**SQL Server内存性能分析**

**内存概念：**

**Working Set = Private Bytes + Shared Memory**

**Working Set**：某个进程的地址空间中，存放在物理内存的那一部分

**Private Bytes**：某个进程提交的地址空间（Commited Memory）中，非共享的部分

**Shared Memory**：对多个进程可见的内存，或存在于多个进程的虚拟地址空间。例如如果两个进程使用相同的DLL，只要该DLL代码装入内存一次，其他所有映射这个DLL的进程只要共享这些代码就可以了

**Virtual Address Space** :2的64次方。Windows会自行决定什么时候放在Physical Memory，什么时候放在Paging file

**Reserved Memory**：应用程序会先Reserve一块内存的地址空间

**Committed Memory**：将Reserve的内存页面正式Commit使用，Commit的页面会最终转换到Physical Memory中的有效页面

**Memory Leak**：一直不断的Reserve或Commit内存资源，即使不再被使用也不会释放给其他用户使用

**Page Fault**：访问一个存放于Viratul Address Space，但不存在于物理内存（Working Set）的页面，就会发生Page Fault。如果目标页面存放于硬盘（例如Pageing File），则会进行硬盘读写，称之为Hard Fault。如果已经存在在物理内存中，但是还没有直接放在这个进程的Working Set下，则需要Windows重新定向一次，不会带来硬盘读写，称之为Soft Fault。由于Soft Fault一般不会带来性能影响，因此一般不使用该计数器，而是使用Pages/sec

**Committed Bytes**：整个Windwos系统（包括Windows自身及其所有用户进程）使用的内存总数，包括Physical Memory和Paging file中的数据

**Pages/sec**：Hard Page Fault每秒钟需要从磁盘上读取或写入的页面数目（包括Windows和所有应用进程的所有磁盘Paging动作）  
Pages/sec = Memory:Pages Input/sec + Pages Output/sec

**Page File:%Usage和Page File:%Peak Usage** ：反映Paging File使用量的多少。数据在Paging File中存的越多，说明Physical Memory和实际需求量差距越大，性能越差

=================================================================================

**Windows系统自身内存使用情况：**  
一般情况下在64为机器上，使用1-2GB左右。如果由于一些硬件驱动造成了内存泄露，则Windows可能会占用到几GB甚至更多的内存。

**系统的Working Set，也就是系统使用的物理内存量，包括高速缓存、页交换区、可调页的ntoskrnl.exe、驱动程序代码和系统映射图等**

**Cache Bytes**：系统使用的物理内存数目

Cache Bytes = System Cache Resident Bytes + System Driver Resident Bytes + System Code Resident Bytes + Pool Paged Resident Bytes

**Memory:System Cache Resident Bytes(System Cache)：**系统高速缓存消耗的物理内存。高速缓存的主要功能是提高文件读写速度（例如读写超大文件）

赋予SQL服务账号“Lock Pages in Memory”权限，以便SQL Server可以将自己所申请的内存锁定放在物理内存中，企业版自动启用该功能。但是NonbufferPool的内存不受限制

SQL Server作为一个用户态为主的应用程序，还是会受限于核心态。如果核心态里发出内存要求，SQL Server就会被迫把自己的内存释放出来。用户态永远也抢不过核心态。

=================================================================================

**检查某个进程的内存使用情况：**  
**Process:Working Set**

**Process:Private Bytes**

**Process:Virtual Bytes：**某个进程所申请的Virtual Address Space，包括Reserved Memory和Committed Memory

如果系统内存有瓶颈，磁盘一般也会很忙，paging也会较多；但如果磁盘很忙，但是paging不高，就不能说明系统内存有瓶颈

=================================================================================

**SQL Server内存:**  
**Traget Server Memory：**SQL Server在理论上能够使用的最多的内存量。如果在SQL Server中设置了 Max Server Memory，则会使用该值作为内存上限

**Total Server Memory：**SQL Server自己分的代码申请的Buffer Pool空间大小，所有的Database Cache和大部分的Consumer（Connection、Query Plan、   
Optimizer等）都保存在Buffer Pool中，对于一些特别长的语句的Query Plan、Optimizer等内存使用一部分的Multi-Page

Buffer Pool = Max Server Memory 或 SQL physical memory-Multi-page

Max Server Memory是Buffer Pool的上限，但不是SQL Server所有内存使用的上限。SQL Server内存的使用包括Buffer Pool和MemToLeave，所以SQL Server实际内存使用量一定大于Max Server Memory。但在正常情况下，MemToLeave的使用会远小于Buffer Pool，控制好Buffer Pool，基本上就控制住了SQL Server的整体内存使用量。   
   
Traget Server Memory和Total Server Memory都是指逻辑上的内存空间大小，而不是物理内存空间大小。数据是存放在page file还是物理内存中是由Windows决定的。

**SQL Server内存使用分类：**

按申请方式分类：

对Database Cache，会先Reserve，再Commit。其他的所有内存使用，基本都是直接Commit，都是Stolen（Memory Leak）

按申请大小分类：

对于所有小于或等于8KB的，直接分配给一个页面:8KB。所有这些页面都集中管理，这块内存被称为Buffer Pool，一次一个页面的分配被称为Single page Allocation

