DB2 10 for z/OS
Technical Overview

John Campbell
IBM Distinguished Engineer
Silicon Valley Lab
DB2 for z/OS Technical Strategy

RAS (reliability, availability, serviceability), scalability and performance. Technology leadership: extend the lead
- Parallel Sysplex: the best scale-out and high availability solution in the industry
- Workload consolidation: System z is the ultimate consolidation platform
- Tight integration between DB2 and the System z hardware and z/OS operating system
- Advanced solutions for compliance with data security and privacy regulations
- 24x7 operations: eliminate all causes of outages
- Storage and CPU optimization, including specialty engines

Application enablement
- Application portability and DB2 family compatibility
- Advanced SQL, XML capability.
- Advanced query processing capabilities

Systems management and automation
- DB technology that can handle larger workloads without requiring more people
- Easy to use interfaces that are clear and intuitive
- Advanced autonics to make the system more self-managing and self-tuning
DB2 for z/OS
The most robust and cost effective data server

DB2
- Deep synergy with System z
- HW Compression
- Consolidation

DB2 9
- 20%-30% Utility CPU savings
- Compress indexes, save 50% disk
- More CPU on specialty engines

DB2 10
- Save 5-10% or more cpu batch/transactions
- On-the-fly data Compression
- Temporal data support
- Skip-level migration

Efficiency
- Unmatched availability
- Unparalleled security
- Industry leading reliability

Resilience
- Flexible context and role security
- Expanded online schema changes
- Volume level backup & recovery

Growth
- Near-linear scalability
- Optimized for SOA
- Flexible development
- Warehousing capabilities

- Seamless integration of XML and relational
- Improved SQL
- Partition by growth
- OLAP expressions

- Enhanced query parallelism
- More SQL compatibility
- Improved pureXML and SQL PL

Beta started: Mar 12, 2010
DB2 10 for z/OS: Faster Answers at Lower Cost

Better performance and reduced CPU use

Up to 10x the number of concurrent users
- Optimized use of System z processor and memory resources
- Enhancing proven performance of DB2 on System z

Accelerated access to most commonly used data
- Direct row access eliminates multi-step indexing

Time travel query and analysis
- Application transparent access to past and future data

5-10% or more reduction in CPU use
Summary: DB2 10

Autonomics
- Compress on the fly during INSERT
- Auto-stats
- Default SAP settings for DB2
- Access path stability and hints enhancements
- Automatic checkpoint interval
- Automated installation, configuration and activation of DB2 supplied stored procedures
- Exploit data set FlashCopy in COPY & inline copy
- Inline image copies for COPY YES indexes
- Reduce need for reorganizations for indices
- Access path manageability for dynamic SQL
- ALTER TABLE ROTATE n TO LAST

Performance
- Hash access path
- Parallel index update at insert
- Numerous optimizer enhancements
- Insert performance improvements

Faster single row retrievals
- Inline LOBs
- LOB streaming between DDF and rest of DB2
- Faster fetch and insert, lower virtual storage consumption
- DEFINE NO for LOBs (and XML)
- Enabling MEMBER CLUSTER for UTS
- Query parallelism enhancements: lifting restrictions
- Option to avoid index entry creation for NULL value (post GA)
- Index include columns
- Buffer pool enhancements

Availability
- More online schema changes for table spaces, tables and indexes via online REORG
- Online REORG for LOB
- Online add log
- Automatically delete CF structures before/during first DB2 restart
- Pre-emptable Rollback

Portability
- Allow non-NULL default values for inline LOBs
- Loading and unloading tables with LOBs
- ‘Last committed’ locking semantics
- Easier SQL paging through result sets
- CREATE TABLE clauses for tablespace attributes

Scalability
- Full 64-bit runtime support
- Reducing internal latch contention
- Workfile spanned records, PBG support, and in-memory enhancements

Security
- More granular DBA privileges

© 2010 IBM Corporation
DB2 10 for z/OS Beta

Announce : Feb. 9
Shipped : Mar. 12

Largest Beta Ever
- Strong customer demand
- 24 WW customers/cross industry
- Extended beta started 3Q
- 73 parties in vendor program

Customer Focus Areas
- Stability / regression
- Out-of-box performance
- Additional performance
- New function

So Far.....
- Most production readiness testing in history
- Solid stability
- Positive feedback on performance and vstor
Highlights of Beta Results

Good Results

• Virtual storage constraint relief
• Insert performance
• HASH Access good for sweet spot
• Complex queries
• INLINE LOBS (SLOBs)
• Latch contention reduction
• REORG, other utilities
• Migration (after some challenges early in the program)

Mixed Results

$ OLTP, but mostly good
• BIND performance
• Online schema change
• Access path lockdown

Some customer performance data is still under analysis
DB2 10 for z/OS: Out-of-the-Box Savings

CPU reductions for transactions, queries, and batch

- Out-of-the-box CPU reductions of 5-10% for traditional workloads
- Up to additional 10% CPU savings using new functions or avoiding constraints
- Out-of-the box CPU reductions of up to 20% for new workloads

Scales with less complexity and cost

- 5-10x more concurrent users – up to 20,000 per subsystem
- Significant scale-up capabilities in addition to existing scale-out support
- Consolidate to fewer LPARs and subsystems

Improved operational efficiencies and lower administration cost

- Automatic diagnostics, tuning, and compression

Even better performance

- Elapsed time improvement for small LOBS and Complex Queries
DB2 & IBM zIIP Add Value to Database Work

Portions of the following DB2 for z/OS V8, DB2 9 and 10 workloads may benefit from zIIP (DB2 9 in blue, DB2 10 in green)*:

1 – DRDA over TCP/IP connections
   • DB2 9 for z/OS Remote native SQL procedures
   • DB2 9 XML parsing via DRDA to fully utilize zIIP
   • PM12256 increased portion of DRDA redirected to zIIPs to 60%
     – Improved performance via reduced processor switching

