

# SQL Server DBA Training

Andrew Fraser, July 2000, <http://andrewfraserdba.com>

This course is a reduced version of Microsoft's 5 day System Administration for Microsoft SQL Server 7.0 course. It focuses on the differences between SQL Server and Oracle and on the main SQL Server tasks a DBA team is likely to have to perform, such as: Installs; Service Access Requests; Cloning; Backups; Restores; Datafile maintenance.

DBA tasks which are out of scope for this course are the SQL Server 6.5 to 7.0 **upgrade** process and performance **tuning**.

## Table of Contents

1. SQL Server Overview .....	2
2. Installing and Configuring SQL Server .....	2
3. Security .....	9
4. Managing Database Files.....	11
5. Backup.....	13
6. Restore .....	13
7. Automating Administrative Tasks with Jobs and Alerts.....	16
8. Transferring Data with the DTS .....	17
9. Monitoring Tools .....	17
10. Maintenance Plans.....	18
11. Replication .....	20
Appendix A – Microsoft Support and Other Helpful Resources .....	21
Appendix B – List of Differences between SQL Server and Oracle .....	21
Appendix C – Remote Management .....	23

# 1. SQL Server Overview

SQL Server is the Microsoft licensed version of the Sybase relational database. It is only available on the Microsoft operating systems: Windows 95; Windows 98; NT Workstation; NT Server; and Windows 2000.

This course focuses on SQL Server version 7.0 rather than the later SQL Server 2000 and 2005. SQL Server 2005 in particular has noticeably different front end DBA interfaces to the earlier versions.

The two principal differences between SQL Server/Sybase and Oracle are:

1. SQL Server has no rollback segments and therefore no rollback/commit functionality, or read consistent data views; other than what little can be accommodated from the on line redo logs.
2. SQL Server architecture is: 1 *machine* to 1 *SQL Server* to many *Databases*. There is no such thing as instances and tablespaces within SQL Server. On line redo logs exist at database level.

A more complete list of differences is given in Appendix B.

Note that SQL commands are not executed by ; or / as in oracle, but with **go**.

# 2. Installing and Configuring SQL Server

## Installation

Various types of SQL Server CD exist

1. Microsoft BackOffice SQL Server – the legitimate licensed edition. The Microsoft BackOffice CD suite is normally kept by the NT team. Note that the first CD in the set has to be run to start the install, before the SQL Server specific CD is asked for. A license number has to be supplied, this should be on the BackOffice box label.
2. MSDN (Microsoft Developer Network) SQL Server. This should be used for our PCs and for test and development servers. The binary files in this installation have a hard coded limit of ten logins, so if this is used on a server which will eventually go live, you will have problems. Note that a license number is asked for even with this CD when installing the enterprise edition.
3. SQL Server 120 day evaluation copy – not to be used for obvious reasons.



## 1.1 Installing SQL Server on client (desktop edition)

This is very simple. You can accept all defaults and let the install run through on Windows 95.

On Windows NT Workstation, choose the startup option user to be **Local System** rather than a *Domain User*.

The TCP/IP network library should be installed. This can be configured post install through the *Client Network Utility*.

### Lab 2.1 – Install SQL Server Desktop Edition

## 1.2 Installing SQL Server on Server (enterprise edition)

### Licensing

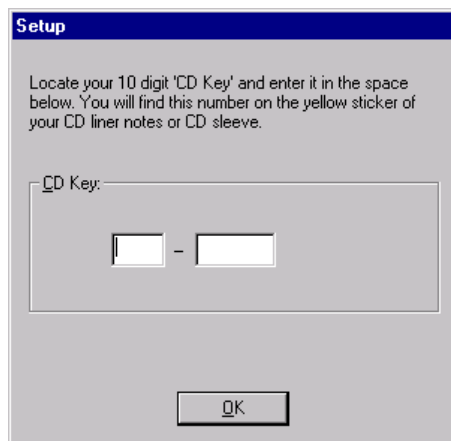
This will ask for a license number, so have this handy before installing. Different projects may use different license numbers – normally this is co-ordinated by the NT team.

Microsoft have two options for licensing:

1. Per Server (individual licenses for individual servers)
2. Per Seat (enterprise wide licenses for clients using any server)

It is cheaper to buy per seat licenses (achieves economies of scale) but that requires inter-project collaboration. So almost all licenses here are per server.

Note that, unlike Oracle, Microsoft actually enforce their licensing within the software. Users can get an error when trying to connect in once all available licenses are in use.



### TCP/IP

Default options can be taken for the rest of the install, except that TCP/IP must be installed. The default port number (1433) should be used.

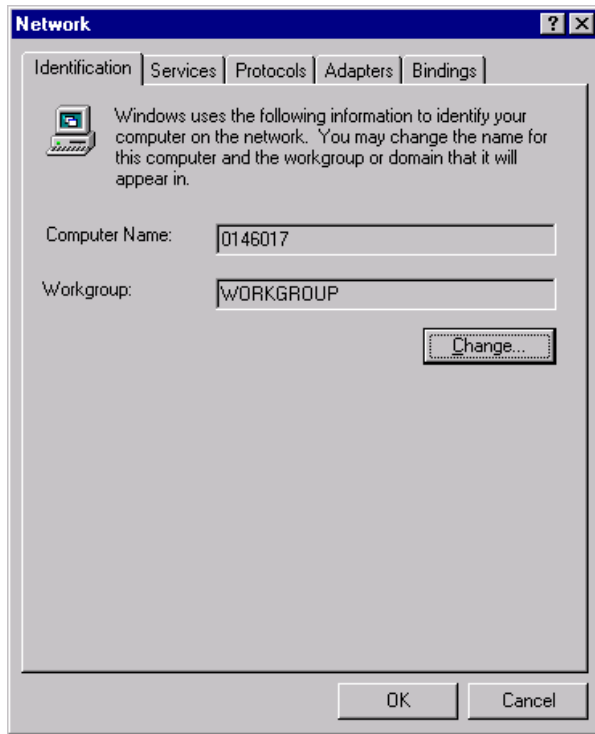
## 1.3 Remote Installs

There are two methods of installing SQL Server onto a remote server. The preferred option is to use NSM to control the remote server. Within this remote server, map your PC's CD-Rom drive as a drive mapping on the remote server. Below are the steps to do this:

### Remote Install Step 1 – Identify Local PC

You need to know the **Computer Name** of your own PC. This can be identified by looking (in your own PC) at **control panel** then **network** and on the **identification** tab. Take a note of the Computer Name as you will need it for step 3.

In the example below, the computer name is 0146017.

