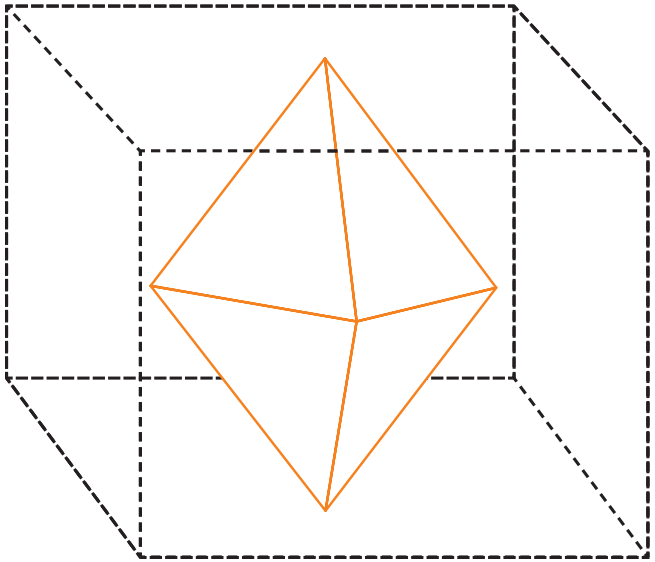
SQL Server INSIDER

Tips for SQL Server pros May 2007

New security model in SQL Server 2005



Organizations traditionally spend little money to secure the database server level. Instead, security is added to the to-do list of DBAs and developers. Get familiar with the security features in SQL Server 2005, and allow them to make your job easier.

INSIDE

- 03 Principals and securables
- 05 New schema model
- 09 Security for CLR
- 14 ARTICLE 1: Database mirroring and its witness
- 17 ARTICLE 2: Find and fix resource-intensive SQL Server queries

New Security Model *in SQL Server 2005*

BY MICHELLE GUTZAIT

Organizations don't tend to invest in securing SQL Server instances and databases. Instead, there is a tendency to leave security considerations to database administrators and developers. The problem is that if there are no defined security standards, it may leave data vulnerable.

Security standards are important for any organization. The problem is that it is not enough just to develop them — they should also be applied, managed and controlled. In many cases, security standards are created after applications and databases are deployed, and that makes them difficult to implement.

Whenever possible, you should plan security standards according to existing and future applications. Security considerations should always be part of the database and application design. For an existing environment, the best approach is to plan and formulate the security modifications step by step.

MICHELLE GUTZAIT

works as a senior database consultant for Itergy International Inc., an IT consulting firm specializing in the design, implementation, security and support of Microsoft products in the enterprise. Gutzait has been involved in IT for 20 years as a developer, business analyst and database consultant. For the last 10 years, she has worked exclusively with SQL Server. Her skills include SQL Server infrastructure design, database design, performance tuning, security, high availability, VLDBs, replication and T-SQL/packages coding.

PRINCIPALS AND SECURABLES IN SQL SERVER 2005

The new security model in SQL 2005 defines two main security objects — principals and securables:

PRINCIPALS are entities that can request SQL Server resources. They can be arranged in a hierarchy. A principal inherits the permissions given to it in a higher level of that hierarchy. Every principal has a security identifier, or SID.

THERE ARE THREE TYPES OF PRINCIPALS:

WINDOWS-LEVEL PRINCIPALS	SQL SERVER-LEVEL PRINCIPAL	DATABASE-LEVEL PRINCIPALS
• Windows domain login • Windows local login	• SQL Server login	• Database user • Database role • Application role

SECURABLES are the resources to which the SQL Server database engine authorization system regulates access. Some securables can be contained within others, creating nested hierarchies called scopes, which can, themselves, be secured.

Access to securables can be

THE SECURABLE SCOPES:

SERVER LEVEL	DATABASE LEVEL	SCHEMA LEVEL

• Endpoint
• Database
	• Database user
	• Database role
	• Application role
	• Assembly
	• Message type
	• Route
	• Service
	• Remote service binding
	• Fulltext catalog
	• Certificate
	• Asymmetric key
	• Symmetric key
	• Contract
	• Schema
		• Type
		• XML schema collection
		• Object
		  • Function
		  • Procedure
		  • Queue
		  • Synonym
		  • Table
		  • View

granted for principals in each level.

