Memory usage (Managed Instance)

Last updated by | Charlene Wang | Sep 25, 2020 at 12:58 AM PDT

```
// top N memory clerks.  it captures max memory per interval
//
MonSqlMemoryClerkStats
| where  TIMESTAMP >= datetime({StartTime}) and  TIMESTAMP <= datetime({EndTime})
| where LogicalServerName =~ "{LogicalServerName}" and AppName =~ "{AppName}"  and NodeName =~ "
{NodeName}"
//| where AppName =~ "{AppName}" and NodeName =~"{NodeName}"
//| extend MemoryInMB = round(pages_kb/1024.0,1)
| extend MemoryInGB = round(pages_kb/1024.0/1024,1)
| top-nested of bin(TIMESTAMP, 5min) by max(MemoryInGB), top-nested 5 of memory_clerk_type by
MaxMemInGB=max(MemoryInGB) desc
| sort by TIMESTAMP asc nulls last
| project TIMESTAMP, memory_clerk_type, MaxMemInGB
| render timechart




//*****************************************************************************************************
// Instance Memory
//*****************************************************************************************************
// M.01
// Memory Manager Perfmon Counters
// Ex: Total Server Memory, Target Server memory
MonDmOsMemoryManagerCounters
| where TIMESTAMP > datetime({StartTime}) and TIMESTAMP < datetime({EndTime})
| where NodeName == "{NodeName}" and ClusterName == "{ClusterName}"
| where LogicalServerName =~ "{LogicalServerName}"   and AppName =~ "{AppName}"
| summarize avg(cntr_value) by bin(TIMESTAMP, 10min), counter_name, NodeName
| render timechart
// M.02
// BPool Counters
// Ex: page lookups
MonDmOsBPoolPerfCounters
| where TIMESTAMP > datetime({StartTime}) and TIMESTAMP < datetime({EndTime})
| where NodeName == "{NodeName}" and ClusterName == "{ClusterName}"
| where LogicalServerName =~ "{LogicalServerName}"   and AppName =~ "{AppName}"
| summarize avg(cntr_value) by bin(TIMESTAMP, 10min), counter_name, NodeName
| render  timechart
// M.03
// Memory Clerks over time (filtered to 10MB and above)
// MonSqlMemoryClerkStats is captured every 5 min
// top N memory clerks
```

```
MonSqlMemoryClerkStats
//| where  TIMESTAMP >= datetime({StartTime}) and  TIMESTAMP <= datetime({EndTime})
| where LogicalServerName =~ "{LogicalServerName}" and AppName =~ "{AppName}"  and NodeName =~ "
{NodeName}"
| extend MemoryInMB = round(pages_kb/1024.0,1)
| top-nested of bin(TIMESTAMP, 5min) by sum(MemoryInMB), top-nested 5 of memory_clerk_type by
TotalMemInMB=sum(MemoryInMB) desc
| sort by TIMESTAMP asc nulls last
| project TIMESTAMP, memory_clerk_type , TotalMemInMB
| render timechart
//Memory clerk over 10MB
MonSqlMemoryClerkStats
| where  TIMESTAMP >= datetime({StartTime}) and  TIMESTAMP <= datetime({EndTime})
| where LogicalServerName =~ "{LogicalServerName}" and AppName =~ "{AppName}" and NodeName =~ "
{NodeName}"
| extend MemoryInMB = pages_kb/1024.0
| summarize TotalMemoryInMB=round(sum(MemoryInMB),0) by bin(TIMESTAMP, 5min), memory_clerk_type
| where TotalMemoryInMB >10
| render timechart
// M.04 RM Ring buffer
// RM Ring buffer -- do we have memory pressure
//  1 -> MEMPHYSICAL_HIGH
//  2 -> MEMPHYSICAL_LOW
//  4 -> MEMVIRTUAL_LOW
MonSqlRmRingBuffer
| where TIMESTAMP > datetime({StartTime}) and TIMESTAMP < datetime({EndTime})
| where notification == "MEMPHYSICAL_LOW"
| where AppName == "{AppName}" and NodeName == "{NodeName}"
| project TIMESTAMP, process_indicators, system_indicators, pool_indicators
//| render timechart
// M.05
// SQL CP cache when system or process memory low
MonSqlCaches
| where TIMESTAMP > datetime({StartTime}) and TIMESTAMP < datetime({EndTime})
| where AppName == "{AppName}"
| where NodeName == "{NodeName}"
| where event == "clock_hand_stats"
| where store_type endswith "CP"
| where is_system_physical_memory_low == 1 or  is_process_physical_memory_low == 1
| project PreciseTimeStamp, store_name, clock_hand_id, rounds_count, visited_last_round, removed_last_round,