2 - Requests that use parallel queries
   DB2 9 higher percentage of parallel queries zIIP eligible
   DB2 10 still more queries eligible for parallelism

3 - DB2 Utilities LOAD, REORG & REBUILD
   • DB2 utility functions used to maintain index structure
   • DB2 10 RUNSTATS

4 - DB2 10 Buffer Pool Prefetch

* zIIP allows a program working with z/OS to have all or a portion of its enclave Service Request Block (SRB) work directed to zIIP. Above types of DB2 work are those running in enclave SRBs, of which portions can be sent to zIIP.
Performance Enhancements with Few Changes (CM)

- SQL runtime improved efficiency
- Address space, memory changes to 64 bit, some REBINDs
- Faster single row retrievals via open / fetch / close chaining
- Distributed thread reuse High Performance DBATs
  - RELEASE(DEALLOCATE) can be more aggressively used with v10 vstor relief
    - IBM measurements show up to 10-20% cpu savings
  - DDF enforcement of RELEASE(COMMIT) is removed
  - KEEPDYNAMIC=YES will not get benefit from this enhancement
- DB2 9 utility enhancements in CM8
- Parallel index update at insert
- Workfile in-memory enhancements
- Smart index prefetching
- JCC Type2 and ODBC for z/OS performance improvements
- Buffer pool enhancements
  - Utilize z10 1MB page size
  - “Fully in memory” option (ALTER BUFFERPOOL)
Parallel Index Update for INSERT

- Beta Customer Test Results for 2M+ INSERTs into table with 9 indexes
  - All jobs in DB2 10 CM9 have less CPU and less elapsed time
    - Would expect same results for CM8
  - the CPU improvement is between 15-30 percent
  - and the Elapsed Time improvement is between 30-43 percent.
  - Improvements depend on the size of the Bufferpool
    - Larger savings observed with smaller BPs
  - Some increase in DBM1 SRB time, but this is zIIP eligible

- Another beta customer reported 84% speedup for INSERTs into table with 5 indexes
# Some Initial Performance Results from Beta Customers

## INSERT Performance

**Customer1**

Cobol Batch concurrent insert (Seq i/o reduced) 14.5% improvement in CM9 without bind, 25% after rebind and NFM (access path change included).

17% CPU reduction due to Include index - Inserted 2 million, fetched 1 million.

**Customer2**

Ran a batch job with 1000 Inserts and a SQLJ application comparing V9 vs V10. Observed up to 20% reduction in CPU class 1 and class 2 time.

**Customer3**

With multi-row insert, observed a 33% improvement vs. V9, 4x improvement vs V8. This is due to LRSN spin avoidance for inserts to the same page.

**Customer4**

Heavy concurrent insert: CPU reduction in CM mode, Result 36% reduction. Higher improvement in inserting in key sequential from multiple threads Result 81% reduction. LC19 reduction average suspends Result 51% reduction
Smart Index Prefetching

- As data is inserted, updated, deleted in an indexed table, index leaf pages become non-sequential.

- **Problem:** performance of queries accessing such tables will not be optimal.
  - Prefetching allows reading of index leaf pages asynchronously in order to speed up performance.
  - DB2 detects sequential index leaf page reads and triggers leaf page prefetching.
  - Non-sequential index leaf pages require synchronous reads.
  - To solve this issue (synchronous leaf page i/o) today, an index reorg is required.

- **Solution:** Smart Index Prefetch enables prefetching for non-sequential leaf pages.
  - Avoids need to reorg the index for clustering
  - Collect the upcoming index leaf page numbers from the non-leaf level of the index to asynchronously prefetch them
  - Ensure we do not prefetch beyond our scan endpoint (stop key)
  - DB2 recognizes when sequential detection is not sufficient and dynamically switches to Smart Index Prefetch on the fly during query execution
  - No rebind necessary
Performance Enhancements requiring REBIND (CM)

- Most access path enhancements
- SQL paging performance enhancements
  - Single index access for complex OR predicates
- IN list performance
  - Optimized Stage1 processing (single or multiple IN lists)
  - Matching index scan on multiple IN lists
- Query parallelism improvements
- More stage 2 predicates can be pushed down to stage 1
- More aggressive merge of views and table expressions
  - Avoid materialization of views
- REBIND enables further SQL runtime improvements
- If migrate from V8, get new RUNSTATS before mass rebind
Performance Enhancements requiring NFM

- Efficient caching of dynamic SQL statements with literals
- Most utility enhancements
- LOB streaming between DDF and rest of DB2
- Faster fetch and insert, lower virtual storage consumption
- SQL Procedure Language performance improvements
- Workfile spanned records, PBG
- Some insert improvement for UTS
- Local JDBC (Type2) and ODBC application performance
  - Limited block fetch, LOB progressive streaming, progressive CLOSE now available for JCC type2 and ODBC z/OS drivers
- Solid State Disk monitoring
Performance Enhancements requiring NFM + DBA work

- Hash access path Alter + Reorg + rebind to activate
- Index include columns Alter + Rebuild + rebind to activate
- Inline LOBs Alter (need UTS and RRF)
  - Index on expression now possible for LOB columns
  - Important for spatial performance
  - LOAD/UNLOAD performance improvements
  - LOB compression for inline portion
- MEMBER CLUSTER for UTS
- DEFINE NO for LOB and XML columns
- Option to avoid index entry creation for NULL value (planned post GA)
Index to Data Access Path

INDEX on AcctID

Select Balance From Accounts WHERE acctID = 17

Traverse down Index Tree
- Typically non-leaf portion of tree in the bufferpool
- Leaf Portion of the tree requires I/O
- Requires searching pages at each level of the index

Access the Data Page
- Typically requires another I/O

For a 5 Level Index
- 6 GETP/RELPs, 2 I/O's, and 5 index page searches

Accounts Table

= Page in Bufferpool
= Page Read from Disk
Hash Access provides the ability to directly locate a row in a table without having to use an index.

