IBM GLOBAL SERVICES

Z10

64 bit storage in DB2 for z/OS

Jay Yothers

IBM DB2 Information Management Technical Conference

Sept. 20-24, 2004

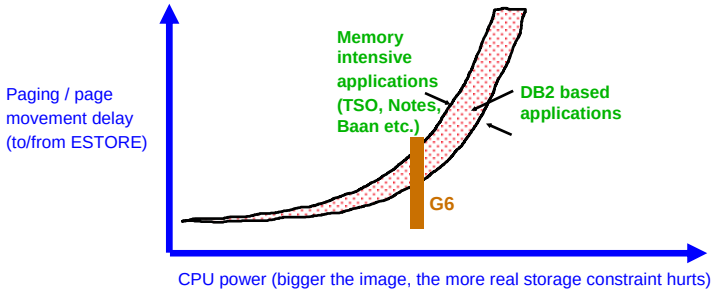
Las Vegas, NV

Agenda

- z/OS architecture -- where are we today?
  - ►Why we did it
  - ►z/OS architectural specifics
- Prior DB2 (pre Version 8) 64-bit architectural exploitation
  - ►A history of DB2 storage exploitation
  - ►DB2 exploiting 64-bit real storage
- DB2 Version 8 exploitation of 64-bit virtual support
- What else is moving above the 2-GB bar?

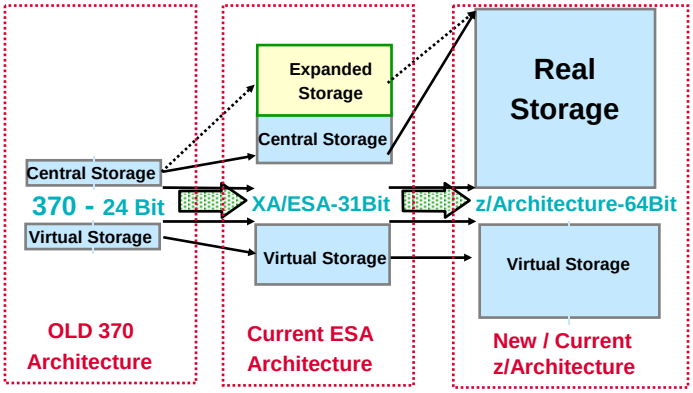
z/OS Architecture

Why a new 64-bit architecture? (the problem)



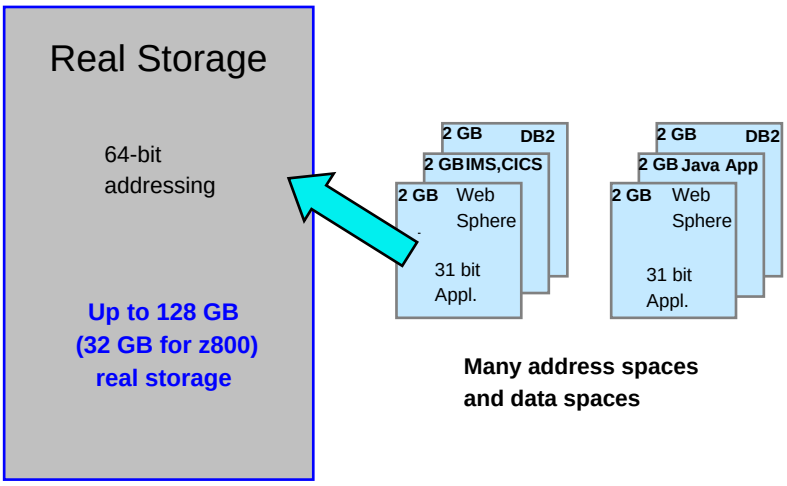
- Adding CPU capacity resulted in little or no additional real work being done
  - Paging overhead increased due to 2 GB (31-bit) central storage limit
- Expanded storage had provided an excellent interim solution
  - G5 / G6 implemented the enhanced MOVE PAGE instruction

64-bit memory architecture (the solution)



- V2R10 provided initial z/Architecture real addressing support:
  - Supporting 24-bit, 31-bit, and 64-bit applications
  - 128 GB (64 GB current max on z900) of central storage
  - 64-bit registers, etc...

