

Windows[®] IT Pro

Best Practices for Upgrading to SQL Server[®] 2005

By Haldong "Alex" Ji

Edited with permission from *SQL Server Magazine*.
Copyright © 2008 Penton Media, Inc.
All rights reserved.



Third-party information brought to you courtesy of Dell.
THIS WHITE PAPER IS FOR INFORMATIONAL PURPOSES ONLY, AND MAY CONTAIN TYPOGRAPHICAL ERRORS
AND TECHNICAL INACCURACIES. THE CONTENT IS PROVIDED AS IS, WITHOUT EXPRESS OR IMPLIED WARRANTIES
OF ANY KIND.

Best Practices for Upgrading to SQL Server 2005

→ Contents

Choose a 64-bit System When You Can	1
Moving to SQL Server 2005: Upgrade-in-place or Migration	3
Run Upgrade Advisor before the upgrade	3
Lessons learned when upgrading through migration.....	4
Transfer logins	5
Moving databases from a SQL Server 7/2000 instance to a SQL Server 2005 instance.....	5
Detach/Attach.....	5
Backup and restore.....	5
Copy Database Wizard.....	5
Build the new instance offline	6
Build the new online and rename the instance later	7
Post upgrade tasks	7
Changing the database compatibility level.....	7
Updating Statistics	7
Conclusion	8



SQL Server 2005 came to market two years ago, about five years after the earlier version, SQL Server 2000, was released. For the most part, SQL Server 2000 has been very stable, so many shops still have not moved to SQL Server 2005. But SQL Server 2005 is a compelling upgrade, for reasons not the least of which include CLR integration, database mirroring, and ease of management (DMVs in particular). As SQL Server 2008 looms on the horizon, the time to move to SQL Server 2005 is now.

This *SQL Server Magazine* white paper focuses on moving from SQL Server 2000 or SQL Server 7 to SQL Server 2005. The methodologies discussed here—best practices learned from helping clients make the move—apply equally to both releases. For versions earlier than that, such as SQL Server 6.5, there is no direct upgrade path to SQL Server 2005. You will have to script the objects and data, and move things over manually. Or, you can try to upgrade those instances to SQL Server 7 first, if you can find a copy of SQL Server 7 media. We will cover only methodologies for upgrading the relational engine; Analysis Services, Reporting Services, and Integration Services will not be covered in this white paper.

Choose a 64-bit System When You Can

When preparing to migrate, the first question to answer is which platform should you choose: 32-bit or 64-bit? The answer is simple: get to 64-bit, if you can.

In 32-bit systems, the virtual address space is 4GB (2 to the power of 32). In today's ever increasing database size, that is insufficient. Sure, the physical memory can go well beyond 4GB, and you can turn on AWE within SQL Server 2000 to use that memory. However, that additional memory can be used only in SQL Server's data cache, not the processor cache. In addition, you have additional overhead in managing AWE and

using it for data cache. So a 32-bit system is not an ideal situation.

