Scaling SQL Server Applications: Application Design and Hardware Considerations

Tarek Bohsali
Microsoft

Microsoft

SESSION SUMMARY

- SQL Server is a proven platform for OLTP workloads
- SQL Server 2008 R2 offers features to assist with OLTP scalability
- How to design hardware and software for scalability

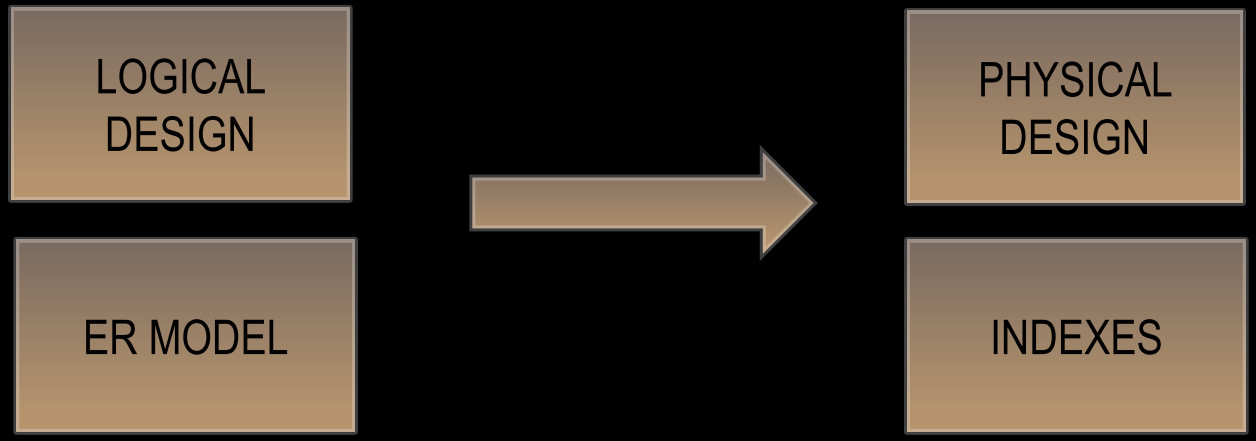
AGENDA

- OLTP workload characteristics
- OLTP application design principles
- Scalability determinants and bottlenecks
- SQL Server 2008 R2 Performance and Scale features
- Demo
- Scaling Up – Hardware to the rescue
- Summary

OLTP WORKLOAD CHARACTERISTICS

- Typically used by line-of-business (LOB) applications
- Has both read-write
- Fine-grained inserts and updates
- High transaction throughput e.g., 10s K/sec
- Usually very short transactions e.g., 1–3 tables
- Sometimes multi-step e.g., financial
- Relatively small data sizes

