



Making the Most of SQL Server 2005: Developing World Class Database Applications



Brian A. Randell
Senior Consultant
<http://www.MCWTech.com>



Kimberly L. Tripp
President and Founder
<http://www.SQLskills.com>



About Brian

- Senior Consultant with MCW Technologies, LLC
- Contact me via my blog or e-mail:
 - <http://www.mcwtech.com/cs/blogs/brianr/>
 - brianr@mcwtech.com
- Visual Basic MVP
- Author of Visual Studio 2005 Team Edition for Database Professionals Hands-on Labs
- Manager and co-author of Microsoft SQL Server 2005 Hands-on Labs
- Author of Core Team System training course
- Co-author of Effective Visual Basic



About Kimberly

- Consultant/Trainer/Speaker/Writer
- Founder, **SY Solutions, Inc.** (www.SQLskills.com)
 - e-mail: Kimberly@SQLskills.com
 - blog: <http://www.SQLskills.com/blogs/Kimberly>
- Microsoft Regional Director and SQL Server MVP
- Writer/Editor for SQL Magazine www.sqlmag.com
- Author of the Database Mirroring HOL (DAT070) and the Table and Indexing HOL (DAT010) as well as presenting these sessions: DAT325, DAT413 and DAT437 and two chalk talks: on Monday-Why Enterprise Edition and on Thursday-SQLCMD Tips/Tricks)
- Author of several SQL Server 2005 Whitepapers on MSDN/TechNet – Table and Index Partitioning, Snapshot Isolation and .NET SQLCLR for DBAs
- Author/Presenter for more than 25 online webcasts on MSDN and TechNet (two series and other individual webcasts)
- Coauthor MPress Title: *SQL Server 2000 High Availability*
- Presenter/Technical Manager for SQL Server 2000 High Availability DVD
- I (we) still love this stuff... Feel free to ask questions!



SQL Server 2000

A Major Release

- ❑ XML support
 - ❑ SELECT ... FOR XML
 - ❑ OpenXML
 - ❑ XML Views
 - ❑ XML Updategrams
 - ❑ XML View Mapper
 - ❑ XML Bulk Load
- ❑ URL and HTTP db access
- ❑ HTTP access to cubes
- ❑ Multi-instance support
- ❑ Integrated Data Mining
- ❑ Full-Text Search in formatted docs
- ❑ English Query for the Web
- ❑ C2 security rating (NSA)
- ❑ Installation disk imaging
- ❑ Active Directory integration
- ❑ Self-management and tuning
- ❑ Distributed Partitioned Views
- ❑ Log Shipping
- ❑ Parallel CREATE INDEX
- ❑ Parallel scan
- ❑ Parallel DBCC
- ❑ Failover clustering
- ❑ Failover cluster management
- ❑ 32 CPU SMP system support
- ❑ 64 GB RAM support
- ❑ VIA SAN support
- ❑ Indexed views
- ❑ ROLAP dimension storage
- ❑ Distributed Partitioned Cubes
- ❑ Online index reorganization
- ❑ Differential backup
- ❑ User-defined functions
- ❑ Server-less snapshot backup
- ❑ SQL Query Analyzer debugger
- ❑ New data types
- ❑ Column-level collations
- ❑ Virtual Cube Editor
- ❑ Linked cubes
- ❑ MDX Builder
- ❑ Dimensions
- ❑ Security in Analysis Services
- ❑ OLAP Actions
- ❑ Custom rollups
- ❑ Cascading referential integrity and actions
- ❑ INSTEAD OF triggers
- ❑ Indexes on computed columns
- ❑ Queued replication
- ❑ DTS enhancements
- ❑ Copy Database Wizard

SQL Server 2005

- .NET Framework**
 - ❑ Common Language Runtime Integration
 - ❑ User-defined Aggregates
 - ❑ User-defined Data Types
 - ❑ User-defined Functions
 - ❑ SQL Server .NET Data Provider
 - ❑ Extended Triggers
- Data Types**
 - ❑ New XML Datatype
 - ❑ nvarchar(max)
- SQL Server Engine**
 - ❑ New Message Service Broker
 - ❑ HTTP Support (Native HTTP)
 - ❑ Database Tuning Advisor
 - ❑ Enhanced Read ahead & scan
 - ❑ Online Index Rebuild
 - ❑ Multiple Active Result Sets
 - ❑ Persisted Computed Columns
 - ❑ Queuing Support
 - ❑ Snapshot Isolation Level
 - ❑ Non-blocking Read Committed
 - ❑ Table and Index Partitioning
 - ❑ NUMA support
- Database Availability and Redundancy**
 - ❑ Fail-over Clustering (up to 8 node)
 - ❑ Enhanced Multi-instance Support
 - ❑ Database Mirroring
 - ❑ Database Snapshots
- XML**
 - ❑ XQUERY Support (Server & Mid Tier)
 - ❑ XML Data Manipulation Language
 - ❑ FOR XML Enhancements
 - ❑ XML Schema (XSD) Support
 - ❑ MSXML 6.0 (Native)
 - ❑ .Net XML Framework Support for XML and CLR integration
- Notification Services**
- Replication**
 - ❑ Auto-tuning Replication Agents
 - ❑ Oracle Publication
 - ❑ Improved Blob Change Tracking
- OLAP and Data Mining**
 - ❑ Analysis Management Objects
 - ❑ Windows Integrated Backup and Restore
 - ❑ UDM, multi-fact table support
 - ❑ DTS and DM Integration
 - ❑ Eight new DM algorithms
 - ❑ Auto Packaging and Deployment
- SQL Integration Services (SQLIS)**
 - ❑ New Architecture (DTR + DTP)
 - ❑ Complex Control Flows
 - ❑ Control Flow Debugging
 - ❑ For Each Enumerations
 - ❑ Property Mappings
 - ❑ Full Data Flow Designer
 - ❑ Full DTS Control Flow Designer
 - ❑ Graphical Package Execution
 - ❑ Immediate Mode and Project Mode
 - ❑ Package (Advanced) Deployment Tools
 - ❑ Custom Tasks and Transformations
- Reporting Services**
 - ❑ Multiple Output Formats
 - ❑ Parameters (Static, Dynamic, Hierarchical)
 - ❑ Bulk Delivery of Personalized Content
 - ❑ Support Multiple Data Sources
 - ❑ STS (Web Parts, Doc Libraries)
 - ❑ Visual Design Tool
 - ❑ Charting, Sorting, Filtering, Drill-Through
 - ❑ Scheduling, Caching
 - ❑ Complete Scripting Engine
 - ❑ Scale Out architecture
 - ❑ Open XML Report Definition
- Dynamic Management Views**
- Management Tools**
 - ❑ SSMS
 - ❑ MDX Query Editor
 - ❑ Version Control Support
 - ❑ SQLCMD Command Line Tool
 - ❑ SQL Service Manager
- Database Maintenance**
 - ❑ Backup and Restore Enhancements
 - ❑ Checksum Integrity Checks
 - ❑ Dedicated Administrator Connection
 - ❑ Dynamic AWE
 - ❑ Fast Recovery
 - ❑ Highly-available Upgrade
 - ❑ Online Index Operations
 - ❑ Online Restore
 - ❑ Parallel DBCC/Index Operations
- Performance Tuning**
 - ❑ Profiler Enhancements
 - ❑ Profiler/Perfmon Integration
 - ❑ Profiling Analysis Services
 - ❑ Exportable XML Showplan
 - ❑ Exportable Deadlock Traces
- Full-text Search**
 - ❑ Indexing of XML Datatype
 - ❑ Index files integrated with SQL backup
- MDAC**
 - ❑ Side by Side installation
 - ❑ Microsoft Installer base setup
 - ❑ Support for Active Directory Deployment
- SQL Client .NET Data Provider**
 - ❑ Server Cursor Support
 - ❑ Bulk load support
- Security**
 - ❑ No default metadata access
 - ❑ Fine Grained Administration Rights
 - ❑ Separation of Users and Schema
 - ❑ SSL Encryption at installation
 - ❑ Data Encryption
 - ❑ Code-signing

Developer and DBA Convergence

Developers and DBA's have traditionally worked to an interface, defined by:

- Table or View definition
- Stored Procedure & result set definition

Communication Barriers

Development doesn't understand good query design

Database Administration doesn't understand our development priorities

Database Architect

Increase communication and collaboration via product integration

Reduce complexity through clear release management processes

- Pockets of information within disciplines
- Unclear delineation of responsibilities
- Conflicting best practices and architectures
- Conflicting strategic goals and objectives

Developer and DBA Convergence

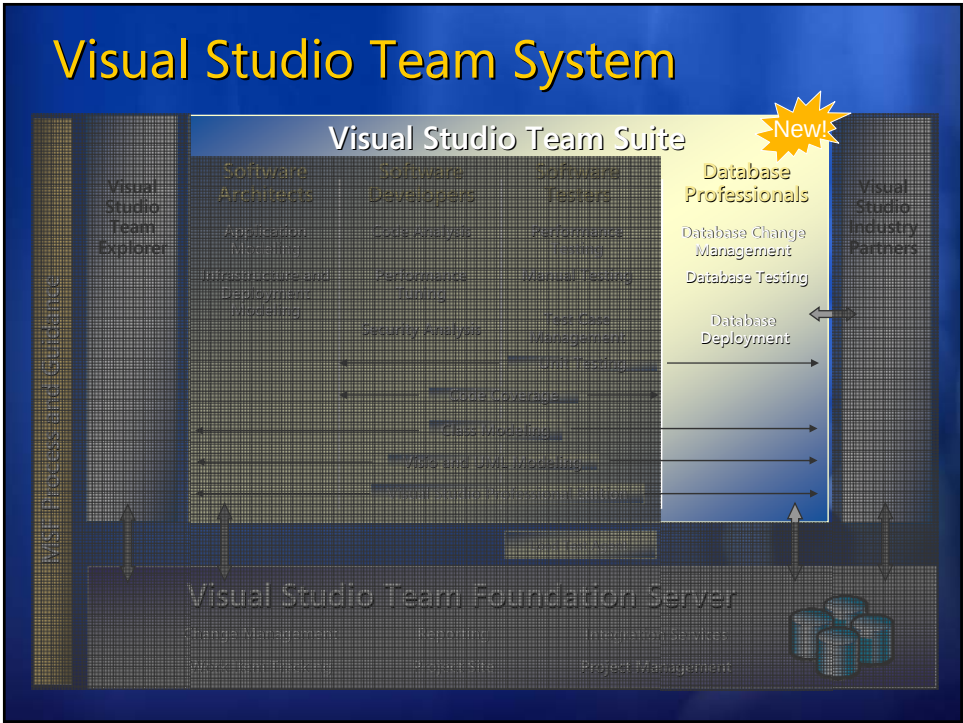
- Developers:

 - Get productive with the new tools
 - Closer integration of DBAs into the development process
 - Design application to be:
 - Supportable
 - Resilient to change
 - Ensure secure, robust implementation
- DBAs:

 - Maintain role as data steward
 - Understand & advise on appropriate technology choices
 - Factor XML & CLR into schema design
 - Ensure secure, robust implementation



Visual Studio Team System



Developing World Class Database Applications – Agenda

- Installation - An Effective Base for Performance and Security
- Initial Configuration
- Optimal Database Structures
- Optimal Table Structures and Design
- Developing on SQL Server 2005
 - Optimizing Procedural Code
 - Choices, choices and more choices
- Caching
- Tuning, Monitoring, and Diagnosing System Problems

Installation

- Which version has the features you need?
- Server Installation
- Initial Configuration

Installing SQL Server 2005

- Getting a current build
- Installation through Windows Installer
- Pre-installation checklist
 - Operating System Choices
 - Visual Studio – Beta?
- Versions from which to choose
 - 32-bit v 64-bit (Itanium and x64)
 - Developer, Enterprise, Express...
- Licensing and features
- SCC (System Configuration Checker)
- Installation

Choosing a Product to Install

- SQL Server 2005 Enterprise Engine
 - Includes Enterprise Edition, Enterprise Evaluation Edition and Developer Edition – only difference is licensing
 - 32-bit and 64-bit (Itanium and x64)
- SQL Server 2005 Standard Edition
 - 32-bit and 64-bit (Itanium and x64)
- SQL Server 2005 Workgroup Edition
 - 32-bit only
- SQL Server 2005 Express Editions
 - SQL Server 2005 Express Edition
 - SQL Server 2005 Express Edition with Advanced Services
 - SQL Server 2005 Express Edition Toolkit
- Can run multiple versions side-by-side

Microsoft Express Product Range

Visual Studio 2005 Express ~~Beta~~ Products

The Express products, expanding the Visual Studio product line to include lightweight, easy-to-use, easy-to-learn tools for hobbyists, enthusiasts, and novices who want to build dynamic Windows applications and Web sites.

					
Develop for Windows	Develop for Windows	Develop for Windows	Develop for Windows	Develop for Web	Develop for Databases
Visual Basic 2005 Express Edition, which was built with a focus on productivity, is ideal for the first time or casual programmer, students, or anyone with prior Visual Basic programming experience.	Visual C# 2005 Express Edition combines some of the best aspects of the Java language and C/C++ to create a great combination of power and productivity for students, hobbyists and enthusiasts.	Visual C++ 2005 Express Edition typically takes longer to master but can offer more horsepower and a finer degree of control than the other Express products.	Visual J# 2005 Express Edition is an implementation of the Java language and is an ideal tool for anyone with prior Java experience or students using Java in school.	Visual Web Developer 2005 Express Edition is focused exclusively on Web development with ASP.NET 2.0. Choose from Visual Basic, C#, or J# languages.	SQL Server 2005 Express Edition complements the other Express products by providing database support that is both powerful and easy to use.

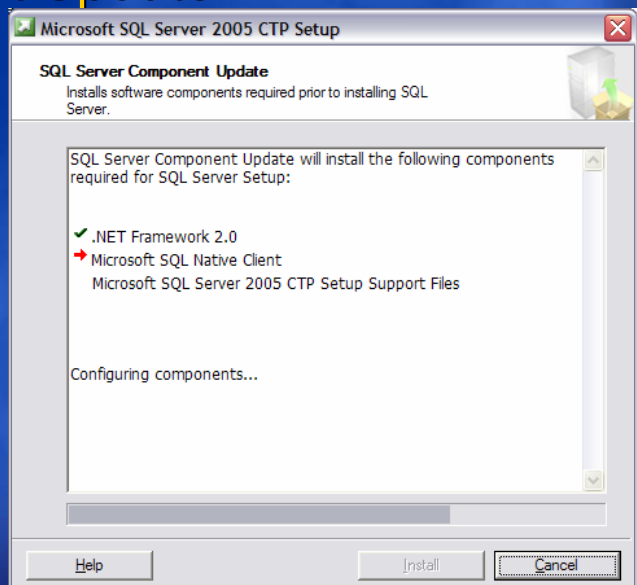
Be sure to check the resource links for Brian's articles on SQL Server Express!

Installation as an "Application"

- Not using Install Shield
 - Third party product
 - Hard to "control"
 - Only stopped installation before file copy: "Installation has enough information to start copying files..."
 - Limited rollback capabilities, uninstalling partial and failed install was messy
- Uses Windows Installer
 - Microsoft product
 - Can download/review Windows Installer SDK
 - MSDN, Win32 and COM Development, Administration and Management, Setup, Windows Installer
 - Components have an MSI file
 - Installation can be rolled back until the very end

Component Update

- .NET Framework 2.0
- SQL Native Client (SNAC)
- Support Files



 Hidden Slide
w/extra details

.NET Framework 2.0

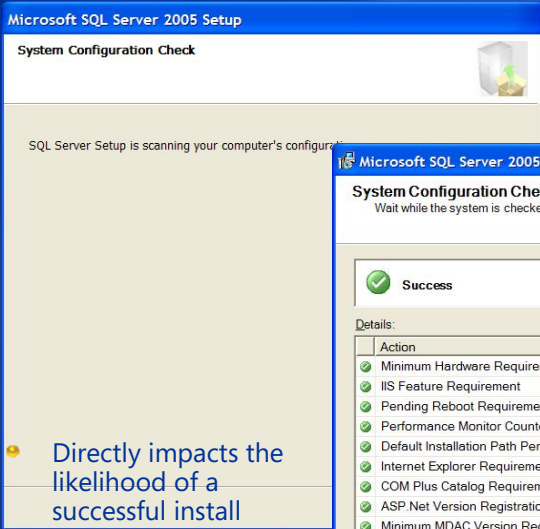
- Required for:
 - SQL Server Management Studio
 - Express Manager (used for SQL Express)
 - Engine (used by SQLCLR)
- How to Identify .NET Framework Version on your Machine
 - %WINDIR%\Microsoft.NET\Framework\version Right-click mscorlib.dll, click Properties, and then click Version.
 - Administrative Tools, .NET Framework 2.0 Configuration. At the top of the right pane, .NET Framework version displays.

 Hidden Slide
w/extra details

SQL Native Client

- Why SNAC?
 - Separates "infrastructure" from "drivers"
 - SNAC is owned by SQL and only updated by SQL (during install/service packs)
 - SNAC is on the "drivers" side
 - Infrastructure owned by OS and only updated by OS
 - Enables side-by-side driver installation with minimal downtime
 - MDAC doesn't allow side-by-side therefore requires stopping the app
 - MDAC to be removed on 64-bit as well

 Hidden Slide
w/extra details



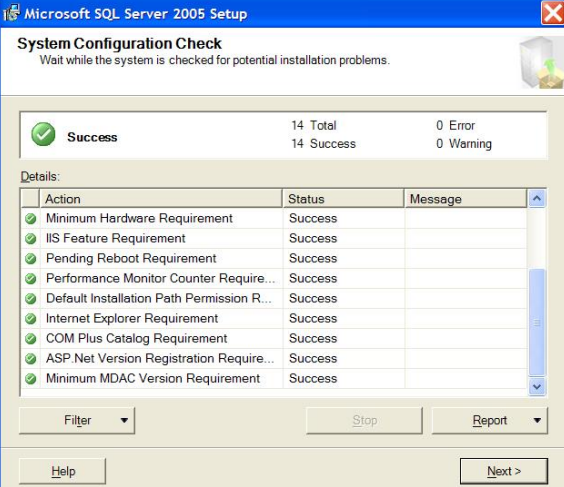
Microsoft SQL Server 2005 Setup

System Configuration Check

SQL Server Setup is scanning your computer's configuration...

System Configuration Checker

- Directly impacts the likelihood of a successful install
- Doesn't mean you'll have success with upgrade...



Microsoft SQL Server 2005 Setup

System Configuration Check

Wait while the system is checked for potential installation problems.