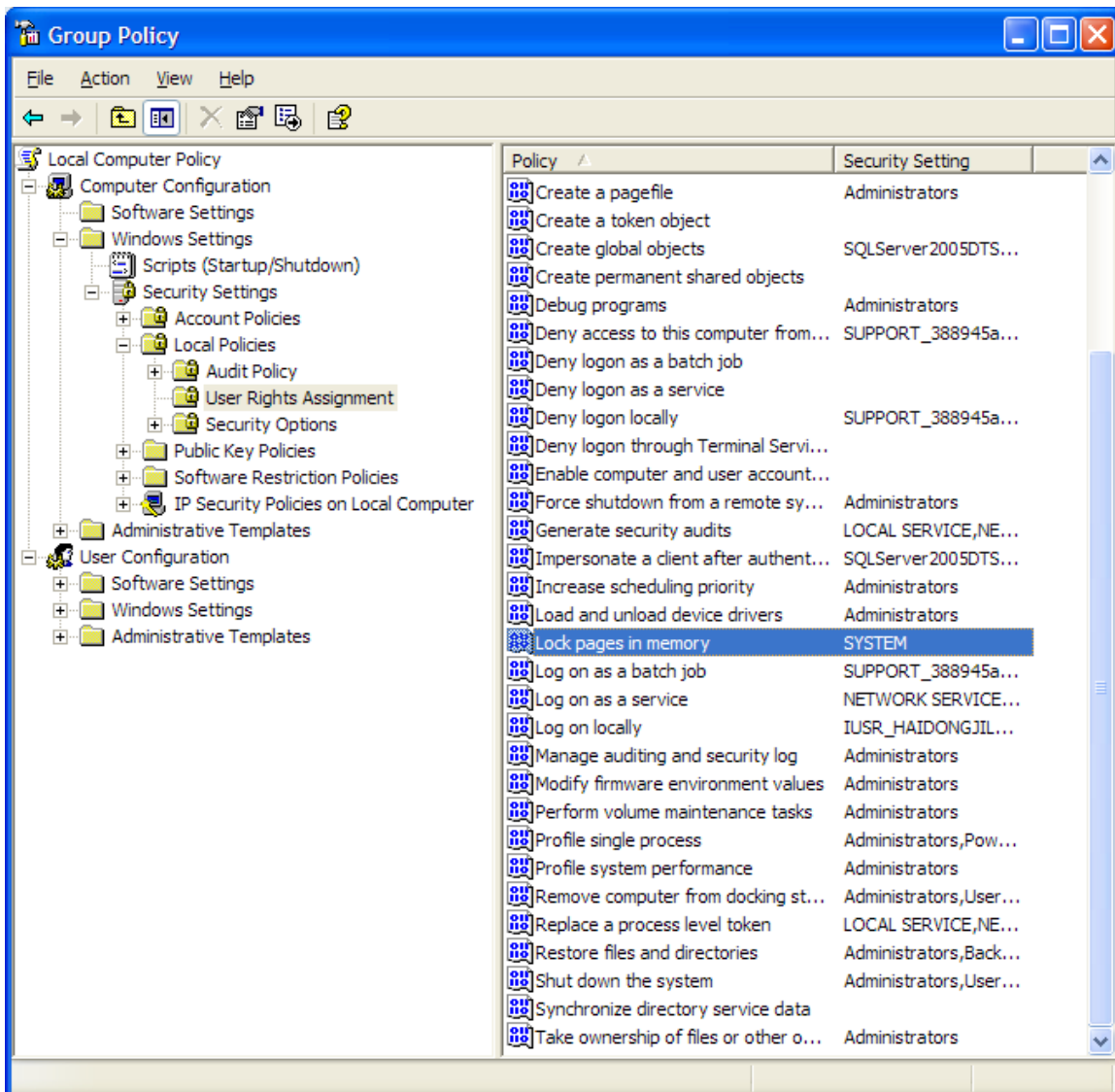
With a 64-bit system, the 4GB limitation is gone and the virtual address space is vastly larger. As such, the Windows® operating system and SQL Server can use the memory more natively and, thus, more efficiently. You will have additional memory for both data and processor cache, without the overhead associated with the 32-bit system.

In addition, the cost for 32-bit and 64-bit software is the same. This holds true for both SQL Server 2005 and Windows operating system software. With the industry

