

SQL Server 2000 Operations

As a DBA it is often difficult to accurately measure how well or badly you are doing your job. Daily catastrophic database failures aside, most users may complain when the database is “running slow” but that is about it. Consequently even if you are doing a fantastic job most DBA’s would only be graded a “C” as they are seen as doing what they always should – managing the database and keeping it running.

So how do you convince your boss that you are doing a superb job and which benchmark can you use to measure yourself against?

To help with this Microsoft have developed an operations framework designed to give the typical DBA a set of processes and management tasks that should be completed on a regular basis, based on best practices gained from across the industry. Adherence to these is a useful measure of your capabilities.

What is the Microsoft Operations Framework?

MOF, as it is called, was created by Microsoft and a number of their key partners to provide a set of planning, deployment and maintenance tasks as they relate to enterprise wide IT projects. MOF itself is an iterative cycle, so designed that processes and procedures can be improved overtime and the delivery of IT back to the business gaining a better reputation.

The MOF cycle takes a typical implementation through the phases of changing, operating, supporting and optimising an IT system. Whilst MOF in its entirety may not be suitable for all organisations cherry picking some of the better ideas is bound to help most practical DBA’s or database operational people.

This month we will look at some of the best ideas in the Microsoft Operations Framework for SQL Server 2000. The framework itself is huge, and we can only scratch the surface but hopefully you will pick up some interesting tips.

SQL Server 2000 DBA Best Practices

At the heart of any significant SQL Server installation is the DBA, responsible for providing a fully functional and optimal database installation. There are a number of typical characteristics that define a decent SQL Server installation, including:

- Full documentation
- Standardised setup
- Best use of automated features of SQL Server
- Maximum reliability of the hardware and software
- Maximum availability of the hardware and software
- Database that is as performant and optimised as possible

Documentation is almost always neglected, coming a poor second when there is a new database feature to play with. Just as a DBA would consider building in redundancy to a SQL Server installation, in case of failure, the knowledge

that you have in your head needs to be documented so that another competent DBA can pick up where you left off in case of the proverbial high street omnibus accident.

Documentation will need to cover all of the operational processes that are undertaken – not rewriting the SQL Server books online, rather ensuring that your own corporate processes for adding logins, database recovery, database backup and the plethora of regular activities are written down. The design of the database tables, components and networking infrastructure needs to be documented and a list of key contacts and help desk personnel that can be called upon in an emergency.

Standardising on a certain configuration or setup which is documented should enable you to rebuild servers far quicker as the settings have already been worked out. Useful standards include the naming convention of databases, database objects and the use of pre-agreed logical drive letters.

SQL Server has a bundle of automated features designed to remove the burden of boring and repetitive tasks. The SQL Server Wizards are useful to set these up but the DBA may prefer to use Transact-SQL (T-SQL) scripts. The scripts themselves could be stored in a software configuration management tool such as Microsoft SourceSafe and subsequent version changes stored safely away so that the originals are always safe. If the DBA prefers to use the SQL Server Enterprise Manager then fine, but use the script “save change” button to keep a scripted version of the changes you made. Quite often it is useful to keep the scripts as stored procedures, and then use some automated auditing during stored procedure execution to keep a check on what has been changed during the script run.

