

## Introduction to SQL Server 2008

### What is SQL Server 2008/RDBMS?

As you most likely know, SQL Server 2008 is primarily thought of as a *Relational Database Management System (RDBMS)*. It is certainly that, but it is also much more.

SQL Server 2008 can be more accurately described as an *Enterprise Data Platform*. It offers many new features and even more enhanced or improved features from previous editions of the product. In addition to traditional RDBMS duty, SQL Server 2008 also provides rich reporting capabilities, powerful data analysis, and data mining, as well as features that support asynchronous data applications, data-driven event notification, and more.

### Database Engine

The Database Engine is the primary component of SQL Server 2008. It is the Online Transaction Processing (OLTP) engine for SQL Server, and has been improved and enhanced tremendously in this version. The Database Engine is a high-performance component responsible for the efficient storage, retrieval, and manipulation of relational and Extensible Markup Language (XML) formatted data.

SQL Server 2008's Database Engine is highly optimized for transaction processing, but offers exceptional performance in complex data retrieval operations. The Database Engine is also responsible for the controlled access and modification of data through its security subsystem. SQL Server 2008's Database Engine has many major improvements to support scalability, availability, and advanced (and secure) programming objects.

### Analysis Services

Analysis Services delivers Online Analytical Processing (OLAP) and Data Mining functionality for business intelligence applications. As its name suggests, Analysis Services provides a very robust environment for the detailed analysis of data. It does this through user-created, multidimensional data structures that contain de-normalized and aggregated data from diverse data sources (such as relational databases, spreadsheets, flat files, and even other multidimensional sources).

### Reporting Services

Reporting Services is a Web service-based solution for designing, deploying, and managing flexible, dynamic Web-based reports, as well as traditional paper reports. These reports can contain information from virtually any data source. Because Reporting Services is implemented as a Web service, it must be installed on a server with Internet Information Services (IIS). However, IIS does not have to be installed on a SQL Server. The Reporting Services databases are hosted on SQL Server 2008, but the Web service itself can be configured on a separate server.



## **Integration Services**

SQL Server Integration Services (SSIS) is Microsoft's new enterprise class data Extract, Transform, and Load (ETL) tool. SSIS is a completely new product built from the ashes of SQL Server 2000's Data Transformation Services (DTS). SSIS offers a much richer feature set and the ability to create much more powerful and flexible data transformations than its predecessor. This huge improvement, however, is not without a cost. SSIS is a fairly complex tool and offers a completely different design paradigm than DTS. Database administrators adept at the former tool are very often intimidated and frustrated by the new SSIS. Their biggest mistake is in thinking that Integration Services would just be an upgrade of Data Transformation Services.

## **Replication Services**

SQL Server 2008 Replication Services provides the ability to automate and schedule the copying and distribution of data and database objects from one database or server to another, while ensuring data integrity and consistency. Replication has been enhanced in SQL Server 2008 to include true Peer-to-Peer replication, replication over HTTP, the ability to replicate schema changes, and, very interestingly, the ability to configure an Oracle server as a replication publisher.

## **Multiple Instances**

SQL Server 2008 provides the capability of installing multiple instances of the database application on a single computer. Depending on the edition of SQL Server being installed, up to 50 instances can be installed. This feature allows for one high-performance server to host multiple instances of the SQL Server services, each with its own configuration and databases. Each instance can be managed and controlled separately with no dependency on each other.

## **Database Mail**

In the past SQL Server relied on a Messaging Application Programming Interface (MAPI) mail client configured on the server to facilitate email and pager notification for administrative and programmatic purposes. What this essentially meant was that to fully utilize administrative notifications, the administrator needed to install Outlook or some other MAPI-compliant client on the server, and then create a mail profile for the service account to use.

Many organizations wanted to take advantage of the SQL Server Agent's ability to send job and event notification via email but were unwilling to install unnecessary and potentially risky software on production server assets. The SQL Server 2008 Database Mail feature removes this requirement by supporting Simple Mail Transfer Protocol (SMTP) for all mail traffic. In addition, multiple mail profiles can be created in the database to support different database applications.

## **SQL Server 2008 Services**

SQL Server runs as a service. In fact, it runs as several services if all the different features of the product are installed. It is important to know what service is responsible for what part of the application so that each service can be configured



correctly, and so that unneeded services can be disabled to reduce the overhead on the server and reduce the surface area of SQL Server.

### **MSSQLServer (SQL Server)**

The MSSQLServer service is the database engine. To connect and transact against a SQL Server 2008 database, the MSSQLServer service must be running. Most of the functionality and storage features of the database engine are controlled by this service.

The MSSQLServer service can be configured to run as the local system or as a domain user. If installed on Windows Server 2003, it can also be configured to run under the Network System account.

### **SQLServerAgent (SQL Server Agent)**

This service is responsible for the execution of scheduled jobs such as scheduled backups, import/export jobs, and Integration Services packages. If any scheduled tasks require network or file system access, the SQLServerAgent service's credentials are typically used.

The SQLServerAgent service is dependent on the MSSQLServer service. During installation, the option is given to configure both services with the same credentials. Although this is by no means required, it is common practice. A frequent problem encountered by database administrators is that jobs that work perfectly when run manually fail when run by the agent. The reason for the failure is because the account that is used when testing the job manually is the logged-in administrator, but when the job is executed by the agent, the account the agent is running under does not have adequate permissions.

### **MSSQLServerOLAPService (SQL Server Analysis Services)**

MSSQLServerOLAPService is the service that Analysis Services runs under. Analysis Services provides the services and functionality to support all of SQL Server 2008's OLAP needs, as well as the new data mining engine included with SQL Server 2008.

### **SQLBrowser (SQL Server Browser)**

The SQLBrowser service is used by SQL Server for named instance name resolution and server name enumeration over TCP/IP and VIA networks.

The default instance of SQL Server is assigned the TCP port 1433 by default to support client communication. However, because more than one application cannot share a port assignment, any named instances are given a random port number when the service is started. This random port assignment makes it difficult for clients to connect to it, because the client applications don't know what port the server is listening on. To meet this need, the SQLBrowser service was created.



### **MSDTSServer (SQL Server Integration Services)**

The MSDTSServer service provides management and storage support for SSIS. Although this service is not required to create, store, and execute SSIS packages, it does allow for the monitoring of SSIS package execution and displaying of a hierarchical view of SSIS packages and folders that are stored in different physical locations.

### **ReportServer (SQL Server Reporting Services)**

The ReportServer service is the process in which Reporting Services runs. The service is accessible as a Web service and provides for report rendering, creation, management, and deploying.

### **MSDTC (Distributed Transaction Coordinator)**

The MSDTC service is used to manage transactions that span more than one instance of SQL Server or an instance of SQL Server and another transaction-based system. It utilizes a protocol known as Two-Phased Commit (2PC) to ensure that all transactions that span systems are committed on all participating systems.

## **SQL Server 2008 Database Objects**

SQL Server 2008 database objects are defined and exist within a defined scope and hierarchy. This hierarchy enables more control over security permissions and organization of objects by similar function. SQL Server 2008 objects are defined at the Server, Database, and Schema levels.

### **Server**

The server scope encompasses all the objects that exist on the instance of SQL Server, regardless of their respective database or namespace. The database object resides within the server scope.

We can install multiple instances of the SQL Server 2008 Data Platform application on a single computer running a Windows operating system.

### **Database**

The database scope defines all the objects within a defined database catalog. Schemas exist in the database scope.

### **Schema**

Each database can contain one or more schemas. A schema is a namespace for database objects. All data objects in a SQL Server 2008 database reside in a specific schema.



## Object Names

Every object in a SQL Server 2008 database is identified by a four-part, fully qualified name. This fully qualified name takes the form of server.database.schema.object. However, when referring to objects, the fully qualified name can be abbreviated. By omitting the server name SQL Server will assume the instance the connection is currently connected to. Likewise, omitting the database name will cause SQL Server to assume the existing connection's database context.

## SQL Server 2008 Databases

There are two types of databases in SQL Server: system databases and user databases. The *system databases* are used to store system-wide data and metadata. *User databases* are created by users who have the appropriate level of permissions to store application data.

### System Databases

The system databases are comprised of Master, Model, MSDB, TempDB, and the hidden Resource database. If the server is configured to be a replication distributor, there will also be at least one system distribution database that is named during the replication configuration process.

#### The Master Database

The Master database is used to record all server-level objects in SQL Server 2008. This includes Server Logon accounts, Linked Server definitions, and EndPoints. The Master database also records information about all the other databases on the server (such as their file locations and names). Unlike its predecessors, SQL Server 2008 does not store system information in the Master database, but rather in the Resource database. However, system information is logically presented as the SYS schema in the Master database.

#### The Model Database

The Model database is a template database. Whenever a new database is created (including the system database TempDB), a copy of the Model database is created and renamed with the name of the database being created. The advantage of this behavior is that objects can be placed in the Model database prior to the creation of any new database and, when the database is created, the objects will appear in the new database.

#### The MSDB Database

I mostly think of the MSDB database as the SQL Server Agent's database. That's because the SQL Server Agent uses the MSDB database extensively for the storage of automated job definitions, job schedules, operator definitions, and alert definitions.



### **The TempDB Database**

The TempDB database is used by SQL Server to store data temporarily. The TempDB database is used extensively during SQL Server operations, so careful planning and evaluation of its size and placement are critical to ensure efficient SQL Server database operations.

The TempDB database is used by the Database Engine to store temporary objects (such as temporary tables, views, cursors, and table-valued variables) that are explicitly created by database programmers. In addition, the TempDB database is used by the SQL Server database engine to store work tables containing intermediate results of a query prior to a sort operation or other data manipulation.

### **The Resource Database**

The last system database is the Resource database. The Resource database is a read-only database that contains all the system objects used by an instance of SQL Server. The Resource database is not accessible during normal database operations. It is logically presented as the SYS schema in every database. It contains no user data or metadata. Instead, it contains the structure and description of all system objects. This design enables the fast application of service packs by just replacing the existing Resource database with a new one. As an added bonus, to roll back a service pack installation, all you have to do is replace the new Resource database with the old one. This very elegant design replaces the older method of running many scripts that progressively dropped and added new system objects.

### **User Databases**

User databases are simply that: databases created by users. They are created to store data used by data applications and are the primary purpose of having a database server.

### **Distribution Databases**

The distribution database stores metadata and transactional history to support all types of replication on a SQL Server. Typically, one distribution database is created when configuring a SQL Server as a replication Distributor. However, if needed, multiple distribution databases can be configured.

A model distribution database is installed by default and is used in the creation of a distribution database used in replication. It is installed in the same location as the rest of the system databases and is named distmdl.mdf.



## What's New in SQL Server 2008?

### New in SQL Server Installation

SQL Server 2008 has new Setup architecture for the following scenarios: installation, upgrade, maintenance, failover clustering, and command prompt installations.

The SQL Server Installation Wizard is Windows Installer-based. It provides a single feature tree for installation of all SQL Server components, so you do not have to install the following components individually:

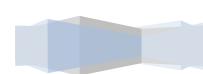
- Database Engine
- Analysis Services
- Reporting Services
- Integration Services
- Replication
- Management tools
- Connectivity components
- Sample databases, samples, and SQL Server Books Online

### New in SQL Server Database Engine:

This latest release of the SQL Server Database Engine introduces new features and enhancements that increase the power and productivity of architects, developers, and administrators who design, develop, and maintain data storage systems.

These are the areas in which the Database Engine has been enhanced.

Topic	Description
Availability Enhancements (Database Engine)	The availability of Microsoft SQL Server 2008 databases is improved by enhancements to database mirroring. Database mirroring enables the creation of hot standby servers that provide rapid failover support with no loss of data from committed transactions.
Manageability Enhancements (Database Engine)	Manageability of the SQL Server 2008 Database Engine is simplified by enhancements to tools and monitoring features.
Programmability Enhancements (Database Engine)	Programmability enhancements in the Database Engine include new data storage features, new data types, new full-text search architecture, and numerous improvements and additions to Transact-SQL.



Scalability and Performance Enhancements (Database Engine)	Scalability and performance enhancements in the Database Engine include filtered indexes and statistics, new table and query hints, and new query performance and query processing features.
Security Enhancements (Database Engine)	Security enhancements in the Database Engine include new encryption functions, the transparent data encryption and extensible key management features, and a clarification of DES algorithms.

## What's New (Replication?)

Replication Monitor includes the following usability improvements:

- In most Replication Monitor grids, you can now do the following: select which columns to view; sort by multiple columns; and filter rows in the grid based on column values.

To access this functionality: right-click a grid, and then select **Choose Columns to Show, Sort, Filter, or Clear Filter**. Filter settings are specific to each grid. Column selection and sorting are applied to all grids of the same type, such as the publications grid for each Publisher.

- The **Common Jobs** tab for the Publisher node has been renamed to **Agents**. The **Agents** tab now provides a centralized location to view information about all the agents and jobs that are associated with publications at the selected Publisher. Agents and jobs that are associated with publications include the following:
  - The Snapshot Agent, which is used by all publications.
  - The Log Reader Agent, which is used by all transactional publications.
  - The Queue Reader Agent, which is used by transactional publications that are enabled for queued updating subscriptions.
  - Maintenance jobs, which are used by all publications.

The Distribution Agent and Merge Agent are associated with subscriptions to publications.



## SQL Server 2008 Architecture

### Components of the SQL Server Engine

Figure 1-1 shows the general architecture of SQL Server, which has four major components (three of whose subcomponents are listed): protocols, the relational engine (also called the Query Processor), the storage engine, and the SQLOS. Every batch submitted to SQL Server for execution, from any client application, must interact with these four components. (For simplicity, I've made some minor omissions and simplifications and ignored certain "helper" modules among the subcomponents.)

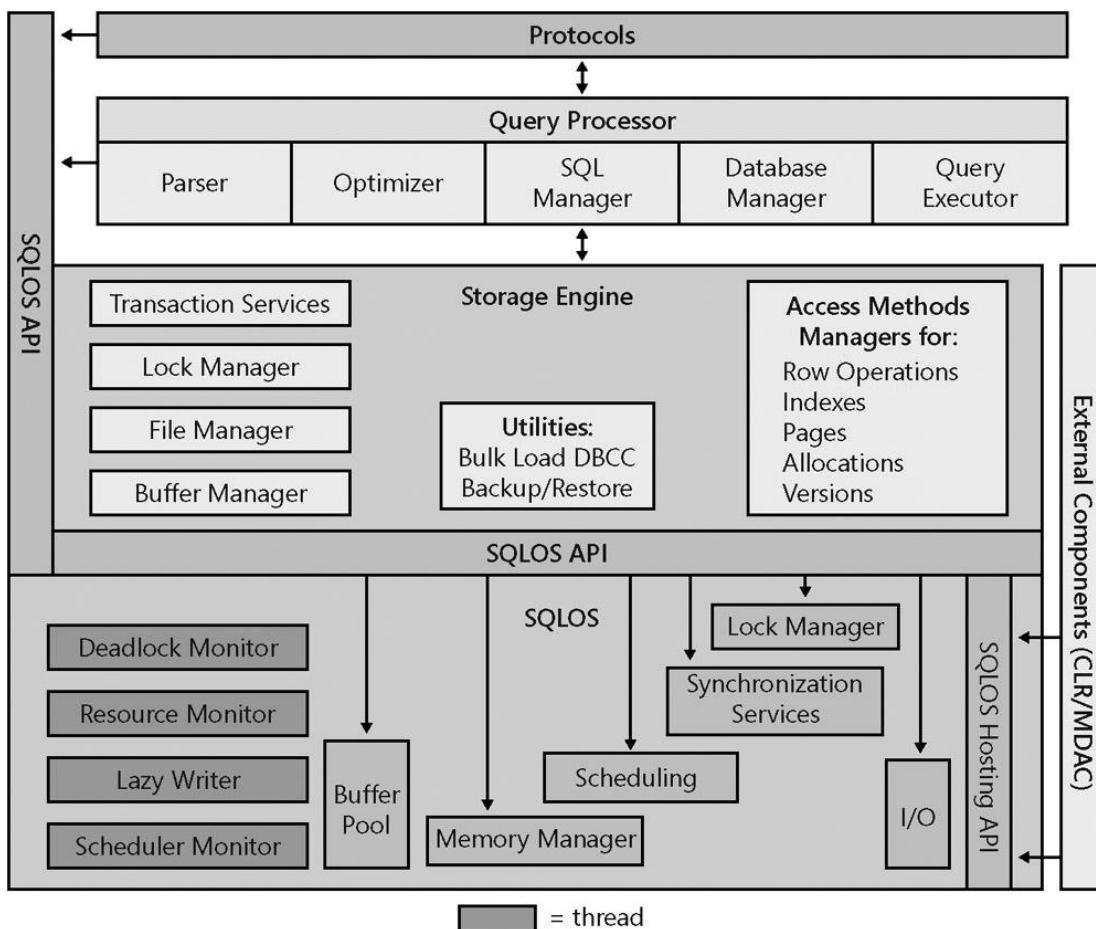


Figure 1-1: The major components of the SQL Server database engine

The protocol layer receives the request and translates it into a form that the relational engine can work with, and it also takes the final results of any queries, status messages, or error messages and translates them into a form the client can understand before sending them back to the client. The relational engine layer accepts SQL batches and determines what to do with them. For Transact-SQL queries and programming constructs, it parses, compiles, and optimizes the request and oversees the process of executing the batch. As the batch is executed, if data is needed, a request for that data is passed to the storage engine. The storage engine manages all data access, both through transaction-based commands and bulk



operations such as backup, bulk insert, and certain DBCC (Database Consistency Checker) commands. The SQLOS layer handles activities that are normally considered to be operating system responsibilities, such as thread management (scheduling), synchronization primitives, deadlock detection, and memory management, including the buffer pool.

## Protocols

When an application communicates with the SQL Server Database Engine, the application programming interfaces (APIs) exposed by the protocol layer formats the communication using a Microsoft-defined format called a *tabular data stream (TDS) packet*. There are Net-Libraries on both the server and client computers that encapsulate the TDS packet inside a standard communication protocol, such as TCP/IP or Named Pipes. On the server side of the communication, the Net-Libraries are part of the Database Engine, and that protocol layer is illustrated in Figure 1-1. On the client side, the Net-Libraries are part of the SQL Native Client. The configuration of the client and the instance of SQL Server determine which protocol is used.

SQL Server can be configured to support multiple protocols simultaneously, coming from different clients. Each client connects to SQL Server with a single protocol. If the client program does not know which protocols SQL Server is listening on, you can configure the client to attempt multiple protocols sequentially. The following protocols are available:

- **Shared Memory** The simplest protocol to use, with no configurable settings. Clients using the Shared Memory protocol can connect only to a SQL Server instance running on the same computer, so this protocol is not useful for most database activity. Use this protocol for troubleshooting when you suspect that the other protocols are configured incorrectly. Clients using MDAC 2.8 or earlier cannot use the Shared Memory protocol. If such a connection is attempted, the client is switched to the Named Pipes protocol.
- **Named Pipes** A protocol developed for local area networks (LANs). A portion of memory is used by one process to pass information to another process, so that the output of one is the input of the other. The second process can be local (on the same computer as the first) or remote (on a networked computer).
- **TCP/IP** The most widely used protocol over the Internet. TCP/IP can communicate across interconnected networks of computers with diverse hardware architectures and operating systems. It includes standards for routing network traffic and offers advanced security features. Enabling SQL Server to use TCP/IP requires the most configuration effort, but most networked computers are already properly configured.
- **Virtual Interface Adapter (VIA)** A protocol that works with VIA hardware. This is a specialized protocol; configuration details are available from your hardware vendor.

## Tabular Data Stream Endpoints

10

SQL Server 2005 also introduces a new concept for defining SQL Server connections: the connection is represented on the server end by a TDS endpoint. During setup, SQL Server creates an endpoint for each of the four Net-Library protocols supported



by SQL Server, and if the protocol is enabled, all users have access to it. For disabled protocols, the endpoint still exists but cannot be used. An additional endpoint is created for the dedicated administrator connection (DAC), which can be used only by members of the sysadmin fixed server role. (I'll discuss the DAC in more detail in configuration chapter.)

## The Relational Engine

As mentioned earlier, the relational engine is also called the query processor. It includes the components of SQL Server that determine exactly what your query needs to do and the best way to do it. By far the most complex component of the query processor, and maybe even of the entire SQL Server product, is the query optimizer, which determines the best execution plan for the queries in the batch.

The relational engine also manages the execution of queries as it requests data from the storage engine and processes the results returned. Communication between the relational engine and the storage engine is generally in terms of OLE DB row sets. (*Row set* is the OLE DB term for a *result set*.) The storage engine comprises the components needed to actually access and modify data on disk.

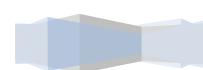
## The Command Parser

The command parser handles Transact-SQL language events sent to SQL Server. It checks for proper syntax and translates Transact-SQL commands into an internal format that can be operated on. This internal format is known as a *query tree*. If the parser doesn't recognize the syntax, a syntax error is immediately raised that identifies where the error occurred. However, non-syntax error messages cannot be explicit about the exact source line that caused the error. Because only the command parser can access the source of the statement, the statement is no longer available in source format when the command is actually executed.

## The Query Optimizer

The query optimizer takes the query tree from the command parser and prepares it for execution. Statements that can't be optimized, such as flow-of-control and DDL commands, are compiled into an internal form. The statements that are optimizable are marked as such and then passed to the optimizer. The optimizer is mainly concerned with the DML statement *SELECT, INSERT, UPDATE, and DELETE*, which can be processed in more than one way, and it is the optimizer's job to determine which of the many possible ways is the best. It compiles an entire command batch, optimizes queries that are optimizable, and checks security. The query optimization and compilation result in an execution plan.

The first step in producing such a plan is to *normalize* each query, which potentially breaks down a single query into multiple, fine-grained queries. After the optimizer normalizes a query, it *optimizes* it, which means it determines a plan for executing that query. Query optimization is cost based; the optimizer chooses the plan that it determines would cost the least based on internal metrics that include estimated memory requirements, CPU utilization, and number of required I/Os. The optimizer considers the type of statement requested, checks the amount of data in the various tables affected, looks at the indexes available for each table, and then looks at a sampling of the data values kept for each index or column referenced in the query.



The sampling of the data values is called *distribution statistics*. Based on the available information, the optimizer considers the various access methods and processing strategies it could use to resolve a query and chooses the most cost-effective plan.

### **The SQL Manager**

The SQL manager is responsible for everything related to managing stored procedures and their plans. It determines when a stored procedure needs recompilation, and it manages the caching of procedure plans so that other processes can reuse them.

The SQL manager also handles auto parameterization of queries. In SQL Server 2008, certain kinds of ad hoc queries are treated as if they were parameterized stored procedures, and query plans are generated and saved for them. SQL Server can save and reuse plans in several other ways, but in some situations using a saved plan might not be a good idea.

### **The Database Manager**

The database manager handles access to the metadata needed for query compilation and optimization, making it clear that none of these separate modules can be run completely separately from the others. The metadata is stored as data and is managed by the storage engine, but metadata elements such as the data types of columns and the available indexes on a table must be available during the query compilation and optimization phase, before actual query execution starts.

### **The Query Executor**

The query executor runs the execution plan that the optimizer produced, acting as a dispatcher for all the commands in the execution plan. This module steps through each command of the execution plan until the batch is complete. Most of the commands require interaction with the storage engine to modify or retrieve data and to manage transactions and locking.

### **The Storage Engine**

The SQL Server storage engine has traditionally been considered to include all the components involved with the actual processing of data in your database. SQL Server 2005 separates out some of these components into a module called the SQLOS. In fact, the SQL Server storage engine team at Microsoft actually encompasses three areas: access methods, transaction management, and the SQLOS.

### **Transaction Services**

A core feature of SQL Server is its ability to ensure that transactions are *atomic*—that is, all or nothing. In addition, transactions must be durable, which means that if a transaction has been committed, it must be recoverable by SQL Server no matter what—even if a total system failure occurs 1 millisecond after the commit was



acknowledged. There are actually four properties that transactions must adhere to, called the ACID properties: atomicity, consistency, isolation, and durability.

**Locking Operations** Locking is a crucial function of a multi-user database system such as SQL Server, even if you are operating primarily in the snapshot isolation level with optimistic concurrency. SQL Server lets you manage multiple users simultaneously and ensures that the transactions observe the properties of the chosen isolation level. Even though readers will not block writers and writers will not block readers in snapshot isolation, writers do acquire locks and can still block other writers, and if two writers try to change the same data concurrently, a conflict will occur that must be resolved. The locking code acquires and releases various types of locks, such as share locks for reading, exclusive locks for writing, intent locks taken at a higher granularity to signal a potential “plan” to perform some operation, and extent locks for space allocation. It manages compatibility between the lock types, resolves deadlocks, and escalates locks if needed. The locking code controls table, page, and row locks as well as system data locks.

### The SQLOS

Whether the components of the SQLOS layer are actually part of the storage engine depends on whom you ask. In addition, trying to figure out exactly which components are in the SQLOS layer can be rather like herding cats. I have seen several technical presentations on the topic at conferences and have exchanged e-mail and even spoken face to face with members of the product team, but the answers vary. The manager who said he was responsible for the SQLOS layer defined the SQLOS as everything he was responsible for, which is a rather circular definition. Earlier versions of SQL Server have a thin layer of interfaces between the storage engine and the actual operating system through which SQL Server makes calls to the OS for memory allocation, scheduler resources, thread and worker management, and synchronization objects. However, the services in SQL Server that needed to access these interfaces can be in any part of the engine. SQL Server requirements for managing memory, schedulers, synchronization objects, and so forth have become more complex. Rather than each part of the engine growing to support the increased functionality, all services in SQL Server that need this OS access have been grouped together into a single functional unit called the SQLOS. In general, the SQLOS is like an operating system inside SQL Server. It provides memory management, scheduling, IO management, a framework for locking and transaction management, deadlock detection, and general utilities for dumping, exception handling, and so on.

Another member of the product team described the SQLOS to me as a set of data structures and APIs that could potentially be needed by operations running at any layer of the engine. For example, consider various operations that require use of memory. SQL Server doesn't just need memory when it reads in data pages through the storage engine; it also needs memory to hold query plans developed in the query processor layer. Figure 1-1 (shown earlier) depicts the SQLOS layer in several parts, but this is just a way of showing that many SQL Server components use SQLOS functionality.



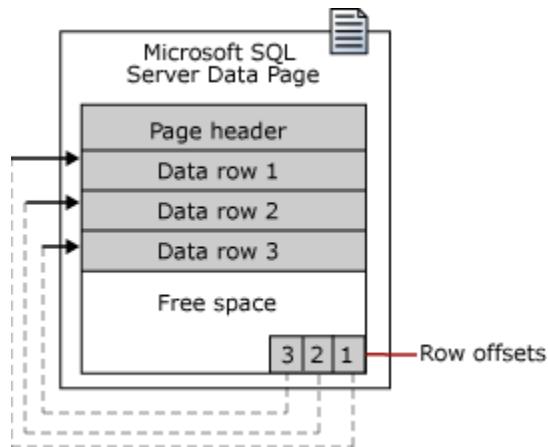
## Pages and Extents

### Pages

The fundamental unit of data storage in SQL Server is the page. The disk space allocated to a data file (.mdf or .ndf) in a database is logically divided into pages numbered contiguously from 0 to  $n$ .

Extents are a collection of eight physically contiguous pages and are used to efficiently manage the pages. All pages are stored in extents. In SQL Server, the page size is 8 KB. Each page begins with a 96-byte header that is used to store system information about the page. This information includes the page number, page type, the amount of free space on the page, and the allocation unit ID of the object that owns the page.

Data rows are put on the page serially, starting immediately after the header. A row offset table starts at the end of the page, and each row offset table contains one entry for each row on the page. Each entry records how far the first byte of the row is from the start of the page. The entries in the row offset table are in reverse sequence from the sequence of the rows on the page.



The maximum amount of data and overhead that is contained in a single row on a page is 8,060 bytes (8 KB).

### Extents

Extents are the basic unit in which space is managed. An extent is eight physically contiguous pages, or 64 KB. This means SQL Server databases have 16 extents per megabyte.

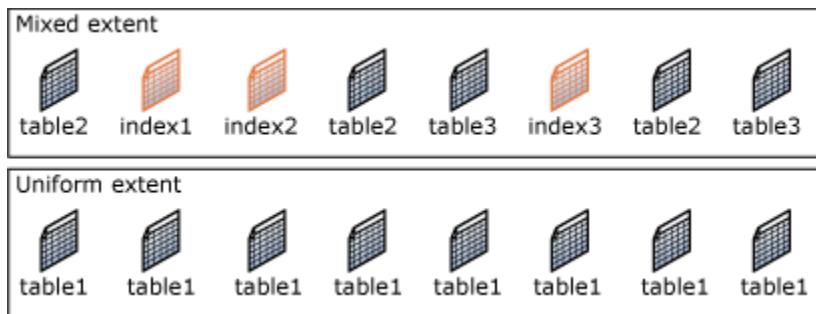
SQL Server has two types of extents:

- Uniform extents are owned by a single object; all eight pages in the extent can only be used by the owning object.
- Mixed extents are shared by up to eight objects. Each of the eight pages in the extent can be owned by a different object.

A new table or index is generally allocated pages from mixed extents. When the table or index grows to the point that it has eight pages, it then switches to use uniform extents for subsequent allocations. If you create an index on an existing



table that has enough rows to generate eight pages in the index, all allocations to the index are in uniform extents.



### Managing Extent Allocations

SQL Server uses two types of allocation maps to record the allocation of extents:

- Global Allocation Map (GAM)

GAM pages record what extents have been allocated. Each GAM covers 64,000 extents, or almost 4 GB of data. The GAM has one bit for each extent in the interval it covers. If the bit is 1, the extent is free; if the bit is 0, the extent is allocated.

- Shared Global Allocation Map (SGAM)

SGAM pages record which extents are currently being used as mixed extents and also have at least one unused page. Each SGAM covers 64,000 extents, or almost 4 GB of data. The SGAM has one bit for each extent in the interval it covers. If the bit is 1, the extent is being used as a mixed extent and has a free page. If the bit is 0, the extent is not used as a mixed extent, or it is a mixed extent and all its pages are being used.

Each extent has the following bit patterns set in the GAM and SGAM, based on its current use.

Current use of extent	GAM bit setting	SGAM bit setting
Free, not being used	1	0
Uniform extent, or full mixed extent	0	0
Mixed extent with free pages	0	1

### Tracking Free Space

Page Free Space (PFS) pages record the allocation status of each page, whether an individual page has been allocated, and the amount of free space on each page. The PFS has one byte for each page, recording whether the page is allocated, and if so, whether it is empty, 1 to 50 percent full, 51 to 80 percent full, 81 to 95 percent full, or 96 to 100 percent full.



## Files and File groups

SQL Server maps a database over a set of operating-system files. Data and log information are never mixed in the same file, and individual files are used only by one database. File groups are named collections of files and are used to help with data placement and administrative tasks such as backup and restore operations

### Database Files

SQL Server databases have three types of files:

- Primary data files

The primary data file is the starting point of the database and points to the other files in the database. Every database has one primary data file. The recommended file name extension for primary data files is .mdf.

- Secondary data files

Secondary data files make up all the data files, other than the primary data file. Some databases may not have any secondary data files, while others have several secondary data files. The recommended file name extension for secondary data files is .ndf.

- Log files

Log files hold all the log information that is used to recover the database. There must be at least one log file for each database, although there can be more than one. The recommended file name extension for log files is .ldf.

SQL Server does not enforce the .mdf, .ndf, and .ldf file name extensions, but these extensions help you identify the different kinds of files and their use.

### Database File groups

Database objects and files can be grouped together in file groups for allocation and administration purposes. There are two types of file groups:

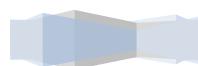
#### Primary

The primary file group contains the primary data file and any other files not specifically assigned to another file group. All pages for the system tables are allocated in the primary file group.

#### User-defined

User-defined file groups are any file groups that are specified by using the FILEGROUP keyword in a CREATE DATABASE or ALTER DATABASE statement.

Log files are never part of a file group. Log space is managed separately from data space.



## Memory Architecture

### 32-bit Vs 64-bit Architecture

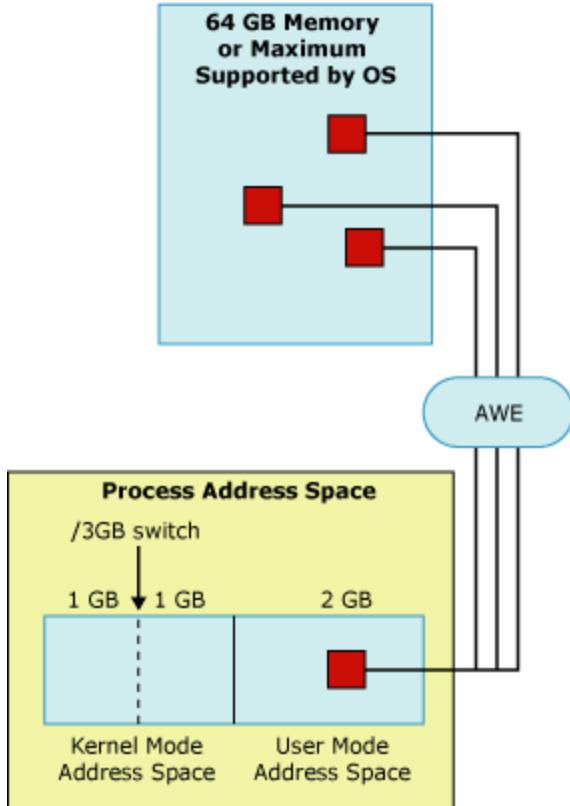
A 32-bit machine can directly address only 4 GB of memory, and by default, Windows itself reserves the top 2 GB of address space for its own use, which leaves only 2 GB as the maximum size of the VAS for any application, such as SQL Server. You can increase this by enabling a /3GB flag in the system's Boot.ini file, which allows applications to have a VAS of up to 3 GB. If your system has more than 3GB of RAM, the only way a 32-bit machine can get to it is by enabling AWE. One benefit in SQL Server 2005 of using AWE, is that memory pages allocated through the AWE mechanism are considered locked pages and can never be swapped out.

On a 64-bit platform, the AWE Enabled configuration option is present, but its setting is ignored. However, the Windows policy Lock Pages in Memory option is available, although it is disabled by default. This policy determines which accounts can make use of a Windows feature to keep data in physical memory, preventing the system from paging the data to virtual memory on disk. It is recommended that you enable this policy on a 64-bit system.

On 32-bit operating systems, you will have to enable Lock Pages in Memory policy when using AWE. It is recommended that you don't enable the Lock Pages in Memory policy if you are not using AWE. Although SQL Server will ignore this option when AWE is not enabled, other processes on the system may be impacted.

All 32-bit applications have a 4-gigabyte (GB) process address space (32-bit addresses can map a maximum of 4 GB of memory). Microsoft Windows operating systems provide applications with access to 2 GB of process address space, specifically known as user mode virtual address space. All threads owned by an application share the same user mode virtual address space. The remaining 2 GB are reserved for the operating system (also known as kernel mode address space). All operating system editions starting with Windows 2000 Server, including Windows Server 2003, have a boot.ini switch that can provide applications with access to 3 GB of process address space, limiting the kernel mode address space to 1 GB.





Address Windowing Extensions (AWE) extend the capabilities of 32-bit applications by allowing access to as much physical memory as the operating system supports. AWE accomplishes this by mapping a subset of up to 64 GB into the user address space. Mapping between the application buffer pool and AWE-mapped memory is handled through manipulation of the Windows virtual memory tables.

To enable support for 3 GB of user mode process space, you must add the **/3gb** parameter to the boot.ini file and reboot the computer, allowing the **/3gb** parameter to take effect. Setting this parameter allows user application threads to address 3 GB of process address space, and reserves 1 GB of process address space for the operating system.

If there is more than 16 GB of physical memory available on a computer, the operating system needs 2 GB of process address space for system purposes and therefore can support only a 2 GB user mode address space. In order for AWE to use the memory range above 16 GB, be sure that the **/3gb** parameter is not in the boot.ini file. If it is, the operating system cannot address any memory above 16 GB.

### **Dynamic Memory Management**

The default memory management behavior of the Microsoft SQL Server Database Engine is to acquire as much memory as it needs without creating a memory shortage on the system. The Database Engine does this by using the Memory Notification APIs in Microsoft Windows.

Virtual address space of SQL Server can be divided into two distinct regions: space occupied by the buffer pool and the rest. If AWE mechanism is enabled, the buffer pool may reside in AWE mapped memory, providing additional space for database pages.



The buffer pool serves as a primary memory allocation source of SQL Server. External components that reside inside SQL Server process, such as COM objects, and not aware of the SQL Server memory management facilities, use memory outside of the virtual address space occupied by the buffer pool.

When SQL Server starts, it computes the size of virtual address space for the buffer pool based on a number of parameters such as amount of physical memory on the system, number of server threads and various startup parameters. SQL Server reserves the computed amount of its process virtual address space for the buffer pool, but it acquires (commits) only the required amount of physical memory for the current load.

The instance then continues to acquire memory as needed to support the workload. As more users connect and run queries, SQL Server acquires the additional physical memory on demand. A SQL Server instance continues to acquire physical memory until it either reaches its max server memory allocation target or Windows indicates there is no longer an excess of free memory; it frees memory when it has more than the min server memory setting, and Windows indicates that there is a shortage of free memory.

As other applications are started on a computer running an instance of SQL Server, they consume memory and the amount of free physical memory drops below the SQL Server target. The instance of SQL Server adjusts its memory consumption. If another application is stopped and more memory becomes available, the instance of SQL Server increases the size of its memory allocation. SQL Server can free and acquire several megabytes of memory each second, allowing it to quickly adjust to memory allocation changes.

### **Effects of min and max server memory**

The min server memory and max server memory configuration options establish upper and lower limits to the amount of memory used by the buffer pool of the Microsoft SQL Server Database Engine. The buffer pool does not immediately acquire the amount of memory specified in min server memory. The buffer pool starts with only the memory required to initialize. As the Database Engine workload increases, it keeps acquiring the memory required to support the workload. The buffer pool does not free any of the acquired memory until it reaches the amount specified in min server memory. Once min server memory is reached, the buffer pool then uses the standard algorithm to acquire and free memory as needed. The only difference is that the buffer pool never drops its memory allocation below the level specified in min server memory, and never acquires more memory than the level specified in max server memory.

The amount of memory acquired by the Database Engine is entirely dependent on the workload placed on the instance. A SQL Server instance that is not processing many requests may never reach min server memory.

If the same value is specified for both min server memory and max server memory, then once the memory allocated to the Database Engine reaches that value, the Database Engine stops dynamically freeing and acquiring memory for the buffer pool.

If an instance of SQL Server is running on a computer where other applications are frequently stopped or started, the allocation and deallocation of memory by the instance of SQL Server may slow the startup times of other applications. Also, if SQL Server is one of several server applications running on a single computer, the system administrators may need to control the amount of memory allocated to SQL Server.



In these cases, you can use the min server memory and max server memory options to control how much memory SQL Server can use.

SQL Server supports Address Windowing Extensions (AWE) allowing use of physical memory over 4 gigabytes (GB) on 32-bit versions of Microsoft Windows operating systems. Up to 64 GB of physical memory is supported. Instances of SQL Server that are running on Microsoft Windows 2000 use static AWE memory allocation, and instances that are running on Microsoft Windows Server 2003 use dynamic AWE memory allocation.

## **Buffer Management**

A buffer is an 8-KB page in memory, the same size as a data or index page. Thus, the buffer cache is divided into 8-KB pages. The buffer manager manages the functions for reading data or index pages from the database disk files into the buffer cache and writing modified pages back to disk. A page remains in the buffer cache until the buffer manager needs the buffer area to read in more data. Data is written back to disk only if it is modified. Data in the buffer cache can be modified multiple times before being written back to disk.

## **Using AWE**

Microsoft SQL Server uses the Microsoft Windows Address Windowing Extensions (AWE) API to support very large amounts of physical memory. SQL Server can access up to 64 gigabytes (GB) of memory on Microsoft Windows 2000 Server and Microsoft Windows Server 2003.

AWE is a set of extensions to the memory management functions of Windows that allow applications to address more memory than the 2-3 GB that is available through standard 32-bit addressing. AWE lets applications acquire physical memory, and then dynamically map views of the nonpaged memory to the 32-bit address space.

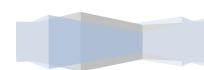
Although the 32-bit address space is limited to 4 GB, the nonpaged memory can be much larger. This enables memory-intensive applications, such as large database systems, to address more memory than can be supported in a 32-bit address space.

Before you configure the operating system for AWE, consider the following:

- AWE allows allocating physical memory over 4 GB on 32-bit architecture. AWE should be used only when available physical memory is greater than user-mode virtual address space.
- To support more than 4 GB of physical memory on 32-bit operating systems, you must add the /pae parameter to the Boot.ini file and reboot the computer. For more information, see your Windows documentation.

If there is more than 16 GB of physical memory available on a computer, the operating system requires 2 GB of virtual address space for system purposes and therefore can support only a 2 GB user mode virtual address space. For the operating system to use the memory range above 16 GB, be sure that the /3gb parameter is not in the Boot.ini file. If it is, the operating system cannot use any physical memory above 16 GB.

Memory management is a huge topic, and to cover every detail would require a whole volume in itself. My goal in this section is twofold: first, to provide enough information about how SQL Server uses its memory resources so you can determine whether memory is being managed well on your system; and second, to describe the



aspects of memory management that you have control over so you can understand when to exert that control.

By default, SQL Server 2008 manages its memory resources almost completely dynamically. When allocating memory, SQL Server must communicate constantly with the operating system, which is one of the reasons the SQLOS layer of the engine is so important.

### The Buffer Pool and the Data Cache

The main memory component in SQL Server is the buffer pool. All memory not used by another memory component remains in the buffer pool to be used as a data cache for pages read in from the database files on disk. The buffer manager manages disk I/O functions for bringing data and index pages into the data cache so data can be shared among users. When other components require memory, they can request a buffer from the buffer pool. A buffer is a page in memory that's the same size as a data or index page. You can think of it as a page frame that can hold one page from a database. Most of the buffers taken from the buffer pool for other memory components go to other kinds of memory caches, the largest of which is typically the cache for procedure and query plans, which are usually called the *procedure cache*.

Occasionally, SQL Server must request contiguous memory in larger blocks than the 8-KB pages that the buffer pool can provide so memory must be allocated from outside the buffer pool. Use of large memory blocks is typically kept to minimum, so direct calls to the operating system account for a small fraction of SQL Server memory usage.



## Thread and Task Architecture

### Overview

Threads are an operating system feature that lets application logic be separated into several concurrent execution paths. This feature is useful when complex applications have many tasks that can be performed at the same time.

When an operating system executes an instance of an application, it creates a unit called a process to manage the instance. The process has a thread of execution. This is the series of programming instructions performed by the application code. For example, if a simple application has a single set of instructions that can be performed serially; there is just one execution path or thread through the application. More complex applications may have several tasks that can be performed in tandem, instead of serially. The application can do this by starting separate processes for each task. However, starting a process is a resource-intensive operation. Instead, an application can start separate threads. These are relatively less resource-intensive. Additionally, each thread can be scheduled for execution independently from the other threads associated with a process.

Threads allow complex applications to make more effective use of a CPU, even on computers that have a single CPU. With one CPU, only one thread can execute at a time. If one thread executes a long-running operation that does not use the CPU, such as a disk read or write, another one of the threads can execute until the first operation is completed. By being able to execute threads while other threads are waiting for an operation to be completed, an application can maximize its use of the CPU. This is especially true for multi-user, disk I/O intensive applications such as a database server. Computers that have multiple microprocessors or CPUs can execute one thread per CPU at the same time. For example, if a computer has eight CPUs, it can execute eight threads at the same time.

### Allocating Threads to a CPU

By default, each instance of SQL Server starts each thread. The operating system then assigns each thread to a specific CPU. The operating system distributes threads from instances of SQL Server evenly among the microprocessors, or CPUs on a computer. Sometimes, the operating system can also move a thread from one CPU with heavy usage to another CPU.

SQL Server administrators can use the affinity mask configuration option to exclude one or more CPUs from being eligible to run threads from a specific instance of SQL Server. The affinity mask value specifies a bit pattern that indicates the CPUs that are used to run threads from that instance of SQL Server. For example, the affinity mask value 13 represents the bit pattern 1101. On a computer that has four CPUs, this indicates that threads from that instance of SQL Server can be scheduled on CPUs 0, 2, and 3, but not on CPU 1. If affinity mask is specified, the instance of SQL Server allocates threads evenly among the CPUs that have not been masked off. Another effect of affinity mask is that the operating system does not move threads from one CPU to another. However, affinity mask is rarely used. Most systems obtain optimal performance by letting the operating system schedule the threads among the available CPUs.



## Using the lightweight pooling Option

The overhead involved in switching thread contexts is not very large. Most instances of SQL Server will not see any performance differences between setting the lightweight pooling option to 0 or 1. The only instances of SQL Server that might benefit from lightweight pooling are those that run on a computer having the following characteristics:

- A large multi-CPU server.
- All the CPUs are running near maximum capacity.
- There is a high level of context switching.

These systems may see a small increase in performance if the lightweight pooling value is set to 1.

We do not recommend that you use fiber mode scheduling for routine operation. This is because it can decrease performance by inhibiting the regular benefits of context switching, and because some components of SQL Server cannot function correctly in fiber mode

## Thread and Fiber Execution

Microsoft Windows uses a numeric priority system that ranges from 1 through 31 to schedule threads for execution. Zero is reserved for operating system use. When several threads are waiting to execute, Windows dispatches the thread with the highest priority.

By default, each instance of SQL Server is a priority of 7, which is referred to as the normal priority. This default gives SQL Server threads a high enough priority to obtain sufficient CPU resources without adversely affecting other applications.

The priority boost configuration option can be used to increase the priority of the threads from an instance of SQL Server to 13. This is referred to as high priority. This setting gives SQL Server threads a higher priority than most other applications. Thus, SQL Server threads will generally be dispatched whenever they are ready to run and will not be pre-empted by threads from other applications. This can improve performance when a server is running only instances of SQL Server and no other applications. However, if a memory-intensive operation occurs in SQL Server, however, other applications are not likely to have a high-enough priority to pre-empt the SQL Server thread.

If you are running multiple instances of SQL Server on a computer, and turn on priority boost for only some of the instances, the performance of any instances running at normal priority can be adversely affected. Also, the performance of other applications and components on the server can decline if priority boost is turned on. Therefore, it should only be used under tightly controlled conditions.



## Hot Add CPU

Hot add CPU is the ability to dynamically add CPUs to a running system. Adding CPUs can occur physically by adding new hardware, logically by online hardware partitioning, or virtually through a virtualization layer. Starting with SQL Server 2008, SQL Server supports hot add CPU.

Requirements for hot add CPU:

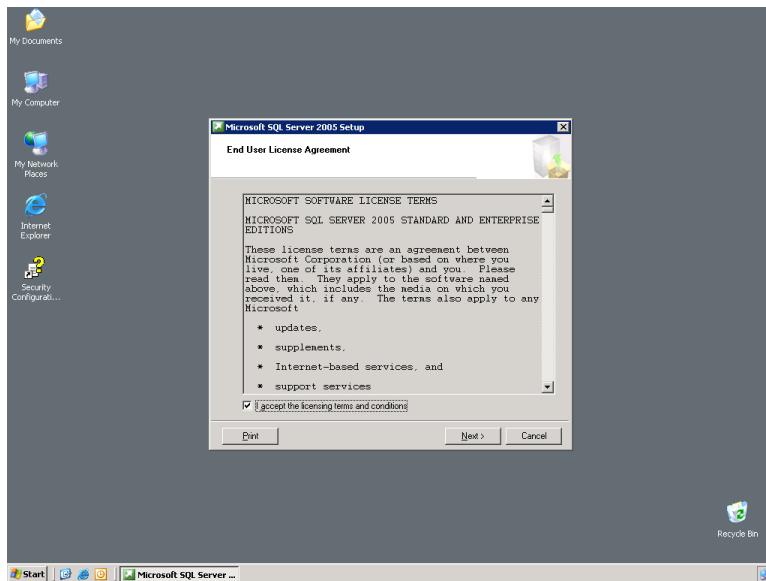
- Requires hardware that supports hot add CPU.
- Requires the 64-bit edition of Windows Server 2008 Datacenter or the Windows Server 2008 Enterprise Edition for Itanium-Based Systems operating system.
- Requires SQL Server Enterprise.

SQL Server does not automatically start to use CPUs after they are added. This prevents SQL Server from using CPUs that might be added for some other purpose. After adding CPUs, execute the RECONFIGURE statement, so that SQL Server will recognize the new CPUs as available resources.



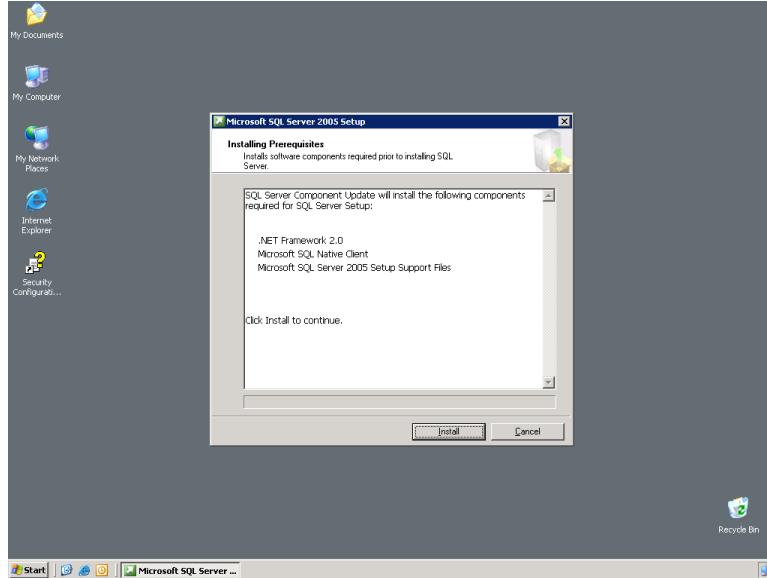
## Installing SQL Server 2005

1. To begin the installation process, insert the SQL Server 2005 DVD into your DVD drive. If the autorun feature on your DVD drive does not launch the installation program, navigate to the root of the DVD and launch **splash.hta**.
2. From the autorun dialog, click **Run the SQL Server Installation Wizard**.
3. On the **End User License Agreement** page, read the license agreement, and then select the check box to accept the licensing terms and conditions. Accepting the license agreement activates the **Next** button. To continue, click **Next**.

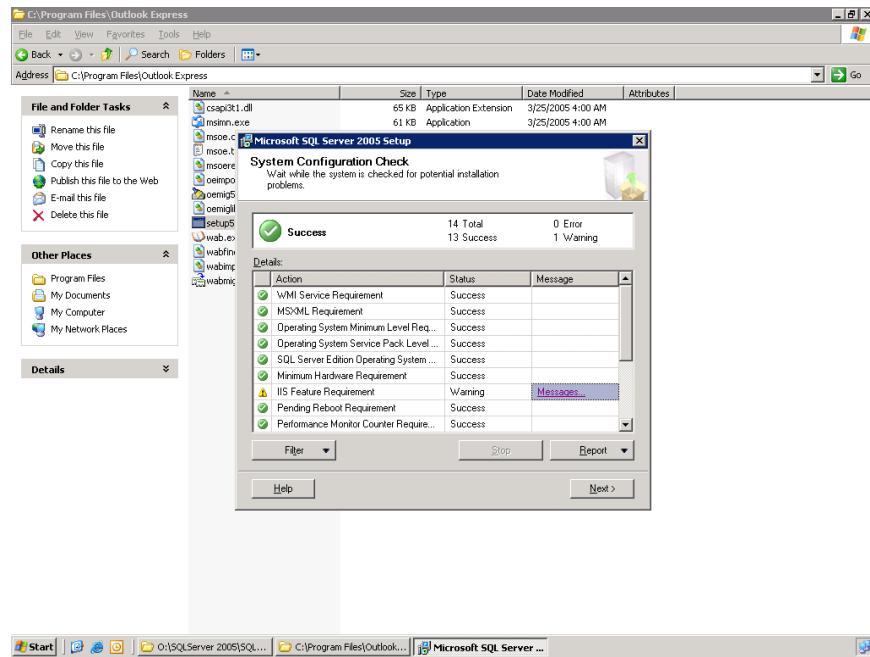


4. On the **SQL Server Component Update** page, Setup installs software required for SQL Server 2005. For more information about component requirements, To begin the component update process, click **Install**. To continue after the update completes, click **Finish**.



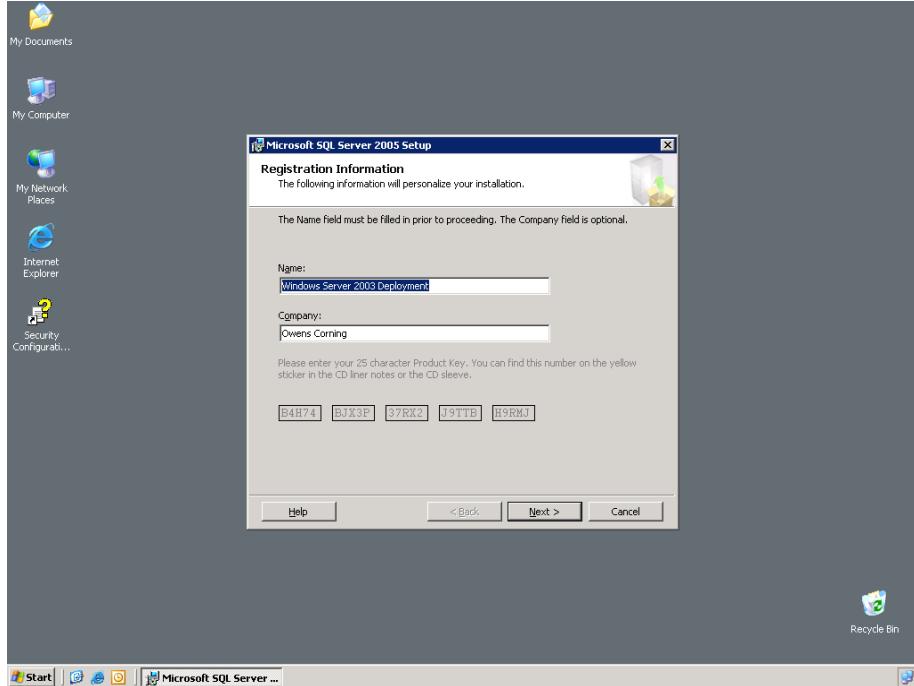


5. On the **Welcome** page of the SQL Server Installation Wizard, click **Next** to continue.
6. On the **System Configuration Check** (SCC) page, the installation computer is scanned for conditions that may block Setup. click **Continue**.

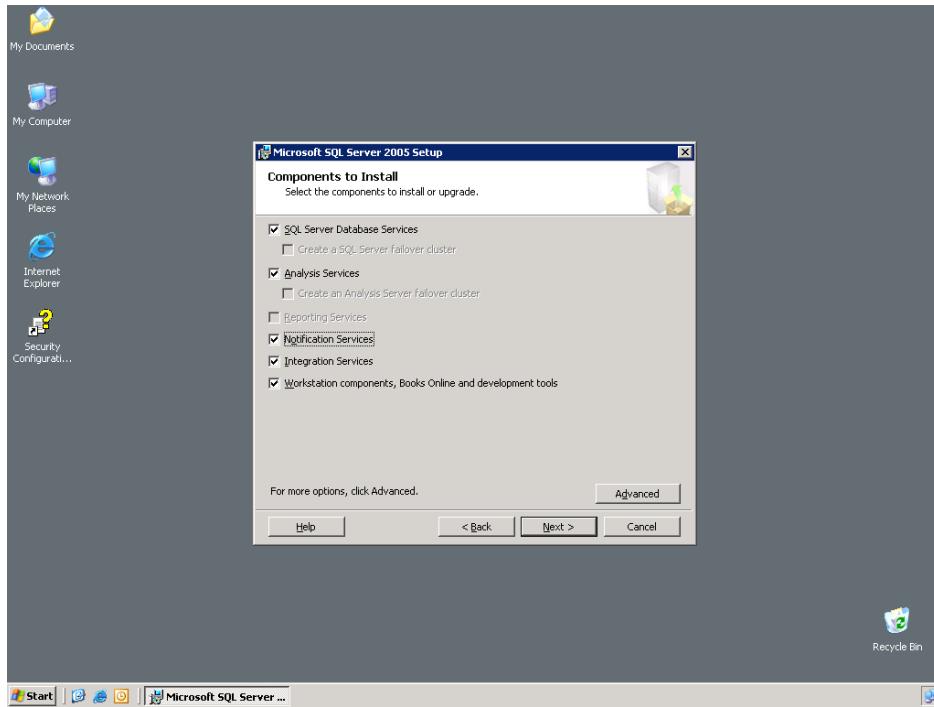


7. On the **Registration Information** page, enter information in the **Name** and **Company** text boxes. To continue, click **Next**.

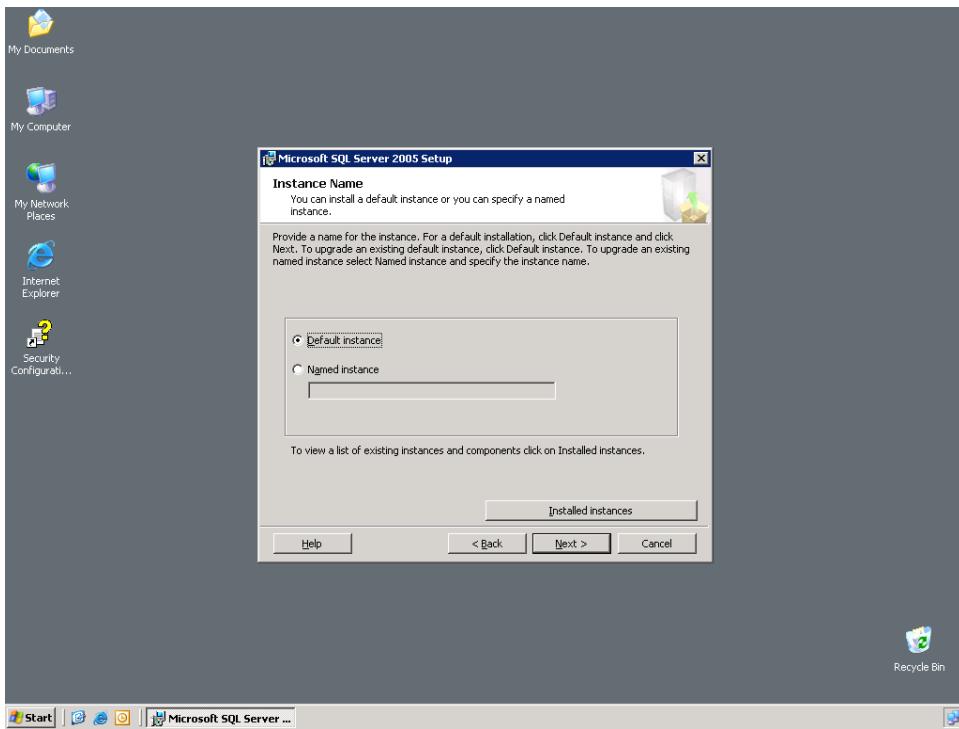




8. On the **Components to Install** page, select the components for your installation. A description for each component group appears in the **Components to be Installed** pane when you select it. You can select any combination of check boxes. When you select SQL Server or Analysis Services, if Setup detects that you are installing to a virtual server, the **Install as a Virtual Server** check box is enabled. You must select this option to install a failover cluster.



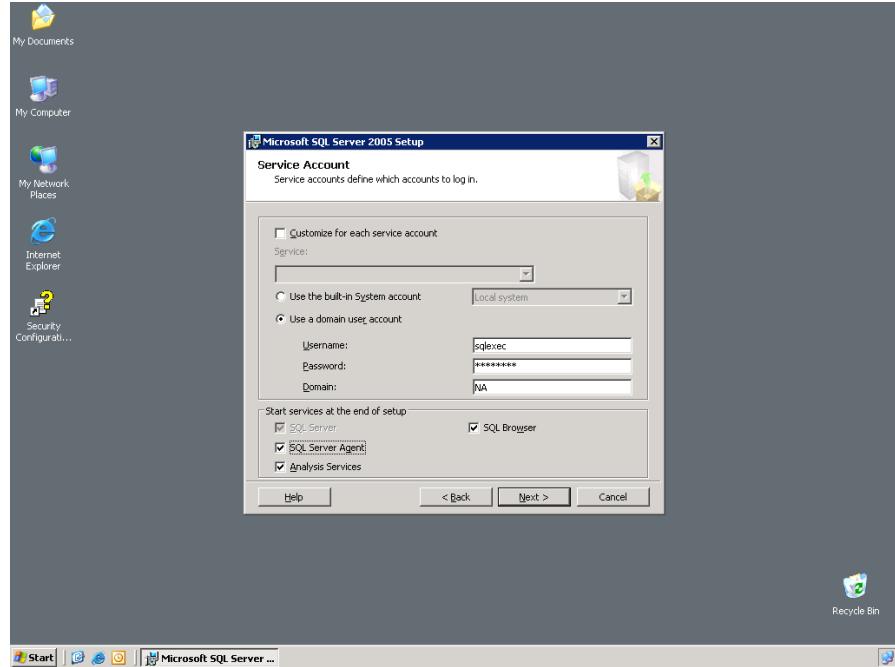
9. To install individual components, click **Advanced**. Otherwise, click **Next** to continue.
10. If you clicked **Advanced** on the previous page, the **Feature Selection** page is displayed. On the **Feature Selection** page, select the program features to install using the drop-down boxes. To install components to a custom directory, select the feature and then click **Browse**. To continue when your feature selections are complete, click **Next**.
11. On the **Instance Name** page, select a default or named instance for your installation. If a default or named instance is already installed, and you select the existing instance for your installation, Setup upgrades it and provides you the option to install additional components. To install a new default instance, there must not be a default instance on the computer. To install a new named instance, click **Named Instance** and then type a unique instance name in the space provided.



12. On the **Service Account** page, specify the user name, password, and domain name or SQL Server service accounts. You can use one account for all of the services.

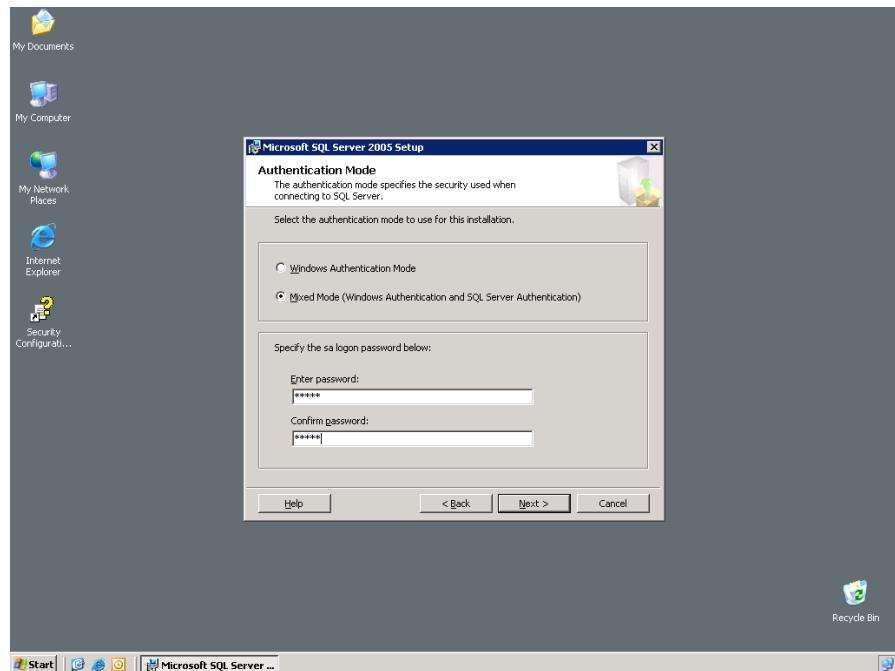
Optionally, you can specify an individual account for each service. To specify an individual account for each service, select **Customize for each service account**, select a service name from the drop-down box, and then provide login credentials for the service. To proceed, click **Next**.





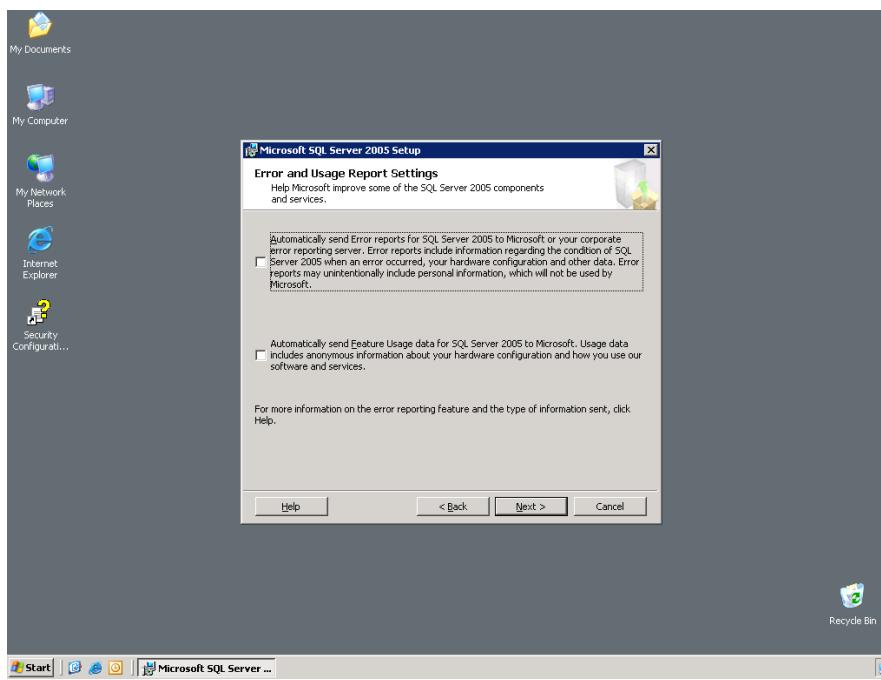
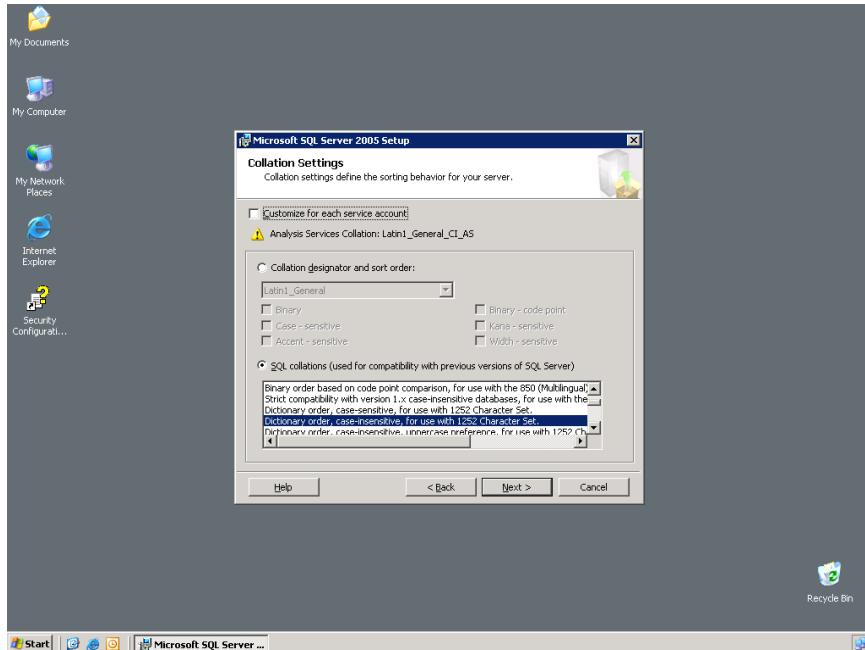
13. On the **Authentication Mode** page, choose the authentication mode to use for your SQL Server installation.

If you select Windows Authentication, Setup creates an **sa** account, which is disabled by default. Enter and confirm the system administrator (**sa**) login when you choose Mixed Mode Authentication. Passwords are the first line of defense against intruders, so setting strong passwords is essential to the security of your system. Never set a blank or weak **sa** password.

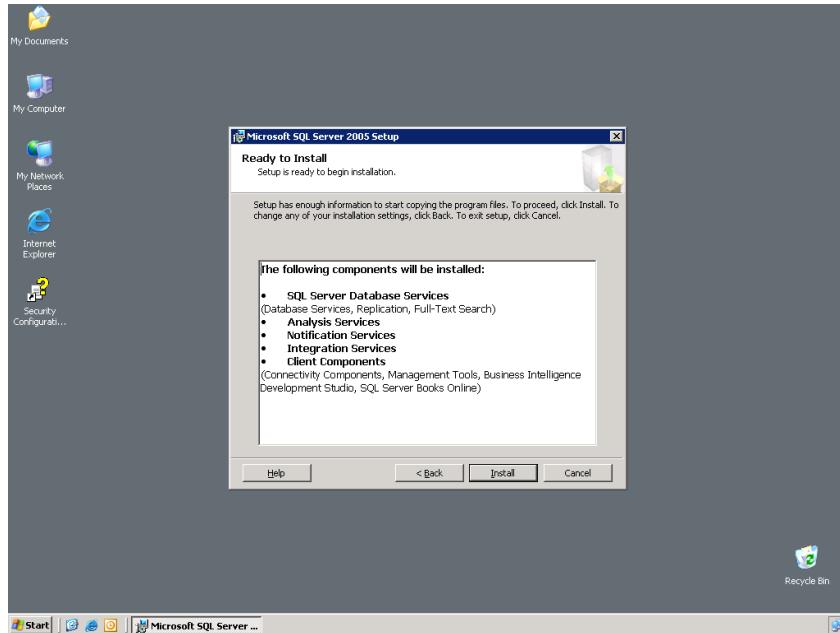


To set separate collation settings for SQL Server and Analysis Services, select the **Customize for each service account** check box. After you select the check box, a drop-down selection box appears. Select a service from the drop-down selection box and then assign its collation. Repeat for each service. To proceed, click **Next**.

14. On the **Error Reporting** page, optionally clear the check box to disable error reporting. For more information about error reporting functionality, click **Help** at the bottom of the page. To proceed, click **Next**.

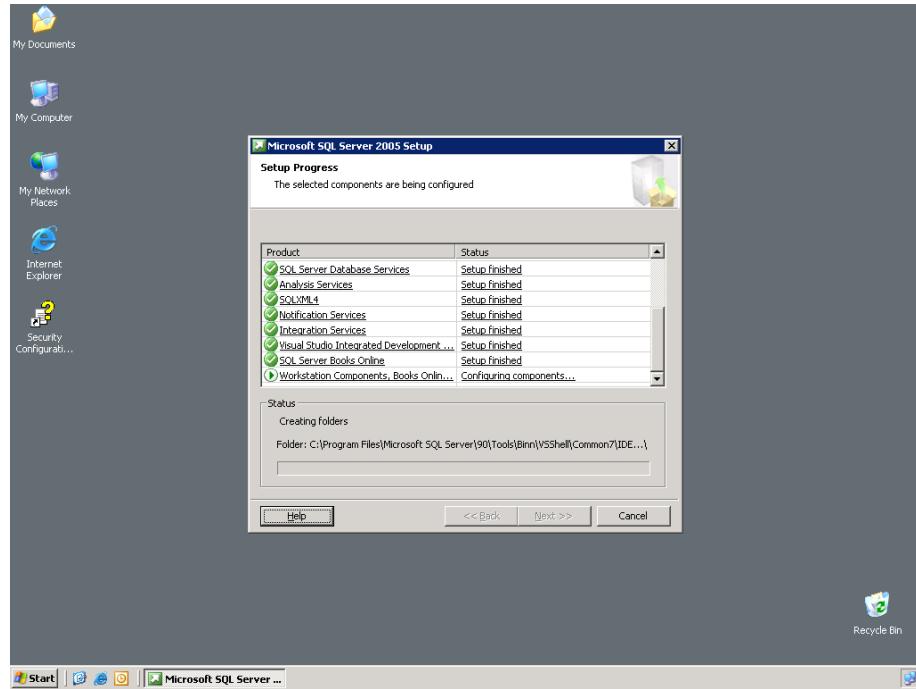


15. On the **Ready to Install** page, review the summary of features and components for your SQL Server installation. To proceed, click **Install**.



16. On the **Installation Progress** page, you can monitor installation progress as Setup proceeds. To view the log file for a component during installation, click the product or status name on the **Installation Progress** page.
17. On the **Completing the Microsoft SQL Server Installation Wizard** page, you can view the Setup summary log by clicking the link provided on this page. To exit the SQL Server Installation Wizard, click **Finish**.





18. If you are instructed to restart the computer, do so now. It is important to read the message from the Setup program when you are done with installation. Failure to restart the computer may cause failures when you run the Setup program in the future.



## SQL Server 2008 Installation

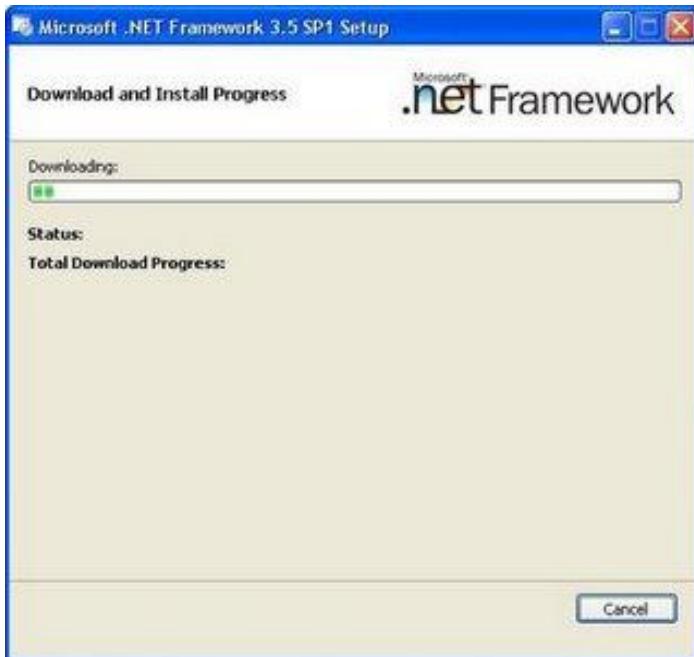
### Pre-Requisites

- You have planned your Instance name if it is a named Instance
- You have planned all the Data Directory. Ie. Where to keep System Databases, TempDB, and User Databases.(if not you can go by Default but it is not generally recommended).
- You have planned Startup Account for all the services. (if not you can go by Default but it is not generally recommended.)
- You have a list of features that you want to install. Like if you are not going to use File stream no need to configure that during installation

### Step 1: Insert the DVD

The system will automatically detect the system configuration and it install the .Net Framework and Windows Installer 4.5 if required.

My installation Screen is as follows :-

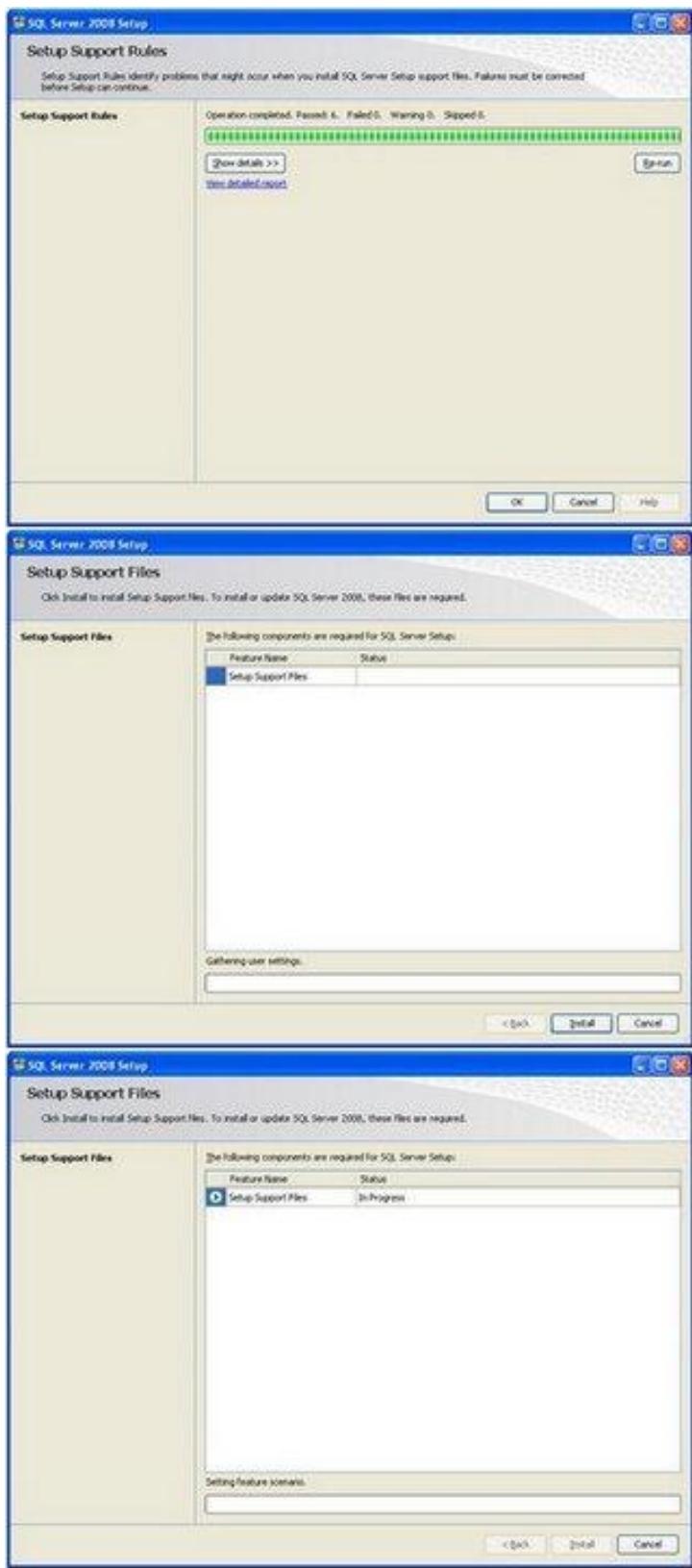


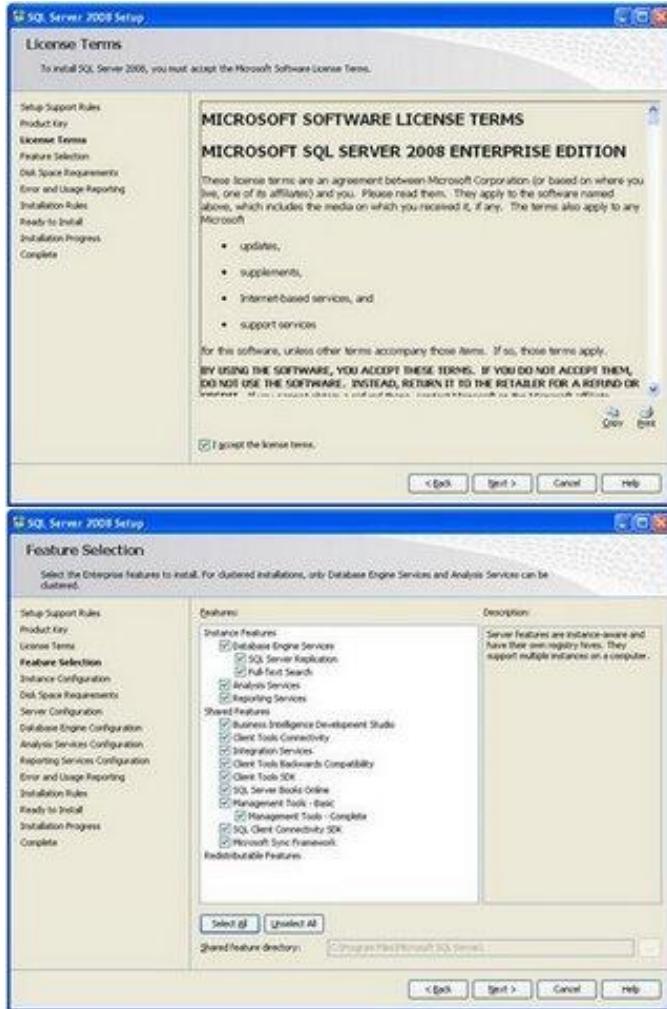


**Step 2 :** System has installed .Net Framework 3.5 and rebooted the system. Next step is to install SQL Server 2008.

In "SQL Server Installation Center" , navigate to "Installation" -- >> click "New Installation or Add Features to an Existing Installation."





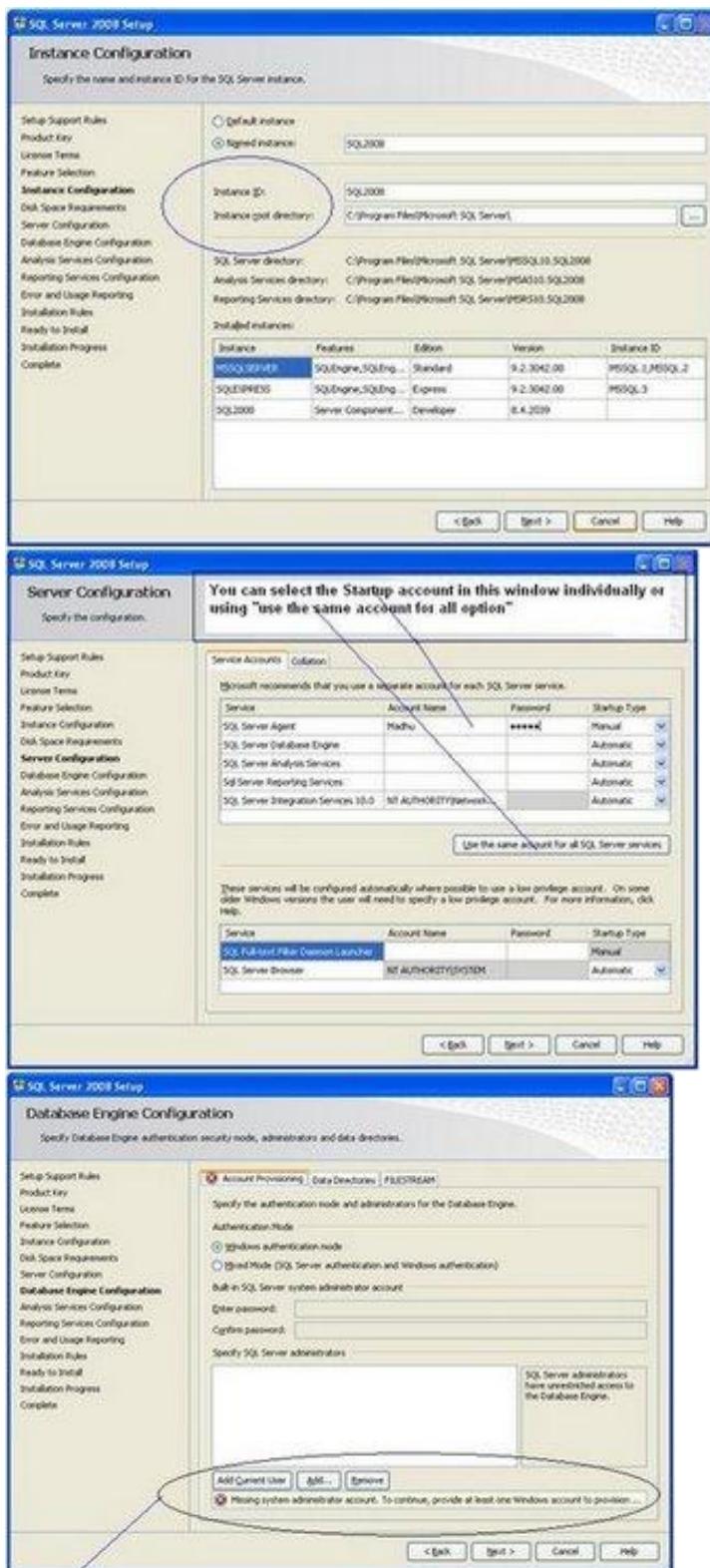


In the Instance Configuration Screen you need to select the instance type (default /named) and other instance related setting. The one new thing in SQL Server 2008 is InstanceID.

#### [Instance Configuration](#)

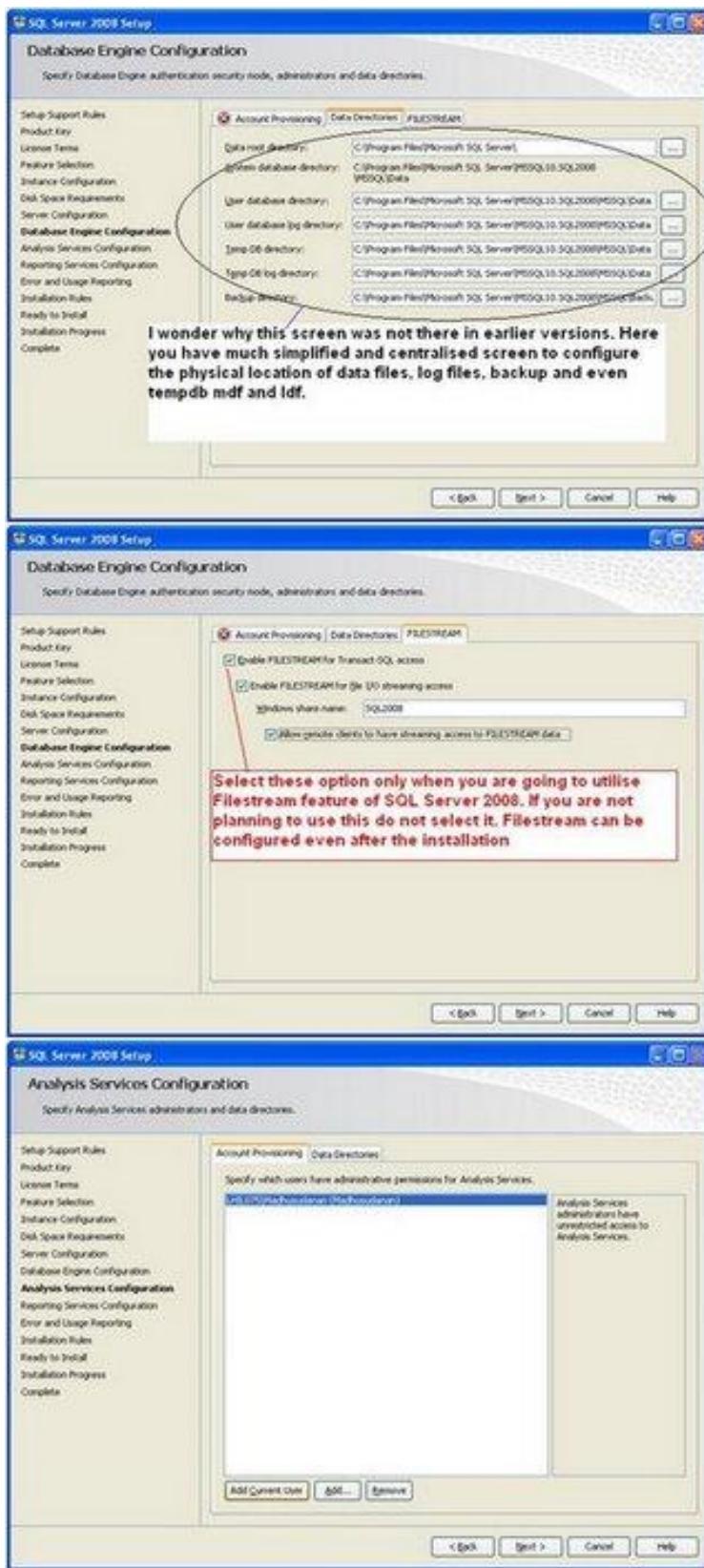
**Instance ID** - By default, the instance name is used as the Instance ID. This is used to identify installation directories and registry keys for your instance of SQL Server. This is the case for default instances and named instances. For a default instance, the instance name and instance ID would be MSSQLSERVER. To use a non-default instance ID, specify it in the Instance ID field.

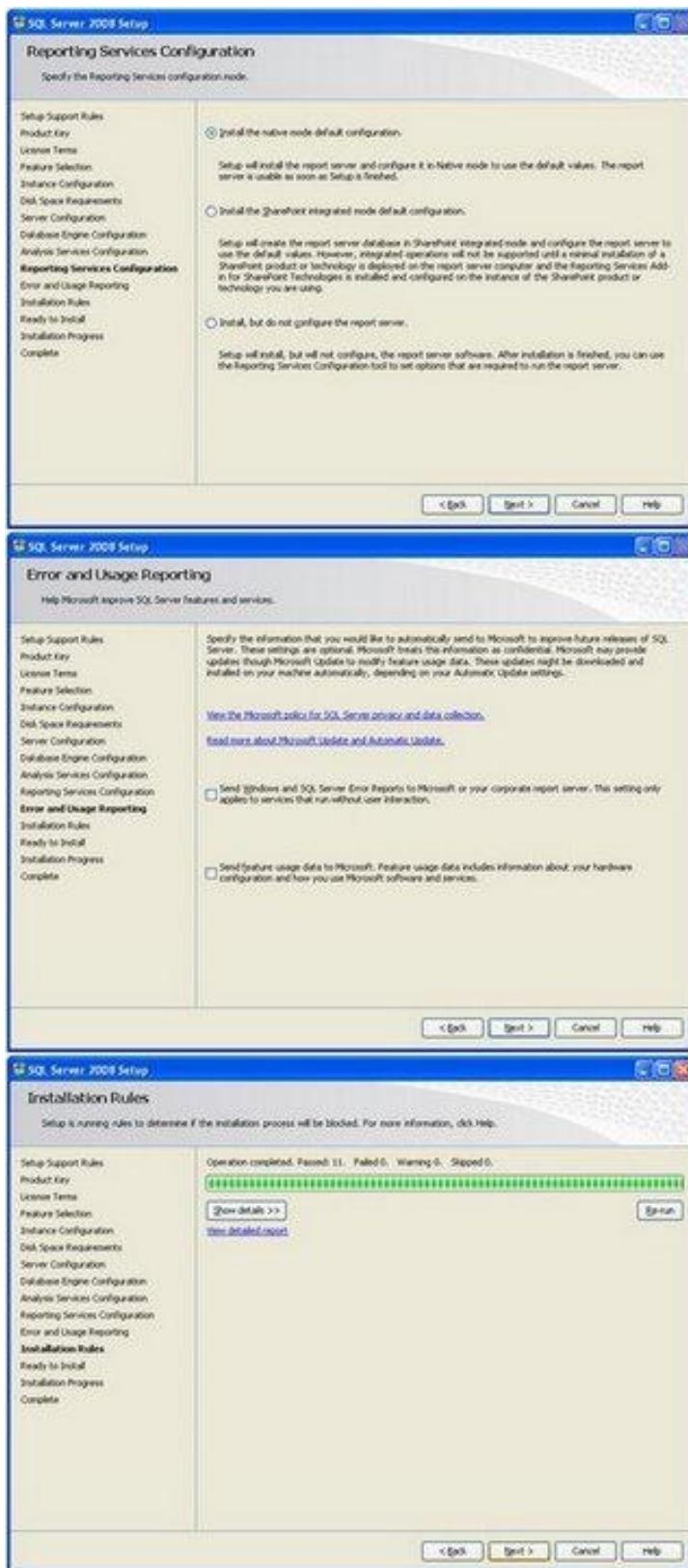


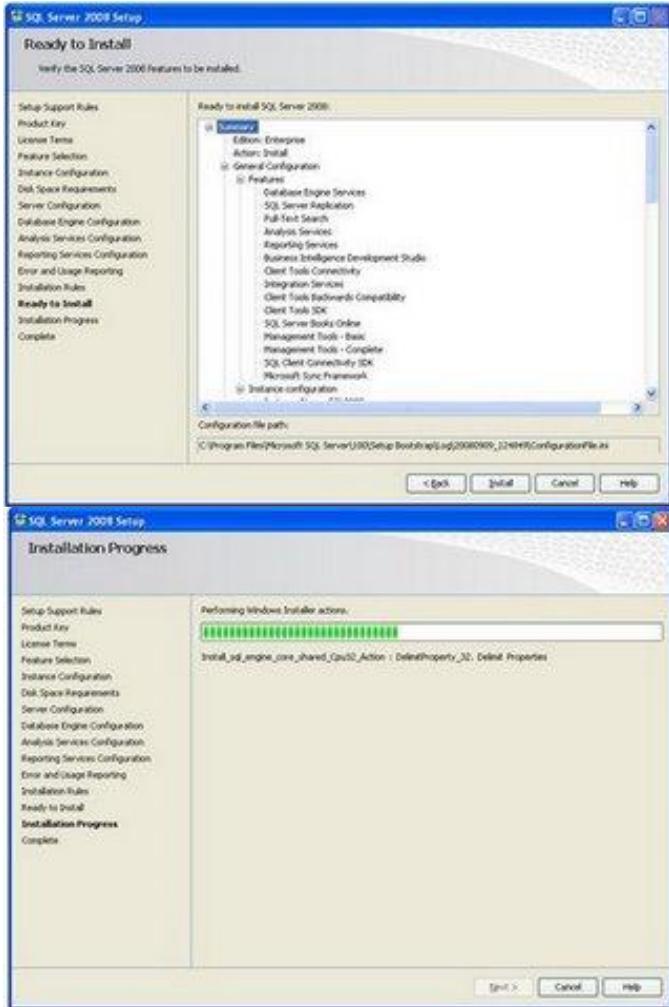


**SQl Server Administrators — You must specify at least one system administrator for the instance of SQL Server. To add the account under which SQL Server Setup is running, click Add Current User. To add or remove accounts from the list of system administrators, click Add or Remove, and then edit the list of users, groups, or computers that will have administrator privileges for the instance of SQL Server**









### Best Practices on Installation:

1. Always fully document installs so that your SQL Server instances can easily be reproduced in an emergency.
2. If possible, install and configure all of your SQL Server instances consistently, following an agreed-upon organization standard.
3. Don't install SQL Server services you don't use, such as Microsoft Full-Text Search, Notification Services, or Analysis Services.
4. For best performance of SQL Server running under Windows, turn off any operating system services that aren't needed.
5. For optimum SQL Server performance, you want to dedicate your physical servers to only running a single instance of SQL Server, no other applications.
6. For best I/O performance, locate the database files (.mdf) and log files (.ldf) on separate arrays on your server to isolate potentially conflicting reads and writes.
7. If tempdb will be used heavily, also put it on its own separate array.
8. Do not install SQL Server on a domain controller.
9. Be sure that SQL Server is installed on an NTFS partition.
10. Don't use NTFS data file encryption (EFS) and compression on SQL Server database and log files.



## Case Study/Practical Troubleshooting

### Failed Installation/Moving System Databases

Accidently you choose program files as well as data files in the same directory, later you realized that that mistake. You are planning to un-install and reinstall the engine with specified settings. Here is the solution how to move system databases from SQL default location to another location.

#### **Tasks for moving system databases:**

##### **1. Moving tempdb databases.**

- Execute the script below.

```
USE master;
GO
alter database tempdb MODIFY FILE (NAME = tempdev,FILENAME='NEW PATH');
GO
alter database tempdb MODIFY FILE (NAME = templog,FILENAME='NEW PATH');
GO
```

#### **Example**

```
SAGARSSYS\SQL...QLQuery1.sql* Object Explorer Details
USE master;
GO
alter database tempdb MODIFY FILE (NAME = tempdev,FILENAME='C:\Temp\tempdev.mdf');
GO
alter database tempdb MODIFY FILE (NAME = templog,FILENAME='C:\Temp\templog.ldf');
GO
```

The file "tempdev" has been modified in the system catalog. The new path will be used the next time the database is opened.  
The file "templog" has been modified in the system catalog. The new path will be used the next time the database is opened.

- Restart services.
- Confirm path of database files using the query USE tempdb; SELECT physical\_name from sys.database\_files.

##### **2. Moving model and msdb databases.**

- Execute the script below.

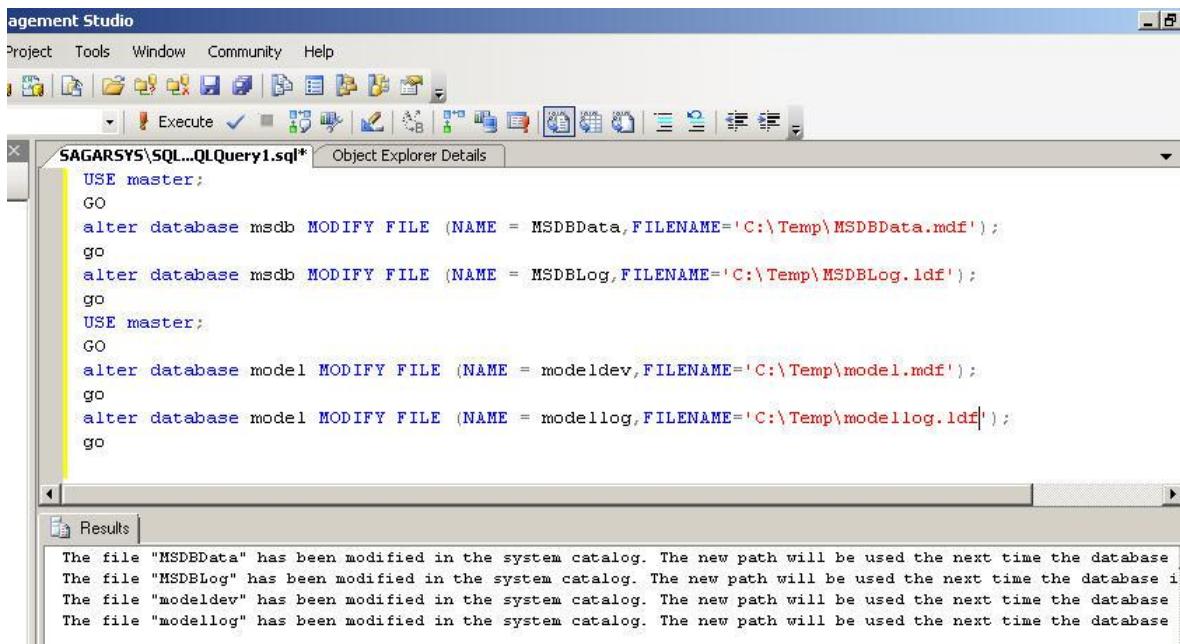
```
USE master;
GO
alter database msdb MODIFY FILE (NAME = MSDBData,FILENAME='NEW PATH');
```



```
alter database msdb MODIFY FILE (NAME = MSDBLog,FILENAME='NEW PATH');
go
```

```
USE master;
GO
alter database model MODIFY FILE (NAME = modeldev,FILENAME='NEW PATH');
go
alter database model MODIFY FILE (NAME = modellog,FILENAME='NEW PATH');
go
```

### Example



The screenshot shows the SQL Server Management Studio interface. In the Object Explorer Details pane, a query named 'SAGARSSYS\SQL...QLQuery1.sql\*' is open. The code within the query is:

```
USE master;
GO
alter database msdb MODIFY FILE (NAME = MSDBData,FILENAME='C:\Temp\MSDBData.mdf');
go
alter database msdb MODIFY FILE (NAME = MSDBLog,FILENAME='C:\Temp\MSDBLog.ldf');
go
USE master;
GO
alter database model MODIFY FILE (NAME = modeldev,FILENAME='C:\Temp\model1.mdf');
go
alter database model MODIFY FILE (NAME = modellog,FILENAME='C:\Temp\modellog.ldf');
go
```

In the Results pane, the output of the query is displayed, showing four messages indicating file modifications:

```
The file "MSDBData" has been modified in the system catalog. The new path will be used the next time the database
The file "MSDBLog" has been modified in the system catalog. The new path will be used the next time the database i
The file "modeldev" has been modified in the system catalog. The new path will be used the next time the database
The file "modellog" has been modified in the system catalog. The new path will be used the next time the database
```

- b.) Stop services
- c.) Copy the files to the new location
- d.) Restart services.
- e.) Confirm path of database files using the below query.

```
USE msdb;
```

```
SELECT physical_name from sys.database_files
```

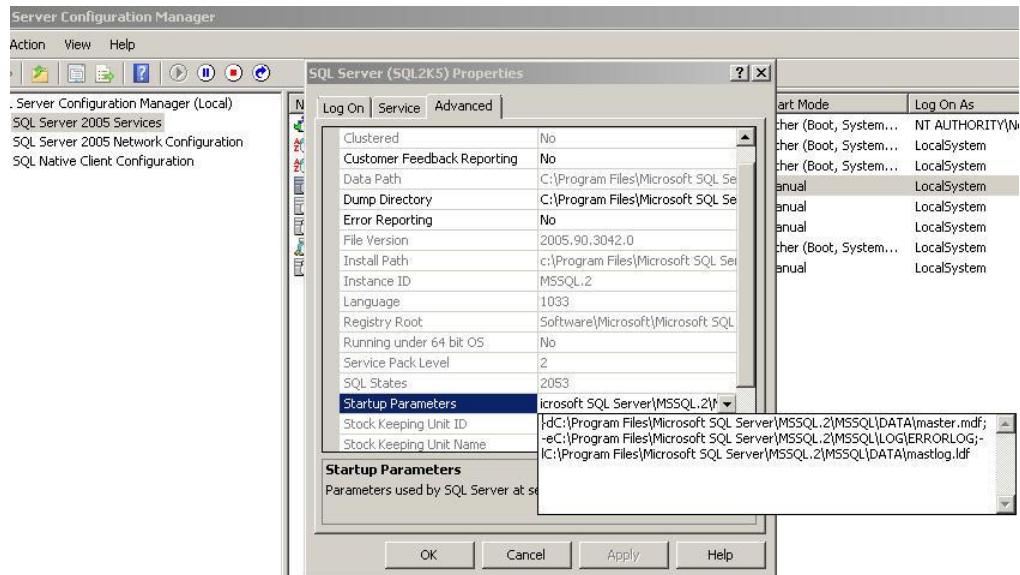
```
USE model;
```

```
SELECT physical_name from sys.database_files
```

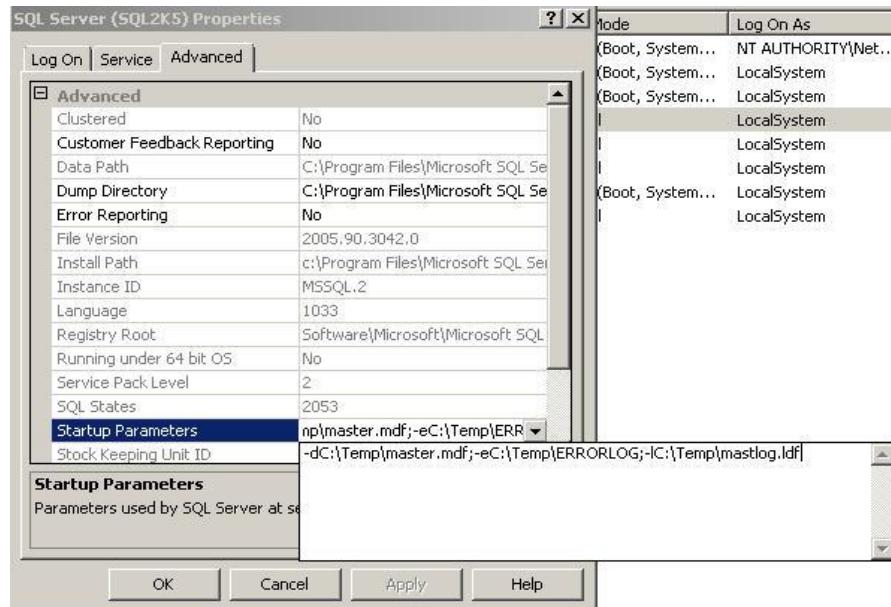
### 3.) Moving master database:

- a.) Edit the startup parameters to reflect the new path for -d, -l and -e parameters.





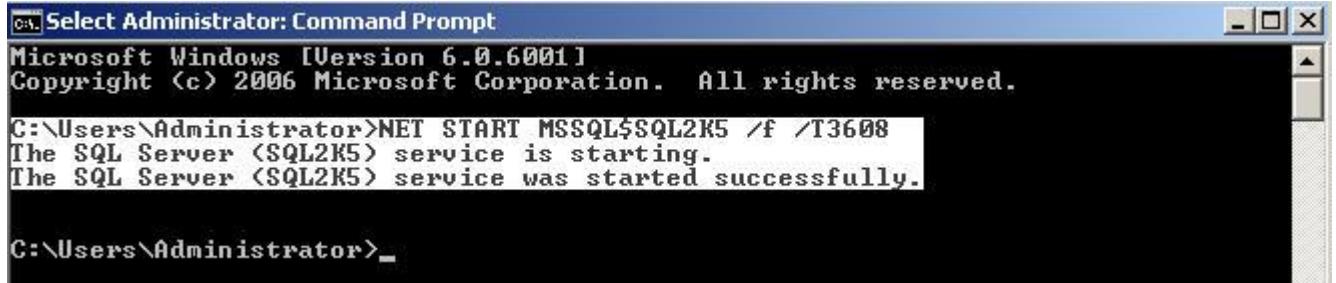
## Example



- b.) Stop the services.
- c.) Move the master and resource database files to the new location
- d.) Start the services using NET START MSSQLSERVER /f /T3608 (\*MSSQLSERVER is for default instance, if you have installed named instance then you need to use NET START MSSQL\$instancename /f /T3608)

## Example





```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation. All rights reserved.

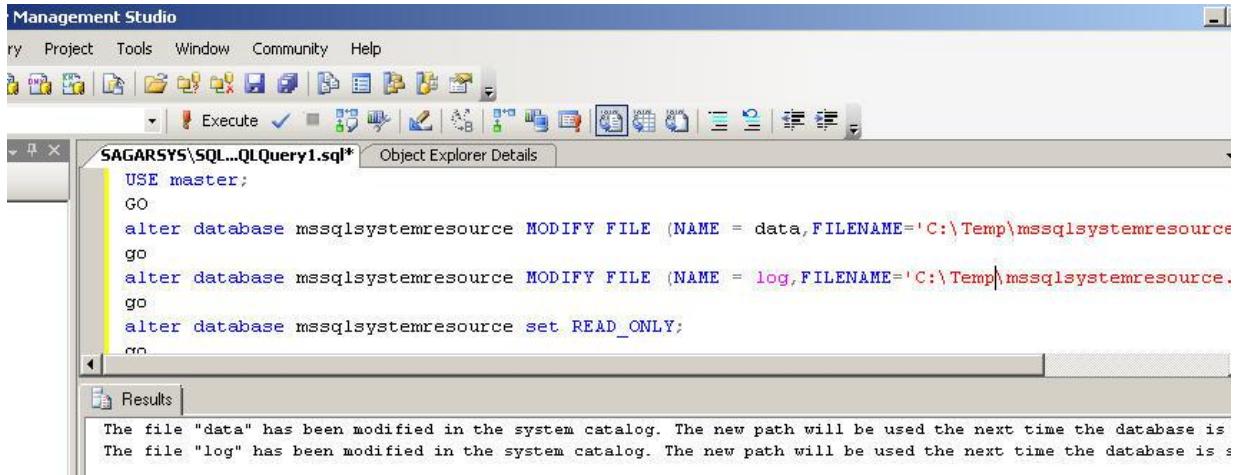
C:\Users\Administrator>NET START MSSQL$SQL2K5 /f /T3608
The SQL Server (SQL2K5) service is starting.
The SQL Server (SQL2K5) service was started successfully.

C:\Users\Administrator>_
```

f.) Execute the script given below from sqlcmd

```
USE master;
GO
alter database mssqlsystemresource MODIFY FILE (NAME = data,FILENAME='NEW
PATH\mssqlsystemresource.mdf');
go
alter database mssqlsystemresource MODIFY FILE (NAME = log,FILENAME='NEW
PATH\mssqlsystemresource.ldf');
go
alter database mssqlsystemresource set READ_ONLY;
go
```

### Example



g.) Stop the services

h.) Start sql services.

i.) Confirm if the data files and log files reside on desired path using the query  
SELECT physical\_name from sys.master\_files.



## Upgrading the SQL server

### Upgrading By applying service pack

SQL Server 2008 Service Pack installation process, which differs quite a bit from SQL Server 2000 and SQL Server 2005 installations.

It is recommended that before applying any service pack on SQL Server, database administrators should take a full backup of all the user and system databases including the [Resource](#) database which was first introduced in SQL Server 2005. The **Resource** database is a read-only database that contains all the system objects that are included with SQL Server.

Database Administrators should read the [readme.txt](#) file which comes with Service Pack Installation before applying the Service Pack. However, Database Administrators should first apply the service pack on development and test servers before applying on production servers. If there are no issues reported in the development and test Environments after testing your applications, then you can apply them on the production servers. Here are the essential steps that can make your SQL Server 2008 Service Pack (SP1) installation as smooth as possible:

1. Development environments:
  1. Issue a full backup of all user and system databases including the Resource database.
  2. Take note of all the Startup parameters, Memory Usage, CPU Usage etc.
  3. Install the service pack on development SQL Servers.
  4. Create a backup and restore plan with the steps for "**What to do if application is not working properly after installing the new Service Pack?**"
2. Test environments:
  1. Issue a full backup of all user and system databases including the Resource database.
  2. Take note of all the Startup parameters, Memory Usage, CPU Usage etc
  3. Install the service pack on test SQL Servers.
  4. Conduct testing for administrative process as well as coordinate testing with the Development and QA Teams to ensure the application is performing as expected.
  5. Test the rollback plan.
3. Production environments:
  1. Plan for a scheduled downtime on the Production Servers as it takes approximately 30 minutes to apply the service pack on SQL Server 2008.
  2. Issue a full backup of all user and system databases including the Resource database.
  3. Take note of all the Startup parameters, Memory Usage, CPU Usage etc
  4. Install the service pack on test SQL Servers.
  5. Validate the application is working properly.

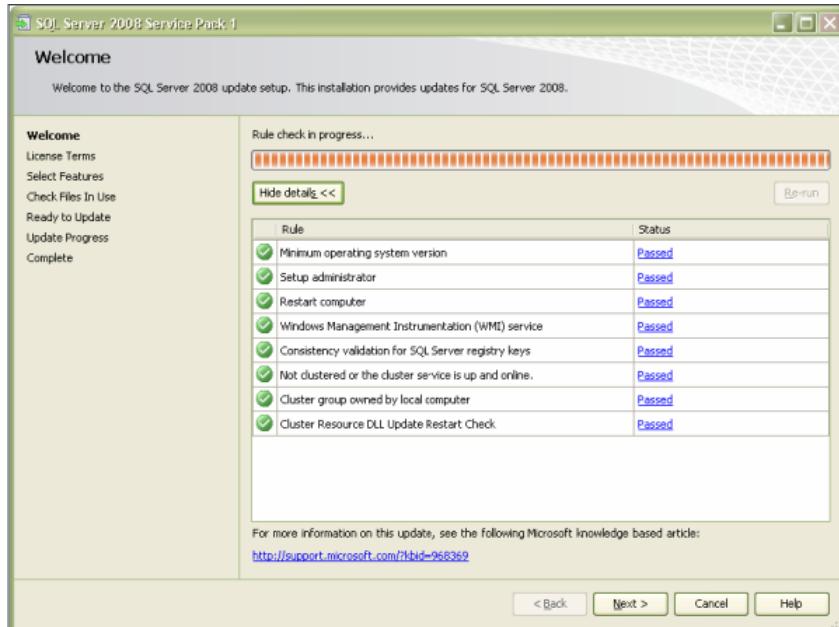


### Steps to Install SQL Server 2008 Service Pack 1 (SP1)

1. Download SQL Server 2008 Service Pack 1 (SP1) from the following [link](#).
2. Double Click **Setup.exe** to extract the Service Pack installation files from the setup.



3. Once setup files are extracted go to the folder where the files are extracted and click **Setup.exe**. This will open up SQL Server 2008 Service Pack 1 screen as shown in the snippet below. In the **Welcome** screen the SQL Server 2008 Setup program will check for few rules before applying the Service Packs. If any of the rules are failing, the installation will not continue further. Hence, you need to fix those issues before continuing with the installation.



46



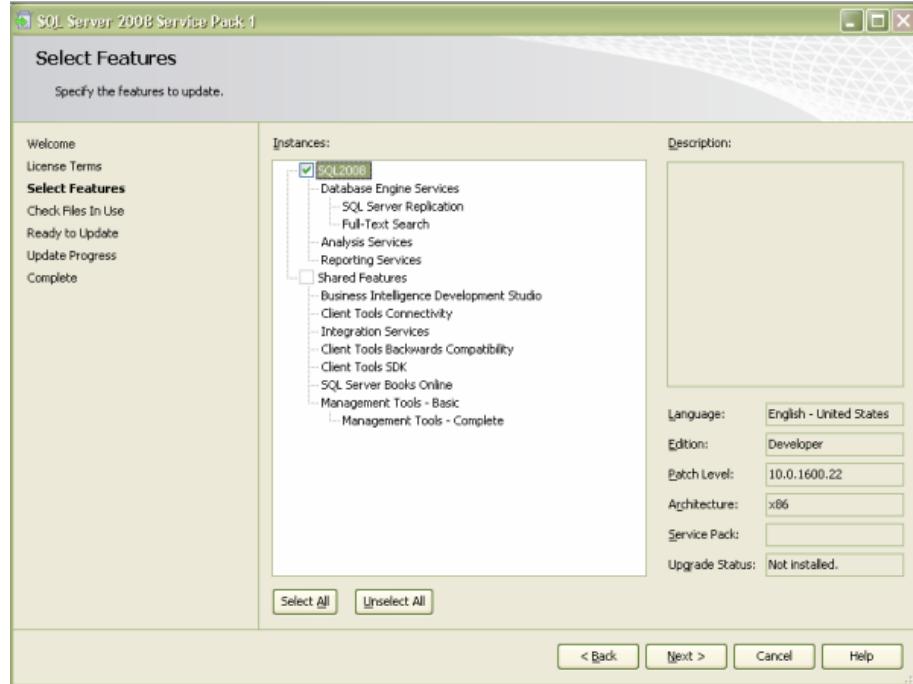
4. In **License Terms** screen, read the license agreement and then select the check box at the bottom of the page to accept the licensing terms and conditions of the product. Click Next to continue with the installation.



5. In **Select Features** screen, select the SQL Server Instance and select the features of SQL Server 2008 which need to be upgraded. A brief description about each feature is shown in the right side panel along with the **Language, Edition, Patch Level, Architecture, Service Pack** and **Upgrade Status** when each of the features is selected. The different components which are available for upgrade within SQL Server 2008 are mentioned below:

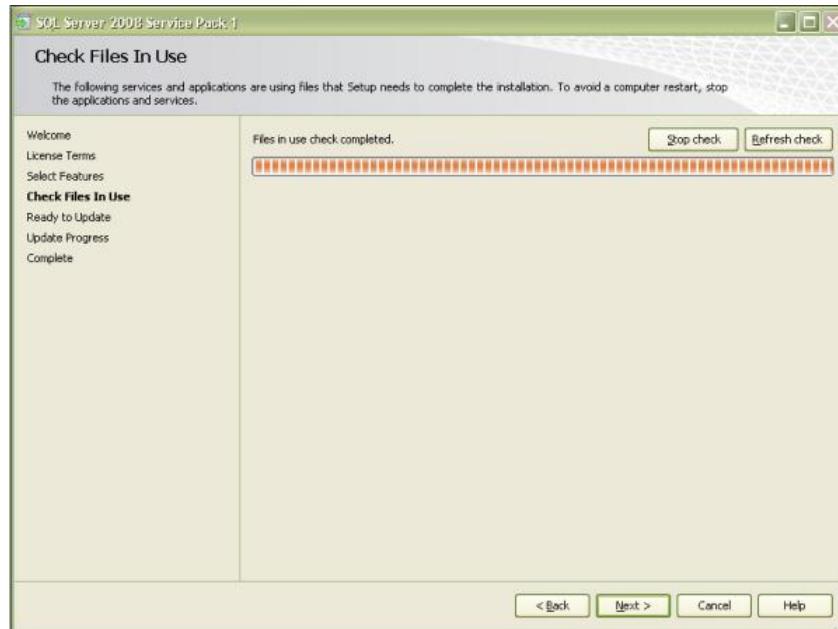
Database Engine Services  
SQL Server Replication  
Full-Text Replication  
Analysis Services  
Reporting Services  
Shared Features  
Business Intelligence Development Studio  
Client Tools Connectivity  
Integration Services  
Client Tools Backwards Compatibility  
Client Tools SDK  
SQL Server Books Online  
Management Tools – Basic  
Management Tools - Complete



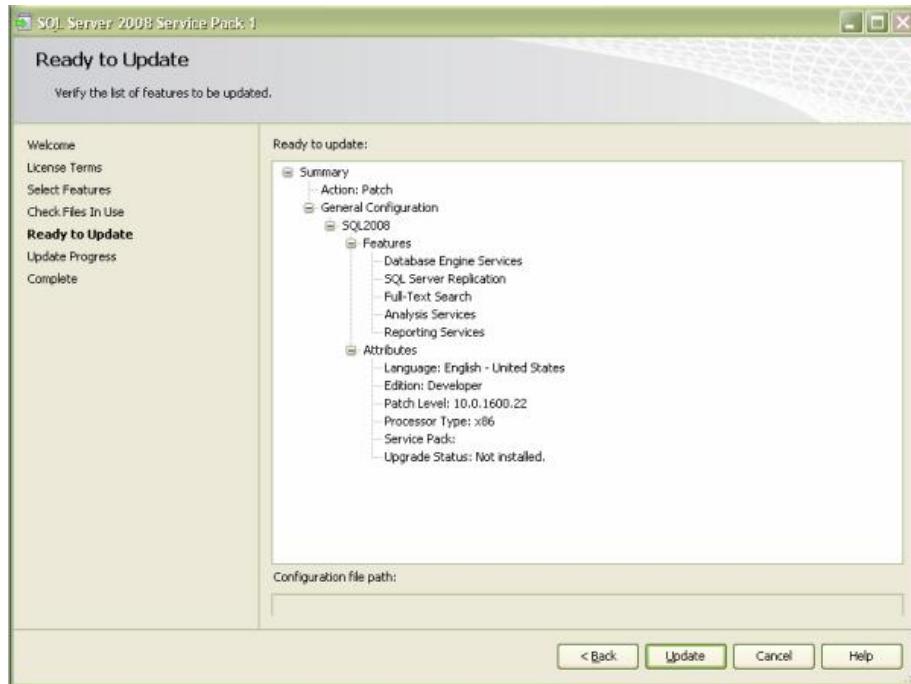


Once all the features are selected click Next to continue with the SQL Server 2008 Service Pack 1 (SP1) installation.