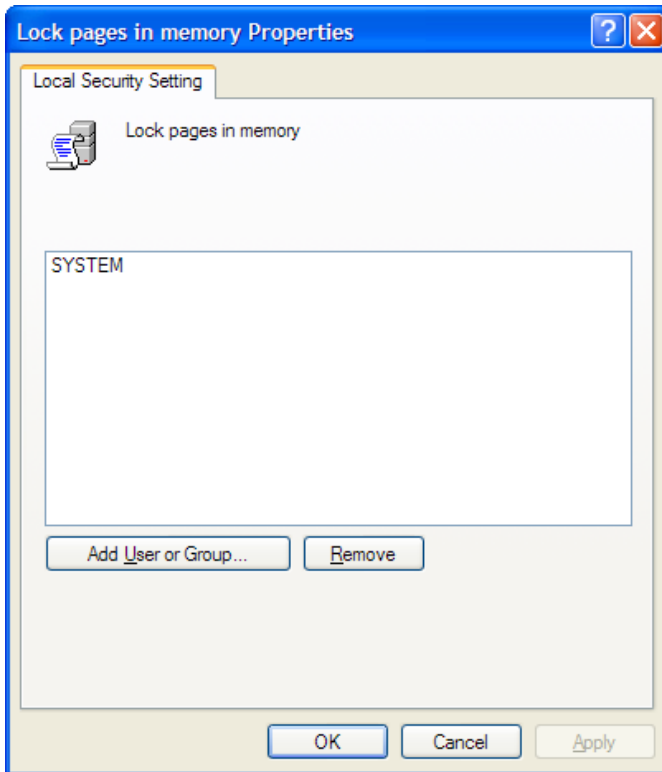
push toward 64-bit systems and the gradual phasing out of 32-bit systems by hardware vendors, you would be hard pressed not to use a 64-bit system.

If you choose a 64-bit system Enterprise Edition, you need to turn on “Lock Pages in Memory” for better memory and paging management. See KB article 918483 for more details (<http://support.microsoft.com/default.aspx/kb/918483>). To turn on “Lock Pages in Memory,” follows these steps:

1. Run gpedit.msc. In the left-hand pane, expand Computer Configuration, expand Windows Settings, expand Security Settings, expand Local Policies, select User Rights Assignment.



- In the right-hand pane, find “Lock pages in memory” and double click, then add your SQL Server startup account in the Local Security Policy Setting tab.



- In SQL Server Management Studio, run:
`sp_configure 'show advanced', 1`
`reconfigure`
`sp_configure 'awe enabled', 1`
`reconfigure`
Ignore the error message below, if you have it.
Msg 5845, Level 16, State 1, Line 1
Address Windowing Extensions (AWE) requires the 'lock pages in memory' privilege, which is not currently present in the access token of the process
- Reboot.

Moving to SQL Server 2005: Upgrade-in-place or Migration

You can use one of two methodologies to move to SQL Server 2005: you can use the existing hardware and upgrade the existing instance (an upgrade-in-place); or you can use a separate server, do a fresh SQL Server 2005 install, and then migrate databases and related jobs and other objects from SQL Server 2000 or SQL Server 7 servers to the new SQL Server 2005 installation. Dell™ PowerEdge™ servers are good candidates for new SQL Server 2005 installs.

During an upgrade-in-place process, the setup program stops the old SQL Server 2000 or SQL Server 7 instance, updates the program with new binary files and registry settings, refreshes the data and metadata so they are consistent with SQL Server 2005 requirements, and restarts the service.

For an upgrade-in-place, you don't need to buy a separate server for SQL Server 2005. The instance name will not change; therefore, it probably is not necessary to change an application's connection string or other database configuration related settings. Furthermore, all database related objects, such as jobs and related system stored procedures, will be in place.

An upgrade-in-place does have disadvantages, however. For example, you will have less control over the upgrade process. You won't be able to perform some regression testing. It will be hard to do a side-by-side comparison of different SQL Server environments, and it will be difficult to roll back to the previous SQL Server environment (although I haven't seen the need for that in my experience). In addition, you won't be running on new hardware that SQL Server 2005 can take better advantage of. By upgrading through migration, you will start from a clean slate. You still have your SQL Server 2000 environment available for a side-to-side comparison. And if you did want to roll back to the SQL Server 2000 environment, you could certainly do that.

The disadvantage of an upgrade through migration is that you will need to move the production databases from the SQL Server 2000 instance to the new SQL Server 2005 instance. The move should also include logins, users, system jobs, and other related objects such as system-stored procedures. In addition, the instance name will have to change (there are ways to get around the name change, which I will discuss later). As such, you will have to change application connection settings.