APPLICATION DESIGN PRINCIPLES



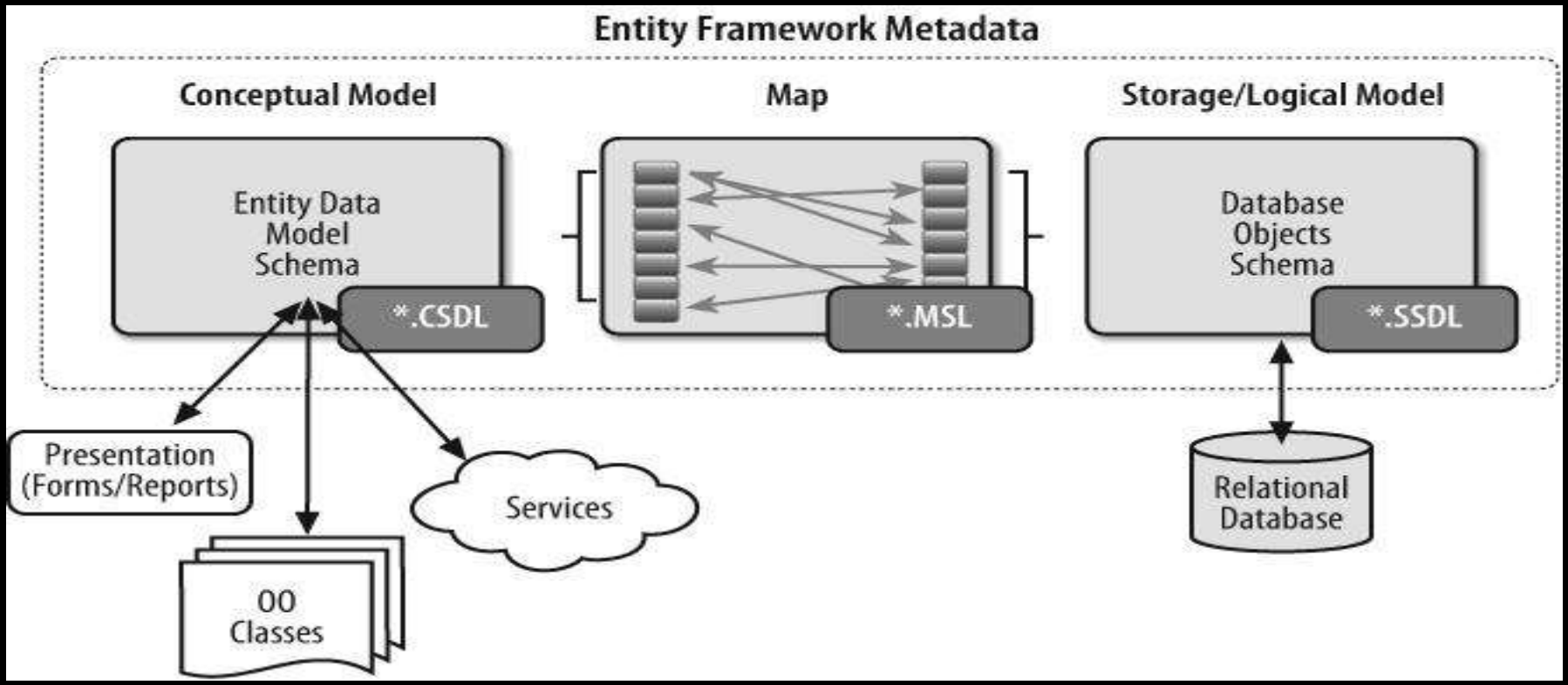
Design to leverage set-oriented processing power of SQL Server
Use development tools Visual Studio for Entity Framework design and DTA for tuning indexes

