporal UPDATE & DELETE statements
  - These statements are allowed
- SQLERRD(3) does not reflect rows added due to a temporal UPDATE / DELETE
  - Consistent with RI handling
- It is possible to have contiguous Business Time ranges with the same non-temporal data in the row
- Should adopt the same high value as System Time
  - 9999-12-30-00.00.00.000000000000

Customers love it

The new temporal functionality in DB2 10 for z/OS will allow us to **drastically simplify** our date-related queries. In addition, we'll be able to **reduce our storage costs** by using cheaper storage for inactive rows and reduce our processing cost by having DB2 handle data movement more efficiently than the custom code we've written to do the same work in the past

Large Insurance Company - DB2 10 Beta Customer

bankdata

"We are really thrilled about "Temporal Data" feature – this feature has the potential to **significantly reduce overheads**. We have estimated that **80% of our existing temporal applications** could have used "the DB2 10 temporal features" instead of application code - this feature will **drastically save developer time, testing time** – and even more importantly make applications easier to understand so **improve business efficiency and effectiveness**"

Additional Temporal Topics

- DB2 for z/OS Temporal Proof of Technology
  - One day learning opportunity
  - Brief lecture / followed by detailed labs
  - Temporal Labs
    - Implementing Business Time
    - Implementing System Time
    - Working with Business Time (showing System Time impacts)
    - Working with System Time (discover row lineage)
    - Temporal Simulation

Temporal Data – IBM DB2 Tools Support

- DB2 Administration Tool and Object Compare
  - Alter, Migrate, Compare, Create
- DB2 Table Editor
  - Insert, Update, Delete
- SQL Performance Analyzer
  - Business Time & System time
- Log Analysis Tool
  - Externalized Business and/or System Time
- Data Studio (3.1.1)
  - Versioning properties, SQL assist, etc.

LOB Enhancements

- Inline LOBs
- Benefits
- Inline LOB DDL
- Inline LOB Support
- Streaming LOB Data

Business value statements

- The use of inline LOBs reduces I/O & CPU consumption since LOB data can now be stored in the base table
- This enables new workload that employs LOB data
- Eases LOB handling and management

Benefits of Inline LOBs

- Reduces auxiliary table storage, CPU, and virtual storage potential
- Inline portion of LOB compressible even though LOBs are not compressed
- Inline portion of LOB can be used in index on expression
  - For example, when using spatial data the index portion of the spatial object can reside in the row
- Inline LOBs stored in the base table can achieve similar performance characteristics as VARCHAR columns

Inline LOBs...

- LOB data can also be split
  - Entirely or partially
    - Split LOB

  - DASD storage associated with small LOBs may be reduced
  - Avoids I/O's associated with LOB table spaces
  - Base table can become larger.
    - SQL limited to non-LOB columns may be impacted

Inline LOB DDL…

- Pre-requisites
  - DB2 10 NFM
  - RRF
  - Default value of other than NULL or empty string is supported
  - Define length of inline portion of LOB
    - DDL
      - *Default to ZPARM*
      - *Valid range 0 – 32680 bytes*
    - New ZParm **LOB_INLINE_LENGTH**
      - *Default 0 bytes*
        - *No inline length. Stored entirely in AUX*
      - *Range 0 – 32680 bytes*
  - Restrictions
    - Cannot alter shorter if used in index on expression or shorter than default data value

Inline LOB DDL…

- Create table with LOB column & Inline portion

```
CREATE TABLE mytable (
ID SMALLINT,
mylobcolumn CLOB(1M) INLINE LENGTH 1000);
```

- Alter table INCREMENT length of Inline LOB portion

```
ALTER TABLE mytable
ALTER mylobcolumn
SET DATA TYPE CLOB(1M) INLINE LENGTH 2000;
```

- Alter to larger size is an immediate change. Base table space is put in AREO* status. If rows exist at time of change, length change is materialized upon REORG

Inline LOB DDL…

- Alter table DECREMENT length of Inline LOB portion

```
ALTER TABLE mytable
ALTER mylobcolumn
SET DATA TYPE CLOB(1M) INLINE LENGTH 200;
```

  - Hard REORG pending status (REORP)
- Alter add a lob column with inline portion

```
ALTER TABLE mytable
ADD COLUMN mylobcolumn2
CLOB(1M) INLINE LENGTH 2000;
```

  - AREO* status for table space

Additional LOB Flexibility

- AUX (and XML) table spaces & dependent indexes can be created DEFINE NO
  - Defers create time to first INSERT operation
  - Improves application installation time
  - Simplifies/improves backup as empty data sets don't have to be backed up
- LOAD/UNLOAD LOBs along with non-LOB columns
  - Spanned record support **UTL**

Streaming LOB (& XML)…

- LOB streaming first introduced in DB2 V8
  - Allowed programs to read LOB objects from applications without knowing their size first
  - Streaming occurs to DDF (out), not to the engine (in)
  - Had to materialize it entirely before sending to DB layer
- In DB2 10 LOBs will be streamed to the engine from DDF without needing to materialize it in memory
- Customers who use LOB & XML objects in DB2 10 will use fewer resources
  - DB2 10 eliminates materialization for LOB (>2MB) and XML objects (>32KB)
  - Class 2 CPU time reduced
  - Virtual storage reduction

Streaming LOB (& XML)

- Streaming between DDF and DBM1 for LOB/XML objects occurs in the following situations
  - INSERT, UPDATE
  - Limited to max 1 LOB or 1 XML column per row
  - LOAD utility using file reference variables
  - When CCSID conversion is required by the LOAD utility
  - When inserting XML data in a DRDA® environment at the remote server

LOB Enhancements – IBM DB2 Tools Support

- DB2 Admin and DB2 Object Compare
  - Inline LOBs
- DB2 Table Edit
  - Insert, Update, Delete
- DB2 Utilities Suite
- DB2 Cloning Tool
- DB2 Recovery Expert

Optimization Evolution

- Literal Replacement
- Access Path Stability Improvements
  - Optimization Hints
  - Plan Management Policies
- Index Usage Improvements
- SQL Pagination (Complex OR)
- Dynamic Row Level Sequential Detection
- Parallelism
- Statistics Enhancements
- Other BIND / REBIND Changes
- EXPLAIN Changes
- Aggressive Merge

Business Value of Optimization Evolution

- Dynamic and Static enhanced workload stability
- Performance via exploitation of enhanced index usage
- More parallelism value via better workload distribution
- Reduced administration for statistics maintenance

What customers are saying...

"Over the past several months, BMW has tested the new version of DB2 10 for z/OS, focusing on specific features and comparing these directly to the same features in DB2 9 for z/OS. One of the IBM design goals expected a general improvement in massive parallel SQL-insert performance, where we achieved **close to 40% CPU** improvement and significant elapse time reduction in direct comparison to DB2 9 for z/OS. For all of our critical tested selects statements, the version 10 optimizer chose the optimal access path, sometimes even improving previous access path choices in version 9. Overall, we are very pleased with the added functionality and architectural enhancements, and are looking forward to this exciting release."

- Philipp Nowak -BMW Group DB2 Product Manager

ITERGO

"We are really excited about the **Plan Stability enhancements** in DB2 10 - we have a sophisticated procedure to assure quality in access path - this means, all packages run with their optimal access path, to assure this, we have to look at each new statement and maybe at each new rebind - this can be a time-consuming partly manual process. **Plan stability** offers an opportunity to **reduce** the daily effort and to **improve the DBA's productivity**"

Walter Janissen- ITERGO Informationstechnologie GmbH

Literal Replacement...

- Dynamic SQL statements with literals can now be made more reusable with literal replacement
- Literals are replaced with a “&” and stored in DSC
- Enabled via:
  - PREPARE: CONCENTRATE STATEMENT WITH LITERALS in the ATTRIBUTES
  - ODBC: LITERALREPLACEMENT specified in the initialization file
  - JCC Driver: enableLiteralReplacement=‘YES’
- Requires 2 DSC lookups
  - A DSC match for the original statement (with literals) happens first
  - If no exact statement match occurs, the literals are replaced and a second lookup occurs
    - This must match another statement with literal replacement, not parameter markers
    - Also requires:
      - *Matching criteria similar to parameter markers (SQL text and auth ID)*
      - *The Literal Reusability Context must be the same.*
        - *The data type and lengths of the literal replacements must be compatible*

Literal Replacement...

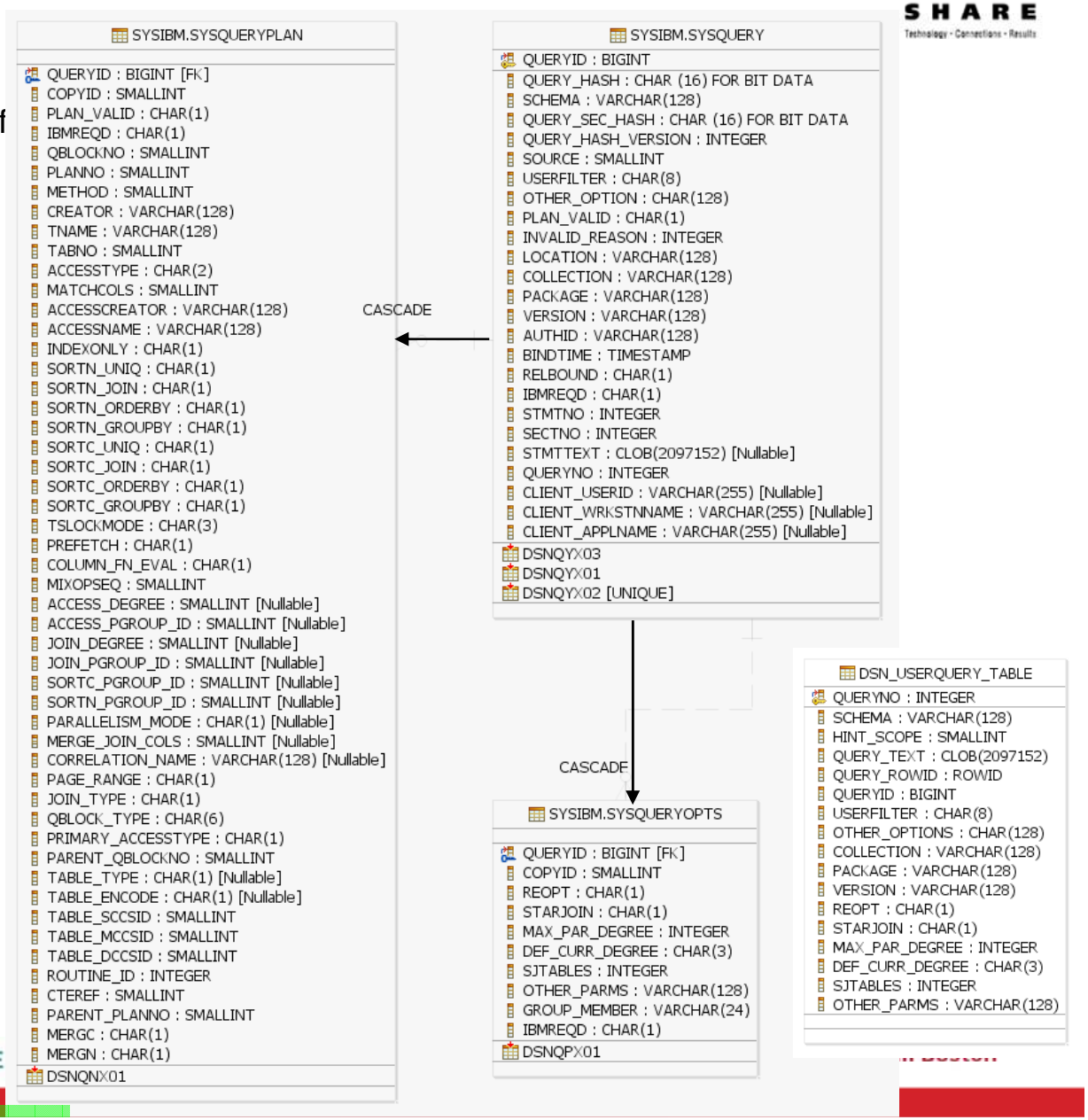
- Statements that benefit by including literal values will run better:
  - Without literal replacement
  - Or with REOPT (all REOPT options supported with literal replacement)
  - These include statements that benefit from distribution or correlation statistics
- Better suited statements are likely smaller workload SQL statements with literal that could benefit from DSC
- EXPLAIN STATEMENT CACHE ALL will populate DSN_STATEMENT_CACHE
  - The LITERAL_REPL column will indentify literal replacement SQL
- Cannot have literal replacement and parameter markers
  - For cases with parameter markers and literals
    - The literals are considered constants that do not change and the values are provided to the optimizer for a better filtering estimate
- No Optimization Hints or EXPLAIN PLAN FOR

Access Path Stability Improvements (APSI)...

- Static SQL provides many benefits, including:
  - Performance
  - Security
  - Predictability / Stability
- Optimization Hints
  - Pre- DB2 10 offered User Level Optimization Hints
  - DB2 10 introduces Statement Level Optimization Hints
    - Access Path Repository
      - *Statement level hints with **BIND QUERY** for static or dynamic SQL*
      - *Can enforce an existing access path from a PLAN_TABLE (hints) or*
      - *Can customize optimization options for a statement*

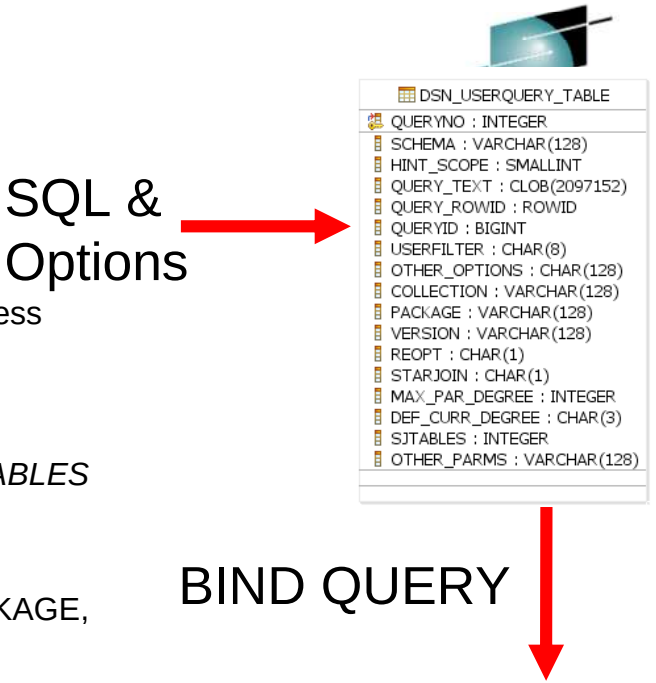
APSI / Optimization Hints...

- The Access Path Repository (APR) is implemented during the ENFM mode of migration
  - SYSIBM.SYSQUERY
  - SYSIBM.SYSQUERYPLAN
  - SYSIBM.SYSQUERYOPTS
- The new input table, DSN_USERQUERY_TABLE, is under the schema of a userid
- SYSQUERY
  - Contains data for all optimization hints
- SYSQUERYPLAN
  - Contains data for optimization hints to enforce an access path
- SYSQUERYOPTS
  - Contains data for hints that customize optimization parameters



APSI / Optimization Hints...

- Statement level stability
- Place a SQL Statement into DSN_USERQUERY_TABLE
  - QUERYNO
    - Match a PLAN_TABLE QUERYNO (with the same AuthID) to enforce an access path
    - Use a QUERYNO that does NOT match a PLAN_TABLE QUERYNO to set optimization parameters
      - *REOPT, STARJOIN, MAX_PAR_DEGREE, DEF_CURR_DEGREE, SJTABLES*
  - HINT_SCOPE:
    - 0: System level access plan hint (matching on statement and schema)
    - 1: Package level access plan hint (matching on optional COLLECTION, PACKAGE, and VERSION)
  - QUERY_TEXT:
    - SELECT
    - INSERT with fullselect
    - DELETE
    - UPDATE
    - MERGE
    - TRUNCATE
- Populate the APR with BIND QUERY
- For dynamic SQL in Dynamic Statement Cache, QUERY will remove the SQL from DSC



APSI / Optimization Hints...

- Statement Level Optimization Hints enablement
  - Create the DSN_USERQUERY_TABLE
  - Same enablement as User Level Optimization Hints
    - OPTHINT System Parameter
    - Add PLAN_TABLE_HINT_IX index to the PLAN_TABLE (Managing Performance Guide)
- Optimization Parameter Hints (SYSQUERYOPTS)
  - When successfully bound, the QUERYID column is updated in DSN_USERQUERY_TABLE with the value established in the catalog
- BIND QUERY LOOKUP(YES)
  - Will look for the DSN_USERQUERY_TABLE rows in the Access Path Repository
  - Messages (DSNT280I / DSNT281I) will be produced indicating which rows exists in the APR
- After a BIND QUERY
  - Consider purging the DSN_USERQUERY_TABLE. This will avoid replacing hints for existing statements during the next BIND QUERY
  - Customized USERFILTER value can be set to group optimization hints
- FREE QUERY can be used to remove optimization hints
  - By ALL, specific QUERYID, or USERFILTER value

```
DSNT280I  DSNT LOOKUP QUERY FOR QUERYNO = 0 SUCCESSFUL
DSNT281I  DSNT LOOKUP QUERY FOR QUERYNO = 201111296 NOT SUCCESSFUL,
            REASON:  THE MATCHING ROWS IN THE SYSIBM.SYSQUERY TABLE
AND THE SYSIBM.SYSQUERYPLAN TABLE ARE NOT FOUND
DSNT290I  DSNT LOOK QUERY COMMAND COMPLETED
```

Access Path Stability Improvements (APSI)...

- Plan Management Policies (PLANMGMT)
  - DB2 9 delivered Access Path / BIND Stability (PLANMGMT) via REBIND
    - Delivered in maintenance stream via PK52523 via REBIND
    - PLANMGMT Rebind keyword or System Parameter
      - *BASIC: Keeps 2 copies of a package (Current and Previous)*
      - *EXTENDED: Keeps 3 copies (Current, Previous, and Original)*
    - Use the Profile tables to set up automatic Plan Management Polices
  - DB2 10 evolves this capability
    - More formal Catalog implementation for PLANMGMT (V9 delivery)
      - *SYSPACKCOPY is populated as part of the ENFM process.*
    - Old copies that are from before V6 will still be invalid

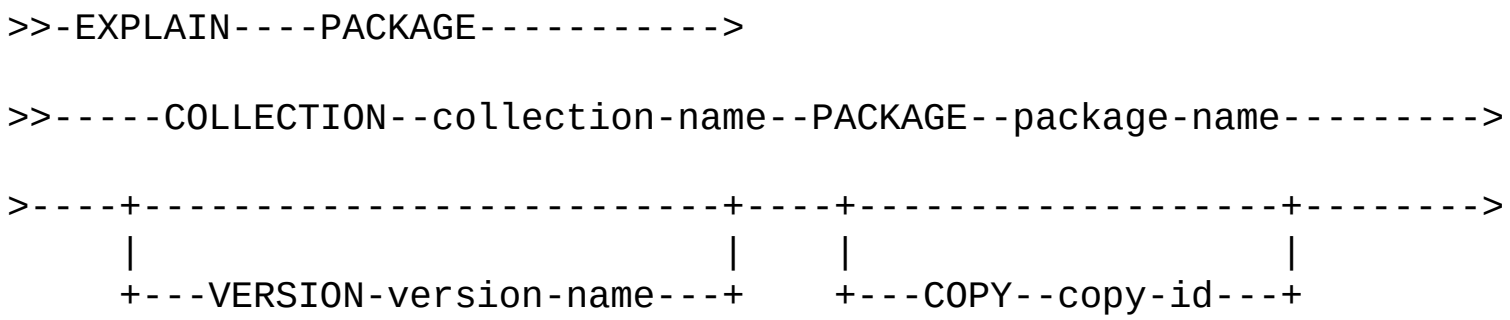
APSI / Plan Management Policies...

- New Catalog Table
  - SYSPACKCOPY
    - Package copy information from V9 is copied into SYSPACKCOPY during ENFM
    - COPYID column
      - *1: Indicates a Previous copy*
      - *2: Indicates an Original copy*
- PLANMGMT
  - System Parameter options are the same as V9 *with default change in CM*
    - OFF (was V9 default)
    - BASIC
    - EXTENDED (V10 default)
      - *Copies are kept by default in DB2 10*
  - REBIND changes
    - APRETAINDUP for PLANMGMT(BASIC or EXTENDED)
      - *APRETAINDUP keyword defaults to YES*
      - *NO: A new access path identical to a Current access path does not get copied to a Previous or Original*

APSI / Plan Management Policies

- PLANMGMTSCOPE
  - New System Parameter and Profile Attribute
  - For now, the only option (and default) is STATIC (V9 behavior)
- PLANMGMT extended to Native SQL Procedures
- EXPLAIN PACKAGE command
  - To retrieve PLAN_TABLE records for existing package
    - Original bind/rebind may have been EXPLAIN(NO)
    - Must have been bound on V9 or V10 to work.
    - Populate explain tables without BIND or REBIND.

SEC

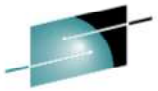


OPT

Access Path Stability Improvements...

- APCOMPARE / APREUSE
  - Options to get stabilize access paths across BIND or REBIND
  - The replaced package must be from DB2 9 or later
  - APCOMPARE
    - Structural comparison of the access path across the BIND / REBIND
    - When differences are discovered (for existing statements), the outcome depends on the option used
      - *WARN: Package is created with an RC=04*
      - *ERROR: No package is created with an RC=08*
  - APREUSE
    - Hints all statements for the new package using the access path from the package to be replaced
    - When differences are discovered (for existing statements), no package is created
      - *ERROR is the only option*

Access Path Stability Improvements...



- APCOMPARE / APREUSE...
  - BIND / REBIND & REBIND TRIGGER PACKAGE options
    - NO/NONE, WARN (APCOMPARE only), ERROR
  - Uses the pre-BIND / REBIND current package for comparison or reuse to determine the next package
  - The previous (pre-BIND / REBIND) package must have been bound / rebound at the DB2 9 level or later
  - Starting in DB2 9 an Explain Data Block (EDB) is kept in the package in SPT01
    - Internal hint
    - A compressed, internal representation of the PLAN_TABLE
  - PM25679 adds this new function
  - PM30425 corrective maintenance to enhance package reusability
    - Packages must have been bound / rebound on this maintenance before expecting this reuse
    - Ex. Information about Virtual Tables and corrected COLUMN_FN_EVAL

Access Path Stability Improvements

- APCOMPARE / APREUSE
  - Success or failure is at the package level
  - Comparisons are done at the statement level
  - Feedback is placed in the PLAN_TABLE if **EXPLAIN YES** or **ONLY** specified
    - HINT_USED contains ‘APREUSE’ when hint succeeds
    - REMARKS used for reuse or comparison failure information
    - These are placed on the ‘new’ PLAN_TABLE rows
    - If an ‘old’ PLAN_TABLE row is unmatched in the ‘new’ set of rows
      - *A new row is created to show the REMARKS*
  - These are not “sticky”; must specify for each BIND/REBIND
  - AUTOBIND defaults to **NO**

Access Path Stability Improvements

- APCOMPARE (NO/NONE, WARN, ERROR)
  - Determine if the static SQL statements have an access path change
  - This is an EDB structural comparison
    - There could be some details of the access path that are different
    - Examples:
      - *MERGE_JOIN_COLS=3; Specific columns & order???*
      - *Only explainable statements can be compared*
  - Compares statements that are exactly the same
    - Statement order in the package does not matter
  - If differences are found:
    - WARN results in an RC=04 and continues
      - *Recommend the use of a Plan Management Policy with REBIND because a new package (and access path) will be created*
    - ERROR results in an RC=08 and that specific BIND/REBIND is terminated
    - Use **EXPLAIN YES** or **ONLY** to get failure reasons

Access Path Stability Improvements

- APCOMPARE
  - DSNT285I lists
    - # of statements successfully compared
    - # of statements with unsuccessful comparison
    - # of statements that could not be compared
      - *Examples include:*
        - *Bound prior to V9*
        - *VALIDATE(RUN)*
        - *New statement during a BIND*
  - The ‘old’ package for comparison is found based on
    - REPLACE or COPY..COPYVER
      - *LOCATION, COLLID, NAME, VERSION*
    - ADD
      - *LOCATION, COLLID, NAME*
      - *DSNT294I reports the VERSION found for comparison*
    - If Plan Management is in use, it always uses the ‘current’, ‘old’ copy.

Access Path Stability Improvements

- APREUSE (NO/NONE, ERROR)
  - Applies a hint from the EDB for the new package
  - Attempts to avoid an access path change
    - Not guaranteed due to:
      - *Missing objects*
      - *Version incompatibilities*
      - *Hint ambiguities (ex. Columns of a merge join)*
  - Can get the same errors as when using OPTHINTs
    - +395 / Reason Code
    - Reason Code 50 added for comparison failure
  - ERROR & unable to reuse an explainable statement results in RC=08
    - That particular BIND / REBIND fails

Access Path Stability Improvements

- APREUSE
  - Matches:
    - LOCATION, COLLID, NAME, VERSION
    - If no match, then LOCATION, COLLID, NAME
    - If no match, DSNT292I and operation continues
  - For packages that bind / rebind successfully
    - SYSPACKSTMT ACCESSPATH='A'
    - SYSPACKSTMT & SYSPACKCOPY REUSE column set
    - Hints are not enforceable 100% of the time
    - DB2 10 may merge query blocks or decorrelate subqueries
- Migration and APCOMPARE / APREUSE
  - Consider using REBIND...EXPLAIN(YES)...APREUSE(ERROR)
  - Then analyze the REBIND failures or use APCOMPARE(WARN)

Index Usage Improvements...

- Safe Query Optimization
  - At BIND / Prepare risk is now accessed by predicate
    - This could choose a index with a slightly higher calculated optimization cost
    - Choosing a lower risk index for a more robust index choice
  - During runtime
    - RID overflows now use a workfile to continue processing
      - *Avoids fallback to table space scan*
    - May increase workfile usage
      - *MAXTEMPS_RID ZParm for maximum WF usage for each RID list*
      - *Sort-type workfile usage (as opposed to DGTT-type)*
      - *Increase is expected to be small*
      - *Note that RID Pool default in V10 has increased from 8MB to 400MB*

Index Usage Improvements...

- Predicate Transitive Closure for IN Lists
  - IN Lists can be generated for columns that are indicated as the same by another predicate
    - Predicate transitive closure already supported for =, BETWEEN, <, <=, >, >=
  - With the additional predicate, the optimizer may choose DEPT or EMP as the first table accessed
    - Provides new choices to optimizer not previously available

```
SELECT *
FROM DEPT A
    ,EMP B
WHERE A.DEPTNO = B.WORKDEPT
  AND B.WORKDEPT IN (?,?,?)
;
```

Rewritten

```
SELECT *
FROM DEPT A
    ,EMP B
WHERE A.DEPTNO = B.WORKDEPT
  AND B.WORKDEPT IN (?,?,?)
  AND A.DEPTNO IN (?,?,?)
;
```

DSN_PREDICAT_TABLE

PREDNO [INTEGER]	TYPE [CHAR(8)]	LEFT_HAND_SIDE [VARCHAR(128)]	RIGHT_HAND_SIDE [VARCHAR(128)]	ADDED_PRED [CHAR(1)]
1	AND			N
2	EQUAL	DEPTNO	WORKDEPT	N
3	IN	WORKDEPT	VALUE	N
4	IN	DEPTNO	VALUE	Y

Index Usage Improvements...