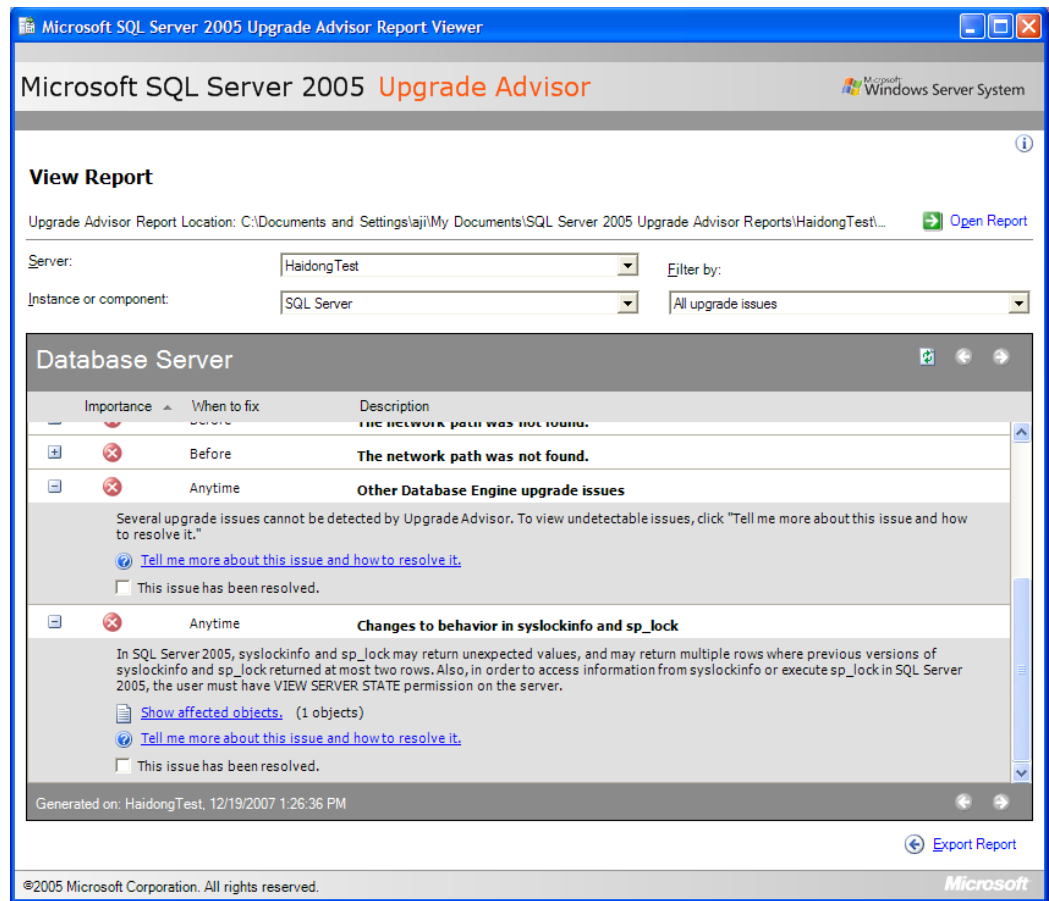
Run Upgrade Advisor before the upgrade

Regardless of what upgrade method you choose, it is always a good idea to inspect the current system for any potential problems. Toward that end, Microsoft provides a free tool, called Upgrade Advisor, to help you identify any possible problems before starting the upgrade process.

SQL Server 2005 Upgrade Advisor scans the existing SQL Server 7 and SQL Server 2000 instances provided. In doing so, the Advisor extracts and analyzes the target instances and verifies them against predefined rules. The

Advisor checks for any potential compatibility issues, deprecated functionalities, and feature and behavior changes, and provides you with a recommended list of tasks that need to be done before and after the upgrade. Below is a report sample generated by the Upgrade Advisor.

