

## Module 4 SQLOS



### Overview

- Understanding Windows Scheduling
- Server hardware and NUMA
- The purpose of SQLOS
  - CPU Scheduling
  - Memory allocation
- Edge cases for tweaking
- DMV monitoring and troubleshooting (basics)

## Windows Scheduling

- Windows NT+ uses a multilevel feedback queue
  - Give preference to short jobs
  - Give preference to I/O bound processes
  - Separate processes into categories based on their need for the processor
  - Threads execute within their quantum
    - Once the quantum expires the thread must wait for NT to schedule them again
  - All threads with the same priority are treated equal
  - Threads with highest priority are scheduled first
  - If a higher-priority thread becomes available to run, the system stops executing lower-priority threads and assigns a full time slice to the higher-priority thread

©SQLskills.com  
SQLskills Immersion Event

SQL

## Windows Scheduling

- Threads priorities are dynamic and can be changed by NT based on a number of factors:
  - Normal priority processes in the foreground are increased so that the priority is greater than any background processes thread priority
  - If a window receives input, such as timer messages, mouse messages, or keyboard input, the scheduler boosts the priority of the thread that owns the window
  - When the wait conditions for a blocked thread are satisfied, the scheduler boosts the priority of the thread
- Low priority threads can be randomly boosted to ensure they execute critical code sections and exit appropriately through priority inversion

©SQLskills.com  
SQLskills Immersion Event

SQL

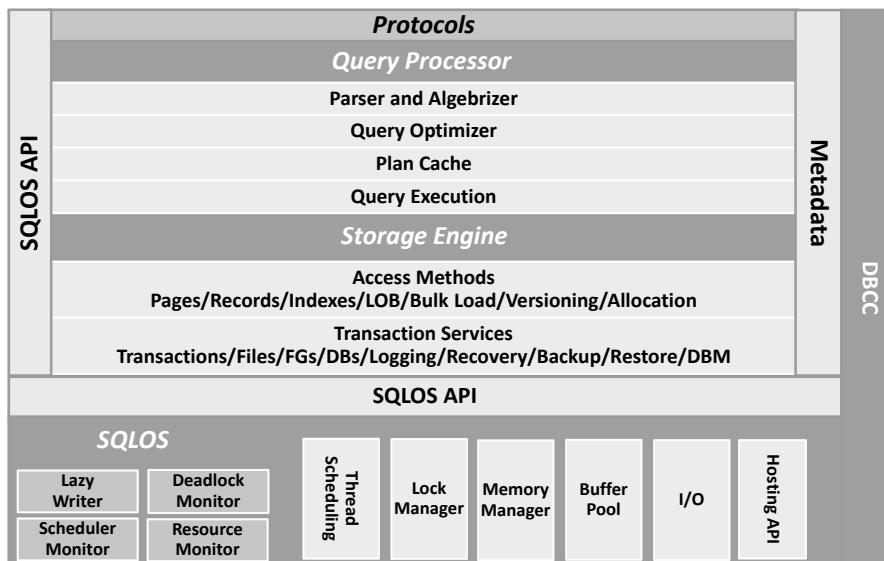
## Context Switching

- The scheduler maintains a queue of executable threads for each priority level known as ready threads
- When a processor becomes available, the system performs a context switch by:
  - Saving the context of the thread that just finished executing
  - Moving the thread that just finished executing at the end of the queue for its priority
  - Find the highest priority queue that contains ready threads
  - Remove the thread at the head of the queue, load its context, and execute it
- The most common reasons for a context switch are:
  - A threads quantum has elapsed
  - A thread with a higher priority has become ready to run
  - A running thread needs to wait and relinquishes the remainder of its time slice