is_system_physical_memory_low, is_process_physical_memory_low, is_process_virtual_memory_low,
pool_physical_memory_indicator_mask
| order by PreciseTimeStamp asc nulls last
// M.06
// Memory usage by pool
// Governor Pool Memory uage
MonGovernorResourcePools
| where max_memory_kb > 0
| where AppName =~ "{AppName}" and NodeName =~ "{NodeName}"
| where  TIMESTAMP >= datetime({StartTime}) and  TIMESTAMP <= datetime({EndTime})
//| where name =="SloHkPool"
| extend avg_used_mem_pct = tolong(used_memory_kb * 100.0 / max_memory_kb)
| summarize Avg_Mem_Usage_Percent=avg(avg_used_mem_pct) by bin(TIMESTAMP, 5min), PoolName=name
//| summarize avg(used_memory_kb), avg(max_memory_kb)  by bin(TIMESTAMP, 5min), PoolName=name
| render timechart
// M.07
// OOF factor, Pools
// memory_node_oom_ring_buffer_recorded xevent
MonSqlMemNodeOomRingBuffer
| where AppName == "{AppName}"
| project TIMESTAMP, AppName, LogicalServerName, failure, factor, last_error, task, pool_metadata_id,
committed_kb, job_object_limit_job_mem_kb, is_system_physical_memory_low, instance_rg_size
// M.08
// Roughly check non-SOS memory usage when OOM occurs.
// memory_node_oom_ring_buffer_recorded xevent
MonSqlMemNodeOomRingBuffer
| where AppName == "{AppName}" and NodeName == "{NodeName}"
| extend nonSosMemKb=job_object_limit_job_mem_kb - committed_kb
| project TIMESTAMP, job_object_limit_job_mem_kb, committed_kb, nonSosMemKb
| render timechart
//*********************************************************************************************************
// Memory grant
//*********************************************************************************************************
// Grant
// G.01 Resource Semaphores
MonQueryResourceSemaphores
| where  TIMESTAMP >= datetime({StartTime}) and  TIMESTAMP <= datetime({EndTime})
| where LogicalServerName =~ "{LogicalServerName}" and AppName =~ "{AppName}" and NodeName =~ "
{NodeName}"
| where granted_memory_kb > 1024 // > 1MB
| project TIMESTAMP, granted_memory_kb, grantee_count, waiter_count

// QDS granted memory
// G.02
MonWiQdsExecStats
| where  TIMESTAMP >= datetime({StartTime}) and  TIMESTAMP <= datetime({EndTime})
| where LogicalServerName =~ "{LogicalServerName}" and   database_name  =~ "{LogicalDatabaseName}" and
AppName == "{AppName}"
| where is_primary  == 1
| extend MaxQueryMemoryMB=round(max_max_query_memory_pages*8.0/1024,1)
| summarize max(MaxQueryMemoryMB) by query_hash
| top 20 by max_MaxQueryMemoryMB  desc nulls last

// Sampled query meory grant waiter
// G.03
// CWarningMemgrantBlocking::WarnOnBlocking
// xesqlminpkg_common.xe on query_memory_grant_blocking
// max_query_memory_kb = max memory grant ALLOWED for one query
```

```
MonQueryProcessing
| where  TIMESTAMP >= datetime({StartTime}) and  TIMESTAMP <= datetime({EndTime})
| where LogicalServerName =~ "{LogicalServerName}" and    AppName == "{AppName}"
| where  event =~ "query_memory_grant_blocking"
| extend sampled_mem_waiter_query_hash =strcat('0x',toupper(tohex(blocking_query_hash)))
| extend session_id=blocking_session_id, requested_memory_mb=requested_memory_kb/1024,
required_memory_mb=required_memory_kb/1024, ideal_memory_mb=ideal_memory_kb/1024
| extend total_granted_memory_mb=total_granted_memory_kb/1024,
max_query_memory_allowed_per_query_mb=max_query_memory_kb/1024,
total_available_memory_mb=total_available_memory_kb/1024,
total_max_memory_mb=total_max_memory_kb/1024
| extend query_cost=round(query_cost, 1)
| project TIMESTAMP, sampled_mem_waiter_query_hash, session_id,dop, wait_time_sec, query_cost,
requested_memory_mb,required_memory_mb,ideal_memory_mb,total_granted_memory_mb,max_query_memor
y_allowed_per_query_mb, total_available_memory_mb, total_max_memory_mb,pool_name, total_waiter_count ,
total_grantee_count
```

How good have you found this content?