- IN List Direct Table Access & List Prefetch
  - In memory tables used to process IN Lists predicates as matching predicates
    - TNAME DSNINnnn(QB): nnn=predicate # ; QB=Query Block
    - TABLE_TYPE="I"
  - ACCESSTYPE "IN" for IN List Direct Table Access
  - Matching on multiple IN List predicates (MATCHCOLS)
  - List Prefetch used (PREFETCH)
  - Other IN-list access paths continue to use pre-V10 IN-list access (ACCESSTYPE='N')

```
SELECT *
FROM EMP
WHERE JOB IN (?,?,?)
  AND EMPNO IN (?,?,?)
;
```

PLAN_TABLE

PLANNO [SMALLINT]	METHOD...	CREATOR ...	TNAME [VARCHAR(128)]	TABNO [SMALLINT]	ACCESSTYPE...	MATCHCOLS...
1	0	DBA015	DSNIN003(01)	2	IN	0
2	1	DBA015	DSNIN002(01)	3	IN	0
3	1	DBA015	EMP	1	I	2

ACCESSCREATOR...	ACCESSNAME ...	PREFETCH ...	TABLE_TYPE
			I
			I
DBA015	EMP	L	T

EMP index on (EMPNO, JOB) →

SQL Pagination

- Prior to DB2 10 OR predicates often result in multi-index access with list prefetch
- DB2 10 supports single index access without list prefetch
- Range List Index Scan requires
  - Every OR predicate
    - has at least one matching predicate
    - is mapped to the same index
  - Therefore this isn't just for positioning or paging.
- ACCESSTYPE="NR" (iN list Range)
  - At runtime, the ordering of the access path steps will be determined
  - PLAN_TABLE represented the order coded in the query
  - Ordering will be based on literal values and the order of the index chosen

```
SELECT *
FROM EMP
WHERE
   (    LASTNAME = ?
    AND FIRSTNME = ?
    AND MIDINIT  > ?
   )
OR
   (    LASTNAME = ?
    AND FIRSTNME > ?
   )
OR
   LASTNAME > ?
ORDER BY LASTNAME
         ,FIRSTNME
         ,MIDINIT
;
```

EMPX1 Index (LASTNAME, FIRSTNME, MIDINIT)

MIXOPSEQ	METHOD	TABLE_NAME	TABNO	ACCESSTYPE	MATCHCOLS	INDEX_NAME
1	0	EMP	1	NR	3	EMPX1
2	0	EMP	1	NR	2	EMPX1
3	0	EMP	1	NR	1	EMPX1

OPTIMIZE FOR 1 ROW

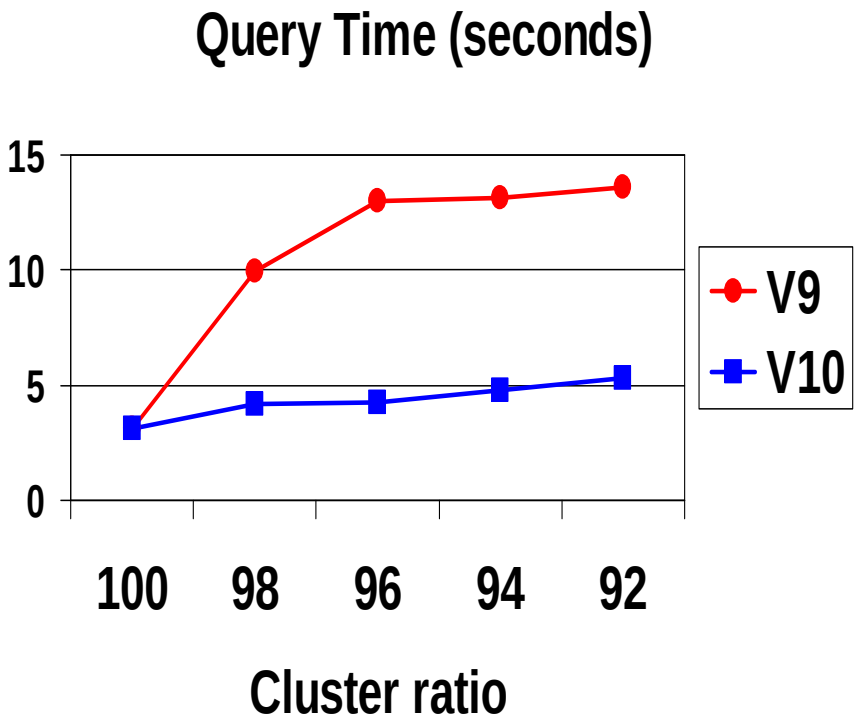
- Prior to DB2 10 OF1N discouraged sorting
  - A sort could still be chosen when most effective for the query
- DB2 10 did not choose sort when OF1R is specified
  - Even when it might be most effective
- PM56845
  - Introduces ZParm OPT1ROWBLOCKSORT
    - ENABLE – DB2 10 behavior of no sorting
    - DISABLE – Pre-DB2 10 behavior where sort is discouraged

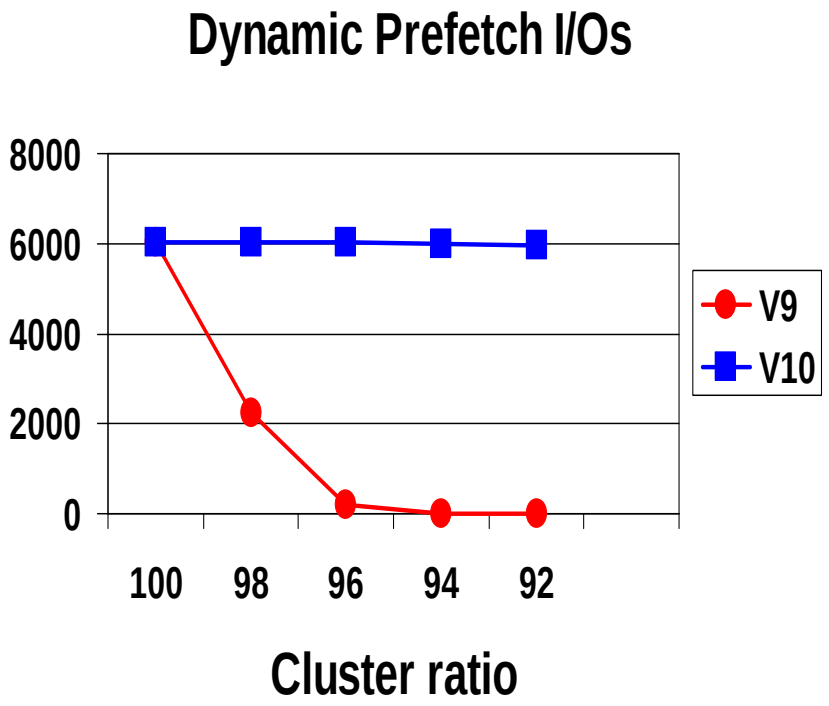
Dynamic Prefetch: Row Level Sequential Detection (RLSD)...

- Problem: Dynamic sequential prefetch could work poorly when the number of rows per page is large
- Solution: DB2 10 has row level sequential detection (RLSD)
  - Count rows, not pages
  - Unclustered row is less likely to cause DB2 to fall out of prefetch
- Since DB2 10 will trigger prefetch more quickly, it will use progressive prefetch quantity:
  - For example, with 4K pages the first prefetch I/O reads 8 pages, then 16 pages, then all subsequent I/Os will prefetch 32 pages
  - Progressive prefetch also applies to indexes (but still counts pages)

Row Level Sequential Detection

Row size = 392 bytes, page size = 32K (83 rows per page)





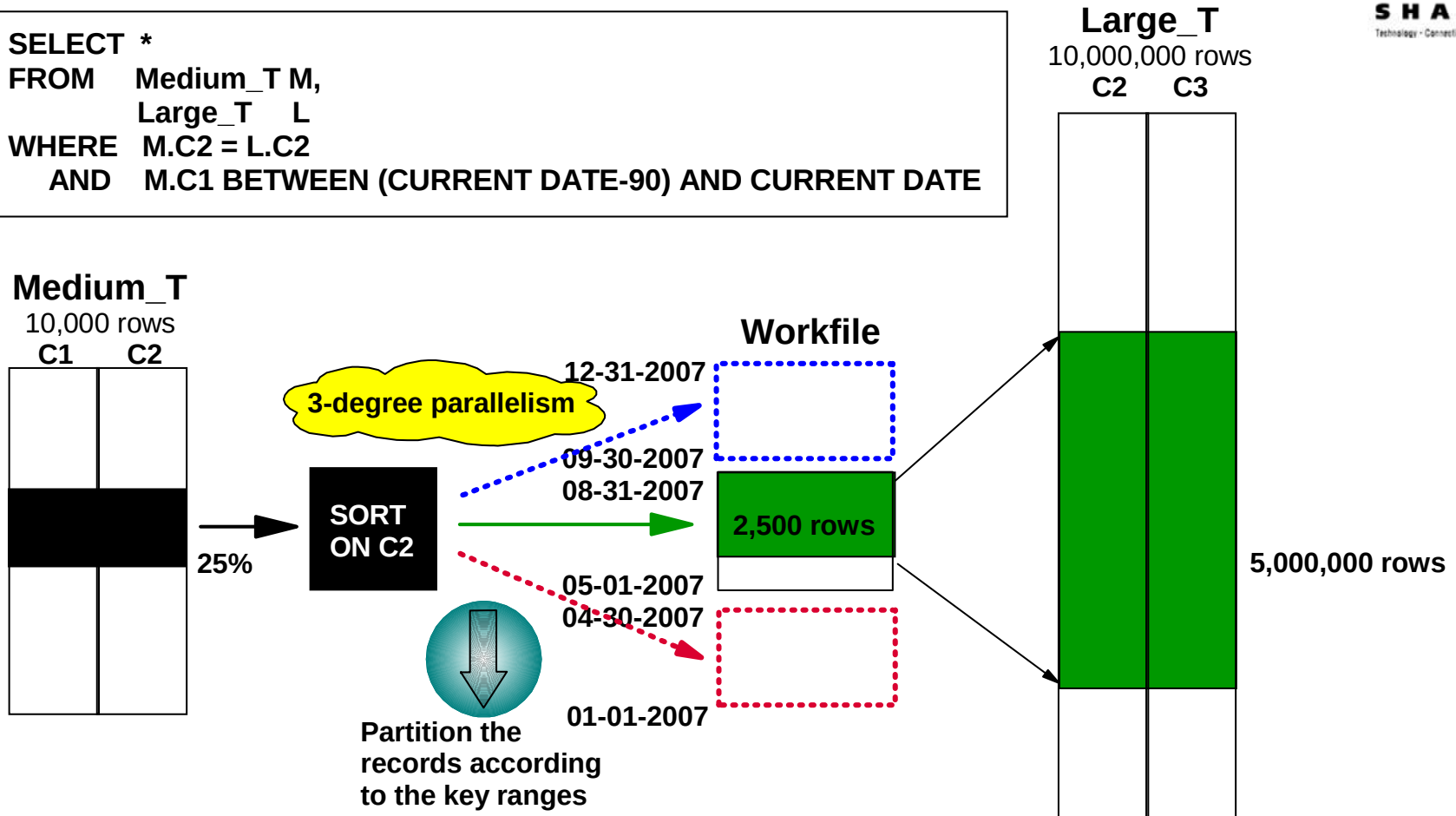
- RLSD ensures that dynamic prefetch continues to prefetch clustered pages as the cluster ratio degrades, hence DB2 10 will not see the degradation in response time seen in DB2 9

Parallelism…

- Support parallelism for multi-row fetch
  - Pre DB2 10
    - Parallelism is disabled for the last parallel group in the top level query block
      - *if there is no more tables to join after the parallel group*
      - *and there is no GROUP BY clause or ORDER BY clause*
  - Example:- SELECT * FROM CUSTOMER
    - There is no parallel group in the query and there are no table joins
    - There is no GROUP BY clause
    - There is no ORDER BY clause
    - So NO PARALLELISM will be used
  - Only effective if CURSOR is DECLARED as READ ONLY
- Allow parallelism if a parallel group contains a work file
  - View or table expression is materialization results in a work file
  - This type of work file is not shared among child tasks prior to DB2 10
    - Hence parallelism is disabled
  - DB2 10 will make the work file shareable
    - only applies to CP mode parallelism and no full outer join case

Parallelism...- Key Range Partitioning Before DB2 10

```
SELECT  *
FROM    Medium_T M,
        Large_T    L
WHERE   M.C2 = L.C2
   AND  M.C1 BETWEEN (CURRENT DATE-90) AND CURRENT DATE
```



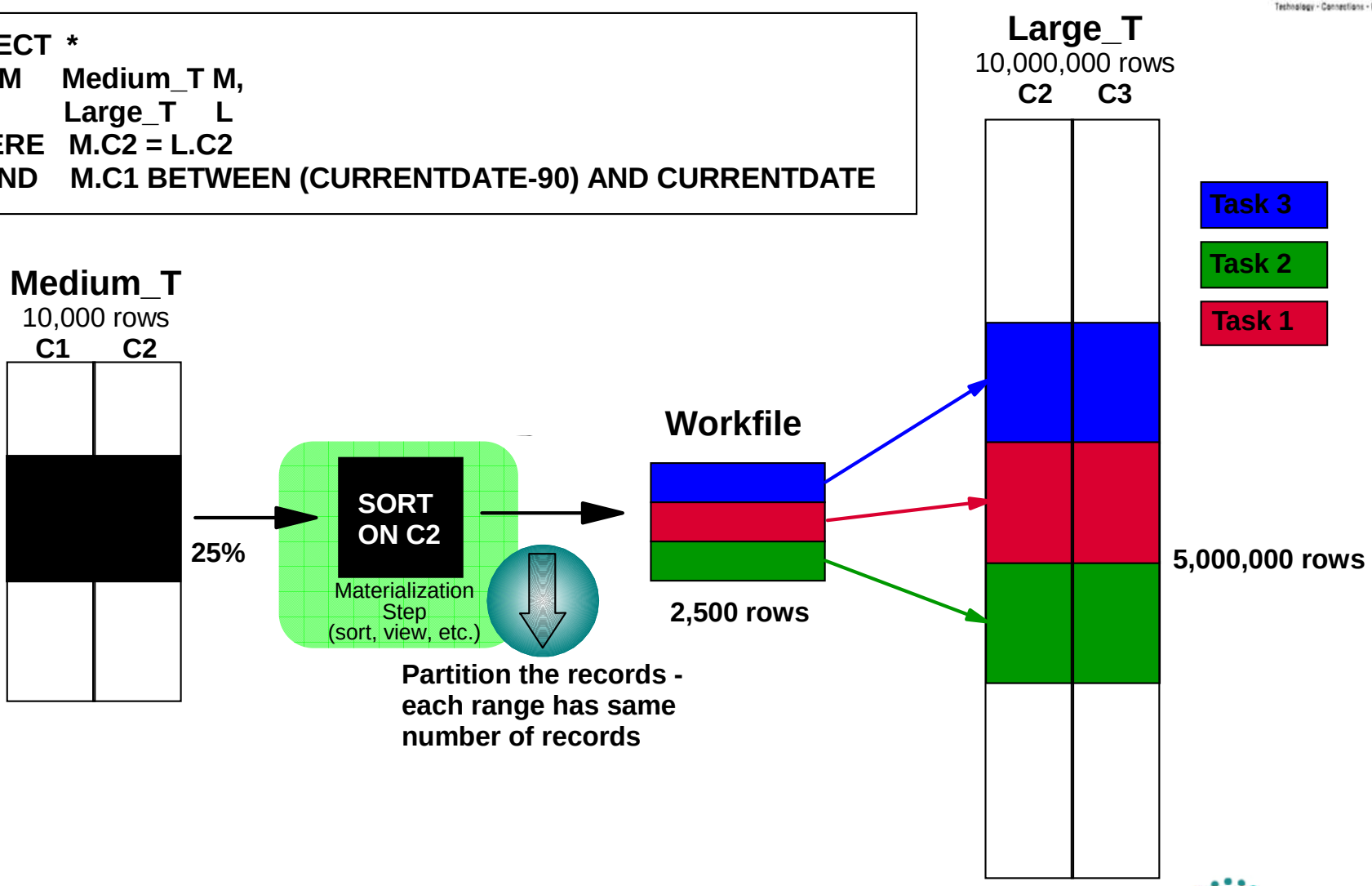
M.C1 is date column, assume current date is 8-31-2007, after the between predicate is applied, only rows with date between 06-03-2007 and 8-31-2007 survived, but optimizer chops up the key ranges within the whole year after the records are sorted :-(

Parallelism…

- Dynamic record range partitioning
  - Intermediate results are divided into ranges
    - Equal number of records
    - Division doesn't have to be on the key boundary
      - *Unless required for group by or distinct function*
  - Record range partitioning is dynamic
    - No longer based on the key ranges decided at bind time
  - Now based on number of
    - Composite side records and
    - Workload elements
  - Not impacted by
    - Data skew,
    - Out of date statistics
  - Will attempt to use in-memory work file for the materialization
  - DSN_PGROUP_TABLE, RANGEKIND = ‘R’

Parallelism...Dynamic record range partition

```
SELECT  *
FROM    Medium_T M,
        Large_T    L
WHERE   M.C2 = L.C2
   AND  M.C1 BETWEEN (CURRENTDATE-90) AND CURRENTDATE
```



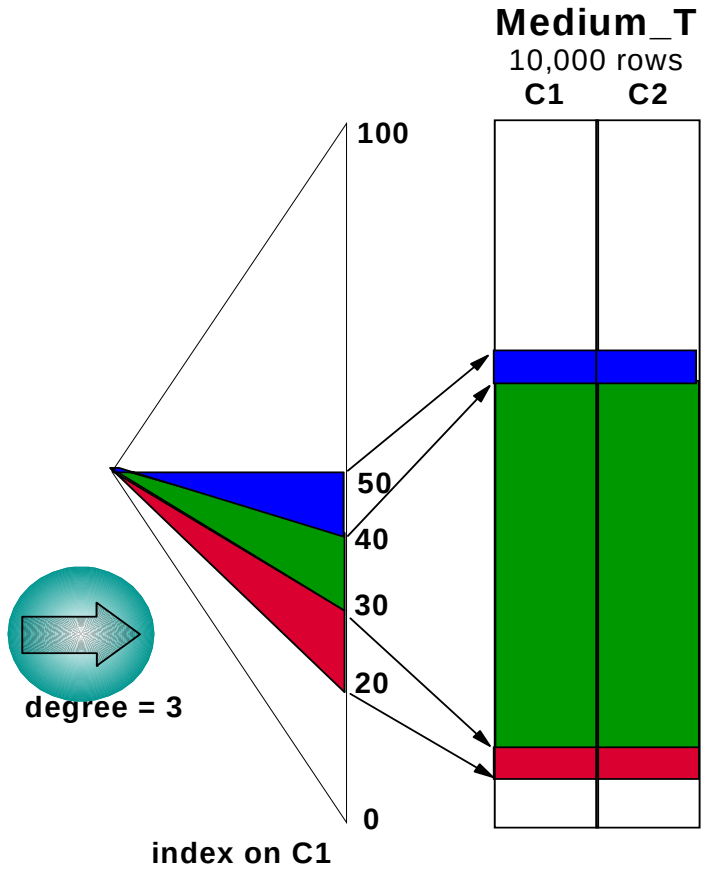
Parallelism…

- Previous releases of DB2 divide the number of keys or pages by the number representing the parallel degree
  - One task is allocated per degree of parallelism
  - The range is processed and the task ends
  - Tasks may take different times to process due to uneven distribution/skew

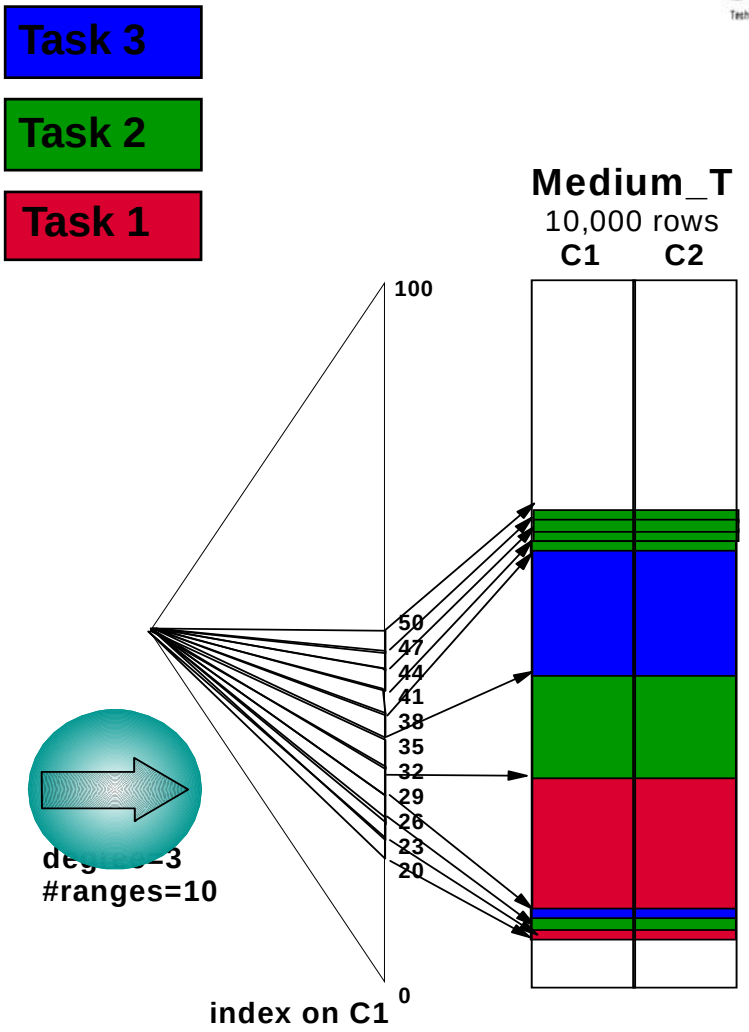
- DB2 10 may use the Straw Model workload distribution method
  - More key or page ranges will be allocated than the number of parallel degrees
  - The same number of tasks as before are allocated (same as degree)
  - Once a task finishes its smaller range it will process another range
  - Skewed data has the opportunity to be divided into a smaller number of pieces

Parallelism... STRAW Model

```
SELECT *
FROM   Medium_T M
WHERE  M.C1 BETWEEN 20 AND 50
```



Divided in key ranges before DB2 10



Divided in key ranges with Straw Model

Sort Enhancements

- FETCH FIRST n ROWS
  - DB2 9 used an in-memory replacement technique
    - When the result is less than 32K
  - DB2 10 extends this to 128K
- Avoid workfile usage for small sorts
  - DB2 9 introduced this for:
    - Final sort when <= 255 rows and < 32K
  - DB2 10 extends this to intermediate sorts

Statistics Enhancements...

- Index Probing & RTS usage (Runtime stats validation)
  - Estimate the number of rids and/or validate number of table rows
    - Within start/stop key range by probing a limited number of index non-leaf pages
    - Validate statistics from RTS
  - Runtime stats validation will be done when both of the following conditions are met.
    - Query has matching index-access local predicate
    - Predicate contain literals, or REOPT(ALWAYS|ONCE|AUTO) is done
  - In addition to the conditions above, at least one of the following must also be satisfied.
    - Predicate is estimated to qualify no rows
    - Stats indicate the table contains no rows
    - Table is defined with VOLATILE or passes NPGTHRSH ZParm threshold check
  - New EXPLAIN table to externalize estimates obtained
    - DSN_COLDIST_TABLE

Statistics Enhancement... AutoStats

- Autostats is implemented though a set of Stored Procedures
  - Stored procedures are provided to enable administration tools and packaged applications to automate statistics collection.
  - **ADMIN_UTL_MONITOR**: Look for statistics needing update
  - **ADMIN_UTL_EXECUTE**: Execute RUNSTATS
  - **ADMIN_UTL_MODIFY**: Clean up the Autostats Repository
  - ADMIN_ADD_TASK: Called by MONITOR/EXECUTE & start up
  - STATS ADVISOR (Data Studio)
- Run automatically honoring specified time windows
- Working together, these procedures
  - Determine what stats to collect
  - Determine when stats need to be collected
  - Schedules and Performs the stats collection
  - Records activity for later review

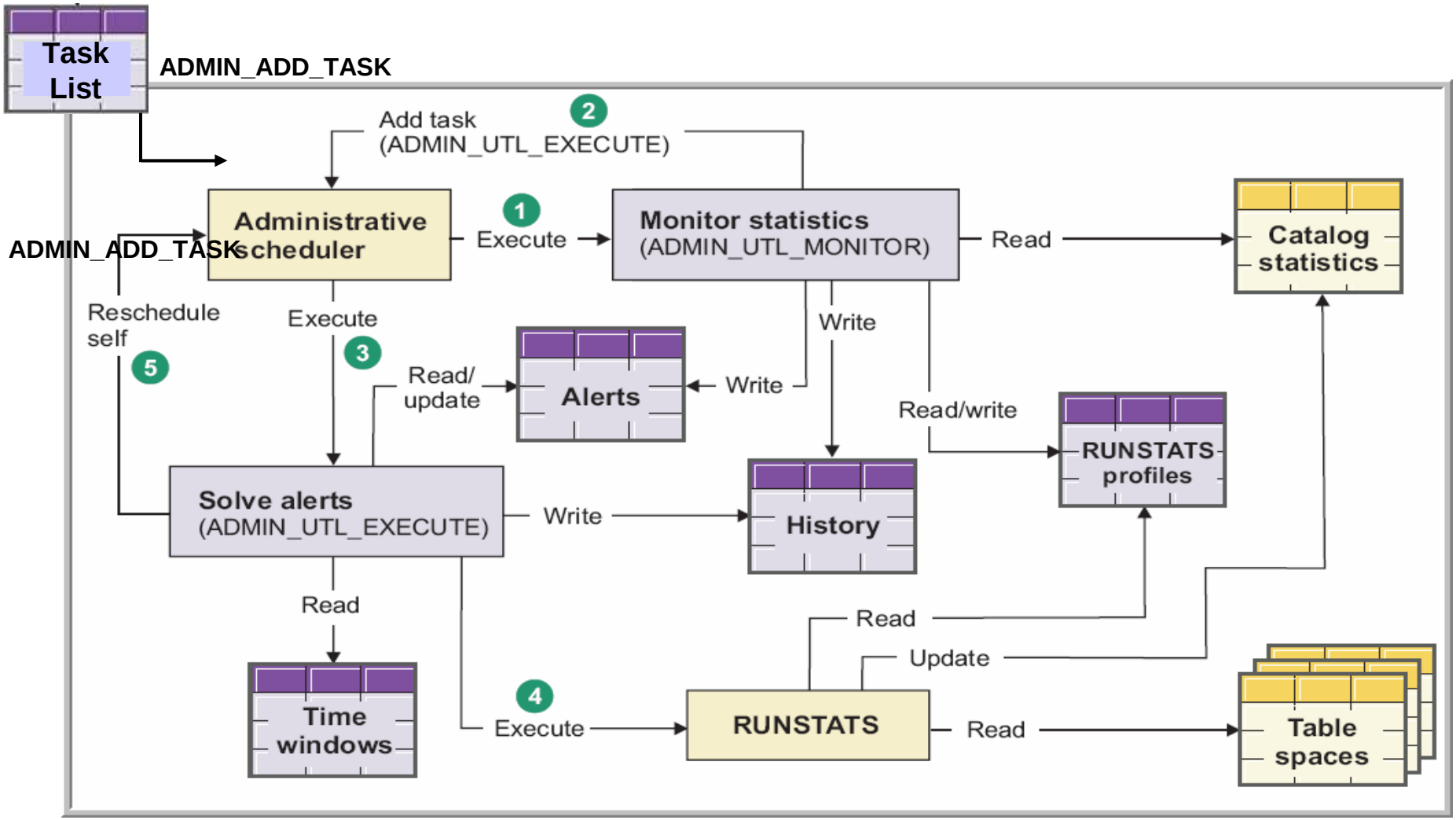
See Chapter 11 "Designing DB2 statistics for performance" in the DB2 10 for z/OS Managing Performance Guide for details on how to configure autonomic monitoring directly within DB2.

Also see the ADMIN_UTL_MONITOR definition for criteria of Missing, Out of Date, or Inconsistent statistics.