These tasks may include coding changes in stored procedures and metadata changes such as indexes and schemas. In many cases, not too many changes will be needed. Some changes can be done by DBAs; other changes need to be done by developers. Because more than one person could be involved in making any changes, you should run the Upgrade Advisor as early as possible in the migration process.



As of this writing, the latest edition of the Upgrade Advisor was released in April 2006. You can get Upgrade Advisor in one of two ways: from Microsoft's Web site or from the SQL Server 2005 installation media. The best approach is to download the Advisor directly from Microsoft's Web site: <http://www.microsoft.com/downloads/details.aspx?familyid=1470e86b-7e05-4322-a677-95ab44f12d75&displaylang=en>

Alternatively, you can find the Upgrade Advisor in a sub-directory called **redist** on the SQL Server 2005 installation media.

Lessons learned when upgrading through migration

As mentioned earlier, you can move to SQL Server 2005 by upgrading in place or upgrading through migration. Because an upgrade-in-place is fairly straightforward, and because an upgrade through migration seems to be the most prevalent upgrade method of choice for many companies, I will focus the rest of the paper on this method.

Do an inventory of the current production system. Before we begin, it is important to understand what exactly needs to be migrated. Databases come to mind first. But other jobs and processes, both inside and outside the database server, also need to be migrated. Here is a list of things to think about.

1. Most database servers have jobs scheduled. These jobs can be maintenance jobs, business process jobs, data transfer jobs, and so on. In most cases, they are an integral part of your company's application, and as such, will need to be moved to the new SQL Server 2005 system. Fortunately, it is easy to script out all SQL Server jobs. You can use the scripts to recreate those jobs on the new system without much trouble.
2. Related to the jobs mentioned above, you need to answer the following questions: If you have a task that transfers data to other RDBMS systems, mainframes, AS400s, do you need to install data drivers on the new server? If yes, do you have the correct setup package? Do you need a 64-bit version of those setup packages? Are they easily available? If not, how do you get them?
3. Other than jobs, do you have system stored procedures

- you would like to move to the new server? If yes, where are they? Can you script them out?
4. Outside the existing production SQL Server instance, but still on the production server, do you have processes that need to be moved—for instance, batch files, SQL scripts, VB or WMI scripts, folders and directory structures?
 5. Outside the server, what other processes, other than production applications, touch the existing SQL Server instance? Will the move affect them in any way? If yes, what needs to be done?
 6. Do applications connect to the database server by name or by IP address? Will the new server have a new IP address?
 7. Do you use SQL Mail on the existing server? If yes, what is the backend mail server? Is it necessary to coordinate with the network administrator to have a proper mail account set up for the new instance?
 8. Does the existing SQL Server use SAN storage? How much space does it take? Most likely, you will need more storage space on the new server. Is the new storage device ready?
 9. Do you have monitoring software on the existing instance? If yes, do you plan to monitor the new instance the same way when it comes on line? Will the monitoring software work with SQL Server 2005?

Transfer logins

Because the upgraded SQL Server will be on a new server, you will need to move all user logins from the old server to the new one. As with mismatched SID issues, when you move a database around on SQL Server 2000 and SQL Server 7 instances you will have the same problem when you move logins from SQL Server 7 or SQL Server 2000 to SQL Server 2005.

If the login is based on Windows authentication, the move will be relatively easy because the authentication is done at the Windows domain level and there will be no misaligned SID issue. The problem happens when the login is based on SQL Server authentication.

SIDs generated on each server are unique to that instance. To preserve that scenario, Microsoft created a stored procedure to help. The advantage is that the script generated by the procedure also preserves the old password. It is not uncommon for a DBA to not know the password for a specific login. You'll find information about this procedure in the Knowledge Base article "How to transfer logins and passwords between instances of SQL Server" at <http://support.microsoft.com/kb/246133>.

The script generated using the procedure provided by

Microsoft does not set the proper default database. Sometimes it is necessary for the login to have a proper default database. Fortunately, that is easy to do.