For example, a database user or a database role, which can contain zero or more users, can get access to a database object or to a database schema, which can contain zero or more objects. Also, a database user or role can get permissions to view definitions and to grant permissions to another database user or role.

As in previous versions, a user who wants to access data from a database must pass through two stages of authentication — one at the SQL Server level (login) and the other at the database level (user).

NEW PERMISSION DELEGATION CAPABILITY

In SQL Server 2005, more permissions can be given both in the instance and in the database level than in previous releases. Also, permissions can now be inherited — for example, permissions given to a schema are inherited by the schema's objects. Here are examples of new permissions:

- CONTROL – functionally equivalent to all permissions granted to the object's owner and inherited by all subentities within its scope. Principals that have CONTROL permission on a securable can grant permission on that securable. For example:

```
-- Grant CONTROL
-- permission on
-- AdventureWorks user
-- Michelle to user Joseph
-- now the user Joseph can
-- grant permissions on the
-- user "Michelle":
USE AdventureWorks;
GRANT CONTROL ON USER::
Michelle TO Joseph;
```

- ALTER ANY – provides the ability to alter properties of an object. Depending on the scope, inheritance can be limited to objects of a specific type. For example, its variation in the form ALTER ANY '*object_type*' grants permissions to modify every instance of '*object_type*' within server or database scope. For example:

```
ALTER ANY DATABASE DDL
TRIGGER
ALTER ANY SCHEMA
ALTER ANY ROLE
```

- IMPERSONATE – permits impersonating another user, without requiring SysAdmin or dbo privileges, as was the case in SQL Server 2000. For example:

```
-- Grants IMPERSONATE
-- permission on user
-- Michelle to
-- AdventureWorks
-- application role
-- Accountants.
-- The role Accountants can
-- now impersonate
-- Michelle:
USE AdventureWorks;
GRANT IMPERSONATE
ON USER::Michelle TO
Accountants;
```

- VIEW DEFINITION – gives read access to an object's metadata via catalog views. For example:

```
-- Grant role "public" to view
-- any object definition
-- in the instance level:
GRANT VIEW ANY
DEFINITION TO public
-- Grant role "public" to
-- view any object definition
-- in the database level:
GRANT VIEW DEFINITION
TO public
-- Grant VIEW DEFINITION
-- permission on
-- AdventureWorks role
-- Accountants together
-- with GRANT OPTION to
-- database user Michelle
-- (now user Michelle can
-- view the definition of the
-- Accountants role and
-- grant it permissions):
USE AdventureWorks;
GRANT VIEW DEFINITION
ON ROLE::Accountants
TO Michelle WITH GRANT
OPTION;
```

THE STRENGTH OF THE NEW SCHEMA MODEL

How many times have you tried to delete a user from your SQL Server 2000 database and couldn't because it was owning database objects? In SQL Server 2005, that problem has been solved by moving the object definition under the "schema" object.

Schema can have an owner, which can be easily reassigned without having to change the ownership of each object. Also, applications will not break if they reference the schema name before the object name when the object's ownership is shifted. It is still possible to grant permission to an object, such as a table. But as a best practice, centralize permissions by schema, not by objects.

Use synonyms when you want to keep an object name under a specific schema, but that object resides in a different schema, different database or a different SQL Server instance.

WHEN TO USE SYNONYMS

A synonym is an alternative name given to a schema-

scoped object. The synonym is created under the schema object but not bound to it. In other words, the deletion of the synonym will not issue an error message if it is used in another object. The synonym is used during runtime, so the object names are not verified during the synonym's creation. You can grant permissions on synonyms. For example:

```
-- First result will be from
-- publishers_1:
DROP SYNONYM publish
CREATE SYNONYM publish
FOR pubs.dbo.publishers_1
SELECT * FROM publish
-- Second result will be from
-- publishers_2:
DROP SYNONYM publish
CREATE SYNONYM publish
FOR pubs.dbo.publishers_2
SELECT * FROM publish
```

Note that the synonym is a database object. If you try to run the above code as a transaction from two different sessions, the second run will wait until the first transaction ends.

