
Microsoft
tech·ed
China | 2008
享
IT Professionals
Developers
未来
微软技术大会
tech·ed 2008
Microsoft | Virtualization
Windows HPC Server 2008
SQL Server 2008
Visual Studio 2008

主 要 内 容

Microsoft tech·ed China 2008

OLTP 系统简介
设计技巧及最佳实践
性能优化
客户案例

Agenda

Microsoft tech·ed China 2008

Overview

- What are the characteristics of OLTP?
- What are the goals of OLTP?

Design, Techniques and Best practices

- Transactions
- Concurrency
- Database design
  - Normalization, Denormalization,
  - Index maintenance issues

Identifying Performance issues

- Resource utilization
- Optimization
  - Estimation and query plan selection
  - Plan re-use & Recompilation
- Useful counters
- OLTP Performance Blueprint

课程编号：DAT351

OLTP 系统简介

- OLTP在线交易系统特性
- OLTP在线交易系统目的

OLTP 在线交易系统特性

Microsoft tech·ed China 2008

大量短而重复的事务/交易

- 处理的数据量较小
- 较大的并发量
- 可预计的数据使用状态 Predictable access patterns

OLTP 系统设计的考虑因素

- 事务/交易的设计 Transactional design
- 数据库的设计 Database design
- 性能的目标 Performance objectives

数据库管理人员的关注

- 锁 Locking/Blocking
- 随机的 I/O Random IO
- 大量的并发 High number of concurrent sessions

OLTP 系统设计性能目标

Microsoft tech·ed China 2008

OLTP performance objectives

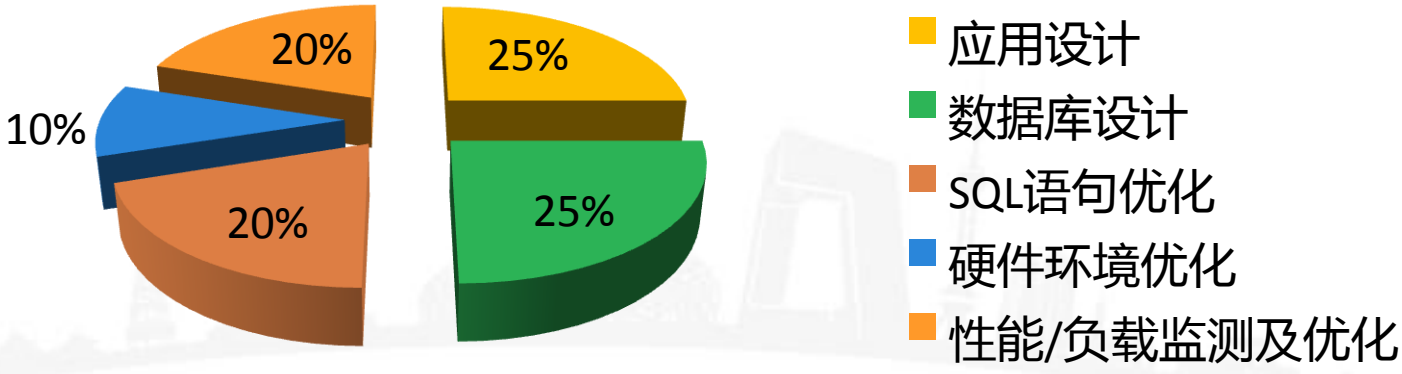
- 快速的事物处理 Fast transactions
  - Set operations preferable over Cursors
  - Indexes allow granular data access and locking
- 优化CPU资源 Maximizing CPU resources
  - High plan re-use
  - Low re-compilation
- 优化I/O资源 Maximizing IO resources
  - Minimize joins
  - Fast transaction log (writelog)
  - Small IOs for Data (io_completion)
  - See SEAS06PT (Performance Tuning) deck

设计技巧及最佳实践

- 事物设计
- 并发
- 数据库设计

设计、调优及最佳实践对系统性能的影响

- 应用、数据库的设计和T-SQL语法对系统性能产生主要的影响
- 典型的系统性能优化影响并不很大
- 性能监测可以帮助在整个系统应用中找到性能瓶颈/优化的重点



Design, Techniques, Best Practices Impact on Performance

Application & Database Design and T-SQL have major impact on performance

'Typical' performance tuning plays smaller role

Performance monitoring can point out some deficiencies (or opportunities for improvement!) throughout application life cycle

App Design	DB Design	SQL	Hardware tuning	Performance Monitoring workload changes
25%	25%	20%	10%	20%

事物、交易的设计

OLTP workload is a mix of selects, inserts, updates, deletes

Consistency and Concurrency

- Locks ensure data consistency
- Incompatible locks can cause blocking
- Short transactions are key for high concurrency in OLTP
- New row versioning isolation levels reduce read / write blocking

Some OLTP environments include reporting requirements which typically cause blocking

- Big transactions e.g. shared locks can be segregated from OLTP using database snapshots or row versioning

Design: Transactions

- OLTP workload is a mix of selects, inserts, updates, deletes
- Consistency and Concurrency
  - Locks ensure data consistency
  - Incompatible locks can cause blocking
  - Short transactions are key for high concurrency in OLTP
  - New row versioning isolation levels reduce read / write blocking
- Some OLTP environments include reporting requirements which typically cause blocking
  - Big transactions e.g. shared locks can be segregated from OLTP using database snapshots or row versioning

数据库设计之一
规范化

数据库设计技巧
- 数据库设计中逐渐减少数据冗余的规则Process of applying increasingly restrictive design rules
- 5种不同的规范，通常使用第三规范（3rd Normal Form)
- Use ORM when designing 4th and 5th NF
- Eliminates redundant data, shrinks row sizes, more tables

优势
- 因为数据行窄，搜索、排序、建立索引更加快速，单一数据页可以储存更多行数据记录
- 理论上结构更清晰，容易危害，更改数据较为直接明了

劣势
- 需要更多表联接

Design: Database 2

Denormalization

Selective relaxation of normalization rules

- based on thorough knowledge of the application
- only if performance issues indicate that it is needed

Benefits

- Minimizing the need for joins
- Reducing the number of foreign keys on tables
- Reducing the number of indexes, saving storage space, and reducing data modification time
- Pre-computing aggregate values at data modification time rather than at select time. Compare update cost vs. select cost
- Reducing the number of tables (in some cases)

Disadvantages

- It usually speeds retrieval but can slow data modification
- application-specific, should be reevaluated if the application changes
- It can increase the size of tables
- In some instances, it simplifies coding; in others, it makes coding more complex

Design: Database 3

Normalization vs. denormalization tradeoffs

Microsoft tech·ed China 2008

Highly normalized models require multi-table joins

- Joins create work tables, tempdb activity
- High concurrency performance objective: reduce joins
- Question?
  - Do we always join 6 tables to get the most trivial information? Maybe too normalized.