Action	Status	Message
Minimum Hardware Requirement	Success	
IIS Feature Requirement	Success	
Pending Reboot Requirement	Success	
Performance Monitor Counter Require...	Success	
Default Installation Path Permission R...	Success	
Internet Explorer Requirement	Success	
COM Plus Catalog Requirement	Success	
ASP Net Version Registration Require...	Success	
Minimum MDAC Version Requirement	Success	

14 Total 0 Error
14 Success 0 Warning

Filter Stop Report Next >

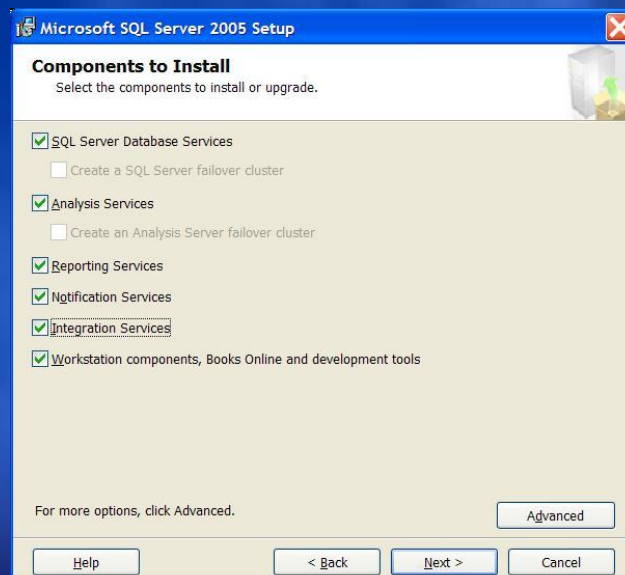
 Hidden Slide
w/extra details

Hardware Recommendations

- File System—NTFS, no compression for installation drive or primary database file or log file
- More smaller/faster disks over fewer/larger disks
- Be sure to update all drivers and firmware!
- OS should be on redundant disks
- SQL installation should be on redundant disks
- Transaction logs should be on fast, redundant and preferably isolated disks
- TempDB should be isolated and if on multi-proc use multiple files (Q328551 – see section “Increase the Number of Tempdb Data Files with Equal Sizing”)
- Databases should be “moved” to support better performance (Q224071)
- KB Articles ⇒ Q224071 (SQL 200x) and Q328551 (SQL 2000 – but best practices still apply)

Components to Install

- Separate “packages” (.MSI files)
- High Level is for “typical” set of features
- Click Advanced for feature tree



Components to Install

SQL Server Database Services	Database Engine, Replication, Full-Text
Analysis Services	Analysis Services server components for Business Intelligence
Reporting Services	Reporting Services and Report Manager
Notification Services	Notification Services (Engine and Client), Bulk Event for XML
Data Transformation Services (Integration Services)	SQL IS replaces SQL DTS
Workstation components, Books Online and development tools	Management Tools, Command Prompt Tools, Reporting Services Tools, Connectivity Components, Programming Models, Management Studio, Computer Manager, Profiler, and Replication Monitor Business Intelligence Development Studio Documentation and samples - Books Online, code samples

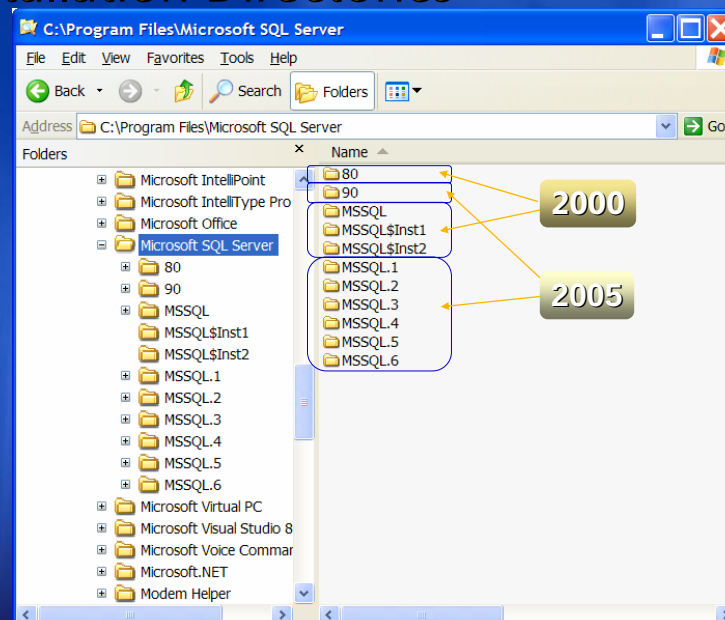
Default and Named Instances

- Default Instance
 - Accessed by "computername"
 - SQL Server Service = mssqlserver
 - SQL Server Agent Service = sqlserveragent
 - Full-Text = MSFTESQL
 - Analysis Server = MSSQLServerOLAPService
- Named Instance
 - Accessed by computername\instancename
 - SQL Server Service = mssql\$instance
 - SQL Server Agent Service = sqlagent\$instance
 - Full-Text = msftesql\$instance
 - Analysis Server = MSOLAP\$instance

Directory Structure

- No longer dependent on instance name
- All directories MSSQL.n even for other components such as Analysis Services
- How can you detect?
 - Registry: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL Server\Instance Names\SQL
 - Name = Instance name (without MSSQL)
 - Data = Install Directory (mssql.n)

Installation Directories



Programmatically Detecting Dir

```

DECLARE @InstanceName sql_variant
DECLARE @SQLDataRoot sql_variant
DECLARE @SQLDataRoot nvarchar
DECLARE @SQLDataRoot nvarchar max

SELECT @InstanceName
ISNULL SERVERPROPERTY 'InstanceName' 'MSSQLServer'

EXECUTE master 'HKEY_LOCAL_MACHINE'
'SOFTWARE\Microsoft\Microsoft SQL Server\Instance Names
\SQL' OUTPUT

SELECT @ExecStr 'EXECUTE master.dbo.xp_regread '
'''HKEY_LOCAL_MACHINE''', '
'''SOFTWARE\Microsoft\Microsoft SQL Server\'
convert varchar
\'Setup'', \'SQLDataRoot\'', @SQLDataRoot OUTPUT'

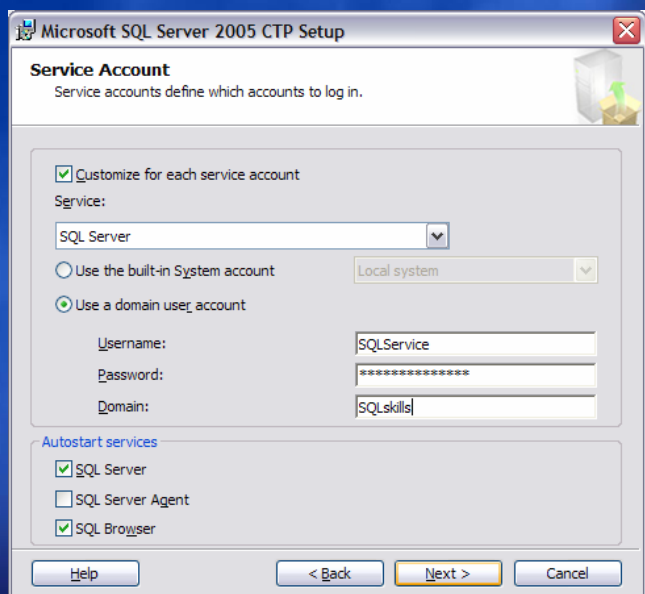
EXEC master sp_executesql @ExecStr
@SQLDataRoot nvarchar(512) OUTPUT'
OUTPUT

```

See Backup2000Restore2005.sql for SQLCMD master script.

Startup Service

- Customize allows you to set each individually and to different accounts
- Local System or
- Domain Account...



Service Account Choices

- Pros and Cons for each, re: security/control
 - ✓ Dedicated Local account (not local administrator)
 - Local service (do other services use this?)
 - ✓ Network service (do other services use this?)
 - ✓ Dedicated Domain account (not local administrator)
 - Local account which is a local administrator
 - Domain account which is a local administrator
- NEVER use:
 - Local System
 - Domain Administrator

Principle of Least Privilege Applies

Never give more rights than necessary!

Authentication

- Windows
- Mixed
- Password only specified for Mixed Mode

Important: Set the sa password immediately after installation to ensure password is known value rather than randomly generated value

Microsoft SQL Server 2005 CTP Setup

Authentication Mode
The authentication mode specifies the security used when connecting to SQL Server.

Select the authentication mode to use for this installation.

☒ Windows Authentication Mode

☐ Mixed Mode (Windows Authentication and SQL Server Authentication)

Specify the sa login password below:

Enter password:

Confirm password:

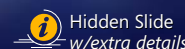
Help < Back Next > Cancel

Windows Authentication

- More Secure
 - Strong Password requirements
 - Account Lockout policies
 - Password expiration
- Can be changed later
- After installation be sure to manually set the password of the sa account otherwise internally set to a randomly generated password—if you change to mixed mode you will “enable” your sa account but not know the password. However, you can easily change it.

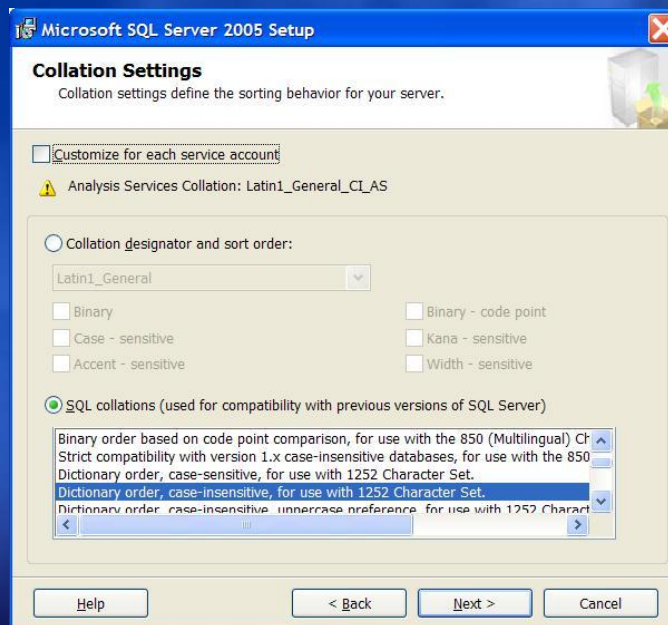
Mixed Mode Authentication

- Windows Authentication plus
- SQL Server Authentication
 - Uses Windows 2003 Server password policies, if installed on Windows 2003 Server
 - Requires sa password to be set during installation if Mixed mode is chosen
 - Stronger password requirements in 2005
 - All passwords must be at least 6 characters long and satisfy at least three of the four criteria:
 - It must contain uppercase letters (A-Z)
 - It must contain lowercase letters (a-z)
 - It must contain numbers (0-9)
 - It must contain non-alphanumeric characters (!, #, %, or \$)
 - See BOL for complete list of requirements



Collation Setting

- Affects searches, lookups and ORDER BY
- Character data is affected:
 - ASCII
 - char
 - varchar
 - text
 - Unicode
 - Nchar
 - Nvarchar
 - Ntext



Collation Concepts



- May be changed at any point in chain
- Different options offer different gains – most in language sorting, some in performance
- Can assign on the fly for just a single query – or within a view for subsequent use
- Computed Columns and Indexed Views can aid in performance for lookups with changed collations (there are likely to be version specific restrictions)

Collation Options

- To see the list of collations
`SELECT * FROM ::fn_helpcollations()`
- To see the server's setting
`sp_helpsort`
- To see the database's setting
`sp_helpdb dbname`
- To see the table's setting (for each column)
`sp_help tname`
- For more information, check out these BOL topics:
 - **COLLATE**
 - "Specifying Collations"

Changing Database Collation

```
ALTER DATABASE TestCollation
    COLLATE Latin1_General_CS_AS_KS_WS
go

-- Confirm the change?
SELECT DATABASEPROPERTYEX 'TestCollation' 'Collation'
go
```

- Be careful moving from sensitive to insensitive, may yield errors due to duplicate object names
- Make sure to do significant application testing before changing a collation in production

Working with Temporary Objects

```
USE DB_with_Diff_Collation_than_TempDB
go

CREATE TABLE #test
(
    [column_name] varchar(100) COLLATE database_default
)
go

USE Tempdb
go

-- Confirm that the column RETAINS the Diff DB Collation!
sp_help 'tempdb..#test'
go
```

- If using a different collation than tempdb, temporary objects may have problems in comparisons, etc. unless you use: COLLATE database_default

Best Practices for Collations

- Have corporate standard, use consistently
- If you don't know, accept the default as is most widely used and is compatible with .NET by default
- Can be changed later (at many levels)...
- Binary Sort – excellent for testing scenarios to make sure you are using consistent coding practices, naming conventions, etc. Most strict environment in which to work



Initial Configuration

- Off by default/SQL Surface Area Configuration
- SSL Encrypted Communications
- SQL Server Configuration Manager
- Special Considerations
 - Running SQL Server 2000 and SQL Server 2005 side-by-side
 - Renaming a server

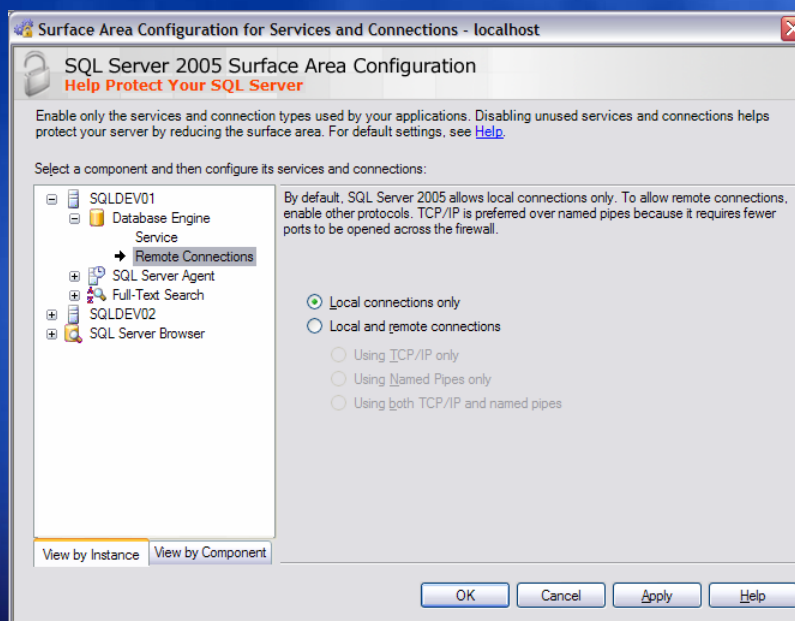
Off by Default

- Principle of least privilege
- Minimal Configuration to run – many installed services (even when chosen during installation) are “off by default”
- Secure by Design
- Secure by Default
- Secure by Deployment
- Where to enable, where to view?
 - sp_configure, Computer Management, catalog views, no single central way
 - ⇒ Enter SQL Surface Area Configuration

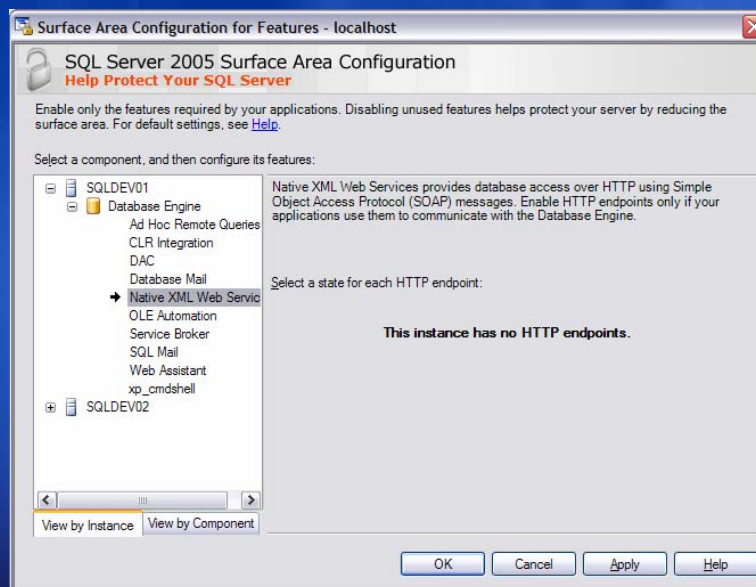
Surface Area Configuration

- SQL Server Program Group, Configuration Tools
- Review SQL Server Setup Help from link on main dialog
- Configure Services and Protocols
 - In this first release more options in Configuration Manager
- Surface Area Configuration for Features
 - "Lockdown" most common/vulnerable features
 - sp_configure settings
 - Catalog View queries to view (endpoints)

Services – Local Only?



SQLSAC – HTTP Endpoints

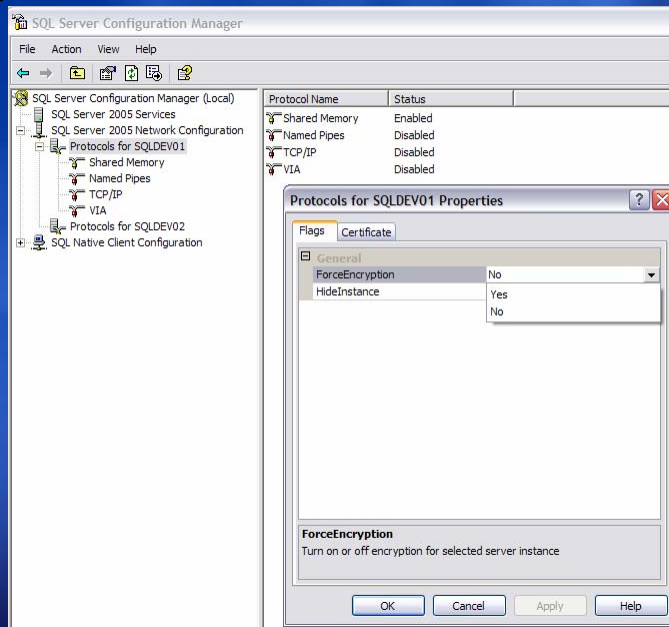


Encrypted Communications

- SSL is used even if not present at installation
 - Checked at installation, if present used
 - If not present, SQL Server generates a 1024 bit certificate
- Logins/passwords encrypted always
- Uses certificate created at installation if SSL not present
- Use "Certificate Picker" in SQL Server Configuration Manager to explicitly choose SSL certificate

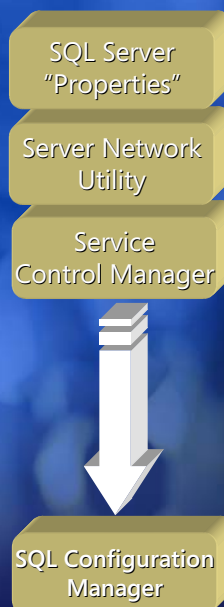
SSL Encryption/Certificate Picker

- Protocol Properties
- Login/pwd encrypted
- ALL traffic encrypted = YES
- Certificate Picker tab

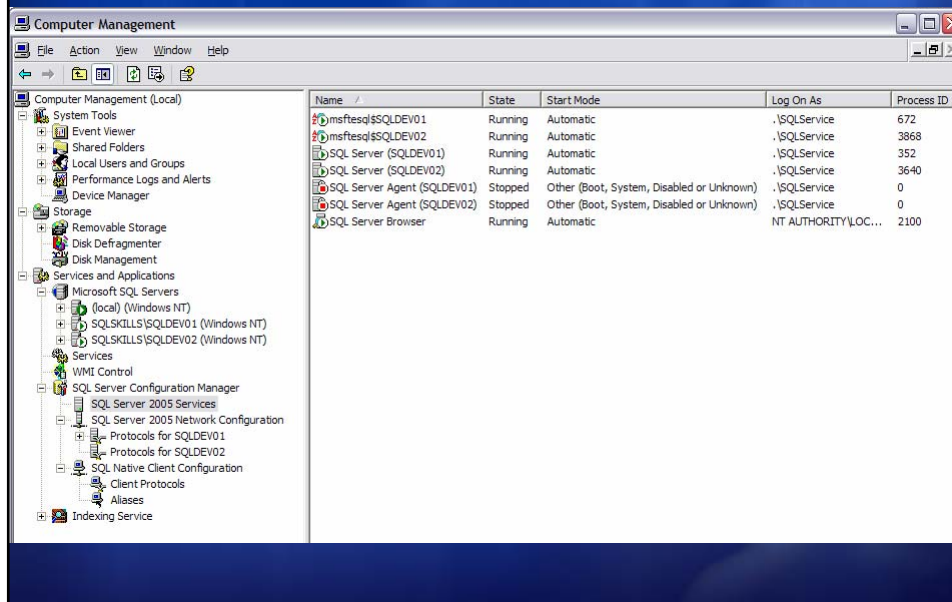


Configuration Manager

- Start and Stop Services
- Change auto start and service properties
- Does not require your services to be started to change properties
- Service Account password changes don't require restart of service
- Control Supported Network libraries and SSL encryption
- Create aliases
- Launch from:
 - SSMS, Right click Configure Services
 - SQL Configuration Manager
 - Computer Management



Computer Manager – 2000/2005



Running 2000/2005 "Side-by-side"

- Possible combination:
 - SQL Server 2000 default/named instance(s)
 - SQL Server 2005 default/named instance(s)
 - SQL Server 2005 can support multiple instances of multiple versions (e.g. Developer and Express)
 - Don't make a design mistake of assuming the version of the default instance
- Possible troubles:
 - Enterprise Manager Database Properties crashes
 - Color coding incorrect in Query Analyzer
- Fix: re-register all COM components
 - At command prompt in 80\tools\binn and mssql\binn dirs, execute:


```
FOR %i IN (*.dll) DO regsvr32 /s %i
```