It seems that the best choice for using synonyms is when you would like to keep an object name under a specific schema, but the object resides in a different schema or in a different database or different SQL Server instance. Use a synonym instead of using a view if the synonym is for a table. For example:

```
-- Table resides in another
-- schema:
CREATE SYNONYM
Schema1.Authors
FOR Schema2.dbo.Authors
-- Table resides in another
-- server (myserver\SQL2005
-- is a Linked Server):
CREATE SYNONYM dbo.
RemoteAuthors
FOR [myserver\SQL2005].
pubs.dbo.authors
-- Synonym for a function:
CREATE SYNONYM
synGetAuthorName FOR
pubs.dbo.fnGetAuthorName;
SELECT dbo.
synGetAuthorName('171-10-
1178')
```

DDL TRIGGERS

The new DDL Triggers option, among other things, allows DBAs to control security issues, such as automating grant permissions or auditing.

Here is an example of a DDL trigger from the Microsoft Developer Network (MSDN):

```
-- Grant VIEW DEFINITION
-- on each created role or
-- user to public:
CREATE TRIGGER
   GrantViewDefOnPrincipal
ON DATABASE
FOR       CREATE_USER,
          CREATE_ROLE
AS
DECLARE
@event_type      sysname,
@principal_name sysname,
@sql       nvarchar(max);
```

```
SELECT @sql =
  '(/EVENT_INSTANCE/'+
  'EventType) [1]';
SELECT @event_type =
   eventdata().value
        (@sql,'sysname'),
@principal_name =
   eventdata().value
        (@sql,'sysname');
IF (@event_type =
      'CREATE_USER')
     SELECT @sql =
      'GRANT VIEW ' +
      'DEFINITION ON ' +
      'USER :: ' +
      @principal_name +
      ' TO PUBLIC ' ;
ELSE
     SELECT @sql =
      'GRANT VIEW ' +
      'DEFINITION ON ' +
      'ROLE :: ' +
     @principal_name +
     ' TO PUBLIC ' ;
EXEC (@sql)
```

SECURITY CONFIGURATIONS FOR SQL SERVER

There are two extreme methods of applying security:

- THE LAZY MODEL – The less security, the better. This may result in either giving too many or too few permissions, but security management is easier and less complicated.
- SECURE EVERYTHING POSSIBLE. This may complicate the environment, producing more management effort.

Most organizations are using a security model that is between these two extremes.

When designing a security model, try to keep it simple but satisfying. For example:

- Decide that the schema is the most granular unit to which you grant permissions. Don't grant permission to an object, unless it is really necessary.
- Use a small number of database roles to which you grant permissions. Don't grant permissions on a per-user basis.
- Use only stored procedures to access the data and grant permissions to these stored procedures.
- Allow only one ownership to all the objects and schemas. This will simplify granting the permissions and will help avoid permission chains.

DEFINING SERVICE ACCOUNTS

In previous versions, it was easier to add SQL Server and SQL Server Agent services accounts to the sysadmin group than to play with the Group Policy Objects or permissions of that account.

In SQL Server 2005, the SQL Server service account:

- Requires less privilege than in previous versions. It can now be defined as a member of the Users group (non-domain user) or Domain Users group (domain user). During installation, the user is automatically placed in the SQL Server service group and the group is granted exactly the privileges that it needs.
- Should be changed only by using SQL Server Configuration Manager or by using the equivalent functionality in the Windows Management Instrumentation (WMI) APIs. Using Configuration Manager ensures that the new service account is placed in the appropriate Windows group and is thus granted exactly the correct privileges to run the service.
- Can be configured by password expiration policies because changing the password of the service account does not require restarting SQL Server 2005.

The SQL Server Agent service account requires sysadmin privileges in the SQL Server instance it is associated with. However, in SQL Server 2005, SQL Server Agent job steps can be configured to use proxies that encapsulate alternate credentials.

SURFACE AREA CONFIGURATION

SQL Server 2005 installation minimizes the "attack surface" because, by default, optional features are not installed. It's possible to turn off the features in SQL Server Surface Area Configuration or use the system stored procedure sp_configure.

