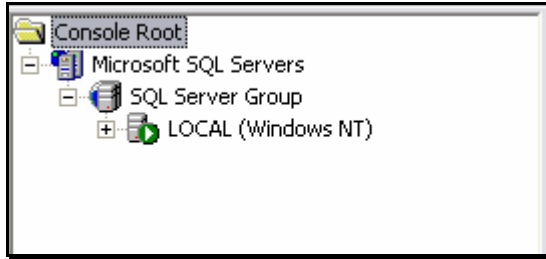


Total Recall SQL Database Maintenance Plan User Manual

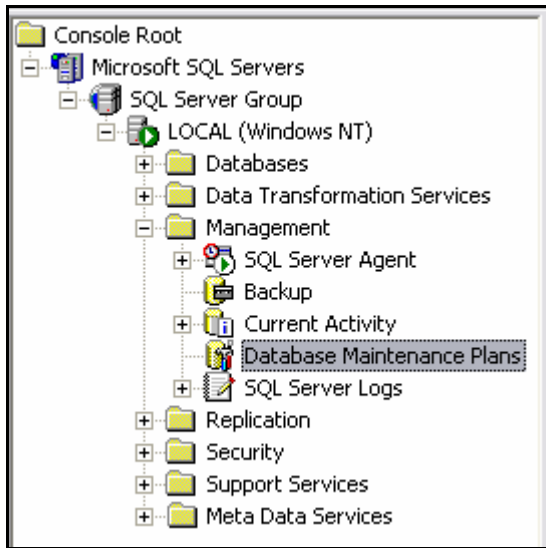
Root Menu



(Fig. 1.1)

When you open SQL Server Enterprise Manager, this root menu will be displayed on the left hand side of the screen.

**Root Menu /
Server and
Management
Expanded**



(Fig. 1.3)

Select your server and expand the Management folder. Within that folder you will find the “Database Maintenance Plans” sub folder. Right clicking this sub folder will open the “Database Maintenance Plan Menu” in *figure 1.4*.

**Database
Maintenance
Plan Menu**



(Fig. 1.4)

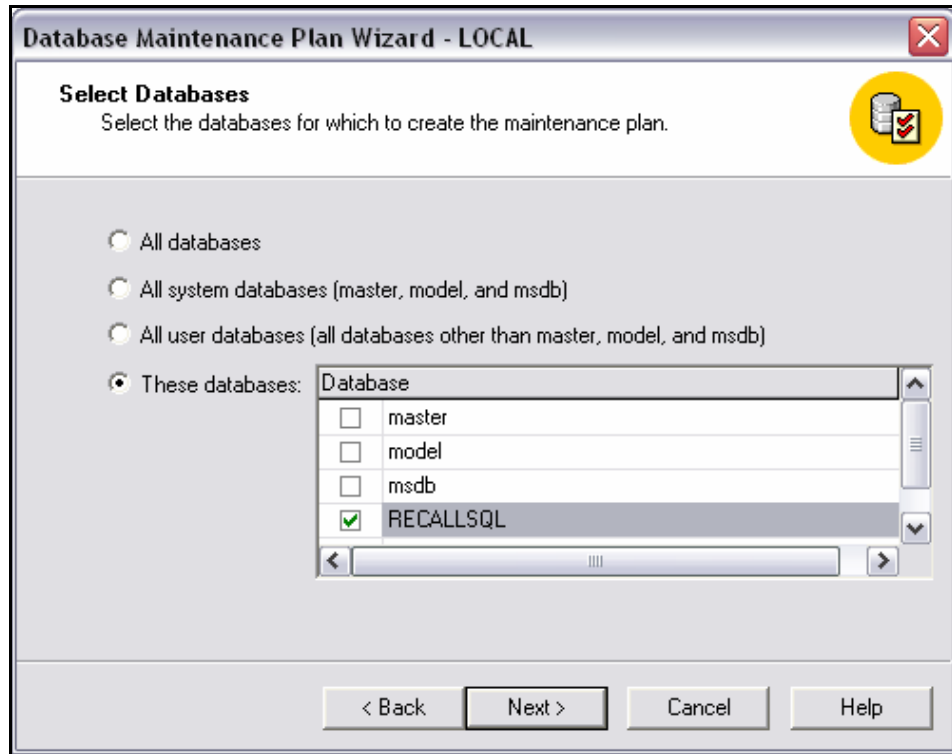
From within this menu you may select <New Maintenance Plan...> to setup a new maintenance plan , <Maintenance Plan History...> to filter previously run maintenance plans based on selected criteria, or any of the other display options shown in *figure 1.4*.