Tradeoffs

- Flexibility vs. performance
- Balancing denormalization issues
  - Tradeoff of update cost vs. select cost
  - Few updates (I,U,D) vs. Many selects: favors denormalization
  - Many updates (I,U,D) vs. Few selects: favors normalization

The most common denormalization techniques are:

1. Adding redundant or derived columns
2. Adding indexed views
3. Collapsing tables
4. Splitting tables (rarely used columns go in 2nd table with

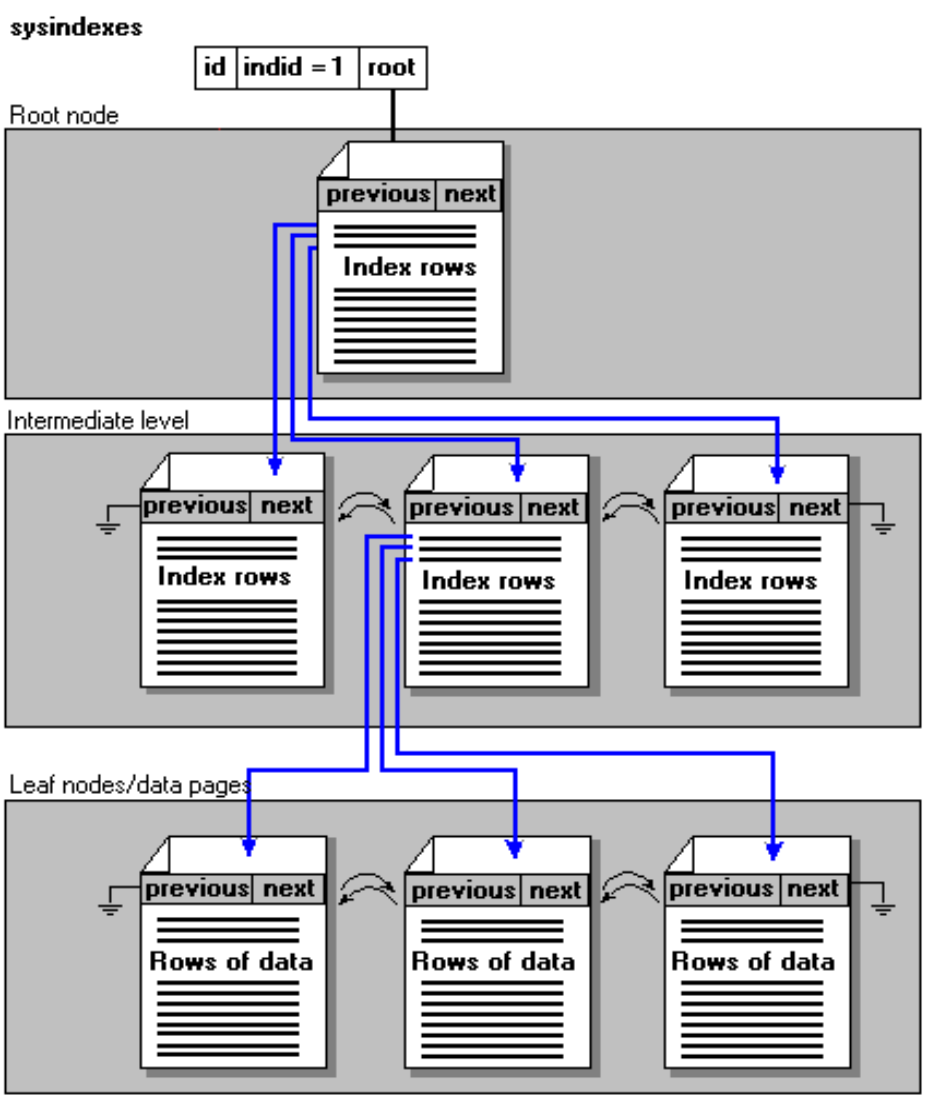
Design: Indexes 1

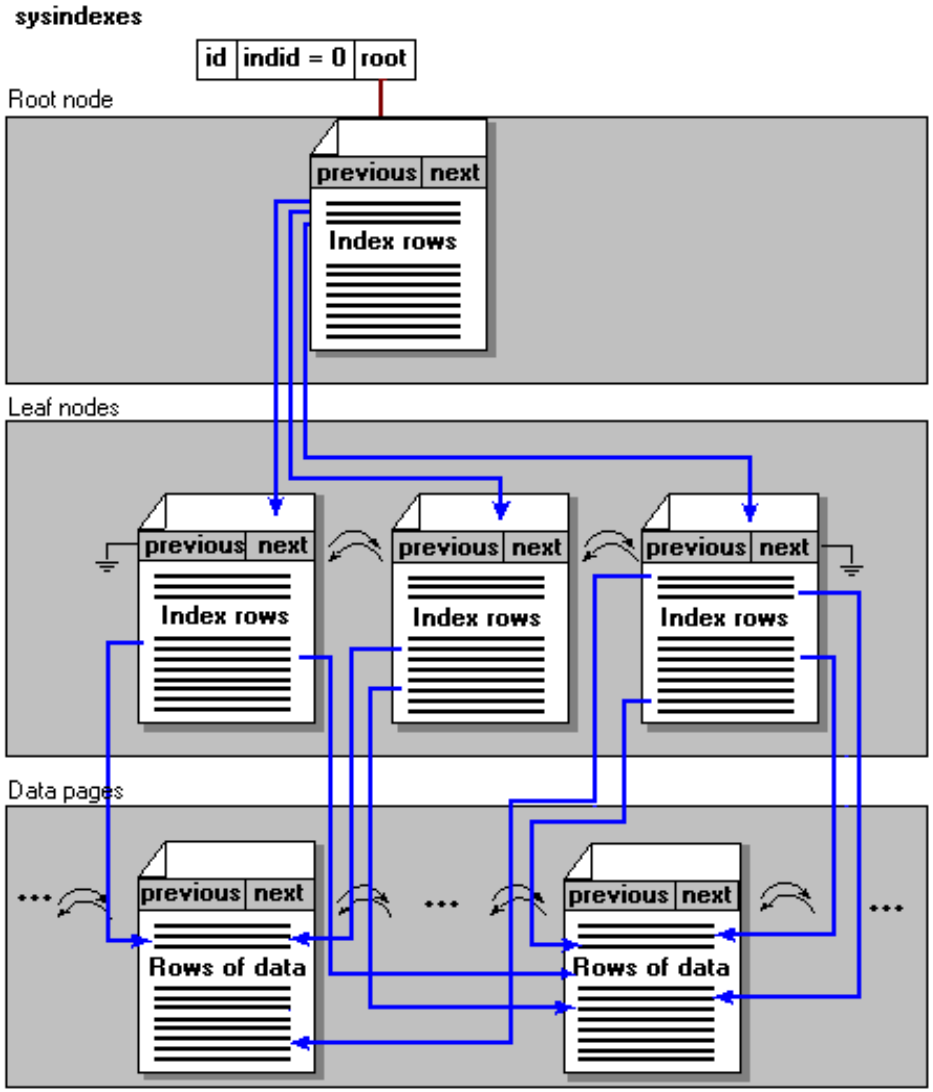
Overview

Index issues

- Indexes provide alternatives to table scans
- OLTP typically has fewer indexes than DSS / reporting
  - Trade off of index usage vs maintenance costs
  - OLTP Indexes should be designed for the predictable, repetitive nature of this workload
- It is preferable to have a clustered index on large tables
- Which index should be clustered?