Upgrading from SQL 2000? If so, few features such as the xp_cmdshell Stored Procedure and ad hoc queries through linked servers — OPENROWSET and OPENDATASOURCE — are disabled by default in a newly installed instance of SQL Server 2005.

Database mail is another great new feature in SQL Server 2005. You might go through the complicated process of configuring SQL Mail in SQL 2000, or you might be using an alternative method of running SQL Mail, like SP_SQLSMTPMail or xp_smtp_sendmail. Note that there is a problem using this feature. If there is a bug in the database program or SSIS package, it can easily flood the mail server.

The SQL Server Surface Area Configuration command-line interface, sac.exe, makes it possible to import and export settings. This enables you to standardize the configuration

of a group of SQL Server 2005 instances. For example:

```
sac in server1.out –S
MyServer
```

INTEGRATION AND SECURITY FOR COMMON LANGUAGE RUNTIME

The security model of the Microsoft SQL Server integration with the Microsoft .NET Framework common language runtime (CLR) manages and secures access between different types of CLR and non-CLR objects running within SQL Server.

Because CLR programs can affect the stability and robustness of the SQL Server environment, it is important to follow these best practices:

- Protect the non-SQL Server resources, such as network and operating system resources, with a higher security level.
- Managed code should not gain a higher security level than it needs -- for example, by impersonating the SQL Server Service or SQL Server Agent service account.
- Managed code should access local resources as much as possible.

SQL Server now integrates the user-based security model of SQL Server with the code access-based security model of the CLR.

DATA ENCRYPTION

Data encryption is a great new feature in SQL Server 2005, but you should use it wisely and only if necessary. Remember that using this feature will result in performance and administrative issues. Generally, SQL Server and database permissions can be enough for most applications when they are well planned and applied.

If you do decide to use encryption, remember that encryption and decryption algorithms are comparably heavy to run. First, never index encrypted columns. Also, test the encryption-decryption performance on the expected amount of data and application functionality before you decide to use it.

OTHER SECURITY ENHANCEMENTS IN SQL 2005

Here are descriptions of a few more security enhancements:

- With SQL Server 2005, you can alter the execution context with the EXECUTE AS clause available as part of the definition of stored procedures, functions, queues and triggers. EXECUTE AS can also be used to set the execution context within a SQL batch instead of SETUSER. The execution context choices are:

- **Execute as caller** — the caller of the procedure (no impersonation). This is the only pre-SQL Server 2005 behavior.
- **Execute as owner** — the owner of the procedure.
- **Execute as self** — the creator of the procedure.
- **Execute as 'username'** — a specific user.

✤ Server 2005 offers a much more granular way of associating privileges with procedural code with code signing. By using the ADD SIGNATURE DDL statement, you can sign the procedure with a certificate or asymmetric key. A user can then be created for the certificate or asymmetric key itself and permissions assigned to that user. When the procedure is executed, the code executes with a combination of the caller's permissions and the key/certificate's permissions.

✤ Direct access to system tables is no longer allowed. Instead, they are exposed through catalog views, encompassing both server and database-wide settings.

✤ SQL Server 2005 can manage SQL Server account password and lockout properties (such as password complexity, password expiration and account lockout) with local and domain-based Group Policies. This functionality is available only on Windows 2003 Server systems. Example:

```
CREATE LOGIN Michelle
WITH
PASSWORD =
'Change$NxtLogin' MUST_
CHANGE,
CHECK_EXPIRATION = ON,
CHECK_POLICY = ON
```

✤ Endpoint-based authentication is used to provide secure communication in scenarios where SQL Server 2005, running on Windows Server 2003, functions natively as a Web service, listening and responding to HTTP SOAP requests.

✤ Permissions on DTS packages in SQL 2000 were difficult to manage. SSIS packages are flexible and can run in different ways.

✤ Microsoft Baseline Security Analyzer (MBSA) is a utility that scans for common insecurities in a SQL Server configuration. Run MBSA on a regularly scheduled basis, either locally or across the network.

XP_CMDSHELL HANDY EXAMPLES

xp_cmdshell is a very powerful

Stored Procedure. In SQL Server 2000, it is enabled by default. In SQL Server 2005, it is disabled by default.

