

mow

M

M

X

Miracle OpenWorld 2010

mow

M

M

X

Miracle OpenWorld 2010

# SQL Server Storage Engine under the hood: How SQL Server performs IO

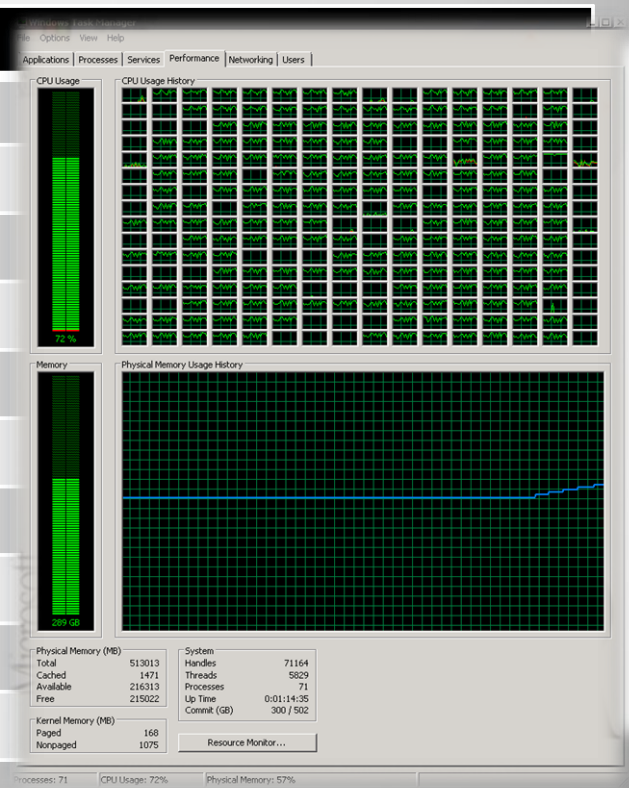
Thomas Grohser =tg=, bwin  
thomas.grohser@bwin.org

tg@grohser.com



# select \* from =tg=

@@Version	Remark
SQL 4.21	First SQL Server ever used (1994)
SQL 6.0	First Log Shipping with failover
SQL 6.5	First SQL Server Cluster (NT4.0 + Wolfpack)
SQL 7.0	2+ billion rows / month in a single Table
SQL 2000	938 days with 100% availability
SQL 2000 IA64	First SQL Server on Itanium IA64
SQL 2005 IA64	First OLTP long distance database mirroring
SQL 2008 IA64	First Replication into mirrored databases
SQL 2008R2 IA64	First 256 CPUs & >500.000 STMT/sec
SQL 11 (Denali)	Can't wait to push the limits even further



## Focus on SQL Server Infrastructure Architecture and Implementation Close Relationship with Microsoft

SQLCAT (SQL Server Customer Advisory Team)

SCAN (SQL Server Customer Advisory Network)

TAP (Technology Adoption Program SQL2008R2 and SQL11)

Close relationship with Hardware Vendors (Focus IA64)



M M X

**Active PASS member  
and PASS Summit Speaker  
SQL Server User Group Austria**



Miracle OpenWorld 2010



**World's biggest publicly listed online gaming platform**

World's **leading provider** of online Sports Betting

One of the largest **Poker networks**

Comprehensive range of **Payment Service Providing**

Integrated gaming portal - **22 languages, 25 core markets**

**Gross gaming revenues 2008 (GGR):**  
EUR **421 million**

More than **20 million registered customers**

**1,500 employees**

bwin builds on the strengths of the web in order to **tie up responsibility and gaming**

**15 million page views** and up to **980,000 users a day**



# Copyright and Thank you notice

Some of the slides and content were originally created and/or the data was collected by

- Thomas Kejser, SQLCAT, Microsoft
- Jürgen Thomas, SQLCAT, Microsoft
- Günter Zink, Performance Engineering, HP