Indexes


sysindexes
id
indid = 1
root
Root node
previous
next
Index rows
Intermediate level
previous
next
Index rows
previous
next
Index rows
previous
next
Index rows
Leaf nodes/data pages
previous
next
Rows of data
previous
next
Rows of data
previous
next
Rows of data


sysindexes
id
indid = 0
root
Root node
previous
next
Index rows
Leaf nodes
previous
next
Index rows
previous
next
Index rows
previous
next
Index rows
Data pages
previous
next
Rows of data
previous
next
Rows of data

Design: Indexes 2

index types

Clustered

- order imposed on data rows
- Leaf is data row
- Row length includes data row
- Non-dense
  - First row on data page recorded in intermediate pages
- Advantage of moving data to other file groups if needed

- Nonclustered
  - Order imposed on index, not on data rows
  - Leaf contains locater to data row
    - PKey if clustered index present
    - RID if no clustered index
  - Row length includes nonclustered index columns and row locater
  - Dense
    - all rows represented at leaf
  - Covering Index
    - When nonclustered index satisfies query ***without*** going to data rows, this is a covered query

Design: Indexes 3

maintenance and performance

- IO story – alternatives to table scans
- Maintenance cost vs. benefit
  - On delete, insert, update maintenance
  - Clustered indexes
    - Page splits
  - Nonclustered indexes
    - Heap tables and forwarded records
- Randomization
  - Indexes can "randomize" insert/update/delete activity (examples: Name or PhoneNum)
    - Avoids hot spot (blocking) but can cause page splits
    - Will need more frequent index reorgs / rebuilds
    - Especially useful with table partitioning to spread IO over multiple partitions.
- Ascending keys
  - Can cause hot spots (e.g. blocking)
  - Row level locking minimizes blocking

Design: Indexes 4

Recommendations

- *Avoid* long (or wide) clustered index key if table has nonclustered (N/C) indexes
  - Leaf of Nonclustered index uses the clustered index key (primary key) to locate the data row
    - Since a wide clustered index key increases size of N/C, (covered) nonclustered range scans results in more IO
- *Avoid* high volume Clustered index seeks & RID lookups (N/C)
- Clustered index benefits
  - high volume lookups (avoids RID lookups)
  - Range scans – access to entire data row

Non-clustered index benefits

Index DMVs & DMFs

See details in SEAS06PT: SQL 2005 Perf Tuning

Missing indexes

- Sys.dm_db_missing_index_group_stats
- Sys.dm_db_missing_index_groups
- Sys.dm_db_missing_index_details
- Sys.dm_exec_query_plan(plan_handle) - Look for <MissingIndexes>

Unused indexes

- Sys.dm_db_index_usage_stats

Index Access, Blocks, Contention e.g. waits

- Sys.dm_db_index_operational_stats()
- Sys.dm_db_index_physical_stats()

Agenda

Microsoft tech·ed China 2008

Overview

- What are the characteristics of OLTP?
- What are the goals of OLTP?

Design, Techniques and Best practices

- Transactions
- Concurrency
- Database design
  - Normalization, Denormalization,
  - Index maintenance issues

Identifying Performance issues

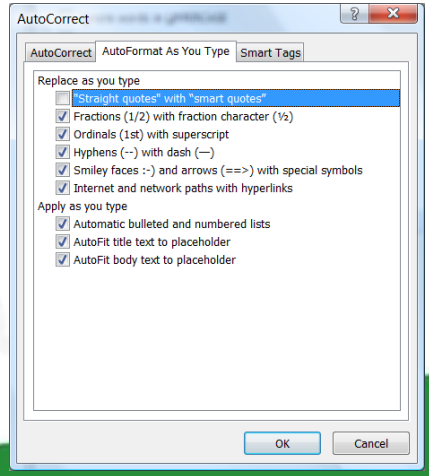
- Resource utilization
- Optimization
  - Estimation and query plan selection
  - Plan re-use & Recompilation
- Useful counters
- OLTP Performance Blueprint

点击此处修改二级标题

- 将幻灯片标题设置为 “Title Case”
- 标题字号为 40分或类似尺寸，并尽可能不换行
- 将二级标题设置为 “sentence case”
- 二级标题的字体颜色已经在PPT模板中定义

• 代码字体和字号

请用次版面展示软件代码
请用展示字体为Courier New 这一单一空间字体



模板颜色

- 字体，字号和颜色模式已经在PPT模板中定义
- 请使用下面展示的颜色模式
- 更多指示请参见下页
- 超文本衔接颜色: www.microsoft.com

Sample Fill | Sample Fill | Sample Fill | Sample Fill | Sample Fill

Agenda

Microsoft tech·ed China 2008

Overview

- What are the characteristics of OLTP?
- What are the goals of OLTP?

Design, Techniques and Best practices

- Transactions
- Concurrency
- Database design
  - Normalization, Denormalization,
  - Index maintenance issues

Performance issues

- Resource utilization
- Optimization
  - Estimation and query plan selection
  - Plan re-use & Recompilation
- Useful counters

Summary

Performance issues

What hinders Performance?

Queuing

- Multiple types of queues (memory, CPU, IO)

Resource limitations

Bad configuration

- Hardware & Software

Bad Queries & Design

- Badly written, poorly designed

Poor indexing

- Not relevant to workload or lack of

Inappropriate optimizer plans

Too many round trips/Network

Performance issues

Shared Resources, Scalability Limits

Database shared resources

- Database performance is limited by maximum **Transaction Log** throughput, only ONE possible transaction log per database!
- Can be resolved by
  - adding multiple spindles
  - Increasing number of databases to provide multiple transaction logs

Server shared resources

- TEMPDB
  - Tempdb in memory vs. less memory for buffer cache
- Memory (64-bit) flat (see SEAS06 SQLOS & VLDB)
- Memory (32-bit)
  - Only data cache can live in 32-bit AWE
  - Proc cache, locks, user connections, sorting restricted to lower 2-3GB of address space
- Can be resolved by partitioning over multiple instances

Machine/node shared resources

Performance Issues
Scalability Rules

Database scalability is limited by the maximum throughput of the transaction log
- Disk I/O

Instance scalability is limited by shared "process" level resources
- Memory

Server scalability is limited by shared "server" / "machine" level resources
- CPU (incl. L1 & L2 cache)
- Network bandwidth

Performance Issues
Disk I/O

Microsoft tech·ed China 2008

Determine I/O pattern
- Writes
  - Transaction Log (~100% sequential)
  - Lazy Writer (random)
- Read
  - Random vs. Sequential

Establish disk I/O baseline or SLA outside SQL Server, using:
- SQLIO or IOMeter (Intel, public domain)

Special cases:
- Transaction log
- 1 Tempdb file for each cpu
- Max Parallel BCP load = 1 BCP / CPU
  - Into SQL Server 2005 partitioned tables

Performance Issues
I/O Bottlenecks 1

I/O bottlenecks are typically easy to find

Be very careful with the transaction log

- Beyond 12 to 15 spindles doesn't buy much
- Keep on separate physical disks for recovery
- Make RAID 10

Beware of write cost on RAID5:

- In RAID 5 each write has to logically read old data + old parity (to compute parity) and write new data and new parity
- Each RAID5 write = 2 READS + 2 WRITES !
  - However: Disk guys work real hard to optimize this
  - Recent bulk load tests showed ~50%

Performance Issues
I/O Bottlenecks 2

Disk subsystem based on I/O throughput required, not size of DB

- E.g. 1TB data / 72GB per drive = 14 drives.
  - Will 14 drives provide sufficient IO throughput?
  - Recommend more smaller drives
  - Random (OLTP) vs. sequential (Reporting) IO/sec
  - Cache on controller – tuned for % read or write

Consider all workloads

- OLTP (typically random IOs)
- Batch (could be random or sequential depending on the type of work done)

Use SQLIO to measure your max throughput

Performance Issues

Optimizing for the log

Profile the log disk

- How many writes / second can your disk sustain?