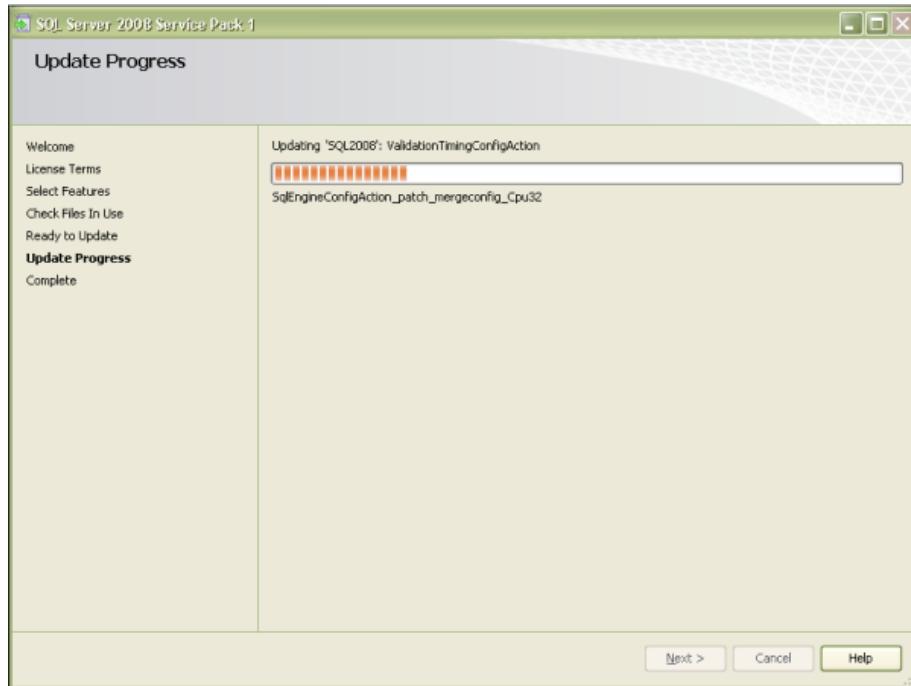
6. In **Check Files In Use** screen the setup will check for SQL Server related files which are currently being used. If there are any such files then they need to be released before performing the Service Pack installation. Click Next to continue with the installation.



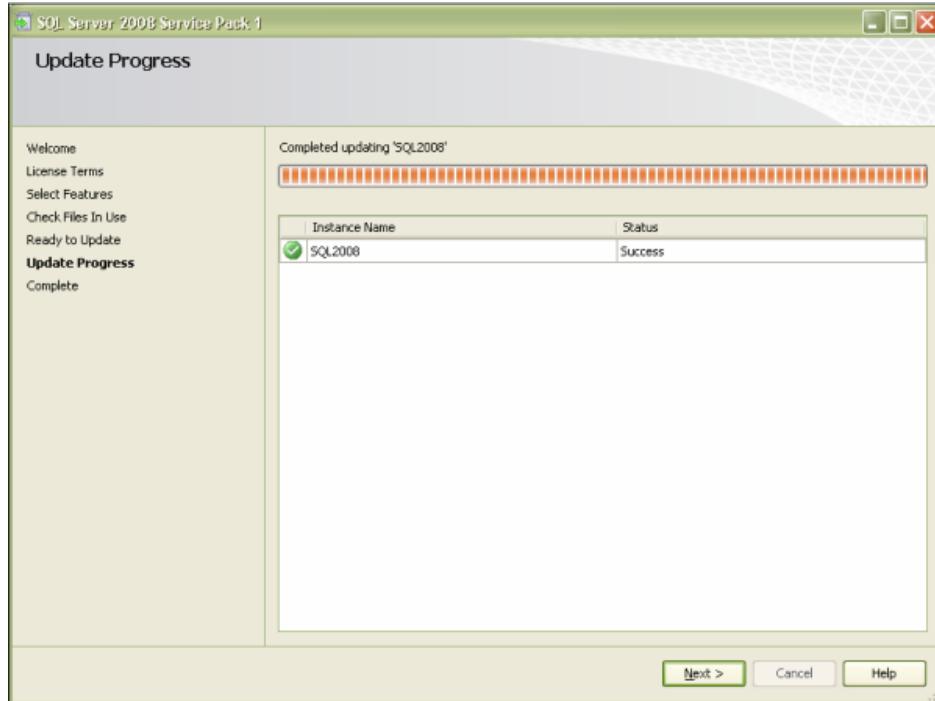
7. In **Ready to Upgrade** screen you can verify the list of features which will be upgraded during the installation. If you are fine with the list of features, then click **Upgrade** to perform the actual Service Pack installation.



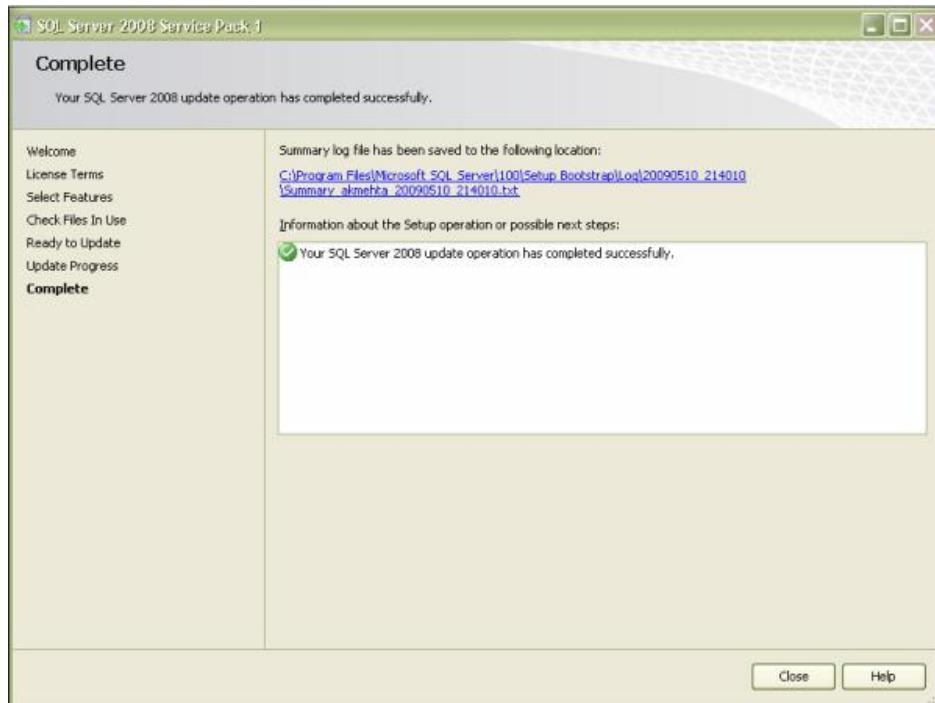
8. In **Upgrade Progress** screen you will be able to see the installation progress. It will approximately take 30 minutes to complete the installation. The installation time varies depending upon the hardware configuration.



9. Once the Service Pack installation is successful you will be able to see the **Success** message as shown in the snippet below.



10. On the **Complete** screen you will be able to see "**Your SQL Server 2008 update operation has completed successfully**" message. Click Close to end the installation setup.



11. You can verify the SQL Server 2008 Service Pack 1 (SP1) installation by executing the query below in SQL Server Management Studio (SSMS).

```
SELECT SERVERPROPERTY('Edition') AS 'Edition',
SERVERPROPERTY('ProductVersion') AS 'ProductVersion',
SERVERPROPERTY('ProductLevel') AS 'ProductLevel',
SERVERPROPERTY('ResourceLastUpdateDateTime') AS
'ResourceLastUpdateDateTime',
SERVERPROPERTY('ResourceVersion') AS 'ResourceVersion'
```

	Edition	ProductVersion	ProductLevel	ResourceLastUpdateDateTime	ResourceVersion
1	Developer Edition	10.0.2531.0	SP1	2009-03-29 12:30:27.077	10.00.2531

## Upgrade Advisor

Upgrading from the previous version of SQL Server to a newer version is always a challenging task for a Database Administrator. In order to improve the SQL Server 2008 upgrade experience, Microsoft introduced a free tool called SQL Server 2008 Upgrade Advisor (SSUA). This tool basically does the analysis of installed components of the earlier SQL Server versions, and generates a report which has details about the issues that need to be fixed either before or after you upgrade to SQL Server 2008.

Microsoft SQL Server 2008 Upgrade Advisor helps database administrators analyze the existing instances of SQL Server 2000 and SQL Server 2005 databases. It identifies features and configuration changes that might affect your successful upgrade to SQL Server 2008. Once the entire analysis is done by the tool, it documents all the upgrade issues along with providing a link for issue resolution. The tool also identifies all potential upgrade blockers that might occur during or after the upgrade without actually performing the actual upgrade. Database Administrators should utilize this wonderful tool during the upgrade planning phase of the project to analyze SQL Server 2000 and SQL Server 2005 instances. This tool can analyze different components of SQL Server such as SQL Server, Analysis Services, Notification Services, Reporting Services, Data Transformation Services, Integration Services, Scripts and SQL Trace Files. I strongly recommend using this tool in advance to avoid any last minute surprises during the actual upgrade.

### [Download Link for Microsoft SQL Server 2008 Upgrade Advisor Tool](#)

SQL Server 2008 Upgrade Advisor Tool is available as a free download from the following link:

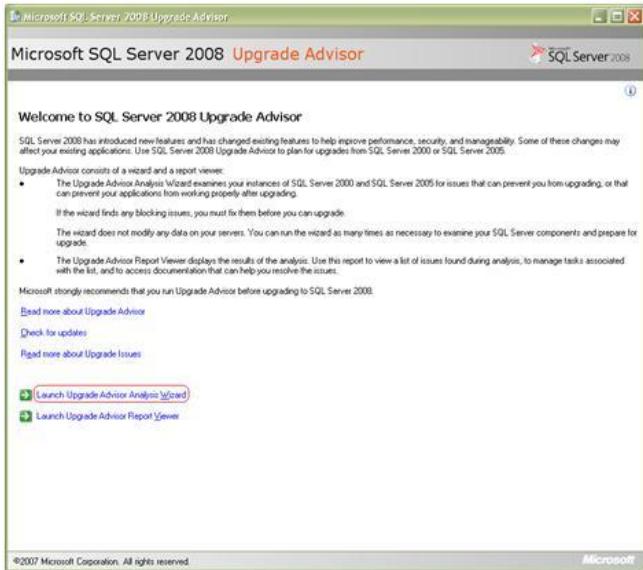
<http://www.microsoft.com/downloads/details.aspx?FamilyId=F5A6C5E9-4CD9-4E42-A21C-7291E7F0F852&displaylang=en>

You can also get the installation copy of SQL Server 2008 Upgrade Advisor Tool from "Servers\Redist\Upgrade Advisor" folder of the SQL Server installation media.

### [Using SQL Server 2008 Upgrade Advisor Tool](#)

After the successful installation of SSUA you can access it from Start menu - Click Start | All Programs | Microsoft SQL Server 2008 | SQL Server 2008 Upgrade Advisor.





On the “Welcome to SQL Server 2008 Upgrade Advisor” screen there are three options namely Read more about Upgrade Advisor, Check for updates and Read more about Upgrade Issues. I advise you to check for latest updates before using the tool, as Microsoft keep adding new rules and functionality to the tool to provide better customer experience while upgrade to a higher versions of SQL Server.

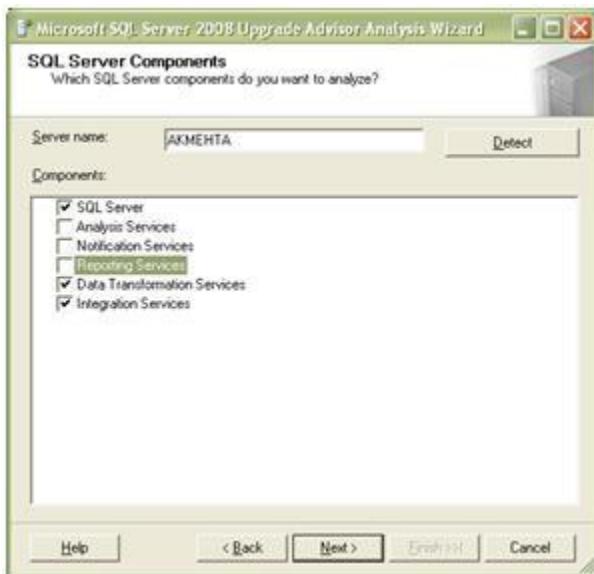
In the same screen you can see two more links to the wizards namely “Launch Upgrade Advisor Analysis Wizard” and “Launch Upgrade Advisor Report Viewer”. The “Launch Upgrade Advisor Report Viewer” will launch the report viewer from where you can see reports which were earlier generated by this tool in the previous analysis. You need to click on Launch Upgrade Advisor Analysis Wizard option to launch the Microsoft SQL Server 2008 Upgrade Advisor Analysis Wizard as shown in the below snippet.



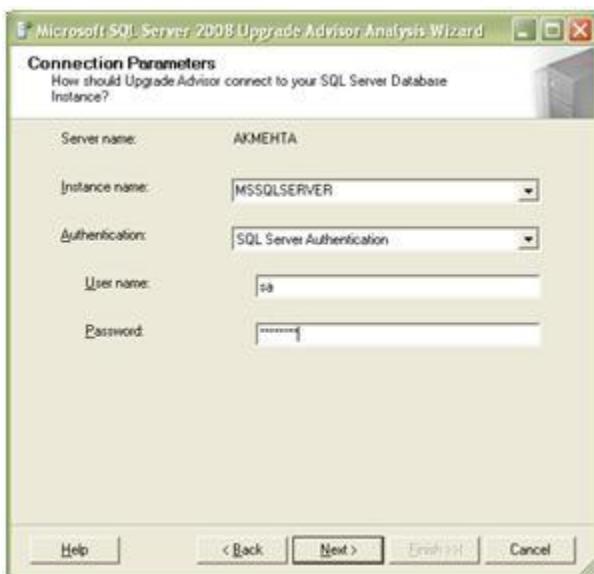
Click Next to see SQL Server Components screen where you will be asked to select the SQL Server Components which you want to analyze using this tool. In the Server



Name textbox provide the name of the SQL Server which needs to be analyzed.



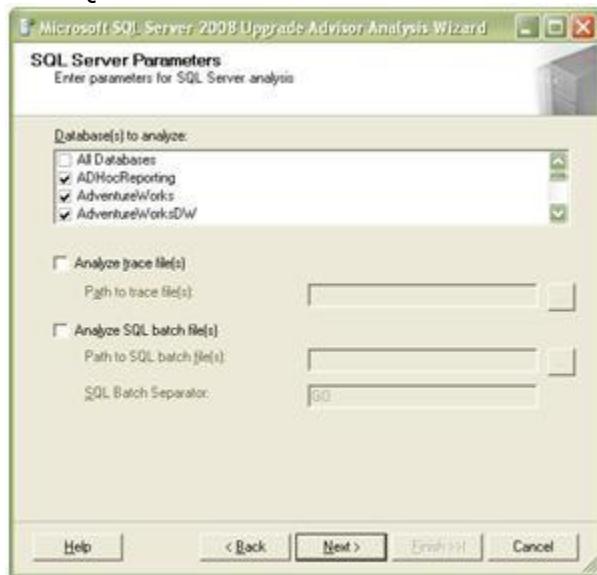
Once the Server Name is provided, you can either select the components manually or click the Detect button. The detect functionality will automatically detect the SQL Server Components which are installed on the SQL Server that needs to be checked for upgrade issues. In case you want to analyze a server which has multiple instances, then you have to specify just the name of the server in the server name box and select the components manually or allow the tool to detect by clicking Detect button. Once the components are identified click Next to see the Connection Parameters screen.



In the Connection Parameters screen select the Instance Name from the dropdownlist (for this article default instance of SQL Server 2005 is used by this tool for analysis) against which you want to do the analysis. In the Authentication screen you can either select Windows Authentication or SQL Server Authentication. In the above snippet you can see that I have selected SQL Authentication and I have provided the Username and Password to establish the connection. Click Next to see



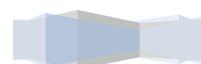
the SQL Server Parameters screen.



In the SQL Server Parameters screen you need to enter the parameters for SQL Server analysis. You can check box All Databases if you want to analysis issues for all the databases present on the SQL Server Instance. Else you can check only those databases against which you want to perform the analysis. It is always a good practice to choose All Databases option when you are planning to do an in-place upgrade. There are two other options like Analyze trace files and Analyze SQL batch file. If you have any you can check the options and provide the path of the files for the analysis. You can also run trace files against Upgrade Advisor. This way you will be able to analyze any adhoc query getting executed from the applications which uses SQL Server. The recommended SQL profile template is SQLProfilerTSQL\_Replay, as this will have unique number of queries. Click Next to see DTS Parameters screen.



In the DTS Parameters screen you have radio buttons to choose the location of DTS packages. You can either analysis all the DTS packages on the SQL Server or you



can analyze DTS packages which are stored as file. If you want to analyze the DTS packages stored as files then you need to select the second option and provide the path of the DTS package files. Click Next to see the SSIS Parameters screen.

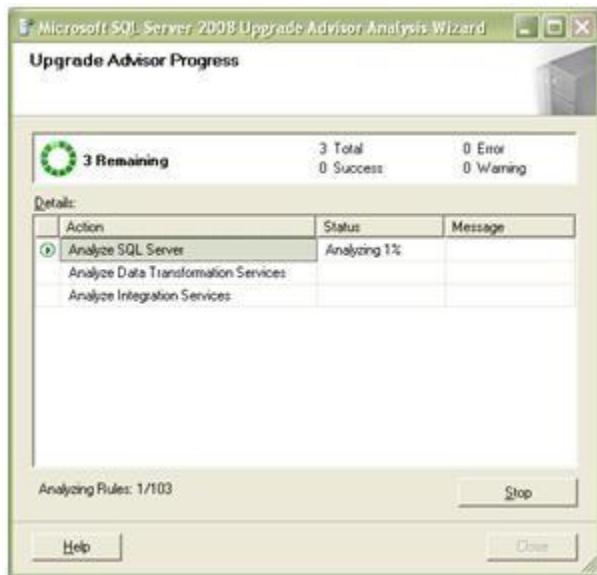


In the SSIS Parameters screen you have radio buttons to choose the location of SQL Server Integration Services (SSIS) packages. You can either analysis all the SSIS packages on the SQL Server or you can analyze SSIS packages which are stored as file. If you want to analyze the SSIS packages stored as files then you need to select the second option and provide the path of the SSIS package files. If your SSIS packages are encrypted then you also need to specify the Password for the package and then Click Next to see the Confirm Upgrade Advisor Settings screen.

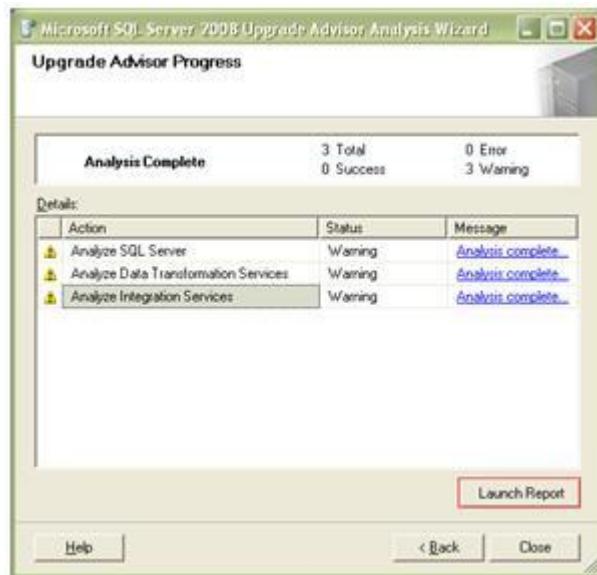


In the Confirm Upgrade Advisor Settings screen you can see the summary of all the options which you have selected in the previous screens. Finally click on Run button to capture the analysis using the Upgrade Advisor Tool.



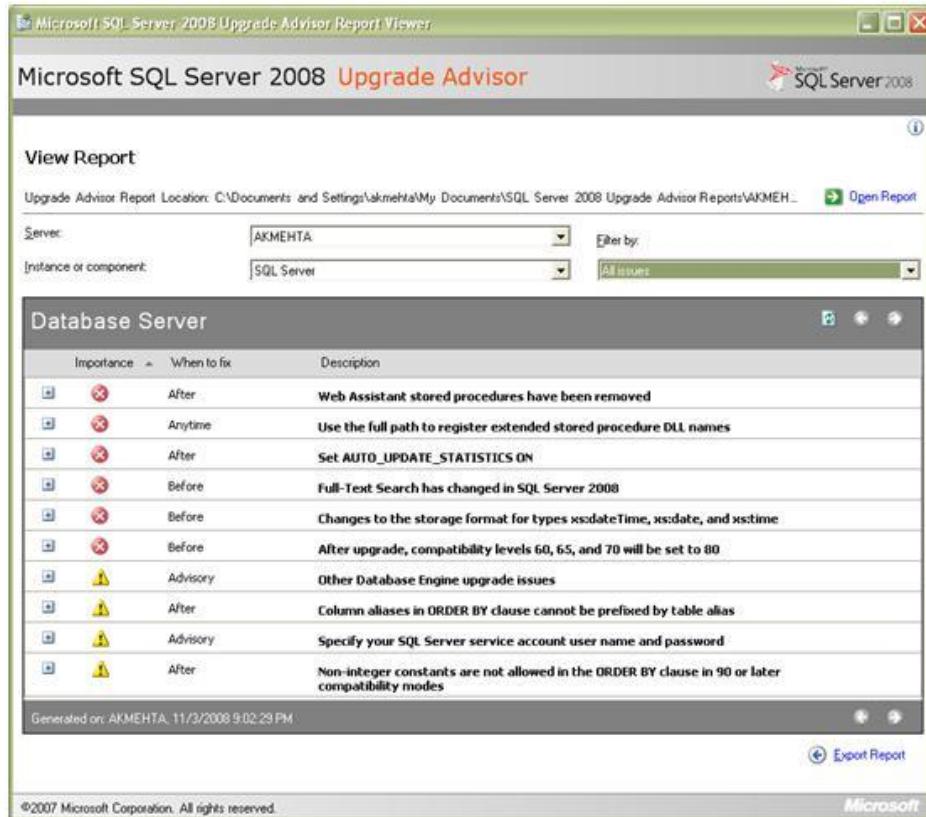


Once you click on Run you will be redirected to Upgrade Advisor Progress screen. You could see in the above snippet that when analysing SQL Server it's basically running 103 rules. Rules are nothing but scenarios according to Microsoft which will prevent you from successful upgrade. Once the analysis has completed you will see the below screen.

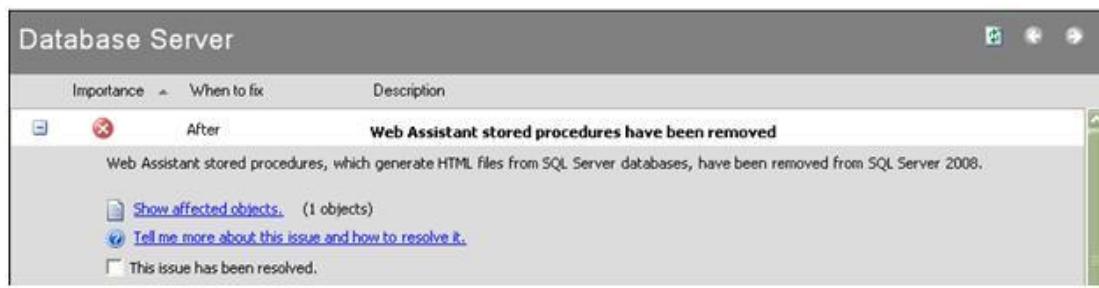


In order to see the detailed analysis report you need to click on Launch Report button which will open up Microsoft SQL Server 2008 Upgrade Advisor Report Viewer as shown in the below snippet. The results of the analysis are basically stored in an XML file in the following folder location "C:\Documents and Settings\NTUserName\My Documents\SQL Server 2008 Upgrade Advisor Reports\SERVERNAME\".





From the Instance or component dropdownlist you can choose components like SQL Server, Database Transformation Services, and Integration Services etc based on the selections which you have done before in the SQL Server Selections screen. The When to Fix column within the report mentions the stage of upgrade in which the issue needs to be fixed. The options under "When to Fix" are Before, After or Advisory. The ones which are classified to be in "Before" category are very important issues as they are upgrade blockers and it needs to be fixed before you start the upgrade of SQL Server. The issues which are classified to be in "After" category can be fixed after the upgrade of SQL Server as they are of Medium priority. The issues which are classified to be in "Advisory" category are informative messages. The issues which are in "After" and "Advisory" category should not be ignored by the DBA even though they won't affect the Upgrade. They basically inform you that your server is not configured as per best practices suggested by Microsoft. You can double click on each and every issue to know the steps for the issue resolution. Once you double click an issue you will see the below screen.



Click on "Show affected objects" to see the list of objects which are affected and needs to be fixed. Similarly if you are not sure about how to resolve the issues identified then you can click on "Tell me more about this issue and how to resolve it" link to see the steps to be followed for issue resolution.

This tool identifies most of the potential upgrade blockers on your existing SQL Server instances and can prevent you from a smooth upgrade. Database Administrators should utilize this wonderful tool during the upgrade planning phase of the project to analyze SQL Server 2000 and SQL Server 2005 instances.

### **In-Place Up gradation from SQL server 2005 to 2008**

1. Insert the SQL Server installation media, and from the root folder, double-click setup.exe. To install from a network share, move to the root folder on the share, and then double-click setup.exe. If the Microsoft SQL Server 2008 Setup dialog box appears, click **OK** to install the prerequisites, then click **Cancel** to exit SQL Server 2008 installation.
2. If the .NET Framework 3.5 SP1 installation dialog box appears, select the check box to accept the .NET Framework 3.5 SP1 License Agreement. Click **Next**. To exit SQL Server 2008 installation, click **Cancel**. When installation of .NET Framework 3.5 SP1 is complete, click **Finish**.
3. Windows Installer 4.5 is also required, and may be installed by the Installation Wizard. If you are prompted to restart your computer, restart, and then run SQL Server 2008 Setup.exe again.
4. When prerequisites are installed, the Installation Wizard will start the SQL Server Installation Center. To upgrade an existing instance of SQL Server 2008, click **Upgrade from SQL Server 2000 or SQL Server 2005**.
5. If Setup support files are required, SQL Server Setup will install them. If you are instructed to restart your computer, restart before you continue.
6. The System Configuration Checker will run a discovery operation on your computer. To continue, click **OK**. Setup log files are created for your installation.
7. On the Product key page, click an option button to indicate whether you are upgrading to a free edition of SQL Server, or whether you have a PID key for a production version of the product.
8. On the License Terms page, read the license agreement, and then select the check box to accept the license terms and conditions. **Click Next to continue**. To end Setup, click **Cancel**.
9. On the Select Instance page, specify the instance of SQL Server to upgrade.
10. On the Feature Selection page, the features to upgrade will be preselected. A description for each component group appears in the right pane after you select the feature name. Be aware that you cannot change the features to be upgraded, and you cannot add features during the upgrade operation. To add features to an upgraded instance of SQL Server 2008 after the upgrade operation is complete.
11. On the Instance Configuration page, specify whether to install a default or a named instance.



**Instance ID** — By default, the instance name is used as the Instance ID. This is used to identify installation directories and registry keys for your instance of SQL Server. This is the case for default instances and named instances. For a default instance, the instance name and instance ID would be MSSQLSERVER. To use a nondefault instance ID, select the **Instance ID** check box and provide a value.

**Instance root directory** — By default, the instance root directory is C:\Program Files\Microsoft SQL Server\. To specify a nondefault root directory, use the field provided, or click **Browse** to locate an installation folder.

All SQL Server service packs and upgrades will apply to every component of an instance of SQL Server.

**Detected instances and features** — The grid will show instances of SQL Server that are on the computer where Setup is running. If a default instance is already installed on the computer, you must install a named instance of SQL Server 2008. **Click Next to continue.**

12. The Disk Space Requirements page calculates the required disk space for the features that you specify, and compares requirements to the available disk space on the computer where Setup is running.
13. Work flow for the rest of this topic depends on the features that you have specified for your installation. You might not see all the pages, depending on your selections.
14. On the Server Configuration -- Service Accounts page, specify login accounts for SQL Server services. The actual services that are configured on this page depend on the features that you are upgrading.

Authentication and login information will be carried forward from the previous instance of SQL Server. You can assign the same login account to all SQL Server services, or you can configure each service account individually. You can also specify whether services start automatically, are started manually, or are disabled. Microsoft recommends that you configure service accounts individually so that SQL Server services are granted the minimum permissions they have to complete their tasks.

To specify the same login account for all service accounts in this instance of SQL Server, provide credentials in the fields at the bottom of the page.

**Security Note** Do not use a blank password. Use a strong password.

When you are finished specifying login information for SQL Server services, click **Next**.

15. Use the **Server Configuration — Collation** tab to specify nondefault collations for the Database Engine and Analysis Services.
16. On the Full-Text Search Upgrade Options page, specify the upgrade options for the databases being upgraded.
17. On the **Error and Usage Reporting** page, specify the information that you want to send to Microsoft that will help improve SQL Server. By default, options for error reporting and feature usage are enabled.



18. The System Configuration Checker will run one more set of rules to validate your computer configuration with the SQL Server features that you have specified before the upgrade operation begins.
19. The Ready to Upgrade page displays a tree view of installation options that were specified during Setup. To continue, click **Install**.
20. During installation, the progress page provides status so that you can monitor installation progress as Setup continues.
21. After installation, the Complete page provides a link to the summary log file for the installation and other important notes. To complete the SQL Server installation process, click **Close**.
22. If you are instructed to restart the computer, do so now. It is important to read the message from the Installation Wizard when you have finished with Setup.

#### **Best Practices while Upgrading:**

1. Run the Upgrade Advisor before upgrading. Make any necessary changes before performing the upgrade.
2. Perform a test upgrade of your test SQL Servers before you upgrade your production servers. And don't forget to test your applications with the new version also.
3. Before you upgrade, be sure you have a plan in place to fall back to in case the upgrade is problematic.
4. Don't upgrade SQL Server clusters in place. Instead, rebuild them on new hardware.
5. If you upgrade from a previous version of SQL Server, you should update all of the statistics in all your databases using either UPDATE STATISTICS or sp\_updatestats. This is because statistics are not automatically updated during the upgrade process.



## Configuring SQL Server

### Configuring Network Protocols from SQL Server configuration manager

SQL Server 2008 is a client-server application designed to efficiently exchange data and instructions over one or more network connections.

### SQL Server 2008 Network Protocols

SQL Server 2008 provides support for four protocols:

- Shared Memory
- TCP/IP
- Named Pipes
- Virtual Interface Adapter (VIA)

By default, the only network protocols enabled for most editions of SQL Server are TCP/IP and Shared Memory. The Developer and Enterprise Evaluation editions are configured with all protocols except Shared Memory disabled during installation, but the remaining protocols can be enabled if required. If a protocol is not enabled, SQL Server will not listen on an endpoint that is configured to utilize that protocol.

The SQL Server Configuration Manager is used to configure server protocols.

#### Shared Memory

The Shared Memory protocol can only be used by local connections, because it is a shared memory and process space used for inter-server communication. It has only one configurable property: Enabled. The Enabled property can be set to Yes or No, resulting in a status of Enabled or Disabled.

#### Named Pipes

Named Pipes uses Inter-Process Communication (IPC) channels for efficient inter-server communication, as well as local area network (LAN) communication. The Named Pipes protocol has some enhancements in SQL Server 2008 including support for encrypted traffic, but because of the excessive overhead of Named Pipes when connecting across networks or firewalls, and the additional port that Named Pipes requires to be opened (445), it is generally a good idea to leave the Named Pipes protocol disabled. However, there are many applications that take advantage of the Named Pipes protocol because they were designed for local network implementations. Named Pipes provides easy access to Remote Procedure Calls (RPC) within a single security domain, and so is advantageous to these applications. If you need to support one of these applications, and the SQL Server is not exposed to external traffic, the risk of enabling the Named Pipes protocol and corresponding endpoint is minimal.

Named Pipes has two configurable properties: Enabled and Pipe Name. The Enabled property works the same as the Shared Memory protocol. The Pipe Name specifies the inter-process pipe that SQL Server will listen on. The default pipe is \\.\pipe\sql\query.



## TCP/IP

The TCP/IP protocol is the primary and preferred protocol for most SQL Server installations. It is configured on two separate tabs on the TCP/IP Properties window: the Protocol tab and the IP Addresses tab, as shown



The Protocol tab has the following four configurable properties:

- Enabled — This works the same as the other protocols.
- Keep Alive — This specifies how many milliseconds SQL Server waits to verify an idle connection is still valid by sending a KEEPALIVE packet. The default is 30,000 milliseconds.
- Listen All — This specifies whether SQL Server will listen on all IP addresses configured on the server.
- No Delay — This option specifies whether the TCP protocol queues small packets to send out larger packets. This queuing is typically undesirable in transaction-based systems, and so it should be left in its default configuration of No.

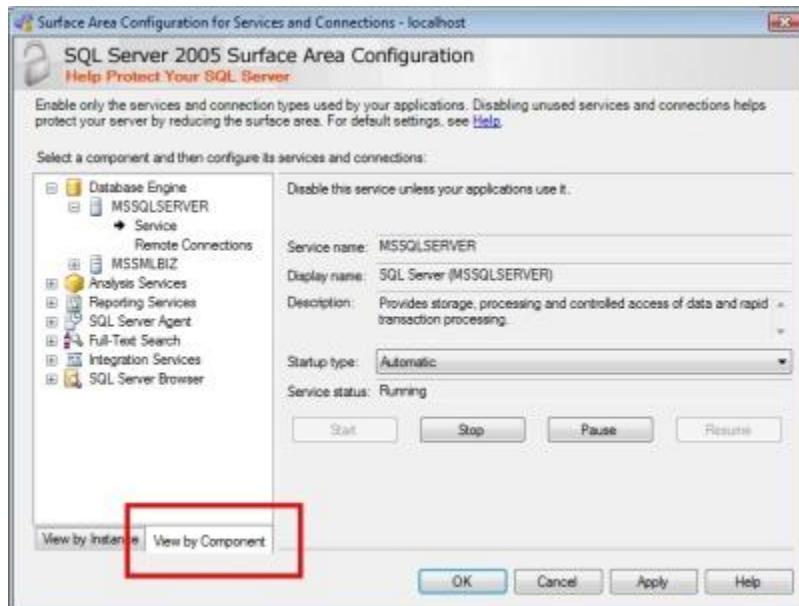
## Surface Area Configuration for services and Connections.

There are two ways to view the various services and connections on a given server. The first is by instance, and this is usually the easiest method.





However, if you have multiple components on the same server, such as multiple instances of the database engine, it may be easier to group by component, which is what the second tab (View by Component) is for:



Once you've settled on your view, the next step is to actually configure each component. Every component listed will have a Service option where you can configure the startup status for the service as well as control the current state of the service itself. For instance, the following image is of the MSSQLSERVER database engine service, which is set to Automatic, meaning it will start when the computer starts up.

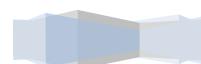
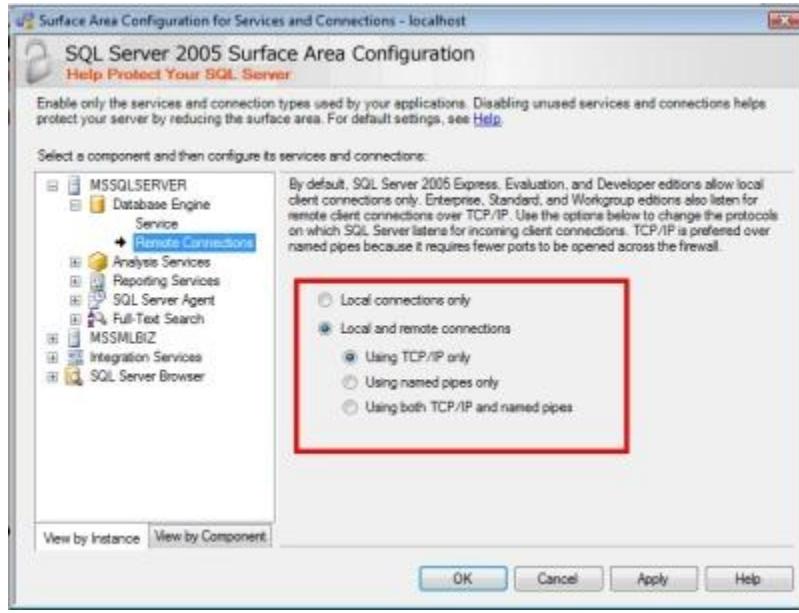




You have the ability to set any of the following startup types:

- **Automatic** - The service will start when the computer starts up.
- **Manual** - The service is capable of being started, but must be started manually by an authorized user (Power Users or Administrators local group membership).
- **Disabled** - The service cannot be started unless the startup type is changed.

In addition to the service configurations, the Database Engine and Analysis Services have additional options. For the database engine, there is the Remote Connections configuration.



As the highlighted section shows, you can configure the database engine either to listen only for local connections (originating from the same computer as SQL Server) or to listen for both local and remote connections. In addition, if you choose for SQL Server to listen for remote connections as well, you'll have the choice whether to use TCP/IP, Named Pipes, or both. In general, for network connections, you'll want to use TCP/IP. In my experience, Named Pipes sometimes suffers from timeout issues that you don't get with TCP/IP.

### The Dedicated Administrator Connection

In SQL Server 2005, Microsoft introduced a new feature called Dedicated Administrator Connection (DAC). Using this feature a SQL Server Database Administrator can connect to a SQL Server Instance when the database engine is not responding to regular connections. During such a scenario a DBA can connect to the SQL Server Instance to troubleshoot and to kill any of the SQL Server Processes which are causing the issues.

The DAC allows database administrators to connect to a SQL Server Instance and to execute T-SQL commands to troubleshoot and fix issues rather than rebooting the SQL Server which could lead to database corruption or other problems. By default, the remote Dedicated Administrator Connection feature is disabled in SQL Server 2005 and later versions. It's a good practice to enable the DAC feature once the SQL Server 2005 or SQL Server 2008 is installed on every instance as this will help you troubleshoot issues when regular connections are not responding. However, only one dedicated administrator connection is allowed at a time on SQL Server 2005 and later versions.

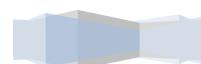
### Enable Dedicated Administrator Connection in SQL Server 2008 Using TSQL

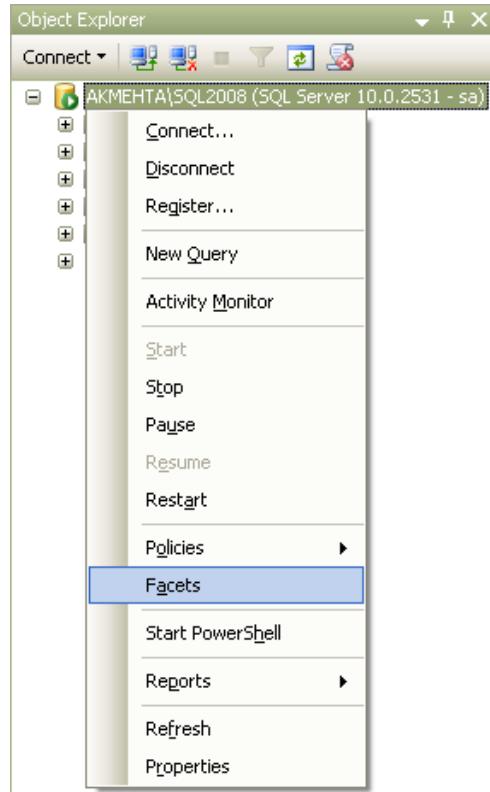
Execute the below T-SQL to enable remote clients to utilize the Dedicated Administrator Connection.

```
Use master
GO
sp_configure 'show advanced options' , 1
GO
/* 0 = Allow Local Connection, 1 = Allow Remote Connections*/
sp_configure 'remote admin connections', 1
GO
RECONFIGURE
GO
```

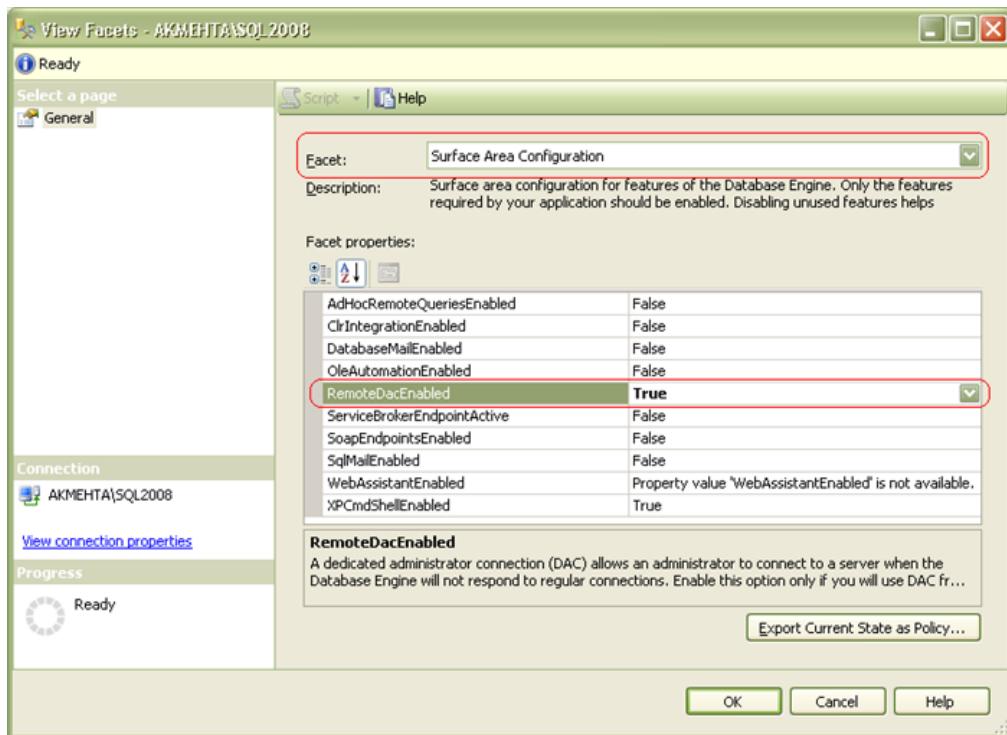
### Enable Dedicated Administrator Connection in SQL Server 2008 Using SQL Server 2008 Management Studio

Database Administrators can also enable Dedicated Administrator Connection Feature using SQL Server 2008 Management Studio. This can be done by right clicking the SQL Server Instance and selecting the **Facets** option from the drop down list as shown in the snippet below.





This will open up **View Facets** window as shown in the snippet below. Here you need to select **Surface Area Configuration** facet as highlighted and then select the option as "**True**" for **RemoteDacEnabled**.



Finally, click OK to save the configuration changes in the **View Facets** window.

You can also enable other database engine features like AsHocRemoteQueriesEnabled,ClrIntegrationEnabled, DatabaseMailEnabled, OleAutomationEnabled, ServiceBrokerEndpointActive, SoapEndpointsEnabled, SQLMailEnabled, WebAssistanceEnabled, XPCmdShellEnabled etc when required using the Surface Server Configuration Facet which is available in SQL Server 2008 Management Studio.

Once the Dedicated Administrator Connection is enabled you can connect to SQL Server 2008 using either SQL Server Management Studio or using SQLCMD.

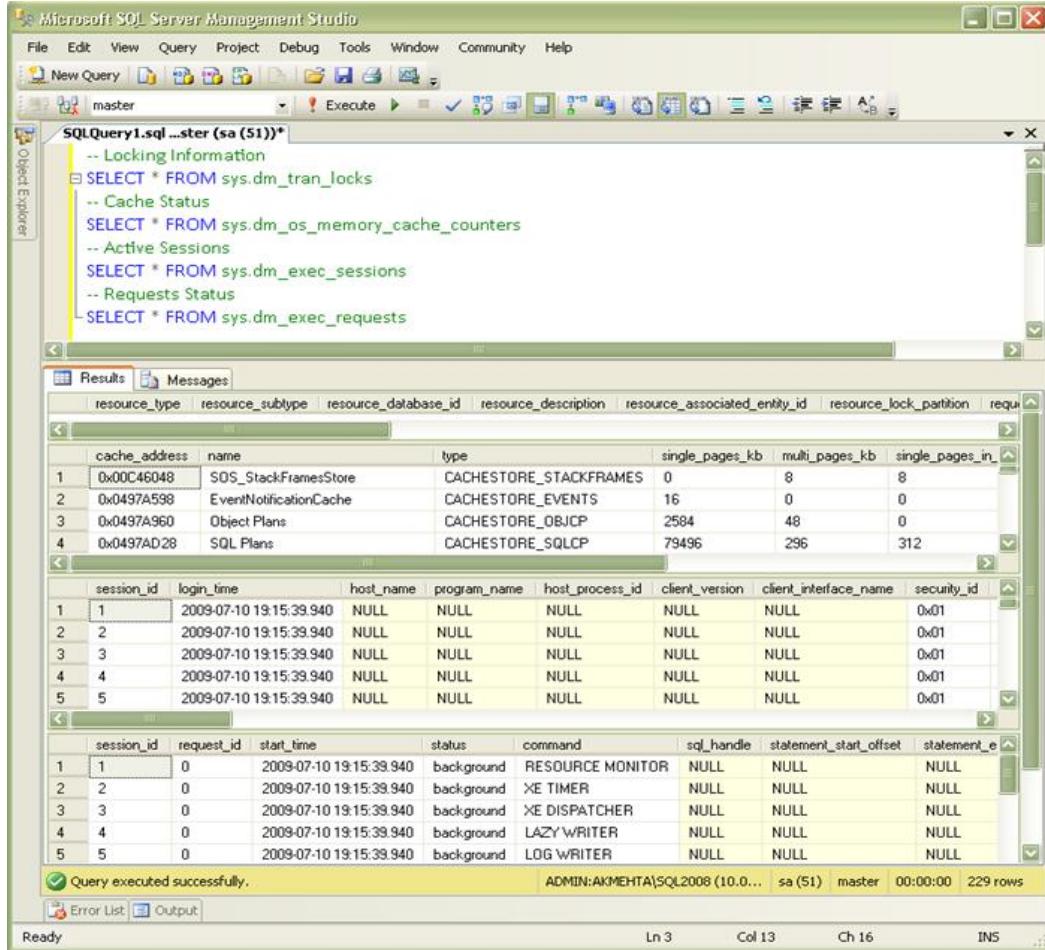
### Using DAC with SQL Server Management Studio

You need to specify "**ADMIN:**" before the SQL Server Instance name when trying to connect to an SQL Server Instance to using DAC feature as shown in the snippet below.



Once you are connected to SQL Server Instance using DAC, then you can execute code such as the code below to check the SQL Server health.

```
-- Locking Information
SELECT * FROM sys.dm_tran_locks
GO
-- Cache Status
SELECT * FROM sys.dm_os_memory_cache_counters
GO
-- Active Sessions
SELECT * FROM sys.dm_exec_sessions
GO
-- Requests Status
SELECT * FROM sys.dm_exec_requests
GO
```



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a file named 'SQLQuery1.sql' is open under the 'master' database. The code in the query window is as follows:

```
-- Locking Information
SELECT * FROM sys.dm_tran_locks
-- Cache Status
SELECT * FROM sys.dm_os_memory_cache_counters
-- Active Sessions
SELECT * FROM sys.dm_exec_sessions
-- Requests Status
SELECT * FROM sys.dm_exec_requests
```

The Results pane displays three tables of data:

cache_address	name	type	single_pages_kb	multi_pages_kb	single_pages_in_kb
1 0x00C46048	SOS_StackFramesStore	CACHESTORE_STACKFRAMES	0	8	8
2 0x0497A598	EventNotificationCache	CACHESTORE_EVENTS	16	0	0
3 0x0497A960	Object Plans	CACHESTORE_OBJCP	2584	48	0
4 0x0497AD28	SQL Plans	CACHESTORE_SQLCP	79496	296	312

session_id	login_time	host_name	program_name	host_process_id	client_version	client_interface_name	security_id
1 1	2009-07-10 19:15:39.940	NULL	NULL	NULL	NULL	NULL	0x01
2 2	2009-07-10 19:15:39.940	NULL	NULL	NULL	NULL	NULL	0x01
3 3	2009-07-10 19:15:39.940	NULL	NULL	NULL	NULL	NULL	0x01
4 4	2009-07-10 19:15:39.940	NULL	NULL	NULL	NULL	NULL	0x01
5 5	2009-07-10 19:15:39.940	NULL	NULL	NULL	NULL	NULL	0x01

session_id	request_id	start_time	status	command	sql_handle	statement_start_offset	statement_end_offset
1 1	0	2009-07-10 19:15:39.940	background	RESOURCE MONITOR	NULL	NULL	NULL
2 2	0	2009-07-10 19:15:39.940	background	XE TIMER	NULL	NULL	NULL
3 3	0	2009-07-10 19:15:39.940	background	XE DISPATCHER	NULL	NULL	NULL
4 4	0	2009-07-10 19:15:39.940	background	LAZYWRITER	NULL	NULL	NULL
5 5	0	2009-07-10 19:15:39.940	background	LOG WRITER	NULL	NULL	NULL

At the bottom of the Results pane, a message states: "Query executed successfully." Below the panes, the status bar shows: Ready, Ln 3, Col 13, Ch 16, INS.

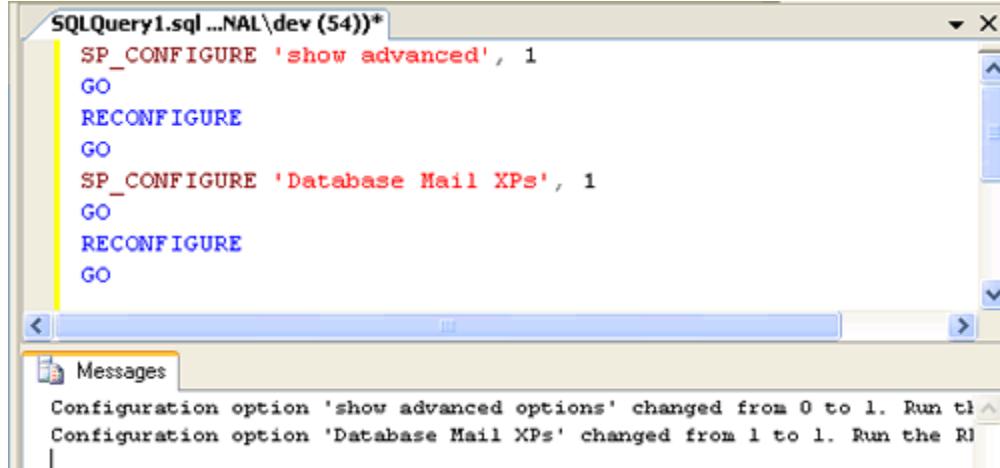
## Configuring Database Mail

In order to send mail using Database Mail in SQL Server, there are 3 basic steps that need to be carried out. 1) Configure Email , 2) Create Profile and Account

To configure it, we need to enable the Database Mail XPs parameter through the sp\_configure stored procedure, as shown here:

```
sp_CONFIGURE'show advanced',1
GO
RECONFIGURE
GO
sp_CONFIGURE'Database Mail XPs',1
GO
RECONFIGURE
GO
```





```

SQLQuery1.sql...NAL\dev (54)*
SP_CONFIGURE 'show advanced', 1
GO
RECONFIGURE
GO
SP_CONFIGURE 'Database Mail XPs', 1
GO
RECONFIGURE
GO

```

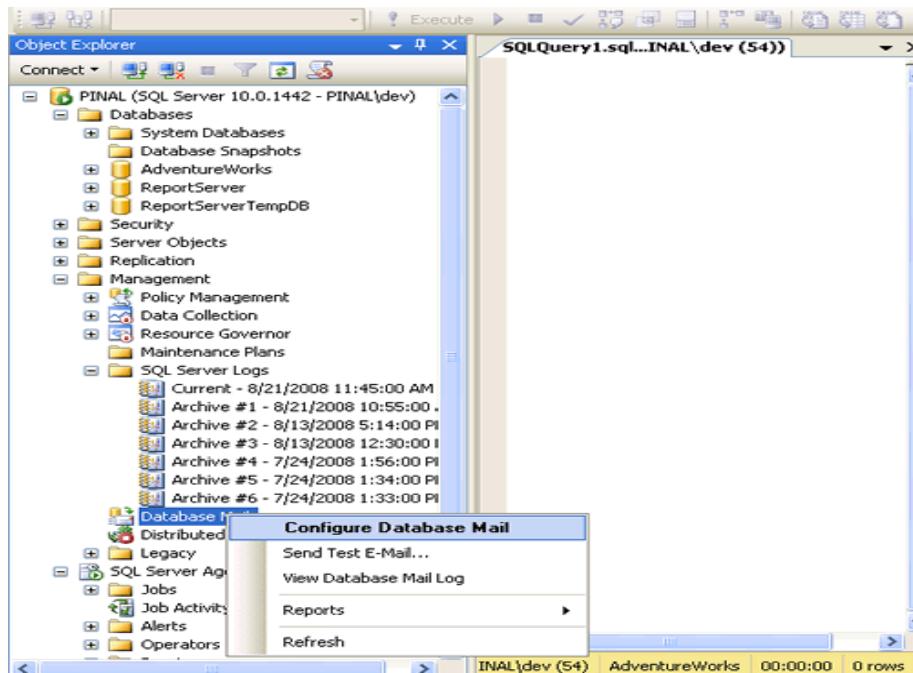
Messages

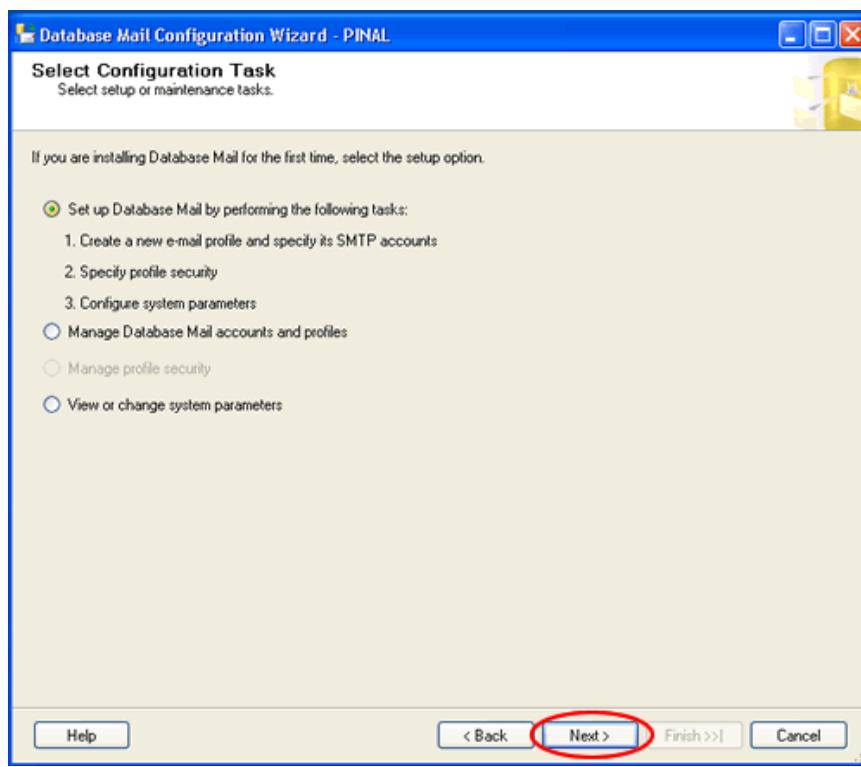
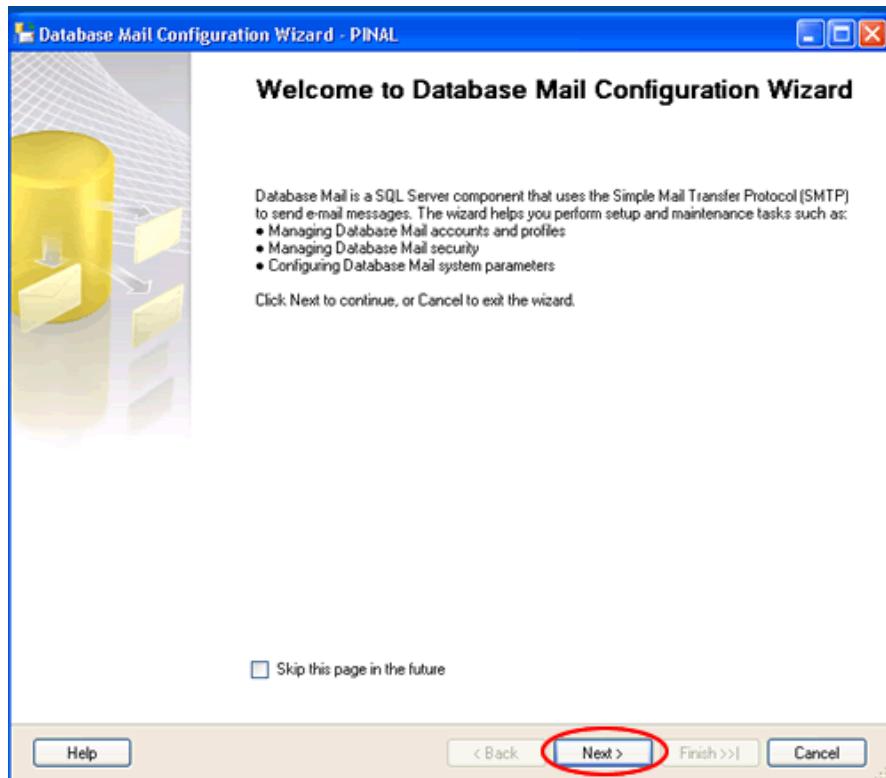
Configuration option 'show advanced options' changed from 0 to 1. Run the RECONFIGURE statement.

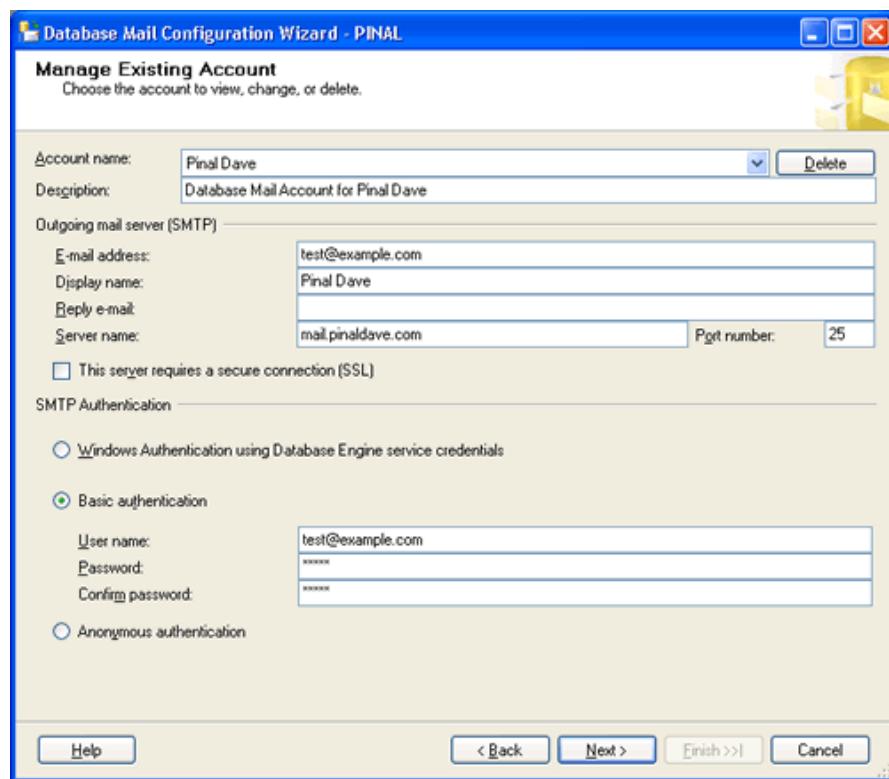
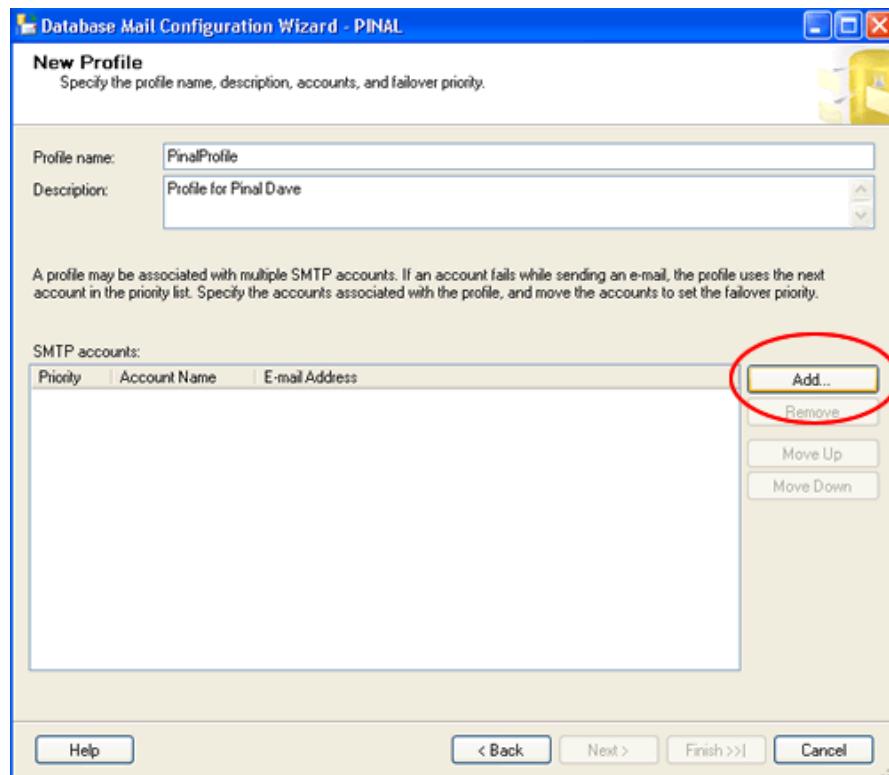
Configuration option 'Database Mail XPs' changed from 1 to 1. Run the RECONFIGURE statement.

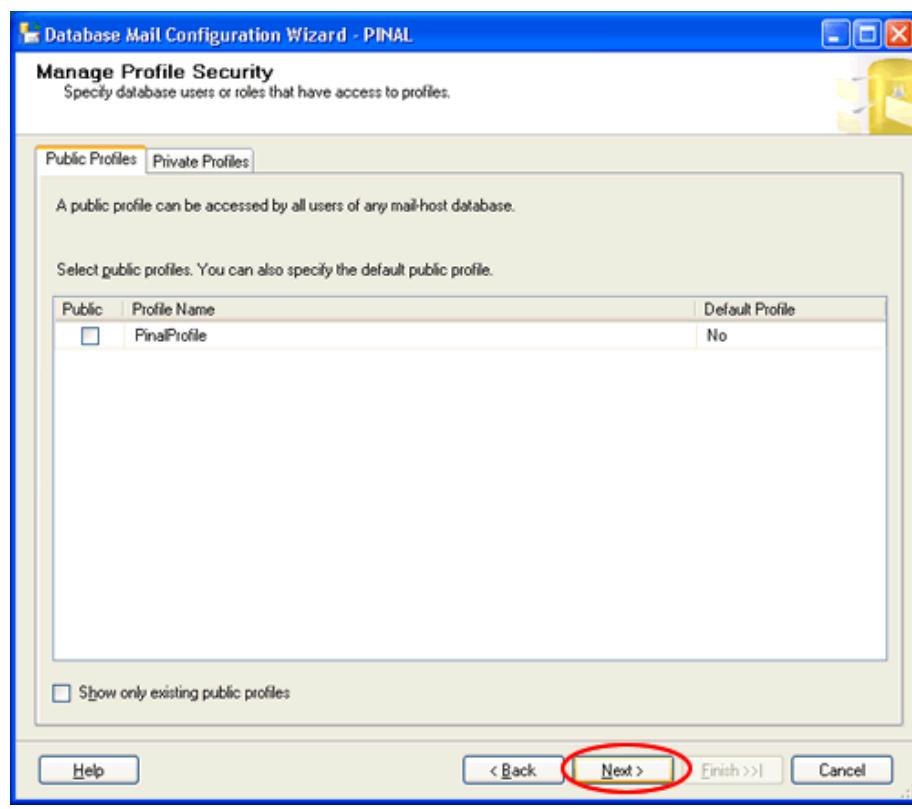
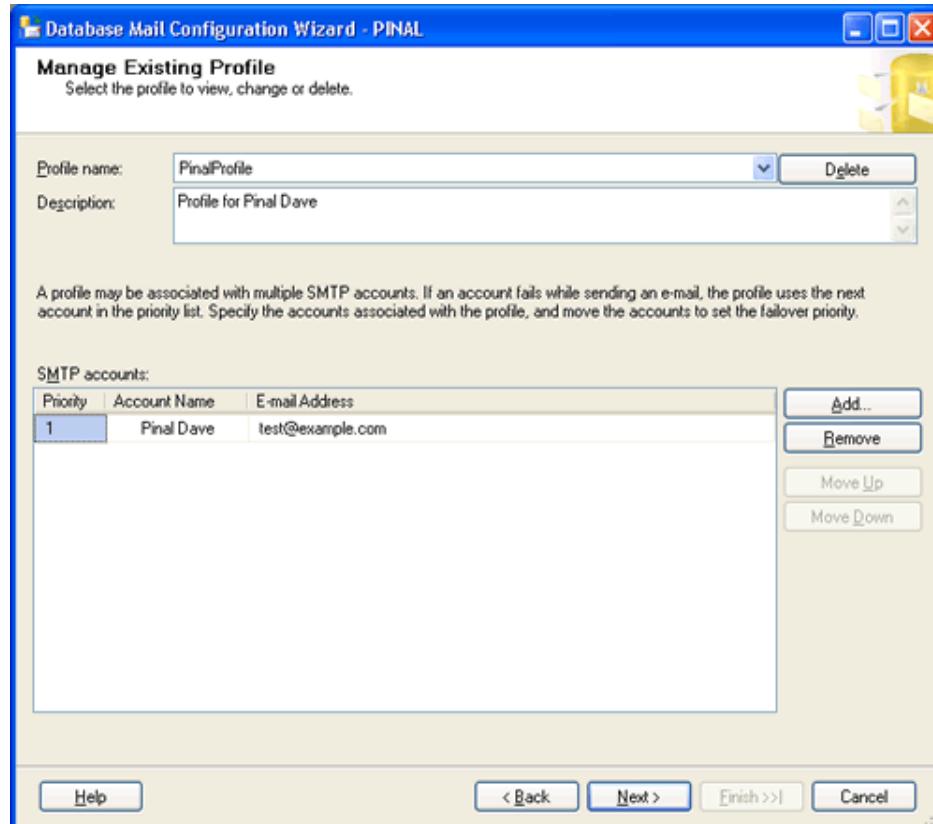
### Step 2) Create Profile and Account:

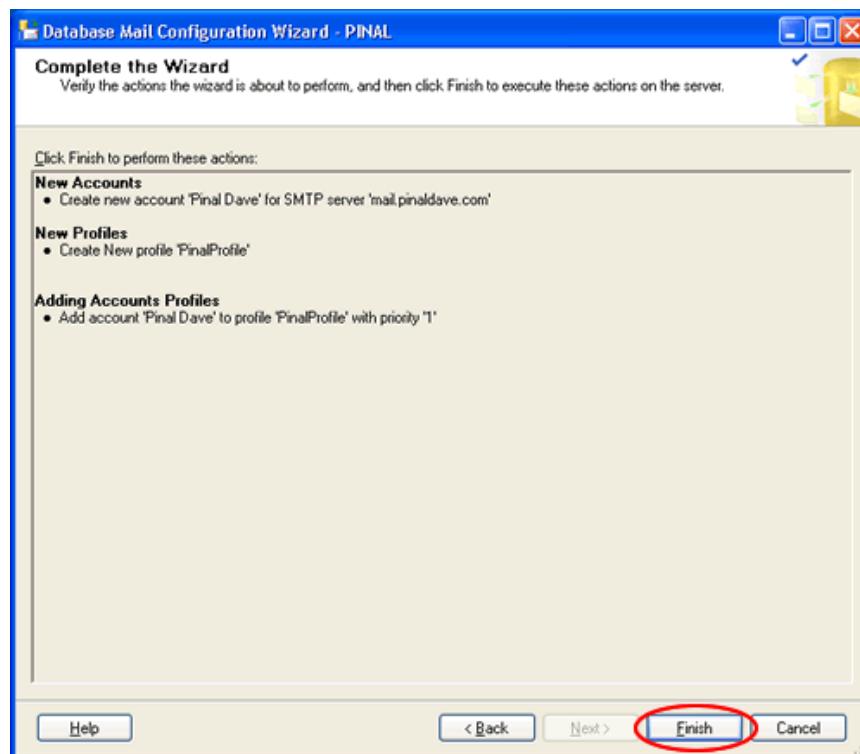
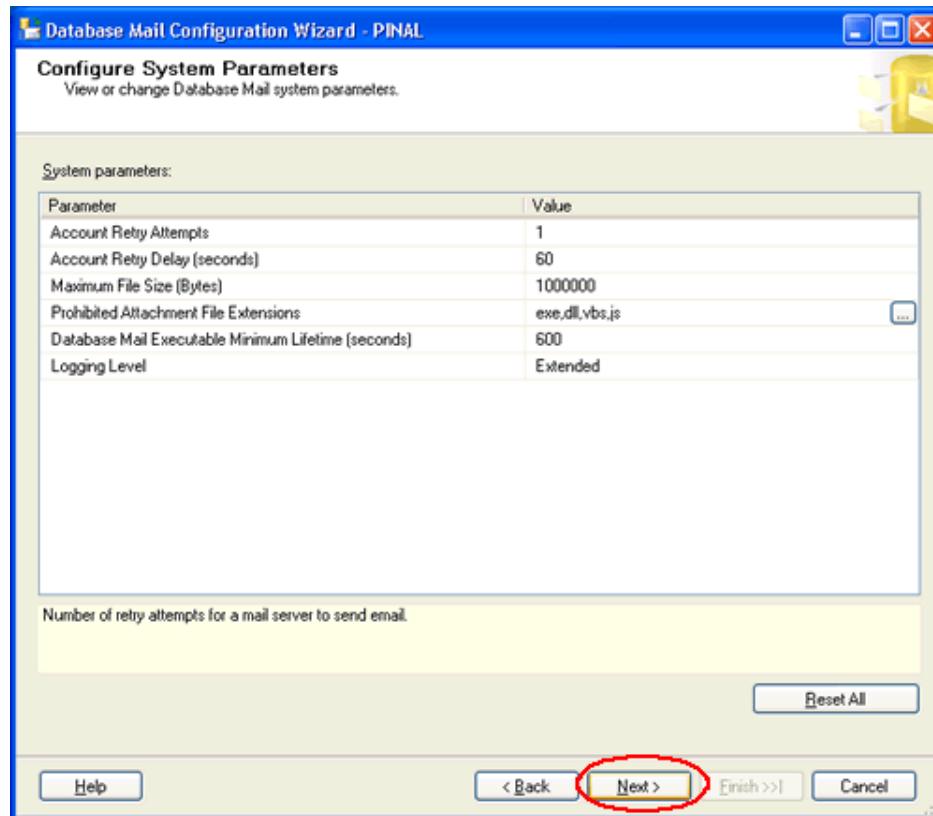
You need to create a profile and account using the Configure Database Mail Wizard which can be accessed from the Configure Database Mail context menu of the Database Mail node in Management Node. This wizard is used to manage accounts, profiles, and Database Mail global settings which are shown below:

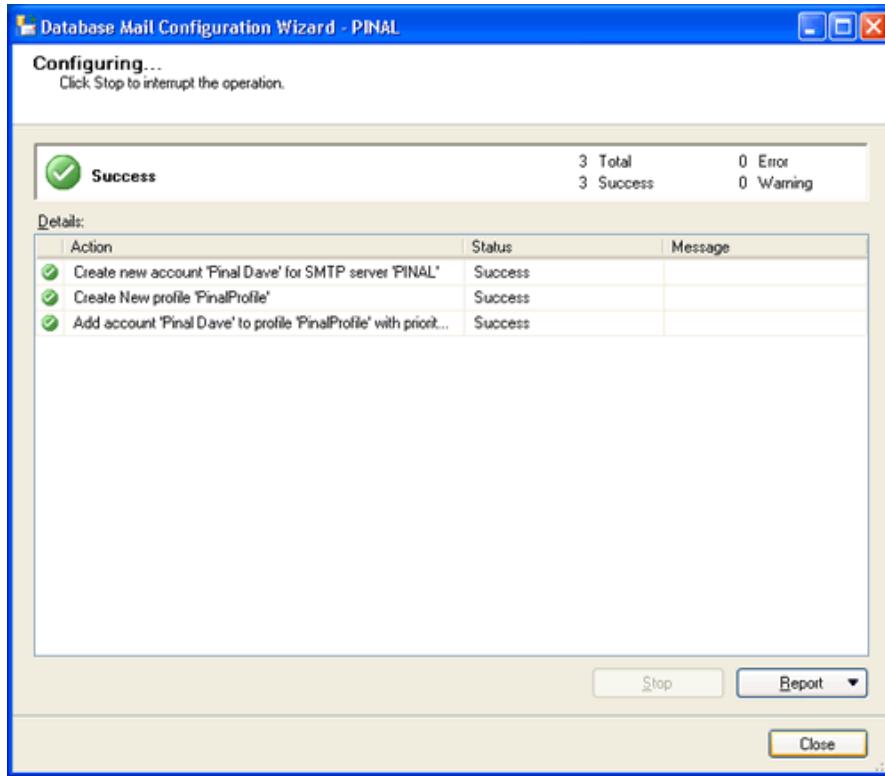












### **Best Practices in Configuration/ Database Settings:**

1. Unless you know exactly what you are doing and have already performed impartial experiments that prove that making SQL Server configuration changes helps you in your particular environment, do not change any of the SQL Server configuration settings.
2. In almost all cases, leave the "auto create statistics" and "auto update statistics" options on for all user databases.
3. In most cases, the settings for the "maximum server memory" and the "minimum server memory" should be left to their default values. This is because the default values allow SQL Server to dynamically allocate memory in the server for the best overall optimum performance. If you use AWE memory, then this recommendation is to be ignored, and maximum memory needs to be set manually.
4. Many databases need to be shrunk periodically in order to free up disk space as older data is deleted from the database. But don't be tempted to use the "auto shrink" database option, as it can waste SQL Server resources unnecessarily. Instead, shrink databases manually.
5. Don't rely on AUTOGROWTH to automatically manage the size of your databases. Instead, proactively monitor and alter database size as circumstances dictate. Only use AUTOGROWTH to deal with unexpected growth.



## **Tempdb Configuration**

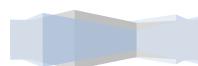
In SQL Server 2005/2008, TempDB has taken on some additional responsibilities. As such, some of the best practice has changed and so has the necessity to follow these best practices on a more wide scale basis. In many cases TempDB has been left to default configurations in many of our SQL Server 2000 installations. Unfortunately, these configurations are not necessarily ideal in many environments. With some of the shifts in responsibilities in SQL Server 2005/2008 from the user defined databases to TempDB, what steps should be taken to ensure the SQL Server TempDB database is properly configured?

### **What is TempDB responsible for in SQL Server 2005/2008**

- Global (# #temp) or local (#temp) temporary tables, temporary table indexes, temporary stored procedures, table variables, tables returned in table-valued functions or cursors.
- Database Engine objects to complete a query such as work tables to store intermediate results for spools or sorting from particular GROUP BY, ORDER BY, or UNION queries.
- Row versioning values for online index processes, Multiple Active Result Sets (MARS) sessions, AFTER triggers and index operations (SORT\_IN\_TEMPDB).
- DBCC CHECKDB work tables.
- Large object (varchar(max), nvarchar(max), varbinary(max) text, ntext, image, xml) data type variables and parameters.

### **Best practices for TempDB**

- Do not change collation from the SQL Server instance collation.
- Do not change the database owner from sa.
- Do not drop the TempDB database.
- Do not drop the guest user from the database.
- Do not change the recovery model from SIMPLE.
- Ensure the disk drives TempDB resides on have RAID protection i.e. 1, 1 + 0 or 5 in order to prevent a single disk failure from shutting down SQL Server. Keep in mind that if TempDB is not available then SQL Server cannot operate.
- If SQL Server system databases are installed on the system partition, at a minimum move the TempDB database from the system partition to another set of disks.
- Size the TempDB database appropriately. For example, if you use the SORT\_IN\_TEMPDB option when you rebuild indexes, be sure to have sufficient free space in TempDB to store sorting operations. In addition, if you are running into insufficient space errors in TempDB, be sure to determine the culprit and either expand TempDB or re-code the offending process.



## Managing Database Services

### Starting the SQL Server Service Automatically

You can configure an instance of Microsoft SQL Server (or SQL Server Agent) to start automatically each time you start the Microsoft Windows 2000 or Windows Server 2003 operating system. You can:

- Use the SQL Server Installer.
- Use SQL Server Configuration Manager.
- Use SQL Server Management Studio.

Normal configuration is to start the SQL Server service automatically. If a server is intentionally restarted because of software or hardware maintenance, or unintentionally restarted due to a power or hardware failure, SQL Server will become available without additional attention from an attendant.

Possible reasons to configure SQL Server not to start automatically include the following scenarios:

- You want to investigate the cause of the restart before making the database available.
- SQL Server is not always needed, and you wish to control conserve computer resources, such as on a laptop computer.

### Starting SQL Server Manually

You can manually start an instance of Microsoft SQL Server or SQL Server Agent using the following methods.

<b>Method</b>	<b>Description</b>
SQL Server Configuration Manager	Start, pause, resume, and stop an instance of a local SQL Server or SQL Server Agent service.
Command prompt	Start an instance of SQL Server or SQL Server Agent service from a command prompt by the <b>net start</b> command or by running <b>sqlservr.exe</b> .

Use **sqlservr.exe** to start SQL Server from a command prompt only to troubleshoot SQL Server. Before you start an instance of SQL Server using **sqlservr.exe** from a command prompt (independent of SQL Server Configuration Manager), consider the following:

- SQL Server runs in the security context of the user, not the security context of the account assigned to run SQL Server during setup.
- All system messages appear in the window used to start an instance of SQL Server.



## **SQL Server can be stopped/started from SQL Server Configuration Manager**

### To start the default instance of SQL Server

- On the Start menu, point to All Programs, point to Microsoft SQL Server 2008, point to Configuration Tools, and then click SQL Server Configuration Manager.
- In SQL Server Configuration Manager, expand Services, and then click SQL Server.
- In the details pane, right-click SQL Server (MSSQLServer), and then click Start.
- A green arrow on the icon next to the server name and on the toolbar indicates that the server started successfully.
- Click OK to close SQL Server Configuration Manager.

### To start a named instance of SQL Server

- On the Start menu, point to All Programs, point to Microsoft SQL Server 2008, point to Configuration Tools, and then click SQL Server Configuration Manager.
- In SQL Server Configuration Manager, expand Services, and then click SQL Server (<instance\_name>).
- In the details pane, right-click the named instance of SQL Server, and then click Start.
- A green arrow on the icon next to the server name and on the toolbar indicates that the server started successfully.
- Click OK to close SQL Server Configuration Manager.

## **Microsoft SQL Server service can be started by using Net commands.**

### To start the default instance of SQL Server

From a command prompt, enter one of the following commands:

```
net start "SQL Server (MSSQLSERVER)" -or-
net start MSSQLSERVER
```

### To start a named instance of SQL Server

From a command prompt, enter one of the following commands. Replace <instancename> with the name of the instance you want to manage.

```
net start "SQL Server ( instancename )" -or-
net start MSSQL$ instancename
```

### To start SQL Server with startup options

Add startup options to the end of the net start "SQL Server (MSSQLSERVER)" statement, separated by a space. When started using net start, startup options use a slash (/) instead of a hyphen (-).

```
net start "SQL Server (MSSQLSERVER)" /f /m -or-
net start MSSQLSERVER /f /m
```

## Starting an Instance of SQL Server (sqlservr.exe)

If the SQL Server Database Engine does not start, one troubleshooting step is to attempt to start the Database Engine from the command prompt.

By default, sqlservr.exe is located at C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\Binn. If a second instance of SQL Server is installed, a second copy of sqlservr.exe is located in a directory such as C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\binn. You can start one instance of SQL Server by using sqlservr.exe from a different instance, but SQL Server will start the version of the incorrect instance as well, including service packs, which may lead to unexpected results. To avoid this, use the MS-DOS change directory (**cd**) command to move to the correct directory before starting sqlservr.exe

**To start a named instance of SQL Server in single-user mode from a command prompt**

From a command prompt, enter the following command:

```
sqlservr.exe -m -s <instancename>
```

## SQL Server Service Startup Options

When you install SQL Server, Setup writes a set of default startup options in the Microsoft Windows registry. You can use these startup options to specify an alternate **master** database file, **master** database log file, or error log file.

Startup options can be set by using SQL Server Configuration Manager.

Default startup options	Description
<b>-d</b> <i>master_file_path</i>	The fully qualified path for the <b>master</b> database file (typically, C:\Program Files\Microsoft SQL Server\MSSQL.n\MSSQL\Data\master.mdf). If you do not provide this option, the existing registry parameters are used.
<b>-e</b> <i>error_log_path</i>	The fully qualified path for the error log file (typically, C:\Program Files\Microsoft SQL Server\MSSQL.n\MSSQL\LOG\ERRORLOG). If you do not provide this option, the existing registry parameters are used.
<b>-l</b> <i>master_log_path</i>	The fully qualified path for the <b>master</b> database log file (typically C:\Program Files\Microsoft SQL Server\MSSQL.n\MSSQL\Data\mastlog.ldf). If you do not specify this option, the existing registry parameters are used.

You can override the default startup options temporarily and start an instance of SQL Server by using the following additional startup options.



<b>-c</b>	Shortens startup time when starting SQL Server from the command prompt. Typically, the SQL Server Database Engine starts as a service by calling the Service Control Manager. Because the SQL Server Database Engine does not start as a service when starting from the command prompt, use <b>-c</b> to skip this step.
<b>-f</b>	Starts an instance of SQL Server with minimal configuration. This is useful if the setting of a configuration value (for example, over-committing memory) has prevented the server from starting.
<b>-s</b>	Allows you to start a named instance of SQL Server. Without the <b>-s</b> parameter set, the default instance will try to start. You must switch to the appropriate BINN directory for the instance at a command prompt before starting <b>sqlservr.exe</b> . For example, if <b>Instance1</b> were to use <code>\mssql\$instance1</code> for its binaries, the user must be in the <code>\mssql\$instance1\binn</code> directory to start <b>sqlservr.exe -s instance1</b> .
<b>-T</b> <i>trace#</i>	Indicates that an instance of SQL Server should be started with a specified trace flag ( <i>trace#</i> ) in effect. Trace flags are used to start the server with nonstandard behavior. For more information, see Trace Flags (Transact-SQL).

### Starting SQL Server in Single-User Mode

Under certain circumstances, you may have to start an instance of Microsoft SQL Server in single-user mode by using the **startup option -m**. For example, you may want to change server configuration options or recover a damaged **master** database or other system database. Both actions require starting an instance of SQL Server in single-user mode. When you start an instance of SQL Server in single-user mode, note the following:

- Only one user can connect to the server.
- The CHECKPOINT process is not executed. By default, it is executed automatically at startup.

### Starting SQL Server with Minimal Configuration

If you have configuration problems that prevent the server from starting, you can start an instance of Microsoft SQL Server by using the minimal configuration startup option. This is the startup option **-f** starting an instance of SQL Server with minimal configuration automatically puts the server in single-user mode.

When you start an instance of SQL Server in minimal configuration mode, note the following:

- Only a single user can connect, and the CHECKPOINT process is not executed.
- Remote access and read-ahead are disabled.
- Startup stored procedures do not run.

79

After the server has been started with minimal configuration, you should change the appropriate server option value or values, stop, and then restart the server.



## Case Study/Practical Trouble Shooting

### Problem:

Moved TempDB but forgot to add the name of the MDF and LDF at the end of the file path  
Error: 5123, Severity: 16, State: 1 when moving TempDB

### Solution:

To move tempdb is a fairly simple task, that can very easily be done incorrectly, which will cause the SQL server to not start up the next time it is restarted, which is generally immediately to put the file moves for tempdb into place. To start off with get the file path information for the current configuration of tempdb, you will need this to fall back to if you have a problem:

```
SELECT name, physical_name
FROM sys.database_files
```

To move the files, you simply use ALTER DATABASE as follows:

```
ALTER DATABASE tempdb MODIFY FILE (NAME = 'tempdev', FILENAME =
'c:\program files\microsoft sql server\mssql.2\mssql\sql\data\tempdb.mdf')
ALTER DATABASE tempdb MODIFY FILE (NAME = 'templog', FILENAME =
'c:\program files\microsoft sql server\mssql.2\mssql\sql\data\tempdb.ldf')
```

To break my instance I am going to omit the file names and only provide the path, which is what was done in both of the posts that inspired this tread:

```
ALTER DATABASE tempdb MODIFY FILE (NAME = 'tempdev', FILENAME =
'c:\program files\microsoft sql server\mssql.2\mssql\sql\data\')
ALTER DATABASE tempdb MODIFY FILE (NAME = 'templog', FILENAME =
'c:\program files\microsoft sql server\mssql.2\mssql\sql\data\'')
```

This will output the following result:

The file "tempdev" has been modified in the system catalog. The new path will be used the next time the database is started.

The file "templog" has been modified in the system catalog. The new path will be used the next time the database is started.

You can rerun the above query to validate the change occurred, but it won't take effect until you restart the service, so I went ahead and restarted my SQL Instance and it fails as expected with the following error in the error log:

### Error: 5123, Severity: 16, State: 1.

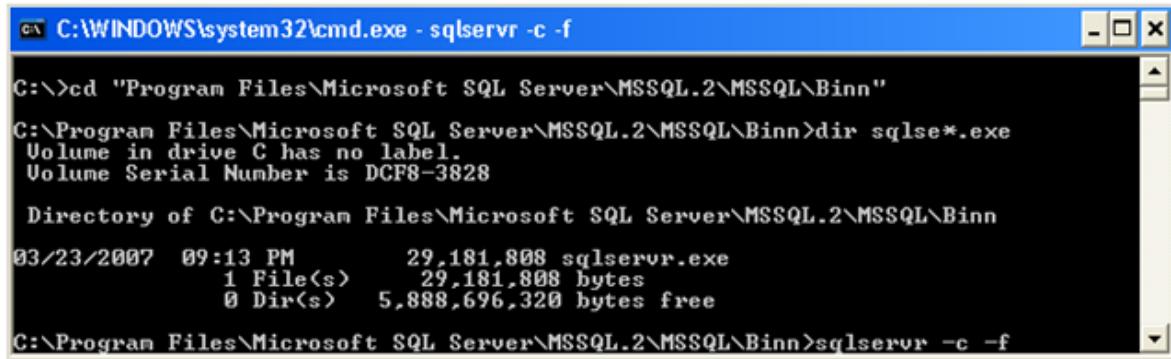
CREATE FILE encountered operating system error 3(The system cannot find the path specified.) while attempting to open or create the physical file 'c:\program files\microsoft sql server\mssql.2\mssql\data\'.

Could not create tempdb. You may not have enough disk space available. Free additional disk space by deleting other files on the tempdb drive and then restart SQL Server. Check for additional errors in the event log that may indicate why the tempdb files could not be initialized.

I had given the error information from the log file for why SQL Server failed to start. So now that we can't get into SQL Server how do we fix it. Well it isn't all that difficult to do, but you have to drop to the command prompt to do it. First open the command prompt by running cmd in the Run box:

Then change directories to the Binn directory under your SQL Instances path:

Then run sqlservr with the -c and -f startup parameters which will start SQL Server in minimal configuration mode.



C:\>cd "Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\Binn"  
C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\Binn>dir sqlservr.exe  
Volume in drive C has no label.  
Volume Serial Number is DCF8-3828  
Directory of C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\Binn  
03/23/2007 09:13 PM 29,181,808 sqlservr.exe  
1 File(s) 29,181,808 bytes  
0 Dir(s) 5,888,696,320 bytes free  
C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\Binn>sqlservr -c -f

NOTE: Do not use any other startup parameters or Trace Flags as these can cause SQL to try to create tempdb from the settings that are wrong and again fail to start.

When you start SQL Server from the command prompt it will spool the log information out to the command prompt screen. When it shows Recovery is complete the SQL Server Instance is running in single user mode and can be connected to through SSMS or sqlcmd.



```
C:\WINDOWS\system32\cmd.exe - sqlservr -c -f

2009-01-06 21:19:38.53 spid5s      CHECKDB for database 'master' finished without errors on 2008-05-10 01:30:00.483 (local time). This is an informational message only; no user action is required.
2009-01-06 21:19:38.54 spid5s      Server started with '-f' option. Auditing will not be started. This is an informational message only; no user action is required.
2009-01-06 21:19:38.57 spid5s      Starting up database 'mssqlsystemresource'.
2009-01-06 21:19:38.59 spid5s      The resource database build version is 9.00.3 042. This is an informational message only. No user action is required.
2009-01-06 21:19:38.98 spid6s      Starting up database 'model'.
2009-01-06 21:19:38.98 spid5s      Server name is 'LT-JKEHAYIAS'. This is an informational message only. No user action is required.
2009-01-06 21:19:39.17 spid6s      CHECKDB for database 'model' finished without errors on 2008-05-10 01:30:19.403 (local time). This is an informational message only; no user action is required.
2009-01-06 21:19:39.18 spid6s      Clearing tempdb database.
2009-01-06 21:19:39.59 Server      A self-generated certificate was successfully loaded for encryption.
2009-01-06 21:19:39.62 Server      Server is listening on [ 'any' <ipv4> 1433].
2009-01-06 21:19:39.62 Server      Server local connection provider is ready to accept connection on [ \\.\pipe\SQLLocal\MSSQLSERVER ].
2009-01-06 21:19:39.64 Server      Server named pipe provider is ready to accept connection on [ \\.\pipe\sql\query ].
2009-01-06 21:19:39.65 Server      Server is listening on [ 127.0.0.1 <ipv4> 1434].
2009-01-06 21:19:39.65 Server      Dedicated admin connection support was established for listening locally on port 1434.
2009-01-06 21:19:39.87 spid6s      Starting up database 'tempdb'.
2009-01-06 21:19:39.98 spid6s      CHECKDB for database 'tempdb' finished without errors on 2008-05-10 01:30:19.403 (local time). This is an informational message only; no user action is required.
2009-01-06 21:19:42.31 Server      The SQL Network Interface library could not register the Service Principal Name (SPN) for the SQL Server service. Error: 0x54b, state: 3. Failure to register an SPN may cause integrated authentication to fall back to NTLM instead of Kerberos. This is an informational message. Further action is only required if Kerberos authentication is required by authentication policies.
2009-01-06 21:19:42.31 Server      SQL Server is now ready for client connections. This is an informational message; no user action is required.
2009-01-06 21:19:42.39 spid5s      Recovery is complete. This is an informational message only. No user action is required.
```

Once you connect to the object explorer details [query analyzer] then run the correct ALTER DATABASE scripts to fix the tempdb path.

```
ALTER DATABASE tempdb MODIFY FILE (NAME = 'tempdev', FILENAME = 'c:\program files\microsoft sql server\mssql.2\mssql\data\tempdb.mdf')
ALTER DATABASE tempdb MODIFY FILE (NAME = 'templog', FILENAME = 'c:\program files\microsoft sql server\mssql.2\mssql\data\tempdb.ldf')
```

Once this has been run, you can close the SQL Server Instance running in the command prompt by pressing Ctrl+C with the window active. Then restart the SQL Service from the Services.msc snapin or the Computer Management Console and you should be back in business.



## Case Study: Rebuilding System databases

We need to rebuild the system databases if the master database is corrupted or damaged. Let us discuss in detail how to rebuild system databases in SQL server 2008.

**Step 1:** Take a full backup of all the System and User databases prior to rebuilding the system databases as we need to restore it later to avoid data loss.

**Step 2:** Copy the SQL 2005 setup files from the CD to the hard disk. In the command prompt, navigate to the folder which contains the setup.exe file. In my case it is available in D:\Setups\SQL 2005\Servers folder. The SQL Server we are going to rebuild is currently running. Now type the below command,

```
start /wait setup.exe /qn INSTANCENAME="MSSQLSERVER" REINSTALL=SQL_Engine  
REBUILDDATABASE=1 SAPWD="XXXX"
```

where XXXX is the name of the password.

INSTANCENAME="MSSQLSERVER" for default instance of SQL 2005 and

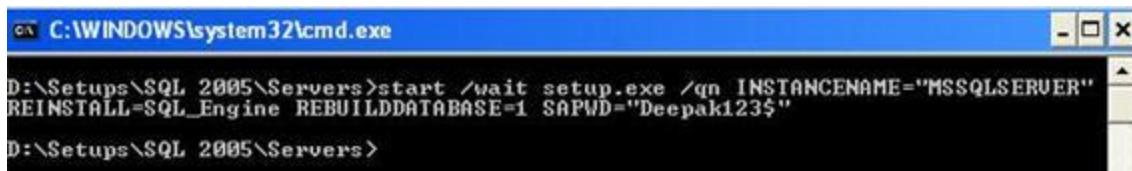
INSTANCENAME="MSSQL\$INSTANCENAME" for named instance of SQL 2005.

For example,

If you have a named instance named as "Deepak\Test" then type as below,

INSTANCENAME="MSSQL\$TEST" in the above command

Refer the below screenshot for the same.



The screenshot shows a Windows Command Prompt window titled 'C:\WINDOWS\system32\cmd.exe'. The command entered is: `D:\Setups\SQL 2005\Servers>start /wait setup.exe /qn INSTANCENAME="MSSQLSERVER"  
REINSTALL=SQL_Engine REBUILDDATABASE=1 SAPWD="Deepak123$"`. The command prompt then shows the directory again: `D:\Setups\SQL 2005\Servers>`.

**Step 3:** After executing the command in step 2 the rebuild process will start and will complete within 5 minutes. You can verify whether the databases are rebuilt by navigating to folder containing the data and log files for the system databases. If you arrange them using modified date it will clearly show the time when it was last modified and it is the time when we executed the command in Step 2.

**Step 4:** Once the rebuild is completed, connect to the SQL server using SSMS. In the object explorer only the system databases will be available.

If any user db were present prior to rebuild it will be lost and we need to perform as below to retrieve it.



1. Restore from the backup taken in Step 1 (or)
2. We can attach from the data and log files of the user db as they will not be cleaned up during rebuild process.

**NOTE :** No Need to detach all the user databases before rebuild as the ldf and mdf files will be present in the same path as usual and will not be overwritten.

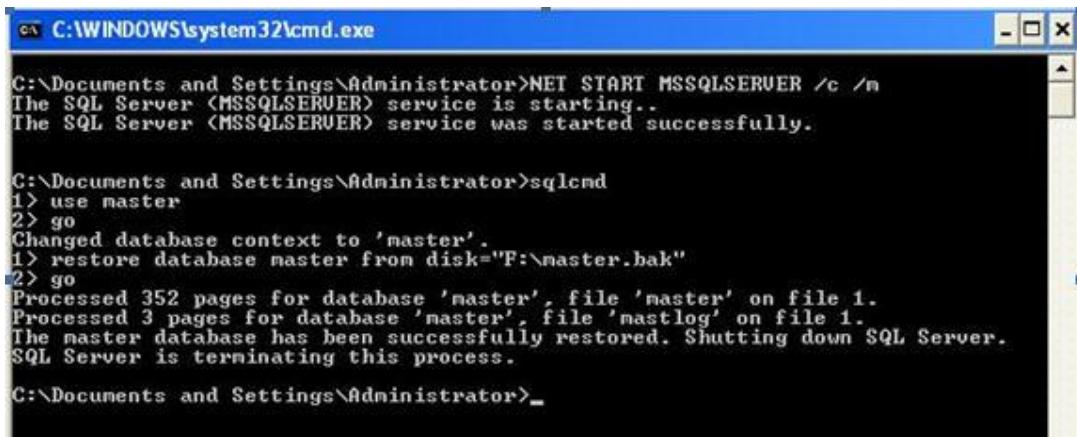
Now we need to restore the system databases from the backup which we took in Step 1.

Master database can be restored only in Single user mode (refer Step 5) and other dbs can be restored normally.

**NOTE :** The ldf and mdf files of the system databases will be overwritten and hence we cannot perform detach/ attach operation.

**Step 5:** In order to restore master database perform the below steps,

- Stop SQL server and start it using the below command from the command prompt
- NET START MSSQLSERVER /c /m which will start SQL in single user mode  
Note: For default instance its MSSQLSERVER, for named instance its MSSQL\$instanceName. Type as shown in the below screenshot. Once the restore is completed SQL server will shut down and hence we need to start it normally and access the databases.



The screenshot shows a Windows Command Prompt window titled 'C:\WINDOWS\system32\cmd.exe'. The command entered is 'NET START MSSQLSERVER /c /m'. The output shows the service starting and then successfully starting. Subsequent commands in SQLCMD show the restoration of the 'master' database from a backup file 'F:\master.bak'. The restoration process is completed, and the SQL Server terminates. The final command is 'C:\Documents and Settings\Administrator>'.

```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\Administrator>NET START MSSQLSERVER /c /m
The SQL Server (MSSQLSERVER) service is starting..
The SQL Server (MSSQLSERVER) service was started successfully.

C:\Documents and Settings\Administrator>sqlcmd
1> use master
2> go
Changed database context to 'master'.
1> restore database master from disk="F:\master.bak"
2> go
Processed 352 pages for database 'master', file 'master' on file 1.
Processed 3 pages for database 'master', file 'mastlog' on file 1.
The master database has been successfully restored. Shutting down SQL Server.
SQL Server is terminating this process.

C:\Documents and Settings\Administrator>
```



## Case Study: Server Collation in SQL Server 2008

While installing SQL Server 2008 we may miss to choose the right collation and we need to rectify this by changing the collation at server level. You can change the collation of sql server without uninstalling. Let's discuss the necessary steps for changing collation for sql server.

Steps for changing collation

- Take backup of all the databases & logins exists in the server for safer side.
- Detach all the user databases
- Insert SQL Server 2008 CD \ DVD into drive.
- Below is the syntax for changing the collation at serverlevel, please note that this will rebuild all the system databases in that instance/.

```
setup.exe /q /ACTION=RebuildDatabase /INSTANCENAME=InstanceName  
/SAPWD="New SA Password" /SQLCollation=CollationName  
/SQLSYSADMINACCOUNTS="Admin ID"
```

Where,

/q - perform silent installation

/Action - We are rebuilding the system databases to change the collation hence the parameter is always RebuildDatabase only

/INSTANCENAME - Name of the instance you are going to change the collation

/SAPWD - Provide new password for SA login

/SQLCollation - Provide the new collation name of SQL Server

/SQLSYSADMINACCOUNTS - Provide a account name which has admin rights in sql server. Please note that this account should be windows authenticated account having sysadmin privilege in sql server

- Once its done check the new collation of sql server
- Attach all the user databases to SQL Server and re-create the logins.
- Check application functionality.

I'm going to test the above steps with an SQL Server 2008 environment which has an existing collation "SQL\_Latin1\_General\_CI\_AS (Latin1-General, case-insensitive, accent-insensitive, kanatype-insensitive, width-insensitive for Unicode Data, SQL Server Sort Order 54 on Code Page 1252 for non-Unicode Data)" to the collation "SQL\_Ukrainian\_CI\_AS (Ukrainian, case-sensitive, accent-sensitive, kanatype-insensitive, width-insensitive for Unicode Data, SQL Server Sort Order 107 on Code Page 1251 for non-Unicode Data)"

Sample exercise to change the collation

- Before changing the collation you can find the collation name.



```
SQLQuery1.sql - W2GZ36Y...)* Object Explorer Details
SELECT @@VERSION as 'Server Version'
SELECT SERVERPROPERTY('COLLATION') as 'Server Collation'

Results | Messages
Server Version
1 Microsoft SQL Server 2008 (RTM) - 10.0.1600.22 (...)

Server Collation
1 SQL_Latin1_General_CI_AS
```

- Execute the below command in Dos prompt to start changing the collation

```
M:\SQL2k8\setup.exe /q /ACTION=RebuildDatabase /INSTANCENAME=SQLEXPRESS
/SAPWD="SQL2K8" /SQLCollation=SQL_Ukrainian_CI_AS_CS_AS
/SQLSYSADMINACCOUNTS="SAGARSYS\Admin"
```

Where M:\ is my DVD drive letter.

```
C:\WINDOWS\system32\cmd.exe
D:\>M:\SQL2k8\setup.exe /q /ACTION=RebuildDatabase /INSTANCENAME=SQLEXPRESS /SAPWD="SQL2K8" /SQLCollation=SQL_Ukrainian_CI_AS_CS_AS /SQLSYSADMINACCOUNTS="SAGARSYS\Admin"
```

- Since this is silent installation it wont ask anything it will just start working on it, once its done the DOS prompt will be like below

```
Microsoft (R) SQL Server 2008 Setup 10.00.1600.22
Copyright (c) Microsoft Corporation. All rights reserved.

D:\>
```

- You can find from the screenshot below is that the new collation will be Cyrillic\_General\_CI\_AI. That's all collation change has successfully completed.

```
SQLQuery1.sql - W2GZ36Y...)* Object Explorer Details
SELECT @@VERSION as 'Server Version'
SELECT SERVERPROPERTY('COLLATION') as 'Server Collation'

Results | Messages
Server Version
1 Microsoft SQL Server 2008 (RTM) - 10.0.1600.22 (...)

Server Collation
1 SQL_Ukrainian_CI_AS
```



## Migrating SQL Server

### Difference between in-place & Side by Side

Upgrade (or in-place upgrade):

- Updates an existing installation while preserving user data
- Instance name remains the same after upgrade
- Automated process

Migration (or side-by-side migration):

- Starts with a new installation
- New & old instance reside side-by-side
- Objects are copied from the old to new instance
- Mostly a manual process

Pre-Upgrade: Preparing the environment:

- Study SQL Server 2008 minimum hardware & software requirements
- Get an inventory of your applications & legacy systems
- Releases, Components, SKU's, Platforms
- Opt for the same or a compatible edition
  - Check features in each SQL Server 2008 SKU
  - Beware of cross-SKU upgrade matrix
- Run Upgrade Advisor
- Examine Upgrade Advisor report
- Fix or work around the backward compatibility issues

Pre-Upgrade: Backward compatibility

- Some features are discontinued:
  - They do not appear in SQL Server 2008
  - Example: Undocumented system stored procedures, Virtual cube, Virtual dimension...
- Some are being deprecated:
  - They won't be supported in the release following SQL Server 2008
  - Example: SQL Mail, Calculated Cell, Cell evaluation list...
- Some editions have a different feature set
  - Example: Express does not have SQL Server Agent
- Check Books Online for a full list as well as replacements and techniques
- Run Upgrade Advisor before any migration or upgrade

**SQL Server 2000 Data Transformation Services**

Importance	When to fix	Description
[+]	Before	You can use SQL Server 2005 tools to edit your existing DTS packages. However, upgrading or uninstalling the last instance of SQL Server 2000 on a computer removes the components required to support this feature. You can retain or restore these components by installing the special Web download, "SQL Server 2000 DTS Designer Components", before or after you upgrade or uninstall SQL Server 2000.
[+]	Before	<b>SQL Server 2000 Meta Data Services packages are not supported.</b>
[+]	After	No DTS packages were found on the server that you selected to analyze. <a href="#">Show affected objects... (1 objects)</a> <a href="#">Tell me more about this issue and how to resolve it.</a> <input type="checkbox"/> This issue has been resolved.
[+]	Advisory	Upgrade Advisor reports only on the most recent version of each DTS package.
[+]	Advisory	SQL Server 2000 Data Transformation Services is deprecated.

Generated on: STU01, 12/15/2005 1:00:11 PM



### Advantages/Disadvantages of In-Place to Side-by-Side

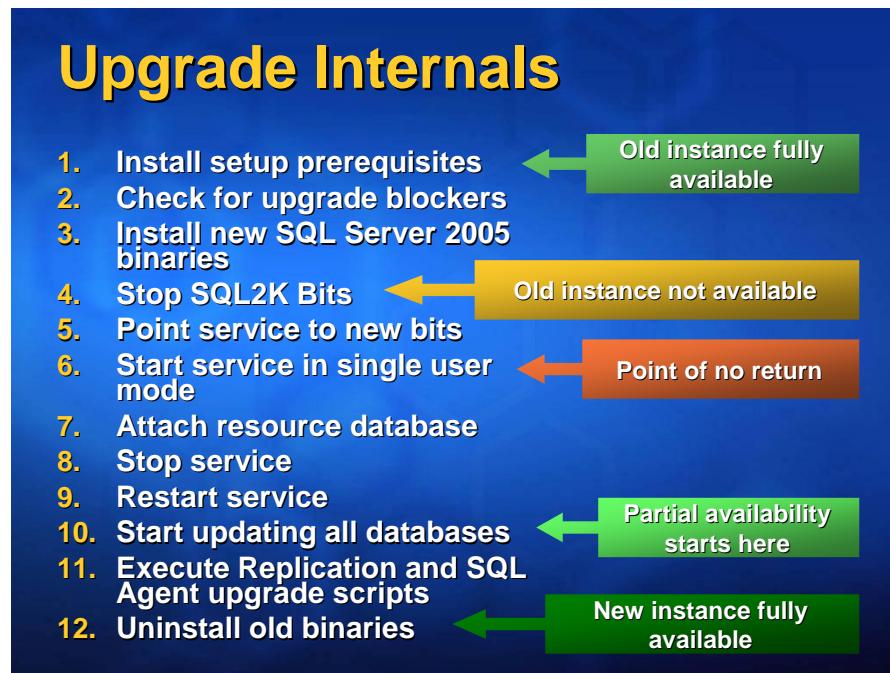
Side By Side Migration:

- 👉 Advantages
  - 👉 Migration provides more granular control over the upgrade process
  - 👉 Having new and old instances side-by-side helps with testing & verification
  - 👉 Legacy instance remains online during migration
  - 👉 Flexibility to implement migration with failover
- 👉 Dis-advantages
  - 👉 May require new or additional hardware resources
  - 👉 Applications need to be directed to new instance

In-Place Migration/Upgrade:

- 👉 Advantages
  - 👉 Easier, faster, less headache for small systems
  - 👉 Requires no additional hardware
  - 👉 Applications remain pointing to old instance
- 👉 Dis-advantages
  - 👉 Less granular control over upgrade process
  - 👉 Instance remains offline during part of upgrade
  - 👉 Not best practice for all components
  - 👉 Analysis Services cubes are recommended to be migrated

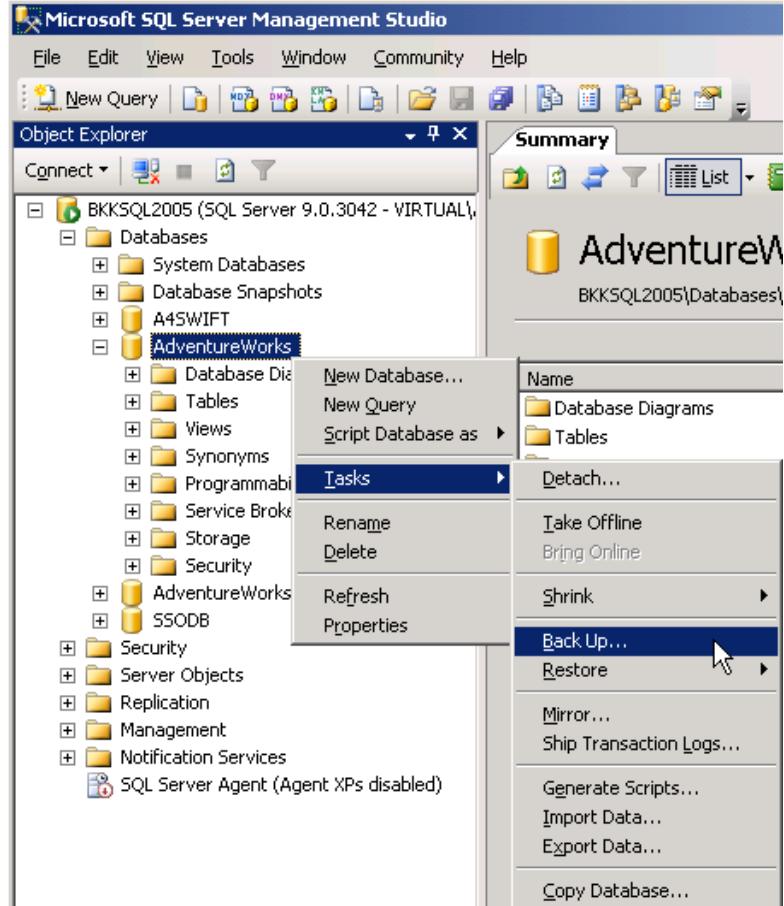
### In-Place Upgrade Internals



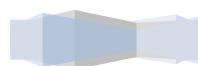
## 1. Migrating the database by using Backup & Restore

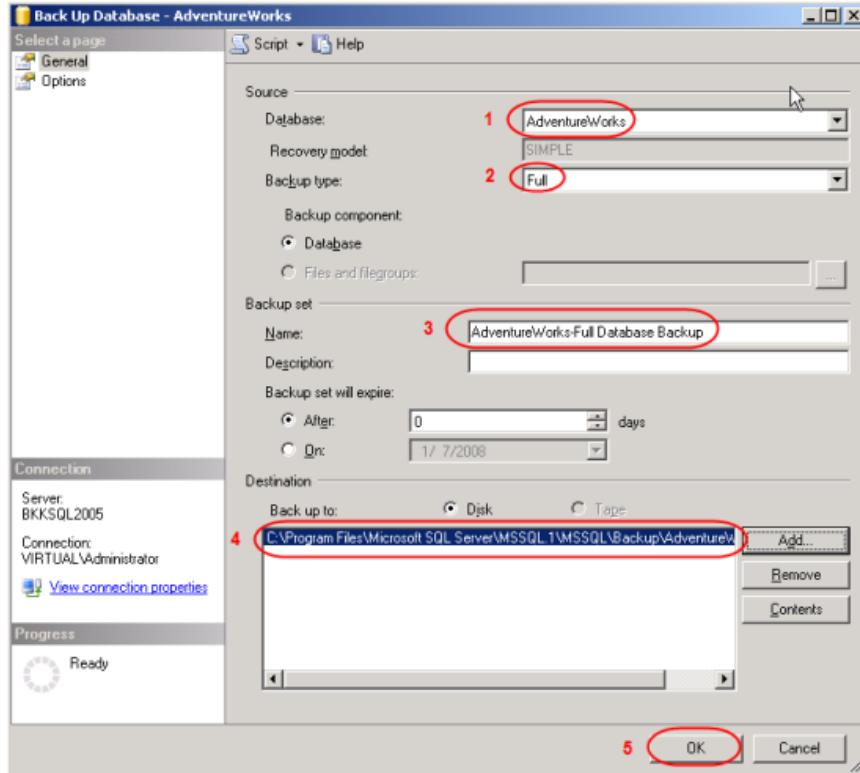
By using backup, you can backup a database without interrupt any transactions on the database. I will backup a database from SQL Server 2005 and restore the database to another SQL Server 2005/2008 instance.

1. Connect to source server. Open Microsoft SQL Server Management Studio and connect to BKSQL2005.
2. Right-click on the Adventure Works database. Select Tasks -> Backup...

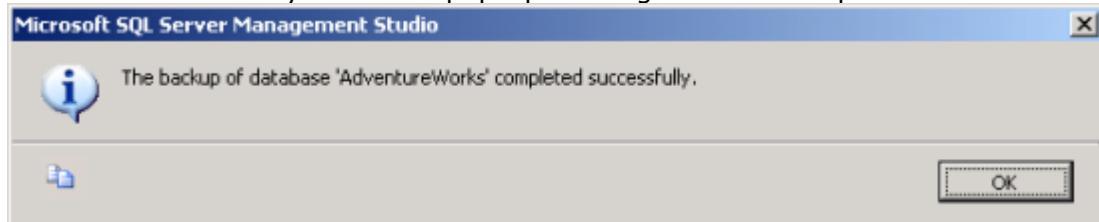


3. On Back up Database window, you can configure about backup information. If you're not familiar these configurations, you can leave default values. Here are some short descriptions.
  1. Database – a database that you want to backup.
  2. Backup type – you can select 2 options: Full and Differential. If this is the first time you backup the database, you must select Full.
  3. Name – Name of this backup, you can name anything as you want.
  4. Destination – the file that will be backup to. You can leave as default. Default will backup to "C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Backup".
  5. Click OK to proceed backup.





4. Wait for a while and you'll see a pop-up message when backup is finished.



5. Browse to the destination, you'll see a backup file (.bak format) which you can copy to other server for restore in the next step. Default backup directory is "C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Backup".

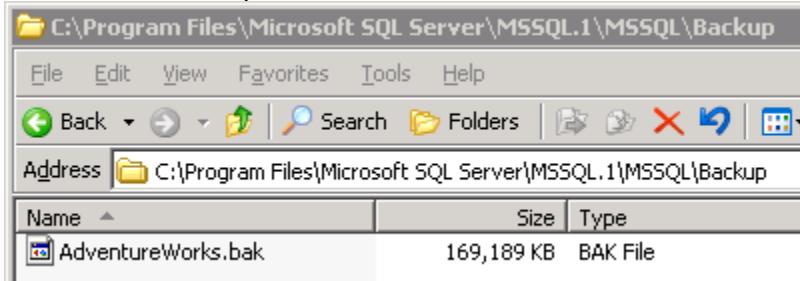


### Restore the database.

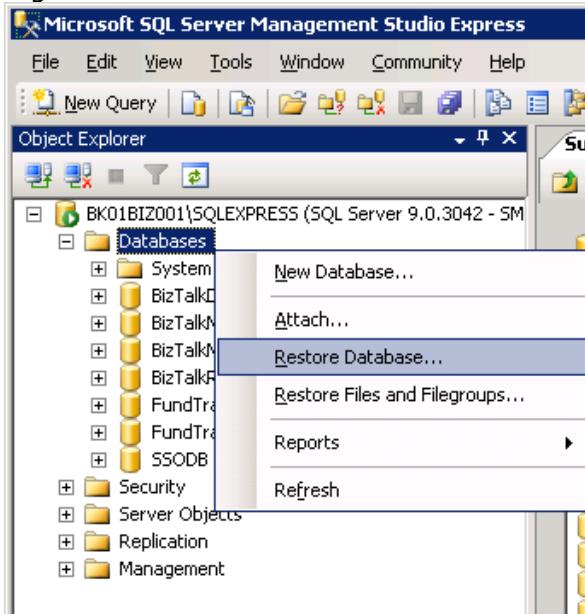
Next, I will restore the Adventure Works database from a file that I've created above to BK01BIZ001 which runs Microsoft SQL Server Express Edition.



1. Copy the backup file from source server to destination server. I've copied into the same directory as source server.

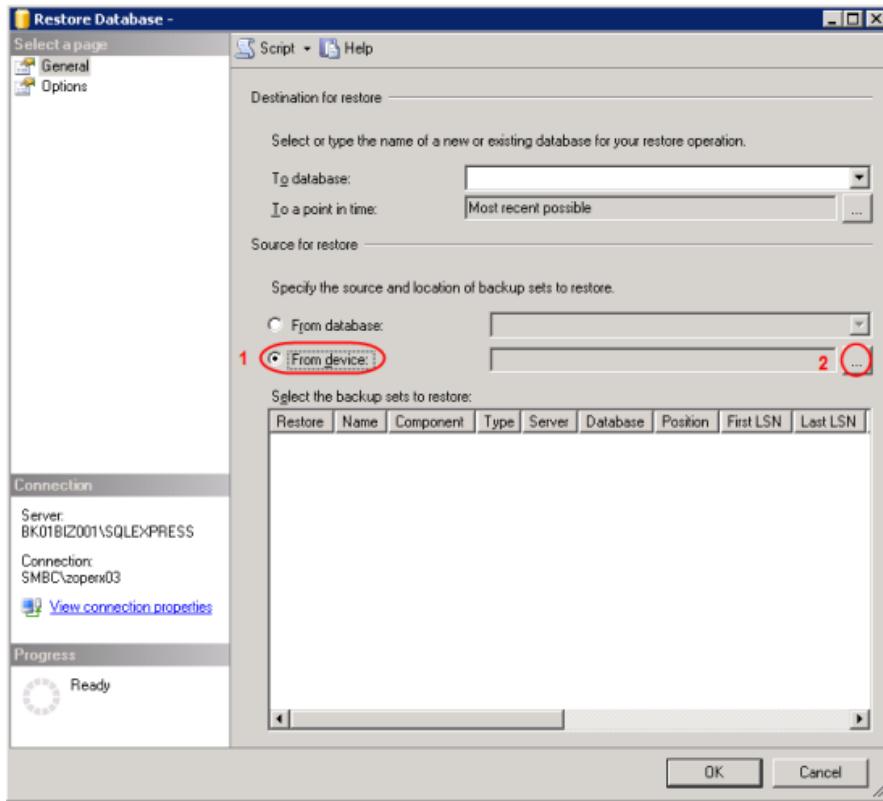


2. Connect to destination server. Open Microsoft SQL Server Management Studio Express and connect to BK01BIZ001.
3. Right-click on Databases. Select Restore Database...