**Welcome
Screen**



(Fig. 1.5)

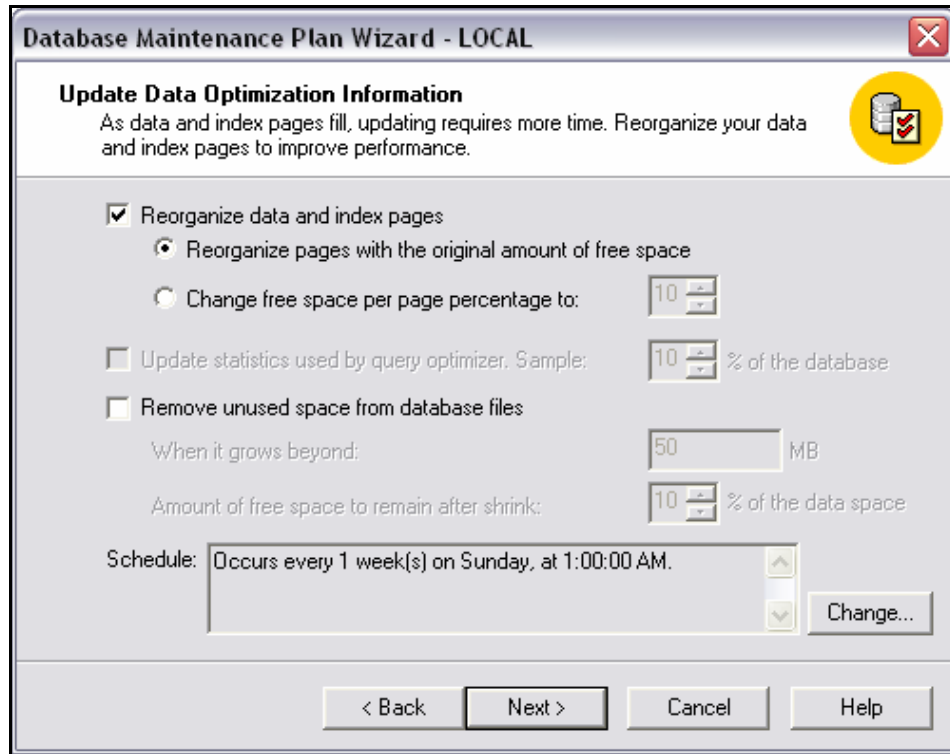
When <New Maintenance Plan...> is selected the Maintenance Plan Wizard will start. This is the Welcome Screen. Click <Next> to continue.

Database Selection

(Fig. 1.6)

This is where you will select the database(s) that applies to the maintenance plan you are setting up. Select your RECALLSQL™ database and click <Next> to continue.

Database Optimization Menu



(Fig. 1.7)

This is where you choose the optimization options for your scheduled maintenance plan. When completed click the <Next> button to proceed to the next screen.