Statistics Enhancement... AutoStats

- Configuration / Communication via DB2 catalog tables
  - SYSAUTOTIMEWINDOWS
    - Defines when autonomic procedures can be run
  - SYSAUTORUNS_HIST
    - Keeps history of what procedures have executed autonomically
  - SYSAUTOALERTS
    - Populated when MONITOR procedure detects that an action needs to be scheduled for execution (e.g. RUNSTATS needs to be scheduled)
  - SYSTABLES_PROFILES
    - Contains the RUNSTATS options for a particular table

Statistics Enhancement... AutoStats



Statistics Enhancement... AutoStats

Call SYSPROC.ADMIN_TASK_ADD

- Set execution times, interval, max invocations, what to execute, etc.
- INPUT establishes objects involved & runtime options
- Output shows assigned TASK_NAME / ID

Name	Type	Value
USERID	VARCHAR(128)...	dba015
PASSWORD	VARCHAR(24) F...	[redacted]
BEGIN_TIMESTAMP	TIMESTAMP	*NULL*
END_TIMESTAMP	TIMESTAMP	2010-12-04 21:30:33
MAX_INVOCATIONS	INTEGER	10
INTERVAL	INTEGER	60
POINT_IN_TIME	VARCHAR(400)...	*NULL*
TRIGGER_TASK_NAME	VARCHAR(128)...	*NULL*
TRIGGER_TASK_COND	CHAR(2) FOR S...	*NULL*
TRIGGER_TASK_CODE	INTEGER	*NULL*
DB2_SSID	VARCHAR(4) F...	DSNT
PROCEDURE_SCHEMA	VARCHAR(128)...	SYSPROC
PROCEDURE_NAME	VARCHAR(128)...	ADMIN_UTL_MONITOR
PROCEDURE_INPUT	VARCHAR(4096...	SELECT 'restrict-ts="DBNAME LIKE "DSN8%"'', 0, 0 ," FROM SYSIBM.SYSDUMMY1
JCL_LIBRARY	VARCHAR(44) F...	*NULL*
JCL_MEMBER	VARCHAR(8) F...	*NULL*
JOB_WAIT	VARCHAR(8) F...	*NULL*
TASK_NAME	VARCHAR(128)...	*NULL*
DESCRIPTION	VARCHAR(128)...	*NULL*

Input parameters

JCL_MEMBER	INPUT	VARCHAR	*NULL*	
JOB_WAIT	INPUT	VARCHAR	*NULL*	
TASK_NAME	IN/OUT	VARCHAR	*NULL*	TASK_ID_0001
DESCRIPTION	INPUT	VARCHAR	*NULL*	
RETURN_CODE	OUTPUT	INTEGER		0
MSG	OUTPUT	VARCHAR		THE TASK TASK_ID_0001 WAS ADDED SUCCESSFULLY.

Output parameters

Statistics Enhancement... AutoStats

- To view the tasks for ADMT

```
1  SELECT *
2  FROM TABLE(DSNADM.ADMIN_TASK_LIST()) ATL
3  ;
```

USERID	END_TIMESTAMP	MAX_INVOCATIONS	INTERVAL	DB2_SSID	PROCEDURE_SCHEMA	PROCEDURE_NAME	PROCEDURE_INPUT
DBA015	2010-12-02 20...	10	60	DSNT	SYSPROC	ADMIN_UTL_MONITOR	SELECT 'restrictts="DBNAME LIKE "DSN8%"'', 0, ...
DBA015	NULL	10	NULL	NULL	SYSPROC	ADMIN_UTL_EXECUTE	SELECT 'stand-alone=no',0,0,'' FROM SYSIBM.SY...

- To see the task(s) status
  - TASK_ID_0001 completed
  - It scheduled DB2 AUTO PROCEDURE EXECUTE
    - Which also completed
    - This ran the ADMIN_UTL_EXECUTE SP

```
SELECT
        SUBSTR(TASK_NAME,1,30) AS TASK_NAME
--     ,TASK_NAME
       ,STATUS
       ,NUM_INVOCATIONS
       ,START_TIMESTAMP
       ,END_TIMESTAMP
--       ,JOB_ID
--       ,MAXRC
--       ,COMPLETION_TYPE
--       ,SYSTEM_ABENDCD
--       ,USER_ABENDCD
       ,MSG
       ,SQLCODE
       ,SQLSTATE
       ,SQLERRP
       ,SQLERRMC
       ,DB2_SSID
       ,USERID
FROM TABLE(DSNADM.ADMIN_TASK_STATUS(10)) ATS
ORDER BY START_TIMESTAMP
;
```

TASK_NAME	STATUS	NUM_INVOCATIONS	START_TIMESTAMP	END_TIMESTAMP	MSG
TASK_ID_0001	COMPLETED	1	2010-11-30-20.38.36.000000	2010-11-30-20.38.41.000000	DSNT400I SQLCODE = 000, SUCCESSFUL
DB2 AUTO PROCEDURE EXECUTE	COMPLETED	10	2010-11-30-20.38.41.000000	2010-11-30-20.38.49.000000	DSNT400I SQLCODE = 000, SUCCESSFUL

Statistics Enhancement... AutoStats

- To view execution status

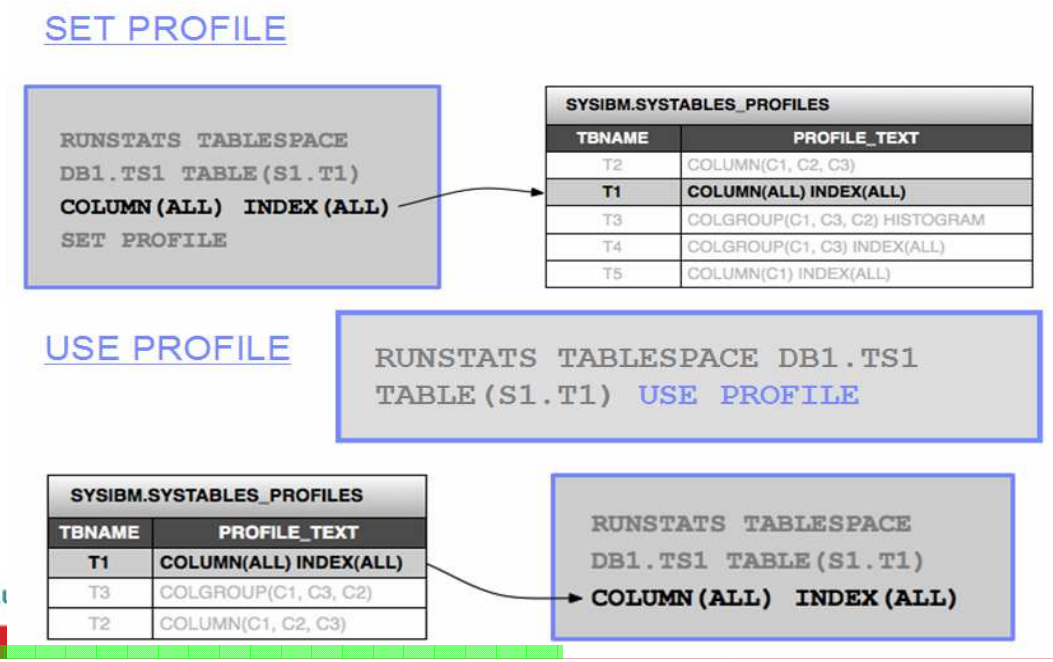
```
SELECT HISTORY_ENTRY_ID
      ,OUTPUT
 FROM SYSIBM.SYSAUTORUNS_HIST
 ;
```

```
2010-11-30 20:38:38.211170> Executing SYSPROC.ADMIN_UTL_MONITOR
2010-11-30 20:38:38.211210> with options: restrictts="DBNAME LIKE 'DSN8%'"
2010-11-30 20:38:38.211218> stored procedure begins at 2010-11-30 20:38:38.139501
2010-11-30 20:38:38.211238> ----------------------->>Used parameters<<--------------------
2010-11-30 20:38:38.211252> STAND-ALONE=NO
2010-11-30 20:38:38.211263> STATISTICS-SCOPE=BASIC
2010-11-30 20:38:38.211278> RESTRICT-TS=
2010-11-30 20:38:38.211288> SAMPLING-RATE=AUTO
2010-11-30 20:38:38.211299> SAMPLING-THRESHOLD=500000
2010-11-30 20:38:38.211308> NUM-CHANGES=100000
2010-11-30 20:38:38.211317> NUM-MASS-DELETES=0
2010-11-30 20:38:38.211332> PCT-CHANGES=20
2010-11-30 20:38:38.211843> COLGROUP-CARD-GREATER-THAN-SUPERSET-COLGROUP-CARD=0.1
2010-11-30 20:38:38.211851> DIFFERENT-COLGROUP-CARD-FROM-COLDIST-AND-INDEX=0.1
2010-11-30 20:38:38.211857> DIFFERENT-COLGROUP-CARD-FROM-INDEXES=0.1
2010-11-30 20:38:38.211863> DIFFERENT-SINGLE-COL-COLGROUP-CARD-FROM-COLDIST-AND-INDEX=0.1
2010-11-30 20:38:38.211869> DIFFERENT-SINGLE-COL-COLGROUP-CARD-FROM-INDEXES=0.1
2010-11-30 20:38:38.211875> DRF-GREATER-THAN-TABCARD=0.1
2010-11-30 20:38:38.211880> DRF-LESS-THAN-NPAGES=0.1
2010-11-30 20:38:38.211886> FREQUENCY-OUT-OF-RANGE=0.1
2010-11-30 20:38:38.211891> INDEX-FULLKEYCARD-LESS-THAN-ANY-KEY-CARD=0.1
2010-11-30 20:38:38.211897> INDEX-FULLKEYCARD-LESS-THAN-FIRSTKEYCARD=0
2010-11-30 20:38:38.211902> MAXIMUM-FREQUENCY-LESS-THAN-RECIPROCAL-OF-COLGROUP-CARD=0.1
2010-11-30 20:38:38.211908> NUMBER-OF-FREQUENCY-RECORDS-GREATER-THAN-COLGROUP-CARD=0.1
2010-11-30 20:38:38.211914> PRODUCT-OF-COLCARD-LESS-THAN-COLGROUP-CARD=0
2010-11-30 20:38:38.211919> QUANTILE-CARD-GREATER-THAN-COLCARD=0.1
2010-11-30 20:38:38.211925> QUANTILE-CARD-GREATER-THAN-COLGROUP-CARD=0.1
2010-11-30 20:38:38.211930> QUANTILE-FREQUENCY-OUT-OF-RANGE=0
```

```
2010-11-30 20:38:41.614274> Executing SYSPROC.ADMIN_UTL_EXECUTE
2010-11-30 20:38:41.614315> with options: stand-alone=no
2010-11-30 20:38:41.614332> stored procedure begins at 2010-11-30 20:38:41.614320
2010-11-30 20:38:41.621030> running in DB2 subsystem DSNT
2010-11-30 20:38:41.621045> ----------------------->>Used parameters<<----------------------
2010-11-30 20:38:41.621472> STAND-ALONE=NO
2010-11-30 20:38:41.641783> ----------------------->>Step 1: detect new alerts<<----------------------
2010-11-30 20:38:41.643971> 523 new alerts with action RUNSTATS detected
2010-11-30 20:38:41.644140> ----------------------->>Step 2: get maintenance windows<<----------------------
2010-11-30 20:38:41.647076> Time window for RUNSTATS alerts: [2010-11-30 20:38:41 -> 2010-12-07 23:59:59]: 1 tasks on DSNT
2010-11-30 20:38:41.647131> ----------------------->>Step 3: calculate alerts execution/duration ratio<<----------------------
```

Statistics Enhancement... Profiles / Performance

- RUNSTATS
  - New options to SET / UPDATE / USE a statistics profile
    - RUNSTATS … TABLE tbl COLUMN(C1)… **SET PROFILE**
      - *Alternatively use* ***SET PROFILE FROM EXISTING STATS***
    - RUNSTATS … TABLE tbl COLUMN(C5)… **UPDATE PROFILE**
    - RUNSTATS … TABLE tbl **USE PROFILE**
  - New option to do page-level sampling
    - RUNSTATS … TABLE tbl **TABLESAMPLE SYSTEM AUTO**



BIND / REBIND Changes

- ACQUIRE(ALLOCATE)
  - This is deprecated.
  - Will result in a warning and be changed to ACQUIRE(USE)
- APRETAINDUP **(REBIND)** — SQL
  - New option to avoid storing package copies identical to previous access path
- CONCURRENTACCESSRESOLUTION — DDF
  - New option
- DBPROTOCOL — SQL
  - Only valid option is DRDA
- DEPLOY **(BIND)** — SEC
  - Now applies to non-inline SQL functions
- EXPLAIN(ONLY)
  - New option to only generate an access path without package creation
- FILTER / QUERYID **(FREE also)**
  - To target a group of queries in the Access Path Repository
- SQLERROR(CHECK) — SEC
  - New option to perform syntax checking without package creation or needing execute authority

OPT

EXPLAIN Changes...

- EXPLAIN tables must be in:
  - V8 or later format (Otherwise -20008, RC 2 is returned)
  - V8 or V9 format results in +20520
  - Unicode encoding scheme
    - Is preferred for V8 & V9 formats
    - Required for V10 format (Otherwise SQLCODE -878 returned)
- EXPLAIN CAPTURE Special Register (defaults to NO)
  - Set to YES
    - Explainable SQL statements run normally and explain information is captured after each statement is prepared and executed
  - Set to EXPLAIN
    - Explainable SQL statements do not execute but explain information is captured during prepare
    - Note that application logic may fail when expecting successful execution
  - Requires a PLAN_TABLE and DSN_STATEMENT_CACHE_TABLE
- The DSNAEXP stored procedure is deprecated
  - This procedure allowed the EXPLAIN of a SQL statement with the authorization for the statement
  - The EXPLAIN privilege or SQLADM should be used

EXPLAIN Changes...

- New PLAN_TABLE columns
  - BIND_EXPLAIN_ONLY
    - Explain information created with BIND....EXPLAIN(ONLY)
  - SECTNOI
    - Section Number of the statement taken from SYSPACKSTMT or SYSSTMT for static statements
    - Differentiates explained statements when QUERYNO = 0 or there are statements with the same QUERYNO
    - If the a static SQL package was bound with the EXPLAIN(YES) option and contains more than one statement with the same value for QUERYNO, use the SECTNOI column in place of QUERYNO
  - EXPLAIN_TIME
    - Timestamp when the explain information was captured
  - MERGC
    - Indicates composite table consolidation before a join
    - Composite rows from the parallel groups are consolidated into a single materialized workfile
    - This happens when the dynamic record range partitioning parallelism enhancement is chosen
  - MERGN
    - Indicates new table consolidation before a join
    - Not currently used. Added for consistency with MERGC for future use
  - Joined to other EXPLAIN tables
    - QUERYNO and BIND_TIME or EXPLAIN_TIME columns

Optimization Evolution – IBM DB2 Tools Support

- Omegamon XE for DB2 PE (OMPE)
  - Literal Replacement (Stmt Cache)
- Bind Manager, Path Checker, SQL PA, QWT
  - Access Path Stability Improvements
  - Optimization Hints
  - Plan Management Policies
- OMPE, Query Monitor, QWT
  - Index Usage Improvements
- Bind Manager, Path Checker, DB2 Admin Tool
  - Other BIND / REBIND Changes
- DB2 Admin Tool, QWT, SQL PA
  - EXPLAIN Changes
  - SQL Pagination (Complex OR)
- OMPE, QM, QWT
  - Parallelism
  - Statistics Enhancements

Advanced Design Options

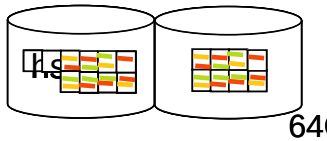
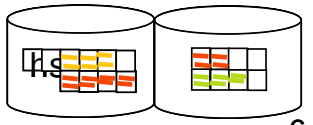
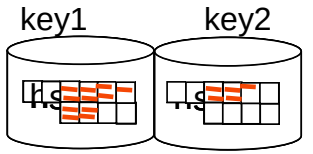
- Table Space history
- Online Schema Evolution
- Compress on Insert
- Implicit Casting
- Hash Access
- Extended Addressability Volumes

Business value of Advanced Design

- Availability
  - Online REORG can move your simple table spaces to new universal table spaces
  - Move a segmented table space at the 64GB limit seamlessly into a partition by growth table space without an outage (aside from the rebinds)
- Productivity
  - Have DB2 materialize online ALTERs at DBA's discretion at a future date
  - No need to CREATE new table spaces, UNLOAD/LOAD, GRANT, DROP, REBIND

Past Table Spaces Options

- **Past table space options**
  - Simple (deprecated v9)
  - Segmented
  - Partitioned
- **Universal TS: partition by growth or by range**

Simple	Multitable		Single deletes	<MC>
Seg	Multitable		Mass deletes	No MC
Part.	Single table		Single deletes	<MC>

*MC = Member Cluster

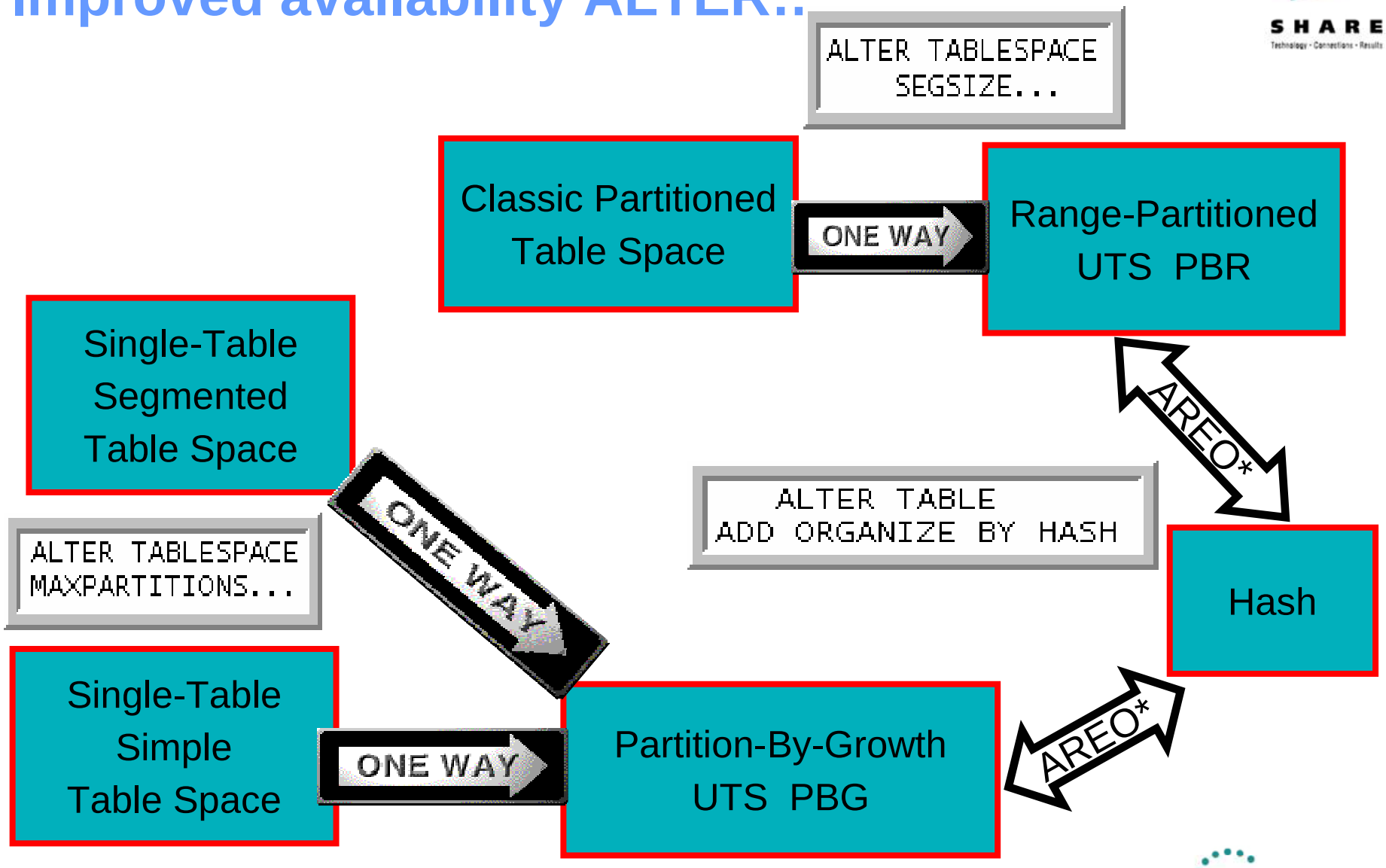
- Remember Simple and Classic Partitioned TS have 2 bits in space map per page
- Segmented and UTS have 4 bits per space map entry

Universal Table Spaces

- What kind of Table Space will be created? (* optional)

CREATE TABLESPACE...	SEGSIZE	NUMPARTS	MAXPARTITIONS	Comments
Segmented	*			*SEGSIZE is optional. Default for explicitly created TS & implicitly created TS for CM8. SEGSIZE defaults to 4.
UTS PBG	*	Optional to indicate # of initial partitions		Default for CM9 and NFM with implicitly created TS. Single table TS. *SEGSIZE will default to 32.
UTS PBR	*			Single table TS *SEGSIZE will default to 32.
Classic Partitioned TS	*			Partitioning TS prior to V9 *DPSEGSZ 0 will create classic partitioned and CM8 behavior is same as V8 NFM

Improved availability ALTER..



Improved ALTER…

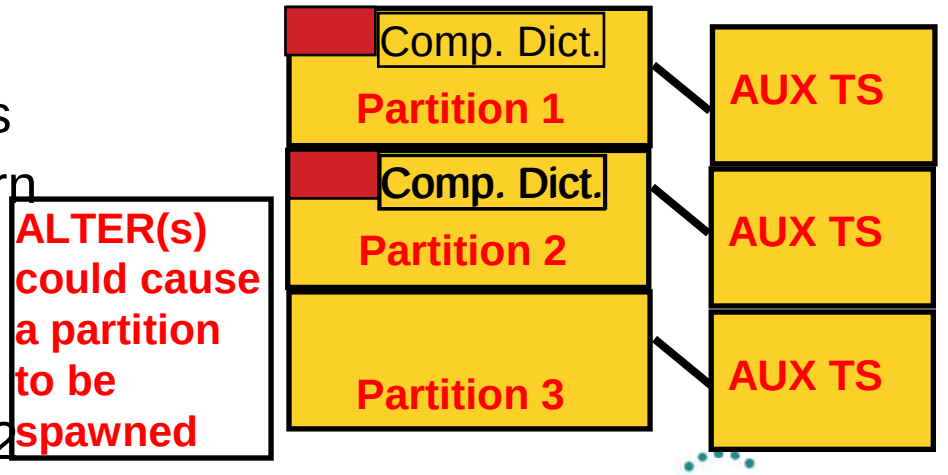
- Pending changes materialized with an online REORG apply to UTS
  - SEGSIZE – no other pending ALTERs can be done before this is materialized
  - DSSIZE – no other pending ALTERs can be done before this is materialized (IMPDSSIZE ZParm for default – 4GB)
    - PM43175 adds DSSIZE 128 and 256
      - *Proportionally decreases the number of partitions*
  - MEMBER CLUSTER – new for UTS
  - MAXPARTITIONS
    - If other pending changes are involved, or changing table space type: it is pending
    - Otherwise it is immediate
  - Page Size (BUFFERPOOL)
    - Can be done with REORG TABLESPACE (for indexes and tables) or REORG INDEX for only index changes
- Other ALTERs are immediate
  - The above statements if TS or IX not defined
  - MAXPARTITIONS (unless changing TS)
    - With PM57001 MAXPARTITIONS can be altered lower if removed part(s) are not allocated
  - BUFFERPOOL PGSTEAL NONE
  - LOB INLINE LENGTH LOB

Improved ALTER…

- Altering SEGSIZE
  - Smaller could cause table space to spawn parts
  - SEGSIZE: 32 is default for UTS
    - Too small lots of space map pages
    - *DB2 for z/OS Performance Monitoring and Tuning Guide*
- Altering buffer pool (pagesize)
  - Likely WILL cause shrinkage or growth in number of populated parts
  - Empty parts are not reclaimed, return to primary quantity
- Implicitly created TS
  - PBG DSSIZE 4GB, MAXPARTITONS 256, SEGSIZE 32

```
CREATE TABLESPACE  emp
DSSIZE 2G
MAXPARTITIONS 3
NUMPARTS 2
LOCKSIZE ANY
SEGSIZE 32;
```

Universal Table space – Partition By Growth



Online Schema – Details on Execute ALTER Statement

- Statement is validated
  - Semantic checking against effective catalog definition
- Assuming all checks out ok:
  - Statement is put on pending list
  - Table space is placed in Advisory-REORG pending :**AREOR** (non-restrictive)
    - Not to be confused with REORG-pending advisory **(AREO*)** which says performance could be impacted
  - Statement completes with SQLCODE +610 to advertise the advisory state
- Drop changes
  - ALTER TABLESPACE… DROP PENDING CHANGES
    - still in AREOR
    - all changes for that table space will be dropped

SYSIBM.SYSPENDINGDDL:

DBNAME	TSNAME	DBID	PSID	OBJSCHEMA	OBJNAME	…	OPTION_ KEYWORD	OPTION_ VALUE	…	STATE MENT_ TEXT

Online Schema – Details on Online REORG

- Pending DDL is materialized - **DSNU1163I**
  - Catalog & Directory are updated with the new attributes
  - Data sets are updated with the new attributes
  - Materialized SYSPENDINGDDL entries are removed
- Stats are collected
  - Default is TABLE ALL INDEX ALL UPDATE ALL HISTORY ALL unless overridden

UTL

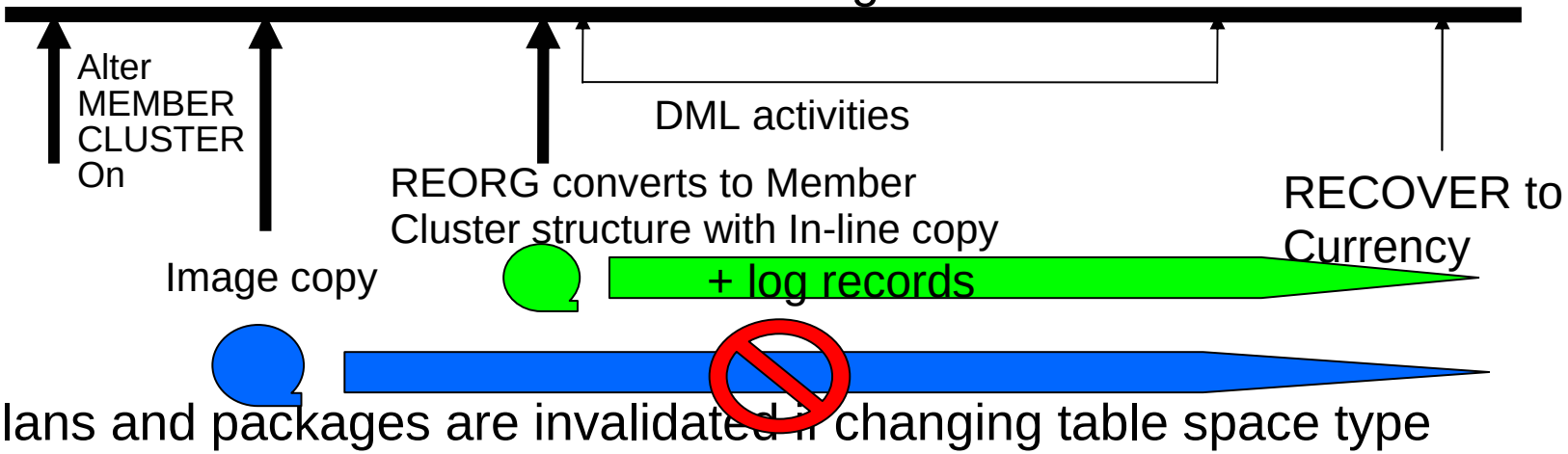
  - Warning message is issued to indicate that some partition statistics may no longer be accurate - DSNU1166I
    - (COLGROUP, KEYCARD, HISTOGRAM …)
- SYSCOPY entries show inability to recover object prior to changes
- AREOR state is reset

Online Schema – SQL Restrictions - NFM

- Pending ALTERs other than changing table space type are supported only for UTS
- ALTERing table space type only for single-table table spaces
- Not permitted to mix immediate and pending options in an ALTER statement (SQLCODE -20385)
- Many immediate DDL statements are not allowed while there is pending DDL awaiting materialization (-20385)
  - CREATE/DROP/ALTER
    - E.g. alter of FREEPAGE for a partition
- Pending DDL only materialized by REORG SHRLEVEL REFERENCE or CHANGE
  - REORG SHRLEVEL(NONE) and part-level REORGs are not blocked, but do not materialize pending DDL
- If MAXPARTITIONS is exceeded during REORG **DSNU1170I will be issued**