Renaming A Server After Startup

- Server name is NOT detected by SQL Server
- Review the list of servernames
`sp_helpserver`
- Drop the old servername
`sp_dropserver 'oldservername'`
- Add the new servername – make sure you define it as the “local” instance of SQL Server
`sp_addserver 'newservername', local`
- May have problems with servername related objects/jobs
– i.e. msdb.dbo.sysjobs

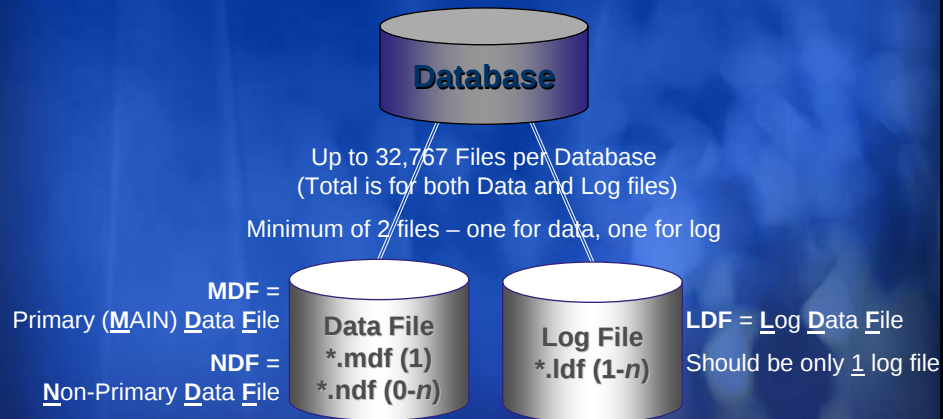
```
UPDATE sysjobs
  SET originating_server = 'newservername'
  WHERE originating_server = 'oldservername'
```

Optimal Database Structures

- Anatomy of Data Modifications
- Optimizing Data/Log Files
- Logging and Recovery
- Transactions and Isolation

Database Structure

Up to 32,767 Databases per Instance



The Anatomy of a Data Modification

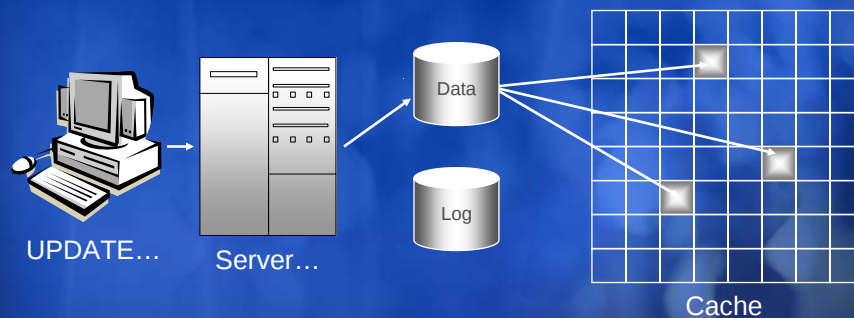
1. User sends UPDATE

- Update is highly selective (only 5 rows)
- Indexes exist to aid in finding these rows efficiently
- The update is a SINGLE statement batch NOT enclosed in BEGIN TRAN...COMMIT TRAN block therefore this is IMPLICIT transaction

The Anatomy of a Data Modification

2. Server receives the request and locates the data in cache OR reads the data from disk into cache
 - Since this is highly selective only the necessary pages are read into cache (maybe a few extra but that's not important here)
 - Let's use an example where the 5 rows being modified are located on 3 different data pages

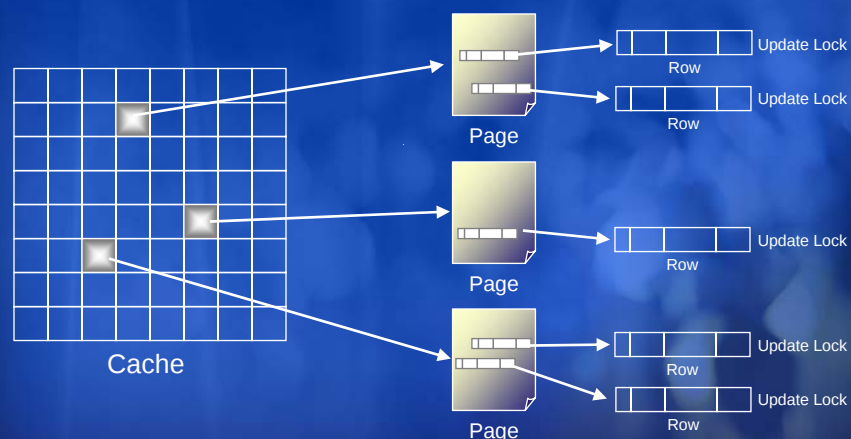
What it looks like: Data



The Anatomy of a Data Modification

3. SQL Server proceeds to lock the necessary data
 - Locks are necessary to give us a consistent point FOR ALL rows from which to start
 - If any other transaction(s) have ANY of these rows locked we will wait until ALL locks have been acquired before we can proceed.
 - In the case of this update (*because it's highly selective and because indexes exist to make this possible*) SQL Server will use row level locking.

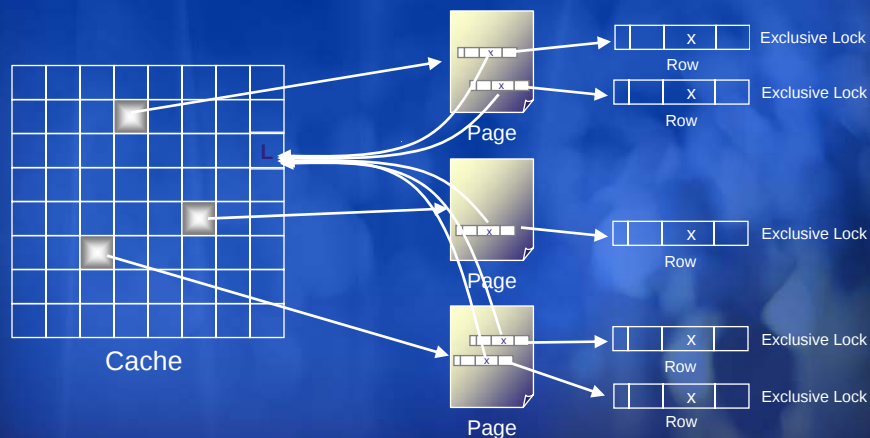
What it looks like: Locks



The Anatomy of a Data Modification

4. SQL Server can now begin to make the modifications – for EVERY row the process will include:
 - Change to a stricter lock (eXclusive lock)
 - An update lock helps to allow better concurrency by being compatible with other shared locks (readers). Readers can read the pre-modified data as it is transactionally consistent
 - The eXclusive lock is required to make the change because once modified no other reads should be able to see this un-committed change
 - Make the modification (in cache)
 - Log the modification to the transaction log pages (also in cache)

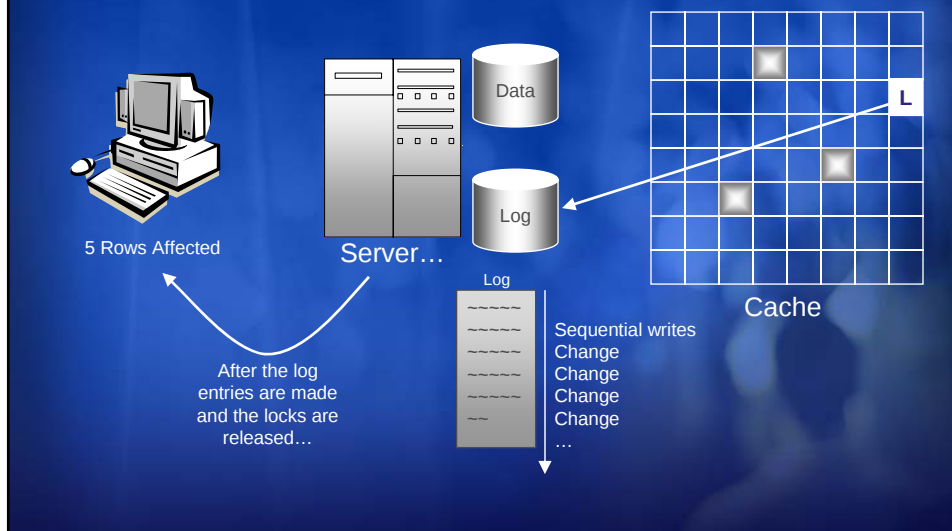
What it looks like: Modifications



The Anatomy of a Data Modification

5. Finally, the transaction is complete – this is the MOST critical step (when @@trancount = 0)
 - All rows have been modified
 - There are no other statements in this transaction – i.e. Implicit transaction
 - Steps are:
 1. Write all log pages to transaction log ON DISK
 2. Release the locks
 3. Send a message to the user:
(5 Rows Affected)

What it looks like: Write-Ahead Logging



So now what?

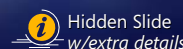
- The transaction log ON DISK – is up to date
- The data in CACHE – is up to date
- But when does the data get written from cache to disk?

Checkpoint

Checkpoint

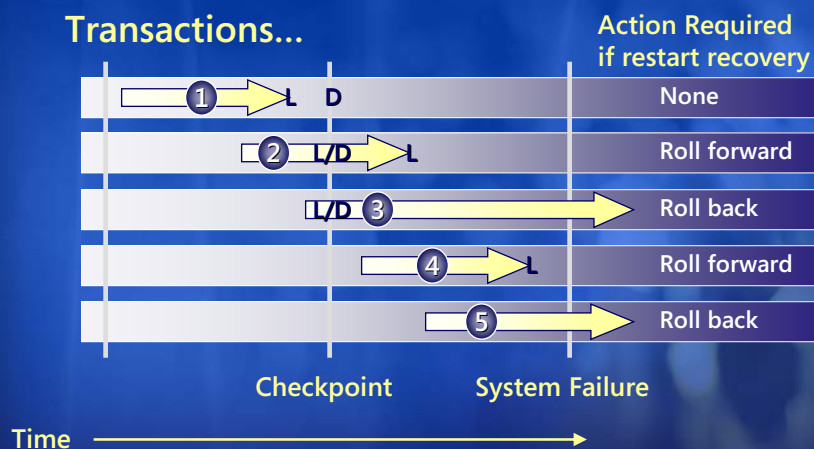
*It's important to realize that a checkpoint does NOT just write committed pages...
Instead a checkpoint writes ALL pages which have changed since they were brought into cache – regardless of the state of the transaction which changed them!*

- Why?
 - To reduce roll-forward recovery time during restart recovery
 - To batch I/Os to disk and reduce disk thrashing for data writes



Hidden Slide
w/extra details

Transaction Recovery and Checkpoints



Hidden Slide
w/extra details

Restart Recovery

- Automatically on system restart
- SQL Server 2000
 1. Redo
 2. Undo
 3. Users allowed access to database
- SQL Server 2005 (Enterprise Edition Only)
 1. Redo
 2. Users allowed access to database
 3. Undo
- Faster access to database – including faster failover on Cluster
- Records are protected (i.e. Locked)

Hidden Slide
w/extra details

Key Points

- Data Portion mostly random reads – except at checkpoint
- Log Portion mostly sequential writes
- Separate physical disks minimizes contention at the drive level – first choice in tuning
- Log is critical in recovery
- Protect the log
- Minimize impact to log
- Be aware of environmental changes

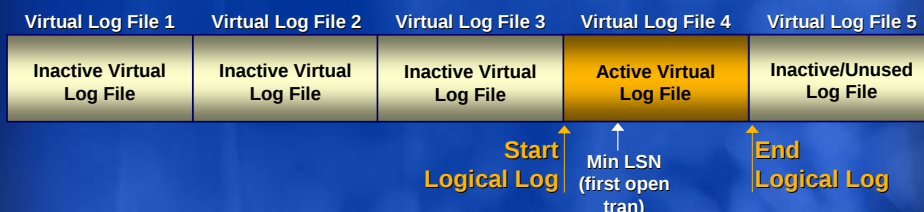
Optimizing Data Files

- Defrag the physical disks
- Effective RAID array configuration
- Pre-allocate to a reasonable initial size
- Don't let auto-growth get out of control
- Consider allowing Instant Initialization
- While a lot of these will help for the log (and I'll explain why in a moment) there are more important things to be aware of IN the data portion – tables and indexes
 - Extent Scan Fragmentation (interleaved objects)
 - Index Fragmentation

Optimizing Log Files

- Isolate **the** Transaction Log (only one!)
- Defrag the physical disks
- Effective RAID array configuration
- Pre-allocate to a reasonable initial size
- Don't let auto-growth get out of control
- Check and fix your internal fragmentation
- Don't be caught up in nothing but speed!
- See Kim's blog entry: [8 Steps to Better Transaction Log Throughput](#), June 25, 2005

How the Transaction Log Works



- On commit, activity is written to the log – active and likely sequential
- Activity moves through log sequentially, fills and then goes to a second file or autogrows
- Excessive autogrowth causes:
 - Slight pause on autogrow
 - Windows call to extend a file (may cause file fragmentation)
 - Adds "VLFs" to the file on each autogrowth

The Effects of Logging

- Log is written AHEAD of the data portion
- Log is the ONLY place where transactional consistency is known (i.e. guaranteed)
- Once a checkpoint occurs SQL Server doesn't need the information in the log – for committed (or inactive) transactions (the log could even be cleared however...)
- Without the transaction log a database cannot function (i.e. marked suspect)
- Need to make sure this is redundant AND optimal...
- What can effect the logging – and therefore performance?

Minimize impact of logging

- Cannot clear the log unless the transactional information is inactive – if a transaction is pending then it stays in the log until it's completed...
- Restart recovery will take longer if there were a lot of long running transactions pending at the time of failure
- Always try to avoid:
 - Long running transactions
 - Transactions which span more than one batch
 - Transactions that require user interaction
 - Nested Transactions (complex and usually longer running)

Consider changing recovery model

What can impact your ability to recover?

- Full Recovery Model ⇒ Everything is Fully Logged
 - All operations allowed and FULLY logged
 - Operations like creating or rebuilding an INDEX takes as large a log as the size of the operation
- Bulk_Logged Recovery Model ⇒ Minimal Logging for SOME Operations (**NOT** non-logged)
 - Operations whose performance is affected, see BOL Topic ⇒ Minimally Logged Operations
 - ALL other operations (i.e. updates, inserts, etc. take the same log space and time as the FULL recovery model
- Simple Recovery Model ⇒ Log Truncation on Checkpoint

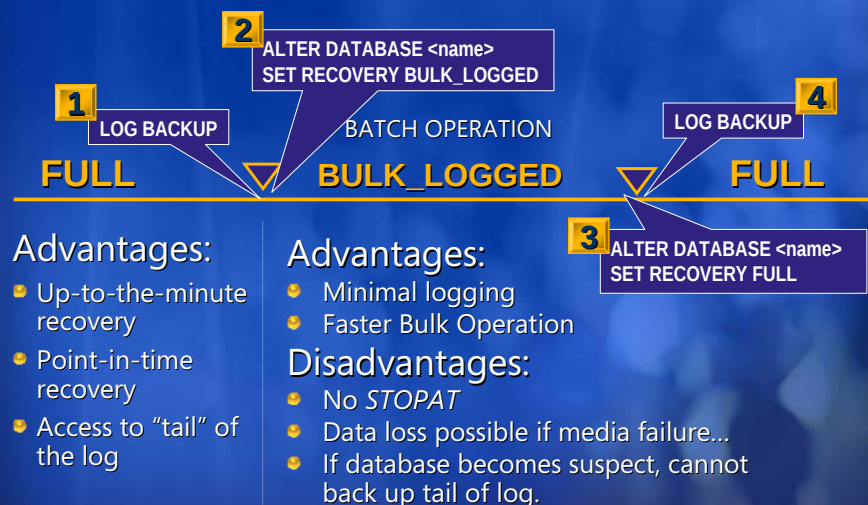
Logging, Recovery Models, Recovery and Performance

- Acceptable to Switch?
- How, Why and Why Not?
- Batch Processing Scenario
- Batch Processing Script
- Switching Recovery Models – Summary

Acceptable to Switch?

- Yes
 - If users are not allowed in database during batch processing
 - No modifications are made during batch process that are not recoverable through some other means (i.e. re-running the batch process)
 - Data loss during the batch process is OK
- No
 - Modifications are made by users/processes that are not recoverable
 - OLTP System requiring 24x7x365
 - Data loss is NOT acceptable
 - Database Mirroring

Switching Recovery Models



What About Locking?

- ACID Transaction Design Requirements
 - Atomicity Consistency Isolation Durability
- Isolation Levels
 - Level 0 – Read Uncommitted
 - Level 1 – Read Committed
 - Level 2 – Repeatable Reads
 - Level 3 – Serializable
- Default Isolation Level in BOTH 2000/2005 is ANSI/ISO Level 1, Read Committed
- In implementation this default level uses locking

ACID Properties

- Atomicity
 - A transaction must be an atomic unit of work; either all of its modifications are performed, or none
- Consistency
 - When completed, a transaction must leave all data and all related structures in a consistent state
- **Isolation**
 - A transaction either sees data in the state it was in before another concurrent transaction modified it, or it sees the data after the second transaction has completed, but it does not see an intermediate state
- Durability
 - Transaction should persist despite system failure

Isolation Level 0 – Read Uncommitted

Phenomenon: Dirty reads

- A read transaction can read another transaction's uncommitted (or in-flight) changes – resulting in "dirty reads"
- DML statements always use exclusive locking
- **In Implementation:** row locks are not used (SCH_S locks are used) for the dirty read transaction and locks against data being accessed are not honored
- **Resulting Phenomenon:** statements execute with the possibility of inaccurate data since the "in-flight" data read may continue to change or even be invalidated (rolled back)

Isolation Level 1 – Read Committed

Phenomenon: Inconsistent Analysis

- The default behavior in ALL releases
- In-flight transaction's data cannot be read by a read committed transaction – only committed changes are visible
- DML statements always use exclusive locking
- **In Implementation:** locks are released (for readers – not modifiers) as resources are read, a row may be read more than once in some scenarios
- **Resulting Phenomenon:** Reads are not repeatable through the life of a transaction – as a result a row may not be read consistently during the life of a transaction

Isolation Level 2 – Repeatable Read

Phenomenon: Phantoms

- In-flight transaction's data cannot be read and data modified is accessible only to the repeatable read transaction
- Data read, but not modified is accessible to other transactions for reads, but not DML
- **In Implementation:** locks are held for the life of a transaction, rows which are read are locked and can be repeatably read during the life of a transaction
- **Resulting Phenomenon:** rows which were not present at the beginning of the transaction can appear – in the result

Isolation Level 3 – Serializable

Key range
and index
reminder

Phenomenon: None

- In-flight transaction's data cannot be read and data modified is accessible only to the serializable transaction
- Data read, but not modified is accessible to other transactions for reads, but not DML
- **In Implementation:** locks are held for the life of a transaction and held at higher levels within indexes to prevent rows from entering the "set"
- **Implementation side effect:** to prevent rows from entering the "set" of data, the "set" of data needs to be locked. If appropriate indexes do not exist then higher levels of locking might be necessary – i.e., table-level locking.

Isolation in Implementation

- All versions of SQL Server – including SQL Server 2005 – use a locking-based implementation by **default**
- In the default environment and to reduce the possibility of various anomalies, transactional duration and possibly higher levels of locking are necessary – resulting in blocking
- There are benefits of locking, if true “isolation” is desired

Locking Prevents Conflicts

At the expense of blocking

- No other transactions can invalidate the data set through modifications
- Is this always what you want or need?
 - Queuing applications typically want locks, if someone is modifying that row – we want to go to another not see last transactional state
 - Very volatile prices – don't want to give them the last price if it's currently being updated, so wait... (but the update's fast)
- What about long running transactions or cases where you don't need absolute current...