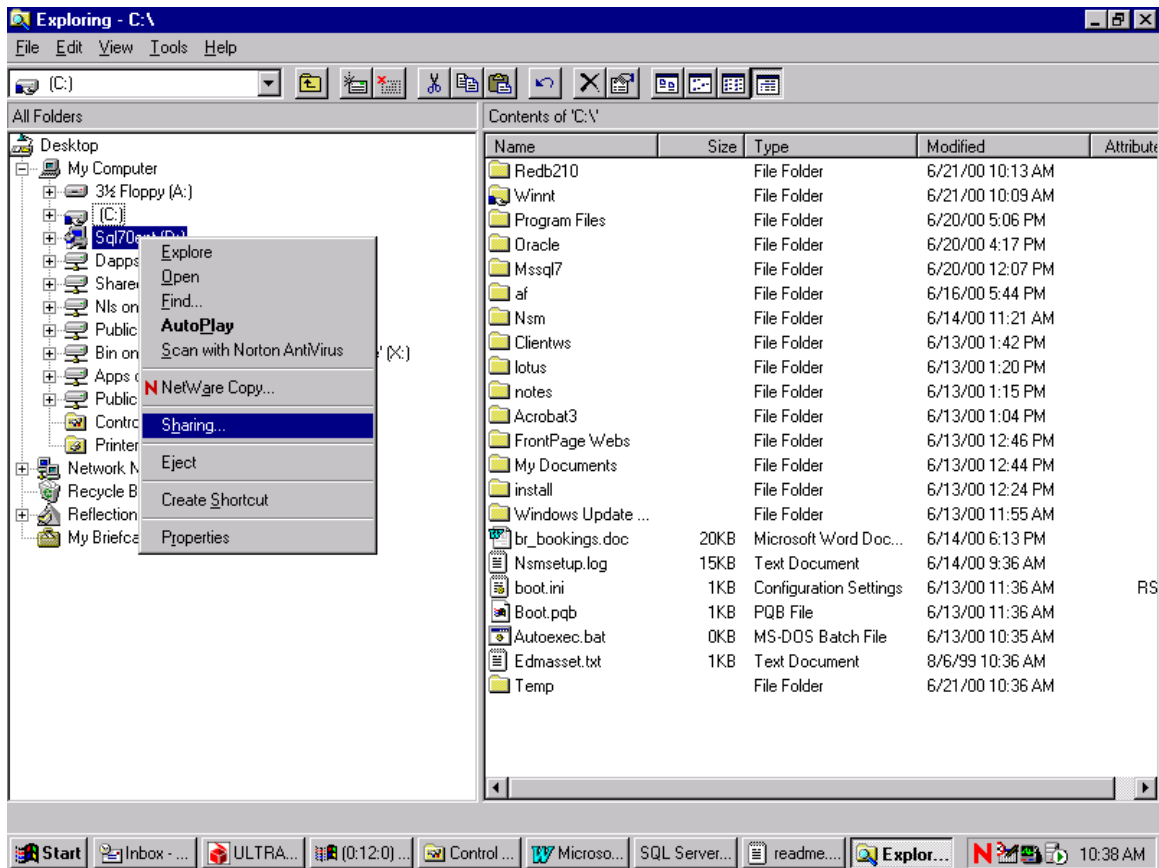


### Remote Install Step 2 - Allow filesharing on local PC

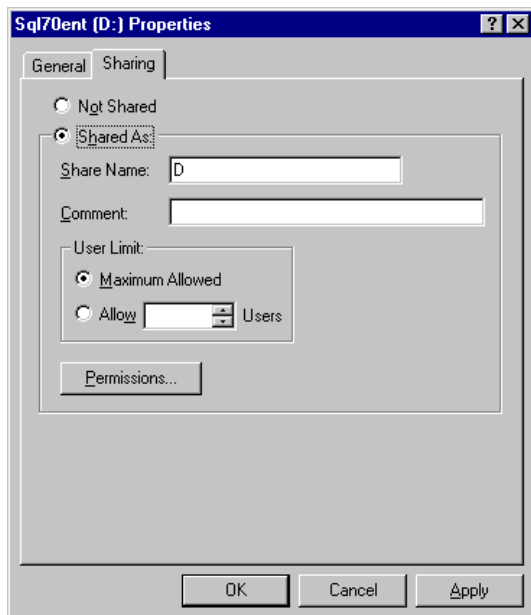
The local PC must be set up to allow filesharing. This is also done through **control panel** then **network**, then on the **Configuration** tab. Click on the **File and Print Sharing** button and tick the "I want to give others access to my files" check box.

### Remote Install Step 3 – Make Local PC CD Rom shareable

You must make the PC CD Rom drive *shareable* (equivalent to the share command in unix). To do this, run **windows explorer** from the **Start** bar. Then **right click** on your CD Rom drive (normally d:) in the right hand window. Select **Sharing...** from the menu.



Make the PC CD Rom drive available to everyone by selecting the **Shared As** radio button. For **Share Name**, you can give the CD Rom drive a meaningful name, or leave it at default of the drive letter (normally D). Either way, take a note of the Share Name, as you will need this for step 3.



### Remote Install Step 4 – Mapping drive on Remote Machine

It is now time to switch into NSM and map the PC CD Rom drive into the remote machine. When logged into the remote machine, bring up **Windows Explorer** and select **Tools** then **Map Network Drive** from the top menu bar.

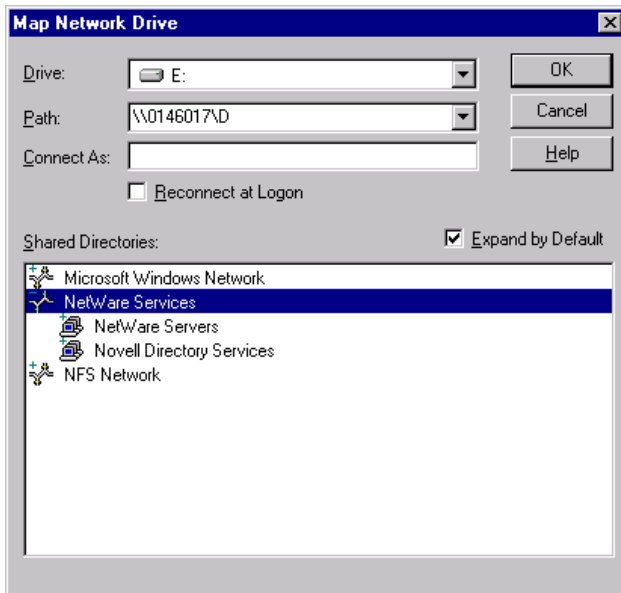
Within the resulting Map Network Drive dialogue box,

**Drive:** select any currently unmapped drive (E: in the example below, but any letter would do)

**Path:** type in here //<Computer Name of local PC from step 1>/<Share Name of local PC CD Rom drive from step 2.

In the example below, the Computer Name is 0146017 and the Share Name is D, so the Path is //0146017/D

**Reconnect at Logon:** Uncheck this box, or the remote machine will be forever attempting to access your local CD Rom drive.



It is theoretically possible to install directly from the path without mapping it first, but that would be a riskier method of installation.

### Remote Install Step 5 – Running the install on Remote Machine

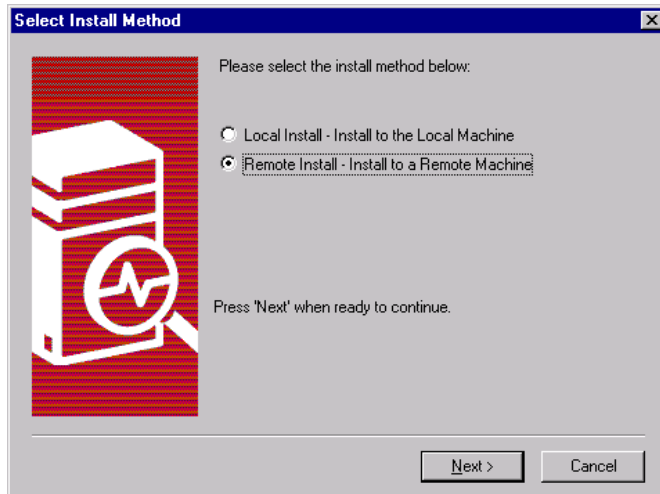
Staying in NSM on the remote machine, run the **autorun.exe** on the SQL Server CD. This can be done from Windows Explorer, or a normal window of the newly mapped drive, or from **Run** in the **Start** bar.

In the example given above, the command to run would be **e:autorun**

After that, the install should proceed just as if it was a standard local installation onto the server.