Keep the log disk purely for the log

- Keeps the disk heads writing sequentially minimizing seeks

Beware of unprotected write back caches

- If power fails, you could lose the entire database – not just the last couple of transactions!
- Check with your SAN / Disk controller vendor

Performance Issues
Blocking

Blocking between sessions can occur due to a combination of incompatible locks and waits on resources

Tools

- Use Profiler block process report and other tools to find blocking processes
- DMVs

New blocking solutions

- Snapshot Isolation - Row Versioning

See SEAS06PT for locking discussion

Performance Issues

How to Evaluate Blocking

DMF sys.dm_db_index_operational_stats() identifies the contention points

- Row locks counts
- Row lock waits counts
- Total wait time for blocks
- Compute blocking percentage and average wait times
- See SEAS06PT Indexes & Row Lock Waits.sql

Finding Resource Bottlenecks Identifying Blocking & Concurrency issues

Sp_block_info – lists real time blocks

Trace – for historical analysis

- ✓ Capture long blocks using the Trace Event "Block Process Report"
  - ✓ Sp_configure "blocked process threshold",15 (seconds)
  - ✓ This is covered in SEAS06PT

- ✓ If blocking is still an issue, Consider row versioning to minimize read / write contention

Performance Issues

row versioning: new blocking solutions

Row versioning-based isolation levels

- Always read a committed value (as compared with dirty reads)
- Reads do not acquire shared (S) locks
- improve concurrency by eliminating blocks for read / write operations.
- Tempdb overhead
  - Stores versions of previously committed row data
- RCSI
  - Advantage: NO APPLICATION CHANGES !
  - Transaction Isolation Level Read Committed & Read_Committed_Snapshot ON database option
  - Statement level read consistency
- Transaction Isolation Level Snapshot

Performance & Resources

TempDB Usage

Tempdb usage is much more common in SS2005

- Tempdb management must be a configuration priority for DBAs

The following uses Tempdb w/ SS2005

1 DBCC CHECKDB - small change
2 Internal objects: work file (hash join, SORT_IN_TEMPDB) - CTEs
3 Internal objects: work table (cursor, spool) - small changes
4 Large object (LOB) variables
5 Service Broker
6 Temporary objects: global/local temp table, table variables
7 Temporary objects: SPs and cursors - small changes
8 Version store: General
9 Version store: MARS
10 Version store: Online index
11 Version store: Row version based isolation levels
12 Version store: Triggers
13 XML

Performance Issues

TempDB capacity planning

Microsoft tech·ed China 2008

On line index:

- 2x-3x size of index – Sort size, temp index and rollback

Versioning:

- [Size of Version Store] = 2 * [Version store data generated per minute] * [Longest running time (minutes) of your transaction] * number of concurrent transactions/users
- Note: Version store data generated per minute and version store size are now perfmon parameters

Performance Issues

Tempdb – Trace Flag 1118

Reduces sgam contention

Still needed in 2005 if you have DDL statements for Create Table and Create Index in stored procedures that are called many times (high volume).

Performance & Resources

Tempdb – Space Used

```
select
    sum(user_object_reserved_page_count)*8 as
    user_objects_kb,
    sum(internal_object_reserved_page_count)*8
    as internal_objects_kb,
    sum(version_store_reserved_page_count)*8
    as version_store_kb,
    sum(unallocated_extent_page_count)*8 as
    freespace_kb
from sys.dm_db_file_space_usage
where database_id = 2
```

Performance & Resources

Tempdb usage: by sql_handle & plan_handle

```
SELECT t1.session_id,
(t1.internal_objects_alloc_page_count + task_alloc) as allocated,
(t1.internal_objects_dealloc_page_count + task_dealloc) as deallocated
     , t3.sql_handle, t3.statement_start_offset
     , t3.statement_end_offset, t3.plan_handle
from sys.dm_db_session_space_usage as t1,
             sys.dm_exec_requests t3,
(select session_id,
   sum(internal_objects_alloc_page_count) as task_alloc,
   sum (internal_objects_dealloc_page_count) as task_dealloc
      from sys.dm_db_task_space_usage group by session_id) as t2
where t1.session_id = t2.session_id and t1.session_id >50
and t1.database_id = 2   --- tempdb is database_id=2
and t1.session_id = t3.session_id
order by allocated DESC
```

Performance & Resources
Database Snapshot

- Database snapshots do consume resources on your server.
  - Example: Buffer Pool
- Tested – TPC-C workload had 15% performance loss with single Database snapshot
- The more database snapshots, the more performance will be impacted.

Performance Issues

3rd Party Performance Tools

Veritas (formerly Precise) InDepth for SQL Server

- Excellent tool for identifying
  - Resource bottlenecks
  - Resources consumed by statement
  - Waits by statement
  - Performance history

Quest Software

- Great tools for monitoring
- Partition management
- Backup with compression (Litespeed)

Performance Issues

Language vs. RPC Events

Server has two distinct and optimized code paths

- Goal is to utilize the correct code path!

Language event

- Every statement not being a (stored) procedure
- extra parsing required to figure out what is in the string
- Adhoc query plans for string (in addition to Stored Proc plans)
- Generic code which executes procedures via a language event, for example OSQL, Query Analyzer etc.

RPC event

- Stored procedure invocations using {call} syntax

Performance Issues

API - Benchmark lessons

Microsoft tech·ed China 2008

OLTP Benchmark lessons

- Big performance gains from best practices
  - Use efficient row length and data types
    - Every byte counts, use correct types
  - Match packet size and batch size
    - Perf of ‘Bind’ on client proportional to batch size
      - For large batches, avoid ODBC Parameter binding with ?
  - ODBC {Call Proc} better than execute proc syntax
    - {call dbo.qi ('M01', 'M01.040704000000002')}
    - exec dbo.qi @v1='M01', @v2='M01.040704000000002' –adds ADHOC query plans due to SQL string parsing
  - Net gain using above - 7x

Performance Issues

Results Handling / Round trips

You always fetch all results and all result sets!

Un-fetched results and result sets can cause concurrency issues on the server

Un-fetched results and result sets will cause an attention signal to be send to the server to cancel the pending stream

SET NOCOUNT ON

- Avoid unnecessary round trips of sending empty result sets for INSERT, UPDATE and DELETE statements

Performance Issues

Cached Objects & plan re-use

Master..Sys.dm_exec_cached_plans

- Procedure or batch name
- *Set* options for plans
- Ref counts, Use counts
- Compiled plan
  - Single copy (serial and parallel)
  - Re-entrant and re-usable
  - Statement level recompilation
- Executable plan
  - Data structure for user context, not re-entrant
  - Look for plan reuse: usecounts > 1

Plan re-use of

- Procs, Triggers, Views
- Defaults, Check constraints, rules

Performance Issues

Cached Objects & plan re-use

- SQL Batch requests/sec
  - Compare to initial SQL Compilations/sec
- SQL Compilations/sec
  - Includes initial compiles AND re-compiles
  - Eliminate re-compilations to get initial compiles
  - Look for identical SQL statements with low usecounts in Sys.dm_exec_cached_plans
    - See SEAS06PT:Worst plan re-use by statement.sql
- SQL Re-compilations/sec
  - Statement Level Recompiles
  - Sys.dm_exec_query_stats (plan_generation_num)
    - when incremented indicates recompilation
  - Check profiler for sp:recompile event to identify SQL statement.