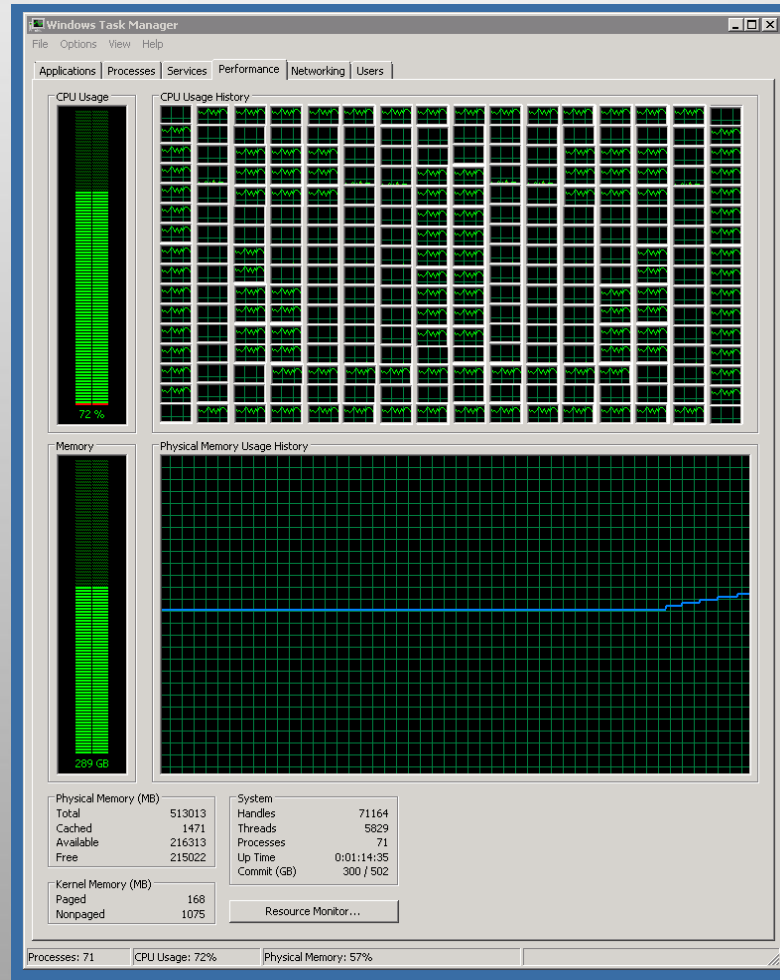
Thanks to all of them for doing such a great job in discovering, collecting and documenting SQL Server “Know How”



# My favorite new SQL2008 R2 Feature



# CREATE INDEX WITH SMILE



M M X

# Agenda

- I/O Basics
- What is an I/O sub system?
- Understanding SQL Server I/O Patterns
- Storage types
- Wrap Up
- Q&A



ATTENTION:  
Important  
Information may be  
displayed at any  
slide at any time!



# I/O Basics

- Random / Sequential
- How data is organized in SQL Server
- When does IO occur during transactions
- Available IO System Technologies

# Random / Sequential

- Sequential:
  - Data is read from the IO subsystem in the same order as it is stored on the IO subsystem.
- Random:
  - Data is read from the IO subsystem in a different order as it is stored on the IO subsystem.

Careful: On bad configured systems sequential IO on non RAID 1 may become random IO



# How data is organized

- SQL Server is an in memory database
- All operations are performed in memory
- The format on disk and in memory is the same !
  
- 8KB pages = 8192 bytes
  - 96 bytes of page header
  - 8096 bytes of data(usage depends of page type [data, index, PFS, ...])
  
- 8 contiguous pages are combined to a segment
  - ➔ Segment size is 64KB
  
- The query processor just requests the page from the buffer pool the buffer pool knows if its in memory or if it has to pull in from disk



# Disk File structure

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Select statement

```
SELECT * FROM T WHERE ID = 5
```

Query processor chooses to do an index seek

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Buffer pool

SELECT \* FROM T WHERE ID = 5

Page Nr										
Page type										
Data										

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Read root page buffer pool

SELECT \* FROM T WHERE ID = 5

Page Nr		8								
Page type	Root									
Data	1:9 12:10									

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Read index page into buffer pool

SELECT \* FROM T WHERE ID = 5

Page Nr	8	9								
Page type	Root	Index								
Data	1:9 12:10	1:11 5:12 8:13 10:14								

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Data page into buffer pool

SELECT \* FROM T WHERE ID = 5

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Freak							

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Transaction

- UPDATE T SET Name = 'Frank' WHERE ID = 6

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Freak							

<b>LOG</b>										
------------	--	--	--	--	--	--	--	--	--	--

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Update Data in memory

- UPDATE T SET Name = 'Frank' WHERE ID = 6

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Frank							

<b>LOG</b>										
------------	--	--	--	--	--	--	--	--	--	--

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Write to Transaction Log

- UPDATE T SET Name = 'Frank' WHERE ID = 6

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Frank							

<b>LOG</b>	<b>ID 6;Old Freak;New Frank</b>											
------------	---------------------------------	--	--	--	--	--	--	--	--	--	--	--