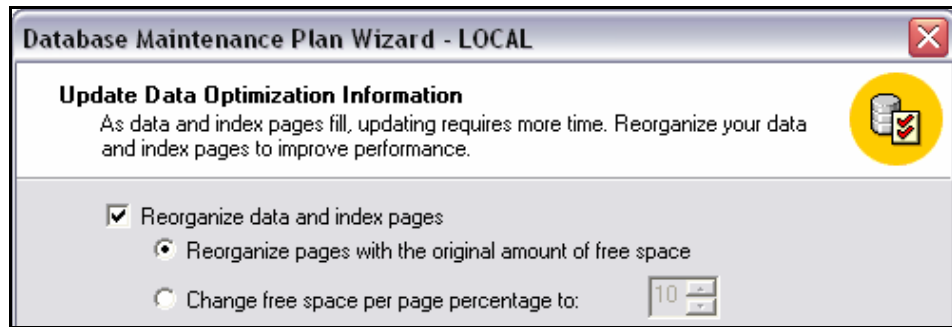
The “**Reorganize data and index pages**” option causes the indexes on the table to be dropped and recreated. The 2 sub options determine how much free space to allocate for each recreated index for future expansion. This option is best run when no users are accessing the database as it is time consuming and may cause slower activity. This option allows you to plan ahead for future expansion to prevent auto growth of the database which may slow your activity.

The “**Update statistics used by query optimizer**” option causes the distribution statistics for indexes to be resampled. This is used by SQL Server™ to speed up transactions and improve table relationships. This option is also time consuming (more so depending on the % of the database) and may cause slower activity.

The “**Remove unused space from database files**” option removes any unused space to shrink the database size. This may cause slower activity and is recommended to be done when no users are accessing the database.

Recommended Settings for Data Optimization

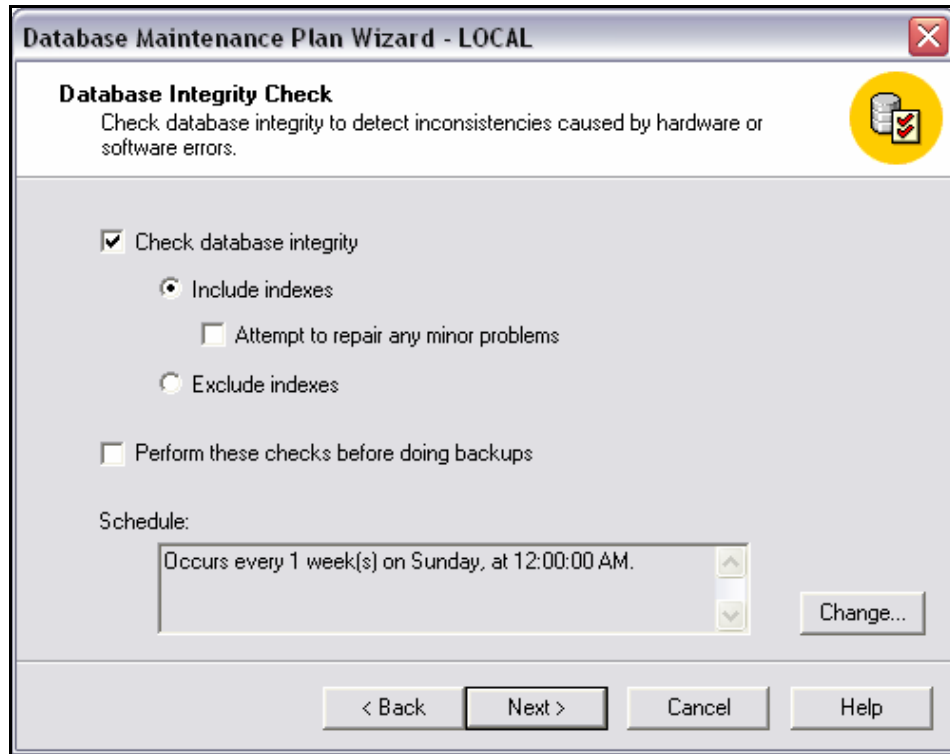
We recommend time be allotted to run at least the first option once a week. Our recommended settings for this plan are below:



**Schedule
Menu**

(Fig. 1.8)

The Schedule Menu (accessed by clicking the <Change...> button from the Optimization, Integrity, Database Backup, or Transaction Log Backup Menus) allows you to set the frequency that the maintenance tasks are executed. When completed click the <Ok> button to return to the previous screen.

