Platform: DB2 UDB for z/OS

Performance Considerations in Migrating to DB2 UDB for z/OS V8

Akira Shibamiya
IBM Silicon Valley Laboratory

Session: SL1
May 26, 2005 – 12:30 pm to 1:40 pm

IDUG® 2005 – North America
Where Business & Data Converge

Notes

- **Abstract: This presentation covers the performance considerations, such as catalog migration time and a potential change in the use of CPU time, I/O time, and virtual/real storage as a DB2 for z/OS subsystem is migrated to V8 compatibility mode and then to new function mode. It also provides performance monitoring and tuning tips on CPU and virtual storage usage of applications migrating to V8.**
- **Speaker: Akira Shibamiya, IBM Silicon Valley Laboratory, shibamiy@us.ibm.com**

Acknowledgment and Disclaimer

- **Measurement data included in this presentation are obtained by the members of the DB2 performance department at the IBM Silicon Valley Laboratory.**
- **The materials in this presentation are subject to**
  - **Enhancements at some future date,**
  - **A new release of DB2, or**
  - **A Programming Temporary Fix**
- **The information contained in this presentation has not been submitted to any formal IBM review and is distributed on an "As Is" basis without any warranty either expressed or implied. The use of this information is a customer responsibility.**

IDUG® 2005 – North America
Where Business & Data Converge

Five Bullet Points

- **Catalog migration performance**
- **Change in CPU usage and reporting**
- **Change in DBM1 address space virtual storage usage**
- **Change in real storage usage**
- **Performance monitoring and tuning tips**

Catalog Migration

- 3 possible migrations to V8
  - V5 to V7 to V8
    - V5 end of service 12/02
  - V6 to V7 to V8
    - V6 end of marketing 6/02, end of service 6/05
  - V7 to V8
- V6 to V7
  - Laboratory measurements show 10 to 100 seconds, depending on the size of catalog/directory (medium to large) and DASD/processor model
    - Large if >5GB, small if <0.5GB, medium in between, based on V7 customer catalog survey in the last 2 years
- V5 to V7
  - 2 to 20 minutes

IDUG® 2005 – North America
Where Business & Data Converge

Catalog Migration - continued

- V7 to V8 Compatibility Mode (CM)
  - 0.2 to 10 minutes, depending on the size of catalog/directory (medium to large)
    - Time heavily depends on DASD and channel model used also
  - No significant change in catalog size from V7 to V8 Compatibility Mode
  - Compatibility Mode
    - DB2 catalog in EBCDIC
    - Fallback to V7 allowed
    - Check for type 1 index
      - If any found, catalog migration rolled back

Catalog Migration -continued

- V8 Compatibility Mode to New Function Mode (NFM)
  - 0.1 to 2 hours depending on the size of catalog/directory (medium to large)
  - 1 to 10% increase in the size of catalog for both data and index
  - New Function Mode
    - DB2 catalog in unicode
    - No fallback allowed

Catalog Migration - continued

- Online Reorg Sharelevel Reference of SPT01 and 17 catalog tablespaces the most time-consuming
- Very Rough Rule-of-Thumb on estimating the time for medium to large catalog/directory = 6min + 5min/GB of SPT01, SYSPKAGE, SYSDBASE, etc.
  - Example: If 10GB SPT01, SYSPKAGE, SYSDBASE, then (6+5x10) = roughly 1 hour
  - Most catalogs are smaller than 10GB and thus faster
- Time also depends on DASD and channel model used
- Catalog reorg can make catalog migration faster, especially if no reorg for a long time

Major Performance Highlight of V8

- 10 to 100 times improvement possible from
  - Materialized Query Table
  - Stage 1 and indexable predicate for unlike data types
  - Distribution statistics on non-indexed columns
  - Other access path selection enhancements
- 2 to 5 times improvement possible from
  - Multi-row Fetch, cursor Update/Delete, Insert
  - Star join with work file index, in-memory work file, more parallelism
  - DBM1 virtual storage constraint relief
  - Partition Load/Reorg/Rebuild with DPSI