### Alternative Remote Install Method

Sql Server allows the install program to run on the local PC, but write to a remote server. To use this method, run the install CD on local PC as normal, but select the **Remote Install** radio button when asked for the Install Method.



This method is out of scope for this course and is not explored further.

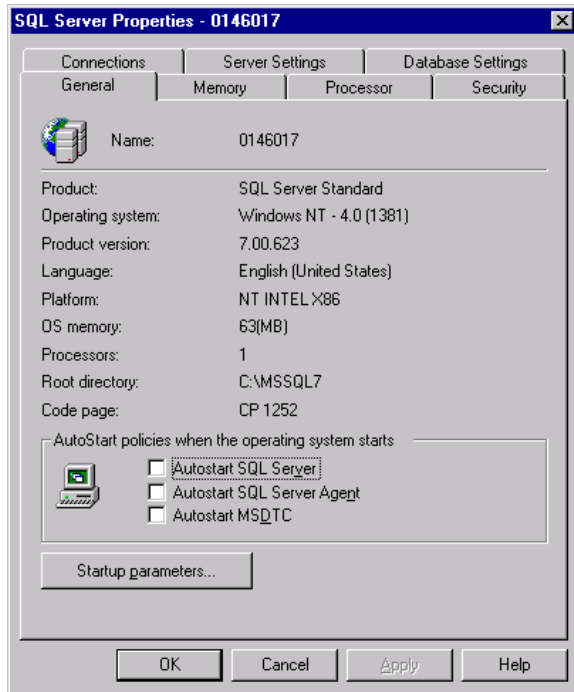
#### 1.4 Service Packs for SQL Server

Service Packs (Microsoft's name for patchsets) are simple to install.

- Download the service pack executable file from the internet onto the target machine
- Run the service pack executable file – this will unpack the contents of the file (equivalent to tar).
- Shutdown all three SQL Server services
- Run the service pack installation executable (normally setup.exe)
- Startup the three SQL Server services

They do **not** have to be installed one after the other. It makes sense to set all new SQL Servers to the latest service pack level, unless there is a requirement to keep a SQL Server at the same level as, say, its development equivalent.

The service pack level of a SQL Server can be worked out by taking the value of Product Version. A table of product version numbers to Service Packs should be available somewhere, although I have not managed to locate it.



SQL Server 7.0 had Service Pack 2 on release as of July 2000

## 1.5 De-Installing SQL Server

De-Installing SQL Server can cause the usual Microsoft Registry and DLL disasters. If you absolutely need to de-install, run the de-install without removing any of the shared library/DLL files.

## 1.6 Post Installation

Three services are required for SQL Server:

1. MSDTC (data transfer functionality)
2. MSSQLServer (SQL Server itself)
3. SQLServerAgent (automatic jobs and alerts functionality)

For enterprise installations, all three of these should be set to start automatically. The startup option should log on as the System Account.

These are configured through **Control Panel** then **Services** then clicking the **Startup** button for each service.



### Lab 2.2 – Set up these services

## 1.7 Remote Administration

SQL Server is most easily controlled from a desktop PC. Each Server can be added to the list (registered) within Enterprise Manager, then fully controlled from the desktop.

We normally use the **sa** account to connect to remote servers.

Once a server is registered, it can then be administered from the desktop enterprise manager just as if it was running on the desktop machine itself.

Registered servers can be grouped together within Enterprise Manager.

See Appendix C for more details.

# 3. Security

## 3.1 Logins and Users

Two separate security accounts exist in SQL Server, rather than the single username in oracle:

1. Logins – A login account is for an entire Server.
2. Users – A user account is for a particular database within a Server

Accounts must have **both** a login account for the server and a user account for a database or they will not be able to connect.

In addition there are NT accounts, typically handled through multi-server domains and maintained by system administrators rather than DBAs.

There are two methods of authentication for logins: Windows NT authentication (equivalent to an ops\$ login in oracle) and mixed mode authentication, which is a username/password separate from the operating system.

So service access requests would involve four stages:

1. Create login account
2. Create user account – one for each database for which access is required
3. Assign login account to above user accounts
4. Assign permissions and roles to the above user accounts

Although the whole process is simplified by using the wizards.

The **sa** login is the system administrator – equivalent to sys in oracle. It should not be used outside the DBA team. Unfortunately, by default it has no password, and application users will often want to keep the password blank and use it as their only login account.

The **guest** account should normally be dropped for security reasons.

*Lab 3.1 – add then revoke logins and Northwind users.*

### 3.2 Permissions and Roles

Roles exist in SQL Server as in oracle. Roles are either Server level or Database level.

There are 7 Fixed Server roles (sysadmin; dbcreator; diskadmin; processadmin; serveradmin; setupadmin; securityadmin), 10 Fixed Database roles, and as many User-defined Database roles as you want to create.

The sysadmin role has DBA+ privileges, and so should not be granted outside the DBA team.

Object Permissions are granted to user accounts or database roles through the Enterprise Manager GUI. As in oracle, permissions can be implemented at column level. Roles can be assigned passwords.

Three separate object permissions exist in SQL Server:

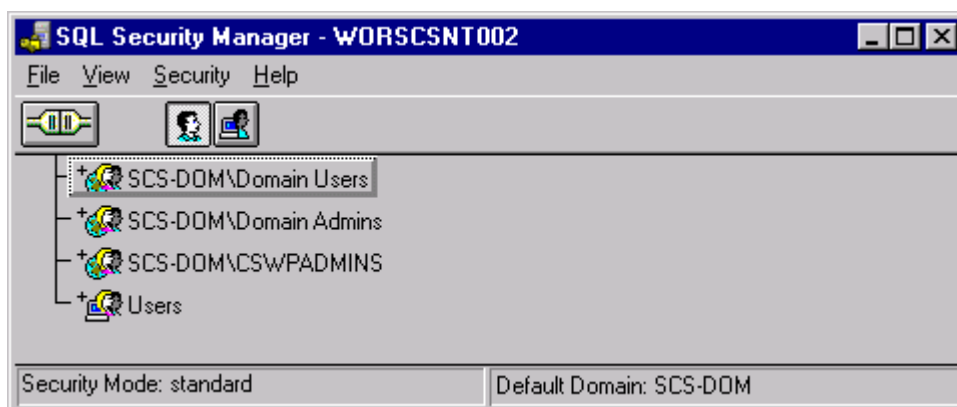
1. **Grant** – can perform action
2. **Deny** – cannot perform action (strong). This applies even if the user account is a member of a role which has been granted the permission.
3. **Revoke** – cannot perform action (weak). This will be overridden by a grant to a role which the user account is a member of.

So Grant and Revoke are the same as in oracle. The additional Deny command is an extra strong form of Revoke.

*Lab 3.2 – create a role and add users, then grant statement and object permissions to users and/or role.*

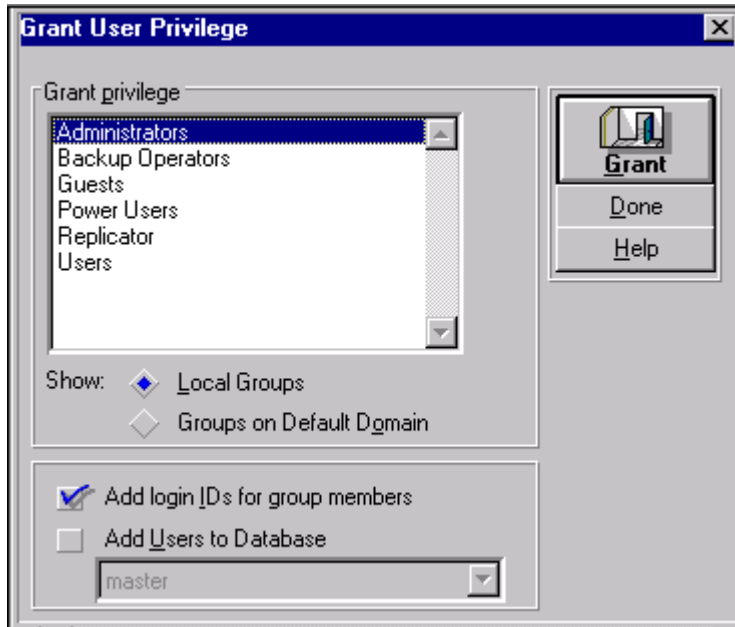
### 3.3 Security in SQL Server 6.5

SQL Server 6.5 has a separate programme for security, SQL Security Manager.



