



## DB2 for z/OS Selected Performance Topics

Roger Miller  
*IBM*



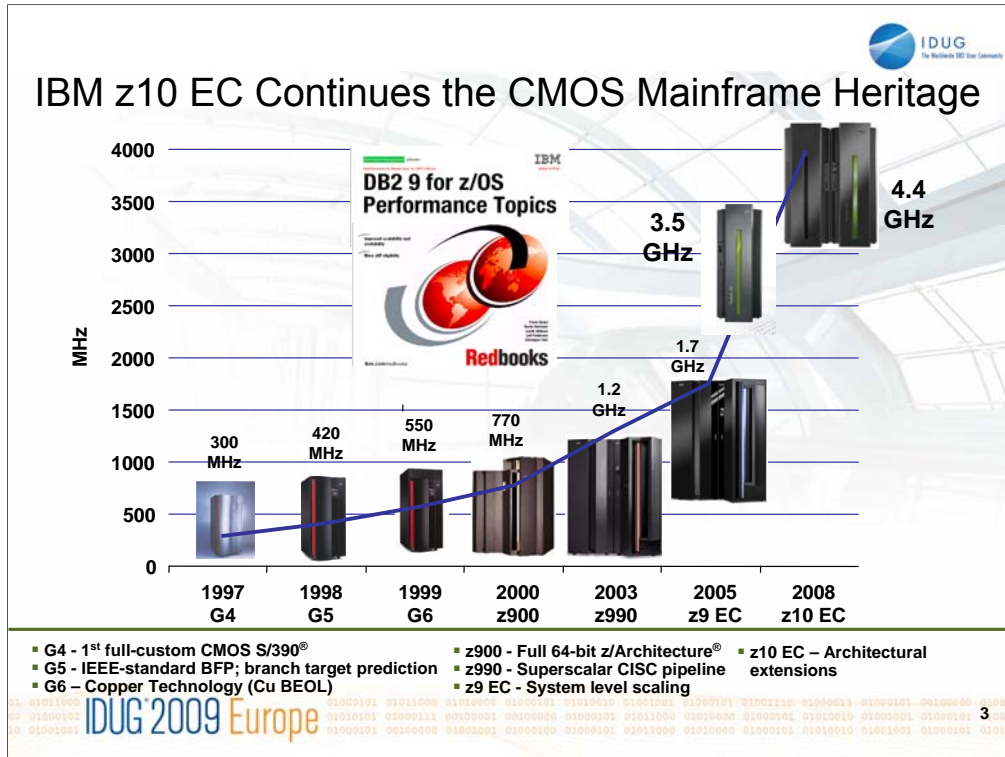
06 October 2009 • 13:00 – 14:00  
Platform: DB2 for z/OS

This talk will present some of the latest challenges and achievements for DB2 performance, noting some recent measurements and what can be achieved with the latest software and the latest hardware. The discussion will be centered on what you can do to help improve performance in your shop, particularly on the items that can help in DB2 9.

## DB2 for z/OS Selected Performance Topics

- System z synergy
- System performance tuning: what can be achieved across the subsystem
- Database design changes that can make significant improvements
- Application tuning: SQL, utilities, and other areas
- Sources and resources

The first part of this presentation is system performance, looking at the subsystem and overall system, covering the aspects of objectives, metrics, managing, resources, tuning and System z. The second part is application performance, with sections for tuning, SQL, languages, utilities, environment, database design, and application design. We'll finish with pointers to many other sources and resources, providing a detailed roadmap of where to look for more information. We expect this information to be usable by many people with highly varied tasks. Performance professionals and database administrators who are concerned about performance will be able to use most of the content. Application designers, architects and programmers will primarily use the application performance part.



The design of the IBM System z10™ processor chip is the most extensive redesign in over 10 years, resulting in an increase in frequency from 1.7 GHz (z9 EC) to 4.4 GHz on the z10 EC. The z10 BC processors run at 3.5 GHz. The average performance increase for the z10 EC over the z9 EC is about 58%, but we see substantial variation in that ratio as workloads change, from 40% to 80% for most workloads, but some improve by a factor of 2.1 times faster, while some can run at very close to the same speed. The number of cycles per instruction increases to roughly 5 cycles per instruction.

It is designed for secure data serving, yet also was enhanced to provide improvement enhances for CPU intensive workloads. The result is a platform that continues to improve upon all the mainframe strengths customers expect, yet opens a wider aperture of new applications that can all take advantage of System z10s extreme virtualization capabilities, and lowest TCO versus distributed platforms.

See section 4.3.1 z10 performance in the latest updates of DB2 9 for z/OS Performance Topics, SG24-7473 for additional detail.

## DB2 for z/OS & IBM zIIP value

Portions of DB2 V8 and **DB2 9 (blue)** workloads may benefit from zIIP\*:

**ERP, CRM, Business Intelligence or other enterprise applications**

- Via DRDA over a TCP/IP connection
- **DB2 9 for z/OS Remote native SQL procedures**
- **DB2 9 XML parsing**



**Data warehousing applications\*:** Large parallel SQL queries

**DB2 9 higher percentage of parallel queries eligible for zIIP**

**DB2 Utilities LOAD, REORG & REBUILD** maintaining index structures

**Sorting for LOAD, REORG (on indexes), REBUILD, & RUNSTATS**

DB2 9 uses zIIP in two new ways, remote native SQL procedures and increased use of parallelism. See IDUG Europe 2007 presentation F06 by Terry Purcell, "Tuning your SQL to get the most out of zIIPs". This session can be obtained from IDUG online Technical Library by searching for code EU07F06.

The zIIP is designed so that a program can work with z/OS to have all or a portion of its enclave Service Request Block (SRB) work directed to the zIIP. The above types of DB2 V8 work are those executing in enclave SRBs, of which portions can be sent to the zIIP. Not all of this work will be run on zIIP. z/OS will direct the work between the general processor and the zIIP. The zIIP is designed so a software program can work with z/OS to dispatch workloads to the zIIP with no anticipated changes to the application – only changes in z/OS and DB2. IBM DB2 for z/OS version 8 was the first IBM software able to take advantage of the zIIP. Initially, the following workloads can benefit: SQL processing of DRDA network-connected applications over TCP/IP: These DRDA applications include ERP (e.g. SAP), CRM (Siebel), or business intelligence and are expected to provide the primary benefit to customers. Stored procedures and UDFs run under TCBs, so they are not generally eligible, except for the call, commit and result set processing. DB2 9 remote native SQL Procedure Language is eligible for zIIP processing. BI application query processing utilizing DB2 parallel query capabilities; and functions of specified DB2 utilities that perform index maintenance. For more, see <http://www.ibm.com/systems/z/ziip/>

The DB2 9 and z/OS System Services Synergy Update paper discusses recent XML benchmark measurements and performance information.

<http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101227>

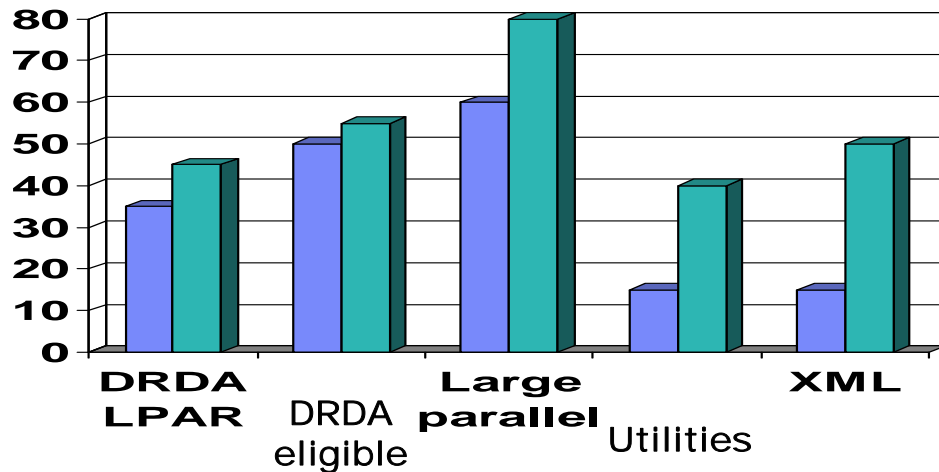
<http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101387>

DB2 9 for z/OS remote native SQL procedures are described in this paper, showing scalability up to 3193 transactions per second for SQL procedures and redirect to zIIP of over 40%

<http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD104524>

\* 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.

## How much CPU gets redirected typically?



IDUG'2009 Europe

5

The range of processing redirected to zIIP and zAAP ranges widely. For some of the example workloads tested, this chart shows common ranges. With different workloads, your numbers will vary. The largest redirection is for large parallel queries, where as much as 80% of the CPU time is redirected. Some large parallel queries cannot be processed entirely in parallel, so the redirect percentage drops. When measuring the full LPAR, with work that cannot be redirected, such as operating system and performance monitors.

The range of utility redirection is large, depending upon the percentage of index processing. Most utility processing is not during the peak time, so this part of work is not as important. The utility CPU time is reduced in DB2 9, and the amount redirected is also reduced.

z/OS XML System Services consumes approximately 15% to 50% of total CPU time in measured XML insert or LOAD operations. This portion of CPU time is eligible to exploit zAAP redirection. The amount of CPU time for z/OS XML System Services will vary widely for other applications, based on the document size, its complexity, and number of indexes defined on XML tables.