Online Schema…

- Restrict RECOVER across materializing REORGs

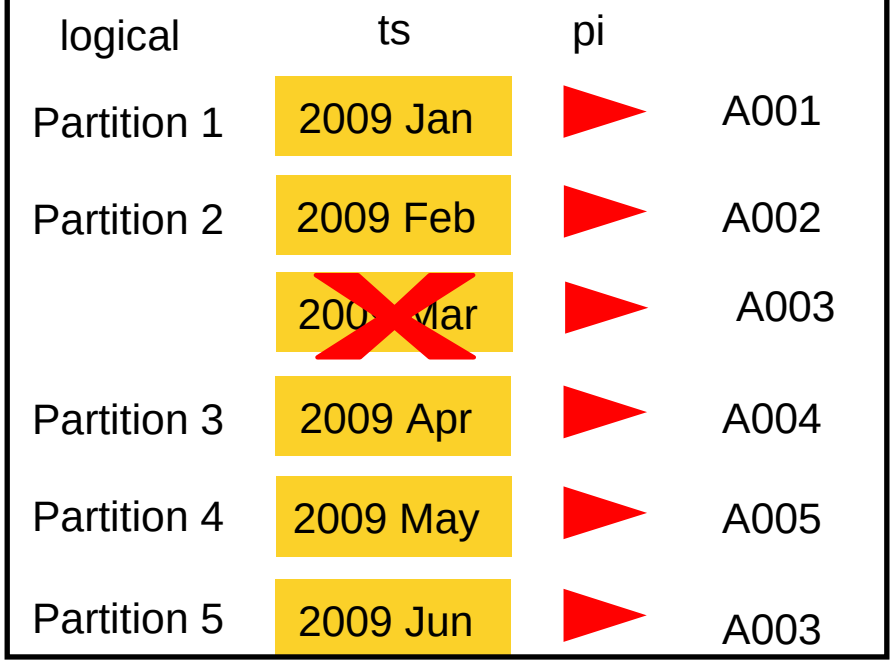


- Plans and packages are invalidated if changing table space type
  - When changing the MAXPARTITIONS attribute of a simple or segmented table space to convert it to a partition-by-growth universal table space
  - The SEGSIZE attribute of a partitioned table space is changed to convert the table space to a range-partitioned universal table space

ALTER TABLESPACE enhancements

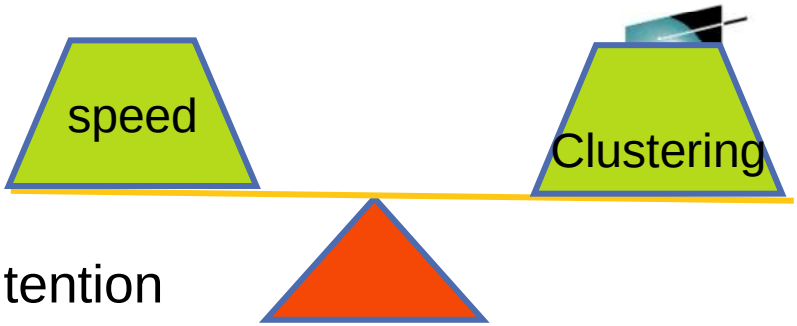
- You can now ROTATE any partition to last
  - LOAD REPLACE first, suggested
  - RESET, no delete triggers activated
  - Not for MQT, PBG, XML
- Also ALTER TABLE ADD PARTITION for partition by growth TS
  - Inherits attributes of previous parts
  - Cannot exceed MAXPARTITIONS (-4701 SQLCODE)
    - MAXPARTS not pending change unless other pending changes exist
  - Helpful for DSN1COPY
    - When moving data to another DB2
    - Synchronize the # of partitions

```
ALTER TABLE TAB1 ROTATE PARTITION 3
TO LAST ENDING AT (2009 JUL') RESET;
```



- **PM43597; ALTER TABLESPACE MAXROWS now sets AREO* instead of AREOR**
  - Can be reset by REORG SHARELEVEL NONE

Member Cluster



- MEMBER CLUSTER added to UTS
  - Avoid hotspots, P-lock, and page latch contention
    - Usually only applicable for data sharing, concurrent access
    - Each member is assigned a set of space map pages
    - Ignores clustering index on insert, REORG to get clustering back
    - Space map covered 199 data pages in DB2 9
    - Space map covers 10 segments in DB2 10
  - New column SYSTABLESPACE.MEMBER_CLUSTER
  - Very important for APPEND tables (DB2 9)

```
ALTER TABLESPACE emp_space
MEMBER CLUSTER (YES);
```

Compress on INSERT

- Data compression occurs when a dictionary exists
- Prior to DB2 10
  - Dictionary not built on a table space with COMPRESS YES attribute until:
    - REORG or
    - LOAD utility was executed
  - For some customers, REORG or LOAD are not executed frequently
  - LOAD COPYDICTIONARY offered in DB2 9
- DB2 10 NFM allows for build of compression dictionary on:
  - INSERT
  - MERGE
  - LOAD utility with REPLACE, RESUME NO, or RESUME YES SHRLEVEL CHANGE, and without KEEPDICTIONARY
- Eliminate need for REORG or LOAD needed to build compression dictionary

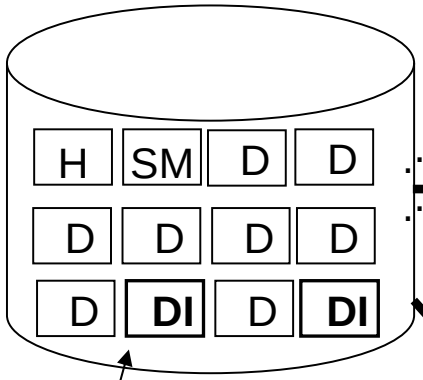
How Compress on INSERT works

- INSERT, MERGE and LOAD trigger the creation of a compression directory if:
  - The table space or partition is defined with COMPRESS YES
  - The table space or partition has no compression dictionary built
  - Inserted data reaches a threshold that allows the build of the compression dictionary
- If threshold is reached, dictionary build is done asynchronously
  - Data continues to be inserted uncompressed until dictionary is ready
  - Rows read with UR in order to build the dictionary
  - DSNU235I – dictionary not built
- Default behavior Compress on insert
  - Use COMPRESS NO if unwanted

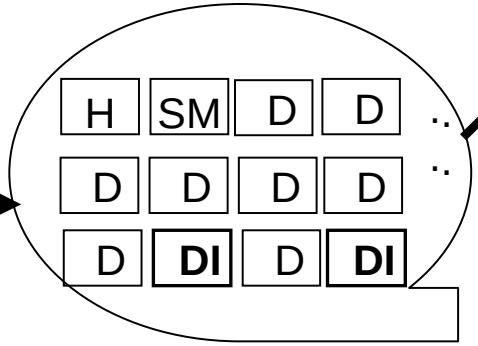
```
DSNU241I @ DSNUZLCR - DICTIONARY WITH
4096 ENTRIES  HAS BEEN SUCCESSFULLY
BUILT FROM 824 ROWS FOR
TABLE SPACE LI864DB.LI864TS, PARTITION 1
```

Location of compression dictionary pages

- If the dictionary pages are scattered throughout table a REORG or COPY SYSTEMPAGES(YES) is needed to UNLOAD rows



COPY TS...
SYSTEMPAGES NO

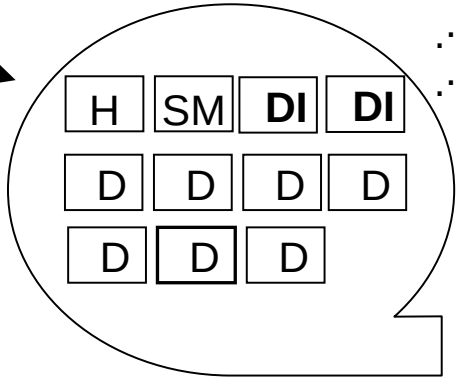


DSNU1232I - COMPRESSED ROW IS IGNORED BECAUSE THE DICTIONARY IS NOT AVAILABLE FOR TABLE table-name

UNLOAD DATA.... FROMCOPY

Dictionary pages not necessarily after space map page! -Could use inline COPY during REORG To ensure dictionary pages up front

COPY TS...
SYSTEMPAGES YES
(default option makes DB2 copy the dictionary pages after space map page)



Implicit Casting of string and numeric data

- V8 could compare CHAR to VARCHAR; SMALLINT to INT
- CHAR FOR BIT DATA -> Binary explicitly in DB2 9
- What about archive table with different data types
  - *Select from MY.emp into ARCH.emp gets SQLCODE -401*
- DB2 10 implicitly casts character or graphic string to DECIMALFLOAT, and numeric back to VARCHAR
- Allows for index matching

```
CREATE TABLE emp
 (empno INTEGER,
   salary DECIMAL(15,2));
```

```
CREATE TABLE emp
 (empno varchar(10),
   salary DECIMAL(15,2));
```

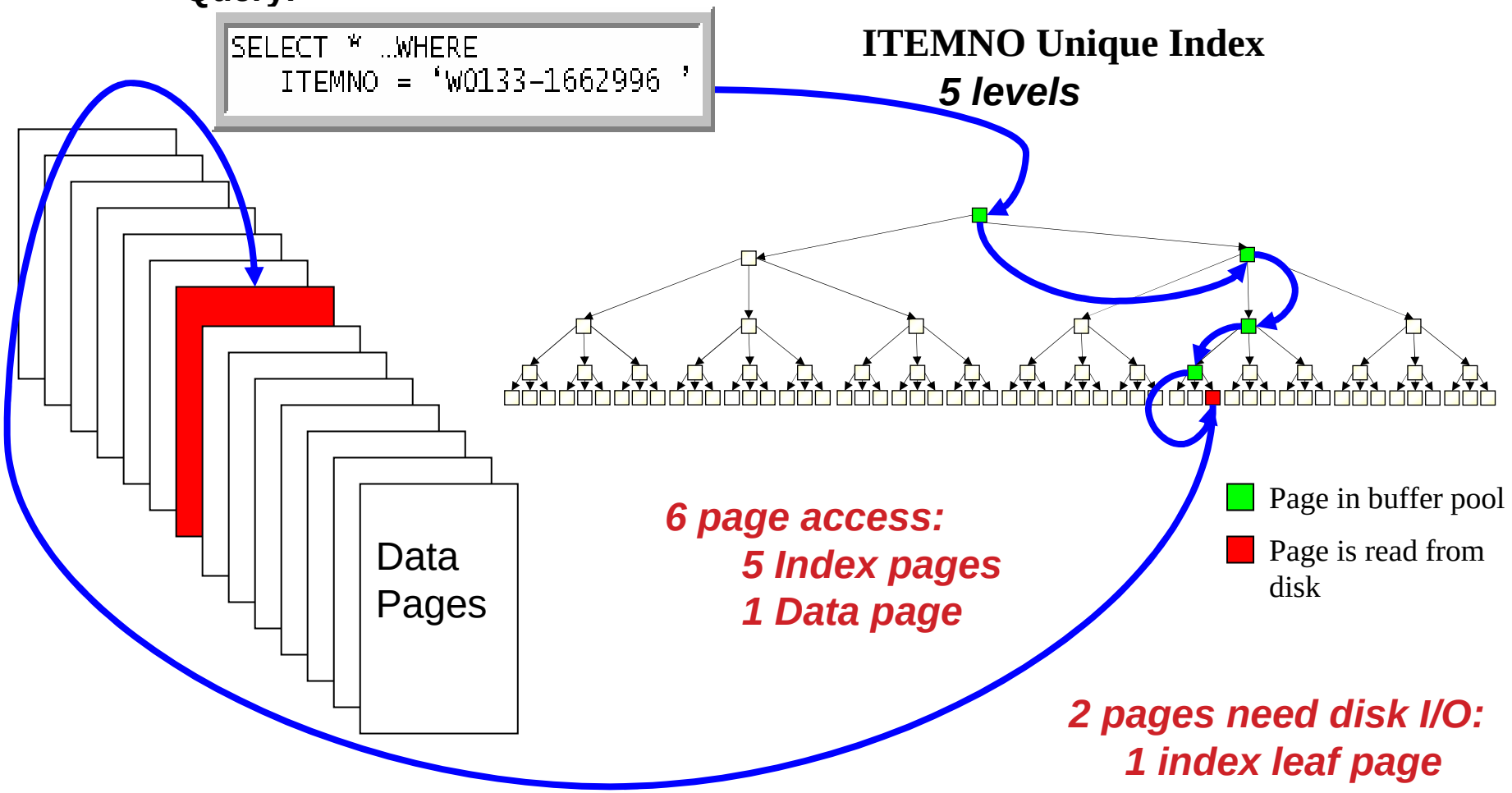
Implicit Casting

Source Data Type	Target Data Type
SMALLINT	VARCHAR(6)
INTEGER	VARCHAR(11)
BIGINT	VARCHAR(20)
NUMERIC/DECIMAL	VARCHAR(precision+2)
REAL	VARCHAR(24)
FLOAT	VARCHAR(24)
DOUBLE	VARCHAR(24)
DECFLOAT	VARCHAR(42)
CHAR	DECFLOAT(34)
VARCHAR	DECFLOAT(34)
GRAPHIC	DECFLOAT(34)
VARGRAPHIC	DECFLOAT(34)

Fastest Available in DB2 V9: Index access

Query:

```
SELECT * …WHERE
   ITEMNO = 'W0133-1662996 '
```

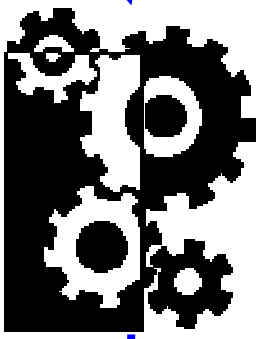


Fastest Available In DB2 V10: Hash Access

Query:

```
SELECT * …WHERE
   ITEMNO = 'W0133-1662996 '
```

Unique Key Value



1 data page access

- **Key finds the row without index**
- **Reduced:**
  - **Page visits**
  - **CPU time**
  - **Elapsed time**

Page in buffer pool

Page is read from disk

1 data page disk I/O
(Possibly in buffer pool)

Rid

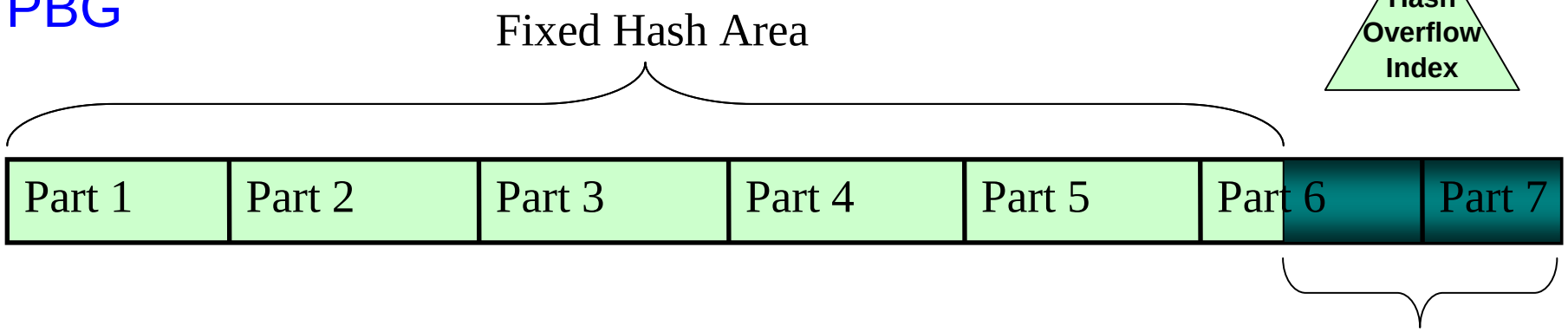
-Trade-off: extra space used

Hash Access Candidates

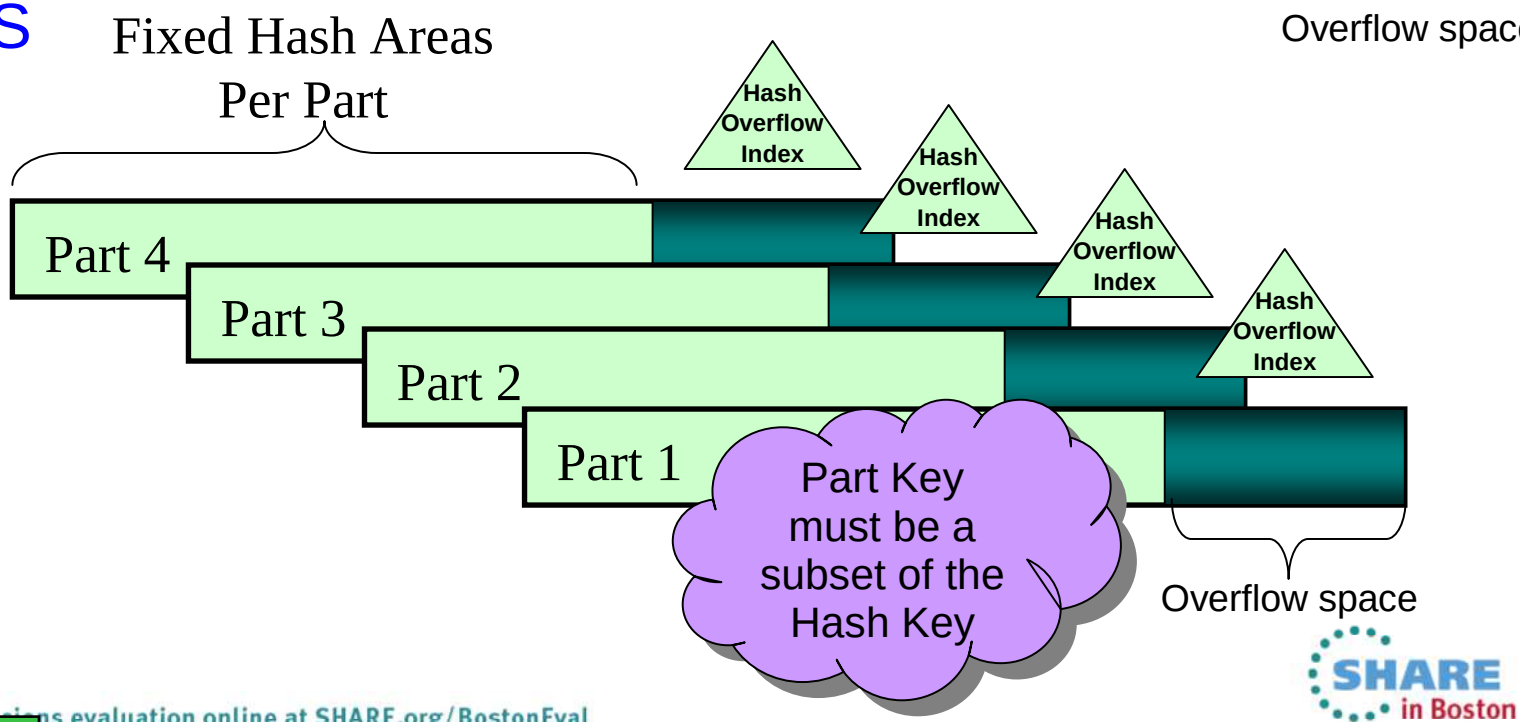
- **Candidate Tables**
  - For queries needing single row access via the unique key
  - Queries having equal predicates on keys
  - With known and static approximate size of data
  - Having large N-level indexes
- **Not for Tables**
  - Needing sequential processing
  - Data size changes frequently
  - Either using BETWEEN or > and <
- **Follow-up**
  - Run REBIND with EXPLAIN option and query the PLAN_TABLE to check access path
  - SYSTABLESPACESTATS.REORGHASHACCESS
    - *Number of times data is read using hash access in the last time interval*
  - Check LASTUSED & REORGINDEXACCESS on overflow and other indexes to validate HASH access
  - PM25652 adds REORG recommendations to DSNACCOX

Hash Organization With Respect to Partitioning

PBG



PBR-UTS



Create Objects with Hash Organization

```
CREATE TABLE…
    …
    ORGANIZE BY HASH UNIQUE
           (LASTNAME, FIRSTNAME)
           HASH SPACE 4G
```

1 PBG

```
CREATE TABLE…
    …
    PARTITION BY RANGE…
          PARTITION 1  …
          ….
          PARTITION 5 …
    …
    ORGANIZE BY HASH UNIQUE
           (LASTNAME, FIRSTNAME)
           HASH SPACE 2G
```

2 PBR

3

***TABLESPACE** options:*

-Table space MUST be UTS PBG or PBR

-Can be created explicitly

-or DB2 creates it implicitly – 64MB for HASH space

4

Overflow INDEX

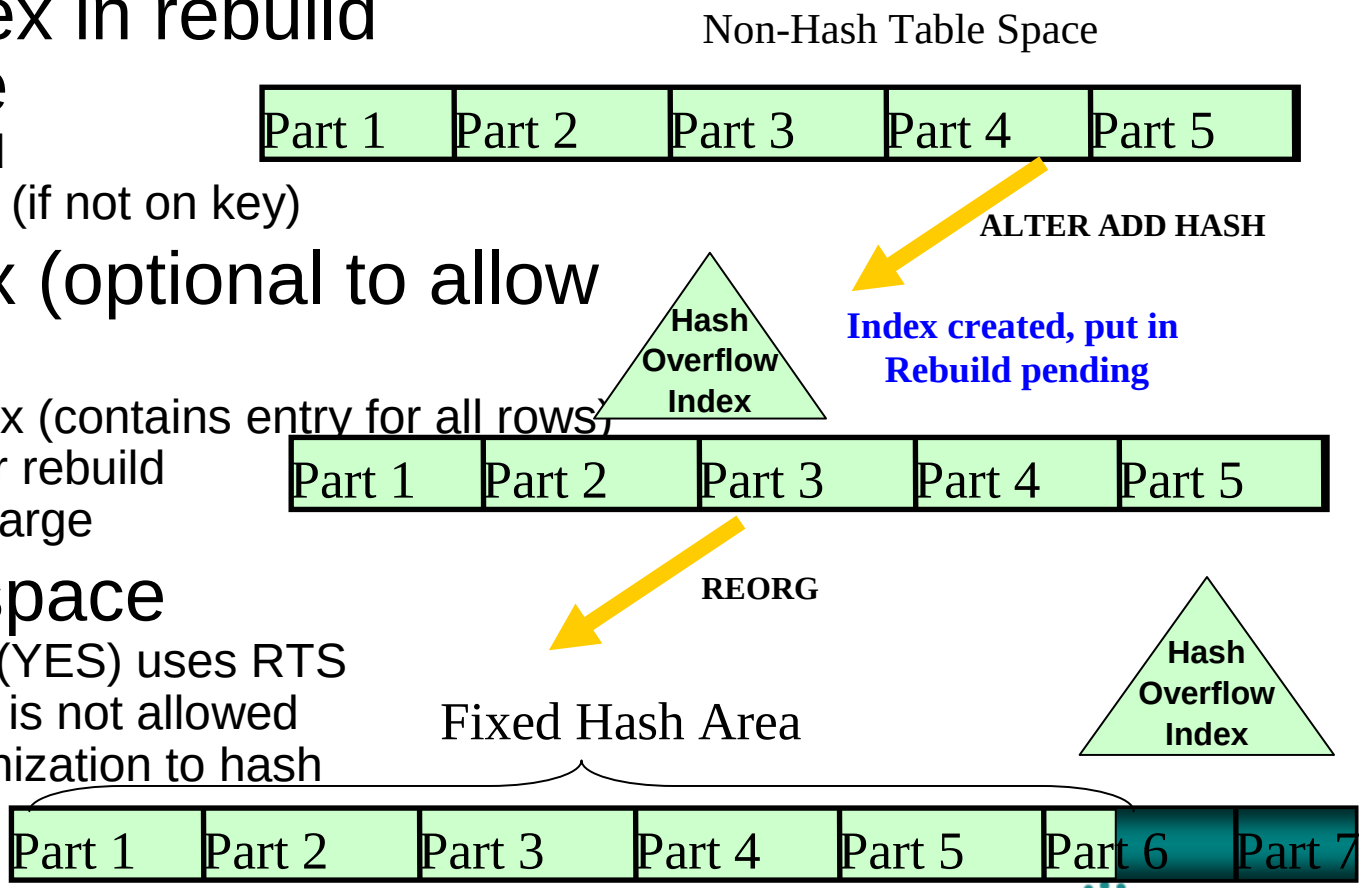
-DB2 creates it implicitly

New variation of index object created to store hash column information.
The index will only contain entries to rows in the overflow area.

1. SYSTABLESPACESTATS.**TOTALROWS:**
   Actual number of rows in the table
2. SYSTABLESPACESTATS.**DATASIZE:**
   Total number of bytes used for rows
3. SYSINDEXSPACESTATS.**TOTALENTRIES:**
   Number of overflow records with keys in the overflow index

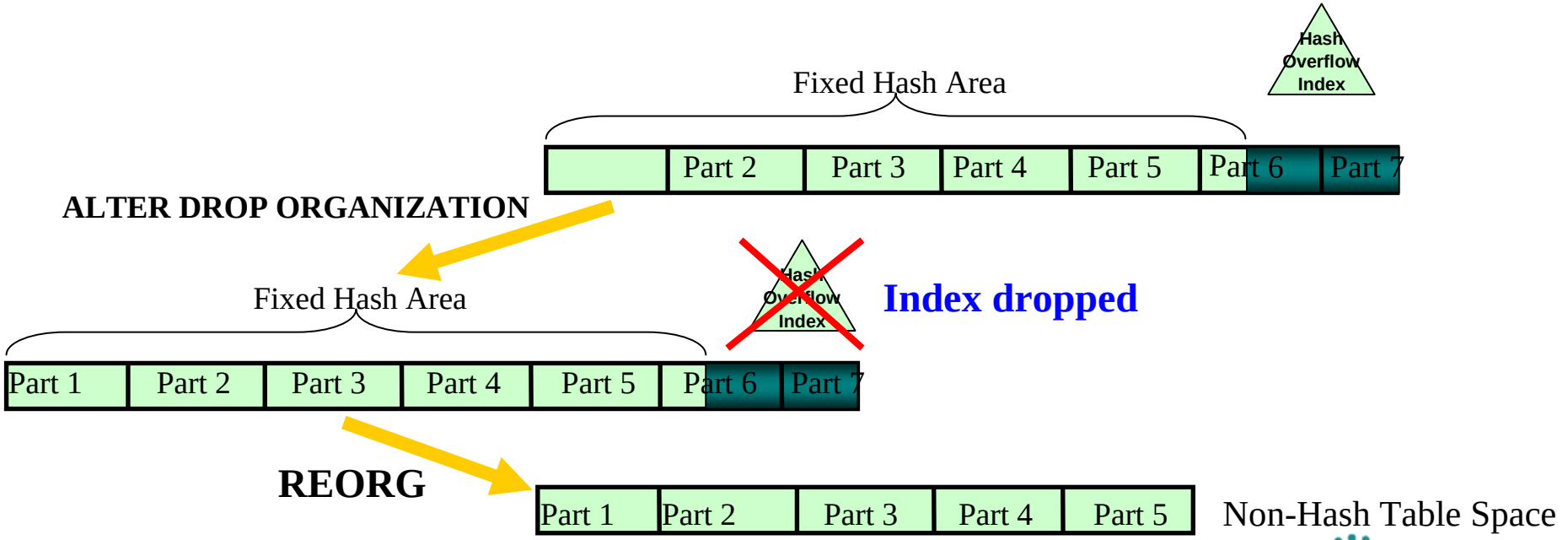
ALTER TABLE ADD ORGANIZE BY HASH

- ALTER is IMMEDIATE to enforce uniqueness
  - Table in advisory reorg state
  - Overflow index in rebuild pending state
    - Inserts not allowed
    - Can delete/update (if not on key)
  - Rebuild Index (optional to allow inserts)
    - Builds a large index (contains entry for all rows)
    - Allows inserts after rebuild
    - Index will be very large
  - Reorg table space
    - AUTOESTSPACE(YES) uses RTS
    - Part level REORG is not allowed
    - Changes the organization to hash
    - Index shrinks



ALTER TABLE DROP ORGANIZATION

- Removes Hash organization
  - Table is placed in REORP
  - Table and related indexes become inaccessible until after REORG
  - Implicitly created hash overflow index dropped during ALTER
  - Table space MUST be Reorged immediately
  - Consider which index will control clustering after this
  - Part level REORG is not allowed after DROP ORGANIZATION



Advanced Design Options – IBM DB2 Tools Support

- DB2 Administration Tool and Object Comparison Tool
  - Improved Alter
  - Universal Tablespaces / Tablespace Enhancements
  - Create with HASH support
- DB2 Table Editor
  - Hash Access
- DB2 Utilities Suite
  - Online Reorg
  - Online Check

Indexing Enhancements

- I/O Parallelism
- Include Columns
- RID Lists
- Sequential Inserts

Business value of index enhancements

- Application performance and CPU savings
  - Sequential insert operations see a vast improvement
  - Lessen frequency of REORG utility while maintaining access path consistency
  - Less waiting on I/O operations for applications

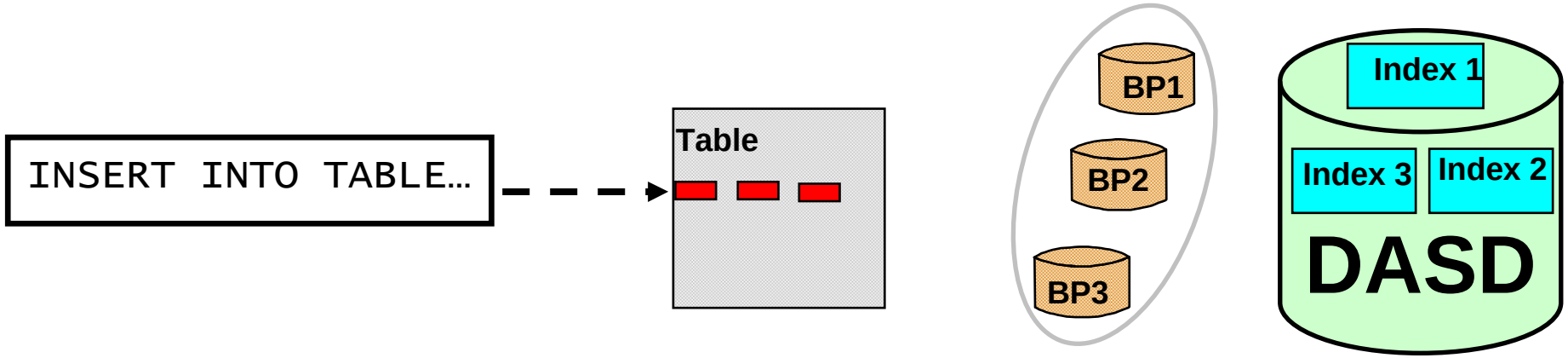
I/O Parallelism for Index Inserts...