It looks like this:

Within SQL Security Manager, use "Security" -> "Grant New" to control users.



This is the preferred method of controlling users in SQL Server 6.5, although the Enterprise Manager GUI is similar in its handling of login and user accounts:



## 4. Managing Database Files

All databases have a primary data file (.mdf) and one or more transaction log files (.ldf) A database may also have secondary data files (.ndf).

Data is stored in 8-kilobytes blocks of contiguous disk space called pages. Tables and Indexes are stored in extents of 8 contiguous pages, or 64kb.

As in Oracle, data is modified in the buffer cache, the modification recorded in the transaction log file, with the checkpoint process periodically writing all completed transactions to the disk files. For this reason, Microsoft recommends that hardware write-caching disk controllers be disabled.

Ideally, the transaction log files should be placed on RAID-1 mirrored disks, and the data files placed on RAID-0 or RAID-5 (parity) striped disks. Transaction log file and data files should exist on separate physical disks with separate I/O controllers.

Filegroups can be created within a database to manually place individual tables and indexes onto individual disk drives. However, disk striping normally produces the same performance benefits as filegroups without all the extra work.

When creating a database, it makes sense to accept the defaults of unlimited file growth in 10% increments. This is especially crucial to the transaction log, as changes cannot be made to the data of a database with a full transaction log. A maintenance plan can be set up (see below) to periodically shrink files. Transaction log files are initially created by default to be 25% of the size of the data files. This default should be accepted unless the database data will have an unusually low number of changes, in which case a smaller transaction log file would be appropriate.

After creating, dropping or modifying a database, back up the master database.

*Lab 4.1 – create and modify a database.*

## 4.2 Database Options

Important options can be set for a database. Leave the 6 options below unchecked in normal circumstances. Database options are set through the options tab within database properties.

Database option	Description
<b>dbo use only</b>	Limits use of the database to the database owner only – use during development.
<b>read only</b>	Defines a database as read-only – use to set security for decision support databases.
<b>select into / bulkcopy</b> (like <code>_disable_redo_logging</code> )	Allows a database to accept non-logged operations – use during bulk copying of data or when using SELECT INTO to conserve transaction log space. Restore from backup operations will not restore a database with non-logged operations to a consistent state.
<b>single user</b>	Restricts database access to one user at a time – use when performing maintenance.
<b>trunc. log on chkpt.</b> (like <code>noarchivelog</code> mode)	Causes the transaction log to be truncated (committed transactions are removed) every time that the checkpoint process occurs – use during development to conserve transaction log space. Do not use in a production database.
<b>Autoshrink</b>	Determines whether the database size shrinks automatically. Databases can also be manually shrunk using the database -> all tasks -> shrink database menu option

	(T-SQL commands DBCC SHRINKDATABASE or DBCC SHRINKFILE).
--	--

*Lab 4.2 – Set and unset some database options.*

## 5. Backup

Backup can be done through Arcserve with the Arcserve agent for SQL Server. But our preferred backup methodology is to use SQL Server's own backup utility. This will dump backup files to disk, which will then be backed up by the Arcserve filesystem backup. The option to delete old backups should be ticked and backups should be kept for 1 week or so on disk, because the disk files will always be available from the Arcserve tape backups.

Backups can be written direct from SQL Server to

- disk file
- tape
- pipe

Most commonly SQL Server uses backups to disk file.

All SQL Server backups are online (hot). While a backup is in progress, it is impossible to:

- create or modify databases;
- autogrow files;
- create indexes;
- perform non-logged operations.

There are three kinds of SQL Server backup:

1. Full backup – backs up data files and the part of the transaction log that took place during the full backup. Equivalent to a full online backup in oracle.
2. Differential backup – backs up the extents which have been modified since the last full backup, and the part of the transaction log that took place during the differential backup.
3. Transaction log backup – backs up and then truncates the transaction log. Equivalent to a log switch in an oracle instance in archivelog mode.
4. Transaction log backup with NO TRUNCATE option – backs up the transaction log. Done before attempting a restore to keep a copy of the current transaction log.

Database maintenance plans are used to perform backups (see section 10 below), except for one-off ad-hoc backups. In normal practice, a full backup followed by a transaction log backup is scheduled every night.

*Lab 5 – run a one-off backup.*

## 6. Restore

When disaster strikes, the procedure to follow is:

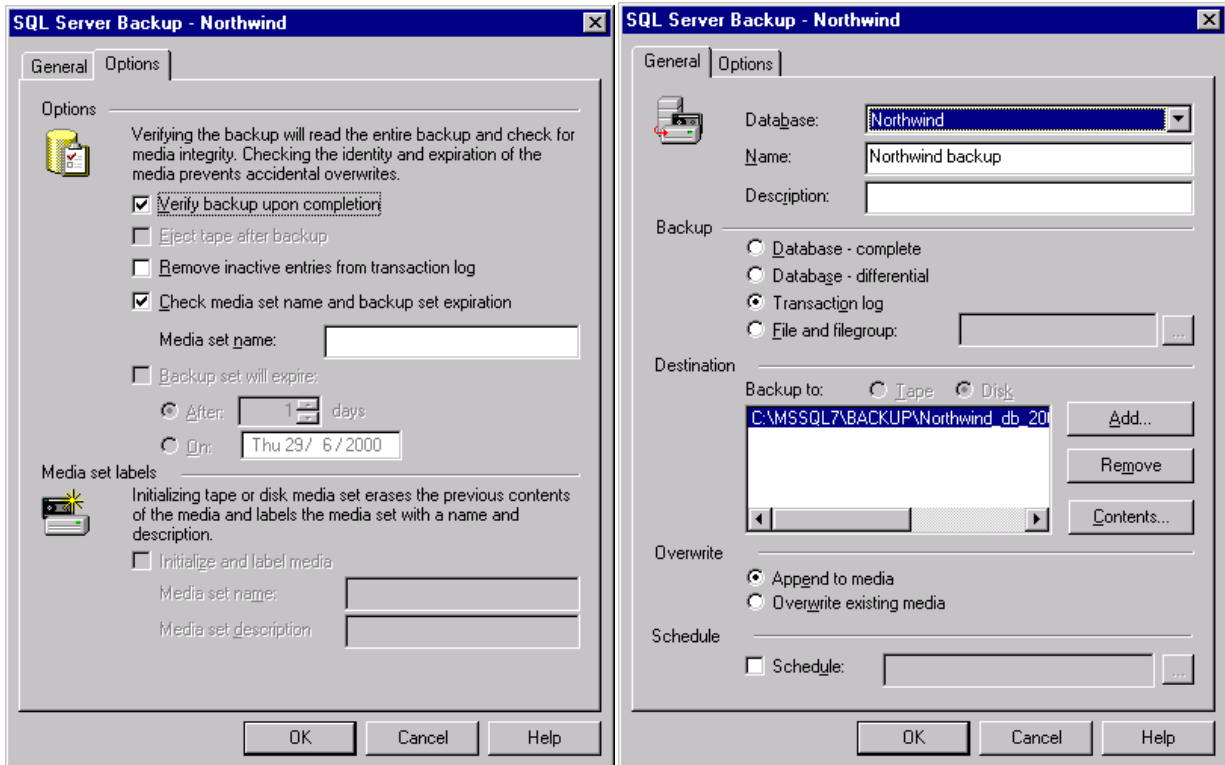
1. Tick the “Single User” and “DBO Use Only” boxes in “database -> properties -> options”. This will prevent users interfering with the restore in progress.
2. Back up the transaction log **without truncate**.