©SQLskills.com  
SQLskills Immersion Event

SQL

## Server Architecture



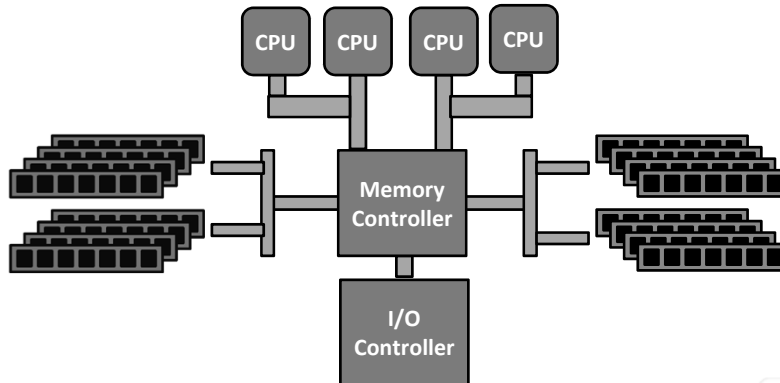
©SQLskills.com  
SQLskills Immersion Event

6

SQL

## Traditional SMP Systems

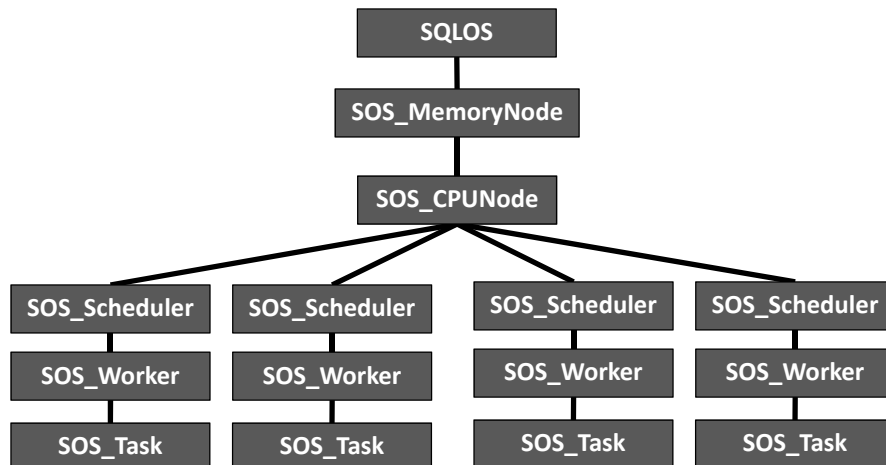
- Processors share front-side bus or cross bar
- Memory access uniform across all CPU cores



©SQLskills.com  
SQLskills Immersion Event

7

## SQLOS Under SMP



©SQLskills.com  
SQLskills Immersion Event

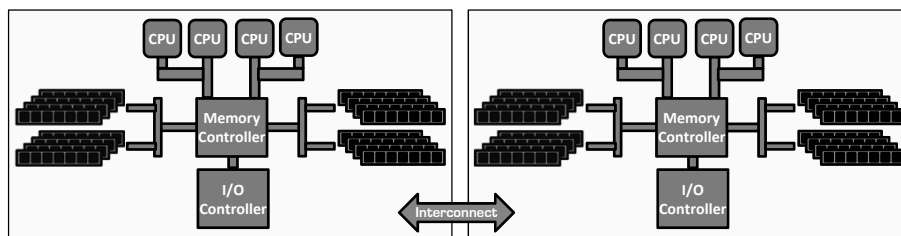
8

## The case for Non-Uniform Memory Access (NUMA)

- Modern CPUs operate faster than the memory to which they are attached
- Only one processor can access memory at a time for coherency
- As the number of processor cores increases, the cross bar/front-side bus becomes the system bottleneck starving multiple processors for memory access

