
SQL Server Audit Framework

Version 2.4

Prepared by Ashish Awasthi

Outlined below is a summary of Infosphere's approach to SQL Server audits. Larger jobs require all these elements to be investigated in detail, while smaller jobs can be expedited. This is a matter of judgement and negotiation with the client.

The following headings will be used in this framework:

SQL Server Security

Reviews the development, implementation and management of security controls.

SQL Server Performance

Observes the health of a SQL Server service and identifies any obvious performance problems with SQL Server.

SQL Server Disaster Recovery Process (Backup/Restore)

Deals with on-site and off-site data storage for the purposes of data restoration and historical archiving, and ensures the physical security of backups and archives.

Scheduled Job and Alert Mechanism Setup

Assesses batch processing tasks in terms of resource use and business impacts. Checks alert mechanisms used for encountering failures.

SQL Server Security Audit

- **Access methods** - Determine the threat level to the server based on user access methods.
- **User accounts and logins** - Review SQL Server security mode. Identify accounts with excess privileges. Identify inappropriate uses of the system administrator account.
- **Service account review** - Evaluate the service accounts used to run SQL Server and the SQL Server agent.
- **Access by applications** - Review each application's needs to determine the minimum required permissions.
- Install the most recent service pack & security patches.
- Assess server's security with Microsoft Baseline Security Analyzer (MBSA).
- Use Windows Authentication Mode for security if feasible in your organization.
- Assign a strong SA password.

- Limit privilege level of SQL Server services.
- Disable SQL Server ports on your firewall.
- Remove the Guest user from databases to keep unauthorized users out. The exception to this is the master and tempdb databases as the guest account is required.
- Set alerts to log failed object access and logins

SQL Server Performance Audit

The goal of this audit is to help us, in a quasi-scientific way, identify any obvious performance problems with your SQL Server. One of the advantages of using this checklist is that it will not only tell you what we can do to boost current performance, it also can be used to help you know what you have already done correctly.

To make the SQL Server Performance Audit easy to carry out, we have divided it into the following sections:

- Using Performance Monitor to Identify SQL Server Hardware Bottlenecks
- Server Hardware Performance Checklist
- SQL Server Configuration Performance Checklist
- Database Configuration Settings Performance Checklist
- Index Performance Checklist (optional)
- Application and Transact-SQL Performance Checklist (optional)

Using Performance Monitor to Identify SQL Server Hardware Bottlenecks

Counter Name	Average	Minimum	Maximum
Memory: Pages/sec			
Memory: Available Bytes			
Physical Disk: % Disk time			
Physical Disk: Avg. Disk Queue Length			
Processor: % Processor Time			
System: Processor Queue Length			
SQL Server Buffer: Buffer Cache Hit Ratio			
SQL Server General: User Connections			

Using Performance Monitor to Help Identify SQL Server Hardware Bottlenecks

The best place to start your SQL Server performance audit is to begin with the Performance Monitor (System Monitor). By monitoring a few key counters over a 24 hour period, we should get a pretty good feel for any major hardware bottlenecks your SQL Server is experiencing.

Once we have captured 24 hours of Performance Monitor data in a log, we can display the recommended counters in the graph mode of the Performance Monitor, and then compare the average, minimum, and maximum values in the table above. By comparing our results with the recommendations, we should be able to quickly identify any potential hardware bottlenecks your SQL Server is experiencing.

SQL Server Hardware Performance Checklist

SQL Server Hardware Characteristics	Describe Here
Number of CPUs	
CPU MHz	
CPU L2 Cache Size	
Physical RAM Amount	
Total Amount of Available Drive Space on Server	
RAID Level of Array Used for SQL Server Databases	
Hardware vs. Software RAID	
Location of Operating System	
Location of tempdb Database	
Location of System Databases	
Location of User Databases	
Location of Log Files	
Is Write Back Cache in Disk Controller On or Off?	
Speed of Disk Drives	
Is this Physical Server Dedicated to SQL Server?	

Auditing SQL Server Hardware

From the previous section, we may have identified some potential hardware bottlenecks that are negatively affecting your SQL Server's performance. In this section, we will take a look at each of the major components of a SQL Server's hardware, and examine what can be done to help maximize the performance of your hardware.

SQL Server Configuration Performance Checklist

SQL Server Configuration Settings	Advanced Setting?	Requires Restart?	Default Value	Current Value
affinity mask	Yes	Yes	0	
awe enabled	Yes	Yes	0	
cost threshold for parallelism	Yes	No	5	
cursor threshold	Yes	No	-1	
fill factor (%)	Yes	Yes	0	
index create memory (KB)	Yes	No	0	
lightweight pooling	Yes	Yes	0	
locks	Yes	Yes	0	
max degree of parallelism	Yes	No	0	
max server memory (MB)	Yes	No	2147483647	
max text repl size (B)	No	No	65536	
max worker threads	Yes	Yes	255	
min memory per query (KB)	Yes	No	1024	
min server memory (MB)	Yes	No	0	
nested triggers	No	No	1	
network packet size (B)	Yes	No	4096	
open objects	Yes	Yes	0	
priority boost	Yes	Yes	0	
query governor cost limit	Yes	No	0	
query wait (s)	Yes	No	-1	
recovery interval (min)	Yes	No	0	
scan for startup procs	Yes	No	0	
set working set size	Yes	Yes	0	
user connections	Yes	Yes	0	

SQL Server Configuration Settings

In this section, we are going to take a look at some of the performance-related SQL Server configuration settings. These are SQL Server-specific settings that can be modified using either Enterprise Manager or `sp_configure`.