To do this:

1. select “database -> all tasks -> backup database”
2. Tick the backup “Transaction Log” radio button in the general tab
3. **Important** - Untick the “Remove inactive entries from transaction log” radio button in the options tab.

Alternatively, use the T-SQL syntax “**backup log <database> with nottruncate**”

This is required to regress in case something goes wrong with the restore. It is the same as the oracle requirement to copy the on line redo logs before starting a restore.



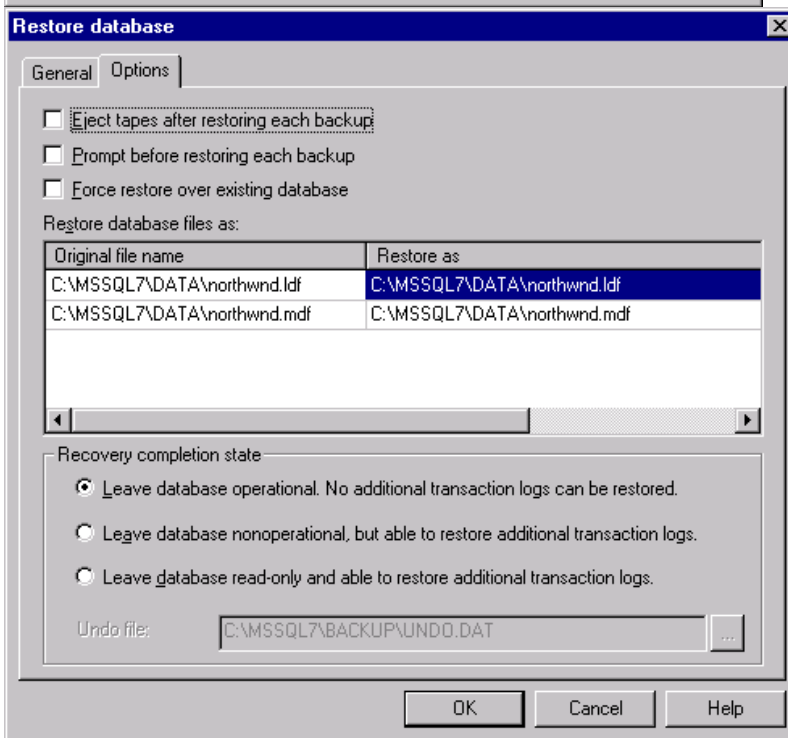
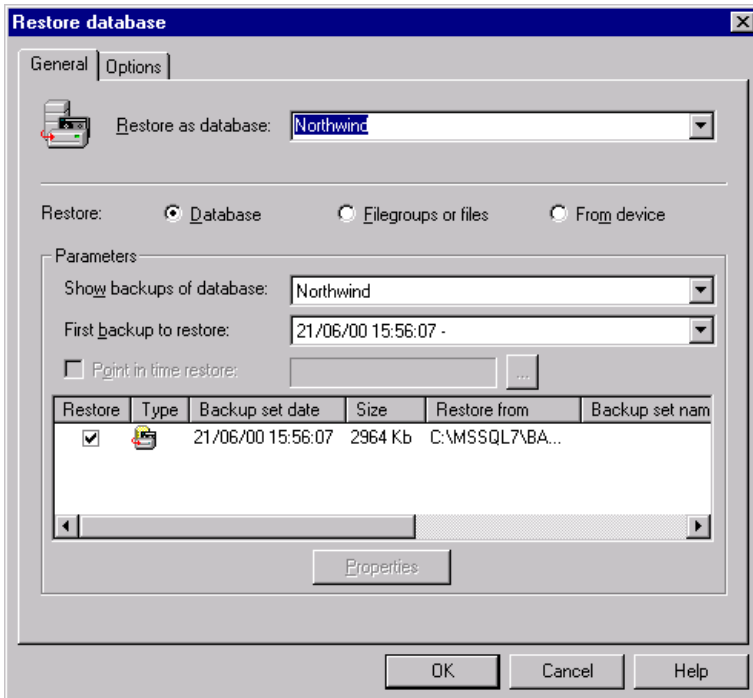
3. Run the restore: database -> all tasks -> restore database.

If you have multiple backups to restore, click the “Leave database non-operational, but able to restore additional transaction logs” button in the options tab. The restore will then have to be run for the extra backups. Cases where this applies are:

- Partial backups are being employed; and/or
  - Transaction log backups are taken at intervals inbetween the full or partial backups
- Either of which would be fairly unusual configurations.

If you wish to restore to a point in time, specify this in the general tab.

Note that you can restore into a different database or into a new database name from the “Restore to database” box in the general tab.



Lab 6.1 – restore a database

6.2 Rebuilding Master

If the master database is lost, it should be restored from backup in exactly the same way as any other database. However, if you cannot even start SQL Server in the first place (because master is damaged), it is possible to rebuild the master, model and msdb databases.

To do this:

1. Run the **Rebuildm.exe** command-prompt utility in Mssql7/Binn.
2. Restart the SQL Server service
3. Restore backups of the master, model and msdb databases in the normal way.

Or, if there are no good backups of these three databases, Manually attach all the databases to master using the **sp\_attach\_db** and **sp\_attach\_single\_file\_db** stored procedures. Also manually recreate all jobs in the msdb database and manually make any required changes to the model database.

### 6.3 Standby Databases

SQL Server standby databases are similar to oracle standby instances. Like oracle:

- Transaction logs are shipped from the live to the standby database and then applied to the standby database
- When recovery is performed (eg “restore database with recovery” T-SQL statement), the standby database is no longer a standby database. It must be rebuilt from a backup of live to function again as a standby database.
- Read only access is allowed to the standby database.

### 6.4 Cloning Databases

To copy a database within the same server, use the DTS export wizard within Enterprise Manager (see chapter 8 below).

To clone a database to another server, use the DTS export wizard within Enterprise Manager, **or** image copy its database files and transaction log files to the target server, then attach it using the **sp\_attach\_db** and **sp\_attach\_single\_file\_db** stored procedures.

To clone a full server to another machine:

1. Place a fresh SQL Server installation on the target machine (note that binaries cannot just be copied over because of the NT registry). Patch this up to the same level as the source server.
2. Create empty databases on the target server with the same filenames and transaction log filenames as are on the source server.
3. Image copy all database files and transaction log files to the target server, when both the source and target servers are down.
4. Use the **sp\_dropserver** and **sp\_addserver** stored procedures to give the target server the correct name.
5. If domain based NT authentication is used, you may have to re-register these logins. (in 6.5, this is done with the SQL Security Manager)

## 7. Automating Administrative Tasks with Jobs and Alerts

For alerts, there are predefined errors , e.g. Error 9002 – Transaction Log Full. User defined errors can be added – these must have error numbers greater than 50000.

