您的潜力，我们的动力

Microsoft®
微软(中国)有限公司

Performance Tuning in SQL2005

孙巍
技术总监
北京中达金桥技术服务有限公司

msdn

MSDN Webcasts

收听本次课程需具备的条件

- SQL Engine
- CPU
- Memory
- IO

Level 300

Agenda

- Introduction
- Common causes of performance issues
  - Resource bottleneck

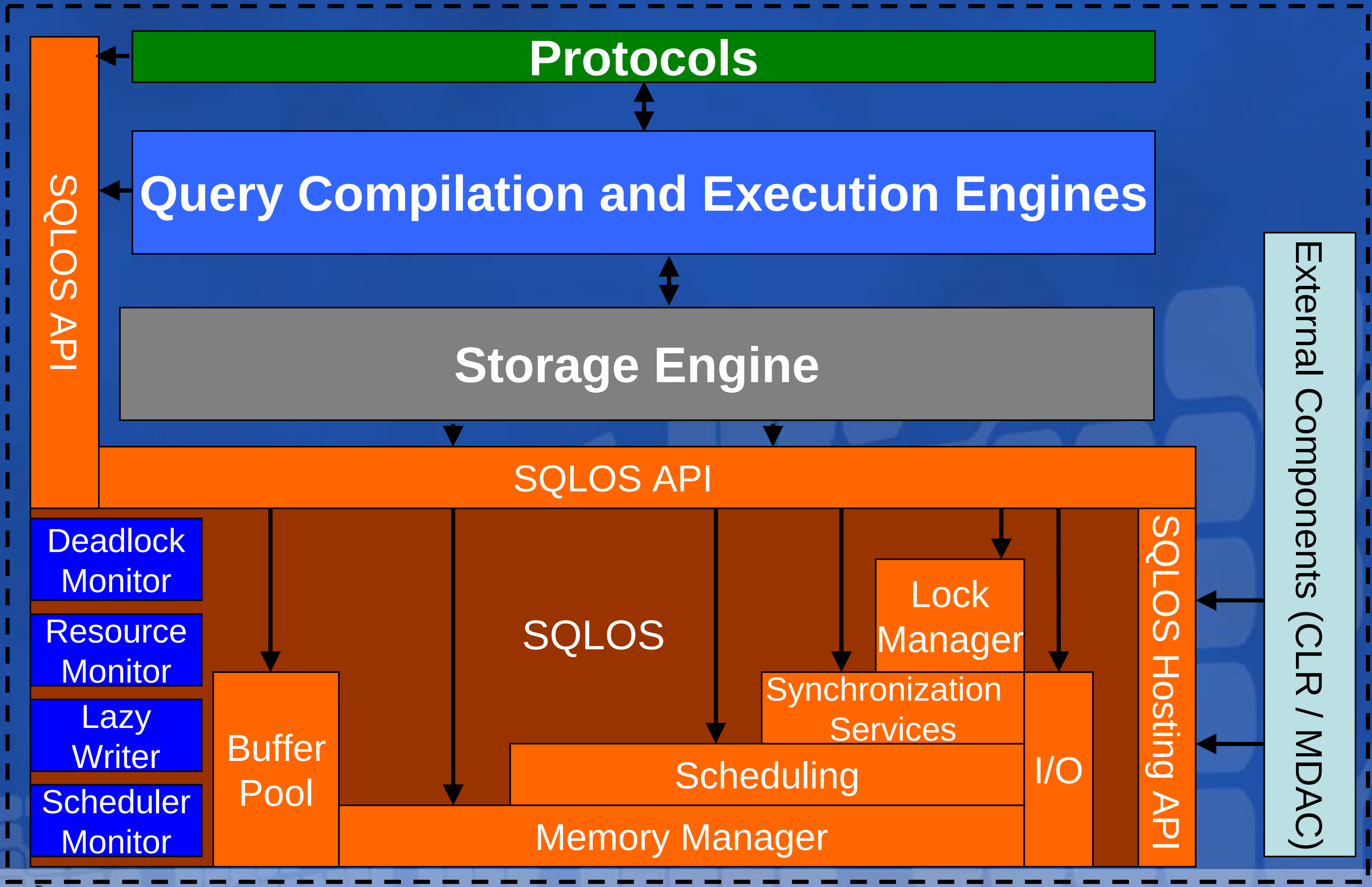
Introduction

- Goal
  – General methodology for perf tuning
- Tools
  – Perfmon
  – Profiler
  – Dynamic Management Views (DMVs)
  – DBCC commands

Performance Diagnostics

- Resource Bottleneck
  - CPU
  - Memory
  - IO

SQL Engine Overview



= thread

CPU Bottleneck

- High CPU usage or usage jump
  - Find out from Perfmon or Taskmgr
- Find out queries use most CPU

```
select top 50
sum(qs.total_worker_time) as TotalCPUTime,
sum(qs.execution_count) as TotalExecutionCount,
count(*) as NumberOfStatements,
qs.plan_handle
from sys.dm_exec_query_stats qs
group by qs.plan_handle
order by sum(qs.total_worker_time) desc
```

Common causes for high CPU usage: recompile

- Excessive compilation/recompilation
  - Compilation: produce query execution plan
  - Recompilation: to update query plan
  - SQL2000 recompiles the whole stored procedure; SQL2005 can do statement level recompile
- What triggers recompile?
  - Schema change; Statistics change; Set option change
  - Deferred compile; temp table change; SP created with RECOMPILE or option used

Common causes for high CPU usage: recompile

- How to detect excessive recompilation
  - Perfmon counters
    - Batch Requests/sec
    - SQL Compilations/sec
    - SQL Recompilations/sec
    - Ration of recompile to batch requests should be very low

Common causes for high CPU usage: recompile

- How to detect excessive recompilation (cont.)
  - SQL Trace
    - SP:Recompile & SQL:StmtRecompile event
    - provide more information