See web resources and papers noted at the end of this presentation:

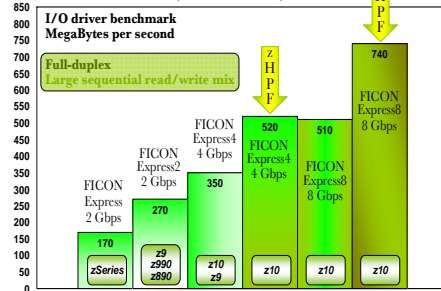
<ftp://ftp.software.ibm.com/software/data/db2/zos/presentations/overview/ziip-zaap-specialty-engines-idug-au-2008-miller.pdf>

## z10 High Performance FICON for System z (zHPF)

- Optimization of storage area network (SAN) traffic using zHPF to improve performance, especially with FICON Express8 and SSD
  - Maximum number of I/Os per second can be increased by up to 100%\*
  - For OLTP workloads (DB2, VSAM, PDSE, and zFS ) that transfer small blocks of fixed size data (4K blocks)

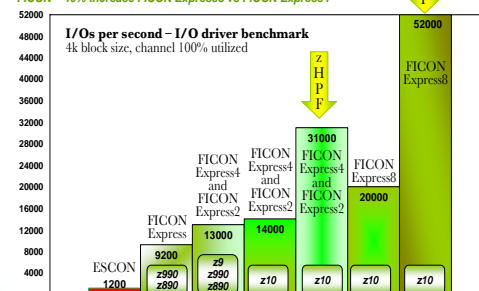
**FICON performance on System z – MBps throughput**

zHPF - 40% increase FICON Express8 vs FICON Express4  
 FICON - 45% increase FICON Express8 vs FICON Express4



**FICON performance on System z – start I/Os**

zHPF - 70% increase FICON Express8 vs FICON Express4  
 FICON - 40% increase FICON Express8 vs FICON Express4



\*This performance data was measured in a controlled environment running an I/O driver program under z/OS. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed.

FICON Express8 operating at 8 Gbps may be able to:

- Provide performance improvements for online transaction processing (OLTP) workloads - diverse business functions including order entry, inventory tracking, hotel reservations – applications such as DB2, VSAM, PDSE, and zFS

Help reduce the duration of backup/copy operations

Performance improvements with FICON Express8 may provide opportunities for consolidation or growth by reducing the number of I/O slots needed – you may be able to “do more with less” High Performance FICON and Solid State Disk more than doubles the random throughput per channel for the same amount of channel time.

Channel consolidation - fewer channels can manage more storage capacity.

Actual throughput depends on the percentage of I/Os eligible for zHPF. DB2 prefetch I/Os are not yet eligible.

Improves SSD response time and throughput by 20%

zHPF requires DS8000 Release 4.1, z10 processor and z/OS 1.10 or SPE to z/OS 1.8

SSD technology speeds up data access and removes bottlenecks imposed by spinning disks – Other important performance features include MIDAW, HyperPAV, AMP, High Performance FICON, 4 gbps FICON links

Optimization of storage area network (SAN) traffic using zHPF to improve performance

Maximum number of I/Os per second can be increased by up to 100%\*

For OLTP workloads (DB2, VSAM, PDSE, and zFS ) that transfer small blocks of fixed size data (4K blocks)

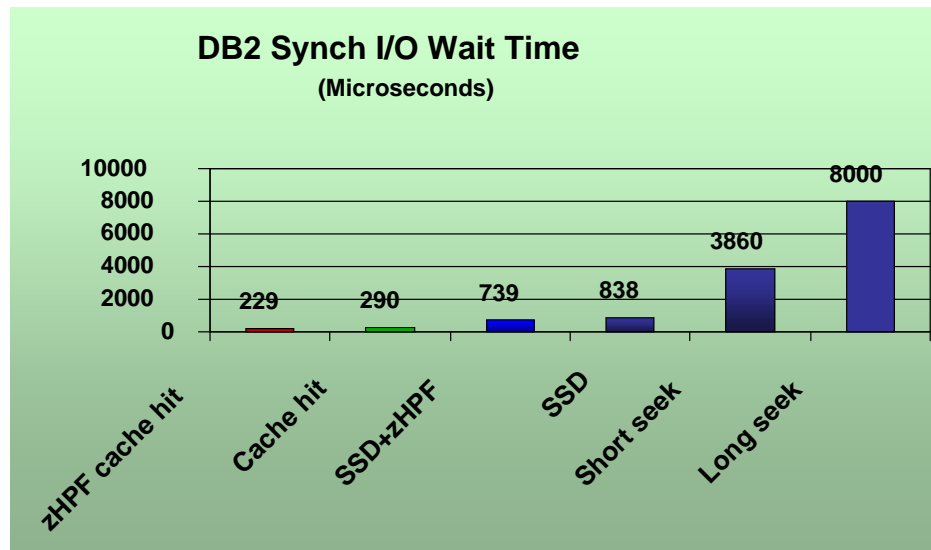
Exclusive to System z10 - FICON Express4 and FICON Express2

Requires Control unit exploitation – IBM DS8000™ Release 4.1

z/OS V1.7 with the IBM Lifecycle Extension for z/OS V1.7 (5637-A01), V1.8, V1.9, or V1.10 with PTFs

<ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/zsw03059usen/ZSW03059USEN.PDF>

## Solid state disk in storage pyramid



Understanding where solid state disk fits into the storage hierarchy is very important. This device is very fast for random IO, but it's much more expensive per byte than magnetic disk, priced for performance. Solid state disk provide much faster random IO times than hard disk. The disk cache is the fastest, at 229 to 290 microseconds. The solid state disk times are 739 to 838 microseconds. Getting data from the spinning disk requires roughly 4 to 8 milliseconds.

If you have very effective buffer pools and disk cache today, then solid state disk might not be as useful. If your IO is more random, so that your average random IO is 4 milliseconds or more, then the solid state disk may be able to hold enough of the highest IO density information to reduce the IO times significantly.

While solid state disk is higher than magnetic disk today, the cost is rapidly lowering, and new technology promises even more reductions with multilevel cells versus single level cell technology.

See Jeff Berger's presentations for more detail or this article on pricing. <http://www.informationweek.com/news/storage/systems/showArticle.jhtm?articleID=219501231&pgno=2&queryText=&isPrev=>

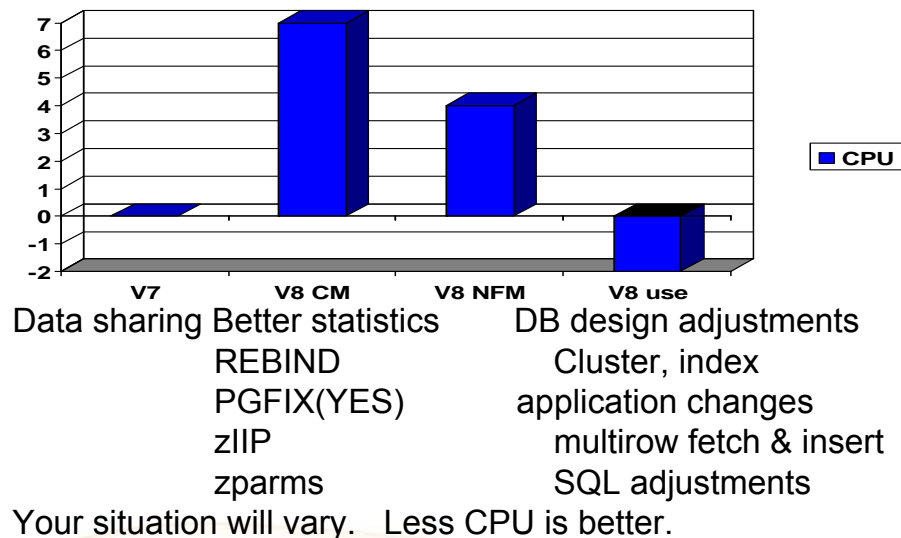
## System tuning

- New versions and service
- Parameter changes
- Buffer pools and other storage
- Use of zIIP and zAAP
- Thread reuse
- Enough performance monitoring,  
not too much

The subsystem tuning options can make a significant difference, although the database administration and application programming options generally provide larger improvements.

New versions and service can make a significant difference in performance. Some of the subsystem parameters can help, particularly when there is a bottleneck, such as memory. Use of zIIP and zAAP engines can help with costs. Thread reuse can help, when used appropriately.

## V8 best practice performance plan example scenario



IDUG'2009 Europe

9

Your situation and mileage will vary, but this is a common shape for a V8 performance plan, starting with zero for the V7 base line. When you move to V8, CPU time generally increases from 5% to 10%, shown here as 7. Start with long term page fix for buffer pools with high numbers of pages read and written. Reorg and collect improved statistics for non-uniform distribution of data on non-indexed columns. The V8 CM performance plan REBINDs the primary packages, and adjusts DSNZPARMs. The CM REBIND process provides most of the improved access paths. Data sharing batching helps in CM. During CM, a zIIP is added if your peak work load includes DRDA SQL, parallel query or LOAD, REORG and REBUILD.

In moving to NFM, some additional DSNZPARMS are adjusted and REBIND all plans and packages. Database designs start taking advantage of new clustering & indexing options, such as NOT PADDED for large varchar indexed columns. After making the design changes, REORG the data; REORG or REBUILD the indexes; get improved statistics & REBIND. The data sharing group is quiesced, and protocol 2 locking is used.

V8 use takes more advantage of the V8 performance improvements: MQTs, DPSI, more not-padded indexes, multi-row Fetch, cursor Update, cursor Delete, & Insert. Use other SQL improvements to reduce V8 CPU, less than V7. The work may grow, but some of the growth uses the zIIP.


## DB2 for z/OS V8 Performance Overview

- Performance / Scalability Enhancements
  - ▶ Improved partitioning scale and flexibility
  - ▶ Many index improvements
- Query / Access Path Performance Enhancements
- Multirow fetch and insert
- Synergy with new hardware: zIIP, MIDAW, DS8000, ...




DB2 for z/OS V8 has many key performance and scalability improvements. The biggest CPU saver for most customers is the ability to insert and select multiple rows, reducing the CPU for crossing address spaces and making a bigger improvement for distributed cases. Optimization improvements, with many new options for partitions and indexes, improved optimization techniques and better information for the optimizer are important.

Synergy with both processor and IO hardware is extended substantially. The ability to use zIIP improves costs. The memory structure expansion to 64 bit addressing helps in many ways. Faster disk speeds keep the system balanced.