IDUG® 2005 – North America
Where Business & Data Converge

For applications not taking advantage of V8 performance features

- Some CPU time increase is expected in order to support a dramatic improvement in user productivity, availability, scalability, portability, family consistency,..
  - DBM1 virtual storage constraint relief with 64bit instructions
  - Long names, long index keys
  - Longer and more complex SQL statements
- I/O time
  - No change for 4K page I/O
  - Significant sequential I/O time improvement possible for 8K, 16K, or 32K page because of bigger Vsam Control Interval size
    - Up to 70% i/o data rate (MB/sec) improvement
    - Also Vsam i/o striping now supported
  - V8 PQ99608 2/05 to fix excessive log write i/o's

CPU change based on laboratory measurements with

no application nor aggressive configuration/environment change

- '+' means cpu increase, '-' means reduction, compared to V7

- -5 to +15% online transaction
- -10 to +10% online transaction in data sharing (NFM)
- -5 to +20% batch
  - -5 to +5% insert
  - -5 to +20% select
  - +5 to 20% fetch, update
- -10 to +15% batch data sharing (NFM)
- -20 to +15% batch DRDA
- -5 to +5% utility
- -20 to +15% query

➲ Numbers subject to change as more data become available

IDUG® 2005 – North America
Where Business & Data Converge

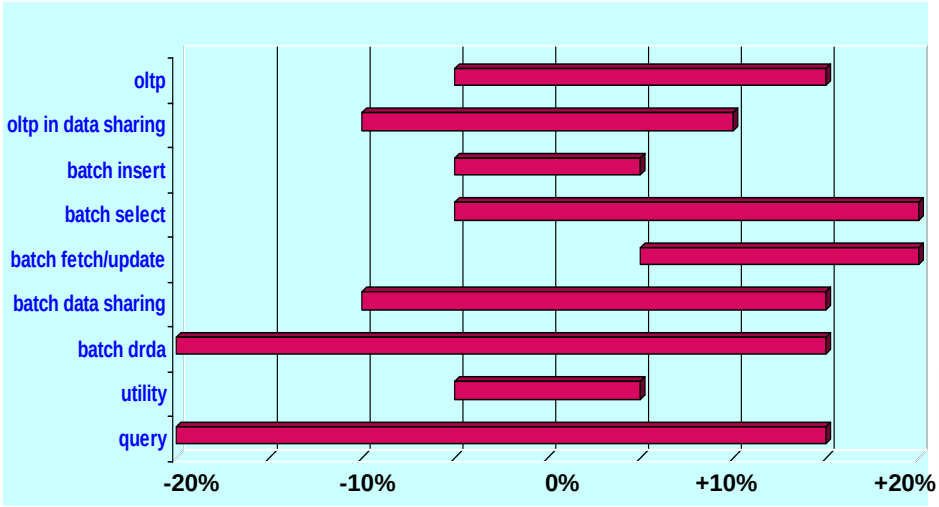
CPU change - continued

➲ Typically, no difference between CM and NFM except in data sharing for workloads with

- No application change
- No aggressive configuration/environment change

- Examples of what CM supports
  - Most access path selection enhancements
  - DBM1 virtual storage constraint relief
  - Lock avoidance in Select Into with CurrentData Yes, overflow rows
  - 180 CI limit removal in list prefetch and castout I/O
  - Long-term page fix option by buffer pool
  - SMF 89 performance enhancement (usage pricing)
  - Data sharing immediate write default change
  - Implicit multi-row operation in DRDA

CPU change - continued



13

V8 vs V7 OLTP measurements

- **ITR=Internal Throughput Ratio, inversely proportional to CPU**
- **One ERP vendor transaction -8% CPU 6/04**
- **IRWW data sharing +7% ITR 11/04**
- **IRWW non data sharing 0% ITR 11/04**
- **5 SVL client/server transaction workloads 0 to -12% ITR 9/04**

➲ **-5 to +15%, or -10 to +10% data sharing, typical CPU observed**

14

V8 vs V7 Query measurements

- Over 200 TPCD and other queries: 0% avg cpu, -20% total cpu 8/04
  - Some queries benefit from improved access path selection
- 160 BW queries: -30 to -40% cpu and elapsed time because of star join enhancement 6/04 (NFM)

➲ -20 to +15% typical CPU observed

IDUG® 2005 – North America
Where Business & Data Converge

V8 vs V7 Batch CPU measurements

- SVL batch 20 column rows 7/04
  - +12% Fetch/Update
  - +5% Fetch/Delete
  - -6% Insert
- IBM Japan batch at SVL 11/04, both CD YES and NO
  - +10 to 15% Fetch loop
  - -4% (if CD YES) to +15% (if CD NO) Select loop
    - V8 PQ89070 8/04 for lock avoidance in Select Into with Cursor Stability and Currentdata Yes
  - +5% Select Max
  - +20% Fetch/Update or Delete loop
  - +2% Insert