- Transactions that perform inserts into tables with many indexes defined on the table previously may have had high response times due to index I/O wait time.
  - DB2 executes index updates sequentially
- I/O parallelism : overlap the I/Os against non-clustering indexes
  - Utilized if there are more than 3 indexes defined on the table and one of them is a clustering index, or 2 indexes if neither is a clustering index
  - DB2 can prefetch pages from different indexes defined on the same table into buffer pool in parallel for insert operation.
- New ZParm INDEX_IO_PARALLELISM with default YES
- This functionality is enabled for DB2 10 Conversion mode

I/O Parallelism for Index Inserts

Preliminary Performance Data

- For a table with 6 indexes and 2000 inserts, elapsed time improvements of up to 50% are measured
- Buffer pool hit ratio affects the measured elapsed time improvement.
  - No elapsed time improvement if all indexes are in the buffer pool
  - Still single threaded update
- IFCID 357 (start) and 358 (end) of I/O parallelism
- Applies to UTS and classic partitioned TS



Additional Non-key Columns In An Index

- Indexes are used to enforce uniqueness constraints on tables
- To achieve index only access on columns not part of the unique constraint, additional indexes are often created for the non-unique columns
  - Slower DB2 transaction time
  - Increased storage requirements
- In DB2 10 Additional Non-key Columns can be defined in a unique index to reduce total amount of needed indexes
- Indexes that participate in referential integrity (RI) will support additional columns, but INCLUDE(d) columns will not be used to enforce RI
- Improves:
  - insert performance as less indexes need to be updated
  - space usage
  - Can stabilize access path as optimizer has fewer similar indexes to choose from

Additional Non-key Columns In An Index ...

- V9 definition

```
CREATE UNIQUE INDEX i1 ON t1(c1,c2,c3)
CREATE INDEX i2 ON t1(c1,c2,c3,c4,c5)
```

- Possible V10 definition

or

```
CREATE UNIQUE INDEX ON t1(c1,c2,c3)  INCLUDE (c4,c5)
```

```
ALTER INDEX i1 ADD INCLUDE (c4)
```

marked RBDP

```
ALTER INDEX i1 ADD INCLUDE (c5)  and DROP INDEX i2
```

- The following restrictions will apply:
  - Index is placed in Rebuild Pending
  - INCLUDE columns are not allowed in non-unique indexes
  - Indexes on Expression will not support INCLUDE columns

More Index Enhancements

- RID list overflows to workfile instead of relational scan (MAXRBLK was 8MB, now 400MB)
  - Eliminate RID list failures from all four causes
  - DB2 9 had it for pair-wise join
  - MAXTEMPS_RID new ZParm
- Referential integrity check performance
  - Sequential detection and index look aside for RI
    - Avoid RI check for each insert of a child under the same parent

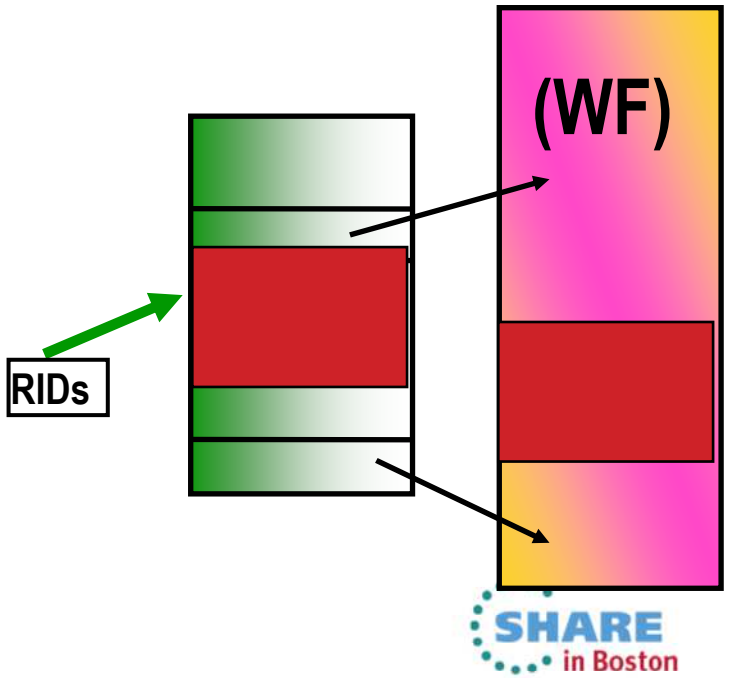
RID limit exceeded:

-RDS limit

-DM limit

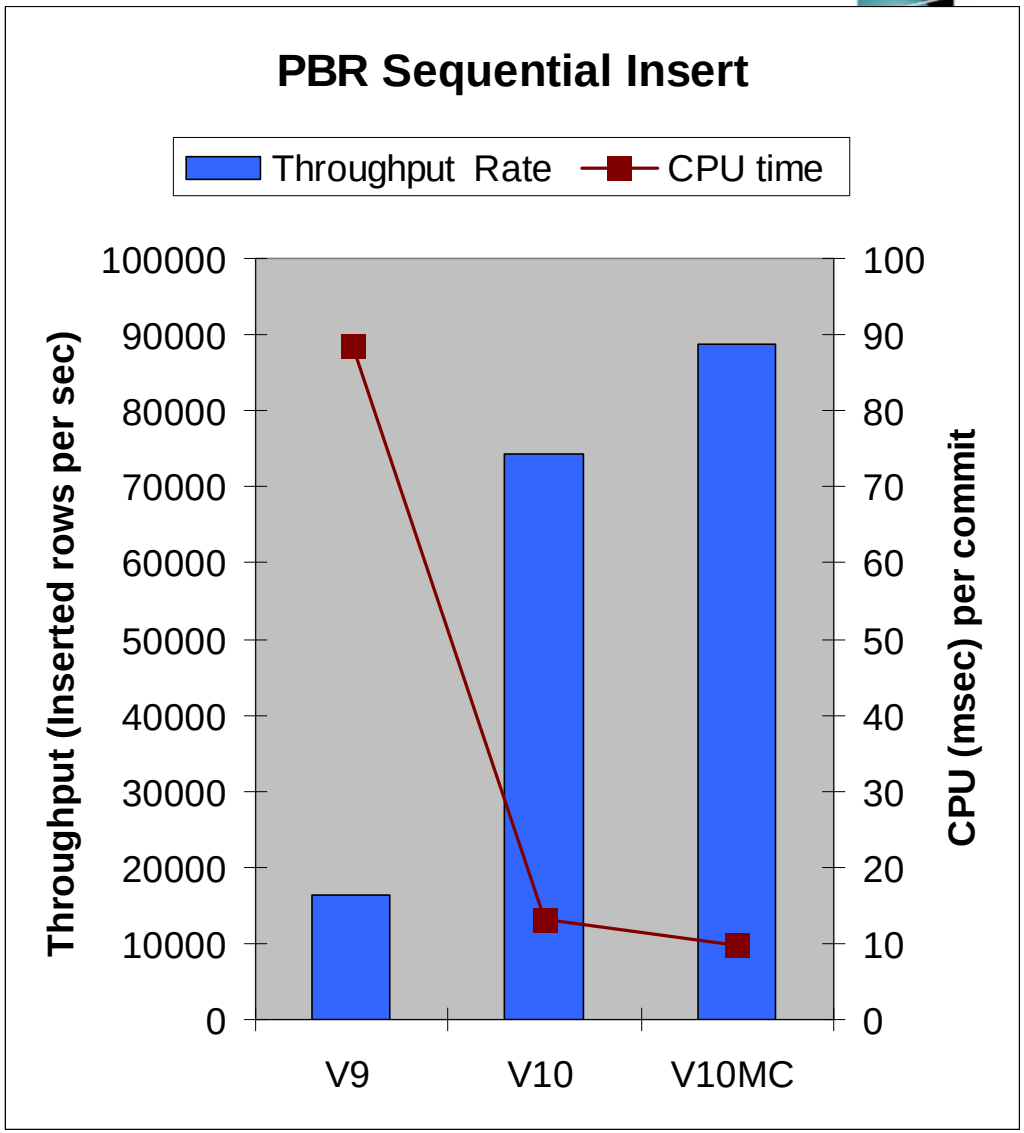
-Exceed proc limit

-No storage



Sequential Inserts

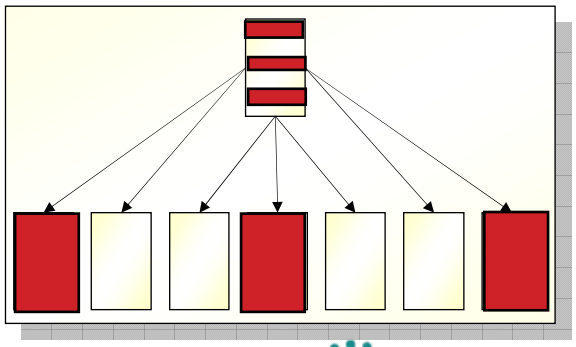
- Performance improvement for sequential inserts into the middle of a cluster index
  - Optimize when index manager picks the candidate RID during sequential insert
    - Significant space search improvement in sequential insert
    - Reduced latch class 006, and 245
    - Higher chance to find open space and avoid a search
- **Test case: Sequential key insert into 3 partitioned TS from 240 clients in 2way data sharing. Multi row insert.**



Member Cluster is only shown as a comparison point

Cost reduction in Index Evaluation

- Optimization in index predicate evaluation process
  - Applicable in any workload but query with many predicate shows higher improvement
  - Preliminary measurements shows average 18% CPU reduction (1% thru 70%) from TPC-H like workload using 100 queries
- Reduced need for REORG
  - Index scan using disorganized index
  - Index manager switches from dynamic prefetch to list prefetch for a disorganized index
  - Index scan can be 2 to 6 times faster, CPU time is less too
  - REORG, CHECK, and RUNSTATS performance improvement
  - Sequential INSERT and DELETE improvement
  - Maximum elapsed time benefit occurs if:
    - *Cache hit ratio is high or index is on SSD*
    - *Index is striped*



Indexing Enhancements – IBM DB2 Tools Support

- OMPE
  - I/O Parallelism
- DB2 Admin Tool and DB2 Object Compare, QWT
  - Include Columns
- OMPE, QWT, QM
  - RID Lists
  - Sequential Inserts
- Query Workload Tuner
  - Index recommendations

Security for Today's Challenges

- Concerns about security
- New Access Control Authorities
- New Bind Options
- New Audit Capabilities
- Row/Column Access control

Auditors concerns about security …

- Worried as much about internal as external threats
- Reduce the number of people with access to sensitive data
- Remove privileged users having access to data
  - Be able to monitor privileged user access to data
- Separation of duties between those who manage the data and those who manage the access to the data
- Policies need to be in place for distributed users

Auditors concerns about security …

- Granularity of Database privileges
  - Privileges are granted at the database object level
  - Difficult to protect personal and sensitive data
  - Can not easily comply with data protection laws

Auditors concerns about security …

- Overloading applications with security logic
  - Security logic can be bypassed by malicious users
  - Hampers the ability to use ad-hoc query tools
  - Difficult to maintain
- Different views for different groups of users
  - Views updatability may not reflect security policies
  - Can be bypassed by malicious users
  - Difficult to maintain
- Evolution of security policies
  - Affect the security logic in applications
  - Affect the organization and number of views

Separation of Duties …

- New ZParm – SEPARATE_SECURITY
  - Specified on DSNTIPB
    - YES – Users with SYSADM can not perform GRANTS on objects created by others
    - NO – Users with SYSADM can administer security for all objects
    - Available in CM Mode
    - Users with INSTALL SYSADM can still perform GRANTS for other users
- SYSADM/INSTALL SYSADM data access remains unchanged
  - Future direction is to only use SECADM for security and INSTALL SYSADM for install activities
  - Users with SYSADM or INSTALL SYSADM can still view all data within tables
  - Recommendation:
    - Limit the use of INSTALL SYSADM and SYSADM to only when needed

Separation of Duties …

- New SECADM authority to manage security of data
  - Does not have access to the data
  - Is able to manage GRANTS on all objects
- New Administrative authorities
  - DATAACCESS – to control who can see data
  - ACCESSCTRL – to control who can govern access to the data
- New DBADM ON SYSTEM authority
  - Gives the individual or role the ability to
    - *Manage all user tables in the subsystem*
      - *With or without DATAACCESS – default is with DATAACCESS*
      - *With or without ACCESSCTRL - default is with ACCESSCTRL*
- Administrative authorities can be divided among individuals without overlapping responsibilities
- Allows security administrator to grant the minimum privilege to a user to perform a specific task

INSTALL SECADM …

- A person or role that manages DB2 objects
  - This separates the object management from data access and data control
- No inherent access to data
- Specified in SECADM1 and SECADM2
  - Install panel DSNTIPB
  - In conjunction with SECADM_TYPE
    - which can be AUTHID or ROLE
- Set INSTALL SECADM before setting SEPARATE_SECURITY to YES
- Activated by SEPARATE_SECURITY ZParm
  - If YES, then SYSADM and SYSCTRL can not perform GRANTS for others

INSTALL SECADM

- What can SECADM do?
  - GRANT Role privileges
  - CREATE, COMMENT, DROP ROLE
  - CREATE, ALTER, COMMENT, DROP TRUSTED CONTEXT
  - New DB2 10 Audit privileges
    - SELECT, INSERT, UPDATE, DELETE on new SYSIBM.SYSAUDITPOLICIES table
  - New DB2 10 row and column access
    - CREATE, ALTER, COMMENT, DROP row permissions and column masks
    - ALTER TABLE to activate row and column level access control
    - CREATE_SECURE_OBJECT privilege
  - SELECT, INSERT,UPDATE, DELETE on catalog tables

SQLADM

- Designed to be used by Performance analyst
- This will allow performance analyst to do all performance work, except access data
- What can a person with SQLADM do?
  - Issue SQL EXPLAIN statement
  - Issue START, STOP and DISPLAY PROFILE commands
  - Perform actions involving
    - EXPLAIN privilege
    - STATS privilege on all user databases
    - MONITOR2 privilege
    - Execute DB2 supplied stored procedures and routines
  - Ability to SELECT, INSERT, UPDATE, DELETE on DB2 catalog tables
  - CAN NOT access data, perform DDL or EXECUTE plans or packages

NEW PRIVILEGES – EXPLAIN

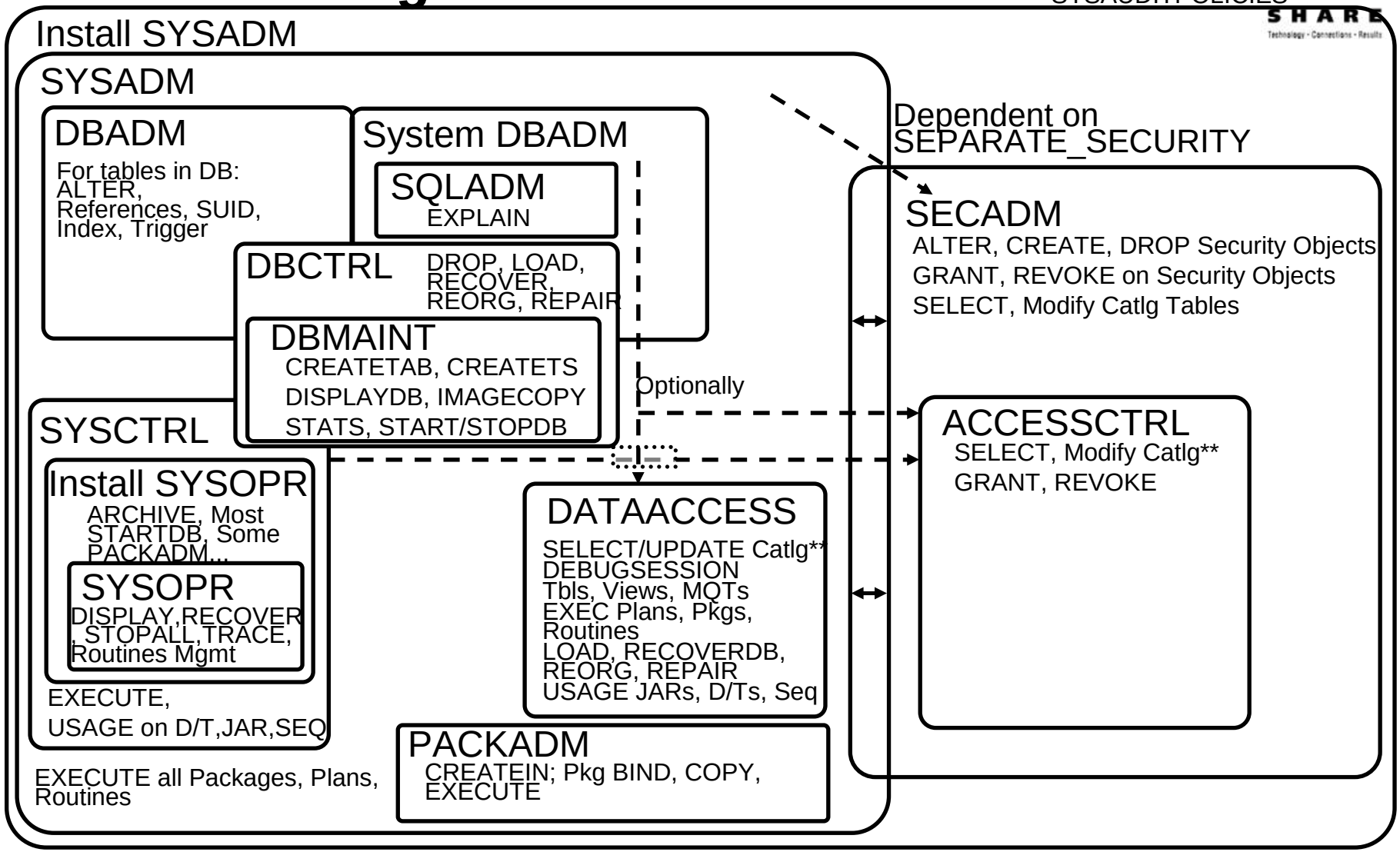
- Designed for the application architect
- What can a user do with the EXPLAIN privilege?
  - Issue SQL EXPLAIN ALL statement without being able to EXECUTE that statement
  - Issue SQL PREPARE and DESCRIBE TABLE statements without having privileges on the object
  - BIND EXPLAIN(ONLY) and SQLERROR(CHECK)
    - REBIND...EXPLAIN(ONLY) added with PM25679
  - Explain dynamic SQL statements executing under new special register
    - CURRENT EXPLAIN MODE = EXPLAIN

OPT

NEW BIND OPTIONS – EXPLAIN(ONLY) & SQLERROR(CHECK)

- EXPLAIN(ONLY)
  - Provides the ability to EXPLAIN statements without the ability to execute them
  - Requires EXPLAIN privilege or necessary BIND privileges
  - Populates the EXPLAIN tables without creating a package
- SQLERROR(CHECK)
  - Provides the ability to syntax and semantic check the SQL statements being bound without the ability to execute the statement(s)

Authorities Diagram



REVOKE DEPENDENT PRIVILEGES …

- Provides additional controls regarding cascading effects of a REVOKE statement
  - INCLUDING DEPENDENT PRIVILEGES
  - NOT INCLUDING DEPENDENT PRIVILEGES
    - When ACCESSCTRL, DATAACCESS, or DBADM ON SYSTEM is revoked,
      - *the default is always NOT INCLUDING DEPENDENT PRIVILEGES and*
      - *the NOT INCLUDING DEPENDENT PRIVILEGES clause must be explicitly specified*

REVOKE DEPENDENT PRIVILEGES …

- ZParm – REVOKE_DEP_PRIVILEGES
  - Panel DSNTIPP1 –
  - Values
    - *NO, YES, SQLSTMT*
    - *NO*
      - *You can not specify INCLUDING DEPENDENT PRIVILEGES*
      - *Dependent privileges CAN NOT be cascaded*
    - *YES - This is pre DB2 10 behavior*
      - *All revokes will include dependent privileges except when ACCESSCTRL, DATAACCESS and SYSTEMDBA are revoked*
    - *SQLSTMT*
      - *Controlled at the SQL statement level as specified in the REVOKE statement. THIS IS THE DEFAULT*

REVOKE DEPENDENT PRIVILEGES without cascading revokes…

```
SET CURRENT SQLID = 'DNET775';
GRANT SELECT ON DNET775.CUSTOMER TO FRANK WITH GRANT OPTION;
SELECT
  SUBSTR(GRANTOR,1,8) AS GRANTOR,
  SUBSTR(GRANTEE,1,8) AS GRANTEE,
  SELECTAUTH
FROM SYSIBM.SYSTABAUTH
WHERE TCREATOR = 'DNET775' AND TTNAME = 'CUSTOMER';
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G
CUSTOMER	DNET775	FRANK	G

REVOKE DEPENDENT PRIVILEGES without cascading revokes…

```
SET CURRENT SQLID = 'FRANK';
GRANT SELECT ON DNET775.CUSTOMER TO STAN;
SELECT
  SUBSTR(GRANTOR,1,8) AS GRANTOR,
  SUBSTR(GRANTEE,1,8) AS GRANTEE,
  SELECTAUTH
FROM SYSIBM.SYSTABAUTH
WHERE TCREATOR = 'DNET775' AND TTNAME = 'CUSTOMER';
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G
CUSTOMER	DNET775	FRANK	G
CUSTOMER	FRANK	STAN	Y

REVOKE DEPENDENT PRIVILEGES without cascading revokes and REVOKE BY

```
SET CURRENT SQLID = 'DNET775';
REVOKE SELECT ON DNET775.CUSTOMER FROM FRANK
  NOT INCLUDING DEPENDENT PRIVILEGES;
SELECT
  SUBSTR(GRANTOR,1,8) AS GRANTOR,
  SUBSTR(GRANTEE,1,8) AS GRANTEE,
  SELECTAUTH
FROM SYSIBM.SYSTABAUTH
WHERE TCREATOR = 'DNET775' AND TTNAME = 'CUSTOMER';
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G
CUSTOMER	FRANK	STAN	Y

```
REVOKE SELECT ON DNET775.CUSTOMER FROM STAN BY FRANK;
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G

DB2 Audit Capability …

- New audit capabilities without additional data collectors
- New Audit Policies are managed in the catalog
  - Audit policy provides wild carding of table names
- Ability to audit (associated column names identified in upper case)
  - Privileged users (SYSADMIN, DBADMIN)
    - Audit policy records each use of a administrative system authority
    - For DBADMIN, optional DBNAME or COLLID (for PACKADM)
  - SQL activity against a table (EXECUTE)
    - Audit policy does not require AUDIT clause to be specified
    - Audit policy generates records for all read and update access, not just first access in the transaction
    - UTS, Classic Partitioned, and Segmented table space support
  - Trusted Context use (VALIDATE)
    - When established or used by another different user (ASUSER connection)

DB2 Audit Capability

- Ability to audit (continued)
  - Authorization & authentication failures (CHECKING)
  - The alter or drop of a table (OBJMAINT + OBJ* identification columns)
  - Utility start, change, or end (CONTEXT)
  - Grants, revokes, or create/alter of a Trusted Context (SECMAINT)
- Various IFCIDs created for the different audit types
- External collectors only report users with a system authority
- Audit Policies can be started at DB2 start
  - DB2START
    - Y: Starts at DB2 start
    - S: Starts at DB2 start and can only be changed by SECADM

DB2 Audit Capability …

- To create an AUDIT POLICY
  - Insert a row into new SYSAUDITPOLICIES table
  - Specify the category and related fields
    - See SQL Reference Appendix A
  - Issue START TRACE command with audit policy name to enable audit policy
  - Issue STOP TRACE command with audit policy name to disable audit policy
  - Up to 8 audit policies can be specified to auto start when DB2 is started

DB2 Audit Capability …

SYSIBM.SYSAUDITPOLICIES table

Column Name	Col No	Col Type	Length
AUDITPOLICYNAME	1	VARCHAR	128
OBJECTSCHEMA	2	VARCHAR	128
OBJECTNAME	3	VARCHAR	128
OBJECTTYPE	4	CHAR	1
CREATEDTS	5	TIMESTMP	10
ALTEREDTS	6	TIMESTMP	10
CHECKING	7	CHAR	1
VALIDATE	8	CHAR	1
OBJMAINT	9	CHAR	1
EXECUTE	10	CHAR	1
CONTEXT	11	CHAR	1
SECMAINT	12	CHAR	1
SYSADMIN	13	VARCHAR	128
DBADMIN	14	VARCHAR	128
DBNAME	15	VARCHAR	24
COLLID	16	VARCHAR	128
DB2START	17	CHAR	1
IBMREQD	18	CHAR	1

DB2 Audit Capability …

- To create a new AUDITADMIN1 policy to audit the SYSADM authority (S) and the SYSOPR authority (O), you can specify SYSADMIN as the category:

```
INSERT INTO SYSIBM.SYSAUDITPOLICIES
   (AUDITPOLICYNAME, SYSADMIN)
 VALUES('AUDITADMIN1','OS');
```

- You can also use the SQL LIKE predicate to audit tables of the same characteristics. For example, you can audit all tables that start with E_P in schema TSCHEMA by issuing the following INSERT statement:
  - OBJECTTYPE ‘T’ means table
  - EXECUTE ‘C’ means Audit on all INSERT,UPDATE,DELETE statements
  - EXECUTE ‘A’ means Audit all access on the table

```
INSERT INTO SYSIBM.SYSAUDITPOLICIES
   (AUDITPOLICYNAME, OBJECTSCHEMA, OBJECTNAME, OBJECTTYPE, EXECUTE)
 VALUES('TEST2','TSCHEMA','''E_P%''','T','C');
```