V8 queries and data warehouses

DB2 UDB for z/OS  
Version 8



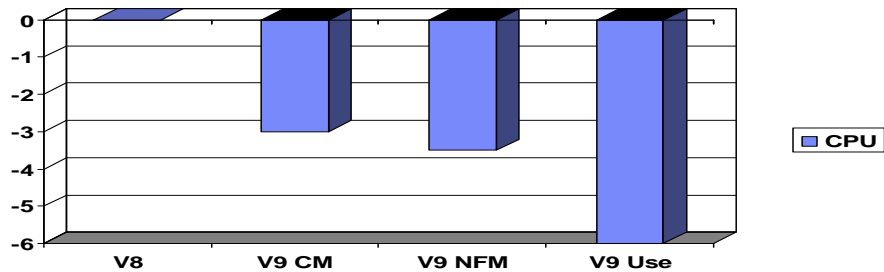
- Optimization Improvements
  - ❑ Materialized Query Tables
  - ❑ Improved optimization techniques
  - ❑ Enhanced data for optimizer
  - ❑ Visual Explain, Optimization Service Center
- Enhanced index options
- New Partitioning options
- QMF improvements
- SQL enhancements, multirow fetch & insert

IDUG 2009 Europe 11

Queries and data warehousing are improved a lot in V8. Optimization improvements provide a performance boost and make the job simpler. Improved optimization techniques like ability to use indexes more, star join and scale improvements allow reduced work for computers and for people. Enhanced data helps get the best access path. Visual Explain improves the ability to analyze and resolve any problems.

The many improvements for indexes, materialized query tables and partitioning can save space and add new options for improved performance and availability, even while simplifying the process. Not padded, clustering, longer and backward scans help indexes. Being able to add, rotate and rebalance partitions improve partitioning options. QMF enhancements build upon these strengths and add new function to reporting, dash boards, and a new platform in WebSphere. SQL enhancements on this page and the next improve portability of the SQL, improve the ability to express queries, and help with performance.

## DB2 9 z10, z9, z890 & z990 performance plan example scenario



Utilities	DB design adjustments
Histogram statistics	Index improvements
REBIND	application changes
DSNZPARMS	native SQL procedures
	SQL adjustments

Your situation will vary. Less CPU is better.

z800 and z900 expect +5% to +10% CPU

IDUG 2009 Europe

12

If you have a z9, z990 or z890, this is expected to be a common shape for a DB2 9 performance plan, starting with zero for the V8 baseline. When you first move to DB2 9, total DB2 CPU time generally decreases from 0% to 5% for z9, z890 and z990 customers, shown here as a first step -3%. Utility CPU reductions help immediately. Some work will be about the same (+/-3%). Start with reorgs and collect improved histogram statistics when useful. The DB2 9 CM performance plan REBINDs the primary packages and adjusts DSNZPARMS. The REBINDs provide most of the improved access paths. On z800 or z900 the initial CPU expectation is +5 to +10% regression, more if there are many columns, so making adjustments is more important.

In moving to NFM, some additional DSNZPARMS are adjusted and all plans and packages are rebound. The DB2 9 use line takes wider advantage of DB2 9 performance improvements. Database designs start taking advantage of new indexing options, such as compression, index on expression and larger pages. After making the design changes, REORG the data and REORG or REBUILD the indexes, get the improved statistics and REBIND. Native SQL procedures, added use of zIIP, and improved SQL continue the improvements in this phase.

Scenario: Customer mix of DB2 CPU time is 30% in utilities, 70% in SQL access. With 10% improvement for the utilities, we get a -3% net, assuming that SQL is the same as before. With optimization improvements, another -1/2% improvement shows up in DB2 9 NFM. Then as design adjustments, reorgs and rebinds are performed, we get improvements from varchar improvements, native SQL procedures and improved SQL, another -3%.

## DB2 9 for z/OS Performance Overview

- Significant CPU time reduction in most utilities
- Synergy with new hardware: zIIP, MIDAW, DS8000, ...
- Performance / Scalability Enhancements
  - Especially Insert, Update & Delete
- Query / Access Path Performance Enhancements
- Other performance enhancements: LOBs, varchar, native SQL procedure, index compression
- Improved virtual storage usage below 2GB DBM1



The key performance improvements in DB2 9 are reduced CPU time in many utilities, deep synergy with System z hardware and z/OS software, improved performance and scalability, especially for insert, update and delete, better LOB performance and scalability, improved optimization for SQL, zIIP processing for remote native SQL procedures, index compression, reduced CPU time for data with varying lengths and better sequential access. This version also improves virtual storage use below the 2 GB bar.

The optimization improvements include more function to optimize, improved information for optimization, better optimization techniques and a new approach to providing information for tuning. V8 SQL procedures were not eligible to run on the zIIP, but changing to use the native SQL Procedure Language on DB2 9 will make the work eligible for zIIP processing. Varying length data can improve substantially if there are large numbers of varying length columns. Several improvements in disk access can reduce the time for sequential disk access and improve data rates.

## Most consumable DB2 9 improvements



- CM very little to no action:
  - Optimization Service Center, Data Studio and Optim Query Tuner
  - Utility CPU reductions
  - Archive log striping, compression
  - Larger prefetch, write & preformat quantities
  - LOB performance
  - DDF VSCR
  - Index lookaside
  - Changed online REORG
- Package stability
- Improved RUNSTATS
- Optimization improvements, EDMPOOL VSCR, increased parallel & zIIP NFM
- LOB lock avoidance
- Logging for data sharing
- Improved index leaf page split
- Reordered row format, native SQL
- Index: larger page sizes, compression, index on expression



Here are some highlights for items that deliver the most quickly and easily:

Very little to no action is required for the utility CPU reductions, logging improvements, improved index page split, larger prefetch, write & preformat quantities, some LOB performance, DDF virtual storage constraint relief. The first group delivers in CM.

The next items require some work. Changed online REORG and other utility improvements require process changes and use of SHRLEVEL(CHANGE).

Improved RUNSTATS statistics needs some analysis to determine where the value is greater than the cost of gathering the new statistics.

Optimization improvements are automatic for dynamic SQL, but require work to REBIND for static SQL. In both cases, we need baselines to check for regression. REOPT(AUTO) for dynamic SQL needs analysis to be sure the improvement is working. EDMPOOL virtual storage constraint relief also requires a REBIND.

Optimization Service Center takes some learning, but should be fast for those who have used Visual Explain in the past. See the new redbook, SG24-7421, DB2 9 for z/OS: New Tools for Query Optimization.

LOB lock avoidance requires a quiesce of all subsystems in NFM until APAR PK62027. Reordered row format requires a REORG in NFM and varying length columns, and use with small columns can reduce compression. See APARs.

Index improvements for larger page sizes, compression, index on expression require database design work to determine where they are applicable. ALTERs, REORGs and creation of new indexes are needed.

## DB2 9 Utilities Performance Improvements

### CPU reductions in LOAD, REORG, and REBUILD

- Reductions mostly due to improved index processing (\* with exceptions)
- 10 to 20% in Image Copy\* (even with forced CHECKPAGE YES)
- 5 to 30% in Load, Reorg, Reorg Partition, Rebuild Index
  - Except REORG TABLESPACE SHR CHG PART with NPSIs
- 20 to 40% in Load
- 20 to 60% in Check Index
- 35% in Load Partition
- 30 to 40% in Runstats Index
- 40 to 50% in Reorg Index
- Up to 70% in Load Replace Partition with dummy input

Reduces redirect to zIIP

We have seen substantial performance improvements in the utilities, with some early customers noting as much as 20% to 30% overall reductions in utility CPU time. If utilities are 30% of the total DB2 CPU time and you average 20% improvements, then the net would be a 6% reduction in total DB2 CPU. We anticipate that most of the savings would not be for the peak processing time.

The utility improvements are broad-based, but not all processing improves. The primary improvements are in index processing, so you will probably have larger improvements if you have more indexes. Many of the CPU savings are in index processing, so the zIIP redirection for DB2 9 utilities is generally less than V8.

The utilities that process only indexes see the most dramatic improvement. For example, REORG INDEX sees a 40-50% improvement while REORG table space is only 5-30%. That's because during REORG TABLESPACE there is a lot of data row processing so that index key processing is proportionally less of the entire job than for REORG INDEX.

COPY is improved with less data movement for pages than in previous releases. This is seen even with some additional CPU needed for the default CHECKPAGE YES option.

For LOAD REPLACE of a part with no data rows, the NPIs must be updated to delete logical keys. The interface to index manager was improved so that the delete of these keys is save up to 70% of the CPU.

## DB2 9 Query Enhancements

- SQL enhancements: INTERSECT, EXCEPT, cultural sort, caseless comparisons, FETCH FIRST in fullselect, OLAP specifications: RANK, ROW\_NUMBER, ...
- pureXML integration and text improvements
- Index improvements
  - ▶ Index on expression            Larger index pages
  - ▶ Index compression            Improved page split
- Improved optimization statistics: Histogram
- Optimization techniques & REOPT(AUTO)
  - ▶ Access paths more parallel, increased zIIP
  - ▶ Cross query block optimization
  - ▶ Generalize sparse index & in-memory data cache method
  - ▶ Dynamic Index ANDing for Star Schema
- Analysis: instrumentation & Optimization Service Center

Query enhancements improve data warehousing and reporting. Today's complex applications include both transactions and reporting, so performing both well is imperative. More queries can be expressed in SQL with new SQL enhancements. The set operators INTERSECT and EXCEPT clauses make SQL easier to write. OLAP extensions for RANK, DENSE\_RANK and ROW\_NUMBER add new capabilities. Other SQL statements improve consistency with the DBMS industry. DB2 9 continues the progress in SQL, with many new functions, statements and clauses. The biggest changes are in XML. New SQL data manipulation statements are MERGE and TRUNCATE. New data types with DECIMAL FLOAT, BIGINT, BINARY and VARBINARY. Improvements in LOBs provide new function, more consistent handling and improved performance. Security is improved with network trusted context and roles. Data definition consistency and usability are improved. DB2 9 is another big step in DB2 family consistency and in the ability to port applications to DB2 for z/OS.

Indexes have many improvements in DB2 9. The key items are the ability to have an index on an expression instead of on a column, compression to save disk space, larger index pages and an improved page split to improve the insert rate.