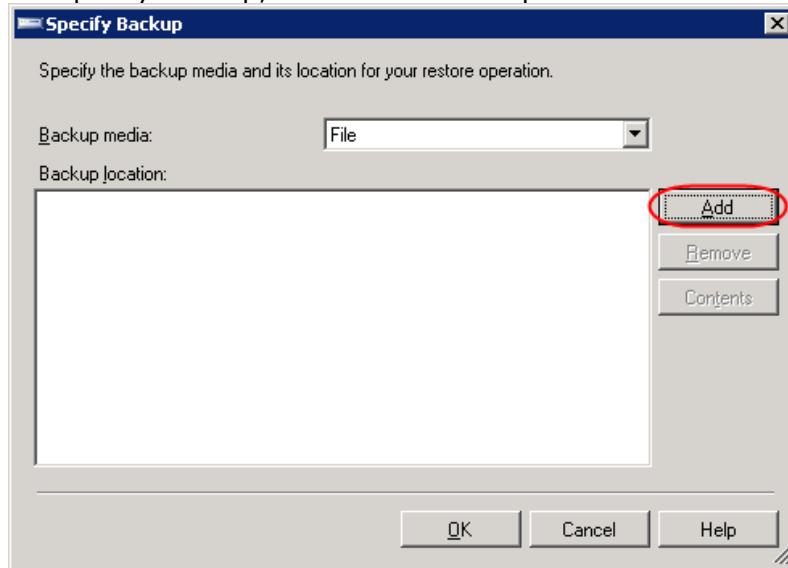


4. Restore Database window appears. On Source for restore, select from device and click [...] button to browse file.

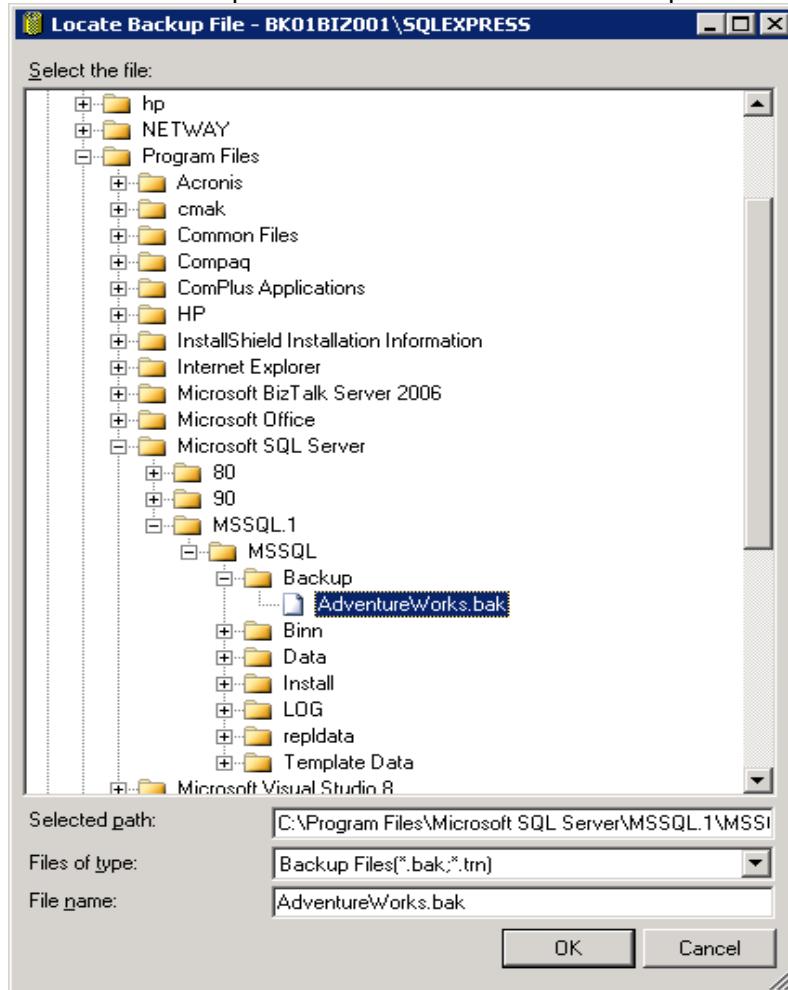




5. On Specify Backup, ensure that Backup media is "File" and click Add.



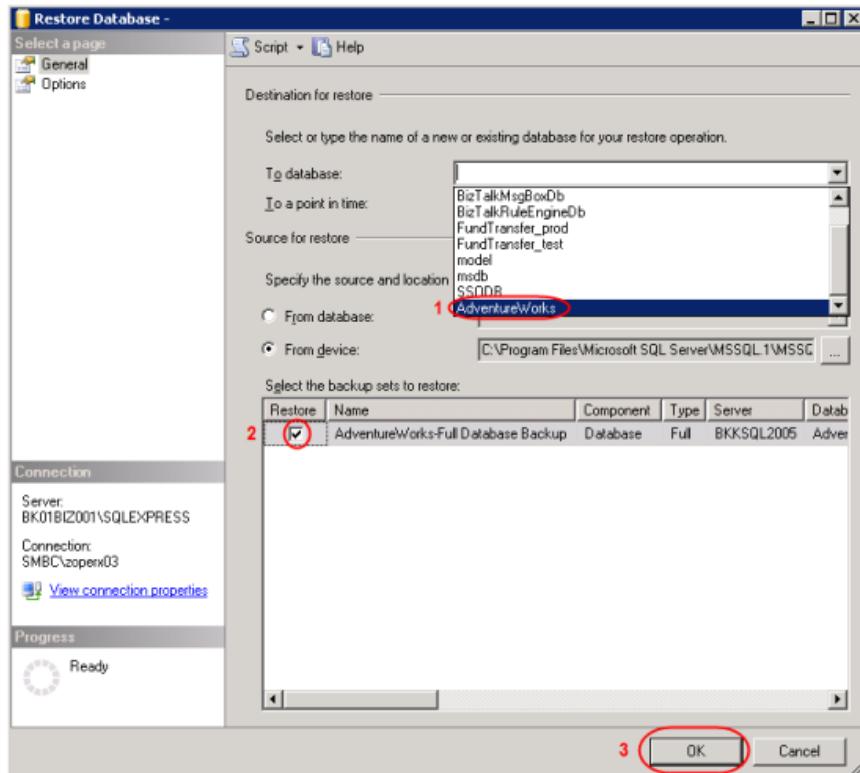
6. On Locate Backup File, select the backup file. This is the backup file that was created in Backup a database section and was copied to this server. Click OK.



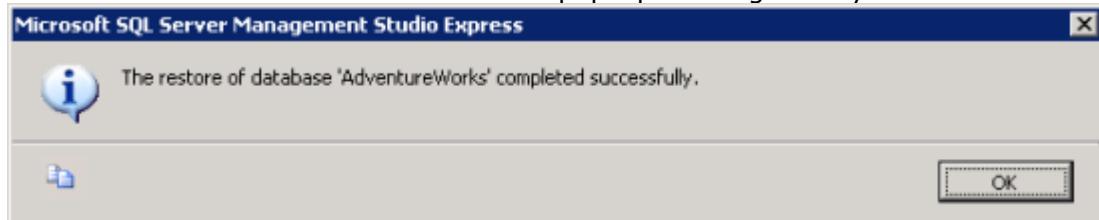
7. Back to Restore Database window.

1. On Destination for restore, select "Adventure Works".  
*Note: If you haven't added the backup file on Source before (step 4-6), you won't see the database name on Destination.*
2. On Source for restore, check the box in front of the backup name (in Restore column).
3. Click OK.

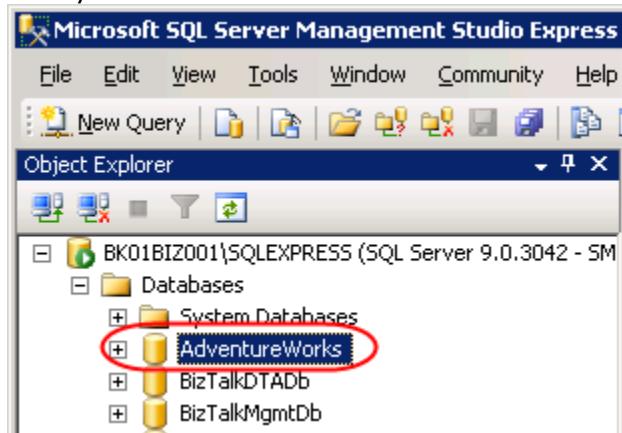




- Wait until restore finish and there'll be a pop-up message notify.



- Now you'll see the restored database on the destination SQL Server.



## 2. Migrating the database using copy database wizard

What is copy database wizard?

Copy Database Wizard is a new feature from SQL Server 2005 onwards. You can make use of this feature to copy \ move databases between different instances of SQL Server. It can be used for the below purposes

- Transfer a database when the database is still available to users by using the SQL Server Management Objects (SMO) method.
- Transfer a database by the faster detach-and-attach method with the database unavailable during the transfer.
- Transfer databases between different instances of SQL Server 2005.
- Upgrade databases from SQL Server 2000 to SQL Server 2005.

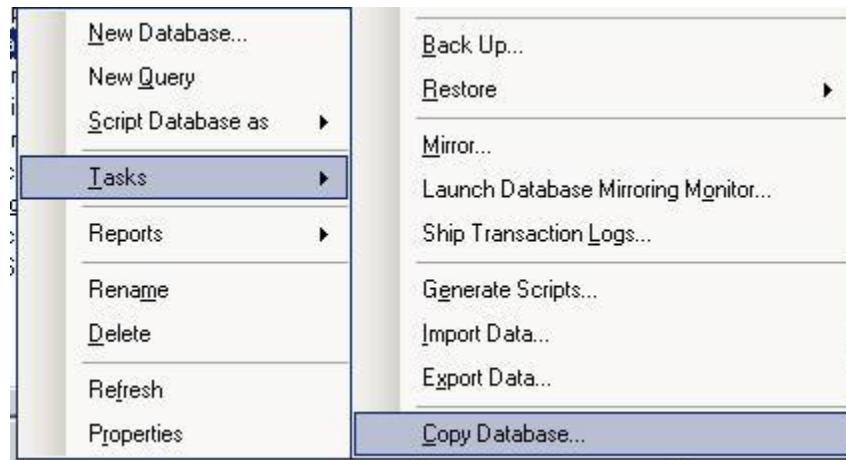
**Note:** The Server from which you are running CDW should be patched with minimum SQL Server SP2 (better update with latest SP) for Copy Database Wizard (CDW) to work properly

### Permission Required:

To use the Copy Database Wizard, you must be a member of the sysadmin fixed server role on the source and destination servers. To transfer databases by using the detach-and-attach method, you must have file system access to the file-system share that contains the source database files.

### How to Use it?

- Open SSMS in source or destination server which is running SQL Server 2005
- Right click on any of the database and then click on Tasks from the select Copy database wizard as shown below

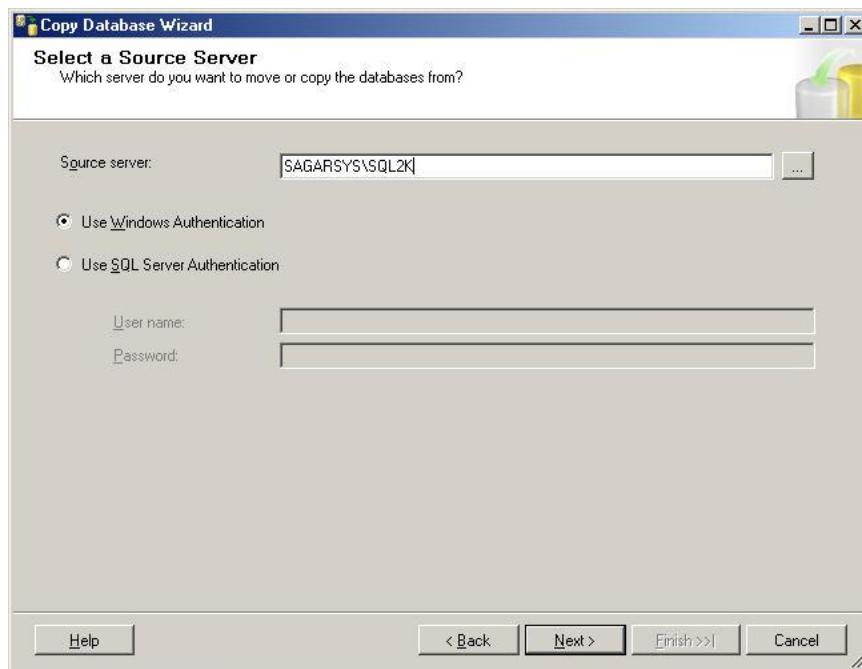


- Once opened you can see the welcome screen as below



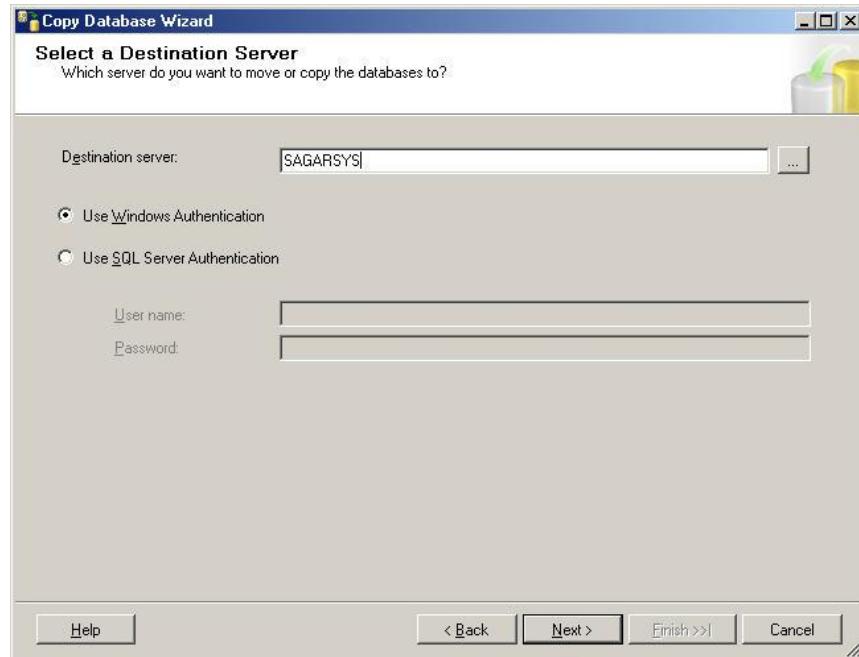


- Click on next to proceed with wizard, In this screen you need to provide the source server name and the credentials

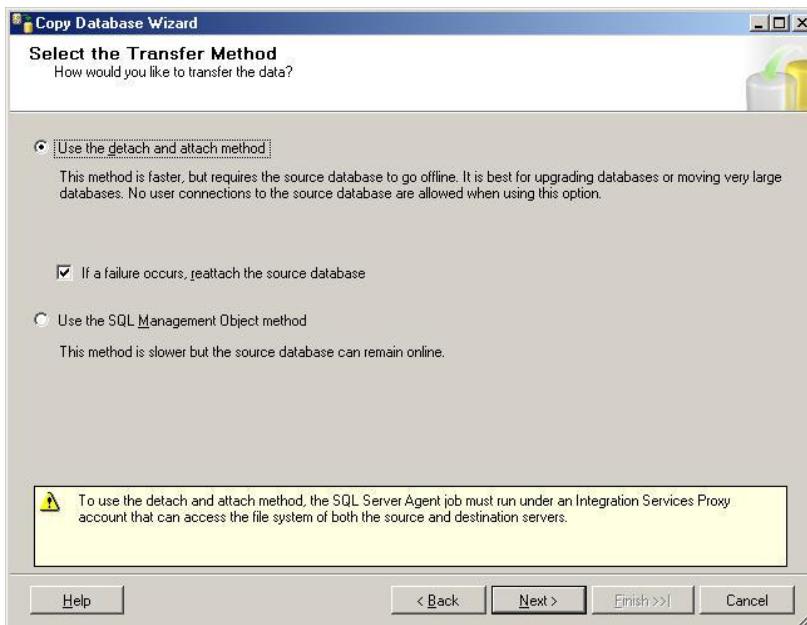


- After this click on next and in this screen provide the destination server(which should be SQL Server 2005) and its credentials



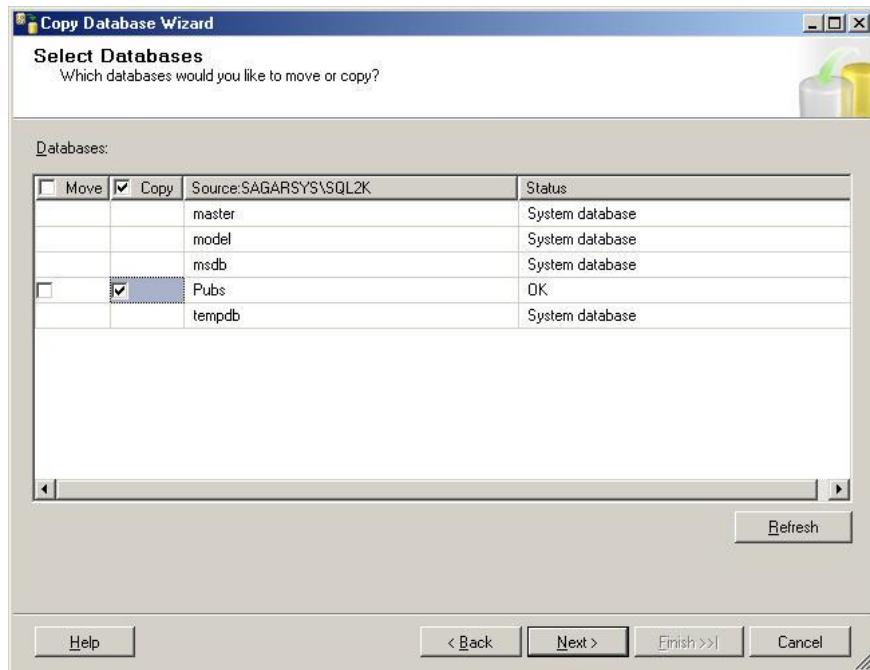


- Once source & destination server details given, you need to select the way by which you are going to copy \ move the database.
  - Detach \ Attach — Faster methods, requires db to be offline. Users will be disconnected and physical files of the db will be copied to the destination server
  - SMO — Slower method, db will be in online state. This will create the db in the destination server with the same name and copy all the data's from source

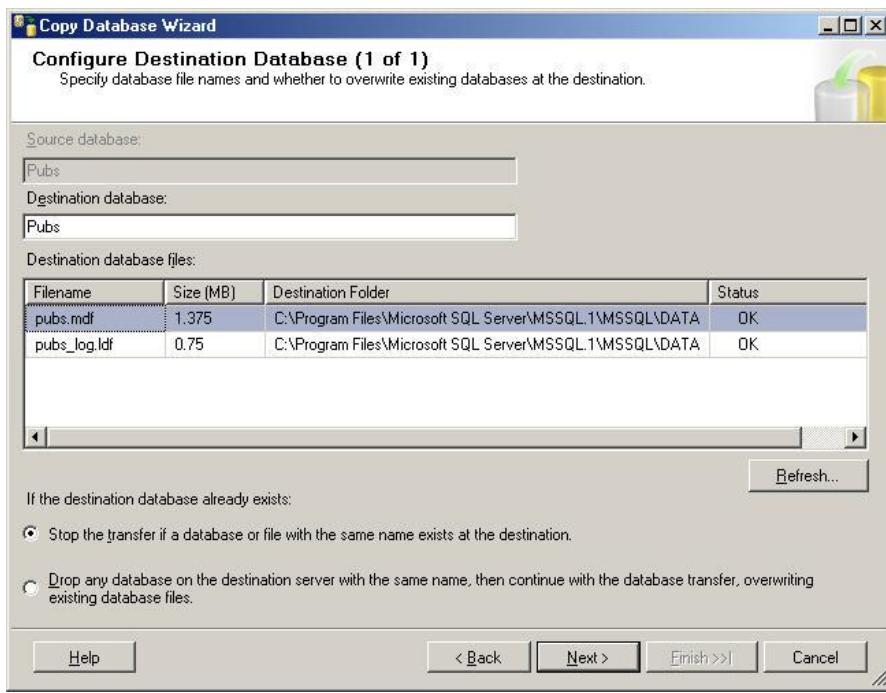


- Once the movement method chosen you can select the databases from the source server and you need to specify whether its move or copy.

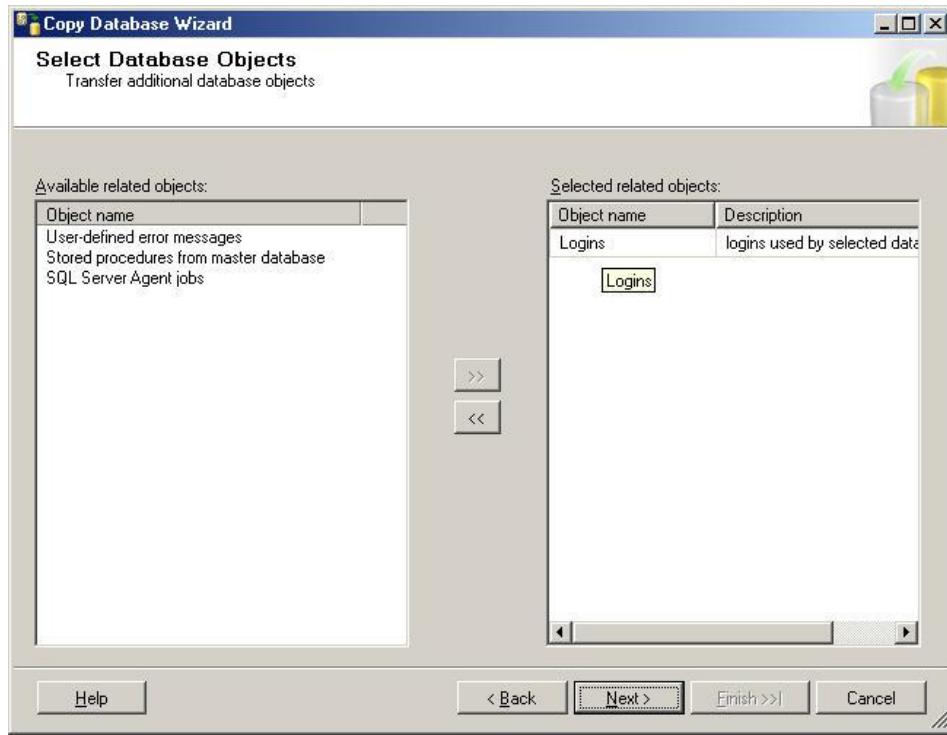
- Copy — Will copy the db to the destination server and the database will be online in both the servers
- Move — In this method the wizard automatically deletes the source database after moving the database to destination



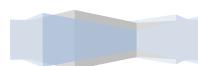
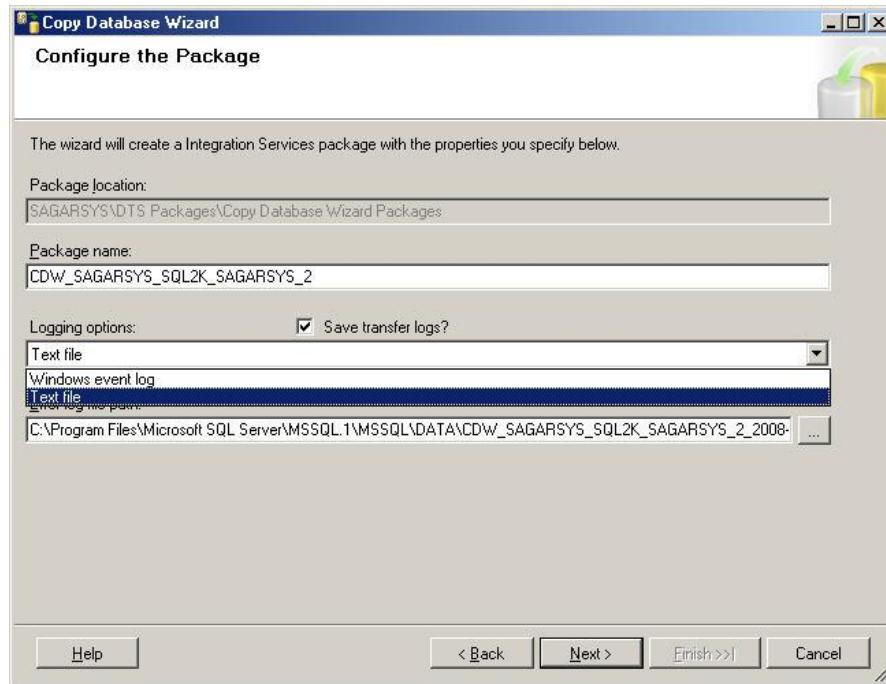
- In the next page you need to provide the new db name and the path where CDW should place the physical files in the destination server



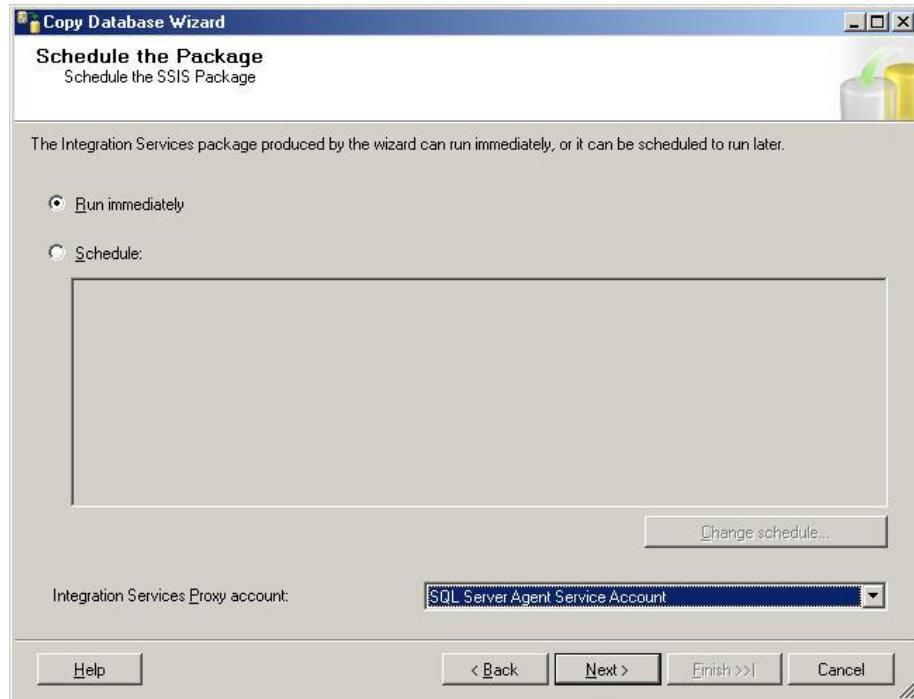
- The next page is the good feature in CDW, here you can select the logins, objects, jobs & SP which is related to the database you are trying to copy making our job simpler. Here I'm just copy the logins alone.



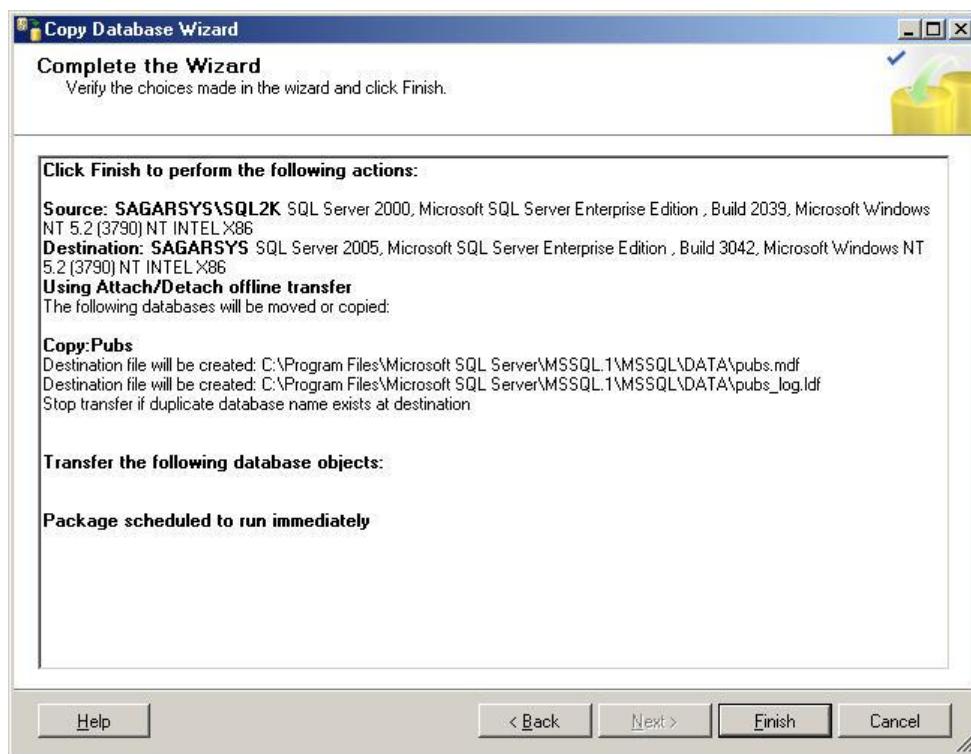
- In the next page you need to provide the package name and the log file for this process, so that incase of failure you can review it.



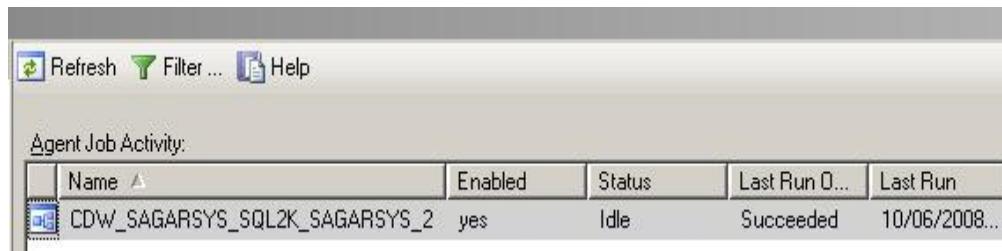
- In the next window you can select the package to execute immediately or else you can schedule the same to run after some time after EOD



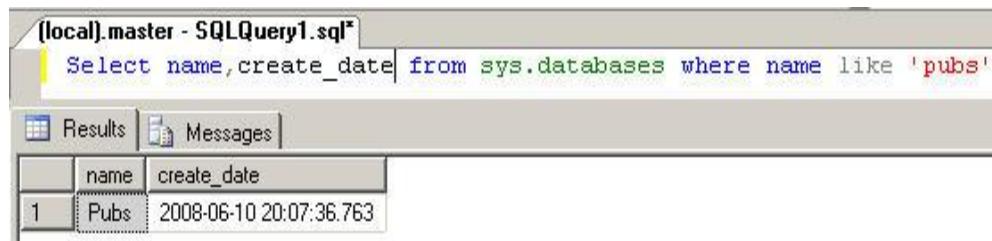
- You can review the full summary before it proceeds as shown in the below screenshot



- Clicking on finish will create a job with the name mentioned in "Configure the package" page. If you have selected to run immediately the job will be scheduled to run in the near time or else it will be scheduled as given in the page.
  - Note:** Once the db copy \ movement is done you can delete the job or else it will be stayed in your job list.



- Once it succeeded the db will be moved \ copied to the destination server. You can query sys.databases catalog view to check the same.



### Considerations:

- You cannot move system databases
- Selecting move option will delete the source db once it moves the db to destination server
- If you use SMO method to move full text catalogs then you need to repopulate it
- SQL Server Agent should be running or else it will fail in job creation step
- You can't move encrypted objects (like objects, certificates etc) using CDW

### 3. Migrating the database by using Detach & Attach method.

If database is to be from one database to another database, the following script can be used detach from old server and attach to new server.

Process to move database :

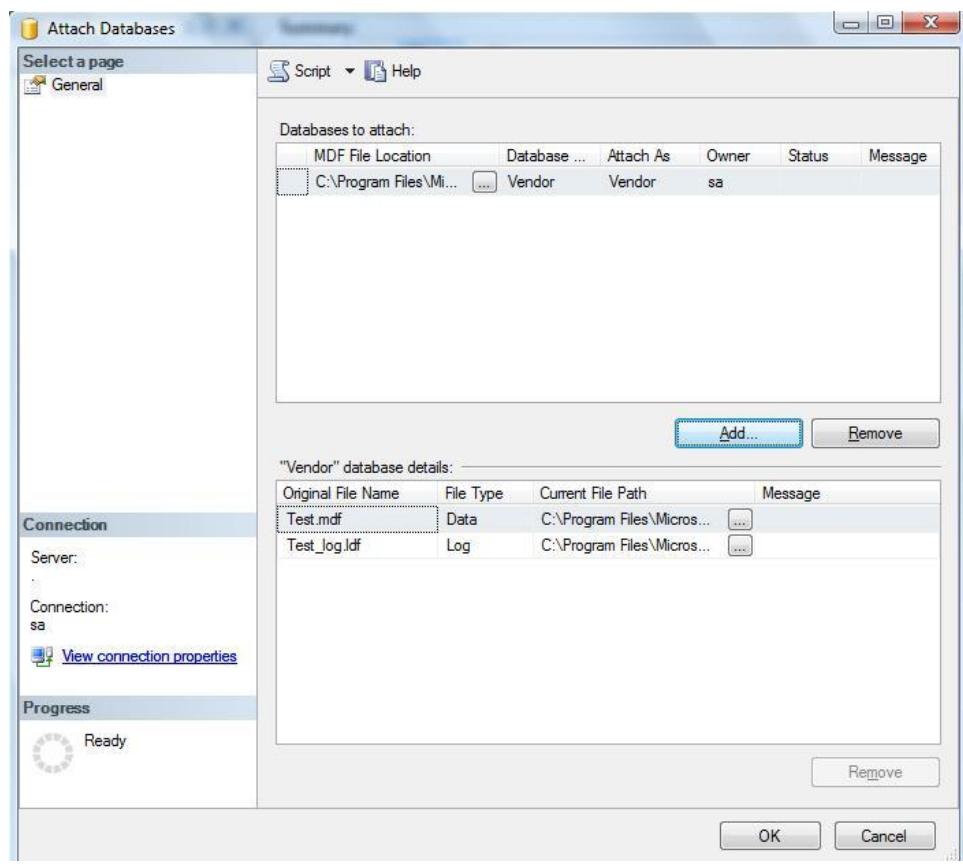
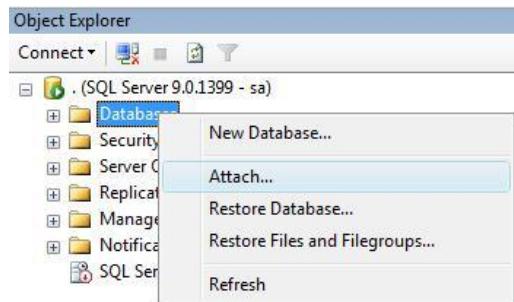
```
----Step 1 : Detach Database using following script
USE [master]
EXEC master.dbo.sp_detach_db @dbname = N'AdventureWorks',
GO
```



----Step 2 : Move Data files and Log files to new location

--Step 3 : Attach Database using following

Open Object Explorer in SQL Server and right click on Databases and select Attach menu item as you see in the below image.

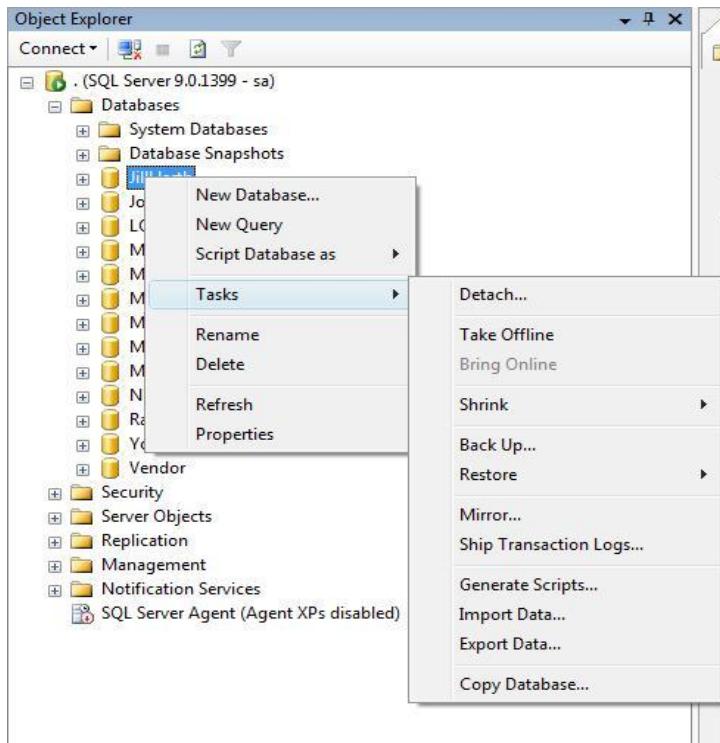


Now, select a file option and use Browse button to browse your MDF and LDF files as seen in below image.

That's it. Your database is attached now and ready to use.

The reverse operation of Attach is detach. The Detach option removes the database from the Databases list in the Object Explorer and saves the files to the given location.

Right click on the database name you would like to detach and select Detach menu item.



### Migrating Logins & Fixing Orphaned Users:

#### To Detect Orphaned Users:

To detect orphaned users, execute the following Transact-SQL statements:

```
USE <database_name>;
GO;
sp_change_users_login @Action='Report';
or
EXEC sp_change_users_login 'Report';
```

103

Note: **sp\_change\_users\_login** cannot be used with SQL Server logins that are created from Windows.



## To Resolve an Orphaned User

-----Script to find orphahend users-----

```
exec sp_change_users_login 'Report'
```

Use Master

```
SET NOCOUNT ON
```

```
SELECT 'EXEC sp_addlogin @loginame = "' + loginname + """
      , @defdb = "' + dbname + """
      , @deflanguage = "' + language + """
      , @encryptopt = "skip_encryption"""
      , @passwd =' 
      , cast(password AS varbinary(256))
      , @sid =' 
      , sid
FROM sys.syslogins
where loginname Not like 'NA%' and
      loginname not like 'Builtin%' and loginname Not like 'sa'
```

Run the above script on Source server copy the result and execute on Destination server

Eg:--

```
EXEC sp_addlogin @loginame = 'CorpCommUser' , @defdb = 'CorpComm'      ,
@deflanguage = 'us_english'      , @encryptopt = 'skip_encryption' , @passwd =
      0x01003F04413C64CEE4767BA2DD0053A02C6056640C4C88C24DFA , @sid
=      0xCEE1766A76520E43A98DCB141B031F7E
```

Mapping a database user to a new SQL Server login:

```
-- --Map database user MB-Sales to login MaryB.
EXEC sp_change_users_login 'Update_One', 'MB-Sales', 'MaryB';
```

Automatically mapping a user to a login:

how to use Auto\_Fix to map an existing user to a login of the same name, or to create the SQL Server login Mary that has the password B3r12-3x\$098f6 if the login Mary does not exist.

```
EXEC sp_change_users_login 'Auto_Fix', 'Mary', NULL, 'B3r12-3x$098f6';
```

## Migrating DTS Packages to SSIS

Determining how you will upgrade your DTS packages to SQL Server Integration Services (SSIS) is the first step in creating a DTS-to-SSIS migration strategy.

To start the upgrade, run the SQL Server 2005 Upgrade Advisor, which has an option specifically for DTS. Since everything has changed between DTS and SSIS, the upgrade will not be easy, but that shouldn't concern you as long as you understand the process and what you'll need to do going forward.

### What happens during the upgrade?

When making the upgrade, you will need to determine how you will manage these legacy DTS packages in the future. The DTS runtime will continue to be available, and will have been updated to enable DTS packages to connect SQL Server 2005 data sources. But the DTS designer components are removed along with SQL Server 2000 Enterprise Manager.

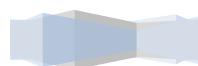
You cannot connect to a SQL Server 2005 instance using the SQL Server 2000 Enterprise Manager. But SQL Server Management Studio, which is the SQL Server 2005 replacement for Enterprise Manager, supports DTS packages.

### Downloading designer tools

In the Object Explorer window, under the Management Legacy nodes, you will find Data Transformation Services. This is the equivalent of local packages, and is the same table as mentioned above. You can import packages and start the DTS-to-SSIS migration wizard from there, but to do any editing work or manage packages you must download and install the Microsoft SQL Server 2005 DTS Designer Components. These can be found at the Microsoft download center as part of the feature pack for SQL Server 2005.

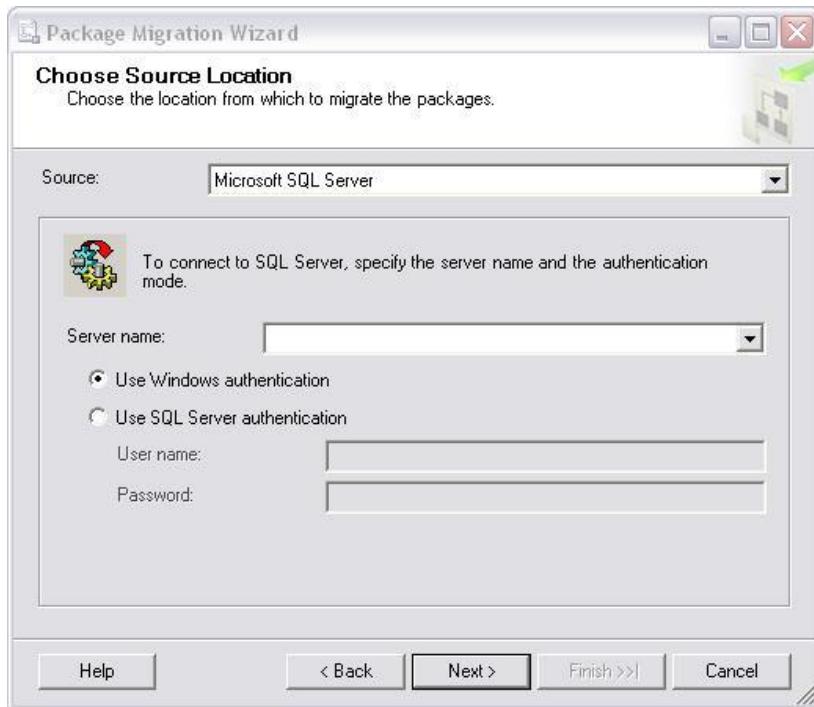
With the full DTS Designer, you can now create or edit DTS packages as you have done in the past. SQL Server Management Studio does not support Meta Data Services, so you will not be able to enumerate or edit packages stored there.

The first screen that will hit you is the splash screen. On here you can choose if you want to see the screen again or not



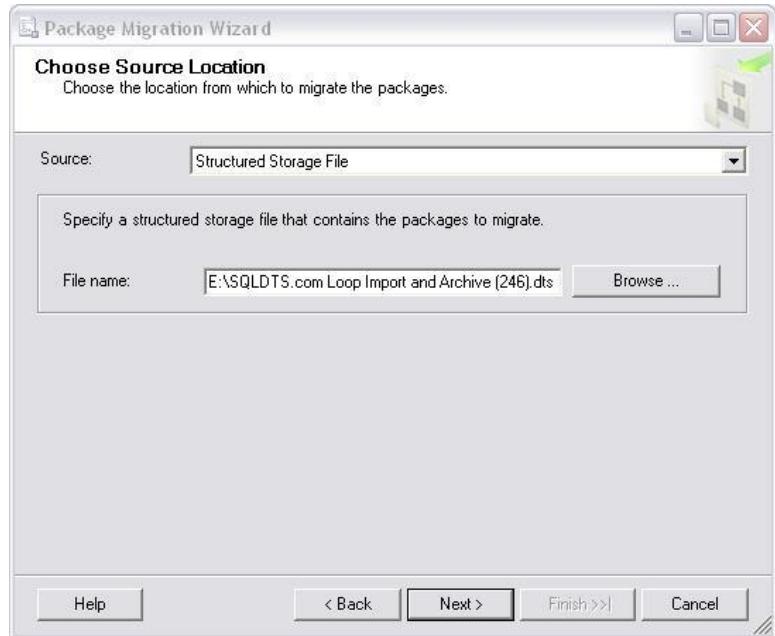


Now you have to choose from where to get your DTS package

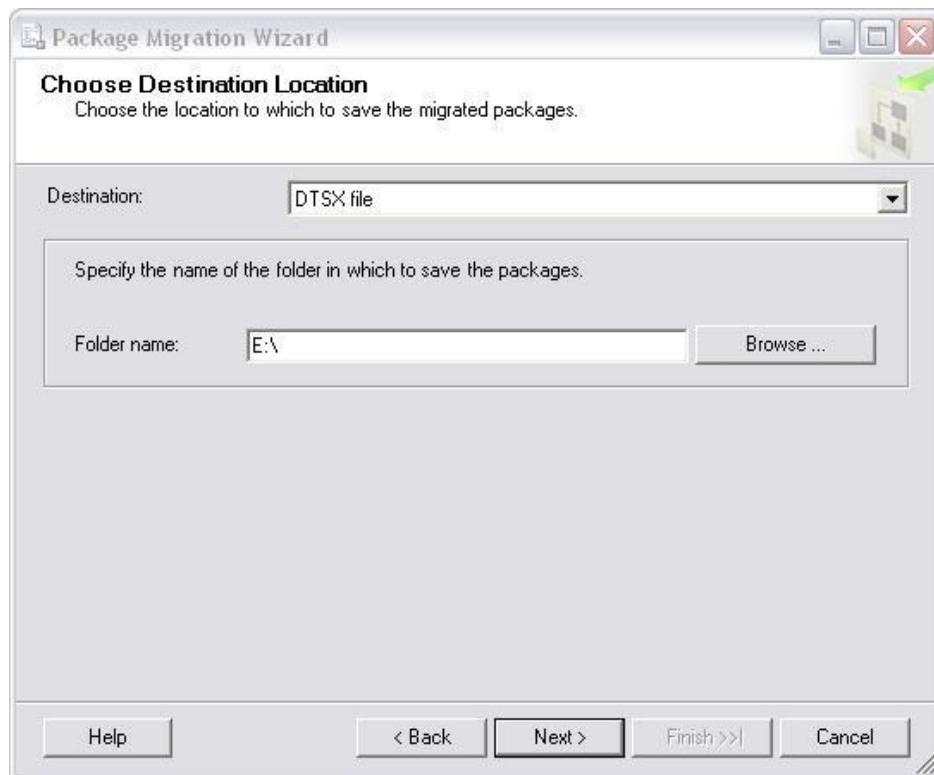


We have used a Structured Storage File



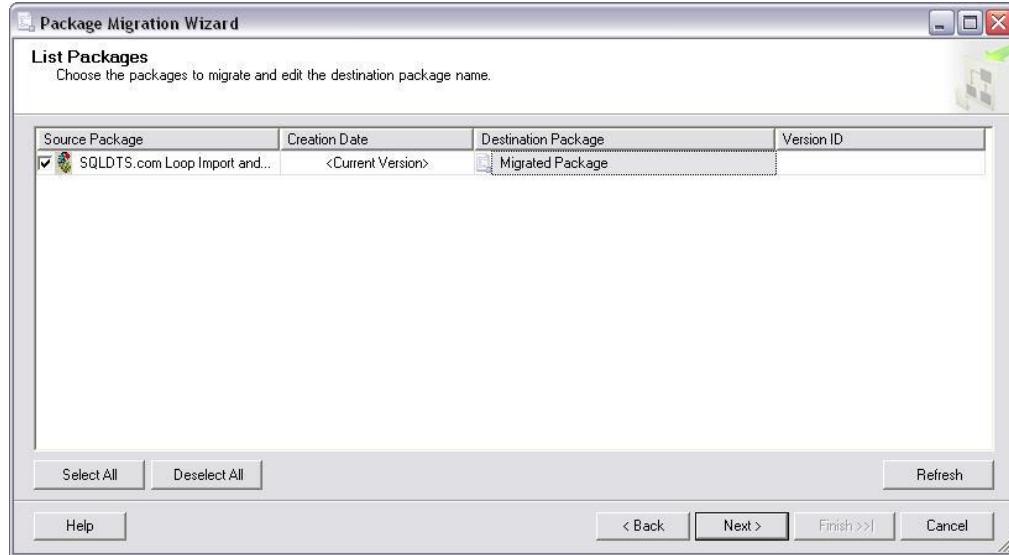


Now you need to tell the wizard into which directory you want to save the package

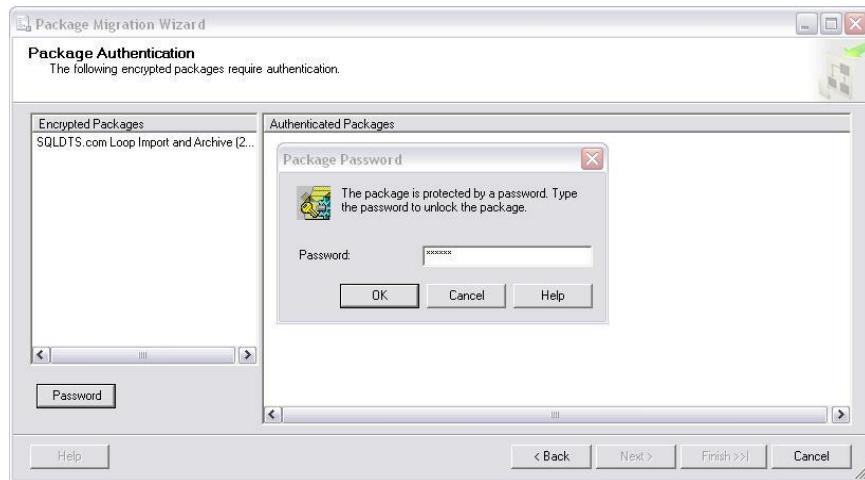


Because a Structured Storage File can store more than one package we need to tell the wizard which package to migrate



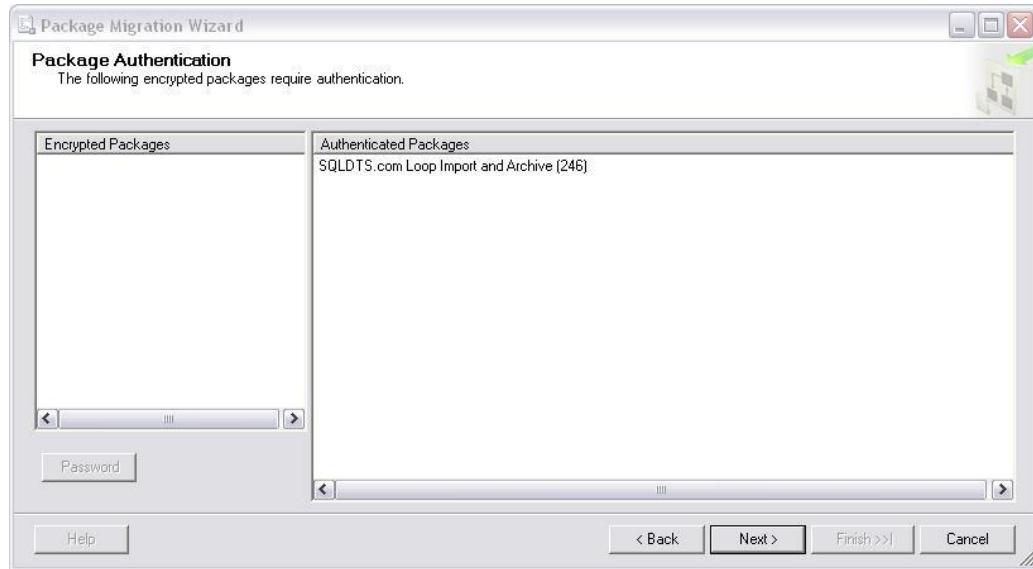


If your package is protected by a password then you will need to supply it here



You should then have an authenticated package



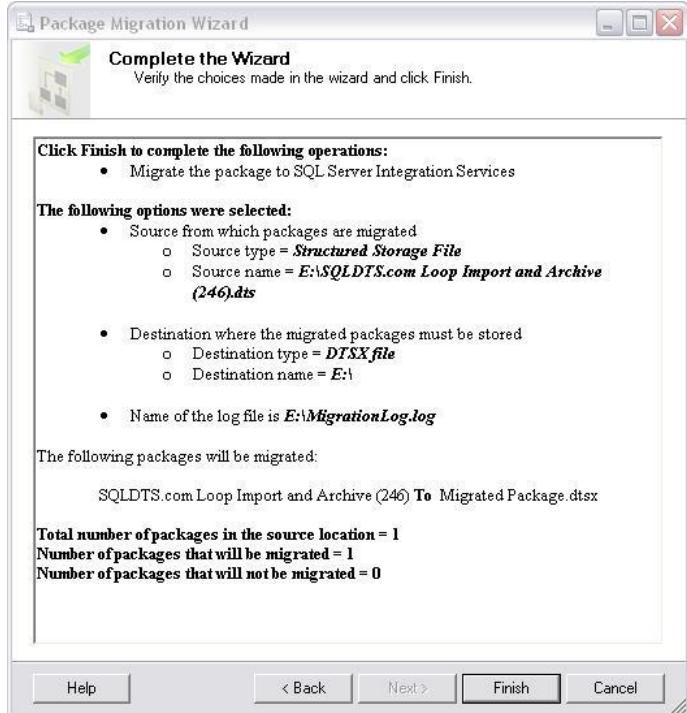


Where do you want to store a log detailing what happened during the migration?

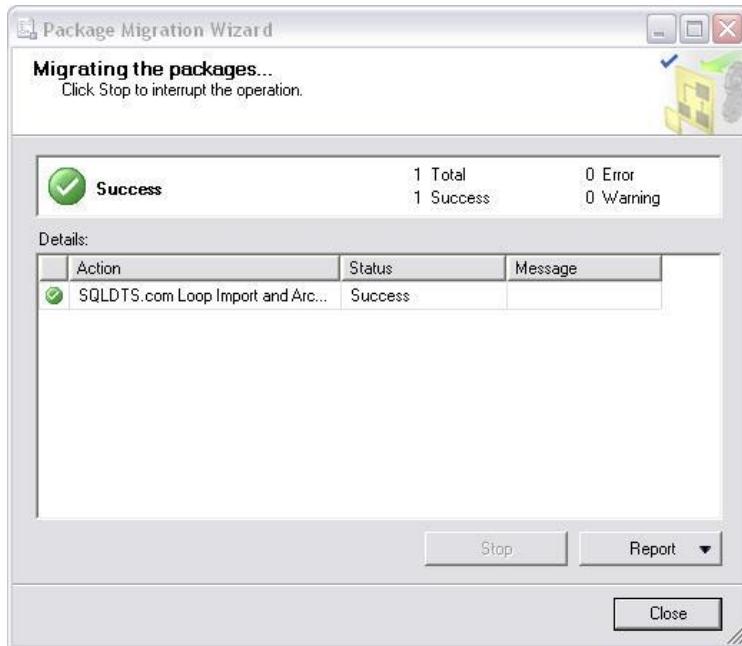


On this screen we can see what the wizard is intending to do



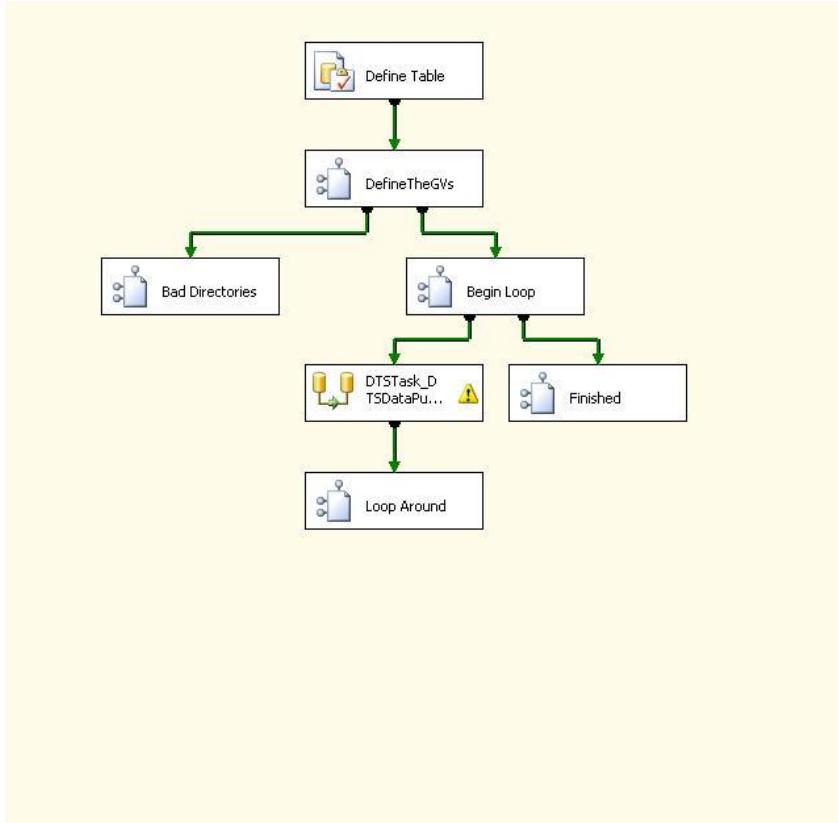


After the wizard completes we get a dialog box.



This is what the migrated package looks like





## The Import and Export Wizard

The import and export wizard was available even with SQL 2000 has remained an important tool for exporting from and importing into SQL Server data from many different kinds of data sources. It can also be used for transferring data between non-Microsoft data sources. In this I will show how to transfer data from MS Excel spreadsheet data to SQL Server 2008. In any of the transformations it is important to realize that data types used in data sources are not exactly the same and that there are differences to be reckoned with. The basic steps to take are to indicate the source of data and the destination to which it needs to be transferred. In order to match the differences some mappings may be necessary if the source and destination are not both SQL Servers.

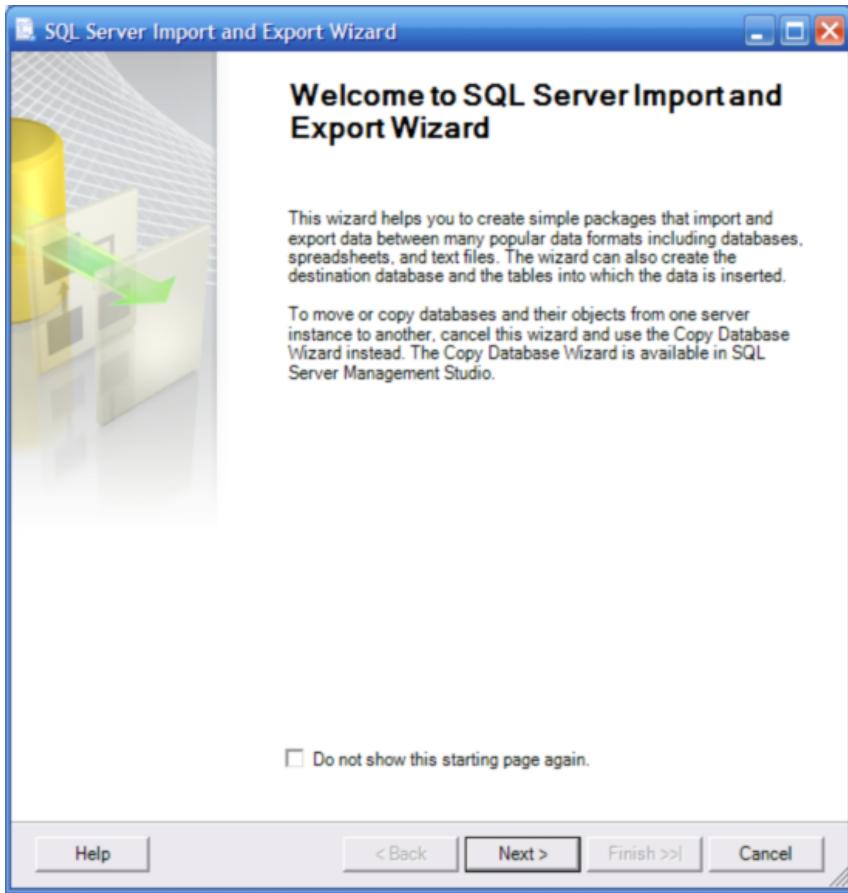
The MS Excel file PrincetonTemp.xls used in this example is a simple spread sheet data that shows the temperature variations during a year and the maximum recorded temperature. The data type used for the column 'Month' is text and of the others are numbers.



Id	Month	Temperature	RecordHigh
1	Jan	40	60
2	Feb	32	50
3	Mar	43	65
4	Apr	50	70
5	May	53	74
6	Jun	60	78
7	Jul	68	70
8	Aug	71	70
9	Sep	60	82
10	Oct	55	67
11	Nov	45	55
12	Dec	40	62

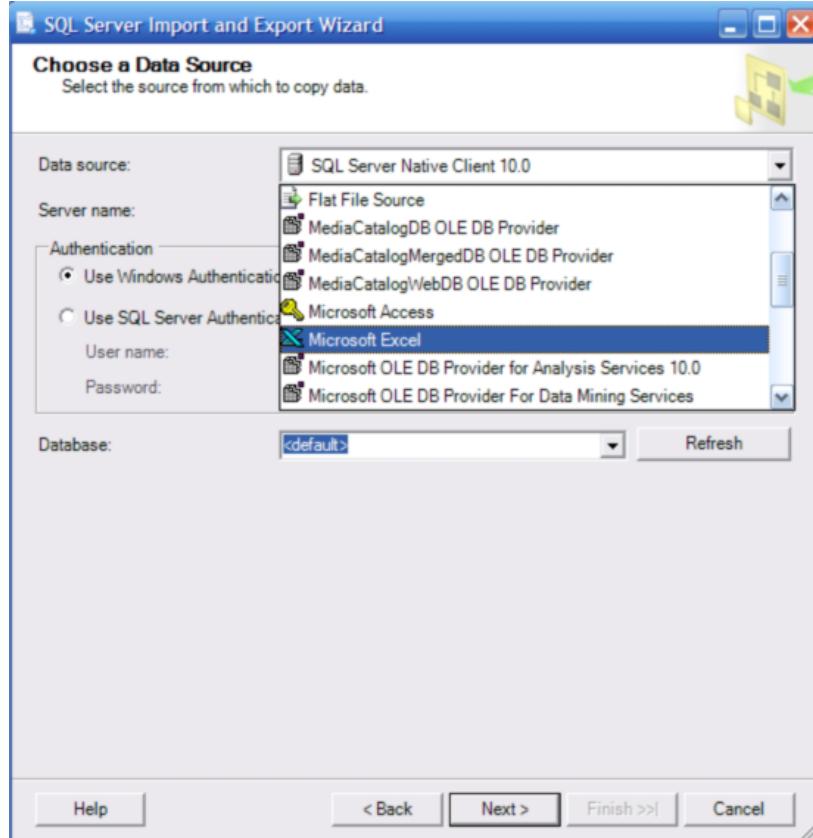
## Invoke the Import and Export Wizard

Bring up the Import and Export wizard from Start → All Programs→ Microsoft SQL Server 2008 →Import and Export Data (32 bit). This pops-up the Welcome Wizard as shown. Make sure you read the explanations provided.



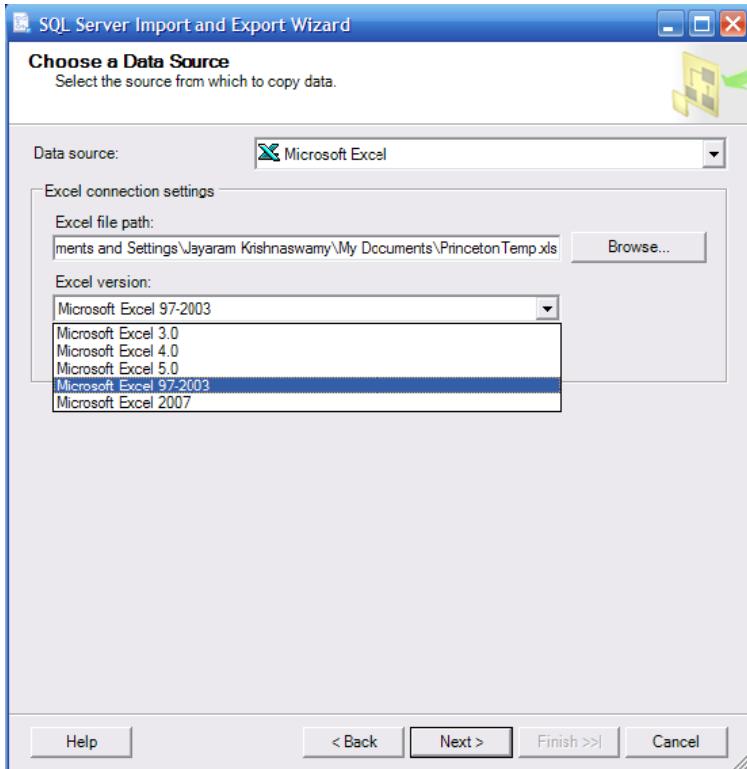
Click Next. The default page gets displayed. In the 'Choose a Data Source' page click on the handle along the data source and choose Microsoft Excel file as the data source as shown.





Click Next. The 'Select the source from which to copy data' shows up. Use the Browse...button to bring in the location information of PrincetonTemp.xls to the window as shown. The Excel version displayed by default (Microsoft Excel 97-2003) is proper for the MS Access version used in this article. Keep the 'First row has column names' option checked. Note that the MS Access 2007 is not supported.

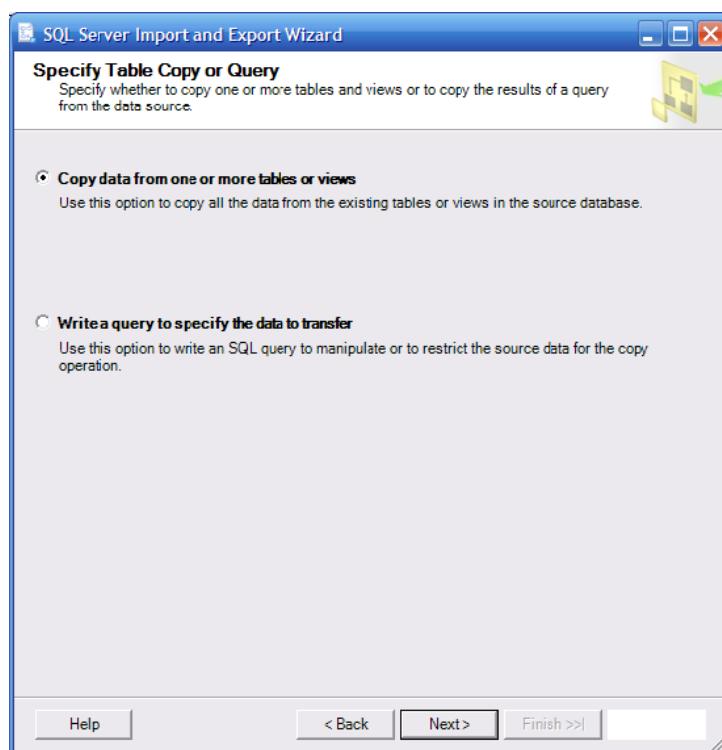
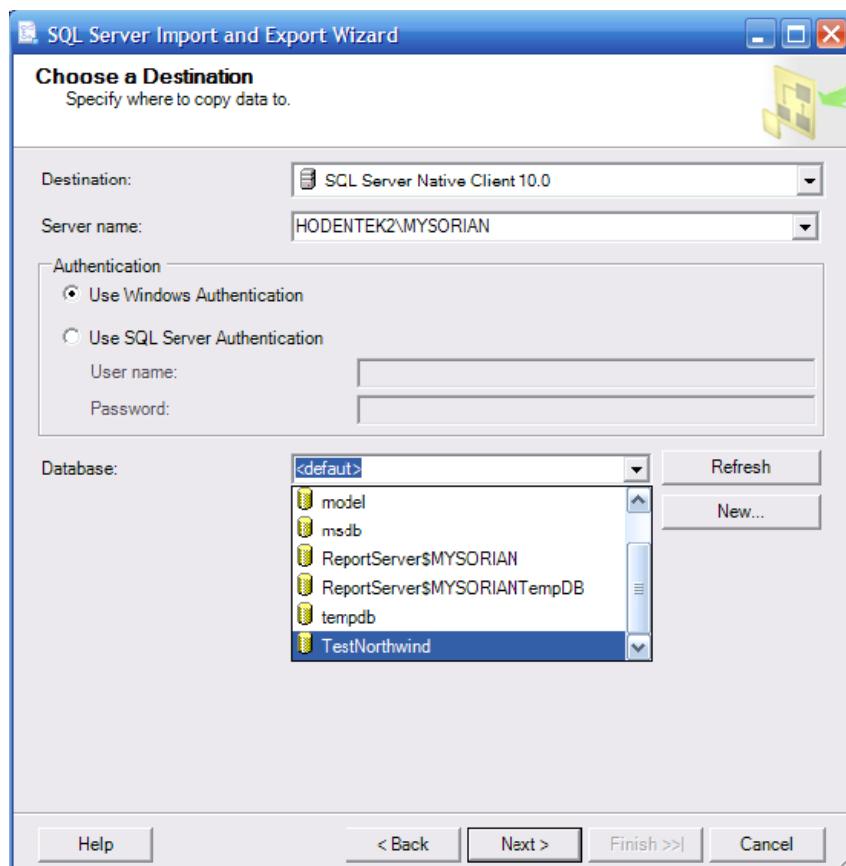




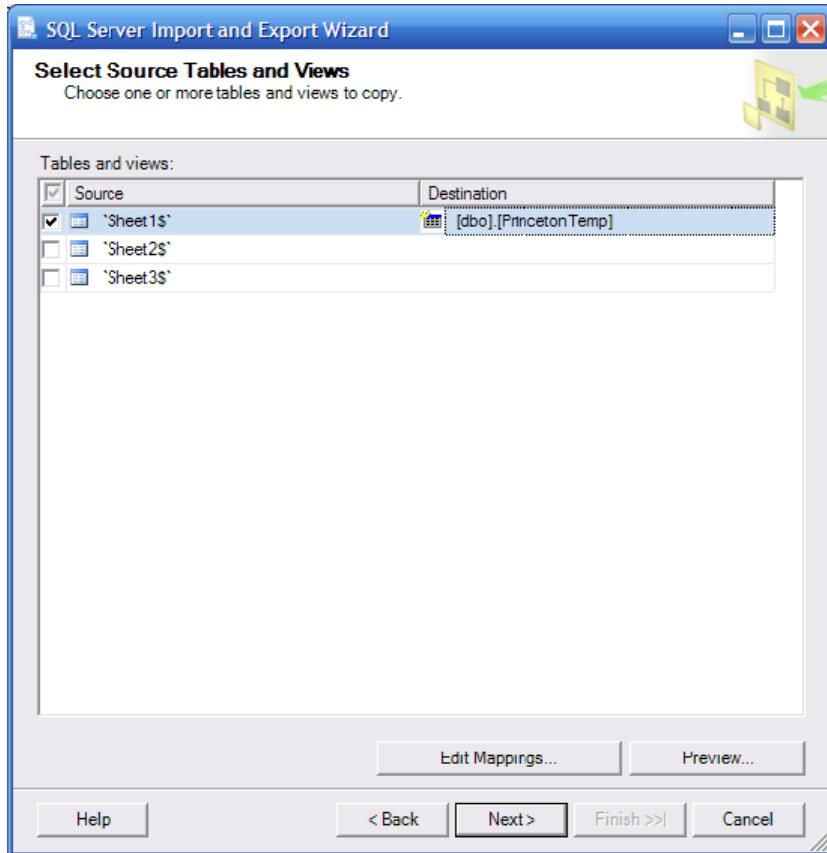
Click Next. The 'Choose the Destination' page shows up with SQL Server Native Client 10.0 as default and the resident server as Hodentek2\Mysorian. The server is configured for Windows authentication. Accept the defaults. In case your server is configured for SQL Server authentication you need to have the information ready. The database is displaying <default>. Click on the handle and choose a database from the drop-down list. Herein TestNorthwind is chosen. You can choose any database including the tempdb. Note that you can begin to create a new database as well, if you choose to do so by using the New...button.

Click Next. The 'specify the Table Copy or Query' page of the wizard shows up. Since we are transferring only one table, accept the default option, 'Copy data from one or more tables or views'.



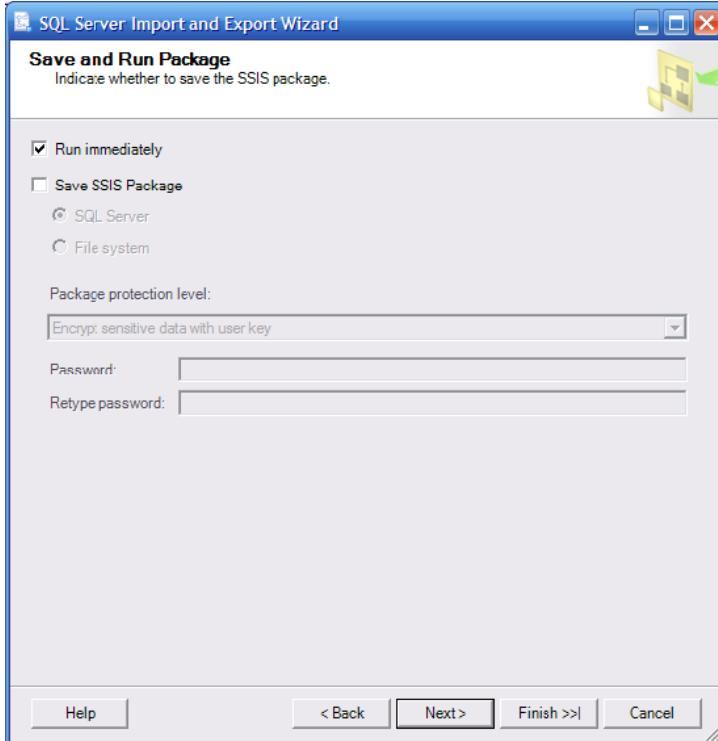


Click Next. Since sheet one has the data place check mark for 'Sheet1\$' as shown. Only Sheet1 has data in this XLS file. Modify the destination column to read dbo.PrincetonTemp instead of the default [dbo] . [Sheet1\$] as shown.

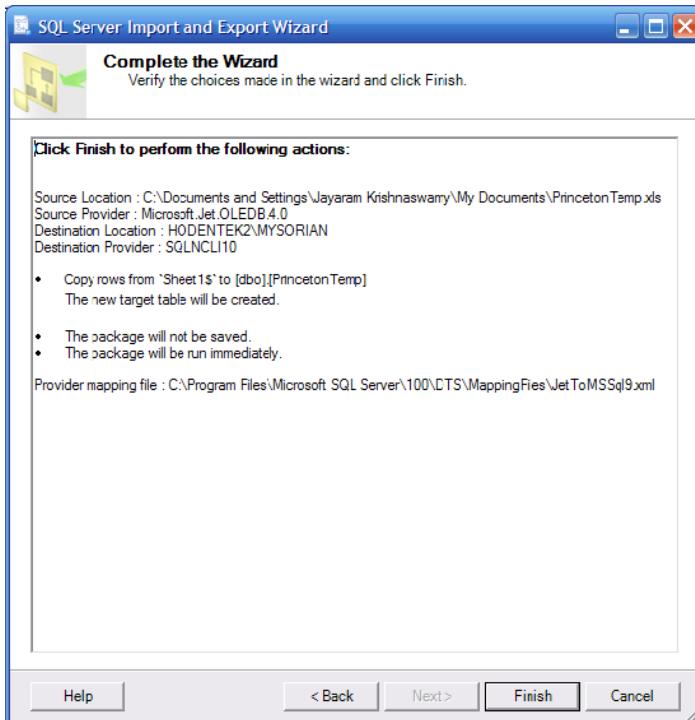


Click Next. In the 'Save and Run Package' page of the wizard accept the defaults shown. You could also save it as a package as well for later use.

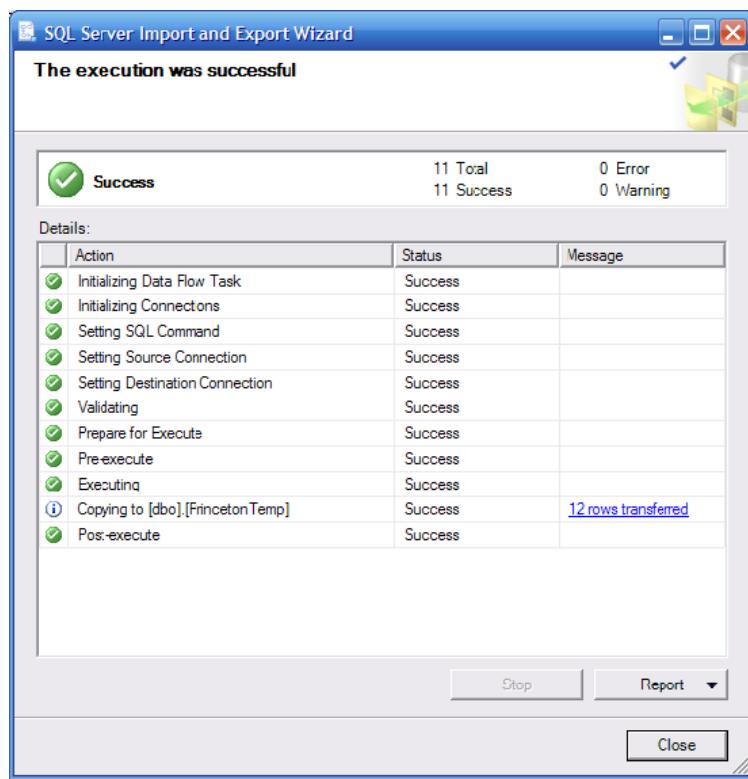
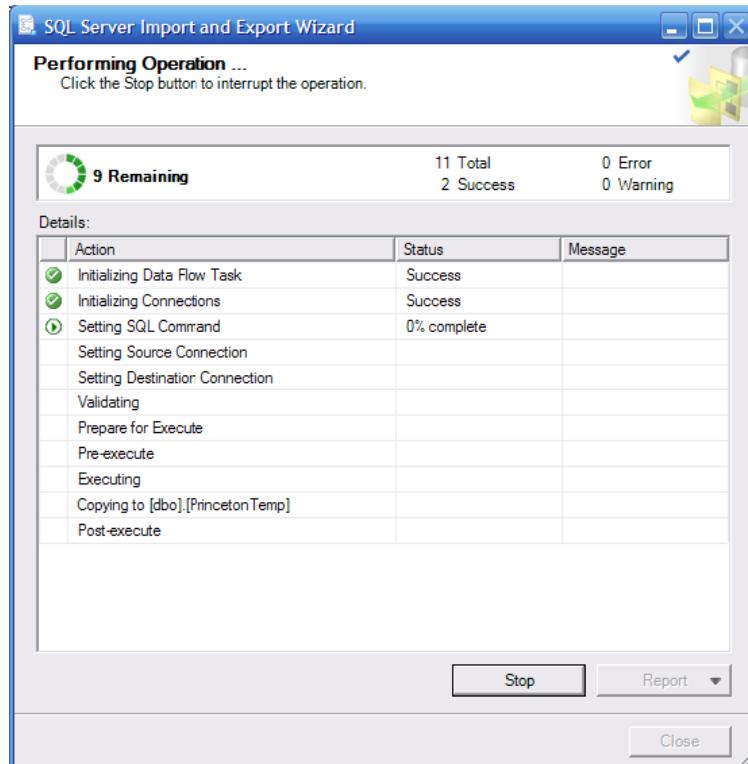




Click Next. The 'Complete the Wizard' page gets displayed. Check if the information is correct (this is a summary of options you have chosen). If it is not correct you can hit the back button and move back to the pages you visited earlier in the reverse order.



Click Finish. The program starts running and you should see a progress window displaying 'Performing Operation...' as shown.

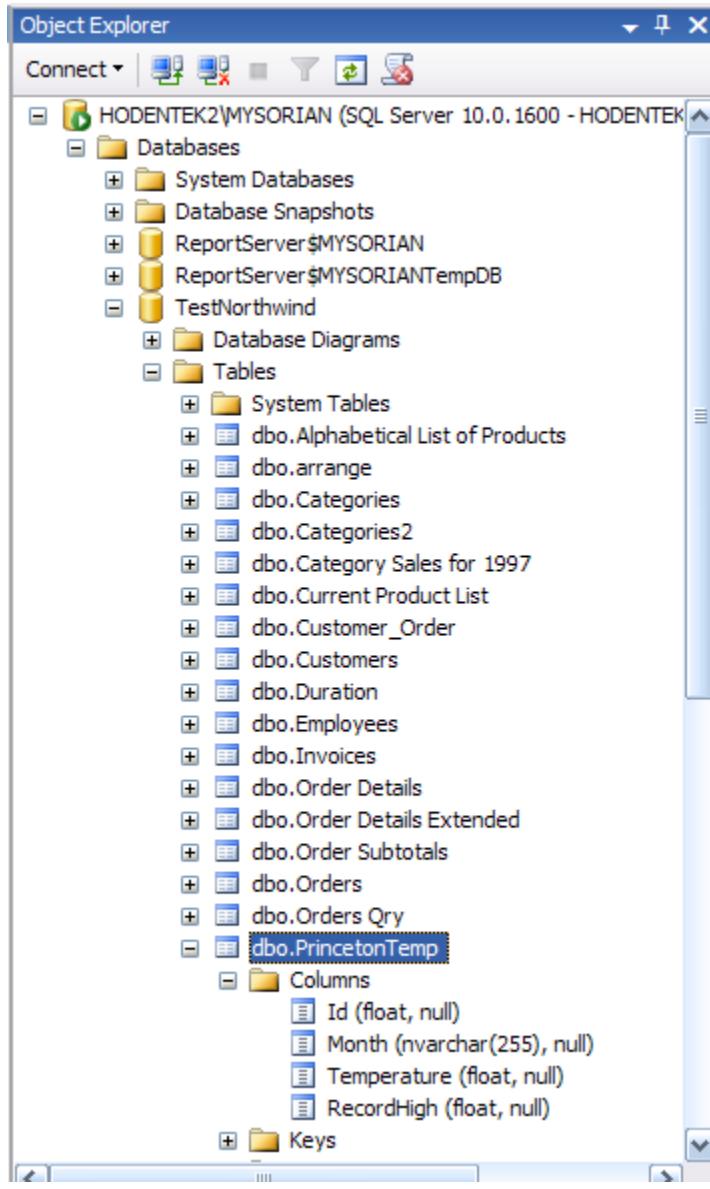


When the operation is completed you should see the following window and you can keep a copy of the report as to how the import was executed using the Report drop-down button.

The import in this case was successful as shown above. If there is an error there should be a hyperlink to the message in the Message column of the above window, presently the message is '12 rows transferred'. Close the wizard. The transfer is finished.

## Verifying the import

Open the Microsoft SQL Server Management Studio and login to display the database engine using your Windows credentials. Expand the databases node and the TestNorthwind database node as shown.



## Data type mismatch and the fix

Also check if the data is brought in correctly as shown by right clicking the dbo.PrincetonTemp table and choose 'Select Top 1000 rows'. You can see that the Month names are all showing 'Null'. The 'text' data type in the XLS file became nvarchar type.

```
SQLQuery1.sql - HODENTEK2...5))
/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP 1000 [Id]
    , [Month]
    , [Temperature]
    , [RecordHigh]
FROM [TestNorthwind].[dbo].[PrincetonTemp]
```

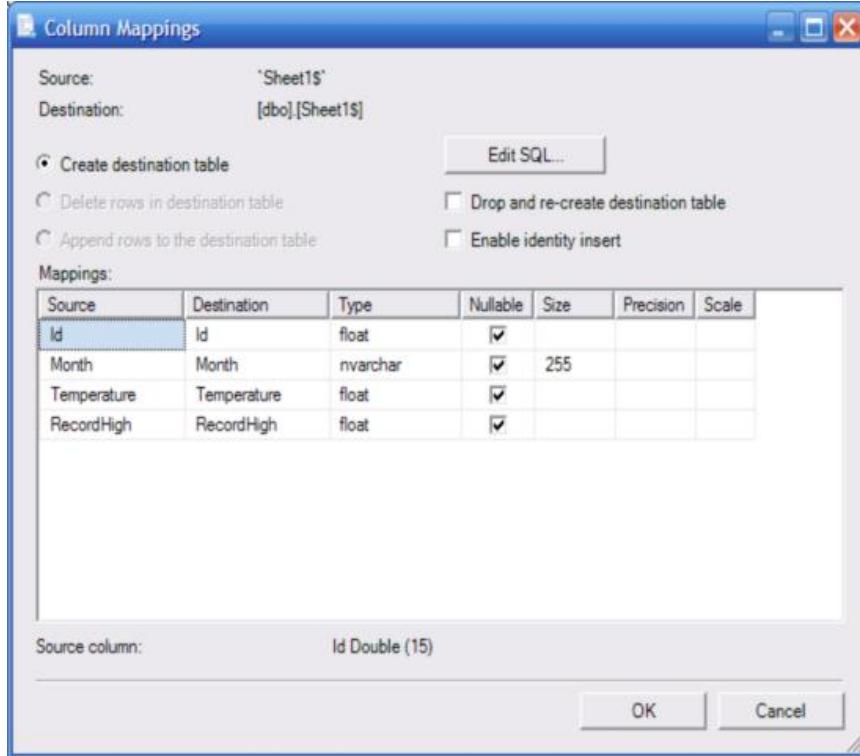
	Id	Month	Temperature	RecordHigh
1	1	NULL	40	60
2	2	NULL	32	50
3	3	NULL	43	65
4	4	NULL	50	70
5	5	NULL	53	74
6	6	NULL	60	78
7	7	NULL	68	70
8	8	NULL	71	70
9	9	NULL	60	82
10	10	NULL	55	67
11	11	NULL	45	55
12	12	NULL	40	62

## Modify the default mappings

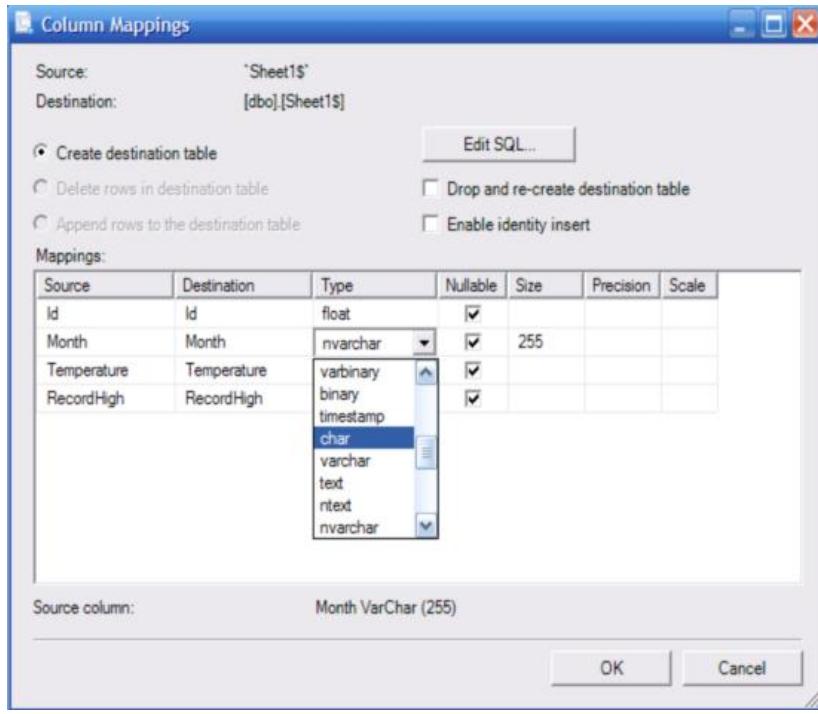
In order to fix this, you can use either Drop table statement or right click and choose delete to delete the table from the TestNorthwind database. In the Delete Object window click OK. Refresh the Tables node by right clicking the Tables and choosing refresh. Now the imported table is gone.

Repeat the process that you did earlier and when you come to the stage shown in Figure.6 click on the table Edit Mappings...button. The Column Mappings page shows up as in the next figure



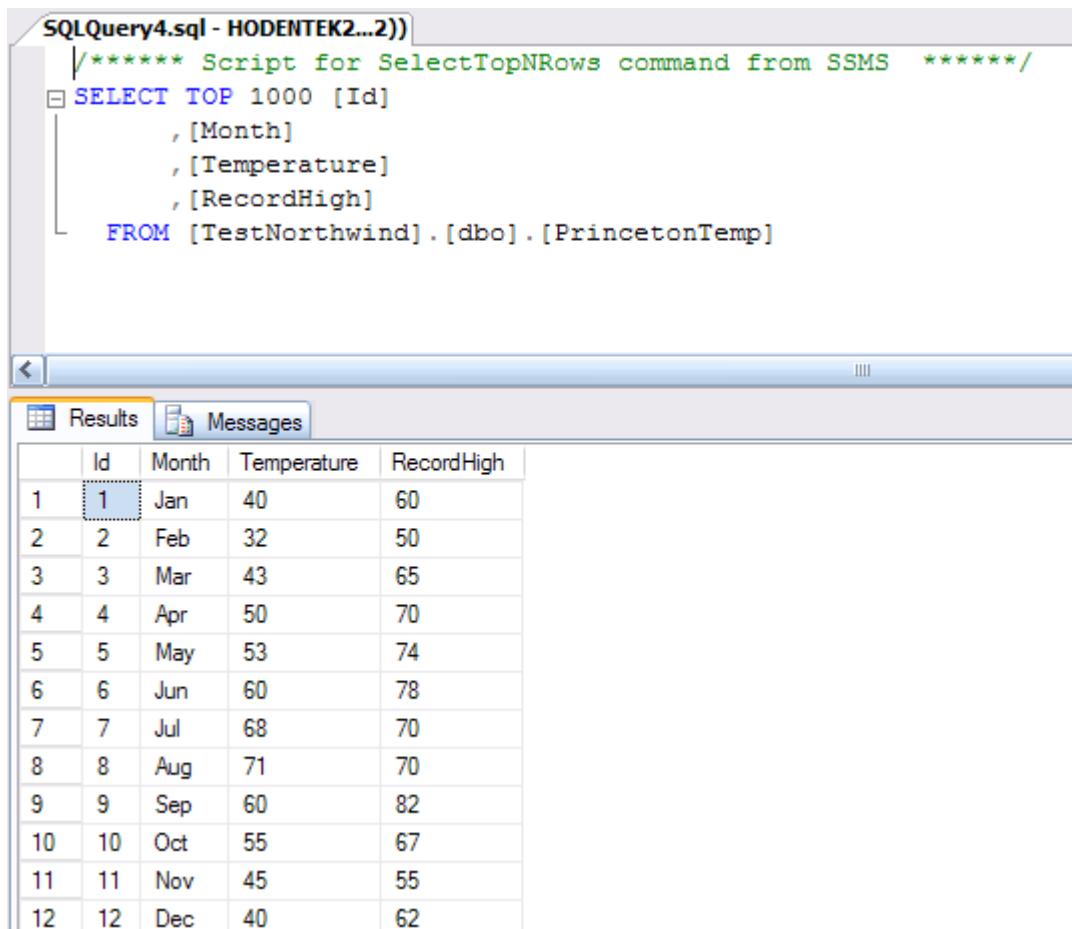


The month column data type for the destination is nvarchar (255). The Source had 'Text' as data type for this column. We need to cast it properly. Click on nvarchar in the 'Type' column and change it to 'char' as shown. Click OK. Change destination table name from [dbo]. [Sheet1\$] to [dbo]. [PrincetonTemp] as done previously. Click Next.



In the 'Save and Run Package' page accept defaults as previously. Click Next. The 'Complete the Wizard' page shows up. Click Finish. You get the wizard announcing 'The execution was successful'. Close the wizard.

Refresh the Tables node of the Northwind database in Management Studio. Now right click the PrincetonTemp and choose to select top 1000 rows as before. You will see that all the data in source is in the destination.



The screenshot shows the SQL Server Management Studio interface. The top window is titled "SQLQuery4.sql - HODENTEK2...2)". It contains a T-SQL script:

```
/* ***** Script for SelectTopNRows command from SSMS *****  
SELECT TOP 1000 [Id]  
    , [Month]  
    , [Temperature]  
    , [RecordHigh]  
FROM [TestNorthwind].[dbo].[PrincetonTemp]
```

Below the script is a results grid. The "Results" tab is selected. The grid has columns: Id, Month, Temperature, and RecordHigh. The data is as follows:

	Id	Month	Temperature	RecordHigh
1	1	Jan	40	60
2	2	Feb	32	50
3	3	Mar	43	65
4	4	Apr	50	70
5	5	May	53	74
6	6	Jun	60	78
7	7	Jul	68	70
8	8	Aug	71	70
9	9	Sep	60	82
10	10	Oct	55	67
11	11	Nov	45	55
12	12	Dec	40	62

### Case Study: Transferring Jobs and Logins using SSIS

We can use SQL Server Integration Services to transfer the logins and jobs from SQL 2005 to another SQL 2005 or SQL 2008. This comes in handy when it's difficult to script each of the jobs. Firstly we need to create an SSIS package.

Open Business intelligence development studio – click file new project – select integration services project as the template and provide a suitable name for it.

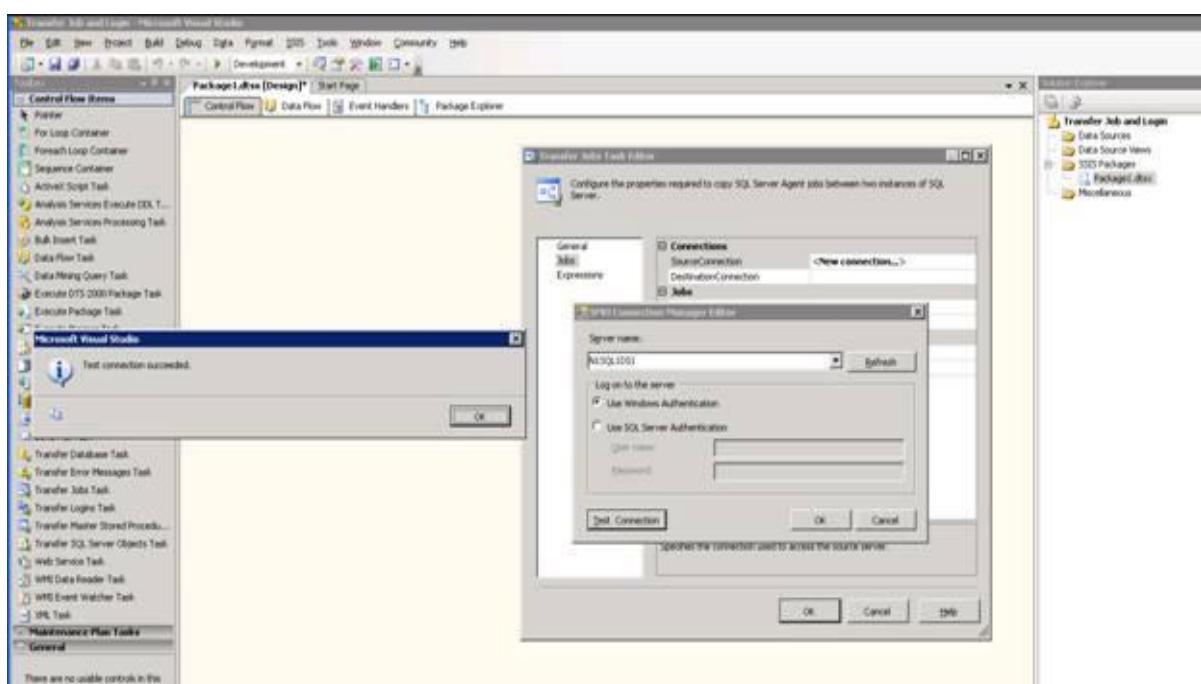
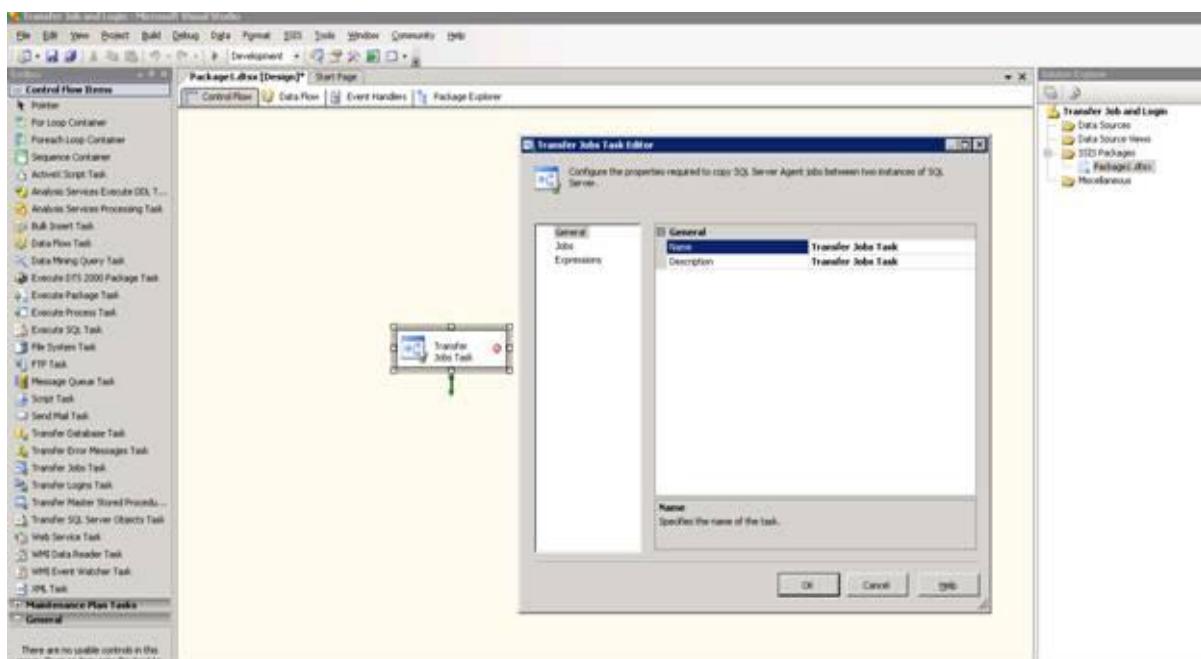


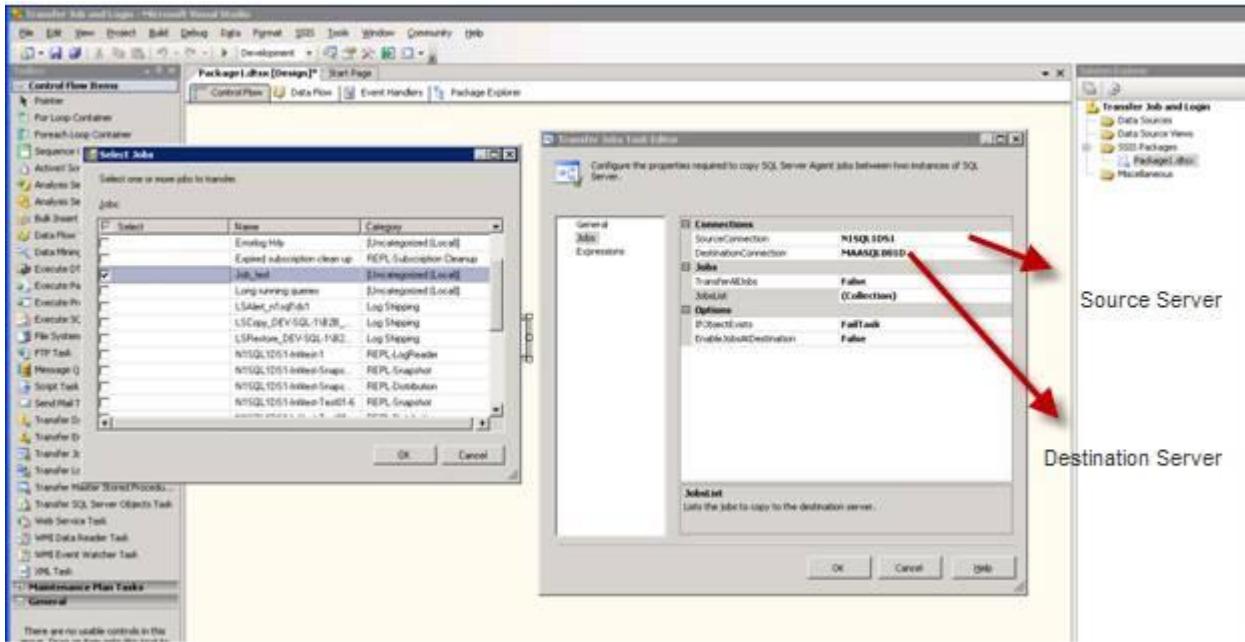


Open solution explorer using Ctrl + Alt + L or go to view and select solution explorer. Expand SSIS under solution explorer and expand SSIS packages – right click and select new SSIS package.

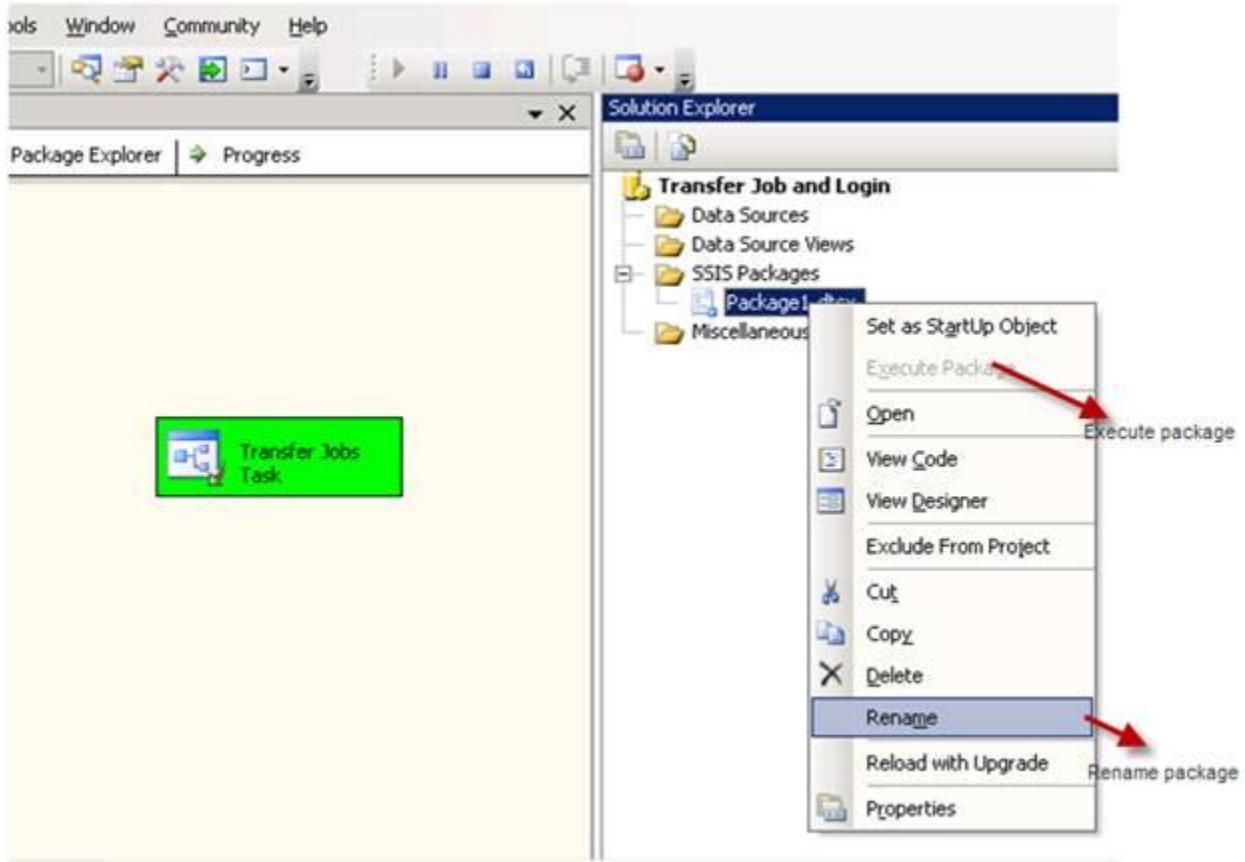
1. Drag and drop Transfer Job task from Control flow item into Control flow tab and double click it
2. In the general tab, give a specific name and description for the transfer job task
3. In Jobs tab, specify the source and destination server name under connections and test the connection as shown below
4. We also have an option to specify whether to transfer all the jobs or specific jobs. In my case I am setting the option as False so as to transfer only specific jobs. Then we need to select the list of jobs that needs to be transferred from the 'JobsList'
5. In the options, if the objects already exist (in our case it is the job) we need to specify what needs to be done. We can either specify it as FailTask if the job exists in destination or skip the object or overwrite it if it's already present
6. Finally we need to specify the option to enable the jobs in destination server after getting transferred



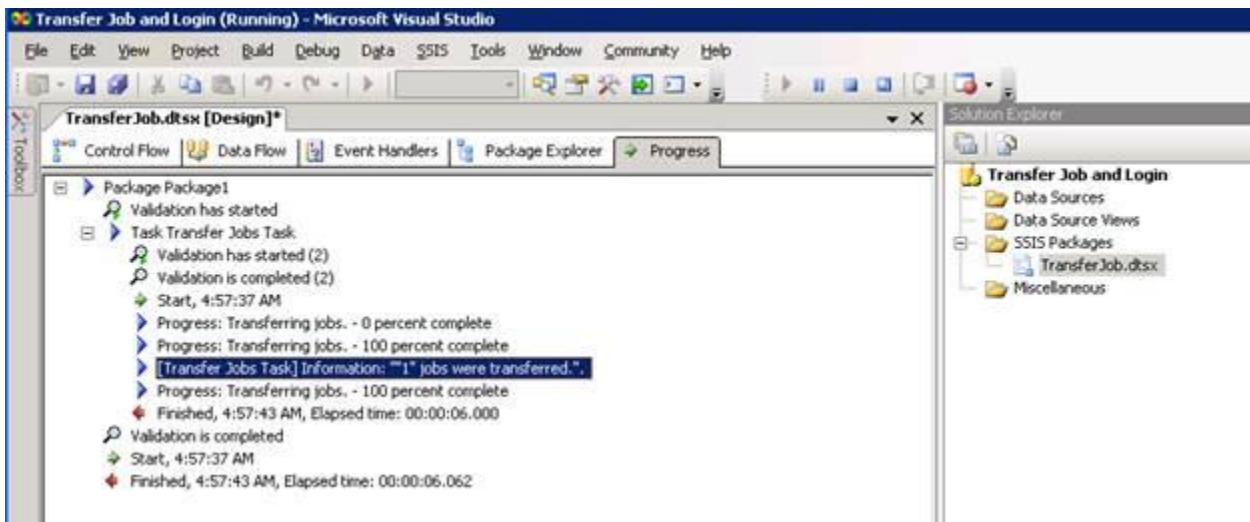




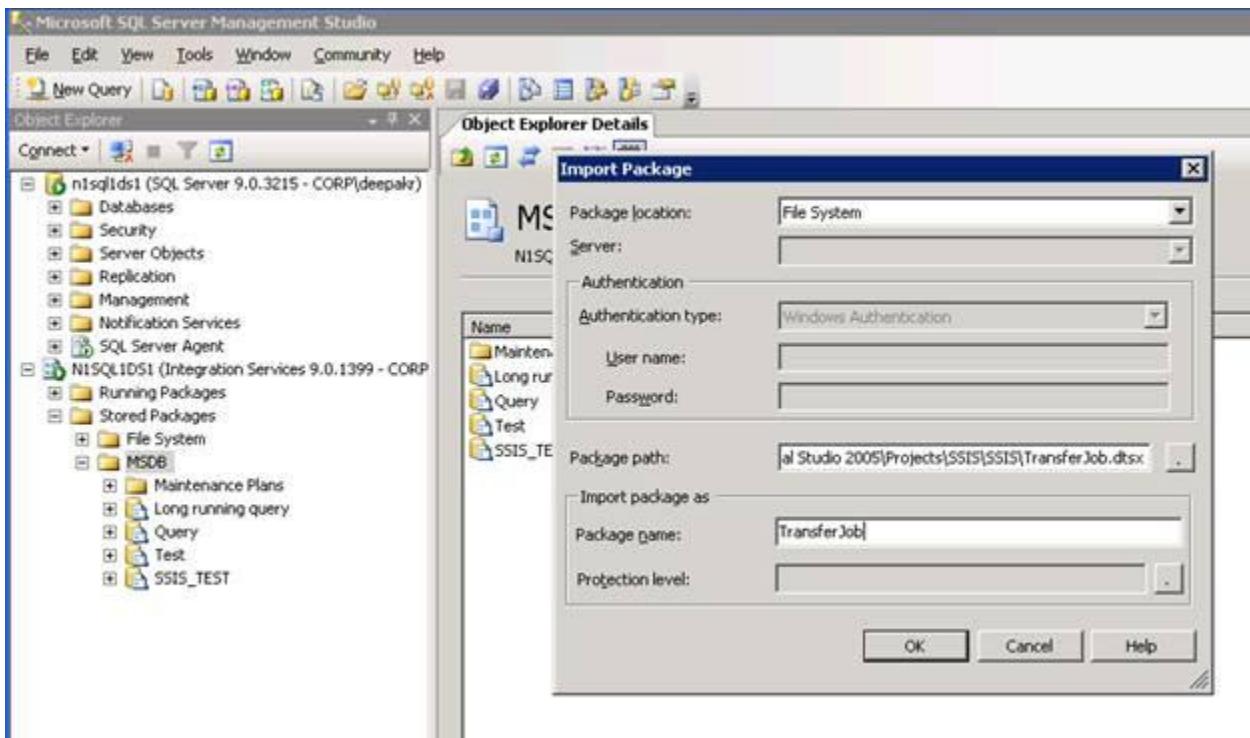
Once the above steps are completed we need to save the package (Ctrl + S) and execute it as shown below. We can view the status of the package execution under 'Progress' tab.



We also have an option to rename the package as indicated by the arrow marks.

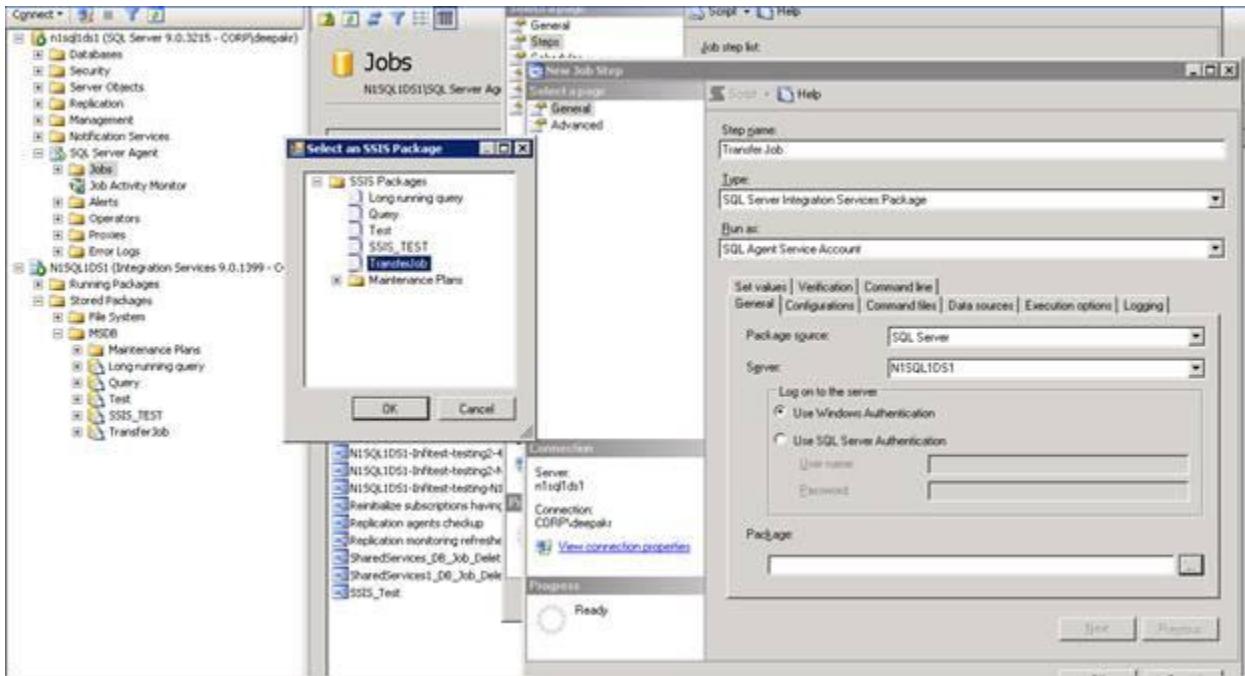


Once the package has executed successfully we can schedule the package as a job. First connect to Integration service – right click MSDB – Import package – select file system – specify the package path – and provide a name for the SSIS package.

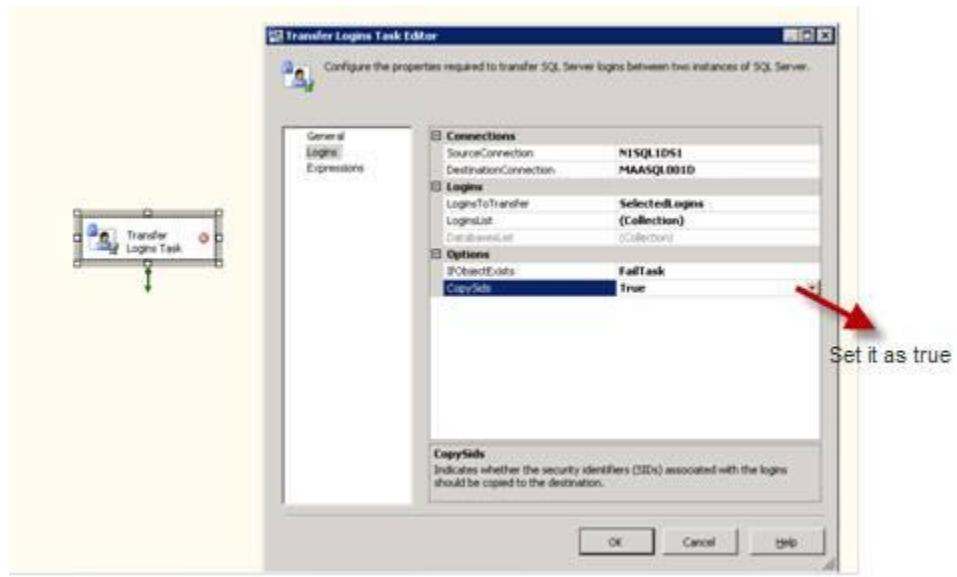


Expand SQL Agent node and right click on Jobs – new job and select the ‘type’ as SQL Server Integration Services Package. Specify server name and package source as SQL Server. Click the ellipse button next to ‘package’ and select the package we created. Specify the schedule for the job as per your desire.





To create a package for Transfer login task, we need to create a new package and drag and drop the transfer login task from control flow items window and specify the source, destination server name. We need to choose the option to transfer all the logins or selected logins for specific databases. Also we have option to overwrite/skip/fail the package similar to the transfer job task. Finally we need to set the option 'CopySids' as true to transfer the security identifiers as well. We need to enable this option in for transferring logins while doing log shipping where there might be mismatched id due to restore operation.



## SQL Server Security

SQL Server 2008 can be configured to work in either the Windows Authentication Mode or the SQL Server and Windows Authentication Mode, which is also frequently called Mixed Mode.

### Windows Authentication Mode

In Windows Authentication Mode only logins for valid Windows users are allowed to connect to SQL Server. In this authentication mode, SQL Server “trusts” the Windows, Windows Domain, or Active Directory security subsystem to have validated the account credentials. No SQL Server accounts are allowed to connect. They can be created, but they cannot be used for login access.

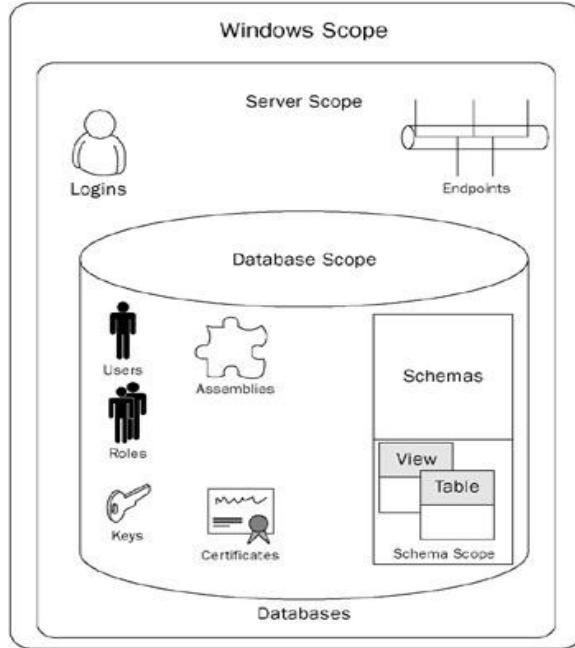
### SQL Server and Windows Authentication Mode (Mixed Mode)

In SQL Server Mode and Windows Authentication Mode or Mixed Mode, valid Windows accounts and standard SQL Server logins are permitted to connect to the server. SQL Server logins are validated by supplying a username and password. Windows accounts are still trusted by SQL Server. The chief advantage of Mixed Mode is the ability of non-Windows accounts (such as UNIX) or Internet clients to connect to SQL Server.