      ObjectID, Spid, Start time, sqlhandle, Text Data
    - Can query trace data

      Select spid, StartTime, Textdata, EventSubclass, ObjectID, DatabaseID, SQLHandle

      from fn_trace_gettable ('e:\recompiletrace.trc' , 1)

      where EventClass in(37,75,166)

      // EventClass 37 = Sp:Recompile, 75 = CursorRecompile, 166=SQL:StmtRecompile
    - Showplan XML For Query Compile

Common causes for high CPU usage: recompile

- How to detect excessive recompilation (cont.)
  - Dynamic Management Views (DMVs)
    - Find the top 25 recompiled stored procedures

```
select top 25
   SQLText.text, sql_handle, plan_generation_num,
   execution_count, dbid, objectid
from sys.dm_exec_query_stats a
   Cross apply sys.dm_exec_sql_text(sql_handle) as SQLText
where plan_generation_num >1
order by plan_generation_num desc
```

    - Find the time used for query optimization

```
Select * from sys.dm_exec_query_optimizer_info
```

Common causes for high CPU usage: recompile

- Resolution for excessive recompile
  - If it’s due to set option change
    - Avoid change set option in SP; minimize change during connection
  - If it’s on temp table
    - Lower recompile threshold for temp table
    - Change temp table to use table variable or use KEEP PLAN query hint
  - Use qualified name to avoid recompilation and ambiguity
    - Select * from dbo.table1

Common causes for high CPU usage: recompile

- Resolution for excessive recompile (cont.)
  - If it's due to deferred compile
    - Avoid interleave DML and DDL statements
    - Avoid use DDL in conditional clause like IF
  - Run Database Tuning Advisor to see if any indexing changes improve the compile time and the execution time of the query
  - Review query and SP using recompile option
    - Try to limit recompile to statement level in SQL2005