Data sizes continue to increase while the SQL grows more complex. The SQL enhancements provide more opportunities for optimization, and DB2 9 adds optimization enhancements to improve query and reporting performance and ease of use. Improved data is provided for the optimizer, with improved algorithms and a rewritten approach to handling performance exceptions. Histogram statistics provide better information about non-uniform distributions of data when there are many values skewed, rather than just a few. Improved algorithms widen the scope for optimization. When exceptions occur, guidance and support are made easier with improved instrumentation, the Optimization Service Center and the Optimization Expert.

## Database design tuning

- Index changes: many new options
  - Ability to use index
  - Clustering
  - Compression
  - Index on expression, XML, ...
- Universal table space

Both DB2 V8 and DB2 9 have large changes to index design options. With more effective options, indexes can be used in situations that were not possible in prior releases. The improvement can be orders of magnitude.

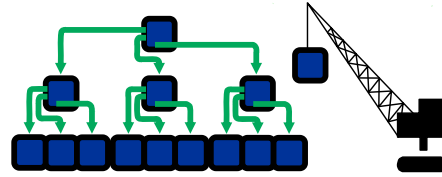
The biggest change for table spaces since Version 2 is the universal table space. It is both partitioned and segmented in the space maps. It can be partitioned even without a partitioning key. This is the table space structure for the future.

## Index: DB2 for z/OS V8



### Index Improvements

- Variable length index keys
- Index-only access for varchar data
- Maximum index key 2000 bytes
- Predicates indexable for unlike types
- Backward Index Scan
- Partitioning separate from clustering
- Data-partitioned secondary indexes (DPSI)
- Create index online during select, insert
- Add column to index



DB2 V8 provides many new opportunities for improving index processing, rebuilding the architecture for indexes.

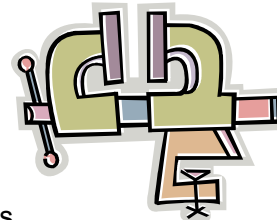
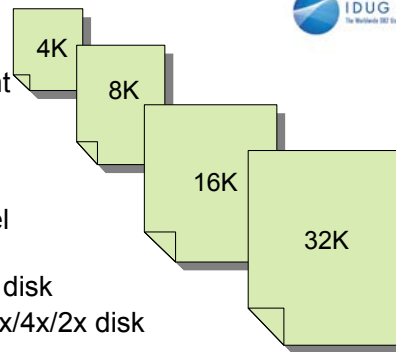
We are able to use indexes more effectively, reducing the space in variable-length indexes, being able to have index-only access with variable-length data and being able to use the index when the predicates do not match.

In some cases, such as backward index scans or partitioning, we will be able to work as efficiently with one less index. Being able to eliminate an index will improve the insert, delete, LOAD, REORG and update processing.

We have more flexibility in indexes, with longer index keys, the ability to partition secondary indexes and the ability to have more effective clustering.

## Indexing Enhancements

- Larger index pages allow more efficient use of storage
  - ▶ Fewer page splits for long keys
  - ▶ More key values per page
- Index compression provides page-level compression
  - ▶ Data is compressed to 4K pages on disk
  - ▶ 32K/16K/8K pages results in up to 8x/4x/2x disk savings
  - ▶ No compression dictionaries
    - Compression on the fly
    - No LOAD or REORG required
- Find unused indexes with real time statistics
- Rebuild Index SHRLEVEL CHANGE
- Index on expression
- Define RANDOM index keys to avoid hot spots with multiple processes inserting sequential keys



IDUG 2009 Europe

19

Indexing improvements contribute to the overall improvements in query performance. Specific improvements include index compression, index on expression, index key randomization, and larger index page sizes.

Larger index pages allow for more efficient use of storage, with fewer page splits for long keys and more key values per page.

Multiple processes inserting sequential keys can create hot spots on indexes. Randomized index keys avoid hot spots. Application insert throughput improved via avoidance of locking conflicts, but retrieval of sequential rows is likely to be slower.

Bigger index page: 4K, 8K, 16K, or 32K page → Up to 8 times less index split

Good for heavy inserts to reduce index splits. Especially recommended if high latch class 6 contention in data sharing.  
Two forced log writes per split in data sharing

Or high latch class 254 contention in non data sharing shown in IFCID 57

## Index Compression

### Differences between data and index compression

Compression: data vs index	Data	Index
Unit for compression	Row	Page (8K, 16K, 32K)
Compression using	Instruction	software
Compressed in Buffer Pool	Yes	No
Compressed Log	Yes	No
Compression Dictionary	Yes	No (2)
Compression technique	Row dictionary	Prefix, RID
Average Comp Ratio	10% to 90%	25 to 75% (3)

Similarities: Data on disk is compressed  
 Use DSN1COMP utility to predict index compression ratio.

Index compression can be a very important way to save disk space, especially if your indexes take more space than your compressed data. Index compression is very different from data compression, so your use will change significantly. Index compression does have some overhead in CPU time and in memory to save disk space. Index compression does not use a dictionary, so index data can be compressed as data is inserted. You can choose whether you want to use index compression by specifying COMPRESS YES or COMPRESS NO on the CREATE INDEX or the ALTER INDEX statements in DB2 9. See the Redpaper for much more. Index Compression DB2 9, REDP4345, <http://www.redbooks.ibm.com/abstracts/redp4345.html>

As in data compression, the DSN1COMP utility can be used to simulate compression and calculate the compression ratio, without performing the actual compression. DSN1COMP also suggests a page size for the index, usually 8K or 16K. Performance with index compression can be better or worse than data compression, but the CPU overhead is critically sensitive to the buffer pool hit ratio. Having the data remain in the buffer pool and be accessed many times is key. Thus a larger index buffer pool is strongly recommended where you use index compression with random index IO.

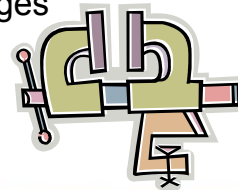
- (1) No compression or decompression in each Insert or Fetch; instead at I/O time  
 CPU overhead critically sensitive to index BP hit ratio  
     Bigger index BP strongly recommended for index compression  
     No change in accounting CPU time if index pages are brought in by prefetch

- (2) Load or Reorg not required for compression

- (3) Based on a limited survey thus far, Higher for relatively unique indexes with long keys  
 Maximum CR limited by index page size: 50% with 8K, 75% with 16K, 87.5% with 32K page

## Compression rules of thumb

- ✓ Calculate disk space used by indexes
- ✓ Avoid high IO situations or add memory to compensate
- ✓ Choose large indexes, over 10 MB
  - ✓ Especially if long index keys (over 400 bytes)
- ✓ Use DSN1COMP to estimate compression
- ✓ Where compression is
  - ✓ Over 45% and under 70%, use 8K pages
  - ✓ Over 70% and under 85%, use 16K pages
  - ✓ Over 85% use 32K pages
- ✓ Test and stage in implementation



Here are my guidelines for a successful implementation of index compression. You need to understand the differences, the opportunity, and stage the implementation carefully. While almost all customers are experienced with data compression, index compression is new and different. Start by understanding the opportunity. Some customers have more disk space occupied by indexes than data, while others have a very small fraction of the total. Index compression has several opportunities for performance regression. The first challenge is making sure that the buffer pool is large enough, as compression and decompression occur with the read and write IO. In most situations, some additional memory and CPU time is needed to save the disk space.

My suggestion for a rule of thumb to set the page size is to use 8K pages when index compression is over 45%, but less than or equal to 70%. This will be most of the compressed indexes. Use 16K pages for compression ratios over 70% to 85%, a few. Use 32K pages where the compression ratio is over 85%.

## Insert/Update/Delete Performance

DB2 9 offers improved Scalability and Performance

- ▶ Especially in data sharing environments
- Reduced Latch contention
  - ▶ Reduced Log Latch (Latch 19) Contention
  - ▶ Reduced LRSN Spin
- Asymmetric leaf page split
- Larger index page sizes
- Increased index look-aside
  - Randomized index key
  - Table space APPEND option (can ALTER on and off)
  - Not logged table spaces

Insert/Update/Delete performance is, and has always been, one of the most challenging issues in any database management system. DB2 9 adds dramatic performance/scalability improvement in this area.

Insert performance increases substantially through a wide range of improvements. DB2 9 has addressed a number of performance issues with the high usage of certain latches and the underlying problem that was causing them. Logging performance is improved with latching improvements and striped archive logging. The newer disk and channel changes, such as DS8000 Turbo, 4 Gigabit per second channels, and MIDAW, improve data rates substantially. Indexes are improved with larger page sizes to reduce the number of page splits and with a better page split. Where performance should be optimized for inserts, rather than for later retrieval, the append option can be used. If the data needs to be randomized to avoid insert hot spots, the new randomized index key is useful.


Latch class 19 reduction can also be achieved by using the new DB2 9 function of having a not logged table space. By removing logging from a table space, you remove any potential for log records that are associated with this table space to contribute to the logging load of the system.

## Asymmetric Index Page Split (NFM)

The effect of page splits after inserts of keys A5 and B5

Asymmetric index page splits lead to more efficient space usage and reduces index tree contention.

- Index split roughly 50/50 (prior to DB2 9)
- Sequential inserts → ~50% free space
- New algorithm dynamically accommodates a varying pattern of inserts
- Up to 90/10 split
- Effective across multiple inserting threads (due to tracking at the page level).
- Improve space utilization and reduce contention.

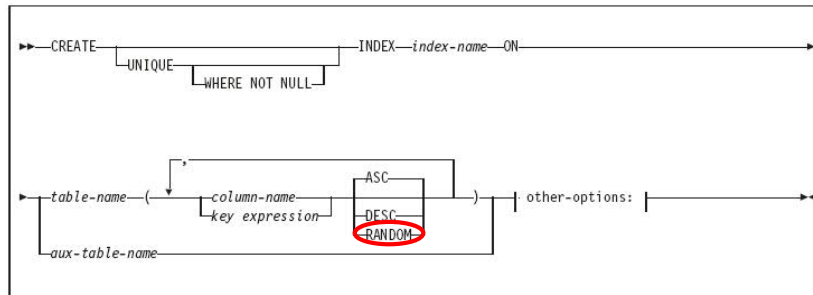

23

Prior to DB2 9, an index page can be done as a 100/0 split if keys are inserted into the end of an index or as a 50/50 split if keys are inserted in the middle of the key range. When using the 50/50 split of an index page, you can experience frequent page splits that leave half of the splitting pages empty. They use up more space and need frequent reorganization.