Real storage support on z/Series (z900 / z800)



Product and application considerations

- 24- and 31-bit applications will run unchanged
- Few products or applications are affected
  - Products that depend on real addresses (LRA)
    - Usually middleware
    - Rarely used by customer applications
  - Performance reporting tools or capacity planning tools, etc.
    - Tools that need to monitor and report on the additional storage
- IBM and vendor product testing and production increasing all the time
  - Extensive testing done by IBM and ISVs prior to 64-bit real delivery
- Customers are now running 64-bit real and virtual in production today
  - Smooth migration reported by customers
  - See z/OS Migration web site for more information on 64-bit real exploitation
    http://www.ibm.com/servers/eserver/zseries/zos/installation/

IBM Software Group | DB2 Information Management Software

Prior DB2 (pre V8) 64-bit architectural exploitation

Increasing processor speeds

- Good news for DB2 customers
  - ►Drives business workloads to greater heights
- Problem
  - ►More workload requires more data
    - –Subsequently means more I/O
- In the past I/O latency improvements had NOT kept pace with CPU speeds:
  - ►I/Os become more and more precious
- Larger impact on general performance
- Solution
  - Larger buffer pools

Important I/O improvements:
Latest disk subsystems use large memory caches and parallel performance
ESS Model 800 sigificantly enhanced sequential I/O rate

Larger buffer pools

- Larger central storage became available
- Gain performance advantages
- Restrictions
  - ►Due to DBM1 VSTOR constraint, DB2 forces the maximum buffer pool sizes < machine memory (on a 2 GB central storage LPAR):
    - –Size of VPOOL limited to 1.6 GB (typically only 1 GB due to constraints)
    - –Frequently in the range 400-800 MB
- Problem
  - ►Can't increase virtual pools as much as needed to avoid I/O for many workloads
    - –Specifically those which repeatedly access same data elements and index keys
    - –Central storage was expensive
- Solution
  - ►Hiperpools (DB2 Version 3) in expanded storage

Hiperpools

- Buffers in hiper spaces (hiper pools) offer some relief
  - Expanded storage exploitation -- cheaper memory option
- Benefits
  - High performance data access
    - Larger amount of data in memory
    - I/O elimination -- large buffer pool without MVS paging
- Problems / issues
  - DB2 limits total size of hiperpool to 8 GB
  - Still requires a substantial virtual pool size for effective usage
    - Limits on size (8 GB max)
  - Contains ONLY clean pages -- NOT updated pages not yet written to disk
  - Page addressable, hence buffers must be in virtual pool before use
  - No direct I/O in and out of a hiper pool
- Solution
  - Data space buffer pools (DB2 Version 5)

Buffer pools in data spaces -- benefits

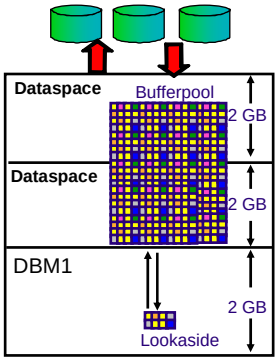
- Data spaces provided a good short term solution by exploiting 64-bit REAL
  - Buffer pools and statement caching in data spaces
    - Frees up space for other work in the DBM1 address space
    - Performance penalty when not 100% backed by real storage
- Advantages of data spaces over hiper pools
  - Read and write cache with direct I/O to data space
  - Byte addressability
  - Very large buffer pool sizes
    - 32 GB for 4K page size
    - 256 GB for 32K page size
- Single buffer pool can span multiple data spaces
- Excellent performance experienced with z900 and large processor storage
  - Performance dependent upon being in 64-bit REAL mode

Buffer pools in data spaces -- problems

- Undesirable as a long term solution due to size limitations
  - Still limited to 32 GB (for 4K buffer pools)
    - Imposed by 8 million PMBs in DBM1, therefore total storage practical for DB2 is around 20 GB total
- Overhead in copying buffers between data spaces and lookaside pool as they are accessed and updated
  - Internal lookaside pool resides in DBM1 - 10s MB