With sp_cmdshell you could run the following command:

```
Exec xp_cmdshell 'del /S c:\'
```

which deletes the c: drive and all its subdirectories in the server where the SQL Server instance is running.

If I am doing so as the sa or in the sysadmin role in SQL and the SQL Server Service account is a sysadmin on the computer running the SQL Server, it can be too powerful.

But here are two examples where you could use xp_cmdshell because it is quicker and more straightforward:

Imagine that you have more than 30 servers hosting SQL Server instances.

Your manager asks you to do two things:

1. **Delete** a specific file on the c:\temp directory in each of these servers.
2. **Collect** the list of the installed programs and tools on each of the servers, prior to consolidation.

You could go computer by computer and do these two tasks. You could write code.

Oops, sorry, you are a DBA—but what could be easier than to create a table with the SQL Server instances names, create a cursor on them and then loop and run what's needed to be run?

In Task 1, you'll see code to delete a file, which results in no output.

In Task 2, you'll see code to show the contents of a directory on a server. Here, you'll find the results of the code in Task 2, assuming it ran against only one server.

NOTE: *These two examples are not secured and are against security best practices. They will run on SQL Server 2000 as well as on SQL 2005.*

LINKS FOR SQL SERVER

Security Considerations for Integration Services
http://msdn2.microsoft.com/en-us/library/ms137833.aspx

Security Considerations for SQL Server
http://msdn2.microsoft.com/en-us/library/ms161948.aspx

CLR Integration Security
http://msdn2.microsoft.com/en-us/library/ms131071.aspx

SQL Server 2005 Best Practices Analyzer
http://www.microsoft.com/downloads/details.aspx?FamilyId=DA0531E4-E94C-4991-82FA-F0E3FBD05E63&displaylang=en

Stately IT

The Pennsylvania State University

"Re-indexing used to take four hours—completing outside our maintenance window. Once we went to SQL Server 2005 on the PowerEdge 6850 servers, the whole process could be completed in half an hour—eight times faster."

— **Alex Pollock**, Lead Database Administrator, The Penn State University

Dell Services helped Penn State upgrade the hardware foundation for its learning management system using Dell™ PowerEdge™ 6850 servers and Microsoft SQL Server 2005

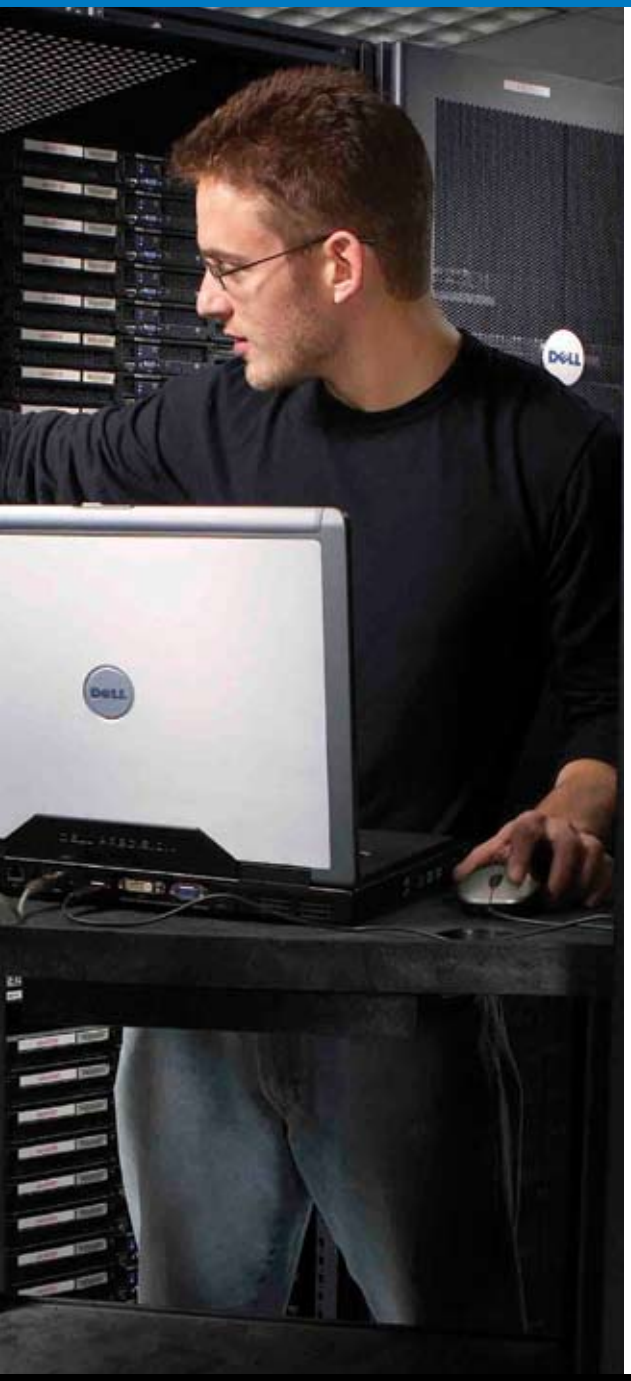
Since its founding in 1855 as a small agricultural college dedicated to applying scientific principles to farming, The Pennsylvania State University—affectionately known as Penn State—has grown into a world-class learning institution with more than 84,000 enrolled students.