In DB2 9 new-function mode, the insert pattern in an index is detected. Based on the insert pattern, DB2 9 can split an index page by choosing from several algorithms. If an ever-increasing sequential insert pattern is detected for an index, DB2 splits index pages asymmetrically using approximately a 90/10 split. With a 90/10 split, most existing keys stay on the original index page, thereby leaving more space on the new index page for future inserts. If an ever-decreasing sequential insert pattern is detected in an index, DB2 splits index pages asymmetrically using approximately 10/90 split. With the 10/90 split, most existing keys are moved to the new index page, thereby making room on the original index page for future inserts. If a random insert pattern is detected in an index, DB2 splits index pages with a 50/50 ratio. Benchmarks show up to 50% reduction in index split, along with CPU/ET reductions:

- Data sharing - 20% class 2 CPU and 31% elapsed time reduction
- Non-data sharing - 10% class 2 CPU and 18% elapsed time reduction

## Randomized Index Key (NFM)



- Lock contention relief
  - Additional getpages
  - Additional read/write I/Os
  - Increased lock requests
  - Cannot support order
  - Can provide dramatic improvement or degradation!
  - Recommend making randomized indexes bufferpool resident
- Vs.**

Be careful with this new option. Index contention, especially on a hot page, can be a major problem in a data sharing environment and can limit scalability. DB2 9 new-function mode introduces a randomized index key order. The randomized key order allows DB2 to spread out index keys within the whole index tree, instead of maintaining an ascending or descending order, thereby minimizing index page contention and turning a hot index page into a cool index page. A randomized index key can reduce lock contention, but can increase the number of getpages, lock requests, and read and write I/Os. Using a randomized index key can produce a dramatic improvement or degradation of performance.

To use a randomized index key, you specify **RANDOM** after the column name on the **CREATE INDEX** command or specify **RANDOM** after the column name of the **ALTER INDEX ADD COLUMN** command. The values of the **HIGH2KEY** and **LOW2KEY** columns in the **SYSCOLSTATS** and **SYSCOLUMNS** catalog tables are not updated by **RUNSTATS** if the columns are randomized-key columns.

## Index Look-aside (CM)

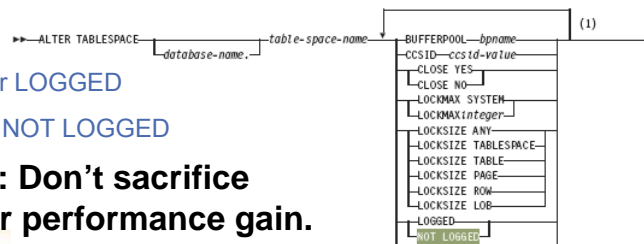
- In V8
  - ▶ Insert – clustering index only
  - ▶ Delete – no index lookaside
- In DB2 9,
  - ▶ Insert & Delete – now possible for additional indexes where CLUSTERRATIO >= 80%
- Potential for big reduction in the number of index getpages with substantial reduction in CPU time
  - ▶ Benchmark Example - Heavy insert
    - Large table, 3 indexes, all in ascending index key sequence,
    - $0+6+6=12$  index Getpages per average insert in V8
    - $0+1+1=2$  in DB2 9

Index lookaside improvements are automatic and beneficial where they apply. The objective of the index look-aside technique is to minimize the number of getpage operations that are generated when an individual SQL statement or DB2 process is executed repeatedly and makes reference to the same or nearby pages. Index look-aside results in a significant reduction in the number of index and data page getpage requests when an index is accessed in a sequential, skip-sequential, or hot spot pattern. This happens often with applications that process by ordered values. DB2 keeps track of the index value ranges and checks whether the required entry is in the leaf page accessed by the previous call. It also checks against the lowest and highest key of the leaf page. If the entry is found, DB2 can avoid the getpage and traversal of the index tree. If the entry is not within the cached range, DB2 checks the parent non-leaf page's lowest and highest key. If the entry is found in the parent non-leaf range, DB2 has to perform a getpage but can avoid a full traversal of the index tree. If an entry is not found within the parent non-leaf page, DB2 starts an index probe from the index root page.

In DB2 9, index lookaside is possible for more indexes in both insert and delete. CPU time can be reduced substantially. In V8 only the clustering index is used for insert, but index lookaside is not used for delete.

## NOT LOGGED table spaces

- Is actually NOT LOGGED tables spaces, tables, indexes, LOB, XML
- ALTER / CREATE a TABLESPACE as NOT LOGGED
  - ▶ ALTER not allowed if in same UOW with an update to the table space
- Indexes, LOB, and XML inherit the logging attribute of the base
  - ▶ These are considered “*Linked*” objects
- Effects the UNDO / REDO records
  - ▶ Control information is still logged
- LOB continue to log system pages & auxiliary indexes
- Unit of Recovery (UR) is still created



LOG YES is a synonym for LOGGED

LOG NO is a synonym for NOT LOGGED

**Recommendation: Don't sacrifice recovery for minor performance gain.**

Use NOT LOGGED only in carefully controlled situations, generally where many insert tasks are used. You can dig yourself a deep hole and fall in.

Normal gain for a single inserter is less than 3% CPU, less than 5% elapsed for a single thread insert. The performance cost can be greater than that if you need the log. The place to turn off logging is a situation where you control all of the work being done, will be inserting, updating and deleting with ten concurrent jobs or more, and will be able to restart with the image copy if anything goes wrong, goes wrong, goes wrong,...

Cannot explicitly specify for XML & Index objects

LOBs can be set independent of the base table. However, if a LOB is LOGGED, the base must also be logged. This “*dissolves the link*” with the base. Not compatible with CHANGE DATA CAPTURE attribute

Applies to any tables in the table space A FULL COPY should be taken

Just before ALTERing to NOT LOGGED

Just after ALTERing to LOGGED

## Application tuning

- Query tuning
- Dynamic to static SQL
- Multirow fetch & insert
- Stored procedures
- REORG Statistics REBIND
- Minimize processing needed

Multirow fetch and insert provide the largest opportunity for improvement, so we'll discuss this topic thoroughly and point to more resources.

Stored procedures can be a performance improvement in some situations and can be a challenge in others.

The process of reorganization, collecting statistics, and then rebinding applications is very important.

Some of the basic rules for minimizing processing need to be followed for the best performance.

Dynamic SQL to static SQL is now a range with many options. Choosing the right ones for your applications can save time.

## Query tuning improved in DB2 9

- Package BIND Stability
- Histogram Statistics
- Page Range Processing
- Global Query Optimization
- Generalized sparse index & in-memory data cache
- Dynamic Index ANDing
- Indexing Enhancements
- Optimization Service Center, Statistics Advisor, & Optim Query Tuner

See IOD presentations from Patrick Bossman on using OSC and Terry Purcell on optimization improvements in DB2 9 for much more depth on this topic.

Also check the IBM Redbooks publications at the end of this presentation, especially

## Dynamic SQL $\leftrightarrow$ Static SQL



### A spectrum of choices

- **Reduce dynamic bind frequency via**
  - **Dynamic statement caching with CACHEDYNAMIC YES**
  - **REOPT(ONCE) in V8 REOPT(AUTO) in DB2 9**
  - **Improved monitoring in V8 Visual Explain**
  - **Next step in Optimization Service Center (DB2 9 and V8)**
  
- **Incremental bind in accounting**
  - **Static plan/package with VALIDATE(RUN) and bind time failure**
  - **Static SQL with REOPT(ALWAYS), or referencing Declared Temp Table, or private protocol in requestor**

Once upon a time we could say that dynamic SQL was slow and static SQL was fast. Now the picture is much less black and white, with many shades of grey, often around the REOPT option. We have made dynamic SQL more static with several versions of dynamic statement caching. We have made static SQL more dynamic when a single access path cannot handle all of the needed options.

There is a wide spectrum of SQL, and performance improvements come from making dynamic SQL more static as well as from making static SQL more dynamic. REOPT has more options in V8 and more again in DB2 9. Programming options still matter as well.

For the program that should be bound, static SQL, one common mistake causes extra CPU time. When the package is bound with VALIDATE(RUN) and the object does not exist at BIND time, then incremental BINDs are performed, using extra CPU time. Other incremental bind causes are use of static SQL with REOPT(ALWAYS), referencing a declared temporary table, and distributed private protocol. Monitor the accounting reports for incremental binds to see if they can be avoided.

## JDBC/SQLJ

- Use CACHEDYN YES for JDBC, or better yet use SQLJ or best choice is pureQuery
- Select/Update/Insert required columns only
  - More important in JDBC/SQLJ environment
- Store numeric as smallint or int to minimize conversion and column processing cost
  - Relative cost: Integer (lowest) -> Float -> Char -> Decimal -> Date/Time -> Timestamp (highest)
- Match Java and DB2 data type
  - V8 enhancement for non-matching data type

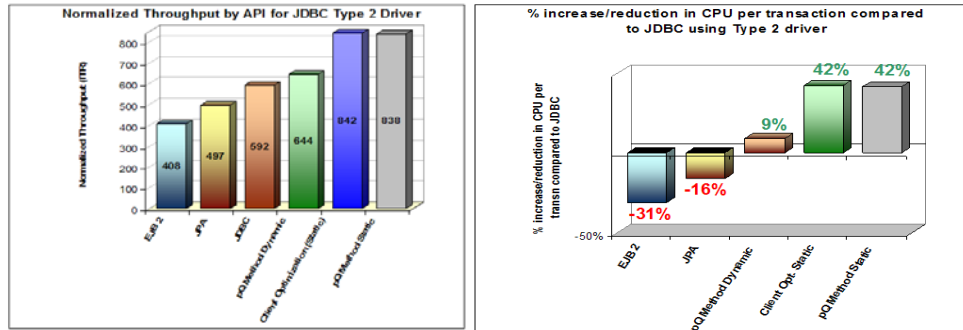
For applications using dynamic SQL, regardless of the technique, dynamic statement caching tends to make a substantial reduction in CPU use. Some techniques, like use of parameter markers, rather than literals, will help with the dynamic statement cache hit ratio. Sometimes this technique will interfere with the best optimization, since the literal value is not known at run time. Using REOPT(ONCE) then allows the first run time value to be used. REOPT(AUTO) tries to improve upon that technique, reoptimizing when needed.