对于大于8KB的内存申请，会被集中在另外一个区域，成为Multi-Page（或MemToLeave），这种分配称为Multipl Page Allocation

**SQL Server性能计数器使用：**

**Memory Manager：监视服务器内存总体使用情况**

**Ttotal Server Memory（KB）**

**Traget Server Memory（KB）**  
Optimizer Memory（KB）：服务器正在用于查询优化的动态内存总数

SQL Cache Memory（KB）：服务器正在用于动态SQL Server高速缓存的动态内存总数

Lock Memory（KB）：服务器用于锁的动态内存总量

Connection Memory（KB）：服务器正在用来维护连接的动态内存总量

Granted Workspace Memory(KB)：当前给予执行哈希、排序、大容量复制和索引创建等操作进程的内存总量

**Memory Grants Pending**：等待工作空间内授权的进程总数。如果该值不等于0，就说明当前有一个用户的内存申请由于内存压力而被延迟。一般来讲，这就意味着有比较严重的内存瓶颈

**Buffer Manager：用于监视内存如何使用**

**Buffer Cache Hit Ratio**：在缓冲区高速缓存中找到而不需要从磁盘中读取的页的百分比。经过很长时间后，该比率的变化应该很小，基本应该在99%以上。如果小于95%，通常就有了内存不足的问题。可以通过增加SQL Server的可用内存来提高

Database Pages:缓冲池中有数据库内容的页数。也就是所谓的Database Cache的大小

**Free pages**：所有空闲可用的总页数。当这个值降低是就说明SQL Server正在分配内存给一些用户。当这个值下降到比较低的值时（例如只剩几百个page了），SQL Server就会开始做Lazy Write，把一些内存让出来，所以该值一般不会为0.但如果该值反复降低，就说明内存存在瓶颈。一个没有内存瓶颈的SQL Server的Free Pages会维持在一个稳定的值   
   
**Lazy writes/sec**：每秒被缓冲区管理器的Lazy writer写入的缓冲区数。Lazy writer是一个系统进程，用于呈批刷新脏的老化的缓冲区（包括更改的缓冲区，必须将这些更改写回磁盘，才能将缓冲区重用于其他页），并使他们可用于用户进程。当SQL Server感觉到内存压力时，就会将最久没有被重用到的数据页和执行计划清理出内存。这些数据页和执行计划，就被称为“老化的缓冲区”，这个清理动作就是由Lazy writer完成的。所以如果SQL Server内存压力不大，Lazy writer就不会被经常触发。如果被经常触发，就应该是有内存瓶颈   
    
**Page life expentancy**：页若不被引用，将在缓冲池中停留的秒数。如果SQL Server没有新的内存需求，或者有空余的空间来完成新的内存需求，那么Lazy writer就不会被触发，页面会一直放在缓冲池中，Page life expentancy就会维持在一个较高的值。如果SQL Server出现了内存压力，Lazy writer就会被触发，Page life expentancy也会突然下降。所以如果Page life expentancy总是高高低低，SQL Server应该就出现了内存瓶颈   
   
**Page reads/sec**：每秒发出的物理数据库页读取数。此统计信息显示的是所有数据库间的物理页读取总数。如果用户访问的数据都缓存在了内存里，那么SQL Server就不需要从磁盘读取页面，不许需要做任何的Page reads。当SQL Server需要读取这些页面时，必须要为他们腾出内存空间。所以当Page reads/sec高时，一般Page life expentancy会下降，Lazy writes/sec会上升   
由于物理I/O开销大，Page Reads动作一定会影响SQL Server性能，可以通过使用更大的数据缓存、智能索引、更有效的查询或更改数据库设计等方法降低Page Reads

**Page writes/sec**：每秒执行的物理数据库页写入数。该值和内存使用没有什么关系，和Checkpoint pages/sec一样，更用户的修改量有关

**Checkpoint pages/sec**：由要求刷新所有脏页的检查点或其他操作每秒刷新到磁盘的页数。该值和内存压力没有直接关系，和用户行为有关。如果用户操作主要是读，Checkpoint值就比较小。如果很多操作都是Insert/Update/Delete，name内存中修改过的数据脏页就会比较多，每次Checkpoint的量也会较大。主要用来分析磁盘I/O

Stolen Pages：用于非Database Pages（包括执行计划缓存）的页数。这里就是Stolen Memory在Buffer Pool里的大小

Target Pages：缓冲池中理想的页数，乘以8KB，就应该是Target Server Memory的值

Total Pages：缓冲池中的页数（包括数据库页、可用页和Stolen页）乘以8KB，就应该还是Total Server Memory的值

**SQL性能分析步骤：**

1）当内存出现瓶颈时，会出现大量的paging动作，磁盘也就会很繁忙。所以要先解决内存瓶颈，才能降低I/O

2）分析磁盘性能是否正常，读/写是否已达到预期值，是读还是写繁忙

3）查看引起高I/O的操作类型（Page Reads、Page Writes、Lazy Writes、Checkpoints、Log Writes等）

**动态性能视图：**  
select \* from sys.sysprocesses

select \* from sys.dm\_os\_wait\_stats

select \*from sys.dm\_exec\_requests

=================================

计数器（2分钟）：

Processor\% Privileged Time   
Processor\% Processor Time   
Memory\Available Mytes   
PhysicalDisk\Avg. Disk sec/Read   
PhysicalDisk\ Avg. Disk sec/Write   
SQLServer:SQL Statistics\Batch Requests/sec   
SQLServer:Memory Manager\Target Server Memory (KB)   
SQLServer:Memory Manager\Total Server Memory (KB)

Process(w3wp)\Working Set  
Process(w3wp)\Private Bytes