Tips to Minimize Blocking

- Write efficient transactions – keep them short and in one batch
- Do not allow interaction in the midst of the batch
- Use indexes to help SQL Server find – and lock – only the necessary data
- Consider estimates for long running queries and/or migrating data to a secondary analysis server
- Problems with locking becoming blocking are likely when long running (and conflicting) transactions execute

SQL Server 2005 Isolation Options

- Default – Uses Locking, no options set
- Read Committed with Statement-level Snapshot
 - DB Option: **READ_COMMITTED_SNAPSHOT** enabled
 - Uses locking for writes, versioning for reads for statement-level consistency
- Transaction-level Snapshot Isolation
 - DB Option: **ALLOW_SNAPSHOT_ISOLATION** enabled
 - Uses default locking for everything, **ALLOW**s snapshot for transactions when requested by changing session isolation setting: **SET TRANSACTION ISOLATION LEVEL SNAPSHOT**
 - If requested, transaction-level consistency through row versioning
- Both Options Configured
 - Statement-level consistency through versioning automatically
 - Transaction-level consistency where requested

Read Committed Snapshot Isolation Database Changed to `READ_COMMITTED_SNAPSHOT`

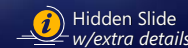
- No phenomena are possible in the bounds of a single statement
- In volatile databases, a multi-statement transaction may yield different results for different statements which access the same data
- Each statement is consistent but only for the execution of that statement, not for the life of the transaction (if the transaction has multiple statements)
- Each time data is read by a new statement the latest version is used

Snapshot Isolation Database Changed to `ALLOW_SNAPSHOT_ISOLATION`

- Setting `ALLOW` allows users to ask for Snapshot Isolation – NOT on by default
- ALL phenomena and ALL default locking behavior is EXACTLY the same unless you explicitly ask for Snapshot Isolation
- Once requests, no phenomena are possible in the bounds of a transaction running under snapshot isolation
- In volatile databases, a multi-statement transaction will always see the transactionally accurate version which existed when the transaction started
- Versions must stick around longer, multi-statement transactions may have conflicts

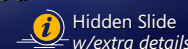
Isolation Levels: In Summary

- READ UNCOMMITTED (Level 0)
 - "Dirty Reads" – An option ONLY for readers
 - Any data (even that which is in-flight/locked) can be viewed
- READ COMMITTED (Level 1 – Default)
 - Only committed changes are visible
 - Data in an intermediate state cannot be accessed
- Read Committed w/statement-level Snapshot
 - Statement-level read consistency
 - New non-blocking, non-locking (ex. SCH_S), version-based Level 1



Isolation Levels: In Summary (cont'd)

- REPEATABLE READS (Level 2)
 - All reads are consistent for the life of a transaction
 - Shared locks are NOT released after the data is processed – does not allow writers (does allow other readers)
 - Does not protect entire set (phantoms may occur)
- SERIALIZABLE (Level 3)
 - All reads are consistent for the life of a transaction
 - Avoids phantoms – no new records
- Snapshot Isolation – 2005
 - Transaction-Level consistency using snapshot
 - New non-blocking, non-locking, version-based transactions



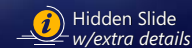
Controlling Isolation Levels

Table-level changes

From Clause, per table (no spaces)

- Level 0 – READUNCOMMITTED, NOLOCK
- Level 1 – READCOMMITTED (locking)
- Level 1 – READCOMMITTED (versioning)
 - Only in 2005 and *only* if the database option to READ_COMMITTED_SNAPSHOT is on
 - Can be overridden with READCOMMITTEDLOCK
- Level 2 – REPEATABLE READ
- Level 3 – SERIALIZABLE, HOLDLOCK

```
FROM dbo.titles WITH(READUNCOMMITTED)
JOIN dbo.publishers WITH(SERIALIZABLE)
```



Hidden Slide
w/extra details

Controlling Isolation Levels

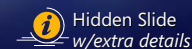
Session-level changes (cont'd)

Session level settings impact entire session but can be overridden with table-level settings

- Level 0 – READ UNCOMMITTED
- Level 1 – READ COMMITTED
- Level 2 – REPEATABLE READ
- Level 3 – SERIALIZABLE

SET TRANSACTION ISOLATION LEVEL opt

- READ UNCOMMITTED
- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE
- SNAPSHOT

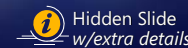


Hidden Slide
w/extra details

Allowing Read Committed using Statement-level Snapshot

- Database option

```
ALTER DATABASE <database_name>
SET READ_COMMITTED_SNAPSHOT ON
WITH ROLLBACK AFTER 5
```
- No other changes necessary...
- No changes to your queries or your applications – they automatically return with statement-level consistency
(NOTE: You may need to modify your applications if you depend on locking – re: queues, readers to wait for change)
- Changes to blocking...
- However, if this is NOT your performance problem (meaning concurrency isn't your bottleneck) then you may hinder performance not improve
- Expect this change in behavior at a cost



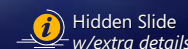
Hidden Slide
w/extra details

Allowing Snapshot Isolation

- Database option

```
ALTER DATABASE <database_name>
SET ALLOW_SNAPSHOT_ISOLATION ON
```
- Session setting

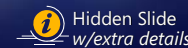
```
SET TRANSACTION ISOLATION LEVEL SNAPSHOT
```
- Changes to applications:
 - Request snapshot isolation
 - Test for conflict detection
- Expect this change in behavior at a higher cost



Hidden Slide
w/extra details

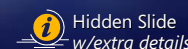
Potential Issues

- Cost in row overhead – when enabled, 14 bytes added to row
- If snapshot, do you depend on locking?
 - OK, most of you will say no but what about status queues...
 - Tip: Use READCOMMITTEDLOCK hint
- If Snapshot Isolation, could you have conflicts?
 - Be sure to have proper conflict detection and error handling, see Kim's Snapshot Isolation whitepaper for details and examples



Management/Monitoring

- Version Store in TempDB
- Versions removed when no longer needed
- Read committed using statement-level snapshot won't hold versions as long – in theory because only statement-level
- Snapshot isolation may have more impact on TempDB as versions held for life of transaction
- Lots of long running transactions may stress TempDB
- See Kim's Snapshot Isolation whitepaper for monitoring with DMVs (Dynamic Management Views)



Optimal Table Structures and Design

- Data Modeling
- Data Types
- The Data Row
- Table Design for Performance

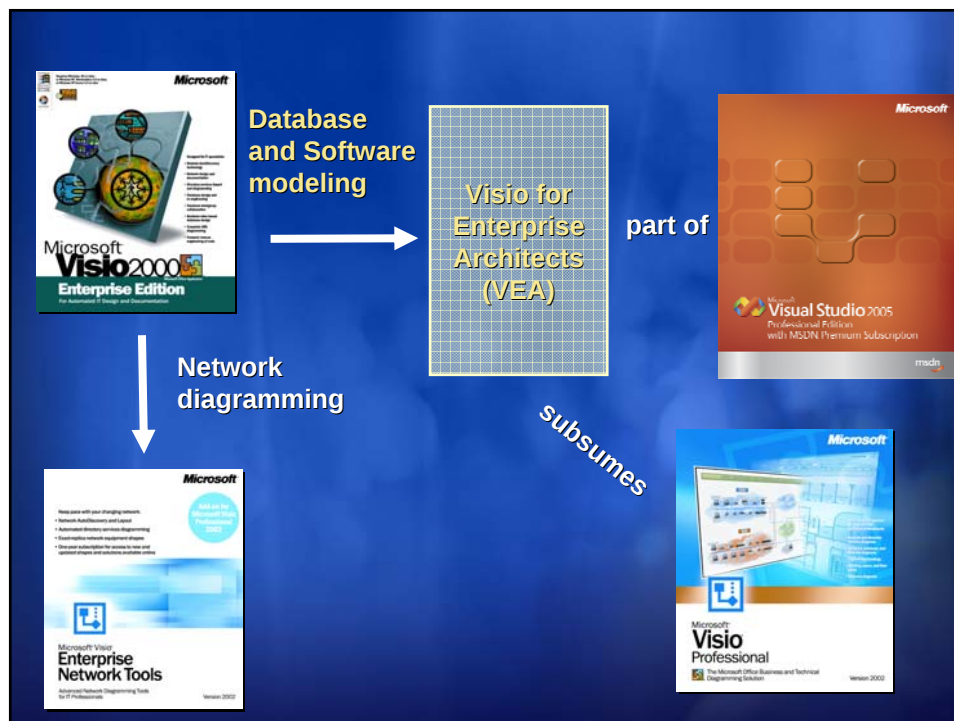
Data Modeling

- Data modeling is the process of planning the logical design of your database
- Use Data modeling to define the relationships between your data and processes



Data Modeling Tools

- Visual Studio .NET Enterprise Architect includes Visio for Enterprise Architects (VSEA)
- Contains great tools for data modeling
- Supports round-tripping between data model and database
- Future has not yet been announced



VSEA Database Design Tools

- Visio-based (VEA)
 - Conceptual data modeling (ORM)
 - Logical database modeling (Relational, IDEF1X, "ER")
 - Physical database modeling (SQL Server, Access, Oracle, DB2, etc.)
 - Forward and reverse engineering, sync, import/export, reports, etc.
- Non-Visio
 - Online physical database design tools
 - SQL query designer

Relational Model

- Introduced by E.F. Codd
- Based upon "relational algebra"
- Relations actually refer to what are more commonly referred to as tables

Remembering Dr. Codd

- Dr. E.F. Codd
 - August 23, 1923—April 18, 2003
 - Father of the Relational Model
 - Released the seminal paper A Relational Model of Data for Large Shared Data Banks in 1970
 - Received the A. M. Turing Award in 1981
 - Considered the highest honor in the computer science field



Normalization

- Normalization is the process of using formal methods to separate data to be stored in a database into multiple, related tables
- Normalization helps maintain the consistency and integrity of your data
- Done primarily for on-line transaction processing (OLTP) systems
 - In contrast, on-line analytical processing (OLAP) systems are often heavily denormalized

Entity Integrity

- Entities are single objects or concepts from the real world
- Entity integrity means that you must be able to uniquely identify each entity you store in a database
- Each entity in a table should have a primary key
 - This can be a single column or multiple columns
 - Multiple columns would be composite key

Domain Integrity

- Domain integrity restricts what can be stored in the database
- Domain integrity represents your business rules
 - SQL Server can help you enforce these rules
 - You can enforce them in your application
 - At multiple layers

Referential Integrity

- Referential integrity preserves the defined relationships between tables when records are entered or deleted
- Tables can have relationships based upon keys
 - Foreign keys and primary keys
 - Foreign keys and unique keys

User-Defined Integrity

- Applies when a business rule does not fit into one of the three other forms
- Implement using constraints, triggers, or stored procedures

Normal Forms

- Codd defined various "normal forms", or levels of normalization
 - See Appendix for details
- Generally you want to get to at least third normal form
- Introduction of native XML data type in SQL Server 2005 introduces design challenges

Denormalization

- The deliberate process of introducing redundant data into your database
- Generally done to increase application performance
- Often occurs when one database is both a TP and DSS source

Entities

- Basic principle behind a table – the way we describe some specific person, place or thing as a base object within the database
- Focus on a single element - Person, place, thing...
 - Authors – Each author is an Entity described by a row in the authors table.
 - Titles – Each title, separate from the author(s) that wrote the title is described by a row in the titles table.
 - Publishers – Each publisher, separate from any title(s) they may have published is described by a row in the publishers table.

Attributes

- Basic principle behind a column – the way we describe some specific element about the entity
- Each column of a table should describe ONLY that entity...
- Core elements of an attribute
 - Attribute Name price
 - Data type money
 - Nullability NULL

Every attribute should describe the primary key, the whole primary key, and nothing but the primary key.

Data Types

System Supplied Data Types

- Character
- Unicode Character
- Numeric
- Date and Time
- Exact Numeric
- Uniqueidentifier
- Binary
- Miscellaneous

Find the right data type for the job:

Use Unicode over ASCII

If the datatype varies:

< 5 chars should be nchar

5-20 chars – questionable

> 20 char – lean more towards
nvarchar

For numeric data:

Find the right range

Standardize on decimal or
numeric

Use uniqueidentifier sparingly

Understand precision and range

Data Type	Max/Exact Storage	Notes
char(n) varchar(n)	8000 bytes	1 byte per char - 8000 Characters
nchar(n) nvarchar(n)	8000 bytes	2 bytes per char - 4000 Characters
tinyint	1 byte	0-255
smallint	2 bytes	-32768 to 32767
int	4 bytes	-2 ³¹ to 2 ³¹ -1
bigint	8 bytes	-2 ⁶³ to 2 ⁶³ -1
uniqueidentifier	16 bytes	98E94963-F193-4E69-9262-7B692125557F
smalldatetime	4 bytes	Jan 1, 1900 with precision to the minute
datetime	8 bytes	Jan 1, 1753 with precision to a timetick (3.33 ms)
smallmoney	4 bytes	- 214,748.3648 through +214,748.3647
money	8 bytes	-922,337,203,685,477.5808 through +922,337,203,685,477.5807

 Hidden Slide
w/extra details

SQL Server 2005 Data Types

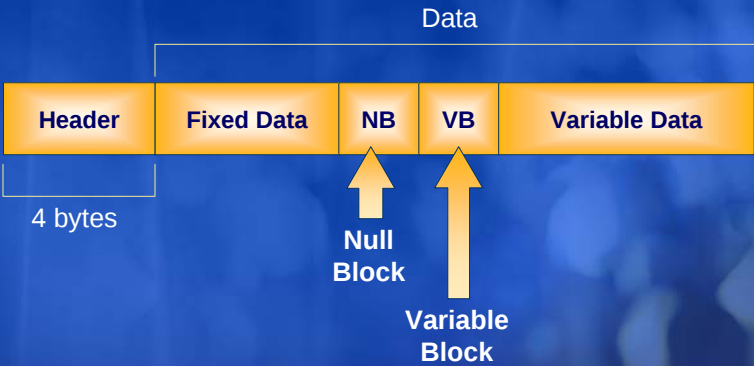
- Large Value Data Types, `varchar(max)`, `varchar(max)` and `varbinary(max)`
 - In row storage limit 8000 bytes (max col width)
 - Work like typical `varchar` but support LOB values (2^{31} bytes)
 - Default behavior is `IN_ROW` when entire LOB fits in the 8060 max `IN_ROW` storage area (NOTE: 8060 bytes is no longer the maximum row size)
- Be aware that LOB types (*text, ntext, image, varchar(max), nvarchar(max), varbinary(max), xml*) added to a table prevent online index rebuilds
- May want to consider vertical partitioning for LOB types

Where to store BLOBs?

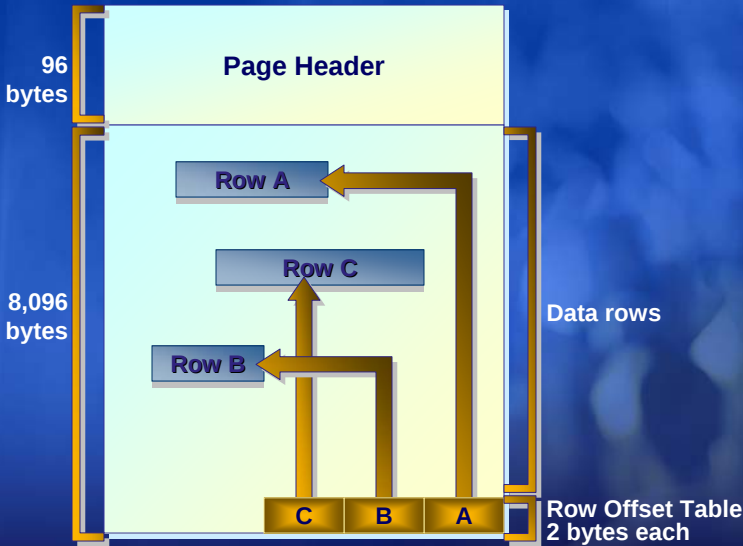
Each of these have pros/cons – which is *right*? (it depends)

- In the *core* base table
 - Prevents online index operations
 - Easily accessible
- In a separate table but in the same filegroup
 - Can't do isolated backup strategies (not desired?)
 - Must be "joined" or accessed when necessary
 - Doesn't put data into memory when not needed (vertical partitioning uses resources more effectively)
- In a separate table and in its own filegroup
 - Allows isolated backup strategies
- Outside of the database
 - Not secured/controlled with database
 - May allow easier access to other applications

The Data Row



Rows on a Page

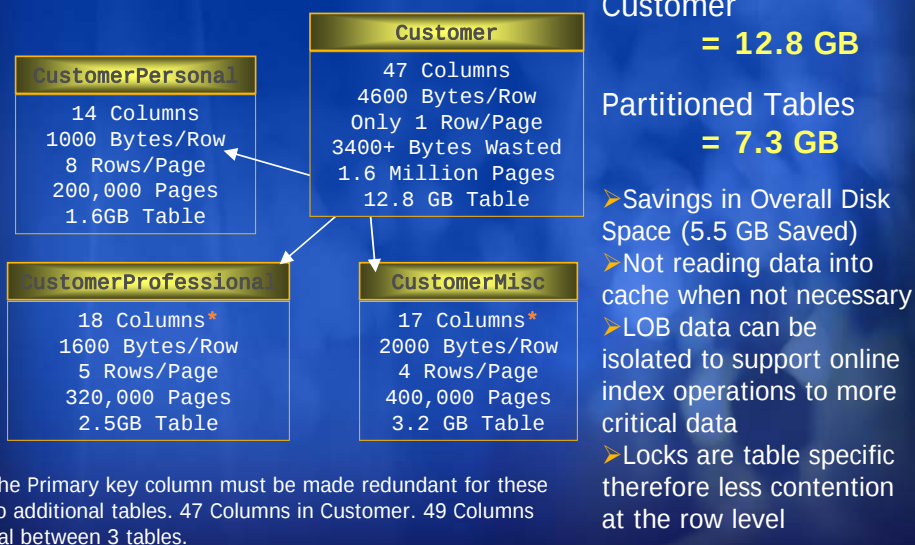


Optimal Row Width

- Consider Table Usage above all else
- Estimate Average Row Length
 - Overhead
 - Fixed Width Columns
 - Estimate Average from realistic sample data
`SELECT avg(datalength(columnname)) FROM tname`
- Calculate Rows/Page

$$\frac{8096 \text{ Bytes/Page}}{??? \text{ Bytes/Row}} = \text{Rows/Page}$$
- Calculate Wasted Bytes – on Disk $\therefore$ in Memory

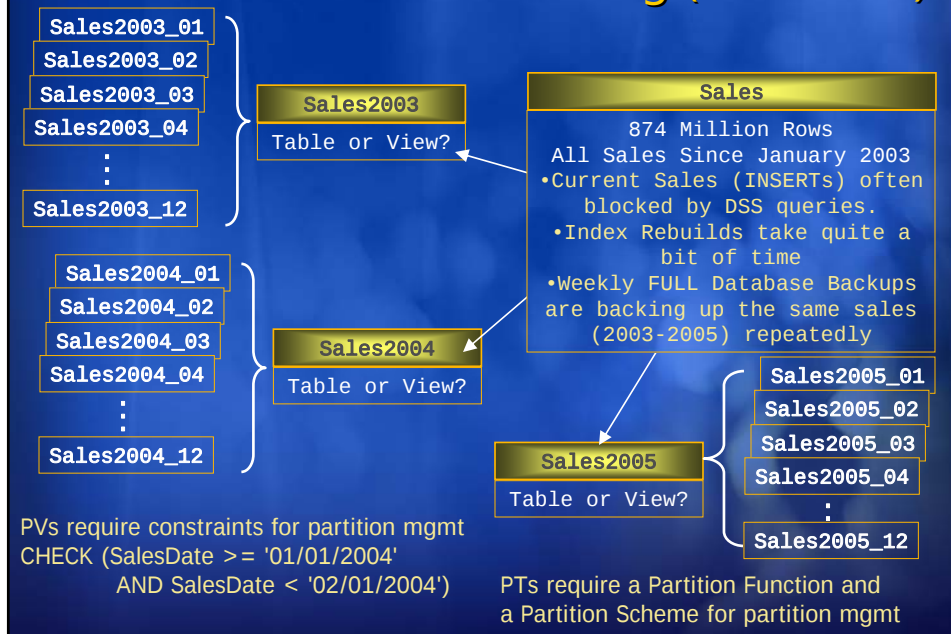
Consider Customer Table with 1,600,000 Rows