**Database
Integrity Menu**

The screenshot shows a window titled "Database Maintenance Plan Wizard - LOCAL". The main heading is "Database Integrity Check" with a sub-description: "Check database integrity to detect inconsistencies caused by hardware or software errors." There is a yellow circular icon with a database cylinder and a checkmark. The options are:

- Check database integrity
 - Include indexes
 - Attempt to repair any minor problems
 - Exclude indexes
- Perform these checks before doing backups

Schedule:

Navigation buttons: < Back, Next >, Cancel, Help

(Fig. 1.9)

This is where you setup the integrity check options and specify how the system should respond to integrity problems. When completed click the <Next> button to proceed to the next screen.

The “**Attempt to repair any minor problems**” option is recommended for your integrity check maintenance plans, but not required.

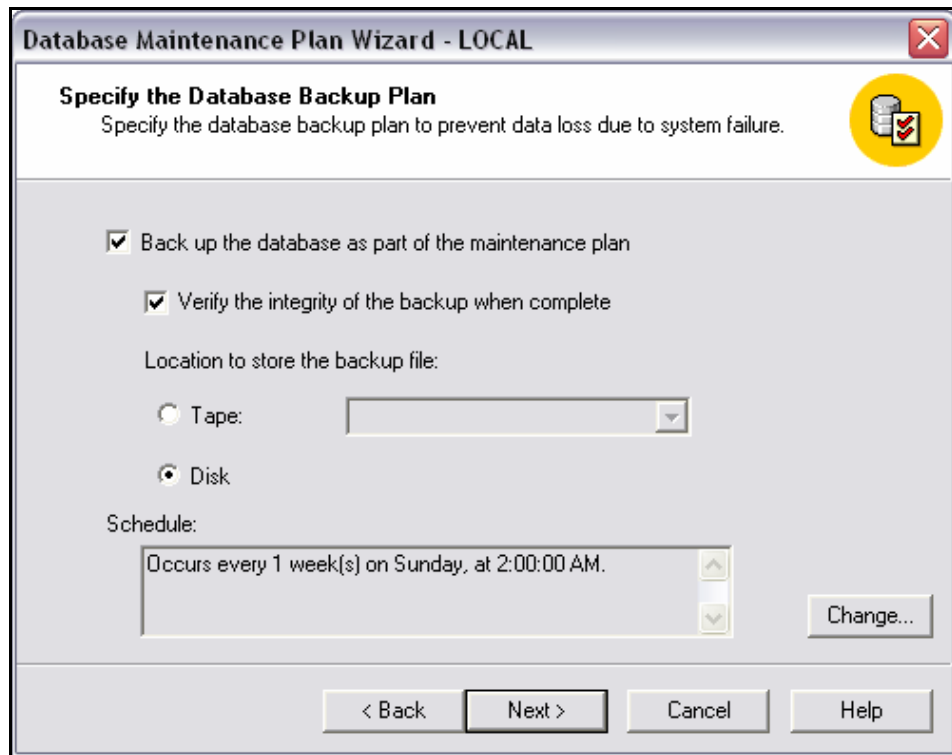
In order to optimize your database backups it is necessary to understand the recovery models and make a decision based on your server specifications and business operations:

Backup Recovery Models

- **Simple Recovery Model** – Requires the least amount of storage. The simple recovery model offers the minimum amount of data required to restore your database to the last successful backup. Any information entered between the last backup and the point of failure will need to be manually reentered.
- **Full Recovery Model** – Requires the most space for a more specific recovery. With a full recovery model your recovery will be a combination of the transaction log backup and the last successful database backup.
- **Bulk Logged Recovery Model** – Requires minimal space, but sacrifices backup comprehensiveness. With a bulk logged recovery model your bulk transactions (such as barcode imports and TMS imports) are not stored in the backup, and more effort is required to recover from a bulk logged backup.