Moving databases from a SQL Server 7/2000 instance to a SQL Server 2005 instance

After the new SQL Server instance has been installed and configured, you will need to move the databases from your SQL Server 7 or SQL Server 2000 instance to the new one. You can accomplish this move in a number of ways:

- Detach/Attach
- Backup and restore
- Copy Database Wizard

Detach/Attach

One of the quickest ways to move a database around is to detach it from the source instance, copy the data and log files to the destination instance, and then attach the database to the destination instance. If the log file is big, it is not even necessary to move the log file, because it can be created directly on the destination instance at the location you define.

The only downside of this approach is that the database has to go down for a while, which is acceptable if you're doing the upgrade during a scheduled down time (which is usually the case).

Backup and restore

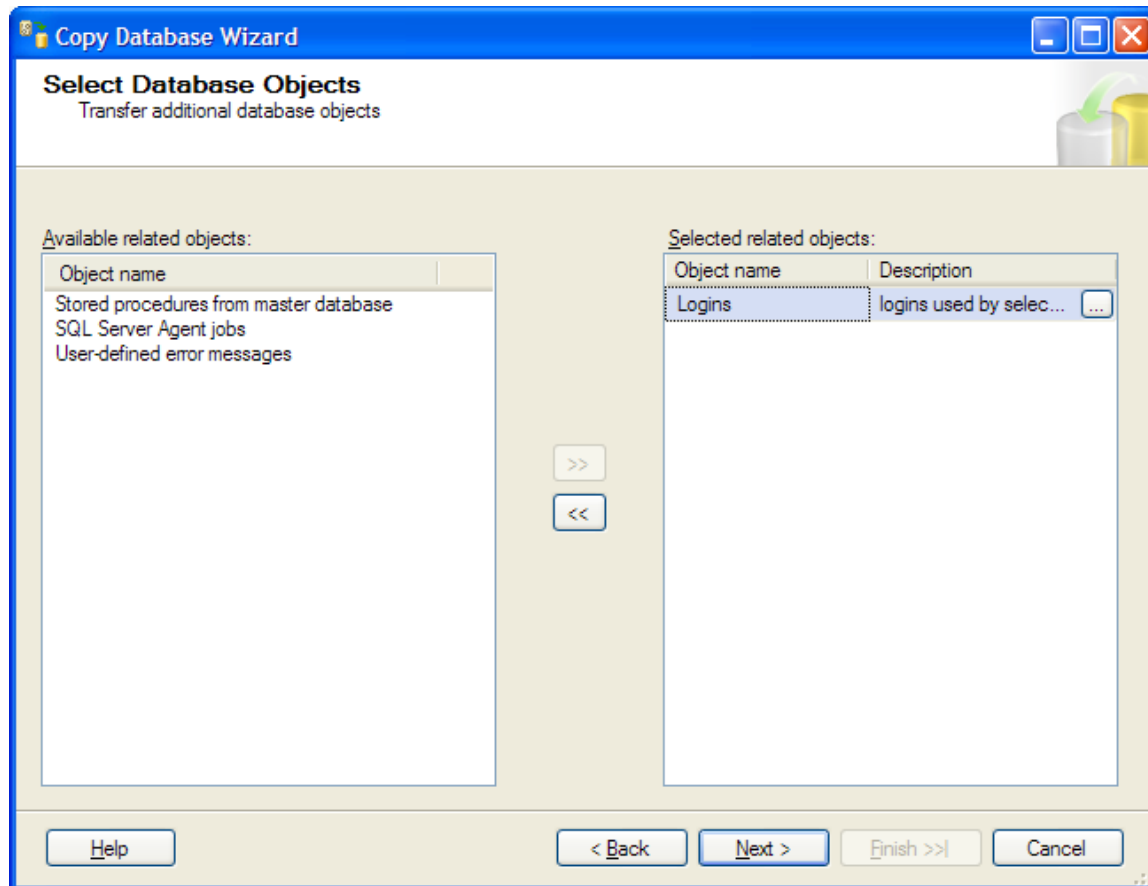
You can back up a SQL Server 7 or SQL Server 2000 database, copy the backup files from the source server to the destination SQL Server 2005 server, and restore it there.