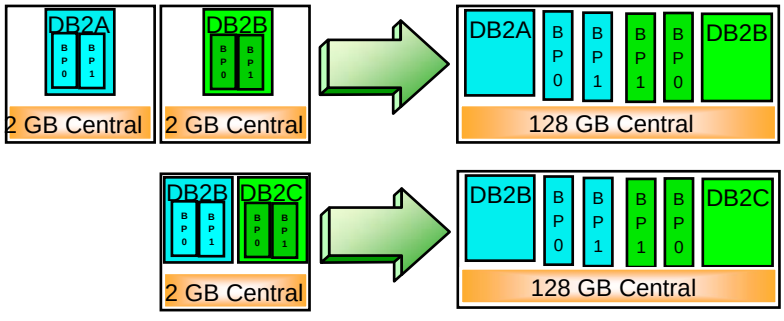
Scalability issue

- VSTOR limit of 2 GB for DBM1 address space is the biggest constraint to achieving linear scalability



Large real memory support

- For **all** versions of DB2 (no application code required)
  - Facilitates system consolidation activities
  - Allows multiple DB2 systems (single images or data sharing) on a single OS image without significant paging activity



- Larger CF structures for DB2 were also introduced with z900 GA2

Other thoughts

- Problems with both hiper pools and data space buffer pools
  - Still require work space and control blocks in the DBM1 address space
  - Require page movement (but still faster than I/O!)
    - Elapsed time
    - CPU time
  - Some amount of monitoring and tuning still required
- Business requirements to manipulate more data
  - Larger objects
    - Image, text, video (LOBs) use data spaces also
  - XML data

DBM1 virtual storage constraint relief

- DBM1 storage constraint - single biggest inhibitor to scaling workload on 31-bit machines will become even larger issue
  - z/Architecture and large 64-bit main memories take hold
  - Processor power keeps increasing
- What's caused the dramatic increase in DBM1 storage?
  - Larger workloads
  - New DB2 functionality
  - Larger REAL storage available on new processors
- z/Architecture and z/OS support for REAL storage > 2 GB
  - Problem made worse: reduced paging, faster CPUs promote larger workloads

Additional pressures on DBM1 storage

- DB2 doesn't get the whole 2 GB addressable virtual storage
  - Large chunks are common storage (CSA / ECSA, etc.) ==> a lot of storage
- EDM pool partly satisfied with dynamic statement caching in data space
- Thread storage
- VSAM data set control blocks
- Compression dictionaries
- More functionality
  - Larger code base therefore larger memory requirement

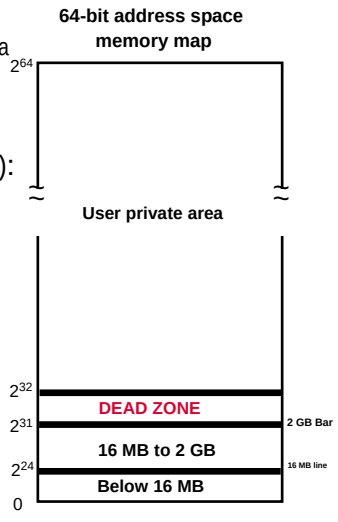
What would be the IDEAL solution?

- Increasing the 31-bit / 2 GB virtual storage limit
- Backing as much as possible with REAL storage

DB2 Version 8 exploitation of 64-bit virtual support

z/OS 64-bit virtual addressing support

- 64-bit virtual storage:
  - ►Applications and middleware can store and manipulate data above 2 GB
  - ►Dominant programming model in the future
  - ►z/OS 1.2 initiated 64-bit virtual support
- z/OS 64-bit Virtual Storage Roadmap (GM13-0076): http://www.ibm.com/servers/eserver/zseries/solutions/s390da/
- DB2 64-bit virtual exploitation in DB2 for z/OS Version 8 together with z/OS 1.3
- Importance of 64-bit VSTOR to DB2:
  - ►Enhanced data caching (performance)
  - ►Relieve virtual storage constraint (scale)
  - ►Ease of managing VSTOR
  - ►Avoidance of abends when users use a bit too much
- ISVs now starting to exploit 64-bit virtual http://www.ibm.com/servers/eserver/zseries/solutions/s390da/