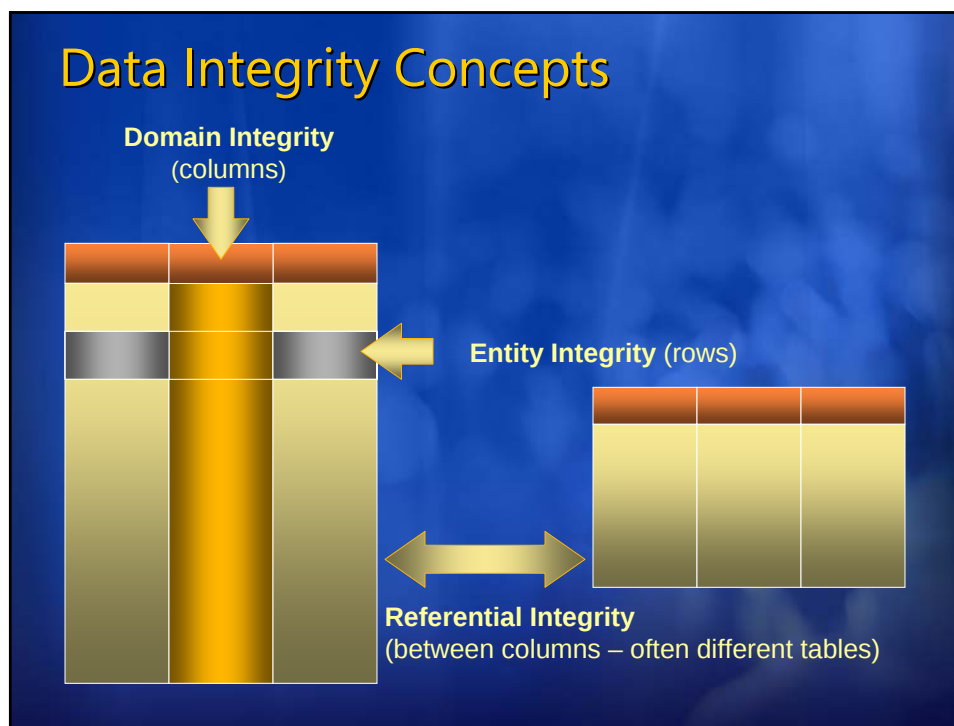


Vertical Partitioning

- Optimizing row size for...
 - Caching
 - More rows on a page, more rows in memory
 - Locking
 - Only locking the columns that are of interest. Minimize row based conflicts
- Usage Defines Vertical Partitions
 - Logically Group Columns to minimize joins
 - Consider Read Only vs. OLTP Columns
 - Consider Columns often used together

Horizontal Partitioning (PVs or PTs)





Enforcing Data Integrity

- Declarative Data Integrity
 - Criteria defined in object definitions
 - SQL Server enforces automatically
 - Implement by using constraints, defaults, and rules
- Procedural Data Integrity
 - Criteria defined in script
 - Script enforces
 - Implement by using triggers and stored procedures
- Why Procedural over Declarative?

The big three: Custom Error Messages, Cross-database RI and Cascading (prior to SQL Server 2000) NOT TO MENTION limitless functionality – you write the code!

Foreign Key References

- Foreign Key enforces domain and referential integrity
 - Domain – restricts set of legal values to set in referenced table/column(s)
 - Referential – enforces relationship between the tables (FK must reference PK or UK)
 - INSERTs/UPDATEs in the referencing table must supply a value which exists
 - DELETEs in the referenced table are not allowed when row is referenced
- Foreign Key reference can cascade
 - ON UPDATE – clean, easy, consistent
 - ON DELETE – Too easy, beware!

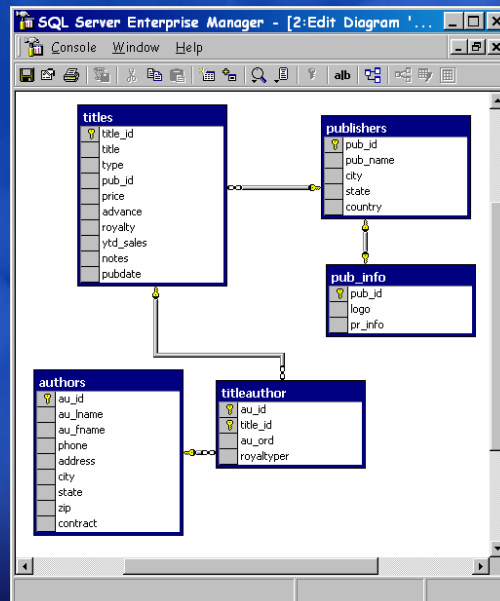
Foreign Key References

- Inserting a row – in the referencing table
 - Must lookup row in referenced table
 - Must reference a primary key or a unique key
 - An index must exist, lookup is fast
- Delete a row – in the referenced table
 - Must determine if any rows reference the key
 - The referencing column (the column on which the foreign key exists) does not have an index (unless manually created)
- Locks must be used to validate the integrity of the relationship

ERD – Entity Relationship Diagram

Relationships

- Titles have a Publisher
- Titles have potentially many Authors
- Authors may have written more than one Title
- For a given title an Author is given an order and a royalty

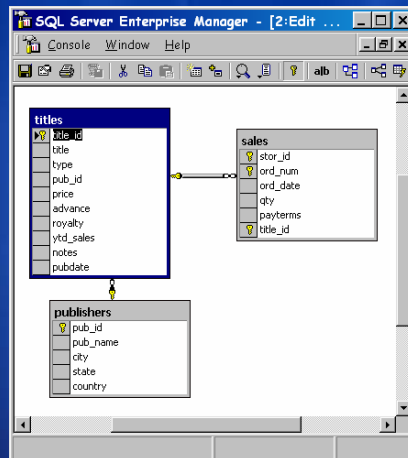


Describing Relationships

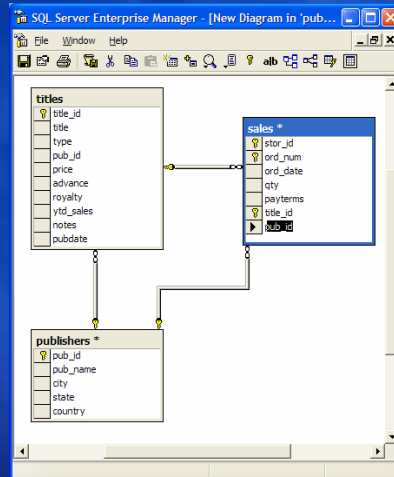
- Mathematically – One to Many (1-n) Relationship
 - For Every Sale there is Only one Title
 - For Each Title there can be Many Sales
The Relationship between Sales and Titles is One to Many
 - For Every Title there is Only one publisher
 - For Each Publisher there can be Many Titles
The Relationship between Titles and Publishers is One to Many
- Directionally
 - Sales to Titles = One Sale to One Title
 - Titles to Publishers = One Title to One Publisher
- Can we describe Sales to Publishers?

Duplicating Keys

BEFORE adding redundant pub_id

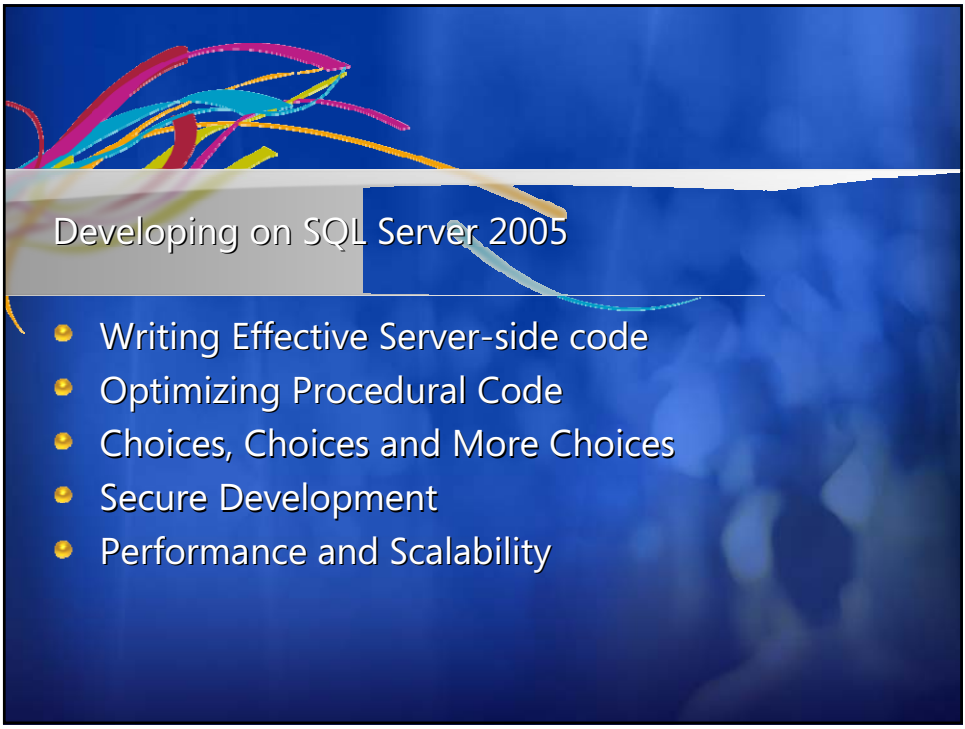


AFTER adding redundant pub_id



Design Techniques Summary

- Rows CAN span pages however, record size impacts disk space, cache utilization, concurrency and maintenance restrictions
- The lowest granularity of locking holds an entire row – partitioning a table vertically or horizontally can improve concurrency, minimize restrictions on online rebuilds and improve cache utilization
- Optimizer is aware of constraints – efficient horizontal partitioning requires constraints
- Duplicating Keys which define relationships can improve join performance by giving the optimizer more options for performing a join



Developing on SQL Server 2005

- Writing Effective Server-side code
- Optimizing Procedural Code
- Choices, Choices and More Choices
- Secure Development
- Performance and Scalability

Batches

- Parsed as a unit – if any statement fails syntax then NO statements (in that batch) are executed. SQL Server proceeds to next batch.
- At execution – if a statement fails then that statement is rolled back and SQL Server exits the batch (or falls into catch block) and proceeds to the next batch. Any statement following the failed statement is NOT executed.

```
SELECT * FROM pubs.dbo.authors
SELECT * FROM pubs.dbo.authors
SELECT * FORM pubs.dbo.authors
go
```

```
Error: Server: Msg 170, Level 15, State 1, Line 3
Line 3: Incorrect syntax near 'form'.
```

Transactions

- Controlled by Transaction Mode of session
 - Defined explicitly
 - Defined at connection level
 - Defined at server level
- Allows multiple statements to be treated as a single logical unit of work
- Handled automatically across databases on the same server (i.e. instance)
- Handled through MSDTC explicitly through
 - BEGIN DISTRIBUTED TRANSACTION
 - COMMIT TRANSACTION
- Requires error handling logic

Transaction Mode

- Autocommit transaction (default)
 - Statement level implicit transaction
 - Each statement commits as a single unit
- Explicit transaction (user-defined)
 - BEGIN TRANSACTION
 - COMMIT TRANSACTION
- Implicit transaction
 - Session Level Setting
 - SET IMPLICIT_TRANSACTIONS ON
- Batch scoped transactions (2005)
 - Only when MARS enabled on client

Transaction Termination (1 of 2)

- Resource Error – Automatically Handled
 - If the statement fails due to a SQL Server resource error (for example, the transaction log fills), SQL Server maintains data integrity automatically.
- User-defined Error – Programmatically Handled
 - Conditionally, when your business rules (i.e. constraint violation or a lock timeout (more on this coming up)) are violated, YOU must programmatically determine the fate of the transaction.

Transaction Termination (2 of 2)

- SET ARITHABORT
(default based on API and driver)
 - ON – rollback statement/transaction when divide-by-zero or arithmetic overflow error occurs
 - OFF – return NULL
- SET XACT_ABORT
(off by default)
 - ON – rollback statement/transaction when run-time error occurs

Understanding Transactions

- BEGIN TRANSACTION
 - Begins a New Transaction
 - Increments @@TRANCOUNT
 - Supports special "marked transactions"
BEGIN TRANSACTION TransactionName
WITH MARK 'important transaction'
- COMMIT TRANSACTION
 - Decrements @@TRANCOUNT by 1
 - Does NOT commit the transaction UNLESS the @@TRANCOUNT is 0

Understanding Transactions

- ROLLBACK TRANSACTION
 - Returns @@TRANCOUNT to 0
 - Cancels all levels of Nested Transactions – even when using ROLLBACK TRANSACTION to a named (i.e. marked transaction).
- SAVE TRANSACTION (a.k.a. Savepoints)
 - Do not affect @@TRANCOUNT
 - When you ROLLBACK to a savepoint you must still "commit" or "rollback" the pending transaction

Nested Transactions

- Issues in Nesting Transactions
 - Use @@trancount to determine nesting level
 - Transactions aren't truly "nested" (even when named), rollback always rolls back to OUTER most transaction (@@trancount = 0)
 - Rollback to a savepoint does NOT decrement @@trancount – must also commit nested transaction to decrement @@trancount
- Handle errors
 - Try/Catch logic
 - Using session variables

System Functions to help handle errors:
 @@ERROR
 @@TRANCOUNT
 @@ROWCOUNT

Handling Errors – The Old Way

- After each statement you need to test the state of @@ERROR
- To centralize error checking code, use labels with GOTO and RETURN statements
- Errors such as data type conversion errors causes batch to terminate; not every error can be trapped
- Makes for a lot of additional code

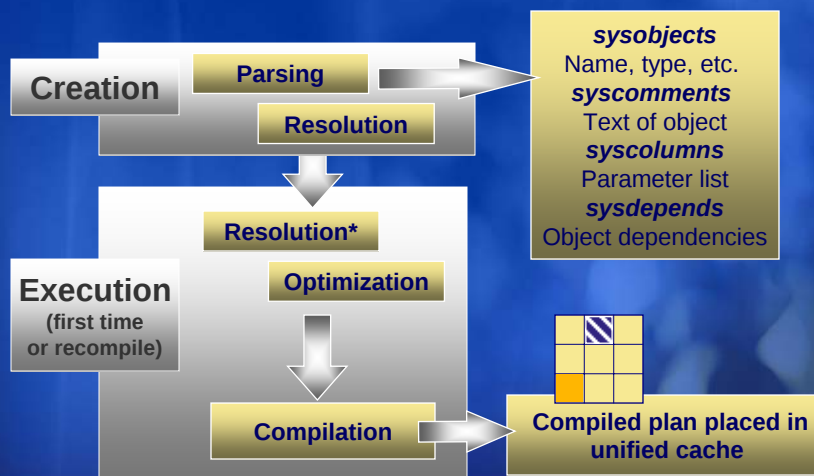
```
IF @@ERROR <> 0
  BEGIN
    RAISERROR...
    ROLLBACK
    RETURN
  END
```

Handling Errors – The New Way

- SQL Server 2005 adds exception handling
 - Error handling in previous SQL Server versions tedious
 - @@ERROR set on each statement
 - set variable with @@ERROR, then check value
 - BEGIN-END TRY BEGIN-END CATCH blocks in SQL 2005
 - semantic equivalent of BEGIN-END blocks
 - additional functions return error info
 - can query transaction status
 - can save @@ERROR equivalent

**See TRY/CATCH in DAT424 on Wednesday – 14:00-15:15, by Bob Beauchemin.*

Processing of Stored Procedures



Compilation/Optimization

- Based on parameters supplied
- Future executions will reuse the plan
- Complete optimization of all code passed (more on this coming up...statement-based recompilation and/or modular code!)
- Poor coding practices can cause excessive locking/blocking
- Excessive recompilations can cause poor performance

Recompilation Issues

RECOMPILATION = OPTIMIZATION
OPTIMIZATION = RECOMPILATION

- When do you want to recompile?
- What options do you have Recompilation?
- How do you know you need to recompile?
- Do you want to recompile the entire procedure or only part of it?
- Can you test it?

When to recompile?

- When the plan for a given statement within a procedure is not consistent in execution plan—due to parameter changes
- Cost of recompilation might be significantly less than the execution cost of a bad plan!
- Why?
 - Faster Execution with a better plan
 - Saving plans for reuse is NOT always beneficial
 - Some plans should NEVER be saved
- Do you want to do this for every procedure?
 - No, but start with the highest priority/expensive procedures first!

CREATE ... WITH RECOMPILE

- When procedure returns widely varying results
- When the plan is not consistent
- For SMALL procedures
- For backward compatibility
- Why?
 - 2000: Only complete procedure recompiles
 - 2005: Statement level recompilation
- Always target the smallest amount possible to recompile!

EXECUTE WITH RECOMPILE

- Excellent for Testing
- Verify plans for a variety of test cases
 - EXEC dbo.GetMemberInfo 'Tripp' WITH RECOMPILE
 - EXEC dbo.GetMemberInfo 'T%' WITH RECOMPILE
 - EXEC dbo.GetMemberInfo '%T%' WITH RECOMPILE
- Do the execution plans match?
- Are they consistent?
- Yes ⇒ then create the procedure normally
- No ⇒ Determine what should be recompiled

Statement-level Recompilation

- The old way: Modularizing your code
 - Doesn't hurt!
 - May allow better reuse of code "snippets"
- The new way: "inline recompilation"
 - OPTION(RECOMPILE)
 - Excellent when parameters cause the execution plan to widely vary
 - Bad because EVERY execution will recompile
 - OPTIMIZE FOR (@variable_name = literal_constant, ...)
 - Excellent when large majority of executions generate the same optimization time
 - You don't care that the minority may run slower with a less than optimal plan?

Modular Code – Still works!

```
IF (expression operator exp)
    SQL Statement Block1
ELSE
    SQL Statement Block2
```

Solution?

Do not use a lot of
conditional SQL
Statement Blocks
Call separate stored
procedures instead!

Scenario 1 – upon first execution...