### Security Architecture:

There are different levels in your security hierarchy. The below figure outlines the different levels of security you need to manage. At the Windows scope, you create Windows users and groups, manage the files and services needed by the SQL Server, as well as the behavior of the server itself. In the server scope, you manage logins, endpoints, and databases. In the database scope, you work with users, keys, certificates, roles, assemblies, and other objects. Also in this scope are schemas, which contain your next set of securables. Finally, within the schema scope, you have data types, XML schema collections, and objects. These objects include your tables, views, stored procedures, and more.





## 2008

Microsoft SQL Server 2008 includes a number of server-level roles that are available to simplify management (and the delegation of management) for SQL logins. These are often referred to as fixed 2008 because membership is the only thing you can really change about these roles. The fixed 2008 are designed to allow you to automatically assign a common set of permissions to a login, based upon the purpose of the role.

Role	Description
sysadmin	Members have full administrative access to the SQL Server, and can perform any action. By default, this includes the BUILTIN\Administrators group.
serveradmin	Members can change server-wide configurations and shut down the server.
securityadmin	Members can manage SQL logins, including changing and resetting passwords as needed, as well as managing GRANT, REVOKE, and DENY permissions at the server and database levels.
dbcreator	Members can create, drop, alter, and restore any database for the server.
diskadmin	Members can manage disk files for the server and all databases.
processadmin	Members can manage and terminate processes on the SQL Server.
setupadmin	Members can add and remove linked servers.
bulkadmin	Members of this role can execute the BULK INSERT statement for any database on the server.



## Creating Logins in Management Studio

To create logins from Management Studio, follow these steps:

1. From the Object Explorer, expand your server.
2. Expand the Security folder.
3. Right-click Logins and select New Login.
4. In the New Login dialog box (see Figure ), either type the Login name you want to add, or click the Search button to browse for a Windows account.

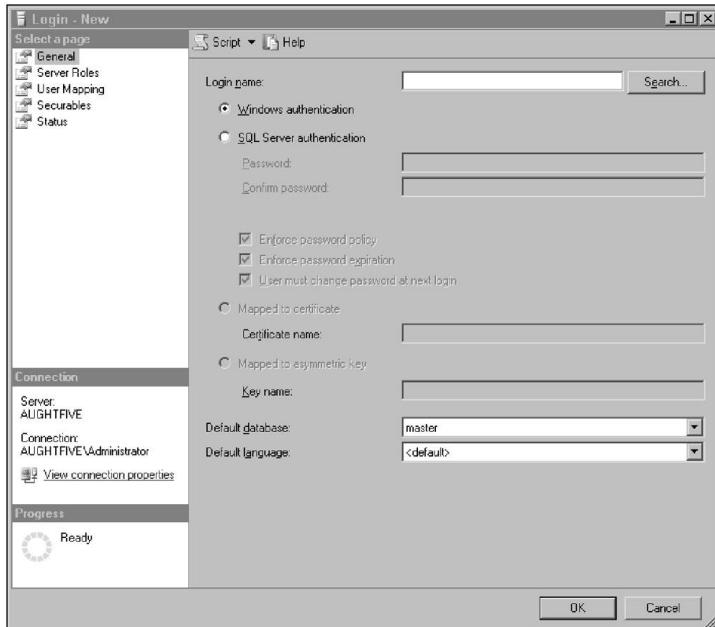


Figure: New Login dialog box

5. If you are creating a SQL Login, select the “SQL Server authentication” radio button.
6. Also, when you select “SQL Server authentication,” you can choose to not enforce the password policies.
7. You may also want to change the user’s default database and language.

## Credentials

Microsoft SQL Server 2005/2008 includes a new feature for mapping SQL Server logins to external Windows accounts. This can be extremely useful if you need to allow SQL Server logins to interact with the resources outside the scope of the SQL Server itself (such as a linked server or a local file system). They can also be used with assemblies that are configured for EXTERNAL\_ACCESS.

Credentials can be configured as a one-to-one mapping, or a many-to-one mapping, allowing multiple SQL Server logins to use one shared Windows account for external access. Logins, however, can only be associated with one credential at a time.



## Creating a New Credential

To create a new credential, follow these steps:

1. In Object Explorer, expand your server.
2. Expand the Security folder.
3. Right-click Credentials and select New Credential.
4. Type a name for the credential (see [Figure](#)).

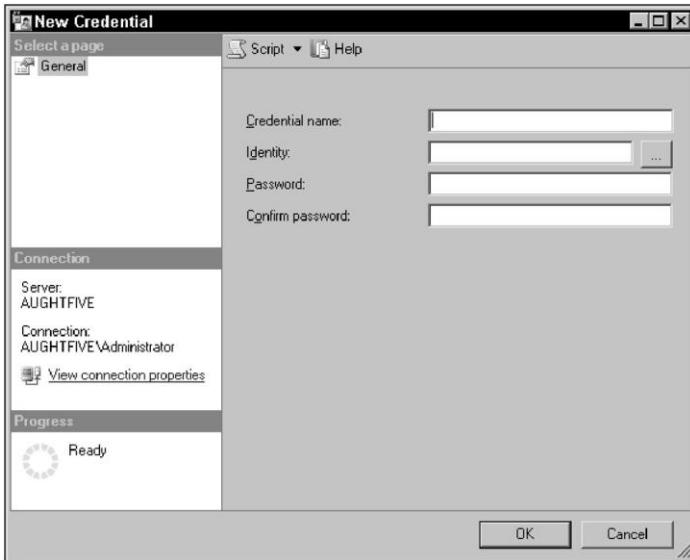


Figure: New Credential properties screen

5. Either type the name of a Windows account, or click the "..." button to browse for an account.
6. Enter the password for the account.
7. Re-enter the password to confirm.
8. Click OK.

View	Description
sys.server_principals	Returns information about all server-level principals.
sys.sql_logins	Returns information about SQL Server logins.
sys.server_role_members	Returns the role ID and member ID for each member of a server role.

## Fixed Database Roles

Every SQL database has a list of fixed database roles that allow you to delegate permissions to users as necessary. As with the fixed 2008, membership is the only thing you can change about these roles. It is important to know how and when to use these roles.

The following table shows the fixed database roles.



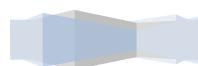
<b>Role</b>	<b>Description</b>
db_accessadmin	This role can add or remove access for Windows logins, Windows groups, and SQL Server logins.
db_backupoperator	This role has the right to back up the database.
db_datareader	Members of this role can read data from all user tables.
db_datawriter	Members of this role can write data from all user tables.
db_ddladmin	This role can execute data definition language (DDL) statements for any object in the database.
db_denydatareader	This role is explicitly excluded from being able to read from any user table with the database.
db_denydatawriter	This role is explicitly excluded from being able to write to any table in the database.
db_owner	Members of this role can perform any activity within the database. New to SQL Server 2008 is the ability for this role to drop the database from the server. The dbo user is automatically a member of this role.
db_securityadmin	This role can manage permissions and role membership within the database.
public	Membership in the public role is automatic. Permissions that apply to the public role apply to everyone who accesses the database.

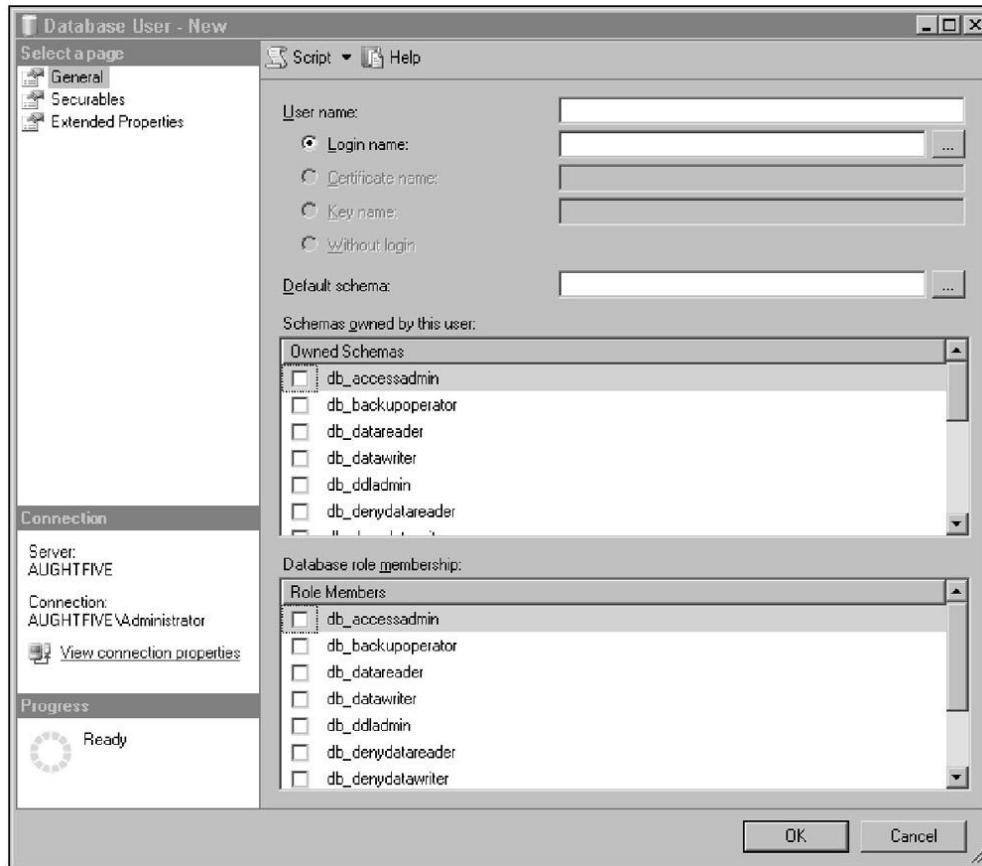
Note that the fixed database roles include db\_denydatareader and db\_denydatawriter. These roles explicitly deny read or write access to user tables in the database, and should be used sparingly. Deny permissions are authoritative and cannot be overridden.

## Database Users

Database users are another component of the security model employed by Microsoft SQL Server 2008. Users are granted access to database securables, either directly or through membership in one or more database roles. Users are also associated with ownership of objects such as tables, views, and stored procedures.

When a login is created, unless it is a member of a fixed server role with database administrative privileges, that login has no explicit permissions within the various databases attached to the server. When this happens, the login is associated with the guest database user, and inherits the permissions of that user account.





## Permissions

Permissions are at the heart of security in SQL Server 2008. In the previous section, you looked at the different types of objects that can be created to help manage security by identifying to whom you can grant access. In this section, you look at permissions that can be applied to the different resources in SQL Server.

To begin with, you should understand there are essentially three permission states that exist: GRANT, GRANT\_W\_GRANT, and DENY. In addition, when a principal does not have an explicit permission defined, the permission is considered "revoked." The following table shows the different permission states.

Permission	Description
GRANT	This state means that you have been given the right to perform this action, or interact with this resource based on what the actual permission is.
GRANT_W_GRANT	Not only can you perform this action, but you also have the right to give others the ability to perform this action.
DENY	You cannot perform this action. This is also known as an "explicit deny," because nothing will allow you to perform this action.
REVOKE	This is not really a permission state as much as it is the absence of a permission state. Revoked permissions will not show up in a

Permission	Description
	sysprotects table or sys.sysprotects view, and are considered an "implicit deny." The idea is that if you haven't been granted this permission, either directly or through membership in a role with that permission, it is safe to assume you shouldn't be doing that. Therefore, you will not be doing that.

To control permission states, you can use the Object Explorer or Transact-SQL. The three commands that you can use to control permission states are GRANT, REVOKE, and DENY, which are described in the following table.

Command	Description
GRANT	This command allows you to grant the right to perform an action or interact with an object in a specific way. The GRANT statement includes the WITH GRANT OPTION option, which also allows the grantee the ability to become a grantor of this permission.
REVOKE	This command removes any explicit permission granted to the grantee, either grant or deny. Revoked permissions will remove the ability to perform that task. Remember that if the user is a member of another role, they may still have the ability to perform the action, unless an explicit deny is specified.
DENY	This command creates an entry that will prevent the user from performing the action. Denied permissions cannot be overridden by grant permissions.

## Best Practices on Security

1. Ensure the physical security of each SQL Server, preventing any unauthorized users to physically accessing your servers.
2. Only install required network libraries and network protocols on your SQL Server instances.
3. Minimize the number of sysadmins allowed to access SQL Server.
4. As a DBA, log on with sysadmin privileges only when needed. Create separate accounts for DBAs to access SQL Server when sysadmin privileges are not needed.
5. Assign the SA account a very obscure password, and never use it to log onto SQL Server. Use a Windows Authentication account to access SQL Server as a sysadmin instead.
6. Give users the least amount of permissions they need to perform their job.
7. Use stored procedures or views to allow users to access data instead of letting them directly access tables.
8. When possible, use Windows Authentication logins instead of SQL Server logins.
9. Use strong passwords for all SQL Server login accounts.
10. Don't grant permissions to the public database role.
11. Remove user login IDs who no longer need access to SQL Server.
12. Remove the guest user account from each user database.
13. Disable cross database ownership chaining if not required.
14. Never grant permission to the xp\_cmdshell to non-sysadmins.



15. Remove sample databases from all production SQL Server instances.
16. Use Windows Global Groups, or SQL Server Roles to manage groups of users that need similar permissions.
17. Avoid creating network shares on any SQL Server.
18. Turn on login auditing so you can see who has succeeded, and failed, to login.
19. Don't use the SA account, or login IDs who are members of the Sysadmin group, as accounts used to access SQL Server from applications.
20. Ensure that your SQL Servers are behind a firewall and are not exposed directly to the Internet.
21. Remove the BUILTIN/Administrators group to prevent local server administrators from being able to access SQL Server.
22. Run each separate SQL Server service under a different Windows domain account.
23. Only give SQL Server service accounts the minimum rights and permissions needed to run the service. In most cases, local administrator rights are not required, and domain administrator rights are never needed. SQL Server setup will automatically configure service accounts with the necessary permissions for them to run correctly, you don't have to do anything.
24. When using distributed queries, use linked servers instead of remote servers.
25. Do not browse the web from a SQL Server.
26. Instead of installing virus protection on a SQL Server, perform virus scans from a remote server during a part of the day when user activity is less.
27. Add operating system and SQL Server service packs and hot fixes soon after they are released and tested, as they often include security enhancements.
28. Encrypt all SQL Server backups with a third-party backup tool, such as SQL Backup Pro.
29. Only enable C2 auditing or Common Criteria compliance if required.
30. Consider running a SQL Server security scanner against your SQL servers to identify security holes.
31. Consider adding a certificate to your SQL Server instances and enable SSL or IPSEC for connections to clients.
32. If using SQL Server 2005, enable password policy checking.
33. If using SQL Server 2005, implement database encryption to protect confidential data.
34. If using SQL Server 2005, don't use the SQL Server Surface Area Configuration tool to unlock features you don't absolutely need.
35. If using SQL Server 2005 and you create endpoints, only grant CONNECT permissions to the logins that need access to them. Explicitly deny CONNECT permissions to endpoints that are not needed by users.

**Problem/Case Study:**

1. How to grant column level permissions on another schema?



## 2. How to Grant limited permissions to create views in another schema?

### Solution

In SQL Server 2005 and 2008 you can grant permissions at the schema level and, in fact, this is what you'll need to do to give them the ability to create the views.

This script below creates an example database along with a role to which we'll assign the permissions to. Note that while I'm using the dbo schema, that's only because there's no logical schema name to use since this isn't a real world example. Typically you would name your schema to group objects and the schema name should reflect what the grouping is. For instance, Person or Product. As can be seen from the example, the LimitedCreatorRights role has the ability to create views in the database and select on tables and views that are located in the dbo schema.

```

CREATE DATABASE yourdb;
GO

USE yourdb;
GO

CREATE ROLE LimitedCreatorRights;
GO

GRANT CREATE VIEW TO LimitedCreatorRights;
GO

GRANT SELECT ON SCHEMA::dbo TO LimitedCreatorRights;
GO

CREATE USER TestUser WITHOUT LOGIN;
GO

EXEC sp_addrolemember 'LimitedCreatorRights', 'TestUser';
GO

CREATE TABLE dbo.ATest (TestId INT);
GO

```

One thing we've not given is the permission to create tables. In the following examples you will see that I am using the `EXECUTE AS` and the `REVERT` commands. The `EXECUTE AS` allows you to still be logged in with sysadmin rights, but run these examples using the TestUser permissions and the `REVERT` returns permissions back to the original user.

So if a user that is a member of this role attempts to create a table in the dbo schema, it'll fail:

```

USE yourdb;
GO

-- This will fail, as TestUser doesn't have CREATE TABLE permissions
EXECUTE AS USER = 'TestUser';
GO

CREATE TABLE dbo.ASecondTable (TestId INT);
GO

```



```
REVERT;
GO
```

Error Msg 262, level 14, state 1, line 2

Create table permission denied in database 'yourdb'

And, in fact, so will the creation of a view:

```
-- This will fail, as TestUser does have CREATE VIEW rights
-- but does not have permission to alter the dbo schema
EXECUTE AS USER = 'TestUser';
GO

CREATE VIEW dbo.AView AS SELECT TestID FROM dbo.ATest;
GO

REVERT;
GO
```

Error Msg 2760, level 16, state 1, procedure Aview, line2

The Specified schema name dbo either does not exist or you don't have permissions

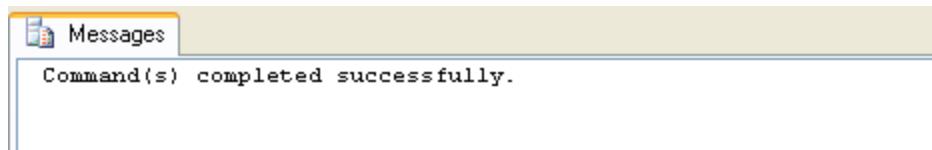
The catch is that the TestUser must have the permission to modify the dbo schema. We can accomplish this by assigning that permission to a role the TestUser is a member of:

```
-- Once permission is granted, re-run the previous CREATE VIEW
-- statement. It will now succeed.
GRANT ALTER ON SCHEMA::dbo TO LimitedCreatorRights;
GO
```

Now, if you go back and re-run the CREATE TABLE and the CREATE VIEW statements above, you'll see the CREATE TABLE statement fails (we didn't give TestUser or any role it is a member of the permission to create a table), but the create view statement will succeed.

Create Table Fails: Error msg create table permission denied in database 'yourdb'

### Create View is Now Successful



### **Case study 3:** How to Recover "SA" password when you forget it. - SQL 2005/2008

What to do if you forgot SA password in SQL Server 2005/2008?

In general if you forgot SA password or if the SA account is disabled these are the options to login into SQL Server 2005 and reset or enable SA.

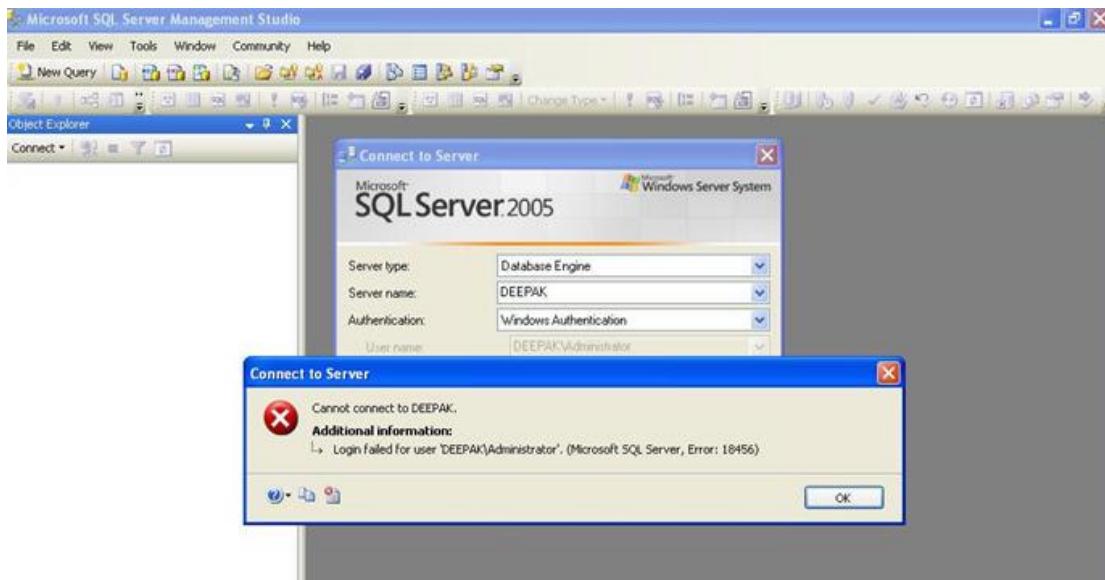
1. If BuiltIn\Administrator is present in SQL Server, you can login with an ID which is member of Administrators group and reset the SA password.
2. Or else if you have some other ID which is having sysadmin privilege in SQL level(this also SQLservice account), you can login with that and reset SA.

But in case if you have the following scenario where,

1. You have disabled SA (or) forgotten SA password
2. You followed the best security practice and removed BuiltIn\administrator from SQL Server

So you cannot login with a sysadmin ID into SQL Server, and you start thinking about uninstalling SQL Server 2005. No need to perform any uninstall and reinstall in such scenarios in SQL Server 2005 as you have this option where the Members of Windows Administrative groups have sysadmin privilege in SQL Server if you start SQL Server 2005 in Single user mode.

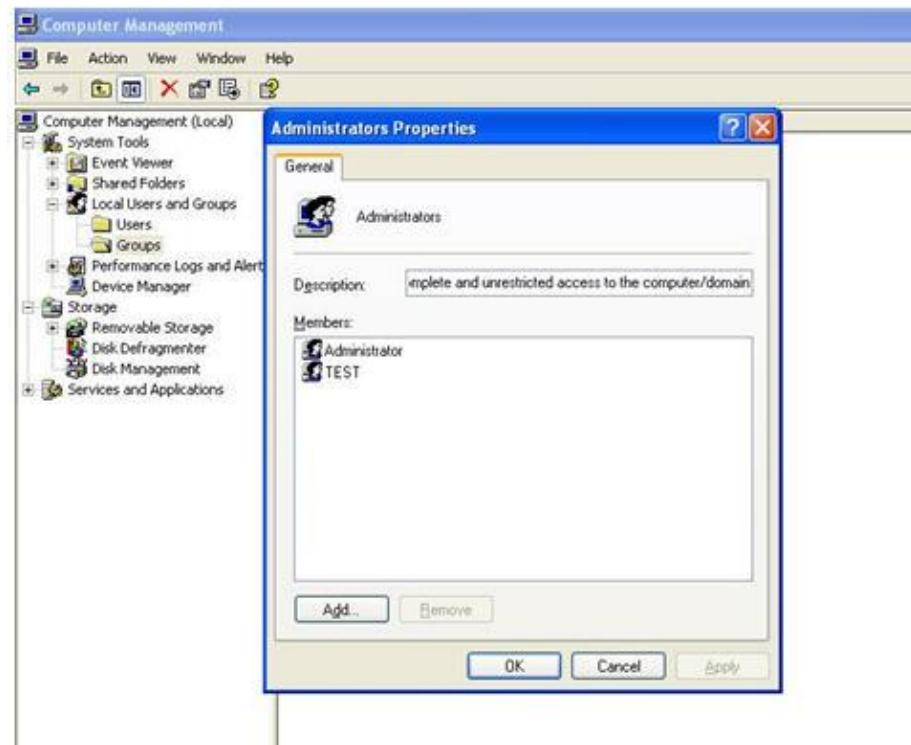
Consider this example, where I have disabled SA and removed BuiltIn\Administrator from SQL Server 2005. Refer the below screenshots which displays the error message I get when I login Administrator and SA,



## SQL Server 2008 DBA



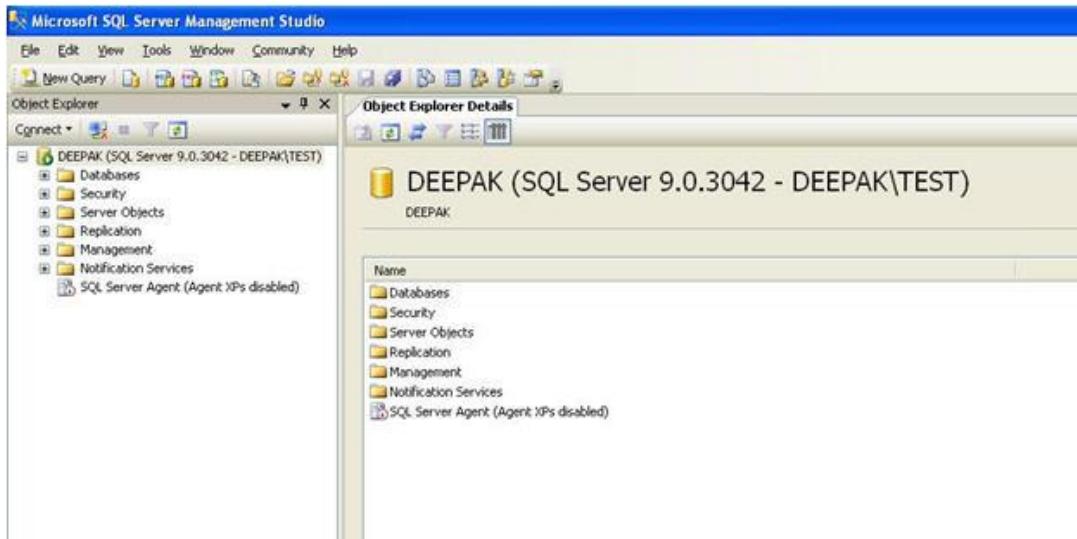
Refer the below screenshot which shows the members of windows administrator group,



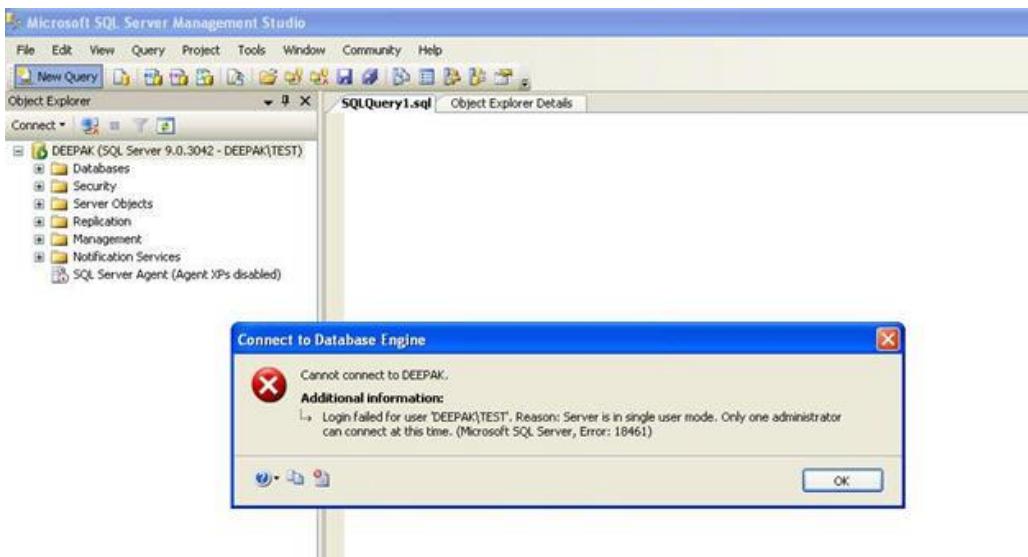
I perform the following steps,

1. Create account "test" at OS level with admin privileges. Login with the ID Test @OS Level
2. Stop SQL Server 2005 using the command, **NET STOP MSSQLSERVER**
3. Start SQL Server 2005 in Single-User mode using the command, **NET START MSSQLSERVER /m**
4. Login into SQL Server 2005 using the ID Test as shown in the below screenshot,





5. Since SQL Server is started in Single-User mode it will allow only one connection and hence you will get the following error if you click the "New Query"

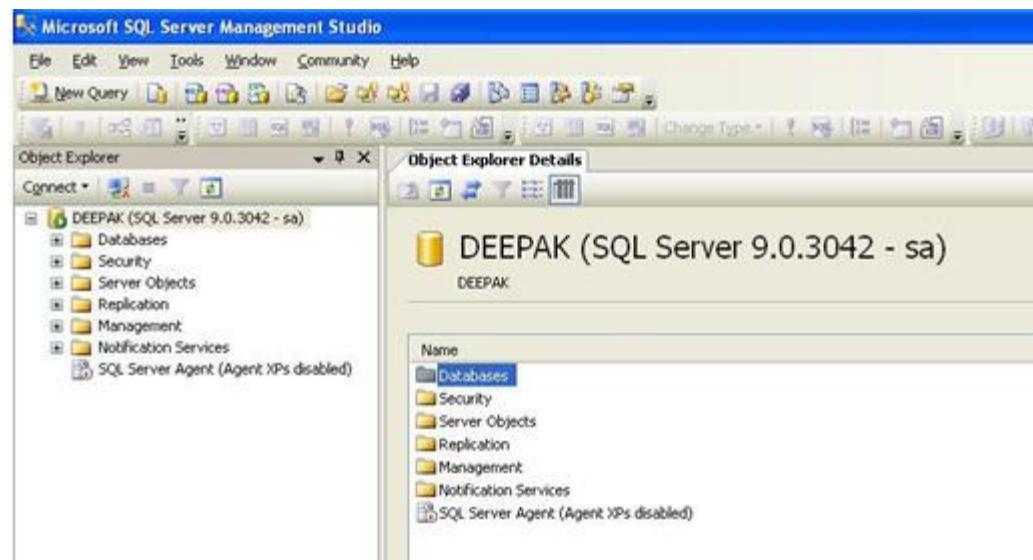


6. Disconnect and close the Object Explorer and then connect using "New Query" you will be able to connect as shown below, and then enable SA login using the command ALTER Login SA enable



The screenshot shows the Microsoft SQL Server Management Studio interface. In the top menu bar, the 'File' and 'Edit' options are visible. Below the menu bar is a toolbar with various icons. A tab bar at the top indicates the current database is 'master' and the query is 'DEEPAK\master - SQLQuery3.sql'. The main pane contains the SQL command: 'ALTER LOGIN SA ENABLE;'. Below the command, the 'Messages' pane displays the message: 'Command(s) completed successfully.'

8. Now you need to Stop SQL Server and start it normally using the command, **NET START MSSQLSERVER** and connect using SA or the new login you created and proceed as shown below,



## Automating Administrative Tasks

### SQL Server Agent

This section examines how to automate tasks on the SQL Server using the Microsoft SQL Server Agent Service. The SQL Agent service runs as Windows service that is dependent on the SQL Service. Each instance of SQL will have its own Agent service to manage jobs, schedules, operators, and alerts. You learn about the essential components of the Agent service for single and multiple server management configurations.

The primary purpose of the SQL Server Agent is to make your job easier. In a perfect world, you could configure your servers, let them run, and never worry about losing data or the database going offline. But, as is too often the case, this isn't a perfect world. And because you can't realistically monitor every server every minute of every day, you can use the SQL Server Agent to leverage against what you can't do.

The SQL Server Agent service is not available in SQL Server 2005 Express Edition. You did not configure the Agent to start automatically; you'll need to know how to start it manually. There are actually four different ways you can start and stop the SQL Server Agent service. One way is to use the

NET START command from a command prompt:

```
NET START SQLSERVERAGENT
```

To stop the service, use the NET STOP command:

```
NET STOP SQLSERVERAGENT
```

### Agent Security

When planning to use the SQL Server Agent service, or allowing other users to access it, you need to ensure that appropriate access is granted. By default, only members of the sysadmin fixed server role have complete access to the Agent service. In the msdb database, additional roles are created with varying levels of rights and permissions, but these roles are empty until a user is explicitly added to these roles. In this section, you learn about each of these roles and the permissions assigned to them.

### SQLAgentUserRole

The SQLAgentUserRole is the most limited of the three Agent roles. Users who are members of this role have the ability to create new jobs and schedules, and can manage only those jobs and schedules they create.

### SQLAgentReaderRole

As with SQLAgentUserRole, SQLAgentReaderRole can enable users to create local jobs and schedules, and manage only those that they create. In addition to these



permissions, they can also view the properties of other local jobs, as well as multi-server jobs.

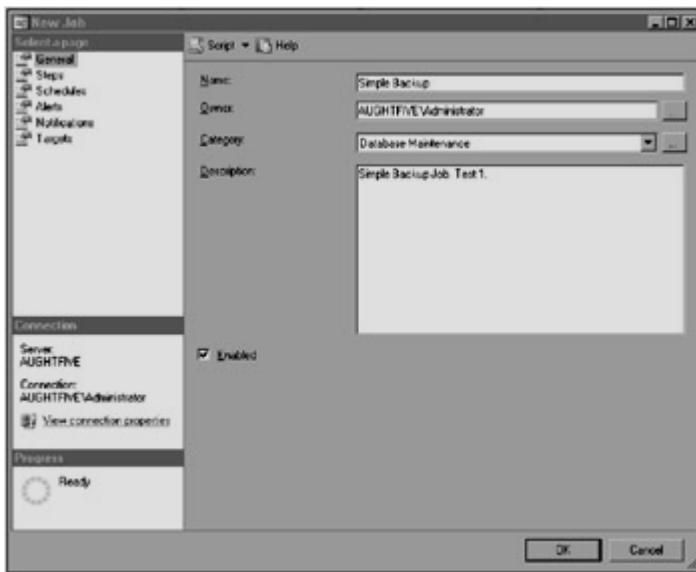
### **SQLAgentOperatorRole**

Members of this role can create local jobs, as well as manage and modify jobs they own. They can also view and delete the job history information for all local jobs. To a limited extent, they can also enable or disable jobs and schedules owned by other users.

### **Creating a New Job**

Begin by creating a new job in SQL Server Management Studio. For this example, you're going to populate only the most basic information about the job from the General properties page.

1. In Object Explorer, expand SQL Server Agent.
2. Right-click Jobs and select New Job.
3. In the New Job dialog box, enter **Simple Backup** as the job name.



3. Leave the Owner as the default.
4. Select Database Maintenance in the Category drop-down list.
5. In the description, enter **Simple Backup Job. Test 1.**
6. Remove the check next to Enabled.
7. Click OK.

This creates a new job, and prevents the job from running once you close the New Job window.

Now you're going to add a Transact-SQL step that will perform a full backup of the AdventureWorks database onto the local disk. Before beginning, you should create a folder called dbBackups on your C: drive.



1. From Object Explorer, expand your server, and then expand SQL Server Agent.
2. Expand Jobs.
3. Right-click Simple Backup and select Properties.
4. Under the Select a Page list, select Steps.
5. Click the New button.
6. In the Step Name box, enter **AdventureWorks Backup**.
7. In the Type drop-down list, ensure that Transact-SQL is listed.
8. Leave "Run as" empty.
9. Ensure that master is the selected database.
10. Enter the following code in the command window:  
11.  
12. `BACKUP DATABASE AdventureWorks TO DISK = 'C:\dbBackups\AWFull.bkf';`  
13. Click OK to close the New Job Step window.  
14. Click OK to close the Job Properties window.  
15. In the SQL Server Management Studio Note, it informs you that the last step will be changed from "Goto Next Step" to "Quit with Success." Click Yes.

You have now created a simple job step. Feel free to enable the job by right-clicking the job and selecting Enable from the context menu. You can also manually run the job at any time, even if it's disabled, by right-clicking and selecting Start Job. The job should execute with success.

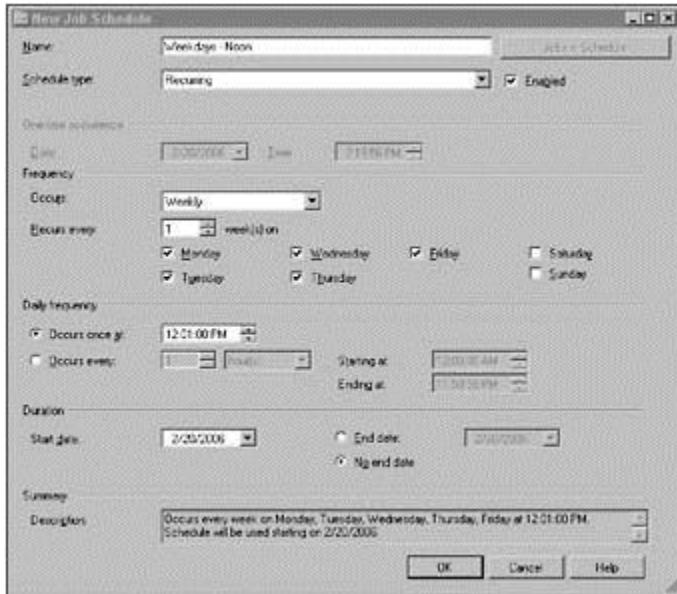
## Creating Schedules

To automate many of the tasks you need to perform to maintain your SQL Server, you must define schedules for when your jobs run. Schedules, not unlike categories, can be created and managed independently of the creation and management of jobs. This allows you to use the same schedule for multiple jobs.

Create a new schedule for your Simple Backup job that will run the job every weekday at noon:

1. From Object Explorer, expand your server, and then expand SQL Server Agent.
2. Right-click Jobs and select Manage Schedules.
3. In the Manage Schedules window, click New...
4. In the New Job Schedule window, enter **Weekdays - Noon** for the schedule name.
5. Ensure that the schedule type is Recurring, and ensure the schedule is Enabled.
6. In the Frequency schedule, make sure that the schedule is set to occur weekly.
7. Select the checkboxes for Monday, Tuesday, Wednesday, Thursday, and Friday.
8. If selected, remove the check in the box next to Sunday.
9. In "Daily frequency," select the radio button marked "Occurs once at:" and set the time to 12:01:00 PM.
10. Leave the "Start date" as the current date, and ensure that "No end date" is selected.
11. Click OK.





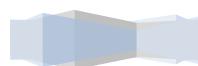
## Creating Operators/Alerts

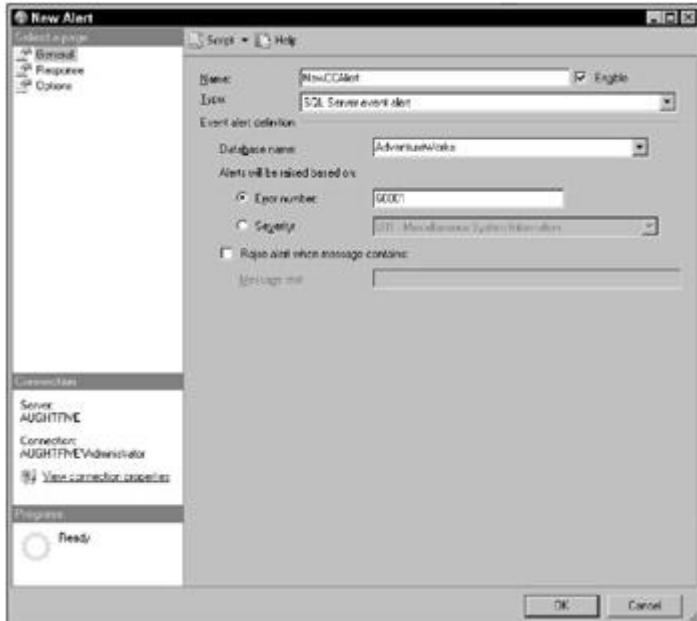
*Operators* are objects that represent a unit of notification for SQL Server Agent jobs and alerts. Operators can represent an individual person, or a group. Operators are not associated with database or server principals, but are exclusive to the SQL Server Agent service.

When you create a new operator, you assign a name to the operator, and then define the methods for notifying the operator. Your options for notifying an operator include email, NET SEND using the Windows Messenger service, and SMTP-enabled pager.

Create a new operator for the administrator account. This operator will be available for paging only on the weekend.

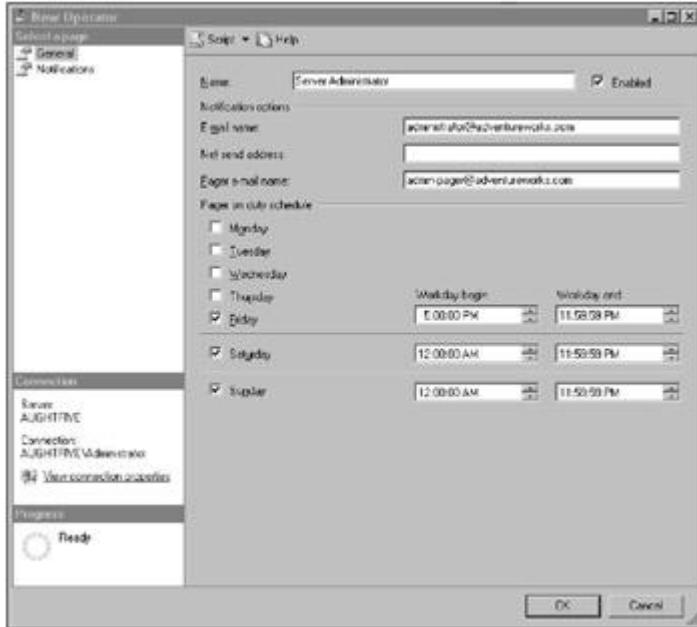
1. From Object Explorer, expand your server, and then expand SQL Server Agent.
2. Right-click the Operators folder and select New Operator.
3. In the New Operator window, enter **Server Administrator** in the Name field.
4. Ensure that the Enabled box is checked.
5. In the "e-mail name" field, enter **administrator@Adventureworks.com**.
6. Leave the "Net send address" field empty.
7. Click OK to close the New Operator properties window.





Creating new alert

If you open the properties of the properties of the operator you just created, you will notice there are two additional pages. The notification page displays a list of jobs and alerts that have sent notifications to this operator. The history page reports the time of the last notification attempt for each notification type.



Creating a new operator



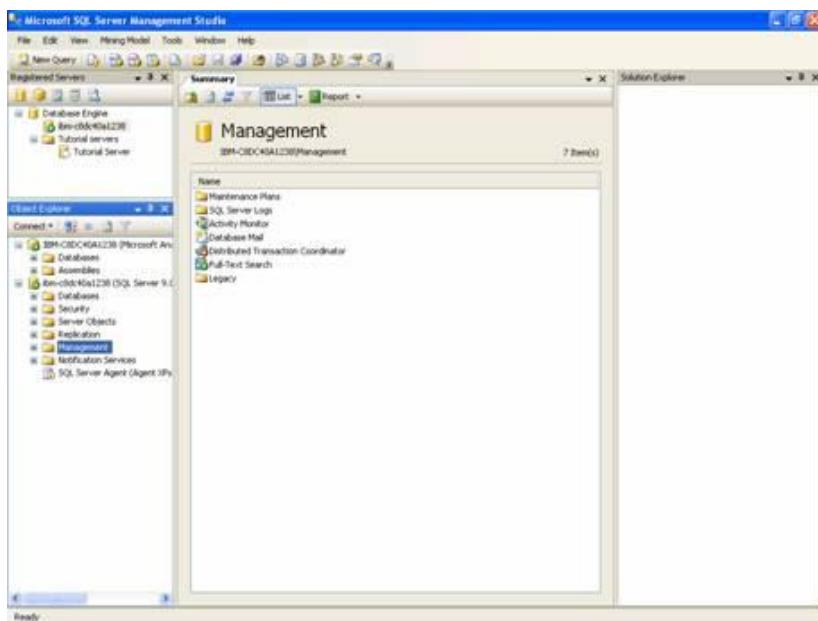
## Maintenance Plans

Maintenance plans can be created using the Maintenance Plan Wizard or using the design surface. The Wizard is useful if the DBA wants to create a basic maintenance plan. If he intends to create enhanced work flow then, it is advisable to use the design surface. Maintenance Plans are displayed only to users connected using the Windows Authentication.

The Maintenance Plan Wizard helps set up the core maintenance tasks for optimum performance of the SQL Server components.

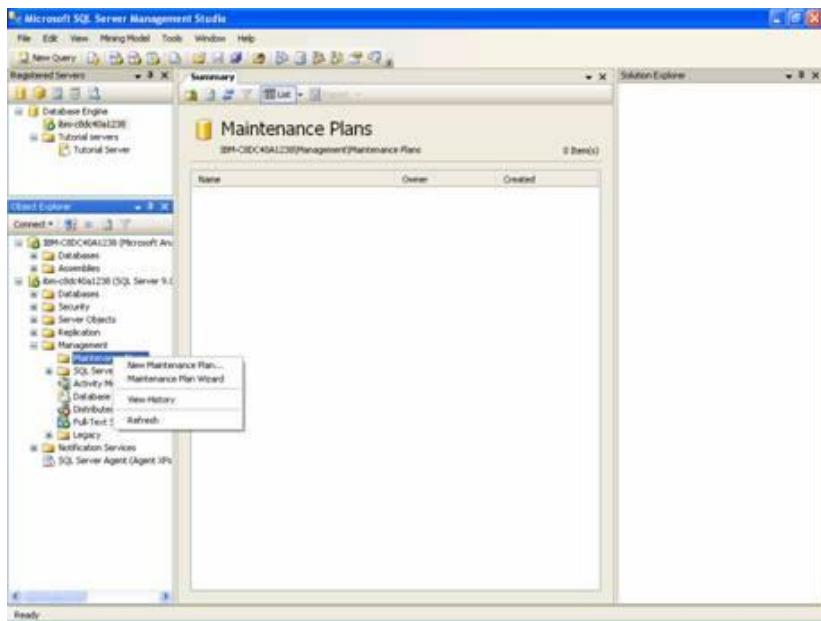
### To start the Maintenance Plan Wizard

1. Expand the server.



2. Expand the **Management** folder.



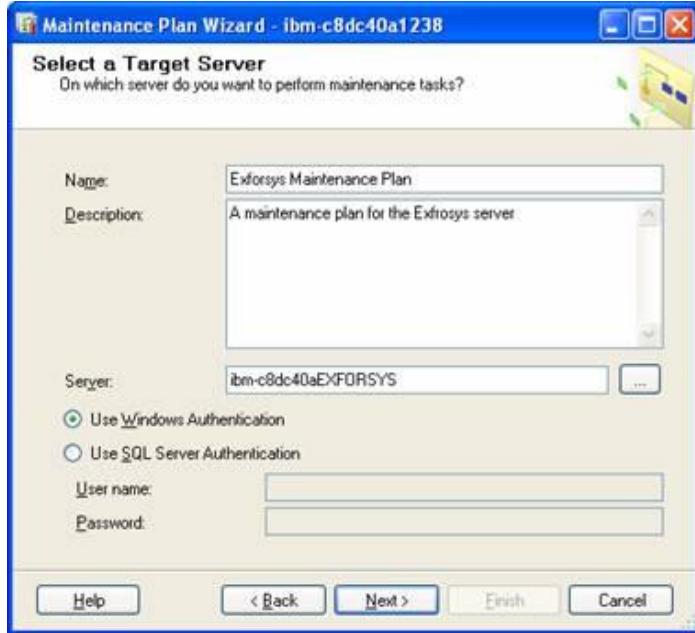


3. Right-click **Maintenance Plans** and select **Maintenance Plan Wizard**. This launches the wizard and you can now step through and create a plan customized to meet your maintenance requirements.

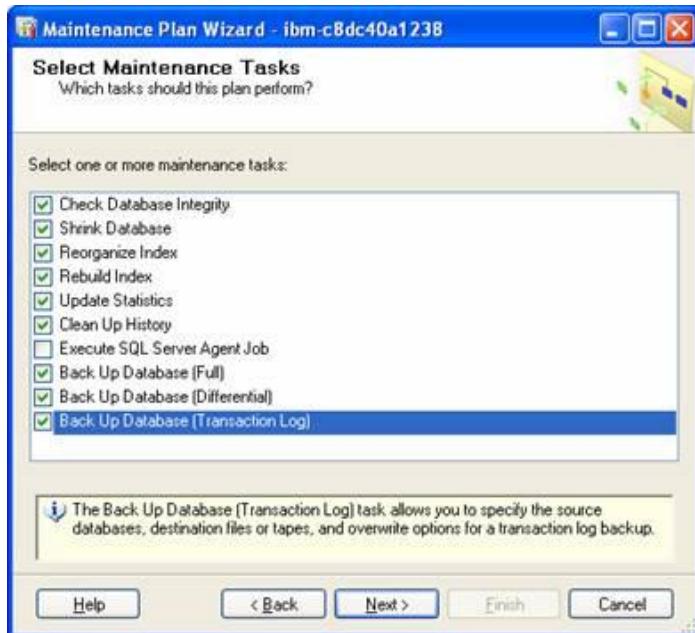


The first step is to give a name to the Maintenance plan along with a description. Enter the name, description, the server details and authentication mode and click Next.



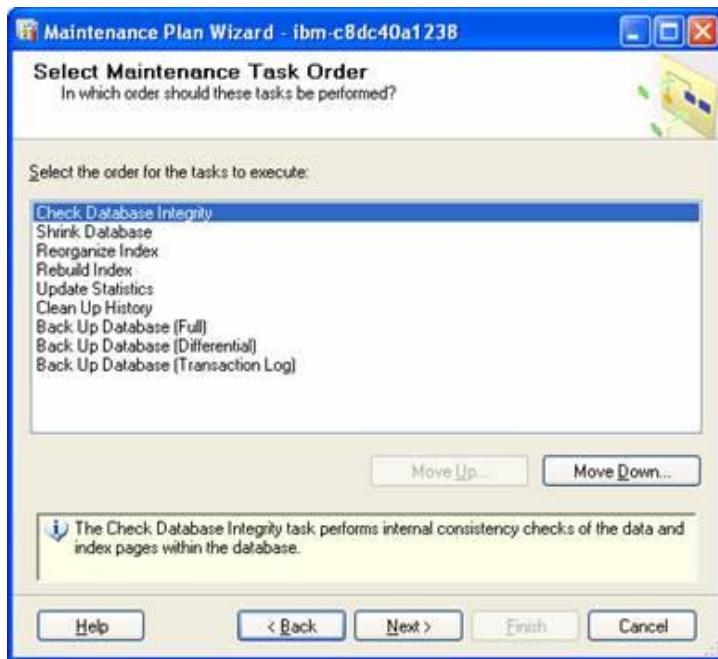


The next step is to specify the areas where the DBA wants to set up Maintenance options.



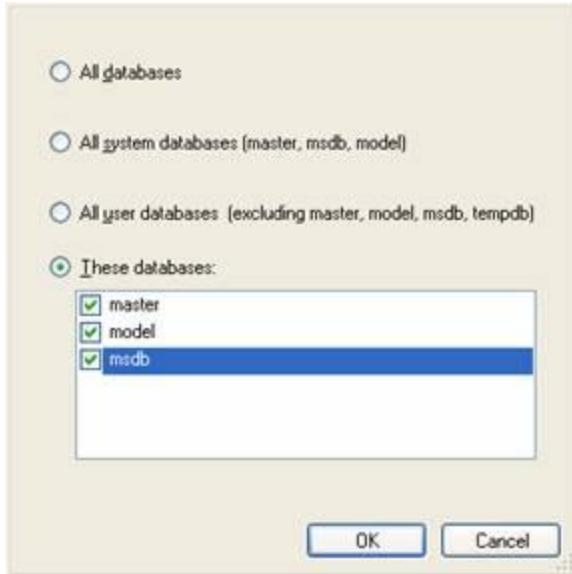
The Next screen prompts the Administrator select the order in which he wants the maintenance tasks performed. He can move the items up and down the list until he is satisfied with the order using the Move up.. and move down... buttons.



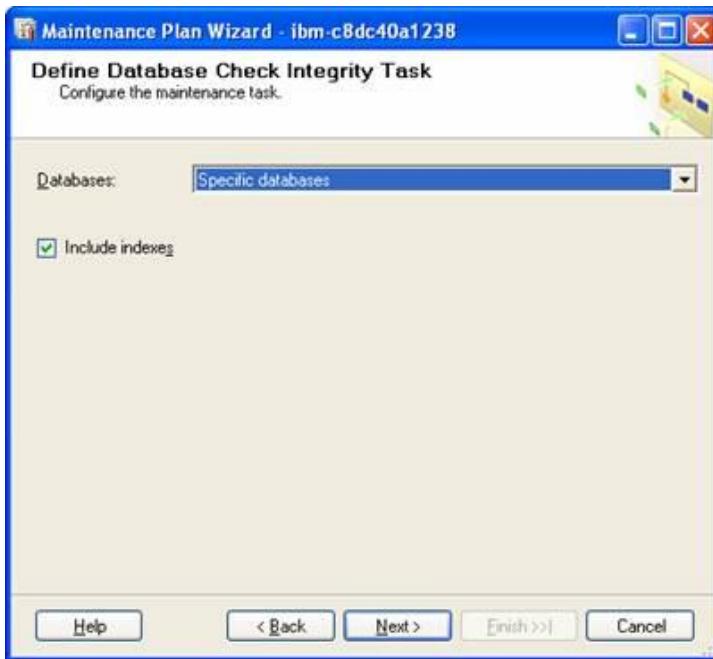


The next task is to specify the details for each task. The DBA is taken through Wizard screens for each of the tasks he has selected. The first is the Shrink Database task as per the list above. The DBA now specifies the databases in the window that pops up(illustrated below) when he clicks on the Databases combo box. After specifying the databases he returns to the Wizard screen and specifies the event at which the Shrinking should happen and the amount of free space that should remain after the shrink. He has also to specify whether the space should be retained or returned to the operating system.



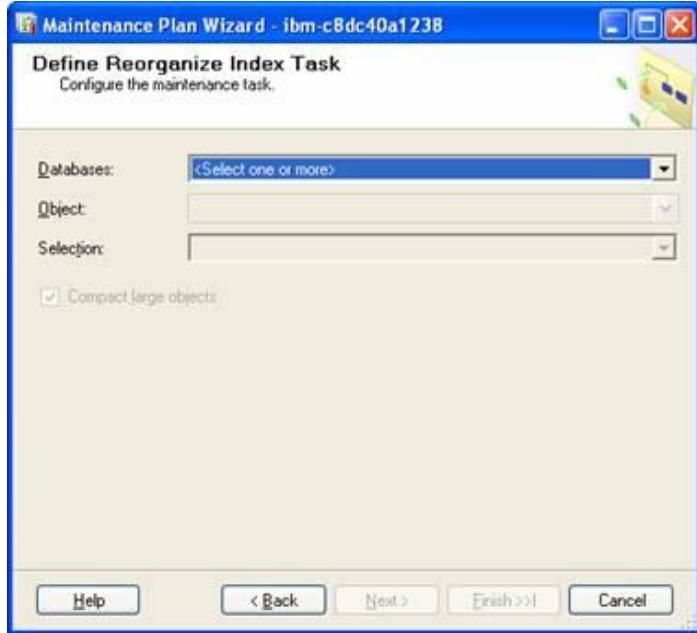


The Next task is the definition of the Database check integrity task. By default this is the first task. The user has to select the databases and specify whether indexes have to be included or not. The window that pops up is similar to the one above.

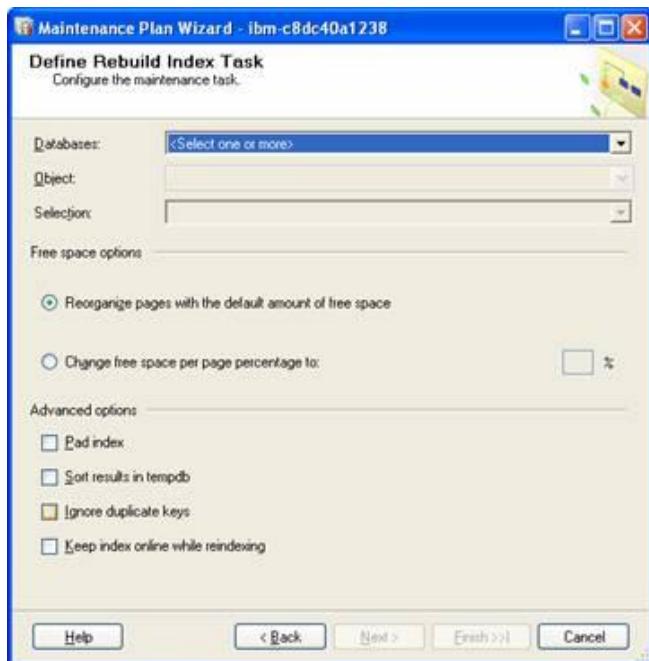


The Next task is to enter the specifications for the Reorganize Index task. Select the tables from the popup window and specify the tables and views required. The Administrator has to also specify whether he wants to compact large objects or not.





Rebuilding the Index involves selection of the databases, objects and setting the space options. The DBA can also set Advanced options such as Pad Index, Sort results to tempdb, Ignore duplicate keys and Keep index online while reindexing.

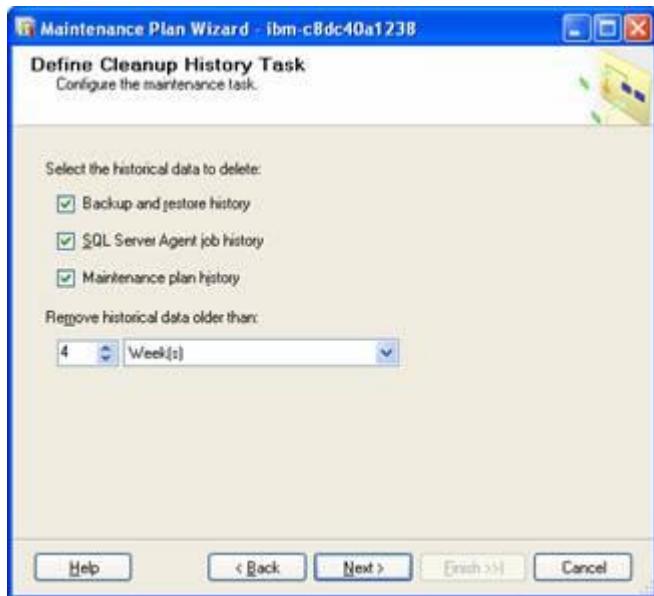


The Update statistics task prompts the user to select the database, objects and the update options.





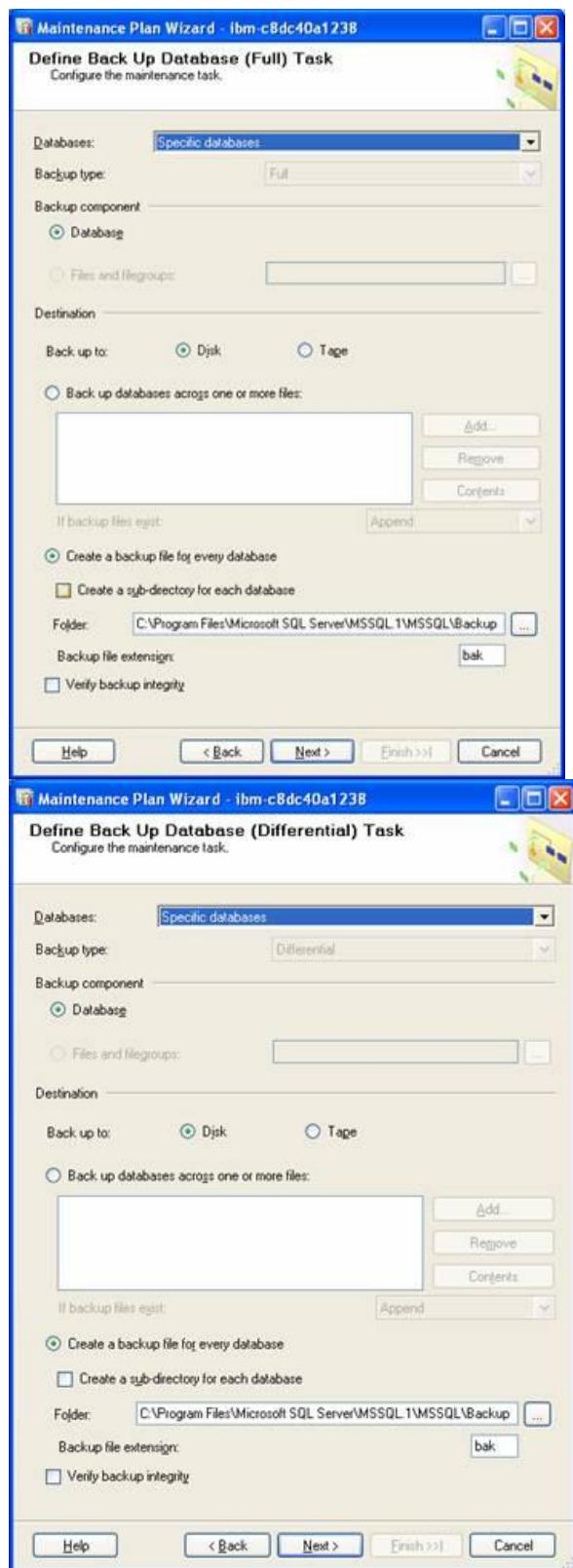
The Cleanup job specifies what needs to be backed up before cleaning.



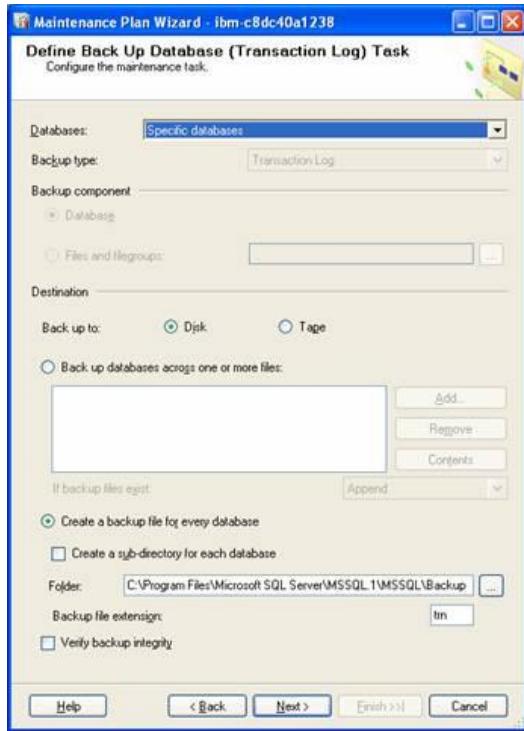
The Next task is the definition of [Back up options](#). The first is for Full task. There are a number of options which the user has to specify.

The next backup options are for differential backup. The user has to specify which databases have to be backed up, where it has to be backed up and so on...The user can also set up the option for a back up verification check by checking the check box provided for the purpose.

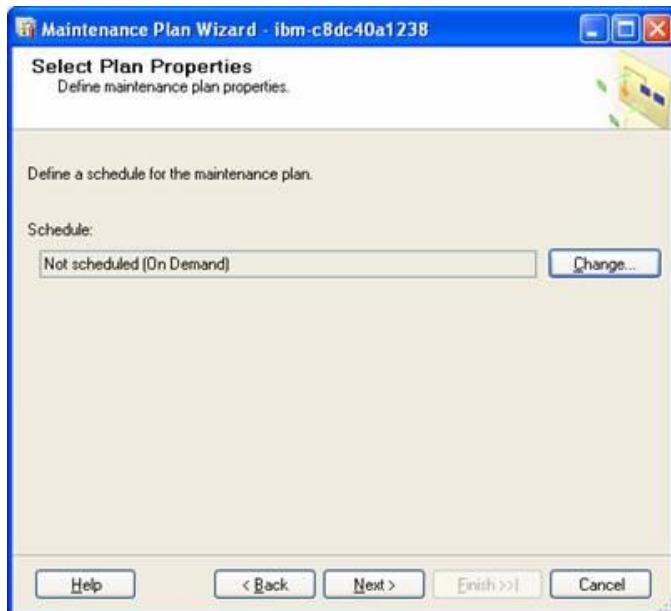


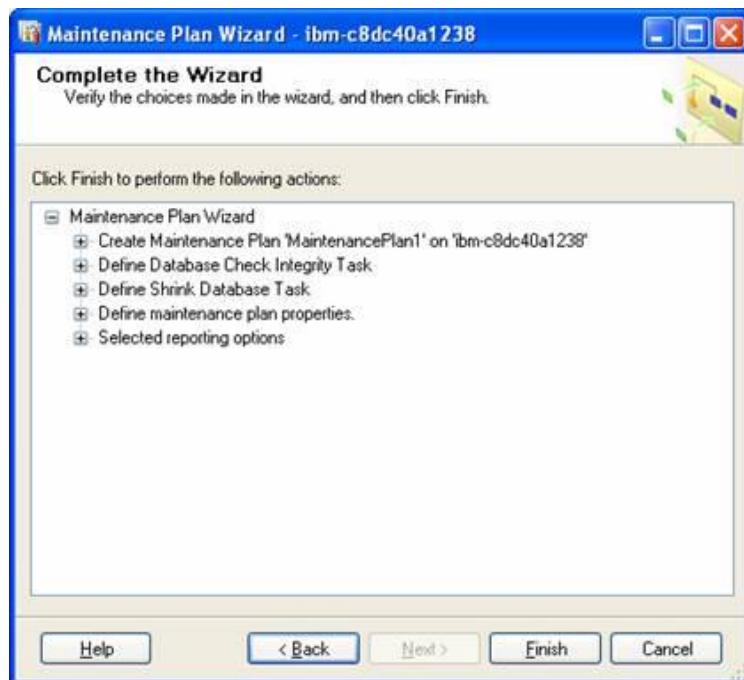
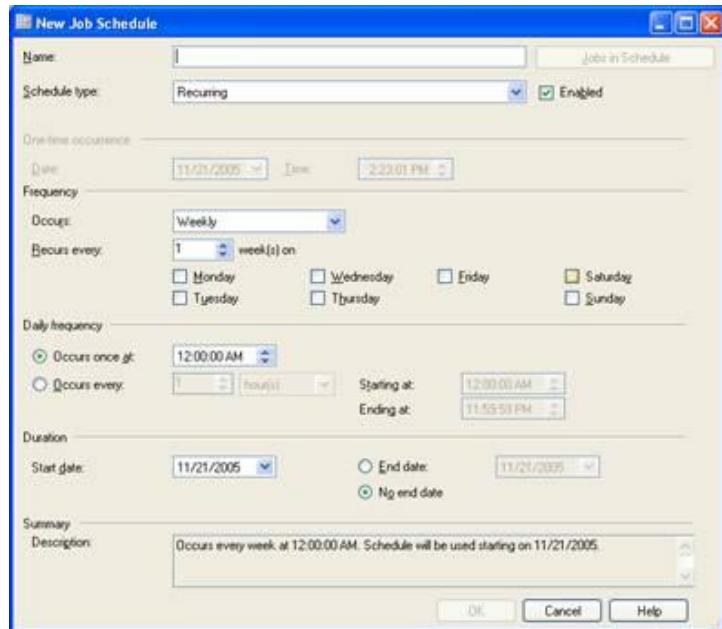


The Next options required to be set are the Transaction log backup options. Again the user has to specify what to backup, where to backup and so on.. He can also specify a backup integrity verification check.

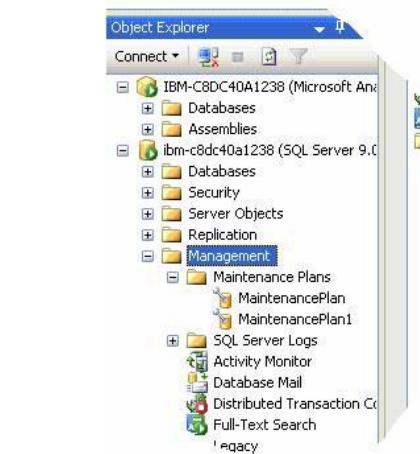
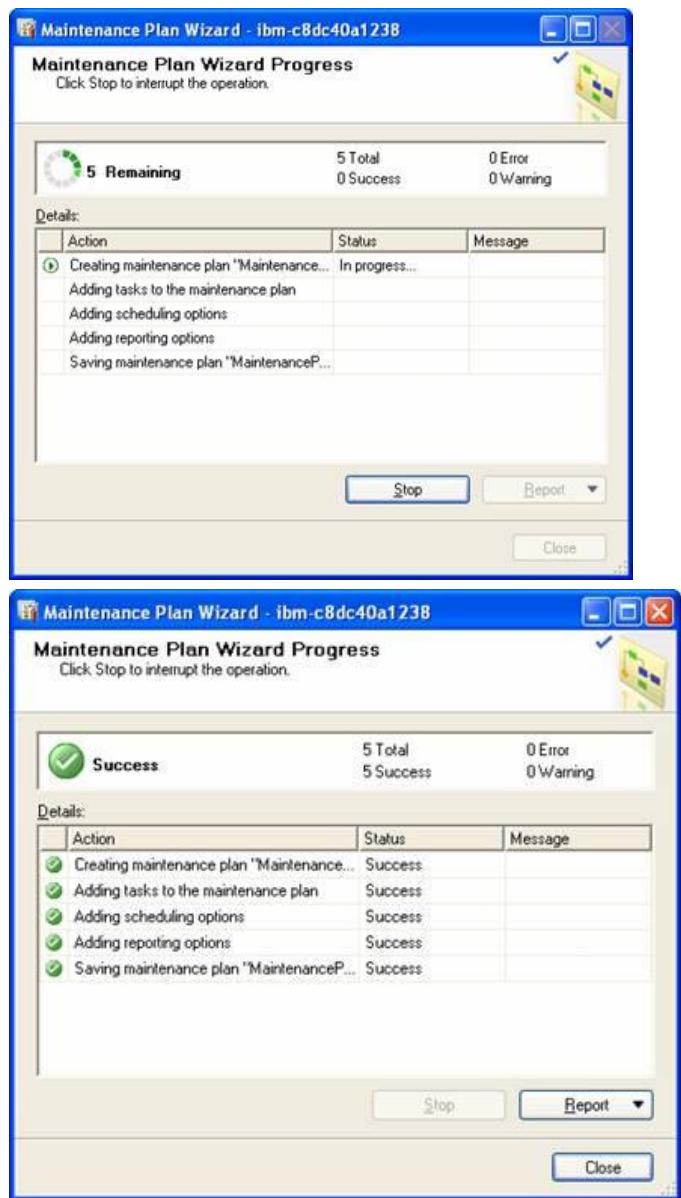


The Next screen prompts for Plan properties. The user has to specify whether he wants to retain default properties or he wants to change them. If he needs to change them, he has to click on the Change button and navigate through the screens setting the options he is prompted to give as illustrated in the screen shots below.



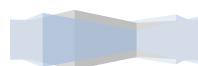


## SQL Server 2008 DBA



### **Best Practices on Job Maintenance**

1. Avoid overlapping jobs on the same SQL Server instance. Ideally, each job should run separately at different times.
2. When creating jobs, be sure to include error trapping, log job activity, and set up alerts so you know instantly when a job fails.
3. Create a special SQL Server login account whose sole purpose is to run jobs, and assign it to all jobs.
4. If your jobs include Transact-SQL code, ensure that it is optimized to run efficiently.
5. Periodically (daily, weekly, or monthly) perform a database reorganization on all the indexes on all the tables in all your database. This will rebuild the indexes so that the data is no longer logically fragmented. Fragmented data can cause SQL Server to perform unnecessary data reads, slowing down SQL Server's performance. Reindexing tables will also update column statistics.
6. Don't reindex your tables when your database is in active production, as it can lock resources and cause your users performance problems. Reindexing should be scheduled during down times, or during light use of the databases.
7. At least every two weeks, run DBCC CHECKDB on all your databases to verify database integrity.
8. Avoid running most DBCC commands during busy times of the day. These commands are often I/O intensive and can reduce performance of the SQL Server, negatively affecting users.
9. If you rarely restart the mssqlserver service, you may find that the current SQL Server log gets very large and takes a long time to load and view. You can truncate (essentially create a new log) the current server log by running DBCC ERRORLOG. Set this up as a weekly job.
10. Script all jobs and store these scripts in a secure area so they can be used if you need to rebuild the servers.



## Monitoring SQL Server

One of the primary responsibilities of the database administrator is the ongoing monitoring of SQL Server performance. Much of this monitoring can be automated, but for the most part, the monitoring results must be interpreted and acted upon in a systematic approach by the DBA. The monitoring job never ends, and it can become quite complex. Knowing what to monitor, when to monitor, and what constitutes acceptable and unacceptable behavior can become a full-time job. Making things worse is the fact that each SQL Server installation is different, making a global recommendation about what indicators identify unacceptable and acceptable performance very difficult.

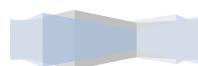
In this we will learn various tools used to monitor SQL Server and provides guidelines on how to use these tools to identify areas for optimization. Monitoring SQL Server can be a challenging process. SQL Server interacts heavily with every operating system subsystem. Some applications rely heavily on RAM, whereas others are CPU- or disk-intensive. SQL Server can be all three at the same time.

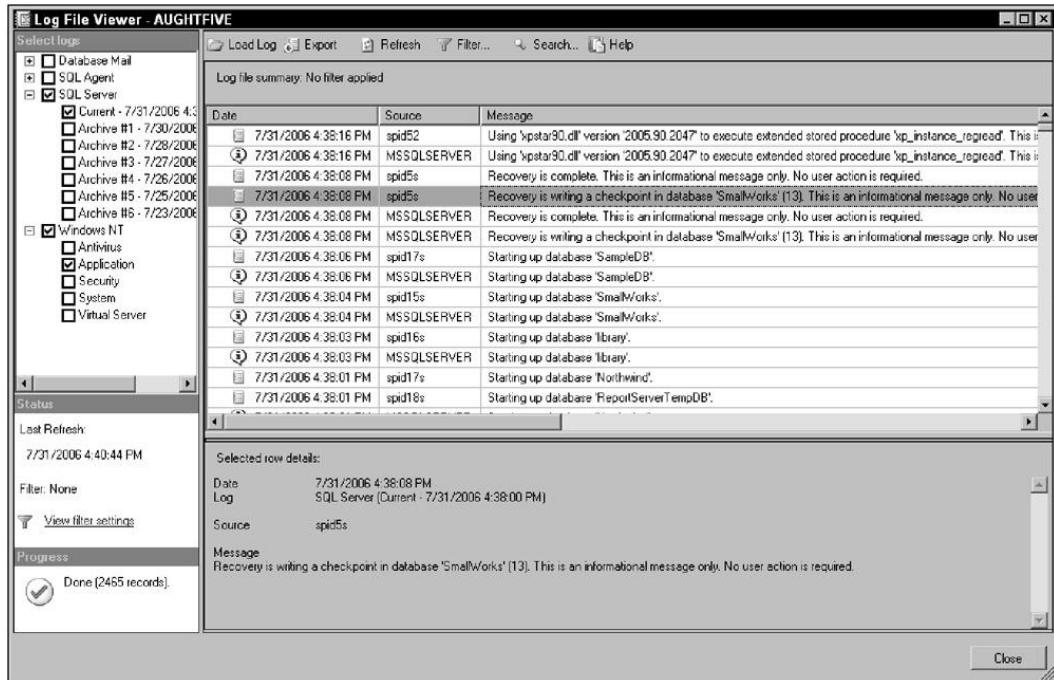
### Tools and Techniques for Monitoring

#### Log File Viewer

The Log File Viewer is an excellent tool for the viewing of SQL Server and operating system logs in a one-time correlated view. For example, memory subsystem errors from the system log can be correlated with SQL Server errors, indicating out-of-memory conditions and allowing you to isolate the problem away from SQL Server. To open the Log File Viewer, expand the Management folder in SQL Server Management Studio, expand SQL Server Logs, right-click the log you want to view, and select View SQL Server Log. Once the Log File Viewer is open, you can choose to open additional SQL Server logs and/or operating system logs by expanding and selecting the logs you want to review. Notice that you can also open up log files for the SQL Server Agent and Database Mail.

SQL Server and SQL Server Agent log files are closed and a new log opened every time the respective service is restarted. In a production system, this may not occur very often, resulting in a large log file. To avoid unacceptably large log files, the contents of the log files should be exported and the files cycled. To cycle the SQL Server Log, execute the `sp_cycle_errorlog` stored procedure. To cycle the Agent Log the `sp_cycle_agent_errorlog` stored procedure is used. These procedures clear the contents of the logs without requiring a service restart.

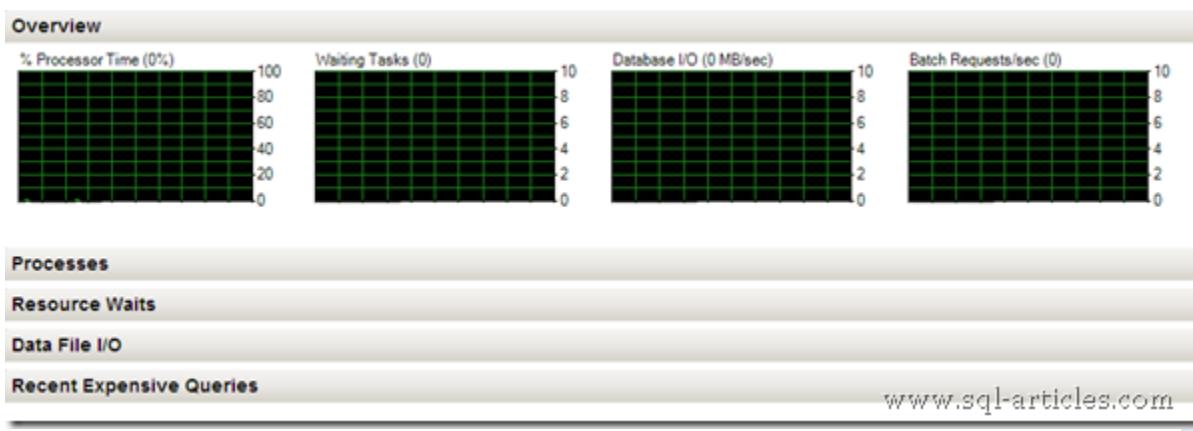




## Activity Monitor – SSMS 2008

Activity monitor has been refreshed in SQL Server 2008 Management studio. Microsoft has recoded the activity monitor written for SQL 2005 to provide more information on SQL Server processes and how these processes affect the current instance of SQL Server. The first change which I experienced is the real time graphical view of processes, CPU time etc, this really helps us to know the status of the server immediately.

Activity Monitor is a tabbed document, which has 5 tabs to show the status of the server as shown below. All these tabs are collapsible, so when you expand a tab it will collect information regarding to that tab and provide the output, if the tab is collapsed then it will stop collecting information from the server. Overview, Active User Tasks, Resource Waits, Data File I/O, and Recent Expensive Queries are the 5 tabs of activity monitor.

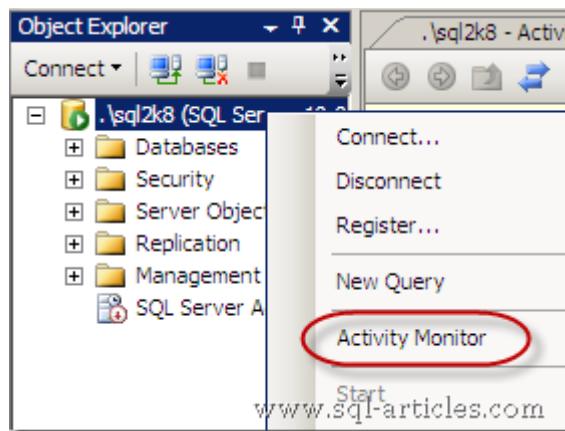


### How to open Activity monitor in SQL Server 2008 Management studio?

In the old version of management studio you can open activity monitor under Management folder, but now its changed in SSMS 2008. A user can open activity monitor in two ways either from object explorer or through tool bar. An additional option is also set to SSMS startup, you can enable activity monitor in SSMS startup so that when you open SSMS 2008 activity monitor will also gets opened.

Open through Object Explorer

- \* Connect to SQL server
- \* Right click on the SQL server instance name ,
- \* Now click on the activity monitor to open it



Open through Toolbar

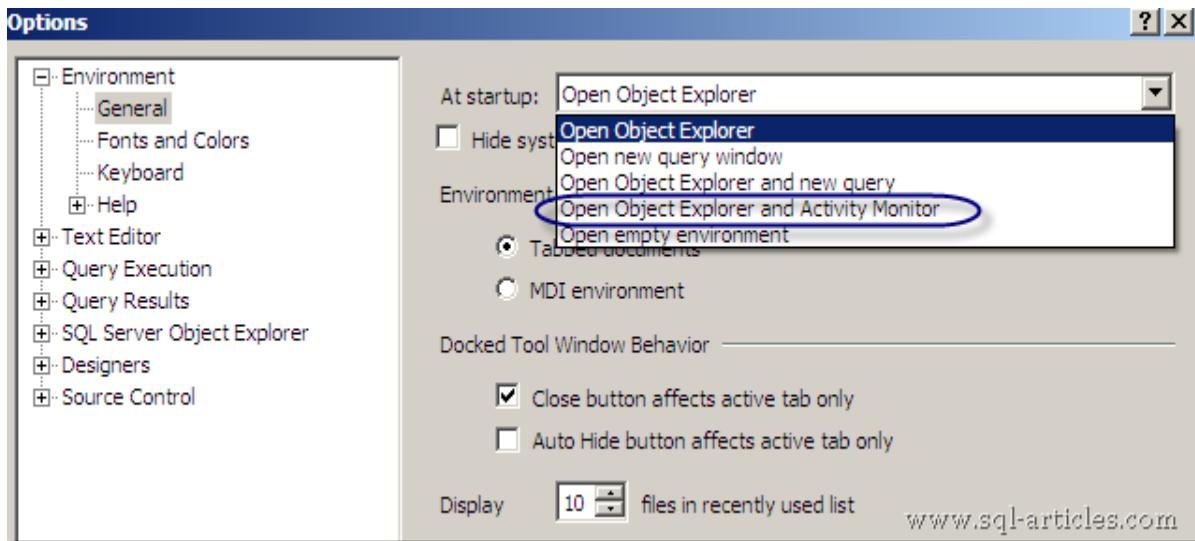
- \* Open Management studio 2008,
- \* In the standard toolbar you can find the icon shown below, clicking on that will open activity monitor.
- \* Once its opened you can connect to SQL server instance to get the status



Setting startup option in SSMS

- \* Open SSMS 2008, Go to Tools—> Options
- \* In the left hand pane expand Environment and click on General
- \* In the right hand pane click on "At startup" drop down menu , now select "Open Object explorer and activity monitor" as shown below
- \* Click Ok and restart SSMS



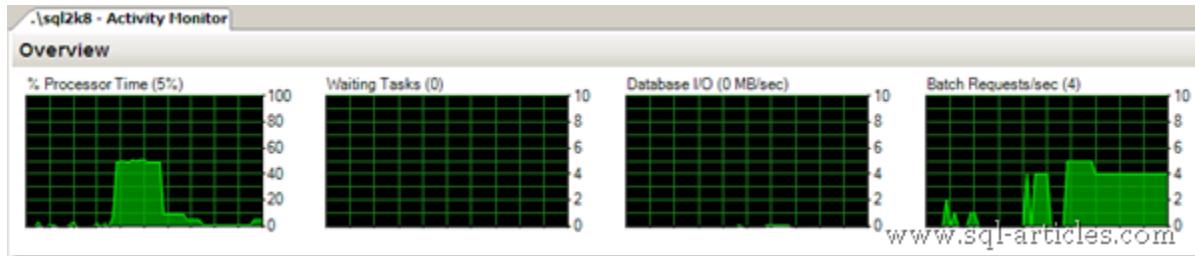


## Five tabs of Activity Monitor

Activity monitor has 5 tabs to describe the server status, lets see one by one. All these tabs are collapsed by default except Overview tab.

### Overview

Overview tab is the first tab and its expanded by default. This pane shows the graphical displays of instance information which has % Processor Time, Waiting Tasks, Database I/O and Batch Requests/sec as shown below



### Active User Tasks

This pane shows information for active user connections to the instance. This tab also provide more information in columns, you can rearrange the columns and sort or filter the columns as per your need. You can also run profiler from here for a particular process, just right click on a process and click on "Trace process in SQL Server profiler", profiler will start capturing the process.

Session ID	User	Proc	Login	Database	Task State	Command	Application	Wait Type	Wait Resource	Bloc By	Hea Bloc	Memory Use (KB)	Host Name	Workload Group
51	1	SAGAR\Ad...		master	RUNNING	SELECT	SqlDxe\admin...						16	SAGARPC
53	1	SAGAR\Ad...		tempdb	RUNNING	SELECT	MosDef S...						16	SAGARPC

The URL 'www.sql-articles.com' is visible at the bottom right.

## Resource Waits

This pane shows information about waits for resources for the instance.

Wait Category	Wait Time (ms/sec)	Recent Wait Time (ms/sec)	Average Waiter Count	Cumulative Wait Time (sec)
Network I/O	4	2	0.0	1
Other	0	0	0.0	9
Buffer I/O	0	0	0.0	36
Buffer Latch	0	0	0.0	0
Latch	0	0	0.0	0
Lock	0	0	0.0	3
Logging	0	0	0.0	1
Memory	0	0	0.0	0

www.sql-articles.com

## Data File I/O

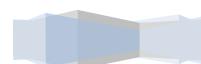
This pane shows information about the database files for the databases that belong to the instance.

Database	File Name	MB/sec Read	MB/sec Written	Response Time (ms)
AdventureWorks	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0
AdventureWorks	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0
AdventureWorks	G:\SQL2005\Data\indexfile.ndf	0.0	0.0	0
master	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0
master	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0
model	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0
model	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0
msdb	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0
msdb	C:\Program Files\Microsoft SQL Server\MSSQ...	0.0	0.0	0

www.sql-articles.com

## Recent Expensive Queries

This pane shows information about the most expensive queries that have been run on the instance over the last 30 seconds. The information is derived from the union of sys.dm\_exec\_requests and sys.dm\_exec\_query\_stats, and includes queries in process and queries that finished during the time period.



Recent Expensive Queries									
Query	Executions/r	CPU (ms/sec)	Physical Reads/sec	Logical Writes/sec	Logical Reads/sec	Average Duration (ms)	Plan Count	Database Name	
SELECT @current_request_count = cntr_value...	4	0	0	0	0	5	1	tempdb	
SELECT cfg.name AS [Name],cfg.configuration...	0	0	0	0	0	676	1	master	
DELETE FROM #am_wait_stats_snapshots W...	5	0	0	0	1	1	1	tempdb	
DELETE FROM #am_request_count WHERE ...	4	0	0	0	0	0	1	tempdb	
INSERT INTO #am_dbfileio (collection_time,to...	4	0	0	0	0	0	1	tempdb	
DELETE FROM #am_resource_mon_snap WH...	10	0	0	0	0	0	1	tempdb	
SELECT @current_total_io_mb = SUM(num_of...	4	0	0	0	0	1	1	tempdb	
INSERT INTO #am_wait_types VALUES (N'Ba...	6	0	0	0	0	www.sql-articles.com			

Activity monitor works with old version of SQL Servers?

Yes, Activity monitor works well when you connect to old versions of SQL. You will get the similar output in old versions too.

### What are the permissions required to use Activity Monitor?

If you are connecting to SQL 2005 or SQL 2008 then the user should have VIEW SERVER STATE permission.

If you are connecting to SQL 2000 then the user should have select permission on sysprocesses and syslocks tables in master table. By default public will have the privilege.

### Best Practices on Monitoring

1. Check OS Event Logs, SQL Server Logs, and Security Logs for unusual events.
2. Verify that all scheduled jobs have run successfully.
3. Confirm that backups have been made and successfully saved to a secure location.
4. Monitor disk space to ensure your SQL Servers won't run out of disk space.
5. Throughout the day, periodically monitor performance using both System Monitor and Profiler.
6. Use Enterprise Manager/Management Studio to monitor and identify blocking issues.
7. Keep a log of any changes you make to servers, including documentation of any performance issues you identify and correct.
8. Create SQL Server alerts to notify you of potential problems, and have them emailed to you. Take actions as needed.
9. Run the SQL Server Best Practices Analyzer on each of your server's instances on a periodic basis.



## Transaction Log Architecture

### 1. Transaction Log Logical Architecture

The SQL Server transaction log operates logically as if the transaction log is a string of log records. Each log record is identified by a log sequence number (LSN). Each new log record is written to the logical end of the log with an LSN that is higher than the LSN of the record before it.

Log records are stored in a serial sequence as they are created. Each log record contains the ID of the transaction that it belongs to. For each transaction, all log records associated with the transaction are individually linked in a chain using backward pointers that speed the rollback of the transaction.

Many types of operations are recorded in the transaction log. These operations include:

- The start and end of each transaction.
- Every data modification (insert, update, or delete). This includes changes by system stored procedures or data definition language (DDL) statements to any table, including system tables.
- Every extent and page allocation or de-allocation.
- Creating or dropping a table or index.

Rollback operations are also logged. Each transaction reserves space on the transaction log to make sure that enough log space exists to support a rollback that is caused by either an explicit rollback statement or if an error is encountered. The amount of space reserved depends on the operations performed in the transaction, but generally is equal to the amount of space used to log each operation. This reserved space is freed when the transaction is completed.

The section of the log file from the first log record that must be present for a successful database-wide rollback to the last-written log record is called the active part of the log, or the *active log*. This is the section of the log required to do a full recovery of the database. No part of the active log can ever be truncated.

### 2. Transaction Log Physical Architecture

The transaction log is used to guarantee the data integrity of the database and for data recovery. The topics in this section provide the information about the physical architecture of the transaction log. Understanding the physical architecture can improve your effectiveness in managing transaction logs.

The transaction log in a database maps over one or more physical files. Conceptually, the log file is a string of log records. Physically, the sequence of log records is stored efficiently in the set of physical files that implement the transaction log.

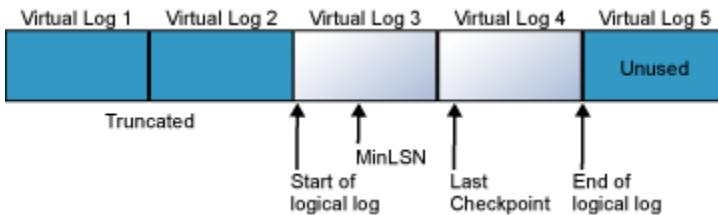
The SQL Server Database Engine divides each physical log file internally into a number of virtual log files. Virtual log files have no fixed size, and there is no fixed number of virtual log files for a physical log file. The Database Engine chooses the size of the virtual log files dynamically while it is creating or extending log files. The Database Engine tries to maintain a small number of virtual files. The size of the



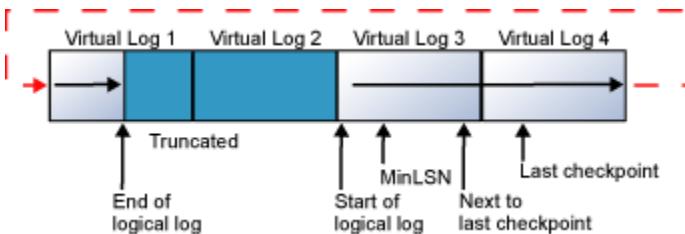
virtual files after a log file has been extended is the sum of the size of the existing log and the size of the new file increment. The size or number of virtual log files cannot be configured or set by administrators.

The only time virtual log files affect system performance is if the log files are defined by small *size* and *growth\_increment* values. If these log files grow to a large size because of many small increments, they will have lots of virtual log files. This can slow down database startup and also log backup and restore operations. We recommend that you assign log files a *size* value close to the final size required, and also have a relatively large *growth\_increment* value.

The transaction log is a wrap-around file. For example, consider a database with one physical log file divided into four virtual log files. When the database is created, the logical log file begins at the start of the physical log file. New log records are added at the end of the logical log and expand toward the end of the physical log. Log truncation frees any virtual logs whose records all appear in front of the minimum recovery log sequence number (MinLSN). The *MinLSN* is the log sequence number of the oldest log record that is required for a successful database-wide rollback. The transaction log in the example database would look similar to the one in the following illustration.



When the end of the logical log reaches the end of the physical log file, the new log records wrap around to the start of the physical log file.



This cycle repeats endlessly, as long as the end of the logical log never reaches the beginning of the logical log. If the old log records are truncated frequently enough to always leave sufficient room for all the new log records created through the next checkpoint, the log never fills. However, if the end of the logical log does reach the start of the logical log, one of two things occurs:

- If the FILEGROWTH setting is enabled for the log and space is available on the disk, the file is extended by the amount specified in *growth\_increment* and the new log records are added to the extension. For more information about the FILEGROWTH setting, see ALTER DATABASE (Transact-SQL).
- If the FILEGROWTH setting is not enabled, or the disk that is holding the log file has less free space than the amount specified in *growth\_increment*, an 9002 error is generated.

If the log contains multiple physical log files, the logical log will move through all the physical log files before it wraps back to the start of the first physical log file.

### 3. Checkpoint Operation

Checkpoints flush dirty data pages from the buffer cache of the current database to disk. This minimizes the active portion of the log that must be processed during a full recovery of a database. During a full recovery, the following types of actions are performed:

- The log records of modifications not flushed to disk before the system stopped are rolled forward.
- All modifications associated with incomplete transactions, such as transactions for which there is no COMMIT or ROLLBACK log record, are rolled back.
- A checkpoint performs the following processes in the database:
  - Writes a record to the log file, marking the start of the checkpoint.
  - Stores information recorded for the checkpoint in a chain of checkpoint log records.

One piece of information recorded in the checkpoint is the log sequence number (LSN) of the first log record that must be present for a successful database-wide rollback. This LSN is called the Minimum Recovery LSN (MinLSN). The MinLSN is the minimum of the:

- LSN of the start of the checkpoint.
- LSN of the start of the oldest active transaction.

The checkpoint records also contain a list of all the active transactions that have modified the database.

- If the database uses the simple recovery model, marks for reuse the space that precedes the MinLSN.
- Writes all dirty log and data pages to disk.
- Writes a record marking the end of the checkpoint to the log file.
- Writes the LSN of the start of this chain to the database boot page.

#### Activities That Cause a Checkpoint

Checkpoints occur in the following situations:

- A CHECKPOINT statement is explicitly executed. A checkpoint occurs in the current database for the connection.
- A minimally logged operation is performed in the database; for example, a bulk-copy operation is performed on a database that is using the Bulk-Logged recovery model.
- Database files have been added or removed by using ALTER DATABASE.

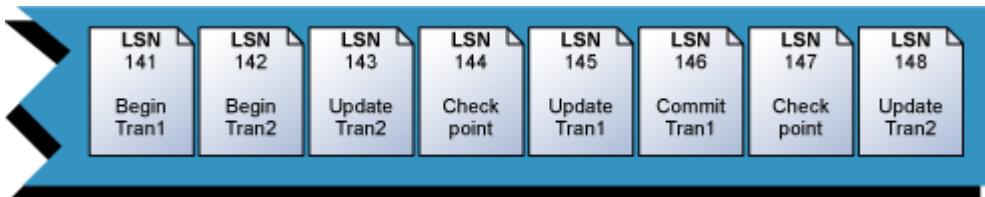


- An instance of SQL Server is stopped by a SHUTDOWN statement or by stopping the SQL Server (MSSQLSERVER) service. Either action causes a checkpoint in each database in the instance of SQL Server.
- An instance of SQL Server periodically generates automatic checkpoints in each database to reduce the time that the instance would take to recover the database.
- A database backup is taken.
- An activity requiring a database shutdown is performed. For example, AUTO\_CLOSE is ON and the last user connection to the database is closed, or a database option change is made that requires a restart of the database.

### Active Log

The section of the log file from the MinLSN to the last-written log record is called the active portion of the log, or the *active log*. This is the section of the log required to do a full recovery of the database. No part of the active log can ever be truncated. All log records must be truncated from the parts of the log before the MinLSN.

The following illustration shows a simplified version of the end-of-a-transaction log with two active transactions. Checkpoint records have been compacted to a single record.



LSN 148 is the last record in the transaction log. At the time that the recorded checkpoint at LSN 147 was processed, Tran 1 had been committed and Tran 2 was the only active transaction. That makes the first log record for Tran 2 the oldest log record for a transaction active at the time of the last checkpoint. This makes LSN 142, the Begin transaction record for Tran 2, the MinLSN.

### Long-Running Transactions

The active log must include every part of all uncommitted transactions. An application that starts a transaction and does not commit it or roll it back prevents the Database Engine from advancing the MinLSN. This can cause two types of problems:

- If the system is shut down after the transaction has performed many uncommitted modifications, the recovery phase of the subsequent restart can take much longer than the time specified in the **recovery interval** option.
- The log might grow very large, because the log cannot be truncated past the MinLSN. This occurs even if the database is using the simple recovery model, in which the transaction log is generally truncated on each automatic checkpoint.



#### 4. Write-Ahead Transaction Log

SQL Server uses a write-ahead log (WAL), which guarantees that no data modifications are written to disk before the associated log record is written to disk.

To understand how the write-ahead log works, it is important for you to know how modified data is written to disk. SQL Server maintains a buffer cache into which it reads data pages when data must be retrieved. Data modifications are not made directly to disk, but are made to the copy of the page in the buffer cache. The modification is not written to disk until a checkpoint occurs in the database, or the modification must be written to disk so the buffer can be used to hold a new page. Writing a modified data page from the buffer cache to disk is called flushing the page. A page modified in the cache, but not yet written to disk, is called a *dirty page*.

At the time a modification is made to a page in the buffer, a log record is built in the log cache that records the modification. This log record must be written to disk before the associated dirty page is flushed from the buffer cache to disk. If the dirty page is flushed before the log record is written, the dirty page creates a modification on the disk that cannot be rolled back if the server fails before the log record is written to disk. SQL Server has logic that prevents a dirty page from being flushed before the associated log record is written. Log records are written to disk when the transactions are committed.

### Managing the Transaction Log

Log truncation, which is automatic under the simple recovery model, is essential to keep the log from filling. The truncation process reduces the size of the logical log file by marking as inactive the virtual log files that do not hold any part of the logical log. In some cases, however, physically shrinking or expanding the physical log file is useful.

#### Transaction Log Truncation

If log records were never deleted from the transaction log, it would eventually fill all the disk space that is available to the physical log files. Log truncation automatically frees space in the logical log for reuse by the transaction log.

Except when delayed for some reason, log truncation occurs automatically as follows:

- Under the simple recovery model, after a checkpoint.
- Under the full recovery model or bulk-logged recovery model, after a log backup, if a checkpoint has occurred since the previous backup

#### How Log Truncation Works

Truncation does not reduce the size of a physical log file. Reducing the physical size of a log file requires shrinking the file.



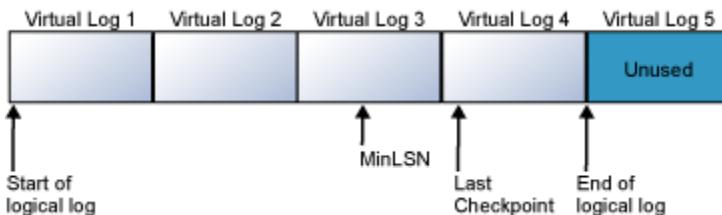
The transaction log is a wrap-around file. When the database is created, the logical log file begins at the start of the physical log file. New log records are added at the end of the logical log and expand toward the end of the physical log. The transaction log in a database maps over one or more physical files. The SQL Server Database Engine divides each physical log file internally into a number of virtual log files. Log truncation frees space in the logical log by deleting inactive virtual log files from the start of the logical log.

Virtual log files are the unit of space that can be reused. Only virtual log files that contain just inactive log records can be truncated. The active portion of the transaction log, the *active log*, cannot be truncated, because the active log is required to recover the database. The most recent checkpoint defines the active log. The log can be truncated up to that checkpoint.

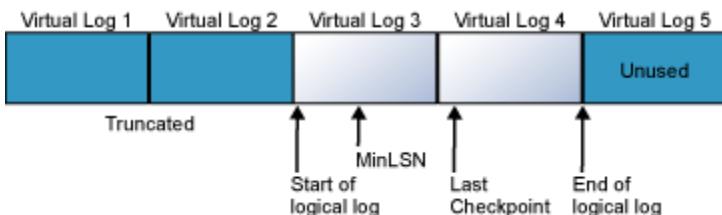
When the checkpoint is performed, the inactive portion of the transaction log is marked as reusable. Thereafter, the inactive portion can be freed by log truncation. Truncation frees the inactive virtual log files for reuse. Eventually, when a new record is written to a freed virtual log, that virtual log file becomes active again.

One piece of information recorded in a checkpoint is the log sequence number (LSN) of the first log record that must be present for a successful database-wide rollback. This LSN is called the *minimum recovery LSN* (*MinLSN*). The start of the active portion of the log is the virtual log that contains the MinLSN. When a transaction log is truncated, only the log records in front of this virtual log file are freed for reuse.

The following illustrations show a transaction log before and after truncation. The first illustration shows a transaction log that has never been truncated. Currently, four virtual log files are in use by the logical log. The logical log starts at the front of the first virtual log file and ends at virtual log 4. The MinLSN record is in virtual log 3. Virtual log 1 and virtual log 2 contain only inactive log records. These records can be truncated. Virtual log 5 is still unused and is not part of the current logical log.



The second illustration shows how the log appears after being truncated. Virtual log 1 and virtual log 2 have been freed for reuse. The logical log now starts at the beginning of virtual log 3. Virtual log 5 is still unused, and it is not part of the current logical log.



## Managing the Size of the Transaction Log File

### Monitoring Log Space Use

You can monitor log space use by using DBCC SQLPERF (LOGSPACE). This command returns information about the amount of log space currently used and indicates when the transaction log is in need of truncation.

For information about the current size of a log file, its maximum size, and the autogrow option for the file, you can also use the **size**, **max\_size**, and **growth** columns for that log file in **sys.database\_files**.

### Shrinking the Size of the Log File

Log truncation is essential because it frees disk space for reuse, but it does not reduce the size if the physical log file. To reduce its physical size, the log file must be shrunk to remove one or more virtual log files that do not hold any part of the logical log (that is, *inactive virtual log files*). When a transaction log file is shrunk, enough inactive virtual log files are removed from the end of the log file to reduce the log to approximately the target size.

### Shrinking the Transaction Log

If you know that a transaction log file contains unused space that you will not be needing, you can reclaim the excess space by reducing the size of the transaction log. This process is known as *shrinking* the log file.

Shrinking can occur only while the database is online and, also, while at least one virtual log file is free. In some cases, shrinking the log may not be possible until after the next log truncation.

### DBCC SHRINKFILE (Transact-SQL)

Shrinks the size of the specified data or log file for the current database or empties a file by moving the data from the specified file to other files in the same filegroup, allowing the file to be removed from the database. You can shrink a file to a size that is less than the size specified when it was created. This resets the minimum file size to the new value.

### Shrinking a data file to a specified target size

The following example shrinks the size of a data file named `DataFile1` in the `UserDB` user database to 7 MB.

```
USE UserDB;
GO
DBCC SHRINKFILE (DataFile1, 7);
GO
```



## Backup & Restore

### Introduction

Backing up a database is one of the most important things you need to do when having a database driven application. It's only all of your data in there, right? But often developers and management don't realize the importance of backups and overall proper backup strategy for the most important side of the business – data and it's consistency.

### How backup works

When you run a backup database command SQL Server performs a Checkpoint on the data pages in memory. Checkpoint means that all transactionally committed dirty pages are written to disk. Dirty pages are simply changed pages in memory that haven't been written to disk yet. After this the data on the disk is backed up in one or multiple files depending on your requirements. A backup must be able to be restored to a transactionally consistent state which means that it will also contain the data from the transaction log needed to undo all of the transactions which are running while the backup is taken.

### Recovery models

#### Full recovery model

Every DML (insert, update, delete) statement is fully logged. Also every bulk insert operation (BCP, BulkInsert, SqlBulkCopy in .Net, SELECT INTO) is fully logged for each row. This recovery model also logs every CREATE INDEX and ALTER INDEX statement which are DDL operations. This means that when you recover a transaction log that contains logged CREATE INDEX you don't have to rebuild an index since it's already built. This behavior has changed since SQL Server 2000 where only the index creation event was logged, and not the whole index. The downside of this model is that it can consume a lot of disk space very fast.

#### Bulk-Logged recovery model

This model differs from the Full recovery in that it doesn't log every row insert on BULK operations I mentioned in Full recovery model. SQL Server does log that the operation has been executed and information about the data page allocations. However all data can still be restored. This is handled by the Bulk Change Map (BCM). SQL Server keeps track of the changed extents under this model by setting a bit representing an extent to 1 in the BCM, if that extent has changed. That is why the BULK operations are also faster under this model than under the Full Recovery, since logging each row is much slower than just setting a bit to 1. When you take a transaction log backup, all changed extents are also backed up by reading the BCM. This means that the transaction log itself is smaller but the transaction log backup can be a lot larger than the one under Full recovery model. Note that under this model you can't do point-in-time recovery on any transaction log backup that contains a bulk-logged transaction.



## Simple recovery model

Under this recovery model you can't backup a transaction log at all. An attempt to do so results in an error, since there's nothing to update. The transaction log gets truncated at every checkpoint (writing data from a log to a disk) which happens at predetermined intervals. Also changing the database recovery model to Simple will immediately truncate the transaction log. A common misunderstanding is that nothing is being logged under this model. That is NOT TRUE. Everything is logged, you just don't have the point-in-time recovery ability. Bulk operations are minimally logged as in Bulk-Logged recovery model.

## Transaction log marks

These are only available under Full and Bulk logged recovery models. Log marks are set in the transaction you want to recover to. You can do this with the begin transaction statement:

```
BEGIN TRANSACTION TranMark1 WITH MARK 'The mark description'
```

The name of the mark in the transaction log is TranMark1. After this transaction commits the mark is inserted into the logmarkhistory table in the msdb database and into the transaction logs of other related databases. Related databases are beyond the scope of this article, but simply put: the databases are related when we make related updates to them (e.g.: update to table1 in db1 has to be followed by update to table2 in db2). With log marks over multiple databases we can recover all databases to a specific related state to each other.

## Types of Backups:

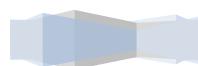
### Full database Backup

This backs up the whole database. In order to have further differential or transaction log backups you have to create the full database backup first.

```
-- Back up the AdventureWorks as full backup  
BACKUP DATABASE AdventureWorks  
TO DISK = N'c:\AdventureWorksDiff.bak'
```

### Differential database backup

Differential database backups are cumulative. This means that each differential database backup backs up all the changes from the last Full database backup and NOT last Differential backup.



```
-- Back up the AdventureWorks as differential backup  
BACKUP DATABASE AdventureWorks  
    TO DISK = N'c:\AdventureWorksDiff.bak' WITH DIFFERENTIAL
```

### Transaction log backup

Transaction log backup aren't possible under the Simple Recovery model. Two transaction logs can't contain the same transactions which means that when restoring you have to restore every backup in the order they were taken

```
-- Back up the AdventureWorks transaction log  
BACKUP LOG AdventureWorks  
    TO DISK = N'c:\AdventureWorksLog.bak'
```

### Tail log backup

There seems to be a lot of confusion about this one since it's a new term in SQL Server 2008 (I haven't heard it being used in SS2k). A Tail log backup is the last Transaction log backup that you make prior to restoring a database. What this means is that if your db crashes for whatever reason, you have to backup your transaction log so that you can do point in time recovery. This last backup is called Tail log backup. If your data file (MDF) is unavailable you need to use WITH NO\_TRUNCATE option:

```
-- Back up the AdventureWorks tail-log  
BACKUP LOG AdventureWorks  
    TO DISK = N'c:\AdventureWorksTailLog.bak' WITH NO_TRUNCATE
```

If your database is in OFFLINE or EMERGENCY state then tail log backup isn't possible.

### Mirrored backup

Mirrored backups simply write the backup to more than one destination. You can write up to four mirrors per media set. This increases the possibility of a successful restore if a backup media gets corrupted. Following statement gives us two backups that we can restore from:

```
-- make one backup on d: disk and a mirror on e: disk  
BACKUP DATABASE AdventureWorks  
    TO DISK = 'd:\AdventureWorksMirror_1.bak'  
    MIRROR  
    TO DISK = 'e:\AdventureWorksMirror_2.bak'  
-- create a new mirrored backup set  
    WITH FORMAT;
```

### **Copy-only backup**

Copy-only backups are new in SQL Server 2008 and are used to create a full database or transaction log backup without breaking the log chain. A copy-only full backup can't be used as a basis for a differential backup, nor can you create a differential copy only backup.

```
-- Back up the AdventureWorks database as copy only  
BACKUP DATABASE AdventureWorks  
    TO DISK = N'c:\AdventureWorksDiff.bak' WITH COPY_ONLY  
  
-- Back up the AdventureWorks transaction log as copy only  
BACKUP LOG AdventureWorks  
    TO DISK = N'e:\AdventureWorksLog.bak' WITH COPY_ONLY
```

### **File and file group backup**

If you have your data spread across multiple files or filegroups you can take a full or differential backup of file(s) or filegroup(s):

```
-- Backup AdventureWorks_file1 file of the  
-- AdventureWorks database to disk  
-- note that AdventureWorks has to have a AdventureWorks_file1 file  
BACKUP DATABASE AdventureWorks  
    FILE = 'AdventureWorks_file1'  
    TO DISK = 'e:\AdventureWorks_file1.bak'  
GO  
  
-- Backup AdventureWorks_filegroup1 filegroup of the  
-- AdventureWorks database to disk  
-- note that AdventureWorks has to have a  
-- AdventureWorks_filegroup1 filegroup  
BACKUP DATABASE AdventureWorks  
    FILEGROUP = 'AdventureWorks_filegroup1'  
    TO DISK = 'e:\AdventureWorks_filegroup1.bak'  
GO
```

For differential backups just add WITH DIFFERENTIAL option to the upper examples.

### **Partial Filegroup backup**

Partial filegroup backups are used to backup large databases with one or more read-only filegroups. In a way they are similar to full database backup, but by default they don't include read-only files or filegroups. This means that they contain the primary filegroup, every read/write filegroup and an optional number of read only files or filegroups.



```
-- Create a partial filegroup backup. backs up all read/write filegroup  
-- and the read only AW_ReadOnly_FileGroup1 filegroup  
BACKUP DATABASE AdventureWorks  
    READ_WRITE_FILEGROUPS, FILEGROUP = 'AW_ReadOnly_FileGroup1'  
    TO DISK = 'e:\AdventureWorks_partial.bak'
```

For differential backups just add WITH DIFFERENTIAL option to the upper example. Note that all file and file group names are logical and not physical names.

### **Restore:**

#### **Recovery states:**

To determine the state of the database after the store operation, you must select one of the options of the Recovery state panel.

#### **RESTORE WITH RECOVERY**

Leave the database ready for use by rolling back the uncommitted transactions. Additional transaction logs cannot be restored.

Recovers the database after restoring the final backup checked in the Select the backup sets to restore grid on the General page. This is the default option and is equivalent to specifying WITH RECOVERY in a RESTORE statement (Transact-SQL).

```
RESTORE DATABASE [AdventureWorksNew]  
    FROM DISK =  
N'\\nas\\Backup\\L40\\SQL2008\\AdventureWorks_backup_200702120215.bak'  
    WITH FILE = 1,  
        MOVE N'AdventureWorks_Data' TO  
N'C:\\Data\\MSSQL.1\\MSSQL\\Data\\AdventureWorksNew_Data.mdf',  
        MOVE N'AdventureWorks_Log' TO  
N'C:\\Data\\MSSQL.1\\MSSQL\\Data\\AdventureWorksNew_Log.ldf'
```

#### **RESTORE WITH NORECOVERY:**

Leave the database non-operational, and do not roll back the uncommitted transactions. Additional transaction logs can be restored. ---Used in Mirroring

Leaves the database in the restoring state. This allows you to restore additional backups in the current recovery path. To recover the database, you will have to perform a restore operation by using the RESTORE WITH RECOVERY option (see the preceding option).

This option is equivalent to specifying WITH NORECOVERY in a RESTORE statement.

If you select this option, the Preserve replication settings option is unavailable.

176

#### **RESTORE WITH STANDBY:**



Leave the database in read-only mode. Undo uncommitted transactions, but save the undo actions in a standby file so that recovery effects can be reverted. () ---Used in log-shipping

Leaves the database in a standby state, in which the database is available for limited read-only access. This option is equivalent to specifying WITH STANDBY in a RESTORE statement.

Choosing this option requires that you specify a standby file in the Standby file text box. The standby file allows the recovery effects to be undone.

Standby file : Specifies a standby file. You can browse for the standby file or enter its pathname directly in the text box.

## Practical troubleshooting

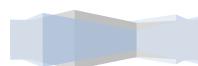
### 1. Backup & Restore for SQL Server 2005&2008

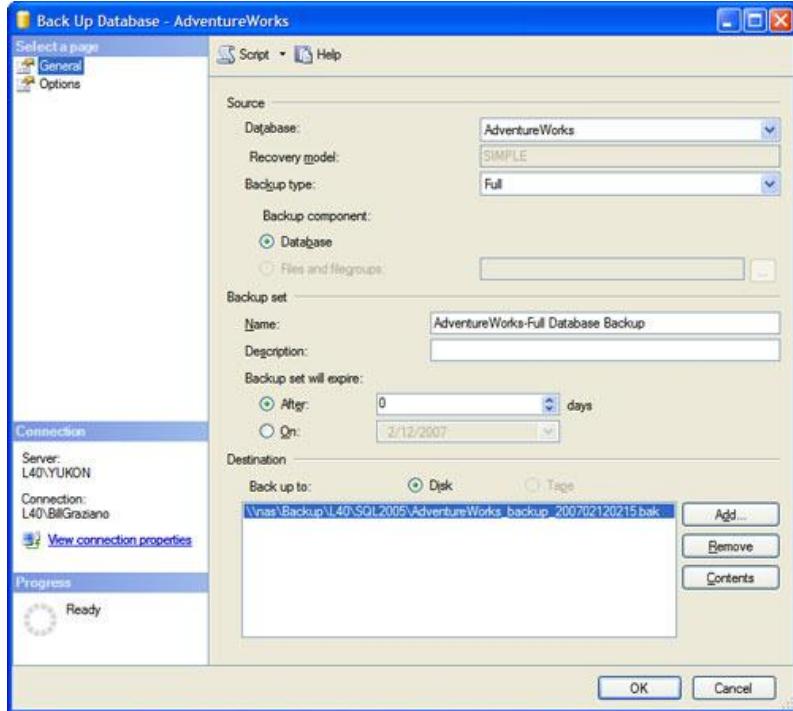
In a typical installation SQL Server stores its data in two files. One has an MDF extension and stores the data itself and the other has an LDF extension and stores the transaction log. You can configure SQL Server to have multiple data files and multiple transaction log files if you'd like but that's beyond the scope of this article. When SQL Server processes a transaction it goes through the following steps:

1. It writes what it's going to do to the transaction log.
2. It makes the change to the data file. This change is typically made on the in-memory copy of that portion of the data file.
3. It writes to the log that the transaction is committed.
4. The CHECKPOINT process writes the portion data file associated with the transaction to disk. This might happen anywhere from seconds to minutes after the step above.
5. It writes to the log that the transaction is "hardened".

The simplest type of backup is the Full Backup. The screen shots below are from SQL Server 2005's Management Studio (SSMS).

At a minimum you need to verify three things on this screen. First, that the correct database is selected. Second, the backup type is set to FULL. Finally you need to choose the backup file name. On the Options tab you can specify whether SQL Server should replace or append the backup to the backup file. Keep in mind that the backup file is relative to where SQL Server is installed and not where you're running SSMS.





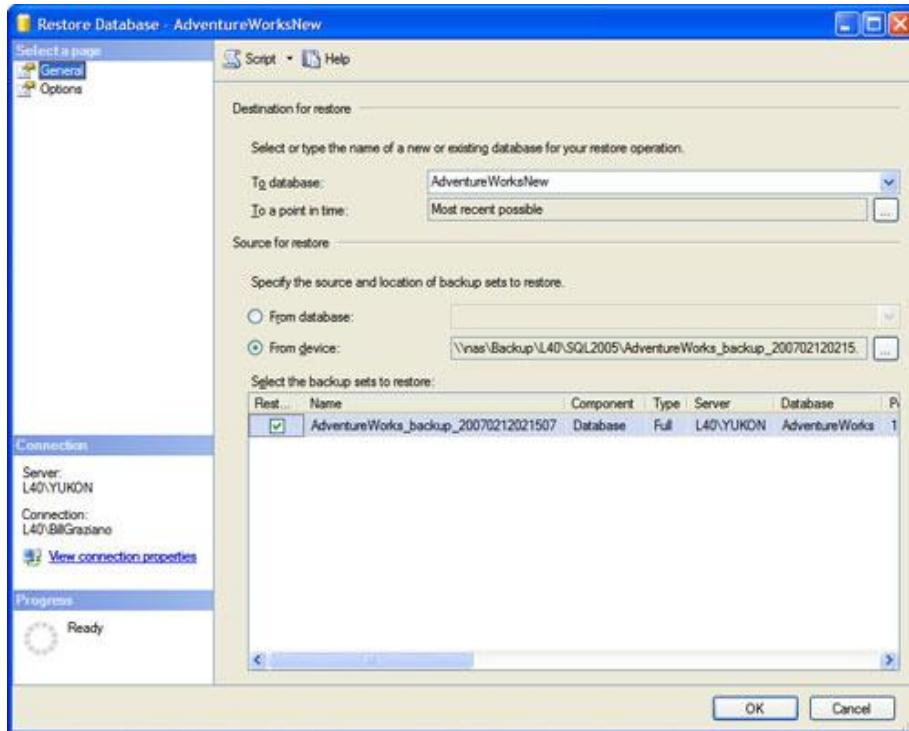
If you want to issue a backup statement yourself you can use SSMS to script it out for you. Click the Script button at the top of the dialog box and SSMS will generate this SQL statement for you:

```
BACKUP DATABASE [Adventureworks] TO
    DISK =
N'\\nas\\Backup\\l40\\sql2005\\Adventureworks_backup_200702120215.bak'
    WITH NOFORMAT, NOINIT, NAME = N'Adventureworks-Full Database
Backup',
    SKIP, NOREWIND, NOUNLOAD, STATS = 10
```

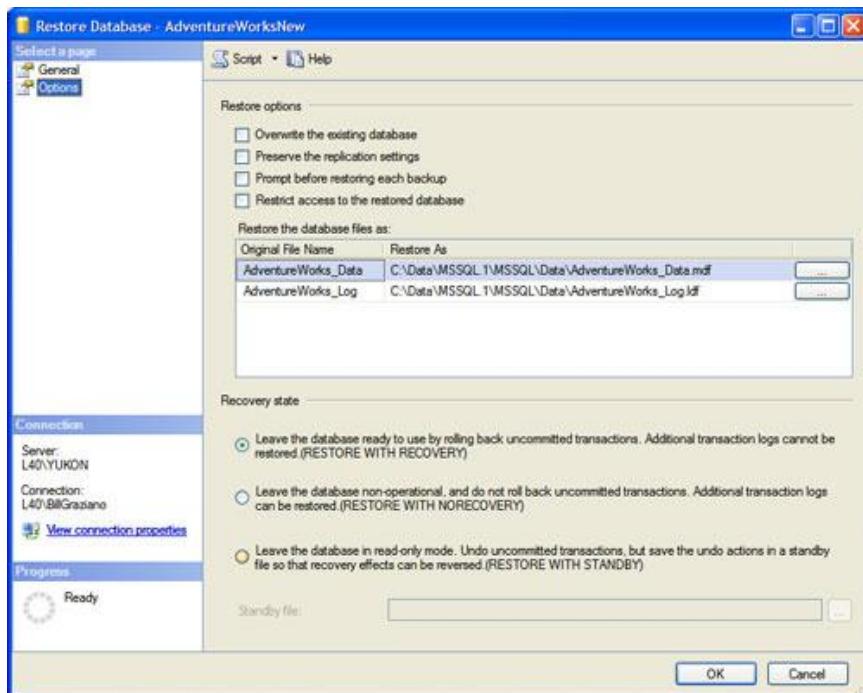
You can see how these options map back to the dialog box. The NOINIT clause is what says to append the backup to the existing backup file. The other option is INIT which will overwrite the backup file. The BACKUP statement will create a single file with a BAK extension that contains what is in your data file and log file. You can backup the database while SQL Server is running and people can still use the database. It might be a little bit slower depending on your disk throughput.