Alerts can be emailed. This requires: an e-mail client on the machine running SQL Server; a domain user account used for the SQL Server services; a mail profile for this account; then SQL Mail configured within Enterprise Manager.

Jobs can be *multiserver* – in that case one SQL Server is the controlling master server and executes the job on a number of target servers.

Note that the SQL Server Agent service **must** be running for jobs and alerts to execute automatically.

Lab 7 – Create jobs and alerts and publish SQL Server data on the Web

## 8. Transferring Data with the DTS

DTS = Data Transformation Services

DTS Export/Import		Simple export/import operations to and from various data sources, including SQL Server, Oracle and Access.
DTS Designer		A 5GL GUI transaction-oriented workflow engine which will perform complex series of operations. Very powerful and easy to program and maintain.
DTS Object Transfer		Transfers objects between SQL Server 7 databases, including tables, procedures, rules, logins, users, etc.
DTS bulk insert		Like direct path sql*load, this is the fastest method of loading text files into a database.
Bcp – bulk copy program		An old command prompt utility like the bulk insert.
Replication		

Lab 8 – DTS Export/Import and Designer

## 9. Monitoring Tools

SQL Server is largely self tuning. Because it is designed for only one operating system, it is tightly integrated into NT memory management. The SQL Server Buffer Cache, for example is determined dynamically and automatically. This leaves little in the way of performance tuning and configuration for the DBA. Parameters such as the buffer cache can be manually forced to a particular value, but this is not recommended without more advanced tuning knowledge than is covered in this course.

Note that SQL Server 6.5 is much poorer in this self tuning ability.

Indexing and optimal coding still has to be performed manually, although SQL Server includes good tools to assist with this. SQL Server uses a cost based optimizer, so statistics have to be refreshed for optimal explain plans.

The DBA may also need to be involved in resolving Locking problems.

There are five main monitoring tools in SQL Server:

**9.1 Microsoft NT Event Viewer** in “Start -> Programs -> Administrative Tools (Common)”  
An expanded equivalent of the unix /var/adm/messages file. The SQL Server logs are also available in “SQL Server Enterprise Manager” under Management

**9.2 SQL Server Performance Monitor** in “Start -> Programs -> Microsoft SQL Server 7.0”  
This shows a huge range of common statistics such as Buffer Cache Hit Ratio, writes per second, etc. A brief definition of each statistic is included.  
Oracle Performance Manager would be the closest Oracle equivalent.

### 9.3 Current Activity in “SQL Server Enterprise Manager” under Management

This lists current processes (like Oracle Top Sessions), and database locks (like Oracle lock manager).

### 9.4 SQL Server Profiler in “SQL Server Enterprise Manager -> Tools”

This is similar to Oracle Trace and Oracle Expert, but has an easier user interface.

The easiest way to use this is with the **Create Trace Wizard** (“Tools -> Create Trace Wizard”). This will collect statistics for a variety of problems. For example, the worst performing queries or the causes of deadlocks. Trace outputs can be saved, especially if the Index Tuning is to be done.

The **Index Tuning Wizard** (“tools -> index tuning wizard”) is also useful. It produces recommendations for adding or dropping indexes based on the type of SQL statements being run. A trace must have been first created and saved.

### 9.5 SQL Server Query Analyzer in “SQL Server Enterprise Manager -> Tools”

This displays the explain plan for SQL Statements. It is also useful as a window for actually executing statements.

*Lab 9 – Use each of these 5 tools*

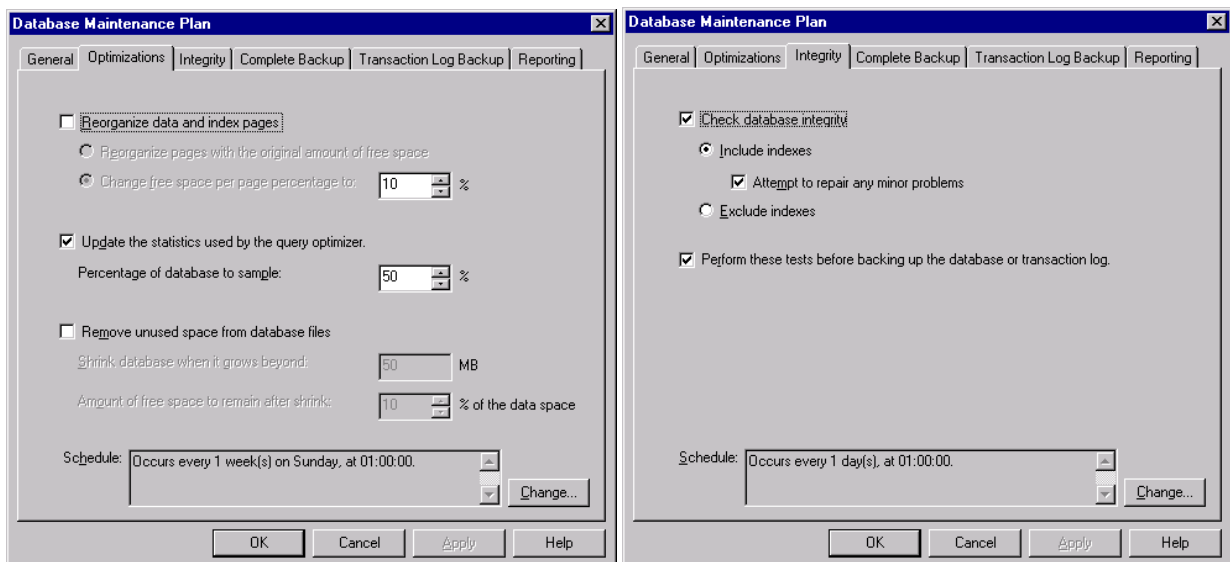
## 10. Maintenance Plans

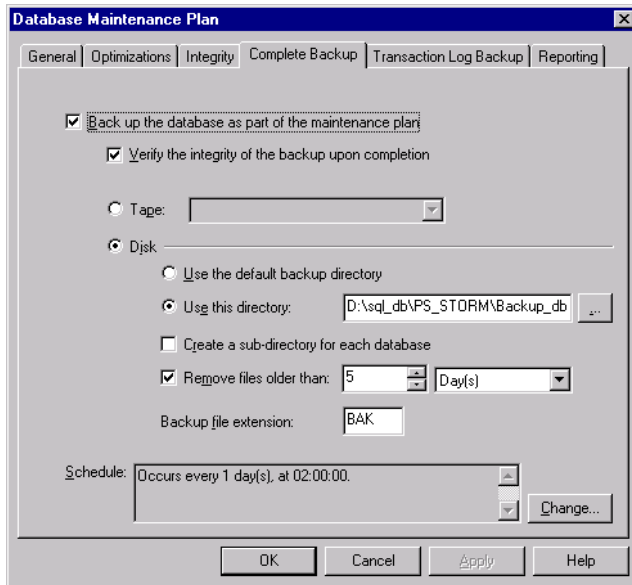
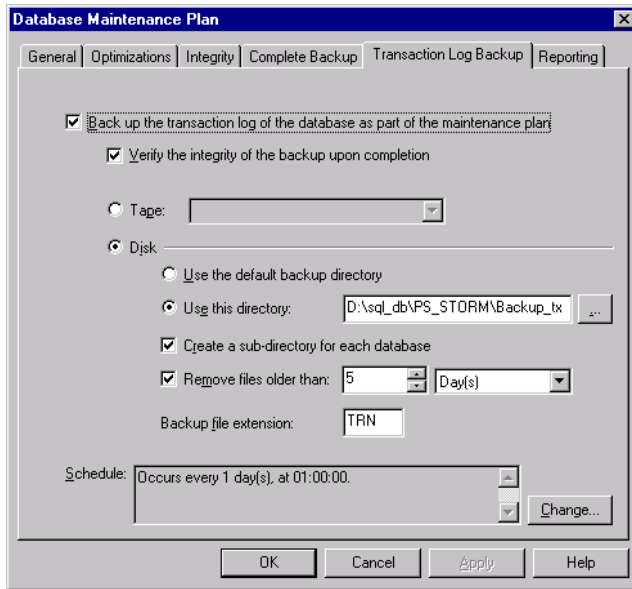
Maintenance Plans perform tasks such as: Backup; Data File Maintenance, Log File Maintenance; Data Integrity Checks; Refresh Data Optimization Information.