➲ -5 to +20% typical CPU observed

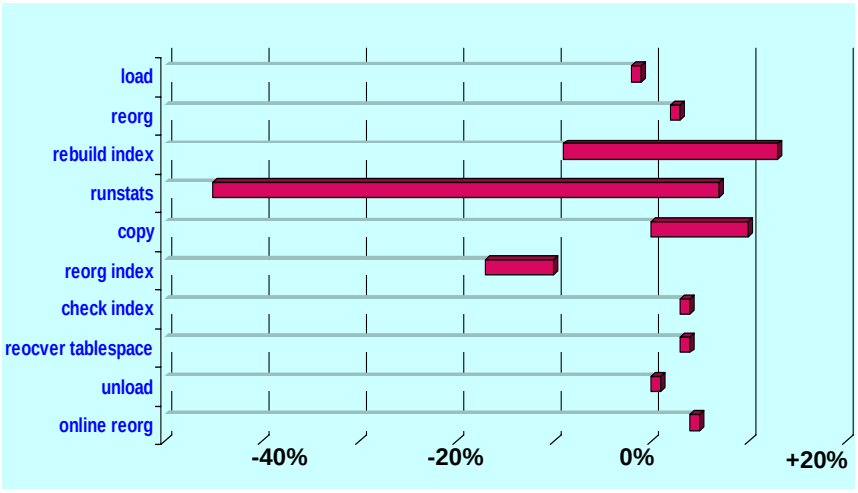
- Could be significantly better if data sharing or DRDA
- V8 multi-row can have a significant improvement here

V8 vs V7 Utility CPU measurements

- SVL Utility 9/04
  - Load -2%
  - Reorg +2%
  - Rebuild Index -9 to +13%
  - Runstats -45% to +7%
  - Copy 0 to +10%
  - Reorg Index -10 to -17%
  - Check Index +3%
  - Recover Tablespace +3%
  - Unload 0%
  - Online Reorg +4%

➲ -5 to +5% typical CPU observed

Utility CPU - continued



Tuning for CPU usage in V8

- If DBM1 virtual storage constrained in V7, recheck various actions taken to reduce virtual storage usage at the cost of additional CPU time, such as
  - Reduced size of buffer pools and other pools
  - Bind option release commit and/or no thread reuse to reduce thread and EDM storage size
  - EDM best fit to reduce EDM pool size
  - MINSTOR and CONTSTOR to reduce thread storage size
  - Lower DSMAX to reduce storage for data set control blocks and compression dictionary

Notes

- Performance enhancements introduced to compensate for increased CPU time to support new V8 functions are described next.

Long-term page fix option for buffer pool (BP) with frequent I/O's

- DB2 BPs have always been strongly recommended to be backed up 100% by real storage
  - To avoid paging which occurs even if only one buffer short of real storage because of Least-Recently-Used buffer steal algorithm
  - Given 100% real storage, might as well page fix all buffers just once to avoid the cost of page fix and free for each and every i/o
- Up to 8% reduction in overall IRWW transaction cpu time
- New option: ALTER BPOOL(name) PGFIX(YES/NO)
- Recommended for BPs with high buffer i/o intensity = [pages read or written]/[number of buffers]

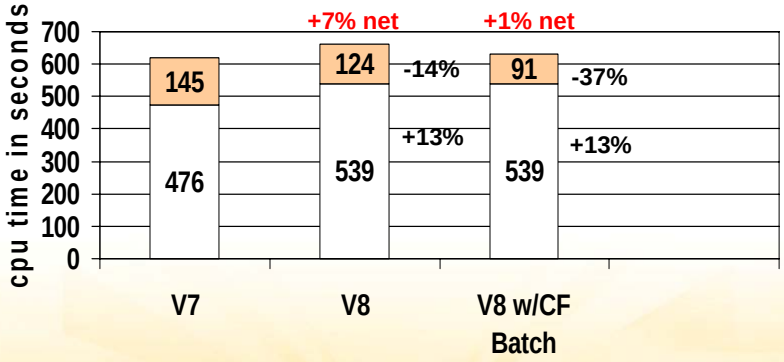
One example with 100,000 buffers total (400MB)