Common causes for high CPU usage: inefficient query plan

- SQL server optimizer tries to find a plan for the fastest response time
- Hash join and sort operation can be CPU intensive
- Nested loop join is usually IO bound
- Inaccurate estimation for cardinality of operators could result inefficient plan

Common causes for high CPU usage: inefficient query plan

- How to detect
  - Find queries use most cumulative CPU through dm_exec_query_stats
  - Look for CPU intensive operators through dm_exec_cached_plans
  - Use Profiler to check for Performance Statistics trace event

Common causes for high CPU usage: inefficient query plan

- Resolution
  - Use Database Tuning Advisor to see if it produces any index recommendations
  - Avoid bad cardinality estimates
    - Write query with WHERE clause to narrow down
    - Run UPDATE STATISTICS
    - Modify query
  - SQL2005 has new plan guide for query hints
    - OPTION (OPTIMIZE FOR)
    - OPTION (FORCE ORDER)
    - OPTION (USE PLAN)

Common causes for high CPU usage: Intra-Query Parallelism

- A parallel query plan uses multiple threads to process the query, to take advantage of multiple CPUs
- Can be custom configured using
  - Server level: max degree of parallelism option
  - Per query level: OPTION (MAXDOP) hint
- Situation could change after parallel plan is chosen
  - New CPU intensive query could come in
  - Shorter CPU time slice

Common causes for high CPU usage: Intra-Query Parallelism

- How to detect:
  - Perfmon:
    - Process - % Processor Time – sqlservr.exe
  - DMVs:
    - Find query plans that may run in parallel

```
select p.*, q.*, cp.plan_handle
from sys.dm_exec_cached_plans cp
   cross apply sys.dm_exec_query_plan(cp.plan_handle) p
   cross apply sys.dm_exec_sql_text(cp.plan_handle) as q
where
   cp.cacheobjtype = 'Compiled Plan' and
   p.query_plan.value('declare namespace
   p="http://schemas.microsoft.com/sqlserver/2004/07/showplan";
               max(//p:RelOp/@Parallel)', 'float') > 0
```

Common causes for high CPU usage: Intra-Query Parallelism

- How to detect: (cont.)
  - DMVs:
    - Only parallel query could use more CPU time than the elapsed time
    - Find such queries

```
select qs.sql_handle, qs.statement_start_offset,
   qs.statement_end_offset, q.dbid, q.objectid, q.number,
   q.encrypted, q.text
from sys.dm_exec_query_stats qs
   cross apply sys.dm_exec_sql_text(qs.plan_handle) as q
where qs.total_worker_time > qs.total_elapsed_time
```

Common causes for high CPU usage: Intra-Query Parallelism

- How to detect: (cont.)
  - SQL Traces:
    - Showplans that have Parallelism operators

      Select EventClass, StmtText
      from ::fn_trace_gettable('c:\temp\high_cpu_trace.trc', default)
      Where StmtText LIKE '%Parallelism%'
    - Parallel query use more CPU time than the elapsed time

      Select EventClass, StmtText
      from ::fn_trace_gettable('c:\temp\high_cpu_trace.trc', default)
      Where EventClass in (10, 12)
      -- RPC:Completed, SQL:BatchCompleted
      And CPU > Duration/1000
      -- CPU is in milliseconds, Duration in microseconds

Common causes for high CPU usage: Intra-Query Parallelism

- Resolution
  - parallel plan is chosen when it exceeds the cost threshold of parallelism
  - Use the Database Tuning Advisor to reduce the cost of query
  - Avoid inaccurate cardinality estimate
    - No MISSING STATS warnings in showplan output
    - Run UPDATE STATISTICS
    - Limit usage of query construct which optimizer can't estimate accurately
      - Multiple table valued function, CLR function, table variables,
  - Rewrite query in more efficient way

Memory Bottleneck

- Memory architecture overview
  - Terms:
    - Virtual Address Space
    - Page file
    - Physical Memory
    - AWE