Like many universities, Penn State uses learning management system (LMS) technology to help manage academic course content. Unfortunately, until recently the university's system was plagued with performance and scalability issues due to an aging hardware infrastructure.

The university's IT staff turned to trusted longtime technology partner Dell for answers. Working closely with Dell Services, the staff tested the LMS on a new hardware architecture. "We found that Dell hardware offered excellent performance for the LMS and enabled us to handle more users than the previous system," explains Lowell Smith, database administrator at Penn State.

Stately IT

The Pennsylvania State University



With such positive test results in hand, the Penn State IT team felt confident deploying a hardware infrastructure, including Dell™ PowerEdge™ 6850 servers, on which Penn State runs Microsoft SQL Server 2005 Enterprise to provide database support for the LMS. According to Alex Pollock, lead database administrator at Penn State, the combination of Dell PowerEdge servers with SQL Server has resulted in impressive performance gains for the LMS compared to the previous system. "Re-indexing used to take four hours—completing outside our maintenance window," Pollock states. "Once we went to SQL Server 2005 on the PowerEdge 6850 servers, the whole process could be completed in half an hour—eight times faster."

Even though the user load has more than doubled, the performance of the ANGEL system is impressive: application availability is exceptional and hardware utilization is nowhere near capacity. "Our user load has skyrocketed in recent months—now we are experiencing 1.4 million Web hits on the system per hour," notes Peter Dawson, manager of mid-tier infrastructure for Administrative Information Services at Penn State. "But even under that load, our servers are running at 25 to 30 percent capacity, which indicates that we have plenty of room to grow.

To view the entire story, go to **www.dell.com**

SQLServer Insider BACKUP AND RECOVERY

Database mirroring and its witness

BY GREG ROBIDOUX

SQL Server 2005's database mirroring feature offers new functionality that allows you to configure database failover much easier than in the past. When configuring database mirroring, one option is to use the High Availability mode. This option allows for synchronizing of transaction writes on both servers, as well as offers the ability of automated failover. When using the High Availability mode, you need to have three instances of SQL Server: the principal, mirror and the witness. Here is a summary of what each component does.

- PRINCIPAL – this is the instance that stores the active database.
- MIRROR – this is the instance that receives transactions to keep the mirrored database in sync.
- WITNESS – this is the instance that communicates with the principal and mirror to determine if failover should occur.

Greg Robidoux

is the president and founder of Edgewood Solutions LLC, a technology services company delivering professional services and product solutions for Microsoft SQL Server. He has authored numerous articles and has delivered presentations at regional SQL Server users groups and national SQL Server events. Robidoux also serves as the SearchSQLServer.com Backup and Recovery expert.

WHAT IS THE ROLE OF THE WITNESS SERVER?