## Traditional NUMA

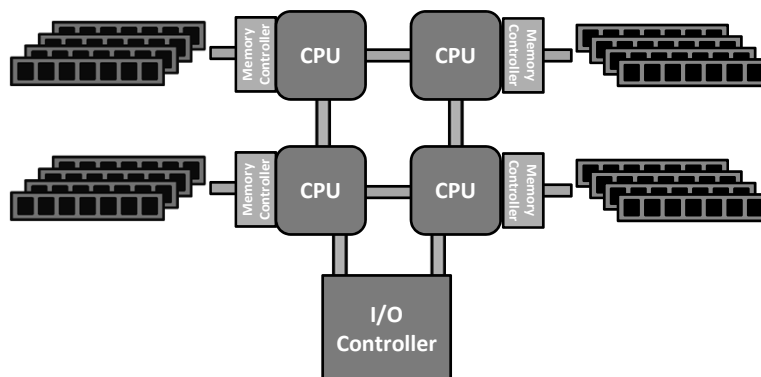
- Traditionally NUMA configurations required special hardware (ex: IBM xSeries Servers)
- Nodes were physically segregated servers connected by special interconnect
  - Two SMP servers running together as a single system
  - Required consideration for I/O and controller placement



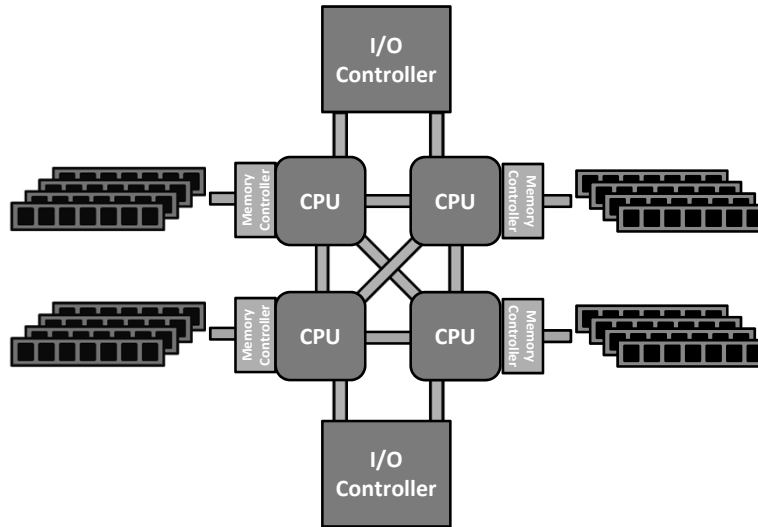
## Modern NUMA

- Memory controllers are built into the processor die
- Each processor socket presents itself as a individual NUMA node to the OS unless node interleaving is enable in the BIOS
- Internode communication paths are extremely fast and foreign memory allocations have a lower impact on overall performance