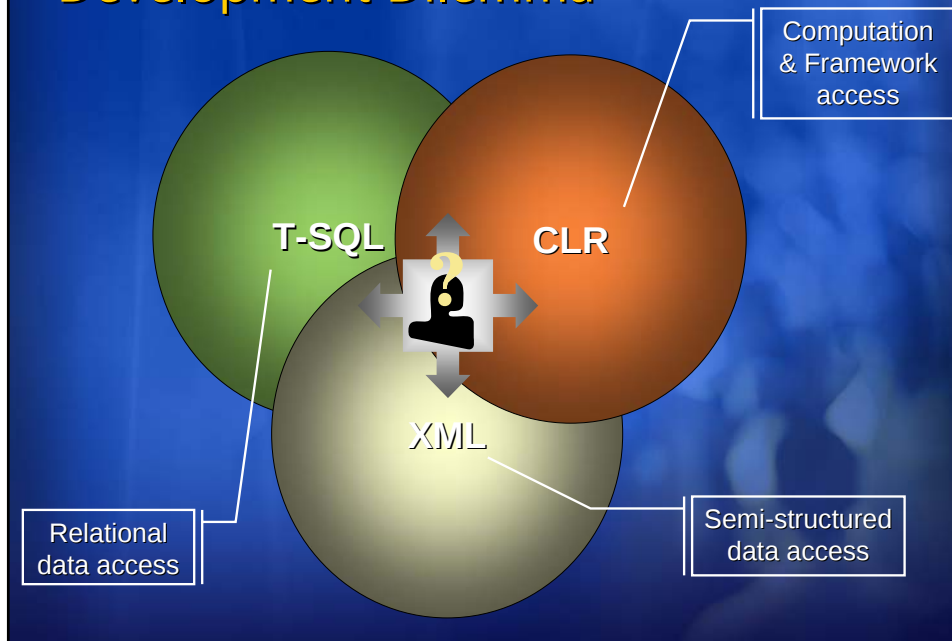
Parameters are passed such that the ELSE condition executes – BOTH Block1 and Block2 are optimized with the input parameters

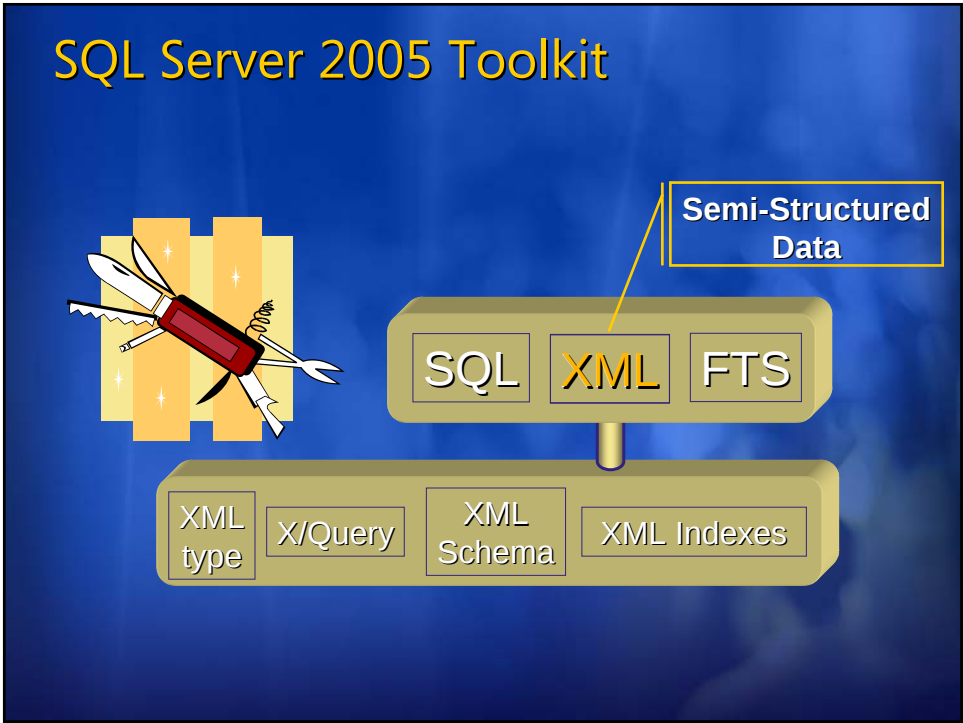
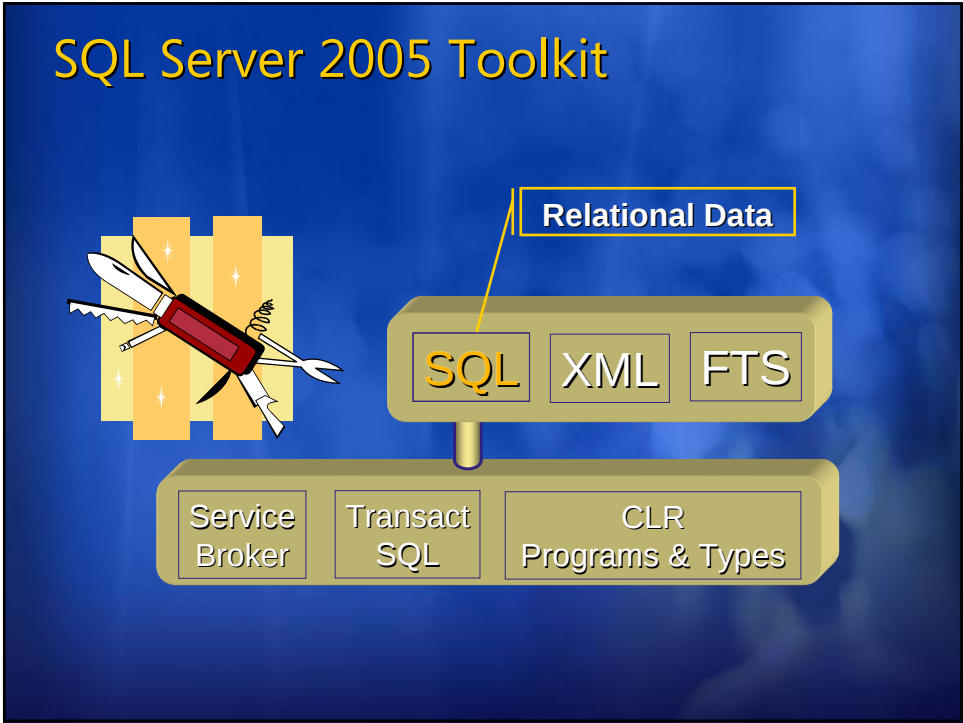
Scenario 2 – upon first execution...

Parameters are passed such that the IF condition executes – ONLY Block1 is optimized. Block2 will be optimized when a parameter which forces the ELSE condition is passed.

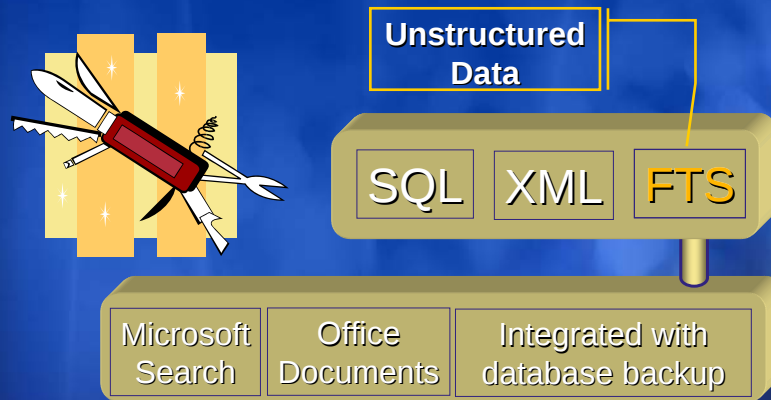
See ModularProcedures.sql

Development Dilemma





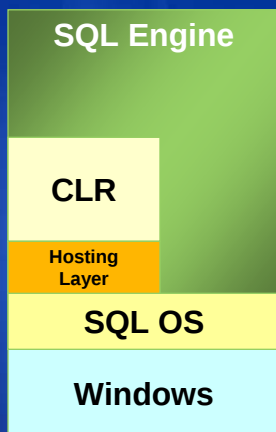
SQL Server 2005 Toolkit



CLR Integration Design Goal

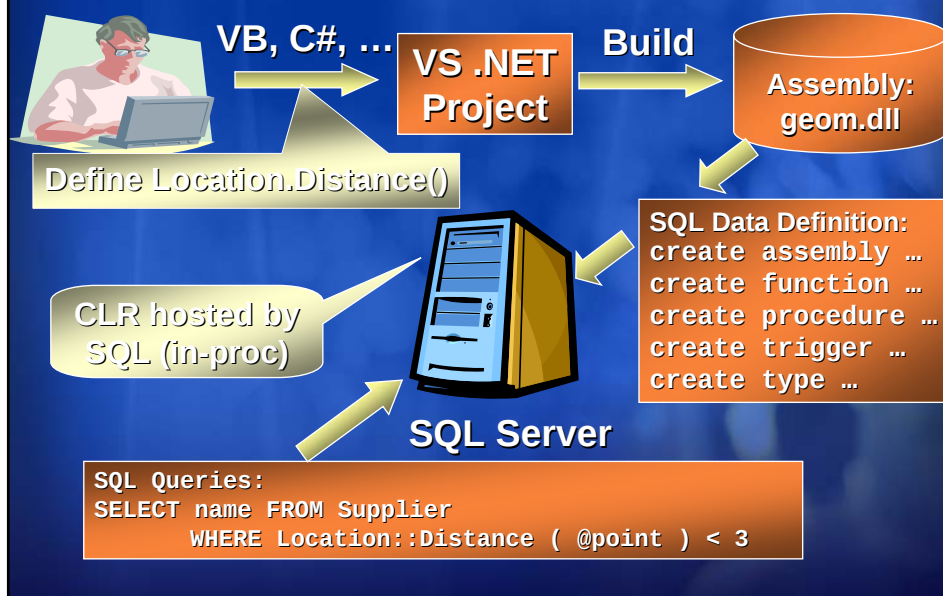
- Bring framework programming model into the database tier
- Allow business logic to easily migrate to the most appropriate tier
- Enable safe database extensions
- Result: stored procedures, triggers, data types defined in managed code

CLR Integration



- CLR Deeply Integrated
- Hosting layer provides coordination of:
 - Threads
 - Memory
 - Garbage Collection
- Allows System Extension
 - Scalar Functions
 - Table Value Functions
 - Types
 - Aggregates
 - Procedures
 - Data Access

SQL CLR Functionality



VS Integration

- Visual Studio supports a new SQL Server 2005 Project type
 - Easy to develop, debug and deploy .NET code for the database
- Integrated debugging experience across mid-tier and database tier
 - Seamlessly step cross-language – TSQL and .NET
 - Set breakpoints anywhere, inspect anything
- Increased Productivity

Good Scenario for CLR Usage

- Data validation & network traffic reduction
- Writing general purpose functions:
 - Data passed as arguments
 - Little/no additional data access
 - Complex computation applied on a row by row basis to the data
- Scalar types & custom aggregations
- Leveraging the power of the .NET Framework
 - Access to a rich set of pre-built functionality
- Replacing Extended Stored Procedures (XP)
 - The CLR is safer:
 - No access violations making SQL Server crash
 - No leaks making SQL Server slow down & crash
 - Better performance & scalability (managed memory model)
 - No security issues...



Bad Scenario for CLR Usage



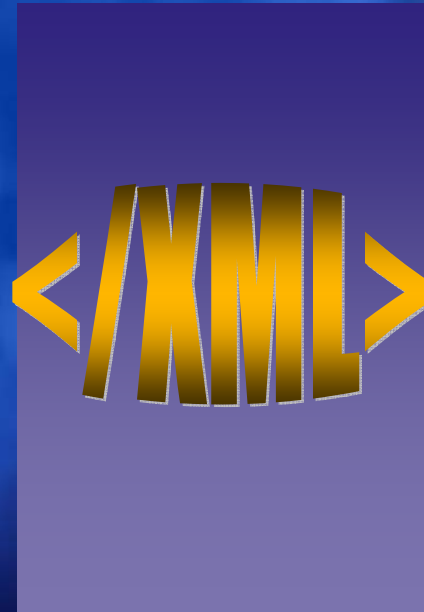
- Heavy data access – Transact-SQL set based access will be faster
 - Don't write SELECT statements as CLR procedures!
- Complex types
 - 8K size limitation
 - All data is read/re-written when updated
- Pre-Aggregation for Reports
 - CLR Aggregates cannot be used in Indexed Views
- Your application must support previous versions of SQL Server
- Technology for technology's sake...

XML And Your database

- Provides for more efficient storage and access to semi-structured data
- Provides for better integration in Service Orientated Architectures
- Provides malleable data that can be consumed and transformed by clients and other services
- Complements, but does not replace a good relational schema

XML Development

- Native XML datatype
 - Optimized on-disk structure (not "just a blob")
- XML Indexes
 - Entities & Attributes
- XQuery
 - Integrated with core relational operators
- Schema Support
 - Strongly typed XML
- CLR/XML Integration
 - XMLTextReader
 - Access to XSLT
- Full Text Search
- And more ...



XML Indexes

- Primary Index:
 - Requires a clustered primary key on the base table
 - B-Tree of element and attribute names, node values, and node types, retains document order and structure, as well as the path from the root of the XML instance to each node for efficient evaluation of path expressions
- Secondary Indexes off of the primary XML index
 - PATH → columns (path, value)
 - PROPERTY → columns (PK, path, value)
 - VALUE → columns (value, path)

Scenario for XML Development



Good Scenario:

- Data is semi-structured, small core of fixed data with many, sparsely populated extended attributes
 - Multi-value Property bags
 - Complex Property bags
 - "WordXML"
- Documents are large but rarely updated
 - Indexing will pay off
- Data is hierarchical
 - regular expressions are well suited for finding data



Bad Scenario:

- "Database in a Cell"
- Documents are large and updated frequently
- Document update contention is likely
- Data is fully structured & populated → candidate for conversion to relational schema
- Data contains large binary objects (2GB limitation)

Transact-SQL Enhancements

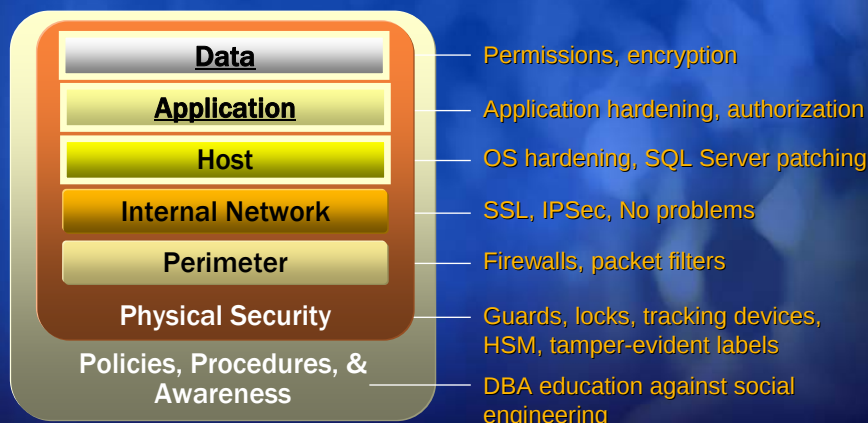
- ROW_NUMBER
- RANK, DENSE_RANK
- Common Table Expressions
- PIVOT/UNPIVOT
- CROSS APPLY and OUTER APPLY
- Error handling – TRY/CATCH
- DDL Triggers (synchronous)
- Event Notifications (asynchronous)
- Parameterized TOP

Whose Fault was SQL Injection?

- OK. We got your attention. It was not you anyway, of course.
- Security for a database developer was a very rude wake-up call.
- We *feel* your pain.
- We can help you – in this session!

Defense in Depth

- Using a layered approach:
 - Increases an attacker's risk of detection
 - Reduces an attacker's probability of success



Application Protection

- *Protection against attacks against the application*
- Techniques and practices used by the database code (modules, stored procedures) to avoid being exploited by callers
 - Permissions and ACLs on the data are also part of application protection from the perspective of the calling application

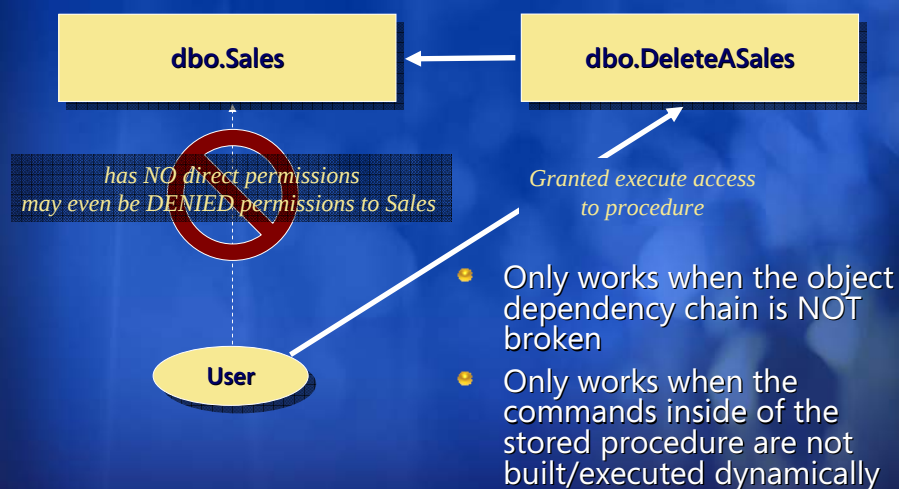
Secure Development

- How things used to work?
- User-schema separation
- Granular permissions
- EXECUTE AS
- SQL Injection (Classic)
- SQL Injection (Future)

Security through Encapsulation

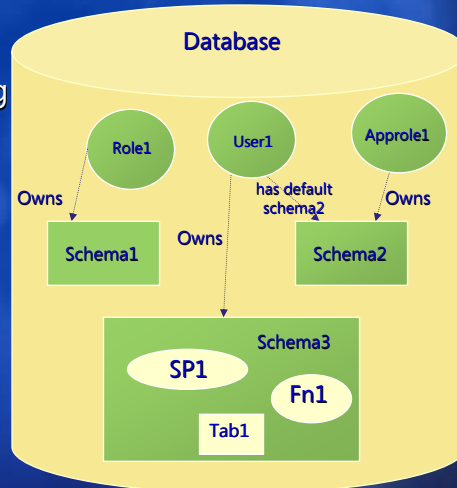
- Granting access to a process or a method without direct access to base objects
- How?
 - Grant access to the stored procedures, views and/or functions without granting access to the base object
- Requires
 - The objects in the dependency chain cannot be broken (object ownership chaining)
 - The user has to have direct base object access (which is what you're trying to avoid)

Do NOT Give Base Object Access The SQL Server 2000 Way



User-Schema Separation

- Database can contain multiple schemas
- Each schema has an owning principal – user or role
- Each user has a default schema for name resolution
- Database objects live in schemas
- Object creation inside schema requires CREATE permission and ALTER or CONTROL permission on the schema
- Ownership chaining still based on owners not schemas



Default Schema

- Used for name resolution purposes
- Common name resolution across multiple users
- No need to rely on DBO schema
- Using DBO schema may result in security issues
 - Object Creation requires higher privileges
 - Mitigates concerns resulting from ownership chaining
- Instead create "buckets" of objects through "schemas" where schemas have owners and developers have default schemas and/or control on needed schemas

Do NOT Give Base Object Access

The SQL Server 2005 Way

- Create a schema – for example one which contains procedures (and possibly even base tables) and then GRANT EXECUTE at the schema level
- Add objects to appropriate schema
- Grant access to the schema or the individual objects – if chaining is required, it is still based on the owner... the owner of the schema

What if Dynamic String Execution

- By default – and for better security – if the stored procedure has a statement which is built dynamically (using EXEC('string') or EXEC(@variable)) then the context under which the dynamically constructed string executes is ALWAYS the caller
- Which is what helps to prevent some forms of SQL Injection
- This is really a **good** thing BUT...
- Can be limiting
 - Enter: *EXECUTE AS*

Module Execution Context

- Execute AS CALLER
 - Default behavior, same as SQL Server 2000
 - Use when caller's permission needs to be checked – or when ownership chaining will suffice
- Execute AS 'UserName'
 - Statements execute as the username specified
 - Impersonate permission required on user specified
- Execute AS OWNER
 - Statements execute as the current owner of the module
 - Impersonate privileges on owner required, at setting time
 - On ownership change, context is new owner
- Execute AS SELF
 - Statements execute as the person specifying the execute as clause for the module – even if the ownership changes
 - May be useful in application scenarios where calling context may change

EXECUTE AS...

- Solves problems
 - Allows permissions to be granted where never possible (e.g. granting truncate table)
 - Wrap ANYTHING inside a stored procedure and set the context to run as someone who has permissions – even dynamic string execution – then give execute permission
- Creates potential for further SQL Injection
 - What if you're code is not well tested and uses dynamically executed strings

SQL Injection

- SQL statement(s) or clause(s) injected into an existing SQL command
- Injected string appended to application input
 - Text boxes
 - Query strings
 - Manipulated values in HTML
- Why SQL injection works?
 - Application accepts arbitrary user input
 - Connection made in context of higher privileged account

SQL Injection Mitigation

- Do not trust the user's input!
 - Look for valid input and reject everything else
 - Protect identifiers with QUOTENAME()
 - Regular expressions are your friend!
- Do not use string concatenation
 - Use parameterized queries to build queries
- Restrict information in error messages
- Use a low-privileged account
 - DO NOT use sa or sysadmin role member

Data Protection

- Use of cryptography to protect the data layer from theft or manipulation
 - Particularly important if offline data is in transit
 - Important for regulatory reasons
 - Prevent admin from access
 - Sarbanes Oxley
- Manipulation (integrity) protection uses digital signatures
 - Not implemented for data in SQL Server 2005
 - Can overcome this with judicious use of encryption alone
 - Implemented for code signing

Check Your OS

- Verify what algorithms and key lengths are supported by your OS, as this depends on your CSP (Cryptographic Services Provider)
 sys.symmetric_keys
- For example, we found the following during testing – this may change in the future
- At present, there is no way to change the CSP...