Memory Bottleneck

- Tools
  - Perfmon
  - DMVs
  - DBCC MEMORYSTATUS
  - Taskmgr
  - Event Viewer, application & system logs

Memory Bottleneck

- Different types of memory pressure
  - External memory pressure
    - Use Taskmgr tool, under performance tab
    - “Available Physical Memory” < 50MB
    - Check Page File size
      - Total commit charge amount exceeds physical memory
      - Page file size is over 2 times of physical memory
    - Need to identify processes consuming most memory
      - Use Taskmgr tool, under process tab, mem usage column
      - Use Perfmon, select Process under Performance Object, select Working Set performance counter
      - Doesn’t show AWE memory

Memory Bottleneck

- Different types of memory pressure
  - Internal memory pressure
    - Memory distribution inside SQL server
    - DBCC MEMORYSTATUS

Buffer Counts	Buffers
Committed	201120
Target	201120
Hashed	166517
Reserved Potential	143388
Stolen Potential	173556
External Reservation	0
Min Free	256
Visible	201120
Available Paging File	460640

    - Target is # of 8K pages buffer pool deems optimal
    - Buffer pool usually is the largest memory consumer under load

Memory Bottleneck

- Different types of memory pressure
  - Internal memory pressure (cont.)
    - Other components use buffer pool for small memory allocations
    - If allocation > 8Kb, use the multi-page allocator interface (memory outside of buffer pool)

```
select type, sum(multi_pages_kb)
from sys.dm_os_memory_clerks
where multi_pages_kb != 0
group by type
```

    - COM objects and linked servers use memory from outside of the buffer pool
    - Need to investigate if multi-page allocation >200MB

Memory Bottleneck

- Internal memory pressure (cont.)
  - Memory consumption by all components
    select * from sys.dm_os_memory_clerks
  - Use script to show major consumers

```
    Total allocated/reserved: 1763 Kb
Component                    Mem allocated/reserved, Mb
------------------------------------------------------------
MEMORYCLERK_SQLBUFFERPOOL          1585
Other                    177
Component                    Mem allocated/reserved, Mb
------------------------------------------------------------
MEMORYCLERK_SQLBUFFERPOOL          1585
USERSTORE_TOKENPERM              78
MEMORYCLERK_SOSNODE              32
CACHESTORE_SQLCP              15
USERSTORE_SCHEMAMGR              14
```

Memory Bottleneck

- Internal memory pressure (cont.)
  - Ring buffer DMV
    select record from sys.dm_os_ring_buffers
    where ring_buffer_type = 'RING_BUFFER_RESOURCE_MONITOR'
  - Track Out Of Memory conditions
    select record from sys.dm_os_ring_buffers
    where ring_buffer_type = 'RING_BUFFER_OOM'
  - Sample record

<Record id="**7301**" type="**RING_BUFFER_OOM**" time="**345640123**">
<OOM>
<Action>**FAIL_VIRTUAL_COMMIT**</Action>
<Resources>**4096**</Resources>
</OOM>

Memory Bottleneck

- Internal memory pressure (cont.)
  - With low memory condition
    - Buffer pool shrinks
    - Turns on low memory notification for other components
  - Query RING_BUFFER_MEMORY_BROKER

    select * FROM sys.dm_os_ring_buffers WHERE ring_buffer_type = 'RING_BUFFER_MEMORY_BROKER'

Memory Bottleneck

- Internal memory pressure (cont.)
  - Virtual address space consumption
    - Find available memory in all free regions

```
SELECT SUM(Size*Free)/1024
  AS [Total avail mem, KB]
FROM VASummary WHERE Free <> 0
```

    - Get size of largest available region

```
SELECT CAST(MAX(Size) AS INT)/1024
  AS [Max free size, KB]
FROM VASummary WHERE Free <> 0
```