Within the Java world, pureQuery is the recommended approach for the best performing SQL.

## Data Studio pureQuery Runtime for z/OS

- In-house testing shows double-digit reduction in CPU costs over dynamic JDBC

<http://www.ibmdatastatemag.com/story/showArticle.jhtml?articleID=208802229>



- IRWW – an OLTP workload, **Type 2 driver (local call)**
- Cache hit ratio between 70 and 85%
- 42% reduction in CPU per transaction** over dynamic JDBC

IDUG'2009 Europe

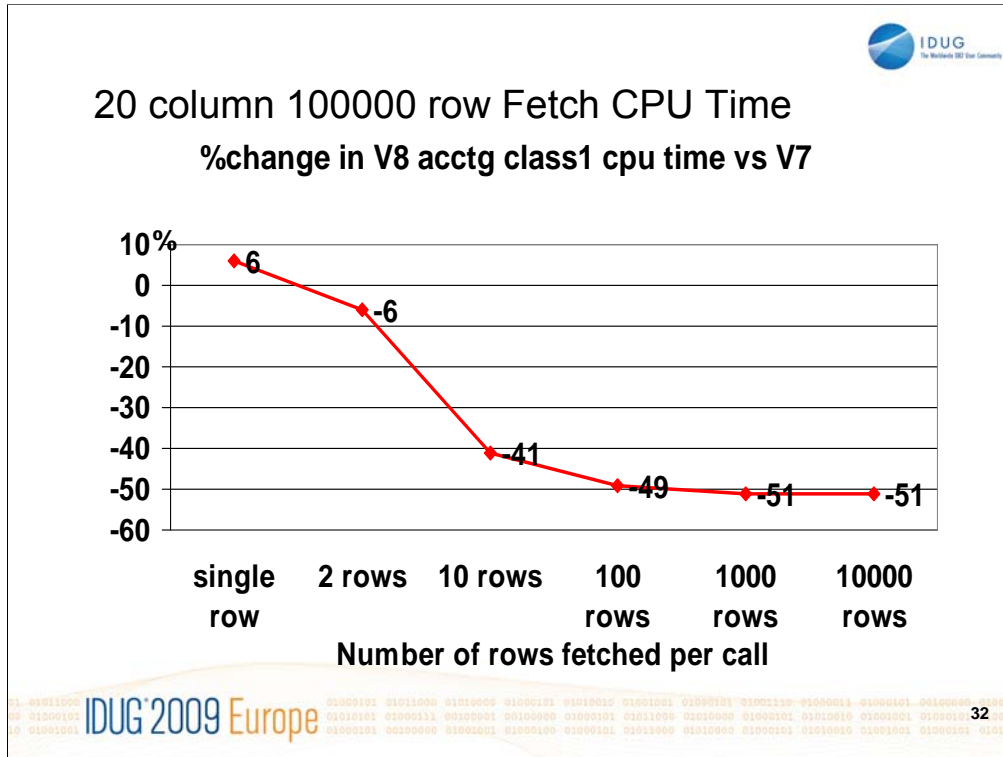
31

See the article, IBM Data Studio pureQuery Runtime for z/OS Performance

<http://www.ibmdatastatemag.com/story/showArticle.jhtml?articleID=208802229>

pureQuery improves application throughput on DB2 for z/OS by making it easy to code and deploy static SQL for Java. The new client optimization feature in Version 1.2 extends the benefits of static SQL to any existing Java application without changing or recompiling Java applications.

Your performance will probably vary from this laboratory measurement.



The graph clearly shows that the percentage improvement goes up as more rows are fetched per Fetch call.

With 1 row fetch, V8 CPU is 6% higher than V7.

However, with 2 row fetch, V8 becomes faster by 6%.

Beyond 100 rows, about 50% improvement continues.

Similarly for elapsed time and class 2 CPU time.

The measurement shown is for a very simple fetch via table space scan fetching 20 columns

Less %improvement for more complex Fetch involving join, sort, index access, more than 20 column fetch

More %improvement for less than 20 column fetch

Here is my rule of thumb for best practice. Use multirow fetch on any application that may be performance sensitive if the number of rows fetched is at least 10. If the number of rows is less than 100, then fetch those rows in a single fetch. If the number of rows can be very large, then do multiple fetches of 100 rows.

## Multi-row Fetch - continued

- **FETCH NEXT ROWSET FROM cursor FOR N ROWS INTO hva1, hva2, hva3**
- **Up to 50% CPU time reduction by avoiding API (Application Programming Interface) overhead for each row fetch (100 rows)**
  - ▶ **% improvement lower if more columns and/or fewer rows fetched per call**
    - **Higher improvement if accounting class 2 on, CICS without OTE, many rows, few columns**
  - ▶ **See later foils for distributed**

When using multi-row fetch operations, available in NFM, you normally fetch a set of rows, or a rowset in a single fetch operation, as shown here in the case of 10 rows:

```
FETCH NEXT ROWSET FROM my-cursor FOR 10 ROWS
INTO :hva1, :hva2, :hva3
```

Since DB2 now fetches 10 rows in a single Application Program Interface (API) crossing (going from the application to DB2 and back), instead of 10 API crossings, one for each row fetched without rowsets, multi-row fetch reduces the multiple trips between the application and the database engine. This reduction produces even more benefits in distributed applications.

Chapter 3.1 of DB2 V8 Performance Topics discusses the numbers in detail.

## Minimize SQL Calls to Reduce API Overhead

- **Filter out unnecessary rows by adding predicates rather than by application program checking**
- **Use DB2 column functions rather than application program code**
- **Example: find how many employees make more than \$10,000/month**
  - 1 **Select, fetching all 100000 employee rows**
  - 2 **Select Where Salary>10000, fetching 1000 rows**
  - 3 **Select Count Where ..., fetching 1 row**
    - **100 times CPU time reduction possible from API elimination**
    - **Watch out for VSAM programmers, IO modules (stage 3 predicates)**

Bonnie Baker would call the first bullet using stage 3 predicates. So if your application programmers are using VSAM techniques or generalized IO modules, then you may see an extra 5 x to 10 x CPU time. In the worst cases, the additional CPU time can be 100 times more than using application programming techniques.

Fetch First N Rows Only in V7, in subquery DB2 9

Limits the number of rows fetched to avoid fetching unwanted rows

Singleton Select (or SELECT INTO) can be used with Fetch First 1 Row even if multiple rows qualify. This technique avoids -811 SQLCODE. V8 supports ORDER BY for more meaningful query. Bigger improvement is possible for CICS attach.

UPDATE without cursor is more efficient than OPEN, FETCH, cursor UPDATE, CLOSE. Up to 30% (possibly more if CICS) CPU time saving possible from singleton Select or Update compared to cursor operation

Reducing #SQL calls improves API path length and Processor MIPS for row processing. Up to 2 to 3 times processor MIPS improvement possible from high-speed processor cache hit by repeated execution of a small set of modules / instructions and reduction in data moves.

V8 multi-row operation can significantly reduce the number of SQL calls issued by up to 50% CPU reduction for simple (short-running) local Fetches, more for distributed.

## Minimize Columns and Host Variables Referenced in SQL Calls

- **Avoid unnecessary columns**
  - **Doubled CPU time possible with 100 additional columns/host variables**
  
- **Increasing order of cost**
  - **Local EBCDIC least -> ASCII or UNICODE or DRDA**  
 -> Single byte conversion -> Double byte conversion
  - **Integer/char least and date/time/timestamp most expensive**

Avoiding unnecessary columns for both input and output is important for high performance, especially with programs which return millions or billions of rows each night. The CPU cost roughly doubles for each extra 100 additional columns. Beware the SELECT \* used, when only a few columns are needed. If the application changes from using 20 parameter markers to 120 parameter markers, you can double the CPU time. If you use 500 parameter markers, then the CPU time can increase by 500%

Conversions are costly compared to simple movement. The highest costs are generally for date, time and timestamp.

## Minimize Predicates Evaluated

- Place most filtering predicates **first** in AND. (for predicates of the same type)

WHERE HOME_STATE='MONTANA'	FF= 1%
AND HAIR='BROWN'	FF=10%
AND SEX='MALE'	FF=50%

- Weighted average of 1.01 predicates evaluated
- If sequence of predicates is reversed, then the weighted average is 1.55, or 50% more predicate evaluation, which can lead to up to 20% CPU increase.
- Conversely, place most filtering predicates **last** in OR and IN-list without ACESSTYPE=N.  
 e.g. STATE IN ('NEW YORK','FLORIDA','MONTANA')

Place the most filtering predicates first in ANDed predicates.

Place the most filtering predicates last in OR and IN-list predicates.

These choices are generally less important than the numbers of rows and columns, but the difference in CPU time can be as much as 20%.

## Minimize SQL Statements in a Program Where Possible



```
DO ....  
  SELECT or INSERT or DELETE or UPDATE  
END  
  
instead of  
SELECT, INSERT, DELETE, or UPDATE  
SELECT, INSERT, DELETE, or UPDATE  
SELECT, INSERT, DELETE, or UPDATE
```

- Reduces EDM pool and thread storage
- Reduces allocate/deallocate cost at SQL execution and commit or deallocation
- Better exploitation of sequential detection and index lookaside
  - Potentially fewer Getpages, Lock requests, and faster I/O
- See Redpaper 4424

Large numbers of SQL statements in a package and large numbers of packages can increase the CPU time. Redpaper 4424 discusses two situations: when a small number of short-running SQL statements out of many SQL statements in a large package are executed and when a set of short-running SQL statements out of many different packages are executed. These two specific situations have become critical in some environments due to the performance difference after migration from DB2® V7 to V8. DB2 V8 has shown an increase in CPU associated with the major changes in functionalities. We show the measured CPU across V7, V8, and V9, highlight the improvement provided by V9, and provide some general tips on reducing CPU utilization by packages.