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Checkpoint, data written to disk

- CHECKPOINT

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Frank							

<b>LOG</b>	<b>ID 6;Old Freak;New Frank</b>											
------------	---------------------------------	--	--	--	--	--	--	--	--	--	--	--

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Freak	8: Hans	10: Pete	12: Rick 14: Sam		

# Checkpoint persisted in Log

- CHECKPOINT

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Frank							

<b>LOG</b>	ID 6;Old Freak;New Frank									
------------	--------------------------	--	--	--	--	--	--	--	--	--

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Frank	8: Hans	10: Pete	12: Rick 14: Sam		

# Transaction Log Backup

- BACKUP LOG ...

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Frank							

<b>LOG</b>	ID 6;Old Freak;New Frank									
------------	--------------------------	--	--	--	--	--	--	--	--	--

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Frank	8: Hans	10: Pete	12: Rick 14: Sam		

# Transaction log is cleared

after the LOG Backup is completed

Page Nr	8	9	12							
Page type	Root	Index	Data							
Data	1:9 12:10	1:11 5:12 8:13 10:14	5: Eric 6: Frank							

<b>LOG</b>										
------------	--	--	--	--	--	--	--	--	--	--

Page Nr	...	8	9	10	11	12	13	14	15	16	...
Page type		Root	Index	Index	Data	Data	Data	Data	Data	Empty	
Data		1:9 12:10	1:11 5:12 8:13 10:14	12:15	1: Andy 2: Bobo	5: Eric 6: Frank	8: Hans	10: Pete	12: Rick 14: Sam		

# Available IO System Technologies

- SAN      Storage Area Network
- DAS      Direct Attached Storage
  - Typically SCSI disks where the controller is in the server and the disk in an external chassis
- SSD      Solid State Device
  - The disk **is** the controller
  - Do not confuse with Solid State Disk that can be used in DAS or SAN environments



# What is an I/O Subsystem?



# Measuring I/O

An I/O subsystem has three characteristics

## 1. Capacity

- Measured in GB/TB



The easy one!

## 2. Throughput

- Measured in MB/sec or IOPs
- Performance Monitor: Logical Disk
  - Disk Read Bytes / Sec
  - Disk Write Bytes / Sec
  - Disk Read / Sec
  - Disk Writes / Sec

## 3. Latency

- Measured in milliseconds (ms)
- Performance Monitor: Logical Disk
  - Avg. Disk Sec / read
  - Avg. Disk Sec / write
- Consistent high values (>15ms) indicate I/O bottleneck



# Specifying I/O

- When building an I/O system, it is important to understand both limitations and growth options
  - Maximum Throughput
    - Depends on:
      - Block sizes requested
      - Sequential vs. Random
      - Read or write pattern
  - Minimum latency
    - The lower limit for latency
    - Important for real time systems
  - Capacity
- Understand how to grow the three factors
- It takes a **LOT** of I/O to saturate SQL Server
  - DW workload typically more I/O intensive than OLTP
  - Not unusual to see machine dedicated SAN for large DW installations



# Basic requirements of storage

- **Stable**

- Storage that must survive system restart or common failure

- Power loss
- Drive failure
- Controller failure

- **Write Ordering**

- Preserving order of I/O operations



# RAID levels

- Both a stability and a performance technology
- Characteristics depend on RAID level
  - Abilities depend on storage vendor implementation
  - RAID 1/0 is typically the fastest option
- The more cache the controllers/arrays have the less the RAID level matters.
  - Except while in degraded state!
    - RAID 5 and RAID 6 much slower when degraded
- **Best Practice:** Benchmark and compare the different RAID levels before deploying



# SQL Server I/O Characteristics



# Understanding I/O patterns

- The I/O pattern generated by SQL Server varies by the workload
- Relational Database Workloads
  - Log Writes
  - Checkpoint / Lazy Writer
  - Index Seeks
  - Table / Range Scans
  - Bulk Load



# Log Writes - Workload Description

- Threads fill log buffers & requests log manager to flush all records up to certain LSN
  - log manager thread writes the buffers to disk
- Log manager throughput considerations
  - SQL Server 2000 SP4 & SQL Server 2005 RTM
    - Limit log writes to 8 outstanding (per database)
  - SQL Server 2005 SP1 or later
    - Limit of 8 (32-bit) or 32 (64-bit) outstanding log writes
    - No more than 480K “in-flight” for either
  - SQL Server 2008 increases “in-flight” per log to 3840K (factor of 8)