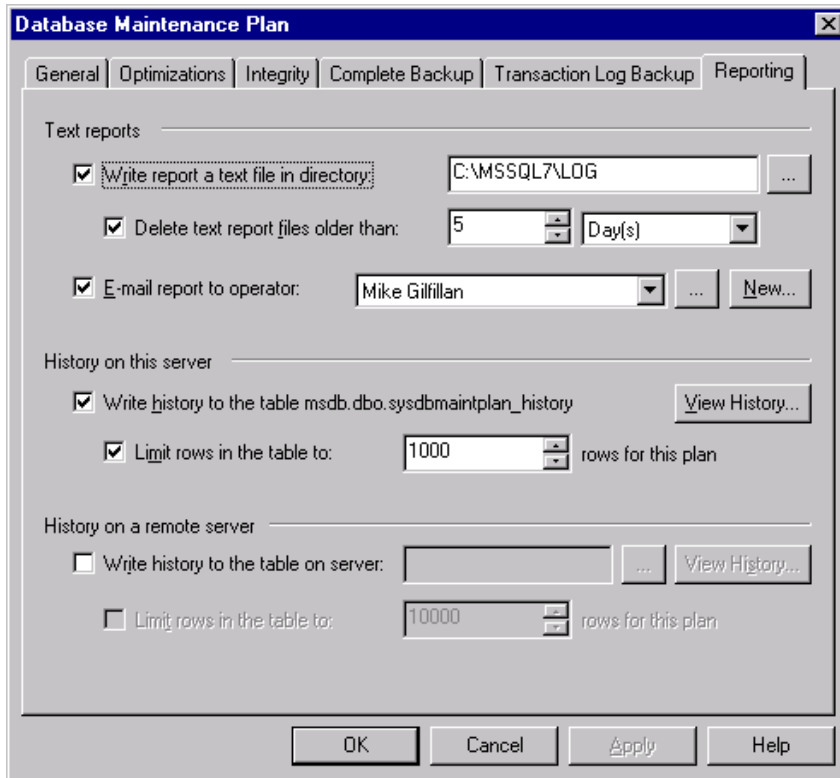
We use the maintenance plan to:

- Backup the databases
- Backup and then truncate the Transaction Logs
- Regenerate statistics for the query optimizer
- Check database integrity

Below is an example of a standard maintenance plan:







### Lab 10 – Create a new maintenance plan

Note that the SQL Server Agent service **must** be running for maintenance plan jobs to execute automatically.

## 11. Replication

Three types of Replication exist

1. **Snapshot Replication** – just like Oracle snapshot replication, data is refreshed periodically. This can have **updating subscribers** (updateable snapshots), where the subscriber can change data in the snapshot and this change will be replicated back to the snapshot.
2. **Transactional Replication** – this replicates data by monitoring transaction logs. Block changes for replication are specifically marked in the transaction logs. This is like snapshot replication, but with constant data copying. So it provides the same functionality as distributed database triggers, but with fewer potential problems. This can have updating subscribers, where the subscriber can change data and this change will be propagated back to the publisher.
3. **Merge Replication** – this uses triggers on each copy of the data. Conflicts are resolved at merge time by means of a “timestamp” column on each row.

Three server types are involved:

1. **Distributor** – responsible for synchronizing data between publishers and their subscribers. The data to be synchronized is stored on disk files on the Distributor machine. A distribution database stores distribution history, and in transactional replication, also keeps the information culled from the redo logs for propagation.
2. **Publisher** – source of data
3. **Subscriber** – receiver of data

In **Push** Subscription the Publisher defines the subscription to the data.

In **Pull** Subscription the Subscribers define the subscriptions to the data.

To set up Replication, use the Configure Publishing and Distribution Wizard in Enterprise Manager.

Note that SQL Server replication can work to subscribers which are not themselves SQL Servers. For example, SQL Server could replicate data to an Oracle or Access database.

*Lab 11 – Set up various types of replication on the desktop.*

## Appendix A – Microsoft Support and Other Helpful Resources

- SQL Server Books Online – look here first for everything, superb documentation installed by default
- Microsoft TechNet – metalink type database contains most previously encountered issues. Available on monthly free issue CDs.
- Microsoft Support phone line – ???????? – we have ????? cover ????? hours. AFchange
- Microsoft SQL Server site [www.microsoft.com/sql](http://www.microsoft.com/sql) – masses of useful documents and programs
- Microsoft “TechNet on the web” SQL Server site [www.microsoft.com/technet/sql](http://www.microsoft.com/technet/sql)
- Microsoft Support site [support.microsoft.com](http://support.microsoft.com) – metalink type facility
- SQL Magazine [www.sqlmag.com](http://www.sqlmag.com) – only partially available without subscription
- Swynk.com [www.swynk.com/sysapps/sql.asp](http://www.swynk.com/sysapps/sql.asp) – independent techie community site
- Microsoft Technical Consultant Mark Hickin, 0118-909-3791, [mhickin@microsoft.com](mailto:mhickin@microsoft.com)
- Independent Consultant Gordon McHarg, 0141-229-5300, [gordonm@api-software.com](mailto:gordonm@api-software.com)

## Appendix B – List of Differences between SQL Server and Oracle

1. In SQL Server, only one instance runs per platform.
2. SQL Server tablespaces are called **databases**, and lie half way between an oracle tablespace and an oracle instance.
3. Five databases always exist: **master** (=system); **tempdb** (=temp); **distribution** (for distributed transactions); **model** (the template for all new database/tablespaces); and **msdb** (automatic scheduled jobs information). Three sample databases are also created on install, but can be safely deleted: pubs; northwind; user1.
4. SQL Server databases comprise one or more datafiles, as in oracle, but also have one or more online redo log files. This is the principal difference between SQL Server and oracle; the idea of the temporary tablespace, system tablespace, and each schema tablespace having their own set of online redo logs takes some getting use to.
5. SQL Server has **logins** and **users**. Logins are specific for a SQL Server instance. Users are specific to a database. A login will have one user for every database it has access to. Logins have systemic privileges (create database, etc.), Users have DDL and DML privileges. Standard practice is to name users the as their parent login - this is done by default.
6. Users with execute permission on a procedure / select permission on a view will fail to use this if there is broken ownership with referenced objects, unless they are explicitly given access to the referenced table.