The advantage of this approach is that there will be no down time required for this operation. But this approach is not as simple and easy as detach and attach.

Copy Database Wizard

Moving a database through Copy Database Wizard could be the slowest way of moving databases around. However, it does not require downtime on either the source or destination server. In addition, the wizard will allow you to move jobs and system-stored procedures as part of the process.

As you can see from the screenshot below, you could copy SQL Server Agent jobs, user-defined error messages and stored procedures in the Master database using Copy Database Wizard.



Of the above mentioned ways of moving databases, detach and attach may prove to be the easiest, because you will probably have to schedule a downtime for the upgrade anyway.

Keeping the same instance name after the upgrade
As mentioned earlier, one of the biggest advantages of upgrading in place is that the instance name will not change after the upgrade. This can be very desirable, because changing the instance name means that application connection settings must be modified. In some cases, applications will have to be recompiled. Therefore, more people will need to be involved, which could slow down the upgrade process.

Can you upgrade through migration and still keep the original instance name? The answer is yes. Here are two ways you can keep the original instance name: build the new instance offline or build the new instance online and remain the instance later.

Build the new instance offline

After the hardware is purchased, you do not have to put it online right away. This will allow you to name the

offline machine the same way as the server it is replacing. Because it is offline, there will be no network conflict. While the server is offline, you can install SQL Server 2005 with the same instance name. Because the server is offline doesn't mean you can't work on it. You can create logins and users and make sure they have the same password as production. You can also move databases to the new server. In addition, you can install any other software needed.

It is important to remember, regardless the method used, that you should patch the Operating System and SQL Server 2005 to its latest service packs, unless you have a compelling reason not to do so, for example, if your application is not compatible with the latest release. You wouldn't want to cause additional interruptions to the production system simply because you forgot to patch it properly from the start. When it is time to switch servers, you can take the old system offline and bring the new system online. You may need to do some data refresh, but if you did your homework, things should work.

Build the new online and rename the instance later

If, for whatever reason, the server needs to be built online, obviously you need to give the new server a new name; otherwise, there will be a network conflict. However, when it is time to install the SQL Server 2005 instance, you can give the instance the same name as the current production instance. There will be no conflict because the server names are different. If the instance it replaces is a default instance, then install SQL Server 2005 as the default instance on the server.

After the installation is done, you can move various bits and pieces from the current instance to the new one, such as logins, databases, and other related objects. During cutover, you can rename the old server. When that is done, you can rename the new server to take over the old server's name. After that, you can run some scripts to modify SQL Server to reflect the name change.

For example, suppose the original server's name is ServerA, and it has a default instance on it. After you get the new server for SQL Server 2005, you can name it ServerB. During cutover, you first need to give the original server, ServerA, another name. Let's call it ServerC. You can rename the server by changing the server name at the property page of the computer.

Now that the original server has a different name, you can let the new one take its place. Let's change its name from Server B to ServerA.

To reflect the name change inside the SQL Server instance, you need to run the script below:

1. EXEC sp_dropserver 'ServerB', 'droplogins';
2. EXEC sp_addserver 'ServerA', 'local';
3. Restart the instance.

Once again, when it is time to do the cutover, you can simply ask users to bring up the applications. If you've done your homework right, everything will work properly. Whether you build the server online or offline, you have the option to roll back, if for whatever reason, that becomes necessary. If you build the new instance offline, all you need to do is to bring the new instance offline, and bring the old one online. For the second scenario, you simply let the old server take its original name.

Post upgrade tasks

Suppose the move is over, and your applications are able to connect to the new SQL Server 2005 instance during initial testing. You need to take care of two more things:

changing the database compatibility level and updating statistics.

Changing the database compatibility level

With each new release of SQL Server, Microsoft introduces new features and keywords, and even behavioral or syntax changes to the T-SQL language. To take advantage of these new features, the database needs to be set at the compatibility level of the current release. To ensure that T-SQL behaves the same as the older version of SQL Server, you can set the database's compatibility level to a lower version.