The witness is a third instance of SQL Server 2005 that acts as an intermediary between the principal and the mirror in order to determine when to fail over. By having a third instance, it creates the ability to have a 2–1 vote that says one of my components is not available and, therefore, I am going to fail over. Because of the need to determine if the components are online or offline before an automatic failover, the witness server is only needed when you implement the High Availability mode and you want, or need, automatic failover. This instance doesn't do much more than communicate with the principal and the mirror to make sure they are still alive. No database activity is occurring on this instance, just communication between the three components.

You can create three instances on one server when setting up database mirroring. But, if you are trying to eliminate downtime caused by hardware failure, install the witness on a different piece of hardware.

WHERE SHOULD THE WITNESS BE?

This really depends on your network configuration and the reliability of your components. If you implement this over a WAN and have periodic network glitches by having the witness near the principal, then you can eliminate some unnecessary failovers. In addition, if your primary data center has some issues and you want to ensure your database stays online, it may make sense to keep this with the mirror. Therefore, if there are any issues in your primary data center, your mirror and witness can communicate and take over the job.

Physical location is not the only point of concern when placing your witness. It would also make sense to install the witness on a different physical server. It is possible to create three instances on one server and set up database mirroring. But, if you are trying to eliminate hardware failure as a possible cause of downtime, the witness should be installed on a different piece of hardware.

If you are trying to eliminate data center outages, it makes sense to have your mirror in a different physical location. Based on this assumption, you should keep the witness and the principal in the same data center and your mirror in a different location.

WHAT VERSION OF SQL SERVER 2005 FOR THE WITNESS?

The witness server can run on any version of SQL Server 2005, including the Express edition. The principal and mirror can only run on the Standard, Enterprise and Developer editions of SQL Server 2005.

WHAT KIND OF SERVER DOES THE WITNESS RUN ON?

You can install the witness on any hardware and operating system that supports the version of SQL Server 2005 you are using for the witness. Because of the nature and role of the witness, I suggest using hardware that you feel is reliable and will not cause further complications when implementing and utilizing database mirroring.

WHAT HAPPENS IF THE WITNESS FAILS?

Because the witness is just one of the three components, if it fails, it does not necessarily mean that a failover will occur. As long as the principal and mirror can still communicate with each other, there is no need for a failover. Therefore, the failure of just the witness will not trigger an automated failover.

HOW DOES FAILOVER WORK?

Since three components make up the High Availability mode, two of these components need to determine that a problem has occurred and then initiate a failover. If the principal server fails and the witness and mirror can still communicate, the failover process will kick in. The mirror will become the principal and the witness will continue to perform its duties as the witness server.

SUMMARY Implementing the High Availability mode of database mirroring is pretty straightforward by just implementing another instance of SQL Server to act as the witness. Although from that perspective it is fairly easy, you should implement database mirroring in different phases until you get the hang of how it works and when it will kick in. Using the High Protection mode as the first implementation probably makes more sense than jumping right into the High Availability mode. Either way, though, take the time to test this new component of SQL Server 2005 before you take the leap into using it as part of your production failover strategy.

Get a step-by-step explanation to setting up database mirroring in a previous article titled Database mirroring setup in SQL Server 2005 found at Search SQL Server.com.

SQLServer Insider PERFORMANCE

Find and fix resource-intensive SQL Server queries

BY JEREMY KADLEC

Taming resource-intensive SQL Server queries is no small task. Finding them can be a challenge and fixing them is typically unique to the query. Here are five common resource-intensive queries with possible resolutions.

HOW TO FIND RESOURCE-INTENSIVE QUERIES

Identifying resource-intensive queries is simple when your application experiences performance issues and users communicate when and where the issues occur. If the overall application is perceived as slow, the root cause and resolution can be much more complex. The following resources will help you address common problems:

- To identify resource-intensive queries, leverage SQL Server 2000 Profiler.

JEREMY KADLEC

is the principal database engineer at Edgewood Solutions, a technology services company delivering professional services and product solutions for Microsoft SQL Server. He has authored numerous articles and delivers frequent presentations regionally and nationally. He authored the "Rational Guide to IT Project Management" and is the SearchSQLServer.com performance and tuning expert.