# Log Writes - Pattern / Monitoring

- Sequential I/O
- Write size varies
  - Dependent on nature of transaction
  - Transaction “Commit” forces log buffer to be flushed to disk
  - Up to 60KB
- SQL Server Wait Stats
  - WRITELOG
  - LOGBUFFER
- Performance Monitor:
  - MSSQL: Databases
    - Log Bytes Flushed/sec
    - Log Flushes/sec
    - Avg. Bytes per Flush = (Log Bytes Flushed/sec) / (Log Flushes/sec)
    - Wait per Flush = (Log Flush Wait Time) / (Log Flushes / sec)



# Checkpoint / Lazy Writer

- Workload Description / Types of Checkpoints
  - Background/automatic checkpoints: Triggered by log volume or recovery interval and performed by the checkpoint thread
  - User-initiated checkpoints: Initiated by the T-SQL CHECKPOINT command.
  - Reflexive checkpoints: Automatically performed as part of some larger operation, such as recovery, restore, snapshot creation, etc.



# Checkpoint / Lazy Writer

- Pattern / Monitoring
  - Random, but SQL Server will attempt to find adjacent pages
  - Up to 256KB in a single I/O request
  - Performance Monitor
    - MSSQL:Buffer Manager
      - Checkpoint pages / sec
      - Lazy Writes / sec

# Checkpoint (continued)

- Checkpoint Throttling
  - Checkpoint measures I/O latency impact and automatically adjusts checkpoint I/O to keep the overall latency from being unduly affected
  - CHECKPOINT [checkpoint\_duration]
    - CHECKPOINT now allows an optional numeric argument, which specifies the number of seconds the checkpoint should take
    - Checkpoint makes a “best effort” to perform in the time specified
    - If specified time is too low it runs at full speed
- NUMA systems spread the checkpoints to lazy writers on each node



# Index Seeks - Workload Description

- Query plans performing loop joins will typically do many index seeks
- Single row lookups in index
- Traverse the B-Tree of the index, retrieve single page / row
- OLTP workloads typically heavy on these
- SQL Server may perform read-ahead
  - Single page request bring in entire 8-page (64KB) extent
  - Helps server come up to speed quicker



# Index Seeks - Pattern / Monitoring

- Random I/O
- 8 KB Block Sizes
  - 64KB when doing read ahead and during startup
- SQL Server Wait Stats
  - PAGEIOLATCH\_<X>
- **dm\_db\_index\_usage\_stats**
  - user\_seeks
  - user\_lookups
- Performance Monitor:
  - MSSQL:Access Methods
    - Index Seeks / Sec
  - MSSQL:Buffer Manager
    - Readahead Pages / sec



# Table / Range Scan - Workload Description

- Query plans doing hash and merge joining
- Aggregation Queries
- Typical for DW workloads
- SQL Server may perform read-ahead
  - Dynamically adjust read-ahead size by table
  - Standard Edition: Up to 128 pages
  - Enterprise Edition: Up to 512 pages



# Table / Range Scan - Pattern / Monitoring

- Sequential I/O
  - 64-512KB Block Sizes
- SQL Server Wait Stats
  - PAGEIOLATCH\_<X>
- dm\_db\_index\_usage\_stats
  - user\_scans
- Performance Monitor:
  - MSSQL:Access Methods
    - Range Scans / Sec
    - Table Scans / Sec
  - MSSQL:Buffer Manager
    - Readahead Pages / sec



# Bulk Load - Workload Description

- Occurs when a bulk load operation is performed
- Typical for DW workloads
- I/O Depends on Data recovery mode
  - SIMPLE / BULK LOGGED mode writes to database
  - FULL writes to transaction log and flush to database

# Bulk Load - Pattern / Monitoring

- Sequential I/O
- 64KB-256 KB
  - Block sizes depend on database file layout
- SQL Server Wait Stats
  - WRITELOG / LOGBUFFER
  - PAGEIOLATCH\_EX
  - PAGELATCH\_UP
    - PFS Contention, not I/O related

# SQL Server I/O Characteristics- Summary

Workload	Type	Block Size
Log writes	Sequential	Up to 60KB
Checkpoint / Lazy Write	Random	Up to 256KB
Index Seeks	Random	8KB 64KB (read Ahead)
Table / Range Scan	Sequential	64KB-512KB
Bulk Load	Sequential	65KB-256KB
Backup Operations	Sequential	64KB-4MB