ENTITY FRAMEWORK 4.0

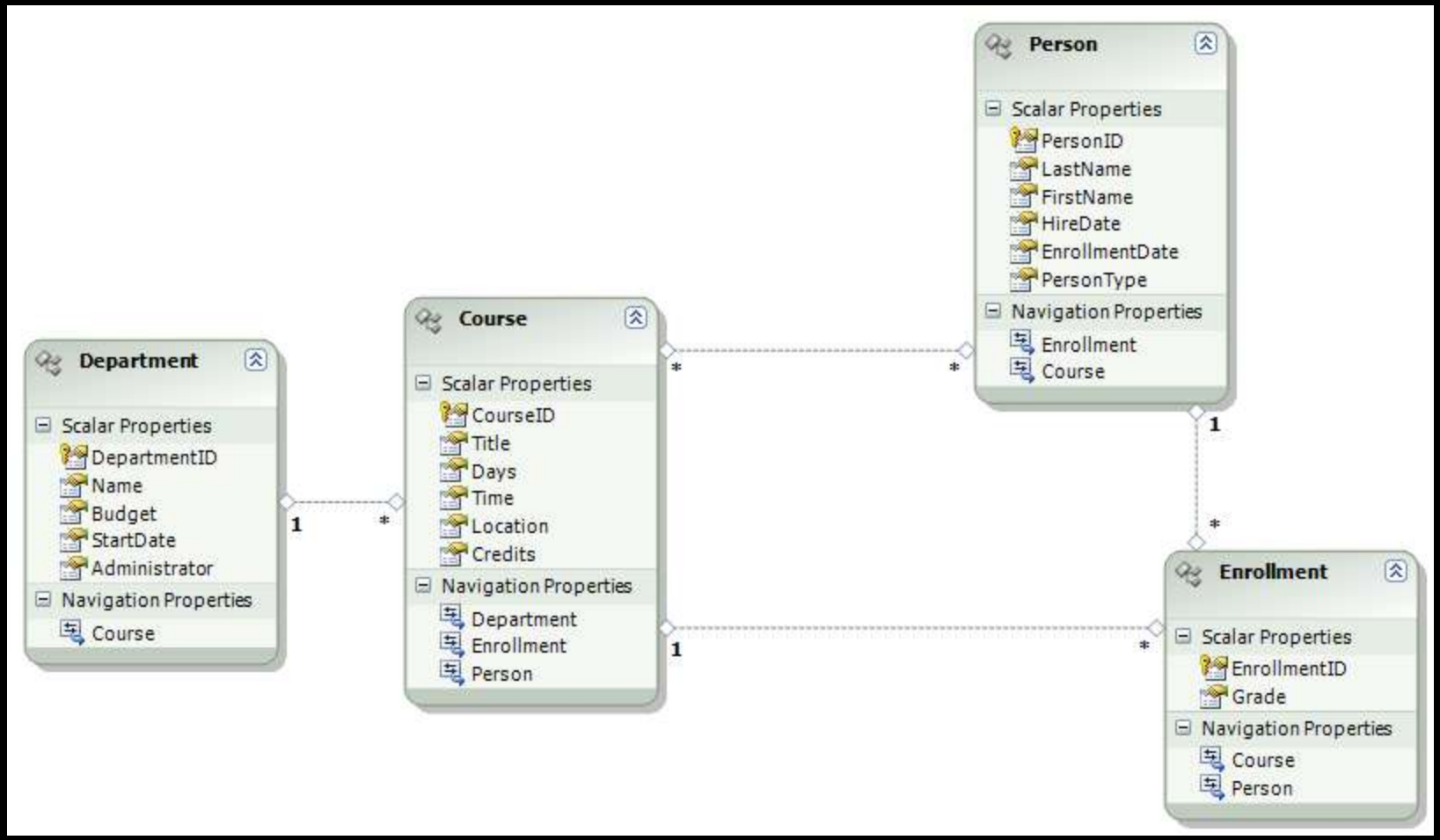
- **Development Approaches**
  - **Model First development** – Start from a Model and then have T-SQL and customized code generated.
  - **Testing**– New interface and guidance for building test suites faster.
- **Architectural Advantages**
  - **Persistence Ignorance** – Use your own classes without needing to introduce interfaces or other elements
  - **Applications Patterns** – Discussing patterns like the Repository and UnitOfWork patterns with guidance on how to use them with the Entity Framework
  - **Building N-Tier applications** – Adding API's and templates that make building N-Tier applications much easier

EXPLORING THE MODEL

- The Three Parts of the Model:



REVERSE ENGINEER DATABASE



APPLICATION DESIGN BEST PRACTICES

- Ensure good logical (E-R Model) and physical (indexes) DB design
- Leverage set-oriented processing power of SQL Server
- Update Statistics – ensure it is up to date!
- Use DTA to assist with physical design
- Avoid too many joins
- Now let's talk Physical Design

PHYSICAL DESIGN BEST PRACTICES

- Reasons for Physical Design changes
  - Performance
  - Availability
  - Security
  - Auditing
- Separate logs and data if possible
- Spend time doing index analysis
- Tune OLTP systems for high I/O per second
- Tune data warehouse for high throughput per second