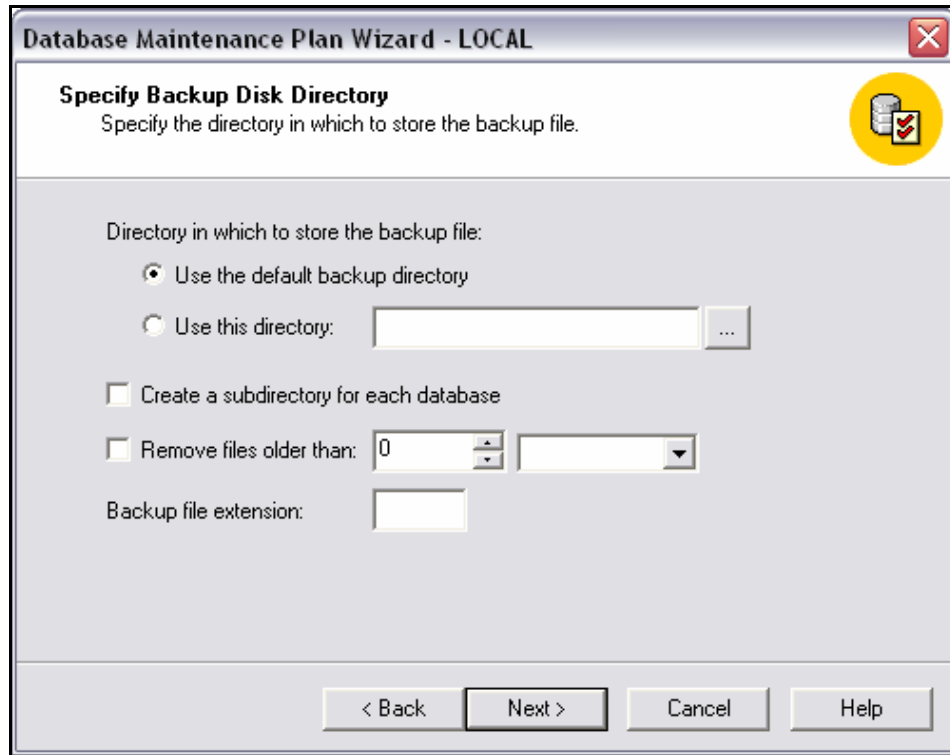
Note: These options are accessed through the Database Options / Options Tab. See figure 3.17.

Database Backup Menu



(Fig. 1.10)

This is where you setup a database backup plan and schedule. When completed click the <Next> button to proceed to the next screen.

**Database
Backup Path**

The screenshot shows a dialog box titled "Database Maintenance Plan Wizard - LOCAL". The main heading is "Specify Backup Disk Directory" with the instruction "Specify the directory in which to store the backup file." Below this, there are several options: "Directory in which to store the backup file:" with two radio buttons: "Use the default backup directory" (selected) and "Use this directory:" followed by a text input field and a browse button "...". There are also two checkboxes: "Create a subdirectory for each database" and "Remove files older than:" followed by two dropdown menus. The first dropdown menu shows "0". Below these is a "Backup file extension:" text input field. At the bottom, there are four buttons: "< Back", "Next >", "Cancel", and "Help".

(Fig. 1.11)

This is where you setup the path your database backup will be saved to, and also allows you to automatically delete older backups. When completed click the <Next> button to proceed to the next screen.