The Upgrade Advisor, if you used it as recommended, should have warned you about compatibility issues, if any. And if you've done your homework, those issues should have been addressed by now. So it should be safe to turn the compatibility to the current level.

Another reason you want to update the compatibility level is this: during any upgrade, it is expected there are minor glitches here and there. As such, the user community is more willing to work with you to resolve any issues that may arise. You might as well take advantage of this opportunity to get things right from the start. Otherwise, if you have to have another service interruption just to update the compatibility level, the user community may not be very receptive to the idea.

So why can't you keep the old compatibility level, you may ask? There are two reasons:

1. Each release has its own compatibility level, and Microsoft usually does not support compatibility levels more than two versions old, so you might as well update sooner rather than later.
2. If the compatibility level is not at 90, many of the new features in SQL Server 2005 will not be available. For example, one very nice feature in SQL Server 2005 is its ability to generate various reports on databases, such as space usage, index data, and expensive queries. If the compatibility level is not set at 90, those reports will not be available to you.

Updating Statistics

Another thing to keep in mind is to refresh statistics. Statistics maintain the distribution of data in your table. In fact, one of the first things the query optimizer looks at, before an execution plan is generated, is the statistics available on columns. Based on different distributions of data, the query engine may choose one execution plan over others.

Because statistics are so important to good SQL Server performance, it is important to keep the statistics up to date. During the upgrade process, some or all statistics need to be refreshed. In fact, I have clients that require an automated statistics update job on their SQL Server 2005 system.

Conclusion

It is not difficult to move from SQL Server 7 or SQL Server 2000 to SQL Server 2005 (the Upgrade Checklist will help). I've worked with many clients that made the move with a simple detach/attach. So don't be intimidated. The key to any successful upgrade is preparation and researching potential problems. Talk to stakeholders, level with them, and get their support, and your upgrade will be a success.

SQL Server 2005 has few changes in the management user interface, called SQL Server Management Studio. And you can use it to manage SQL Server 2000 serv-

ers. I highly recommend that you install SQL Server Management Studio and use it to manage your SQL Server 2000 servers. You'll gain valuable experience with the new tool set and you'll have a better handle on things when all your instances are moved to SQL Server 2005.

Haidong "Alex" Ji is an independent consultant focusing on Microsoft SQL Server, Oracle, MySQL on Solaris and Linux, and database interoperability issues. He has done VB6/COM, ASP and ASP.NET development work and he is an MCSD and MCDBA. Alex is a regular columnist for SqlServerCentral.com and is co-author of three books: *Professional SQL Server 2005 Integration Services*, *Professional SQL Server 2005 Administration*, and *Professional SQL Server 2005 Performance Tuning*. He also has presented at various user group meetings and conferences, including the PASS SQL Server European Conference in Barcelona in February 2006. He can be reached at Haidong.Ji@gmail.com and <http://www.HaidongJi.com/technology>

Upgrade Checklist

Here is a quick check list for you to use during your upgrade:

1. Decide on which version to upgrade to; if possible, pick a 64-bit edition.
2. Run Upgrade Advisor to look for any potential problems, and address those problems with stakeholders as early as possible, because it may take time to fix them.
3. Take an inventory of the current production system. I have provided questions to ask in a previous section of this white paper.
4. Decide on an upgrade method: upgrade-in-place or side-by-side migration.
5. Make sure the latest Windows and SQL Server 2005 Service Packs are installed.
6. When the upgrade is done, make sure database statistics is updated and the compatibility level is set to 90.
7. Pay extra attention to the production system after the upgrade. Address any issues promptly.

Dell and PowerEdge are trademarks of Dell Inc. Microsoft, SQL Server, and Windows are registered trademarks of Microsoft Corporation. Other trademarks and trade names may be used in this document to refer to either the entities claiming the marks and names or their products. Dell disclaims proprietary interest in the marks and names of others.