- Single GETP/RELP in most cases
- 1 Synch I/Os in common case
  - 0 if In Memory Table
- Greatly reduced Search CPU expense
Queries against Hashed Tables

- **Hash Access Path**
  - Great for Equality and IN predicates
  - Can’t do range scans
- Secondary indexes can be defined for Range Scans
- Table Space Scans still supported
- Hash Access can be used to enforce Primary Key and Unique constraints

**Index Lookup/Scan**
Select Balance From Accounts WHERE CustID = 1717

**Hash Access**
Select Balance From Accounts WHERE AcctID = 17

Select CustID From Accounts WHERE Balance > 100000
What happens at insert time

- We compute a ‘Random’ Page to insert on
  - If the Page is Full, the new entry becomes an overflow
  - The Page is ‘Randomly’ chosen, so there will be statistical variation
- Suppose 20 Rows fit in a Page
- What if we hash 20 Million rows into 1 Million pages
  - On Average, 20 rows will hash to each page
Hashing Summary

- **Provides fast, direct location of most rows**
  - Reduces I/O and CPU in most cases
  - Can replace an existing Primary or Unique Key Index
    - Faster Insertion/Deletion when removing an index
- **Size of Fixed Size Hash Area is important**
  - Too small and performance degrades
  - Too large and space is wasted
- **DB2 helps you manage the size**
  - REORG AUTOESTSPACE YES
  - RTS tracks the number of overflowed entries
Full 64-bit Runtime – 10x more DB2 threads

**Challenge**

Virtual storage in DBM1 address space restricted to 2 GB
- Restricts maximum number of SAP work processes served by single DB2 subsystem

**Solution**

64-bit support for all relevant storage areas in DBM1
- Particularly, DB2 threads live above the 2 GB bar

Goal: 5 to 10 times more active threads per DB2 subsystem

A few storage structures related to interfacing with z/OS still 31 bit
- Like data set management

Stack storage space also 64-bit enabled
DB2 10 beta test results with SAP

- **SAP SD benchmark**
  - Run with 2504 concurrent active threads on DB2 10 NFM single member with excellent response time
    - DB2 9 limit was about 400 threads
- **19% ITR improvement for SAP SD workload in POK with 12 CPs on z10**
  - Additional 3 to 4% savings due to REOPT(ONCE) improvement in DB2 10
Running a Large Number of Threads

**Today**

<table>
<thead>
<tr>
<th>LPAR1</th>
<th>LPAR2</th>
<th>LPAR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2A (500 thds)</td>
<td>DB2B (500 thds)</td>
<td>DB2C (500 thds)</td>
</tr>
<tr>
<td>DB2D (500 thds)</td>
<td>DB2E (500 thds)</td>
<td>DB2F (500 thds)</td>
</tr>
</tbody>
</table>

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

**DB2 10**

<table>
<thead>
<tr>
<th>LPAR1</th>
<th>LPAR2</th>
<th>LPAR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2A (2500 thds)</td>
<td>DB2B (2500 thds)</td>
<td>DB2C (2500 thds)</td>
</tr>
</tbody>
</table>

- Easier growth, lower costs, easier management
- More threads per DB2 image
- More efficient use of large n-ways
- Potential for fewer members / LPARs
- Rule of thumb: save ½% CPU for each member reduced, more on memory
- Data sharing and Parallel Sysplex still required for very high availability and scale
Other System Scaling Improvements

- **Other bottlenecks can emerge in extremely heavy workloads**
  - several improvements planned to reduce latching and other system serialization contention
  - new option to for readers to avoid waiting for inserters
  - eliminate UTSEIAL lock contention for utilities
  - Use 64-bit common storage to avoid ECSA constraints

- **Concurrent DDL/BIND/Prepare processes can contend with one another**
  - restructure parts of DB2 catalog to avoid the contention

- **SPT01 64GB limit can be a constraint, especially if package stability is enabled**
  - Allow many more packages by using LOBs

- **Improved accounting rollup, compress SMF option**
  - Reduced SMF data volume

- **Lower overhead for very large buffer pools**
DB2 10 Catalog and Directory Improvements

- Remove links and enable row level locking on key catalog tables
  - Improved concurrency for BIND, PREPARE, and DDL
  - DSN1CHKR no longer needed in DB2 10 NFM

- Move long strings such as SQL statements and package binaries to LOBs
  - Removes max size constraint for SPT01 and others
  - Easier to query SQL statements from catalog

- Online REORG for all catalog and directory table spaces
- Allow SELECT from SYSLGRNX
- Easier management: DB2 managed and SMS controlled
Data Sharing Improvements

- ACCESS DATABASE command wildcarding support - V9 PK80925
- Sub-group attach (V9 usermod)
- BP scan avoidance
- Delete data sharing member (planned for post GA)
  - Offline utility for “deactivate”, “reactivate”, “destroy”
- MEMBER CLUSTER support for UTS
- DDF Restart Light enhancements: Handle DDF indoubt URs
- Online DDF changes
- Auto rebuild CF lock structure on long IRLM waits during restart
  - Can avoid group-wide shutdowns
- LRSN spin avoidance for inserts to the same page (e.g. Multi Row Insert)
- IFCID 359 for index split
- New zparm to force deletion of CF structures on group start (e.g. DR testing)
- Faster GBP DELETE_NAME processing
  - Usually at pseudo close or physical close time when pageset/part becomes non GBP-dependent
  - Avoid sending XI signals by deleting data only
  - Avoid potential lock timeout conditions when there are lots of directory entries for an object
Online Schema Problem: Some are still drop and recreate

- **Change of table - or index space attributes require an outage**
- **Change of table space attributes**
  - Unload data
  - Drop table space
  - Recreate table space, tables, indexes, views
  - Re-establish authorization & RI
  - Reload data
- **Change of index space attributes**
  - Alter index
    - Index placed in RBDP
  - Rebuild index
- **Undo of DDL changes**
  - Same as above
Online Schema - Solution

- Execute ALTER statement
- Changes are cached & materialized by next REORG
  - SHRLEVEL REFERENCE|CHANGE
- Undo of DDL changes if not materialized
  - ALTER TABLESPACE… DROP PENDING CHANGES
  - All pending changes are removed
- Undo of DDL changes if materialized
  - Perform compensating ALTER & schedule REORG
    - Assumes no dependencies on prior ALTER have evolved
Online Schema - What Attributes are ALTERable?