http://www.microsoft.com/technet/prodtechn

Performance & Resources

CPU Utilization

Waiting to run

- Runnable queue – pure CPU waits
- CPU pressure measured by signal waits

Plan compilation & requests

- Perfmon: SQLServer:SQL Statistics
  - Batch requests / sec { >1000' s/sec server is busy}
  - SQL Compilations / sec {>10s/sec could be problem}
  - SQL Recompilations / sec {OLTP should avoid high recomps}
- Ratio of compiles / requests is important
  - Compiles – recompiles = initial compiles
  - Plan re-use = (Batch requests – initial compiles) / Batch requests
    - (compared with batch requests, low initial compiles indicates plan re-use)
- Recompile reasons:
  - Change in schema state – schema altered, etc.
  - Previously parallelized plan needs to run serially

Performance Issues

Plan re-use vs. CPU usage

CPU used for plan determination

- OLTP characterized by high numbers of identical small transactions
  - Plan re-use desirable
  - See usecounts in Sys.dm_exec_cached_plans

Stored procedure estimates are based on initial parameter values

- Re-use is generally good for OLTP,
- re-use can be bad when when results sets can *significantly vary in size*.

Performance Issues

Plan estimation & re-use issues

- Plan selection is based on estimates
  - *Set Statistics Profile on*
    - Shows estimates vs. actuals
    - Look for huge differences (examples)
      - OverEstimates are 100x actuals
      - UnderEstimates are 1% actuals
- Overestimation
  - Favors fixed cost (hash) strategy
  - Extreme cases can improve
    - with LOOP JOIN hint
    - Execute P1 with recompile
- Underestimation
  - Favors variable cost (e.g. nested loops) strategy

Performance Issues
Profiler events

Plan re-use (or lack of)

- Compare batch requests to SQL compiles/sec

IO

- Reads and writes

Recompilation

Cache hit, insert, miss, remove

Index usage (or lack of)

Object access

Performance Issues

Profiler events for query plans

The Profiler events that track cache management include:

- SP:CacheMiss (event ID 34 in Profiler)
- SP:CacheInsert (event ID 35 in Profiler)
- SP:CacheRemove (event ID 36 in Profiler)
- SP:Recompile (event ID 37 in Profiler)
- SP:CacheHit (event ID 38 in Profiler)

SP:Starting lists stored procedure execution

SP:StmtStarting will show corresponding SQL statement

- Example: sequence is
  - SP:StmtStarting
  - SP:CacheMiss (no plan found)
  - SP:CacheInsert (plan created)
- *Watch out: Heavy profiler use will affect performance !*

Add *Eventsubclass* data column to display recompilation reason

Performance Issues

CPU: Recompilation

Plan determination is CPU Intensive

- Recomp good if benefit of new plan > CPU cost

Profiler

- Lists recomp events and statements
- Data column for reason: **EventSubClass**

locks on system tables

- Re-compiling stored procedure plans serialize other users during high concurrency
  - places lock on single compile plan

Re-compilation based on

- Rows changed thresholds (rowmodctr)
- DDL placement, schema changes
- Code practice & temp tables (P1 & P2)

Performance Issues

EventSubClass	Description
1	Schema changed.
2	Statistics changed.
3	Deferred compile.
4	SET option changed.
5	Temporary table changed.
6	Remote rowset changed.
7	FOR BROWSE permission changed.
8	Query notification environment changed.
9	Partitioned view changed.
10	Cursor options changed.
11	OPTION (RECOMPILE) requested.

Performance Issues

Useful Performance Counters

Memory: Page faults/sec
Memory: pages/sec
Physical Disk: Avg. Disk Queue Length
Physical Disk: Avg. Disk sec/Transfer
Physical Disk: Avg. Disk sec/Read
Physical Disk: Avg. Disk sec/Write
Physical Disk: Current Disk Queue Length
Processor: %Processor Time
SS Access Methods: Forwarded Records/sec
SS Access Methods: Full Scans/sec
SS Access Methods: Index Searches/sec
SS Access Methods: Page Splits/sec
SS Access Methods: Range Scans/sec
SS Access Methods: Table Lock escalations/sec
SS Buffer Manager: Checkpoint pages/sec
SS Buffer Manager: Lazy writes/sec
SS Buffer Manager: Page Life expectancy
SS Buffer Node:Foreign Pages
SS Buffer Node:Page Life expectancy
SS Buffer Node:Stolen Pages

SS Databases: Log Flush Wait time
SS Databases: Log Flush Waits/sec
SS General Statistics: User Connections
SS Latches: Average Latch Wait Time(ms)
SS Latches: Latch Waits/sec
SS Latches: Total Latch Wait Time (ms)
SS Locks: Average Wait Time(ms)
SS Locks: Lock requests/sec
SS Locks: Lock Wait Time (ms)
SS Locks: Lock Waits/sec
SS Memory Manager: Memory grants pending
SS SQL Statistics: Auto-Params attempts/sec
SS SQL Statistics: Batch requests/sec
SS SQL Statistics: Safe Auto-Params/sec
SS SQL Statistics: SQL Compilations/sec
SS SQL Statistics: SQL Re-Compilations/sec
System: Processor Queue Length

Agenda

Overview
- What are the characteristics of OLTP?
- What are the goals of OLTP?

Design, Techniques and Best practices
- Transactions
- Concurrency
- Database design
  - Normalization, Denormalization,
  - Index maintenance issues

Identifying Performance issues
- Resource utilization
- Optimization
  - Estimation and query plan selection
  - Plan re-use & Recompilation
- Useful counters
- OLTP Performance Blueprint

OLTP Performance Blueprint

DB Design (values can be debated)

Resource Issue	Rule	Description	Value	Source	Problem Description
DB Design	1	High Frequency queries having # table joins	>4	Sys.dm_exec_sql_text Sys.dm_exec_cached_plans	High Frequency queries with lots of joins may be too normalized for high OLTP scalability
	2	Frequently updated tables having # indexes	>3	Sys.indexes sys.dm_db_operational_index_stats	Excessive index maintenance for OLTP
	3	Big IOs range scans table scans	>1	Perfmon object SQL Server Access Methods Sys.dm_exec_query_stats	Missing index, flushes cache
	4	Unused Indexes	index not in*	* Sys.dm_db_index_usage_stats	Index maintenance for unused indexes

OLTP Performance

Resource Issue	Rule	Description	Value	Source	Problem Description
IO	1	Avg Disk seconds / read	> 10 ms	Perfmon object • Physical Disk	Reads should take 4-8ms with NO IO pressure
	2	Avg Disk seconds / write	> 10 ms	Perfmon object • Physical Disk	Writes (sequential) can be as fast as 1ms for transaction log.
	3	Big IOs range scans table scans	>1	Perfmon object • SQL Server Access Methods	Missing index, flushes cache
	4	If Top 2 values for Wait stats includes: ASYNCH_IO_COMPLETION IO_COMPLETION LOGMGR WRITELOG PAGEIOLATCH_x	Top 2	Sys.dm_os_wait_stats	If top 2 wait_stats values include IO, there is an IO bottleneck