Restoring a database is a little more complicated. Right-clicking on Databases in SSMS brings up a dialog box like this:





I've already changed the database name to AdventureWorksNew. I clicked the From Device radio button and navigated to my backup file. If you're restoring on the same computer where the original database resides you can just leave the From Database radio button selected and choose the database. It will automatically select the backup. Clicking on the options tab brings us to the second part of the dialog:



Notice that it wants to restore the two file names right on top of the file names for AdventureWorks. SQL Server won't actually let you do that unless you check the "Overwrite the existing database" checkbox above. You'll need to edit those filenames to change the name. If I script this statement out it gives me this:

```
RESTORE DATABASE [AdventureworksNew]
    FROM  DISK =
N'\\nas\Backup\L40\SQL2005\Adventureworks_backup_200702120215.bak'
    WITH  FILE = 1,
        MOVE N'Adventureworks_Data' TO
N'C:\Data\MSSQL.1\MSSQL\Data\AdventureworksNew_Data.mdf',
        MOVE N'Adventureworks_Log' TO
N'C:\Data\MSSQL.1\MSSQL\Data\AdventureworksNew_Log.ldf',
    NOUNLOAD,  STATS = 10
```

Notice the MOVE commands have the new file name that I typed in.

One thing to be aware of is the SQL Server Recovery Model. If you right-click on a database and choose Properties and then click the Options tab you'll see the recovery model as the second item listed. The two main settings for this are Simple and Full. In Simple Recovery SQL Server doesn't keep transactions in the transaction log that have already been "hardened" to disk. They are automatically removed and the space in the file is reused. In Full Recovery mode SQL Server keeps every transaction in the transaction log file until you explicitly backup the transaction log. Simple Recovery mode is better for developers or servers that are only backed up nightly. In Full Recovery mode you'll need to do transaction log backups which I'll cover in a future article. If you see your database growing larger and larger the most likely cause is a growing transaction log. To resolve this, change the recovery model to Simple, backup the database and then shrink the database. You can shrink the database by right-clicking on the database and choosing Tasks -> Shrink -> Database and then clicking OK.

When you create a database, SQL Server starts with a copy of the "model" database. If you set the Recovery Model of the "model" database to Simple all future databases will start out in Simple Recovery mode.

## **2. Copy Only Backup for SQL Server 2005 and SQL Server 2008**

### **Problem**

I have implemented a backup plan for my SQL Server databases. I take a daily full backup supported with hourly differential backups. A restore plan is documented based on this backup plan along with the location and time of the backups mentioned in the plan. Often I am required to update the test or development server with a recent copy of the production database. For this purpose I take a full backup of the required database. The problem is that such ad hoc backups interrupt my planned



recovery sequence in case of a needed recovery. Is there any way that my ad hoc backups will not interrupt the sequence of my backup plan?

### **Solution**

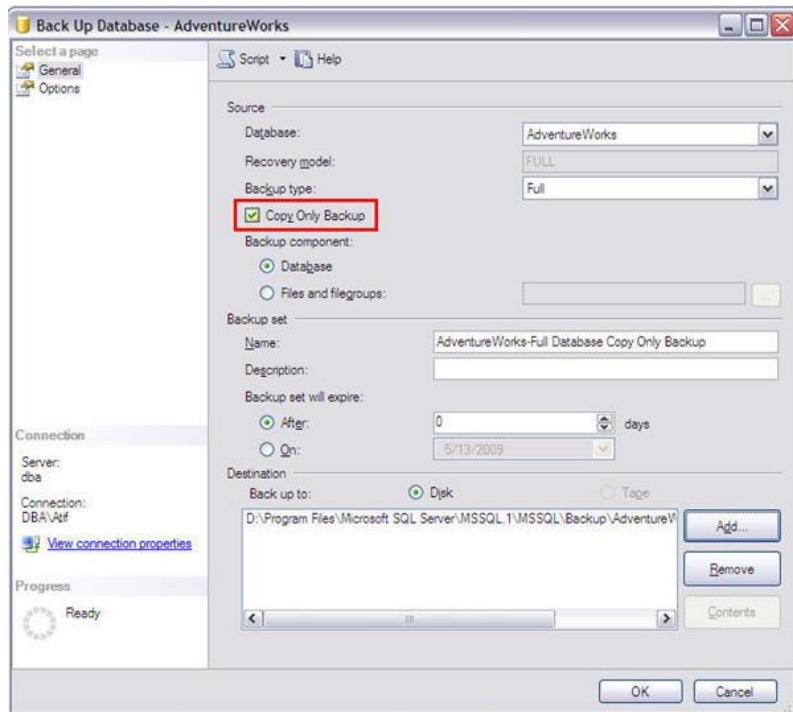
Fortunately in SQL Server 2005 and onwards we have a backup option for such a situation. This backup option is known as Copy Only backups. This option is specifically for creating an ad hoc backup that will not disturb the planned restore sequence for that database.

Copy Only backups can be used for creating a full backup or a transaction log backup. This option is not implemented for differential backups. In practical scenarios you will rarely need to create a Copy Only log backup, however the copy only option may be frequently used with full backups.

Although the Copy Only option is available for SQL Server 2005 there is not a way to create them using SSMS in SQL 2005. If you have SSMS 2008 you can use the GUI or you have to use a T-SQL statement to create Copy Only backups.

First let's look at how this can be done using SSMS 2008 to create Copy Only backup

- Go to the backup database window as you would for a normal backup



- Just below the "Backup type" menu, you will find a check box for "Copy Only Backup"
- Click this check box
- Fill out all other related information as you would for a normal backup and click OK



Note: The check box for "Copy Only Backup" will also be active for differential backups, but it will have no affect other than creating a normal differential backup.

Second, to create a Copy Only backup with T-SQL you can issue the following command:

```
-- Create full backup with Copy Only option
BACKUP DATABASE AdventureWorks
TO DISK = 'D:\WithoutCopyOnly_AdventureWorks.bak'
WITH COPY_ONLY
GO
```

Before going further it will be good to summarize the concept that a full backup with the copy only option is independent of the sequence of your other normal backups. So after you create a backup with the copy only option, you would be able to work with the recovery plan based on your scheduled backups without this impacting your restore process.

We will use LSN (log sequence number) information to track the full backup that is the base for the differential backups. First we will note the LSN for the differential base of the AdventureWorks database.

#### Script # 1: Note current differential base LSN

```
SELECT DB_NAME(database_id) AS [DB Name],
differential_base_lsn AS 'Note differential base LSN'
FROM sys.master_files
WHERE database_id = DB_ID('AdventureWorks')
AND type_desc = 'ROWS'
GO
```

The differential\_base\_lsn affects the sequence in which combination of backups are to be restored in a recovery. You may also get detailed information about the LSN of any database from table msdb..backupset.

	DB Name	Note differential base LSN
1	AdventureWorks	4000000007500037

Now we have to confirm that a full backup without the Copy Only option will update the differential\_base\_lsn. For this purpose we will issue a full backup command without the Copy Only option and we will note the change in the LSN to prove that the updated differential base LSN is the LSN of our last full backup.



**Script # 2: Create full backup and compare LSN information**

```
-- Create full backup
-- Run script after changing Backup path
BACKUP DATABASE AdventureWorks
TO DISK = 'D:\WithoutCopyOnly_AdventureWorks.bak'
GO

-- Get differential_base_lsn after full backup
SELECT DB_NAME(database_id) AS [DB Name],
differential_base_lsn AS 'Updated differential base LSN'
FROM sys.master_files
WHERE database_id = DB_ID('AdventureWorks')
AND type_desc = 'ROWS'
GO

-- Get LSN of recent full backup for match purpose
SELECT database_name, backup_start_date, is_copy_only,
first_lsn as 'LSN of full backup'
FROM msdb..backupset
WHERE database_name = 'AdventureWorks'
ORDER BY backup_start_date DESC
GO
```

In the following result set, we can verify that LSN has been changed for the differential backup restore base and matches our full backup LSN. Both marked LSNs below are the same which confirms that the last backup is our differential base.

	DB Name	Updated differential base LSN
1	AdventureWorks	40000000078800037

	database_name	backup_start_date	is_copy_only	LSN of full backup
1	AdventureWorks	2008-05-11 15:52:32.000	0	40000000078800037
2	AdventureWorks	2008-05-11 15:33:43.000	1	40000000077100034
3	AdventureWorks	2008-05-11 15:30:16.000	0	40000000075000037
4	AdventureWorks	2008-05-11 15:15:36.000	1	40000000073300034
5	AdventureWorks	2008-05-11 15:15:08.000	0	40000000071200037
6	AdventureWorks	2008-05-11 15:00:41.000	1	40000000069500034

Now we will create a full backup with the Copy Only option and it will be proved that the full backup with the Copy Only option will not affect the differential base LSN of our database. In other words the full backup with the Copy Only option will not affect the base full backup for the differential backups.



**Script # 2: Create full backup with copy only option and compare LSN information**

```
-- Create full backup with copy only option
-- Run script after changing Backup path
BACKUP DATABASE AdventureWorks
TO DISK = 'D:\CopyOnly_AdventureWorks.bak'
WITH COPY_ONLY
GO

-- Get differential_base_lsn after full backup with copy only option
SELECT DB_NAME(database_id) AS [DB Name],
differential_base_lsn AS 'Un changed differential base LSN'
FROM sys.master_files
WHERE database_id = DB_ID('AdventureWorks')
AND type_desc = 'ROWS'
GO

-- Get LSN of recent full backup with copy only option for match purpose
SELECT database_name, backup_start_date, is_copy_only,
first_lsn as 'LSN of last full backup'
FROM msdb..backupset
WHERE database_name = 'AdventureWorks'
ORDER BY backup_start_date DESC
GO
```

In the following image we can verify that after a full backup with the Copy Only option the differential base LSN is unchanged and it matches the LSN of the previous full backup (both are marked red). Also note that the last full backup with the Copy Only option is also there (marked green).

	DB Name	Un changed differential base LSN
1	AdventureWorks	40000000078800037

	database_name	backup_start_date	is_copy_only	LSN of last full backup
1	AdventureWorks	2009-05-14 15:56:23.000	1	40000000080900034
2	AdventureWorks	2009-05-14 15:52:32.000	0	40000000078800037
3	AdventureWorks	2009-05-14 15:33:43.000	1	40000000077100034
4	AdventureWorks	2009-05-14 15:30:16.000	0	40000000075000037
5	AdventureWorks	2009-05-14 15:15:36.000	1	40000000073300034

Following considerations will be helpful while using backups with the Copy Only option.

- Copy Only option will also work for compatibility level 80 databases in a SQL Server 2005 instance
- Transaction log backups with the Copy Only option preserves the existing log archive point, hence it will not truncate the transaction logs of that database.



- There is no enhanced consideration required while restoring a backup created with the Copy Only option.
- A full backup with Copy Only option can not be used as a base for restoring differential backups
- A log backup with Copy Only option may be created for databases with recovery model full or bulk logged only.
- A full backup with Copy Only option may be created for databases with any recovery model.

### 3. How to restore a Suspect database:

If you find your Database in Suspect mode then please keep your nerve strong. Just proceed step by step what I am written bellow. I think you will get out of this trouble. SQL Server 2008 introduced a new DB Status called Emergency. This mode can change the DB from Suspect mode to Emergency mode, so that you can retrieve the data in read only mode. The steps are... After executing the script given below you will get back your Database in operational mode.

```
Exec Sp_resetstatus 'Yourdbname'
```

```
Alter database yourdbname set emergency
```

```
Dbcc checkdb(yourdbname)
```

```
Alter database yourdbname set single_user with rollback immediate
```

```
Dbcc checkdb ('yourdbname ', repair _allow_data_loss)
```

```
ALTER DATABASE yourDBname SET MULTI_USER
```

### 4. Backup and restore of the Resource database:

The [Resource database](#) (shortly referred to as **RDB**) is a hidden, read-only database that contains all the system objects that are included with SQL Server 2005. This is the reason why it does not appear in SQL Server Management Studio. It contains all the system objects that ship with SQL Server 2005. These objects physically exist in the Resource database but logically appear in the **sys** schema of every database on the instance. It complements the **master** database in a sense as the SQL Server service now also depends on this database. The **Resource** database makes it easy for service packs to be applied or rolled back whenever necessary. In SQL Server 2000, whenever a service pack is applied, all the system objects that reside on both system and user databases will be updated, making it more difficult to rollback the change whenever necessary. It is also the reason why Microsoft recommends that you backup all the system and user databases before applying a service pack.

In SQL Server 2005, changes will only be made to the this database and will be reflected on all the system and user databases on the instance. If you need to apply a service pack on multiple instances, all you need to do is copy the Resource database's MDF and LDF files to the target instances. Rolling back the changes is as simple as overwriting the database files with an older copy. The physical file names of the **Resource** database are **mssqlsystemresource.mdf** and



**mssqlsystemresource.ldf** and are located, by default, in **<drive>:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data\**

### **Why is it important?**

The **Resource** database appears to be a critical system database as the SQL Server service is now dependent on this. You can verify by renaming the database files while the service is stopped. You will not be able to start the service after this. You can also try moving the **master** database on a different location without moving the **Resource** database together with it and you will not be able to start the service. Its location is dependent on the **master** database. This is critical during a disaster recovery process as we have gotten used to dealing with only the **master** database in previous versions.

### **Backing up the Resource database**

Since the **Resource** database is not available from the SQL Server tools, we cannot perform a backup similar to how we do it with the other databases. You can backup the database using the following options:

1. You can use a simple **xcopy** command to copy from the source location to a destination where you keep your daily database backups. Use the **-Y** option to suppress the prompt to confirm if you want to overwrite the file. You can create a scheduled task to do this on a daily basis. If you want to keep multiple copies of the database files, you can create an automated script to rename them after the copy process.

```
xcopy <drive>:\Program Files\Microsoft SQL  
Server\MSSQL.1\MSSQL\Data\mssqlsystemresource.mdf <destination folder>  
/Y  
  
xcopy <drive>:\Program Files\Microsoft SQL  
Server\MSSQL.1\MSSQL\Data\mssqlsystemresource.ldf <destination folder>  
/Y
```

2. You can use your file-based backup utilities such as NTBackup, IBM Tivoli Storage Manager, Symantec BackupExec, etc.

### **Restoring the Resource database**

It is important to document the location of your **master** database as part of your disaster recovery process. In previous versions of SQL Server, all we need to do to restore the server instance is to worry about the **master** database.

After a SQL Server 2005 instance has been rebuilt a restore of the **master** database will be done, the **Resource** database files should go along with it should a WITH MOVE option be required. This means that if the old location of the **master** database will be different from the one after the restore, the **Resource** database files should already be there prior to restoring the **master** database. This is very critical if a hardware failure occurred and you need to move the system databases on a different drive during the server instance rebuild.

To restore the **Resource** database, just copy the database files to the location of the **master** database files. If you have an older version of the **Resource** database, it is

important to re-apply any subsequent updates. This is why the recommended approach is to simply do a daily backup of these files.

## **Best Practices on High Availability, Backup & Restore**

### **General High Availability:**

1. Physically protect your SQL Servers from unauthorized users.
2. Physically document all of your SQL Server instances. Incorporate effective change management.
3. Always use a RAIDed array or SAN for storing your data.
4. Use SQL Server clustering, database mirroring, or log shipping to provide extra fault tolerance.
5. Replication is not an effective means to protect your data.
6. Always use server-class hardware, and standardize on the same hardware as much as possible.
7. Use hardware and software monitoring tools so you can quickly become aware of when problems first arise.
8. After testing, apply all new service packs and hot fixes to the OS and SQL Server.
9. Cross-train staff so that there are multiple people who are able to deal with virtually any problem or issue.

### **Disaster Recovery**

1. You must create a disaster recovery plan and include every detail you will need to rebuild your servers.
2. As your SQL Servers change over time, don't forget to update your disaster recovery plan.
3. Write the disaster recovery plan so that any computer literate person will be able to read and follow it. Do not assume a DBA will be rebuilding the servers.
4. Fully test your disaster recovery plan at least once a year.

### **Backup**

1. All production databases should be set to use the full recovery model. This way, you can create transaction log backups on a periodic basis.
2. Whenever possible, perform a daily full backup of all system and user databases.
3. For all production databases, perform regular transaction log backups, at least once an hour.
4. Perform full backups during periods of low user activity in order to minimize the impact of backups on users.
5. Periodically test backups to ensure that they are good and can be restored.
6. Backup first to disk, then move to tape or some other form of backup media.
7. Store backups offsite.
8. If using SQL Server 2005 encryption, be sure to backup the service master key, database master keys, and certificates.
9. If you find that backup times take longer than your backup window, or if backup file sizes are taking up too much space on your storage device, consider a third party backup program, such as SQL Backup Pro.
10. Document, step-by-step, the process to restore system and user databases onto the same, or a different server. You don't want to be looking this information up during an emergency.



## Log- Shipping

### What is Log Shipping

Log Shipping is one of the methods for creating a Standby server, by god forbid if something happens to our production server we need a standby server and Microsoft has come up with this idea and introduced Log Shipping in SQL 2000 itself. But in SQL 2005 it has been enhanced further by making it more user friendly. It is mainly used in OLTP environment (Online Transaction Processing). It is used as a High Availability Solution and also in Disaster recovery situations.

### Overview of Log Shipping

Let's discuss the overview of log shipping. The basic idea is nothing but backup and restore of the database and transaction logs sequentially from the primary server to secondary or standby server. We might be having some extremely critical databases in our production environment and need them to be online 24\*7. If the production server is down due to some natural disasters or in some cases we need to reboot the server after applying some service packs or in case of some hardware upgradation i.e adding extra hard disks, all those require some downtime of the production server in that case we can make use of log shipping.

### Pre-Requisites

The following are the prerequisites to configure Log Shipping SQL 2008

1. SQL Server 2008 Enterprise Edition, SQL Server 2008 Workgroup Edition, or SQL Server 2008 Standard Edition must be installed on all server instances involved in log shipping.
2. The servers involved in log shipping should have the same case sensitivity settings.
3. Enable Remote connections to the server in SQL Server surface area configuration, by navigating to start menu--->All Programs--->Microsoft SQL Server 2008--->Configuration tools--->Surface Area Configuration tool--->Select Surface Area configuration for Services and Connections option--->Select Remote Connections--->Select Local and Remote connections by using TCP and Named Pipes option or else you can also use local and remote connections using TCP\IP only and click OK.
4. The SQL Services in both the primary and secondary server should be the same with same password. Preferably --a domain account.
5. The DB to be log Shipped should be in Full Recovery or Bulk logged recovery model, so that T-logs can be applied, else you cannot configure log shipping. Use the below command to change the recovery model or else you can change in the SSMS by right clicking the DB properties.
6. Shared folder should be created in Primary server to hold the tran log backups.



\* Read/Write permission required --->For SQL service account of Primary for the tran log backup to be successful

\* Read/Write permission required --->For SQL Agent account of Secondary for the copy job to be successful

\* Read/Write permission required --->For SQL Service Account of Secondary for the restore job to be successful

Configuration:

I have used 3 instances to configure log shipping and they are as follows,

Deepak--->Primary

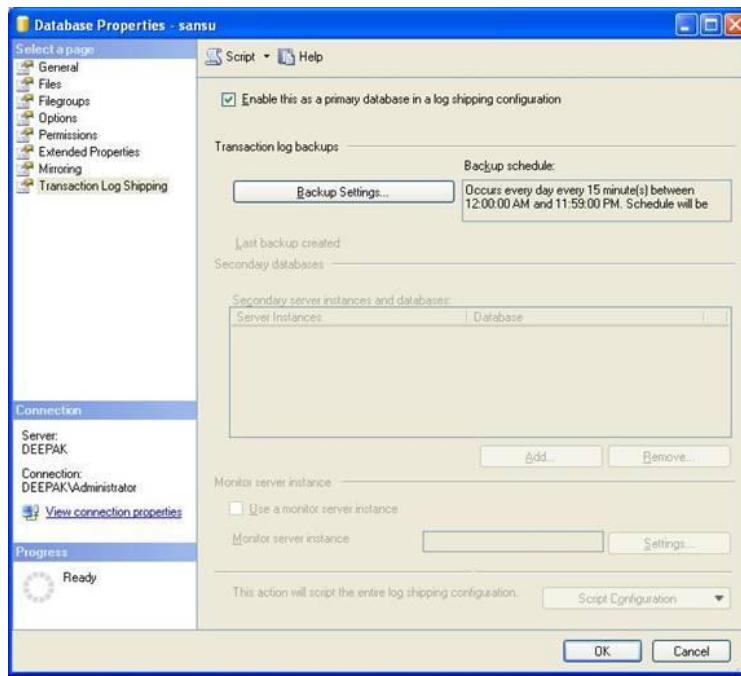
Deepak\Sansu--->Secondary

Deepak\Test--->Monitor

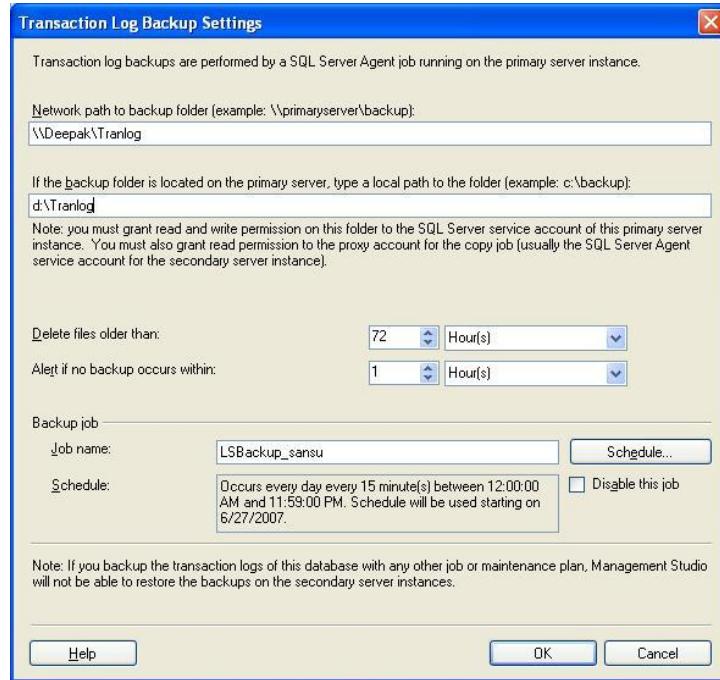
The database for which I am configuring Log shipping is named as "sansu" and it is present in primary server.The following are the steps I followed,

**Step 1:** Take a full backup of the DB to be log shipped in primary server (i.e sansu) and restore it in secondary server *WITH STANDBY* option

**Step 2:** Right Click the DB to be log shipped in primary server, select Task and Ship Transaction logs option and ensure that you enable the option "*enable this as a primary database in log shipping*"



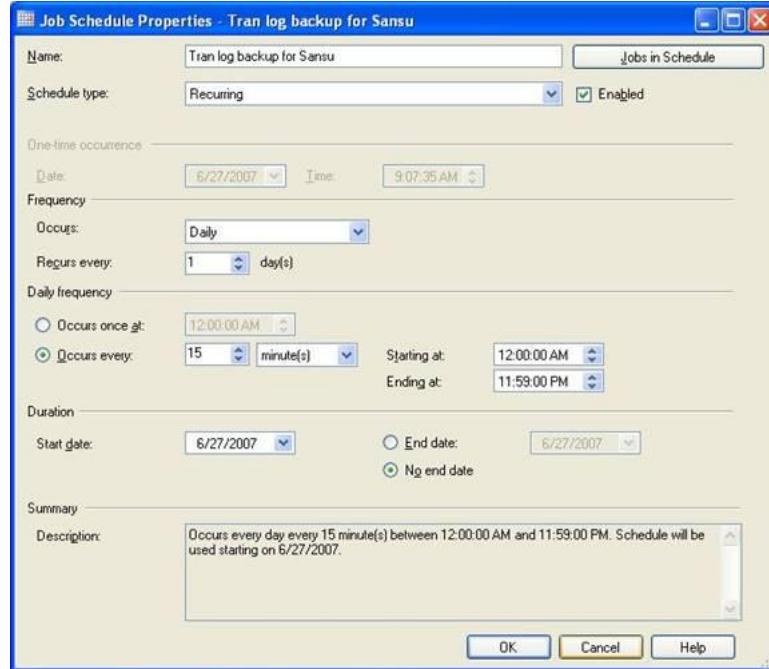
**Step 3 :** Click Backup settings and give the primary server name\\tran log older(shared folder) and also mention the local hard drive in the primary server where the shared folder resides in the subsequent box



**Step 4 :** There are options "*Delete files older than*" it helps to remove the old transaction log backup files from the shared folder in the primary and "*Alert if no backup occurs within*" it sends an alert if a tran log backup has not happened for a stipulated time mentioned there.

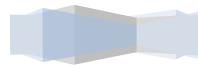
**Step 5 :** Click the schedule option for the tran log backup and schedule as per your desire

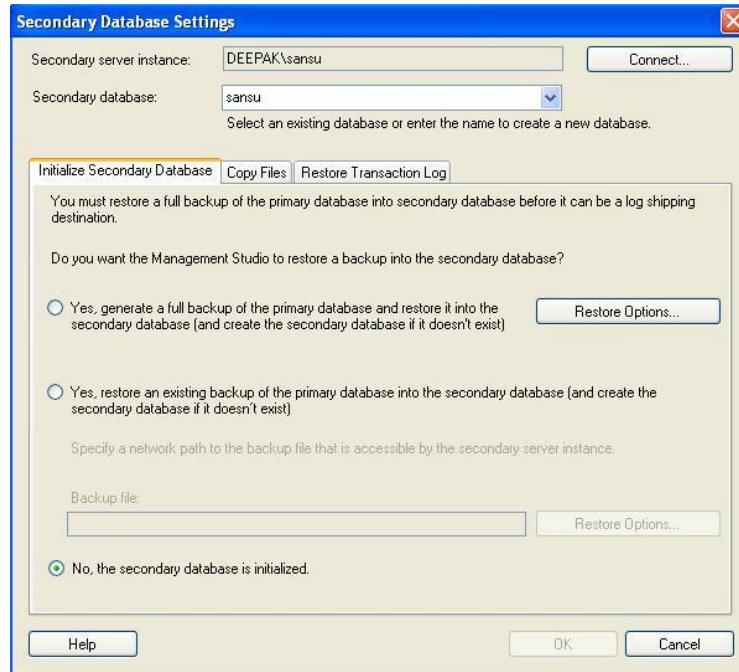




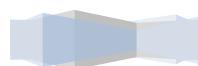
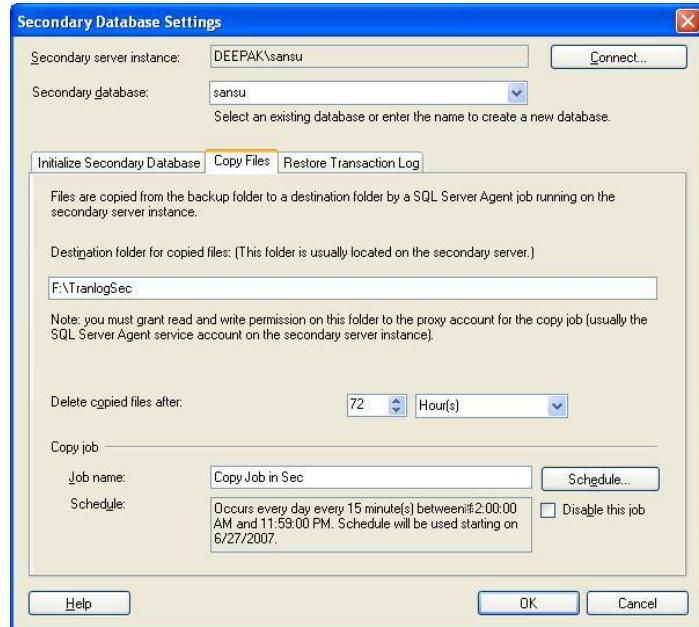
**Step 6 :**As mentioned in Step 2,below the enable option select "Add" option to add the secondary server and connect to it via the connect option.

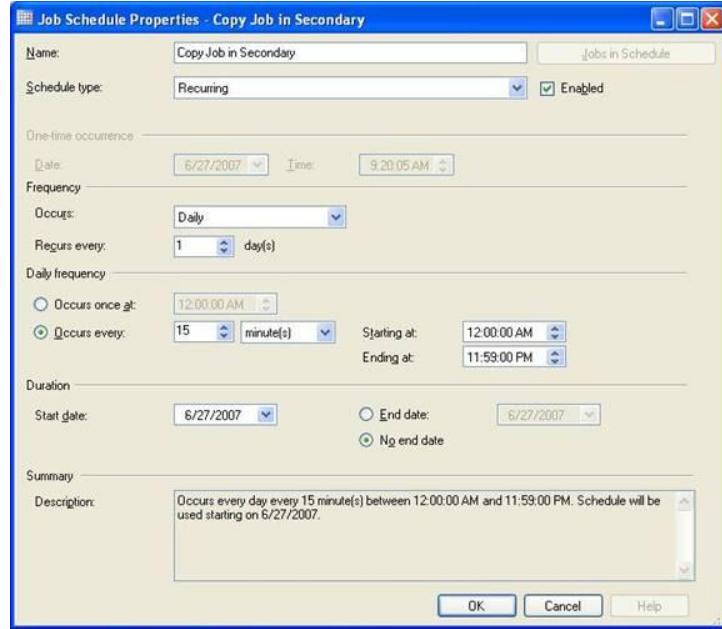
**Step 7 :**In the "*Initialize secondary database*" option choose the 3rd option "No, secondary database is initialized" because we have manually taken a full backup and restored as in Step1.If we havent done Step1,we can choose options 1 which will take a fresh full backup and restore it in secondary server or option 2 which will restore from an existing backup rather than a fresh backup.In that case you need to point the location where the backup file resides and also the name of the data and log files in the restore options.



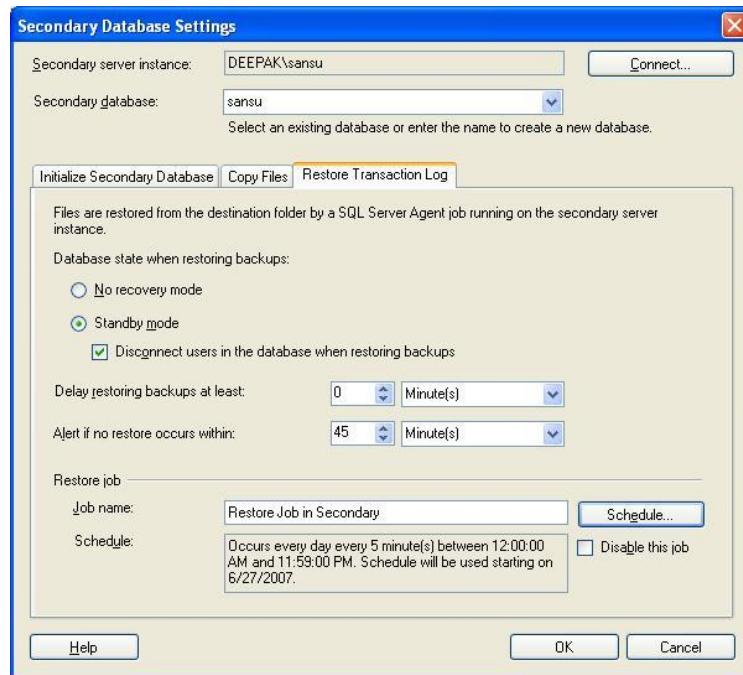


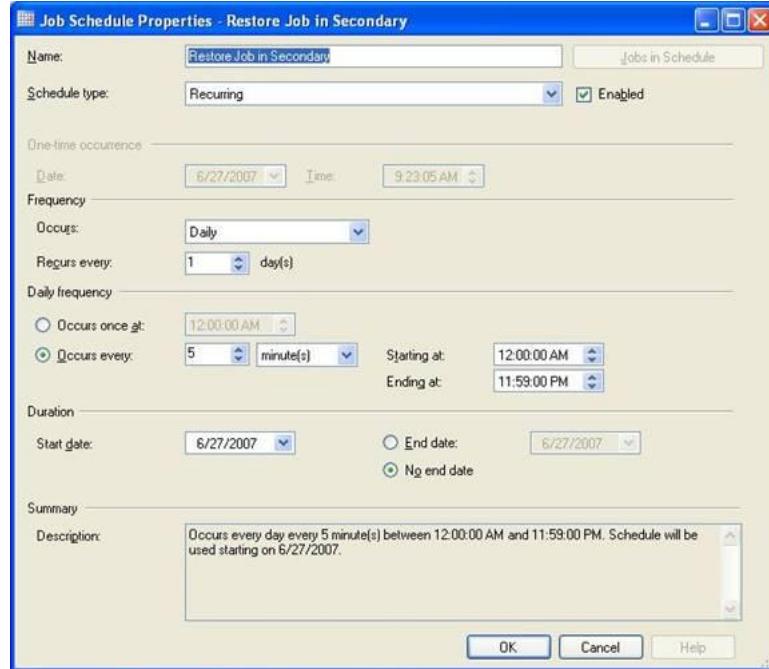
**Step 8 :** Choose the Copy files tab and specify the path where the tran log is copied and put in the secondary and schedule the job and fill the delete old files option.



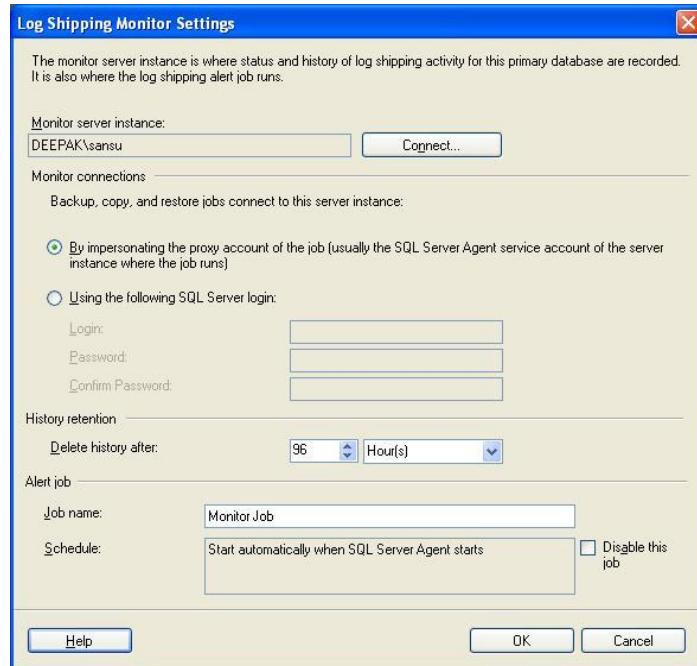


**Step 9 :** In the Restore transaction log tab choose STANDBY mode and "disconnect users in db when restoring backup", because if the users are accessing the db restoration will fail. In "Delay Restoring Backups" if it is 0 minute(s) as soon as copy job completes, restore job will start immediately without any delay. In "Alert if no restore occurs within" option if restore does not happen for a stipulated amount of time mentioned here an alert is issued. Also schedule the restore job as per your desire.





**Step 10 :** Choose the page as in Step2 and enable the "Use a monitor server instance" and connect to a new sql instance(Deepak\Test)which will monitor the log shippings backup,copy and restore jobs and will help in troubleshooting. We can also configure the secondary server instance(Deepak\sansu) as monitor.Both are correct it all depends on your requirements.Click Ok and the log shipping is configured with all 3 jobs and status will be shown as success.



If you see the below screenshot which shows the secondary db sansu which is in read only mode, further 3 jobs will created 1 in the primary server which is the

backup job and other 2 in secondary server which is the copy and restore job. In the monitor server an alert job will be created to generate alerts if the copy or restore job fails for a stipulated amount of time. You can view the jobs created by navigating to SQL server agent and then beneath it you have a folder called Jobs, which displays the jobs present in that server.

## **Failover in SQL 2008 Log Shipping**

The most important aspect in Log Shipping is Failover. Lets discuss it detail ! ! ! If the primary server in Log Shipping becomes unavailable or if it is going to be down due to some manual intervention, DBA should immediately perform the following steps for failover.

**Step 1:** Try to backup the Tail end of the transaction log in primary server with NORECOVERY option i.e perform a tran log backup if the primary is still accessible.

Backup log DBName to disk = "Local path or Network path" with NORECOVERY ---> A

Else execute the below T-SQL in secondary server to bring the secondary online,

Restore database DBName with Recovery ---> C

**Step 2:** If you were able to perform ---> A in step 1 then proceed with ---> B in step 2 to bring the secondary db online from read-only state. If you were able to perform only ---> C in step1 then go to step 3

Restore log DBName from disk = "Local path or Network path" with RECOVERY ---> B

**Step 3:** The syslogins and sysusers table in primary and secondary server should be in sync otherwise the DB users and other application users from primary will not be able to login into SQL server or into the DB in secondary server after failover occurs.

Two ways are there to accomplish the above task namely

\* Create the highly critical application users logins in the secondary server similar to primary just before configuring log shipping. Use the below sps to resolve orphaned users

```
USE
GO
sp_change_users_login @Action='update_one', @UserNamePattern='',
@LoginName=
GO
```

### **Script to find orphahend users**

```
exec sp_change_users_login 'Report'
```

```
Use Master
```



```
SET NOCOUNT ON

        SELECT 'EXEC sp_addlogin @loginame = "' + loginname + """
        , @defdb = "' + dbname + """
        , @deflanguage = "' + language + """
        , @encryptopt = "skip_encryption"""
        , @passwd =' 
        , cast(password AS varbinary(256))
        , @sid =' 
        , sid
FROM sys.syslogins
where loginname Not like 'NA%' and
      loginname not like 'Builtin%' and loginname Not like 'sa'
```

Run the above script on Source server copy the result and execute on Destination server

Eg:--

```
EXEC sp_addlogin @loginame = 'CorpCommUser' , @defdb = 'CorpComm' ,
@deflanguage = 'us_english' , @encryptopt = 'skip_encryption' , @passwd =
0x01003F04413C64CEE4767BA2DD0053A02C6056640C4C88C24DFA , @sid
= 0xCEE1766A76520E43A98DCB141B031F7E
```

**Step 4:** Also Disable the log shipping jobs in the primary and secondary servers,once failover occurs.

**Step 5:** Once the failover occurs the original secondary server is configured as primary and log shipping is again newly configured from this new primary server(original secondary) to original primary(now secondary).

**Step 6:** When you once again want to revert to the original state of log shipping i.e original primary was primary and original secondary was secondary, you need take a full backup in new primary server(original secondary) and restore it in original primary and reconfigure the log shipping from the original primary to original secondary.



## Frequently Raised Errors In Log-Shipping

**Question : IS it possible to log ship database between SQL 2000 & SQL 2008?**

**Answer:** No, that's impossible, In SQL 2008 transaction log architecture is changed compared to SQL 2000 and hence you won't be able to restore tlog backups from SQL 2000 to SQL 2008 or vice versa.

**Question:I'm getting the below error message in restoration job on secondary server, WHY?**

[Microsoft SQL-DMO (ODBC SQLState: 42000)]

Error 4305: [Microsoft][ODBC SQL Server Driver][SQL Server]The log in this backup set begins at LSN 7000000026200001, which is too late to apply to the database. An earlier log backup that includes LSN 6000000015100001 can be restored.

[Microsoft][ODBC SQL Server Driver][SQL Server]RESTORE LOG is terminating abnormally.

**Answer:** Was your sql server or agent restarted Y'day in either source or destination ? because the error states there is a mismatch in LSN. A particular tran log was not applied in the destination server hence the subsequent tran logs cannot be applied as a result !

You can check log shipping monitor \ log shipping tables to check the which transaction log is last applied to secondary db, if the next consecutive transaction logs are available in the secondary server share folder you manually RESTORE the logs with NORECOVERY option, Once you restored all the logs automatically from the next cycle the job will work fine.

Incase if you are not able to find the next transaction log in secondary server shared folder, you need to reconfigure log shipping. Try the below tasks to re-establish log shipping again.

- Disable all the log shipping jobs in source and destination servers
- Take a full backup in source and restore it in secondary server using the With Standby option
- Enable all the jobs you disabled previously in step1



**Question: Is it possible load balance in log shipping?**

**Answer:** Yes of course it's possible in log shipping, while configuring log shipping you have the option to choose standby or no recovery mode, and there you select STANDBY option to make the secondary database read-only.

**Question: Can I take full backup of the log shipped database in primary server??**

**Answer:** In SQL Server 2000 you won't be able to take full backup of log shipped database, because this will break the LSN chain and it directly affects the log shipping.

In SQL Server 2008, yes its possible. You can take full backup of log shipped database and this won't affect the log shipping.

**Question : Can I shrink log shipped database log file??**

**Answer:** Yes of course you can shrink the log file, but you shouldn't use WITH TRUNCATE option. If you use this option obviously log shipping will be disturbed.

**Question : Can I take full backup of the log shipped database in secondary server??**

**Answer:** No chance, you won't be able to execute BACKUP command against a log shipped database in secondary server.

**Question: I've configured Log shipping successfully on standby mode, but in the restoration job I'm getting the below error. What I do to avoid this in future??**

**Message**

2006-07-31 09:40:54.33 \*\*\* Error: Could not apply log backup file 'C:\Program Files\Microsoft SQL

Server\MSSQL.1\MSSQL\Backup\LogShip\TEST\_20060731131501.trn' to secondary database 'TEST'.(Microsoft.SqlServer.Management.LogShipping) \*\*\*

2006-07-31 09:40:54.33 \*\*\* Error: Exclusive access could not be obtained because the database is in use.

RESTORE LOG is terminating abnormally.(.Net SqlClient Data Provider) \*\*\*



**Answer:** To restore transaction logs to the secondary db, SQL Server needs exclusive access on the database. When you configure it in standby mode, users will be able to access the database and run query against the secondary db. Hence If the scheduled restore jobs runs at that time, the db will have a lock and it won't allow SQL Server to restore the tlogs. To avoid this you need to check "Disconnect users in the database when restoring backups" options in log shipping configuration wizard.

**Question : Suddenly I'm getting the error below, How can I rectify this???**

[Microsoft SQL-DMO (ODBC SQLState: 42000)] Error 4323: [Microsoft][ODBC SQL Server Driver][SQL Server]The database is marked suspect. Transaction logs cannot be restored. Use RESTORE DATABASE to recover the database.

[Microsoft][ODBC SQL Server Driver][SQL Server]RESTORE LOG is terminating abnormally

Answer : We had the same issue some time ago, this was related to a new file being created in a filegroup on the source. Don't know if this applies to your case, but restoring a backup of this new file on the secondary server solved the problem.

**Question : Is it possible to log ship database from SQL server 2005 to SQL server 2008 and vice versa?**

Answer : Yes you can log ship database from SQL server 2005 to SQL Server 2008 this will work. However log shipping from SQL Server 2008 to SQL Server 2005 is not possible because you won't be able to restore SQL server 2008 backup to SQL Server 2005 (downgrading version)

**Error message 14420 and error message 14421 that occur when you use log shipping:**

**Error message 14420** Error: 14420, Severity: 16, State: 1

The log shipping destination %s.%s is out of sync by %s minutes.

**Error message 14421** Error: 14421, Severity: 16, State: 1

The log shipping destination %s.%s is out of sync by %s minutes.

If you are using SQL Server 2008, the description for these error messages are different:



**Error message 14420** Error: 14420, Severity: 16, State: 1

The log shipping primary database %s.%s has backup threshold of %d minutes and has not performed a backup log operation for %d minutes. Check agent log and logshipping monitor information.

**Error message 14421** Error: 14421, Severity: 16, State: 1

The log shipping secondary database %s.%s has restore threshold of %d minutes and is out of sync. No restore was performed for %d minutes. Restored latency is %d minutes. Check agent log and logshipping monitor information.

Log shipping uses Sqlmaint.exe to back up and to restore databases. When SQL Server creates a transaction log backup as part of a log shipping setup, Sqlmaint.exe connects to the monitor server and updates the **log\_shipping\_primaries** table with the **last\_backup\_filename** information. Similarly, when you run a Copy or a Restore job on a secondary server, Sqlmaint.exe connects to the monitor server and updates the **log\_shipping\_secondaries** table.

As part of log shipping, alert messages 14220 and 14221 are generated to track backup and restoration activity. The alert messages are generated depending on the value of **Backup Alert** threshold and **Out of Sync Alert** threshold respectively.

The alert message 14220 indicates that the difference between current time and the time indicated by the **last\_backup\_filename** value in the **log\_shipping\_primaries** table on the monitor server is greater than value that is set for the **Backup Alert** threshold.

The alert message 14221 indicates that the difference between the time indicated by the **last\_backup\_filename** in the **log\_shipping\_primaries** table and the **last\_loaded\_filename** in the **log\_shipping\_secondaries** table is greater than the value set for the **Out of Sync Alert** threshold.

**Resolution:**

**Troubleshooting Error Message 14420**

By definition, message 14420 does not necessarily indicate a problem with log shipping. The message indicates that the difference between the last backed up file and current time on the monitor server is greater than the time that is set for the

200



### **Backup Alert** threshold.

There are several reasons why the alert message is generated. The following list includes some of these reasons:

1. The date or time (or both) on the monitor server is different from the date or time on the primary server. It is also possible that the system date or time was modified on the monitor or the primary server. This may also generate alert messages.
2. When the monitor server is offline and then back online, the fields in the **log\_shipping\_primaries** table are not updated with the current values before the alert message job runs.
3. The log shipping Copy job that is run on the primary server might not connect to the monitor server **msdb** database to update the fields in the **log\_shipping\_primaries** table. This may be the result of an authentication problem between the monitor server and the primary server.
4. You may have set an incorrect value for the **Backup Alert** threshold. Ideally, you must set this value to at least three times the frequency of the backup job. If you change the frequency of the backup job after log shipping is configured and functional, you must update the value of the **Backup Alert** threshold accordingly.
5. The backup job on the primary server is failing. In this case, check the job history for the backup job to see a reason for the failure.

### **Troubleshooting Error Message 14421**

By definition, message 14421 does not necessarily indicate a problem with Log Shipping. This message indicates that the difference between the last backed up file and last restored file is greater than the time selected for the **Out of Sync Alert** threshold.

There are several reasons why the alert message is raised. The following list includes some of these reasons:

1. The date or time (or both) on the primary server is modified such that the date or time on the primary server is significantly ahead between consecutive transaction log backups.



2. The log shipping Restore job that is running on the secondary server cannot connect to the monitor server **msdb** database to update the **log\_shipping\_secondaries** table with the correct value. This may be the result of an authentication problem between the secondary server and the monitor server.
3. You may have set an incorrect value for the **Out of Sync Alert** threshold. Ideally, you must set this value to at least three times the frequency of the slower of the Copy and Restore jobs. If the frequency of the Copy or Restore jobs is modified after log shipping is set up and functional, you must modify the value of the **Out of Sync Alert** threshold accordingly.
4. Problems either with the Backup job or Copy job are most likely to result in "out of sync" alert messages. If "out of sync" alert messages are raised and if there are no problems with the Backup or the Restore job, check the Copy job for potential problems. Additionally, network connectivity may cause the Copy job to fail.
5. It is also possible that the Restore job on the secondary server is failing. In this case, check the job history for the Restore job because it may indicate a reason for the failure.

## Best Practices of Log-Shipping

- If you don't currently employ clustering or database mirroring for your SQL Servers because of cost, consider employing log shipping to help boost your high availability. It provides reasonably high availability at low cost.
- If you take advantage of SQL Server 2000 or 2005 log shipping capability, you will want to keep the log shipping monitoring service on a SQL Server of its own, not on the source or destination
- Servers participating in log shipping. Not only is this important for fault tolerance, but because the log shipping monitoring service incurs overhead that can affect the performance of the source and destination servers.
- Monitor log shipping daily to ensure that it is working successfully.
- Learn what you need to know to fix shipping if synchronization is lost between the production and backup databases.
- Document, and test your server recovery plan, so you will be ready in case of a server failure.



### **Case Study: How to add files to a log-shipped database**

How to add a file to a log shipped database without reconfiguring log shipping.  
I did it in SQL 2008 Enterprise edition server to find out the information.

Steps:

1. I configured log shipping between the databases infisystem
2. Stopped and disabled all the backup, copy and restore jobs.
3. Added a secondary file named Infi\_data1.ndf to the log shipped infisystem database in primary server

The following is the script to add a new file to the infisystem database in primary server:

```
ALTER DATABASE infisystem  
ADD FILE  
(  
NAME = Infi_data1,  
FILENAME ='C:\Program Files\Microsoft SQL  
Server\MSSQL10.MSSQLSERVER\MSSQL\DATA\Infi_data1.ndf',  
SIZE = 5MB,  
MAXSIZE = 15,  
FILEGROWTH = 10%  
)
```

4. Manually took a transaction log backup for the database after adding the file
5. Manually copied the transaction log backup file to secondary server
6. Manually restored that particular transaction log backup using the WITH MOVE option and WITH NORECOVERY clause in secondary server

Basically while using the WITH MOVE option I would mention the newly created secondary file i had recently added to the log shipped database.

```
RESTORE filelistonly from disk='D:\Database\SQLDBA\Infisys.trn'  
RESTORE log Infisystem FROM Disk='D:\Database\SQLDBA\Infisys.trn'  
WITH MOVE 'Infi_data1' TO 'C:\Program Files\Microsoft SQL  
Server\MSSQL10.KATMAI\MSSQL\DATA\Infi_data1.ndf',  
norecovery
```

Enable all Jobs and see the log-shipping status in monitor server.



## Database Mirroring

### Overview of Mirroring:

Mirroring is mainly implemented for increasing the database availability. Similar to log shipping mirroring is also implemented on per database basis. Database mirroring maintains two copies of a single database that must reside on different instances of SQL Server Database Engine (server instances). Typically, these server instances reside on computers in different locations. One server instance serves the database to clients (the principal server), while the other server instance acts as a hot or warm standby server (the mirror server).

Mirroring provides a hybrid solution i.e

1. Provides a copy of the database like Log Shipping and
2. Rapid failover capabilities like Clustering

### Advantages of Mirroring :

\* Increases data protection ---> Depending on the mode of operation Mirroring provides minimal data loss.

\* Increases availability of a database ---> In the event of a disaster, in high-safety mode with automatic failover, failover quickly brings the standby copy of the database online (with no data loss). In the other operating modes, the database administrator has the alternative of forcing service (with possible data loss) to the standby copy of the database.

\* Improves the availability of the production database during upgrades ---> During service packs installation or any patch applied on Principal server which requires downtime, the standby comes into effect.

### Components in Mirroring:

Database mirroring consist of the following components

1. Principal ---> The Principal is the originating server i.e it is the source server which contains the database which is configured for mirroring. There can be only one principal database and it has to be in a separate SQL Server instance than the mirror database.

2. Mirror ---> The Mirror is the receiving database in a mirror pair i.e it is the destination server which contains the mirrored database. There can be only one mirror for each principal database. The mirror needs to be on its own separate SQL Server instance preferably on separate physical server.

3. Mirrored Pair ---> A Principal and Mirror operating together are called a Mirrored Pair. The changes on the principal are reflected in the mirrored database

204



4. Witness ---> A Witness is optional and it monitors the Mirrored Pair. It ensures that both principal and mirror are functioning properly. The Witness is also a separate SQL Server instance preferably on a separate physical server than principal and mirror. One Witness server can monitor multiple Mirrored Pairs.

5. Quorum ---> A Quorum is the relationship between the Witness, Principal and the Mirror.

6. Endpoint ---> Endpoint is the method by which SQL Server Database engine communicates with applications. In the context of Database mirroring endpoint is the method by which the Principal communicates with the Mirror. The mirror listens on a port defined in the endpoint. The default is 5022. Each database mirror pair listens on its own unique port.

To list all the database mirror endpoints run,

---> Select \* from sys.database\_mirroring\_endpoints

To list all the endpoints

---> Select \* from sys.tcp\_endpoints

Database Mirroring can be configured for three different operating modes:

**High Availability Operating Mode** - This provides durable, synchronous transfer of data between principal and mirror, including automatic failure detection and failover. There is performance overhead on this mode because a transaction is not considered committed until SQL Server has successfully committed it to the transaction log on both the principal and the mirror database. And as the distance between the principal and the mirror increases, the performance impact also increases. There is a continuous ping process between all three to detect failover. If the witness server is not visible from the mirror, you must either reconfigure the operating mode for the database mirroring session or turn off the witness.

Alternatively, you can manually fail over a database mirroring session at the mirror in High Availability Mode by issuing the following command at the principal. You can also issue the same command if you have to take principal down for maintenance.

ALTER DATABASE SET PARTNER FAILOVER

**High Performance Operating Mode** - In this configuration you dont need a WITNESS Server and the Mirror Server acts as an WARM standby and does not support automatic failure detection or failover. There is any asynchronous data transfer between principal and mirror. This mode provide better performance and you can have geographic dispersion between the principal and the mirror.

**High Protection Operating Mode (Recommended Mode)** - This mode is the same as High Availability Mode except failover is manual and you have to manually promote the mirror to be the principal. Data transfer is synchronous.



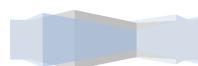
## Prerequisites for Database Mirroring

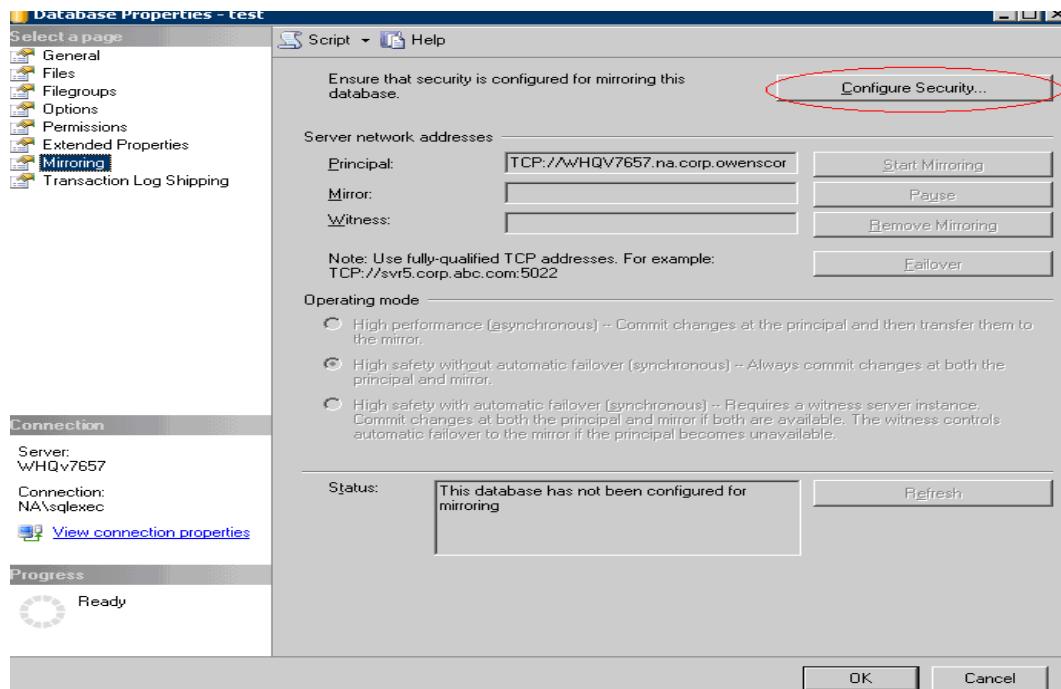
1. Make sure that the two partners that is the principal server and mirror server, are running the same edition of Microsoft SQL Server 2005. The partners require either **SQL Server 2005 Standard Edition or SQL Server 2005 Enterprise Edition or SQL Server 2005 Developer Edition**.
2. If you are using a witness, make sure that SQL Server 2005 is installed on its system. The witness can run on any reliable computer system that supports **SQL Server 2005 Standard Edition, Enterprise Edition, Workgroup Edition, or Express Edition**.
3. **SQL 2005 SP1 or later version** is required for Mirroring
4. The principal database must be in the **FULL recovery model**. Log records that result from bulk-logged operations cannot be sent to the mirror database.
5. Verify that the mirror server has **enough disk space** for the mirror database.
6. All of the server instances in a mirroring session should use the **same master code page and collation**. Differences can cause a problem during mirroring setup.
7. The mirror database must have the **same name** as the principal database.
8. The mirror database must be initialized from a restore of the principal database with **NORECOVERY**, followed by restores in sequence of principal transaction log backups. Prior to configuring mirroring ensure that **at least 1 tran log is restored** in addition to full backup with **NORECOVERY** mode.

First take full backup and one Transaction log backup from Principal server and restore it on Mirror server with **NO Recovery** Option.

Please follow the screen shots to setup Database mirroring.

1. Right Click on Database <AGTest> Go to Properties- Click on Mirroring.



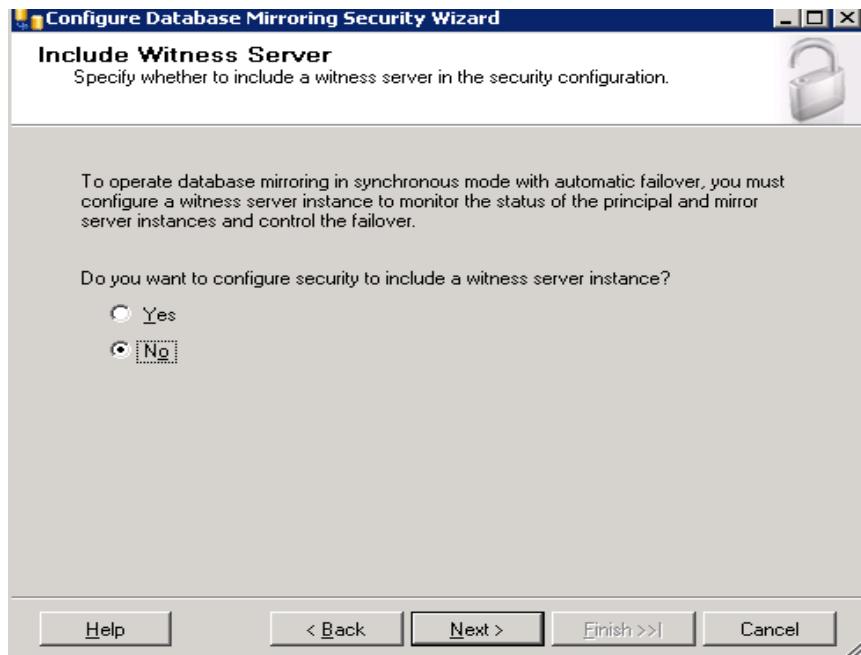


2. Click on **Configure Security** button.



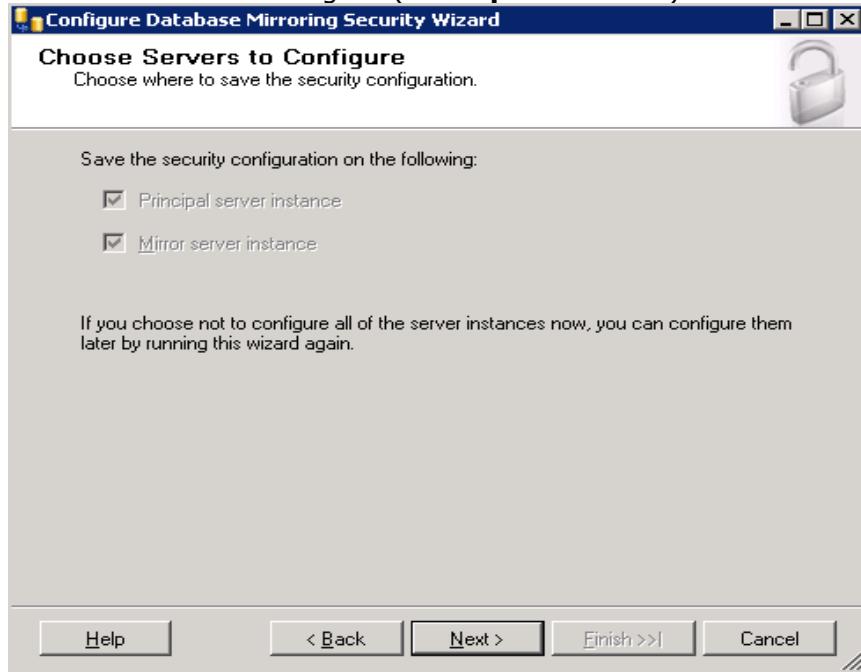
3. Click on **Next** Button





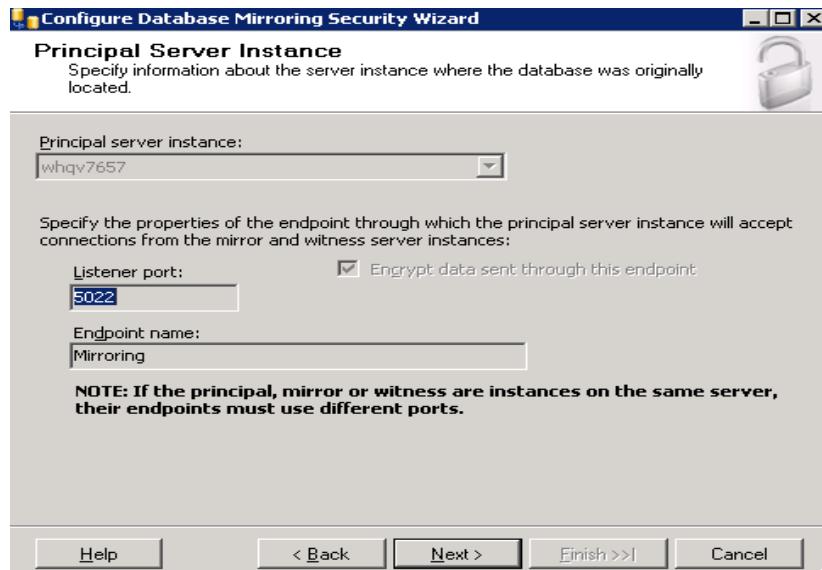
**Note:** If you wish to configure **Witness server** Click on **Yes** radio Button if not Click **No** button. This configuration is without **Witness box**.

4. Choose the server's to configure (**Principal & Mirror**) Click on Next Button

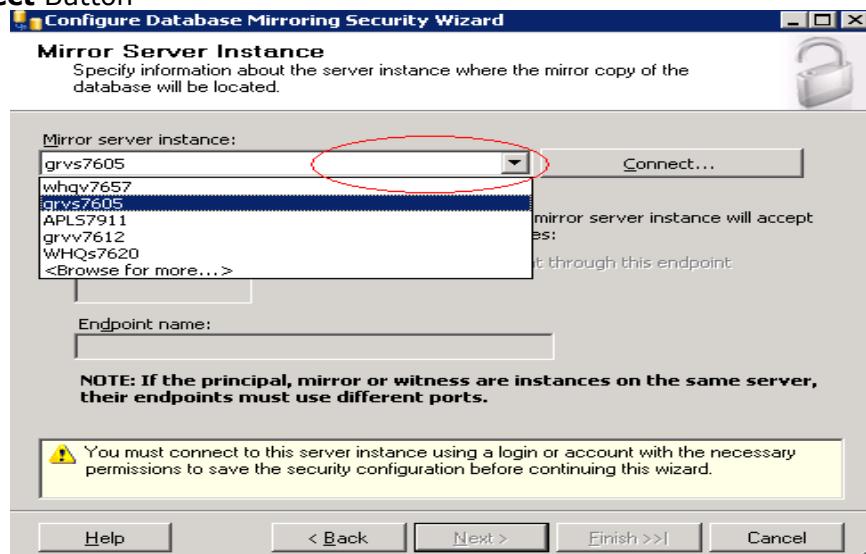


5. Set your Principal Server with Default settings.





6. Click on **Next** to Select **Mirror Server Instance** by using Pull down Menu and hit **Connect** Button

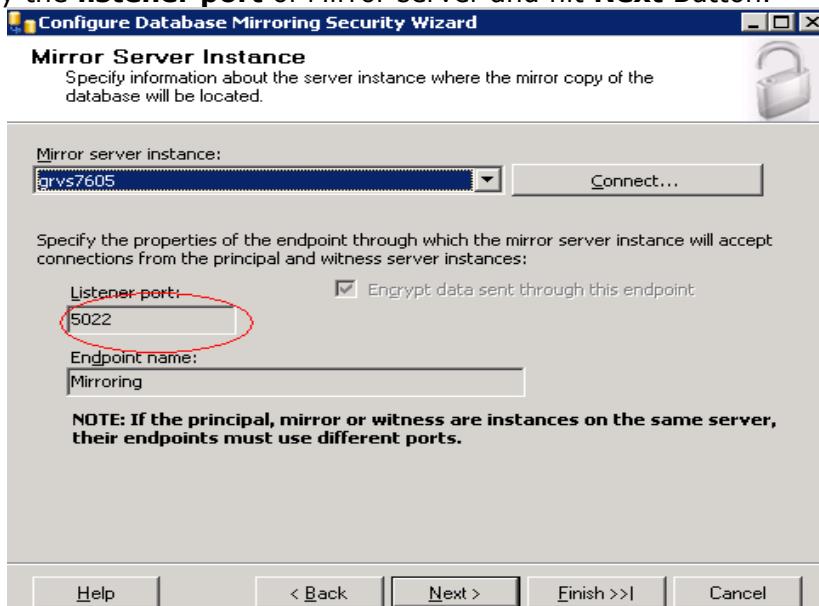


7. Verify **Server name** and **Authentication** and hit Connect



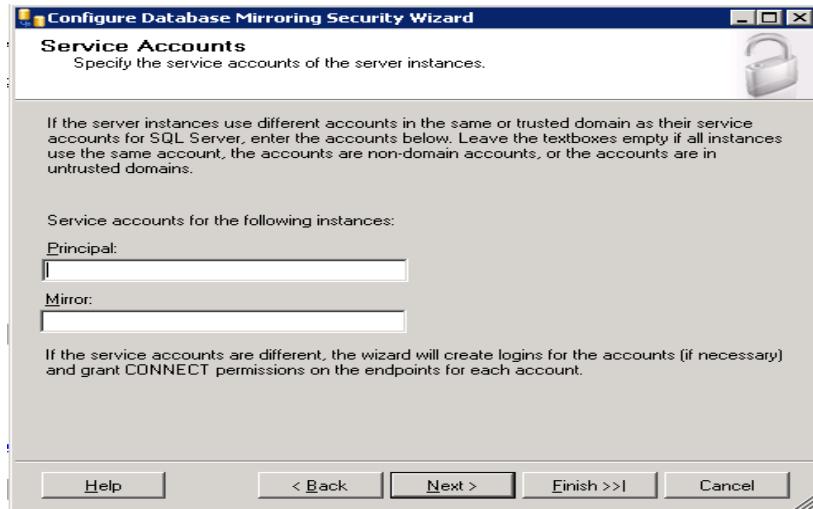


8. Verify the **listener port** of Mirror server and hit **Next** Button.



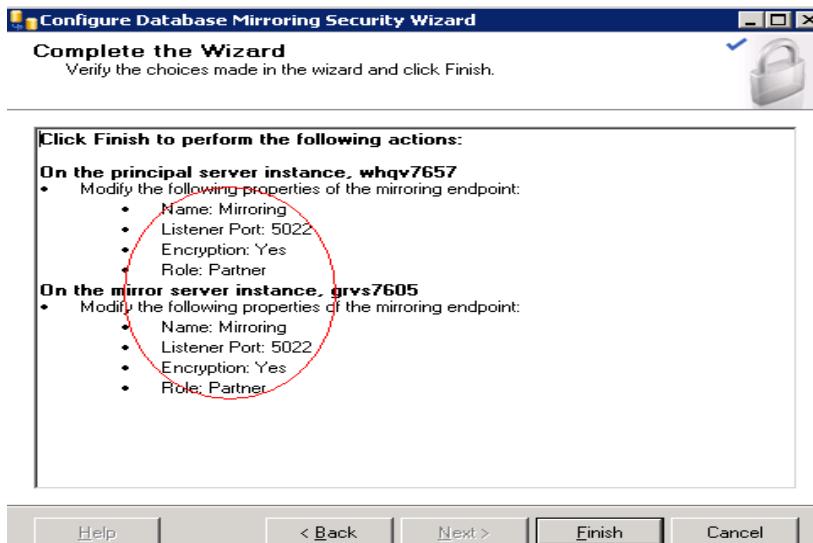
9. Service Accounts Screen Leave Both **Principal** and **Mirror Fields** Blank





- Note:** If both servers are on same Domain, same subnet mask & has same service domain account then both **Principal** and **Mirror** Fields should be left blank **or** need to mention the a specify account that can be used for configuration.

10. Verify the configuration before you hit finish Button



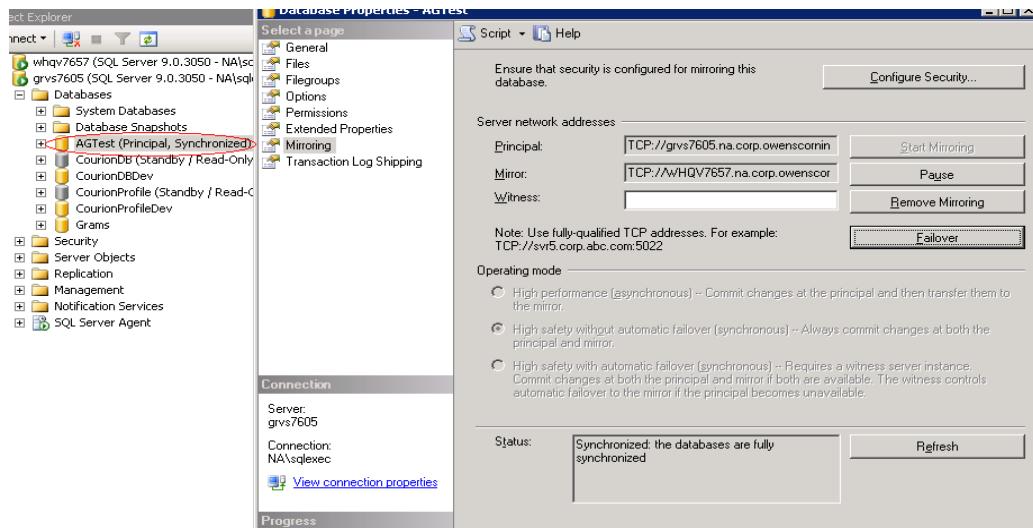
Once you hit Finish Button then you will get the below screen

11. Click on **Start Mirroring** Button.

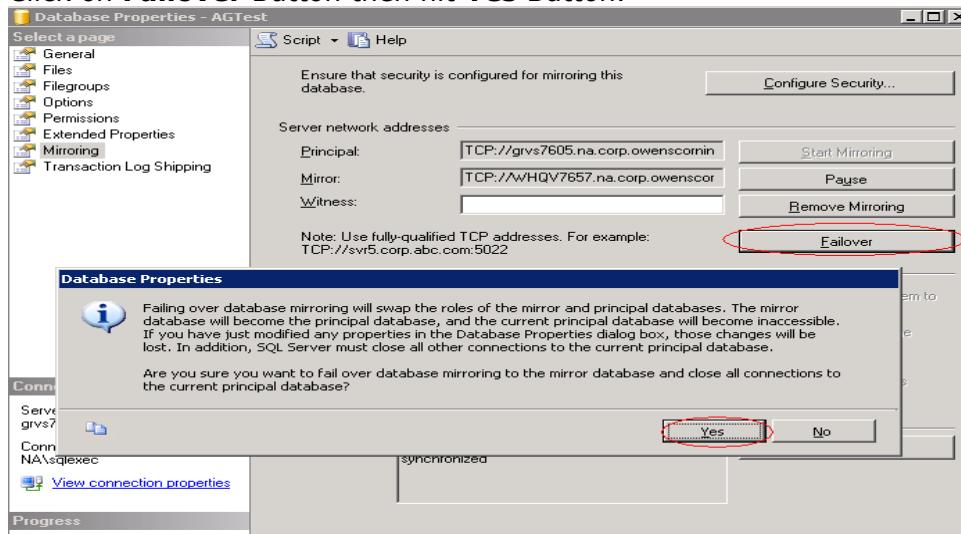


**Failover in Database Mirroring using GUI mode where both Principal and mirror server's are available.**

1. On Principal Server -Database -Right click -Go to -Properties and Click on Mirroring.



2. Click on Failover Button then hit Yes Button.



Please refresh both Principal and Mirror server's to view the changes

### **How can I bring mirror database online after principal server is down ?**

#### **Safety FULL with Witness:**

Well the answer for this 'depends on the mode in which mirroring is configured'. If mirroring is configured in High Availability mode (Full safety) then we don't need to worry about failover as the mirror server will form a quorum with witness and will initiate an automatic failover. The safety level can be set using the below command,

212



```
ALTER DATABASE dbname SET SAFETY FULL
```

```
ALTER DATABASE dbname SET SAFETY OFF
```

### **Safety FULL without Witness:**

This scenario provides high safety, but automatic failover is not allowed. This mode is called as High Protection mode. In the event of failure of the principal, the database service becomes unavailable. You need manual intervention to make the database service available. You must break the mirroring session and then recover the mirror database.

For example, prior to the failure, Server\_A and Server\_B acted as principal and mirror respectively. Server\_A fails. You need to execute the following on Server\_B to make the database service available:

```
ALTER DATABASE dbname SET PARTNER OFF  
RESTORE DATABASE dbname WITH RECOVERY
```

### **Safety OFF :**

In the event of failure of the principal, the database service becomes unavailable. You can perform a force service to make the database service available on the mirror. However, since the safety level is OFF, it is possible that there were transactions that didn't make it to the mirror at the time of the failure of the principal. These transactions will be lost. Therefore, manual failover with safety OFF involves acknowledging the possibility of data loss. For example, prior to the failure, Server\_A and Server\_B acted as principal and mirror respectively. Server\_A fails. You need to execute the following on Server\_B to make the database service available:

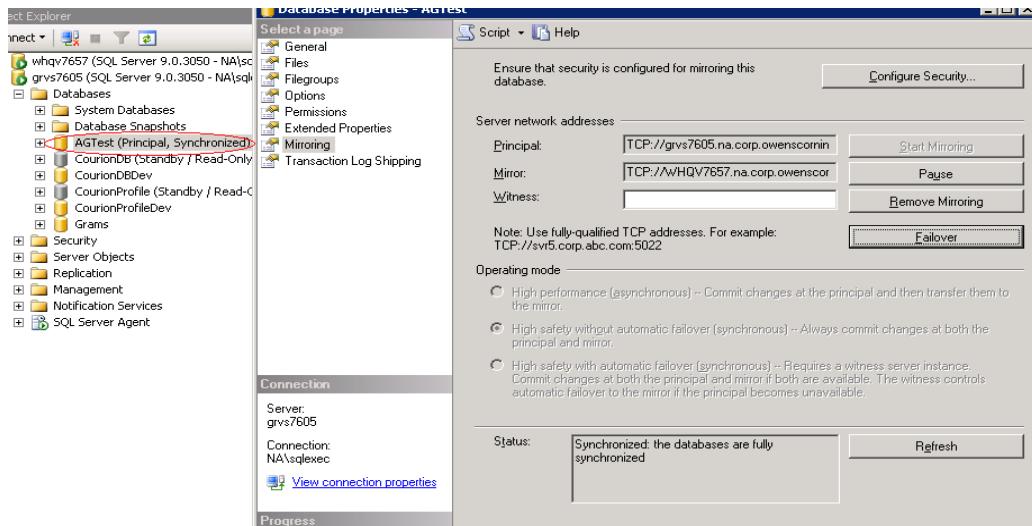
```
ALTER DATABASE dbname SET PARTNER FORCE_SERVICE_ALLOW_DATA_LOSS
```

Once the database on Server\_A becomes operational, it automatically assumes the role of the mirror. However, the mirroring session remains SUSPENDED, and you will need to manually RESUME the mirroring session.

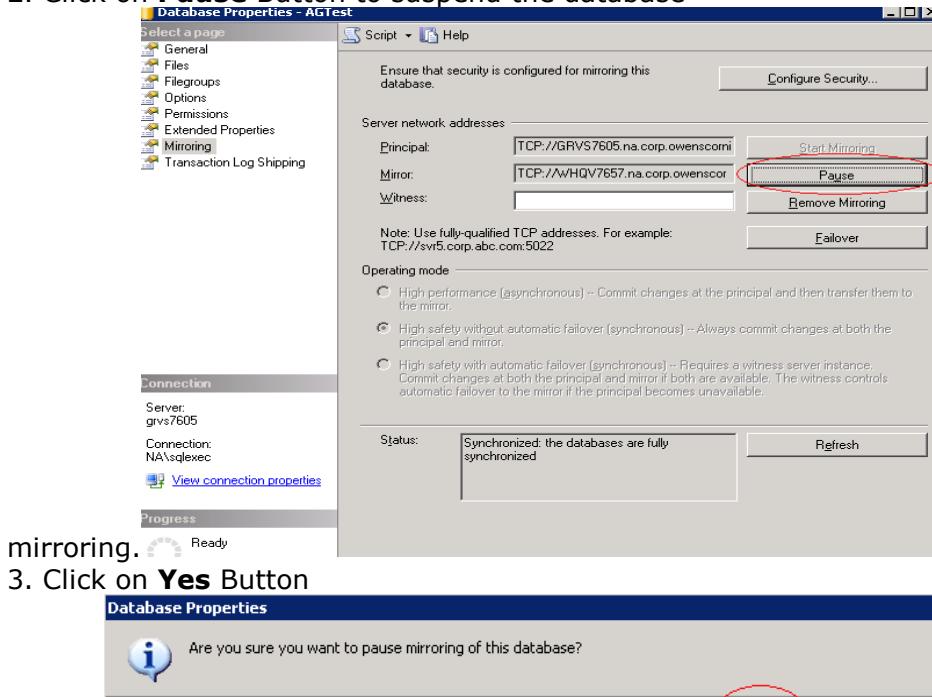
This can be achieved using GUI Mode also.

1. On Principal Server - <Database> -Right click -Go to -Properties and Click on Mirroring.

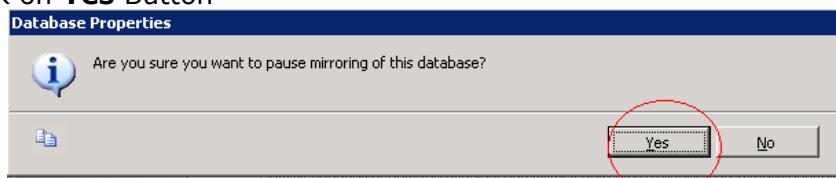




2. Click on **Pause** Button to suspend the database



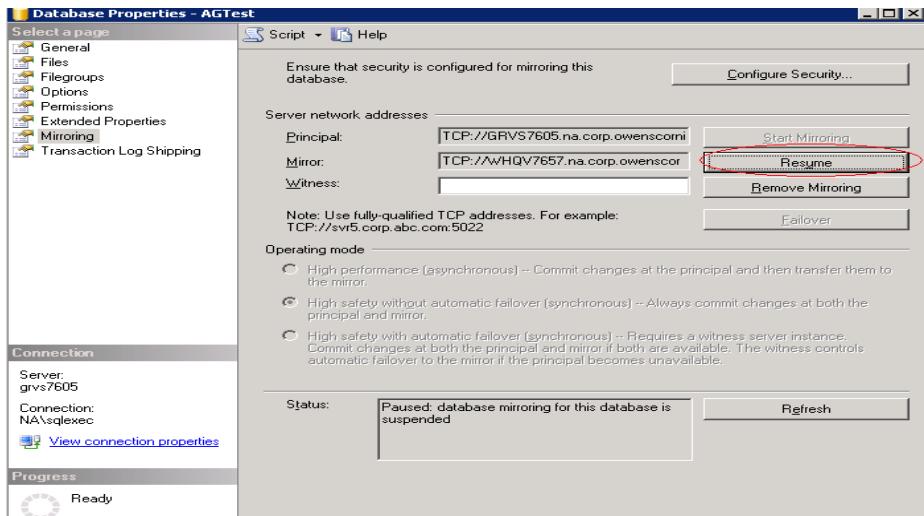
3. Click on **Yes** Button



Now database mirroring is in Suspended mode.

4 .To Resume the database mirroring on same screen Click on **Resume** Button





### Advantages Database Mirroring:

- Database mirroring architecture is more robust and efficient than Database Log Shipping. It can be configured to replicate the changes synchronously to minimized data loss.
- It has automatic server failover and client failover mechanism.
- Configuration is simpler than log shipping and replication, and has built-in network encryption support (AES algorithm).
- Because propagation can be done asynchronously, it requires less bandwidth than synchronous method (e.g. host-based replication, clustering) and is not limited by geographical distance with current technology.
- Database mirroring supports full-text catalogs.
- Does not require special hardware (such as shared storage, heart-beat connection) and clusterware, thus potentially has lower infrastructure cost

### Disadvantages of Log Shipping:

- Potential data lost is possible in asynchronous operation mode. RTO will vary and depend on several factors, such as propagation interval time and bandwidth speed.
- Mirror server/database is not available for user operation.
- It only works at database level and not at server level. It only propagates changes at database level, no server level objects, such as logins and fixed server role membership, can be propagated.
- Automatic server failover may not be suitable for application using multiple databases.



## Mirroring Vs Clustering

Clustering	Database Mirroring
<p>Clustering failover using SQL Virtual Server provides the highest availability because it immediately fails over to the second node. This failover is transparent to the end-user/client application. Clustering failover provides protection against SQL Server failures, Windows operating system crashes and hardware failures (other than disk subsystem).</p> <p>Clustering failover requires special hardware. Also, failover clustering uses a shared disk subsystem, and therefore, the computers must be physically located in the same data center, unless you plan to implement Distance Clustering. Failover clustering does not protect you against a failure in the disk subsystem and the data loss that results because of the hardware failure.</p>	<p>This same can be achieved with High-Availability operation mode without additional witness server. Db mirroring provides protection against SQL server, database failures.</p> <p>Here incase of db mirroring you don't require any additional hardware. To configure db mirroring, we need an SQL instance or server linked to the primary server which will provide you high availability.</p>

## Troubleshooting Information on Database Mirroring

Mirroring Catalog Views:

A database mirroring session consists of the relationship formed between the partner servers and potentially the witness server. Each of the participating servers keeps some metadata about the session and the current state of the databases.

\* *sys.database\_mirroring* ---> provides information about principal and mirror

\* *sys.database\_mirroring\_witnesses* ---> provides information about witness server

All the metadata required for database mirroring (in particular the mirroring failover lsn and partner server names) are kept by the mirroring partners. The witness only keeps data necessary for its role as a witness in a High Availability mode, in particular the role sequence number, which tracks the number of role changes in the session.

### Database mirroring states and transition:

Database states for each server are kept during the database mirroring session, recorded on each partner server, and reported by the *sys.database\_mirroring* catalog view. The *mirroring\_state* column returns a number for the state, and the *mirroring\_state\_desc* column returns the descriptive name for the state. State information about the witness is also reported from the same catalog view.

In addition to the states reported for each database, there are three phrases that are useful in describing the servers and databases involved in database mirroring.



1. **Exposed** - The data on the principal is exposed when it is processing transactions but no log data is being sent to the mirror. When a principal database is exposed, it is active with user connections and processing transactions. However, no log records are being sent to the mirror database, and if the principal should fail, the mirror will not have any of the transactions from the principal from the point the principal entered the exposed state. Also, the principal's transaction log cannot be truncated, so the log file will be growing indefinitely.
2. **Cannot serve the database** - When a principal server does not allow any user connections to the database and any transactions to be processed. When a witness has been set, if the principal server cannot form a quorum with another server, it will stop serving the database. It will not allow user connections and transactions on the principal database, and will disconnect all current users. As soon as it can form a quorum again, it will return to serving the database.
3. **Isolated** - A server is isolated when it cannot contact any of the other servers in the database mirroring session, and they cannot contact it. A server may be operational but communication lines are down between it and both other servers in the database mirroring session. In that case, we'll call the server isolated. If a witness has been set, then, if the principal server becomes isolated, it will no longer be able to serve the database, because there is no server in the session with which it can form a quorum.

When safety is FULL, the principal first enters the SYNCHRONIZING state and as soon as it synchronizes with the mirror, both partners enter the SYNCHRONIZED state. When safety is OFF, the partner databases start with the SYNCHRONIZING state. Once the mirror has caught up, the state goes to SYNCHRONIZED and stays there regardless of how far behind it is. For both safety settings, if the session is paused or there are redo errors on the mirror, the principal enters the SUSPENDED state. If the mirror becomes unavailable, the principal will enter the DISCONNECTED state.

#### **In the DISCONNECTED and SUSPENDED states:**

- \* When a witness has been set, if the principal can form a quorum with the witness or mirror server, the principal database is considered exposed. That means the principal database is active with user connections and processing transactions. However, no log records are being sent to the mirror database, and if the principal should fail, the mirror will not have any of the transactions from the principal from the point the principal entered that state. Also, the principal's transaction log cannot be truncated, so the log file will be growing indefinitely.
- \* When a witness has been set, if the principal cannot form a quorum with another server, it cannot serve the database. All users will be disconnected and no new transactions will be processed.
- \* When safety is OFF, the principal database is considered exposed, because no transaction log records are being sent to the mirror



## Database Snapshots

### What is a Database Snapshot?

- \* The Database Snapshot is a new feature that is available in Enterprise edition of SQL Server 2005.
- \* It provides a read-only, static view of a database.
- \* Multiple snapshots can exist for a single database on the same server instance as the database.

Database Snapshots are completely unrelated to Snapshot Replication and Snapshot Isolation.

### Uses of Database Snapshot

The following are the important usage of snapshots namely,

1. **Reporting Purposes** --> When your db is used for reporting purposes you can create snapshots so that they provide a static view of the data and also prevents blocks from occurring as insert or update statements are prevented as snapshots are read-only copies of your source db.
2. **Querying data from standby servers** --> In Database mirroring the mirror db will be in NORECOVERY mode and you cannot read the contents of the db. If you want to read through the contents of mirrored db or if you want to use the mirrored db for reporting purposes you can make use of Database Snapshots.
3. **System upgrades** --> Before applying service packs the DBA will take backup for all the databases which will be time consuming and also takes lot of space. In such scenarios database snapshots comes handy. Creating the snapshot consumes less amount of time and also the space required is very minimal.

### Creating Database Snapshot

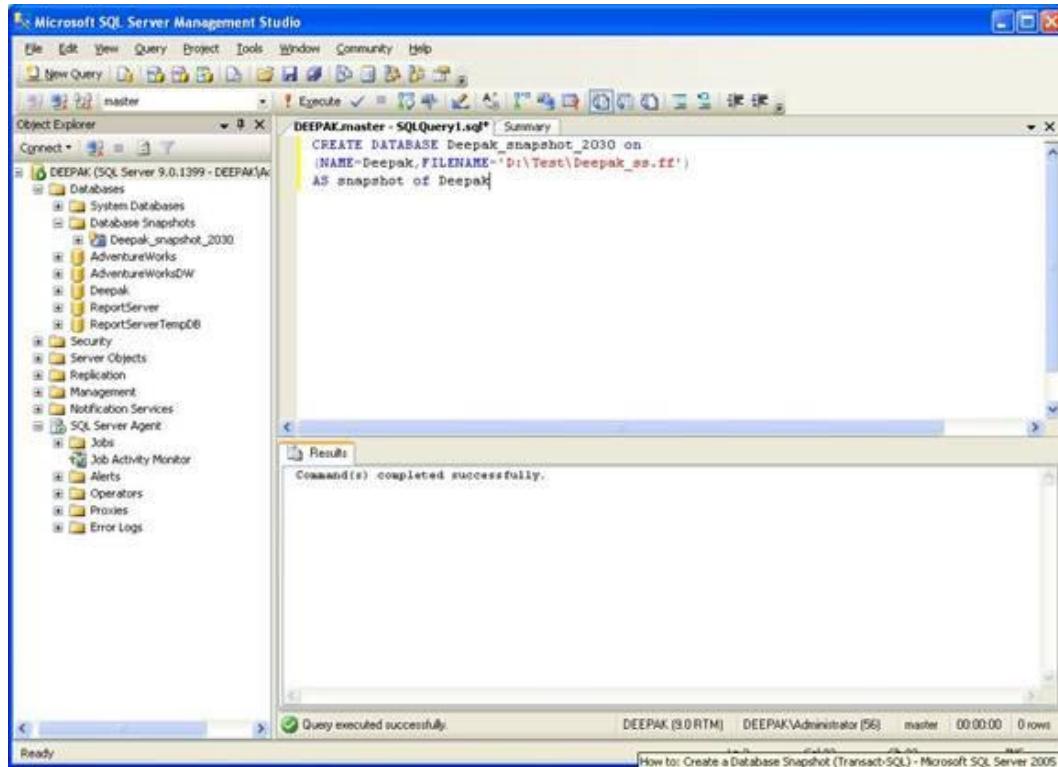
One of the exasperating things while creating a snapshot is that you cannot use SQL Server Management Studio (SSMS) instead you need to make use of T-SQL. I am creating a snapshot of the database "Deepak" as follows,

```
CREATE DATABASE Deepak_snapshot_2030 on  
(NAME=Deepak,FILENAME='D:\Test\Deepak_ss.ss')  
AS SNAPSHOT OF Deepak
```

The Snapshot name is "Deepak\_snapshot\_2030". It can be any name but its advisable to include the time @ which the snapshot was created, in my case it is 2030(i.e 8:30 pm) so that it will be easy to identify. While creating a snapshot a Sparse file is created in my case it is named as "Deepak\_ss.ss".

You need to ensure that the drive containing this sparse file must have NTFS file system else you will not be able to create the snapshot. Initially I tried to create a

database snapshot in F:\drive which has FAT file system and received the below error. Refer the below figure,



## Best Practices of Mirroring

- The principal database and the mirror database should be on separate physical hardware, and ideally, in different physical locations.
- The witness server should be on separate physical hardware, and be on a separate network (best if at a third location).
- Initial database mirroring setup should be done during less busy times, as the setup process can negatively affect performance of the production database being mirrored.
- Use high availability mode whenever possible, and high performance mode only when required.
- The hardware, along with the OS and SQL Server configuration, should be identical (at least very similar) between the two servers.
- While a fast connection is not required between mirrored servers, the faster the connection, and the better quality the connection, the better.
- You will want to optimize the performance of the mirrored database as much as possible to reduce the overhead caused by the mirroring process itself.
- Thoroughly test database mirroring before putting it into production.
- Monitor database mirroring daily to ensure that it is working properly, and is meeting performance goals.
- Develop a formal operational and recovery procedure (and document) to support mirroring. Periodically test the failover process to ensure that it works.



### **Case study/practical troubleshooting:**

#### **1: Login Failures while connecting to new principal database after failover?**

After configuring database mirroring in SQL Server 2008 and performing failover, the original mirror database now becomes the new principal database. We might have even created the same login (as in principal) in original mirror server prior to failover. But after failover if we try to connect or if the application tries to connect, the following error will be returned,

Cannot open database requested by the login. The login failed.

In that case we need to map the login to the user in the database using the procedure `sp_change_users_login` after which the application or the user will be able to successfully connect to the new principal database.

This problem occurs because the SIDs the SQL Server logins on each server do not match. Although the names for the logins are the same, the login is resolved via the SID. This is not a problem with Windows/Domain user/group logins because the SIDs for these logins are created based on the domain SID for the user/group, and hence will be the same for the same given user/group no matter what SQL Server the user/group is added to.

In order to make the `sp_change_users_login` synchronization step unnecessary, we need to create the SQL Server logins on the mirror server not only with the same name, but also with the same SID as on the principal server. This can be accomplished by using the SID specification in the 'CREATE LOGIN' statement when creating the logins on the mirror server. Here is an example where we create a the same login in mirror server as the one in principal server.

```
CREATE LOGIN WITH PASSWORD ='password',SID ='sid for same login on principal server'
```

To retrieve the SID for each login from the principal server query the `sys.sql_logins` catalog view.

You can also create all the logins with same SID in mirror server from principal server using `sp_help_revlogin` procedure. Consider this step as pre-requisite for configuring db mirroring.

#### **2: How to move the database files of a mirrored database to a new location without any downtime.**

Principal Server – N1SQL1VS1 – Dbname is “Deepak”

Mirror Server – N1SQL1DEV1MAA\KATMAI

Witness Server – N1SQL1VS1\SQL2008 - Dbname is “Deepak”



## SQL Server 2008 DBA



I have used witness server to facilitate automatic failover. All the 3 machines are having SQL 2008 RTM Enterprise edition as shown in the below screenshot.

I am moving the log file from C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\DATA folder to C:\Temp folder.

sp_helpdb Deepak					
Results		Messages			
name	db_size	owner	dbid	created	status
Deepak	741.38 MB	CORP\Deepakr	5	Jan 2 2009	Status=ONLINE, Updateability=READ_WRITE, UsedAcc...
name	fileid	Rename			
Deepak	1	C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\DATA\Deepak.mdf			
Deepak_log	2	C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\DATA\Deepak_log.ldf			

The steps I am mentioning below are applicable for SQL 2005 and SQL 2008 servers.

**Step1:** Execute the below query in Principal server,

USE Master

ALTER DATABASE Deepak

MODIFY FILE (NAME='Deepak\_log',

FILENAME='C:\TEMP\Deepak\_log.ldf')

221

The following will be the output of the above command.

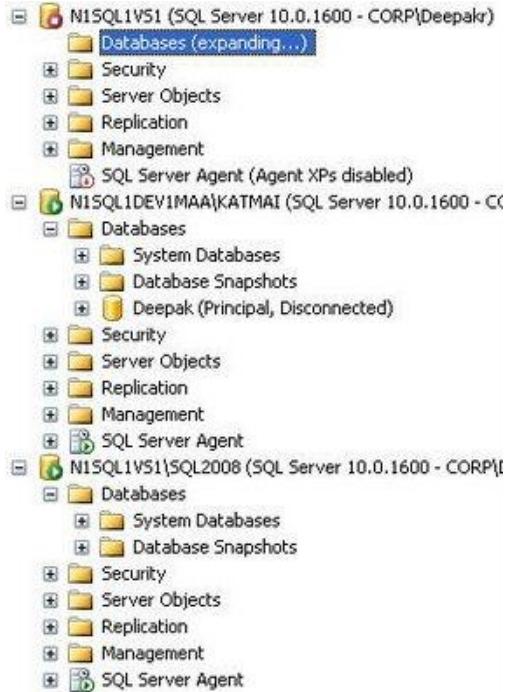


The file "Deepak\_log" has been modified in the system catalog.

The new path will be used the next time the database is started.

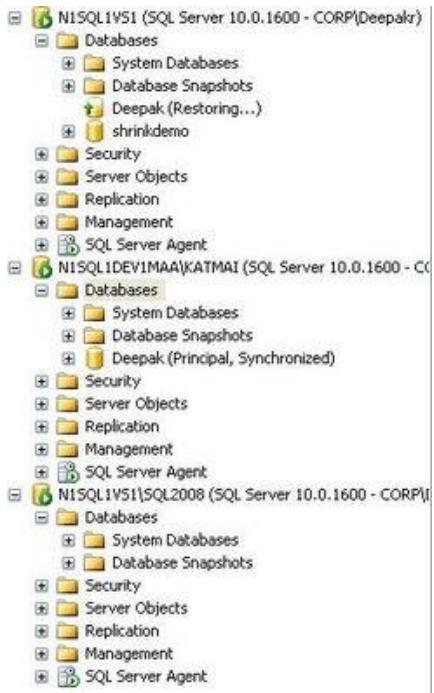
**Step2:** Now stop SQL Server in principal server N1SQL1VS1, move the deepak\_log.ldf file to C:\temp

In principal server, As shown in the below screenshot the mirror server N1SQL1DEV1MAA\KATMAI now becomes the new principal server and will be in disconnected state as the SQL Server in original principal server is down.



**Step3:** Start SQL Service in N1SQL1VS1 and it will now become the new mirror server and the database will be in "Restoring state". The new principal server is "N1SQL1DEV1MAA\KATMAI" and will be in "Synchronized state" as shown in the below screenshot.





**Step4:** Now to move the log file in the new principal server

N1SQL1DEV1MAA\KATMAI we have to repeat the steps mentioned in Step1. The database files reside in D:\database\data\ folder and will be moved to C:\temp folder.

**Step5:** Execute the below query in new Principal server, "N1SQL1DEV1MAA\KATMAI"

USE Master

ALTER DATABASE Deepak

MODIFY FILE

(NAME='Deepak\_log', FILENAME='C:\TEMP\Deepak\_log.ldf')

The following will be the output of the above command.

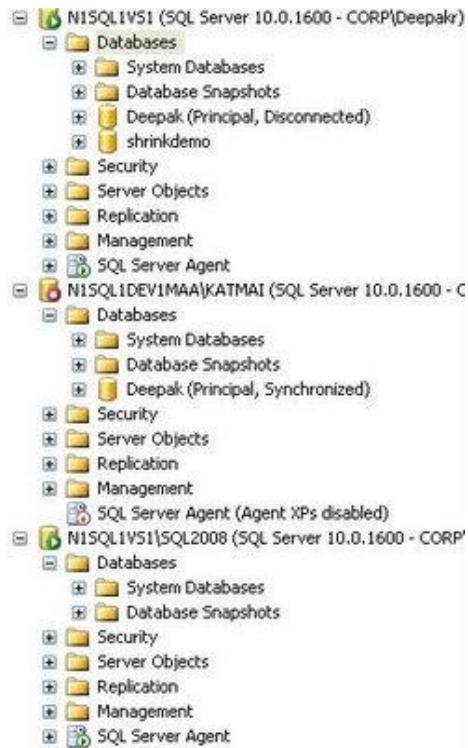
The file "Deepak\_log" has been modified in the **system catalog**

. The new **path** will be used the **next time the database is started**.

**Step6:** Now stop SQL Service in the new principal server N1SQL1DEV1MAA\KATMAI (failover will happen and the now the principal server is N1SQL1VS1 and mirror server will be N1SQL1DEV1MAA\Katmai) and move the database files to the new location C:\Temp and start SQL Server as shown in the below screenshot.

223





**Step7:** Connect to the new principal server N1SQL1VS1 and run `sp_helpdb deepak` and check the location of the database files. As shown in the below screenshot it will now reside in the new location C:\temp.

Similarly do a failover and run `sp_helpdb deepak`, you can see that the new location for the new database as C:\Temp.

Results										
Messages										
name	db_size	owner	dbid	created	status	compatibility_level	file_id	filename	filegroup	size
Deepak	741.38 MB	CORP\Deepakr	5	Jan 2 2009	Status=ONLINE, Updateability=READ_WRITE, UsedAcc...	100				
Deepak	1	C:\Program Files\Microsoft SQL Server\MSSQL10.MSS...			PRIMARY	133376 KB	Unlimited	1024 KB	data-only	
Deepak_log	2	C:\TEMP\Deepak_Log.ldf				NULL	625792 KB	2147483648 KB	10% log only	



## Replication

As it relates to SQL Server, replication is a way of keeping data synchronized in multiple databases. Implementing and maintaining replication might not be a simple proposition: If you have numerous database servers that need to be involved in various types of replication, a simple task can quickly become complex. Implementing replication can also be complicated by the application architecture.

Microsoft SQL Server has supported replication since version 6.0, and setting up replication has become significantly easier over the years (in fact, 99 percent of replication setup can be accomplished by clicking through replication wizards). However, replication involves much more than setup, and unfortunately there aren't many sources of information for implementing and troubleshooting it. The only way to learn replication is to dig through the knowledge base articles and figure things out on your own.

### Replication Terminology

SQL Server replication is commonly described by using the publisher/subscriber metaphor. A database server that makes data available for replication (source server) is referred to as the *publisher*; a collection of one or more database objects that are enabled for replication is called a *publication*. SQL Server supports replicating tables, views, stored procedures, and user-defined functions.

One or more servers that get data and/or transactions from the publisher are called *subscribers*. Replication is managed by the system database, which by default is called *distribution*. A distribution database—which can reside on the publisher, subscriber, or on a separate server—is created when you configure replication.

The server that hosts the distribution database is referred to as the distribution server or *distributor*.

It is recommended that you always use a server that is dedicated to distributing transactions. Thus, the distribution server should be used for nothing but replication.

Each database server can act as a publisher and subscriber at the same time. Each publisher can have multiple subscribers, and each subscriber can receive transactions from multiple publishers.

You should also become familiar with replication *agents*, which are implemented as SQL Server jobs that perform a particular task according to their schedule.



## Snapshot Replication

*Snapshot replication* simply takes a "snapshot" of the data on one server and moves that data to another server (or another database on the same server). After the initial synchronization snapshot, replication can refresh data in published tables periodically—based on the schedule you specify. Although snapshot replication is the easiest type to set up and maintain, it requires copying all data each time a table is refreshed.

Between scheduled refreshes, data on the publisher might be very different from the data on subscriber. In short, snapshot replication isn't very different from emptying out the destination table(s) and using a DTS package to import data from the source.

## Transactional Replication

*Transactional replication* involves copying data from the publisher to the subscriber(s) once and then delivering transactions to the subscriber(s) as they occur on the publisher. The initial copy of the data is transported by using the same mechanism as with snapshot replication: SQL Server takes a snapshot of data on the publisher and moves it to the subscriber(s). As database users insert, update, or delete records on the publisher, transactions are forwarded to the subscriber(s).

To make sure that SQL Server synchronizes your transactions as quickly as possible, you can make a simple configuration change: Tell it to deliver transactions continuously. Alternatively, you can run synchronization tasks periodically. Transactional replication is most useful in environments that have a dependable dedicated network line between database servers participating in replication. Typically, database servers subscribing to transactional publications do not modify data; they use data strictly for read-only purposes. However, SQL Server does support transactional replication that allows data changes on subscribers as well.

## Merge Replication

*Merge replication* combines data from multiple sources into a single central database. Much like transactional replication, merge replication uses initial synchronization by taking the snapshot of data on the publisher and moving it to subscribers. Unlike transactional replication, merge replication allows changes of the same data on publishers and subscribers, even when subscribers are not connected to the network. When subscribers connect to the network, replication will detect and combine changes from all subscribers and change data on the publisher accordingly. Merge replication is useful when you have a need to modify data on remote computers and when subscribers are not guaranteed to have a continuous connection to the network.

Replication can be used effectively for many different purposes, as discussed in the following sections.

## Providing High Availability

Occasionally, you might consider using replication for high availability; that is, to replicate transactions from the main server to a standby server. If the main server fails, you can then point your data sources to the standby server. Be aware that using replication for high availability takes careful planning and testing. Replication



does not provide any sort of automatic fail-over. SQL Server supports other methods of providing high availability, such as clustering and log-shipping, which might be more appropriate for your environment.

## Transporting Data

Another common use for replication is to simply move data changes from publishers to subscribers. This method is particularly useful for moving transactional data to a data warehousing server, in which it is transformed and aggregated for OLAP reporting. SQL Server provides other ways of transporting data: DTS, BCP, BULK INSERT statements, and others. Be sure to carefully consider the alternatives before implementing replication because other solutions might be cheaper or even faster than replication.

Replication needs to be planned carefully. Setting things up is easy, but there is no magic UNDO button that will reverse all your actions. Therefore, be sure to test your plan thoroughly before implementing a replication solution. The following sections discuss some of the planning steps necessary for transactional replication.

## Replication Agents

Transactional replication involves three agents: snapshot, log reader, and distribution. The snapshot agent takes a snapshot of records on the publisher and copies the data out to the snapshot folder. The snapshot agent also generates scripts for database schema and includes CREATE TABLE and CREATE INDEX scripts.

The snapshot agent doesn't have to run continuously for transactional replication. If you get your replication working properly the first time, you might never have to run the snapshot agent again after the initial synchronization. However, if you do have problems with the subscriber servers missing data, the snapshot agent is there to help.

The log reader agent reads the transaction log on the published databases. This agent moves transactions that are marked for replication to the distribution database. The distribution agent delivers transactions from the distribution database to the subscribers. Log reader and distribution agents have to run continuously (or at scheduled intervals) to keep replication working.

In addition to snapshot, log reader, and distribution agents, replication also uses a few other jobs (agents) to keep things organized. The history cleanup agent, for example, is used to delete transactions that have already been delivered from the distribution database. Indeed, if this agent did not work, the distribution database would grow very large.

## Agent Profile Settings

Replication agents are implemented as SQL Server jobs that call executable files with certain parameters. You should be aware that clicking through the replication wizards configures agents to run with default parameters. If you need to tweak agent parameters for troubleshooting or for performance reasons, you'll have to modify the replication agent's profile. (I'll discuss replication agents' parameters in the next article.)



## Security Considerations

Replication agents must have appropriate security permissions to read data on the publisher, move transactions to the distributor and apply the data and transactions to the subscribers. You can allow replication agents to run using security credentials of SQL Server Agent service; alternatively, you can define a login that has a more limited set of permissions. Security is not a joking matter: Allow your replication agents too much freedom, and a hacker can destroy your data on publishers as well as subscribers. On the other hand, not granting sufficient permissions to the agents prevents replication from working properly.

### Considerations for Transactional Replication

There are a number of considerations for transactional replication:

- Transaction log space.

For each database that will be published using transactional replication, ensure that the transaction log has enough space allocated. The transaction log of a published database might require more space than the log of an identical unpublished database, because the log records are not truncated until they have been moved to the distribution database.

If the distribution database is unavailable, or if the Log Reader Agent is not running, the transaction log of a publication database continues to grow. The log cannot be truncated past the oldest published transaction that has not been delivered to the distribution database. We recommend that you set the transaction log file to auto grow so that the log can accommodate these circumstances.

We recommend that you set the **sync with backup** option on the distribution database, which delays the truncation of the log on the publication database until the corresponding transactions in the distribution database have been backed up. This can result in a larger transaction log in the publication database.

- Disk space for the distribution database.

Ensure that you have enough disk space to store replicated transactions in the distribution database:

If you do not make snapshot files available to Subscribers immediately (which is the default): transactions are stored until they have been replicated to all Subscribers or until the retention period has been reached, whichever is shorter.

If you create a transactional publication and make the snapshot files available to Subscribers immediately: transactions are stored until they have been replicated to all Subscribers or until the Snapshot Agent runs and creates a new snapshot, whichever is longer. If the elapsed time between Snapshot Agent runs is greater than the maximum distribution retention period for the publication, which has a default of 72 hours, transactions older than the retention period are removed from the distribution database.



Although making the snapshot available to Subscribers immediately improves the speed with which new Subscribers have access to the publication, the option can result in increased disk storage for the distribution database. It also means that a new snapshot is generated each time the Snapshot Agent runs. If the option is not used, a new snapshot is generated only if there is a new subscription.

- Primary keys for each published table.

All published tables in transactional replication must contain a declared primary key. Existing tables can be prepared for publishing by adding a primary key using the Transact-SQL statement

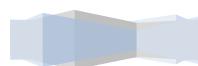
Configuring the Transactional Replication with SQL server 2005.

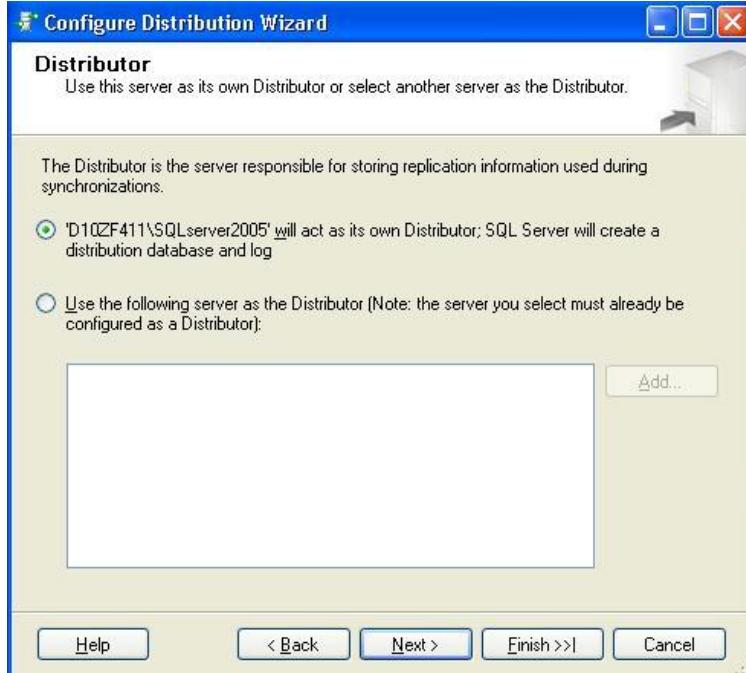
### **Configuring a Distributor**

SQL Server 2005 introduced numerous welcome improvements to replication, not the least of which is shorter wizards. Following a wizard isn't difficult, but fewer wizard screens certainly make replication setup quicker. As a rule, replication wizards in SQL Server 2005 are nearly 50% shorter than those in SQL Server 2000.

The first step in configuring replication is designating a server that will take care of storing and delivering replicated transactions—the distributor. A single server can act as a publisher, distributor, and a subscriber, all at the same time. However, in a realistic scenario you're likely to use two different servers as publisher and subscriber. Using a separate server as the distributor can help to reduce the load on the publisher.

To invoke the Configure Distribution Wizard, connect to an instance of SQL Server by using the SQL Server Management Studio (SSMS), navigate to the "replication" folder, right-click this folder, and choose Configure Distribution from the pop-up menu. Replication wizards are no longer modal; that is, you can continue working with SSMS while the wizard is active. The first screen of the wizard simply informs you of the tasks that this wizard can help you to accomplish. If you don't ever want to see this screen again, simply check the option to skip the introductory screen in the future. The next screen asks whether you want to use the local server or a different server as the distributor. Fig-1





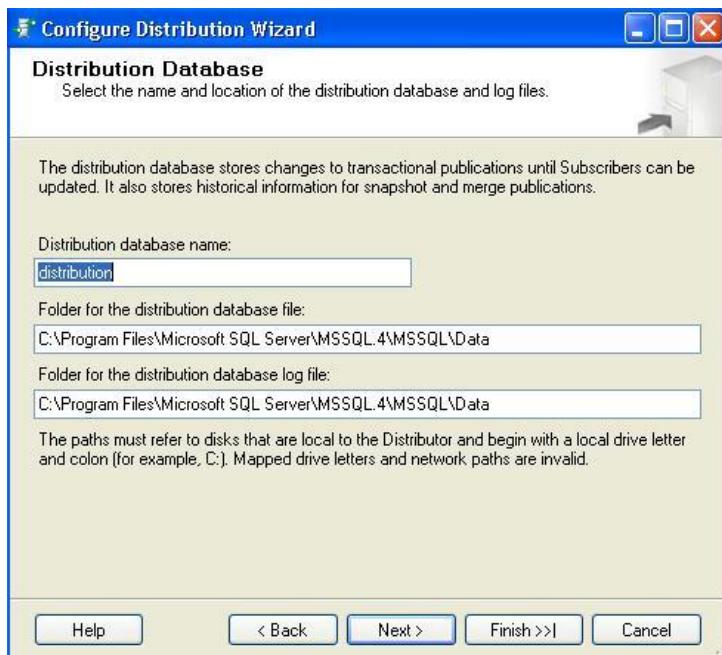
If you want to use a remote distributor, you must first run the Configure Distribution Wizard on that server. For this example, I'll use the same instance as both publisher and distributor. The next screen allows you to specify the snapshot folder where data and schema of the published database will be stored. By default, the snapshot folder is called ReplData and is created within the directory where the current SQL Server instance is installed.



Notice the warning in the dialog box, indicating that the current directory doesn't support pull subscriptions. To use pull subscriptions, you need a network folder for

storing snapshots. Because both publisher and subscriber instances of SQL Server in this example will reside on the same computer, I can safely disregard this message, and simply click Next.

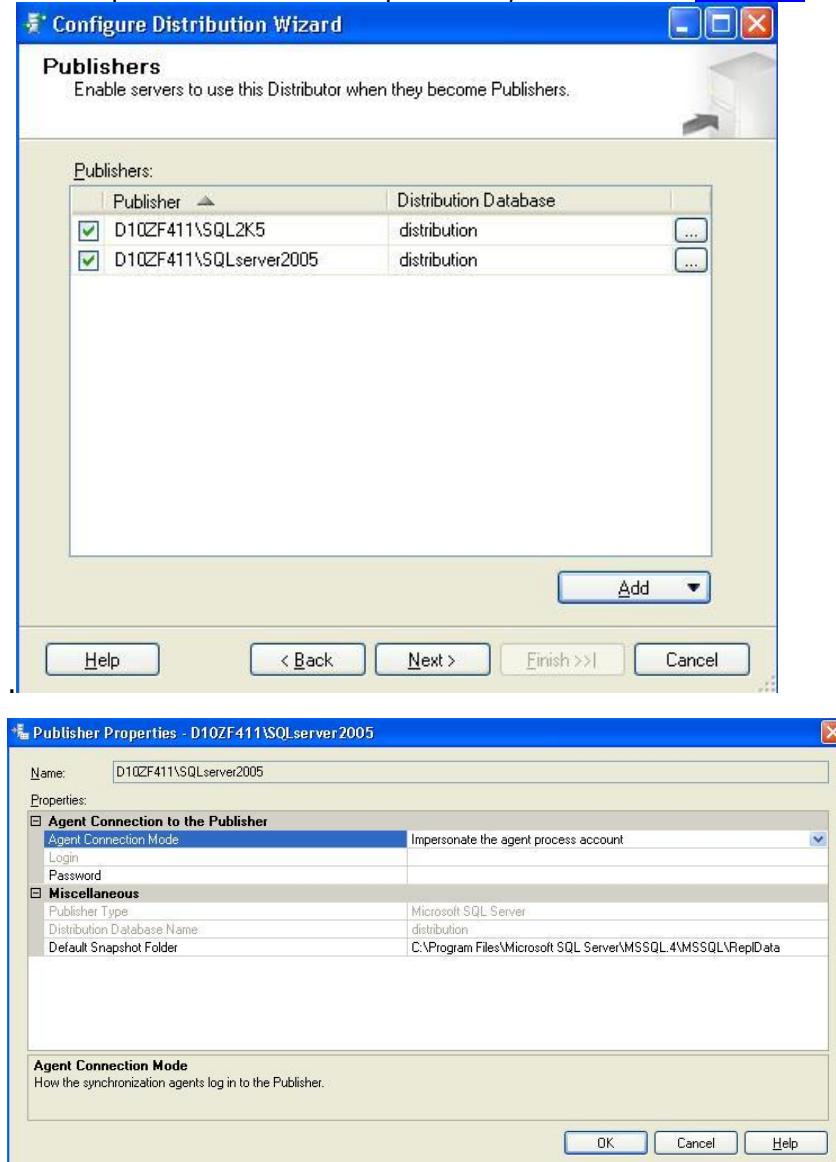
The following screen allows for configuring the distribution database's name and the location for its data and transaction log files. By default, the distribution database is called distribution; you can modify the name if you have a compelling reason to do so. For example, if you have dozens or hundreds of publications, you might want to have multiple distribution databases, with descriptive names for each one. The wizard will use the default location for database and log files. You can configure the default location on the Database Settings tab in the Server Properties dialog box in SSMS (right-click the server and choose Properties to access the dialog box). Alternatively, you can change file locations in the wizard, Fig-3



The next screen enables servers to use the current distributor when they're configured as publishers ([see Figure 4](#)). This screen has a couple of interesting options. First, if you click the ellipsis (...) button next to a publisher, you'll get a dialog box that allows you to configure the log reader agent's security credentials as



well as the snapshot folder for this publisher, as shown in [Figure 5](#)



Second, the Add button allows you to add a SQL Server or Oracle publisher. This feature is worth your attention because using the distribution database for an Oracle publisher wasn't available in previous versions.