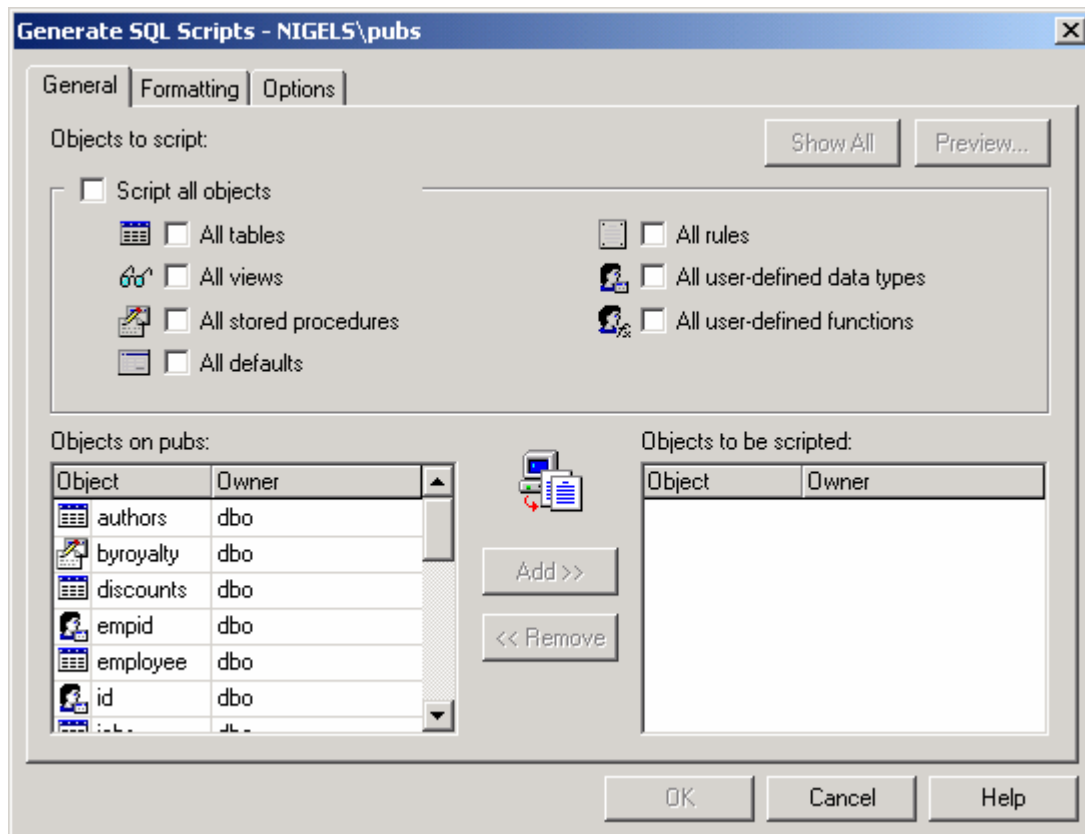


Fig. Generating scripts from SQL Server

Database performance is often the key metric used to measure the success or failure of a database installation. All of the usual performance practices apply, and the DBA will become expert at fine tuning their installation. Regular operations such as the rebuild of indexes should be completed at minimum stress time for the server, if possible, and of course the DBA should use scripting to manage and execute the index management tasks. Remember to manage the statistics for your installation, and in a particularly volatile database environment your statistics may not be as up to date as possible, so a scripted refresh may be in order. If doing this then turn off the auto update stats option.

Availability and reliability are somewhat two of the same. Of course you have a backup in place, but is it as comprehensive and as frequent as it should be? Has a backup been restored on a regular basis to ensure it works? Indeed is the backup too frequent and therefore affecting database performance? A well documented backup and restore plan is crucial, and it must cover applications as well as the operating system and database.

A useful idea is to have a service level agreement or SLA in place with the business. This provides the framework that determines the required availability and reliability of the database and enables you a better degree of protection if your technical recommendations are not acted upon. It is also a useful for when you ask the business about levels of availability and they cite 99.999% uptime as you can document that requirement in the SLA and watch

their faces when you go back with the huge budget that is required to run such an available system.

Monitoring a SQL Server Installation

Once a system has been setup and starts to remain stable the DBA can move it into a monitoring phase – that is, watching a variety of server parameters to determine the state of the server and undertake any preventative work that these parameters may indicate.

The breadth and depth of your system monitoring will depend upon a number of factors, including the complexity of the system, the DBA skills available and more importantly the time you have to monitor a server.

The ongoing monitoring of a SQL Server has been defined by the Microsoft Operations Framework as either proactive or exception management.