# Typical I/O Workloads

- OLTP (Online Transaction Processing)
  - Typically, heavy on 8KB random read / writes
  - Some amount of read-ahead
    - Size varies – multiples of 8K (see read-ahead slide)
    - Many “mixed” workloads observed in customer deployments
  - **Rule of Thumb:** Optimize for Random I/O
- RDW (Relational Data Warehousing)
  - Typical 64-256KB sequential reads (table and range scan)
  - 128-256KB sequential writes (bulk load)
  - **Rule of Thumb:** Optimize for Sequential I/O

# SQL Server View of I/O

Tool	Monitors	Granularity
<b>sys.dm_io_virtual_file_stats</b>	Latency, Number of IO's	Database files
<b>sys.dm_exec_query_stats</b>	Number of ... Reads (Logical Physical) Number of writes	Query or Batch
<b>sys.dm_db_index_usage_stats</b>	Number of IO's and type of access (seek, scan, lookup, write)	Index or Table
<b>sys.dm_db_index_operational_stats</b>	IO latch wait time, Page splits	Index or Table
<b>sys.dm_os_wait_stats</b>	PAGEIOLATCH waits	SQL Server Instance level (cumulative since last start – most useful to analyze over time periods).
<b>Xevents</b>	PAGEIOLATCH waits	Query and Database file

# Windows View of I/O

Counter	Description
<b>Disk Reads/Writes per Second</b>	Measures the Number of I/O's per second Discuss with vendor sizing of spindles of different type and rotational speeds Impacted by disk head movement (i.e. short stroking the disk will provide more I/O per second capacity)
<b>Average Disk/sec Read &amp; Write</b>	Measures disk latency. Numbers will vary, optimal values for averages over time: 1 - 5 ms for Log (Ideally 1ms or better) 5 - 20 ms for Data (OLTP) (Ideally 10ms or better) <=25-30 ms for Data (DSS)
<b>Average Disk Bytes/Read &amp; Write</b>	Measures the size of I/O's being issued. Larger I/O tend to have higher latency (example: BACKUP/RESTORE)
<b>Current Disk Queue Length</b>	Hard to interpret due to virtualization of storage. Not of much use!
<b>Disk Read &amp; Write Bytes/sec</b>	Measure of total disk throughput. Ideally larger block scans should be able to heavily utilize connection bandwidth.

# Storage Types



# Storage Selection - General

- Understanding the I/O characteristics and availability requirements is key
- Engage the engineers from all sides, early on
- Number of spindles matter
  - More spindles typically yield better speed
    - True for both SAN and DAS
    - New Game on SSD
- There is no one single “right” way to configure storage for SQL Server
  - Physical isolation practices become more important at the high end
- **Best Practice:** Validate and compare configurations prior to deployment



# Storage Selection-Common Pitfalls

- There are barriers between DBA's and storage administrators
  - Each needs to understand the others “world”
- Sizing only on “capacity” is a common problem
  - Must take latency and throughput into consideration
- One size fits all type configurations
  - Storage vendor should have knowledge of SQL Server and Windows best practices when array is configured
    - Especially when advanced features are used (snapshots, replication, etc...)
- SAN is complex
  - Generally involves multiple organizations to put a configuration in place