- **ALTER TABLESPACE**
  - Page size (not XML) (BUFFERPOOL)
  - DSSIZE
  - SEGSIZE
  - Table space type
    - Single table simple -> PBG (inherit MC)
    - Single table segmented -> PBG
    - Classic partitioned -> PBR (inherit MC)
    - PBG -> Hash
  - MEMBER CLUSTER

- **ALTER INDEX**
  - Page size (BUFFERPOOL)
    - In DB2 9 this was immediate with RBDP set

- **Other schema change enhancements**
  - Table space no longer needs to be stopped to alter MAXROWS
  - REORG or LOAD can now drop empty partitions from a PBG
Online Schema - Optimizations

- **Undefined table or index spaces**
  - ALTERs take immediate effect

- **ALTER BUFFERPOOL**
  - Data sharing, the object no longer needs to be stopped

- **ALTER BUFFERPOOL (no pagesize change)**
  - ALTERs take immediate effect
  - Unless other pending operations exist
Universal Table Space Enhancements - Requirements

- No MEMBER CLUSTER support
- No pre-allocation of partitions for PBG
- DSN1COPY difficult since user cannot explicitly create PBG partitions
- Need drop and create to change table space type
MEMBER CLUSTER support for UTS

- ALTER support
  - Pending ALTER
- New column MEMBER CLUSTER in SYSTABLESPACE
  - Value in existing TYPE column used to populate new column on DB2 10 migration
UTS Enhancements – Solutions …

• Provide ability to add new partitions to PBG
  • ALTER TABLE … ADD PARTITION;
    • Aux objects implicitly created
    • Single partition at a time
• Provide ability to create multiple partitions on create of PBG
  • Primarily to support hash access
    CREATE TABLESPACE …
    MAXPARTITIONS 10
    NUMPART 10
    IN …
  ▪ CREATE will take longer when pre-allocating partitions
UTS Enhancements - Solution

- Deprecate classic partitioned table spaces
  - By default, creation of new classic partitioned table spaces requires explicit specification of SEGSIZE 0
  - Old syntax will create PBR
    - Default SEGSIZE for UTS in 10 is 32
  - New Zparm to specify default SEGSIZE
    - Setting this Zparm to zero will allow the old syntax to create classic partitioned table spaces
Other Availability Improvements

- Access currently committed data
- Change DDF location alias names online (planned for post GA)
  - New MODIFY DDF ALIAS command
- Online DDF CDB changes (planned for post GA)
  - LOCATIONS, IPNAMES, IPLIST
Logging Enhancements - Solution

- **Dynamic add of active logs**
  - New –SET LOG NEWLOG option
  - New active log must be IDCAMS defined & preformatted by DSNJLOGF
  - Only a single log dataset at a time
    - Issue command twice for dual logging
  - Limit is still 93 active log pairs
  - No dynamic delete of active logs

- **Pre-emptable backout**
  - Pre-DB2 10, abort/backout schedules non-preemptable SRB
    - On single CPU system may give impression of DB2 hang
  - DB2 10: Create enclave at restart for preemptable SRB backout processing
Introduce new AUX keyword for REORG

- UTS or classic partitioned
- Allows movement of base rows by REORG even though LOB columns exist
  - Essential for PBG
- Allows REBALANCE even though LOB columns exist
- Would allow pruning of PBGs even though LOB columns exist...
- Allows DISCARD to delete associated LOB values
- Default is AUX NO unless:
  - Multi-part REORG of PBG with LOB columns
  - REBALANCE of PBR/classic partitioned with LOB columns
  - REORG of PBR/classic partitioned with multiple parts in REORP
- No mapping table change
- Restrictions
  - No XML column support
REORG Enhancements - Solution

- **REORG & LISTDEF support for multiple part ranges**
  - REORG TABLESPACE... PART 1,23:48,596,3042:3800
  - Retrofit REORG support to DB2 9 in PK87762

- **Allow REORG to cancel threads**
  - Option to cancel all or just read claimers to ensure drain succeeds
    - FORCE(NO|READERS|ALL)

- **Support REORG SHRLEVEL REFERENCE or CHANGE if REORP**
  - Previously SHRLEVEL NONE was only option after alter of limitkey
  - Provides restartability

- **Support REORG SHRLEVEL CHANGE for REBALANCE**

- **Reduce outage by updating inline stats after drain released in UTILTERM**
REORG Enhancements - Solution

- REORG SHRLEVEL CHANGE for LOB page sets
  - No mapping table required
  - No access to base table, but not permitted if base is NOT LOGGED

- REORG SHRLEVEL NONE for LOBs deprecated in DB2 10 NFM
  - Will run but is a no-op with a message saying nothing done
Backup/Recovery Enhancements - Requirement

- Improve COPY/RECOVER performance & reduce overhead
- Faster PIT recovery
- Allow creation of consistent copies with no outage
- COPY problems with CHANGELIMIT if no copy taken
- COPY problems with incrementals if no pages changed
- System level backups not taken into account for REPORT RECOVERY
FlashCopy Support - Solution