- To determine how the optimizer processes the code internally, review individual query plans in a graphical format using Query Analyzer.
- To access query plans, use the T-SQL command SET SHOWPLAN_ALL or SET SHOWPLAN_TEXT for a textual view of the output from the SQL Server optimizer.

CALCULATION QUERIES

Users in management and executive management positions issue calculation queries throughout the day. They calculate figures over a long period of time with a primarily static data set (i.e., calculating year-to-date sales or monthly inventory figures). Depending on your applications, the calculations may be different, although the premise remains the same. While users have to ask for these figures for business reasons, the queries may cause a significant resource drain.

To balance the need to run resource-intensive queries and retrieve timely data for users, change the process to execute a stored procedure on a predefined basis, which populates a table that stores the aggregate results. Then have users access the aggregated data instead of issuing the resource-intensive query.

TABLE SCANNING

Table scanning is probably the single biggest offender of draining SQL Server resources. The good news is that the problem is usually easy to fix. The best way to diagnose this resource drainer is to review the query plan. The SQL Server optimizer will indicate which portion of the query is scanning tables by table and column name. With this information, you can create the necessary index to support the query and avoid costly table scanning.

LARGE RESULT SETS

Querying for hundreds or thousands of rows while only displaying 10 to 50 rows in the application is certainly a drain on SQL Server, especially when the query is frequently issued by the same user. Since the data isn't going to change, the application's throughput would benefit from caching that data on the Web server using ADO.NET. Another option would be to cache the IDs or the unique identifier for the result set and query for the detailed data as the data is browsed. A final option I have seen work well is to issue the query with a COUNT clause and let users know how much data will be returned. If it is a significant amount of data, fine-tune the query param-

eters to reduce the result set; not many users will be able to review a large result set, which just becomes overwhelming.

CURSORS

Cursors are notorious for quickly turning a high-end server into a single-user machine. Cursors typically build a large data set and process data one row at a time, which often serializes the processing. Originally developed for ISAM and VSAM databases, Microsoft included support for this processing from the earliest versions of SQL Server. Although they are a viable way to perform data processing, they are not efficient — and your goal should be to migrate away from cursors and use set-based logic.

SINGLE QUERIES THAT RUN REPEATEDLY

Some of the most deviant sets of queries are single queries that execute one or more times per second using few resources — but the number of aggregate resources used is staggering. They don't only eat up SQL Server resources, but also an excessive amount of network round trips. You can expect to see this happen in Web-based applications. If you store data in a session variable or cookie, the problem is resolved.

SUMMARY Take a step back and think about how your applications interact with SQL Server from a functional perspective. Think about complaints users have had historically about the application as well as long-running processes. Observe how users work with the applications and make performance improvements in the code based on how users have evolved with the application.

SQL Server INSIDER

is brought to you by SearchSQLServer.com. The stories "Database mirroring and its witness" and "Find and fix resource-intensive SQL Server queries" originally appeared on SearchSQLServer.com.

EDITORS
Christine Casatelli
Heidi Sweeney

COPY EDITOR
Martha Moore

DESIGN DIRECTOR
Ronn Campisi
www.ronncampisi.com

Additional Resources from Dell

- **Embracing a new level of user experience:**
  Dell Services helped Penn State upgrade the hardware foundation for its learning management system
  http://www.dell.com/content/topics/global.aspx/casestudies/en/2007_penn?c=us&cs=555&l=en&s=biz

- **SQL Server 2005: Preparing for a Smooth Upgrade**
  http://www.dell.com/downloads/global/power/ps1q06-20060126-Microsoft.pdf

- **Maximizing SQL Server Performance**
  http://www.dell.com/downloads/global/power/ps4q05-20050272-Symantec.pdf

- **The Scalable Enterprise Technology Center**
  http://www.dell.com/content/topics/global.aspx/power/en/setc?c=us&cs=555&l=en&s=biz

- **Microsoft SQL Server 2005 Virtualization**
  http://www.dell.com/downloads/global/power/ps4q06-20060405-Muirhead.pdf

- **The Definitive Guide to Scaling Out SQL Server 2005**
  http://www.dell.com/content/topics/global.aspx/alliances/en/ebook_landing?c=us&cs=555&l=en&s=biz