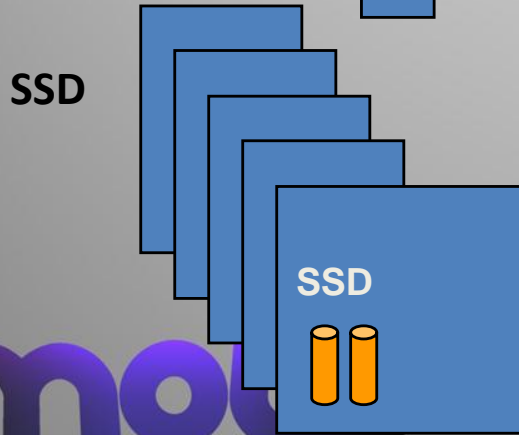
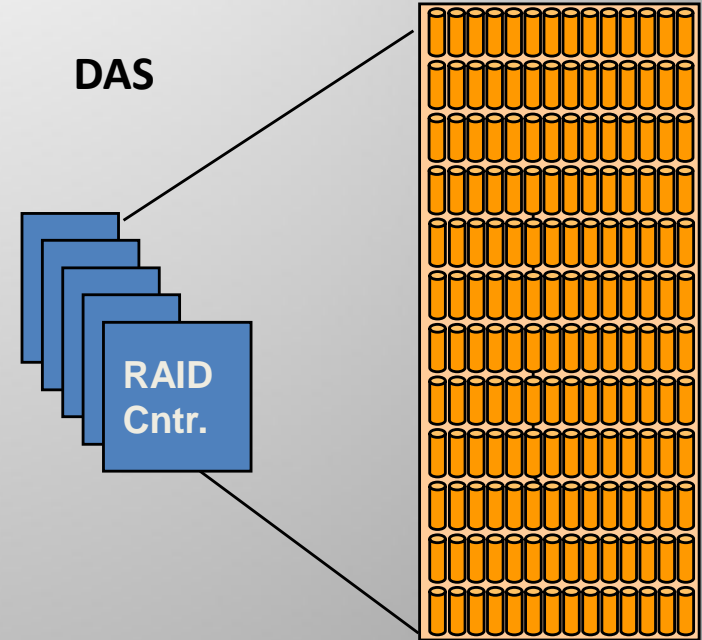
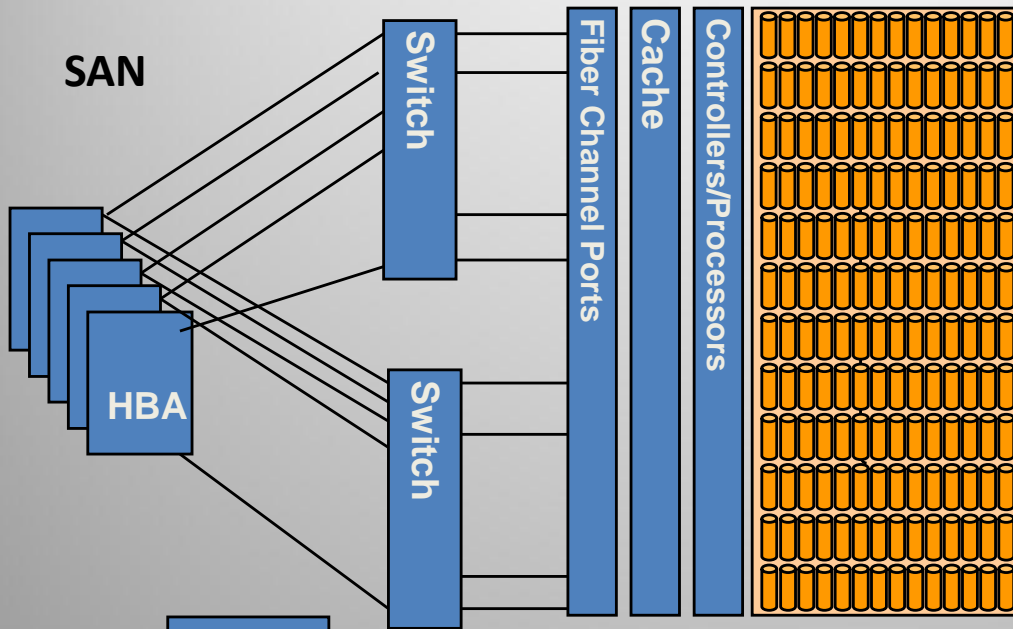


# SAN vs. DAS vs. SSD

Feature	SAN	DAS	SSD
Cost	High - but may be offset by better utilization	Low - But may waste space	Very High
Flexibility	Virtualization allows online configuration changes	Better get it right the first time	
Skills required	Steep learning curve	Simple and well understood	New Technology
Additional Features	Snapshots Storage Replication	None	None
Performance	Contrary to popular belief, SAN is not a performance technology	High performance for the investment	Very High
Reliability	Very high reliability	Typically less reliable. - May be offset by higher redundancy on RAID levels	Wear down time is relatively short
Clustering Support	Yes	No	No