## AMD NUMA – HyperTransport



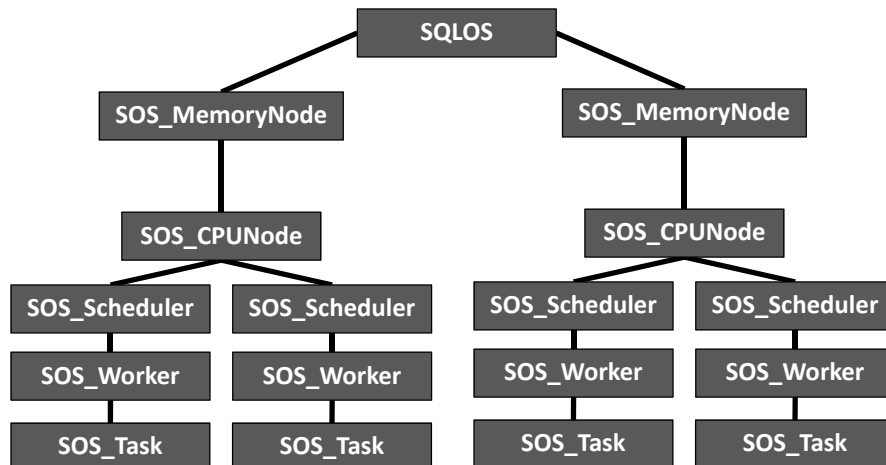
## Intel Nehalem – QPI



©SQLskills.com  
SQLskills Immersion Event

13

## SQLOS Under Hardware-NUMA



©SQLskills.com  
SQLskills Immersion Event

14

## Is NUMA important still?

- Even with enhancements NUMA still matters
  - Separate lazywriter process per NUMA node
  - Impact to checkpoint operations
  - Latency with remote memory still matters
- Max Degree of Parallelism
  - Number of physical cores per NUMA node
  - Prevent cross-node scheduling of parallel processing
    - Leverage Level 1 cache on die to prevent translation look-aside buffer (TLB) misses
      - The TLB is a CPU cache that memory management hardware uses to improve virtual address space translation speed
      - The TLB maps virtual and physical address spaces to reduce the number of page table lookups performed, which read multiple memory areas to compute the address needed

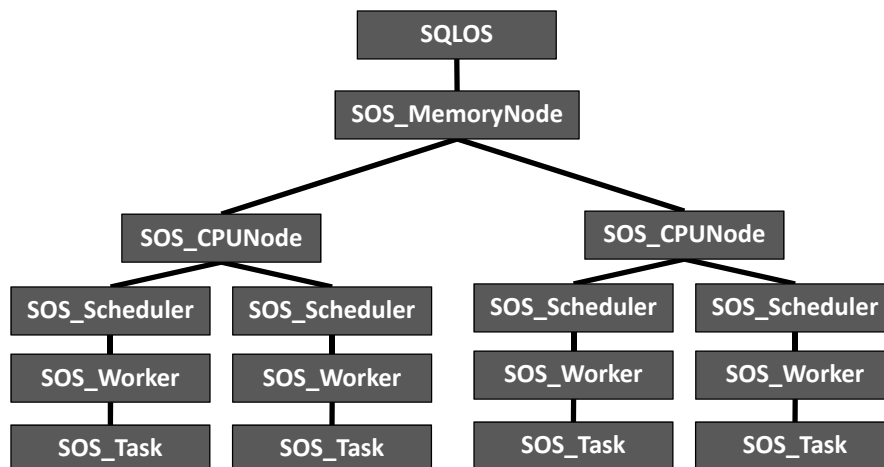
## Node Interleaving

- The option to run the server in an SMP configuration
- Sufficiently Uniform Memory Architecture (SUMA) configuration
  - Memory mapped in 4KB regions to nodes in round robin even fashion instead of as a single sequential block under NUMA
- Beneficial to certain workloads, but should not generally be used with SQL Server

## Soft NUMA

- Creates a custom NUMA configuration independent of hardware NUMA configuration
  - Registry settings control Soft NUMA configuration
  - Can provide greater performance, scalability, and manageability on SMP as well as on real NUMA hardware for specific workloads
  - ETL world record set using Soft NUMA configuration
  - Allows the ability to mask independent nodes to specific TCP ports for increased performance
- Excludes the usage of hardware NUMA when enabled
  - Soft NUMA interleaves nodes in the SQLOS regardless of the BIOS/hardware NUMA configuration

## SQLOS Under Soft-NUMA



## SOS Scheduler

- Provides a thin layer between SQL Server and the OS
  - Provides cooperative scheduling and I/O processing for SQL Server instead of preemptive scheduling used by Windows
    - Cooperative scheduling – tasks have an execution quantum (duration of time) and voluntarily yield the CPU to other tasks
    - Preemptive scheduling – highest priority task gets the CPU
- Two modes of execution: thread or fiber
  - Thread is default
  - Fiber can provide performance boost for specific usage scenarios
    - Rarely used in production environment
    - CLR not supported under fiber mode

## SOS Workers (Threads)

- Default configuration of 0
    - 32-bit SQL Server
      - Actual number is:  $((\text{NumberOfSchedulers} - 4) * 8) + 256$
    - 64-bit SQL Server
      - Actual number is:  $((\text{NumberOfSchedulers} - 4) * 16) + 512$
- SELECT max\_workers\_count  
FROM sys.dm\_os\_sys\_info**
- Edge cases for changing default
    - Database mirroring on 32-bit servers
    - Principal server uses 1 global thread, 2 threads per mirrored database
    - Mirror server uses 1 global thread, 2 threads per mirrored database and 1 additional thread per mirrored database for every 4 processors