- Dataset level FlashCopy for utilities
  - COPY
  - REORG inline copy
  - LOAD inline copy
  - FlashCopy of indexes for LOAD, REORG, REORG INDEX, REBUILD INDEX
- Can combine with sequential copy if required
- ZPARMs for global settings & utility parms for local settings
- FlashCopy backups can be used as input to:
  - RECOVER
  - COPYTOCOPY
    - Create sequential copies from FlashCopy
  - DSN1COPY, DSN1PRNT
    - Remove performance issue with DSN1COPY of inline copies
  - Cannot unload from FlashCopy
    - Use COPYTOCOPY and unload from that
FlashCopy Support - Solution

- REORG, REBUILD, LOAD SHRLEVEL NONE always produce consistent copies
- COPY, LOAD SHRLEVEL NONE produce consistent copies if FLASHCOPY CONSISTENT specified
  - Copy made consistent by backing out uncommitted updates against copy as shadow
  - Recovery can be elongated
- FlashCopies are dataset level but may be copied to single dataset to create sequential copy
- REPORT RECOVERY support for system level backups
COPY - Solution

- **COPY CHANGELIMIT**
  - Delay allocating output dataset until CHANGELIMIT checked
  - &ICTYPE in template will no longer be a “C”, instead will reflect the correct type of image copy
  - Use RTS to decide between incremental or full

- **Incremental copies**
  - Delay allocating output dataset until pages to be copied are found
  - Insert dummy SYSCOPY record to register empty IIC
PIT Recovery - Solution

- **New BACKOUT option on RECOVER**
  - Roll back on log from current point instead of restoring recovery base and rolling forward
  - Works with PIT consistency, so changes prior to logpoint may be backed out
  - Can only be done once for a given log range
DB2 10 Productivity – Doing More With Less!

- Auto statistics collection
- Easier scaling, simpler memory management
- Reduce contention, more online processing
- Reduced need for REORG
  - Build compression dictionary on the fly
  - Index list prefetch enhancements
- Configure IBM UDFs and stored procedures
- Allow one SDSNEXIT data set for many subsystems
- Monitoring enhanced
  - Timeout / deadlock diagnostics
  - Identify SQL statements

Manual invocation of:
- RUNSTATS
- COPY/BACKUP SYSTEM
- QUIESCE
- MODIFY RECOVERY
- REORG
Autonomics and DBA Productivity …

- Checkpoint intervals based on both time and # log records
- Identify unused packages

SQL Statement level monitoring
- Statement ID introduced
- Trace records & messages extended to include statement ID
- New trace class for statement detail
  - GetPages, Locks, I/Os, cpu/elapsed time, etc. at statement level
  - Available via IFI for online monitors or via SMF/GTF

Exploit z/OS 1.10 WLM service to temporarily boost priority of blocking lock holders
- Complements V9 health monitor task which handles latches

Manage max threads, connections, idle thread timeout on an application basis
- Warning or exceptions issued when threshold is hit
- Profiles can be set based on userids, packages, IP addresses, member names, …
Autonomics and DBA Productivity …

- New universal language interface module
  - Previously, DB2 apps had to be link edited with a specific language interface module depending on the attachment type. E.g. DSNRLI for RRS or DSNALI for call attach
    - Had to maintain separate load module library for each type of attachment
  - V10 introduces a new universal language interface module, DSNULI, which can be used in any environment
    - Now a single library can be used for all attachment types
    - Easier to manage and administer

- New ASID keyword on –START TRACE command

- Split out lock and latch wait time in accounting trace records

- Allow user to specify table space attributes for implicitly created table space on a CREATE TABLE statement
  - Two new zparms for implicit DSSIZE and TRACKMOD options

- ALTER TABLE Rotate First to Last
  - Can be used to simulate DROP PARTITION
Autonomics and DBA Productivity …

- **New DSNTIDXAXB installation CLIST for SAP customers**
  - Pre-configured with SAP recommended settings

- **Update CLIST migration input member (DSNTIDxx) with current DB2 settings**
  - New DSNTIJJXZ job to update DSNTIDxx with current DB2 parameter settings
  - Makes migration to new DB2 version easier by avoiding error-prone manual editing
Logging Enhancements - Solution

- **Provide ability to checkpoint based on both time and number of log records**
  - Meaning of CHKFREQ is unchanged
    - Minimum # of log records raised from 200 to 1000
  - New ZPARMs to control new behavior
    - CHKLOGR – number of log records between checkpoints
      - 1000 – 99,999,999
    - CHKMINS – number of minutes between checkpoints
      - 1-1439
    - CHKTYPE SINGLE|BOTH – govern old/new
  - Set by dynamic ZPARM or –SET LOG command
    - -SET LOG change does not persist across restart
  - -DIS LOG command indicates settings and if mode is SINGLE or BOTH

- **OUTBUFF default changed to 4MB (used to be 400KB)**
Optimization Stability and Control

Provide unprecedented level of stability for query performance by stabilizing access paths for

- Static SQL - Relief from REBIND regressions
- Dynamic SQL
  - Remove the unpredictability of PREPARE
  - Extend Static SQL benefits to Dynamic SQL

Support:
- Access path repository
- Versioning
- “Fallback”
- “Lockdown” (targeting post GA)
- Manual overrides. Hints: easily influence access paths without changing apps
- Per-statement BIND options

- Safe query optimization: assess “reliability” of access path choices
- Adaptive algorithms, e.g. RID pool overflow to workfiles
DB2 10: Business Security and Compliance

- **Protect sensitive data from privileged users & improve productivity**
  - SECADM & DBADM without data access
  - Usability: DBADM for all DB
  - Revoke without cascade

- **Separate authorities to perform security related tasks, e.g. security administrator, EXPLAIN, performance monitoring and management**