CLUSTERED INDEX GUIDELINES

- Good when queries select large number of adjacent rows (range queries)
  - Create on the frequently used columns (in JOINs and WHERE with “=“, “<“, “>“, “BETWEEN”)
  - If number of returned rows is small – non-clustered index may be as efficient
  - Preferred on narrow and highly selective columns
- Remember cost of maintenance:
  - Updates reorganize the table
    - Performance impact
    - Causes index fragmentation over time

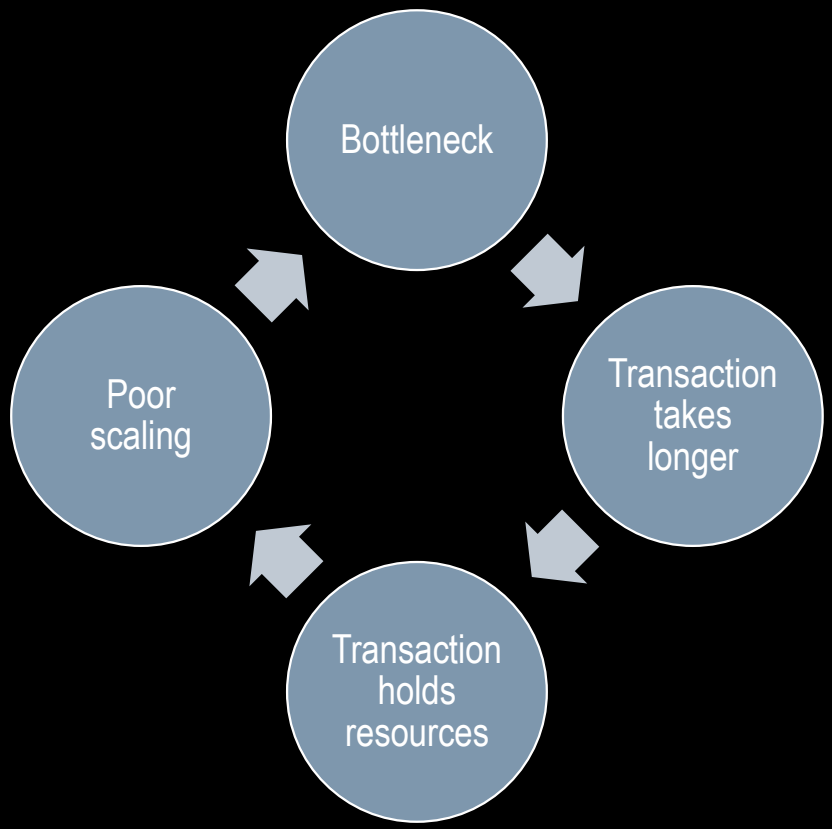
NON-CLUSTERED INDEX GUIDELINES

- Create for frequent search columns
- Use on narrow and highly selective columns
- Place on foreign key constraints (for join queries)
- Check the workload for “covering” queries
  - Consider adding included columns
- The drawback: maintenance cost
  - Frequent updates will ruin perf where there are too many indexes
- Evaluate benefits of [not] indexing small tables

OLTP SCALABILITY DIMENSIONS & DETERMINANTS

Dimensions

- Transaction throughput
- No. of concurrent users
- Data size and growth rate



Resources

- CPU
- Memory
- IO
- Network

Key Design Pattern for Scalability: Divide and Conquer

TYPICAL CPU SCALING ISSUES

Symptoms

- Plan compilation and recompilations
  - Plan reuse < 90% is bad
- Parallel queries
  - Parallel wait type cxpacket > 10% of total waits
- High runnable tasks or sos_scheduler_yield waits

Causes

- Queries not parameterized
- Inefficient Query plan
- Not enough stored procedures
- MAXDOP is not set to 1
- Statistics not updated
- Table scan, range scan
- SET option changes within SP