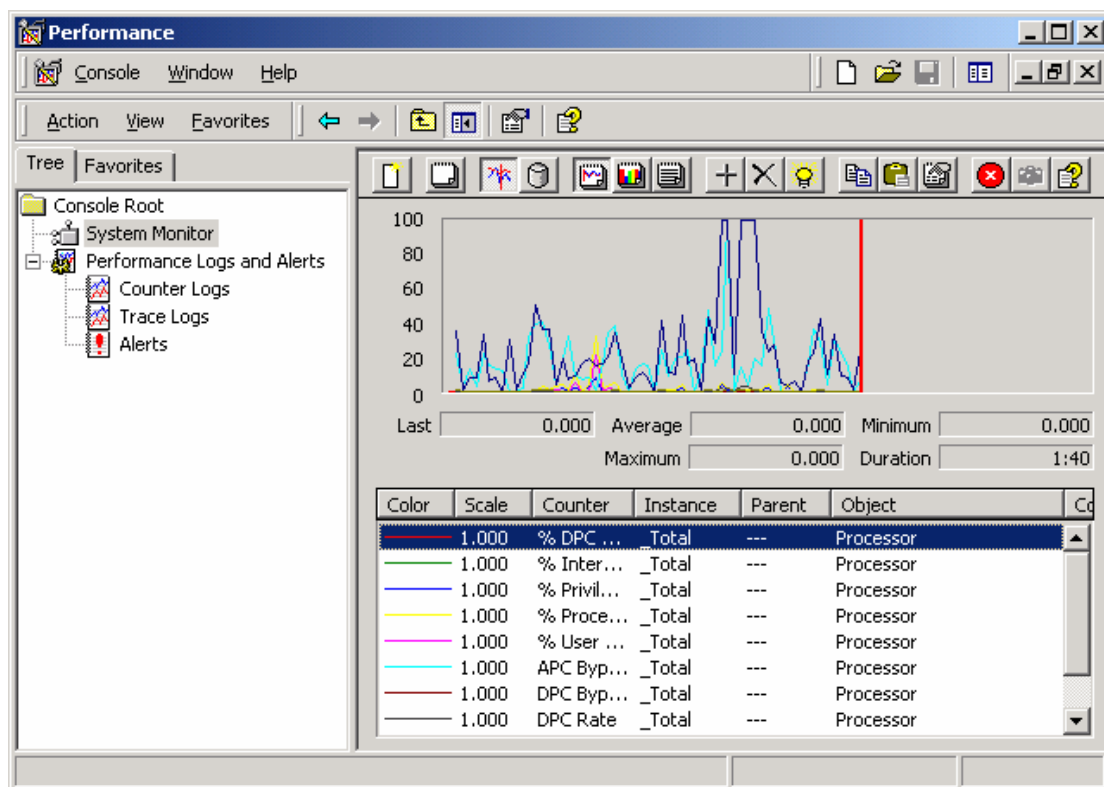


Fig. Monitoring SQL Server performance using a graph

Proactive SQL Server Monitoring

Once your server has settled in, you need to determine an acceptable level of system performance that defines the steady state. This benchmark can then be referred to during the monitoring phase to compare how the system performs.

This benchmark may be created over a period of time that represents a typical operations cycle, including peak loads and troughs in terms of server usage and throughput. The DBA can then draw up a list of problem types in two

categories – those that need intervention and those that are not possible to fix with current resources.

Typical base line counters will include, amongst many other counters:

- Memory – Pages/sec. This is a measure of the pages read from memory as opposed to those read from disk. Pages served from memory are normally faster than those served from disk.
- Network Interface – Bytes total/sec. How busy is the network? If this changes significantly it may be throttling your system
- PhysicalDisk - Disk Transfers/sec. Basic measurement of how busy the disk is at any point
- Processor - % Processor Time. If the processor is running flat out at 100% then user requests are probably in a queue and performance is suffering

The best way of presenting this information is on a baseline chart, created within Enterprise Manager.

Exception Management

This is designed for DBA's in danger of overwork – which I guess is most of them. Only events that are exceptions to normal service running are reported upon, enabling the DBA to only concern themselves about significant problems. Typically a lot of use is made of SQL Server alerts to signify important issues, and only a few SQL Server performance counters are monitored regularly including memory, buffer and CPU measurements.