## Scheduler Monitor

- Per node monitoring for issues every 5 seconds
  - 17883: Non-yielding task, single scheduler
  - 17887: I/O completion stall, single scheduler
  - 17884: No work progressing, all schedulers
  - 17888: 50% of same resource, all schedulers
- Ring buffer entries – sys.dm\_os\_ring\_buffers and Extended Events

## Scheduler DMVs

- sys.dm\_os\_schedulers
- sys.dm\_io\_pending\_io\_requests
- sys.dm\_os\_workers
- sys.dm\_os\_tasks
- sys.dm\_os\_waiting\_tasks
- sys.dm\_os\_wait\_stats

## Demo

### Scheduling DMVs



## Memory Nodes

- Represent the memory attached to all CPUs in SMP configuration or to a single NUMA node
- Supports several different memory allocators
  - Single page allocator - buffer pool memory
  - Multiple page allocator - memory outside of buffer pool
  - Reserved page allocator – DAC memory
- DMVS and DBCC MEMORYSTATUS provide information about the memory nodes in the system
  - sys.dm\_os\_memory\_nodes

## Memory Allocators

- Allocate memory pages internally to components for usage
  - The SQLOS page size is 8KB for memory allocations
- Single page allocator
  - Buffer Pool
  - Plan Cache on 64 bit systems (stolen pages)
- Multi page allocator
  - Linked Servers, SQLCLR, SQLXML, OLE Automation,
- Large page allocator
  - Used for Buffer Pool and Plan Cache when Large Page support is enabled by Trace Flag 834
- Reserved page allocator
  - Used for the Dedicated Administrative Connection

©SQLskills.com  
SQLskills Immersion Event

SQL

## Memory Clerks

- Track memory usage for specific component
- Receives notifications about memory state changes and responds to pressure
  - SQLCLR garbage collection hooks into this mechanism
- Utilize memory node allocators for memory allocation
- DMVS and DBCC MEMORYSTATUS provide information about the memory clerks in the system
  - sys.dm\_os\_memory\_clerks

©SQLskills.com  
SQLskills Immersion Event

26

SQL

## Memory Caches

- Caches are memory clerks
- Buffer pool
  - Data page cache
- Cache Store
  - Generic cache framework
  - Procedure cache & system rowset cache
- User Store
  - Generic cache framework
  - Schema manager, security & metadata Caches
- sys.dm\_os\_memory\_cache\_counters

## Cache Store

- Generic cache mechanism with storage
  - sys.dm\_os\_memory\_cache\_hash\_tables
  - sys.dm\_os\_memory\_cache\_entries
- Implements size control by utilizing LRU (clock) algorithm
  - External clock hand controls memory consumed by all caches
  - Internal clock hand controls memory consumed by a given cache
  - sys.dm\_os\_memory\_cache\_clock\_hands
- User Store is exactly the same as Cache Store but doesn't have storage

## Buffer Pool General

- Largest Cache Store (and memory consumer) for SQL Server
- Single 8K page allocations
  - Data page cache
  - Procedure cache on 64 bit instances as well
- Reserves virtual address space upfront
  - Leaves space for thread stacks + 256MB (modified by -g) on 32 bit systems
- Memory is committed and mapped on demand

## Buffer Pool under NUMA

- Has a single memory clerk that spans NUMA nodes
  - DBCC MEMORYSTATUS incorrectly reports memory allocations under NUMA
  - Use Buffer Node performance counters to track NUMA memory allocations
- Separate lazywriter per NUMA node
- Max server memory divided evenly across nodes
  - 32GB max server memory with 8 NUMA nodes allocates 4GB per node
  - Taking a node offline results in its memory being redistributed to remaining nodes and foreign page allocations