## LOB Improvements

- Progressive Streaming for LOB Locator Values
  - ▶ DB2 uses LOB size to determine whether to send LOB data to Java or DB2 CLI clients in one go (<32KB), in chunks (<1MB) or as LOB locator (>=1MB)
    - Transparent to application using LOB locators
- FETCH CONTINUE
  - ▶ Allows applications to retrieve LOB/XML data in pieces without the use of locators
- File reference variables
  - ▶ A file reference variable allows direct transfer of LOB data between DB2 and the file named in the variable
- Utility Changes
  - ▶ LOAD / Cross load LOB column lengths > 32KB supported
  - ▶ Logging for > 1GB LOBs
  - ▶ REORG LOB reclaim space
  - ▶ Online CHECK LOB and DATA
- Elimination of LOB locks for improved availability and performance

IDUG 2009 Europe

38

For Java and DB2 CLI programs that use locators with LOBs. Improves performance and less network traffic for LOBs that are less than 1MB Default behavior if using DB2 9 for z/OS Requires DB2 Connect 9.1 FP 1

No changes required to programs using locator values. DB2 Client and Type-4 driver manage progressive streaming of data to program. DB2 for z/OS determines whether to flow LOB values or Locators to client based on size thresholds for JDBC, SQLJ, and CLI. For small LOBs, (Default <= 32KB) the performance should approximate that of retrieving a VARCHAR column of comparable size. Medium size LOBs (Defaults > 32KB and <= 1MB) For large LOBs (Default over 1MB) locators are still used

Specific FETCH that contains LOB or XML columns used with programs that materialize LOBs. Application uses a buffer that might not be large enough to hold the entire LOB or XML value. If any of the fetched LOB or XML columns do not fit, DB2 returns information about truncated columns and the actual length.

Retrieve LOB or XML data in multiple pieces without use of locators. Must specify WITH CONTINUE on initial FETCH. Subsequent fetches use FETCH CURRENT CONTINUE Application must manage buffers & reassemble data. Not required to fetch entire object before moving to next SQLCA indicates whether data is truncated

LOAD / Cross load LOB column lengths > 32KB supported

Logging for > 1GB LOBs

REORG LOB reclaim space: SHRLEVEL(REFERENCE). Allows LOG NO.

Online CHECK LOB and DATA

Elimination of LOB locks: Now using LRSN & page latching for consistency checks

Prior to DB2 9, LOB locks were held until commit, Even for UR

Space search for LOB allocation: No LOB locks acquired for space search

Read LSN granularity improved to page level in LOB table space

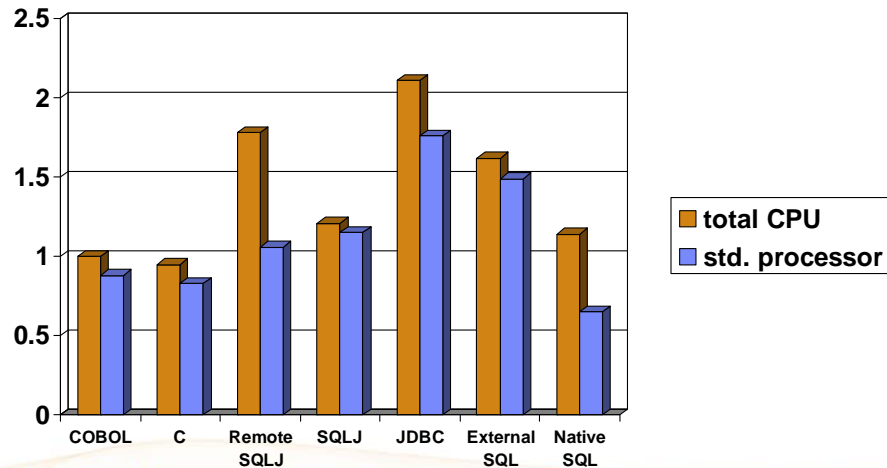
Improved availability & performance, particularly for UR readers

Requirements: NFM "Locking protocol 3" GBP changes Automatic in non-data sharing

Clean group-wide shutdown in data sharing once NFM enabled until PK62027

## Which language should I use?

Stored Procedures – relative cost of various languages with and without zIIP and zAAP for specific OLTP workload



IDUG'2009 Europe

39

Many considerations are involved in choosing a language, such as available skills and a fit between the task and the objectives. This chart uses the performance numbers from a presentation on programming language considerations for use in stored procedures by Todd Munk and Gopal Krishnan.

The primary message from these performance measurements is that performance is not a major factor in choice of language, skills in a language and applicability for the purpose can make a bigger difference than the choice of language. These measurements are for a single, specific workload which may be very different from yours.

The taller, orange bars show the total CPU used for an example workload, relatively light transactions while comparing various languages. The shorter blue bars show the amount of general purpose CPU used for remote stored procedures, removing the time which is redirected to zIIP and zAAP.

See the numbers and discussion on the next chart.

## Which language should I use? ...

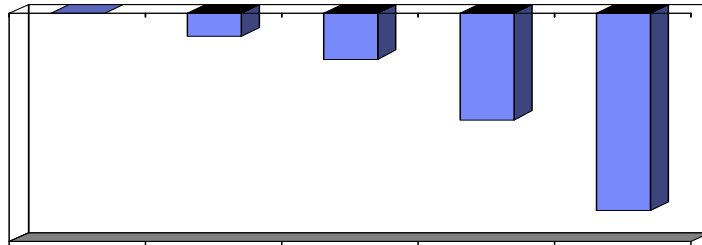
Stored Procedures - Performance of different languages with and without zIIP and zAAP for specific OLTP workload

Language	Base CPU	GP CPU after zIIP and/or zAAP acceleration
COBOL stored proc	1X (BASE)	.88x
C stored proc	.95x	.83x
Remote SQLJ	1.78x	1.06x
SQLJ stored proc	1.21x	1.15x (zIIP + zAAP)
JDBC stored proc	2.11x	1.76x (zIIP + zAAP)
External SQL stored proc	1.62x	1.49x
Native SQL stored proc	1.14x	.65x

For this specific workload, COBOL and C have the lowest base CPU time. All of the other CPU times are calculated relative to the COBOL time.

When zIIP and zAAP time are removed, the general purpose CPU time is smallest for native SQL stored procedures. COBOL, C and external SQL stored procedures are also reduced by about 12% for the CALL, COMMIT, and stored procedure result set processing. Remote SQLJ is reduced more, and the Java options have reductions for both zIIP and zAAP.

DB2 X preliminary performance plan → significant CPU reductions, best with latest processors



8 use	9 CM	9 use	X CM	X use
	Transactions		DB design adjustments	
	Batch		Hash access	
	REBIND		application changes	
			SQL adjustments	

Your situation will vary. Less CPU is better.

Processors z10, z9, z890, z990 & later      z/OS 1.10 & later

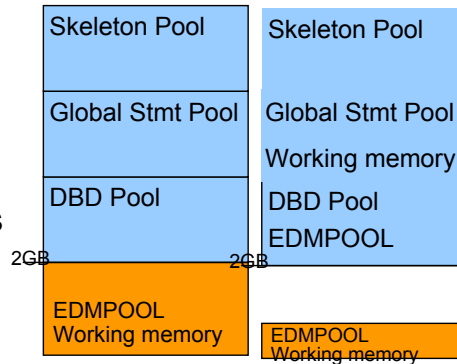
Reducing CPU from DB2 9 to DB2 X without significant administration or application changes is the primary thrust of the performance work.

This work is very preliminary, but the performance plan for DB2 X is much more aggressive than in any recent version. The last version which contained significant improvements for reducing CPU time in transactions and batch was Version 2 in 1988. Versions 3 to 9 made improvements in queries and in utility CPU time and provided many scalability improvements, but little reduction in transaction CPU time. We expect DB2 X to run only on z10, z9, z890, z990, and later processors, and to provide CPU reductions from the beginning, with improvements in CM, but more dramatic reductions for applications that can take advantage of the improvements in application design.

## 64 bit Evolution (Virtual Storage Relief)

**Virtual storage constraint is still an important issue for many DB2 customers.**

- DB2 9 helps (10% – 15%)
- DB2 X expects to move 80% - 90%+
  - ▶ More concurrent work
  - ▶ Reduce need to monitor
  - ▶ Able to consolidate LPARs
  - ▶ Reduced cost
  - ▶ Easier to manage
  - ▶ Easier to grow



The DB2 9 virtual storage objective was 10-15% relief. The DB2 X target is 80% to 90% of the DBM1 address space. We expect the result to be the ability to run much more concurrent work, with an early guess of 3 to 5 times more threads.

Storage monitoring should be drastically reduced. Customers are consolidating LPARs. Sometimes they need to have more than one DB2 subsystem on an LPAR, costing real storage and CPU. With these changes, work can run in one DB2 subsystem, rather than needing more members.

The net for this change is expected to be reduced cost, improved productivity, easier management, and the ability to grow DB2 use much more easily.

## Performance Sources and Resources

RTFW

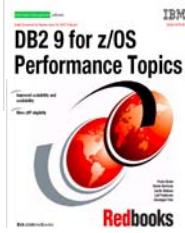

RTFW is the acronym for Read the Friendly Web. Let's take a short walk on the wild, wild web. The problem with the web is not too little information, but rather too much information. The experience is a bit like trying to take a drink from a fire hose. So I'd like to help a bit by narrowing the search with the twenty five cent tour of a few of my favorite DB2 web sites. A lot more information has been added in the past month or two, with many new books and web pages.