# Understand the path to the drives



# SQL Server on SAN – Common Pitfalls

- **Non-disk related bottlenecks**
  - Many shared components (fiber ports / switches, array cache, service processors, etc...) in SAN
  - Disks may not be the bottleneck – understand the full path to the drives
- **Physical design matters!**
  - Think about splitting workloads with very different I/O configurations at the physical (disk) level
  - Provides predictable performance
  - Sharing disks between servers may be a very *bad* idea
  - This is true for both DAS and SAN
- **SAN Cache does *not* solve all performance problems**
  - Limited benefit for random read and for read-ahead operations within SQL Server
  - Best to tune for writes; supporting low log latency and absorbing checkpoint operations
  - Cache trashing may occur, especially with heavy DW workload.
- **Configuration Issues**
  - Queue depth set too low, multi-pathing improperly configured
  - Get the right drivers
  - HBA placement: Avoid overloading single PCI bus with HBA traffic



# SQL Server on DAS - Common Pitfalls

- Beware of non disk related bottlenecks
  - SCSI controller may not have bandwidth to support disks
  - PCI-X bus should be fast enough
  - Example: Need PCI-X 8x to consume 1GB/sec
- Can use Dynamic Disks to stripe LUN's together
  - Bigger, more manageable partitions
- Cannot grow storage dynamically
  - Buy enough
  - ... or plan database layout to support growth
- Inexpensive way to create very high performance I/O system
- No SAN = No Cluster!

# Solid State Devices

- Emerging technology
- Storage device based on DRAM and NAND flash (SLC)
- Fits into PCI slot
  - Drivers required
- Advantages
  - Performance, weight, power consumption, more durable
  - Random = Sequential !
- Disadvantages
  - Cost per GB
  - Limited experience for enterprise use
- Most appealing for
  - IOPs intensive “Tier 0” storage (LOG files)
  - Mobile devices

# Solid State Disks

- Emerging technology
- Storage device based on NAND flash (MLC & SLC)
- Fits into regular HDD slot
  - Utilizes the same command set and interface
  - Can be used both in SAN and DAS
- Advantages
  - Performance, weight, power consumption, more durable
  - Random = Sequential !
- Disadvantages
  - Controller is the limit
  - Cost per GB, shifting bottleneck
  - Writes are expensive relative to reads
- Most appealing for
  - IOPs intensive “Tier 0” storage RANDOM READS
  - Mobile devices



# Tools

- SQLStress
  - SQLStress is the tool to stress test a Microsoft SQL Server installation. It can also be used for hardware sizing, system tuning, benchmarking or verifying "High Availability Features" like clustering and database mirroring.
  - <http://www.sqlstress.com/>
- Diskpar.exe
  - Windows 2000 Companion CD
  - [http://www.microsoft.com/windows2000/techinfo/reskit/en-us/default.asp?url=/windows2000/techinfo/reskit/en-us/prork/pree\\_exa\\_oori.asp](http://www.microsoft.com/windows2000/techinfo/reskit/en-us/default.asp?url=/windows2000/techinfo/reskit/en-us/prork/pree_exa_oori.asp)
- SQLIO
  - Used to stress an I/O subsystem – Test a configuration's performance
  - <http://www.microsoft.com/downloads/details.aspx?FamilyId=9A8B005B-84E4-4F24-8D65-CB53442D9E19&displaylang=en>
- SQLIOSim
  - Simulates SQL Server I/O – Used to isolate hardware issues
  - 231619 HOW TO: Use the SQLIOStress Utility to Stress a Disk Subsystem  
<http://support.microsoft.com/?id=231619>
- Fiber Channel Information Tool
  - Command line tool which provides configuration information (Host/HBA)
  - <http://www.microsoft.com/downloads/details.aspx?FamilyID=73d7b879-55b2-4629-8734-b0698096d3b1&displaylang=en>



# KB Articles

- KB 824190 Troubleshooting Storage Area Network (SAN) Issues
  - <http://support.microsoft.com/?id=824190>
- KB 304415: Support for Multiple Clusters Attached to the Same SAN Device
  - <http://support.microsoft.com/?id=304415>
- KB 280297: How to Configure Volume Mount Points on a Clustered Server
  - <http://support.microsoft.com/?id=280297>
- KB 819546: SQL Server support for mounted volumes
  - <http://support.microsoft.com/?id=819546>
- KB 304736: How to Extend the Partition of a Cluster Shared Disk
  - <http://support.microsoft.com/?id=304736>
- KB 325590: How to Use Diskpart.exe to Extend a Data Volume
  - <http://support.microsoft.com/?id=325590>
- KB 328551: Concurrency enhancements for the tempdb database
  - <http://support.microsoft.com/?id=328551>
- KB 304261: Support for Network Database Files
  - <http://support.microsoft.com/?id=304261>