OLTP Performance Blueprint

Resource Issue	Rule	Description	Value	Source	Problem Description
Blocking	1	Block percentage	>2 %	Sys.dm_db_index_operational_stats	Frequency of blocks
	2	Block process report	30 sec	• Sp_configure “blocked process threshold” • profiler “blocked process report”	Report of long blocks e.g. statements
	3	Avg Row Lock Waits	> 100 ms	Sys.dm_db_index_operational_stats	Duration of blocks
	4	If Top 2 values for wait stats are any of the following: 1. LCK_x	Top 2	Sys.dm_os_wait_stats	If top 2 wait_stats values include locking, there is a blocking bottleneck

OLTP Performance Blueprint

Resource Issue	Rule	Description	Value	Source	Problem Description
CPU	1	Signal Waits	> 25%	Sys.dm_os_wait_stats	Time in runnable queue is pure CPU wait.
	2	Plan Re-use	< 90%	Sys.dm_os_wait_stats Perfmon object • SQL Server Statistics	OLTP identical transactions should ideally have >95% plan re-use
	3	Parallelism: CXPACKET waits	> 5%	Sys.dm_os_wait_stats	Parallelism reduces OLTP throughput

OLTP Performance

Resource Issue	Rule	Description	Value	Source	Problem Description
Memory	1	Average Page Life Expectancy	< 300 sec	Perfmon object • SQL Server Buffer Mgr • SQL Server Buffer Nodes	1. Cache flush,due to big read 2. Possible missing index
	2	Average Page Life Expectancy	Drops by 50%	Perfmon object • SQL Server Buffer Mgr • SQL Server Buffer Nodes	1. Cache flush,due to big read 2. Possible missing index
	3	Memory Grants Pending	> 1	Perfmon object • SQL Server Memory Manager	Current number of processes waiting for a workspace memory grant

Agenda

Overview
- What are the characteristics of OLTP?
- What are the goals of OLTP?

Design, Techniques and Best practices
- Transactions
- Concurrency
- Database design
  - Normalization, Denormalization,
  - Index maintenance issues

Identifying Performance issues
- Resource utilization
- Optimization
  - Estimation and query plan selection
  - Plan re-use & Recompilation
- Useful counters
- OLTP Performance Blueprint

OLTP Summary

Lessons learned

- Challenge: Scheduling a mix workload evenly across Schedulers
- Database Log to handle 60,000+ database tx/sec
- Real time reporting and loading data
  - Indexes are both good and bad
- OLTP general goal: limit recompiles
- Multiple database logs for scalability
- Read-only queries: consider another database via replication, log shipping or Shared Scalable Database

OLTP Summary

Gotchas

Database design driven by workload requirements

- Indexes
- Denormalization decisions
- Transactions

Maximizing resources

- Plan re-use – normally desirable for OLTP
- Recompilation – generally try to avoid with OLTP
- Set based operations more efficient than cursors
- Reduce parallel queries to improve concurrency
  - Sp_configure “max degree of parallelism”,1 -- turns off

OLTP Summary

Microsoft tech·ed China 2008

OLTP applications require appropriate

- database design
  - Index usage
- Transaction usage
  - High concurrency - must minimize blocking
- Application design
  - Use code coding techniques for plan re-use, minimize recompiles
- API
  - Maximize performance with most efficient calls
- Access methods
  - Efficient query plans for OLTP

Other Resources

SQL Server 2005 Batch Compilation, Recompilation, and Plan Caching Issues

- http://www.microsoft.com/technet/prodtechnol/sql/2005/recomp.mspx

SQL Customer Advisory Team internal site

- http://sqlserver/sites/sqlcat

SQL Customer Advisory Team blog

- http://blogs.msdn.com/sqlcat

SQL Server Webcasts

- http://www.microsoft.com/technet/prodtechnol/sql/webcasts/default.mspx

Appendix

SS2005 Technical Learnings

- Upgrade
- Management – DMV(s), Profiler, Perfmon, DQ tracing
- TempDB
- Database Mirroring, DB Snapshot
- Development – MARS, CLR, UDTcur
- Security, Catalog

Upgrading to SS2005

Most customer testing: 300-400 customer applications worldwide with Application Compatibility labs

Upgrade Advisory is a must

- Get the latest version http://microsoft.com/sql
- Go back and run it against your SS2K database systems

Upgrading to SS2005
the biggest findings

Microsoft tech·ed China 2008

SS2K statistics invalid after upgrade

- Update statistic will kick in automatically upon first execution of queries using sample data
- Default update statistics sampling could be very small for very large tables, possibly <1%
- Recommendation: manually update statistics after upgrade. Full if possible, suggest: 10% for Very Large tables.
  - Example: *Update statistics sales.salesorder with sample 10 percent;*

Recommendation: Remember to run surface area configuration manager

- May not recognize an important component is off by default: named pipes, Service Broker, CLR, FTS, Dedicated Administration Connection, etc.

Upgrading to SS2005
the biggest findings

Recommendation: Specify the WITH keyword when using table hints

- Optional in SS2K / and SS2005. May be mandatory with next version.
- Using WITH is ANSI SQL Compliant
- Example: ***UPDATE Production.Product WITH (TABLOCK) SET ListPrice = ListPrice * 1.10 WHERE ProductNumber LIKE 'BK-%'***

Recommendation: Remove references to undocumented system tables

Using Integrated Security and have Windows group called SA will break install !

Engine – Lessons Learned cont

SOS – SQL Server Operating System

- User Mode Scheduler – CPU management
- Memory management
- IO Completion port

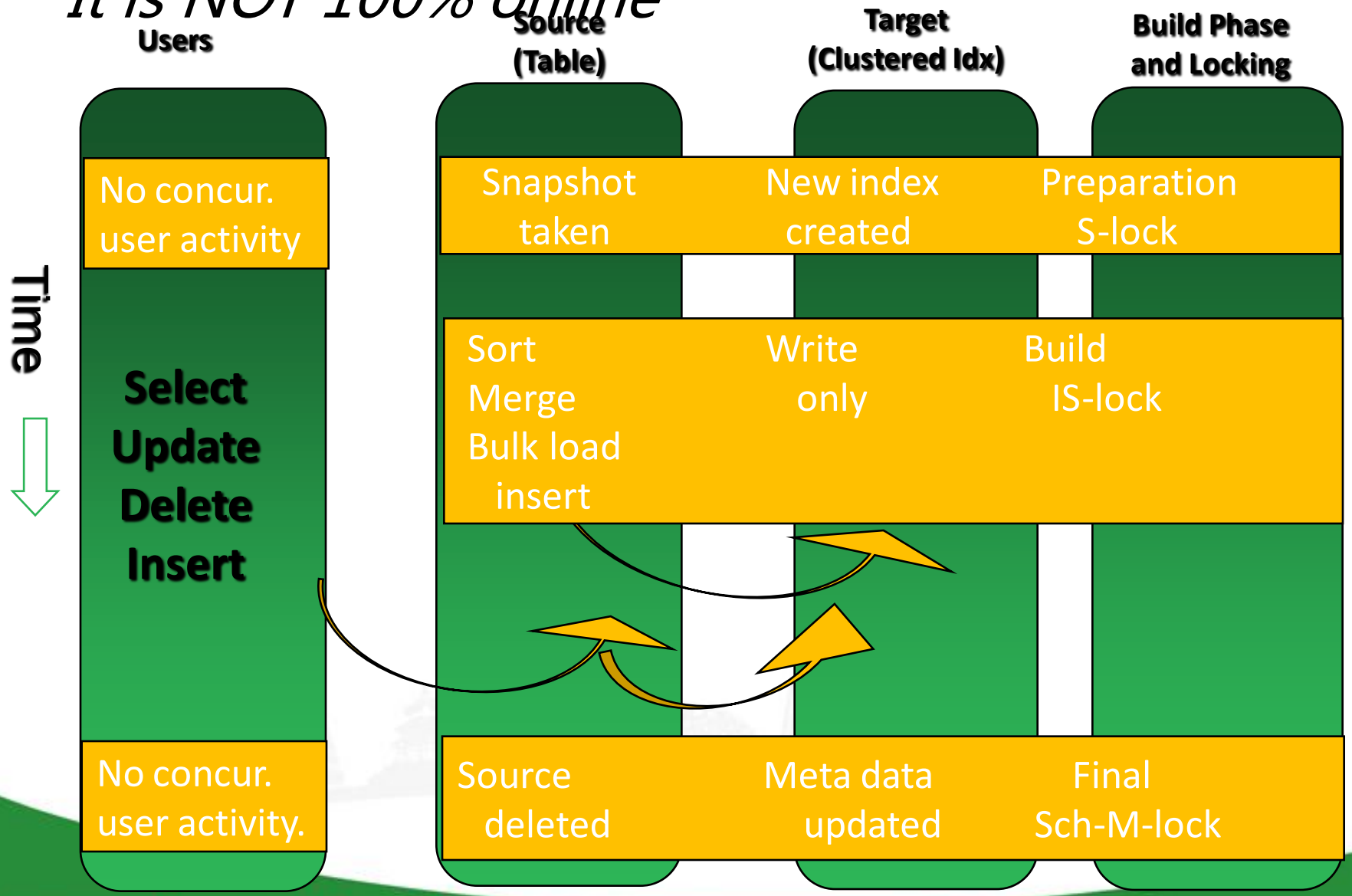
In SS2K a connection was assigned to a UMS and stayed there and in SS2005, a thread within a connection is what is assigned to a UMS