## Memory Management (32 Bit Processes)

- User mode virtual Address Space (VAS) limited to 2GB ( $2^{32}$ )
  - Amount of memory addressable through calls to VirtualAlloc() to an application in Windows
- PAE - Physical Address Extension
  - Allows access to memory above the 4 GB address space (4 GB to 64 GB)
- Address Windowing Extensions (AWE)
  - Switches from a 32 bit pointer to a virtual 36 bit pointer
  - Allows an application to quickly manipulate physical memory greater than 4GB up to 64GB RAM
  - Limits /3GB tuning to under 16GB RAM due to reduced PTE entries to map memory in kernel mode VAS

## Memory Management (32 Bit Processes - /3GB and /PAE)

- /3GB Tuning
  - For servers with < 16GB RAM the /3GB boot.ini option can be used to increase the user mode VAS by 1GB
  - The /USERVA boot.ini option can be used to tune the amount of Page Table Entries (PTEs) for system stability
- -g SQL Server Startup parameter
  - Used to reserve additional startup virtual address space (VAS) also known as the MemToLeave (for the internal function used by SQL Server in 32 bit operation) to reserve VAS for use by the sqlservr.exe process for non-buffer pool allocations
    - Linked Servers, SQLCLR, OLE Automation, SQLXML, large execution plans, Extended Stored Procedures, etc.

## Memory Management

### (64 Bit Processes)

- User mode VAS increased dramatically from 2GB to 16 exabytes ( $2^{64}$ ) but limited to 8TB virtually by Windows
- Eliminates the need for specialized configuration for SQL Server to allocate the available memory from the server OS

## Lock Pages in Memory

- Lock Pages in Memory allows SQL Server to allocate memory using `AllocateUserPhysicalPages()` instead of `VirtualAlloc()`
- Required for AWE on 32 bit SQL Servers
- Prevents paging of the buffer pool on 64 bit instances
  - Does not require configuration of "AWE enabled" in SQL Server
  - Trace Flag -T845 required for Standard Edition instances
- Detected at startup only

## Large Pages

- 32-bit the OS page is 4KB
- 64-bit can use large pages (2-16 MB)
  - Helps avoid translation look-aside buffer (TLB) misses
- Requires
  - Enterprise Edition
  - Locked pages in memory
  - Trace flag -T834
  - <http://support.microsoft.com/kb/920093>
- All memory allocated at startup – must be contiguous
- Recommended for virtual servers implemented on VMware to minimize virtualized TLB lookups
  - [http://www.vmware.com/files/pdf/sql\\_server\\_virt\\_bp.pdf](http://www.vmware.com/files/pdf/sql_server_virt_bp.pdf)

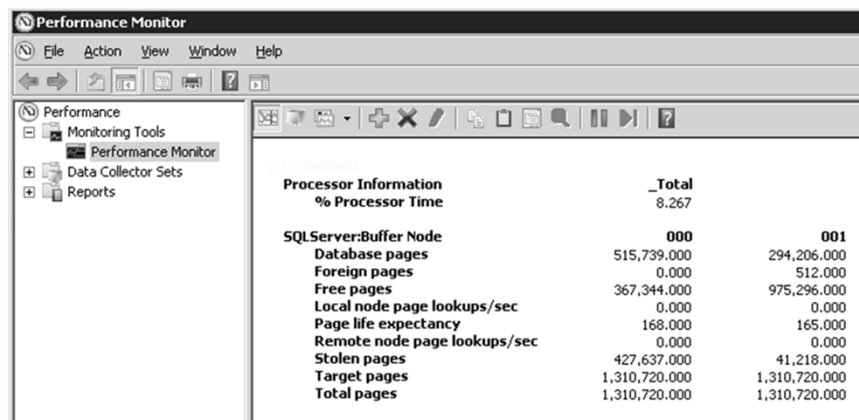
## Server Memory Configuration (Max Server Memory)