Use stored procedures and parameterize queries where possible

TYPICAL IO SCALING ISSUES

Symptoms

- High average disk seconds per read (> 10 msec) and write (> 2 msec) for spindle based devices
- Top 2 values for wait stats are one of - ASYNCH_IO_COMPLETION, IO_COMPLETION, LOGMGR, WRITELOG, PAGEIOLATCH_x

Causes

- Aggravated by Big IOs such as table scans (bad query plans)
- Non covering indexes
- Sharing of storage backend – combine OLTP and DW workloads
- TempDB bottleneck
- Too few spindles, HBA's

OLTP applications need to be designed for random I/O

TYPICAL BLOCKING ISSUES

Symptoms

- High average row lock or latch waits
- Will show up in
  - sp_configure “blocked process threshold” and Profiler “Blocked process Report”
  - Top wait statistics are LCK_x. See sys.dm_os_wait_stats.

Causes

- Higher isolation levels
- Index contention
- Lock escalation
- Slow I/O
- Sequence number problem

Use RCSI/Snapshot isolation

TYPICAL MEMORY ISSUES

Symptoms

- Page life expectancy < 300 secs
- SQL Cache hit ratio < 99%
- Lazy writes/sec constantly active
- Out of memory errors

Causes

- Too many large scans (I/O)
- Bad query plans
- External (other process) pressure

Eliminate table scans in query plans

Use WSRM for non SQLServer processes on machine

PERFORMANCE AND SCALE FEATURES IN SQL SERVER 2008 R2

- Better query plans
  - Plan guides
  - Optimize for Unknown
- Lock escalation hints
- Resource governor
- Transparency and Diagnostics – Xevent, DMV's
- > 64 thread support
- Dynamic affinity (hard or soft)
- Hot-add CPU support
- Data Compression
  - Especially if you have I/O issues
- Partitioning
- Snapshot Isolation, RCSI
- Control Point

PLAN GUIDES

- Guide optimizer to use a fixed query plan
- Helps with plan predictability
- **Use when you can't change the application**
- Simple example
  - *SELECT TOP 1 * FROM Sales.SalesOrderHeader ORDER BY OrderDate DESC;*
  - **sp_create_plan_guide** @name = N'Guide2', @stmt = N'SELECT TOP 1 * FROM Sales.SalesOrderHeader ORDER BY OrderDate DESC', @type = N'SQL', @module_or_batch = NULL, @params = NULL, @hints = N'OPTION (MAXDOP 1)';