7. **Revoke** (neutral) and **Deny** (strong) are both available in place of Oracle Revoke. The difference is whether or not they can be overridden by role or user privileges. Deny appears as a red cross, Grant as a green tick, and Revoke does not appear in Enterprise Manager privilege box. [p109]
8. Objects can have their schema changed in SQL Server without being rebuilt, using **sp\_changeobjectowner** *object, owner*. But SQL has to be rewritten to specify the right user, especially with broken ownership chains, so probably this is more trouble than it is worth.
9. SQL Server does not have synonyms.
10. Standard Practice on SQL Server is to have all objects, even views and stored procedures, owned by the dbo user. Other schemas would tend to have their own database/tablespace. However, this is just standard practice, and does not have to be followed. But problems could arise be with lack of synonyms and broken ownership chains.
11. SQL Syntax is slightly different. SQL Server books online has full syntax with a search capability. However DDL and DBA operations should be done by the Enterprise Manager GUI, which has wizards available. SQL Server Enterprise Manager is vastly superior to the equivalent Oracle GUIs. A few operations cannot be done through Enterprise Manager: Column permissions have to be modified through SQL; Filegroups have to be created through SQL.
12. In DDL, whitespace and special characters must be covered with []. E.g.: **REVOKE ALL ON [order details] FROM PUBLIC; DENY CREATE DATABASE TO Eva, [Corporate\ErikB], Ivan.**
13. SQL Server has its equivalent of v\$ and dba\_ views. It also has a collection of system stored procedures which return v\$ information or actually perform system DDL.
14. SQL Server has **application roles** in addition to standard roles. Application roles are password identified while standard roles are always enabled for a user. When an application role is enabled, no other privileges are apparent, with the exception of public privileges. Application roles would typically be enabled and disabled through a VB front end script. Syntax is: **exec sp\_setapprole 'approle\_name', 'password'**
15. Databases have a primary datafile (\*.mdf), possibly some secondary datafiles (\*.ndf); and one or more on line redo log files (\*.ldf).
16. There is no reason to give a database more than one datafile, except for backup/recovery streamlining for very large database/tablespaces.
17. SQL Server has a fixed block size of 8k, and a fixed extent size of 64k. This may not mean an end to defragmentation requests, since there is still a reorganise utility. The reorganise utility can be scheduled to run after backup, and does hot reorgs. Small tables can, apparently, share an extent.
18. Rows cannot span blocks, so the maximum row size is 8k. This means chaining does not happen in SQL Server, but problems will occur if a row physically cannot fit into 8k.
19. The default size of on line redo log files is 25% of the total size of all datafiles. It is also recommended that autoextend be switched on on online redo log files.
20. A maintenance wizard will decide whether or not to grow or shrink on line redo log files, among other things.
21. Autoextend can be in extensions of a fixed size or a percentage of current size.
22. Autoshrink is available, but recommend that it is switched off and shrinking is done by maintenance jobs after backups.
23. Databases can be dbo use only (in development phase) (=restricted session); and can be in single user mode (when doing restores, etc.).
24. **Truncate log on checkpoint** is equivalent to noarchivelog mode in Oracle. Truncating the log on backup is equivalent to archivelog mode in Oracle.
25. On line redo logs are not archived, except on backup. They will therefore grow to much larger sizes than oracle On line redo logs.
26. By default, a database is created with one filegroup, named default. Microsoft recommend that filegroups should only be used for backup purposes. Performance should be handled by striping, even with tables and indexes.

27. Unlike our practice with Oracle on NT, but like our practice with Oracle on Sun: Microsoft recommend on line redo log files should be on separate physical disks, with separate disk controllers, from the datafiles. This is recommended both for performance and for fault tolerance. [p160]
28. During an online backup: cannot create or alter databases; create indexes; perform nonlogged operations such as bulk copy and writetext. These will be failed if attempted after backup is started, or cause backup to stop these are already running.
29. Three types of backup: full backup (=online backup); online redo log backup (=archive); and differential backup. The last type is completely new to SQL Server. It backs up just those blocks which have been modified since last full backup.
30. Create Index statements force the data and index filegroups to be backed up simultaneously.
31. There are no rollback segments in SQL Server. Rollback information is obtained from the online redo logs. This should improve performance.
32. SQL Server is read inconsistent. This will improve performance for some SQL jobs, but produce inconsistent results.
33. Standby databases can be up and available read-only on SQL Server. On Oracle7 they are unavailable.
34. SQL Server creates snapshot disk files, rather than snapshot log tables.
35. SQL Server has *transactional replication*, which replicates data by monitoring redo logs. block changes for replication are specifically marked in the redo logs. This is like snapshot replication, but with constant data copying. So it provides the same functionality as distributed database triggers, but with fewer potential problems.
36. A distribution database stores distribution history, and in transactional replication, also keeps the information culled from the redo logs for propagation.
37. SQL Server does not have parallel Server, although this may be possible via NT operating system.
38. SQL Server does not have sequences. Instead columns can be given the *identity* property.
39. By default, autocommits instantly, rollback unavailable.
40. With begin transaction...commit/rollback transaction statements, get commit, rollback, endpoint functionality. However, because SQL Server does not have rollback segments, modified rows in a transaction are locked. Users cannot even select from the entire table in most cases.

### **Syntactical Differences**

Page = Block

Identity property = sequence

Replication:

Publisher = master site

Distributor = replication process

Subscriber = copy site

Publication = snapshot group

Article = snapshot

Push Subscription is initiated on publisher

Pull Subscription is initiated on subscriber

Merge Replication = updateable snapshots

## **Appendix C – Remote Management**

Most elements of SQL Server can be administered from your desktop *without* going through NSM – which has a tendency to crash the NT server.

Only some of the Remote Administration described below is available with Windows95 - it makes sense to upgrade your desktop PC to NT Workstation.

## Desktop Enterprise Manager to administer remote SQL Servers

Within Enterprise Manager, different servers are registered. We normally use the **sa** account to connect to remote servers.

Once a server is registered, it can then be administered from the desktop enterprise manager just as if it was running on the desktop machine itself.

Registered servers can be grouped together within Enterprise Manager.

## Mapping NT Server Drives to desktop

Files can be accessed directly from the PC. This is one of the few benefits of NT over unix. NT server drives can be mapped to a PC drive in exactly the same way as Novell Server drives, except

1. Specify a \$ at the drive name, e.g. \\hamsprnts01\d\$ for the D: drive in Hamsprnts01
2. The first time you connect, you must supply a username and password to connect to the drive. Username is in the format of <domain name>\username.

For example:

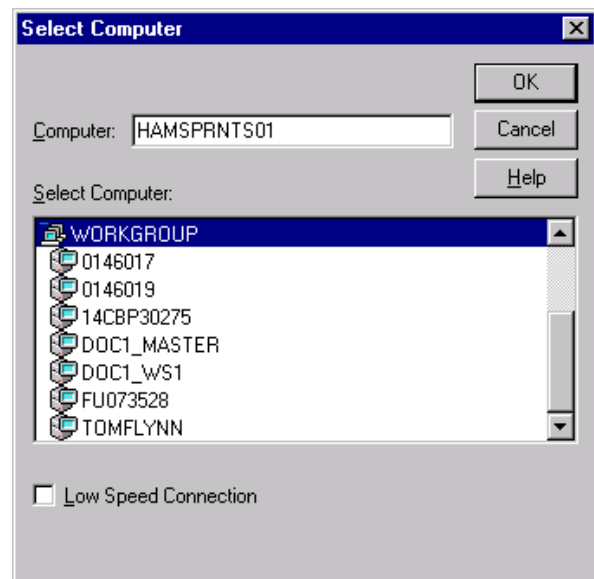


## Event Viewer

You can also use Event Viewer to see remote logs

Use the "File" -> "Select Computer" and type in the name of the remote computer.

For example:



### **Performance Monitor**

The SQL Server Performance Monitor can also view remote servers just as if they were the local machine. Like the Event Viewer, a drop down list of servers to monitor is available.

### **SQL Server Service Manager Console**

The SQL Server Service Manager Console can stop and start any or all of the three SQL Server services on remote servers just as if they were the local machine. A drop down list of servers to control is available.