Row and Column level access …

- What is the purpose of row level security?
  - Filter rows out of answer set
  - Policy can use session information like SQL ID is in what group or user is using what role to control when row is returned in result set
  - Applicable to SELECT,INSERT, UPDATE, DELETE & MERGE
  - Defined as a row permission:

```
CREATE PERMISSION policy-name ON table-name
  FOR ROWS WHERE search-condition
  ENFORCED FOR ALL ACCESS ENABLE
```

  - Optimizer inserts search condition in all SQL statements accessing table. If row satisfies search-condition, row is returned in the answer set

Row and Column level access …

- What is the purpose of column level security?
  - Mask column values in answer set
  - Applicable to the output of outermost subselect
  - Defined as column masks:

```
CREATE MASK mask-name ON table-name FOR COLUMN
  column-name RETURN CASE expression ENABLE;
```

  - Optimizer inserts CASE statement in all SQL accessing table to determine mask value to return in answer set

Row and Column level access …

- Define a column or row policy based on who is accessing the table
  - SESSION-USER
    - Primary authorization ID of the process
  - CURRENT SQLID
    - SQL authorization ID of the process
    - SET CURRENT SQLID = some authorization id
  - VERIFY_GROUP_FOR_USER (new BIF)
    - Get authorization IDs for the value in SESSION_USER
      - *Gets both primary and secondary auth ids*
      - *Return 1 if any of those auth IDs are in the argument*

```
WHERE
  VERIFY_GROUP_FOR_USER(SESSION_USER,'MGR','PAYROLL') = 1
```

  - VERIFY_ROLE_FOR_USER (new BIF)
    - Get the role for the value in SESSION_USER
    - Return 1 if the role is in the argument list

```
WHERE
  VERIFY_ROLE_FOR_USER(SESSION_USER,'MGR','PAYROLL') = 1
```

Row and Column level access

- Row and Column Access Control
  - When activated row and column access controls:
    - Make row permissions and column masks become effective in all DML
      - *All row permissions are connected with 'OR' to filter out rows*
      - *All column masks are applied to mask output*
      - *Rebind required for dependent packages*
      - *Modified statements shown in DSN_PREDICAT_TABLE*
      - *IFCID 145 names the Mask / Permission enabled at prepare / bind time*
    - Halts all access to the table if no user-defined row permissions

```
ALTER TABLE table-name
   ACTIVATE ROW LEVEL ACCESS CONTROL
   ACTIVATE COLUMN LEVEL ACCESS CONTROL;
```

  - When deactivated row and column access controls:
    - *Make row permissions and column masks become ineffective in DML*
  - Opens all access to the table

```
ALTER TABLE table-name
   DEACTIVATE ROW LEVEL ACCESS CONTROL
   DEACTIVATE COLUMN LEVEL ACCESS CONTROL;
```

Row and Column level access – Banking example …

- A Simple banking scenario

ACCOUNT	NAME	PHONE	INCOME	BRANCH
1111-2222-3333-4444	Alice	111-1111	22,000	A
2222-3333-4444-5555	Bob	222-2222	71,000	B
3333-4444-5555-6666	Louis	333-3333	123,000	B
4444-5555-6666-7777	David	444-4444	172,000	C

Row and Column level access – Banking example …

- Determine access control rules for customer service rep
  - Allow access to all customers of the bank (a row permission)
  - Mask all INCOME values (a column mask)
    - Return value 0 for incomes of 25000 and below
    - Return value 1 for incomes between 25000 and 75000
    - Return value 2 for incomes between 75000 and 150000
    - Return value 3 for incomes above 150000
  - All are in the CSR group (who)
- Create a row permission for customer service representatives

```
CREATE PERMISSION CSR_ROW_ACCESS ON CUSTOMER
FOR ROWS WHERE
VERIFY_GROUP_FOR_USER (SESSION_USER,'CSR') = 1
 AND BRANCH = 'B'
ENFORCED FOR ALL ACCESS;
```

Row and Column level access – Banking example…

- Create a column mask on INCOME for customer service rep

```
CREATE MASK INCOME_COLUMN_MASK ON CUSTOMER
FOR COLUMN INCOME RETURN
  CASE WHEN (VERIFY_GROUP_FOR_USER (SESSION_USER,'CSR') = 1)
     THEN CASE WHEN (INCOME > 150000) THEN 3
               WHEN (INCOME > 75000) THEN 2
               WHEN (INCOME > 25000) THEN 1
          ELSE 0
     END
  ELSE 0
  END
ENABLE;
```

Row and Column level access – Banking example…

- Activate Row-level and column-level access control

```
ALTER TABLE CUSTOMER
  ACTIVATE ROW    ACCESS CONTROL
  ACTIVATE COLUMN ACCESS CONTROL;
```

- What Happens in DB2?
  - A default row permission is created implicitly to prevent all access to table customer (WHERE 1=0)
  - All packages and cached statements that reference table CUSTOMER are invalidated

Row and Column level access – Banking example…

```
SELECT ACCOUNT, NAME, INCOME, PHONE,BRANCH
FROM CUSTOMER
WHERE BRANCH IN ('A','B');
```

ACCOUNT	NAME	INCOME	PHONE	BRANCH
2222-3333-4444-5555	Bob	1	222-2222	B
3333-4444-5555-6666	Louis	2	333-3333	B

INCOME is automatically masked by DB2

If the user is not a member of the CSR group, then no rows at all will be returned

Row and Column level access – Banking example

DB2 effectively evaluates the following revised query:

```
SELECT ACCOUNT,NAME,
   CASE WHEN(VERIFY_GROUP_FOR_USER(SESSION_USER,'CSR') = 1)
     THEN CASE WHEN(INCOME > 150000) THEN 3
               WHEN(INCOME >  75000) THEN 2
               WHEN(INCOME >  25000) THEN 1
               ELSE 0
          END
   ELSE NULL
   END AS INCOME,
   PHONE,BRANCH
FROM CUSTOMER
WHERE BRANCH IN ('A','B')
  AND ((1=0) OR (VERIFY_GROUP_FOR_USER(SESSION_USER,'CSR')) = 1)
  AND BRANCH = 'B';
```

If the user is not in the GROUP CSR, the VERIFY_GROUP_FOR_USER returns 0 and no rows are returned

Security – IBM DB2 Tools Support

- Guardium, Guardium Encryption, RACF, TCIM
  - Concerns about security (DB2, IMS, VSAM*)
- DB2 Admin Tool and Object Compare
  - New Access Control Authorities
- DB2 Admin Tool, Bind Manager
  - New Bind Options
- Guardium
  - New Audit Capabilities
- DB2 Administration Tool
  - Row/Column Access control

Distributed Data Performance

- New JDBC/ ODBC features
- DDF enhancements
- High Performance DBATs
- Online CDB Changes
- Elimination of private protocol
- Data Sharing Sysplex Workload Balancing Migration

Business value of DDF Enhancements

- Lessen CPU consumption of remote threads
  - High performance DBAT can reuse threads very efficiently
  - Continued improvement of applications using JCC driver to attach locally
  - Recent addition of zAAP eligibility for type 2 connections

Local JDBC and ODBC Application Performance

- **Local Java and ODBC applications did not always perform faster compared to the same application called remotely**
  - Over last DB2 versions, DDF optimized processing with DBM1 that was not available to local ODBC and JDBC application. (RRS attach)
  - zIIP eligibility significantly reduced chargeable CP consumption (thru DDF)
- **Extend DDF optimization in DBM1 to local JCC type 2 and ODBC z/OS driver**
  - Limited block fetch and MRF recently added to DB2 9
  - LOB progressive streaming
  - zAAP offload for local type 2 attach (PK77599)
- **Expect significant performance improvement for applications with**
  - Queries that return a single row
  - Queries that return LOBs (eliminate materialization of >2MB LOBs)

LOB

DDF Enhancements (available in CM8 or CM9)

- **Improved performance by means of internal restructuring of distributed processing on the server**
  - Optimizes communication between DDF and DBM1 (internal)
  - Flow DRDA message flows in UNICODE
- **Support of implicit close for cursors declared WITH HOLD and FETCH FIRST FOR 1 ROWS ONLY**
  - Avoids network trip for CLOSE CURSOR if only 1 row is qualifying and cursor is defined WITH HOLD
- **DDF restart LIGHT(YES) – concerned with DB2 participant role**
  - Allows other DB2s to regain control of remote in-doubt UR threads
  - No new connections

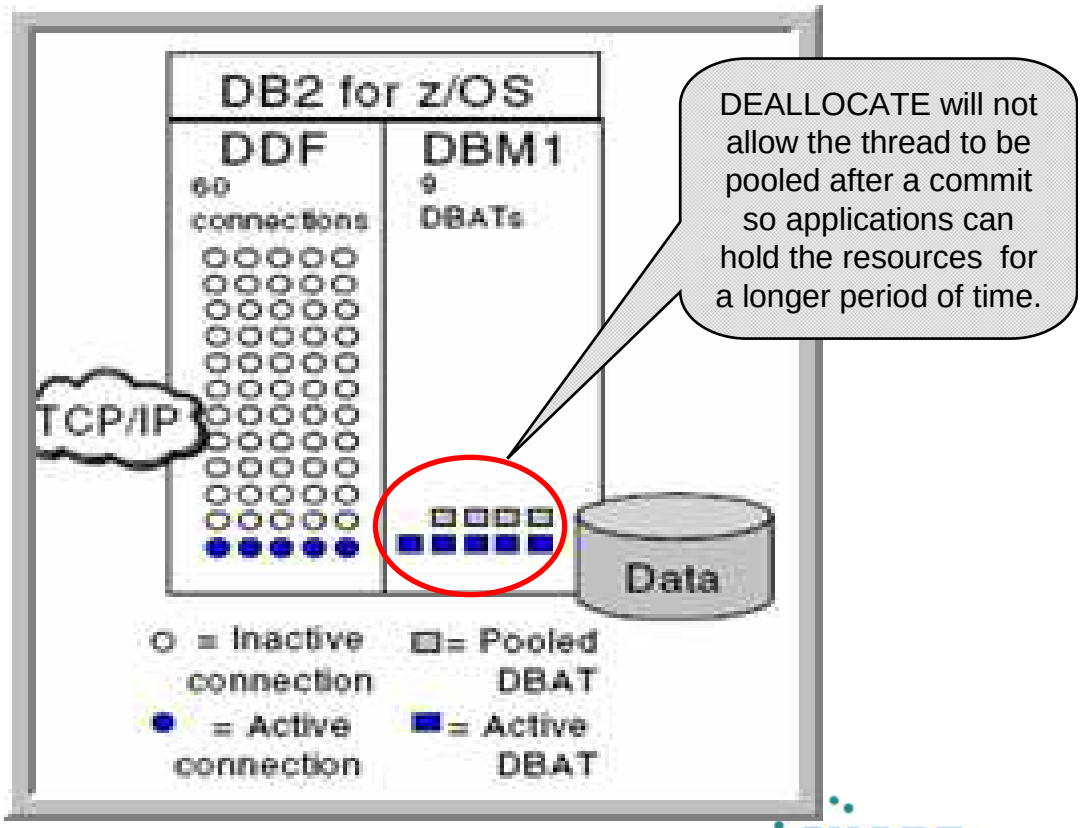
Pre-High Performance DBAT

- **Distributed usage of RELEASE(DEALLOCATE) magnified data maintenance restrictions**
  - Customers could not break in to do DDL, BIND
  - STOP DDF MODE(SUSPEND) was provided to purge pooled DBATs but was not sufficient
- **V6 introduced type-2 inactive connection support (DBATs are pooled)**
  - Initially, packages were allocated based on their RELEASE bind option
    - Pooled DBATS could linger (until POOLINAC expired) and/or get created with RELEASE(DEALLOCATE) package sections and hold table space intent locks
  - PQ63185 was implemented to ALWAYS allocate packages on DRDA DBATs with RELEASE(COMMIT)

DB2 z/OS Inactive Thread Support

- The benefits for DB2 z/OS Thread Pooling and CMTSTAT=INACTIVE
  - CPU savings in DB2, by avoiding repeated creation and destruction of DBATs
  - Real memory savings in z/OS, by reducing the number of DBATs
  - Virtual storage savings in DBM1, by reducing the number of DBATs
  - Greater capacity to support DRDA connections
- RELEASE (DEALLOCATE)
  - Holds thread across commits, limiting pooling

- Candidates for High performance DBATs should be short running highly active threads similar to CICS trans that use protected threads



High Performance DBAT…in 2 steps

- **High Performance DBATs reduce CPU consumption by**
  - Supporting RELEASE(DEALLOCATE) to avoid repeated package allocation/deallocation
  - xPROCs, CTs and PTs, lookaside and prefetch will not have to be re-initialized
  - Bigger CPU reduction for short transactions
- **High Performance DBAT behavior …**STEP 1 package must be RELEASE(DEALLOCATE)
  - DBAT will stay active with connection when DBAT is about to go be pooled and there is at least one RELEASE(DEALLOCATE) package still existing
  - Normal idle thread time-out detection will be applied to these DBATs. If DBAT is in completed unit-of-work status, Connection will turn inactive instead of being canceled (POOLINAC value controls this)
    - Connections will turn inactive after 200 times (not changeable) to free up DBAT (same as in previous releases)

High Performance DBAT…

- **New -MODIFY DDF PKGREL(BNDOPT/COMMIT) command** … STEP 2
  - to alter DDF's inactive connection processing which is activated via the ZPARM, CMTSTAT=INACTIVE
  - Display command shows DSNL106I message with PKGREL= BNDOPT or COMMIT
  - 2 options
    - PKGREL(BNDOPT) honors package bind option
    - PKGREL(COMMIT) forces package bind option RELEASE(COMMIT)
      - *Same as v6- DB2 9 inactive connection behavior*
      - *Will allow BIND and DDL to run concurrently with distributed work*

Online Remote Connection Changes

- Online CDB Changes
  - DDF is notified when a new connection added / committed in CDB
  - DDF will read the CDB and for updated information without refresh
  - **-DIS LOCATION** to see location attributes

```
-MODIFY DDF ALIAS(DB2A1) START
 STARTRBA=0000219E4000,ENDRBA=00002ID67FFF
 DSNL314I  -DB2A DSNLILNR THE ALIAS DB2A1 IS STARTED
STR JOB MSGLEVEL=1
```

- -MODIFY DDF ALIAS..
  - 40 dynamic aliases available
    - In addition to 8 static (DSNJU003)
  - Add, Remove, Start/Stop, Change attributes for location aliases while DDF is running
  - Stop an Alias to edit
  - **-DIS DDF**
    - Alias information & status
    - Does not show in DSNJU004

```
DSNL080I  -DB2A DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
DSNL081I STATUS=STARTD
DSNL082I LOCATION           LUNAME            GENERICLU
DSNL083I STLEC1             USIBMSY.SYEC1DB2  -NONE
DSNL084I TCPPORT=446   SECPORT=0     RESPORT=5001  IPNAME=-NONE
DSNL085I IPADDR=::9.30.88.185
DSNL086I SQL    DOMAIN=FVTEC706.vmec.svl.ibm.com
DSNL087I ALIAS              PORT  SECPORT STATUS
DSNL088I DB2A1              5446  0       STARTD
DSNL090I DT=I  CONDBAT=      64 MDBAT=    64
DSNL092I ADBAT=     0 QUEDBAT=      0 INADBAT=      0 CONQUED=      0
DSNL093I DSCDBAT=       0 INACONN=       0
DSNL105I CURRENT DDF OPTIONS ARE:
DSNL106I PKGREL = COMMIT
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
***
```

Handling of Private Protocol Requests in V10

- Jobs to analyze and prepare for private protocol elimination
  - PK64045 – delivered DSNTP2DP update, as well as DSNTPPCK
- Ability to Enable or Disable Private protocol (PK92339)
  - Via ZPARM PRIVATE_PROTOCOL in DSN6FAC Macro
  - Enables testing. PRIVATE_PROTOCOL=NO ZParm will mimic DB2 10 behavior in V8/V9
- DB2 10 will respond to a Private protocol response as follows, from a v9 or prior system
  - Reject request
    - VTAM sense code ‘10086021’
  - Requestor will receive
    - -904 Reason Code ’00D31026’ Type ‘1001’
  - DB2 10 DDF will reject a BIND with Private protocol
    - SQLCODE -30104
- Attempt to load/execute object with DBPROTOCOL column = ‘P’ will fail with SQLCODE -904, reason code 00E3001E, saying it needs to convert to DRDA, except the following case
  - If a package with PRIVATE PROTOCOL accesses local data only it is allowed. If it attempts to access remote data it will fail

Plan Ownership Authorization Elimination

- Plan Ownership Authorization is a Private Protocol security semantic that has been allowed with DRDA for previous releases of DB2 DB2 z/OS Client to DB2 z/OS Server
  - Plan Owner must have EXECUTE authority for SQL at the DB2 Requester
- With Private Protocol now eliminated in DB2 10, Plan Ownership Authorization is being removed from DRDA
  - PM17665 removes this behavior in DB2 V8 and DB2 9 with PRIVATE_PROTOCOL=NO
  - PM17665 removes this behavior in DB2 10
  - The Primary Authorization ID must have EXECUTE authority of the package at the server
- PM17665 introduces a potential release incompatibility in DB2 10
- PM37300
  - Introduces PRIVATE_PROTOCOL=AUTH to DB2 V8, 9, and 10
    - This allows Private Protocol Authorization without allowing Private Protocol
    - For DB2 V8 and 9 this defaults to YES
    - For DB2 10 this defaults to NO
  - PRIVATE_PROTOCOL=NO supports secondary authorization IDs for SQL EXECUTE authority at the DB2 Server

Data Sharing Sysplex Workload Balance Migration

- PM24292 addresses the DB2 server reporting mixed levels of Product Identification and DRDA Support
- Possible migration issue
  - Data Sharing groups that remain up during
    - Conversion to NFM
    - Reversion from NFM
  - Remote applications could encounter error codes -4212, -4499
- To allow for a smooth, seamless migration
  - Review II14619
  - Review PM24292 & apply UK65312 to DB2 10 (the target version)
  - IBM Data Server Client support in V9.7 FP 3A

- PM43293 introduced connection queue depth and wait time monitors (off by default)
  - MAXCONQN: Max number of inactive or new connection requests waiting for a DBAT
  - MAXCONQW: Max wait time for a connection request to wait for a DBAT.

DSH

DDF

Distributed Data – IBM DB2 Tools Support

- DB2 Administration Tool and Data Studio
  - DDF enhancements
  - Online CDB Changes
- OMPE , pureQuery, QWT
  - New JDBC/ ODBC features
  - High Performance DBATs

Utility Enhancements

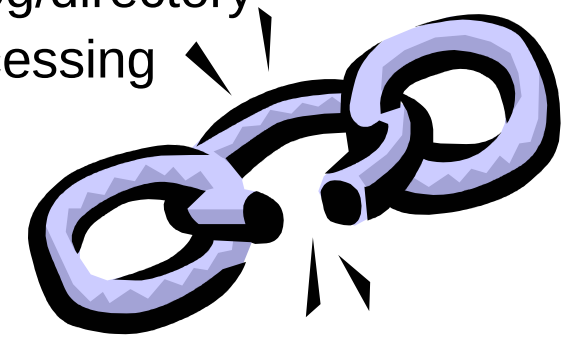
- UTSERIAL Lock
- REORG Table Space
- REORG AUX
- REORG INDEX
- LOAD
- UNLOAD
- COPY
- BACKUP/RECOVER

Business value statements

- Utilities are an integral part of DB2 to ensure availability while maintaining the highest degree of integrity and minimizing downtime
- Availability of the system catalog is improved by the removal of UTSERIAL locks so more utilities an run simultaneously
- Improved access to LOB data during REORG's by allowing Reorg share level change
- We can now ensure utilities will complete by canceling blocking threads
- Recovery times can be minimized by rolling backward through the log

Utility Enhancements

- Unless explicitly stated, new utility functions are only available in NFM
- Improved access to SYSUTILX by removal of the UTSERIAL lock
- REORG SHRLEVEL(CHANGE) for complete catalog/directory
  - Hashes and links are removed during ENFM processing
  - Row level locks are relied upon
  - No changes to utility jobs are necessary
  - Greatly improves access to catalog/directory
- Improves concurrency for online REORG
  - Update of catalog statistic information for inline REORGs occurs after access to the table is allowed



REORG with LOB columns

- New keyword AUX used in REORG to specify how to handle AUX table spaces
  - Allows LOB rows to flow with base table rows for partitioned table spaces
  - Allows REBALANCE
  - Allows ALTER LIMITKEY
  - Makes sure associated LOB rows are deleted with DISCARD
  - May leave auxiliary spaces COPYP
- REORG…AUX(YES|NO)
  - YES – Include LOB table spaces in the REORG
  - NO – v9 behavior
  - Default = YES in the following cases:
    - REORG of entire user PBG space with LOB(s)
    - REORG with REBALANCE, DISCARD, or to materialize altered limit keys
    - REORG of SPT01 if in REORP or AREOR status

REORG...

- REORG TABLESPACE
  - SHRLEVEL CHANGE/REFERENCE
  - Materializes pending changes
    - Table space type, DSSIZE, SEGSIZE, data page size, index page size, member cluster
    - Partitioned table spaces must be REORG'd in their entirety
  - Runs even if table space is in hard REORP
    - Prior to DB2 10 required SHRLEVEL(NONE)
  - REORG SHRLEVEL(NONE)
    - Will not materialize pending DDL changes, but will proceed

REORG

- LOB REORG
  - **SHRLEVEL NONE in CMx / ENx**
    - *DSNU125I REORG SHRLEVEL NONE ON LOB TABLE SPACE WILL BE RESTRICTED IN A FUTURERELEASE OF DB2.*
  - **SHRLEVEL NONE** is not supported in NFM
    - *The job will end with RC=0 and NO LONGER SUPPORTED message*
    - *DSNU126I REORG SHRLEVEL NONE ON LOB TABLE SPACE IS NO LONGER SUPPORTED.*
- REORG TS with SHRLEVEL REFERENCE / CHANGE with AUX YES
  - pending changes that are associated with the base table space are materialized
  - pending changes that are associated with the LOB table spaces are not materialized.

REORG TABLESPACE…

- PM55051 PART REORG performance enhancement with NPSIs (SHRLEVEL REFERENCE or CHANGE)
  - SORTNPSI keyword, defaults to REORG_PART_SORT_NPSI (defaults to NO) system parameter
    - AUTO is an option for the REORG keyword and the system parameter
  - As the ratio of data included in the REORG increases, sort of full index keys provides a performance benefit
  - Up to a 60% ET reduction
  - All RIDs from parts are sorted once, instead of separating the sorts non-REORGed part keys and REORGed part keys
  - Also available in DB2 9
- REORG of multiple non-contiguous partitions allowed
  - Changes behavior from DB2 9
  - Customers who perform multi-part REORGs can now specify multiple parts in a single command
    - Ex: REORG TABLESPACE PARTS (1,5,10:15, 20:23)
  - Retro-fitted into v9 via PM22336

REORG TABLESPACE…

- REORG can CANCEL threads on last RETRY processing
  - Ensures REORG can break in on applications
  - Allows REORG to cancel threads that hold claims on objects that prevent drains
    - FORCE NONE|READERS|ALL
- New Advisory REORG Pending status – AREOR
  - Indicates that there are pending alters that require a table space level REORG
- Statistics
  - Now defaults to STATISTICS TABLE ALL INDEX ALL UPDATE ALL *HISTORY ALL*
  - RTS tracks REORGCLUSTERSENS
    - Times SQL was sensitive to cluster since REORG / CREATE
    - Can indicate a reduced need to REORG
- PM37622 enabled zIIP redirect for REORG UNLOAD

REORG INDEX

- REORG INDEX SHRLEVEL(REFERENCE or CHANGE)
  - Will materialize pending index changes so long as there are no pending table space changes
  - If there are pending TS changes, REORG proceeds and message DSNU275I RC=4 is issued indicating pending changes were not materialized
  - For indexes defined COPY YES, an entry will be made in SYSCOPY
- REORG INDEX SHRLEVEL(NONE)
  - Will not materialize pending index changes
  - DSNU1165I RC=4 will be issued indicating pending changes were not done

REORG INDEX

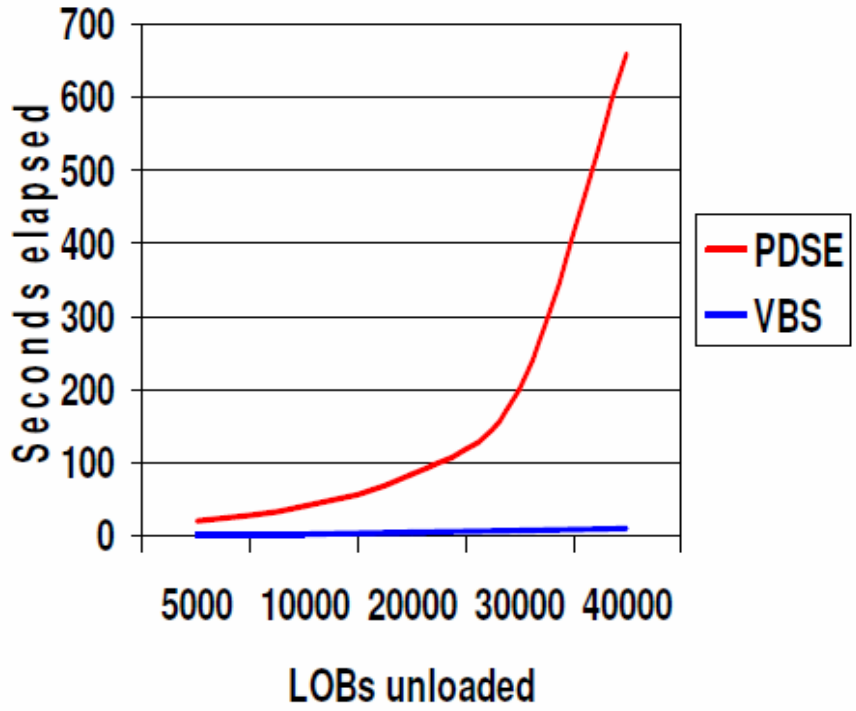
- When pending definitions are materialized in REORG INDEX with SHRLEVEL REFERENCE or CHANGE
  - Index statistics are collected by default and updated in the catalog
  - Default = STATISTICS UPDATE ALL HISTORY ALL
- STATISTICS if used
  - Will override defaults
  - New warning message will indicate some partition stats may no longer be accurate due to materialization of pending changes.
  - Users should execute RUNSTATS to recollect partition statistics to re-gather COLGROUP, KEYCARD, HISTOGRAM, frequency stats with NUMCOLS>1, and statistics for extended indexes

UNLOAD Spanned Record Support…

- LOB and XML columns can now be UNLOADed or LOADed to/from the same data set with other non-LOB/XML columns via spanned records
  - Performance of reading from or writing to a single sequential file is much faster than using separate files or partition data set members because the utilities will not have to use Open/Close for each LOB
  - When unloading LOB/XML columns to a sequential file, LOBs and XML documents will be written at the end of the record in their column definition order