	#buffers	Pages read or written	Buffer i/o intensity
BP0 catalog/directory	2000	10000	5 #2
BP1 workfile	10000	20000	2 #3
BP2 in-memory index or data	30000	100	0.003
BP3 other index	53000	100000	1.9 #4
BP4 other data	5000	100000	20 #1

➲ BP4, 0, 1, 3 in that order. Don't bother with BP2

Batching of multiple CF write and castout read requests into one CF access with z/OS1.4 and CF level 12 7/04 (NFM)

Impact on Fetch/Update batch transaction



More Tuning for CPU usage in V8

- Rebind plans/packages
  - Better access path selection, especially beneficial for complex query
  - Enable SPROC (fast column processing) for 64bit
  - Take advantage of some ALTERed objects; for example
    - Matching index access or index-only after Alter Index Add Column
    - Index-only after Alter padded to non padded index

More CPU Tuning - continued

- PQ86071/89919 6/04 to remove 180 VSAM Control Interval limit in list prefetch and castout i/o
  - 2 to 3% overall CPU time reduction for IRWW
- Group-wide shutdown and restart to reduce global and false contentions for pageset/partition locks when release commit in data sharing (NFM)
  - -6% overall cpu for 2way data sharing IRWW

Notes

- Less regression possible if
  - IRLM PC YES and LOCKPART YES in V7
  - Hiperpool/Dataspace buffer pool used in V7
  - SMF 89 (usage pricing) active
    - CPU captured in class 1 but not class 2 acctg
  - DRDA, DSNTEP4, DSNTIAUL which can automatically exploit multi-row operation
    - Up to 75, 35, and 50% cpu reduction, respectively, compared to V7
  - Bind option release commit in data sharing
  - I/O-intensive workload
    - Long-term page fix and/or 180 CI limit removal help

More CPU Tuning - continued

- PQ89070 8/04 to avoid locks in Select Into in CS and Currentdata Yes
- PQ95867 12/04 for faster DB2 storage management

➲ Ultimately, aggressive exploitation of V8 performance features via application rewrite or configuration change, eg multi-row operation, can make V8 perform significantly better than V7 in affected areas.

Notes

- More regression possible if
  - Non padded index (default with V8 install) with small varchar columns
  - DPSI in some SQL calls
  - Many column processing
    - ➲ Consider Alter to less expensive column type
      - V5 Alter Varchar length, but Varchar no longer necessary to alter length

Caution on observed CPU time increase in V8

- In general, higher %cpu increase in acctg
  - But lower %cpu increase, or even reduction, in MSTR, DBM1, and IRLM address spaces possible
  - For lower %cpu increase overall
- Example of IRWW 11/04

	Non data sharing	Data sharing
Acctg	+4%	+14%
MSTR/ DBM1/ IRLM	-15%	-48%
Total	0%	-8%

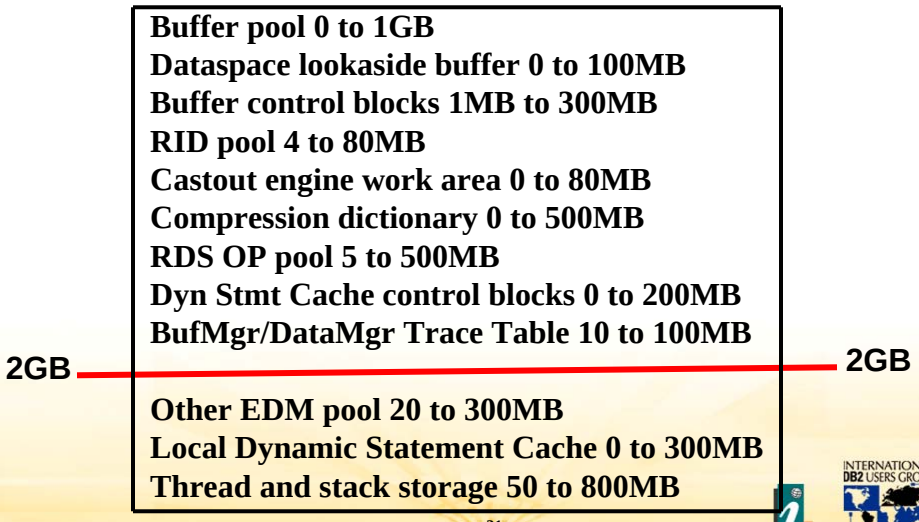
IDUG® 2005 – North America
Where Business & Data Converge