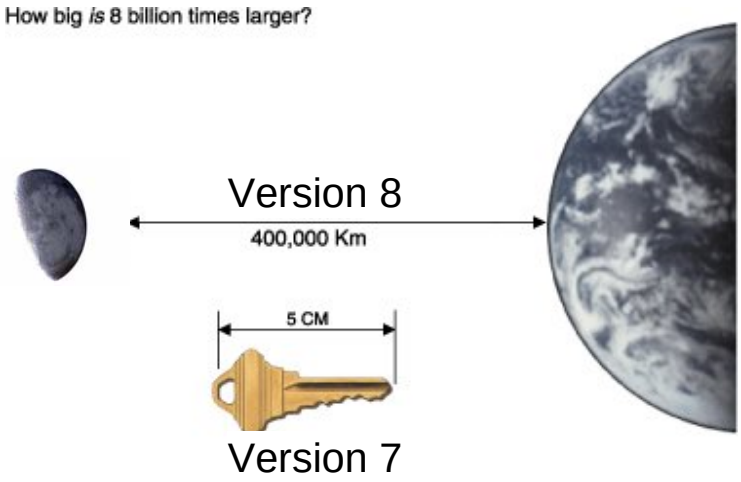


How big is 64 bit virtual?

Kilobyte K / KB	2^{10} Bytes	1024 Bytes
Megabyte M / MB	2^{20} Bytes	1,048,576 Bytes
Gigabyte G / GB	2^{30} Bytes	1,073,741,824 Bytes
Terabyte T / TB	2^{40} Bytes	1,099,511,627,776 Bytes
Petabyte P / PB	2^{50} Bytes	1,125,899,906,842,624 Bytes
Exabyte E / EB	2^{60} Bytes	1,152,921,504,606,846,976 Bytes

16 Exabytes = 18,446,744,073,709,400,000 Bytes!!

How much bigger is 8 billion times?



Why implement 64-bit VSTOR support?

- **Vertical scalability** is of significant importance to DB2:
  - Relieving virtual storage constraint will provide scalability
  - Maximum configuration based upon REAL storage available on processor
  - Monitor paging and ensure sufficient auxilliary storage available
- Enhanced data caching in memory will deliver higher performance
- Increase maximum buffer pool sizes
- Eliminating need for hiper pools and data spaces
- Simplifies DB2 systems management and operational tasks

DB2 benefits of 64-bit VSTOR support

- VSTOR relief utilizes 64-bit virtual addressing to move data areas above 2 GB bar in DBM1 address space:
  - Buffer pools
  - Page Manipulation Blocks
  - Castout buffers
- DB2's "data access" modules enhanced to access 64-bit addressable buffers "in place"
- NO internal data movement as per data space buffer pools today

Other 64-bit enhancements

- Additional DBM1 data areas now resident above the 2GB bar:
  - Sort pools
  - RID pools
  - Compression dictionaries
  - DBDs
- 64-bit serviceability enhancements:
  - 64-bit dump formatter
  - Continued IPCS support
- Locks now reside above the 2-GB bar
  - With IRLM V2.2 **"PC=YES"** forced - Extended Private Area

DB2 code considerations

- AMODE(64) callers not supported -- i.e., applications
- Exits called in 31-bit mode

64-bit virtual buffer pool support

- DB2 Version 8 is 64-bit exclusive
  - Buffer pools always allocated above 2-GB bar
  - Eliminates need for hiper pools and data space pools
  - Simplifying DB2 systems management and operational tasks
- Terminology
  - As of V8, terms **buffer pool** and **virtual pool** become "synonymous"
  - Data space pools, hiper pools no longer exist -- not supported
- Sizing and placement
  - Buffer pool max size is 1 TB
    - **The actual maximum = the REAL storage available**
  - Total buffer pool max size is 1 TB
  - PMBs moved above the bar
  - Castout buffers (data sharing) above the bar
- V7 buffer pool information saved for fallback

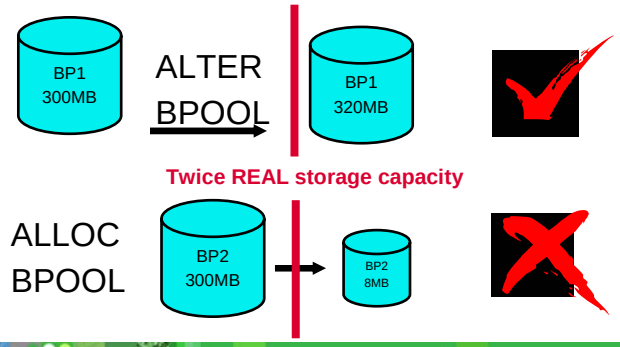
Limit changes

- Max number of read, write, castout engines increased to 600
- DB2 ensures 4 TB of virtual using MEMLIMIT keyword
- Additional enhancements:

Parameter	Value
BP0	Minimum 2000 (from 56) Default 20000
BP8K0	Default 1000
BP16K0	Default 500
BP32K	Default 250

Initial sizing of buffer pools

- Do not over allocate relative to available storage
  - ►DSNB536I indicates when total VSTOR BPOOL requirement > available REAL capacity of image
  - ►DB2 will limit allocation of buffer pools if aggregated (allocated) pool size > 2 x REAL storage size of system image
    - –DSNB610I indicates reduction to 2000 pages for 4K, 1000 pages for 8K, 500 pages for16K and 250 pages for 32K



Migration sizing of buffer pools

- Data spaces and virtual pools (NO hiper pool) -- VPSIZE is used
- Virtual pools with a corresponding hiper pool -- VPSIZE + HPSIZE is used
- VPSEQT, VPPSEQT and VPXPSEQT keep their sizes
  - ►Even if buffer pool size is determined by VPSIZE + HPSIZE
- DB2 V8 maintains original virtual pool and hiper pool sizes for fallback
- No change for new installations -- values taken from install process
- Net result is that their sizes should be their sizes after migration

Additional buffer pool information

- Buffer pool attributes still changed via ALTER BUFFERPOOL command
- Names unchanged: BP0, BP1
- Pages size options remain at 4K, 8K, 16K and 32K
- ALTER BUFFERPOOL command parameters NO LONGER supported:
  - VPTYPE, HPSIZE, HPSEQT, CASTOUT
    - If the above are specified, ONLY a warning message is issued
- Parameters remaining unchanged:
  - VPSEQT, VPPSEQT, VPXPSEQT, DWQT, VDWQT and PGSTEAL
- LSTATS report removes references to hiper pool related counters

Planning

- zSeries processor running 64-bit mode and z/OS 1.3 (or later)
  - Introduce through test and development environments
  - Consider disaster recovery implications
  - Parallel Sysplex and cross system restart
- Estimating storage: remember DB2 allocates space above 2GB bar in DBM1
  - Free up significant storage in the 31-bit addressable area
    - More concurrent threads
    - Higher transaction throughput
- z/OS V1.2 provides a new MEMLIMIT JCL keyword
  - Controls how much VSTOR above 2-GB bar is available in address space
  - DB2 sets MEMLIMIT value to 4 TB (minimum) ensuring sufficient memory for operation

Other recommendations / information

- Ensure a fully tested stand-alone dump is available
  - Install the z/OS version of SAD with the "High Virtual Option"
- Watch and grow ECSA when increasing thread limits
  - ECSA formerly used by IRLM now freed up
- Start by running a DB2 for z/OS and OS/390 Version 7 subsystem under a 64-bit virtual O/S
- Set low values for everything in the beginning
  - Buffer pools, EDM, thread limits
- MVS runs out of auxilliary storage slots at 4 TB

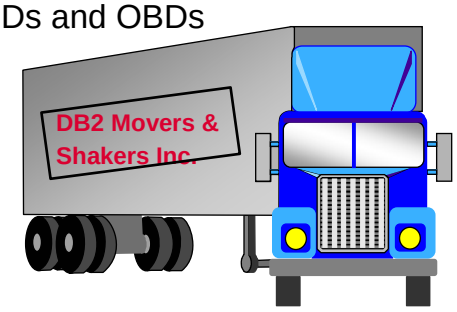
IBM Software Group | DB2 Information Management Software

What else is moving above the 2GB Bar?

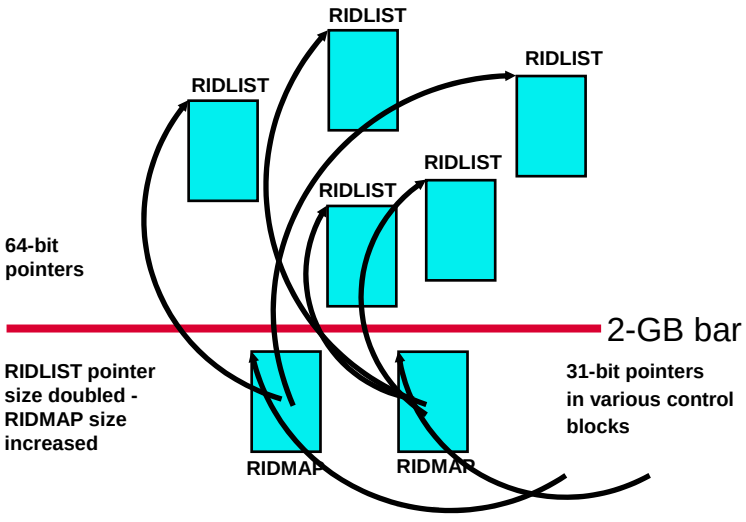
In addition to buffer pools, PMBs and castout buffers

What is moving above the 2-GB bar?