	XP SP2	WS2003
DES	56 (64)	56 (64)
3DES	128	128
AES128	-	128
AES192	-	192
AES256	-	256
RC2	128	128
RC4	40	40

Before You Begin

- Your DBA must ensure that the server has a Service Master Key (SMK) that agrees with your disaster recovery strategy and a Database Master Key (DMK) has been created for each database that will use encryption
 - There are serious implications on security (use strong password for SQL service account) and for disaster recovery
 - ALTER SERVICE MASTER KEY can be used to change recovery options
 - CREATE MASTER KEY ENCRYPTION BY PASSWORD =
 - Use a strong password, write it down, keep it safe

Key Generation

- The key should be impossible to guess. Preferably random.
 - CREATE SYMMETRIC... will generate a fairly random key for you – good!
 - You can base the key on data supplied by you, use KEY_SOURCE* clause – good for generating identical keys from a high-quality password
 - Note: KEY_SOURCE may be renamed to DERIVED_FROM in the released version

Passwords

- Make sure passwords used to protect or create keys are very strong
- In our opinion, it is better to create a very long and complex password that you will have to write down and store in a well-protected safe in your company
 - You can divide it into two halves and store in separate envelopes in different safes
- E.g.: *87(Hyfd1kRM?_764#{ (**%GRtj*(NS£" _+^\$(
 - No dictionary words, more than 20 characters, many non-printing characters
 - Challenge: usability!
- Consider also:
 - Not keeping passwords as text in your code, but store and retrieve them through DPAPI and a .NET component
 - Using good quality password generators

Key Protection

- SQL Server 2005 insists that the key you create is further encrypted for its protection
 - Yes, that's double-encryption, but that is not double-security. Actually, it reduces security a little in some cases
- CREATE SYMMETRIC...ENCRYPT BY
 - PASSWORD
 - Your (v. good) password generates a key to 3DES encrypt the key you are protecting
 - Note, 3DES is less secure than AES, so this
 - CERTIFICATE
 - Your key is encrypted using the public key of a certificate
 - This, in essence, is hybrid encryption
 - If private key is kept secure (and offline), this is a very good way to protect a symmetric key
 - Or another SYMMETRIC or ASYMMETRIC key – less useful but interesting

Encryption

1. Create or retrieve the key
2. Open the key – this means decrypt it with its (secondary) password or certificate or other key
3. Use function ENCRYPTBYKEY inside DML

Decryption

1. Create or retrieve the key
2. Open the key
3. Use function DECRYPTBYKEY inside SELECT and all other DML

Performance Issues

- As you know, security is expensive
- *Asymmetric* encryption alone is not viable for large amounts of data
- *Symmetric* cryptography can be both strong and fast: AES
- *Hybrid* encryption is fast enough and solves the problem of symmetric key distribution logistics
- Don't encrypt everything, this rarely makes sense and would be very costly – and does not allow indexing (re: salting the data)
- But, our (suspect) performance tests showed that all algorithms were as fast as each other. This is not what we expected, and either signifies other overheads which cancel out the differences...

Secure Connections

- SQL Server 2005 can use SSL to encrypt the channel between clients and the server
- Encryption is "auto-on" for initial authentication
- Must be turned on (ForceEncryption=Yes) to ensure secure communication
- Security must be weighed against performance

Top 10 Best Practices in Security

1. Risk assessment and threat analysis
2. Trap "developer quality" error messages
3. Avoid using dynamic string execution
4. Think encryption for sensitive data
5. Do not create objects in DBO schema
6. Hide underlying application schema
7. Use Roles for managing permissions
8. Account for catalog security
9. Avoid encryption as it precludes indexing and performance
10. Know your CSP. Check allowed key lengths if needed. Chose carefully!

Performance and Scalability

- Using Indexes
- Index Concepts
- Table and Index Internals
- Initial Table Design
 - The Clustered Index Debate
- Secondary Non-clustered Indexes
- Letting ITW/DTA Help!

Index Concepts – Book Analogy

- Think of a book – with indexes in the back
- The book has one form of logical ordering
- For references – you use the indexes in the back... to find the data in which you are interested you look up the key
- When you find the key – you must lookup the data based on its location... i.e. a "bookmark" lookup
- The bookmark always depends on the (book) content order

Index – Species Common Name

Index – Animal by Type, Name
Bird, Mammal, Reptile, etc...

Index – Animal by
Country, Name



Index – Animals by Habitat, Name
Air, Land, Water

Index – Species Scientific Name

Index – Animal by
Continent, Country, Name

Index Concepts – Tree Analogy

- If a tree were data and you were looking for leaves with a certain property, you would have two options to find that data....
 1. Touch every leaf – interrogating each one to determine if they held that property...SCAN
 2. If those leaves (which had that property) were grouped such that you could start at the root, move to the branch and then directly to those leaves...SEEK

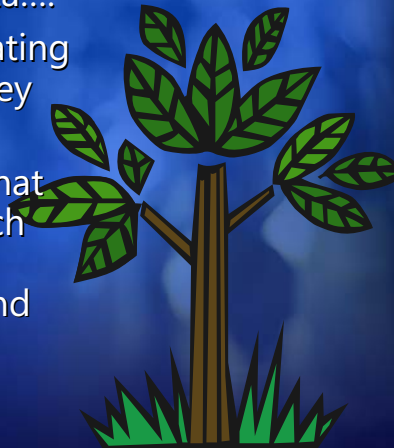


Table Structure – Heap

- Table without a Clustered Index
- Records are NOT ORDERED
- No Doubly-Linked List
- Access via HoBT Map – Heap or B-Tree Map
 - HoBT = 8K Page (Chain) which Tracks Object Usage
 - 1 per Table PARTITION (Heap or B-tree)
 - For each Partition 1 per type (Data, LOB, Row-overflow), per File, per 4 GB
- If NO Indexes exist – a full Table Scan required.
At least 4000 I/Os on the Employee Table Heap.

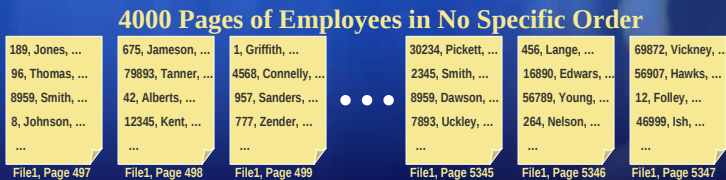
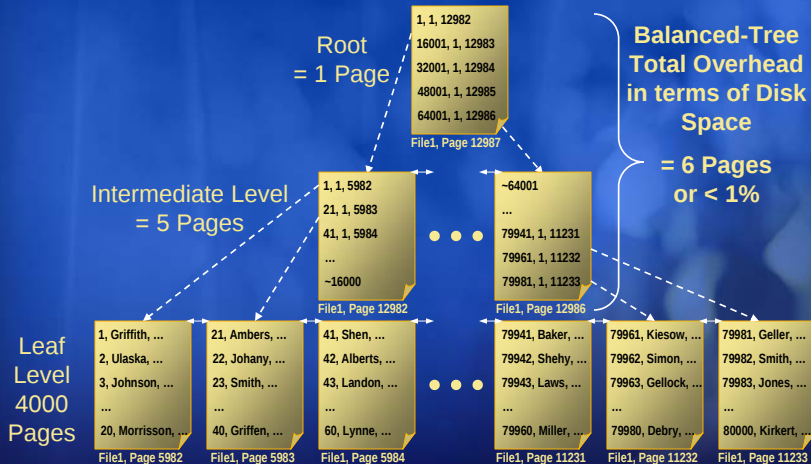


Table Structure – Clustered Table

- Two components – B-tree/Leaf Level
- Ordered Structure – maintained logically after creation



Clustered Index Overview

- Not Required – Although Highly Recommended
- Only One Per Table
- Physical Order Applied at Creation
- Logical Order Maintained through a Doubly-Linked List
- Requires ongoing and automated maintenance
- Need to choose WISELY!

Clustered Index Criteria

- Unique
 - Yes – No overhead, data takes care of this criteria
 - NO – SQL Server must “uniquify” the rows on INSERT. This costs time and space. Each duplicate has a 4-byte “uniquifier.”
- Narrow
 - Yes – Keeps the NC indexes narrow
 - NO – Possibly wastes space
- Static
 - Yes – Improves Performance
 - NO – Costly to maintain during updates to the key

In fact, a column that's ever increasing (identity) is often ideal...

Clustering Key Choices

- Identity column
- Order Date, identity
 - Not date alone as not unique
- GUID
 - Populated by client-side call to .NET client guid generator – NOT as the CL key but may be the Primary Key
 - Populated by server-side newid() function
 - No pattern
 - Populated by NEW server-side newsequentialid() function
 - Creates an ever-increasing pattern
- Key point: As narrow as possible

Finding the Right Balance

- Determine Primary Usage of Table – OLTP vs. OLAP vs. Combo?
 - This determines Clustered Index
- Create Constraints – Primary Key and Alternate/Candidate Keys
- Manually Add Indexes to Foreign Key Constraints
- Capture a Workload(s) and analyze with Database Tuning Advisor (2005)
- Add additional indexes to help improve SARGs, Joins, Aggregations
- Are you done?

NO!

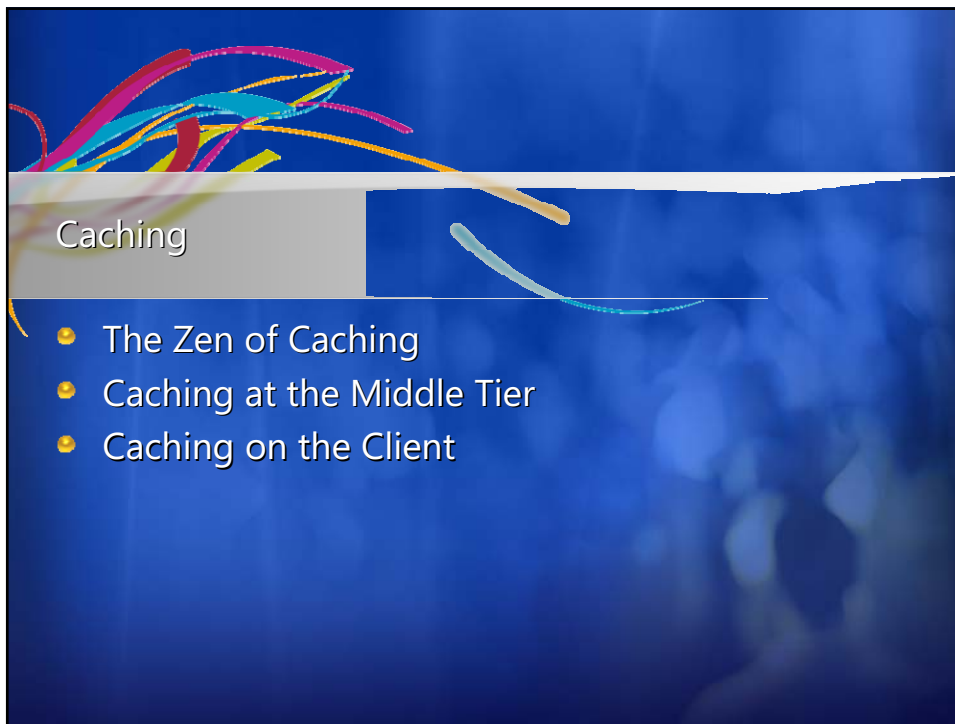
Fragmentation in the Leaf Level of an Index (regardless of type)

- Data Modifications [can] lead to Fragmentation
- INSERT
 - Yes – Key value is not ever increasing/decreasing
 - NO – Key is ever increasing/decreasing
- UPDATE
 - Yes – Updates... to variable width columns – where the values are getting wider and resulting row length no longer fits on page
 - NO – Columns are fixed width, columns have “place holder” values (i.e. DEFAULT constraints) to minimize row expansion on update OR no updates after insert
- DELETE
 - Yes – Deletes are singleton deletes (swiss cheese problem)
 - NO – Deletes are RANGE deletes for archival purposes (where entire pages are emptied)

Indexing Summary

- Indexing is the closest thing to a magic bullet
- But – you need to “find the right balance” and prioritize – know your data!
- Proper maintenance policies/procedures are what’s going to keep you in check
- Good design practices will give you better performance AND more access to features!
- And – if you’re working hard on your database structures then you can take your performance even further with disk and log improvements!
- Testing, Testing, Testing

**See Advanced Indexing Strategies in DAT437 on Thursday – 16:30-17:45, by Kimberly L. Tripp*



Caching

- The Zen of Caching
- Caching at the Middle Tier
- Caching on the Client

Caching Zen

- Moving information to a location where it can be accessed very quickly
 - From main memory to high-speed memory on the processor
 - From the hard drive to main memory
 - From main memory on one machine to another machine
- Caching is framework for managing state
 - State being data + status

Caching Zen: Why?

- Performance
 - Store the data as close as possible to consumer
- Scalability
 - Common data is often processed the same way for multiple users
 - Process it once and store the results
- Availability
 - Provide access to data even when the primary data store or service is unavailable

Caching Zen: When?

- On-demand: cache the data after it has been requested and formatted
- A priori: preload the cache with both raw and formatted data
- Combine the two

Caching Zen: Where?

- Generally as close to the data consumer as possible—wherever accessing the cache can help with
 - Performance
 - Scalability
 - Availability
- Actual cache storage can be
 - Memory resident—often on servers
 - Disk resident—often on client

Caching Zen: Issues

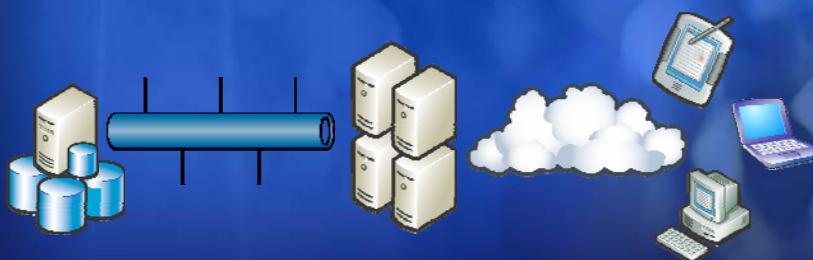
- Staleness
- Coherency
- Security

Staleness

- Data that is cached becomes obsolete
- Staleness is the difference between the master data's current value and the current version that is cached
- Staleness is relative to how often the master copy of the data changes and how tolerant an application is to stale data

Coherency

- Large systems often have multiple caches
 - Often seen with web server farms
- Caches need to be kept in sync otherwise clients can get inconsistent results



Security

- Data stored in cache needs to be protected
- Application logic can enforce access rules to in-memory cache on servers
- Client-side cache might require both in-memory and disk level protection
 - Spyware
 - Unsafe networks
 - Users

Staleness Solution

- Cache expiration settings based upon the application's tolerance to stale data
- Can be time driven where cache is reloaded based upon an expiration setting
- Can be data driven where cache is updated when master data is updated

Coherency Solution

- Depending upon the number of distinct cache stores, updates can occur
 - At the same time
 - Works when cache is small and speed of update is quick
 - At defined intervals
 - Works when cache is large and cache is load-balanced

Security Solution

- Store cache data encrypted
- Ensure communications between cache and data source are secure

Cache Sync

- New feature of SQL Server 2005
- Cache source creates data based subscription that server will monitor
- When data changes, server notifies subscriber of change and reason
 - Uses new Service Broker infrastructure
- Exposed in SQLClient 2.0 and ASP.NET 2.0
 - High-level and low-level implementations available

State Transformation Pipeline

- State can be presented in many formats
 - Raw data object to business object to presentation object
- Caching possible at any and all points in the pipeline



Caching at the Middle Tier

- Prefer existing host processes and services
 - COM+ components and object pooling
 - IIS with ASP.NET
- Use custom services as needed
 - Windows Services with custom endpoint and cache engines

Caching at the Client Tier

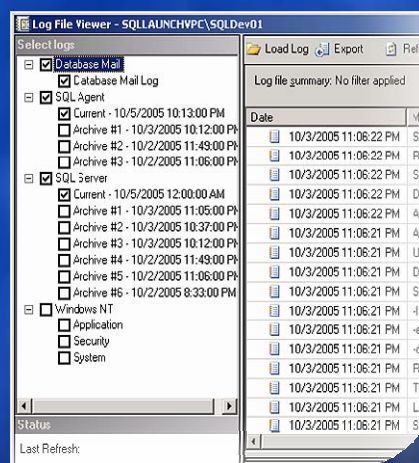
- Local services
 - SQL Server Express
 - Custom Windows Services
- In-memory
 - SQL Everywhere
 - System.Web.Caching
 - DataSets
 - Custom objects

Tuning, Monitoring, and Diagnosing System Problems

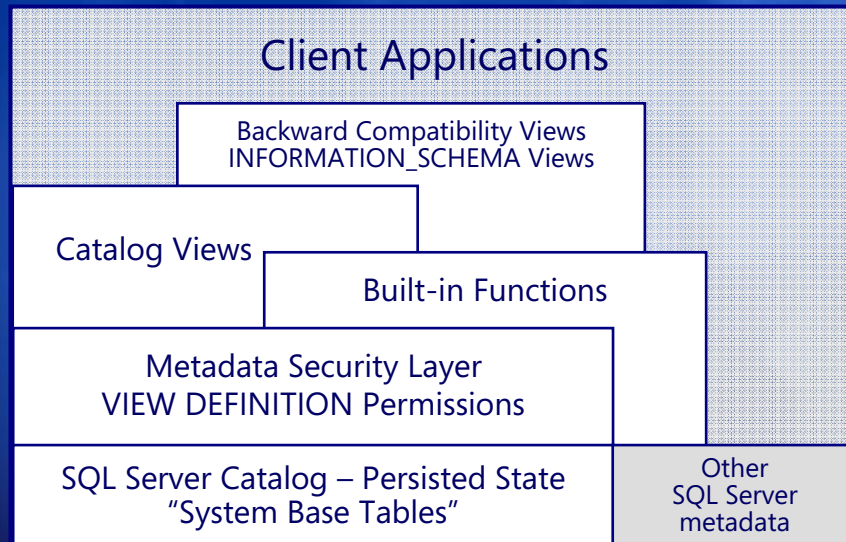
- SQL Server Profiler
- System Monitor
- Database Tuning Advisor

Monitoring A Unified Approach

- Supports multiple logs
 - SQL Server Database Engine Information
 - SQL Server Agent Information
 - Windows Event Viewer Application Log
 - Database Mail Logs
- Across all logs, allows:
 - Searching
 - Filtering
 - Exporting



Accessing "System" Information



Backward Compatibility Views

- Used to access many frequently used columns from SQL Server 2000 system tables (e.g. sysobjects table is now sys.objects view)
- Many limits removed (can have more than 32767 users). Columns that expect a smallint may cause overflow, or return 0/NULL
- Some columns/values no longer used (e.g. sysindexes.FirstIAM and sysindexes.RowModCtr)
- Should move to catalog views and functions for replacement (be aware that it's NOT just simple search and replace for sysobjects ⇒ sys.objects)
 - sysobjects.id ⇒ sys.objects.object_id
 - sysindexes.indid ⇒ sys.indexes.index_id

Accessing System Information

Using the new "sys" schema

- The schema that contains all the engine's system objects
 - The catalog views and compatibility views
 - System stored procedures, functions, etc...
- The sys schema *appears* in every database
- Metadata describing user objects is materialized through catalog views in the resource database
- Metadata describing the system objects themselves is materialized through other catalog views in the sys schema