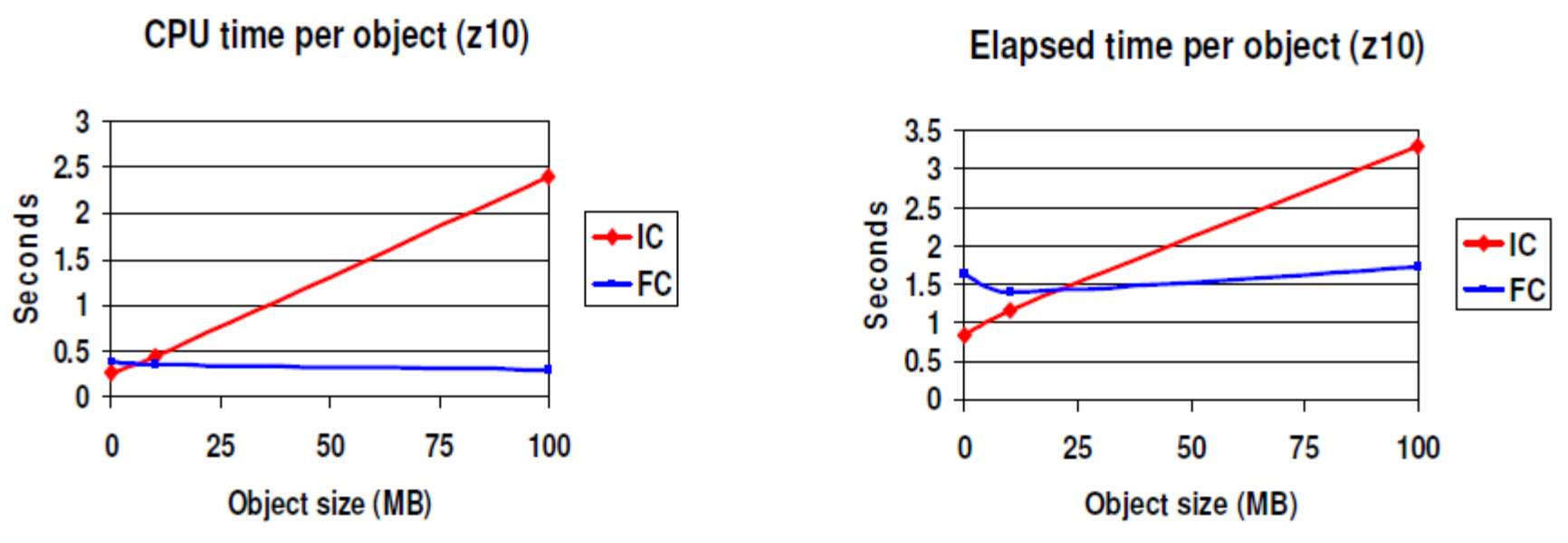
UNLOAD

- Spanned records support for UNLOAD
  - UNLOAD TABLESPACE xxx.yyy SPANNED YES FROM TABLE ttt (field spec list with CLOB and xml data at end)
  - Option for records > 32K
  - FRV were needed in the past



FLASHCOPY SUPPORT…

- Data set-level Flashcopy support
  - For COPY, REORG, LOAD, REBUILD INDEX, REORG INDEX, CHECK
  - Significant CPU and elapsed time reductions vs image copy



COPY

- Create transaction consistent image copies from a COPY SHRLEVEL CHANGE run using Flashcopy
  - DB2 10 uses the Flashcopy copy to make a transaction consistent image copy.
  - Don’t have to wait for blockers. No application outage and no quiesce.
  - Uncommitted units of work are rolled back against the Flashcopy
  - Image copy is then taken from the Flashcopy
  - Objects must be SMS managed residing on FC v2 volumes
- Improved data set management & performance
  - Will not allocate incremental copy data set unless copy taken
  - &ICTYPE now matches actual image copy
    - Regardless of the type specified on a template i.e. TYPE=C
  - Utilize RTS for CHANGELIMIT performance

RECOVER

- Point-in-time recovery with BACKOUT
  - Prior to DB2 10 recovered from a backup image then applied logs to reach the desired point in time.
  - DB2 now provides an option on utilities that specifies if the recovery should be performed by processing the log backwards or in the traditional restore and roll forward through the log process.
  - Changes are backed out from the current state.
  - Customer decides if BACKOUT is desirable option to use
    - IBM Recovery Expert can assist in making the best Recovery decision
  - In order to work successfully, indexes must be marked COPY YES since DB2 needs the SYSCOPY records for non-recoverable events to know whether the RECOVER can proceed or not.
    - PM30991 HIPER should be on to disallow a BACKOUT recovery when certain unlogged events exist.
  - PM45650 needed for RECOVER BACKOUT with LOBs

```
RECOVER TABLESPACE myschema.mytablespace BACKOUT YES
TOLOGPOINT x'00000518DFEF'
```

Utility Enhancements – IBM DB2 Tools Support

- DB2 Utilities Suite, DB2 Automation Tool, DB2 Admin Tool, High Performance Unload
  - UTSERIAL Lock
  - REORG Table Space
  - REORG AUX
  - REORG INDEX
  - LOAD/UNLOAD
  - UNLOAD
  - COPY
  - BACKUP/RECOVER
- DB2 Recovery Expert
  - Backup/Recover
  - Backup System
  - Recover System
- DB2 Cloning Tool
  - Copy replace
  - Unload replace
  - Load replace
  - Recover replace

Utilities Management Workshop for DB2 on z/OS

- Have you reviewed and updated your utilities management procedures to take advantage of the utility changes introduced with DB2 9 and DB2 10?
- If not, schedule a Utilities Management Workshop to:
  - Understand the characteristics of your DB2 utility maintenance.
  - Understand how to leverage features in the Utilities Suite.
  - Learn what utility features can save you time and money.
  - Understand your utility maintenance needs to meet SLAs.
  - Begin developing plans to reduce costs and increase performance and availability.
- Even if you are using tools to manage your utilities, you will find great benefit from this workshop. The tools may not automatically adjust for the newer versions of DB2.
- Contact Doug Clifton – cliftonw@us.ibm.com

Virtual Storage Improvements

- More Storage Above the Bar
  - Ability to run more threads
- In-memory Buffer pool objects
- Workfile Enhancements
- Additional support for 64 bit runtime
- 1 MB Page size
- System Latching changes
- On Demand Buffer Pool storage
- Log Buffer Page Fix

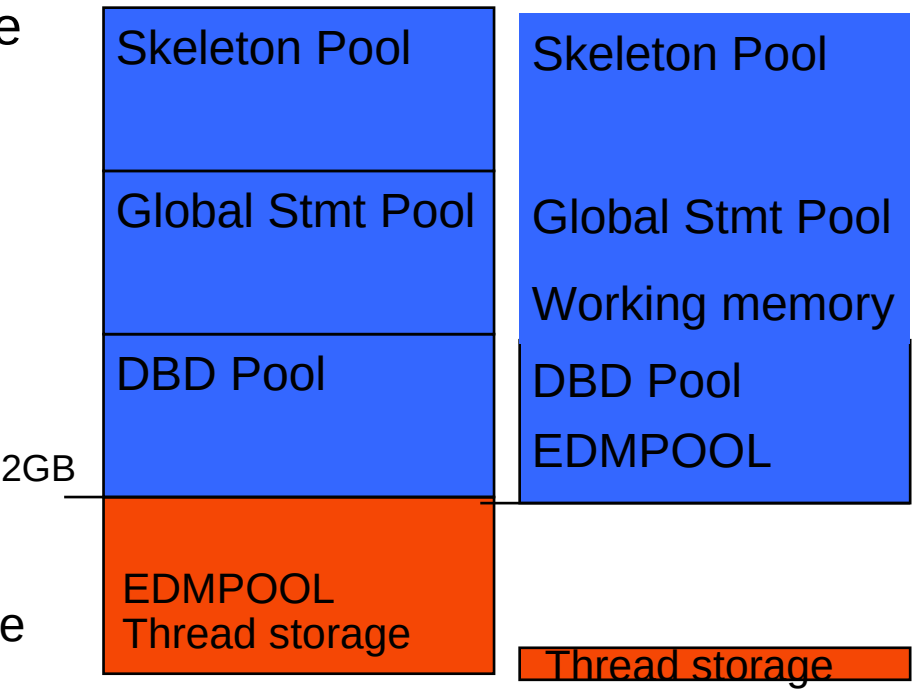
Virtual Storage Business Value

- Changes that began in force in V8
- Can offer significant constraint relief
- Leads to greater scalability
- Can reduce administration and reduce the need for horizontal scaling
- Enables new performance options in DB2 10

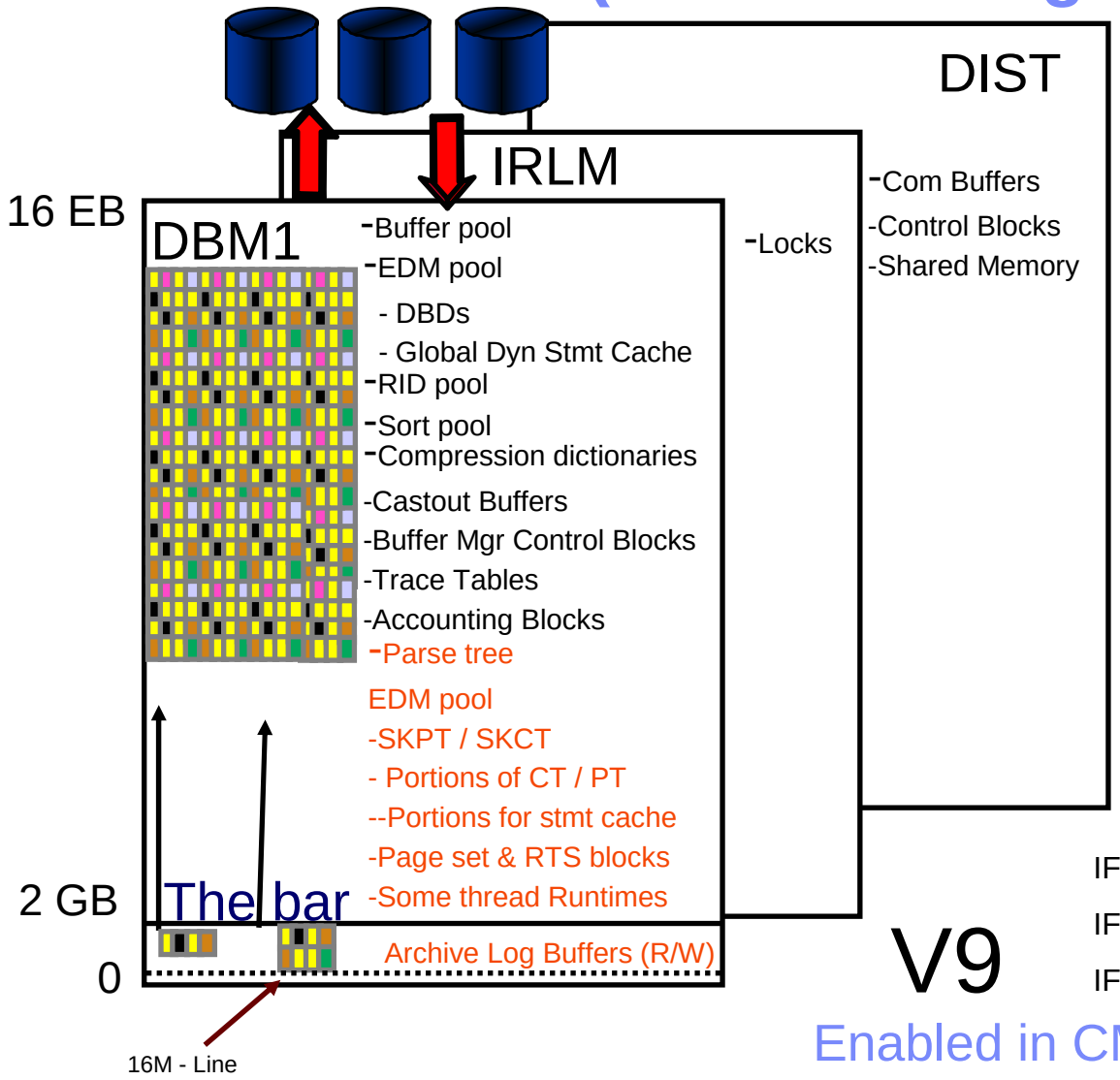
DB2 10: 64 bit Evolution (Virtual Storage Relief)

Scalability: Virtual storage constraint is still an important issue for many DB2 customers, until DB2 10

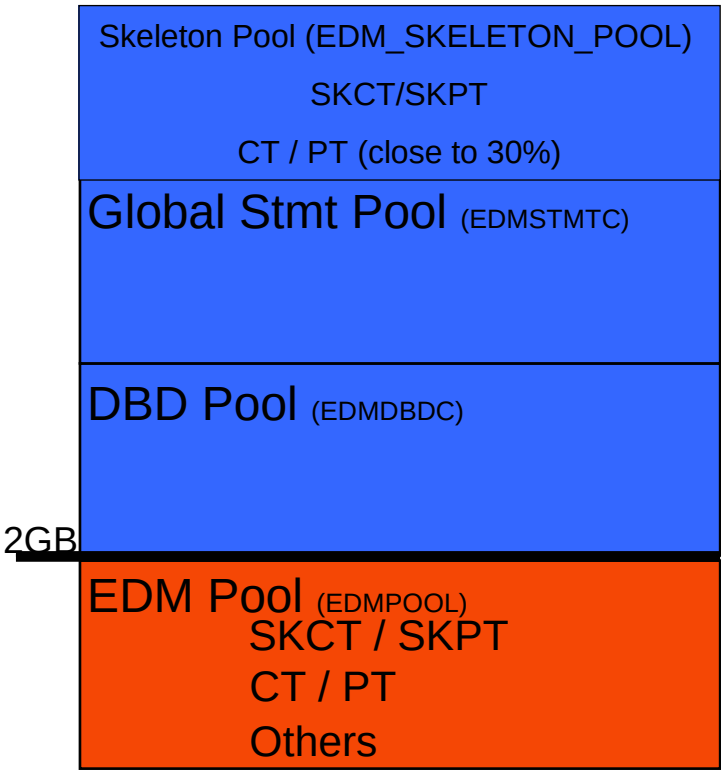
- DB2 10 supports 5-10x more active threads, up to 20,000 per member
  - 80-90% of thread storage moved above the bar
  - More concurrent work
  - Reduce need to monitor
  - Consolidate members and LPARs
  - Reduced cost, easier to manage, easier to grow
  - REBIND required to get most of the savings



64 bit Evolution (Virtual Storage Relief)



Environmental Descriptor Manager Pool



V9

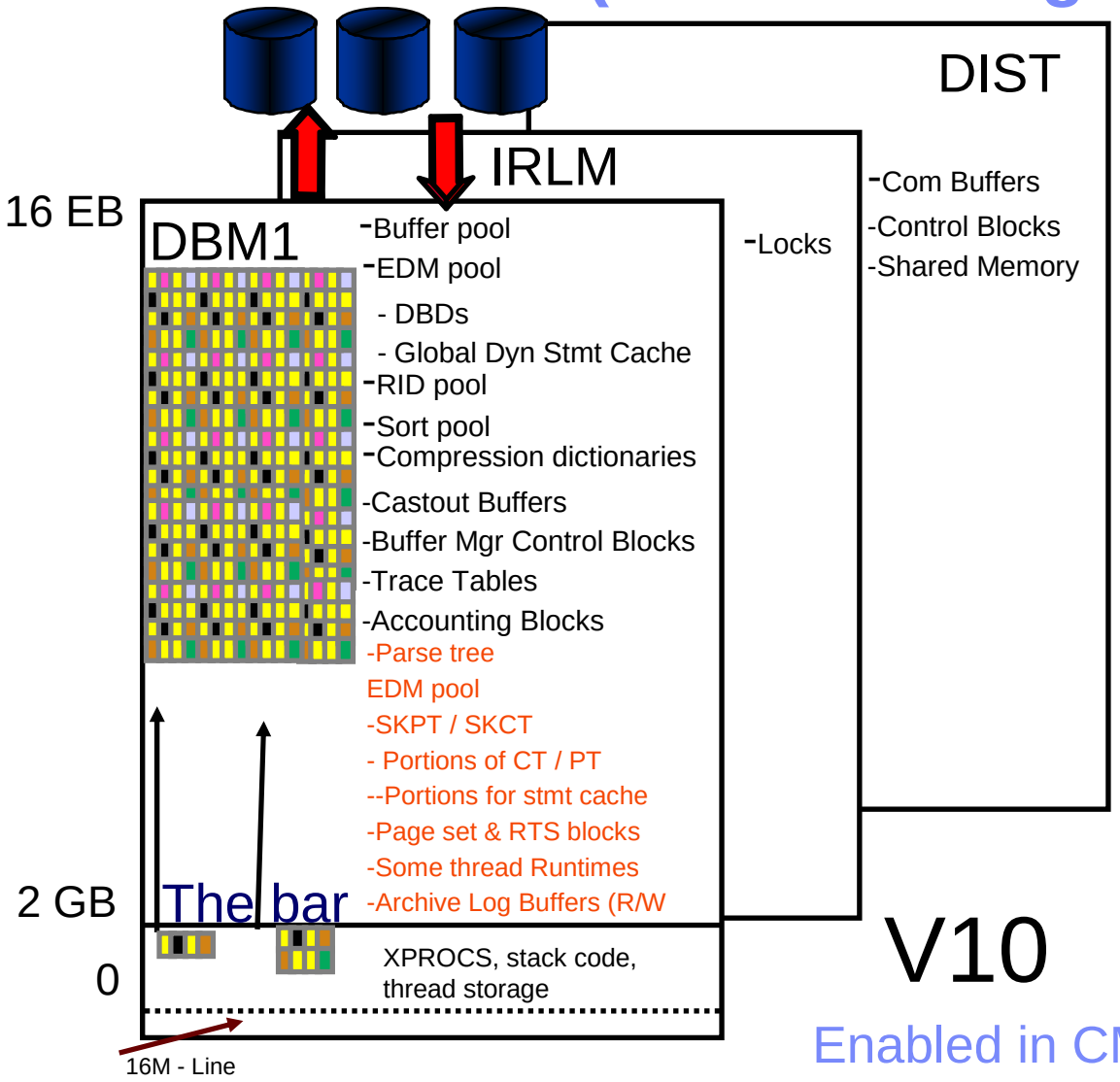
Enabled in CM

IFCID 217: detailed DBM1 virtual storage health

IFCID 225: consolidated DBM1 virtual storage health

IFCID 002: Mapping macro for EDM Pool, DSNDQISE

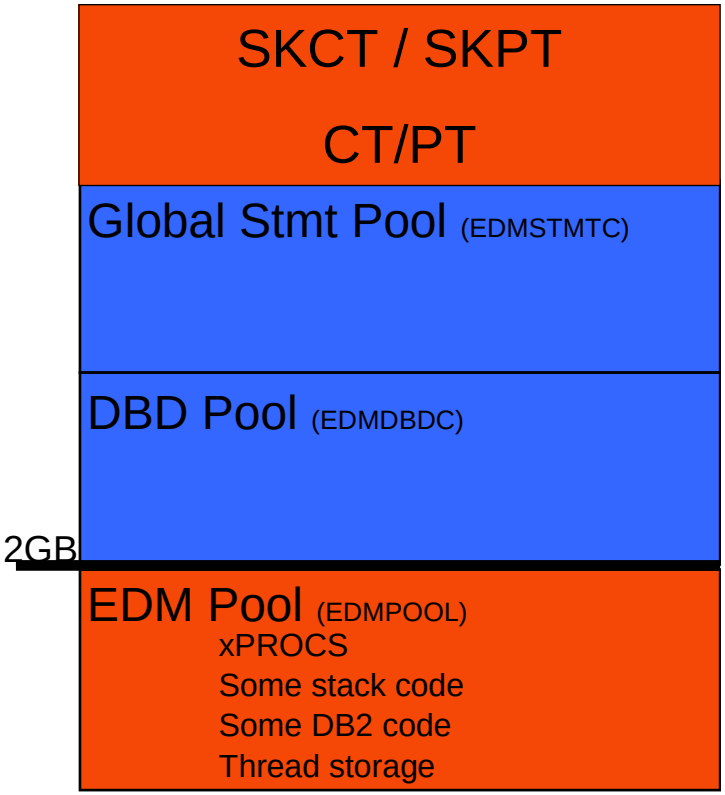
64 bit Evolution (Virtual Storage Relief)



V10

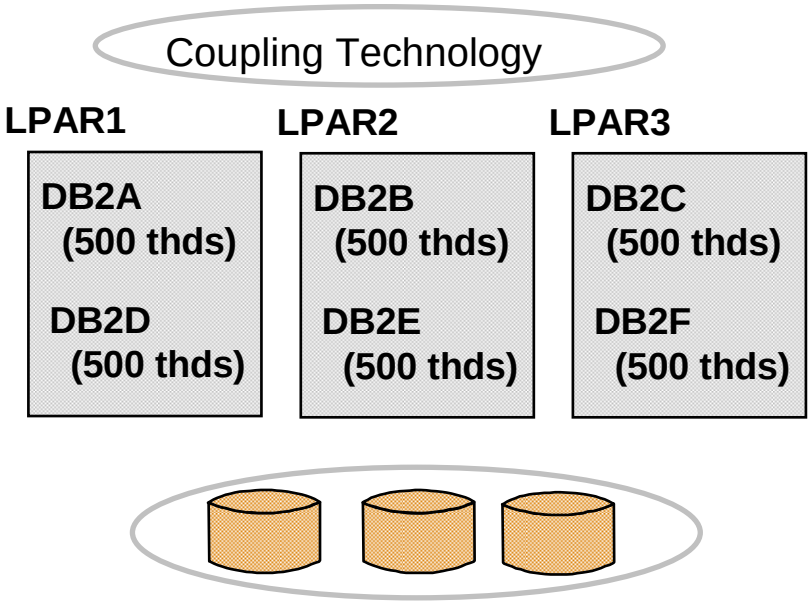
Enabled in CM

Environmental Descriptor Manager Pool



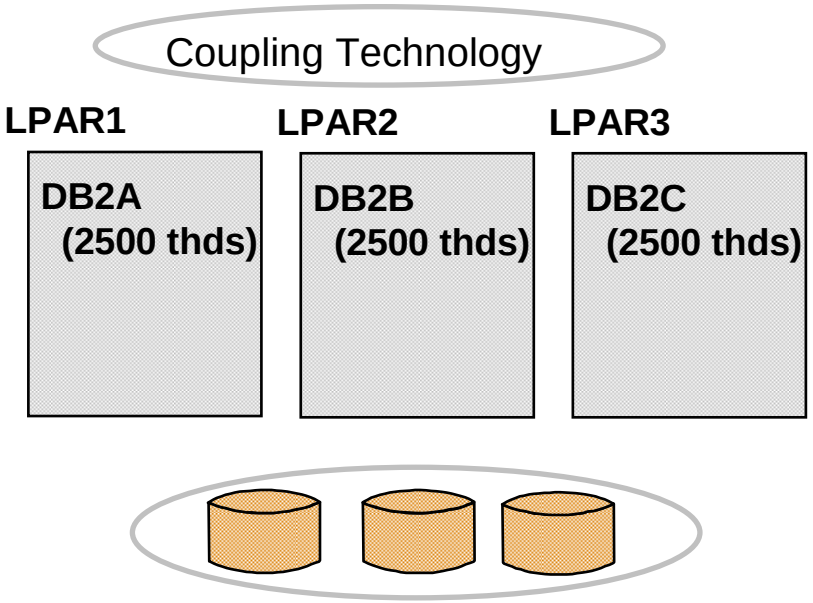
Running a Large Number of Threads

Pre DB2 10



- Data sharing and sysplex allows for efficient scale-out of DB2 images
- Sometimes multiple DB2s / LPAR

DB2 10



- More threads per DB2 image
- More efficient use of large n-ways
- Easier growth, lower costs, easier management

In Memory Table Spaces…

- We are able to let DB2 know that you have cached an entire tablespace into a buffer pool.
  - *Data will be preloaded into buffer pool when object is opened and will remain until closed.*
    - *Specified by PGSTEAL=NONE*
  - *Excellent for lookup tables and indexes*
    - *Small tables that have high I/O rates*
  - *Avoid LRU chain maintenance and LC14 contention*
  - *Avoid unnecessary prefetch and LC24 contention*
  - *IFCID 201 & 202 will be updated to signify this condition*

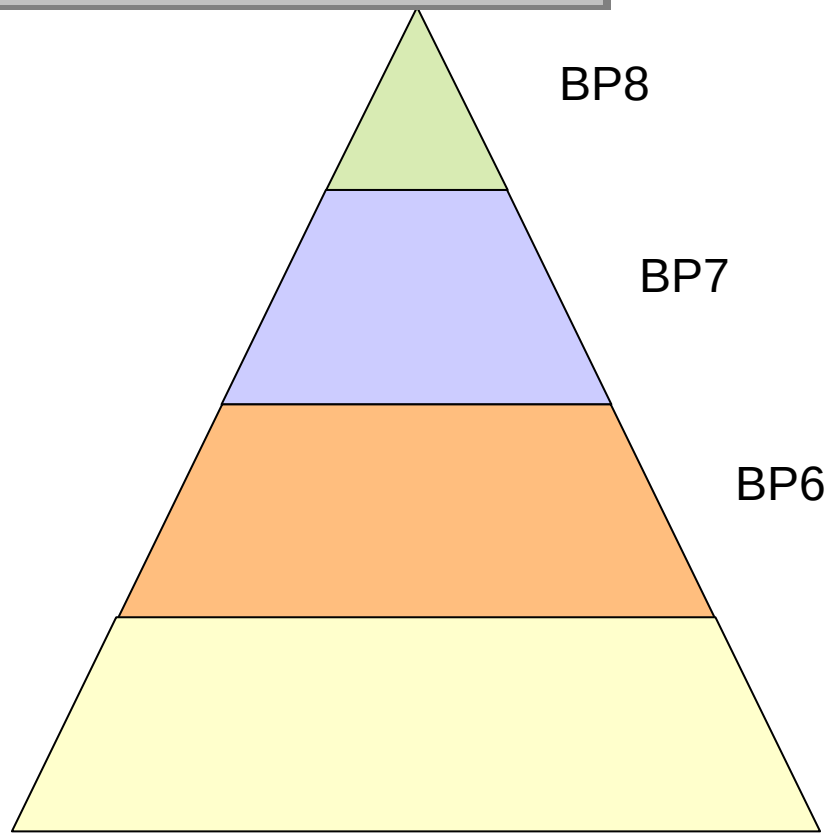
In Memory Table Spaces…

```
ALTER BUFFERPOOL BP6 PGSTEAL=NONE
ALTER BUFFERPOOL BP7 PGSTEAL=NONE
```

```
CREATE TABLESPACE TS1 IN DB1
PRIQTY 7200 USING BUFFERBOOL BP6
CREATE TABLE STATE_ABBR
  STATE_ABBR  CHAR(2)
  STATE_NAME  CHAR(20)
  CITY_NAME   CHAR(20);
```

TS1

```
SELECT STATE_ABBR
FROM STATE
WHERE CITY_NAME = ?
```



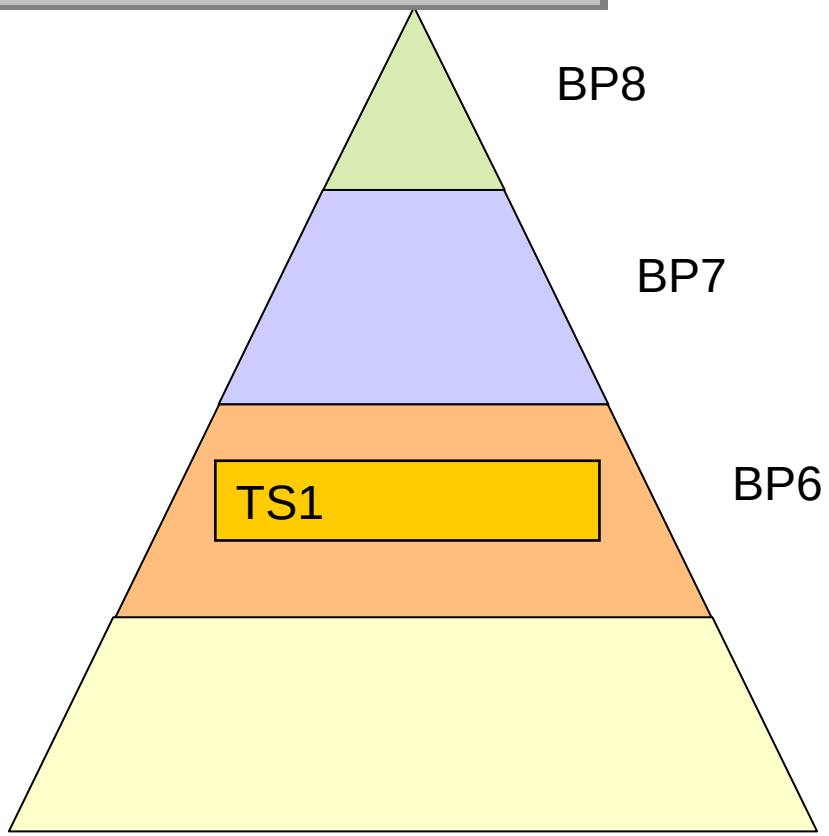
In Memory Table Spaces…

```
ALTER BUFFERPOOL BP6 PGSTEAL=NONE
ALTER BUFFERPOOL BP7 PGSTEAL=NONE
```

```
CREATE INDEX IX1 ON STATE
  (CITY_NAME,STATE_ABBR)
USING BP7
```