- RID pool
- Sort pool
- Compression dictionaries
- EDM pool - DBDs and OBDs



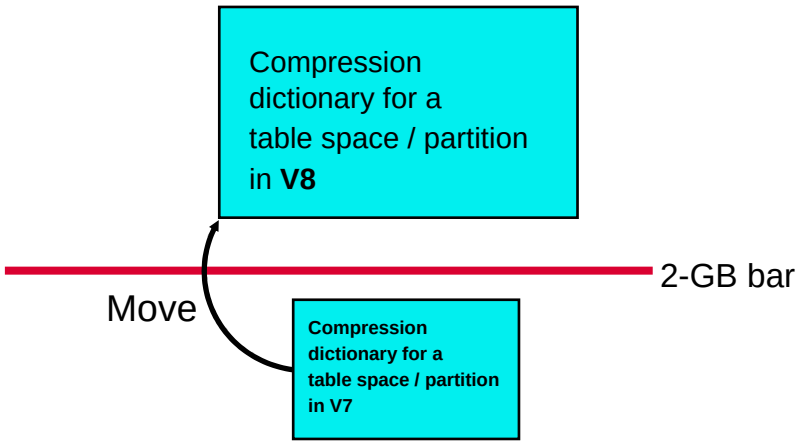
RID pool



RID pool -- external changes

- Split into two parts
  - ►Small part below 2-GB bar storing RIDMAPs
  - ►Larger part above 2-GB bar storing RIDLISTs (bulk of RID pool storage)
- No change to the install panel
- Slight modification to guidelines for estimating the size
  - ►Same size RIDMAPs would have held half as many RIDLISTs
    - –RIDMAP size is doubled to accommodate the same number of 8 byte RIDLISTs
    - –Each RIDLIST now holds twice as many RIDs
    - –RIDBLOCK size is now 32K
- New RIDPOOL calculation:
  - ►Each RIDMAP contains over 4000 RIDLISTs
  - ►Each RIDLIST contains 6400 RID entries
    - –Hence each RIDMAP / RIDLIST combination can contain over 26 million rids
    - –(vs. roughly 13 million in V7 and below)