OPTIMIZE FOR UNKNOWN

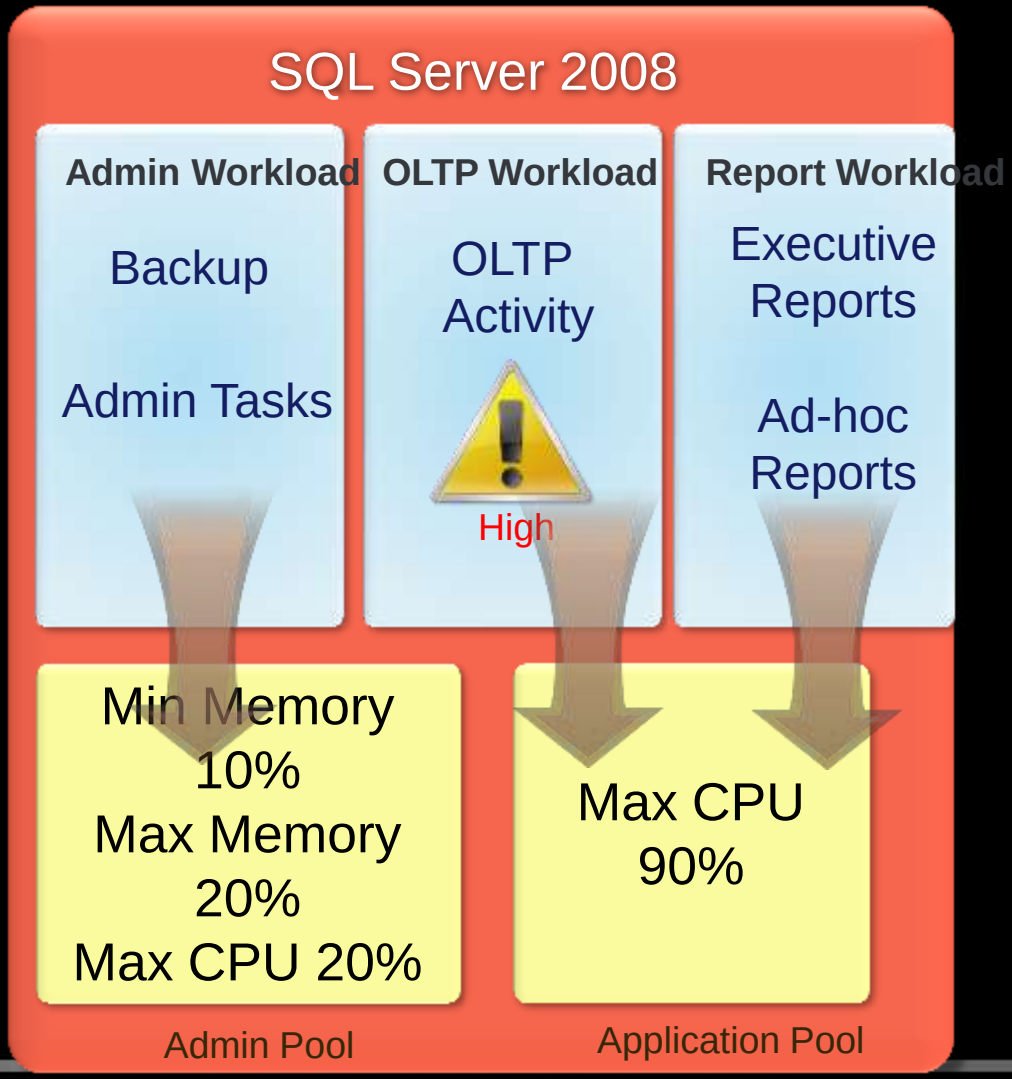
- OPTIMIZE FOR UNKNOWN
  - Hint directs the query optimizer to treat as if no parameters values had been passed
  - Helps solve case where specific parameter values in query result in a bad plan for other values
  - Example
    - *@p1=1, @p2=9998,*
    - *SELECT * FROM t WHERE col > @p1 or col2 > @p2 ORDER BY col1 OPTION (OPTIMIZE FOR (@p1* ***UNKNOWN****, @p2* ***UNKNOWN****))*

DEMO

LOCK ESCALATION CONTROLS

- Check if lock escalation is causing blocking before disabling
- Disable lock escalation at an object or table level
- Enable lock to be escalated to the partition of the table
- If the lock is escalated to partition (Hobt), it is not escalated further
  - *Alter table T1 set (LOCK_ESCALATION = DISABLE)*

RESOURCE GOVERNOR



Benefits

- Provide deterministic Quality Of Service
- Prevent run-away queries
- Tames ill behaved Apps
- DW & Consolidation scenarios