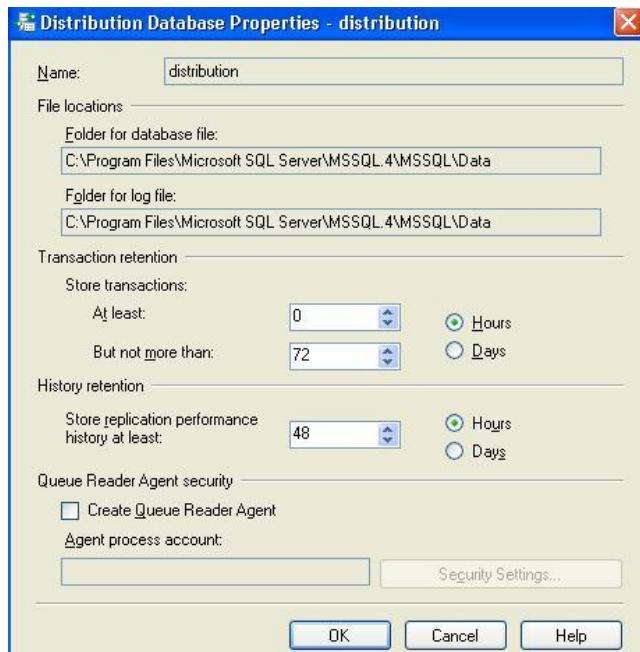
After you've enabled the publishers, you can set a password for remote publishers ([see Figure 6](#)). You must enter the same password twice. SQL Server 2005 allows the administrator to enforce password policies and password expiration. Hence, the wizard warns you that the password you enter for a remote publisher must meet the password policy requirements.





After you click Next on this screen, you can configure distribution right away, save the script for later execution, or perform both actions. If you choose to save the script, you'll be asked for the location where you want to save the file. At this point, the wizard presents a synopsis of the steps it's about to undertake; once you click Finish, the wizard will create the script for adding a distributor and/or save the script, depending on what you specified.

Once you've configured the distribution database, you can read or modify the distributor properties by right-clicking the replication folder and choosing Distributor Properties. The resulting dialog box has two pages—a "general" page and a "publishers" page. The "general" page allows you to view distribution database properties and modify settings for transaction retention and/or history retention ([see Figure 7](#)).



Notice that you're also allowed to create and configure a queue reader agent from this screen. The queue reader agent is beyond the scope of this article.

The "publishers" page of the Distribution Database Properties dialog box lets you add a publisher or change existing publishers' properties.

#### WARNING

The Configure Distribution Wizard adds a login called "distributor\_admin" to your server. This login is a member of the sysadmin server role—meaning that it has no limitation as far as what it can do on the server. This is why it's absolutely imperative to create a strong password for connecting to the distributor.

### **Creating a Publication**

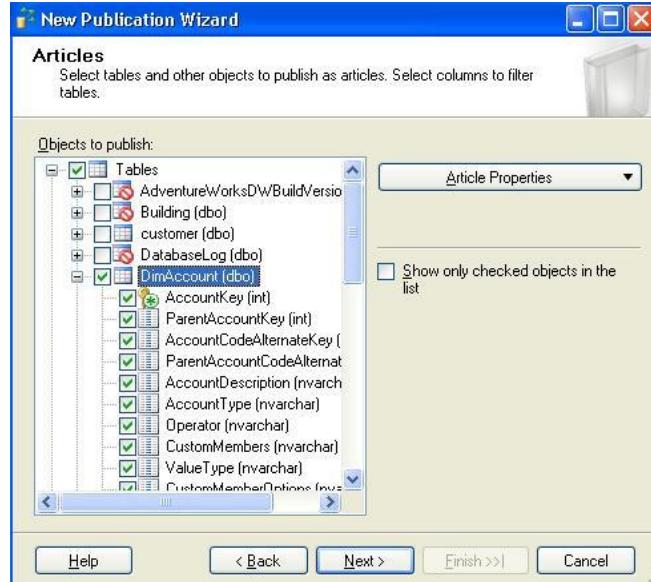
Once you've configured a distributor, you're ready to create publications. To invoke the publication wizard, right-click the local publications folder and choose New Publication from the pop-up menu. As with the Distribution Configuration Wizard, the first screen of this wizard is introductory in nature and can be skipped. The second screen allows you to choose the database in which you want to create a publication; for purposes of this article, I'll create a publication within the AdventureWorksDW database that can be created as part of SQL Server 2005 installation. After selecting the database, you must choose the publication type. The wizard offers the following options:

- Snapshot Publication
- Transactional Publication
- Transactional Publication with Updatable Subscriptions
- Merge Publication

The wizard includes a brief description of each type of publication. I'll use the transactional publication for this example; refer to my earlier articles for more info about other publication types.

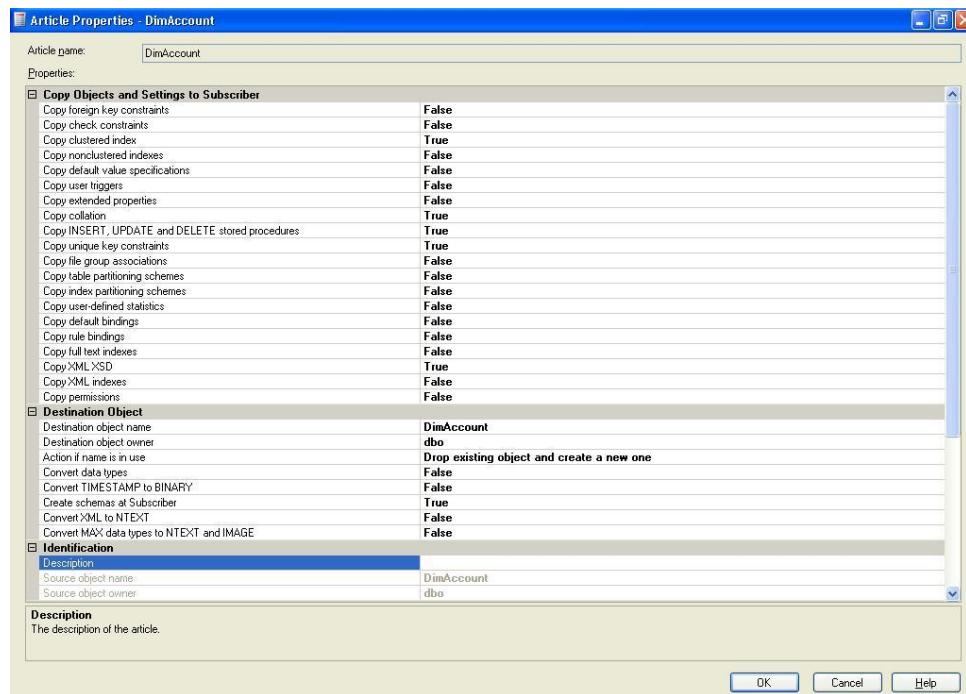
A transactional publication can contain one or more articles. An *article* can be a table, a view (including indexed views), a user-defined function, or a stored procedure. For this example, I'll replicate the dimAccount table from the AdventureWorksDW database. As shown in [Figure 8](#), I can replicate all columns or a subset of all columns within a given table.





Replication has certain rules as far as which columns can be filtered. Transactional replication prohibits filtering primary-key columns. In addition, if your publication allows updateable subscriptions, you must replicate the msrepl\_tran\_version column (added by SQL Server when you create such publications). Further, publications that allow updateable subscriptions must replicate any column that doesn't allow nulls, doesn't have a predefined default, and isn't an identity column.

If you check the box Show Only Checked Objects in the List, the wizard limits the list of articles to only those that have been checked. The Article Properties button allows you to set properties for the highlighted article or for all table articles. As [Figure](#) shows, you can set a multitude of replication-related properties for each article.



Most properties you can set for table articles are self-explanatory; for example, the Copy Foreign Key Constraints option instructs the replication to include foreign key constraints when creating the table in the subscriber database.

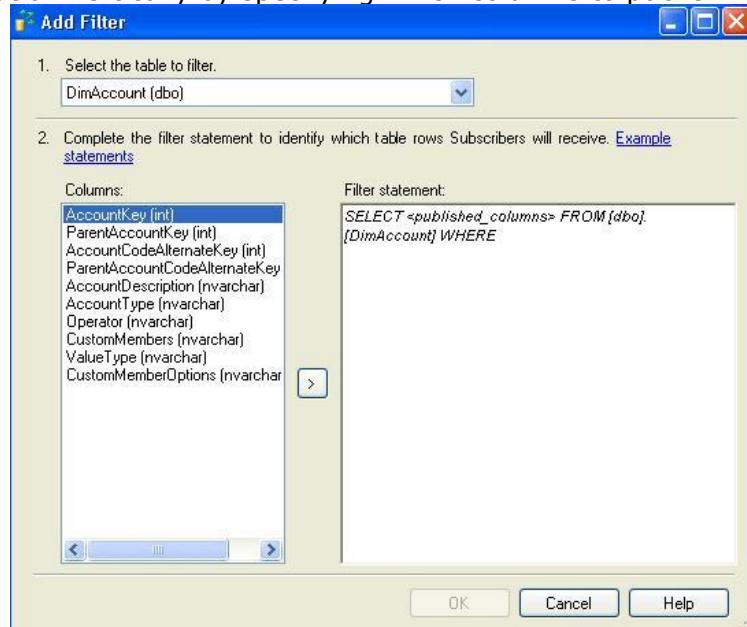
A few properties deserve additional consideration:

- **Destination Object Name, Destination Object Owner.** The destination table isn't required to have the same name or the same owner as the source object.
- **Convert Data Types.** This option automatically changes a user-defined data type to the base data type, because the user-defined data type might not exist on the subscriber(s).
- **Convert TIMESTAMP to BINARY.** When replicating a column with a TIMESTAMP data type, you can convert it to BINARY. The TIMESTAMP data type tracks the sequence of modifications; every time you change a data row, SQL Server will automatically change the value of the column with the TIMESTAMP data type. This is important because, if you're not careful, you might end up with different values in the column with the TIMESTAMP data type on the publisher and the subscriber.
- **Convert MAX Data Types to NTEXT and IMAGE.** This option translates VARCHAR(MAX) and VARBINARY(MAX) data types, which are new in SQL Server 2008, to respective data types supported in previous versions.
- **Convert XML to NTEXT.** Translates the new XML data type to NTEXT.
- Another option that wasn't available through wizards in previous versions of SQL Server is automatic identity range management. This option allows the database administrator to set the ranges of valid values for the identity column in the publisher and subscriber databases. For example, we could assign values 1,000,000 and greater to the publisher and 1 to 1,000,000 to the subscriber. When the publisher database reaches the upper limit for the identity range, it will automatically assign a new range so that publisher and subscriber identity values don't overlap.

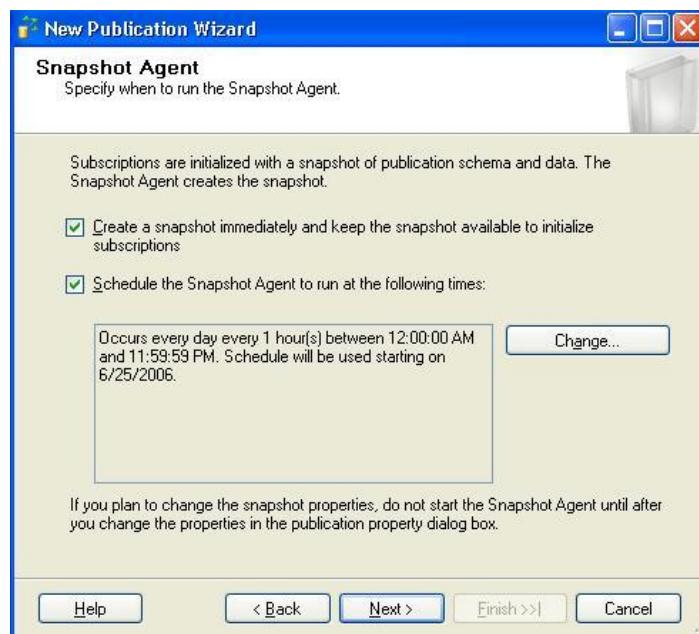


- The final group of options (not shown in Figure 9) determines how to replicate INSERT, UPDATE, and DELETE statements to the subscriber.

Once you've set the necessary properties for the article you want to replicate, you can add publication filters (see Figure 10). In previous versions of SQL Server, these filters were referred to as *horizontal filters*—you create them by supplying a WHERE clause to limit the number of published rows. As shown earlier, now you can filter the publication vertically by specifying which columns to publish.



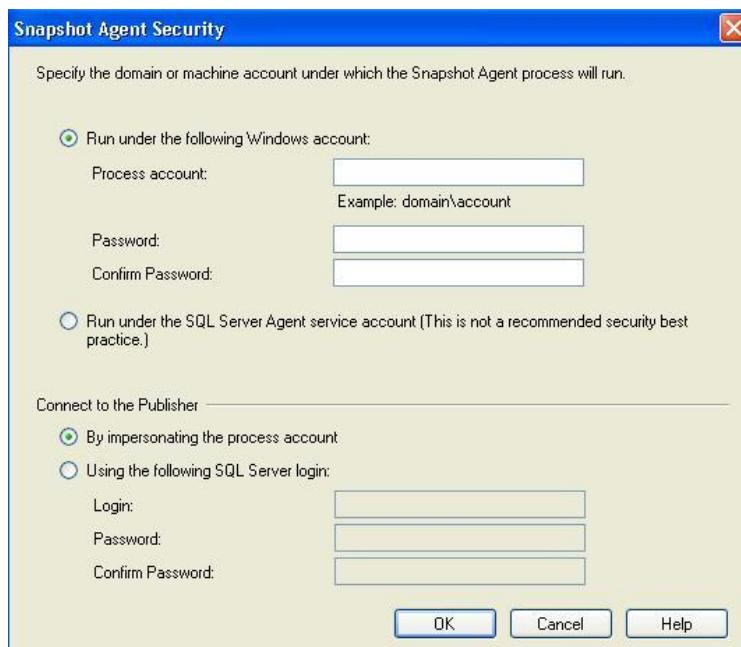
The next step is to create a snapshot and/or specify the snapshot agent's schedule, as shown in



The snapshot agent copies the schema and data of the replicated article(s) into the snapshot folder. If you click the Change button on this screen, you'll get the typical dialog box for creating job schedules; you can run the snapshot agents monthly, weekly, daily, or even multiple times per day.

Next you specify the security settings for the snapshot and log reader agents ([see Figure 12](#)). I'll discuss replication security in greater detail in a later article about transactional replication agents. For now, you just need to know that you can customize security for each agent or use different credentials for each.

The wizard next offers you the option to script the commands for creating the publication. Review the synopsis of the steps the wizard is about to undertake; then specify the publication name and click Finish to create the publication.



You can view the newly created publication's properties by expanding the local publications folder, right-clicking the publication, and choosing Properties from the pop-up menu. The properties dialog box has several pages, each of which has a specific purpose:

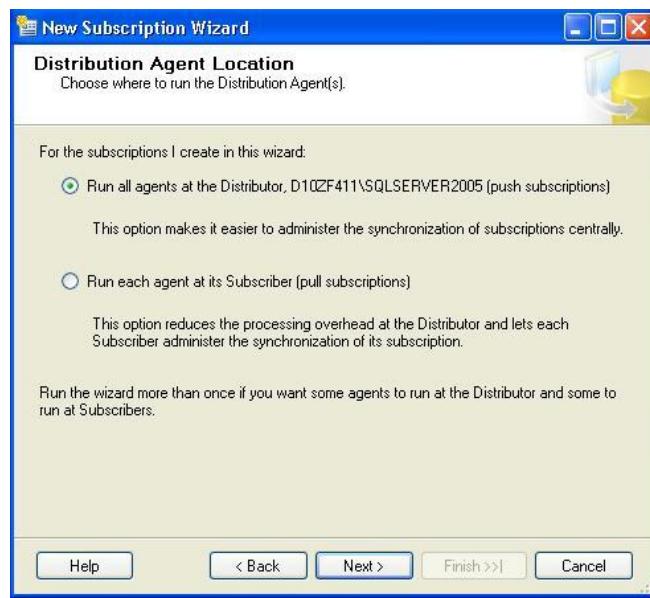
- **General.** Shows the publication's name, description, type, and the database on which the publication is based. You can modify subscription expiration options from this page.
- **Articles.** Lets you review the published articles, modify their properties, or add new articles to the publication.
- **Filter Rows.** Allows you to create horizontal filters for articles.



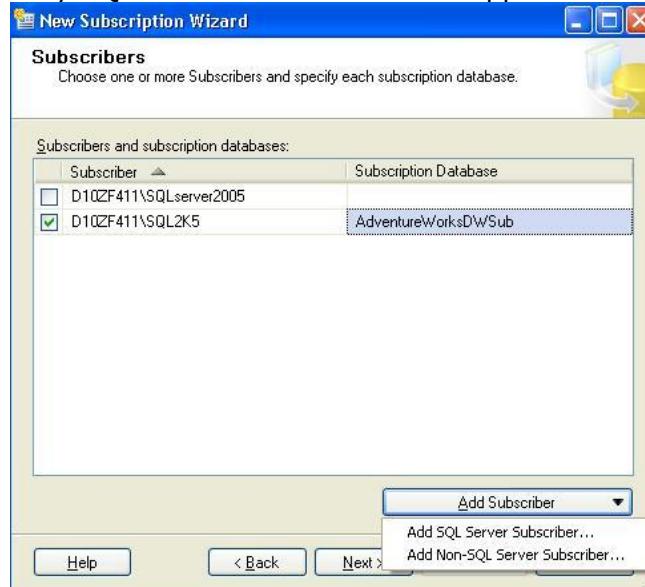
- **Snapshot.** Enables you to specify the snapshot folder location, snapshot format, or additional scripts to be executed before and after applying the snapshot.
- **Agent Security.** Controls security settings for the log reader and snapshot agents.
- **Subscription Options.** Provides a multitude of options for subscribers to the current publication. The following table describes the subscription options you can set through the Publication Properties dialog box. Note that several of these options are new in SQL Server 2005.

### Creating Subscriptions

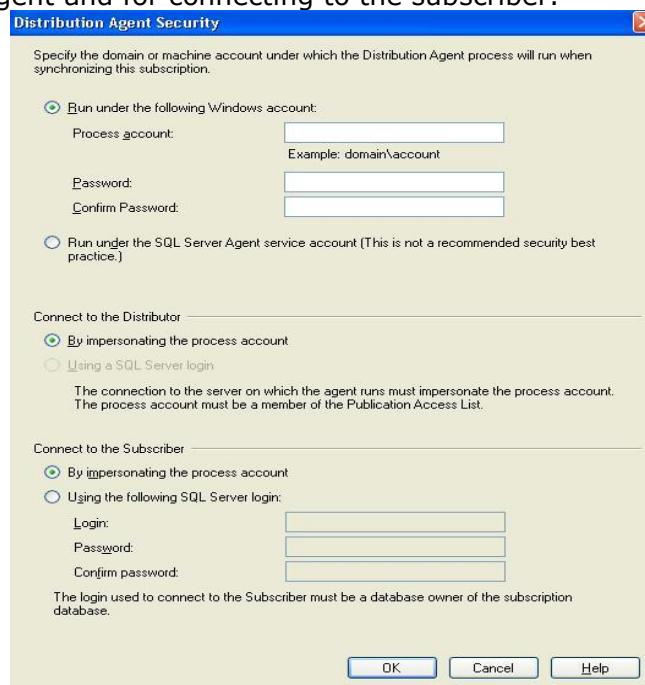
Unlike previous versions, of SQL Server 2005 allows you to use the same wizard to create either pull or push subscriptions. To invoke the new subscription wizard right-click the publication (or the local subscriptions folder) and choose New Subscriptions from the pop-up menu. After you get past the introductory screen, select the publication for which you want to create subscription(s). Next, indicate whether you want to use pull or push subscriptions (see Figure 13). Pull subscriptions reduce the load on the publisher, whereas push subscriptions make it easy to administer all subscriptions at the central location. For this example, I'll use push subscriptions, but the wizard screens are nearly identical for pull subscriptions.



Next you choose a subscribing server and database, as shown in [Figure 14](#). You can use an existing database or create a new database; if you choose to create a new database on the subscribing server, you'll get the typical dialog box for creating databases. More interestingly, note that the wizard allows you to use a non-SQL Server subscriber. You can choose either an Oracle or IBM DB2 subscriber for push subscriptions; only SQL Server subscribers are supported if using pull subscriptions.



After specifying the subscriber server and database, you need to configure distribution agent, keep in mind that you can either impersonate the SQL Server Agent or use a separate Windows login or SQL Server login for the distribution agent. For this example, I'll use the SQL Server Agent service account for running the distribution agent and for connecting to the subscriber.



Now it's time to define a synchronization schedule—how often you want the replicated transactions to be delivered to the subscriber(s). Replicating transactions *continuously* is the best option if you want to achieve minimal latency; however, this option requires more work on the publisher for push subscriptions and on the subscriber for pull subscriptions. *Scheduled delivery* is a good option if you want to minimize the load during business hours and deliver commands only at certain times each day. *On-demand delivery* can be a viable option if you want to synchronize your databases only occasionally.

After indicating the desired synchronization schedule, you can initialize the subscription database. During initialization, replication creates the published objects' schemas and copies data from the snapshot folder to the subscription database; in addition, the stored procedures used for replication are created in the subscriber database. In the dialog box, you can specify that you don't want to initialize the subscriptions—this option is useful if the schema and data already exist on the subscriber. Other options are to initialize subscriptions immediately or at first synchronization—that is, the first time the snapshot agent runs.



You're done specifying all the information that the wizard needs to create subscriptions. At this point, you have the option to script the subscription and/or to create subscriptions. The wizard allows you to review the summary of the steps it's about to undertake before you click the Finish button.

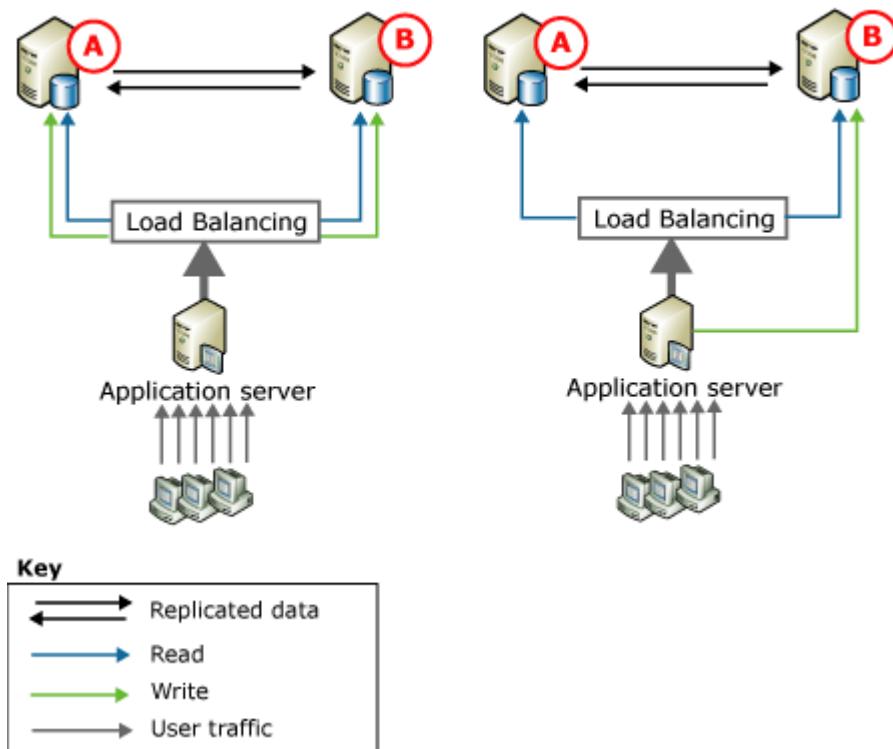


## Peer To Peer Replication

Peer-to-peer replication provides a scale-out and high-availability solution by maintaining copies of data across multiple server instances, also referred to as *nodes*. Built on the foundation of transactional replication, peer-to-peer replication propagates transactionally consistent changes in near real-time. This enables applications that require scale-out of read operations to distribute the reads from clients across multiple nodes. Because data is maintained across the nodes in near real-time, peer-to-peer replication provides data redundancy, which increases the availability of data.

With Peer to Peer replication the idea is that any subscriber is also a publisher, so data can move both ways such as bi-directional replication. With Peer to Peer replication there is a mechanism that knows when a transaction has been replicated, so the transaction does not continue to be replicated over and over again. The one area that still needs to be handled is if the same data is modified on multiple servers and therefore some type of conflict resolution may need to take place.

Below is a diagram that shows how Peer to Peer replication works. This diagram shows two different configurations: on the **left** side both nodes A and B are used for reading and writing and on the **right** side only node B is used for reading and writing where node A is used only for reading data. The application server, which could be a web server, determines which node is used for which component.



The following scenarios illustrate typical uses for peer-to-peer replication.



This model could be further extended to have a node C, D, etc... and the application server could determine which server is utilized for reading and/or writing. In most cases database servers are more read intensive than write intensive, so having more read only nodes would allow you to get additional I/O throughput on your database servers. If you also needed to have more nodes handling writes, you could build this into your application servers to determine which writes are done to what servers.

With Peer to Peer replication when data updates take place the data is then replicated to all other Peers, so the offloading of read activities could be directed to any one of the nodes. From a failover solution, since all nodes act as both publishers and subscribers you can easily point your application server to any of the nodes to act as your primary database.

### **Case Study: How to Configure Peer-to-Peer replication**

P2P replication is new to SQL server 2005 and though looks like complex in theory its one of the easiest to configure and maintain on long run. The following steps take you through the configuration process.

#### **Configuring the distributor**

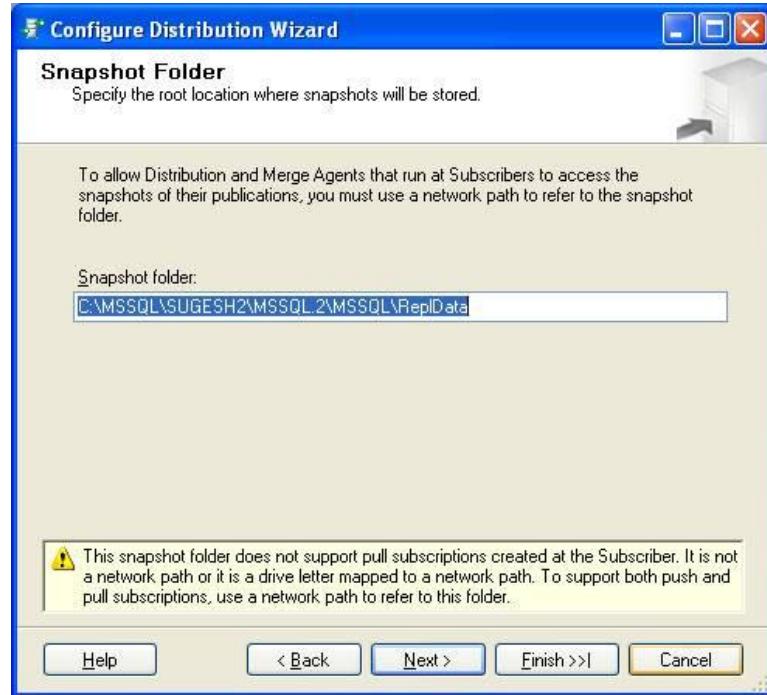
Distributor has to be configured for all the nodes participating in the replication. Each node should have its own distributor. Once you click the configure distribution you will be taken to the welcome screen. Click next.



Next you will be asked for the details of the distribution database server and name. Confirm the detail and click next.



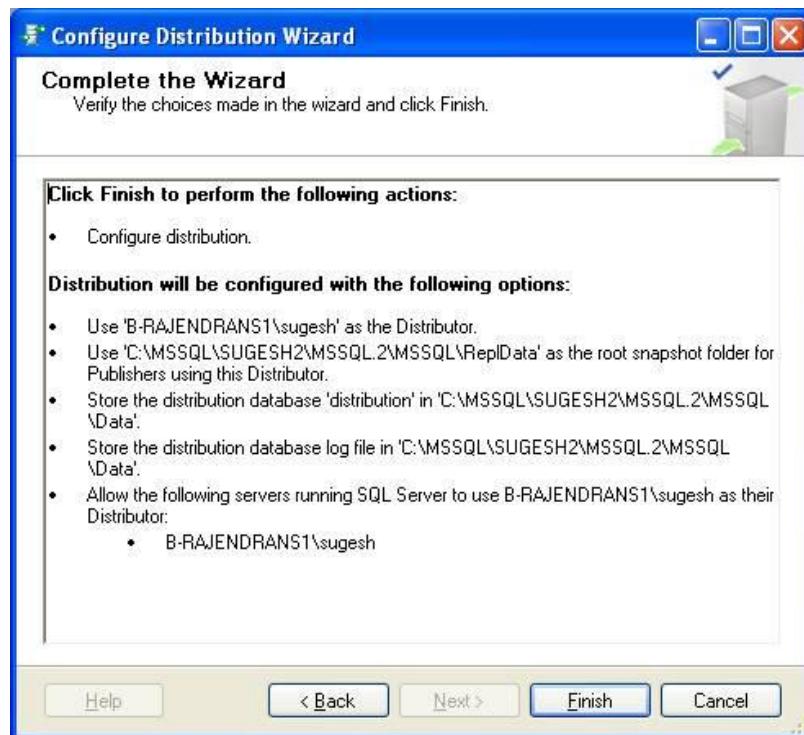
Next you will be asked for the shared folder where the snapshot will be placed.  
Confirm and Click next.



Next you will be asked to confirm the details of Distribution database click next after confirmation.



Once you are done with this, click finish so that the distributor will be configured for the server. Remember to follow these steps for all nodes participating in the replication. If you already have it configured the you are all set to start with the data replication.





### Configuring Publication database:

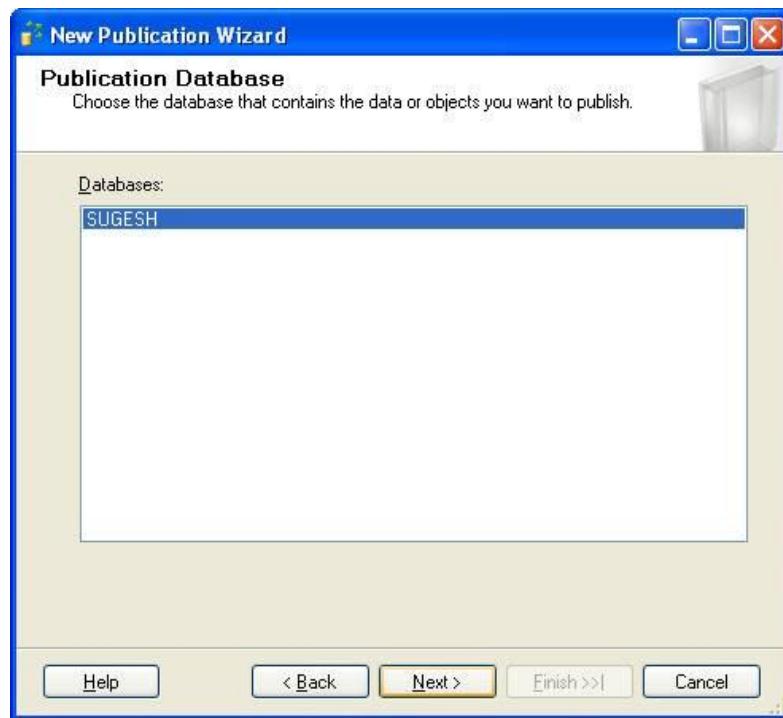
The next step involved is configuring the publication for the replication instance. Login to one of the node and create a new publication following the below steps.

On the welcome screen, click next.

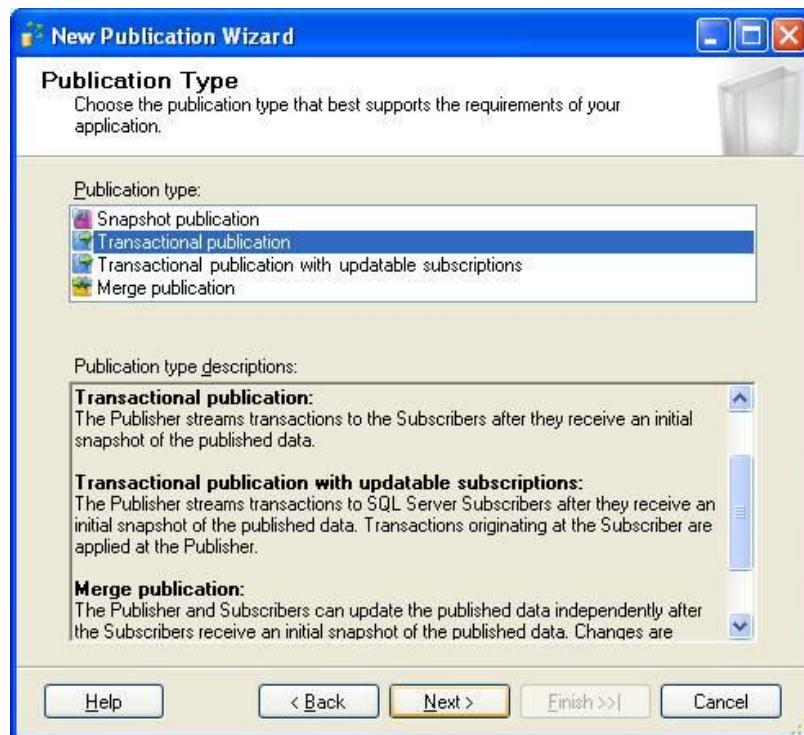


246

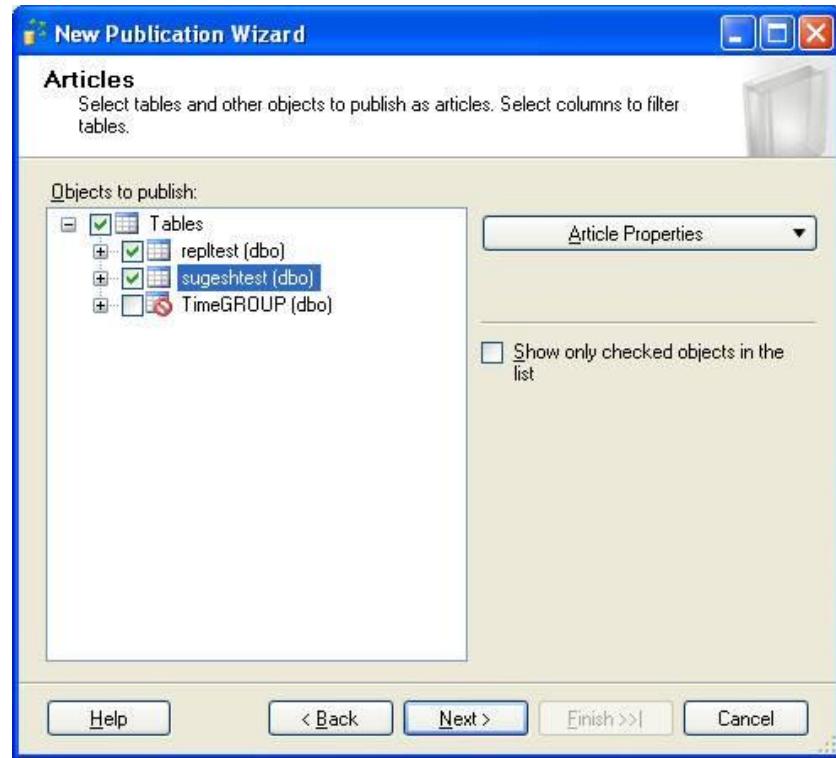
Confirm the database to be published and click next.



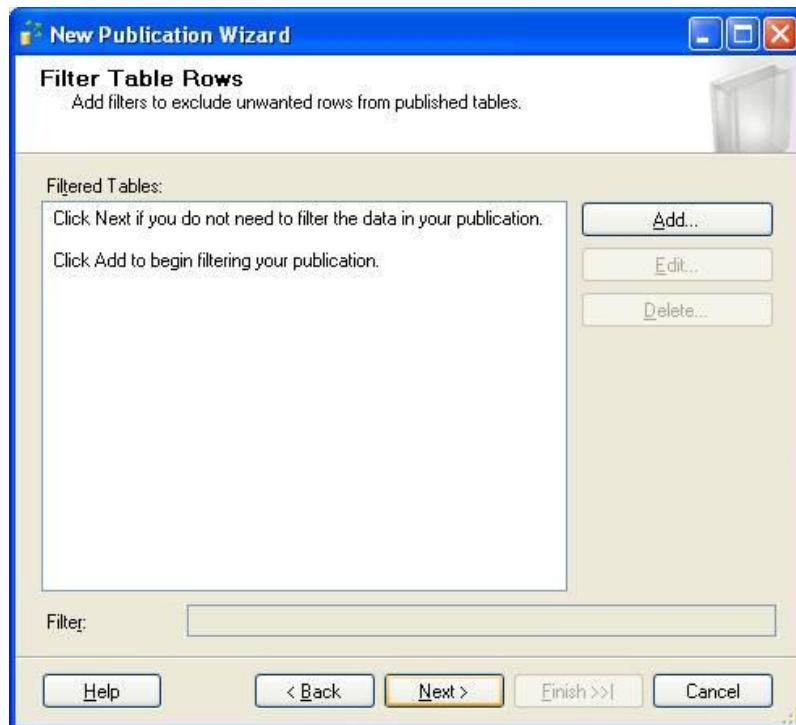
Click on transactional replication and click next.



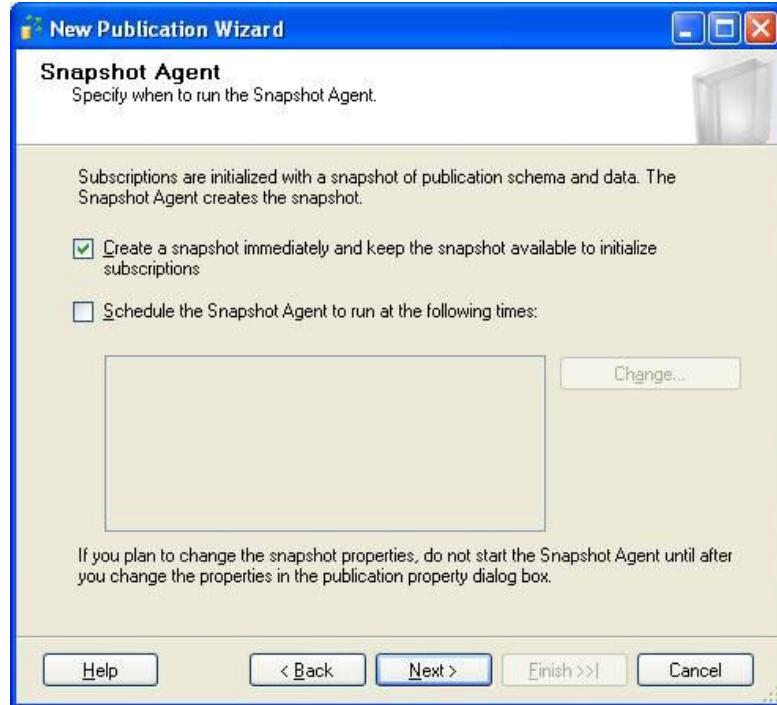
Select the articles to be published and click next. Remember that P2P replication is a kind of transactional replication you can add articles only having a primary key column.



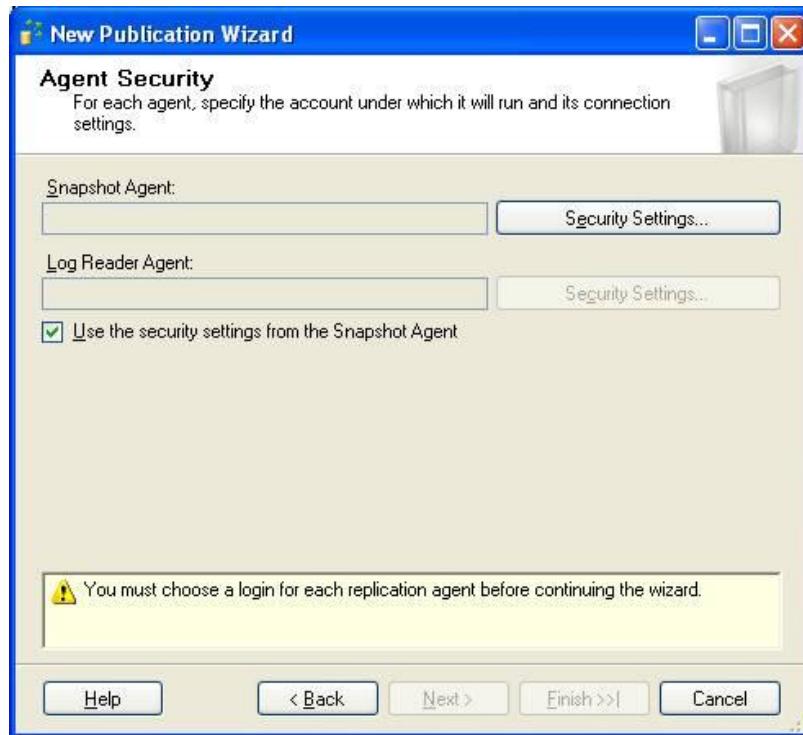
You cannot filter rows in P2P replication so ignore the next screen clicking next.



Next screen is about creating the snapshot for the publication. Choose what you need and click next.



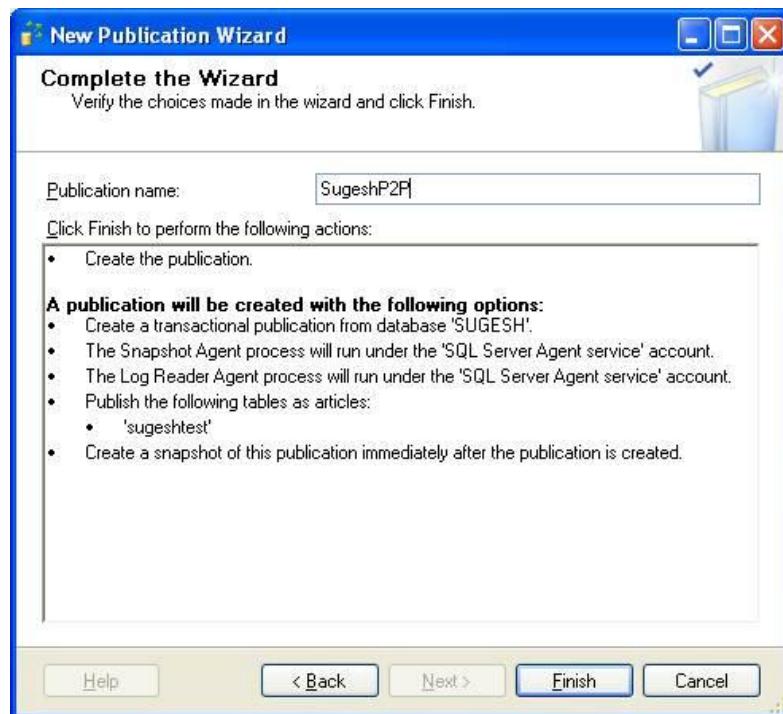
Configure the agent security for the publication and click next. This is the login which the replication agents will use to communicate within themselves.



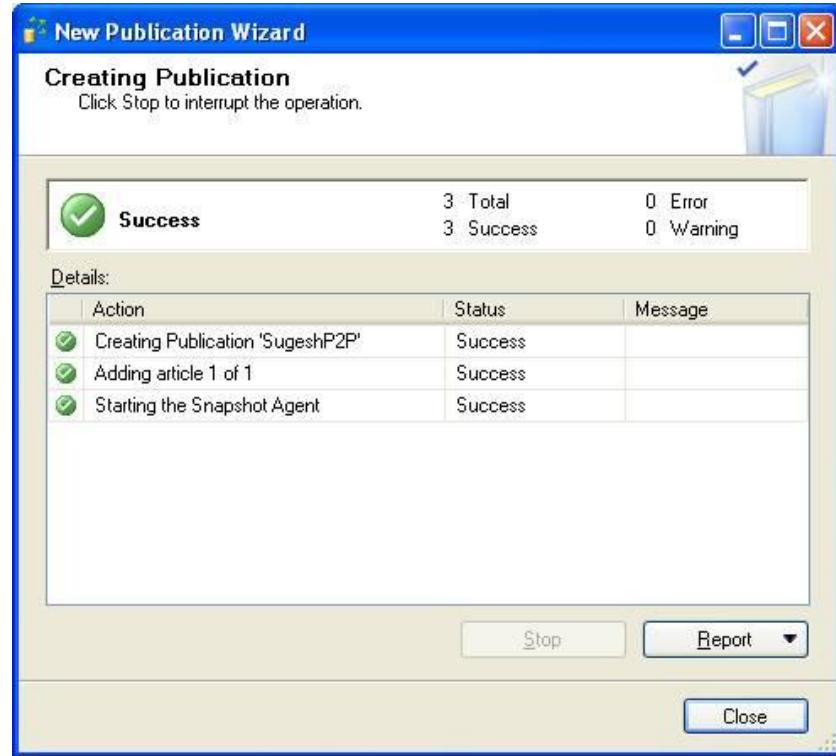
Click on the next screen for creating the publication.



Enter a Name for the publication and click finish.

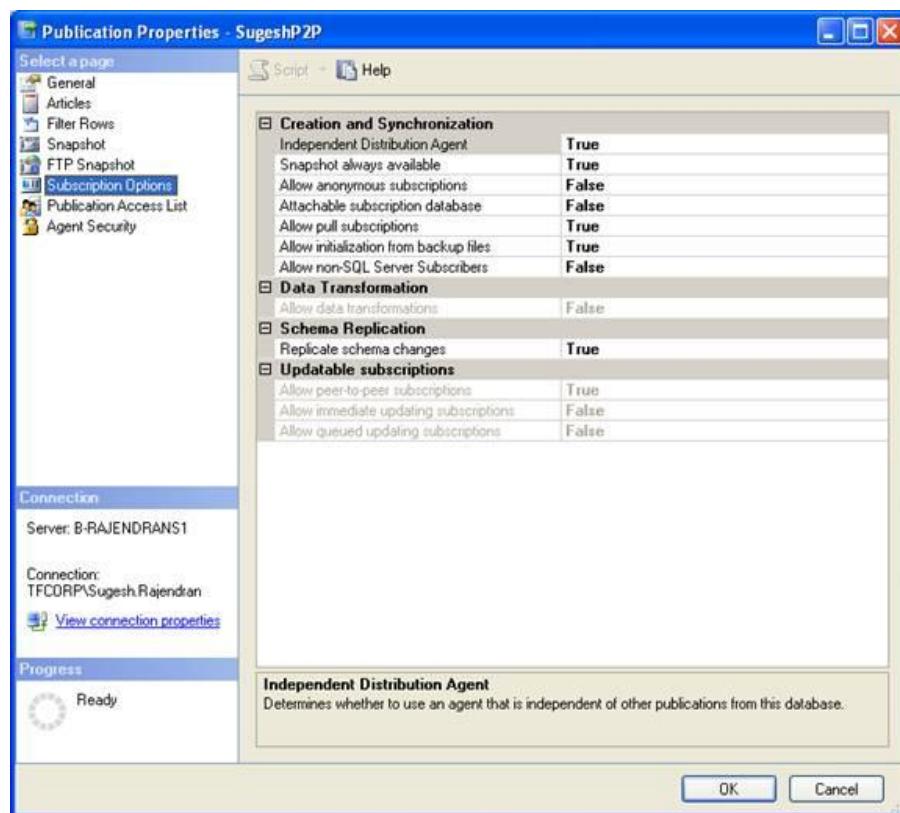
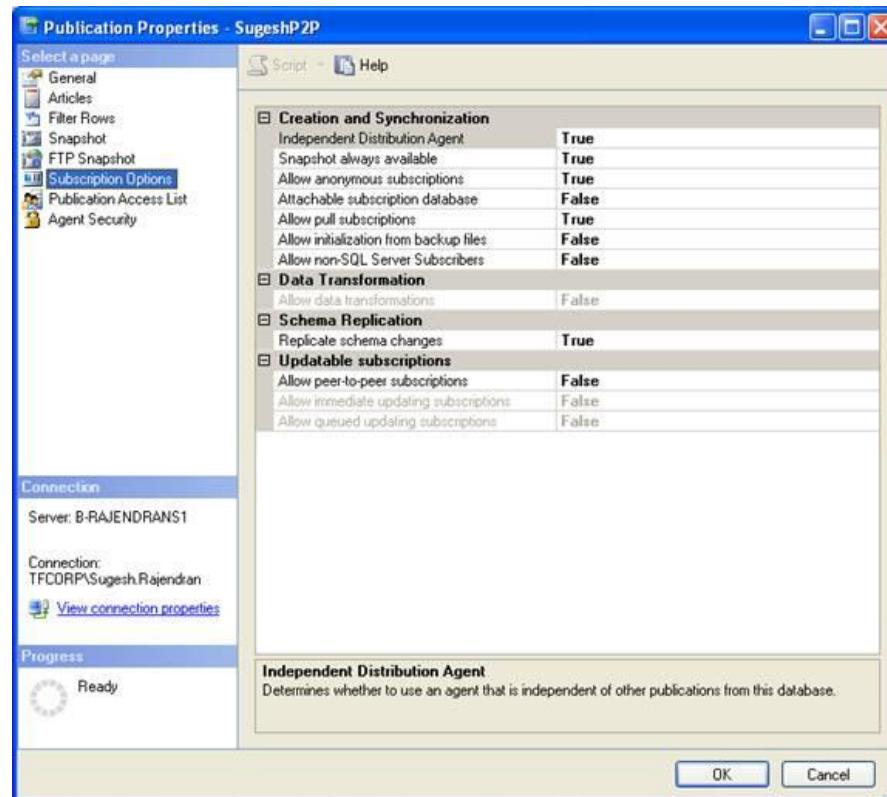


Once you see success in all the steps click close. You have created the publication for this replication instance. The next step is in configuring this publication for P2P replication.



Click on the properties of the publication. Move to the subscriptions options page. You will see an option Allow Peer-to-Peer replication. Change the value from False to True.

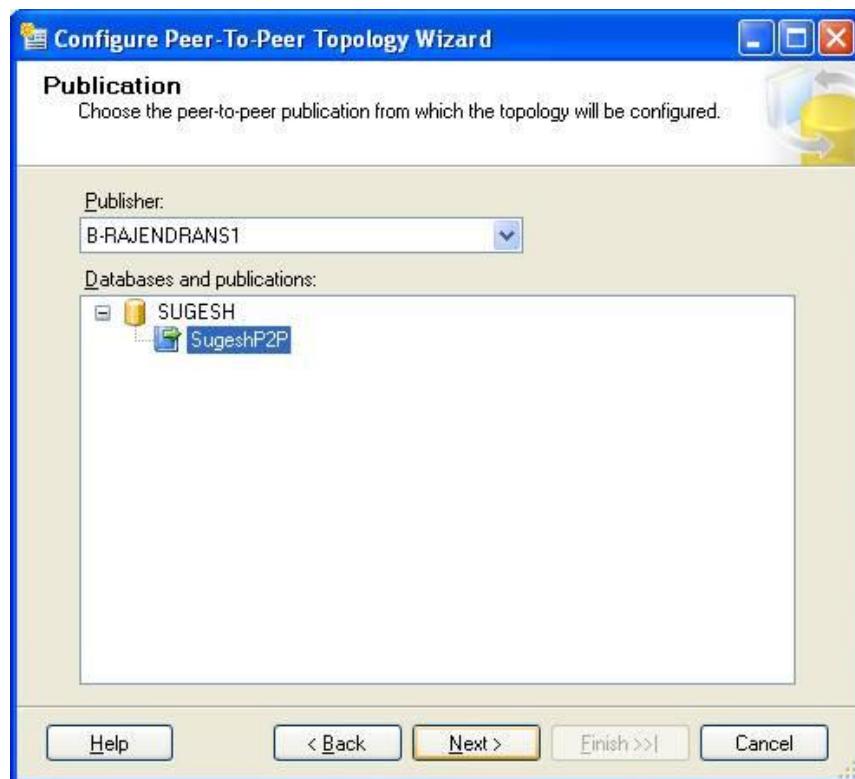




Then Right click the publication and you will see configure peer-to-peer replication. Click that to proceed. Click the publication for which you need to configure the replication instance and click next.



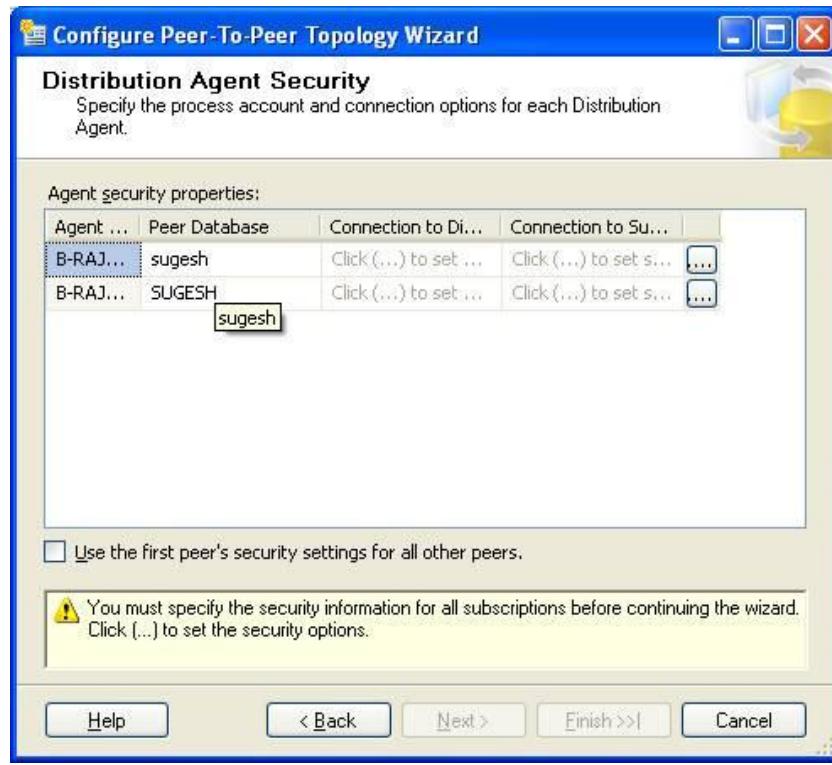
Add the peer instances and click next



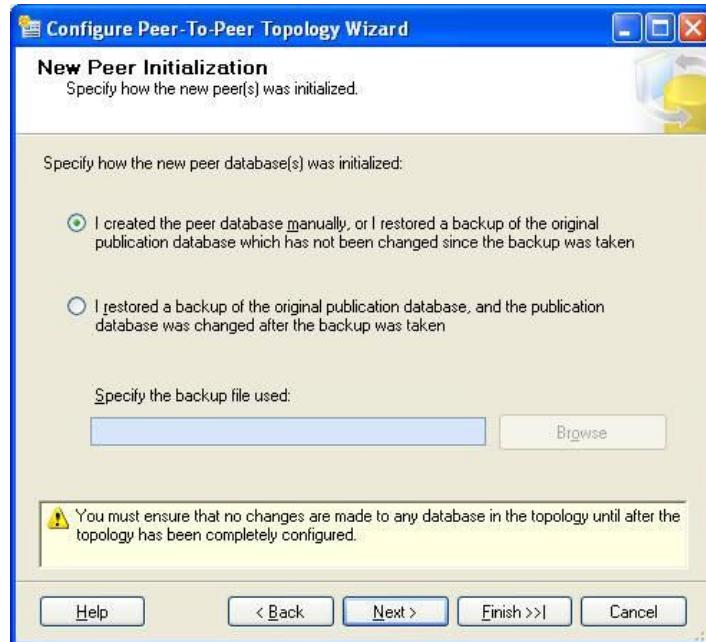
Configure the log reader agent security for the peers and click next.



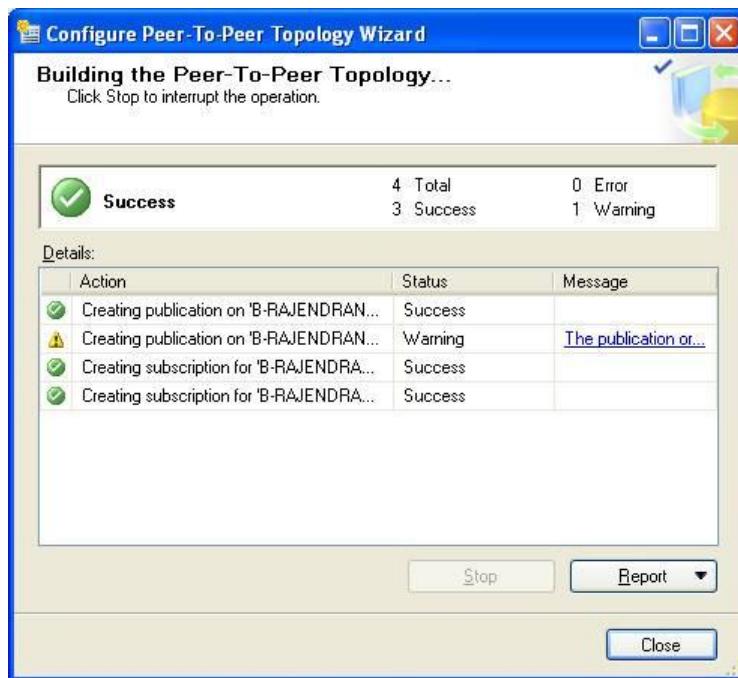
Configure the distributor agent security and click next.



Here you will need to provide the details of how you initialized the database in the peer. You can either create database manually or restore a database. In this case I have restored the database with no changes gone to the database in the publisher. Click next.



Click finish. Check for the errors and warnings in the results screen.



Once all steps have been configured successfully, you are done with configuring P2P replication.

## Best Practices on Replication

- Replication needs should be clearly defined before creating a replication topology. Successful replication can be difficult and requires much pre-planning.
- Ideally, publishers, distributors, and subscribers should be on separate physical hardware.
- Create, document, and test a backup and restore strategy. Restoring replicated databases can be complex and requires much planning and practice.
- Script the replication topology as part of your disaster recovery plan so you can easily recreate your replication topology if needed.
- Use default replication settings, unless you can ensure that a non-default setting will actually improve replication performance or other issues. Be sure that you test all changes to ensure that they are as effective as you expect.
- Fully understand the implications of adding or dropping articles, changing publication properties, and changing schema on published databases, before making any of these changes.
- Periodically, validate data between publishers and subscribers.
- Regularly monitor replication processes and jobs to ensure they are working.
- Regularly monitor replication performance, and performance tune as necessary.
- Add alerts to all replication jobs so you are notified of any job failures.



## SQL Server Clustering

### Introduction

If your mission-critical SQL Server should experience a motherboard failure, how long will it be down? One hour, four hours, a day, or longer? How much will this cost your business in lost sales or productivity? And perhaps even more important to you, what will it do to your stress level?

Being a SQL Server DBA can be demanding and stressful, especially as the success of your company is often a function of your SQL Server's uptime. While we, as DBAs, have some control over the uptime of our SQL Servers, we don't have full control. There is not much we can do if a motherboard fails on a server, other than be prepared.

As you may already be aware, there is one way to help boost your SQL Server's uptime, and that is by clustering SQL Servers. This way, should one SQL Server fail in the cluster, another clustered server will automatically take over, keeping downtime to minutes, instead of hours or more.

The purpose of this article is to introduce you to SQL Server clustering, along with its pros and cons. If you are considering clustering SQL Server to help reduce potential downtime, this article is a good place to start.

### What is Clustering?

Clustering can be best described as a technology that automatically allows one physical server to take over the tasks and responsibilities of another physical server that has failed. The obvious goal behind this, given that all computer hardware and software will eventually fail, is to ensure that users running mission-critical applications will have little or no downtime when such a failure occurs. Downtime can be very expensive, and our goal as DBA is to help reduce it as much as possible.

More specifically, clustering refers to a group of two or more servers (generally called nodes) that work together and represent themselves as a single virtual server to a network. In other words, when a client connects to clustered SQL Servers, it thinks there is only a single SQL Server, not more than one. When one of the nodes fails, its responsibilities are taken over by another server in the cluster, and the end-user notices little, if any differences before, during, and after the failover.

Microsoft added clustering features to its operating system when they introduced Windows NT Server 4.0 Enterprise Edition several years ago. The actual clustering feature was called MSCS (Microsoft Clustering Server). While some brave folks actually put the software into production, I personally avoided it as it was not as dependable as Microsoft led you to believe. Also, about this same time, SQL Server 6.5 Enterprise Edition was released, allowing it to be clustered. This was a very crude attempt at clustering SQL Server that was rarely implemented in the real world.

One very important aspect of clustering that often gets overlooked is that it is not a complete backup system for your applications. It is only one part of a multi-part strategy required to ensure minimum downtime and 100% recoverability.



The main benefits that clustering provides is the ability to recover from failed server hardware (excluding the shared disk) and failed software, such as failed services or a server lockup. It is not designed to protect data, to protect against a shared disk array from failing, to prevent hack attacks, to protect against network failure, or to prevent SQL Server from other potential disasters, such as power outages or acts of God.

Clustering is just one part of an entire strategy needed to help reduce application downtime. You will also need to purchase a shared disk array (more on this later) that offers redundancy, make tape backups, put the server behind a firewall, make sure your network connections have redundancy, use battery backup, and locate the server in a secure facility, among many other steps you can take. So don't think that clustering is all you need for creating a highly available SQL Server. It is just one part.

### What Are the Types of Clustering?

When you decide you want to cluster SQL Server, you have a choice of configuring what is called Active/Active or an Active/Passive cluster. Each has its own pros and cons. Let's look at each, in the context of a two-node SQL Server cluster.

An Active/Active SQL Server cluster means that SQL Server is running on both nodes of a two-way cluster. Each copy of SQL Server acts independently, and users see two different SQL Servers. If one of the SQL Servers in the cluster should fail, then the failed instances of SQL Server will failover to the remaining server. This means that then both instances of SQL Server will be running on one physical server, instead of two.

As you can imagine, if two instances have to run on one physical server, performance can be affected, especially if the server's have not been sized appropriately.

An Active/Passive SQL Server cluster refers to a SQL Server cluster where only one instance of SQL Server is running on one of the physical servers in the cluster, and the other physical server does nothing, other than waiting to takeover should the primary node should fail.

From a performance perspective, this is the better solution. On the other hand, this option makes less productive use of your physical hardware, which means this solution is more expensive.

Personally, I prefer an Active/Passive configuration as it is easier to set up and administer, and overall it will provide better performance. Assuming you have the budget, this is what I recommend.

When the installation process does begin, the setup program recognizes all the nodes, and once you give it the go ahead to install on each one, it does, all automatically. SQL Server 2005 binaries are installed on the local drive of each node, and the system databases are stored on the shared array you designate.

In the next section are the step-by-steps instructions for installing a SQL Server 2005 instance in a cluster. The assumption for this example is that you will be installing this instance in a 2-node active/passive cluster. Even if you will be

installing a 2-node active/active or a multi-node cluster, the steps in this section are virtually the same. The only real difference is that you will have to run SQL Server 2005 setup for every instance you want to install on the cluster, and you will have to specify a different logical drive on the shared array.

## Two/Four-Node Clustering?

SQL Server can be clustered using two nodes (using Windows 2000 Advanced Server), or it can be clustered using more than two nodes (using Windows 2000 Datacenter). Since I don't personally have any experience in three or four node clustering, I won't be discussing it here. But for the most part, what I say about two-node clustering also applies to three- or four-node clustering.

## How Does Clustering Work?

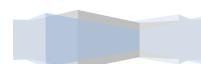
Clustering is a very complex technology, so I will focus here on the big picture. In a two-cluster node, one of the SQL Servers is referred to as the primary node, and the second one is referred to as the secondary node. In an Active/Passive cluster design, SQL Server will run on the primary node, and should the primary node fail, then the secondary node will take over.

When you build a two-node cluster using Windows 2000 Advanced Server and Microsoft Clustering Service, each node must be connected to a shared disk array using either SCSI cables or fibre channel.

Typically, this shared disk array is a stand-alone unit that houses a RAID 5 or RAID 10 disk array. All of the shared data in the cluster must be stored on this disk array, otherwise when a failover occurs, the secondary node in the cluster cannot access it. As I have already mentioned earlier, clustering does not help protect data or the shared disk array that it is stored on. Because of this, it is very important that you select a shared disk array that is very reliable and includes fault tolerance.

Besides both servers being connected to a shared disk array, both nodes of the cluster are also connected to each other via a private network. This private network is used for each node to keep track of the status of the other node. For example, if the primary node experiences a hardware failure, the secondary node will detect this and will automatically initiate a failover.

So how do clients who are accessing SQL Server know what to do when a failover occurs in a cluster? This is the cleverest part of Microsoft Cluster Service. Essentially what happens in a SQL Server cluster is that you assign SQL Server its own virtual name and virtual TCP/IP address. This name and address is shared by both of the servers in the cluster.



Typically, a client will connect to the SQL Server cluster using the virtual name used by the cluster. And as far as a client is concerned, there is only one physical SQL Server, not two. Assuming that the primary node of the SQL Server cluster is the node running SQL Server on an Active/Passive cluster design, then the primary node will respond to the client's requests. But if the primary node fails, and failover to the secondary node occurs, the cluster will still retain the same SQL Server virtual name and TCP/IP address, although now a new physical server will be responding to client's requests.

During the failover period, which can last several minutes (the exact amount of time depends on the number and sizes of the databases on SQL Server, and how active they are), clients will be unable to access SQL Server, so there is a small amount of downtime when failover occurs.

How the client software reacts to the failover process depends on the software. Some software will just wait the failover out, and when the failover has completed, it will continue just as nothing had happened. Some software will present a message box on the screen, describing a lost connection. Other client software will not know what to do, and users may have to exit, and then reload the client before they can access SQL Server again.

As part of the testing process when implementing a SQL Server cluster, it is important to find out how all of the client software that connects to SQL Server reacts to a failover. This way, you can inform your users of what to expect, so they are better able to deal with it when it does occur.

Once a failover occurs, you will want to find out what caused the failover, and then take the necessary action and correct the problem. Once the problem has been fixed, the next step is to failover SQL Server back to the primary node from the secondary node. You can schedule to do this anytime, preferably when user activity is light on the system.



### What are the Pros and Cons of Clustering?

Implementing SQL Server clustering is a big decision, and one fraught with many gochas.

Before you undertake such a large and important project, you will want to carefully evaluate the pros and cons of clustering, which include, but are not limited to:

#### *Pros of SQL Server Clustering*

- Reduces downtime to a bare minimum.
- Permits an automatic response to a failed server or software. No human intervention is required.
- It allows you to perform upgrades without forcing users off the system for extended periods of time.
- It allows you to reduce downtime due to routine server, network, or database maintenance.
- Clustering doesn't require any servers to be renamed. So when failover occurs, it is relatively transparent to end-users.
- Failing back is quick, and can be done whenever the primary server is fixed and put back on-line.
- In some cases, clustering can be used to increase the scalability of an application. For example, if a current cluster is getting too busy, another server could be added to the cluster to expand the resources and help boost the performance of the application.

#### *Cons of Clustering*

- More expensive than other failover alternatives, such as log shipping or stand-by servers.
- Requires more set up time than other alternatives.
- Requires more on-going maintenance than other alternatives.
- Requires more experienced DBAs and network administrators.



## Clustering SQL Server

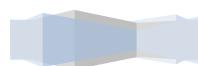
To begin the installation, run Setup.exe. After an introductory screen, you will get the first install dialog box as shown in the figure below.



The Installing Prerequisites dialog box lists the prerequisites that need to be installed before installation of SQL Server 2005 can begin. The number of components may vary from the above figure, depending on what you have already installed on your nodes. What is interesting to note here is that these prerequisite components will only be installed immediately on the active node. They will be installed on the passive node later during the installation process. This is done automatically and you don't have to worry about it.

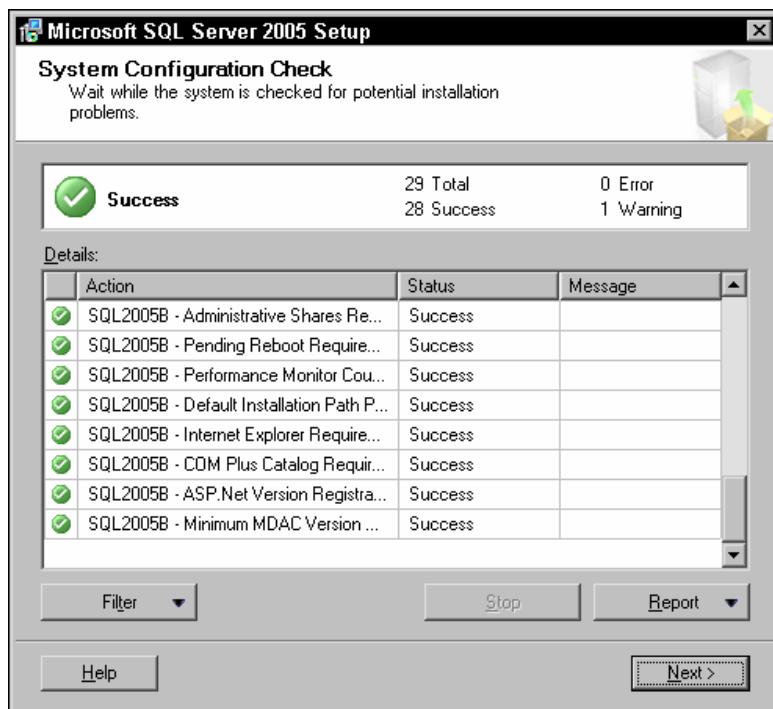
Click Install to install these components. When completed, you will get a dialog box telling you that they were installed successfully, and then you can click Next to proceed. On occasion, I have seen these components fail to install correctly. If this happens, you will have to troubleshoot the installation. Generally speaking, try rebooting both nodes of the cluster and try installing them again. This often fixes whatever caused the first setup try to fail.

Once the prerequisite components have been successfully installed, the SQL Server Installation Wizard launches, as you can see in the figure below.





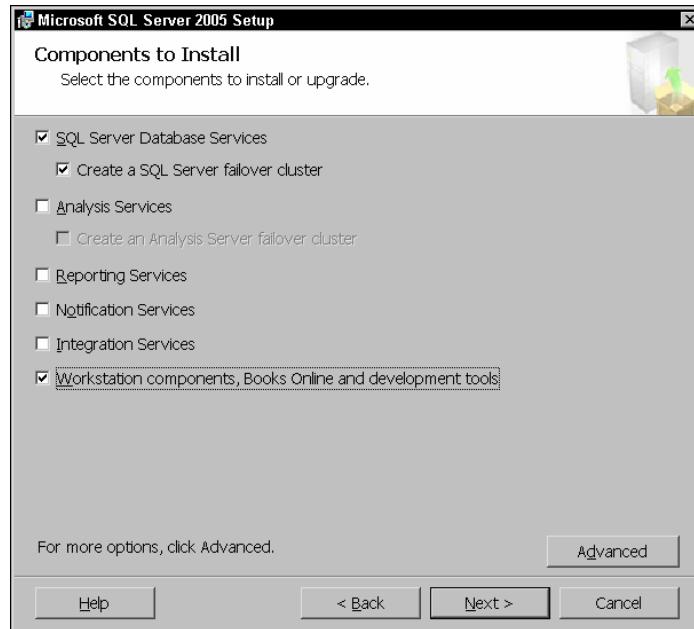
Click Next to proceed.



The next step is for the SQL Server Installation Wizard to perform a System Configuration Check. This is very similar to the check that was performed with clustering services when you installed [Windows Server 2003 Clustering](#). Ideally, you want all checks to be successful, with a green icon. If you get any yellow warning or red error icons, then you need to find out what the problem is, and correct it before proceeding. In some cases, yellow warning icons can be ignored, but red error icons cannot. If you have any yellow or red icons, you may have to abort the setup process, fix the problem, then restart the setup process. Assuming all is well, click Next to proceed.

The next dialog box is Registration, where you enter your company name and license key, if applicable.

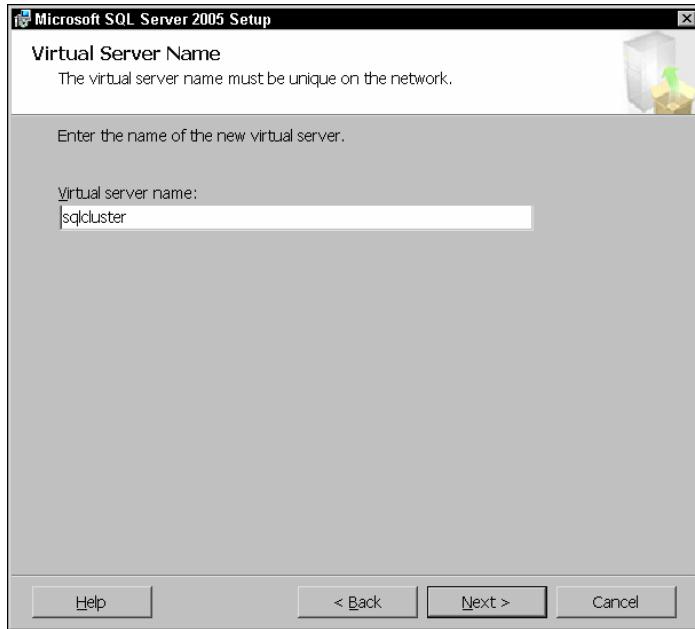
Next, you must select the SQL Server 2005 components to install. See below.



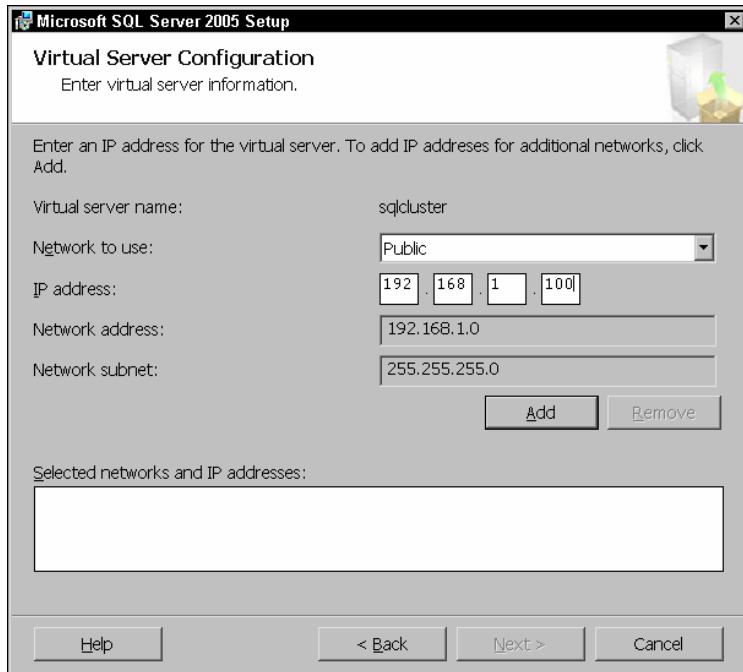
I want to point out the options to "Create a SQL Server failover cluster" and to "Create an Analysis Server failover cluster" (currently grayed out). Since we are creating a SQL Server 2005 cluster, you must select the "Create a SQL Server failover cluster." If you are going to install Analysis Services (not covered in this example) then you must select "Create an Analysis Server failover cluster." Once you have selected all the components you need to include, click Next.



As with any install of SQL Server 2005, the next step is to select the name of the instance to be installed. You can choose between a default instance and a named instance. Click Next to proceed.



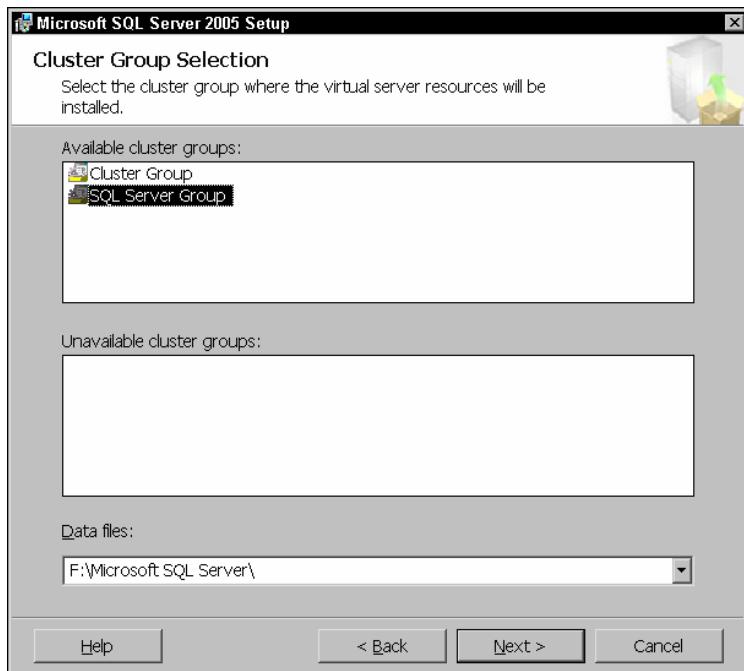
Now, here is a very important step. This is when you enter the name of the virtual SQL Server 2005 instance you are currently installing. This is the name that clients will use to connect to this instance. Ideally, you have already selected a name to use that makes the most sense to your organization. Click Next to proceed. If you ever need to change this virtual name, you will have to uninstall and then reinstall SQL Server 2005 clustering.



This is also a very important step. This is where you enter the virtual IP address for this instance of SQL Server 2005. Like the cluster virtual name, it is used by clients to connect to this instance of SQL Server 2005. The IP address must belong to the same subnet as the IP addresses used by all of the nodes.

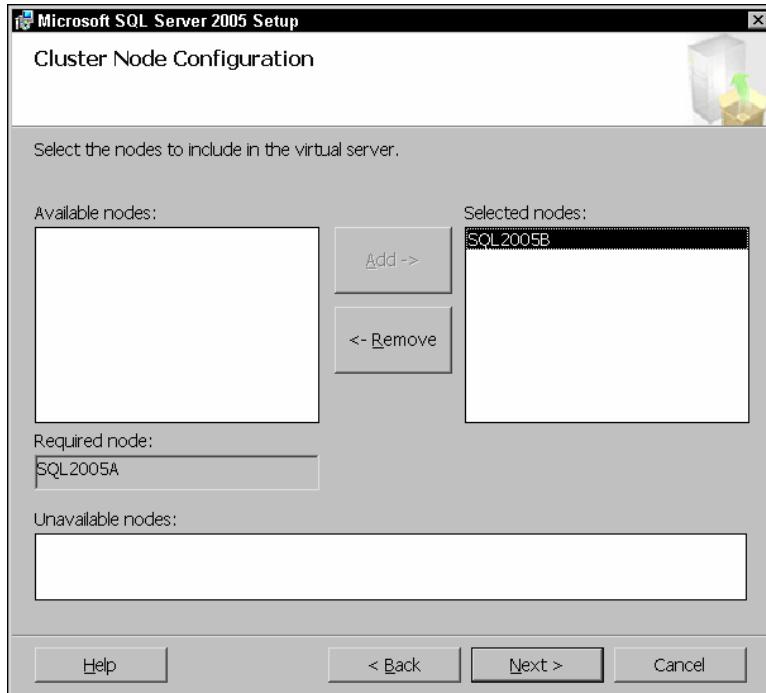
In addition, in this dialog box you must select the network to be used for the public network—the network used by the clients to connect to this instance of SQL Server 2005. All of the available networks will be listed in the drop-down box next to Network to use. If you have named the public and private networks Public and Private, respectively, it will be very easy for you to select the correct network, as I have above.

Once you have entered the IP address and selected the public network, click on Add, so that the information you just selected is in the Selected networks and IP addresses box. Then click Next.



In this dialog box, select the SQL Server Group as the group where you want to create the SQL Server resources. In addition, be sure that the Data files will be created on the correct logical drive of the shared array using the folder name you choose. Click Next to proceed.

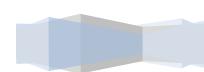


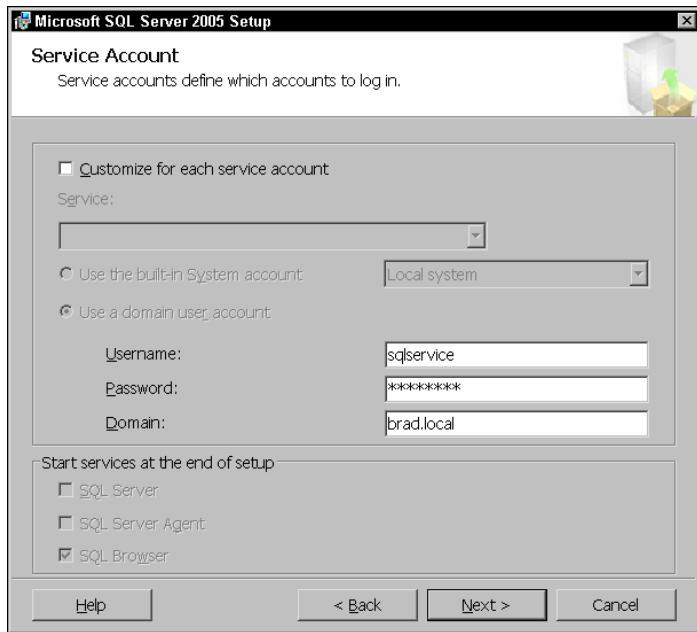


Now, you specify which nodes you want to install this instance of SQL Server on. Because our example is for only two nodes, the default setting works for us. Notice that under Required node is SQL2005A, which is the name of the physical node where we are running the setup program. And under Selected nodes is SQL2005B, the second physical node in our 2-node cluster. Click Next to proceed.

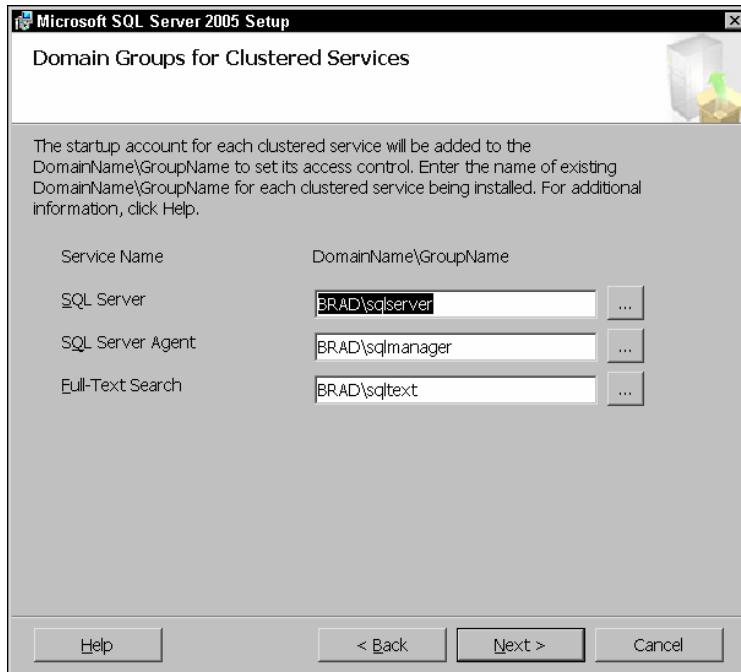


In this dialog box, we must select an account (with password) that has administrative rights on all of the nodes where we want to install this instance of SQL Server 2005. This can be any domain account that is a local administrator of all the nodes. Click Next to proceed.





The Service Account dialog box is identical to the one you see when you install SQL Server 2005 on a non-cluster, and it is configured the same. Click Next to proceed.

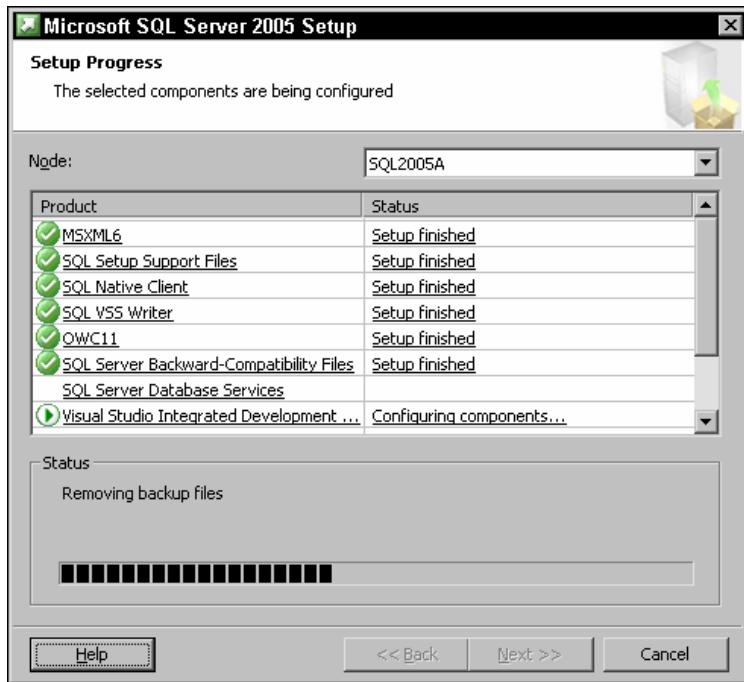


In this dialog box, you must select pre-existing global domain groups that are used to contain the startup account for each clustered service. You can choose to add all three services to the same global domain group, or to create separate global domain groups, one for each service, as has been done above. Once you have selected appropriate domain groups, click Next to proceed.

The next four dialog boxes of the Installation Wizard, not shown here, are the same as for any other installation of SQL Server 2005. After you have completed these



steps, the installation of this instance of SQL Server 2005 begins, and you see the following dialog box.



The installation process will take some time as it is installing the binaries on both nodes of the cluster, and installing the system data files on the shared array. The Setup Progress step shows the status of the first node's install. If you want to see the status of the second node's install, you can change the drop-down box next to Node to the second node and watch its progress.

As the installation proceeds, you will want to see all green icons next to each installation step. If any step should fail, then the entire installation process will need to be rolled back, any problems fixed, and SQL Server 2005 installed fresh. In most cases, cancelling a bad installation will uninstall what has already been installed, but not always.

Sometimes, if the installation breaks, it just dies and a rollback of what has been done so far will not occur. If this is the case you can either choose to reinstall on top of the existing bad install (which often does not work), manually uninstall the failed installation (check Microsoft's Web site for assistance in this area), or rebuild your cluster from scratch (starting with the operating system).

If the install was a success, you will see a final dialog box, where you can click Finish. SQL Server 2005 clustering had now been successfully installed on the two cluster nodes.

### **Clustering Analysis Services**

SQL Server 2005 Analysis Services can be clustered just like SQL Server 2005, and in fact, is installed using the same setup program used to install SQL Server 2005. Below are some points to keep in mind if you should decide to cluster SQL Server 2005 Analysis Services.



- SQL Server 2005 Analysis Services can be installed by itself, or with SQL Server 2008. Because some of the features of Analysis Services require components of the SQL Server 2005 database engine, it is generally a good idea to install both of them in your cluster.
- SQL Server 2005 Analysis Services is installed using the same setup program as SQL Server 2005. When running the setup program, you select, or not select, Analysis Services to be installed in the "Components to Install" screen.
- Because SQL Server 2005 Analysis Services needs to store program files, data files, and shared files, you will have to specify the location of your shared array, where they will reside. These files must reside on a shared array if you want Analysis Services to run after a failover. To specify the location of the shared array, you must select the "Advanced" button from the "Components to Install" screen in the setup wizard.

Other than the above, installing SQL Server 2005 Analysis Services in a cluster is virtually identical to installing SQL Server 2005 in a cluster.

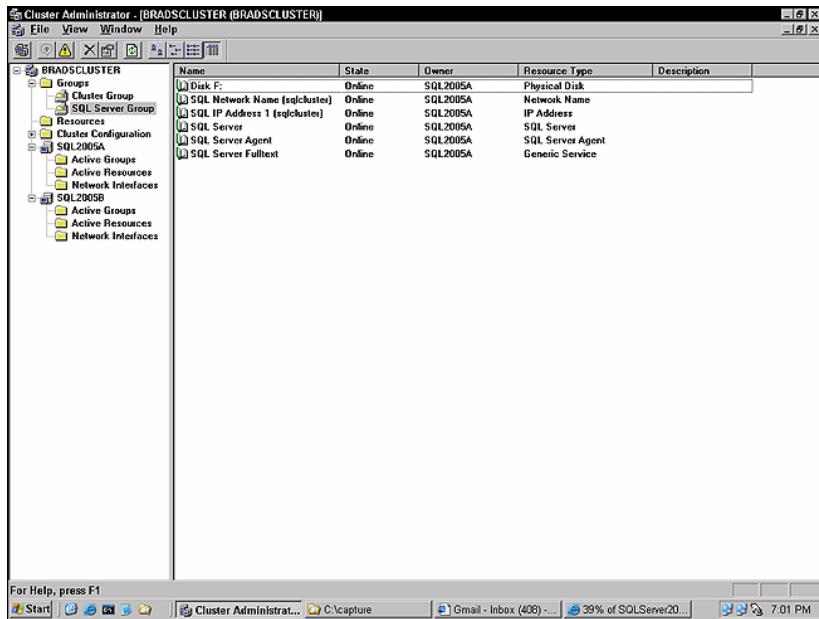
### **Installing the Service Pack and Hot Fixes**

Once you have installed SQL Server 2005 clustering, your next step is to install the latest SQL Server 2005 service pack and hot fixes, which can be downloaded from Microsoft's Web site. Installing a service pack or hot fix is fairly straightforward because they are cluster-aware. Once the service pack or hot fix setup program is started, it detects that you have a cluster and will upgrade all nodes simultaneously. Once setup is complete, you may need to reboot your servers and failover the nodes. Generally, once I have run the service pack, I like to reboot the active node first. Once it has rebooted, then I reboot the passive node. This way, failover and fallback is automatic.

### **Checking the SQL Server 2005 Installation from Cluster Administrator**

Once an instance of SQL Server 2005 clustering is installed, you can view its cluster resources by going to Cluster Administrator and opening up the SQL Server Group resource, as shown below.





This figure shows the cluster resources for the SQL Server 2005 cluster we just built. We see all of the names of the resources, their state, and which node the resources are running on. As I have already mentioned, Cluster Administrator is a great tool for seeing if all the resources are up and running and which node is the current active node.

Here is a brief rundown on each of the SQL Server 2005 cluster resources:

- Disk F: This is the shared disk array where the SQL Server data files and logs are stored.
- SQL Network Name (sqlcluster): This is the virtual SQL Server name used by clients to connect to this clustered instance of SQL Server 2005. The name "sqlcluster" is the name I have assigned this cluster instance, and will not be the same as your cluster, unless you name yours the same as mine.
- SQL IP Address (sqlcluster): This is the virtual SQL Server IP address used by clients to connect to this clustered instance of SQL Server 2005. Again, the name "sqlcluster" is the name of the virtual server, and is the one I have used for this cluster. Your name will most likely be different.
- SQL Server: This is the SQL Server service.
- SQL Server Agent: This is the SQL Server Agent service.
- SQL Server FullText: This is the SQL Server FullText service. Even though you may not use this service, it is automatically installed as a cluster resource.

## Installing Clustering on Multiple Nodes

When I talk about installing SQL Server 2005 clustering on multiple nodes, I am referring to two different scenarios. They include:

A 2-node active/active cluster, where you are running a single instance of SQL Server 2008 on each node. If one of the two active nodes should fail, then the failed active instance would failover to the other active node, with the end result that you are running two active instances of SQL Server 2005 on the same physical node.

A 3-node to 8-node SQL Server 2005 cluster where one of the nodes is designated as a passive node for failover purposes, and the rest are active nodes, with each one running a single instance of SQL Server 2005. Should any of the active nodes fail, then the failover would go to the designated passive node to run on.

Installing multiple instances of SQL Server in a cluster is virtually identical to installing a SQL Server 2005 cluster as described above. In general, here is what you need to know about installing multiple instances of SQL Server 2005 in a cluster:

- All of the nodes in the cluster should have identical hardware and software, and be configured identically.
- You will need a hub or switch for the private network connection among the nodes.
- You will need a separate shared drive for each instance of SQL Server 2005 installed. These are besides the shared drive required for the quorum. You only need one quorum drive for your cluster.
- You will need distinct virtual names and IP addresses for each SQL Server 2005 instance.
- Each SQL Server 2005 instance must be in its own distinct resource group in the cluster.
- You will need to run SQL Server 2005 Setup for each separate instance of SQL Server 2005 you want in the cluster.
- You will need to configure each active node, should a failover occur, to failover to the designated passive node.

Because running more than a single instance of SQL Server 2005 on a cluster is complex, I highly recommend that you build this cluster from scratch, and test it thoroughly before putting it into production.

### **Test, Test, and Test Again**

Once you have installed SQL Server 2005 clustering on your nodes, you need to thoroughly test the installation, just as you did after first installing Windows Server 2003 Clustering. But not only do you want to test SQL Server 2005 clustering, you also want to test how your clients "react" to failovers. Because of this, the following testing section is very similar to the one you previously read, but has been modified to meet the more complex needs of the additional client testing you need to do.

Below are a series of tests you can perform to verify that your SQL Server 2005 cluster, and their clients, works properly during failover situations. After you perform each test, verify if you get the expected results (a successful failover), and also be sure you check the Windows log files for any possible problems. If you find a problem during one test, resolve it before proceeding to the next test.

### **Preparing for the Testing**

Identify a workstation that has Cluster Administrator on it, and use this copy of Cluster Administrator for interacting with your cluster during testing.

Now for the hard part. Essentially, you need to test how each client will be accessing your SQL Server 2005 cluster. In other words, you want to test to see what will happen to each client should a failover occur. Some client software deals with clustering failovers automatically, while others choke and die. The only way to know

272



for sure is to test them. Of course, to test your applications, you will have to install the appropriate databases on the cluster before you begin.

To test them, you must first identify all the client applications, which might be one product, or a dozen products. Each of these products will have to be configured to access the virtual server name (and IP address) on the new SQL Server instance. In addition, for the clients to work, you will have to have the appropriate databases restored or installed on the new cluster. Yes, this is a lot of work. But this is necessary if you want a highly available clustering solution you can count on.

Once you have at least one copy of each of your client applications connected to the SQL Server 2005 instance, you are ready for testing. Keep in mind, that while testing, you are testing multiple things, including the Windows Server 2003 cluster, the SQL Server 2005 cluster, and the client applications.

### **Move Groups between Nodes**

The easiest test to perform is to use Cluster Administrator to manually move the cluster and SQL Server resource groups from the active node to a passive node, and then back again. To do this, right-click on a resource group and then select Move Group. This will initiate the move of the resources groups from your active node to the designated passive node.

Once this happens, check Cluster Administrator and each of the client applications. Each should continue to operate as if no failover had occurred. Cluster Administrator should pass this test easily. The clients are another story. You will need to check each client to see if they continue to work as before. If not, you need to determine why not, which is not always easy. Most clients that stop working after a failover will reconnect if you exit and restart the client.

Once the group has been successfully moved from the active node to a passive node, then use the same procedure above to move the group back to the original node. And as before, check Cluster Administrator, the clients, and the Event Logs to see if there were any problems. If you have Cluster Service or SQL Server 2005 problems due to the test failover, you need to resolve them now before proceeding. If you have a client problem, you can continue with your testing and try to resolve them later. In most cases, if a client fails this first test, it will fail all of the tests.

### **Manually Initiate a Failover in Cluster Administrator**

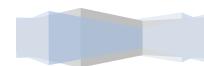
This test is also performed from Cluster Administrator. Select any of the resources found in the SQL Server Group resource group (not the group itself), right-click on it, and select Initiate Failure. Because the cluster service always tries to recover up to three times from a failover, if it can, you will have to select this option four times before a test failover is initiated.

As above, after the first failover, check for any problems, then failback using the same procedure. Then check again for problems.

### **Manually Failover Nodes by Turning Them Off**

273

Turn off the active node. Once this happens, watch the failover in Cluster Administrator and the clients. As before, check for any problems. Next, turn on the node and wait until it boots back up successfully. Then turn off the now current



active node by turning it off hard. And again, watch the failover in Cluster Administrator and the clients, and check for problems. Turn the node back on when done.

### **Manually Failover Nodes by Breaking the Public Network Connections**

Unplug the public network connection from the active node. This will cause a failover to a passive node, which you can watch in Cluster Administrator and the clients. Check for any problems. Now, plug the public network connection back into the server. And unplug the public network connection from the now active node. This will cause a failover to the current passive node, which you can watch in Cluster Administrator. And again, watch the failover in Cluster Administrator and the clients, and check for problems. Once the testing is complete, plug the network connection back into the server.

### **Manually Failover Nodes by Breaking the Shared Array Connection**

From the active node, remove the shared array connection. This will cause a failover, which you can watch in Cluster Administrator and the clients. Check for any problems. Next, reconnect the broken connection from the now active node, and remove the shared array connection. Watch the failover in Cluster Administrator. And again, watch the failover in Cluster Administrator and the clients, and check for problems. When done, reconnect the broken connection.

If you pass all of these tests the first time, it would almost be a miracle. But I do believe in miracles. If you run into problems, you have to figure them out.

### **Ready for Production**

Once you have successfully tested your production SQL Server 2005 cluster, you are ready to go into production. If you have time, you might want to consider running the cluster in test mode for a while, "playing" with it to learn more about how it works under various conditions. But even if you don't have any extra time to "play" with your cluster, it should be ready to go into production. Now pat yourself on the back for a job well done.

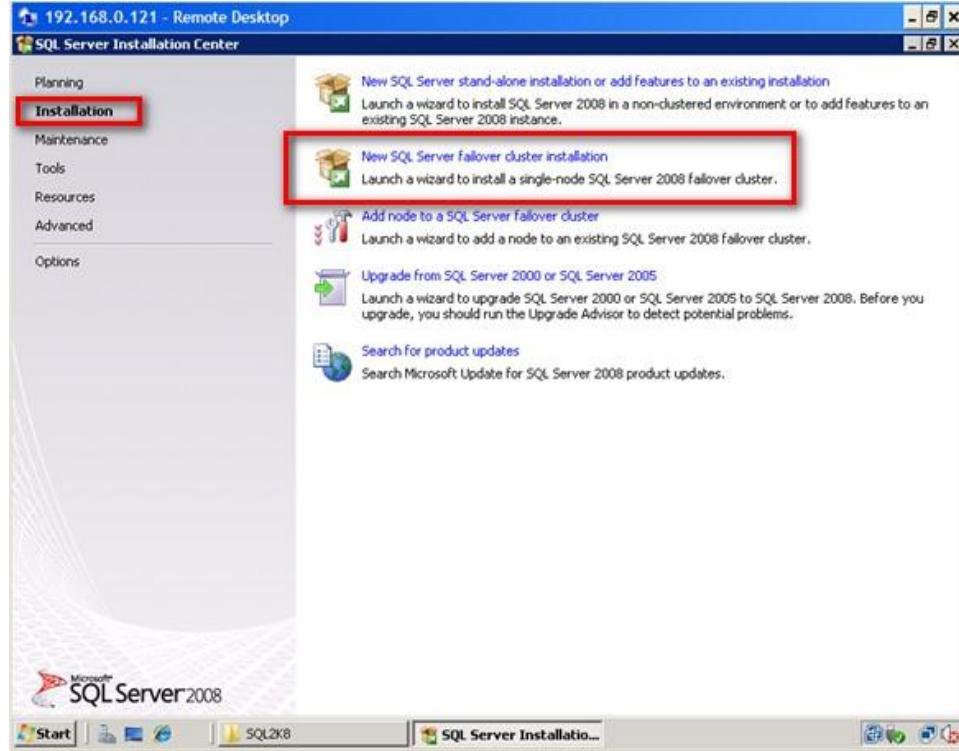
### **Installing SQL Server 2008 on a Windows Server 2008 cluster**

There are two options to install SQL Server 2008 on a cluster. The first one is by using the integrated failover cluster install with Add Node option and the second one is the Advanced/Enterprise installation option. The process outlined below will take into account the first option.

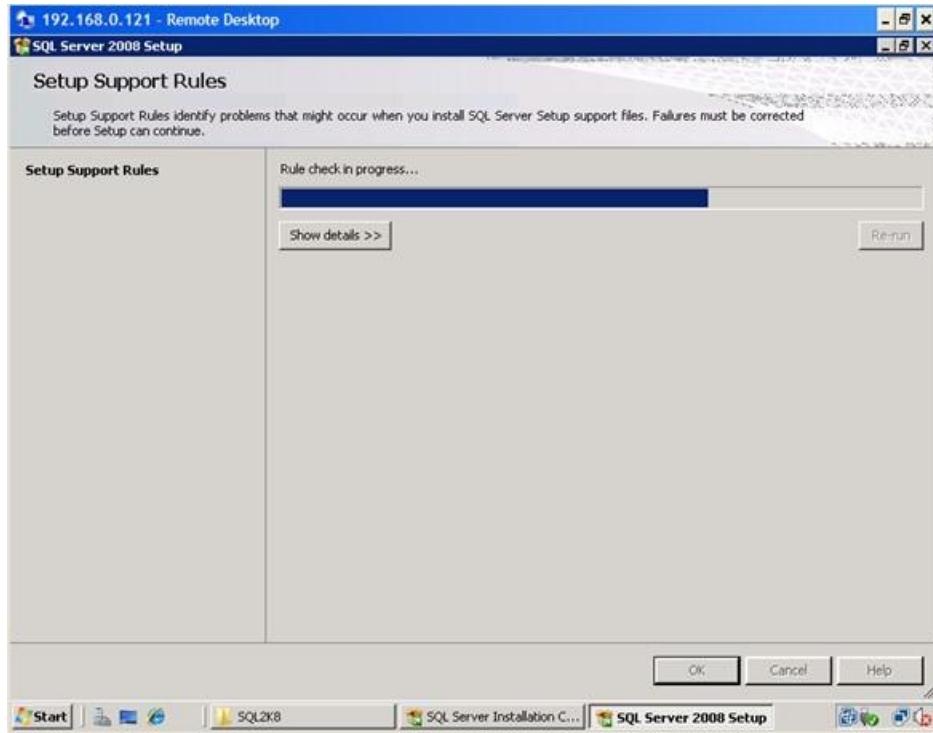
To install SQL Server 2008:

1. Run **setup.exe** from the installation media to launch **SQL Server Installation Center**. Click on the **Installation** link on the left-hand side
2. Click the **New SQL Server failover cluster installation** link. This will run the **SQL Server 2008 Setup** wizard

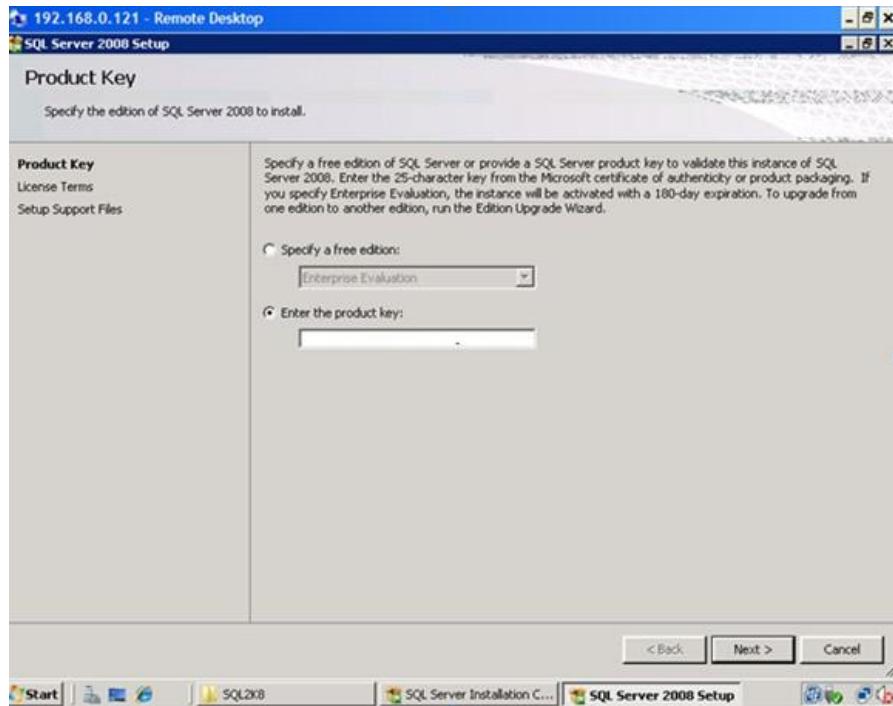




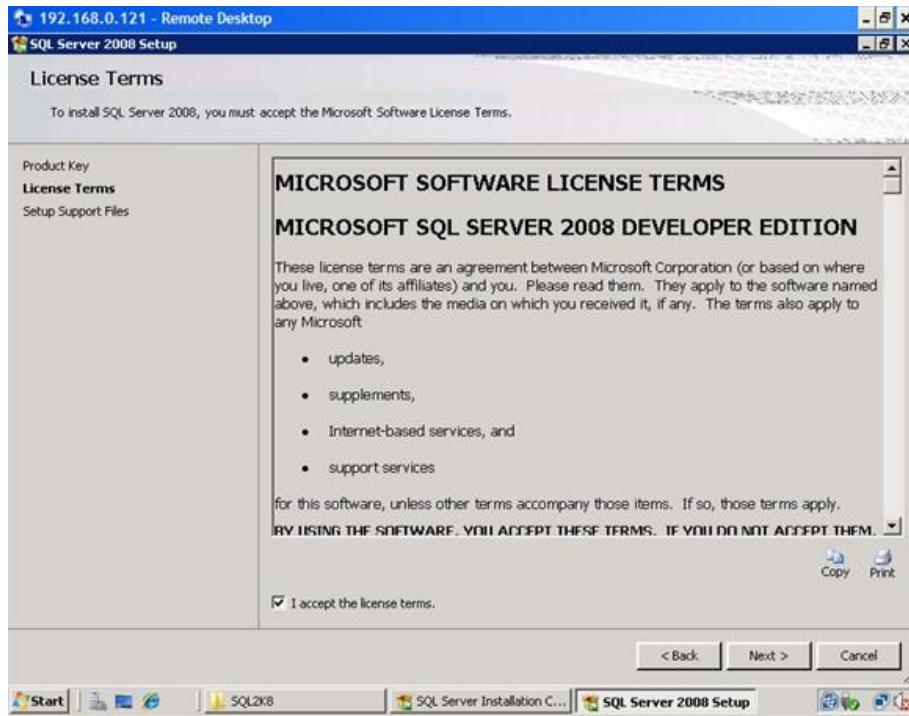
3. In the **Setup Support Rules** dialog box, validate that the checks return successful results and click **Next**.



4. In the **Product Key** dialog box, enter the product key that came with your installation media and click **Next**.



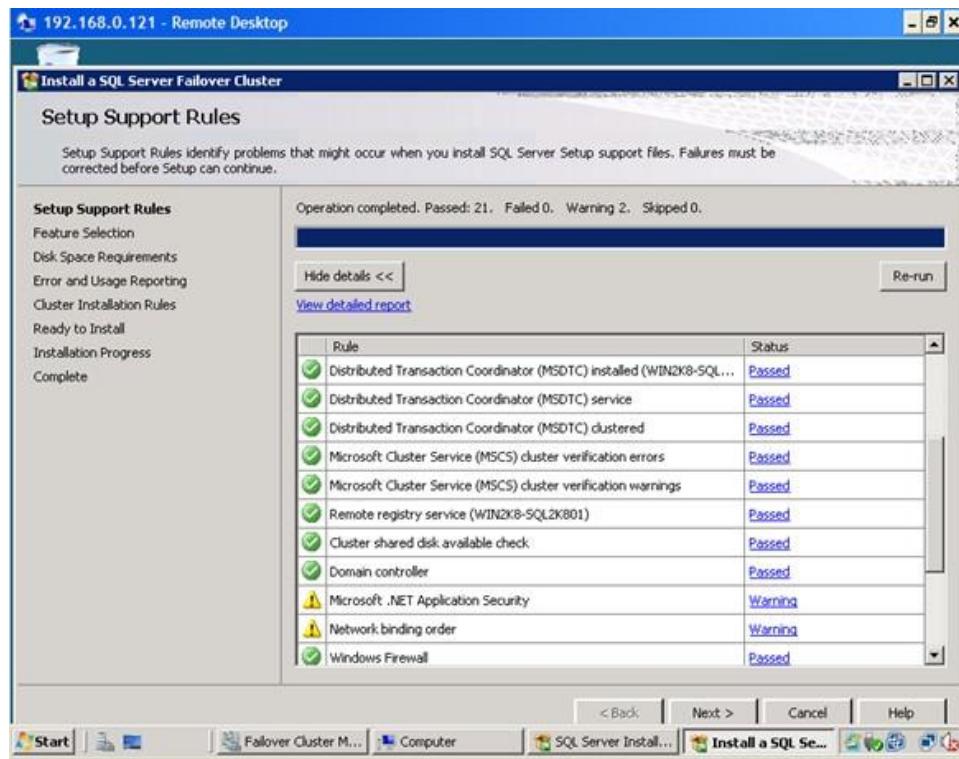
5. In the **License Terms** dialog box, click the **I accept the license terms** check box and click **Next**. You probably haven't read one of these, but if you feel inclined go for it.



6. In the **Setup Support Rules** dialog box, click **Install**. Validate that the checks return successful results. If the checks returned a few warnings, make

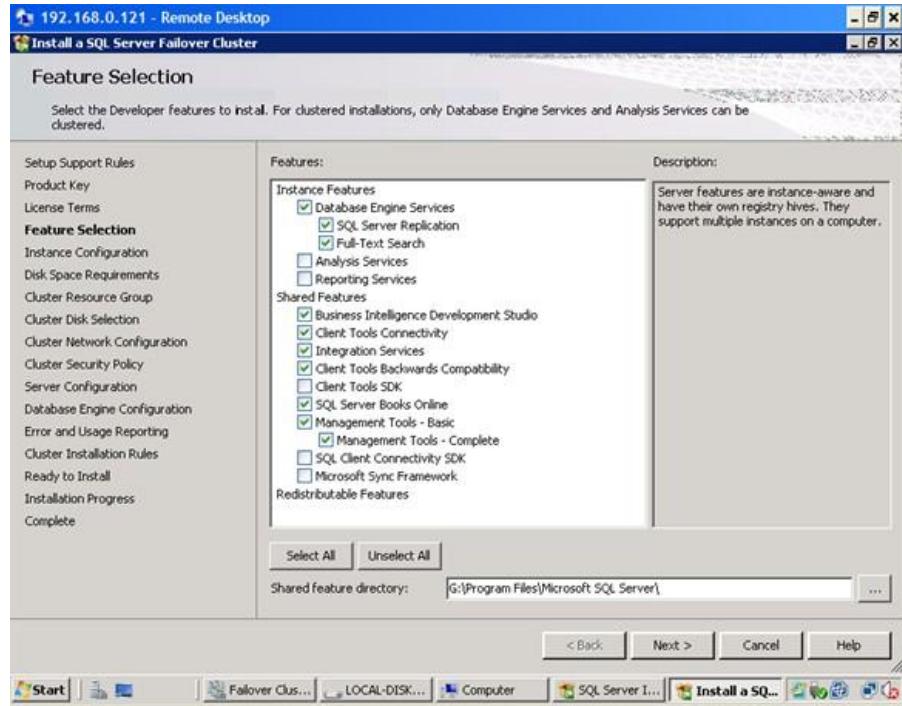
sure you fix them before proceeding with the installation. An example of this is the **Network binding order**. The public network cards should be first on both nodes. Also, you can disable NETBIOS and DNS registration on the network cards to avoid network overhead. Be sure to check your binding order as well.

For the Windows Firewall, make sure that you open the appropriate port number on which SQL Server will communicate. You can do this after the installation. Alternatively, you can disable Windows Firewall during the installation and enable it later with the proper configuration. Click **Next** to proceed.



7. In the **Feature Selection** dialog box, select only the components that you want installed. For the **Shared feature directory**, you can keep the default path if you have sufficient disk space on your C:\ drive or anywhere that is a **local disk** as this will be used by the SQL Server installation process later on. The directory for the clustered database engine will be different. Click **Next**.



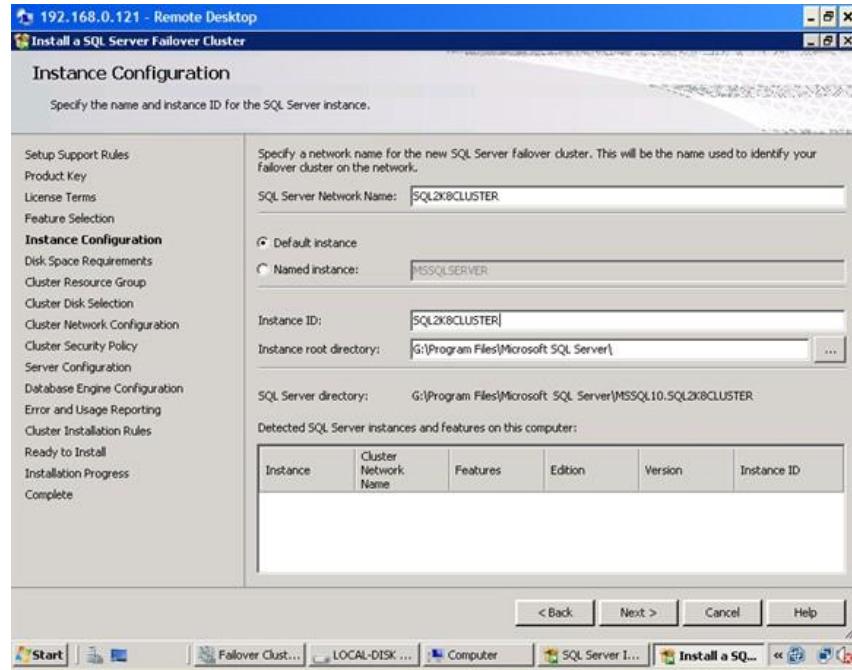


8. In the **Instance Configuration** dialog box, enter the SQL Server Network Name. This is the name that will be available on the network for the clients. This will vary depending on your selection of whether it is a default or named instance. In this example, default instance is selected.

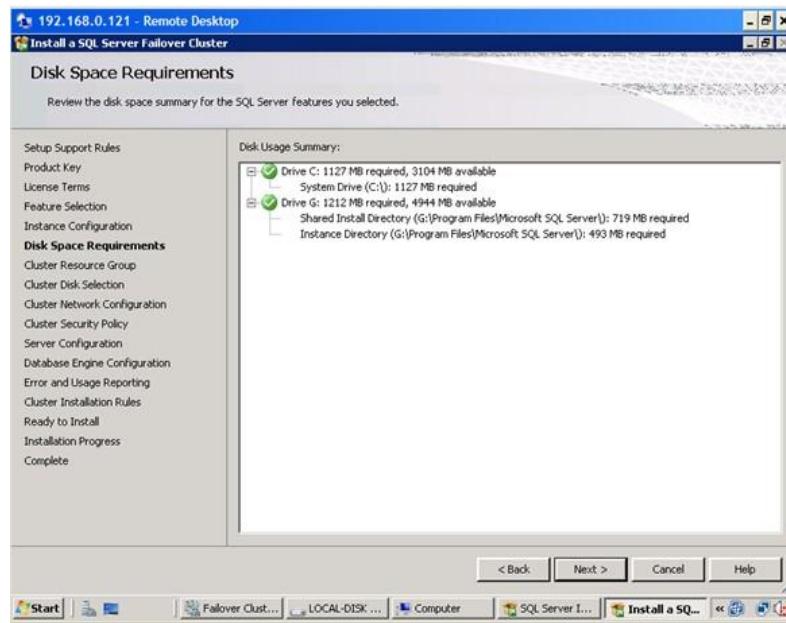
A couple of things need highlighting in this section. By default, the instance name is used as the **Instance ID**. This is used to identify installation directories and registry keys for your instance of SQL Server and is helpful when you want to run multiple instances in a cluster. This is the case for default instances and named instances. For a default instance, the instance name and instance ID would be **MSSQLSERVER**. To use a non-default instance ID, you should select the **Instance ID** box and specify a value.

The section on **Detected SQL Server instances and features on this computer** would make sense if there are other SQL Server instances running on your server.

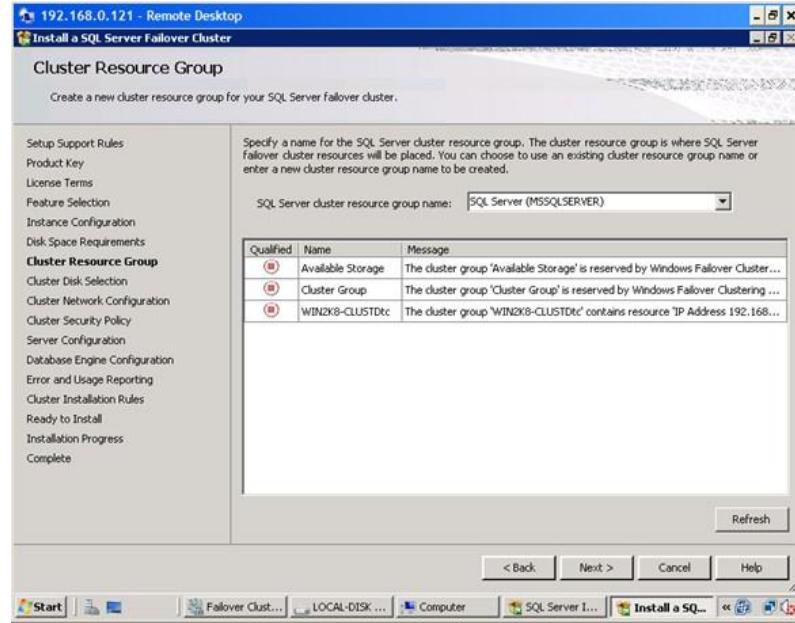




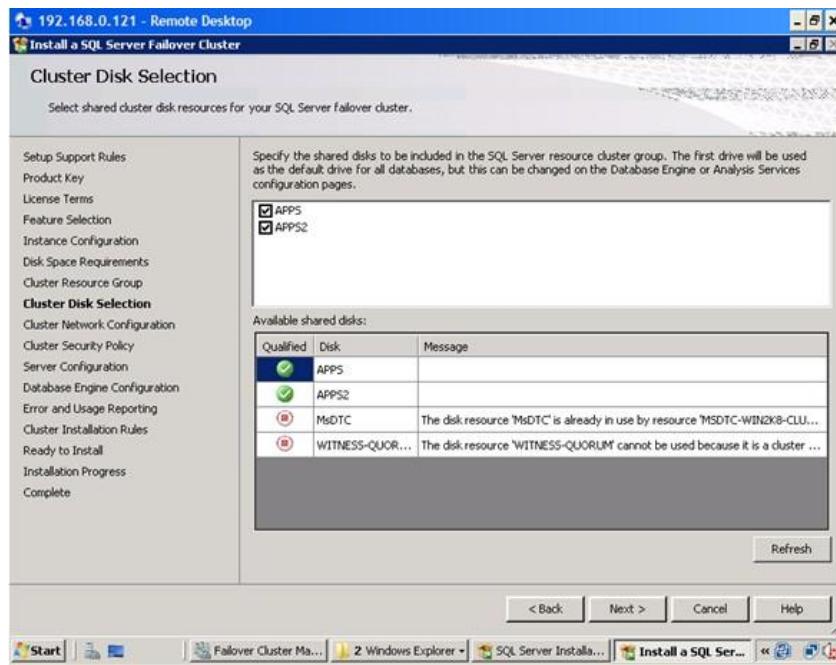
9. In the **Disk Space Requirements** dialog box, check that you have enough space on your **local disks** to install the SQL Server 2008 binaries and click **Next**.