We will review the various configuration settings and then compare them to default settings in order to see which ones, if any, have been changed from the defaults. Once we have identified any changed settings, our next goal will be to find out why they were changed. If we can't find out why, or if we do find out why, but the reasoning behind the change is flimsy, then we will want to change the settings back to the default values. Once we have done this, our next step is to review all of the other settings (those that were set to default when you started) and evaluate each one in order to see if there might be a benefit of changing the value from the default value to a more appropriate value.

SQL Server Database Settings Performance Checklist

Database Configuration Settings	Default Value	Current Value
auto_close	off	
auto_create_statistics	on	
auto_update_statistics	on	
auto_shrink	off	
read_only	off	
torn_page_detection	on in 2000 off in 7.0	
compatibility level	80 for 2000 70 for 7.0	
database auto grow	on	
transaction log auto grow	on	

Each Database Needs to Be Audited

As part of our performance audit, We need to examine each database located on your server and examine some basic database settings.

As a part of our database settings audit, we will be taking a look at two different types of settings: database options and database configuration settings. We will only focus on those database settings that are directly related to performance.

SQL Server Database Index Performance Checklist (optional)

Indexing Checklist	Response
Have you run the Index Tuning Wizard recently?	
Does every table in each database have a clustered index?	
Are any of the columns in any table indexed more than once?	
Are there any indexes that are not being used in queries?	
Are the indexes too wide?	
Are tables that are JOINed have the appropriate indexes on the JOINed columns?	
Are the indexes unique enough to be useful?	
Are you taking advantage of covering indexes?	
How often are indexes rebuilt?	
What is your index fillfactor?	

Auditing Index Use is not an Easy Task but it is Critical to Your Server's Performance

Indexes can play an important part in SQL Server performance but time constraints may prevent a complete review. We will assess this with you in terms of budget and relevance.

SQL Server Application and Transact-SQL Performance Checklist (optional)

Transact-SQL Checklist	Response
Does the Transact-SQL code return more data than needed?	
Are cursors being used when they don't need to be?	
Are UNION and UNION SELECT properly used?	
Is SELECT DISTINCT being used properly?	
Are temp tables being used when they don't need to be?	
Are hints being properly used in queries?	
Are views unnecessarily being used?	
Are stored procedures being used whenever possible?	
Inside stored procedures, is SET NOCOUNT ON being used?	
Do any of your stored procedures start with sp_?	
Are all stored procedures owned by DBO, and referred to in the form of databaseowner.objectname?	
Are you using constraints or triggers for referential integrity?	
Are transactions being kept as short as possible?	
Application Checklist	
Is the application using stored procedures, strings of Transact-SQL code, or using an object model, like ADO, to communicate with SQL Server?	
What method is the application using to communicate with SQL Server: DB-LIB, DAO, RDO, ADO, .NET?	
Is the application using ODBC or OLE DB to communication with SQL Server?	

Is the application properly opening, reusing, and closing connections?	
Is the Transact-SQL code being sent to SQL Server optimized for SQL Server, or is it generic SQL?	
Does the application return more data from SQL Server than it needs?	
Does the application keep transactions open when the user is modifying data?	

Application and Transact-SQL Code Greatly Affect SQL Server Performance

Of all the areas that can negatively affect the performance of SQL Server, the application code used to access SQL Server data, including Transact-SQL code, has the biggest potential for hurting performance. Unfortunately, it is one of the most time consuming aspects of an audit. Our aim will be to try and get a feel for the relevance of this component before we embark on a detailed review.

We will also try and catch the easy performance-related issues of your application and Transact-SQL code that accesses SQL Server data. Besides the tips listed here, there are a lot more factors that affect SQL Server's performance, but the ones listed here are a good start. This aspect of the audit is confined to in-house developed code.

SQL Server Disaster Recovery Process (Backup/Restore)

Disaster recovery is all-too-often ignored until it's too late. Our first step is to ask few basic questions like

- What would be the result if an hour's worth of database changes were lost?
- A day's worth?
- What about a complete loss of the database?

The answers to these questions form the starting point for the development of a backup and restore plan tailored to the unique needs of your organization.

1) The first step in developing a solid plan is to develop an idea of what constitutes an acceptable loss for your organization.

2) Consider the loss of access to the database itself. What would be the ultimate result if your end users were not able to access information for an extended period of time.

- 3) Make a list of plausible disasters.
- 4) Evaluate the existing disaster recovery plan, if any.
- 5) Create effective operational processes by documenting all your planing decisions, bearing in mind that high system availability involves people and process as much as it involves technology.
- 6) To prevent downtime and data loss implement technologies which satisfy the unique needs of your organization.

Scheduled Job and Alert Mechanism Audit

Alerts should be set up for all known failure states. Each job should generate an alert when a failure is encountered. Failed login attempts should also generate an alert. If there is a known limit to the number of users that can access the database, an alert should be launched when that user count is approached.

- Audit/Create a scheduled job for database backup (the backup method will be identified as part of the disaster recovery plan)
- Audit/Create a scheduled job for system database backup
- Audit/Create a scheduled job for database optimization (includes reorganise data and index pages, remove unused space and update statistics).
- Audit/Create a scheduled job for database integrity checking including indexes.
- Audit/Create Operating System level alert to notify administrator if SQL Server is not running. (This alert will not work if the Operating System itself goes down.)
- Audit/Create alert to notify administrator if SQL Server is running out of space.
- Audit/Create alert to notify administrator if SQL Server is getting used excessively.
- Audit/Create alert to notify administrator if any scheduled job fails.