- **Audit privileged users**

- **Row and column access control**
  - Allow masking of value
  - Restrict user access to individual cells

*Use disk encryption*
DB2 10 Security Benefits

- More flexible authorization
- Separation of duties
- Do job without access to data
- Policies for audit
- Simpler control
- Tighter security
- Avoid cascade delete
- Avoid views and application security logic
- Allow more tools
- Evolve security policies
- Easier to manage security policy
  → Improved productivity & tighter security

Use disk encryption
Fine Grained Access Control (FGAC)

- A table level authorization function fully implemented in the database that provides:
  
  - Row level access control based on customer-supplied rules
    - A doctor can see rows representing his patients only
    - A manager can see rows representing his employees only
  
  - Column level access control based on customer-supplied rules
    - This is also called *data masking*
    - A teller can see only the last 4 digits of the credit card number column
  
- Defined and managed by SECCADM
Use case scenario

- CUSTOMER table contains information about the bank customers
- INTERNAL_INFO table contains information about bank employees
- The bank has a security policy for access to customer information
- Table: CUSTOMER

<table>
<thead>
<tr>
<th>Account</th>
<th>Name</th>
<th>Income</th>
<th>Branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>1111-2222-3333-4444</td>
<td>Alice</td>
<td>22,000</td>
<td>A</td>
</tr>
<tr>
<td>2222-3333-4444-5555</td>
<td>Bob</td>
<td>71,000</td>
<td>B</td>
</tr>
<tr>
<td>3333-4444-5555-6666</td>
<td>Carl</td>
<td>123,000</td>
<td>B</td>
</tr>
<tr>
<td>4444-5555-6666-7777</td>
<td>David</td>
<td>172,000</td>
<td>C</td>
</tr>
</tbody>
</table>
**Use Case Scenario …**

**ROW PERMISSION FOR TELLERS**

- CREATE PERMISSION TELLER_ROW_ACCESS
- ON CUSTOMER FOR ROWS WHERE
  - VERIFY_ROLE_FOR_USER (USER, 'TELLER') = 1 AND
  - BRANCH = (SELECT HOME_BRANCH FROM INTERNAL_INFO WHERE EMP_ID = USER)
- ENFORCED FOR ALL ACCESS ENABLE

**ROW PERMISSION FOR CSR and TELEMARKETER**

- CREATE PERMISSION CSR_ROW_ACCESS
- ON CUSTOMER FOR ROWS WHERE
  - VERIFY_ROLE_FOR_USER (USER, 'CSR') = 1 OR
  - VERIFY_ROLE_FOR_USER (USER, 'TELEMARKETER') = 1
- ENFORCED FOR ALL ACCESS ENABLE
Use Case Scenario …

COLUMN MASK FOR ACCOUNT NUMBER

CREATE MASK ACCOUNT_COL_MASK
ON CUSTOMER FOR COLUMN ACCOUNT RETURN
CASE WHEN (VERIFY_ROLE_FOR_USER (USER, 'TELEMARKETER') = 1)
  THEN 'xxxx-xxxx-xxxx-' || SUBSTR(ACCOUNT,13,4)
  ELSE ACCOUNT
END ENABLE

ACTIVATE ROW AND COLUMN LEVEL ACCESS CONTROL

ALTER TABLE CUSTOMER
ACTIVATE ROW LEVEL ACCESS CONTROL
ACTIVATE COLUMN LEVEL ACCESS CONTROL
Use Case Scenario …

- **Teller Amy issues the following query**

```
SELECT ACCOUNT, NAME, BRANCH FROM CUSTOMER
```

<table>
<thead>
<tr>
<th>Account</th>
<th>Name</th>
<th>Income</th>
<th>Branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>2222-3333-4444-5555</td>
<td>Bob</td>
<td>71,000</td>
<td>B</td>
</tr>
<tr>
<td>3333-4444-5555-6666</td>
<td>Carl</td>
<td>123,000</td>
<td>B</td>
</tr>
</tbody>
</table>

- **Telemarketer Pat issues the following same query**

<table>
<thead>
<tr>
<th>Account</th>
<th>Name</th>
<th>Income</th>
<th>Branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXX-XXXX-XXXX-4444</td>
<td>Alice</td>
<td>22,000</td>
<td>A</td>
</tr>
<tr>
<td>XXXX-XXXX-XXXX-5555</td>
<td>Bob</td>
<td>71,000</td>
<td>B</td>
</tr>
<tr>
<td>XXXX-XXXX-XXXX-6666</td>
<td>Carl</td>
<td>123,000</td>
<td>B</td>
</tr>
<tr>
<td>XXXX-XXXX-XXXX-7777</td>
<td>David</td>
<td>172,000</td>
<td>C</td>
</tr>
</tbody>
</table>
DB2 10 New Application Features

- Data versioning, temporal data
- pureXML enhancements
- Large object improvements
  - Allow non-NULL default values for inline LOBs
  - Loading and unloading tables with LOBs
    - LOBs in input/output files with other non-LOB data
- Currently committed locking semantics
- Implicit casting or loose typing
- Timestamp with timezone
- Greater timestamp precision – to picoseconds
- Moving Sum, Moving Average
Temporal Based Analysis

- **System-maintained temporal tables**
  - DB2 generated history
  - AS OF query

- **User-maintained temporal tables**
  - User provide time period
  - Automatic business time key enforcement.
  - Query over any current, any prior, future point/period in business time.
  - New time range update/delete statements support automatic row splitting, exploited by the merge statements.