Notes

- Less DBM1 SRB time from
  - Long term page fix option especially in prefetch and write i/o
  - 180 VSAM Control Interval limit removal in list prefetch and castout i/o PQ86071/PQ89919 6/04
  - z/OS 1.4 Coupling Facility Level 12 batching of multiple CF write and castout read requests into 1 CF access
    - Bigger benefit for Insert/Update/Delete-intensive application
- Less MSTR SRB time from
  - Default immediate write change from NO(Phase2) to Phase1
    - MSTR SRB shifted to Acctg TCB time – no net change
- Less IRLM time from
  - Reduced global and false contention for pageset/partition locks when release commit in data sharing
    - Less need for release deallocate bind option

V8 Virtual Storage Usage in DBM1 Address Space

'typical' V7 below 2GB storage usage shown



Notes

- **LOB, global dynamic statement cache, and dataspace buffer pool are not shown here as these can be in dataspace rather than below 2GB DBM1 address space in V7.**
  - **These are all in above 2GB DBM1 address space in V8.**

Virtual storage - continued

- DBM1 virtual storage constraint relief improves scalability of performance
  - As the processor power continues to grow, linear scalability, or ability to exploit increasing processor power without encountering a bottleneck which prevents the full CPU usage, becomes more important.
  - Bigger buffer pool and cache to reduce i/o bottleneck and CPU overhead
  - ➲ Without 64bit support, it was difficult to exploit more than 20GB of real storage
    - Up to 32GB on z800, 64GB on z900, 256GB on z990
  - CPU-Storage trade-off
- However, thread storage is still below 2GB in V8
  - Hence, maximum number of threads supported, such as CTHREAD (2000) and MAXDBAT (1999), is not increased.

Notes

- Bigger DBM1 virtual storage constraint relief if V7 with bigger
  - Buffer pool below 2GB
  - Buffer control blocks for virtual pool, hiperpool, dataspace buffer pool
  - Dataspace buffer pool lookaside buffer
  - Compression dictionary
  - Castout buffers
  - RID pool, sort pool (in RDS Op pool)
  - Data Manager/Buffer Manager trace table
  - Dynamic statement cache control block
  - IRLM PC=NO
- Less relief from more
  - User and system thread storage with associated stack
  - Local dynamic statement cache

Estimation of V8 Below 2GB DBM1 Use, based on V7 Stats

- The following average estimates are very preliminary and subject to change as more customer data become available
  - Thread storage: 0 to +90% (0 for Image Copy utility, +40% for system, +40 to 90% for user thread)
  - Stack storage: +100%
  - Dynamic statement cache: +60%
  - EDM pool: roughly the same
  - RID pool: -90%
  - Others: -100%
- V7 PQ99658 Storage stats in class 1

Notes

- For a fair comparison, use the same pool size, # of threads, etc. instead of the defaults which may have changed.
- Bigger thread/stack storage in V8 for
  - Long names, keys, statements, other new functions
  - A portion of RDS op pool for dynamic SQL
  - More system agents

➲ How much more room in DBM1 address space below 2GB depends on % of storage used for threads, stacks, and local DSC versus others

Customer 1 (Europe 1/05)	V7 measured	V8 estimated
Virtual Pool	0 MB	0 MB
Buffer control block	78	0
Dataspace lookaside buffer	5	0
Castout buffer	38	0
EDM pool	118	118
Compression dictionary	340	0
1170 system agents	73	103
25 user threads	18	35
RDS OP pool	29	0
RID pool	92	10
DSC control block	53	0
Trace table	15	0
Stack storage	49	98
TOTAL DBM1 below 2GB	908MB	364MB

IDUG® 2005 – North America
Where Business & Data Converge

Notes

- Negligible local dynamic statement cache
- No virtual buffer pool as dataspace buffer pool is used instead.
- Assumes
  - Same number of system agents in V8
  - 90% increase in user thread storage

➲ Good DBM1 virtual storage constraint relief for this customer
  ➲ Even though dataspace buffer pool was used exclusively
  ➲ Because of large compression dictionary and small thread/stack storage