- Required on 64 bit servers to prevent memory pressure and out of memory conditions
- Only applies to the buffer pool, does not set the total amount of memory used by SQL Server
  - Additional memory for thread stack and multipage allocators
  - Memory for thread stack = (max worker threads) \*(stack size)
  - X32 = 512K; x64 = 2MB; ia64 = 4MB
- (Total system memory) – (memory for thread stack) – (OS memory requirements ~ 2-4GB) – (memory for other applications)
  - How I typically do it: (1GB for the OS + 1GB for each 4GB between 4-16GB + 1GB for each 8GB above 16GB in the server)
- Best to monitor Memory\Available Mbytes > 150-300MB and adjust as necessary

## Monitoring Memory Usage

- Performance counters
  - sys.dm\_os\_performance\_counters
  - Performance Monitor
- DMVs
  - Detailed information about memory usage by SQL Server at all levels
- DBCC MEMORYSTATUS
  - <http://support.microsoft.com/kb/907877>
- Ring buffers
  - sys.dm\_os\_ring\_buffers
  - History of memory notifications

## Monitoring Memory Usage (PerfMon BufferNode under NUMA)



| Processor Information        |               |               |
|------------------------------|---------------|---------------|
| % Processor Time             |               | _Total        |
|                              |               | 8.267         |
| <b>SQLServer:Buffer Node</b> |               |               |
|                              | <b>000</b>    | <b>001</b>    |
| Database pages               | 515,739.000   | 294,206.000   |
| Foreign pages                | 0.000         | 512.000       |
| Free pages                   | 367,344.000   | 975,296.000   |
| Local node page lookups/sec  | 0.000         | 0.000         |
| Page life expectancy         | 168.000       | 165.000       |
| Remote node page lookups/sec | 0.000         | 0.000         |
| Stolen pages                 | 427,637.000   | 41,218.000    |
| Target pages                 | 1,310,720.000 | 1,310,720.000 |
| Total pages                  | 1,310,720.000 | 1,310,720.000 |

## Resource Monitor

- Provides a mechanism to respond to memory pressure
  - Implemented as a regular task in the system using its own dedicated hidden scheduler
  - Invokes garbage collection in SQLCLR
- Outputs information to the OS ring buffers
  - ring\_buffer\_type="RING\_BUFFER\_RESOURCE\_MONITOR"

## Memory DMVs (System Information)

- (Reference slide)
- sys.dm\_os\_sys\_info
- sys.dm\_os\_sys\_memory
- sys.dm\_os\_loaded\_modules

## Memory DMVs (SQL Server)

- (Reference slide...)
- sys.dm\_os\_memory\_clerks
- sys.dm\_os\_memory\_objects
- sys.dm\_os\_memory\_allocations
- sys.dm\_os\_memory\_caches
- sys.dm\_os\_memory\_pools
- sys.dm\_os\_memory\_heaps
- sys.dm\_os\_virtual\_addresses
- sys.dm\_os\_virtual\_address\_dump
- sys.dm\_os\_stacks
- sys.dm\_os\_ring\_buffers
- sys.dm\_os\_buffer\_descriptors
- sys.dm\_os\_memory\_cache\_entries

©SQLskills.com  
SQLskills Immersion Event

41

SQL

## Demo

### Memory Monitoring with DMVs and DBCC MEMORYSTATUS

**SQL**  
skills

## Changes in SQL Server 2012

- Memory manager in SQLOS rewritten to have a new “any size page” allocator instead of separate single page and multi-page allocators
  - The ‘max server memory’ sp\_configure option is now the actual maximum for the instance process
  - AWE has been deprecated from SQL Server
  - Consistent out-of-memory handling and management across all the internal components
- New optimized memory broker for Column Store indexes
- Hot Add Memory enabled for VMs

## Review

- Server hardware and NUMA
- CPU scheduling
- Memory allocation
- DMV monitoring and troubleshooting (basics)
- Edge cases for tweaking