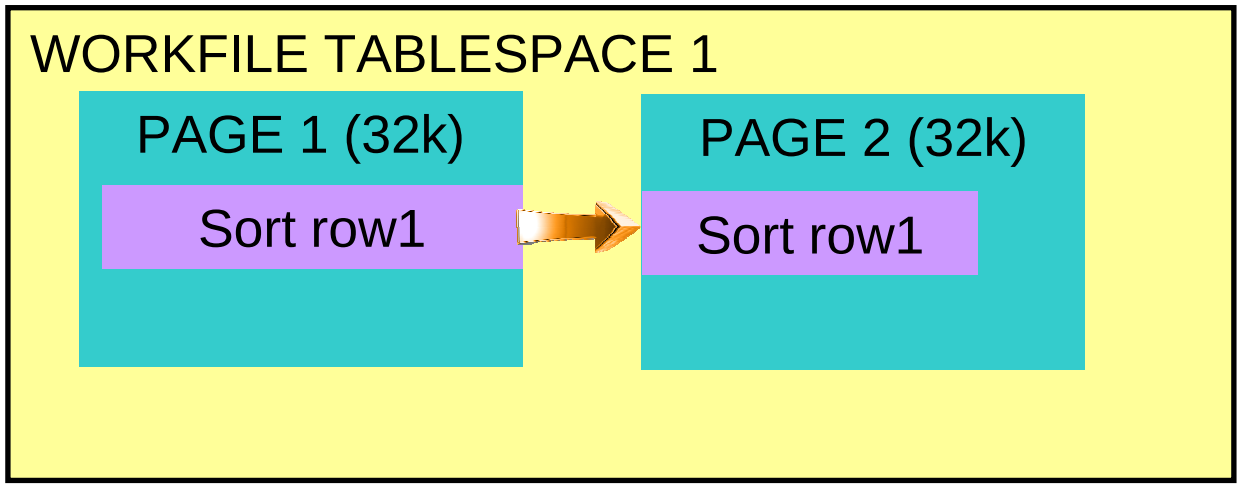
IX1

```
SELECT STATE_ABBR,STATE_NAME FROM
STATE_ABBR
WHERE STATE_ABBR = 'CA'
```



Spanned Workfiles

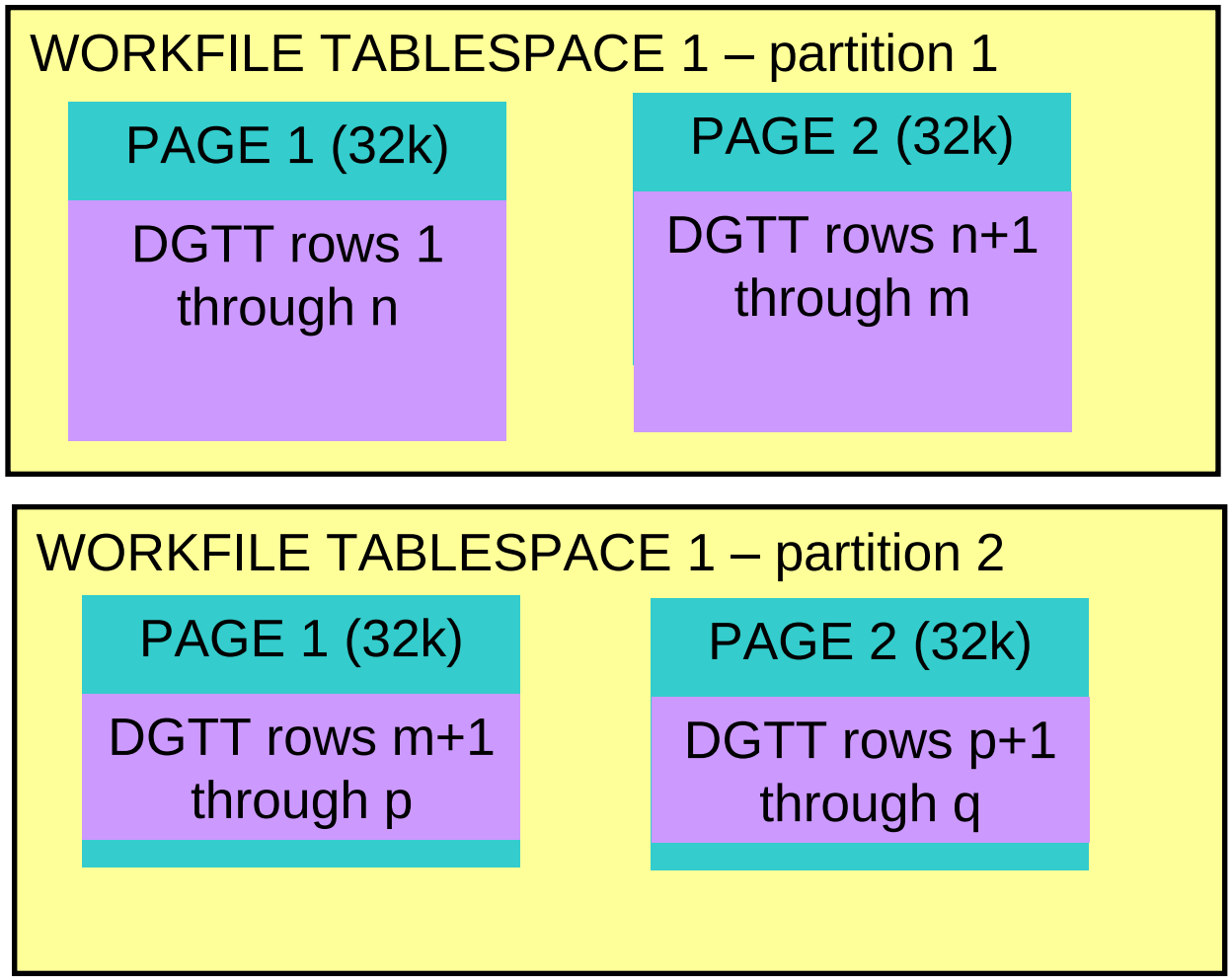
- Spanned workfile records solves problems when sorting > 32K records
- Records for joins & large sorts can now span pages
- Max record length for workfiles is increased to 65,529 bytes
- Max length of sort key is increased from 16,000 to 32,704 bytes
- Alleviate -670 when length of join is larger than max page size in workfile db
- NFM Only



DGTT changes …

- V9 Consolidated temp data base and work file data base
  - DGTT and other work file usage (sort work) share the work file database
- APARS/PTFs introduced to alleviate contention issues
  - Separation by workload
    - Secondary Quantity = 0 for Sort Work
    - Secondary Quantity >0 for DGTT
  - See http://www.ibm.com/support/docview.wss?uid=isg1II14587 for details
- V10 supports PBG table spaces in the work file
  - DGTT can get beyond the 64GB limitation
  - MAXPARTITIONS allowed for table spaces in work file, as well as DSSIZE and NUMPARTS (CREATE ONLY / no ALTER)
  - SYSTABLESPACESTATS will record work file data at the partition level for PBG tables spaces
  - New partitions will remain through DB2 restart

DGTT changes



You can define the workfiles as PBGs so that the DGTT can grow larger than 64 GB in one tablespace.

There are workarounds in v9 to simulate this behavior.

1 MB Frame Size

- **DB2 takes advantage of the new 1 MB frame size on the z10 and later**
- **Must specify PGFIX=YES to get 1 MB frame size**
- **Must be backed by real storage**
- **Must allocate space above the bar with LFAREA parm in IEASYSxx in Parmlib**
- **Potential for significant performance improvements**
- **Using the 1 MB frame size will enable efficiencies in the hardware**
- **See OA34024 for information about LFAREA sizing**
- DISPLAY VIRTSTOR,LFAREA
  - To see if LFAREA has been allocated & High Water Mark information
  - Example display shows no 1MB frames available

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF SYSLOG   7962.103 MVSA MVSA 05/03/2012 13W         COMMAND ISSUED
COMMAND INPUT ===> /DISPLAY VIRTSTOR,LFAREA                 SCROLL ===> CSR
RESPONSE=DEMOMVS
 IAR019I  06.52.50 DISPLAY VIRTSTOR 184
  SOURCE = DEFAULT
  TOTAL LFAREA = 0M
  LFAREA AVAILABLE = 0M
  LFAREA ALLOCATED (1M) = 0M
  LFAREA ALLOCATED (4K) = 0M
  MAX LFAREA ALLOCATED (1M) = 0M
  MAX LFAREA ALLOCATED (4K) = 0M
```

1 MB Frame Size

- When 1 MB Page Frames are in use and none available
  - Reverts to 4K pages
  - No longer request 1 MB pages and pages in use remain mixed
- DISPLAY BUFFERPOOL(BP1) SERVICE=4
  - DSNB999I - how many 1MB size page frames are in use

```
DSNB999I  DSNT DSNB1DBP  SERVICE( 4 )OUTPUT
DSNB999I  DSNT 4K PAGES 246
DSNB999I  DSNT 1M PAGES 0
DSN9022I  DSNT DSNB1CMD '-DISPLAY BUFFERPOOL' NORMAL COMPLETION
***
```

System latching changes

- Log latch reduction
  - Hold log latch for bare minimum of time
  - reduced Latch Class 19 contention
  - reduced other latch classes
  - Will help with both data sharing and non data sharing
- LRSN in data sharing does not have to be unique within a page. DSH
  - Major performance improvement for data sharing environments

On Demand Buffer Pool storage

- This gives you the ability to grow your buffer pool over time as opposed to allocate it all at first usage
- Maximum size will be VPSIZE
- This can be helpful to large buffer pools, where the pool will not grow to the maximum size over the length of time until DB2 is recycled.
- For Index only query, no data pages will be allocated in the buffer pool
  - Only index pages will be allocated
  - V9 – Index only on non padded indexes on varchar columns
  - v10 – Additional columns in index can make more queries index only

More Storage Management

- MAXSPACE for DUMPS
  - Prior to DB2 10 8G was considered OK
  - DB2 10 12G to 16G should be considered
  - Avoid very long dump capture times and partial dumps
- 64 Bit Common
  - 6GB for each DB2 sub system
  - Lightly utilized
- Page fixing the log buffers can reduce MSTR SRB time by up to 30% in CM
  - Consider that this needs to be backed by real storage (OUTBUFF)

Virtual Storage – IBM DB2 Tools Support

- Omegamon XE for DB2 Performance Expert
  - More Storage Above the Bar
  - Workfile Enhancements
  - Additional support for 64 bit runtime
  - 1 MB Page size
  - System Latching changes
  - On Demand Buffer Pool storage
  - Log Buffer Page Fix
- DB2 Buffer Pool Analyzer
  - Buffer Pool Objects

Data Sharing

- Coexistence
- Member Consolidation
- LC19 reduction
- Buffer Pool Scan Avoidance
- MEMBER CLUSTER
- Local Subgroup Attach
- ACCESS DB
- Index Split Improvements
- Restart Improvements
- Log Buffer Page Fix
- ALTER TABLESPACE / INDEXSPACE enhancement
- Optimization when removing GBP Dependency
- New ZParms

Data Sharing Improvements...

- Coexistence possible with V8 (Mode CM8) or V9 (Mode CM9)
  - Private Protocol only support with V8 / V9 members during CM8 / CM9
- Possible Member Consolidation
  - Reduction in virtual storage below the bar
    - CTHREAD increases
  - Reduced resource contention
    - UTSERIAL lock removed
    - LC19 reduction with LRSN “spin” enhancement
- LC19 reduction
  - LRSNs do not have to be unique for INSERTs
    - Ensure PM51093 is applied before NFM (Red Alert) for recovery and LPL/GRECP
  - Targeting Multirow INSERT operations
  - Internal measurements have shown a 70 – 130% improvement in throughput for workloads constrained by LC19
- MEMBER CLUSTER for UTS

Data Sharing Improvements...

- Subgroup Attach for Local Connections
  - Available in V9 via Usermod
  - Defined in the IEFSSNxx parameter lib member
  - Same naming rules as Group Attach names
    - Should not be the same as existing DB2 member (not enforced)
  - The Subgroup Attach Name can only belong to one Group Attach definition
  - The Subgroup Attach Name must have a Group Attach Name
  - The Subgroup Attach Name cannot have the same name as the Group Attach Name
  - A DB2 member may only belong to at most one subgroup attach.
  - A DB2 member does not need to belong to a subgroup attach
    - DIS GROUP DETAIL modified to show subgroups.

```
SUBSYS SUBNAME(ssname) INITRTN(DSN3INI)
INITPARM('DSN3EPX,cmd-prefix<,scope<,GRP<,SGRP>>>')
```

Data Sharing Connection Queue Management

- Connections waiting for a DBAT could be redirected to another member if notified that the current member cannot service the request in a timely manner
- PM43293 introduced connection queue depth and wait time monitors (off by default)
  - **MAXCONQN**: Max **number** of inactive or new connection requests waiting for a DBAT
  - **MAXCONQW**: Max **wait time** for a connection request to wait for a DBAT.
  - -DIS DDF DETAIL enhanced to show settings
- Only effective for:
  - DB2 members in a data sharing group
  - Using CMTSTAT=INACTIVE
- DSNL030I with Reason Code 00D31053 (QN) or 00D31054 (QW)
- Additionally
  - Reduces DB2 system health as connections pass 80% & 90% of CONDBAT (with DSNL074I)
    - 50% & 25% of calculated heath until condition relieved (with DSNL075I)
  - For KEEPDYNAMIC threads exceeding 1 hour or inactive for 20 minutes and inactive
    - Now produces messaging to indicate termination
    - DSNL027I with 00D3003E or 00D3003F

Data Sharing Improvements...

DDF

- Restart Lite for DDF Indoubt URs
- ACCESS DB command wildcarding
  - Also in V9 via PK80925
- Index split recording
  - IFCID 359 reports index splits when performance class 4 is active
  - Also reports if index was group buffer pool dependent when the split occurred
- Group restart avoidance of lock resolution problems
  - Occasional lock resolution problems have been resolved by a lock structure rebuild
  - DB2 10 IRLM detects waits and rebuilds the structure automatically
- Page fixing the log buffers (not specific to data sharing)

VST

Data Sharing Improvements...

- Table and Index Spaces no longer need to be stopped to change the buffer pool
  - Pre-DB2 10 / Data Sharing, the table space / index space needed to be stopped before the ALTER statement was issued
- Optimization when removing Group Buffer Pool Dependency
  - IRLM lock timeouts could occur during CF “delete name” requests when removing inter-DB2 Read Write interest
  - Typically associated with a DB2 member physically separated from the CF
  - Process is changed to only delete CF data entries
    - Avoiding potentially massive amounts XI signaling
    - CF “delete name” happens more quickly
    - Avoid impact to transactions
    - The directory entry will be deregistered when the local buffer pool pages are reclaimed via a “lazy” cleanup process

Data Sharing Improvements...

- DSN_STATEMENT_CACHE table has a GROUP_MEMBER column
  - EXPLAIN STMTCACHE output can distinguish between cache of different members
- ZParm DEL_CFSTRUCTS_ON_RESTART
  - Old CF structures have caused DR test difficulties
  - Defaults to NO
  - If YES
    - Attempts to delete CF structures on group restart
    - Recommended to be the same across members
    - Could be set to YES in the DR ZParm
    - Only the first member to access the structures will trigger the delete
  - Added via PM28925
  - IRLM PM31807 needed to delete Lock Structure
- Red Alert PM51655 should be reviewed and applied
  - Reverses DB2 10 READ CASTOUT CLASS changes
- PM79520 for DB2 10 (any mode) timing window data loss (3/3013)

Data Sharing – IBM DB2 Tools Support

- Omegamon XE for DB2 Performance Expert, Query Monitor
  - Buffer Pool Scan Avoidance
  - MEMBER CLUSTER
  - Index Split Improvements
  - New ZParms
- DB2 Admin Tool
  - ALTER TABLESPACE / INDEXSPACE enhancement
  - New ZParms
- DB2 Cloning Tool
  - Add new members

Serviceability and Instrumentation

- Active Log Data Sets
- Autonomic Checkpoint
- Enhanced Monitoring Support
  - Statement Level
  - System Level
- SMF Management
- zIIP Expansion
- WLM Stored Procedure Address Space Minimum

Active Log Data Sets

- Ability to add new active log data sets without having to recycle DB2.
  - NEWLOG and COPY keywords added to the –SET LOG command.
    *SET LOG NEWLOG(*dsn*) COPY(*log-copy*)*
  - Changes are pervasive.
- Data set must be defined with IDCAMS before issuing command.
- Recommendations:
  - Format data set with DSNJLOGF utility before issuing command.
  - Add both copy 1 and 2 for dual logging.
- New DB2 messages:
  - DSNJ363I:
    COPY*log-copy* LOG DATA SET *dsn* ADDED TO THE ACTIVE LOG INVENTORY
  - DSNJ364I: NEWLOG OPERATION FAILED: *reason*

Autonomic Checkpoint

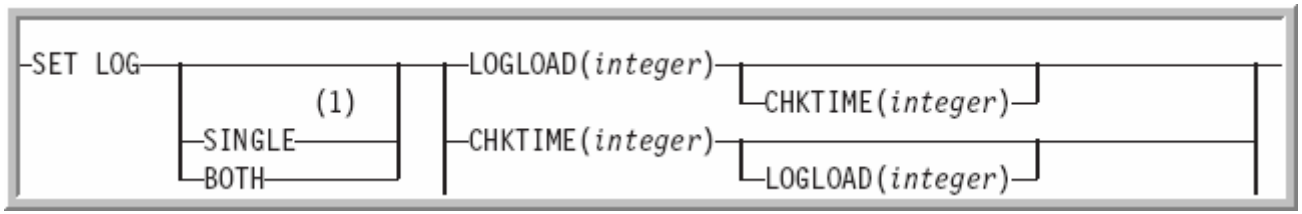
- Currently DB2 allows checkpoints to be scheduled based on either number of log records created or a fixed time interval.
- DB2 10 enhances checkpoints thresholds to be scheduled on either or both.
  - Provides ability to use log records when subsystem is busy, and time interval when subsystem is relatively inactive.

Autonomic Checkpoint…

- ZParm changes:
  - New **CHKTYPE** defines threshold method being used.
    - **SINGLE** – either number of log records or time interval
    - **BOTH** – both methods used
  - New **CHKLOGR** defines log record threshold, when CHKTYPE = BOTH. Otherwise set to NOTUSED.
  - New **CHKMINS** defines time interval threshold, when CHKTYPE = BOTH. Otherwise set to NOTUSED.
  - Existing **CHKFREQ** defines log record or time interval threshold, when CHKTYPE = SINGLE. Otherwise set to NOTUSED.
  - **Note:** The Panel choices that drive these parameter selections are
    - **LOGRECS**
    - **MINUTES**
    - **BOTH**

Autonomic Checkpoint…

- SET LOG command changes:
  - Provides ability to temporarily make changes to checkpoint method being used and values.
    - Values are reset to values specified in ZParm when DB2 recycled.
  - Parameters SINGLE and BOTH added
  - Syntax:



    - If SINGLE is specified, then CHKTIME or LOGLOAD must be specified.
    - If BOTH is specified, and either (or both) CHKTIME or LOGLOAD is not specified, then previously set values or defaults are used.
      - *CHKTIME default = 5*
      - *LOGLOAD default = 500000*
  - Use **DISPLAY LOG** command to display current checkpoint method being used and values.

Enhanced Monitoring Support

- DB2 10 enhanced monitoring support is provided as two independent functions.
  - Statement level
  - System level
- At the statement level this supports performance monitoring and problem determination for both static and dynamic SQL.
  - Uses IFI to capture and externalize information for consumption by tooling.
  - Introduces a unique statement execution identifier (STMT_ID).
- At the system level this provide increased granularity of monitoring system level activities.

Enhanced Monitoring Support… Statement Level

- Problem determination enhancements at the statement level:
  - Externalize statement type (dynamic or static) and STMT_ID in existing messages, including those related to deadlock, timeout, and lock escalation.
  - Information associated with thread information (THREAD-INFO) in these messages.
  - Can be used to correlate the statement execution on the server with the client application.
- Performance monitoring enhancements at the statement level:
  - Existing trace records modified to capture:
    - statement type
    - STMT_ID
    - statement-level performance metrics
  - New trace records are introduced to provide:
    - Access to performance monitoring statistics in real time.
    - Allow tooling to retrieve monitoring data without requiring disk access.

Enhanced Monitoring Support… Statement Level

- STMT_ID:
  - Facilitate monitoring and tracing at the statement level
  - Defined at the DB2 for z/OS server
  - Captured in IFCID records for both static and dynamic SQL
    - Dynamic SQL:
      - *Available when dynamic statement cache is enabled*
      - *Matches value in STMT_ID column in DSN_STATEMENT_CACHE_TABLE*
    - Static SQL:
      - *Matches value in new STMT_ID column in SYSPACKSTMT*
      - *Packages must be rebound in DB2 10 NFM to populate column and load into package.*

```
DSNT376I  DSNT PLAN=DSNESPCS WITH  681
          CORRELATION-ID=SYS014
          CONNECTION-ID=TSO
          LUWID=USIBMNR.NDCDB206.C87B4E3A3B76=79
          THREAD-INFO=SYS014:TSO:SYS014:SYS014:DYNAMIC:43:*:*
           IS TIMED OUT. ONE HOLDER OF THE RESOURCE IS PLAN=DSNESPCS WITH
          CORRELATION-ID=DBA015
          CONNECTION-ID=TSO
          LUW-ID=USIBMNR.NDCDB206.C87B4DE8D9CD=76
          THREAD-INFO=DBA015:TSO:DBA015:DBA015:DYNAMIC:46:*:*
          ON MEMBER DSNT
```

Statement text in DSN_STATEMENT_CACHE_TABLE

Enhanced Monitoring Support… System Level

- Enhancements for monitoring system level activities:
  - Profile table facility enhanced to provide improved threshold monitoring and filtering through new fields/keywords.
    - Enhancements apply only to system level monitoring related to threads and connections, not to statement level monitoring and tuning.
  - Monitor profile extended to support monitoring of:
    - idle thread timeout
    - number of threads
    - number of connections
  - Monitor profile includes ability to filter by ROLE and client product-specific identifier (PRDID).
    - Provides a finer degree of control over the monitor profiles.
  - Allows thresholds to be enforced at a more granular level, which were previously available at the system level via ZParm.
  - Provides greater flexibility and control for allocating resources to particular clients, applications and users according to priorities or needs.
  - Apply PM64376 & PM79521 for memory management
    - Even is SPM is not used

Enhanced Monitoring Support… System Level

- Monitoring system level activities related to threads and connections:
  - Following profile tables must be created:
    - DSN_PROFILE_TABLE
    - DSN_PROFILE_HISTORY
    - DSN_PROFILE_ATTRIBUTES
    - DSN_PROFILE_ATTRIBUTES_HISTORY
    - Some may have been created in DB2 9 or by optimization tools, and may require modifications.
    - Tables will be created by DSNTIJSG job.
  - Following tasks must be completed.
    1. Specify activity to monitor by creating a profile in DSN_PROFILE_TABLE.
       - *Filtering categories:*
         - *IPADDR, PRDID, ROLE/AUTHID, COLLID/PKGNAME, Client accounting*

Enhanced Monitoring Support… System Level

- Monitoring system level activities related to threads and connections: (continued)
  2. Define monitoring by inserting a row into DSN_PROFILE_ATTRIBUTES.
     - *Three KEYWORDS with two ATTRIBUTE columns:*
       - ***MONITOR THREADS****: Number of concurrent active threads… cannot exceed MAX_REMOTE_ACTIVE (MAXDBAT).*
       - ***MONITOR CONNECTIONS****: Number of active and inactive remote connections using TCPIP… cannot exceed MAX_REMOTE_CONNECTED (CONDBAT).*
       - ***MONITOR IDLE THREADS****: Time (in seconds) that an active server thread should be allowed to remain idle... Adjusts IDLE_THREAD_TIMEOUT (IDTHOIN).*
       - ***ATTRIBUTE1:*** *Specify WARNING or EXCEPTION and can include suffix of "_DIAGLEVEL1" or "_DIAGLEVEL2".*
       - ***ATTRIBUTE2:*** *Specify threshold value.*

Enhanced Monitoring Support… System Level

- Monitoring system level activities related to threads and connections: (continued)
  3. Load or reload the profile tables into memory by issuing the START PROFILE command.
  4. Stop monitoring by:
     - *Disabling monitoring function for a specific profile by:*
       - *Deleting the appropriate row or changing the PROFILE_ENABLED column to N for the row in DSN_PROFILE_TABLE.*
       - *Reload the profile tables by issuing the START PROFILE command.*
     - *Disabling all monitoring by issuing the STOP PROFILE command.*

Enhanced Monitoring Support… System Level

- Monitoring system level activities related to threads and connections: (continued)
  - Additional details on ATTRIBUTE1:
    - WARNING: Processing continues… console message issued at most every 5 minutes.
    - EXCEPTION:
      - *MONITOR THREADS: Thread queued and suspended, and may fail thread depending on filtering scope.*
      - *MONITOR CONNECTIONS: Message issued and new connections are rejected.*
      - *MONITOR IDLE THREADS: Messages issued and terminates thread.*
    - DIAGLEVEL1 provides minimal information in messages. Default when only WARNING or EXCEPTION specified.
    - DIAGLEVEL2 includes profile ID and reason code in message text.
  - New IFCID 402 introduced to Stats Class 4
    - Written when a profile warning or exception condition occurs.
  - Additional details: Monitoring threads and connections by using profiles

Statistics Management

- SMF Compression
  - Via the new **SMFCOMP** ZParm
  - Accounting records may compress 80-90%
  - CPU Overhead ~1%
  - When **ACCUMACC** ZParm is also set (default to 10) can compress up to 99%
    - DB2 10 rollup accounting provides more detail
  - PM27872
    - Adds job DSNTSMFD to decompress SMF records
- Statistics interval changes
  - STATIME default reduced to 1 minute
    - IFCIDs 0105, 0106, 0199, and 0365
  - IFCIDs 0001, 0002, 0202, 0217, 0225, and 0230 are no longer controlled by STATIME
    - Corresponding trace records are written at fixed, one-minute intervals

zIIP Expansion & WLM

- zIIP Expansion
  - Prefetch & Deferred Write processes
    - PM30468 for proper reporting
  - RUNSTATS
  - PM37622 for REORG UNLOAD enablement (retrofitted to DB2 9)
- WLM Minimum # of Address Spaces
  - PM27097 adds MNSPAS parameter to the WLM startup JCL
  - Starts this number of address spaces initially
  - Available to DB2 9 as well
- -DISPLAY THREAD
  - Now displays WLM classification information via message V482
    - Service Class Name
    - Service Class Period Number
    - Importance Level of the Period
    - Performance Index of the Service Class Period
  - PM45572 & OA35822

UTL

SQL Access to the Directory

- PM35190 (enabling APAR PM42331)
  - NFM
  - If PTF is applied after NFM
    - **CATMAINT UPDATE UNLDDN PM35190** enables this access
  - Adds SYSLGRNX, SYSUTIL, & SYSUTILX definitions into the Catalog
- SELECT allowed on
  - SYSLGRNX
  - SYSUTIL
  - SYSUTILX
- DB2 10 NFM
  - Not enabled for new installs or systems in 10 NFM before applying this maintenance
  - See APAR for CATMAINT to enable

Serviceability and Instrumentation – IBM DB2 Tools Support

- OMPE, Buffer Pool Analyzer
  - Autonomic Checkpoint
  - Enhanced Monitoring Support
    - Statement Level
    - System Level
  - SMF Management
  - zIIP Expansion
  - WLM Stored Procedure Address Space Minimum
- DB2 Query Monitor, QWT
  - Statement level monitoring
- DB2 SQL PA, QWT
  - Statement level tuning
  - Index advice
- DB2 pureQuery
  - Statement performance improvement
- DB2 Admin Tool, OMPE
  - ZParm Changes

Summary of DB2 10 Business Value

- Integrated XML / Relational Support
- Extending the lead with Security enhancements
- Performance
- Manageability
- Extended Distributed Computing Performance
- Virtual Storage Management
- Data Warehousing
- Time Travel Queries with Temporal (Versioned) Data support.