    - If largest region < 4MB, likely VM pressure

Memory Bottleneck

- Trouble shooting for common memory errors
  - Is server under external memory pressure?
  - Collect performance monitor counters for SQL Server: Buffer Manager, SQL Server: Memory Manager
  - Verify memory configuration parameters (sp_configure)
    - ‘min memory per query’, ‘min/max server memory’,
    - ‘awe enabled’, ‘Lock pages in memory’ privilege
  - Is server under internal memory pressures?
  - DBCC MEMORYSTATUS
  - Check workload (# of concurrent sessions, # of queries)

Memory Bottleneck

- Common memory errors
  - 701: There is insufficient system memory to run this query
  - Indicates a failed memory allocation
  - Need to check for server memory distribution
  - Solution:
    - Remove external memory pressure
    - Increase server max memory setting
    - Free caches

Memory Bottleneck

- Common memory errors (cont.)
  - 802: There is insufficient memory available in the buffer pool
  - May not indicate out of memory
  - Buffer pool memory could be used by other components
  - Troubleshooting and solution:
    - Similar to 701 error

Memory Bottleneck

- Common memory errors (cont.)
  - 8628: A time out occurred while waiting to optimize the query. Rerun the query
  - Indicates failed memory acquisition during query compilation
  - Troubleshooting and solution:
    - Use general troubleshooting steps
    - DBCC memorystatus
    - Reduce workload if possible

Memory Bottleneck

- Common memory errors (cont.)
  - **8645:** A time out occurred while waiting for memory resources to execute the query. Rerun the query
  - Indicates many concurrent memory intensive queries being executed on the server
    - Sort (order by), join and parallel queries are memory intensive
  - Troubleshooting and solution:
    - Use general troubleshooting steps
    - Identify problematic queries
    - Check sp_configure parameters

IO Bottleneck

- How to detect
  - Perfmon
    - PhysicalDisk Object: Avg. Disk Queue Length > 2
    - Avg. Disk Sec/Read > 0.12, Avg. Disk Sec/Write > 0.12
    - %Disk Time > 50%
    - Avg. Disk Reads/Sec > 85%, Avg. Disk Writes/Sec > 85%
  - Adjustment for RAID
    - Raid 0 -- I/Os per disk = (reads + writes) / number of disks
      Raid 1 -- I/Os per disk = [reads + (2 * writes)] / 2
      Raid 5 -- I/Os per disk = [reads + (4 * writes)] / number of disks
      Raid 10 -- I/Os per disk = [reads + (2 * writes)] / number of disks

IO Bottleneck

- How to detect (cont.)
  - Latch wait
    - Physical IO wait when reading and writing buffer pages

```
select wait_type, waiting_tasks_count, wait_time_ms
from sys.dm_os_wait_stats
where wait_type like 'PAGEIOLATCH%'
order by wait_type
```

- Output

wait_type	waiting_tasks_count	wait_time_ms	signal_wait_time_ms
PAGEIOLATCH_DT	0	0	0
PAGEIOLATCH_EX	1230	791	11
PAGEIOLATCH_KP	0	0	0
PAGEIOLATCH_NL	0	0	0
PAGEIOLATCH_SH	13756	7241	180
PAGEIOLATCH_UP	80	66	0

IO Bottleneck

- How to detect (cont.)
  - Pending IO request

```
select database_id, file_id, io_stall,io_pending_ms_ticks,
 scheduler_address
from sys.dm_io_virtual_file_stats(NULL, NULL)t1,
    sys.dm_io_pending_io_requests as t2
where t1.file_handle = t2.io_handle
```

  - Pending IO request

Database_id	*File_Id*	*io_stall*	*io_pending_ms_ticks*	*scheduler_address*
6	*1*	*10804*	*78*	*0x0227A040*
6	*1*	*10804*	*78*	*0x0227A040*
6	*2*	*101451*	*31*	*0x02720040*