- **Bi-temporal, combination of the above two**
Motivation for Temporal Data

- Dramatic cost savings via Application Simplification
- Dramatic Reduction in Time to Deployment
- Auditing compliance advantages by moving logic from application to database layer
- Provides an easy way for applications to manage time sensitive data
  - Powerful, novel, yet intuitive SQL
  - End users deploy without IT Staff intervention
  - Query past/future data with current queries
  - Same schema name, simply add novel temporal clause
  - Response time for current DML/queries preserved
  - Response time for past/future queries comparable
Customer Input Which Motivated Temporal Data

- “We work in an environment of constantly changing state and local laws, we need to support the evolution of a policy”
- “We have complicated procedures to ensure that our customer has exactly one position effective at any point in time, and our code still lets some bad data through”
- “We have packaged apps that we need to enable for compliance through history generation of data base changes made without changing the packaged app.”
- “We would like to take an existing SQL app and run it as of last quarter w/o changing the app”
- “We need audit history for to be able to catch and correct errors, our compliance mandate requires us to keep the history of the evolution of all policies we have issued.”
Current and History

Current SQL Application

Current

History

History Generation

Auditing SQL Application Using ASOF

Transparent/automatic Access to satisfy ASOF Queries

Current

Jul 2008
Aug 2008
Sep 2008
### System Maintained Temporal Table

**CREATE TABLE employees**

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Constraints</th>
<th>Generated</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>EmpID</td>
<td>INTEGER NOT NULL</td>
<td>PRIMARY KEY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dept</td>
<td>VARCHAR(10)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System_start</td>
<td>TIMESTAMP NOT NULL</td>
<td>generated always as row begin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System_end</td>
<td>TIMESTAMP NOT NULL</td>
<td>generated always as row end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trans_start</td>
<td>TIMESTAMP(12)</td>
<td>generated always as transaction start id implicitly hidden</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERIOD SYSTEM_TIME</td>
<td>(System_start, System_end)</td>
<td>in emp_space;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CREATE TABLE employees_history** like employees in hist_space;

**ALTER TABLE employees**

**ADD VERSIONING USE HISTORY TABLE employees_history;**

**Notes:**
- Defining SYSTEM_TIME PERIOD enforces that system_end > system_start.
- Alternatively you can also provide the complete column definition for the history table.
- Transaction start ID column allows DB2 to identify the changes made by the same transaction.
**Insert and Update**

On 11/15/1995, Employee 12345 and 67890 were hired into the department J13 & K25.

*INSERT INTO employees (EmpID,Dept) VALUES (12345,'J13'), (67890,'K25')*

<table>
<thead>
<tr>
<th>EmpID</th>
<th>Dept</th>
<th>System_start</th>
<th>System_end</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>J13</td>
<td>11/15/1995</td>
<td>12/31/9999</td>
</tr>
<tr>
<td>67890</td>
<td>K25</td>
<td>11/15/1995</td>
<td>12/31/9999</td>
</tr>
</tbody>
</table>


*UPDATE employees SET Dept='M24' WHERE EmpID=12345*

<table>
<thead>
<tr>
<th>EmpID</th>
<th>Dept</th>
<th>System_start</th>
<th>System_end</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>M24</td>
<td>01/31/1998</td>
<td>12/31/9999</td>
</tr>
<tr>
<td>67890</td>
<td>K25</td>
<td>11/15/1995</td>
<td>12/31/9999</td>
</tr>
</tbody>
</table>

12345 was in J13 between 11/15/1995 and 1/31/1998

*Note: DATE instead of TIMESTAMP type is used in examples to simplify display*
Delete and Update

On 3/31/2000, Employee 67890 left the company.

DELETE FROM employees WHERE EmplID=67890

<table>
<thead>
<tr>
<th>EmplID</th>
<th>Dept</th>
<th>System_start</th>
<th>System_end</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>M24</td>
<td>01/31/1998</td>
<td>12/31/9999</td>
</tr>
</tbody>
</table>

67890 was in K25 between 11/15/1995 and 3/31/2000

On 5/31/2000, Employee 12345 joined the department M15.

UPDATE employees SET Dept='M15' WHERE EmplID=12345

<table>
<thead>
<tr>
<th>EmplID</th>
<th>Dept</th>
<th>System_start</th>
<th>System_end</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>M15</td>
<td>05/31/2000</td>
<td>12/31/9999</td>
</tr>
</tbody>
</table>

12345 was in M24 between 1/31/1998 and 5/31/2000
Query

1. Which department is employee 12345 in?
   SELECT Dept
   FROM employees
   WHERE EmpID=12345
   Note that without the FOR SYSTEM_TIME clause, it defaults to the current table only.

2. Which department was employee 12345 in on 12/01/1997?
   SELECT Dept
   FROM employees FOR SYSTEM_TIME AS OF '12/01/1997'
   WHERE EmpID=12345

3. How many departments has employee 12345 worked in?
   SELECT count(distinct Dept)
   FROM employees FOR SYSTEM_TIME FROM MINVALUE TO MAXVALUE
   WHERE EmpID=12345

© 2010 IBM Corporation
### Financial planner’s account Timeline

<table>
<thead>
<tr>
<th>planner</th>
<th>account</th>
<th>OT</th>
<th>hourly_rate</th>
<th>Bus_start</th>
<th>Bus_end</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>Eileen</td>
<td>N</td>
<td>100</td>
<td>01/31/2008</td>
<td>03/31/2008</td>
</tr>
<tr>
<td>12345</td>
<td>James</td>
<td>N</td>
<td>120</td>
<td>03/31/2008</td>
<td>11/30/2008</td>
</tr>
<tr>
<td>12345</td>
<td>Tom</td>
<td>N</td>
<td>110</td>
<td>03/01/2009</td>
<td>06/30/2009</td>
</tr>
<tr>
<td>12345</td>
<td>Lisa</td>
<td>N</td>
<td>100</td>
<td>06/30/2009</td>
<td>12/31/2009</td>
</tr>
<tr>
<td>12345</td>
<td>Mary</td>
<td>N</td>
<td>100</td>
<td>02/01/2010</td>
<td>04/30/2010</td>
</tr>
</tbody>
</table>