# General Storage References

- Microsoft Windows Clustering: Storage Area Networks
  - <http://www.microsoft.com/windowsserver2003/techinfo/overview/san.mspx>
- StorPort in Windows Server 2003: Improving Manageability and Performance in Hardware RAID and Storage Area Networks
  - <http://www.microsoft.com/windowsserversystem/wss2003/techinfo/plandeploy/storportwp.mspx>
- Virtual Device Interface Specification
  - <http://www.microsoft.com/downloads/details.aspx?FamilyID=416f8a51-65a3-4e8e-a4c8-adfe15e850fc&DisplayLang=en>
- Windows Server System Storage Home
  - <http://www.microsoft.com/windowsserversystem/storage/default.mspx>
- Microsoft Storage Technologies – Multipath I/O
  - <http://www.microsoft.com/windowsserversystem/storage/technologies/mpio/default.mspx>
- Storage Top 10 Best Practices
  - <http://sqlcat.com/top10lists/archive/2007/11/21/storage-top-10-best-practices.aspx>



# SQL Server Storage References

- SQL Server Consolidation on the 64-Bit Platform
  - <http://www.microsoft.com/technet/prodtechnol/sql/2000/deploy/64bitconsolidation.mspx>
- SQL Server Consolidation on the 32-Bit Platform using a Clustered Environment
  - <http://www.microsoft.com/technet/prodtechnol/sql/2000/deploy/32bitconsolidation.mspx>
- SQL Server 2000/2005 I/O Basics on TechNet
  - <http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/sqlIObasics.mspx>
  - <http://www.microsoft.com/technet/prodtechnol/sql/2005/iobasics.mspx>
- Microsoft SQL Server I/O subsystem requirements for the tempdb database
  - <http://support.microsoft.com/kb/917047>
- SQL Server AlwaysOn Partner program
  - <http://www.microsoft.com/sql/alwayson/default.mspx>
- SQL Server PreDeployment Best Practices
  - <http://www.microsoft.com/technet/prodtechnol/sql/bestpractice/pdplio bp.mspx>
- Scalability and VLDB Resources on Microsoft.com
  - <http://www.microsoft.com/sql/techinfo/administration/2000/scalability.asp>



# Case Studies

- ICE Reference case
  - <http://msevents.microsoft.com/CUI/EventDetail.aspx?EventID=1032341825&Culture=en-US>
- SQL Server Case Studies
  - <http://www.microsoft.com/sql/casestudies/default.mspx>



# References – Vendor Specific

- Microsoft SQL Server OLTP Applications for the Adaptable Modular Storage 1000
  - [http://www.hds.com/related-information/best-practices-for-microsoft-sql-server-oltp-applications.html?WT.ac=hp\\_sp2\\_r0\\_oltp\\_072508](http://www.hds.com/related-information/best-practices-for-microsoft-sql-server-oltp-applications.html?WT.ac=hp_sp2_r0_oltp_072508)
- Hitachi Adaptable Modular Storage 2000 Family Best Practices with Microsoft® SQL Server
  - <http://www.hds.com/assets/pdf/hitachi-adaptable-modular-storage-2000-family-bp-guide-for-sql-server.pdf>
- Hitachi Dynamic Provisioning Software Best Practices Guide - Guidelines for the Use of Hitachi Dynamic Provisioning Software with the Microsoft® Windows Operating System and Microsoft SQL Server 2005, Microsoft Exchange 2003 and Microsoft Exchange 2007
  - <http://www.hds.com/assets/pdf/hitachi-dynamic-provisioning-software-best-practices-guide-microsoft.pdf>
- Tuning Microsoft SQL Server 2005 Performance
  - <http://www.hds.com/assets/pdf/tuning-microsoft-sql-server-2005-performance-wp.pdf>
- Hitachi HiCommand® QoS for Microsoft SQL Server
  - [http://www.hds.com/assets/pdf/br\\_hicommand\\_qos\\_ms\\_sql\\_server\\_568.pdf](http://www.hds.com/assets/pdf/br_hicommand_qos_ms_sql_server_568.pdf)
- Microsoft® SQL Server “Always On” Technologies Hitachi Data Systems Contributes “Always On” Storage Solutions
  - <http://www.hds.com/assets/pdf/microsoft-sql-server-always-on-technologies.pdf>
- Solutions : Microsoft SQL Server
  - <http://www.hds.com/solutions/applications/microsoft/ms-sql-server.html>



Thank You

Questions?



M M X