IO Bottleneck

- Resolution:
  - Find out queries generating most IO

```
select top 5 (total_logical_reads/execution_count) as
    Avg_logical_reads,
                 (total_logical_writes/execution_count) as
    Avg_logical_writes,
          (total_physical_reads/execution_count) as Avg_physical_reads,
          Execution_count, statement_start_offset, sql_handle,
    plan_handle
from sys.dm_exec_query_stats
order by (total_logical_reads + total_logical_writes)/execution_count
    Desc
```

  - Examine IO intensive query plans
    - Choose better plan to minimize IO
    - Use Database Tuning Advisor

IO Bottleneck

- Resolution:
  - Check memory configuration
    - Buffer Cache hit ratio
    - Page Life Expectancy
    - Checkpoint pages/sec
    - Lazywrites/sec
  - Increase IO bandwidth
    - Add more disk drives and replace with faster drives
    - Add faster or additional disk controller

Q&A

您的潜力，我们的动力
Microsoft
微软(中国)有限公司

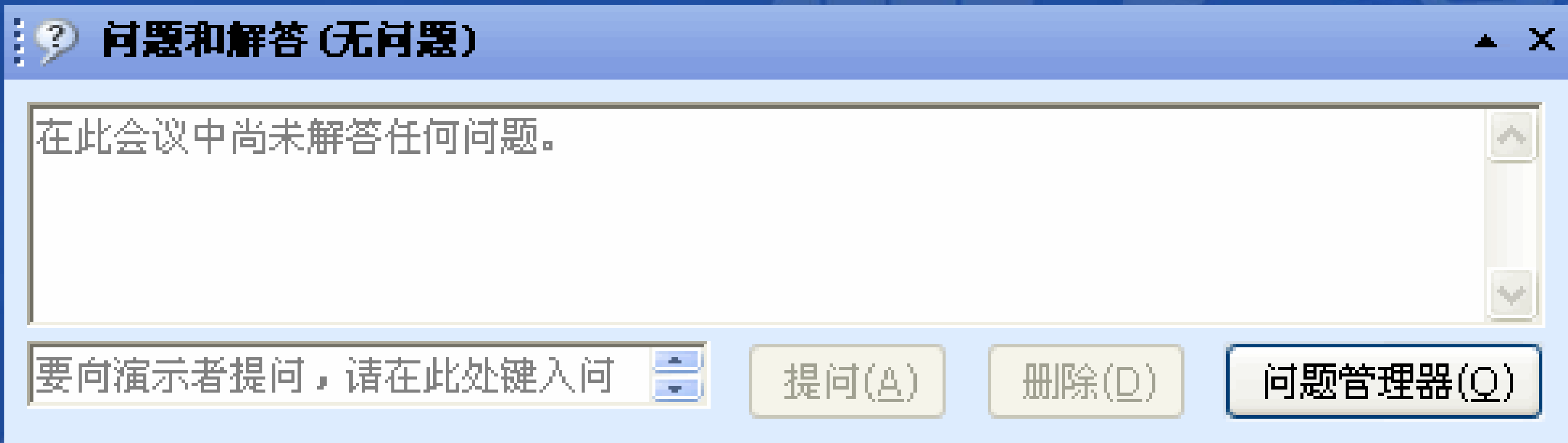
- A lot of information
- See BOL DMV documentation
- Contents in this slide will be published as white paper at www.microsoft.com

获取更多MSDN资源

- **MSDN中文网站**
  http://www.microsoft.com/china/msdn
- **MSDN中文网络广播**
  http://www.msdnwebcast.com.cn
- **MSDN Flash**
  http://www.microsoft.com/china/newsletter/case/msdn.aspx
- **MSDN开发中心**
  http://www.microsoft.com/china/msdn/DeveloperCenter/default.mspx

Question & Answer

如需提出问题，请单击“提问”按钮并在随后显示的浮动面板中输入问题内容。一旦完成问题输入后，请单击“提问”按钮。



您的潜力，我们的动力

Microsoft®

微软（中国）有限公司

Microsoft®

msdn

MSDN Webcasts