- See example next slide

Dirty reads can bypass schema locks. Will be fixed in future SP.

Online Index Build – Lessons Learned

It is NOT 100% online



Index Operations Lessons Learned cont…

Online index build logs the entire contents of the target index pages. Hence log space requirements increase significantly.

- You can specify to use TempDB to avoid log sizing and contention.
- Recommendation: specify sort_in_tempdb option for online index build and tune temp_db to avoid excessive log activity

Online index not available on a per-partition basis for partitioned tables.

ALTER INDEX … REBUILD of a corrupted non-clustered index may use another index (and that may be theoretically corrupted as well).

- Recommendation: Therefore use DBCC repair rather than alter index…rebuild

Index Operations Lessons Learned cont…

CREATE INDEX operations for Standard Edition are single threaded (were potentially multithreaded in SQL Server 2000)

DBCC CHECK* operations use database snapshot technology and tempdb

- Recommendation: Make sure you configure Tempdb in SS2005 for performance. Separate fast set of drives.

Backup / Restore

An online restore requires an exclusive lock on the database at the beginning of the restore.

- Once the first data transfer of the first restore begins, the database can be opened up to other users

Restoring offline filegroups taken at different times must occur serially, making a full piecemeal restore much longer than a parallel monolithic restore.

- Recommendation: If you have several filegroup restores you should consider full database restore it could end up being faster

Dedicated Admin Connection

High priority connection that listens on its own TCP/IP port and has a dedicated scheduler

Great to fix a problem when a perceived system hang. In SS2K this would happen when all worker threads were allocated by locks.

- Reboot SS2K
- Logon DAC (ADMIN:*<instance_name>*) in SS2005 and fix problem

Potential Concern:

- Worried about DBAs logging on DAC by default to get higher priority and dedicated thread.
- Who fixes the problem if DAC causes it?
- Recommendation: Highly discourage DBAs from doing this with normal DBA activities. It will be disastrous if they

Database Mirroring

Database Mirroring is not dropped or cut or abandoned, etc.

- Being released with SP1
- Performance looking pretty good
- Failover is fast

This HA feature is far too important not to be 100% confident with quality

Database Mirroring lessons learned: Not so transparent failover

With Safety=full (Sync), you need to connect to the mirrored database if you want automatic redirection to work, since the primary and mirror are communicated back to the client on connection where it is cached as part of the login acknowledgement

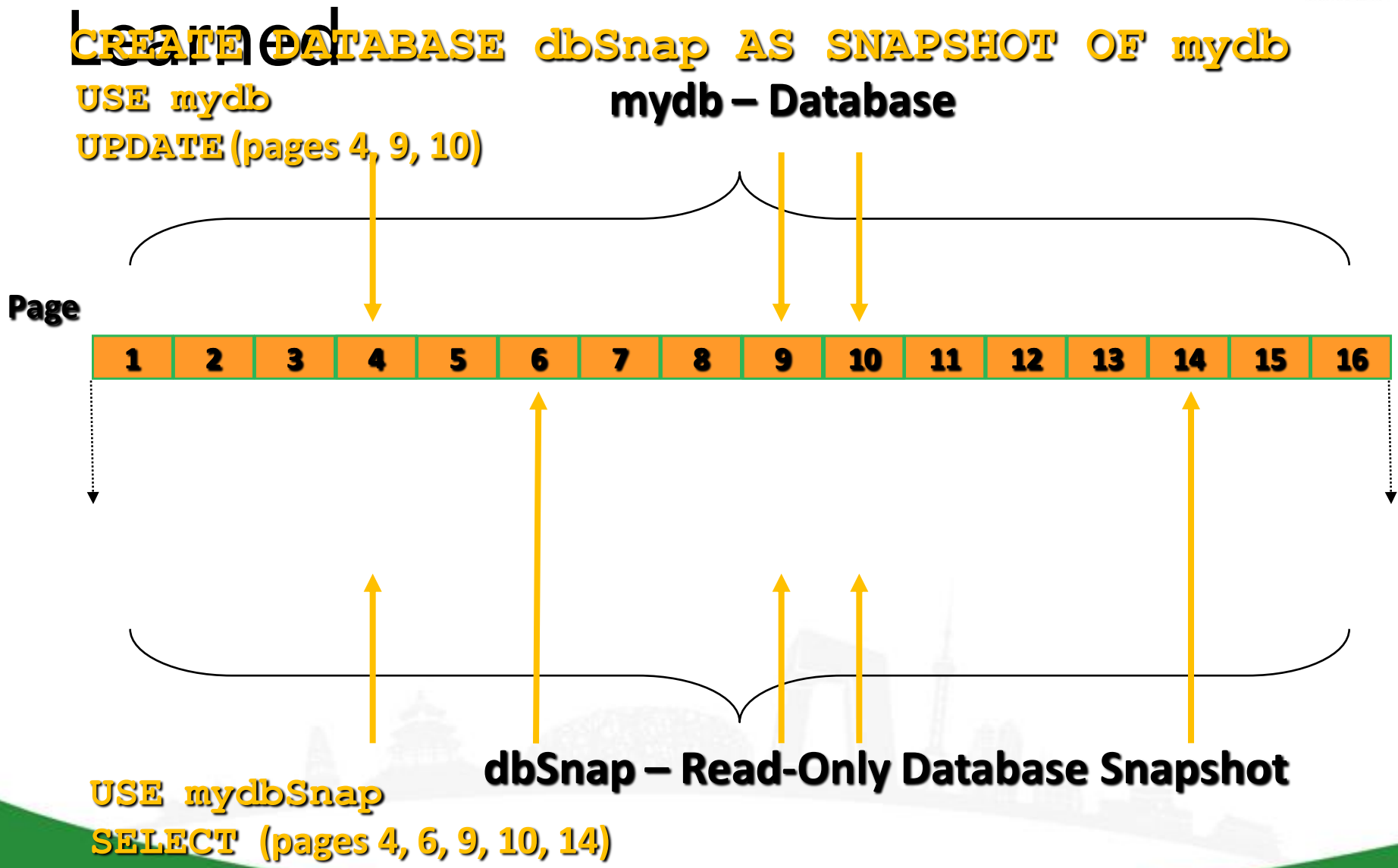
- This means your connect string needs to contain the user database, like FAILOVER_SERVER=myMirrorServerName;
- If you do not have an established connection, your primary and mirror are not cached on your client, you can not failover automatically.