SQL Server 2008 RG

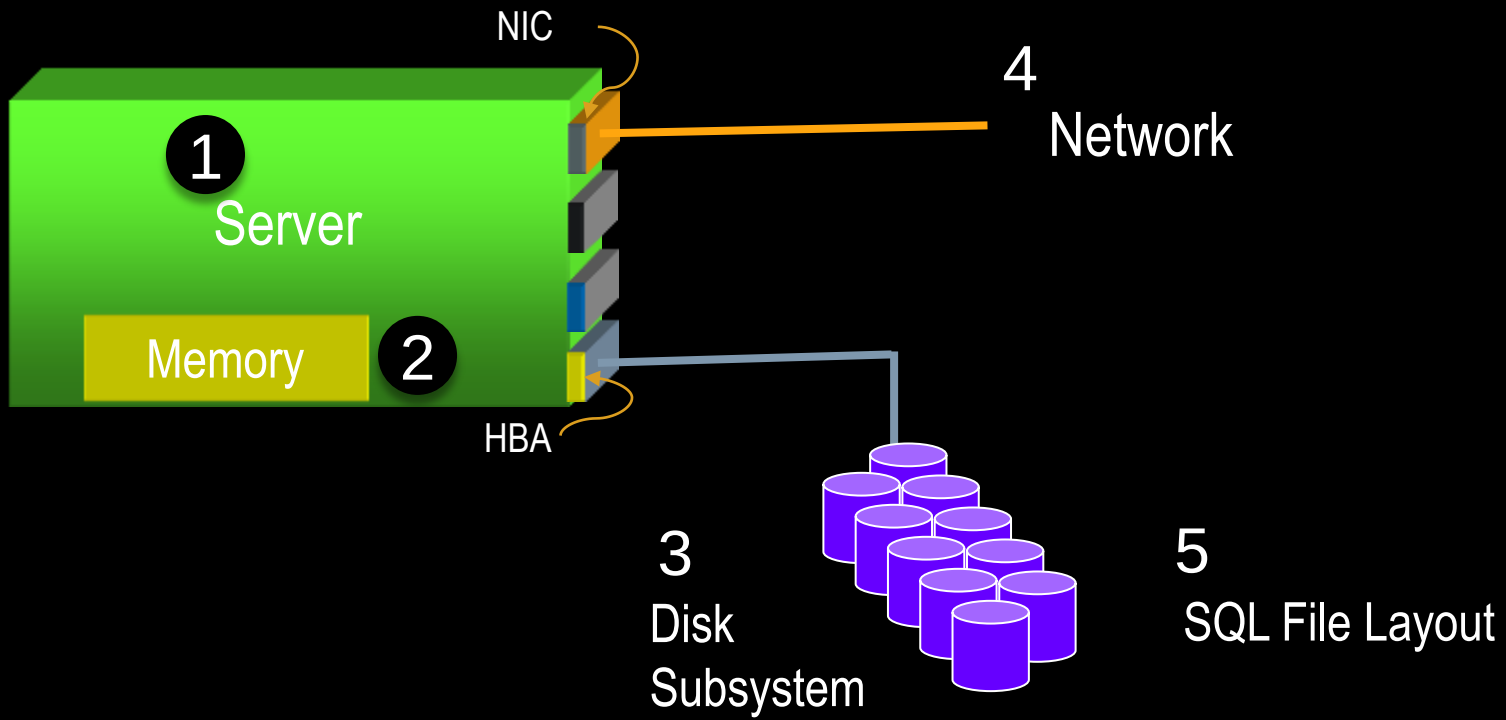
- Workloads are mapped to Resource Pools
- Online changes of groups and pools
- Real-time Resource Monitoring
- Up to 20 Resource Pools

EXTENDED EVENTS (XEVENT)

- Extremely high performance and extensible event and trace mechanism
- Dynamic data collection on event fire
- Integrated with ETW (Event Tracing for Windows)
  - Enables correlation with events exposed by Windows and third party applications
- Hundreds of event points throughout SQL Server code base
- Can identify session/statement level wait statistics