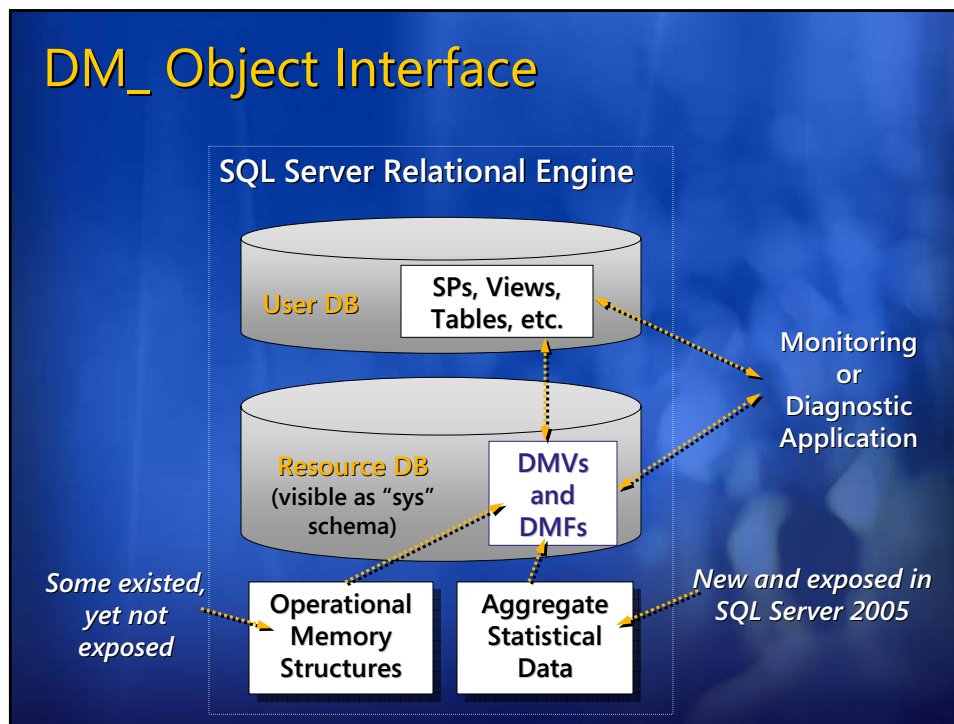
```
SELECT * FROM sys.system_objects
```
- Similar concept to INFORMATION_SCHEMA; however, sys is not an ANSI/ISO standard

Dynamic Management Views

- Views built on top of internal structures
- Ideal for tracking performance

Server Level	Component Level
dm_exec_* Execution of user code and associated connections	dm_repl_* Replication
dm_os_* Memory, locking & scheduling	dm_broker_* SQL Service Broker
dm_tran_* Transactions & isolation	dm_fts_* Full Text Search
dm_io_* I/O on network and disks	dm_qn_* Query Notifications
dm_db_* Databases and database objects	dm_clr_* Common Language Runtime

DM_ Object Interface



Finding DM Objects

Three object types:

- View
- Inline Table-valued Function
- Multi-statement Table-valued Function

```
SELECT so.*  
FROM sys.system_objects AS so  
WHERE so.name LIKE N'dm_%'  
ORDER BY so.type
```

Using DM Objects

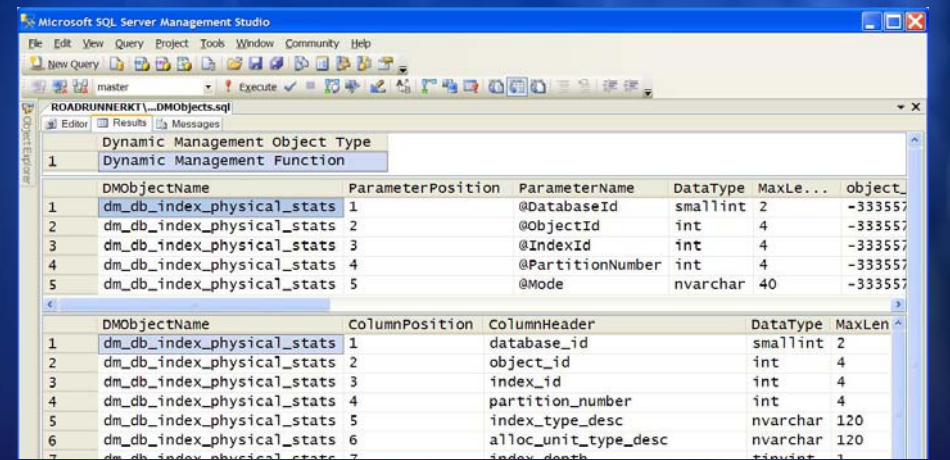
- What type of object is it?
`SELECT dbo.DMObjectType(N'dm_obj_name')`
- If it's a function – what are the input parameters, their order and type?
`SELECT *
FROM dbo.DMFunctionParams(N'dm_obj_name')
ORDER BY ParameterPosition`
- What's the result set?
`SELECT *
FROM dbo.DMObjectColumns(N'dm_obj_name')
ORDER BY ColumnPosition`

To create these procedures use: "DMObjects.sql"

 Hidden Slide
w/extra details

Using DM Objects

- Find out all of this with one simple sproc:
`EXEC dbo.sp_GetDMObjectInfo N'dm_obj_name'`



The screenshot shows the 'Results' pane of SQL Server Enterprise Manager. The query executed is `EXEC dbo.sp_GetDMObjectInfo N'dm_db_index_physical_stats'`. The results are displayed in two tables. The first table shows the parameters of the dynamic management function, and the second table shows the columns of the result set.

DMObjectName	ParameterPosition	ParameterName	DataType	MaxLe...	object...
dm_db_index_physical_stats	1	@DatabaseId	smallint	2	-333557
dm_db_index_physical_stats	2	@ObjectId	int	4	-333557
dm_db_index_physical_stats	3	@IndexId	int	4	-333557
dm_db_index_physical_stats	4	@PartitionNumber	int	4	-333557
dm_db_index_physical_stats	5	@Mode	nvarchar	40	-333557

DMObjectName	ColumnPosition	ColumnHeader	DataType	MaxLen
dm_db_index_physical_stats	1	database_id	smallint	2
dm_db_index_physical_stats	2	object_id	int	4
dm_db_index_physical_stats	3	index_id	int	4
dm_db_index_physical_stats	4	partition_number	int	4
dm_db_index_physical_stats	5	index_type_desc	nvarchar	120
dm_db_index_physical_stats	6	alloc_unit_type_desc	nvarchar	120
dm_db_index_physical_stats	7	index_depth	smallint	2

To create these procedures use: "DMObjects.sql"

 Hidden Slide
w/extra details

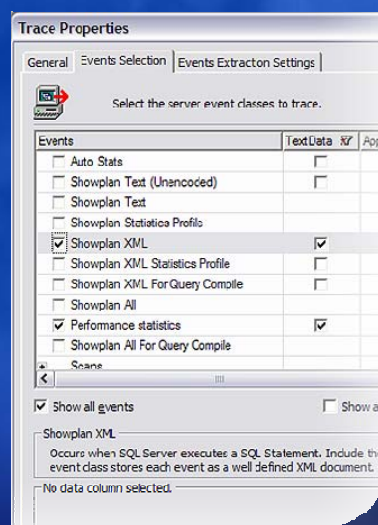
How DMVs Improve Access

- SQL Server 2000: DBCC SHOWCONTIG
 - Not tabular by default
 - Difficult to programmatically analyze
 - Pre-create temp table, complete definition with data types
 - Must use dynamic string execution to execute results into temp table
- SQL Server 2005: `sys.dm_db_index_physical_stats(params)`
 - Table-valued function returns tabular set
 - EASY to programmatically analyze
 - Can use SELECT INTO to catch result set
 - No dynamic string execution needed!

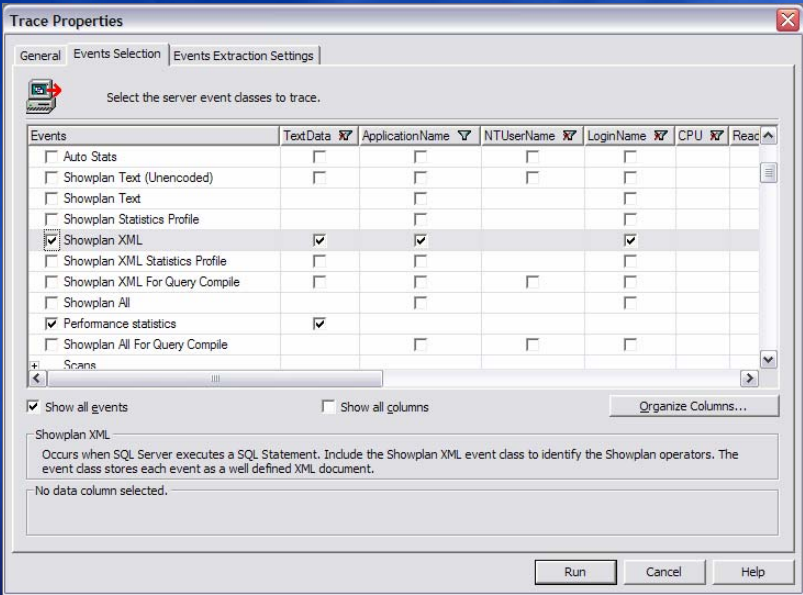
See Link in Resources to "Troubleshooting Performance Problems" for whitepaper.

SQL Server Profiler

- Analyze the SQL Server Database Engine and Analysis Services
- Significantly easier to setup (Events, Data Columns and Filter dialogs combined)
- Special Events: Service Broker, Notification Services, etc...
- Special Event types: Showplan XML and Deadlock Graph – can be saved to files
- Supports pause and modify
- Can Profile SQL Server 2000 and 2005
- Permissions to profile are grantable



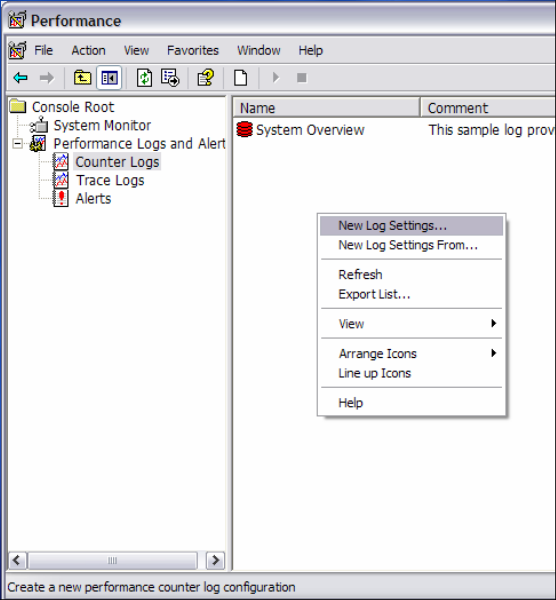
Profiler UI Condensed



Performance Monitor Log

Define

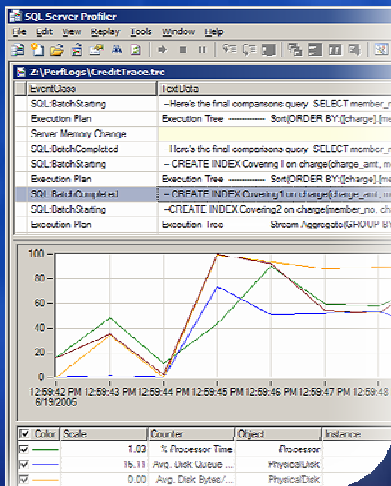
- Objects (complete groups)
- Counters
- Sample Interval
- File Location
- Start log (time)
- Stop log (time)



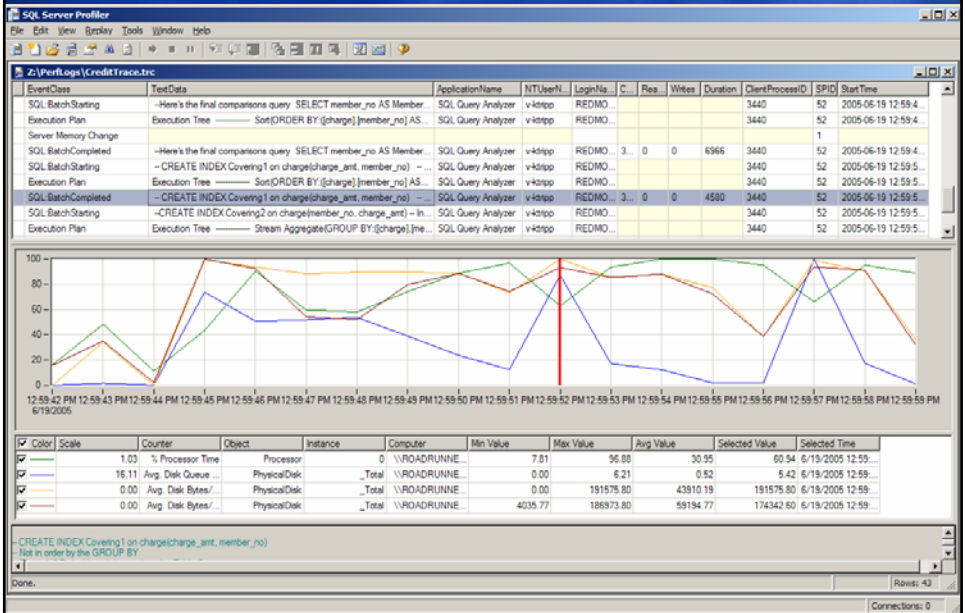
System Monitor Integration

Performance Counter Logs

- Create a Profiler Trace
- Create a Performance Monitor Log
- Open Trace (complete load), Use File – Import Performance Data, Select Objects/Counters...
- Works solely based on time – make sure the two clients (if different) are time correlated
- Can select in either data set and indicators correlate
- Works with SQL Server 2000 and SQL Server 2005

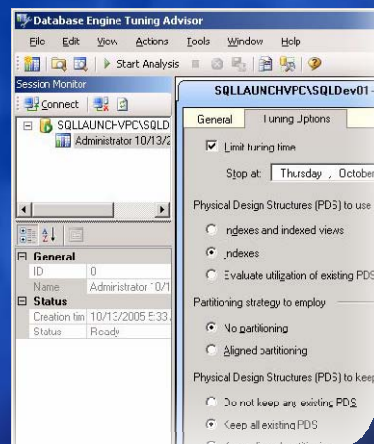


Profiler with Performance Monitor



Database Engine Tuning Advisor

- Partitioning recommendations
- Time-bound tuning
- Indexes with Included columns
- XML Input/Output
- Drop ONLY mode
- Parameterized command line execution
- Import previously saved Session Definition (XML format)
 - Can be a *.trc, *.sql or *.xml format
 - Can be a SQL Server Table



Session Summary

- Building a scalable, reliable, performant, real-world system takes MANY steps...
- **There's no magic bullet!**
- But, it is possible...
 - using a myriad of techniques!*
- Review Blog Entries for demo scripts and all resource links:
 - Kimberly's Blog:
 - <http://www.SQLskills.com/blogs/Kimberly>
 - Brian's Blog:
 - <http://www.mcwtech.com/cs/blogs/brianr/>

TechNet Webcast Series

<http://www.microsoft.com/events/series/technetsqlserver2005.mspix>

- Session 1: **A Fast-paced Feature Overview and Series Introduction**
- Session 2: **Security**, *presented by* Bob Beauchemin, SQLskills.com
- Session 3: **Understanding Installation Options and Initial Configuration**
- Session 4: **Upgrade Considerations and Migration Paths**
- Session 5: **Effective Use of the New Management Tools**
- Session 6: **New Application Design Patterns for Scalability and Availability and the Operational Impacts of Service Broker**, *presented by* Bob Beauchemin, SQLskills.com
- Session 7: **Technologies and Features to Improve Availability**
- Session 8: **Implementing Database Mirroring, Part 1 of 2**, *presented by* Mark Wistrom, Database Mirroring Program Manager, Microsoft Corp.
- Session 9: **Implementing Database Mirroring, Part 2 of 2**
- Session 10: **Recovering from Human Error**
- Session 11: **Best Practices and Series Wrap-up**

MSDN Webcast Series

www.microsoft.com/events/series/msdnsqlserver2005.mspix

- Session 1: **Interaction Between Data and Log**
- Session 2: **Recovery Models**
- Session 3: **Table Optimization Strategies**
- Session 4: **Optimization Through Indexes**
- Session 5: **Optimization Through Maintenance**
- Session 6: **Isolation, Locking, and Blocking**
- Session 7: **Optimizing Procedural Code**
- Session 8: **Table and Index Partitioning**
- Session 9: **Profiling/Server-side Trace Queues**
- Session 10: **Common Roadblocks, A Series Wrapup**

ONLINE Indexing Presentations

- Microsoft® SQL Server™ 2005
 - SQL Server Index Creation Best Practices
<http://www.microsoft.com/emea/itsshowtime/sessionh.aspx?videoid=29>
 - SQL Server Index Defragmentation Best Practices
<http://www.microsoft.com/emea/itsshowtime/sessionh.aspx?videoid=30>
- Microsoft® SQL Server™ 2000
 - Indexing for Performance – Finding the Right Balance
<http://msevents.microsoft.com/CUI/EventDetail.aspx?EventID=1032254503&Culture=en-US>
 - Indexing for Performance – Proper Index Maintenance
<http://msevents.microsoft.com/CUI/EventDetail.aspx?EventID=1032256511&Culture=en-US>

Indexing Whitepapers

- SQL Server 2005
 - Improving Performance with SQL Server 2005 Indexed Views
<http://www.microsoft.com/technet/prodtechnol/sql/2005/ipsql05iv.mspix>
 - Statistics Used by the Query Optimizer in Microsoft SQL Server 2005
<http://www.microsoft.com/technet/prodtechnol/sql/2005/qrystats.mspix>

Indexing Whitepapers

- SQL Server 2000
 - Improving Performance with SQL Server 2000 Indexed Views
<http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/indexvw.msp>
 - Statistics Used by the Query Optimizer in Microsoft SQL Server 2000
<http://msdn.microsoft.com/library/en-us/dnsq12k/html/statquery.asp>
 - Microsoft SQL Server 2000 Index Defragmentation Best Practices
<http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/ss2kidbp.msp>

PDF of Slide Content is Available Online

- <http://www.SQLskills.com/resources/Conferences/200606TechEdPreconf.zip>
- Use a password of **Tech!Ed@2006** to unzip
- Please respect our IP and do not give out the link/password to people who did not attend this workshop!
Thanks so much, kt & br

Thank you! Hope to see you again

Please double-check the times and locations

Kim

- DAT Chalk Talk: SQL Server 2005: Leveraging the Availability and Scalability Features of the Enterprise Edition, **Monday**, June 12, 17:00-18:15
- DAT325: SQL Server Always On Technologies: Best Practices in Building Robust, Recoverable and Reliable Systems, **Tuesday**, June 13, 10:15-11:30, 52 AB
- DAT413: SQL Server Always On Technologies: Disaster Recovery Strategies for Isolated Damage and Human Error, **Wednesday**, June 14 2:00 PM - 3:15 PM, 151 AB
- DAT Chalk Talk: SQL Server 2005: SQLCMD Tips and Tricks, **Thursday**, June 15, 14:45-16:00
- DAT437: SQL Server 2005: Advanced Indexing Strategies, **Thursday**, June 15, 16:30-17:45, 253 ABC

Please remember to complete your session evaluation!

Thank you! Hope to see you again

Please double-check the times and locations

Brian

- DEV307: Delving into Visual Studio 2005 Team Edition for Software Developers
Monday, June 12 13:30-14:45, Grand Ballroom A
- DEV307R: Delving into Visual Studio 2005 Team Edition for Software Developers
Wednesday, June 14 15:45-17:00, 109 AB
- DAT338: Deploying SQL Server Applications
Friday, June 16 10:45-12:00, 259 AB
- DAT Customer Q&A
Friday, June 16 12:00-13:30, DAT TLC

Please remember to complete your session evaluation!

Thank you.
Please fill out your evaluation.



Brian A. Randell
Senior Consultant
<http://www.MCWTech.com>



Kimberly L. Tripp
President and Founder
<http://www.SQLskills.com>



Fill out a session
evaluation on
CommNet and
Win an XBOX 360!





© 2006 Microsoft Corporation. All rights reserved. Microsoft, Windows, Windows Vista and other product names are or may be registered trademarks and/or trademarks in the U.S. and/or other countries. The information herein is for informational purposes only and represents the current view of Microsoft Corporation as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.