10. In the **Cluster Resource Group** dialog box, check the resources available on your Windows Server 2008 cluster. This will tell you that a new Resource Group will be created on your cluster for SQL Server. To specify the SQL Server cluster resource group name, you can either use the drop-down box to specify an existing group to use or type the name of a new group to create it. Click **Next**.



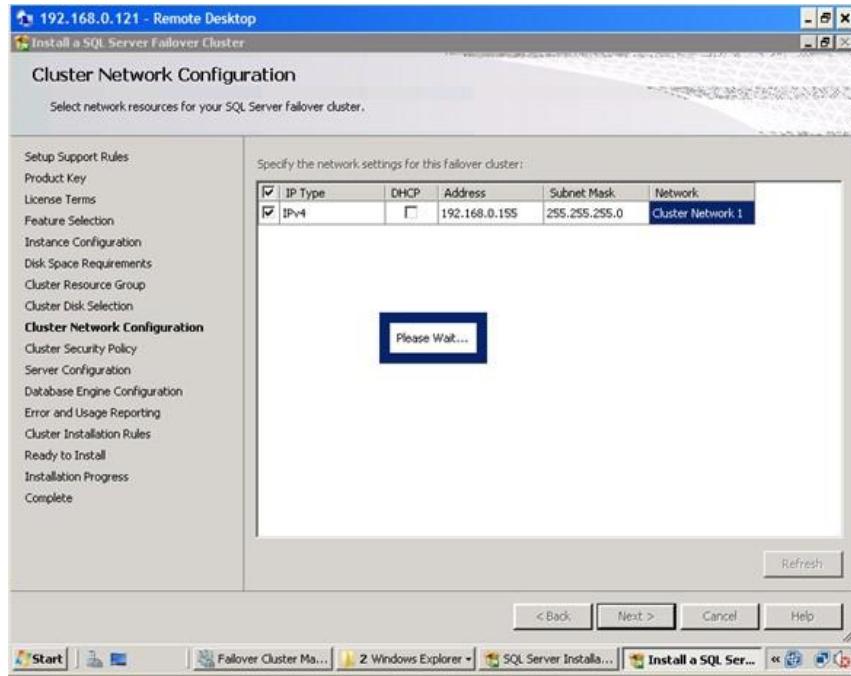
11. In the **Cluster Disk Selection** dialog box, select the available disk groups that are on the cluster for SQL Server 2008 to use. In this example, two clustered disk groups – **APPS** and **APPS2** – have been selected to be used by SQL Server 2008. I will be using one disk resource for the system databases while the other one for the user databases. Click **Next**.



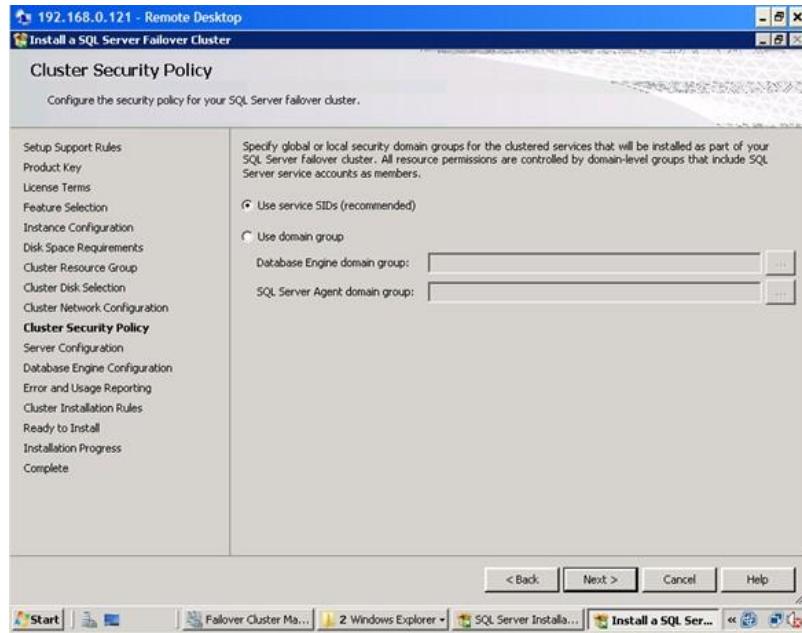
12. In the **Cluster Network Configuration** dialog box, enter the IP address and subnet mask that your SQL Server 2008 cluster will use. Deselect the checkbox under the **DHCP** column as you will be using static IP addresses. If you have not disabled your IPv6 adapters and protocols, it would be better to uncheck the row for **IPv6**

280



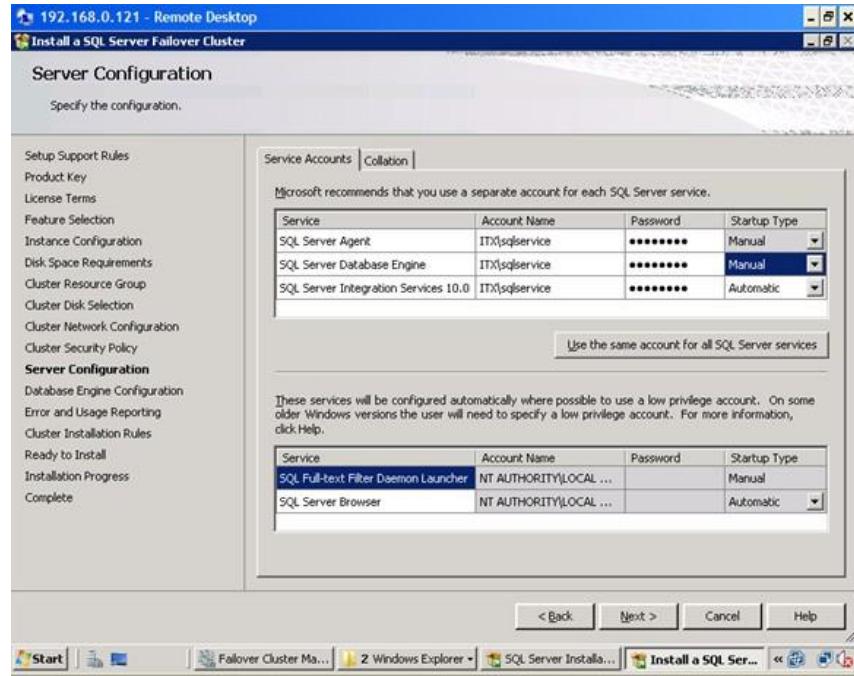


13. In the **Cluster Security Policy** dialog box, accept the default value of **Use service SIDs (recommended)**. In Windows Server 2003, we specify domain groups for all SQL Server services but in Windows Server 2008, this is the recommended option.

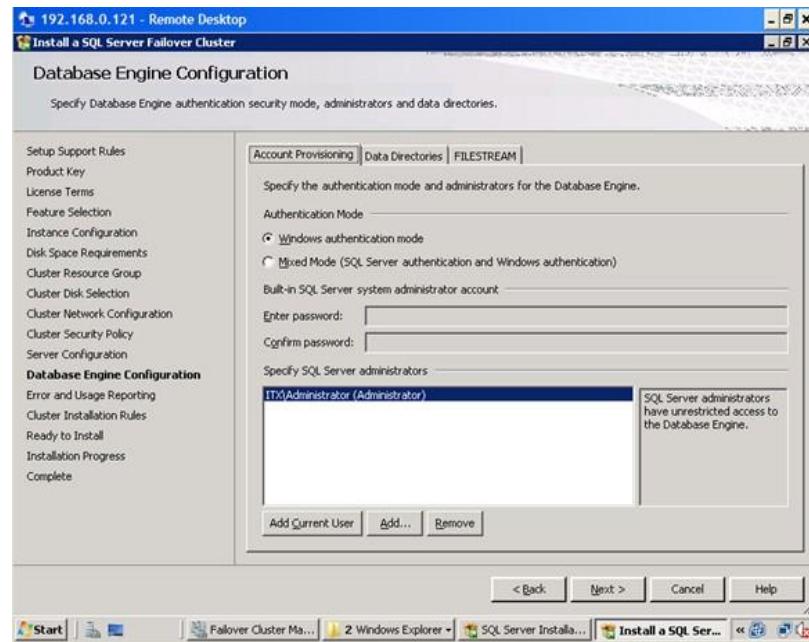


14. In the **Server Configuration** dialog box, enter the credentials that you will use for your SQL Server service accounts in the **Service Accounts** tab. In the **Collation** tab, select the appropriate collation to be used by SQL Server. Note that the startup type is set to manual for all cluster-aware services and cannot be changed during the installation process. Click **Next**.





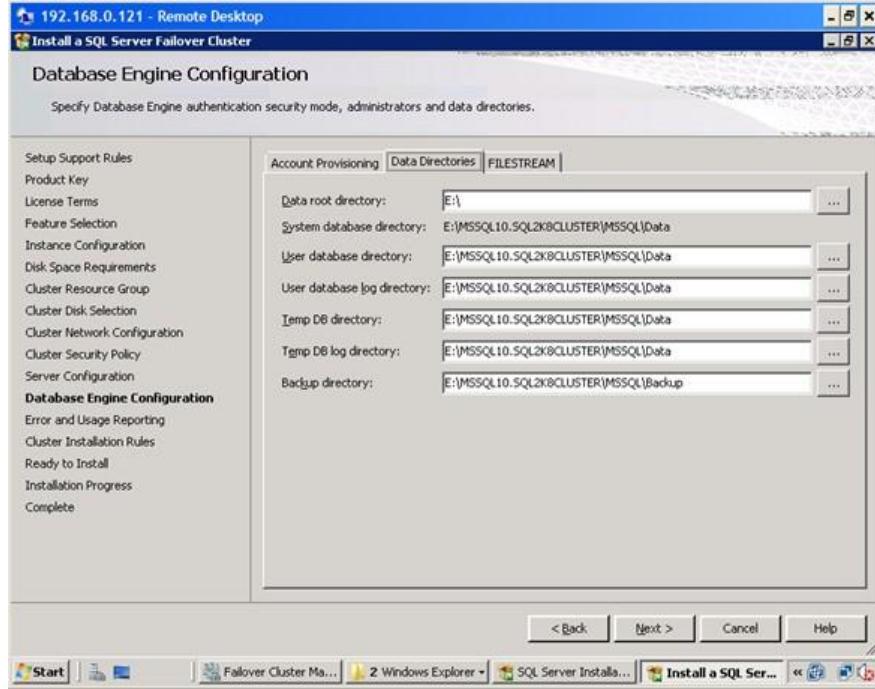
15. In the **Database Engine Configuration** dialog box, select the appropriate **Authentication Mode**. If you want to add the currently logged on user to be a part of the SQL Server administrators group, click the **Add Current User** button.



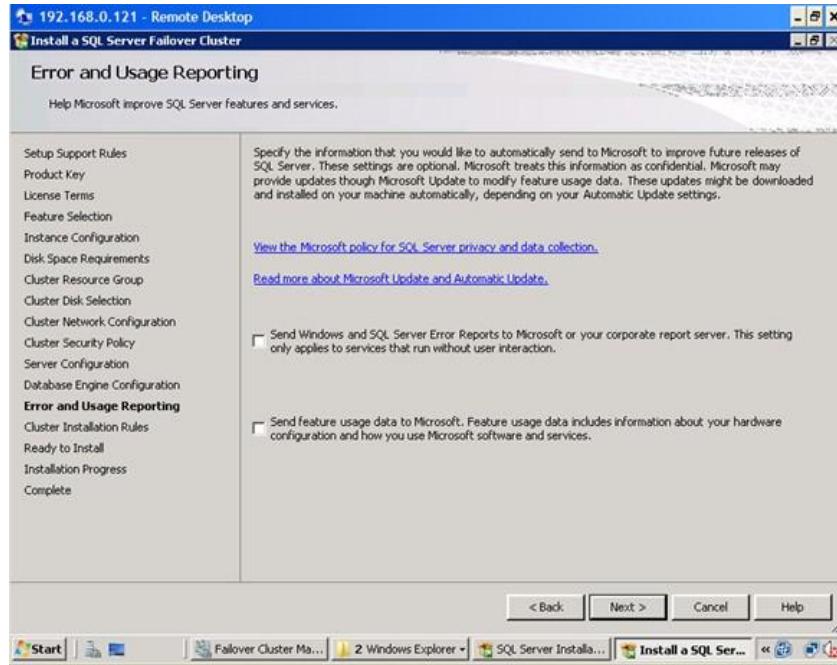
On the **Data Directories** tab, enter the path where your system and user database files will be created. This will default to the first shared disk in the cluster so in case you want to change it to the other shared disks to be used by SQL Server 2008, modify accordingly. If you intend to use the new



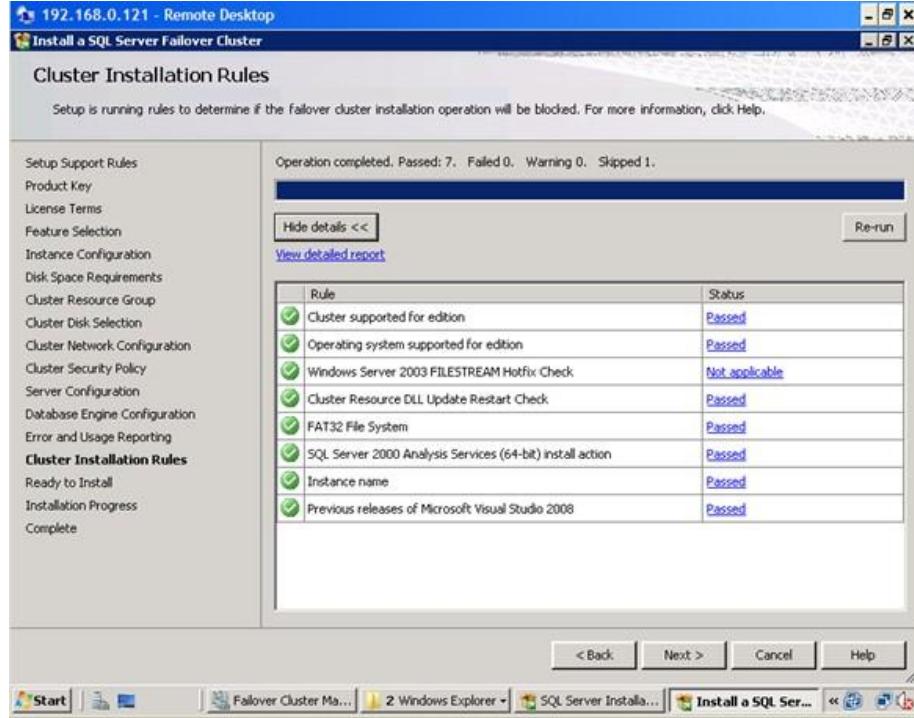
FILESTREAM feature, click the **FILESTREAM** tab and set the appropriate configurations. Click **Next**



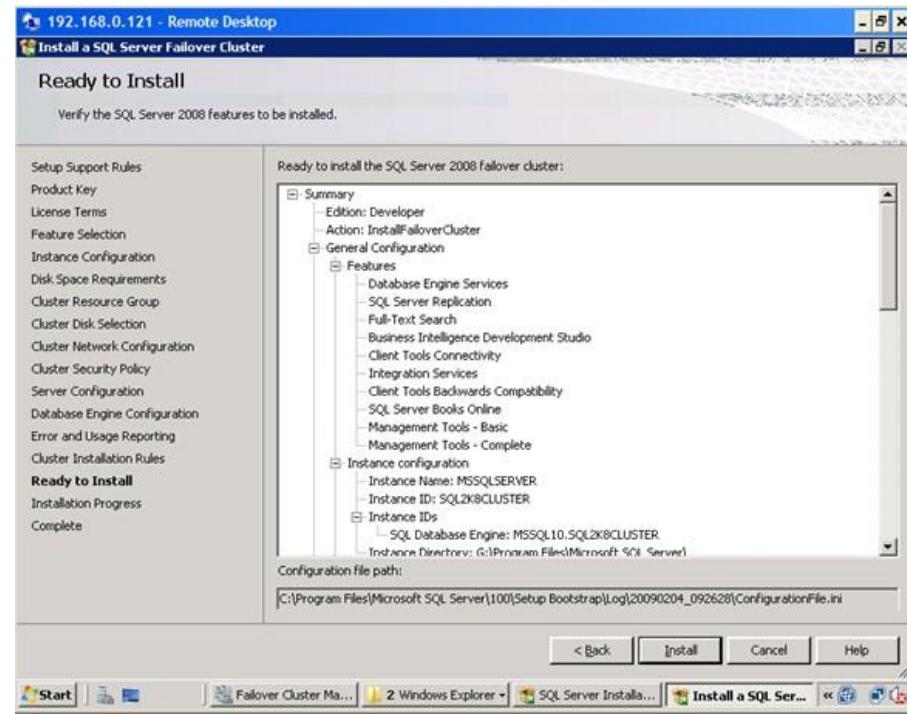
16. In the **Error and Usage Reporting** dialog box, click **Next**.



17. In the **Cluster Installation Rules** dialog box, verify that all checks are successful and click **Next**.

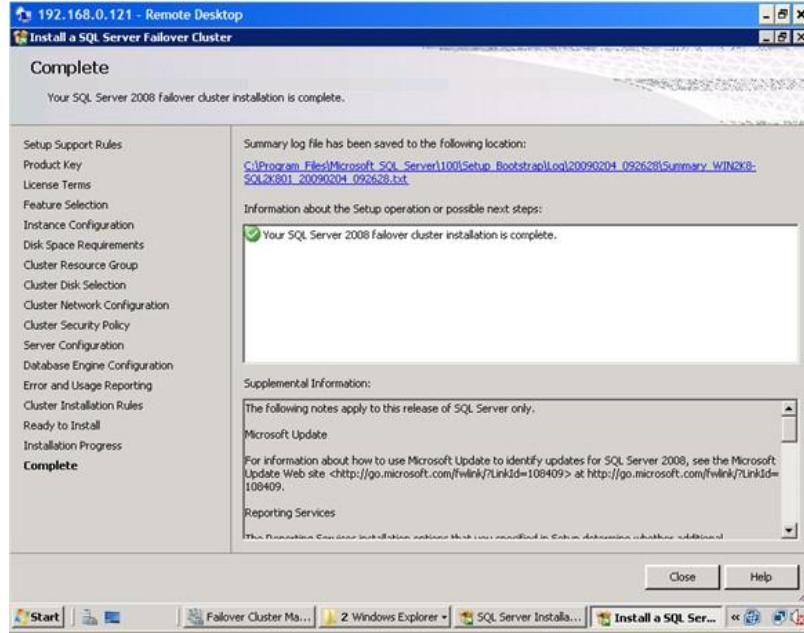


18. In the **Ready to Install** dialog box, verify that all configurations are correct.  
Click **Next**.

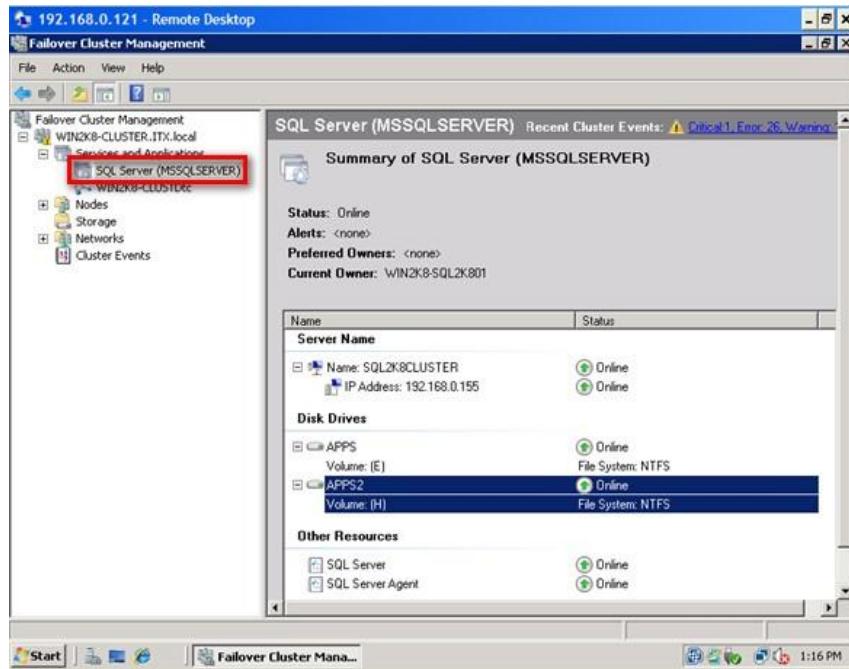


19. In the **Complete** dialog box, click **Close**. This concludes the installation of a SQL Server 2008 Failover Cluster





At the completion of a successful installation and configuration of the node, you now have a fully functional failover cluster instance. To validate, open the **Failover Cluster Management** console, and click on **SQL Server (MSSQLSERVER)** under **Services and Applications**. Make sure that all dependencies are online



Although we do have a fully functioning SQL Server 2008 failover cluster, it does not have high-availability at this point in time because there is only one node in the failover cluster. We still have to add the second node to the SQL Server 2008 cluster. In the last part of this series, we will add the second node in the failover cluster and install the latest cumulative update

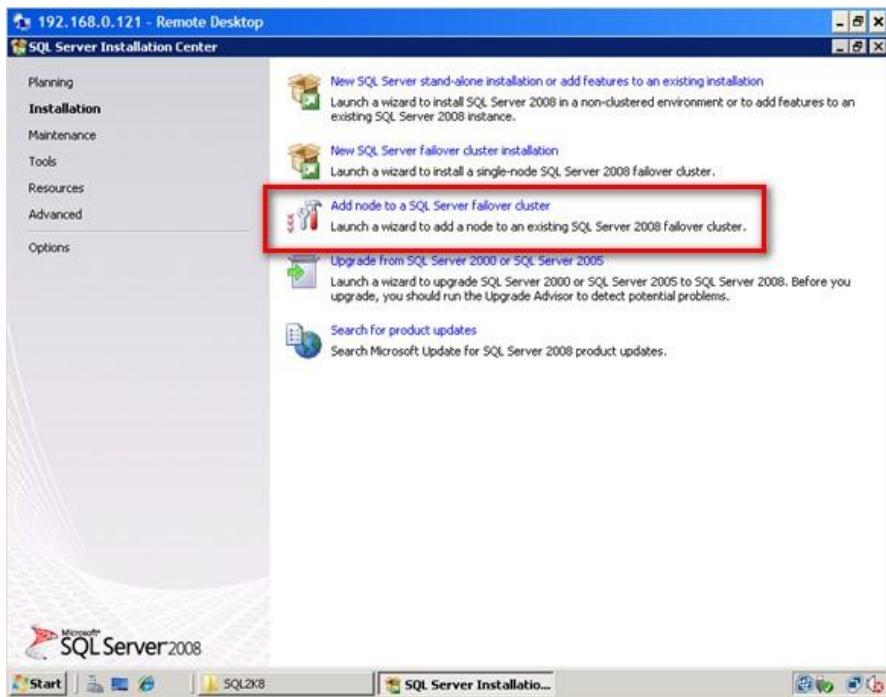


## Adding a node on a SQL Server 2008 Failover Cluster

To add a node on a SQL Server 2008 failover cluster:

1. Run **setup.exe** from the installation media to launch **SQL Server Installation Center**
2. Click on the **Installation** link on the left-hand side. Click the **Add node to a SQL Server failover cluster** link. This will run the **SQL Server 2008 Setup** wizard.

There are a couple of glitches when you get to this point. One of them is a popup error with an error message "*failed to retrieve data for this request*" while in this step. I've seen a [Microsoft Connect](#) item on this but refers to CTP6 so I was thinking it has already been resolved. After a few searches and questions asked, [SQL Server MVP Geoff Hiten](#) advised that prior to adding another node in the cluster, any cumulative update should be pre-applied to the node before the main installation as the cluster install of the RTM version has some bugs. This creates a patched install script for the RTM installer to use. The fix started with cumulative update 1 so, technically, you can apply any cumulative update. Sounds weird, but it works. You still have to apply the patch after the installation.



3. In the **Setup Support Rules** dialog box, validate that the checks return successful results and click **OK**.
4. In the **Product Key** dialog box, enter the product key that came with your installation media and click **Next**.

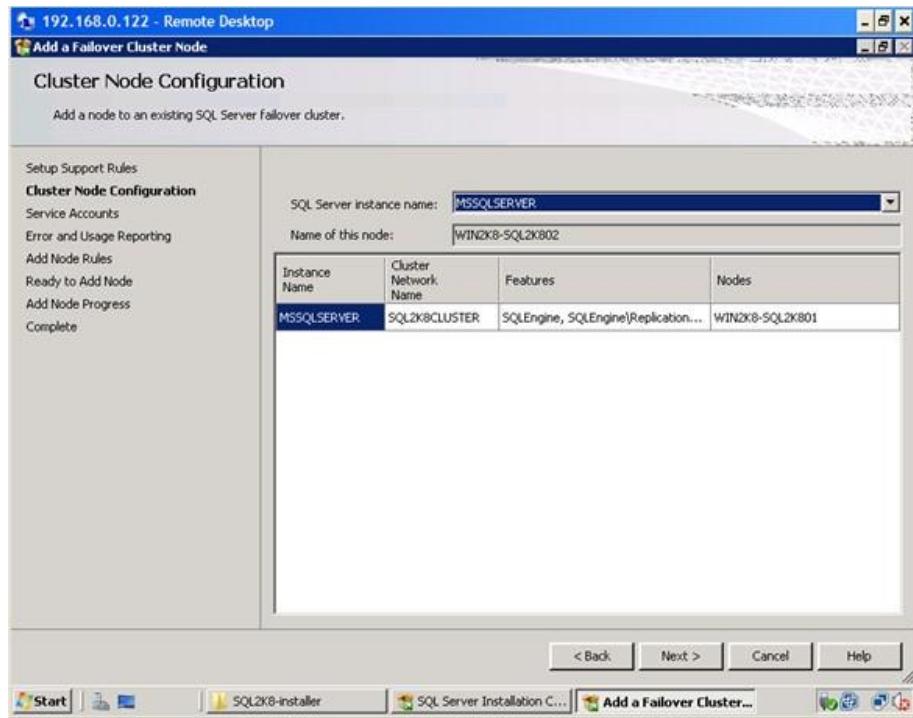
Again, a few glitches on this step. This might seem unusual as you are only being asked about the Product Key. There is also a [Microsoft Connect](#) item for this which basically asks you to run the **setup.exe** in command prompt.

There is a popup error with an error message "*The current SKU is invalid*" while in this step. This usually happens when you use a media with a supplied

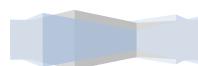


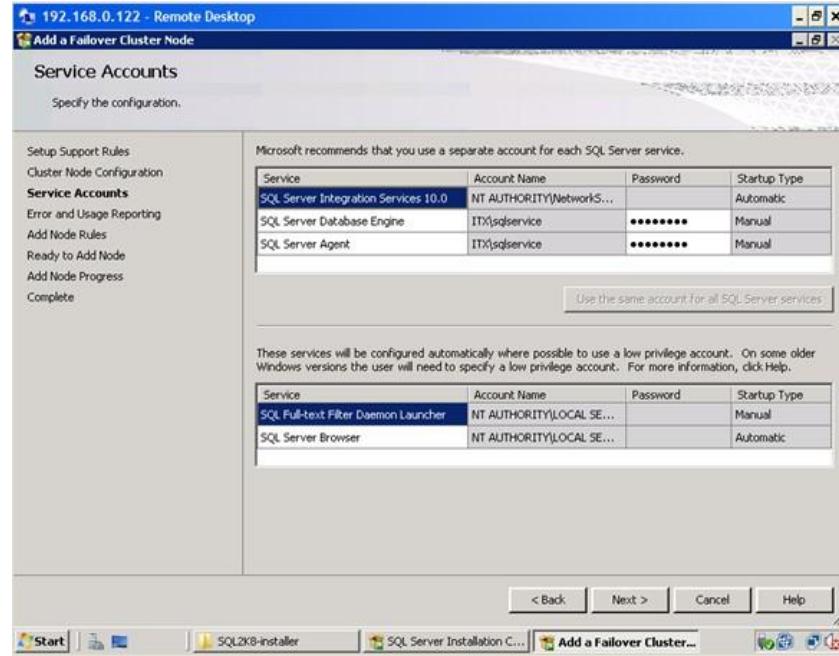
product key, like the one that comes with an MSDN subscription. What worked for me was to copy the installation media on a local disk locate the file **DefaultSetup.ini** file from the installation files and delete it or move it to different location. If you opt to delete the file, make sure you note down the product key written on this file as you will need to manually key this in during the installation process.

5. In the **License Terms** dialog box, click the **I accept the license terms** check box and click **Next**.
6. In the **Setup Support Rules** dialog box, click **Install**. Validate that the checks return successful results. Again, make sure to fix any errors returned by this check before proceeding with the installation.
7. In the **Cluster Node Configuration** dialog box, validate that the information for the existing SQL Server 2008 cluster is correct.

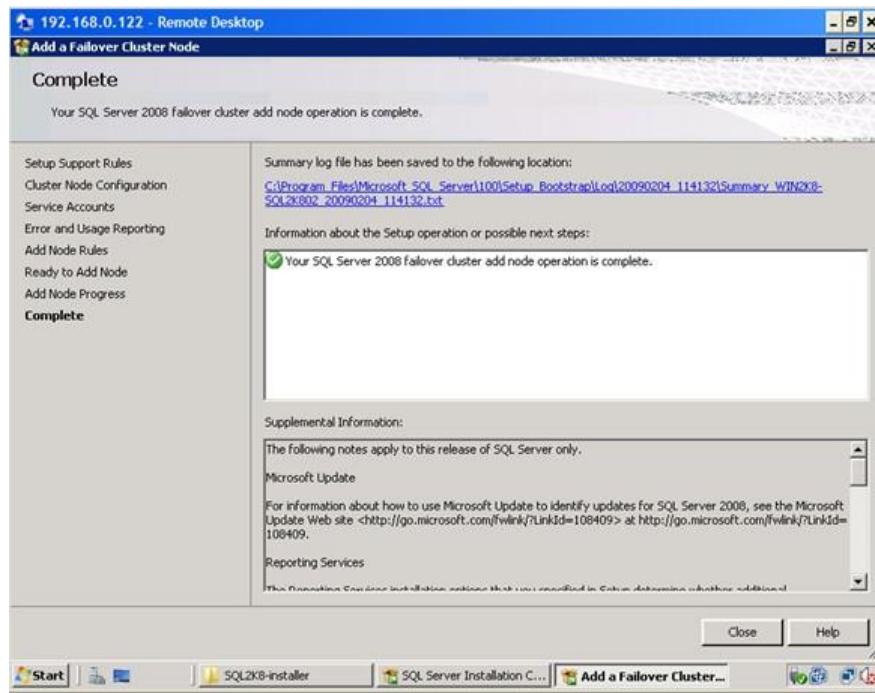


8. In the **Service Accounts** dialog box, verify that the information is the same as what you have used to configure the first node.

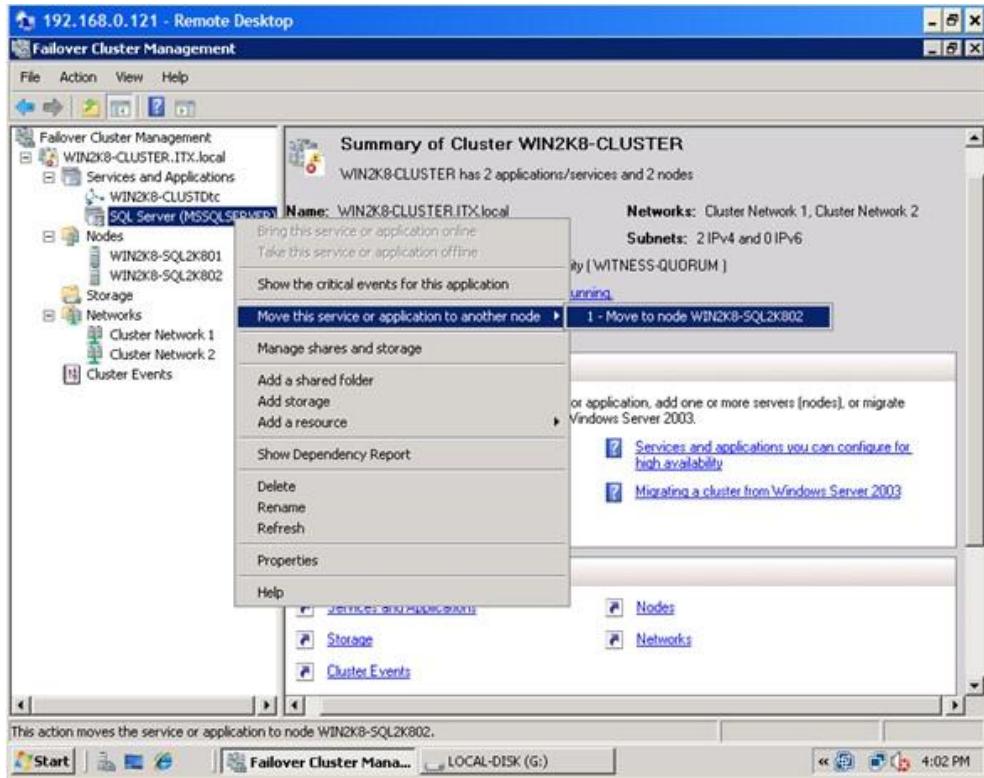




9. In the **Error and Usage Reporting** dialog box, click **Next**
10. In the **Add Node Rules** dialog box, verify that all checks are successful and click **Next**
11. In the **Ready to Add Node** dialog box, verify that all configurations are correct and click **Install**
12. In the **Complete** dialog box, click **Close**. This concludes adding a node to a SQL Server 2008 Failover Cluster



You can validate your cluster installation by expanding the **Services and Applications** node and check the cluster name of your SQL Server instance. You can now see an option to move the service to another node, in this case, the node you've just added in your failover cluster



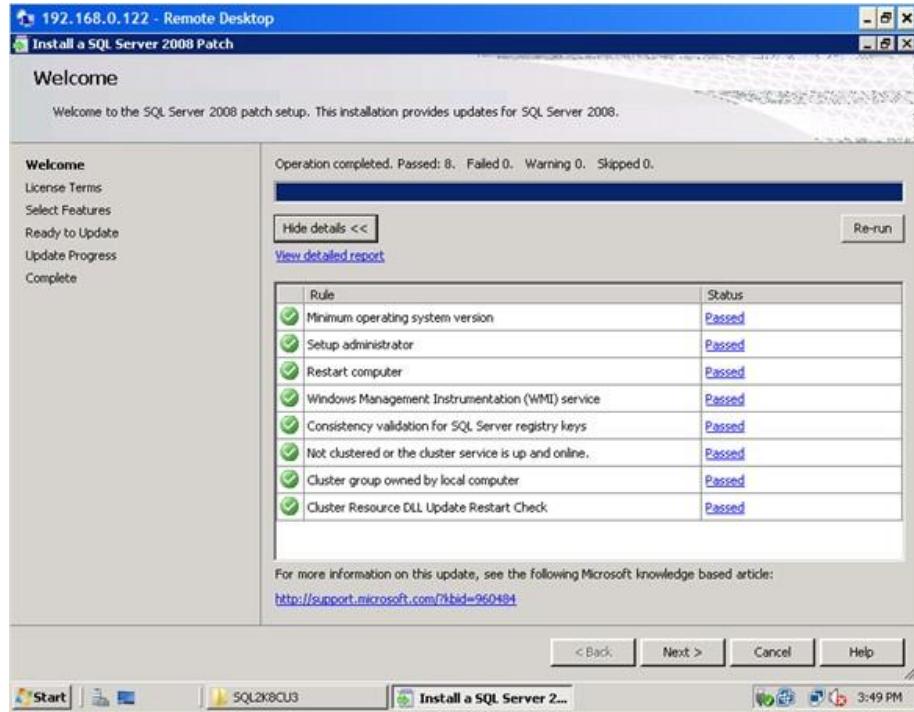
### Applying patches on a SQL Server 2008 cluster

Part of the tasks of a DBA is to apply patches on the database engine and a SQL Server 2008 failover cluster is no exception. In fact, it is not as straight-forward as applying patches and service packs on a stand-alone server. It is important to note that when applying patches or service packs to a SQL Server failover cluster, you should apply them first on the passive node. After completing the installation on the passive node, failover the SQL Server 2008 cluster resource to this node making it the active node. Once the SQL Server service and all other dependencies are up, you can, then, apply the patches on the new passive node. The latest available patch for SQL Server 2008 is cumulative update 4 and is available for request from Microsoft. For more information, check out this [Microsoft KB article](#). You will have to request for the patch from Microsoft as it is not available from the Microsoft Download Center. The screenshots below show cumulative update 3 (version **10.0.1600.22**) but the process is basically the same. Also, note that even though you may have already applied the cumulative update due to the bug mentioned above for adding a node in a failover cluster, you still have to apply the patch on both nodes

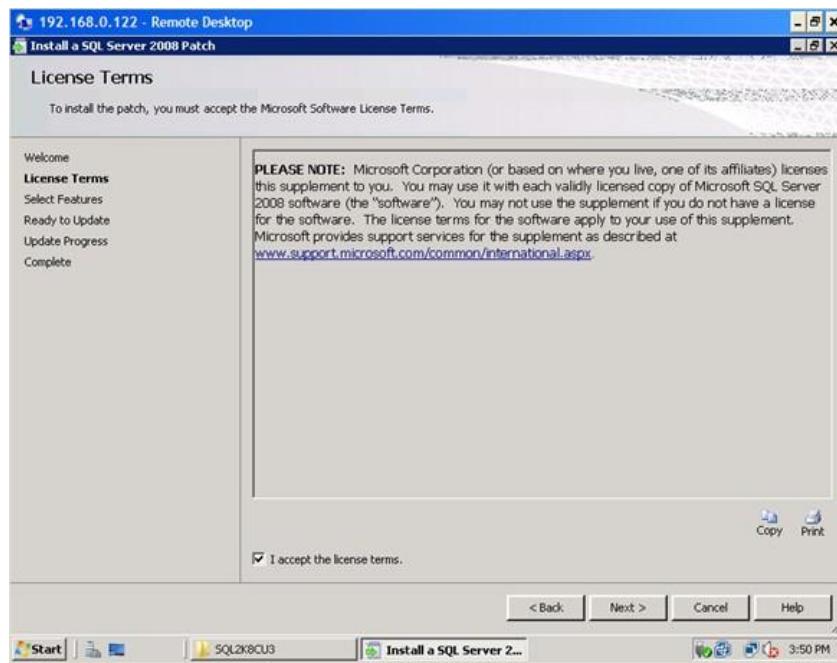
To apply patches on a SQL Server 2008 failover cluster node:



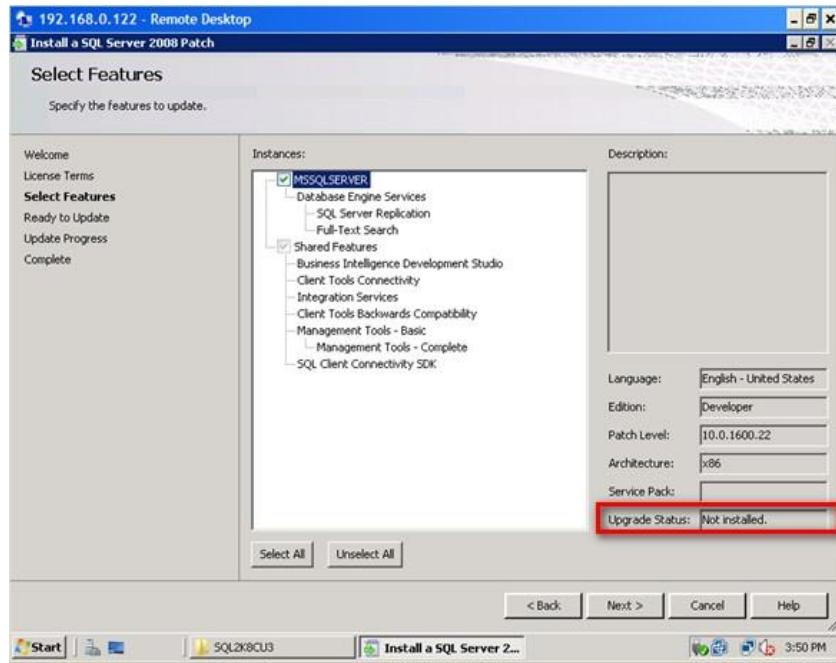
1. Run **SQLServer2008-KB960484-x86.exe** (this would depend on the cumulative update that you want to apply) from the hotfix package you have requested from Microsoft
2. In the **Welcome** dialog box, validate that the checks return successful results.



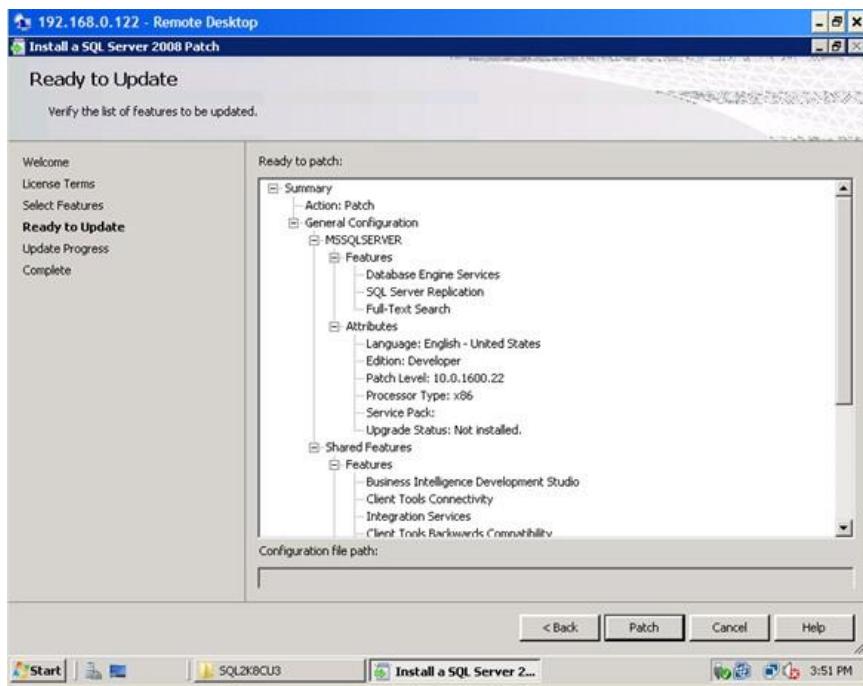
3. In the **License Terms** dialog box, click the **I accept the license terms** check box and click **Next**



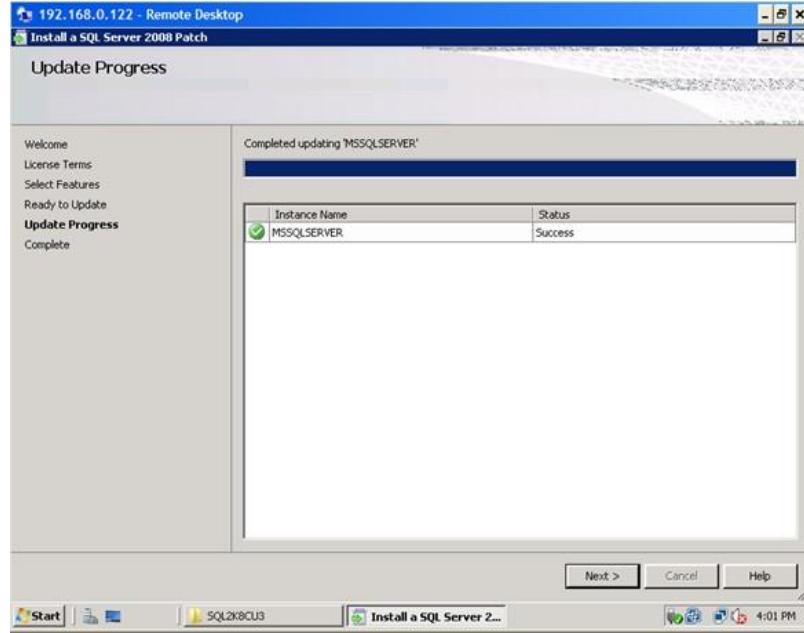
4. In the **Select Features** dialog box, validate the SQL Server 2008 components by clicking on the check box. The **Upgrade Status** field will tell you whether or not the patch has already been applied. Click **Next**



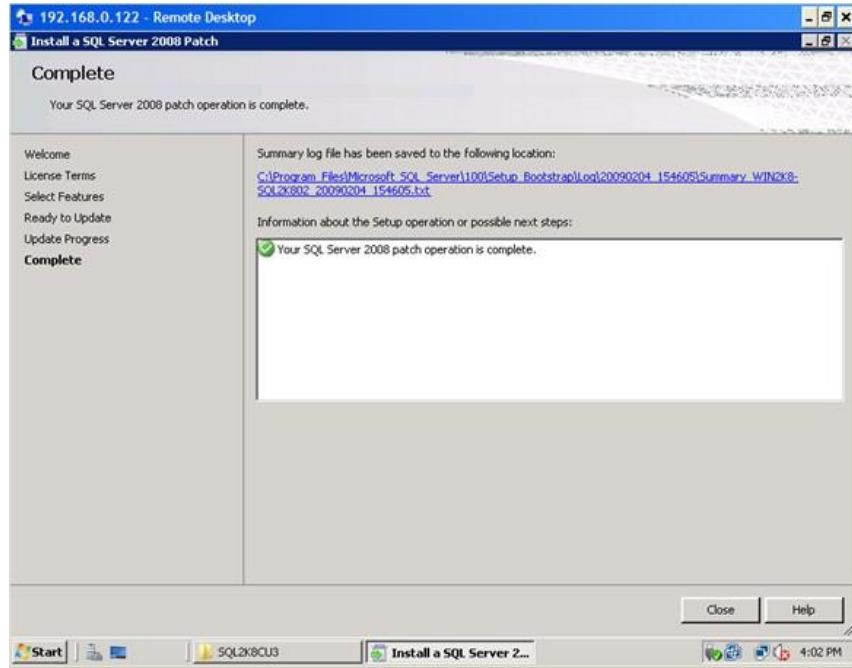
5. In the **Ready to Update** dialog box, verify that all configurations are correct and click **Patch**



6. In the **Update Progress** dialog box, validate that the installation was successful.



7. In the **Complete** dialog box, click Close. This concludes patching the passive node of a SQL Server 2008 Failover Cluster



After successfully installing the patch on the passive node, move the SQL Server 2008 cluster resource to this node so it will become the new active node. Make sure that all the SQL Server 2008 cluster dependencies are online prior to applying the patch on the other node. Repeat the process outlined above to the new passive node. A more comprehensive approach for applying a SQL Server 2008 patch to a failover cluster instance is defined in this [Microsoft KB article](#)



## Best Practices on Clustering

- Detailed planning is critical to the success of every SQL Server cluster installation. Fully plan the install before performing the actual install.
- An expensive cluster is of little value if the supporting infrastructure is not also fault tolerant. For example, don't forget power redundancy, network redundancy, etc.
- Run only a single instance of SQL Server per node. Whether you have two or eight nodes in your cluster, leave one node as a failover node.
- Cluster nodes must not be domain controllers, and all nodes must belong in the same domain and should have access to two or more domain controllers.
- All cluster hardware must be on the Microsoft Windows Clustering Hardware Compatibility List, and certified to work together as part of a cluster.
- Since clustering is not designed to protect data (only SQL Server instances), the shared storage device used by the cluster must incorporate fault tolerant technology. Consider log shipping or mirroring to further protect your production databases.
- When initially installing Windows and SQL Server Clustering, be sure that all drivers and software are up-to-date, including the latest service packs or hot fixes.
- Each node of a cluster should have identical hardware, drivers, software, and configuration settings.
- Fiber channel shared arrays are preferred over SCSI, and Fiber channel has to be used if you include more than two nodes in your cluster.
- The Quorum drive must be on its own fault-tolerant, dedicated, logical drive.
- Once the cluster has been installed, test it thoroughly for every possible failure scenario.
- Do not run antivirus or antispyware on a SQL Server cluster.
- If you need to reconfigure any Windows or SQL Server clustering configuration options, such as IP addresses or virtual names, you will need to uninstall clustering and then reinstall it.
- Monitor active production clusters on a daily basis, looking for any potential problems. Periodically test failover on production servers to ensure all is working well.
- Once you have a stable SQL Server Cluster running, be very leery about making any changes to it, whatsoever.

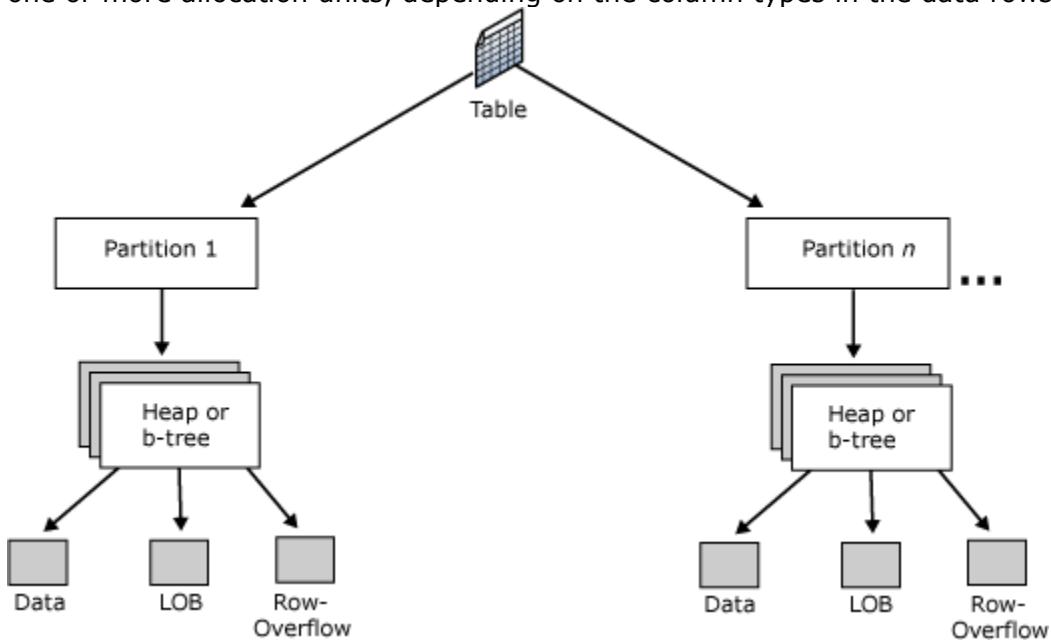


## Table & Index Architecture

Indexes are the other significant user-defined, on-disk data structure besides tables. An index provides fast access to data when the data can be searched by the value that is the index key. To really understand the benefit that indexes can provide and how to determine the best indexes for your environment, we need to know into the organization of Microsoft SQL Server indexes. Tables and indexes are stored as a collection of 8-KB pages.

### Table Organization

The following illustration shows the organization of a table. A table is contained in one or more partitions and each partition contains data rows in either a heap or a clustered index structure. The pages of the heap or clustered index are managed in one or more allocation units, depending on the column types in the data rows.



### Partitions

Table and index pages are contained in one or more partitions. A partition is a user-defined unit of data organization. By default, a table or index has only one partition that contains all the table or index pages. The partition resides in a single filegroup. A table or index with a single partition is equivalent to the organizational structure of tables and indexes in earlier versions of SQL Server.

When a table or index uses multiple partitions, the data is partitioned horizontally so that groups of rows are mapped into individual partitions, based on a specified column. The partitions can be put on one or more filegroups in the database. The table or index is treated as a single logical entity when queries or updates are performed on the data. To view the partitions used by a table or index, use the sys.partitions (Transact-SQL) catalog view.



## Clustered Tables, Heaps, and Indexes

SQL Server tables use one of two methods to organize their data pages within a partition:

- Clustered tables are tables that have a clustered index.

The data rows are stored in order based on the clustered index key. The clustered index is implemented as a B-tree index structure that supports fast retrieval of the rows, based on their clustered index key values. The pages in each level of the index, including the data pages in the leaf level, are linked in a doubly-linked list. However, navigation from one level to another is performed by using key values.

- Heaps are tables that have no clustered index.

The data rows are not stored in any particular order, and there is no particular order to the sequence of the data pages. The data pages are not linked in a linked list. For more information, see [Heap Structures](#).

Indexed views have the same storage structure as clustered tables.

When a heap or a clustered table has multiple partitions, each partition has a heap or B-tree structure that contains the group of rows for that specific partition. For example, if a clustered table has four partitions, there are four B-trees; one in each partition.

## Nonclustered Indexes

Nonclustered indexes have a B-tree index structure similar to the one in clustered indexes. The difference is that nonclustered indexes do not affect the order of the data rows. The leaf level contains index rows. Each index row contains the nonclustered key value, a row locator and any included, or nonkey, columns. The locator points to the data row that has the key value.

## Allocation Units

An allocation unit is a collection of pages within a heap or B-tree used to manage data based on their page type. The following table lists the types of allocation units used to manage data in tables and indexes.

Allocation unit type	Is used to manage
IN_ROW_DATA	Data or index rows that contain all data, except large object (LOB) data. Pages are of type Data or Index.
LOB_DATA	Large object data stored in one or more of these data types: text, ntext, image, xml, varchar(max), nvarchar(max), varbinary(max), or CLR user-defined types (CLR UDT). Pages are of type Text/Image.
ROW_OVERFLOW_DATA	Variable length data stored in varchar, nvarchar, varbinary, or sql_variant columns that exceed the 8,060 byte row size limit. Pages are of type Data.



A heap or B-tree can have only one allocation unit of each type in a specific partition. To view the table or index allocation unit information, use the sys.allocation\_units catalog view.

### **IN\_ROW\_DATA Allocation Unit**

For every partition used by a table (heap or clustered table), index, or indexed view, there is one IN\_ROW\_DATA allocation unit that is made up of a collection of data pages. This allocation unit also contains additional collections of pages to implement each nonclustered and XML index defined for the table or view. The page collections in each partition of a table, index, or indexed view are anchored by page pointers in the sys.system\_internals\_allocation\_units system view.

Each table, index, and indexed view partition has a row in sys.system\_internals\_allocation\_units uniquely identified by a container ID (container\_id). The container ID has a one-to-one mapping to the partition\_id in the sys.partitions catalog view that maintains the relationship between the table, index, or the indexed view data stored in a partition and the allocation units used to manage the data within the partition.

The allocation of pages to a table, index, or an indexed view partition is managed by a chain of IAM pages. The column first\_iam\_page in sys.system\_internals\_allocation\_units points to the first IAM page in the chain of IAM pages managing the space allocated to the table, index, or the indexed view in the IN\_ROW\_DATA allocation unit.

sys.partitions returns a row for each partition in a table or index.

- A heap has a row in sys.partitions with index\_id = 0. The first\_iam\_page column in sys.system\_internals\_allocation\_units points to the IAM chain for the collection of heap data pages in the specified partition. The server uses the IAM pages to find the pages in the data page collection, because they are not linked.
- A clustered index on a table has a row in sys.partitions with index\_id = 1. The root\_page column in sys.system\_internals\_allocation\_units points to the top of the clustered index B-tree in the specified partition. The server uses the index B-tree to find the data pages in the partition.
- Each nonclustered index created for a table or a view has a row in sys.partitions with index\_id > 1. The root\_page column in sys.system\_internals\_allocation\_units points to the top of the nonclustered index B-tree in the specified partition.

### **ROW\_OVERFLOW\_DATA Allocation Unit**

For every partition used by a table (heap or clustered table), index, or indexed view, there is one ROW\_OVERFLOW\_DATA allocation unit. This allocation unit contains zero (0) pages until a data row with variable length columns (varchar, nvarchar, varbinary, or sql\_variant) in the IN\_ROW\_DATA allocation unit exceeds the 8 KB row size limit. When the size limitation is reached, SQL Server moves the column with the largest width from that row to a page in the ROW\_OVERFLOW\_DATA allocation unit. A 24-byte pointer to this off-row data is maintained on the original page.



Text/Image pages in the ROW\_OVERFLOW\_DATA allocation unit are managed in the same way pages in the LOB\_DATA allocation unit are managed. That is, the Text/Image pages are managed by a chain of IAM pages.

### **LOB\_DATA Allocation Unit**

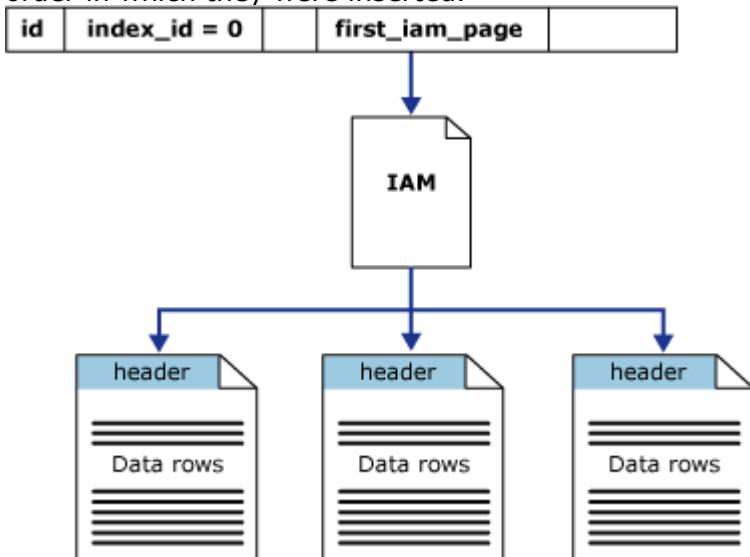
When a table or index has one or more LOB data types, one LOB\_DATA allocation unit per partition is allocated to manage the storage of that data. The LOB data types include text, ntext, image, xml, varchar(max), nvarchar(max), varbinary(max), and CLR user-defined types.

## Heap Structures

A heap is a table without a clustered index. Heaps have one row in sys.partitions, with **index\_id = 0** for each partition used by the heap. By default, a heap has a single partition. When a heap has multiple partitions, each partition has a heap structure that contains the data for that specific partition. For example, if a heap has four partitions, there are four heap structures; one in each partition.

The column **first\_iam\_page** in the **sys.system\_internals\_allocation\_units** system view points to the first IAM page in the chain of IAM pages that manage the space allocated to the heap in a specific partition. SQL Server uses the IAM pages to move through the heap. The data pages and the rows within them are not in any specific order and are not linked. The only logical connection between data pages is the information recorded in the IAM pages.

Table scans or serial reads of a heap can be performed by scanning the IAM pages to find the extents that are holding pages for the heap. Because the IAM represents extents in the same order that they exist in the data files, this means that serial heap scans progress sequentially through each file. Using the IAM pages to set the scan sequence also means that rows from the heap are not typically returned in the order in which they were inserted.



The following figure shows how the SQL Server Database Engine uses IAM pages to retrieve data rows in a single partition heap.



## Clustered Index Structures

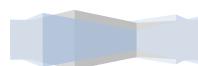
In SQL Server, indexes are organized as B-trees. Each page in an index B-tree is called an index node. The top node of the B-tree is called the root node. The bottom level of nodes in the index is called the leaf nodes. Any index levels between the root and the leaf nodes are collectively known as intermediate levels. In a clustered index, the leaf nodes contain the data pages of the underlying table. The root and intermediate level nodes contain index pages holding index rows. Each index row contains a key value and a pointer to either an intermediate level page in the B-tree, or a data row in the leaf level of the index. The pages in each level of the index are linked in a doubly-linked list.

Clustered indexes have one row in sys.partitions, with **index\_id = 1** for each partition used by the index. By default, a clustered index has a single partition. When a clustered index has multiple partitions, each partition has a B-tree structure that contains the data for that specific partition. For example, if a clustered index has four partitions, there are four B-tree structures; one in each partition.

Depending on the data types in the clustered index, each clustered index structure will have one or more allocation units in which to store and manage the data for a specific partition. At a minimum, each clustered index will have one IN\_ROW\_DATA allocation unit per partition. The clustered index will also have one LOB\_DATA allocation unit per partition if it contains large object (LOB) columns. It will also have one ROW\_OVERFLOW\_DATA allocation unit per partition if it contains variable length columns that exceed the 8,060 byte row size limit.

The pages in the data chain and the rows in them are ordered on the value of the clustered index key. All inserts are made at the point where the key value in the inserted row fits in the ordering sequence among existing rows. The page collections for the B-tree are anchored by page pointers in the **sys.system\_internals\_allocation\_units** system view.

For a clustered index, the **root\_page** column in **sys.system\_internals\_allocation\_units** points to the top of the clustered index for a specific partition. SQL Server moves down the index to find the row corresponding to a clustered index key. To find a range of keys, SQL Server moves through the index to find the starting key value in the range and then scans through the data pages using the previous or next pointers. To find the first page in the chain of data pages, SQL Server follows the leftmost pointers from the root node of the index.



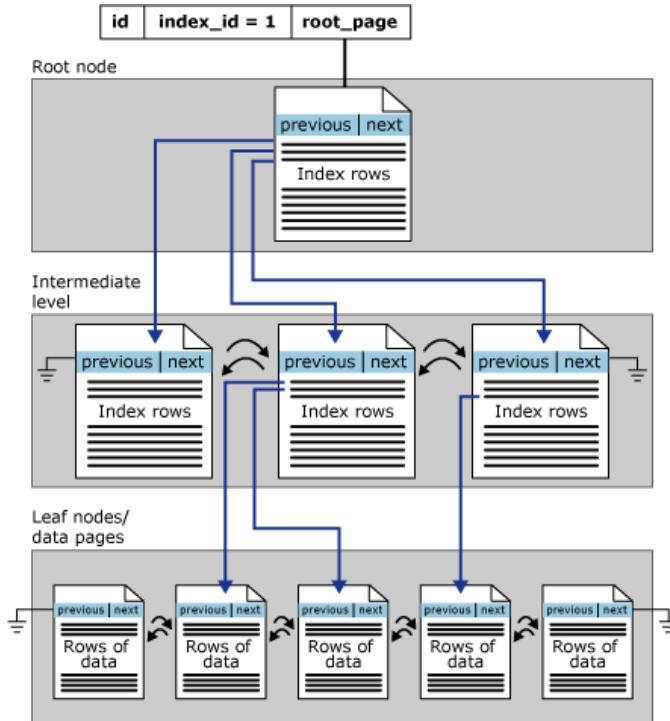


Figure shows the structure of a clustered index in a single partition.

## Non clustered Index Structures

Non clustered indexes have the same B-tree structure as clustered indexes, except for the following significant differences:

- The data rows of the underlying table are not sorted and stored in order based on their non clustered keys.
- The leaf layer of a non clustered index is made up of index pages instead of data pages.

Non clustered indexes can be defined on a table or view with a clustered index or a heap. Each index row in the non clustered index contains the non clustered key value and a row locator. This locator points to the data row in the clustered index or heap having the key value.

The row locators in non clustered index rows are either a pointer to a row or are a clustered index key for a row, as described in the following:

- If the table is a heap, which means it does not have a clustered index, the row locator is a pointer to the row. The pointer is built from the file identifier (ID), page number, and number of the row on the page. The whole pointer is known as a Row ID (RID).
- If the table has a clustered index, or the index is on an indexed view, the row locator is the clustered index key for the row. If the clustered index is not a

unique index, SQL Server makes any duplicate keys unique by adding an internally generated value called a **uniqueifier**. This four-byte value is not visible to users. It is only added when required to make the clustered key unique for use in non clustered indexes. SQL Server retrieves the data row by searching the clustered index using the clustered index key stored in the leaf row of the non clustered index.

Non clustered indexes have one row in sys.partitions with **index\_id > 0** for each partition used by the index. By default, a non clustered index has a single partition. When a non clustered index has multiple partitions, each partition has a B-tree structure that contains the index rows for that specific partition. For example, if a non clustered index has four partitions, there are four B-tree structures, with one in each partition.

Depending on the data types in the non clustered index, each non clustered index structure will have one or more allocation units in which to store and manage the data for a specific partition. At a minimum, each non clustered index will have one IN\_ROW\_DATA allocation unit per partition that stores the index B-tree pages. The non clustered index will also have one LOB\_DATA allocation unit per partition if it contains large object (LOB) columns. The page collections for the B-tree are anchored by **root\_page** pointers in the **sys.system\_internals\_allocation\_units** system view.

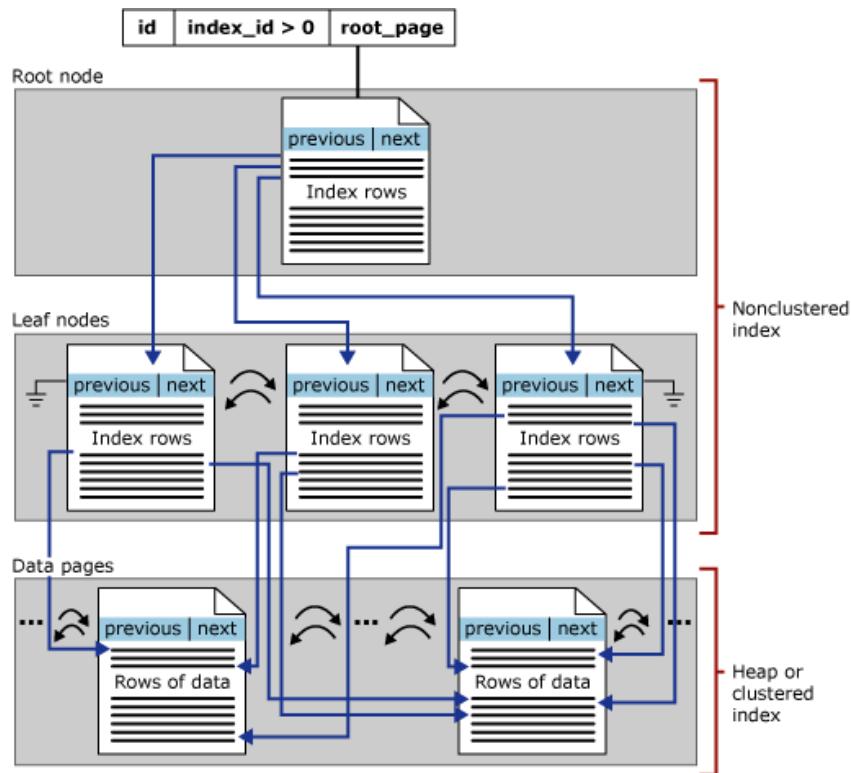


Figure shows the structure of a non clustered index in a single partition.



## **Index Fragmentation in SQL Server**

Index fragmentation is a phenomenon where the index contents are scattered. Normally the contents are in contiguous fashion which helps in fast retrieval of the underlying data. When the indexes are fragmented the data access becomes time consuming because of the scattered data that needs to be searched and read.

Fragmentation occurs as data is modified. The following are the two types of Index fragmentation:

1. Internal fragmentation
2. External fragmentation

### **Internal fragmentation:**

This happens when space is available within your index page i.e. when the index pages have not been filled as full as possible. Due to internal fragmentation the index is taking up more space than it needs to. Thus when scanning the index it results in more read operations. Internal fragmentation also happens due to specifying a low value of fill factor (which determines the % of space to be filled in a leaf level page).

This is also caused by rows that are removed by DELETE statements or when pages are split and only filled to about half. Empty space on pages means there are less rows per page, which in turn means more page reads.

### **External Fragmentation:**

External fragmentation occurs when the pages are not contiguous on the index. If the pages in a book are NOT ordered in a logical way (page 1, then page 2, then page 3 and so on) causing you to go back and forward to compound the information and make sense of the reading. External fragmentation happens when there are frequent UPDATES and INSERTS in a table having small amount of free space in the index page.

Since the page is already full or only has less free space left and if it is not able to accommodate the new row inserted or updated, as a result Page split happens in order to allocate the new row. Due to page split, original page will be split such that half the rows are left on the original page and the other half is moved to the new page. Mostly the new page is not contiguous to the page being split. Page split is an expensive operation and should always be avoided.

### **How to determine fragmentation?**

The following query will give the fragmentation information of a particular table named person.address in adventureworks database. Please modify the query to replace the database name and table name according to your requirements.

301

```
SELECT CAST(DB_NAME(database_id) AS varchar(20)) AS [Database Name],  
CAST(OBJECT_NAME(object_id) AS varchar(20)) AS [TABLE NAME], Index_id,  
Index_type_desc, Avg_fragmentation_in_percent, Avg_page_space_used_in_percent
```



```
FROM  
sys.dm_db_index_physical_stats(DB_ID('AdventureWorks'),OBJECT_ID('person.address'),NULL,NULL,'Detailed')
```

If you wish to identify the fragmentation information for the tables in a particular database please use the below query. I am using it to find the fragmentation in Adventureworks database.

```
SELECT CAST(DB_NAME(database_id) AS varchar(20)) AS [Database Name],  
CAST(OBJECT_NAME(object_id) AS varchar(20)) AS [TABLE NAME], Index_id,  
Index_type_desc, Avg_fragmentation_in_percent, Avg_page_space_used_in_percent  
FROM  
sys.dm_db_index_physical_stats(DB_ID('AdventureWorks'),NULL,NULL,NULL,'Detaile  
d')
```

Take a look at the output of the following columns in the above query:

#### **Avg\_fragmentation\_in\_percent:**

If the value is >5 and <30 you need to REORGANIZE the index using ALTER index REORGANIZE command. If the value is >30 you need to REBUILD the indexes using ALTER index REBUILD command.

#### **Avg\_page\_space\_used\_in\_percent:**

This value represents the amount of page space used in an index. If the value is <75% and >60% we need to REORGANIZE the indexes else REBUILD the indexes.

#### **Defragmenting the Indexes:**

We need to either use ALTER INDEX REBUILD or ALTER INDEX REORGANIZE commands to remove the fragmentation from the indexes. Generally its advisable to schedule a job to do this operations in OFF-Production hours as they consume lots of resources.

## **Best Practices on Indexing**

- Periodically, run the Index Wizard or Database Engine Tuning Advisor against current Profiler traces to identify potentially missing indexes.
- Remove indexes that are never used.
- Don't accidentally create redundant indexes.
- As a rule of thumb, every table should have at least a clustered index.  
Generally, but not always, the clustered index should be on a column that monotonically increases — such as an identity column, or some other column where the value is increasing — and is unique. In many cases, the primary key is the ideal column for a clustered index.
- Since you can only create one clustered index per table, take extra time to carefully consider how it will be used. Consider the type of queries that will be used against the table, and make an educated guess as to which query (the most common one run against the table, perhaps) is the most critical, and if this query will benefit from having a clustered index.



- If a column in a table is not at least 95% unique, then most likely the query optimizer will not use a non-clustered index based on that column. Because of this, you generally don't want to add non-clustered indexes to columns that aren't at least 95% unique.
- Keep the "width" of your indexes as narrow as possible. This reduces the size of the index and reduces the number of disk I/O reads required to read the index, boosting performance.
- If possible, avoid adding a clustered index to a GUID column (uniqueidentifier data type). GUIDs take up 16-bytes of storage, more than an Identity column, which makes the index larger, which increases I/O reads, which can hurt performance.
- Indexes should be considered on all columns that are frequently accessed by the JOIN, WHERE, ORDER BY, GROUP BY, TOP, and DISTINCT clauses.
- Don't automatically add indexes on a table because it seems like the right thing to do. Only add indexes if you know that they will be used by the queries run against the table.
- When creating indexes, try to make them unique indexes if at all possible. SQL Server can often search through a unique index faster than a non-unique index because in a unique index, each
- Row is unique, and once the needed record is found, SQL Server doesn't have to look any further.
- If you perform regular joins between two or more tables in your queries, performance will be optimized if each of the joined columns has appropriate indexes.
- Don't automatically accept the default value of 100 for the fill factor for your indexes. It may or may not best meet your needs. A high fill factor is good for seldom changed data, but highly
- Modified data needs a lower fill factor to reduce page splitting.
- Don't over index your OLTP tables, as every index you add increases the time it takes to perform INSERTS, UPDATES, and DELETES. There is a fine line between having the ideal number of
- Indexes (for SELECTs) and the ideal number to minimize the overhead that occurs with indexes during data modifications.
- If you know that your application will be performing the same query over and over on the same table, consider creating a non-clustered covering index on the table. A covering index, which is a form of a composite index, includes all of the columns referenced in SELECT, JOIN, and WHERE clauses of a query. Because of this, the index contains the data you are looking for and SQL Server doesn't have to look up the actual data in the table, reducing logical and/or physical I/O, and boosting performance.



## Performance Tuning

Introduction to Performance Troubleshooting Methodology:

SQL Server is an enterprise-level database server that has been used by organizations to run their mission-critical applications worldwide. These applications impose the toughest requirements in terms of availability, performance, and scalability. Very few enterprise workloads, if any, have requirements that exceed these. Microsoft SQL Server 2005 has successfully met or exceeded the requirements of these workloads under demanding conditions.

You may wonder what we mean by performance because the word may mean different things to different people. For the discussions here, we will focus on the following three terms.

- **Response time** Refers to the interval between the time when a request is submitted and when the first character of the response is received.
- **Throughput** Refers to the number of transactions that can be processed in a fixed unit of time.
- **Scalability** Refers to how the throughput and/or the response time changes as we add more hardware resources. In simple terms, scalability means that if you are hitting a hardware bottleneck, you can alleviate it simply by adding more resources.

### Factors That Impact Performance

Most users perceive the performance of their SQL Server based on the responsiveness of their application. While SQL Server itself does play a part in application responsiveness, before we can isolate SQL Server as the source of a performance issue, we first need to understand the factors that impact the performance of your application. We will look into these factors for completeness before we switch our focus to troubleshooting performance problems in SQL Server.

At a high level, many factors can impact the performance and scalability that can be achieved in your application. We will be looking at the following:

- Application Architecture
- Application Design
- Transactions and Isolation Levels
- Transact-SQL Code
- Hardware Resources
- SQL Server Configuration

In this I am explaining about the Transactions and Isolation levels, Hardware resources and SQL server configuration

### Transactions and Isolation Levels

304

Applications interact with SQL Server using one or more transactions. A transaction can be started explicitly by executing the BEGIN TRANSACTION Transact-SQL



statement or implicitly, by SQL Server, for each statement that is not explicitly encapsulated by the transaction. Transactions are very fundamental to database systems. A transaction represents a unit of work that provides the following four fundamental properties:

- Atomicity Changes done under a transaction that are either all commit or are rolled back. No partial changes are allowed. It is an all-or-nothing proposition.
- Consistency Changes done under a transaction database from one consistent state to another. A transaction takes a database from one consistent state to another.
- Isolation Changes done by a transaction are isolated from other concurrent transactions until the transaction commits.
- Durability Changes done by committed transactions are permanent.

## Hardware Resources

Hardware resources are the horsepower you need to run your application. It will do you no good if you have a well-designed application, but it is running on hardware that is not on a par with the demands of the workload. Most often, an application is tested in a small-scale environment with simulated workload. While this does serve a useful purpose, clearly it is not a replacement for measuring the performance of the workload and the hardware it actually runs on in production. So when your application hits performance issues, it can be the hardware, however, that is not necessarily so. Any hardware resource (CPU, I/O, memory, or network) that is pushed beyond its operational capacity in any application tier will lead to a slowdown in your application. Note, too, that hardware bottlenecks may also be caused by poor application designs, which ultimately need to be addressed. In such cases, upgrading hardware is a short-term solution at best.

## SQL Server Configuration

SQL Server is designed to be self-tuning to meet the challenges of the workload. In most cases, it just works well with out-of-the-box configuration settings. However, in some cases, you may need to tweak configuration parameters for maximum performance. Inside SQL Server 2005: The Storage Engine (Microsoft Press, 2006) has some additional details about configuration options. In this section, we will list the options that you will most likely need to monitor to track down performance issues. The relevant configuration options can be considered to be either CPU-related or memory-related options.

### CPU-Related Configuration Options

The CPU-related configuration options are used, for example, to control the number of CPUs or sockets that can be used by a SQL Server instance, maximum degree of parallelism, and the number of workers. Some commonly used options in this category include the following:

- Affinity Mask This option can be used to control the mapping of CPUs to the SQL Server process. By default, a SQL Server uses all processors available on the Server box. A general recommendation is not to use affinity mask option, as SQL Server performs best in the default setting. You may, however, want to use this option under two situations.



- First, if you are running other applications on the box, the Windows operating system may move around process threads to different CPUs under heavy load. By using affinity masks, you can bind each SQL Server scheduler to its own CPU. This can improve performance by eliminating thread migration across processors, thereby reducing context switching.
- Second, you can use this parameter to limit the number of CPUs on which a SQL Server can run. This is useful if you are running multiple SQL Server instances on the same Server box and want to limit CPU resources taken by each SQL Server and to minimize their interference.
- Lightweight Pooling When this configuration is enabled, SQL Server makes use of Windows fibers. A worker can map to a Windows thread or to a fiber. A fiber is like a thread, but it is cheaper than normal thread because switching between two workers (that is, fiber threads) can be done in user mode instead of kernel mode. So if your workload is experiencing a CPU bottleneck with significant time spent in kernel mode and in context switching, you may benefit by enabling this option. You must test your workload with this option before enabling it in the production system, as more often than not this option may cause performance regression. You should also keep in mind that CLR integration is not supported under lightweight pooling.
- Max Worker Threads You can think of workers as the SQL Server threads that execute user or batch requests. A worker is bound to a batch until it completes. So the maximum number of workers limits the number of batches that can be executed concurrently. By default, SQL Server sets the Max Worker Threads as described in the following table.

**Number of CPUs    32-bit computer    64-bit computer**

<= 4 processors	256	512
8 processors	288	576
16 processors	352	704
32 processors	480	960

- For most installations, this default setting is fine. Each worker takes 512-KB memory on a 32-bit, 2 MB on X64, and 4 MB on IA64. To preserve memory, the SQL Server starts off with a smaller number of workers. The pool of workers grows or shrinks based on the demand. You may want to change this configuration under two conditions. The first is if you know that your application uses a smaller number of workers. By configuring it to a lower number, the SQL Server does not need to reserve memory for the maximum number of workers. The second is if you have many long-running batches (presumably involving lock waits), such that the number of workers needed may exceed the default configuration. This is not a common case, and you may want to look at your application design to analyze this.
- Max Degree of Parallelism This configuration parameter controls the maximum number of processors or cores that can be deployed to execute a query in parallel. Parallel queries provide better response time but take more CPU resources. You need to look into this configuration parameter if you are encountering CPU bottleneck, described later in this chapter.



## Memory-Related Configuration Options

The memory-related configuration options are used to control the memory consumed by SQL Server. Some of the commonly used configuration options in this category include the following:

**Max and Min Server Memory** This is perhaps the most critical of the configuration options, especially in 32-bit configurations, from a performance perspective. This configuration parameter is often confused with the total memory configured for a SQL Server, but it is not the same. It represents the configured memory for the buffer pool. SQL Server, or any database server for that matter, is a memory-hungry application. You want to make as much memory available for SQL Server as possible. The recommendation on 64 bits is to put an upper limit in place to reserve memory for the OS and allocations that come from outside the Buffer Pool. You will also need to cap the memory usage by SQL Server when other applications (including other instances of SQL Server) are running on the same Server box.

**AWE Enabled** On a 32-bit box, the SQL Server process can only address 2 GB of virtual memory, or 3 GB if you have added the /3 GB parameter to the boot.ini file and rebooted the computer, allowing the /3 GB parameter to take effect. If you have physical memory greater than 4 GB and you have enabled this configuration option, the SQL Server process can make use of memory up to 64 GB normally, and up to 16 GB if you have used /3 GB parameter. Additionally, the SQL Server process requires Lock Pages in Memory privilege in conjunction with AWE-Enabled option. There are some restrictions in terms of the SQL Server SKU and the version of the Windows operating system under which you are running. You should be aware of certain key things when using the AWE option. First, though it allows the SQL Server process to access up to 64 GB of memory for the buffer pool; the memory available to query plans, connections, locks, and other critical structures is still limited to less than 2 GB. Second, the AWE-mapped memory is non pageable and can cause memory starvation to other applications running on the same Server box. Starting with SQL Server 2005, the AWE memory can be released dynamically, but it still cannot be paged out. SQL Server may release this memory in response to physical memory pressure. Third, this option is not available on a 64-bit environment. However, if the SQL Server process has been granted Lock Pages in Memory privilege, the buffer pool will lock pages in memory and these pages cannot be paged out.

## Best Practices on Performance Tuning

- Regularly monitor your SQL Servers for blocked transactions.
- Regularly monitor system performance using System Monitor. Use System Monitor for both real-time analysis and for historical/baseline analysis.
- If running SQL Server 2005, SP2 or later, install the free SQL Server Performance Dashboard. It can be used for real-time monitoring and performance troubleshooting.
- Regularly monitor activity using Profiler. Be sure that traces are taken during the busiest times of the day so you get a more representative trace of what is



going on in each server. When running the Profiler, do not collect more data than you need to collect.

- Perform performance monitoring from a computer that is not the SQL Server you are monitoring. Run monitoring tools on a separate desktop or server.

### **Practical Trouble shooting on Performance**

This document was prepared based on the experience I had in the project (screenshots/queries/examples) and from some blogs (content) on the web.

I divided this into two parts:

Part A discuss about how to setup counters and how to collect the data for analysis.

Part B discusses about the performance counters which are to be used in sql server and threshold limits.

#### **A. SQL Server: Performance Counters Setup-Collect-Analyze**

#### **B. SQL Server: Performance Counters- Thresholds**

### **SQL Server: Performance Counters Setup-Collect-Analyze**

This document was prepared based on the experience I had in the project (screenshots/queries/examples) and from some blogs (content) on the web.

Table of Contents

Setting up performance monitor counters

Case 1: Collection of performance data with SQL Server Agent jobs

Case 2: Collection of performance data with SQL Trace

Case 3: Collection of performance data using command line tool

Case 4: Load the performance data to SQL Server

Typical practical issue faced in the project and analysis

Setting up Performance Monitor Counters

Step 1:

How can we setup Performance Monitor to collect data all of the time, so we can go back and review the data when needed?

- Perf-Mon comes with all OS versions – best to use this as it is free ☺
- There are many ways to configure this tool : I would like to share the method that I have been using and it how it has worked for me

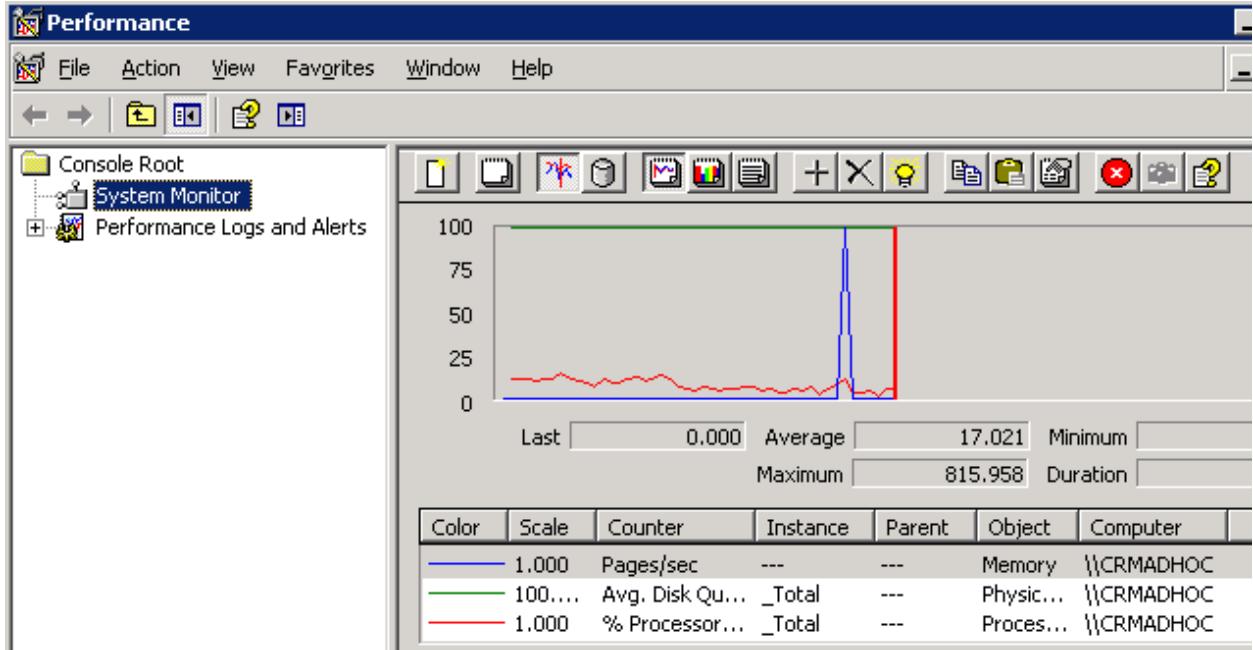
P.S -> There is no exact threshold for lot of the counters to determine if there is an issue on your server, so it is very important to collect performance counter data while the server is running healthy, so you can give yourself a baseline to measure against ☺

#### **Start Perfmon**

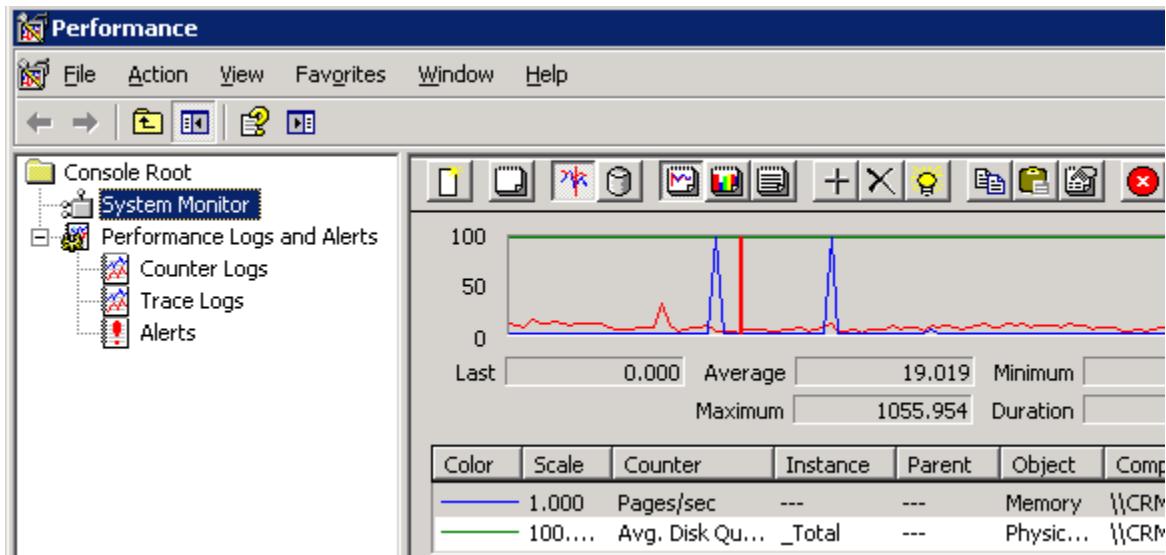
There are many ways to start perfmon and here are a few methods. Here it goes "Click **Start**, click **Run**, and then type **perfmon.msc** or **perfmon** in the Open box". When it starts you will get a screen like the following

308



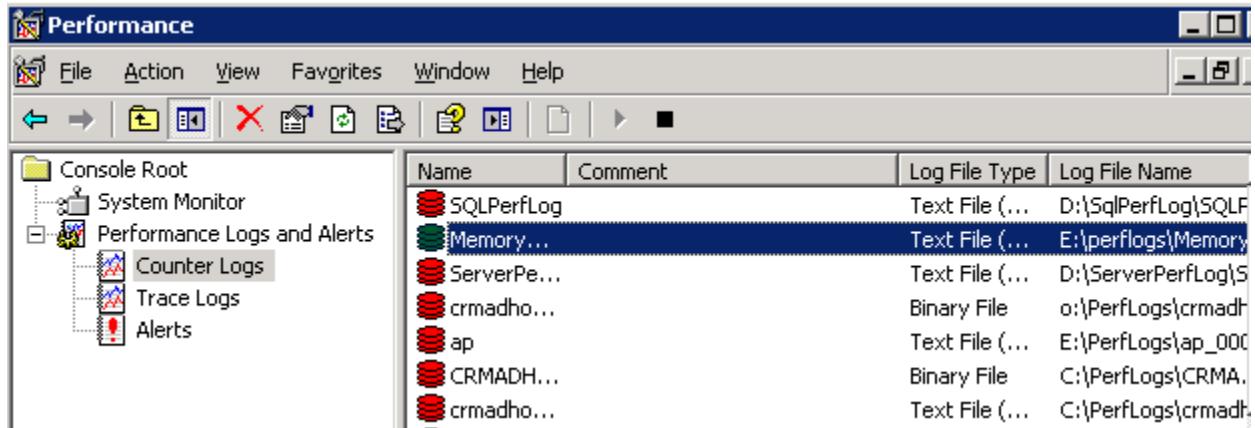


Click the plus sign next to "Performance Logs and Alerts" to expand.



Click "Counter Logs"

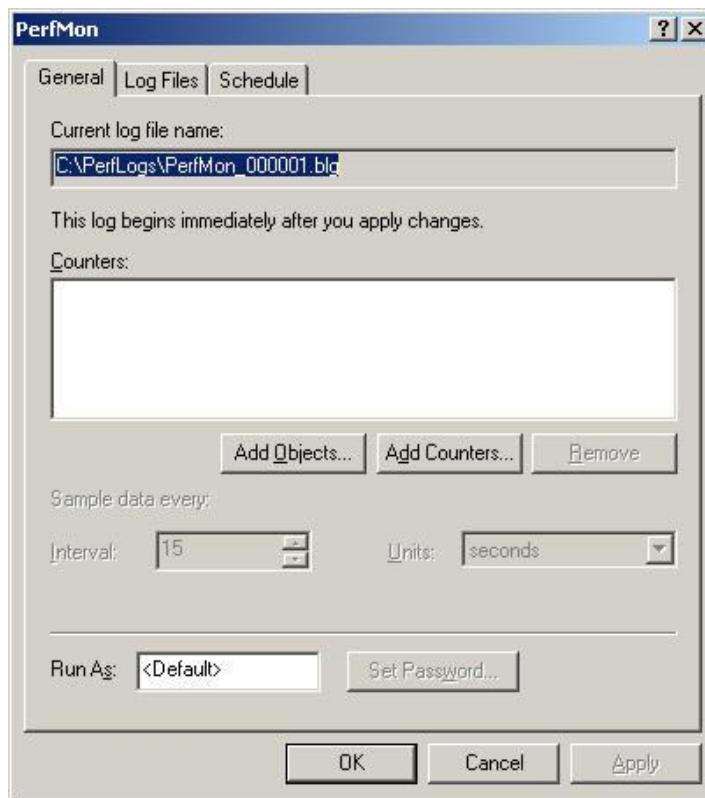




On the menu select Action -> New Log Settings and enter a name that makes sense to you. I usually put either the server name or a generic name like "PerfMon". If you store these files in a central place for several servers, it helps to use the server name as the filename. For this example, I am just using a generic name and clicking "OK"



Now it will open a new window as below.



**Setup PerfMon:** There are three areas that you have to setup.

1. **Counters to monitor:** what are the areas that you want to monitor
2. **Log file type:** format for storing the data
3. **Schedule:** how often you want to track and store the collected data

### Setting up counters to monitor

Let's add some counters. Click "Add Counters..." to choose individual counters or if you wish you can choose "Add Objects...". One thing to note is that if you select "Add Objects..." you may add counters that you never need and it will not only waste storage, but also resources from the system, so I don't recommend it.

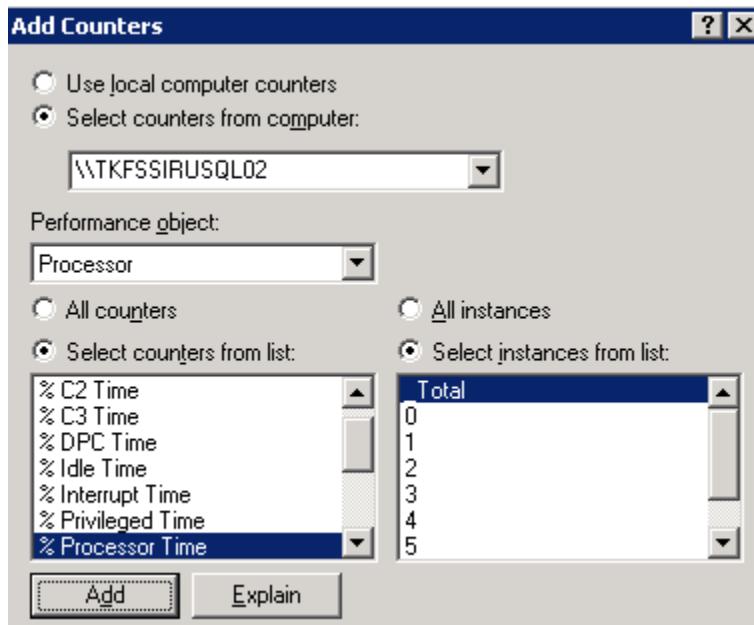
So, let's choose three counters for our example.

#### Counter 1:

Performance objects: Processor

Select counters from list: % Processor Time

Choose Select instance from list: \_Total



#### Counter 2:

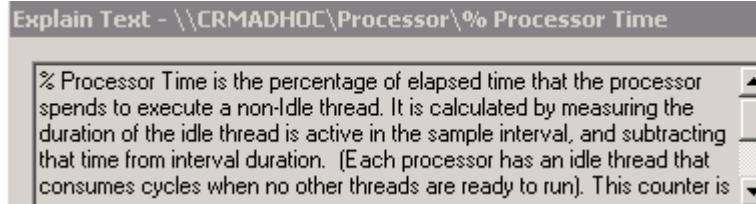
Performance objects: Memory

Select counters from list: Pages/sec

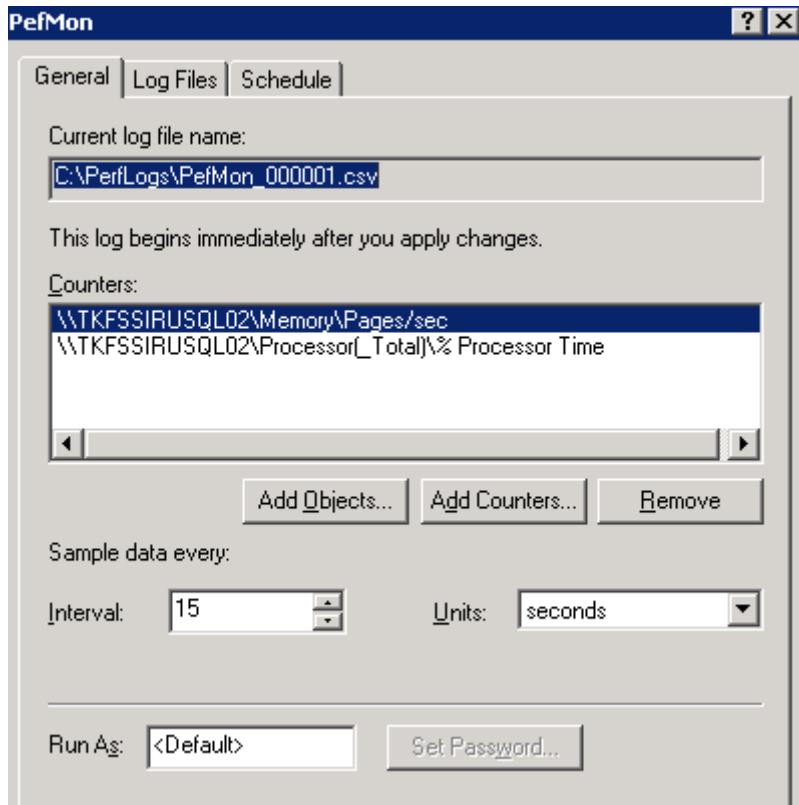
Choose Select instance from list: empty

If you are not sure what information the counter will give you, you can click "Explain", it will open another window and offer an explanation of the counter that you chose. Here is the example of Counter 1, % Processor Time.





Once you are done, your screen should like the below screenshot.



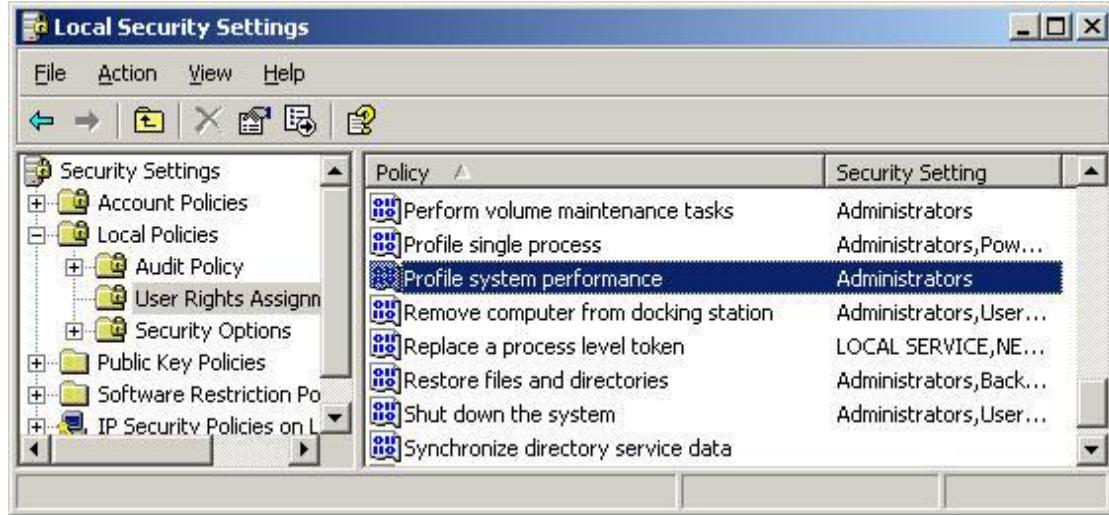
For production servers, obviously you don't want to collect data every 15 seconds which may put too much pressure on your server and also waste storage as well. Depending on what you do, you may want to choose to collect every 1 min to every 15min. I do not recommend setting longer than 10 min though, because that may not capture enough information to troubleshoot an issue. I will choose every 1 minute for this example.

Now, if you are collecting to the server locally, you do not need to set "Run As:" but if you are collecting data from a remote server, then you need to set the service account, which has proper permissions to collect the data. The service account needs to have at least "Profiler system performance" rights from Local Security Settings.

For more information, you can read this article:

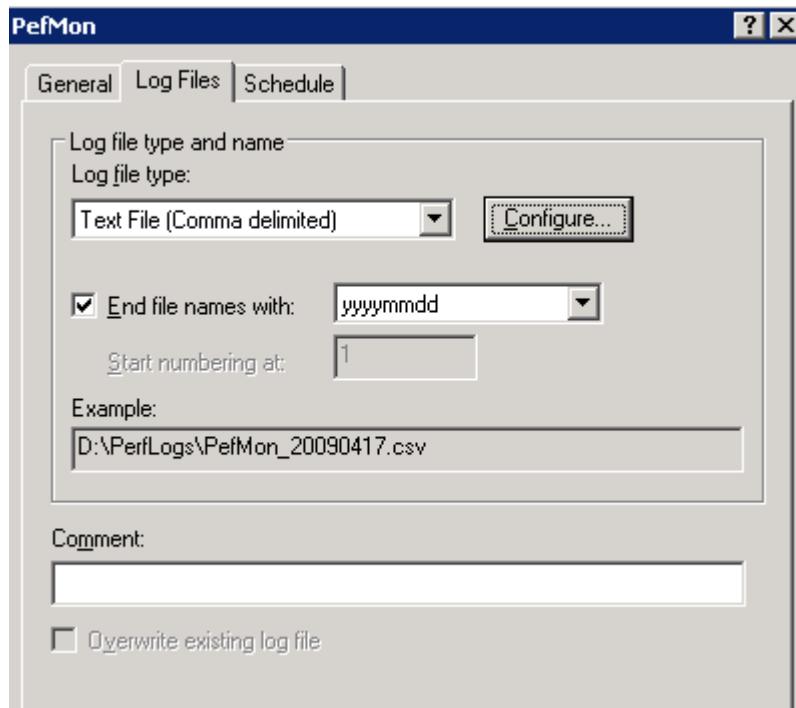
<http://www.microsoft.com/technet/prodtechnol/windows2000serv/reskit/qp/551.mspx?mfr=true>





### Setting "Log Files" type.

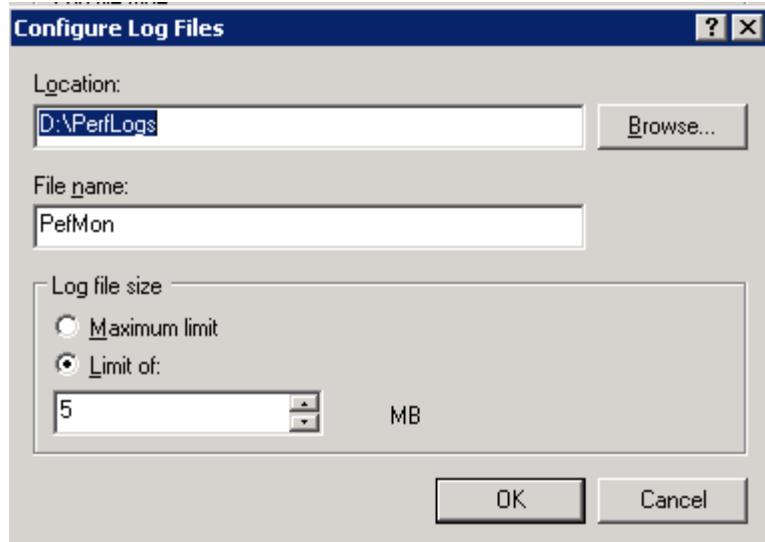
This will decide how the data will be stored once it is collected. For this example I chose "**Text file (Comma delimited)**" and chose "**End file names with:** **yyyymmdd**"



And then choose "**Configure**".

*Don't save the perf logs data on C\$ as there might be chance that C\$ will run out of space on the C drive which then impacts the O/S*



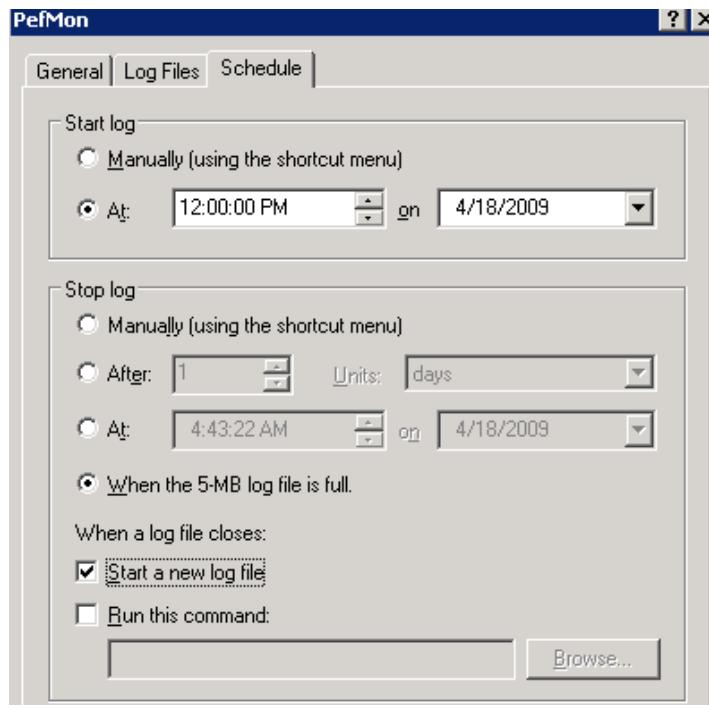


### Schedule

this is how I usually setup my schedules.

Start Log At: 12:00:00 PM on **FOLLOWING DAY** and Stop Log After **1 days** and choose "**Start a new log file**"

Here is the example, assume I set this up on 4/18/2009, here is the example how it will look like.

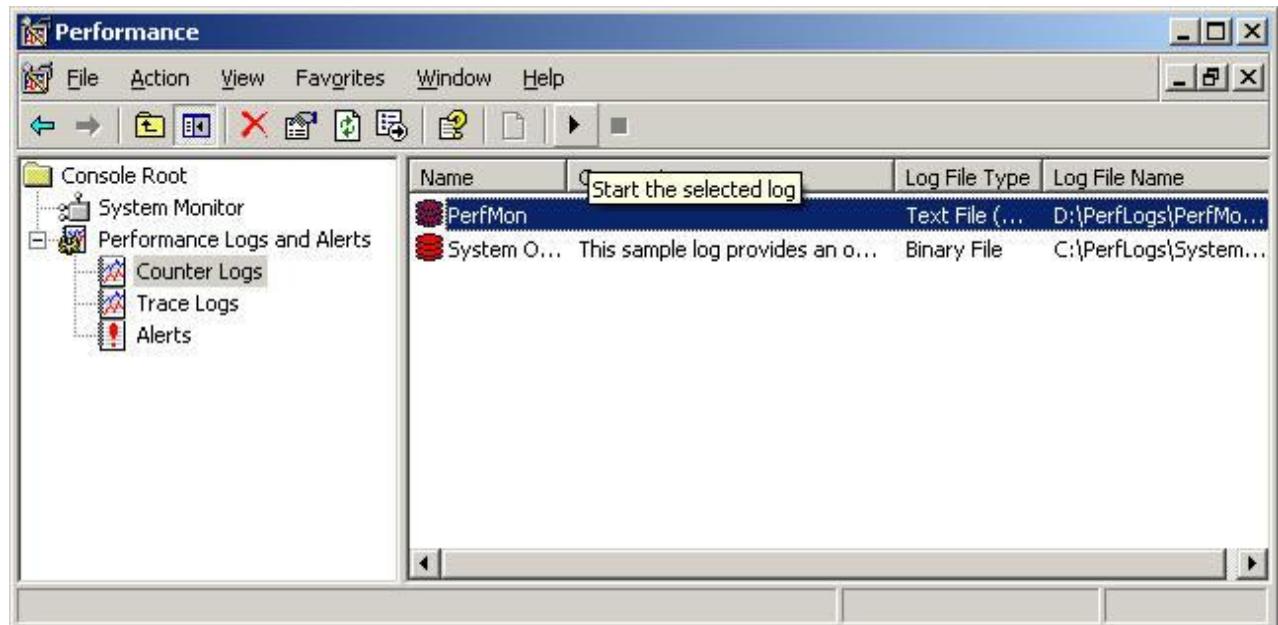


This will create one file each day with a filename like  
**D:\PerfLogs\PerfMon\_20090417.csv** Click "OK" to finish this step.

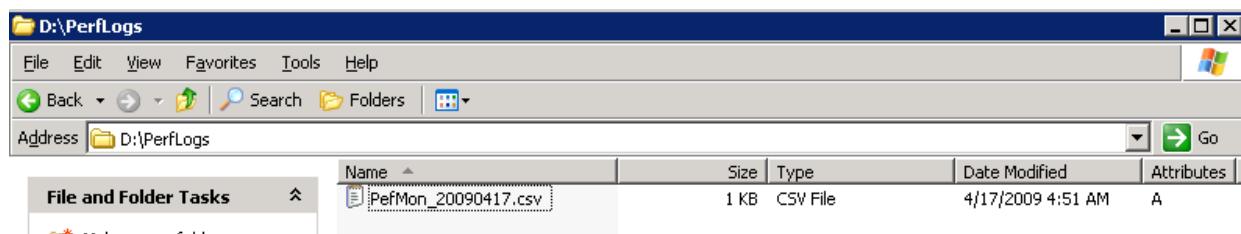
*Before you setup the schedule, you can first run it manually by clicking this icon*



*This will let you make sure you are collecting the correct data before scheduling it to run.*



For the test, check to make sure it created the file correctly.



To look at the data using Excel, just double click on the CSV file that was created to see if you are collecting the counters correctly.



*Tip> Once you are done testing, go back to the "Schedule" section and reset the schedule correctly once again especially "start a new file" section.*

*Tip> Also setup a script to delete old files, so you don't run out of space on your file system.*

Now I will show you how I use Excel to analyze the data to help determine where your bottlenecks may be and also an easy way to create quick reports and charts for your SQL Servers.

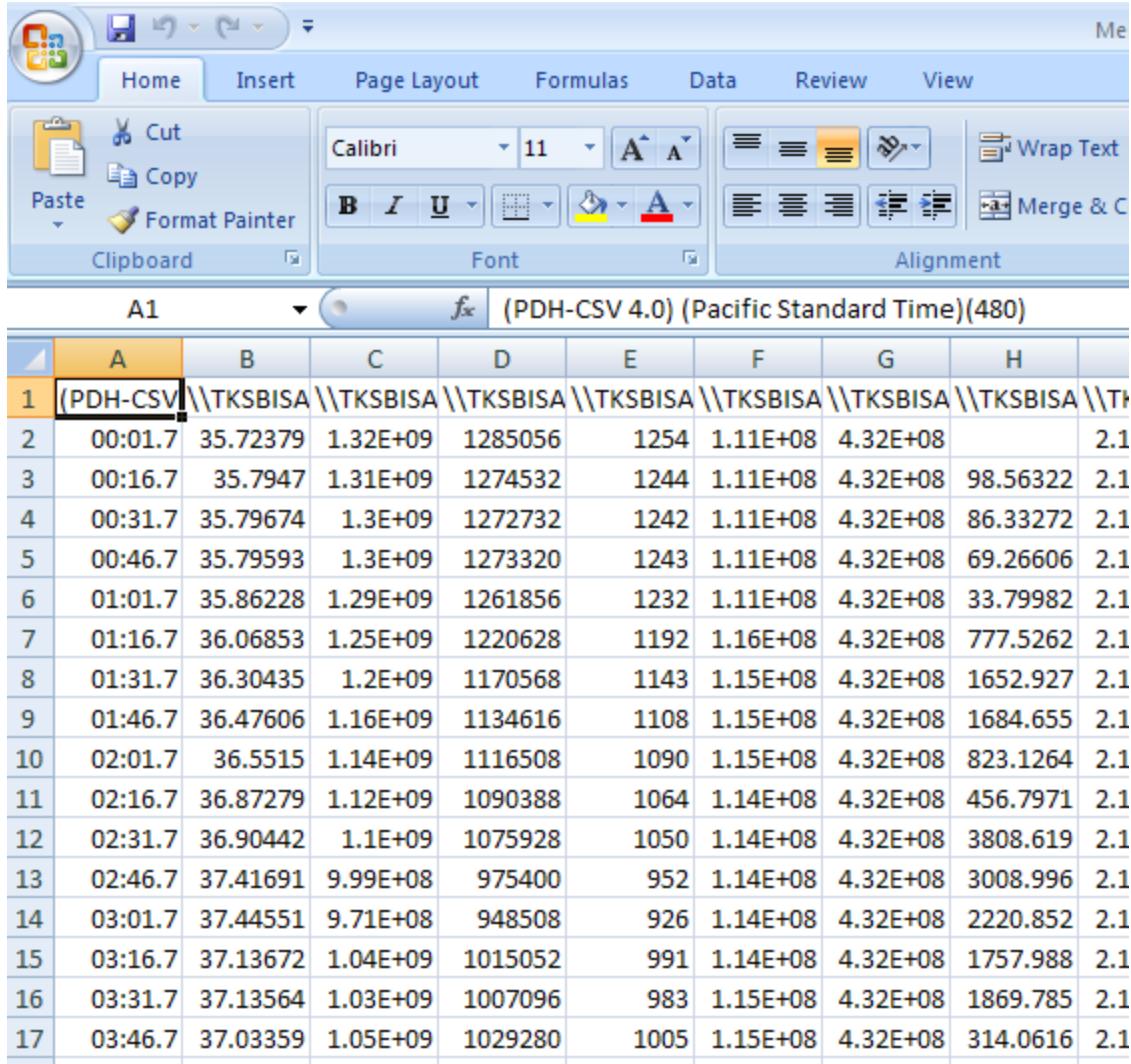
Before we get started here are a couple of things you will need for this tip.

- Microsoft Excel 2007 - you also can use Excel 2003 or earlier version but for this tip, I used the latest version.
  - Perfmon trace files at least one day in "**csv**" format. - If you have a file in "blg" format, you can easily convert it by using the "[relog](#)" tool.

## **Step 1: Open the csv file**

Once you have collected the performance data you can open the csv file using Excel and you should see data similar to the following.





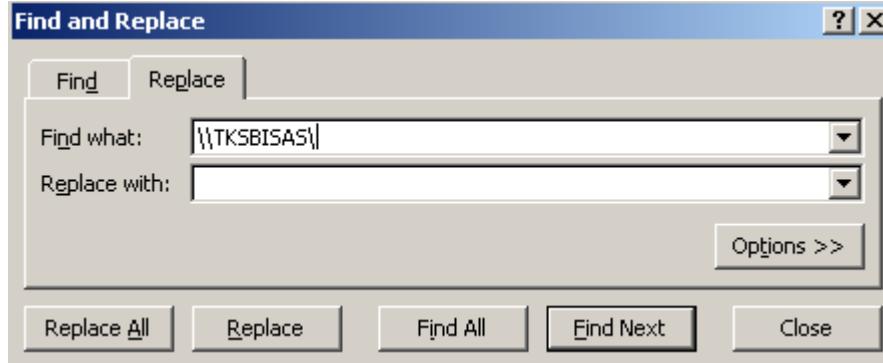
	A	B	C	D	E	F	G	H	
1	(PDH-CSV	\\\TTSBISA	\\\T						
2	00:01.7	35.72379	1.32E+09	1285056	1254	1.11E+08	4.32E+08		2.1
3	00:16.7	35.7947	1.31E+09	1274532	1244	1.11E+08	4.32E+08	98.56322	2.1
4	00:31.7	35.79674	1.3E+09	1272732	1242	1.11E+08	4.32E+08	86.33272	2.1
5	00:46.7	35.79593	1.3E+09	1273320	1243	1.11E+08	4.32E+08	69.26606	2.1
6	01:01.7	35.86228	1.29E+09	1261856	1232	1.11E+08	4.32E+08	33.79982	2.1
7	01:16.7	36.06853	1.25E+09	1220628	1192	1.16E+08	4.32E+08	777.5262	2.1
8	01:31.7	36.30435	1.2E+09	1170568	1143	1.15E+08	4.32E+08	1652.927	2.1
9	01:46.7	36.47606	1.16E+09	1134616	1108	1.15E+08	4.32E+08	1684.655	2.1
10	02:01.7	36.5515	1.14E+09	1116508	1090	1.15E+08	4.32E+08	823.1264	2.1
11	02:16.7	36.87279	1.12E+09	1090388	1064	1.14E+08	4.32E+08	456.7971	2.1
12	02:31.7	36.90442	1.1E+09	1075928	1050	1.14E+08	4.32E+08	3808.619	2.1
13	02:46.7	37.41691	9.99E+08	975400	952	1.14E+08	4.32E+08	3008.996	2.1
14	03:01.7	37.44551	9.71E+08	948508	926	1.14E+08	4.32E+08	2220.852	2.1
15	03:16.7	37.13672	1.04E+09	1015052	991	1.14E+08	4.32E+08	1757.988	2.1
16	03:31.7	37.13564	1.03E+09	1007096	983	1.15E+08	4.32E+08	1869.785	2.1
17	03:46.7	37.03359	1.05E+09	1029280	1005	1.15E+08	4.32E+08	314.0616	2.1

### Step 2: Adjust the format

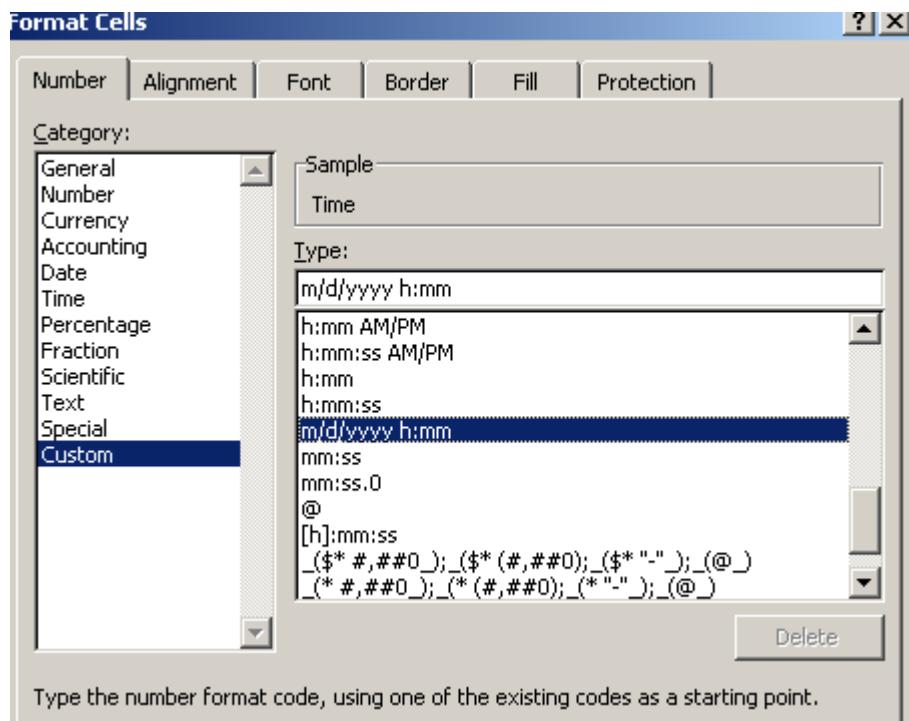
To allow easy reporting of the data there are a few things that I do to adjust the data.