With Safety =off (ASync), you need to manage the failover within your application.

Database Mirroring: Lessons Learned cont….

- If you have multiple databases linked together and they need to fail over together you will need to failover within your application
- Database Mirroring depends on the network for information about the health of the other servers. How the network sends errors back to the server affects how mirroring works.
- Failover time may not be consistent. Monitoring the REDO_QUEUE perf counter correlates directly with the expected failover time.
- Mirroring a database consumes 1 global worker thread and 2 worker threads per database on the principal. On the mirror, mirroring consumes 1 global worker thread

Database Snapshot Lessons Learned



Snapshot Isolation Lessons Learned

Example

```
CREATE TABLE t1 (c1 int unique, c2 int)
INSERT INTO t1 VALUES (1, 5)
```

Transaction 1	Transaction 2 (Snapshot Isolation)	Transaction 3 (RCSI)
BEGIN TRAN	SET TRANSACTION ISOLATION LEVEL SNAPSHOT	
UPDATE t1 SET c2 = 9 WHERE c1 =1	BEGIN TRAN	BEGIN TRAN
	SELECT c2 FROM t1 WHERE c1 = 1 -- SQL Server returns 5	SELECT c2 FROM t1 WHERE c1 = 1 -- SQL Server returns 5
COMMIT TRAN	SELECT c2 FROM t1 WHERE c1 = 1 -- SQL Server returns 5	SELECT c2 FROM t1 WHERE c1 = 1 -- SQL Server returns 9
	COMMIT TRAN	COMMIT TRAN
	SELECT c2 FROM t1 WHERE c1 = 1 -- SQL Server returns 9	SELECT c2 FROM t1 WHERE c1 = 1 -- SQL Server returns 9

Time

Snapshot Isolation lessons learned cont….

- You can only enable RCSI on database before it is set for mirroring. This restriction is not there for Snapshot Isolation.
- You cannot run DTC on Snapshot Isolation, but you can use RCSI in a DTC
- To enable a database for RCSI, there should only be one active session in the database.
- Optimized bulkload operation blocks RCSI and SI queries and vice-versa

Misc. Lessons Learned – MARS, CLR, UDTs, etc.

- MARS is not a replacement for cursors. There are some scenarios where cursors can be replaced by MARS, but it needs analysis of usage pattern to decide if it is suitable solution or not.
- UDTs are meant for small scalar types. They should not be used for modeling complex business objects (i.e. no Object Relational).
- UDAggs have a size limitation of 8K for the final instance value. Many customers wanted to do string concatenation, but this only

Misc. Lessons Learned cont…

CLR Positioning

- In SS
  - Functions, User Defined Types/Aggregates
  - Replacement for XP’ s
  - Run mid-tier data access logic in server
  - Long non-data centric TSQL code (Example: Parsing)
  - Large data results that can be manipulated and aggregated on the server vs mid-tier
- In Middle tier
  - Scale out middle tier
  - Database diagnostic
  - Advanced logic but simple data manipulation with small results sets

Security Lessons Learned

Microsoft tech·ed China 2008

Catalog security – you can no longer see all system tables

Security features – they’ re much more granular now, more flexible, but there are lots of concepts to learn.

- master keys, db keys, certs, roles, fine grained permissions, run as

Security Lessons Learned cont.

Encryption – cannot index an encrypted column

- Example: An encrypted Social Security Number cannot be indexed because encryption of a value doesn't always yield the same guid.
- WORKAROUND: is to create a hash of the value, based on encryption key, and store in a separate column. This can be indexed and searched by hash of a query predicate, but slightly increases risk.

Surface Area Configuration - SAC

Many components off by default

Recommendation: Absolutely run configuration manager after installation and/or upgrade

A couple of surprises

- Components
  - AS, FTS, RS, SSIS, NS
- Features
  - DQ, CLR, DAC, DB Mail, XML Web Services, OLE Automation SPs, Service Broker, XP
- Networking
  - Server: Named Pipes, etc.
  - SNAC: TCP/IP, Named Pipes, etc.

Threats and vulnerabilities

100% of projects are still vulnerable

- SQL Injection – adding a valid sql command at the end of an input value on a screen
- XSS – Cross Side script – replacing a screen value with a script that can run on the client side after next retrieval

//team/sites/aceteam

Alias: AceSEC

These features from earlier versions are not supported in SQL Server 2005.

Category	Discontinued feature	Replacement
Command prompt utilities	**isql** utility	Use sqlcmd
Configuration options	**'allow updates'** option of **sp_configure**.	Option is present but direct updates to system tables are not supported
Configuration options	**'open objects'** option of **sp_configure**.	The 'open objects' option has been left in **sp_configure** to ensure backward compatibility with existing scripts.
Configuration options	'set working set size' option of **sp_configure**	Option is present but its functionality has been deactivated.
Database creation	DISK INIT DISK RESIZE	Legacy behavior from SQL Server 6.x
Database creation	FOR LOAD option of CREATE DATABASE	RESTORE operations can create a database
DBCC	DBCC DBREPAIR	Use DROP DATABASE to remove a damaged database.

These features from earlier versions are not supported in SQL Server 2005.

Category	Discontinued feature	Replacement
DBCC	DBCC NEWALLOC	DBCC CHECKALLOC
DBCC	DBCC PINTABLE, DBCC UNPINTABLE	None.
DBCC	DBCC ROWLOCK	Row-level locking is automatic.
DBCC	DBCC TEXTALL DBCC TEXTALLOC	DBCC CHECKDB DBCC CHECKTABLE
Extended store procedure programming	Use of SRV_PWD field in the SRV_PFIELD structure when there has been an impersonation context switch from the original login.	None.
Network protocols	The following protocols: NWLink IPX/SPX, AppleTalk, Banyan Vines, Multiprotocol.	TCP/IP sockets, named pipes, VIA, or shared memory.
Rebuild master	Rebuildm.exe	Use Setup.exe
Sample databases	**Northwind** and **pubs**	Use **AdventureWorks**; however, **Northwind** and **pubs** are available as downloads, or can be copied from a previous installation of SQL Server.
Transact-SQL	***=** and **=*** outer join operators	Use the JOIN syntax of the FROM clause.
Virtual tables	**syslocks**	**sys.dm_tran_locks**

疑问和解答

感谢您参与此会场！

您的意见与建议对我们非常重要。

请您填写反馈表。

Microsoft®

您的潜力，我们的动力

Microsoft | Virtualization | Windows HPC Server 2008 | SQL Server 2008 | Visual Studio 2008