Customer 2 (US 1/05)	V7 measured	V8 estimated
Virtual Pool	98 MB	0 MB
Buffer control block	13	0
Dataspace lookaside buffer	6	0
Castout buffer	17	0
EDM pool	71	71
Compression dictionary	51	0
837 system agents	64	90
493 user threads	291	553
RDS OP pool	420	0
Dynamic Statement Cache	41	66
DSC control block	42	0
Trace table	36	0
Stack storage	143	286
TOTAL DBM1 below 2GB	1293MB	1066MB

Notes

- Negligible RID pool
- Both virtual buffer pool and dataspace buffer pool used here
- Large RDS OP pool due to lots of concurrent sort
- Assumes
  - Same number of system agents in V8
  - 90% increase in user thread storage

➲ DBM1 virtual storage constraint relief not as good as the Customer 1 because of large thread/stack storage

Virtual Storage-Related Tuning

- **Bind option release commit vs deallocate**
  - **Commit releases pageset/partition locks, RDS sections, and storages sooner, resulting in better concurrency and less DBM1 virtual storage usage**
    - **Recommended default**
  - **Deallocate holds on to these resources longer, resulting in possibly less CPU time (0 to 20%)**
    - **Recommended only for frequently-executed, high-volume, and performance-sensitive packages or plans**
    - **DB2PM/PE Acctg Report Short shows a list of pkgs/plans executed along with #occurrences and avg# SQL statements, elapsed time, and cpu time.**

Notes

- **If BP size increase is considered,**
  - **Make sure there is sufficient real storage to back it up 100%.**
    - **For BP with 100,000 buffers as an example, if 99,999 real storage frames are available, then every I/O could result in paging because of Least Recently Used buffer steal algorithm.**
  - **Deferred write threshold (VDWT) may need to be reduced in order to avoid "hiccup effect" at checkpoint.**
    - **Eg 5% VDWT of much bigger BP can have many more updated pages to be written out at checkpoint.**
- **V8 deferred write threshold default change**
  - **Buffer pool threshold (DWT): 50% to 30%**
  - **Dataset threshold (VDWT): 10% to 5%**
- **If migrating from VP+HP configuration with new BP size = VP+HP size, adjust BP thresholds, which are based on VP but not HP, appropriately.**

Virtual Storage-Related Tuning - continued

- If necessary, reduce MAXKEEPD to reduce local DSC
  - Rely more on global DSC which is above 2GB
- V8 PQ96772 2/05 to move dynamic statement cache control blocks above 2GB
- CONTSTOR and/or MINSTOR to reduce thread storage, especially for >1MB per thread storage

Notes

- Extended CSA size could be reduced in V8 if IRLM PC=NO was used in V7, enabling additional DBM1 virtual storage available below 2GB
- V7/V8 PQ98043 1/05 to reduce excessive stack storage for inactive or disconnected threads with dynamic SQL
- Stay current with service.
- Watch Info APAR II10817.

Real Storage (RS) Usage

- From V1 R1 in 1985 to the present, real storage usage growing at about 20 to 30% yearly to support performance scalability
  - More and bigger buffer pools, other pools, threads, …
- V8 continues a similar trend
  - By removing bottlenecks which would have prevented the exploitation of bigger real storage
  - ➲ If everything under user control is kept constant, 1 to 20% increase in real storage typically observed
    - Less %increase for larger DB2 subsystem
    - Bigger %increase for smaller DB2 subsystem

Notes

- Without 64bit support, difficult to exploit more than 20GB of RS because of DBM1 virtual storage constraint
  - RS of up to 32GB on z800, 64GB on z900, and 256GB on z990
- Example of more real storage usage
  - Higher default and maximum buffer pool size, RID pool size, sort pool size, EDM pool size, …
  - Bigger and possibly more threads
  - Bigger modules, control blocks, internal working storage
  - More deferred write, castout, and GBP write engines (300->600 max each)
  - More parallelism enabled
    - Parallel sort for multiple tables in composite
    - Parallel multi-column merge join

Reference

- V8 manuals, especially Performance Monitoring and Tuning section of Administration Guide
- Redbooks at www.redbooks.ibm.com
  - DB2 UDB for z/OS V8 Technical Review SG24-6871
  - DB2 UDB for z/OS V8 Everything you ever wanted to know… SG24-6079
  - DB2 UDB for z/OS V8 Performance Topics SG24-6465
- More DB2 for z/OS information at www.ibm.com/software/db2zos
  - E-support (presentations and papers) at www.ibm.com/software/db2zos/support.html