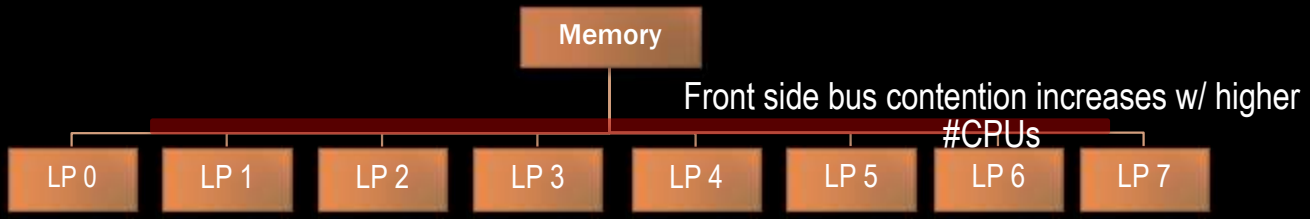
CORE SYSTEM COMPONENTS

The key is to build a Balanced System without bottlenecks

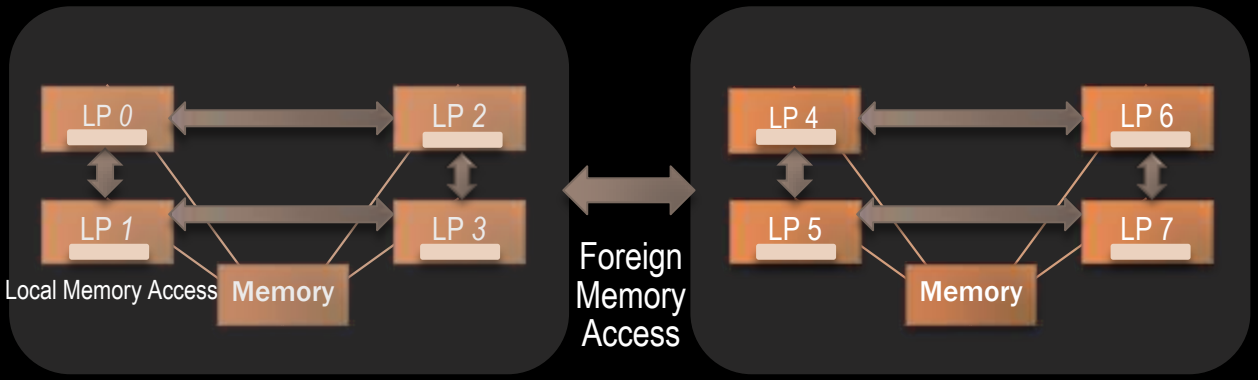


SQL Server is only part of the equation. **Eco system needs to scale.**

CONCEPTS - NUMA



Symmetric Multiprocessor Architecture



Non-Uniform Memory Access

Foreign memory access > local memory access

> 64 thread support exploits NUMA

DISK SUBSYSTEM CONFIGURATION

Trends

- Disk sizes grew by 100 times over last 10 years
- Disk access times only decreased by factor 10
- Disk configuration of high-end systems is not just sizeof(data) but matter of expected I/O workload
- Solid State Disks now more prevalent

Configuration

- Scale throughput with multiple HBA's, spindles
- If using RAID 10 get HBA that can do simultaneous read of the mirrors
- Use multipathing for load balancing
- HBA Queue Depth – default 32 too low at times
- Configure to ensure healthy disk latencies < 10 msec

For OLTP Design for IO/sec
and data warehouse design for throughput

NETWORK

Trends

- Gigabit is standard today. Usable bandwidth typically ~350 Mbps
- 10GBit Ethernet adapters available now – high demand for iSCSI
- Bandwidth not always bottleneck cause
  - Lack of parallel processing of network interrupts

Configuration

- Use Windows Server 2008
  - Offers Distributed network DPC processing
- Suggest one NIC per NUMA node; maximum 4 to 8 cores per NIC
- Use Adapter teaming

Upgrade to Windows Server 2008 to gain these benefits

TOP STATISTICS – SQL SERVER DOES SCALE

Category	Metric
Largest single database	80 TB
Largest table	20 TB
Biggest total data 1 customer	2.5 PB
Highest transactions per second 1 db	36,000
Fastest I/O subsystem in production	18 GB/sec
Fastest “real time” cube	15 sec latency
Data load for 1TB	20 minutes
Largest cube	4.2 TB

SUMMARY

- SQL Server 2008 R2 and Windows together offer an ecosystem to scale the most demanding OLTP applications
- Good application design is a precursor to great scalability

Microsoft®