Let's start with the DB2 family. I'll generally show the short form or alias of the URL, omitting http://

Here are some tips for avoiding the 404. You don't need www in front of ibm.com in most situations. There is often something after www, such as the -306 in www-306 when you get the URL back from the browser. Remove the hyphen and number when you save the URL, since that number changes more quickly than the rest of the URL.

Get updated books October 30, 2009  
<http://publib.boulder.ibm.com/infocenter/imzic/>

- ✓ Performance Guide
- ✓ Redbooks
- ✓ Administration Guide
- ✓ Data Sharing: Planning and Administration
- ✓ Utility Guide and Reference
- ✓ Installation Guide

IDUG '2009 Europe

44

For performance, you need many books. Some are optional, for example the data sharing book is not needed if you don't use data sharing. You can get most of the books from the DB2 Technical References web page. The books were updated in December 2007 and February, March, June, and August 2008, with some coming later, so get the latest ones. Some of the Redbooks are important. You need books from the z/OS Library as well.

<http://www.ibm.com/support/docview.wss?rs=64&uid=swg27011656>

<http://www.ibm.com/support/docview.wss?rs=64&uid=swg27011658>

<http://www.ibm.com/systems/z/os/zos/bkserv/r9pdf/>

Be sure to use the latest information to save time and problems. Some of the IBM Redbooks publications have always been updated and added lately (next page).

## DB2 9 in IBM Redbooks Publications



1. DB2 9 Technical Overview SG24-7330
2. DB2 9 Performance Topics SG24-7473 another update soon
3. DB2 9 Stored Procedures SG24-7604
4. Index Compression with DB2 9 for z/OS redp4345
5. SQL Reference for Cross-Platform Development
6. Enterprise Database Warehouse, SG24-7637
7. 50 TB Data Warehouse on System z, SG24-7674
8. New Tools for Query Optimization SG24-7421
9. LOBs with DB2 for z/OS SG24-7270
10. Deploying SOA Solutions SG24-7663
11. Enhancing SAP - DB2 9 SG24-7239
12. SAP Application on Linux z SG24-6847
13. Best practices SAP BI - DB2 9 SG24-6489-01
14. Data Sharing in a Nutshell, SG24-7322
15. Securing DB2 & MLS z/OS SG24-6480-01
16. Data Sharing: Distributed Load Balancing & Fault Tolerant Configuration redp4449
17. Considerations on Small & Large Packages redp4424
18. Backup and Recovery Considerations redp4452
19. Powering SOA with IBM Data Servers SG24-7259
20. Packages Revisited, SG24-7688
21. Data Studio V2.1 Web Services redp4510
22. Ready to Access Solid-State Drives redp4537
23. Distributed Functions SG24-6952
24. Buffer Pool Monitoring & Tuning redp4604
25. Securing & Auditing Data SG24-7720
26. Serialization and Concurrency SG24-4725-01 coming soon



IDUG 2009 Europe

45

DB2 library more information <http://www.ibm.com/software/data/db2/zos/library.html>

Many IBM Redbooks publications, Redpapers and one cross-platform book on DB2 9 are published, in addition to the standard library, with more in the works. Check for updates.

<http://www.redbooks.ibm.com/cgi-bin/searchsite.cgi?query=db2+AND+9+AND+for+AND+z/os>

1. DB2 9 Technical Overview, SG24-7330 <http://www.redbooks.ibm.com/abstracts/SG247330.html>
2. DB2 9 Performance Topics, SG24-7473, <http://www.redbooks.ibm.com/abstracts/SG247473.html>
3. DB2 9 Stored Procedures, SG24-7604, <http://www.redbooks.ibm.com/abstracts/SG247604.html>
4. Index Compression DB2 9, REDP4345, <http://www.redbooks.ibm.com/abstracts/redp4345.html>
5. Deploying SOA Solutions SG24-7663, <http://www.redbooks.ibm.com/abstracts/SG247259.html>
6. Cross-Platform Development Version 3, <http://www.ibm.com/developerworks/db2/library/techarticle/0206sqlref/0206sqlref.html>  
[ftp://ftp.software.ibm.com/ps/products/db2/info/xplatsql/pdf/en\\_US/cpsqlrv3.pdf](ftp://ftp.software.ibm.com/ps/products/db2/info/xplatsql/pdf/en_US/cpsqlrv3.pdf)
7. Enterprise Data Warehousing, SG24-7637, <http://www.redbooks.ibm.com/abstracts/sg247637.html>
8. LOBs: Stronger & Faster SG24-7270, <http://www.redbooks.ibm.com/abstracts/SG247270.html>
9. Securing DB2 & MLS z/OS, SG24-6480-01, <http://www.redbooks.ibm.com/abstracts/sg246480.html>
10. Enhancing SAP, SG24-7239, <http://www.redbooks.ibm.com/abstracts/SG247239.html>
11. Best practices SAP BI, SG24-6489-01, <http://www.redbooks.ibm.com/abstracts/sg246489.html>
12. New Tools for Query Optimization, SG24-7421, <http://www.redbooks.ibm.com/abstracts/sg247421.html>
13. Data Sharing in a Nutshell, SG24-7322, <http://www.redbooks.ibm.com/abstracts/sg247421.html>
14. DB2 9 for z/OS Data Sharing: Distributed Load Balancing and Fault Tolerant Configuration <http://www.redbooks.ibm.com/abstracts/redp4449.html>
15. Considerations on Small and Large Packages redp4424 <http://www.redbooks.ibm.com/abstracts/redp4424.html>
16. Backup and Recovery Considerations redp4452 <http://www.redbooks.ibm.com/abstracts/redp4452.html>
17. Powering SOA IBM Data Servers, SG24-7259 <http://www.redbooks.ibm.com/abstracts/SG247259.html>
18. Packages Revisited, SG24-7688 <http://www.redbooks.ibm.com/abstracts/SG247688.html>
19. 50 TB Data Warehouse Benchmark on IBM System z <http://www.redbooks.ibm.com/redpieces/abstracts/sg247674.html>
20. SAP on DB2 9 for z/OS: Implementing Application Servers on Linux for System z <http://www.redbooks.ibm.com/redpieces/abstracts/sg246847.html>
21. IBM Data Studio V2.1: Getting Started with Web Services on DB2 for z/OS <http://www.redbooks.ibm.com/redpieces/abstracts/redp4510.html>
22. Ready to Access DB2 for z/OS Data on Solid-State Drives <http://www.redbooks.ibm.com/abstracts/redp4537.html>
23. Parallel Sysplex Operational Scenarios <http://www.redbooks.ibm.com/redpieces/abstracts/sg242079.html>
24. Distributed Architecture <http://www.redbooks.ibm.com/redpieces/abstracts/sg246952.html>
25. Buffer Pool Monitoring & Tuning <http://www.redbooks.ibm.com/redpieces/abstracts/redp4604.html>
26. Securing and Auditing Data <http://www.redbooks.ibm.com/redpieces/abstracts/sg247720.html>
27. Watch for new books on DB2 serialization & concurrency, SG24-4725-01; utilities

Installable option

<http://www.ibm.com/support/docview.wss?rs=865&uid=pub1sk5t737700>  
<http://publib.boulder.ibm.com/infocenter/imzic/>

This is the Information Center, with a wide spectrum of information and access to books for DB2 for z/OS, DB2 tools, QMF, IMS, IMS tools and more. You can get to this page from the Library page, by clicking Information Center. The Information Center provides information across the books and across multiple products.

If you click “Troubleshooting and Support”, then expand under “Searching knowledge base” and click “Web search:...”, you’ll find a helpful Web search page. From this page, you can search IBM support, DeveloperWorks, or even the whole Internet using Google.

The latest change in this area is an installable Information Center, so that you can use the facility even when the internet is not accessible.

This is the main DB2 for z/OS web page. You can get to the other DB2 for z/OS pages from here, so I often call this my home page. This page changes frequently, so look at the highlighted NEW items. Do you want to look in a DB2 book? Click on Library to see books on DB2 and QMF Version 8 (about 40), Version 7, 6 or even 5. V6 and V5 are out of service. You can check the latest changes by looking at the Information Updates or go to the Information Center. From this page, you can look for conferences (Events), specific classes (Education), or services. If you want to see the latest on DB2 9 or DB2 Version 8, click on the DB2 9 or the V8 link. If your primary concern is application development, the Developer Domain is for you. DB2 Magazine covers a broad range of topics about DB2. The latest machines System z9, z990 and z890 are on the System z page. Click DB2 and IMS Tools to see the wide range of help we provide.

## Important Disclaimer

**THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY.**

**WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.**

**IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE.**

**IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION.**

**NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, OR SHALL HAVE THE EFFECT OF:**

- CREATING ANY WARRANTY OR REPRESENTATION FROM IBM (OR ITS AFFILIATES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS); OR
- ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice. Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in the operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only. This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

**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, AS/400, DATABASE 2, DB2, e-business logo, Enterprise Storage Server, ESCON, FICON, OS/400, Netfinity, RISC, RISC SYSTEM/6000, System i, System p, System x, IBM, Lotus, NOTES, WebSphere, z/Architecture, z/OS, System z,**

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

Session: B06



DB2 for z/OS Selected Performance Topics

# Questions?

Roger Miller

IBM Silicon Valley Lab

millerrl@us.ibm.com



IDUG'2009 Europe 49

Thanks for coming.

Title – DB2 for z/OS technical evangelist, strategist, architect, designer, developer, writer, service, DB2 factotum (from the Latin for does everything or jack of all DB2 trades, master of several).

Current Projects - Roger is working to roll out DB2 9 for z/OS, to design the next improvements in DB2.

Technical accomplishments - Roger Miller is a DB2 for z/OS technical evangelist, architect and designer who worked on many facets of DB2, ranging from overall design issues to SQL, languages, install, security, audit, standards, performance, concurrency, and availability. He has worked for 30 years on DB2 development, product design and strategy. He often helps customers to use the product, answers many questions and presents frequently to user groups.

Fun facts - Roger likes hiking, bicycling, reading, Shakespeare, Yosemite and bears. He has learned to like working out with a personal trainer.