- Each row presents account information for the period indicated.
- The column OT stands for whether it is possible to charge overtime.
- The column hourly_rate indicates how much the account total is increased.
- Applications maintain both Bus_start and Bus_end.
- Given the planner 12345 and a point in time, there is at most one account that this planner is assigned.
- Some time gaps can exist for the business time period. In this example, these gaps represent the period when the planner is not assigned to any account.
Defining a user-maintained temporal table

```sql
CREATE TABLE accounts
(planner INTEGER NOT NULL,
account VARCHAR(10),
OT CHAR,
hourly_rate INTEGER,
bus_start DATE NOT NULL,
bus_end DATE NOT NULL,
PERIOD BUSINESS_TIME (bus_start, bus_end),
PRIMARY KEY (planner, BUSINESS_TIME WITHOUT OVERLAPS));
```

Notes

- Defining BUSINESS_TIME PERIOD enforces that bus_end > bus_start.
- Period primary key:
  (planner, Business_time without overlaps): for a given planner and a business time (date in this case), we can find at most one working account. Constraint enforcement is provided automatically by DB2.
Find whose account planner 12345 will be working on as of 7/1/2009.

```sql
SELECT account
FROM accounts
  FOR BUSINESS_TIME AS OF '07/01/2009'
WHERE planner=12345
```

Find the number of accounts assigned to planner 12345 in 2008.

```sql
SELECT count(distinct account)
FROM accounts
  FOR BUSINESS_TIME BETWEEN '01/01/2008' AND '12/31/2008'
WHERE planner=12345
```
DB2 10 More New Application Features

- SQL stored procedure enhancements
  - SQL PL in Scalar UDFs & XML support
- 64-bit ODBC – also DB2 9 PK83072
- EXTENDED INDICATOR VARIABLES to indicate value not supplied or default
- DRDA support of Unicode for system code points
- CURRENT EXPLAIN MODE special register
- Allow caching of dynamic SQL statements with literals
- Data-dependent paging
  - When only a specific part of the result set is needed
  - Efficient access to desired portions of result set, based upon current position
- MAX/MIN function max input argument length raised to 32764
pureXML Performance and Usability Improvements

- XML schema validation in the engine for improved usability and performance
- Binary XML exchange format improves performance
- XML multi-versioning for more robust XML queries
- Allow easy update of sub-parts of XML document
- Stored proc, UDF, Trigger enhanced support
- XML index matching with date/timestamp
- CHECK utility checks XML
DB2 10 Query Enhancements

- CPU time reductions for queries, batch, & transactions
- SQL enhancements: Moving Sum, Moving Average, temporal, timestamp, implicit cast, SQL PL, ...
- Access improvements: Index include columns, hash, index list prefetch, workfile spanned records, ...
- Optimization techniques
  - Remove parallelism restrictions and more even parallel distribution
    - increased zIIP use
  - In-memory techniques for faster query performance
  - Access path stability and control
- Analysis: instrumentation, Data Studio & Optim Query Tuner
- Advanced query acceleration techniques
  - IBM Smart Analytics Optimizer
DB2 10 for z/OS: Skip-Level Migration

May move from V8 to DB2 10, but just because you can, doesn’t mean you always should....

Data sharing mixed release coexistence fully supported V8/10 or V9/10

Key considerations:

- Risk/reward analysis
  - What’s the risk? Tolerance level?
  - How will you do it? What’s your mitigation plan? Are ISVs ready?
  - What workloads do you need to test and can you test them properly?
  - Am I missing out on DB2 9 value in the meantime?

- Migration cost savings is not 2x versus two migrations
  - Migration considerations for two releases still apply
  - Larger migration project, longer migration timeline
  - Applications and ISVs may not be ready
Key Details About DB2 10: Getting Ready

Prerequisites: migrate from DB2 9 for z/OS or DB2 for z/OS V8

- z/OS V1.10 SMS-managed DB2-managed DB2 catalog
- System z10, z9, z890, z990, and above (no z800, z900)
- DB2 Connect 9 FP1, 9.7 FP3 for many 10 functions, FP2 beta
- IMS 10 & 11 (not 9) CICS compilers (See announcement)
- Info APARs for migration II14477 (9), II14474 (8)
- SPE PK56922 PK69411 PK61766 PK85956 PM04680 PK87280 PK87281 PM08102 PM08105
- Premigration check DSNTIJPA PM04968

Items deprecated in earlier versions eliminated: more for V8 mig.

- Private protocol → DRDA (DSNTP2DP, PK92339, PK64045)
- Old plans and packages V5 or before → REBIND
- Plans containing DBRMs → packages PK62876 PK79925 (V8)
- ACQUIRE(ALLOCATE) → ACQUIRE(USE)
- Old plan table formats → DB2 V8 or 9, Unicode, 59 cols PK85068
- BookManager use for DB2 publications → Info Center, pdf
Compelling Reasons to Prepare for DB2 10

- **Business needs to save money**
  - Reduce CPU time
  - Service Oriented Architecture

- **Application developers need**
  - Productivity via XML
  - Easier porting to DB2 for z/OS

- **Database Administrators need**
  - Improved performance
  - Availability, scalability & memory management
  - Simple security and regulatory compliance
  - More productive database administration
Summary - Top 10 in DB2 10 for z/OS

1. CPU reductions for transactions, queries, & batch
2. Ten times more users by avoiding memory constraints
3. More concurrency for catalog, utilities, and SQL
4. More online changes for data definition, utilities and subsystems
5. Improved security with more granularity
6. Temporal or versioned data
7. SQL enhancements improve portability
8. pureXML performance and usability enhancements
9. Hash, index include columns, access path stability, skip migration, … Pick your favorite!
10. Productivity improved for database & systems administrators, and application programmers
Questions?

Thank You