- Replace server name with an empty string - it helps to make reading the counter names easier. In this case I am replacing "\\\TTSBISA\" the name of my server to nothing. (*This is optional, but recommended*)



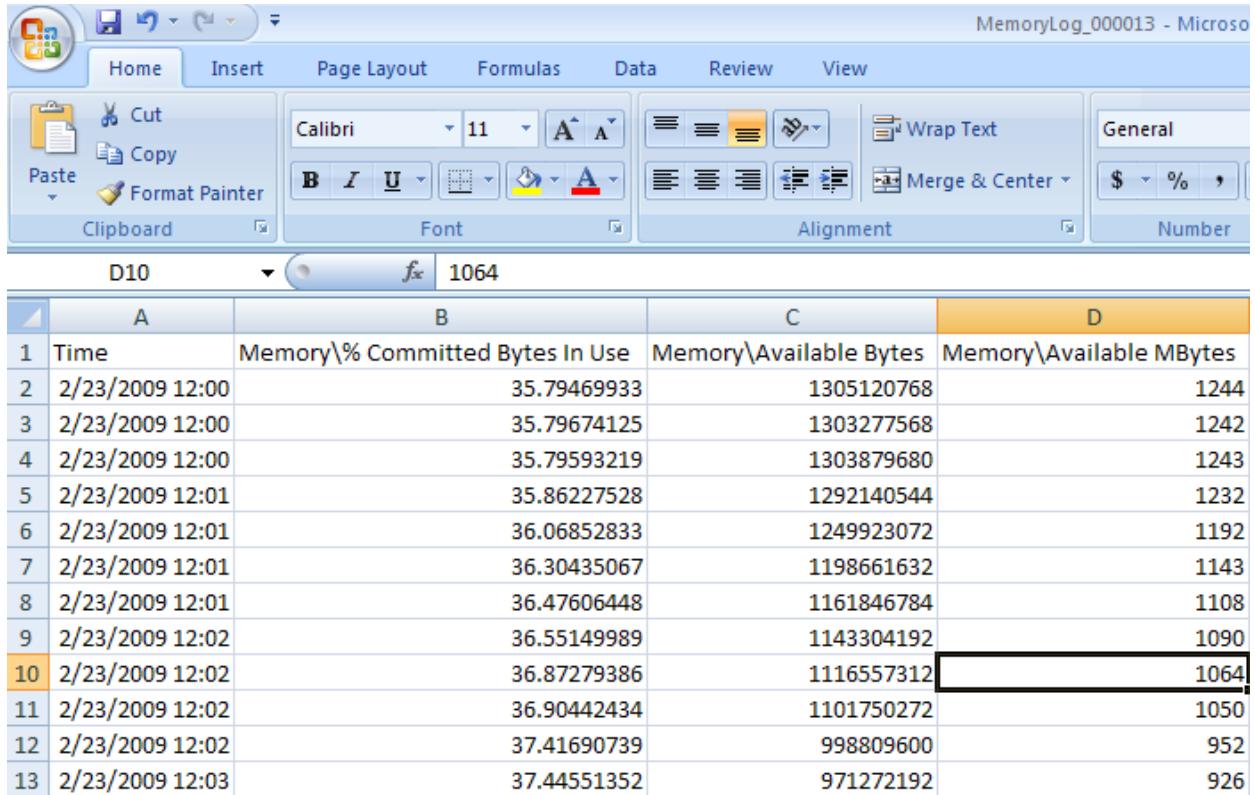


- Cell - A1: Replace "(PDH-CSV 4.0) (Pacific Standard Time)(480)" with "Time" (*Optional, but recommended*)
- Delete the second row - very often, the first data row has bad data
- Change **COLUMN A** cell format to "date time"



Final look before we start using it the data.





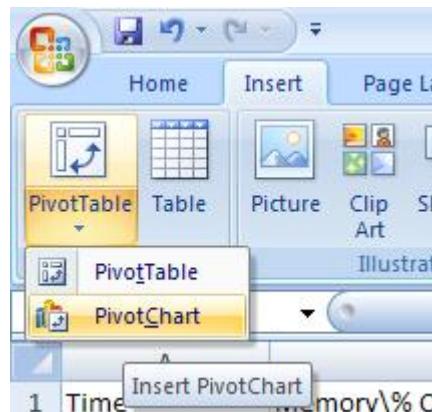
The screenshot shows a Microsoft Excel spreadsheet titled "MemoryLog\_000013 - Microsoft Excel". The ribbon menu is visible at the top with tabs for Home, Insert, Page Layout, Formulas, Data, Review, and View. The Home tab is selected. The clipboard section of the ribbon shows "Cut", "Copy", "Paste", and "Format Painter". The font section shows "Calibri 11pt" and bold/italic/underline options. The alignment section shows "Wrap Text", "Merge & Center", and "General" number format. The table below has columns A, B, C, and D. Column A contains row numbers from 1 to 13. Column B contains "Time" and "Memory \% Committed Bytes In Use" values. Column C contains "Memory \Available Bytes" values. Column D contains "Memory \Available MBytes" values. Row 10 is highlighted in orange, and cell D10 (row 10, column D) contains the value 1064.

	A	B	C	D
1	Time	Memory \% Committed Bytes In Use	Memory \Available Bytes	Memory \Available MBytes
2	2/23/2009 12:00	35.79469933	1305120768	1244
3	2/23/2009 12:00	35.79674125	1303277568	1242
4	2/23/2009 12:00	35.79593219	1303879680	1243
5	2/23/2009 12:01	35.86227528	1292140544	1232
6	2/23/2009 12:01	36.06852833	1249923072	1192
7	2/23/2009 12:01	36.30435067	1198661632	1143
8	2/23/2009 12:01	36.47606448	1161846784	1108
9	2/23/2009 12:02	36.55149989	1143304192	1090
10	2/23/2009 12:02	36.87279386	1116557312	1064
11	2/23/2009 12:02	36.90442434	1101750272	1050
12	2/23/2009 12:02	37.41690739	998809600	952
13	2/23/2009 12:03	37.44551352	971272192	926

---

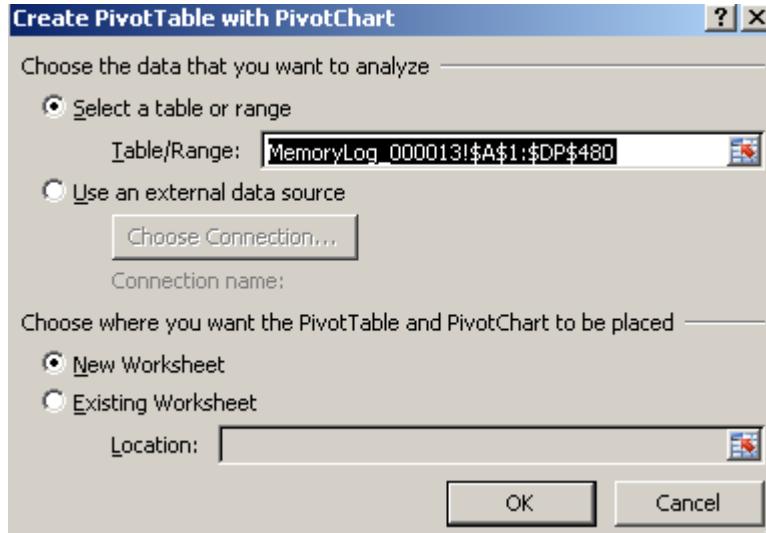
### Step 3: Create PivotTable with PivotChart

- From the Insert menu select PivotTable and then select PivotChart as shown below

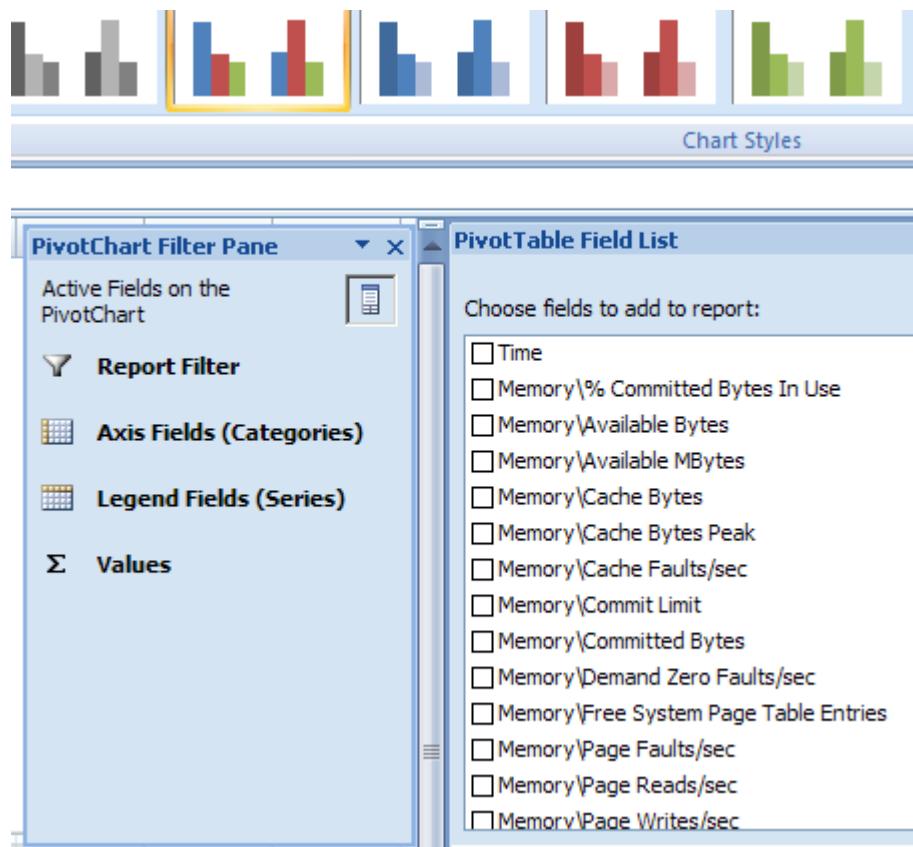


- Take the default settings and click "OK"





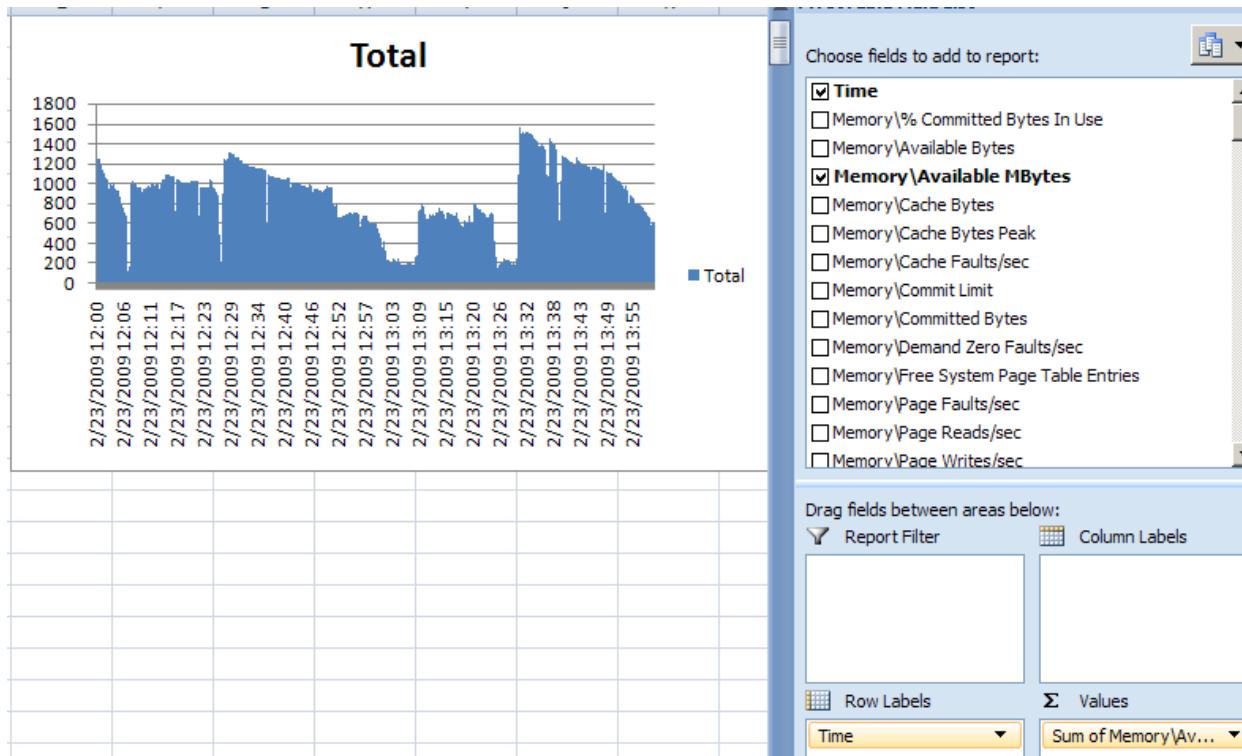
- After you select the above you will get a screen similar to the following. (to get a bigger workspace area you can close the "PivotChart Filter Pane")



### **Step 4: Let's generate our first graph**

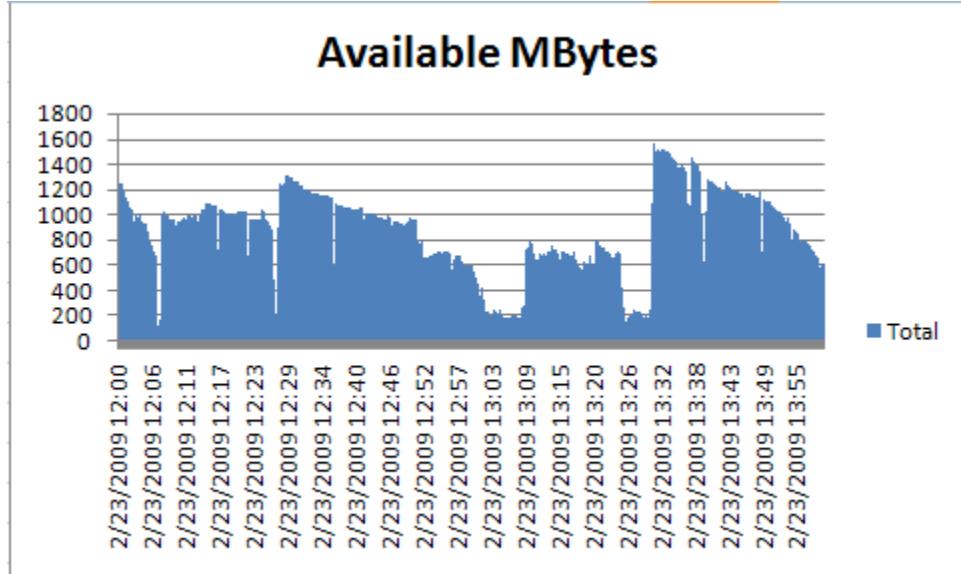
For this example we will look at CPU

- From the "PivotTable Field List" select "Time" and drag it into the "Axis Fields (Categories)" area
- From the "PivotTable Field List" select "Memory\Available MBytes" and drag it into the "Values" area
- At this point you will have a graph similar to the one shown below

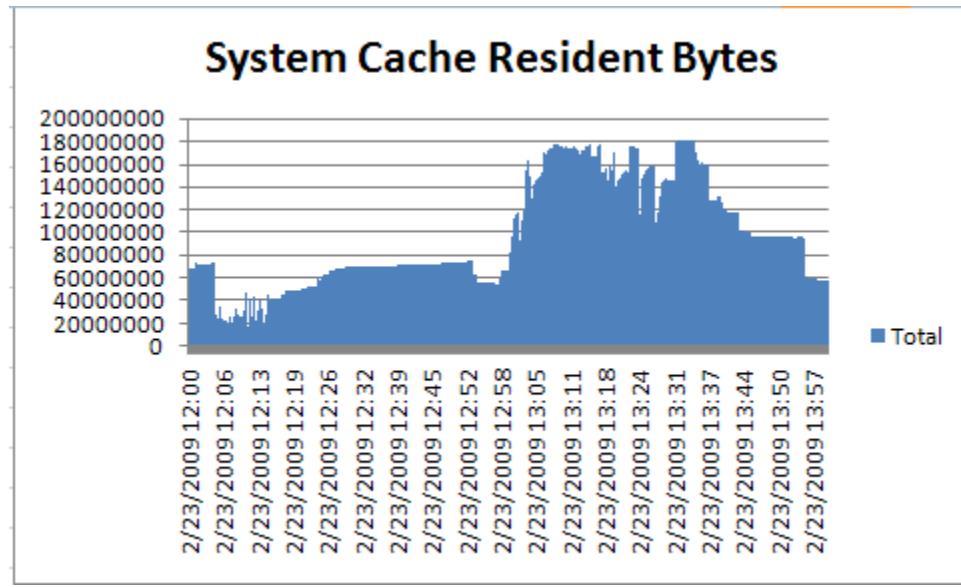


- You can now just select the chart and copy and paste it into a report, an email, Word document etc... as shown below





- If you want to change it from processor time to batch requests you can remove "Memory\Available MBytes" and select "Memory\System Cache Resident Bytes" and you will get a chart like below



#### Case 1:

Most customers asked me how I can schedule the counter [automatically] in such a way that it runs during processing window/during data load window.

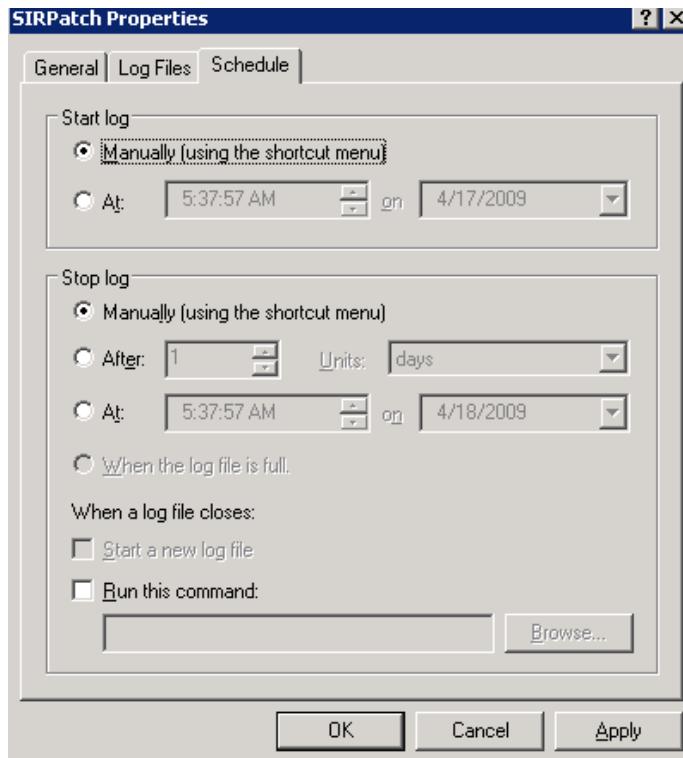
Here is the solution:

322

In the schedule window please select "Manually (using the shortcut option)" under Start Log and Stop Log. Since we will be starting and stopping this from the



command line we can set all of the settings to manual. Once you are done click OK and your Counter Log has been saved.



To start and stop this collection of counters called **SIRPatch** we can issue the following commands from the Windows command line, from a batch job or from a job step in a SQL Server job.

#### To start

```
logman start SIRPatch
```

#### To Stop

```
logman stop SIRPatch
```

After you start and stop the data collection, it will create a file such as D:\PerfLogs\SIRPatch\_000001.csv. This file can then be opened with the Performance Monitor tool to see the data for the counters that you just collected.

If you run the commands again it will create a new file called D:\PerfLogs\SIRPatch\_000002.blg, so the first file is not overwritten.

#### Case 2:

When troubleshooting a SQL Server performance problem, one of the tools to use is Profiler. This tool allows you to collect metrics on statements that are run on your SQL Server for analysis and troubleshooting. The problem with using Profiler is that it is a client tool and unless it is run on the server itself the connection may be lost and your trace stops. This usually happens right before the problem you're trying to troubleshoot occurs and you don't end up collecting that valuable information you need.



One alternative to using Profiler is to run a Server Side Trace. This process runs on the server and collects trace statistics pretty much the same way that you do using Profiler, but the process involves running a T-SQL script to start, run and stop the trace vs. using the Profiler GUI.

The server side trace can be modified to collect any event that the trace process can capture, but for this example we are just looking at SQL: StmtCompleted events which occur when a T-SQL statement has completed. For a complete list of events please check the following location.

<http://msdn.microsoft.com/en-us/library/aa260314.aspx>

EventNumber	Event	Description
45	SQL:StmtCompleted	Occurs when the Transact-SQL statement has completed.

In addition to collecting information on certain events, you can also specify what data to collect. In this example we are collecting the statements or Text Data, the SPID, Duration etc... For a complete list of columns

<http://msdn.microsoft.com/en-us/library/aa260314.aspx>

Column Number	Column	Description
1	TextData	Text value dependent on the event class that is captured in the trace.
12	SPID	Server Process ID assigned by SQL Server to the process associated with the client.
13	Duration	Amount of elapsed time (in milliseconds) taken by the event. This data column is not populated by the Hash Warning event.
14	StartTime	Time at which the event started, when available.
15	EndTime	Time at which the event ended. This column is not populated for starting event classes, such as <b>SQL: BatchStarting</b> or <b>SP: Starting</b> . It is also not populated by the <b>Hash Warning</b> event.
16	Reads	Number of logical disk reads performed by the server on behalf of the event. This column is not populated by the <b>Lock: Released</b> event.
17	Writes	Number of physical disk writes performed by the server on behalf of the event.

To create the trace for these events and columns the command would look as follows:

```
*****  
/* Server Side Trace */  
*****
```

324



```

-- Declare variables
DECLARE @rc INT
DECLARE @TraceID INT
DECLARE @maxFileSize bigint
DECLARE @fileName NVARCHAR(128)
DECLARE @on bit

-- Set values
SET @maxFileSize = 5
SET @fileName = N'C:\TestTrace'
SET @on = 1

-- Create trace
EXEC @rc = sp_trace_create @TraceID output, 0, @fileName, @maxFileSize, NULL

-- If error end process
IF (@rc != 0) GOTO error

-- Set the events and data to collect
EXEC sp_trace_setevent @TraceID, 45, 1, @on
EXEC sp_trace_setevent @TraceID, 45, 12, @on
EXEC sp_trace_setevent @TraceID, 45, 13, @on
EXEC sp_trace_setevent @TraceID, 45, 14, @on
EXEC sp_trace_setevent @TraceID, 45, 15, @on
EXEC sp_trace_setevent @TraceID, 45, 16, @on
EXEC sp_trace_setevent @TraceID, 45, 17, @on

-- Set Filters
-- filter1 include databaseId = 6
EXEC sp_trace_setfilter @TraceID, 3, 1, 0, 6
-- filter2 exclude application SQL Profiler
EXEC sp_trace_setfilter @TraceID, 10, 0, 7, N'SQL Profiler'

-- Start the trace
EXEC sp_trace_setstatus @TraceID, 1

-- display trace id for future references
SELECT TraceID=@TraceID
GOTO finish

-- error trap
error:
SELECT ErrorCode=@rc

-- exit
finish:
GO

```

There are basically four components to this to get this running:

325

- [sp\\_trace\\_create](#) - this procedure creates the trace and has 5 parameters
  - TraceID - the ID of the trace
  - Options - various options that can be set



- TraceFile - physical file name where you want to write the trace file
- MaxFileSize - size of the file, before closing and creating subsequent files
- StopTime - time to stop the trace
- **[sp\\_trace\\_setevent](#)** - this procedure specifies what event to capture and what column to capture
  - TraceID - the ID of the trace
  - EventID - the ID of the event you want to capture
  - ColumnID - the ID of the column you want to capture
  - On - whether you want to turn this event on or off
- **[sp\\_trace\\_setfilter](#)** - this procedure specifies the filters to set. This determines whether you include or exclude data
  - TraceID - the ID of the trace
  - ColumnID - the ID of the column you want to set the filter on
  - LogicalOperator - specifies whether this is an AND or OR operation
  - ComparisonOperator - specify whether the value is equal, greater than, less than, like, etc...
  - Value - the value to use for your comparison
- **[sp\\_trace\\_setstatus](#)**
  - TraceID - the ID of the trace
  - Status - stop, start or close a trace

To add additional events and columns you would just include additional `sp_trace_setevent` commands such as the following to collect event 10 RPC: Completed for the same columns that we were collecting above.

```
EXEC sp_trace_setevent @TraceID, 10, 1, @on
EXEC sp_trace_setevent @TraceID, 10, 12, @on
EXEC sp_trace_setevent @TraceID, 10, 13, @on
EXEC sp_trace_setevent @TraceID, 10, 14, @on
EXEC sp_trace_setevent @TraceID, 10, 15, @on
EXEC sp_trace_setevent @TraceID, 10, 16, @on
EXEC sp_trace_setevent @TraceID, 10, 17, @on
```

To start, stop and delete a trace you use the following commands.

Task	Command	Notes
To find traceid	<code>SELECT * FROM ::fn_trace_getinfo(default)</code>	This will give you a list of all of the traces that are running on the server.
To start a trace	<code>sp_trace_setstatus traceid, 1</code>	TraceId would be the value of the trace
To stop a trace	<code>sp_trace_setstatus traceid, 0</code>	TraceId would be the value of the trace
To close and delete a trace	<code>sp_trace_setstatus traceid,0</code> <code>sp_trace_setstatus traceid, 2</code>	To delete you need to stop the trace first and then you can delete the trace. This will close out the trace file that is written.

Once the data has been collected you can load the data into a trace table and then run queries against the trace file. Following are some commands that can be used to load the trace data into a trace table.



Task	Command	Notes
To load a trace	<pre>--Load into a new table SELECT * INTO sqlTableToLoad FROM ::fn_trace_gettable('traceFileName', DEFAULT)  --Load into an existing table INSERT INTO sqlTableToLoad SELECT * FROM ::fn_trace_gettable('traceFileName', DEFAULT)</pre>	<ul style="list-style-type: none"> <li>sqlTableToLoad – replace this with the table where you will load the data to</li> <li>traceFileName – use the correct path for the file that you will be reading the data from. If you are on the server use the UNC path.</li> <li>default – if this is set to default the load will load the file you specified as well as all additional sequenced files that exist. If you want to only load one file change the word 'default' to a number of files you want to load.</li> </ul>
To query the table	<pre>SELECT * FROM sqlTableToLoad</pre>	

Example:

So let's say we have a job called LoadOLAPData that runs a stored procedure called spLoadOLAPData. To collect the performance and trace data for this job we can setup additional job steps to start these processes before the process and then stop them after the process.

Before we get into the job steps, let's assume the following have been created already.

1. Stored procedure spLoadOLAPData exists
2. The performance counter log called "PerfMon" has been setup
3. A stored procedure called "spStartTrace" that has our server side trace settings has been setup. This stored procedure takes one parameter called @filename that is used for the name of the trace files. The default path is "D:\perflogs".

Job Step	Command	Command Type	Purpose
1	spStartTrace 'LoadOLAPData'	T-SQL	Starts the server side trace with an output file name of D:\perflogs\LoadOLAPData.trc
2	logman update sqlcounters -o D:\perflogs\LoadOLAPData	CmdExec	This modifies the output file name for the performance counters to D:\perflogs\LoadOLAPData.

3	logman start perfmon	CmdExec	This starts the performance counter collection.
4	EXEC spLoadOLAPData	T-SQL	This is our normal job step to load the data or whatever the job does.
5	<pre>DECLARE @traceID int SELECT @traceID=traceID FROM ::fn_trace_getinfo(default) where value = 'd:\perflogs\' + 'LoadOlapData'  EXEC sp_trace_setstatus @traceID, 0  EXEC sp_trace_setstatus @traceID, 2</pre>	T-SQL	This step identifies which trace has been setup for the LoadOlapData process. It gets the TraceID and stops and then closes the trace, so you can see the results.
6	logman stop perfmon	CmdExec	This stops the performance counter collection.

When the job is run and completes there will be at least two files created one for the performance counters and a second for the trace. These files can then be loaded using either Performance Monitor or Profiler so you can see the results of the job

### Case 3:

TYPEPERF.EXE: Collect performance data with command line tool ☺

TYPEPERF.EXE is a command line tool included with the Windows operating system that writes performance data to the command window or to a file



```
C:\>typeperf /?

Microsoft r TypePerf.exe <5.2.3790.0>
c Microsoft Corporation. All rights reserved.

Typeperf writes performance data to the command window or to a log file. To
stop Typeperf, press CTRL+C.

Usage:
typeperf <counter [counter ...]> | -cf <filename> | -q [object]
| -qx [object] > [options]

Parameters:
<counter [counter ...]> Performance counters to monitor.

Options:
-? Displays context sensitive help.
-f <CSV:TSV:BIN:SQL> Output file format. Default is CSV.
-cf <filename> File containing performance counters to
monitor, one per line.
-si <[[hh:]mm:]ss> Time between samples. Default is 1 second.
-o <filename> Path of output file or SQL database. Default
is STDOUT.
-q [object] List installed counters (no instances). To
list counters for one object, include the
object name, such as Processor.
-qx [object] List installed counters with instances. To
list counters for one object, include the
object name, such as Processor.
-sc <samples> Number of samples to collect. Default is to
sample until CTRL+C.
-config <filename> Settings file containing command options.
-s <computer_name> Server to monitor if no server is specified
in the counter path.
-y Answer yes to all questions without prompting.

Note:
Counter is the full name of a performance counter in
"\\"<Computer>"<Object><<Instance>>\<Counter>" format,
such as "\Server1\Processor<0>\% User Time".

Examples:
typeperf "\Processor\_Total"\% Processor Time"
typeperf -cf counters.txt -si 5 -sc 50 -f TSV -o domain2.tsv
typeperf -qx PhysicalDisk -o counters.txt
```

The goal of using TYPEPERF is to capture performance data in a repeatable way; e.g. specify your options in a batch file that you can execute as required. The default is to display the performance data in the command window; alternatively you can use the -f option to specify a CSV file (comma separated values), TSV file (tab separated values), etc.

To get started let's figure out what performance objects are available then setup TYPEPERF to capture some performance data. There are two options that you can use to get the list of performance objects on a particular machine:

- -q [object] lists the installed counters without the instances
- -qx [object] list the counters including the instances

In both cases [object] is an optional parameter which filters the list to just that object. The default is to query the performance objects on your current machine; you can include -s <computer name> to specify another machine. To get the list of counters for the SQL Server Buffer Manager object enter the following command:

TYPEPERF -q "SQLServer: Buffer Manager"

You will see output similar to the following:



```
C:\>TYPEPERF -q "SQLServer:Buffer Manager"
\SQLServer:Buffer Manager\Buffer cache hit ratio
\SQLServer:Buffer Manager\Page lookups/sec
\SQLServer:Buffer Manager\Free list stalls/sec
\SQLServer:Buffer Manager\Free pages
\SQLServer:Buffer Manager\Total pages
\SQLServer:Buffer Manager\Target pages
\SQLServer:Buffer Manager\Database pages
\SQLServer:Buffer Manager\Reserved pages
\SQLServer:Buffer Manager\Stolen pages
\SQLServer:Buffer Manager\Lazy writes/sec
\SQLServer:Buffer Manager\Readahead pages/sec
\SQLServer:Buffer Manager\Page reads/sec
\SQLServer:Buffer Manager\Page writes/sec
\SQLServer:Buffer Manager\Checkpoint pages/sec
\SQLServer:Buffer Manager\AWE lookup maps/sec
\SQLServer:Buffer Manager\AWE stolen maps/sec
\SQLServer:Buffer Manager\AWE write maps/sec
\SQLServer:Buffer Manager\AWE unmap calls/sec
\SQLServer:Buffer Manager\AWE unmap pages/sec
\SQLServer:Buffer Manager\Page life expectancy

The command completed successfully.
```

To get a list of counters with instances enter the following command:

```
TYPEPERF -qx "SQLServer:Databases" | FIND "tempdb"
```

You will see output similar to the following:

```
C:\>C:\WINDOWS\system32\cmd.exe

C:\>TYPEPERF -qx "SQLServer:Databases" | FIND "tempdb"
\SQLServer:Databases(tempdb)\Data File(s) Size (KB)
\SQLServer:Databases(tempdb)\Log File(s) Size (KB)
\SQLServer:Databases(tempdb)\Log File(s) Used Size (KB)
\SQLServer:Databases(tempdb)\Percent Log Used
\SQLServer:Databases(tempdb)\Active Transactions
\SQLServer:Databases(tempdb)\Transactions/sec
\SQLServer:Databases(tempdb)\Repl. Pending Xacts
\SQLServer:Databases(tempdb)\Repl. Trans. Rate
\SQLServer:Databases(tempdb)\Log Cache Reads/sec
\SQLServer:Databases(tempdb)\Log Cache Hit Ratio
\SQLServer:Databases(tempdb)\Bulk Copy Rows/sec
\SQLServer:Databases(tempdb)\Bulk Copy Throughput/sec
\SQLServer:Databases(tempdb)\Backup/Restore Throughput/sec
\SQLServer:Databases(tempdb)\DBCC Logical Scan Bytes/sec
\SQLServer:Databases(tempdb)\Shrink Data Movement Bytes/sec
\SQLServer:Databases(tempdb)\Log Flushes/sec
\SQLServer:Databases(tempdb)\Log Bytes Flushed/sec
\SQLServer:Databases(tempdb)\Log Flush Waits/sec
\SQLServer:Databases(tempdb)\Log Flush Wait Time
\SQLServer:Databases(tempdb)\Log Truncations
\SQLServer:Databases(tempdb)\Log Growths
\SQLServer:Databases(tempdb)\Log Shrinks
\SQLServer:Databases(tempdb)\Tracked transactions/sec
\SQLServer:Databases(tempdb)\Write Transactions/sec
\SQLServer:Databases(tempdb)\Commit table entries
```

Instances in this case (-x option) report the performance counters for the SQLServer: Databases object for each SQL Server database (there is also a \_Total instance which combines all databases). The above output was filtered to include just the tempdb database by piping to the FIND command. When you are working with a named instance of SQL Server, the performance objects will reflect the SQL Server instance name. For example I am running an instance of SQL Server 2005



Enterprise Edition which is named SQL2005; the performance objects are named MSSQL\$SQL2005 instead of SQLServer as shown above.

Use the -q or -qx options to get the list of performance counters, redirect the list to a text file, then edit the file as necessary to get just the performance counters that you want to capture. Include the -cf <filename> option on your TYPEPERF command line to get the list of counters to report on from a text file.

Now we are ready to use TYPEPERF to report some performance data. Here is a sample command:

Let us create text file named C:\PerfMon.txt with the following counters (one per line)

```
\SQLServer:Databases(_Total)\DBCC Logical Scan Bytes/sec  
\SQLServer:Databases(tempdb)\Percent Log Used  
\SQLServer:Buffer Manager\Buffer cache hit ratio  
\SQLServer:General Statistics\User Connections  
\SQLServer:Locks(_Total)\Lock Requests/sec  
  
\SQLServer:SQL Statistics\Batch Requests/sec  
  
TYPEPERF -cf PerfMon.txt
```

The above command will display the counters in the text file PerfMon.txt in the command window every second. Hit Ctrl-C to cancel.

```
C:\>TYPEPERF -cf PerfMon.txt  
<(PDH-CSU 4.0>,"\\TKFSSIRUSQL02\SQLServer:Databases(_Total)\DBCC Logical Scan B  
ytes/sec","\\TKFSSIRUSQL02\SQLServer:Databases(tempdb)\Percent Log Used","\\TKFS  
SIRUSQL02\SQLServer:Buffer Manager\Buffer cache hit ratio","\\TKFSSIRUSQL02\SQLS  
erver:General Statistics\User Connections"  
"04/17/2009 06:26:27.973","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:28.973","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:29.989","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:30.989","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:32.004","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:33.004","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:34.020","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:35.020","0.000000","59.000000","100.000000","16.000000"  
"04/17/2009 06:26:36.036","0.000000","59.000000","100.000000","16.000000"
```

Here is another example:

```
TYPEPERF -f CSV -o PerfMon.csv -si 15 -cf PerfMon.txt -sc 60
```

The above example writes the counter values to PerfMon.csv every 15 seconds. It stops after writing out the counters 60 times (i.e. 15 minutes).

An example of the output is shown below in Excel 2007:



	A	B	C	D	E
1	Time	SQLServer:Databases(_Total)\DBCC Logical Scan Bytes/sec	SQLServer:Databases(tempdb)\Percent Log Used	SQLServer:Buffer Manager\Buffer cache hit ratio	SQL
2	4/17/2009 6:30		59	99.95696457	
3	4/17/2009 6:31	0	59	99.85517741	
4	4/17/2009 6:31	0	59	99.30555556	
5	4/17/2009 6:31	0	59	100	
6	4/17/2009 6:31	0	59	100	
7	4/17/2009 6:32	0	59	100	
8	4/17/2009 6:32	0	59	100	
9	4/17/2009 6:32	0	59	100	
10	4/17/2009 6:32	0	59	100	
11	4/17/2009 6:33	0	59	100	
12	4/17/2009 6:33	0	59	100	
13	4/17/2009 6:33	0	59	100	
14	4/17/2009 6:33	0	59	100	

In the above screen shot the custom format used for the Time column is m/d/yyyy hh:mm:ss.

#### Case 4:

#### How to Load Performance data to SQL Server

Are you ready to push counter data you collected into SQL Server? ☺

To do this use relog

To use relog to input your performance monitor counters into SQL Server you must first select a database you wish to push them into and then create a system DSN to this SQL Server database (any version of SQL Server will work from SQL 2000 to SQL 2008). Use Windows Authentication as you don't want to worry about saving the account and password in the DSN.

- Open up the Data Sources (ODBC) (In the Control Panel applet in the Administrative Tools section)
- Under "User DSN" click "Add" and select SQL Server for your driver and click "Finish"
- Give your System DSN a name – call it "relog", and then point to a SQL Server in the drop down list or type in the server name and click "Next"
- Select Windows Authentication (ensure that your windows login has dbo rights in the database that you wish to write your performance monitor counters to). and click "Next"
- Select your database from the dropdown and click "Next"
- Click "Finish"
- Click "Test Data Source..." to test your data source
- If the test was successful click "OK" and click "OK" again and then close this applet

332

Now push your performance monitor counters to your SQL Server database by using the following command. Server Name is the name of the server which you collected



the data on. This name will be written to the SQL Server table DisplayToID that is created and you can query on it when you want to look at your counters.

You will want to run this command in the folder that has the "blg" file that was created or you will need to specify the path too. Also, you need to make sure the filename that was created is what is used for the command.

```
relog PerfMon.blg -f SQL -o SQL:relog!ServerName
```

## Analyze the Data

Now that the data has been loaded it is time to query the data.

The collection tables that are created when the data is loaded are the following:

- DisplayToID - table containing information about your collection
- CounterDetails - contains details about your perfmon counters
- CounterData - contains the actual counter data

Here is a sample query illustrating how to access your perfmon counter data. Here we are looking for context switches (Page Faults/sec). This will group the data in one minute intervals.

```
SELECT TOP 10 MachineName,
    CONVERT(DATETIME, CONVERT(VARCHAR(16), CounterDateTime)) as [Date],
    AVG(CounterValue) as Average,
    MIN(CounterValue) as Minimum,
    MAX(CounterValue) as Maximum
FROM CounterDetails
JOIN CounterData ON CounterData.CounterID = CounterDetails.CounterID
JOIN DisplayToID ON DisplayToID.GUID = CounterData.GUID
WHERE CounterName = 'Page Faults/sec'
GROUP BY MachineName,
    CONVERT(DATETIME, CONVERT(VARCHAR(16), CounterDateTime))
```

Here is a sample result set

	MachineName	Date	Average	Minimum	Maximum
1		2009-03-30 01:25:00.000	7.08030282168564	0	563.304959486976
2		2009-03-30 03:29:00.000	4.93235676248729	0	458.328954403987
3		2009-03-31 11:14:00.000	4.12029935164263	0	177.67615867995
4		2009-03-30 10:56:00.000	12.9325732518171	0	677.639407679966
5		2009-03-31 04:49:00.000	8.04388065161407	0	560.00671576124
6		2009-03-30 19:29:00.000	4.54754532711677	0	435.428813318894
7		2009-04-02 13:39:00.000	15.2806824783095	0	1068.69529940639



## SQL Server: Performance Counters- Thresholds

Performance – Non Disk Counters

Performance – Disk Counters

Performance – SQL Server Counters

To troubleshoot overall Database system performance issue, analyzing performance counters is the best way to start. By collecting performance counters during busy period for few days consistently and analyzing those data would give a better idea about overall system problems regarding Memory, CPU, and/or Disk I/O. Please note, for troubleshooting a particular SQL problem such as a stored procedure or a piece of T-SQL, it is better to look at the query execution plan and SQL Trace data and identify the need of redesigning a query or table indexes.

Following are some key performance counters to use while assessing a performance issues on SQL Server.

Memory and Disk I/O complements each other. Memory issues on the system could affect disk I/O and vice versa. It is very critical to carefully observe the trend of performance counters data over a long period of time to identify the real problem.

Performance Non-Disk Counters			
Object	Counter	Preferred Value	Description
Memory	<b>Available Mbytes</b>	> 100MB	Available MBytes is the amount of physical memory available to processes running on the computer, in Megabytes. Note that this counter displays the last observed value only. It is not an average.
Memory	Pages Input/Sec	< 10	Higher the value poor the performance. Pages Input/sec is the rate at which pages are read from disk to resolve hard page faults. Hard page faults occur when a process refers to a page in virtual memory that is not in its working set or elsewhere in physical memory, and must be retrieved from disk. See KB 889654.
Memory	<b>Pages/Sec</b>	See Description	<p>Pages/sec is the rate at which pages are read from or written to disk to resolve hard page faults. This counter is a primary indicator of the kinds of faults that cause system-wide delays. Investigate if over 100 pages per second on a system with a slow disk, usually even 500 pages per second on a system with a fast disk subsystem may not be an issue.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>· Values of &gt;20 pages that appear in many other sources of documentation are out of date.</li> <li>· A high value for the Memory:</li> </ul>

			Pages/sec counter does not necessarily indicate memory pressure or a System Monitor reporting error. To gain an accurate reading of your system, you must also monitor other counters (Pages Input/Sec, %Usage, %Usage Peak). See KB 889654.
Paging File	<b>%Usage</b>	< 70%	The amount of the Page File instance in use in percent. See KB 889654.
Paging File	%Usage Peak	< 70%	The peak usage of the Page File instance in percent. See KB 889654.
Process (sqlservr)	<b>%Processor Time</b>	< 80%	% Processor Time is the percentage of elapsed time that all of process threads used the processor to execution instructions. An instruction is the basic unit of execution in a computer, a thread is the object that executes instructions, and a process is the object created when a program is run. Code executed to handle some hardware interrupts and trap conditions are included in this count.
Process (msmdsrv)	<b>%Processor Time</b>	< 80%	For the SSAS. 80% is for a server which is dedicated to SSAS.
Processor	%Privileged Time	< 30% of Total %Processor Time	% Privileged Time is the percentage of elapsed time that the process threads spent executing code in privileged mode.
Processor	<b>%Processor Time</b>	< 80%	% Processor Time is the percentage of elapsed time that the processor spends to execute a non-Idle thread.
System	<b>Processor Queue Length</b>	< 4 per CPU	For standard servers with long Quanta <= 4 per CPU Excellent < 8 per CPU Good < 12 per CPU Fair

### Performance Disk Counters

When the data files are places on a SAN ignore the following!! Use the performance tools provided by the SAN vendor instead

Object	Counter	Preferred Value	Description								
PhysicalDisk	<b>Avg. Disk Sec/Read</b>	< 8ms	<p>Measure of disk latency. Avg. Disk sec/Read is the average time, in seconds, of a read of data from the disk.</p> <p>More Info:</p> <p>Reads</p> <table> <tr> <td>Excellent &lt; 08 Msec ( .008 seconds )</td> <td style="vertical-align: bottom;">335</td> </tr> <tr> <td>Good &lt; 12 Msec ( .012 seconds )</td> <td></td> </tr> <tr> <td>Fair &lt; 20 Msec ( .020 seconds )</td> <td></td> </tr> <tr> <td>Poor &gt; 20 Msec ( .020 seconds )</td> <td></td> </tr> </table>	Excellent < 08 Msec ( .008 seconds )	335	Good < 12 Msec ( .012 seconds )		Fair < 20 Msec ( .020 seconds )		Poor > 20 Msec ( .020 seconds )	
Excellent < 08 Msec ( .008 seconds )	335										
Good < 12 Msec ( .012 seconds )											
Fair < 20 Msec ( .020 seconds )											
Poor > 20 Msec ( .020 seconds )											

PhysicalDisk	<b>Avg. Disk sec/Write</b>	< 8ms (non cached)  < 1ms (cached)	Measure of disk latency. Avg. Disk sec/Write is the average time, in seconds, of a write of data to the disk. Non cached Writes Excellent < 08 Msec ( .008 seconds ) Good < 12 Msec ( .012 seconds ) Fair < 20 Msec ( .020 seconds ) Poor > 20 Msec ( .020 seconds )  Cached Writes Only Excellent < 01 Msec ( .001 seconds ) Good < 02 Msec ( .002 seconds ) Fair < 04 Msec ( .004 seconds ) Poor > 04 Msec ( .004 seconds )
--------------	----------------------------	--	--



<b>SQL Performance Counters</b>			
<b>Object</b>	<b>Counter</b>	<b>Preferred Value</b>	<b>Description</b>
SQLServer:Access Methods	Forwarded Records/sec	< 10 per 100 Batch Requests/Sec	Rows with varchar columns can experience expansion when varchar values are updated with a longer string. In the case where the row cannot fit in the existing page, the row migrates and access to the row will traverse a pointer. This only happens on heaps (tables without clustered indexes). Evaluate clustered index for heap tables. In cases where clustered indexes cannot be used, drop non-clustered indexes, build a clustered index to reorg pages and rows, drop the clustered index, then recreate non-clustered indexes.
SQLServer:Access Methods	Full Scans / sec	(Index Searches/sec)/(Full Scans/sec) > 1000	This counter monitors the number of full scans on base tables or indexes. Values greater than 1 or 2 indicate that we are having table / Index page scans. If we see high CPU then we need to investigate this counter, otherwise if the full scans are on small tables we can ignore this counter. A few of the main causes of high Full Scans/sec are <ul style="list-style-type: none"> <li>• Missing indexes</li> <li>• Too many rows requested</li> </ul> Queries with missing indexes or too many rows requested will have a large number of logical reads and an increased CPU time.
SQLServer:Access Methods	Index Searches/sec	(Index Searches/sec)/(Full Scans/sec) > 1000	Number of index searches. Index searches are used to start range scans, single index record fetches, and to reposition within an index. Index searches are preferable to index and table scans. For OLTP applications, optimize for more index searches and less scans (preferably, 1 full scan for every 1000 index searches). Index and table scans are expensive I/O operations.
SQLServer:Access Methods	<b>Page Splits/sec</b>	< 20 per 100 Batch Requests/Sec	Number of page splits per second that occur as the result of overflowing index pages. Interesting counter that can lead us to our table / index design. This value needs to be low as possible. If you find out that the number of page splits is high, consider increasing the fillfactor of your indexes. An increased fillfactor

			helps to reduce page splits because there is more room in data pages before it fills up and a page split has to occur. Note that this counter also includes the new page allocations as well and doesn't necessarily pose a problem. The other place we can confirm the page splits that involve data or index rows moves are the fragmented indexes on page splits.
SQL Server:Buffer Manager	<b>Buffer Cache hit ratio</b>	> 90%	This counter indicates how often SQL Server goes to the buffer, not the hard disk, to get data. The higher this ratio, the less often SQL Server has to go to the hard disk to fetch data, and performance overall is boosted. Unlike many of the other counters available for monitoring SQL Server, this counter averages the Buffer Cache Hit Ratio from the time the last instance of SQL Server was restarted. In other words, this counter is not a real-time measurement, but an average of all the days since SQL Server was last restarted. In OLTP applications, this ratio should exceed 90-95%. If it doesn't, then you need to add more RAM to your server to increase performance. In OLAP applications, the ratio could be much less because of the nature of how OLAP works. In any case, more RAM should increase the performance of SQL Server OLAP activity.
SQL Server:Buffer Manager	Free list stalls/sec	< 2	Free list stalls/sec is the frequency with which requests for available database pages are suspended because no buffers are available. Free list stall rates of 3 or 4 per second indicate too little SQL memory available.
SQL Server:Buffer Manager	Free pages	> 640	Total number of pages on all free lists.
SQL Server:Buffer Manager	Lazy Writes/Sec	< 20	This counter tracks how many times a second that the Lazy Writer process is moving dirty pages from the buffer to disk in order to free up buffer space. Generally speaking, this should not be a high value, say more than 20 per second or so. Ideally, it should be close to zero. <sup>338</sup> If it is zero, this indicates that your SQL Server's buffer cache is plenty big and SQL Server doesn't have to free up dirty

			pages, instead waiting for this to occur during regular checkpoints. If this value is high, then a need for more memory is indicated.
SQL Server:Buffer Manager	<b>Page Life Expectancy</b>	> 300	This performance monitor counter tells you, on average, how long data pages are staying in the buffer. If this value gets below 300 seconds, this is a potential indication that your SQL Server could use more memory in order to boost performance.
SQLServer:Buffer Manager	Page lookups/sec	(Page lookups/sec) / (Batch Requests/sec) < 100	Number of requests to find a page in the buffer pool. When the ratio of page lookups to batch requests is much greater than 100, this is an indication that while query plans are looking up data in the buffer pool, these plans are inefficient. Identify queries with the highest amount of logical I/O's and tune them.
SQL Server:Buffer Manager	Page reads/sec	< 90	Number of physical database page reads issued. 80 – 90 per second is normal, anything that is above indicates indexing or memory constraint.
SQL Server:Buffer Manager	Page writes/sec	< 90	Number of physical database page writes issued. 80 – 90 per second is normal, anything more we need to check the lazy writer/sec and checkpoint counters, if these counters are also relatively high then, it's memory constraint.
SQLServer:General Statistics	<b>Logins/sec</b>	< 2	> 2 per second indicates that the application is not correctly using <b>connection pooling</b> .
SQLServer:General Statistics	<b>Logouts/sec</b>	< 2	> 2 per second indicates that the application is not correctly using connection pooling.
SQLServer:General Statistics	<b>User Connections</b>	See Description	The number of users currently connected to the SQL Server.  <b>Note:</b> It is recommended to review this counter along with "Batch Requests/Sec". A surge in "user connections" may result in a surge of "Batch Requests/Sec". So if there is a disparity (one going up and the other staying flat or going down), then that may be a cause for concern. With a blocking problem, for example, you might see user connections, lock waits and lock

			wait time all increase while batch requests/sec decreases.
SQL Server:Latches	<b>Latch Waits/sec</b>	(Total Latch Wait Time) / (Latch Waits/Sec) < 10	This is the number of latch requests that could not be granted immediately. In other words, these are the amount of latches, in a one second period that had to wait.
SQL Server:Latches	<b>Total Latch Wait Time (ms)</b>	(Total Latch Wait Time) / (Latch Waits/Sec) < 10	This is the total latch wait time (in milliseconds) for latch requests in the last second
SQL Server:Locks	Lock Wait Time (ms)	See Description"	<p>Total wait time (milliseconds) for locks in the last second.</p> <p><b>Note:</b> For "Lock Wait Time" it is recommended to look beyond the Avg value. Look for any peaks that are close (or exceeds) to a wait of 60 sec. Though this counter counts how many total milliseconds SQL Server is waiting on locks during the last second, but the counter actually records at the end of locking event. So most probably the peaks represent one huge locking event. If those events exceeds more than 60seconds then they may have extended blocking and could be an issue. In such cases, thoroughly analyze the blocking script output. Some applications are written for timing out after 60 seconds and that's not acceptable response for those applications.</p>
SQL Server:Locks	<b>Lock Waits/sec</b>	0	<p>This counter reports how many times users waited to acquire a lock over the past second. Note that while you are actually waiting on the lock that this is not reflected in this counter—it gets incremented only when you "wake up" after waiting on the lock. If this value is nonzero then it is an indication that there is at least some level of blocking occurring. If you combine this with the <b>Lock Wait Time</b> counter, you can get some idea of how long the blocking lasted. A zero value for this counter can definitively prove out blocking as a potential cause; a nonzero value will require looking at other information to determine whether it is significant.</p>



SQL Server:Locks	Number of Deadlocks/sec	< 1	The number of lock requests that resulted in a deadlock.
SQLServer:Memory Manager	<b>Total Server Memory(KB)</b>	See Description	The Total Server Memory is the current amount of memory that SQL Server is using. If this counter is still growing the server has not yet reached its steady-state, and it is still trying to populate the cache and get pages loaded into memory. Performance will likely be somewhat slower during this time since more disk I/O is required at this stage. This behavior is normal. Eventually Total Server Memory should approximate Target Server Memory.
SQLServer:SQL Statistics	<b>Batch Requests/Sec</b>	See Description	<p>This counter measures the number of batch requests that SQL Server receives per second, and generally follows in step to how busy your server's CPUs are. Generally speaking, over 1000 batch requests per second indicates a very busy SQL Server, and could mean that if you are not already experiencing a CPU bottleneck, that you may very well soon. <b>Of course, this is a relative number, and the bigger your hardware, the more batch requests per second SQL Server can handle.</b> From a network bottleneck approach, a typical 100Mbs network card is only able to handle about 3000 batch requests per second. If you have a server that is this busy, you may need to have two or more network cards, or go to a 1Gbs network card.</p> <p><b>Note:</b> Sometimes low batch requests/sec can be misleading. If there were a SQL statements/sec counter, this would be a more accurate measure of the amount of SQL Server activity. For example, an application may call only a few stored procedures yet each stored procedure does lot of work. In that case, we will see a low number for batch requests/sec but each stored procedure (one batch) will execute many SQL statements that drive CPU and other resources. As a result, many counter thresholds based</p>

			<p>on the number of batch requests/sec will seem to identify issues because the batch requests on such a server are unusually low for the level of activity on the server.</p> <p>We cannot conclude that a SQL Server is not active simply by looking at only batch requests/sec. Rather, you have to do more investigation before deciding there is no load on the server. If the average number of batch requests/sec is below 5 and other counters (such as SQL Server processor utilization) confirm the absence of significant activity, <i>then there is not enough of a load to make any recommendations or identify issues regarding scalability.</i></p>
SQLServer:SQL Statistics	<b>SQL Compilations/sec</b>	< 10% of the number of Batch Requests/Sec	The number of times per second that SQL Server compilations have occurred. This value needs to be as low as possible. If you see a high value such as over 100, then it's an indication that there are lots of adhoc queries that are running, might cause CPU usage, solution is to re-write these adhoc as stored procedure or use sp_executeSQL.
SQLServer:SQL Statistics	<b>SQL Re-Compilations/sec</b>	< 10% of the number of SQL Compilations/sec	This needs to be nil in our system as much as possible. A recompile can cause deadlocks and compile locks that are not compatible with any locking type.



### Problem Description:

One of OLAP processing job that runs daily failing with the following error

A time out occurred while waiting for memory resources to execute the query. Rerun the query

Production support team used to restart OLAP services when the job fails and restart the job again. This issue is recurring and production team manually monitoring the OLAP job during its 3 hr run window. Due to this the processing SLA is missing daily.

### Customer Request/Escalation:

Please review the configuration of the server to see why we are getting these errors almost every day. Is there anything that can be modified in the SQL or Analysis server configuration to resolve this issue?

### Observations:

Here are our observations:

- 1) SQL Server Version : SQL 2000 SP4
- 2) RAM : 8 GB
- 3) AWE enabled
- 4) sp\_configure [I have pasted output for min and max memory only]

name	minimum	maximum	config_value	run_value
max server memory (MB)	4	2147483647	3844	3844
min server memory (MB)	0	2147483647	3844	3844

SQL Server configured to use max 3.75 GB

- 5) Please find the boot.ini info on the server here:

```
[boot loader]
default=multi(0)disk(0)rdisk(0)partition(1)\WINNT
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINNT="Windows Server 2003, Datacenter"
/fastdetect /sos /PAE /3GB /debug=disable,noumex /debugport=com1
/baudrate=115200
C:\CMDCONS\BOOTSECT.DAT="Microsoft Windows Recovery Console" /cmdcons
```

I could see both the switches are present [/PAE and /3GB].

- 6) The memory settings I could see on the server for Analysis services is like below

Analysis Manager -> Server Right click properties -> Environment Window

Under Memory settings

Minimum Allocated Memory : 2047 MB

Memory Conservation Threshold : 3047 MB

If you enable the **/3 GB** switch, you should not set the **Memory conservation threshold** setting to more than approximately 2.7 GB. Setting this value slightly



below the 3-GB memory limit ensures that the cleaner thread has sufficient time to respond to low memory conditions and to reduce allocated memory before Analysis Services uses the entire 3-GB address space

- 7) I have created a perfmon log named 'MemoryLog' on OLAP Server and scheduled it to collect the data when OLAP job runs.

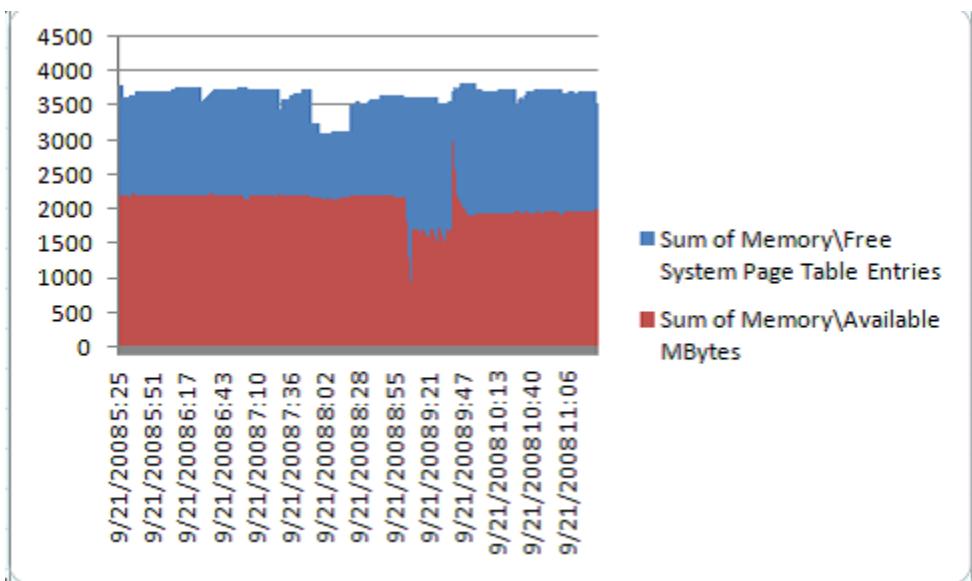
I have created two jobs to collect Memory related data when OLAP job runs on the server.

Job Names:

PerfMon-MemoryLog Start  
PerfMon-MemoryLog Stop

**Analysis done and Recommendations:**

- a) We need to change the **Memory conservation threshold** to 2765 MB [~2.7 GB] and restart Analysis services.
- b) If we have both /3GB and /PAE in boot.ini, we need to very carefully evaluate the Memory->Free System Page Table Entries counter in PerfMon. The ideal value of this counter should be > 7000. If the value is less than 7000 then connectivity and performance became problematic.
- c) I have collected perfmon data [memory object - counters] for the last 5 days when OLAP processing is in progress on the server and the graph for Free System Page Table Entries is like below [Graph is almost similar on those 5 days]. **The value is hovering around 3500.**



That's getting into the dangerously low range. We would suggest that you consider adding a /userva switch with a value of 2900 [/userva=2900]

That should get the free system PTEs into a range closer to 7000.

Please change the settings to boot.ini [if you still face performance issues on OLAP Server]



- 1) Take backup of current boot.ini file
- 2) Change the configuration of boot.ini as below:

```
[boot loader]
default=multi(0)disk(0)rdisk(0)partition(1)\WINNT
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINNT="Windows Server 2003, Datacenter"
/fastdetect /sos /PAE /3GB /Userva=2900 /debug=disable,noumex
/debugport=com1 /baudrate=115200
C:\CMDCONS\BOOTSECT.DAT="Microsoft Windows Recovery Console" /cmdcons
```

- 3) Reboot the OLAP Server

The above settings will keep the server running, but it's definitely living on the edge. These setting will delay the inevitable for a bit, and going to x64 with AS 2000 will delay the inevitable a bit longer (in that environment, the server process can access up to ~3.9GB of memory). If it's absolutely essential to stick with 2000, IA64 would be a much better option. For the longer term, AS 2008 on an x64 platform is really the way to go.



## Frequently Asked Questions:

What is RDBMS?

Relational Data Base Management Systems (RDBMS) are database management systems that maintain data records and indices in tables. Relationships may be created and maintained across and among the data and tables. In a relational database, relationships between data items are expressed by means of tables. Interdependencies among these tables are expressed by data values rather than by pointers. This allows a high degree of data independence. An RDBMS has the capability to recombine the data items from different files, providing powerful tools for data usage.

What is normalization?

Database normalization is a data design and organization process applied to data structures based on rules that help build relational databases. In relational database design, the process of organizing data to minimize redundancy. Normalization usually involves dividing a database into two or more tables and defining relationships between the tables. The objective is to isolate data so that additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database via the defined relationships.

What are different normalization forms?

1NF: Eliminate Repeating Groups

Make a separate table for each set of related attributes, and give each table a primary key. Each field contains at most one value from its attribute domain.

2NF: Eliminate Redundant Data

If an attribute depends on only part of a multi-valued key, remove it to a separate table.

3NF: Eliminate Columns Not Dependent On Key

If attributes do not contribute to a description of the key, remove them to a separate table. All attributes must be directly dependent on the primary key

BCNF: Boyce-Codd Normal Form

If there are non-trivial dependencies between candidate key attributes, separate them out into distinct tables.

4NF: Isolate Independent Multiple Relationships

No table may contain two or more 1:n or n:m relationships that are not directly related.

5NF: Isolate Semantically Related Multiple Relationships

There may be practical constraints on information that justify separating logically related many-to-many relationships.

ONF: Optimal Normal Form

A model limited to only simple (elemental) facts, as expressed in Object Role Model notation.

DKNF: Domain-Key Normal Form

A model free from all modification anomalies.

Remember, these normalization guidelines are cumulative. For a database to be in 3NF, it must first fulfill all the criteria of a 2NF and 1NF database.

346

What is Stored Procedure?

A stored procedure is a named group of SQL statements that have been previously created and stored in the server database. Stored procedures accept input parameters so that a single procedure can be used over the network by several



clients using different input data. And when the procedure is modified, all clients automatically get the new version. Stored procedures reduce network traffic and improve performance. Stored procedures can be used to help ensure the integrity of the database.

e.g. sp\_helpdb, sp\_renamedb, sp\_depends etc.

#### What is Trigger?

A trigger is a SQL procedure that initiates an action when an event (INSERT, DELETE or UPDATE) occurs. Triggers are stored in and managed by the DBMS. Triggers are used to maintain the referential integrity of data by changing the data in a systematic fashion. A trigger cannot be called or executed; the DBMS automatically fires the trigger as a result of a data modification to the associated table. Triggers can be viewed as similar to stored procedures in that both consist of procedural logic that is stored at the database level. Stored procedures, however, are not event-drive and are not attached to a specific table as triggers are. Stored procedures are explicitly executed by invoking a CALL to the procedure while triggers are implicitly executed. In addition, triggers can also execute stored procedures.

**Nested Trigger:** A trigger can also contain INSERT, UPDATE and DELETE logic within itself, so when the trigger is fired because of data modification it can also cause another data modification, thereby firing another trigger. A trigger that contains data modification logic within itself is called a nested trigger.

#### What is View?

A simple view can be thought of as a subset of a table. It can be used for retrieving data, as well as updating or deleting rows. Rows updated or deleted in the view are updated or deleted in the table the view was created with. It should also be noted that as data in the original table changes, so does data in the view, as views are the way to look at part of the original table. The results of using a view are not permanently stored in the database. The data accessed through a view is actually constructed using standard T-SQL select command and can come from one to many different base tables or even other views.

#### What is Index?

An index is a physical structure containing pointers to the data. Indices are created in an existing table to locate rows more quickly and efficiently. It is possible to create an index on one or more columns of a table, and each index is given a name. The users cannot see the indexes, they are just used to speed up queries. Effective indexes are one of the best ways to improve performance in a database application. A table scan happens when there is no index available to help a query. In a table scan SQL Server examines every row in the table to satisfy the query results. Table scans are sometimes unavoidable, but on large tables, scans have a terrific impact on performance.

Clustered indexes define the physical sorting of a database table's rows in the storage media. For this reason, each database table may have only one clustered index.

Non-clustered indexes are created outside of the database table and contain a sorted list of references to the table itself.

#### What is the difference between clustered and a non-clustered index?

A clustered index is a special type of index that reorders the way records in the table are physically stored. Therefore table can have only one clustered index. The leaf nodes of a clustered index contain the data pages.



A nonclustered index is a special type of index in which the logical order of the index does not match the physical stored order of the rows on disk. The leaf node of a nonclustered index does not consist of the data pages. Instead, the leaf nodes contain index rows.

What are the different index configurations a table can have?  
A table can have one of the following index configurations:

- No indexes
- A clustered index
- A clustered index and many nonclustered indexes
- A nonclustered index
- Many nonclustered indexes

What is the use of DBCC commands?

DBCC stands for database consistency checker. We use these commands to check the consistency of the databases, i.e., maintenance, validation task and status checks.

E.g. DBCC CHECKDB - Ensures that tables in the db and the indexes are correctly linked.

DBCC CHECKALLOC - To check that all pages in a db are correctly allocated.

DBCC CHECKFILEGROUP - Checks all tables file group for any damage.

What is a Linked Server?

Linked Servers is a concept in SQL Server by which we can add other SQL Server to a Group and query both the SQL Server dbs using T-SQL Statements. With a linked server, you can create very clean, easy to follow, SQL statements that allow remote data to be retrieved, joined and combined with local data.

Storped Procedure sp\_addlinkedserver, sp\_addlinkedsrvlogin will be used add new Linked Server.

What is Collation?

Collation refers to a set of rules that determine how data is sorted and compared. Character data is sorted using rules that define the correct character sequence, with options for specifying case-sensitivity, accent marks, kana character types and character width.

What are different type of Collation Sensitivity?

Case sensitivity

A and a, B and b, etc.

Accent sensitivity

a and á, o and ó, etc.

Kana Sensitivity

When Japanese kana characters Hiragana and Katakana are treated differently, it is called Kana sensitive.

Width sensitivity

When a single-byte character (half-width) and the same character when represented as a double-byte character (full-width) are treated differently then it is width sensitive.

What's the difference between a primary key and a unique key?

Both primary key and unique enforce uniqueness of the column on which they are defined. But by default primary key creates a clustered index on the column, where



are unique creates a nonclustered index by default. Another major difference is that, primary key doesn't allow NULLs, but unique key allows one NULL only.

What is a NOLOCK?

Using the NOLOCK query optimiser hint is generally considered good practice in order to improve concurrency on a busy system. When the NOLOCK hint is included in a SELECT statement, no locks are taken when data is read. The result is a Dirty Read, which means that another process could be updating the data at the exact time you are reading it. There are no guarantees that your query will retrieve the most recent data. The advantage to performance is that your reading of data will not block updates from taking place, and updates will not block your reading of data. SELECT statements take Shared (Read) locks. This means that multiple SELECT statements are allowed simultaneous access, but other processes are blocked from modifying the data. The updates will queue until all the reads have completed, and reads requested after the update will wait for the updates to complete. The result to your system is delay(blocking).

What is difference between DELETE & TRUNCATE commands?

Delete command removes the rows from a table based on the condition that we provide with a WHERE clause. Truncate will actually remove all the rows from a table and there will be no data in the table after we run the truncate command.

**TRUNCATE**

TRUNCATE is faster and uses fewer system and transaction log resources than DELETE.

TRUNCATE removes the data by deallocating the data pages used to store the table's data, and only the page deallocations are recorded in the transaction log.

TRUNCATE removes all rows from a table, but the table structure and its columns, constraints, indexes and so on remain. The counter used by an identity for new rows is reset to the seed for the column.

You cannot use TRUNCATE TABLE on a table referenced by a FOREIGN KEY constraint.

Because TRUNCATE TABLE is not logged, it cannot activate a trigger.

TRUNCATE can not be Rolled back using logs.

TRUNCATE is DDL Command.

TRUNCATE Resets identity of the table.

**DELETE**

DELETE removes rows one at a time and records an entry in the transaction log for each deleted row.

If you want to retain the identity counter, use DELETE instead. If you want to remove table definition and its data, use the DROP TABLE statement.

DELETE Can be used with or without a WHERE clause

DELETE Activates Triggers.

DELETE Can be Rolled back using logs.

DELETE is DML Command.

DELETE does not reset identity of the table.

When is the use of UPDATE\_STATISTICS command?

This command is basically used when a large processing of data has occurred. If a large amount of deletions any modification or Bulk Copy into the tables has occurred, it has to update the indexes to take these changes into account.

UPDATE\_STATISTICS updates the indexes on these tables accordingly.



What is SQL Profiler?

SQL Profiler is a graphical tool that allows system administrators to monitor events in an instance of Microsoft SQL Server. You can capture and save data about each event to a file or SQL Server table to analyze later. For example, you can monitor a production environment to see which stored procedures are hampering performance by executing too slowly.

Use SQL Profiler to monitor only the events in which you are interested. If traces are becoming too large, you can filter them based on the information you want, so that only a subset of the event data is collected. Monitoring too many events adds overhead to the server and the monitoring process and can cause the trace file or trace table to grow very large, especially when the monitoring process takes place over a long period of time.

Which TCP/IP port does SQL Server run on? How can it be changed?

SQL Server runs on port 1433. It can be changed from the Network Utility TCP/IP properties -> Port number.both on client and the server.

What are the authentication modes in SQL Server? How can it be changed?

Windows mode and mixed mode (SQL & Windows).

To change authentication mode in SQL Server click Start, Programs, Microsoft SQL Server and click SQL Enterprise Manager to run SQL Enterprise Manager from the Microsoft SQL Server program group. Select the server then from the Tools menu select SQL Server Configuration Properties, and choose the Security page.

Where are SQL server users names and passwords are stored in sql server?

They get stored in master db in the sysxlogins table.

Which command using Query Analyzer will give you the version of SQL server and operating system?

`SELECT SERVERPROPERTY('productversion'), SERVERPROPERTY ('productlevel')`,

What is SQL server agent?

SQL Server agent plays an important role in the day-to-day tasks of a database administrator (DBA). It is often overlooked as one of the main tools for SQL Server management. Its purpose is to ease the implementation of tasks for the DBA, with its full-function scheduling engine, which allows you to schedule your own jobs and scripts.

What is log shipping?

Log shipping is the process of automating the backup of database and transaction log files on a production SQL server, and then restoring them onto a standby server.

Enterprise Editions only supports log shipping. In log shipping the transactional log file from one server is automatically updated into the backup database on the other server. If one server fails, the other server will have the same db can be used this as the Disaster Recovery plan. The key feature of log shipping is that it will automatically backup transaction logs throughout the day and automatically restore them on the standby server at defined interval.

What command do we use to rename a db?

`sp_renamedb 'oldname' , 'newname'`

If someone is using db it will not accept sp\_renamedb. In that case first bring db to single user using sp\_dboptions. Use sp\_renamedb to rename database. Use sp\_dboptions to bring database to multi user mode.



What is sp\_configure commands and set commands?

Use sp\_configure to display or change server-level settings. To change database-level settings, use ALTER DATABASE. To change settings that affect only the current user session, use the SET statement.

What are the different types of replication? Explain.

The SQL Server 2000-supported replication types are as follows:

Transactional

Snapshot

Merge

Snapshot replication distributes data exactly as it appears at a specific moment in time and does not monitor for updates to the data. Snapshot replication is best used as a method for replicating data that changes infrequently or where the most up-to-date values (low latency) are not a requirement. When synchronization occurs, the entire snapshot is generated and sent to Subscribers.

Transactional replication, an initial snapshot of data is applied at Subscribers, and then when data modifications are made at the Publisher, the individual transactions are captured and propagated to Subscribers.

Merge replication is the process of distributing data from Publisher to Subscribers, allowing the Publisher and Subscribers to make updates while connected or disconnected, and then merging the updates between sites when they are connected.

What are the OS services that the SQL Server installation adds?

MS SQL SERVER SERVICE, SQL AGENT SERVICE, DTC (Distribution transaction coordinator)

What are three SQL keywords used to change or set someone's permissions?

GRANT, DENY, and REVOKE.

What does it mean to have quoted\_identifier on? What are the implications of having it off?

When SET QUOTED\_IDENTIFIER is ON, identifiers can be delimited by double quotation marks, and literals must be delimited by single quotation marks. When SET QUOTED\_IDENTIFIER is OFF, identifiers cannot be quoted and must follow all Transact-SQL rules for identifiers.

How to rebuild Master Database?

Shutdown Microsoft SQL Server 2000, and then run Rebuildm.exe. This is located in the Program Files\Microsoft SQL Server\80\Tools\Binn directory.

In the Rebuild Master dialog box, click Browse.

In the Browse for Folder dialog box, select the \Data folder on the SQL Server 2000 compact disc or in the shared network directory from which SQL Server 2000 was installed, and then click OK.

Click Settings. In the Collation Settings dialog box, verify or change settings used for the master database and all other databases.

Initially, the default collation settings are shown, but these may not match the collation selected during setup. You can select the same settings used during setup or select new collation settings. When done, click OK.



In the Rebuild Master dialog box, click Rebuild to start the process.

The Rebuild Master utility reinstalls the master database.

To continue, you may need to stop a server that is running.

Source: [http://msdn2.microsoft.com/en-us/library/aa197950\(SQL.80\).aspx](http://msdn2.microsoft.com/en-us/library/aa197950(SQL.80).aspx)

What are the basic functions for master, msdb, model, tempdb and resource databases?

The master database holds information for all databases located on the SQL Server instance and is the glue that holds the engine together. Because SQL Server cannot start without a functioning master database, you must administer this database with care.

The msdb database stores information regarding database backups, SQL Agent information, DTS packages, SQL Server jobs, and some replication information such as for log shipping.

The tempdb holds temporary objects such as global and local temporary tables and stored procedures.

The model is essentially a template database used in the creation of any new user database created in the instance.

The Resource Database is a read-only database that contains all the system objects that are included with SQL Server. SQL Server system objects, such as sys.objects, are physically persisted in the Resource database, but they logically appear in the sys schema of every database. The Resource database does not contain user data or user metadata.

What is data integrity? Explain constraints?

Data integrity is an important feature in SQL Server. When used properly, it ensures that data is accurate, correct, and valid. It also acts as a trap for otherwise undetectable bugs within applications.

A PRIMARY KEY constraint is a unique identifier for a row within a database table. Every table should have a primary key constraint to uniquely identify each row and only one primary key constraint can be created for each table. The primary key constraints are used to enforce entity integrity.

A UNIQUE constraint enforces the uniqueness of the values in a set of columns, so no duplicate values are entered. The unique key constraints are used to enforce entity integrity as the primary key constraints.

A FOREIGN KEY constraint prevents any actions that would destroy links between tables with the corresponding data values. A foreign key in one table points to a primary key in another table. Foreign keys prevent actions that would leave rows with foreign key values when there are no primary keys with that value. The foreign key constraints are used to enforce referential integrity.

A CHECK constraint is used to limit the values that can be placed in a column. The check constraints are used to enforce domain integrity.

A NOT NULL constraint enforces that the column will not accept null values. The not null constraints are used to enforce domain integrity, as the check constraints.

352

What is a table called, if it does not have either Cluster nor Non-cluster Index?

What is it used for?

Unindexed table or Heap. Microsoft Press Books and Book On Line (BOL) refers it as



### Heap.

A heap is a table that does not have a clustered index and, therefore, the pages are not linked by pointers. The IAM pages are the only structures that link the pages in a table together.

Unindexed tables are good for fast storing of data. Many times it is better to drop all indexes from table and then do bulk of inserts and to restore those indexes after that.

### What is BCP? When does it used?

BulkCopy is a tool used to copy huge amount of data from tables and views. BCP does not copy the structures same as source to destination.

### How do you load large data to the SQL server database?

BulkCopy is a tool used to copy huge amount of data from tables. BULK INSERT command helps to Imports a data file into a database table or view in a user-specified format.

### Can SQL Servers linked to other servers like Oracle?

SQL Server can be linked to any server provided it has OLE-DB provider from Microsoft to allow a link. E.g. Oracle has a OLE-DB provider for oracle that Microsoft provides to add it as linked server to SQL Server group.

### How to know which index a table is using?

```
SELECT table_name,index_name FROM user_constraints
```

### How to copy the tables, schema and views from one SQL server to another?

Microsoft SQL Server 2000 Data Transformation Services (DTS) is a set of graphical tools and programmable objects that lets user extract, transform, and consolidate data from disparate sources into single or multiple destinations.

### What is an execution plan? When would you use it? How would you view the execution plan?

An execution plan is basically a road map that graphically or textually shows the data retrieval methods chosen by the SQL Server query optimizer for a stored procedure or ad-hoc query and is a very useful tool for a developer to understand the performance characteristics of a query or stored procedure since the plan is the one that SQL Server will place in its cache and use to execute the stored procedure or query. From within Query Analyzer is an option called "Show Execution Plan" (located on the Query drop-down menu). If this option is turned on it will display query execution plan in separate window when query is ran again.

### What is Data Compression?

In SQL SERVER 2008 Data Compression comes in two flavors:

Row Compression

Page Compression

### Row Compression

Row compression changes the format of physical storage of data. It minimizes the metadata (column information, length, offsets etc) associated with each record. Numeric data types and fixed length strings are stored in variable-length storage format, just like Varchar. ([\\_](#))



## Page Compression

Page compression allows common data to be shared between rows for a given page. Its uses the following techniques to compress data:

Row compression.

Prefix Compression. For every column in a page duplicate prefixes are identified. These prefixes are saved in compression information headers (CI) which resides after page header. A reference number is assigned to these prefixes and that reference number is replaced where ever those prefixes are being used.

Dictionary Compression.

Dictionary compression searches for duplicate values throughout the page and stores them in CI. The main difference between prefix and dictionary compression is that prefix is only restricted to one column while dictionary is applicable to the complete page.

What is Filestream?

Filestream allows you to store large objects in the file system and have these files integrated within the database. It enables SQL Server based applications to store unstructured data such as documents, images, audios, videos etc. in the file system. FILESTREAM basically integrates the SQL Server Database Engine with New Technology File System (NTFS); it basically stores the data in varbinary (max) data type. Using this data type, the unstructured data is stored in the NTFS file system and the SQL Server Database Engine manages the link between the Filestream column and the actual file located in the NTFS. Using Transact SQL statements users can insert, update, delete and select the data stored in FILESTREAM enabled tables.

What is Dirty Read ?

A dirty read occurs when two operations say, read and write occurs together giving the incorrect or unedited data. Suppose, A has changed a row, but has not committed the changes. B reads the uncommitted data but his view of the data may be wrong so that is Dirty Read.

What is SQLCMD?

sqlcmd is enhanced version of the isql and osql and it provides way more functionality than other two options. In other words sqlcmd is better replacement of isql (which will be deprecated eventually) and osql (not included in SQL Server 2005 RTM). sqlcmd can work two modes - i) BATCH and ii) interactive modes.

UNION ALL

The UNION ALL command is equal to the UNION command, except that UNION ALL selects all values.

The difference between Union and Union all is that Union all will not eliminate duplicate rows, instead it just pulls all rows from all tables fitting your query specifics and combines them into a table. (.)

What is B-Tree?

The database server uses a B-tree structure to organize index information. B-Tree generally has following types of index pages or nodes:



**root node:** A root node contains node pointers to branch nodes which can be only one.

**branch nodes:** A branch node contains pointers to leaf nodes or other branch nodes which can be two or more.

**leaf nodes:** A leaf node contains index items and horizontal pointers to other leaf nodes which can be many.

#### What is Service Broker?

Service Broker is a message-queuing technology in SQL Server that allows developers to integrate SQL Server fully into distributed applications. Service Broker is a feature which provides facility to SQL Server to send an asynchronous, transactional message. It allows a database to send a message to another database without waiting for the response, so the application will continue to function if the remote database is temporarily unavailable.

#### What is Policy Management?

Policy Management in SQL SERVER 2008 allows you to define and enforce policies for configuring and managing SQL Server across the enterprise. Policy-Based Management is configured in SQL Server Management Studio (SSMS). Navigate to the Object Explorer and expand the Management node and the Policy Management node; you will see the Policies, Conditions, and Facets nodes. ([\\_](#))

#### What is Replication and Database Mirroring?

Database mirroring can be used with replication to provide availability for the publication database. Database mirroring involves two copies of a single database that typically reside on different computers. At any given time, only one copy of the database is currently available to clients which are known as the principal database. Updates made by clients to the principal database are applied on the other copy of the database, known as the mirror database. Mirroring involves applying the transaction log from every insertion, update, or deletion made on the principal database onto the mirror database.

#### What are Sparse Columns?

A sparse column is another tool used to reduce the amount of physical storage used in a database. They are the ordinary columns that have an optimized storage for null values. Sparse columns reduce the space requirements for null values at the cost of more overhead to retrieve nonnull values.

#### What does TOP Operator Do?

The TOP operator is used to specify the number of rows to be returned by a query. The TOP operator has new addition in SQL SERVER 2008 that it accepts variables as well as literal values and can be used with INSERT, UPDATE, and DELETE statements.

#### What is CTE?

CTE is an abbreviation Common Table Expression. A Common Table Expression (CTE) is an expression that can be thought of as a temporary result set which is defined within the execution of a single SQL statement. A CTE is similar to a derived table in that it is not stored as an object and lasts only for the duration of the query.



Which are new data types introduced in SQL SERVER 2008?

The GEOMETRY Type: The GEOMETRY data type is a system .NET common language runtime (CLR) data type in SQL Server. This type represents data in a two-dimensional Euclidean coordinate system.

The GEOGRAPHY Type: The GEOGRAPHY datatype's functions are the same as with GEOMETRY. The difference between the two is that when you specify GEOGRAPHY, you are usually specifying points in terms of latitude and longitude.

New Date and Time Datatypes: SQL Server 2008 introduces four new datatypes related to date and time: DATE, TIME, DATETIMEOFFSET, and DATETIME2.

DATE: The new DATE type just stores the date itself. It is based on the Gregorian calendar and handles years from 1 to 9999.

TIME: The new TIME (n) type stores time with a range of 00:00:00.0000000 through 23:59:59.9999999. The precision is allowed with this type. TIME supports seconds down to 100 nanoseconds. The n in TIME (n) defines this level of fractional second precision, from 0 to 7 digits of precision.

The DATETIMEOFFSET Type: DATETIMEOFFSET (n) is the time-zone-aware version of a datetime datatype. The name will appear less odd when you consider what it really is: a date + a time + a time-zone offset. The offset is based on how far behind or ahead you are from Coordinated Universal Time (UTC) time.

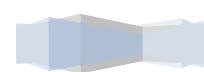
The DATETIME2 Type: It is an extension of the datetime type in earlier versions of SQL Server. This new datatype has a date range covering dates from January 1 of year 1 through December 31 of year 9999. This is a definite improvement over the 1753 lower boundary of the datetime datatype. DATETIME2 not only includes the larger date range, but also has a timestamp and the same fractional precision that TIME type provides

What is RAID and what are different types of RAID configurations?

RAID stands for Redundant Array of Inexpensive Disks, used to provide fault tolerance to database servers. There are six RAID levels 0 through 5 offering different levels of performance, fault tolerance. MSDN has some information about RAID levels and for detailed information, check out the RAID advisory board's homepage

How to restart SQL Server in single user mode? How to start SQL Server in minimal configuration mode?

SQL Server can be started from command line, using the SQLSERVR.EXE. This EXE has some very important parameters with which a DBA should be familiar with. -m is used for starting SQL Server in single user mode and -f is used to start the SQL Server in minimal configuration mode. Check out SQL Server books online for more parameters and their explanations.



What is blocking and how would you troubleshoot it?

Blocking happens when one connection from an application holds a lock and a second connection requires a conflicting lock type. This forces the second connection to wait, blocked on the first. Read up the following topics in SQL Server books online: Understanding and avoiding blocking, Coding efficient transactions. Explain CREATE DATABASE syntax Many of us are used to creating databases from the Enterprise Manager or by just issuing the command: CREATE DATABASE MyDB.

What is a deadlock and what is a live lock? How will you go about resolving deadlocks?

Deadlock is a situation when two processes, each having a lock on one piece of data, attempt to acquire a lock on the other's piece. Each process would wait indefinitely for the other to release the lock, unless one of the user processes is terminated. SQL Server detects deadlocks and terminates one user's process. A livelock is one, where a request for an exclusive lock is repeatedly denied because a series of overlapping shared locks keeps interfering. SQL Server detects the situation after four denials and refuses further shared locks. A livelock also occurs when read transactions monopolize a table or page, forcing a write transaction to wait indefinitely. Check out SET DEADLOCK\_PRIORITY and "Minimizing Deadlocks" in SQL Server books online. Also check out the article Q169960 from Microsoft knowledge base.

What are the steps you will take to improve performance of a poor performing query?

This is a very open ended question and there could be a lot of reasons behind the poor performance of a query. But some general issues that you could talk about would be: No indexes, table scans, missing or out of date statistics, blocking, excess recompilations of stored procedures, procedures and triggers without SET NOCOUNT ON, poorly written query with unnecessarily complicated joins, too much normalization, excess usage of cursors and temporary tables. Some of the tools/ways that help you troubleshooting performance problems are: SET SHOWPLAN\_ALL ON, SET SHOWPLAN\_TEXT ON, SET STATISTICS IO ON, SQL Server Profiler, Windows NT /2000 Performance monitor, Graphical execution plan in Query Analyzer. Download the white paper on performance tuning SQL Server from Microsoft web site. Don't forget to check out [sql-server-performance.com](http://sql-server-performance.com)



As a part of your job, what are the DBCC commands that you commonly use for database maintenance?

DBCC CHECKDB, DBCC CHECKTABLE, DBCC CHECKCATALOG, DBCC CHECKALLOC, DBCC SHOWCONTIG, DBCC SHRINKDATABASE, DBCC SHRINKFILE etc. But there are a whole load of DBCC commands which are very useful for DBAs

What are statistics, under what circumstances they go out of date, how do you update them?

Statistics determine the selectivity of the indexes. If an indexed column has unique values then the selectivity of that index is more, as opposed to an index with non-unique values. Query optimizer uses these indexes in determining whether to choose an index or not while executing a query. Some situations under which you should update statistics: 1) If there is significant change in the key values in the index 2) If a large amount of data in an indexed column has been added, changed, or removed (that is, if the distribution of key values has changed), or the table has been truncated using the TRUNCATE TABLE statement and then repopulated 3) Database is upgraded from a previous version. Look up SQL Server books online for the following commands: UPDATE STATISTICS, STATS\_DATE, DBCC SHOW\_STATISTICS, CREATE STATISTICS, DROP STATISTICS, sp\_autostats, sp\_createstats, sp\_updatestats

What are the different ways of moving data/databases between servers and databases in SQL Server?

There are lots of options available, you have to choose your option depending upon your requirements. Some of the options you have are: BACKUP/RESTORE, detaching and attaching databases, replication, DTS, BCP, logshipping, INSERT...SELECT, SELECT...INTO, creating INSERT scripts to generate data.

How to determine the service pack currently installed on SQL Server?

The global variable @@Version stores the build number of the sqlservr.exe, which is used to determine the service pack installed. To know more about this process visit SQL Server service packs and versions.



What's the maximum size of a row?

8060 bytes. Don't be surprised with questions like 'what is the maximum number of columns per table'. 1024 columns per table. Check out SQL Server books online for the page titled: "Maximum Capacity Specifications". Explain Active/Active and Active/Passive cluster configurations Hopefully you have experience setting up cluster servers. But if you don't, at least be familiar with the way clustering works and the two clustering configurations Active/Active and Active/Passive. SQL Server books online has enough information on this topic and there is a good white paper available on Microsoft site. Explain the architecture of SQL Server This is a very important question and you better be able to answer it if consider yourself a DBA. SQL Server books online is the best place to read about SQL Server architecture. Read up the chapter dedicated to SQL Server Architecture.

What is a Schema in SQL Server 2005? Explain how to create a new Schema in a Database.

A schema is used to create database objects. It can be created using CREATE SCHEMA statement. The objects created can be moved between schemas. Multiple database users can share a single default schema.

CREATE SCHEMA sample;

Table creation

```
Create table sample.sampleinfo
{
id int primary key,
name varchar(20)
}
```

What are Page Splits?

When there is not enough room on a page for a new row, a Server splits the page, allocates a new page, and moves some rows to the new page.

What is SQL Server Agent?

SQL Server Agent is a Microsoft Windows service that executes scheduled administrative tasks called jobs. SQL Server Agent uses SQL Server to store job information. Jobs contain one or more job steps. We generally schedule the backups on the production databases using the SQL server agent. In SQL Server 2005 we have roles created for using SQL Server agents.

- SQLAgentUserRole



- SQLAgentReaderRole
- SQLAgentOperatorRole

SQL Server Agent for SQL Server 2005 provides a more robust security design than earlier versions of SQL Server. This improved design gives system administrators the flexibility they need to manage their Agent service.

### **Problem**

The phone interview. It has tales of bringing normally rationale people to a terrified state. I have even heard of a DBA that was so worried about a SQL Server phone interview that 'they just happened to be in the office park' where the company was located and actually wanted the interview face to face. The reality is, just about all organizations that I work with have a phone interview as a right of first passage in the process. The employer wants to quickly determine if the DBA candidate could be qualified for the position from a technical perspective and if they will fit into the team. As a DBA, what sorts of things should you be on the lookout for during a phone interview? What do you think the employer is expecting? Is this the technical interview or not? Should you try to avoid the phone interview all together and just 'pop-in' for a face to face interview?

### **Solution**

Let's address the last question first, that being should you just 'pop-in' to the office rather than having a phone interview? Phone interviews are setup for a reason. They are intended as a simple means to determine if someone is worth going through the entire interview process. Some organizations have a fairly structured process and follow it closely while other organizations really conduct interviews over lunch or based on a personal contact's network. You need to be the judge and assess the situation for yourself and make the call. In the story I was told about, the results were not positive from either the employer or DBA perspective. So keep that in mind.

With that behind us, let's get into the employer and DBA views of the phone interview as well as some potential questions you should be ready to answer.

### ***Employer's Perspective***

In some respects, employers handle phone interviews in such a manner that they can use the same questions to assess the skills of the candidates as a means to compare and contrast their skills to determine a candidate ranking. By this I mean candidates in terms of best to worst skill as well as who should and should not progress to the next step in the process, the on-site interview.

From an employer's perspective, they are trying to determine a few different items during the phone interview:

- Communication skills
- Personality



- Technical experience and background
- Leadership qualities
- How they could fit into the team

### ***DBA Perspective***

Here are some thoughts from a DBA perspective when it comes to a phone interview:

- First, be ready for the phone interview and expect it as a portion of the interview process.
- Prepare for the phone interview just like you would the on-site interview. Remember if you do not make a good impression with the phone interview that the on-site interview may not be a reality.
- Remember the phone interview is a rite of passage, so first impressions can mean a great deal. Make sure your first impressions are what you want them to be. Simple items like stuttering, stumbling over your words, smoking during the call, chewing gum, etc. may turn off the interviewer quickly.
- Next, figure out your 30 second elevator pitch and make sure you outline your most important experience and skills as well as how you are going to help the organization.
- Just like with your resume, be sure you do not lie. If you do not know the answer to a question, just say you do not know.
- Be prepared for technical questions from either a technical or non-technical interviewer. Be sure to respond in a way that they can understand the response. The interviewer may be looking for just buzz words or may be not. A good technical interview, from a knowledgeable DBA, can really dig into the details to make sure you truly understand the technology.
- As much as the interview process is about the employer selecting the right employee, also keep in mind that the candidate should select the right organization for themselves. As such, use the phone interview as a means to learn about the organization. If you are given the opportunity, ask the questions you have prepared.
- The topic of salary and compensation may be discussed. Be prepared for the question. Historically, the response has been to push off the salary figures to as late in the process as possible. As an employer and employee, I disagree. I think it only makes sense to state a range to make sure one party does not have different expectations than the other. If the figures are not even close, it might make sense for either party to stop the process rather than spending a significant amount of time only to be disappointed at the end. Just something to consider.

### ***Phone Interview Questions***

Although no two phone interviews are the same, below outlines some potential questions to keep in mind as you prepare for a SQL Server DBA phone interview:

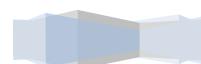
- Can you explain your skill set?
  - Employers look for the following:
    - DBA (Maintenance, Security, Upgrades, Performance Tuning, etc.)



- Database developer (T-SQL, DTS, SSIS, Analysis Services, Reporting Services, Crystal Reports, etc.)
    - Communication skills (oral and written)
  - DBA's
    - This is your 30 second elevator pitch outlining your technical expertise and how you can benefit the organization
- Can you explain the environments you have worked in related to the following items:
  - SQL Server versions
  - SQL Server technologies
    - Relational engine, Reporting Services, Analysis Services, Integration Services
  - Number of SQL Servers
  - Number of instances
  - Number of databases
  - Range of size of databases
  - Number of DBAs
  - Number of Developers
  - Hardware specs (CPU's, memory, 64 bit, SANs)
- What are the tasks that you perform on a daily basis and how have you automated them?
  - For example, daily checks could include:
    - Check for failed processes
    - Research errors
    - Validate disk space is not low
    - Validate none of the databases are offline or corrupt
    - Perform database maintenance as available to do so
  - For example, automation could include:
    - Setup custom scripts to query for particular issues and email the team
    - Write error messages centrally in the application and review that data
    - Setup Operators and Alerts on SQL Server Agent Jobs for automated job notification
- How do you re-architect a process?
  - Review the current process to understand what is occurring
  - Backup the current code for rollback purposes
  - Determine what the business and technical problems are with the process
  - Document the requirements for the new process
  - Research options to address the overall business and technology needs
    - For example, these could include:
      - Views
      - Synonyms
      - Service Broker
      - SSIS
      - Migrate to a new platform
      - Upgrade in place
  - Design and develop a new solution
  - Conduct testing (functional, load, regression, unit, etc.)
  - Run the systems in parallel
  - Sunset the existing system
  - Promote the new system



- What is your experience with third party applications and why would you use them?
  - Experience
    - Backup tools
    - Performance tools
    - Code or data synchronization
    - Disaster recovery\high availability
  - Why
    - Need to improve upon the functionality that SQL Server offers natively
    - Save time, save money, better information or notification
- How do you identify and correct a SQL Server performance issue?
  - Identification - Use native tools like Profiler, Perfmon, system stored procedures, dynamic management views, custom stored procedures or third party tools
  - Analysis - Analyze the data to determine the core problems
  - Testing - Test the various options to ensure they perform better and do not cause worse performance in other portions of the application
  - Knowledge sharing - Share your experience with the team to ensure they understand the problem and solution, so the issue does not occur again
- What are some of the new T-SQL commands with SQL Server 2005 that you have used and what value do they offer?
  - ROW\_NUMBER - Means to page through a result set and only return the needed portion of the result set
  - EXCEPT - The final result set where data exists in the first dataset and not in the second dataset
  - INTERSECT - The final result set where values in both of the tables match
  - PIVOT\UNPIVOT - Expression to flip rows to columns or vice versa
  - Synonyms - Alias to an object (table, stored procedure, function, view) to maintain the original object and refer to the new object as well
  - NOTE - Many more commands do exist, this is an abbreviated list.
- What are the dynamic management views and what value do they offer?
  - The DMV's are a set of system views new to SQL Server 2005 to gain insights into particular portions of the engine
  - Here are some of the DMV's and the associated value:
    - sys.dm\_exec\_query\_stats and sys.dm\_exec\_sql\_text - Buffered code in SQL Server
    - sys.dm\_os\_buffer\_descriptors
    - sys.dm\_tran\_locks - Locking and blocking
    - sys.dm\_os\_wait\_stats - Wait stats
    - sys.dm\_exec\_requests and sys.dm\_exec\_sessions - Percentage complete for a process
- What is the process to upgrade from DTS to SSIS packages?
  - You can follow the steps of the migration wizard but you may need to manually upgrade portions of the package that were not upgraded by the wizard
  - For script related tasks, these should be upgraded to new native components or VB.NET code
- What are some of the features of SQL Server 2008 that you are looking into and why are they of interest?
  - Change Tracking



- Plan Guides
- SQL Data Collector
- Data Auditing
- Data compression
- NOTE - Many more new features do exist, this is an abbreviated list.

Keep in mind that these questions are primarily related to the relational engine, so a BI DBA would have a whole different set of questions. In addition, the more you know about the organization and role should guide you down a path for the types of questions you should be prepared for during the phone interview.

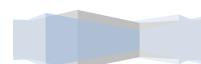
- What is normalization? Explain different levels of normalization?
  - Check out the article Q100139 from Microsoft knowledge base and of course, there's much more information available in the net. It'll be a good idea to get a hold of any RDBMS fundamentals text book, especially the one by C. J. Date. Most of the times, it will be okay if you can explain till third normal form.
- What is denormalization and when would you go for it?
  - As the name indicates, denormalization is the reverse process of normalization. It's the controlled introduction of redundancy in to the database design. It helps improve the query performance as the number of joins could be reduced.
- How do you implement one-to-one, one-to-many and many-to-many relationships while designing tables?
  - One-to-One relationship can be implemented as a single table and rarely as two tables with primary and foreign key relationships. One-to-Many relationships are implemented by splitting the data into two tables with primary key and foreign key relationships. Many-to-Many relationships are implemented using a junction table with the keys from both the tables forming the composite primary key of the junction table. It will be a good idea to read up a database designing fundamentals text book.
- What's the difference between a primary key and a unique key?
  - Both primary key and unique enforce uniqueness of the column on which they are defined. But by default primary key creates a clustered index on the column, where unique creates a nonclustered index by default. Another major difference is that, primary key doesn't allow NULLs, but unique key allows one NULL only.
- What are user defined datatypes and when you should go for them?
  - User defined datatypes let you extend the base SQL Server datatypes by providing a descriptive name, and format to the database. Take for example, in your database, there is a column called Flight\_Num which appears in many tables. In all these tables it should be varchar(8). In this case you could create a user defined datatype called Flight\_num\_type of varchar(8) and use it across all your tables. See sp\_addtype, sp\_droptype in books online.
- What is bit datatype and what's the information that can be stored inside a bit column?



- Bit datatype is used to store boolean information like 1 or 0 (true or false). Until SQL Server 6.5 bit datatype could hold either a 1 or 0 and there was no support for NULL. But from SQL Server 7.0 onwards, bit datatype can represent a third state, which is NULL.
- Define candidate key, alternate key, composite key.
  - A candidate key is one that can identify each row of a table uniquely. Generally a candidate key becomes the primary key of the table. If the table has more than one candidate key, one of them will become the primary key, and the rest are called alternate keys. A key formed by combining at least two or more columns is called composite key.
- What are defaults? Is there a column to which a default can't be bound?
  - A default is a value that will be used by a column, if no value is supplied to that column while inserting data. IDENTITY columns and timestamp columns can't have defaults bound to them. See CREATE DEFAULT in books online.
- What is a transaction and what are ACID properties?
  - A transaction is a logical unit of work in which, all the steps must be performed or none. ACID stands for Atomicity, Consistency, Isolation, Durability. These are the properties of a transaction. For more information and explanation of these properties, see SQL Server books online or any RDBMS fundamentals text book. Explain different isolation levels An isolation level determines the degree of isolation of data between concurrent transactions. The default SQL Server isolation level is Read Committed. Here are the other isolation levels (in the ascending order of isolation): Read Uncommitted, Read Committed, Repeatable Read, Serializable. See SQL Server books online for an explanation of the isolation levels. Be sure to read about SET TRANSACTION ISOLATION LEVEL, which lets you customize the isolation level at the connection level. Read Committed - A transaction operating at the Read Committed level cannot see changes made by other transactions until those transactions are committed. At this level of isolation, dirty reads are not possible but nonrepeatable reads and phantoms are possible. Read Uncommitted - A transaction operating at the Read Uncommitted level can see uncommitted changes made by other transactions. At this level of isolation, dirty reads, nonrepeatable reads, and phantoms are all possible. Repeatable Read - A transaction operating at the Repeatable Read level is guaranteed not to see any changes made by other transactions in values it has already read. At this level of isolation, dirty reads and nonrepeatable reads are not possible but phantoms are possible. Serializable - A transaction operating at the Serializable level guarantees that all concurrent transactions interact only in ways that produce the same effect as if each transaction were entirely executed one after the other. At this isolation level, dirty reads, nonrepeatable reads, and phantoms are not possible.
- CREATE INDEX myIndex ON myTable(myColumn)What type of Index will get created after executing the above statement?
  - Non-clustered index. Important thing to note: By default a clustered index gets created on the primary key, unless specified otherwise.
- What's the maximum size of a row?



- 8060 bytes. Don't be surprised with questions like 'what is the maximum number of columns per table'. 1024 columns per table. Check out SQL Server books online for the page titled: "Maximum Capacity Specifications". Explain Active/Active and Active/Passive cluster configurations. Hopefully you have experience setting up cluster servers. But if you don't, at least be familiar with the way clustering works and the two clustering configurations Active/Active and Active/Passive. SQL Server books online has enough information on this topic and there is a good white paper available on Microsoft site. Explain the architecture of SQL Server. This is a very important question and you better be able to answer it if consider yourself a DBA. SQL Server books online is the best place to read about SQL Server architecture. Read up the chapter dedicated to SQL Server Architecture.
- What is lock escalation?
- Lock escalation is the process of converting a lot of low level locks (like row locks, page locks) into higher level locks (like table locks). Every lock is a memory structure too many locks would mean, more memory being occupied by locks. To prevent this from happening, SQL Server escalates the many fine-grain locks to fewer coarse-grain locks. Lock escalation threshold was definable in SQL Server 6.5, but from SQL Server 7.0 onwards it's dynamically managed by SQL Server.
- What's the difference between DELETE TABLE and TRUNCATE TABLE commands?
  - DELETE TABLE is a logged operation, so the deletion of each row gets logged in the transaction log, which makes it slow. TRUNCATE TABLE also deletes all the rows in a table, but it won't log the deletion of each row, instead it logs the deallocation of the data pages of the table, which makes it faster. Of course, TRUNCATE TABLE can be rolled back. TRUNCATE TABLE is functionally identical to DELETE statement with no WHERE clause: both remove all rows in the table. But TRUNCATE TABLE is faster and uses fewer system and transaction log resources than DELETE. The DELETE statement removes rows one at a time and records an entry in the transaction log for each deleted row. TRUNCATE TABLE removes the data by deallocating the data pages used to store the table's data, and only the page deallocations are recorded in the transaction log. TRUNCATE TABLE removes all rows from a table, but the table structure and its columns, constraints, indexes and so on remain. The counter used by an identity for new rows is reset to the seed for the column. If you want to retain the identity counter, use DELETE instead. If you want to remove table definition and its data, use the DROP TABLE statement. You cannot use TRUNCATE TABLE on a table referenced by a FOREIGN KEY constraint; instead, use DELETE statement without a WHERE clause. Because TRUNCATE TABLE is not logged, it cannot activate a trigger. TRUNCATE TABLE may not be used on tables participating in an indexed view.
- Explain the storage models of OLAP
  - Check out MOLAP, ROLAP and HOLAP in SQL Server books online for more infomation.
- What are the new features introduced in SQL Server 2000 (or the latest release of SQL Server at the time of your interview)? What changed between the previous version of SQL Server and the current version?



- This question is generally asked to see how current is your knowledge. Generally there is a section in the beginning of the books online titled "What's New", which has all such information. Of course, reading just that is not enough, you should have tried those things to better answer the questions. Also check out the section titled "Backward Compatibility" in books online which talks about the changes that have taken place in the new version.
- What are constraints? Explain different types of constraints.
  - Constraints enable the RDBMS enforce the integrity of the database automatically, without needing you to create triggers, rule or defaults. Types of constraints: NOT NULL, CHECK, UNIQUE, PRIMARY KEY, FOREIGN KEY. For an explanation of these constraints see books online for the pages titled: "Constraints" and "CREATE TABLE", "ALTER TABLE"
- What is an index? What are the types of indexes? How many clustered indexes can be created on a table? I create a separate index on each column of a table. What are the advantages and disadvantages of this approach?
  - Indexes in SQL Server are similar to the indexes in books. They help SQL Server retrieve the data quicker. Indexes are of two types. Clustered indexes and non-clustered indexes. When you create a clustered index on a table, all the rows in the table are stored in the order of the clustered index key. So, there can be only one clustered index per table. Non-clustered indexes have their own storage separate from the table data storage. Non-clustered indexes are stored as B-tree structures (so do clustered indexes), with the leaf level nodes having the index key and it's row locator. The row located could be the RID or the Clustered index key, depending up on the absence or presence of clustered index on the table. If you create an index on each column of a table, it improves the query performance, as the query optimizer can choose from all the existing indexes to come up with an efficient execution plan. At the same time, data modification operations (such as INSERT, UPDATE, DELETE) will become slow, as every time data changes in the table, all the indexes need to be updated. Another disadvantage is that, indexes need disk space, the more indexes you have, more disk space is used.
- What is RAID and what are different types of RAID configurations?
  - RAID stands for Redundant Array of Inexpensive Disks, used to provide fault tolerance to database servers. There are six RAID levels 0 through 5 offering different levels of performance, fault tolerance. MSDN has some information about RAID levels and for detailed information, check out the RAID advisory board's homepage
- What are the steps you will take to improve performance of a poor performing query?
  - This is a very open ended question and there could be a lot of reasons behind the poor performance of a query. But some general issues that you could talk about would be: No indexes, table scans, missing or out of date statistics, blocking, excess recompilations of stored procedures, procedures and triggers without SET NOCOUNT ON, poorly written query with unnecessarily complicated joins, too much normalization, excess usage of cursors and temporary tables. Some of the tools/ways that help you troubleshooting performance problems are: SET SHOWPLAN\_ALL ON, SET



- SHOWPLAN\_TEXT ON, SET STATISTICS IO ON, SQL Server Profiler, Windows NT /2000 Performance monitor, Graphical execution plan in Query Analyzer. Download the white paper on performance tuning SQL Server from Microsoft web site. Don't forget to check out [sql-server-performance.com](http://sql-server-performance.com)
- What are the steps you will take, if you are tasked with securing an SQL Server?
    - Again this is another open ended question. Here are some things you could talk about: Preferring NT authentication, using server, database and application roles to control access to the data, securing the physical database files using NTFS permissions, using an unguessable SA password, restricting physical access to the SQL Server, renaming the Administrator account on the SQL Server computer, disabling the Guest account, enabling auditing, using multiprotocol encryption, setting up SSL, setting up firewalls, isolating SQL Server from the web server etc. Read the white paper on SQL Server security from Microsoft website. Also check out My SQL Server security best practices
    - What is a deadlock and what is a live lock? How will you go about resolving deadlocks?
      - Deadlock is a situation when two processes, each having a lock on one piece of data, attempt to acquire a lock on the other's piece. Each process would wait indefinitely for the other to release the lock, unless one of the user processes is terminated. SQL Server detects deadlocks and terminates one user's process. A livelock is one, where a request for an exclusive lock is repeatedly denied because a series of overlapping shared locks keeps interfering. SQL Server detects the situation after four denials and refuses further shared locks. A livelock also occurs when read transactions monopolize a table or page, forcing a write transaction to wait indefinitely. Check out SET DEADLOCK\_PRIORITY and "Minimizing Deadlocks" in SQL Server books online. Also check out the article Q169960 from Microsoft knowledge base.
      - What is blocking and how would you troubleshoot it?
        - Blocking happens when one connection from an application holds a lock and a second connection requires a conflicting lock type. This forces the second connection to wait, blocked on the first. Read up the following topics in SQL Server books online: Understanding and avoiding blocking, Coding efficient transactions. Explain CREATE DATABASE syntax Many of us are used to creating databases from the Enterprise Manager or by just issuing the command: CREATE DATABASE MyDB.
        - But what if you have to create a database with two filegroups, one on drive C and the other on drive D with log on drive E with an initial size of 600 MB and with a growth factor of 15%?
          - That's why being a DBA you should be familiar with the CREATE DATABASE syntax. Check out SQL Server books online for more information.
        - How to restart SQL Server in single user mode? How to start SQL Server in minimal configuration mode?
          - SQL Server can be started from command line, using the SQLSERVR.EXE. This EXE has some very important parameters with which a DBA should be familiar with. -m is used for starting SQL Server in single user mode and -f is used to start the SQL Server in minimal configuration mode.



- Check out SQL Server books online for more parameters and their explanations.
- As a part of your job, what are the DBCC commands that you commonly use for database maintenance?
  - DBCC CHECKDB, DBCC CHECKTABLE, DBCC CHECKCATALOG, DBCC CHECKALLOC, DBCC SHOWCONTIG, DBCC SHRINKDATABASE, DBCC SHRINKFILE etc. But there are a whole load of DBCC commands which are very useful for DBAs. Check out SQL Server books online for more information.
- What are statistics, under what circumstances they go out of date, how do you update them?
  - Statistics determine the selectivity of the indexes. If an indexed column has unique values then the selectivity of that index is more, as opposed to an index with non-unique values. Query optimizer uses these indexes in determining whether to choose an index or not while executing a query. Some situations under which you should update statistics: 1) If there is significant change in the key values in the index 2) If a large amount of data in an indexed column has been added, changed, or removed (that is, if the distribution of key values has changed), or the table has been truncated using the TRUNCATE TABLE statement and then repopulated 3) Database is upgraded from a previous version. Look up SQL Server books online for the following commands: UPDATE STATISTICS, STATS\_DATE, DBCC SHOW\_STATISTICS, CREATE STATISTICS, DROP STATISTICS, sp\_autostats, sp\_createstats, sp\_updatestats
- What are the different ways of moving data/databases between servers and databases in SQL Server?
  - There are lots of options available, you have to choose your option depending upon your requirements. Some of the options you have are: BACKUP/RESTORE, dettaching and attaching databases, replication, DTS, BCP, logshipping, INSERT...SELECT, SELECT...INTO, creating INSERT scripts to generate data.
  - Explain different types of BACKUPs avaialabe in SQL Server? Given a particular scenario, how would you go about choosing a backup plan?
    - Types of backups you can create in SQL Sever 7.0+ are Full database backup, differential database backup, transaction log backup, filegroup backup. Check out the BACKUP and RESTORE commands in SQL Server books online. Be prepared to write the commands in your interview. Books online also has information on detailed backup/restore architecture and when one should go for a particular kind of backup.
  - What is database replication? What are the different types of replication you can set up in SQL Server?
    - Replication is the process of copying/moving data between databases on the same or different servers. SQL Server supports the following types of replication scenarios: Â· Snapshot replication Â· Transactional replication (with immediate updating subscribers, with queued updating subscribers) Â· Merge replication See SQL Server books online for indepth coverage on replication. Be prepared to explain how different replication agents function, what are the main system tables used in replication etc.



- How to determine the service pack currently installed on SQL Server?
  - The global variable @@Version stores the build number of the sqlservr.exe, which is used to determine the service pack installed. To know more about this process visit SQL Server service packs and versions.
- What are cursors? Explain different types of cursors. What are the disadvantages of cursors? How can you avoid cursors?
  - Cursors allow row-by-row processing of the resultsets. Types of cursors: Static, Dynamic, Forward-only, Keyset-driven. See books online for more information. Disadvantages of cursors: Each time you fetch a row from the cursor, it results in a network roundtrip, whereas a normal SELECT query makes only one roundtrip, however large the resultset is. Cursors are also costly because they require more resources and temporary storage (results in more IO operations). Further, there are restrictions on the SELECT statements that can be used with some types of cursors. Most of the times, set based operations can be used instead of cursors. Here is an example: If you have to give a flat hike to your employees using the following criteria: Salary between 30000 and 40000 — 5000 hike Salary between 40000 and 55000 — 7000 hike Salary between 55000 and 65000 — 9000 hike. In this situation many developers tend to use a cursor, determine each employee's salary and update his salary according to the above formula. But the same can be achieved by multiple update statements or can be combined in a single UPDATE statement as shown below:
- ```
UPDATE tbl_emp SET salary = CASE WHEN salary BETWEEN 30000 AND 40000 THEN salary + 5000 WHEN salary BETWEEN 40000 AND 55000 THEN salary + 7000 WHEN salary BETWEEN 55000 AND 65000 THEN salary + 10000 END
```
- Another situation in which developers tend to use cursors: You need to call a stored procedure when a column in a particular row meets certain condition. You don't have to use cursors for this. This can be achieved using WHILE loop, as long as there is a unique key to identify each row. For examples of using WHILE loop for row by row processing, check out the 'My code library' section of my site or search for WHILE. Write down the general syntax for a SELECT statements covering all the options. Here's the basic syntax: (Also checkout SELECT in books online for advanced syntax).
- ```
SELECT select_list [INTO new_table_] FROM table_source [WHERE search_condition] [GROUP BY group_by_expression] [HAVING search_condition] [ORDER BY order_expression [ASC | DESC] ]
```
- What is a join and explain different types of joins.
  - Joins are used in queries to explain how different tables are related. Joins also let you select data from a table depending upon data from another table. Types of joins: INNER JOINS, OUTER JOINS, CROSS JOINS. OUTER JOINS are further classified as LEFT OUTER JOINS, RIGHT OUTER JOINS and FULL OUTER JOINS. For more information see pages from books online titled: "Join Fundamentals" and "Using Joins".
- Can you have a nested transaction?
  - Yes, very much. Check out BEGIN TRAN, COMMIT, ROLLBACK, SAVE TRAN and @@TRANCOUNT



- What is an extended stored procedure? Can you instantiate a COM object by using T-SQL?
  - An extended stored procedure is a function within a DLL (written in a programming language like C, C++ using Open Data Services (ODS) API) that can be called from T-SQL, just the way we call normal stored procedures using the EXEC statement. See books online to learn how to create extended stored procedures and how to add them to SQL Server. Yes, you can instantiate a COM (written in languages like VB, VC++) object from T-SQL by using sp\_OACreate stored procedure. Also see books online for sp\_OAMethod, sp\_OAGetProperty, sp\_OASetProperty, sp\_OADestroy. For an example of creating a COM object in VB and calling it from T-SQL, see 'My code library' section of this site.
- What is the system function to get the current user's user id?
  - USER\_ID(). Also check out other system functions like USER\_NAME(), SYSTEM\_USER, SESSION\_USER, CURRENT\_USER, USER, SUSER\_SID(), HOST\_NAME().
- What are triggers? How many triggers you can have on a table? How to invoke a trigger on demand?
  - Triggers are special kind of stored procedures that get executed automatically when an INSERT, UPDATE or DELETE operation takes place on a table. In SQL Server 6.5 you could define only 3 triggers per table, one for INSERT, one for UPDATE and one for DELETE. From SQL Server 7.0 onwards, this restriction is gone, and you could create multiple triggers per each action. But in 7.0 there's no way to control the order in which the triggers fire. In SQL Server 2000 you could specify which trigger fires first or fires last using sp\_settriggerorder. Triggers can't be invoked on demand. They get triggered only when an associated action (INSERT, UPDATE, DELETE) happens on the table on which they are defined. Triggers are generally used to implement business rules, auditing. Triggers can also be used to extend the referential integrity checks, but wherever possible, use constraints for this purpose, instead of triggers, as constraints are much faster. Till SQL Server 7.0, triggers fire only after the data modification operation happens. So in a way, they are called post triggers. But in SQL Server 2000 you could create pre triggers also. Search SQL Server 2000 books online for INSTEAD OF triggers. Also check out books online for 'inserted table', 'deleted table' and COLUMN\_UPDATED()
- There is a trigger defined for INSERT operations on a table, in an OLTP system. The trigger is written to instantiate a COM object and pass the newly inserted rows to it for some custom processing. What do you think of this implementation? Can this be implemented better?
  - Instantiating COM objects is a time consuming process and since you are doing it from within a trigger, it slows down the data insertion process. Same is the case with sending emails from triggers. This scenario can be better implemented by logging all the necessary data into a separate table, and have a job which periodically checks this table and does the needful.
  - What is a self join? Explain it with an example.
  - Self join is just like any other join, except that two instances of the same table will be joined in the query. Here is an example: Employees table



which contains rows for normal employees as well as managers. So, to find out the managers of all the employees, you need a self join.

- CREATE TABLE emp ( empid int, mgrid int, empname char(10) )
- INSERT emp SELECT 1,2,'Vyas' INSERT emp SELECT 2,3,'Mohan'  
INSERT emp SELECT 3,NULL,'Shobha' INSERT emp SELECT 4,2,'Shridhar'  
INSERT emp SELECT 5,2,'Sourabh'
- SELECT t1.empname [Employee], t2.empname [Manager] FROM emp  
t1, emp t2 WHERE t1.mgrid = t2.empid Here's an advanced query using a  
LEFT OUTER JOIN that even returns the employees without managers (super  
bosses)
- SELECT t1.empname [Employee], COALESCE(t2.empname, 'No  
manager') [Manager] FROM emp t1 LEFT OUTER JOIN emp t2 ON t1.mgrid =  
t2.empid