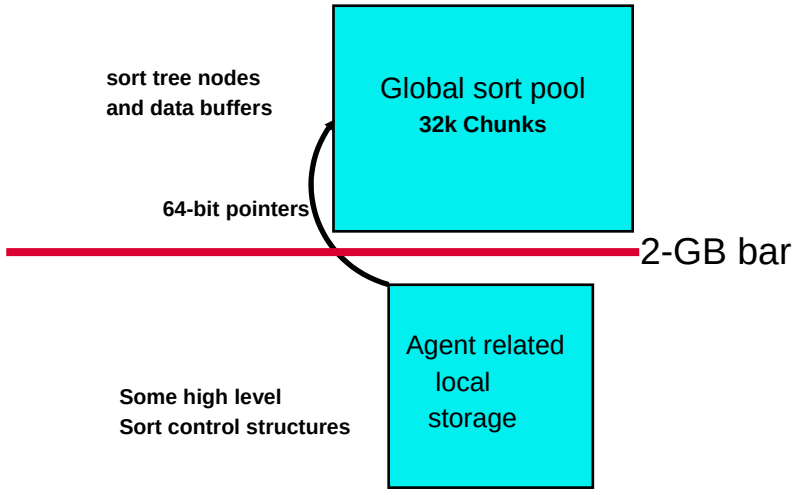
Compression dictionaries



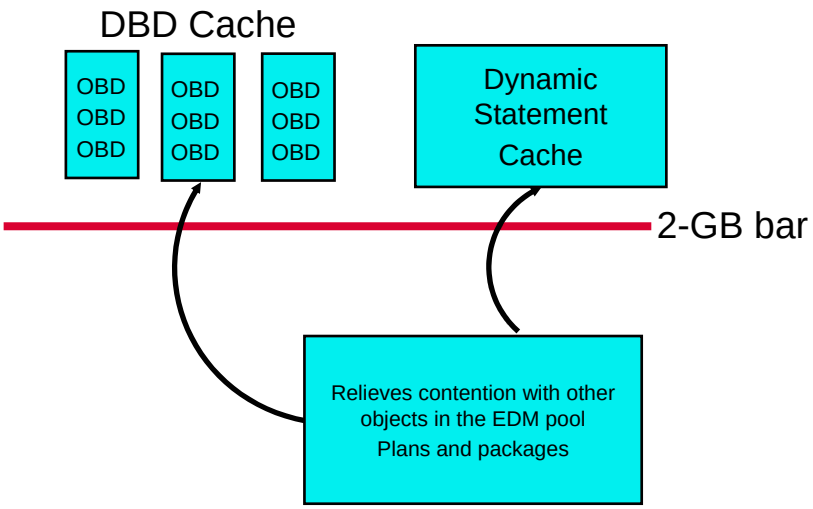
Compression dictionaries -- details

- Can occupy 64K of storage per data set
  - **512 MB for 8000 open, compressed partitions and table spaces**
- What was changed:
  - The compression dictionary will be loaded above the bar after it is built
  - All references to the dictionary now use 64-bit pointers
  - Use standard 64-bit hardware compression assembler instructions
- Standalone utilities
  - Still load dictionary below the bar

Sort pool



EDM pool -- DBD cachings & OBDs



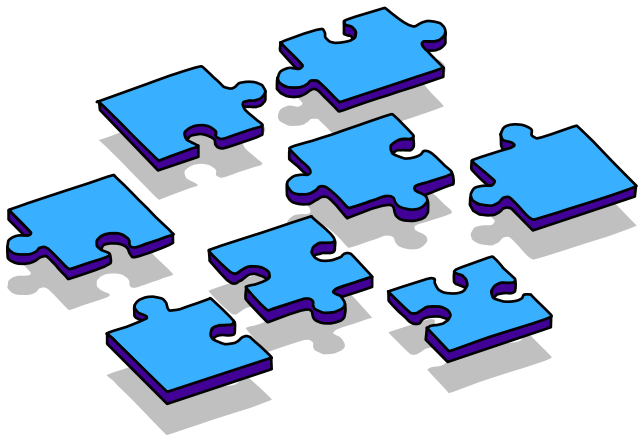
EDM pool -- DBD and statement caching

- Use of data space for dynamic caching is deprecated
  - New cache created above the 2GB bar
  - Today if "cache dynamic" is on:
    - Statements are cached in data space
    - EDM pool
  - Now "cache dynamic" statements ALWAYS cached in the dynamic statement cache pool above 2-GB bar
- New EDM DBD cache created above 2 GB
  - Will give DBDs needed space to grow and relieve contention with other objects

EDM pool -- install panel (DSNTIPC) values

- Change
  - ►EDMPOOL DATASPACE SIZE to
  - ►EDMPOOL DBD CACHE SIZE (Above 2-GB bar)
- Change
  - ►EDMPOOL DATA SPACE MAX to
  - ►the EDM Pool Statement Cache Size (Above 2-GB bar)
- No changes to estimation and defaults of EDMPOOL storage
  - ►except that instead of data space, it will be storage above 2-GB bar

64-bit virtual support question time



For more z/OS information

- z/OS web site
  - **ibm.com**/servers/eserver/zseries/zos
  - **ibm.com**/servers/eserver/zseries/zose
- z/OS migration and installation web pages
  - "z/OS and z/OS.e Planning for Installation Guide" GA22-7504
    ibm.com/servers/eserver/zseries/zos/bkserv/find_books.html
- Order z/OS online
  - **ibm.com**/software/ShopzSeries/
- z/OS 64-bit Virtual Storage Roadmap
  - **ibm.com**/servers/eserver/zseries/library/whitepapers/pdf/gm130076.pdf
- z/OS publications
  - **ibm.com**/servers/eserver/zseries/zos/bkserv/find_books.htm.
- System Center flashes
  - **ibm.com**/support/techdocs/