**Database
Transaction
Log Backup
Menu**

The screenshot shows a Windows-style dialog box titled "Database Maintenance Plan Wizard - LOCAL". The main heading is "Specify the Transaction Log Backup Plan" with a subtitle "Specify the transaction log backup plan to prevent failures and operator errors." and a yellow circular icon with a database cylinder and a checkmark. The dialog contains several options:

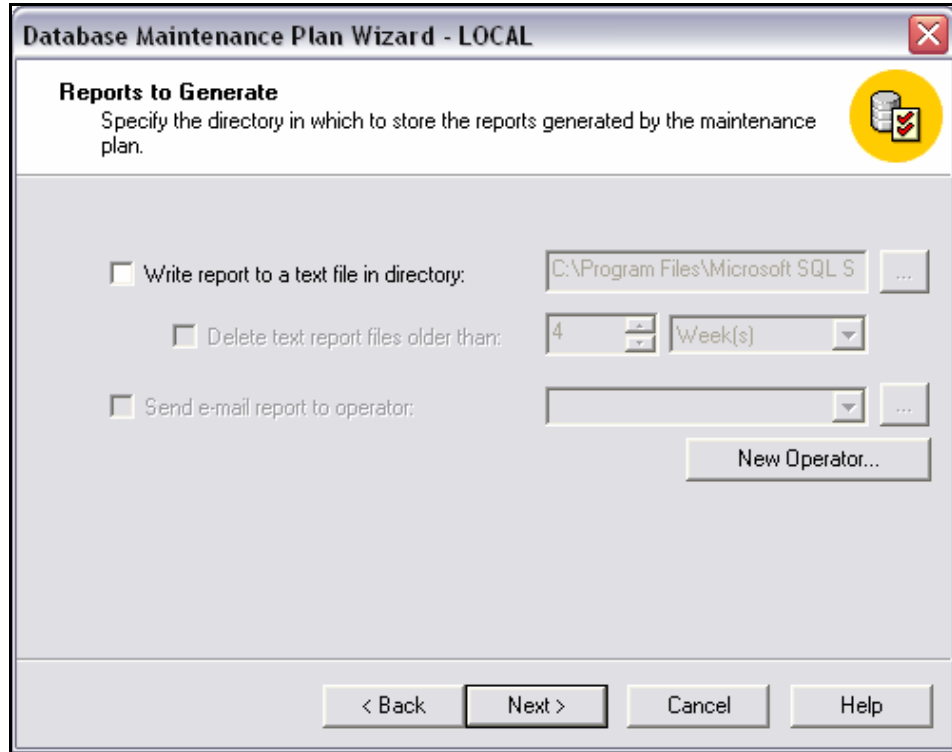
- Back up the transaction log as part of the maintenance plan
- Verify the integrity of the backup when complete
- Location to store the backup file:
 - Tape: [text box with dropdown arrow]
 - Disk
- Schedule: [text box containing "There are no operations to schedule." with up/down arrows and a "Change..." button]

 At the bottom are four buttons: "< Back", "Next >", "Cancel", and "Help".

(Fig. 1.12)

This is where you setup a transaction log backup plan. The transaction log backup is crucial to recover your database as close to the time of failure as possible. This can typically be set to run on a daily basis or more frequently. When completed click the <Next> button to proceed to the next screen.

Note: In order to utilize the transaction log backup it is necessary to use the Full Recovery Model. See figure 3.17.

Reports Menu

The screenshot shows the 'Database Maintenance Plan Wizard - LOCAL' window. The title bar includes a close button (X). The main area is titled 'Reports to Generate' and contains the instruction: 'Specify the directory in which to store the reports generated by the maintenance plan.' To the right of this text is a yellow circular icon with a database cylinder and a checkmark. Below the instruction are three options, each with a checkbox:

- Write report to a text file in directory: C:\Program Files\Microsoft SQL S ...
- Delete text report files older than: 4 Week(s)
- Send e-mail report to operator: ...

A 'New Operator...' button is located below the email option. At the bottom of the window are four buttons: '< Back', 'Next >', 'Cancel', and 'Help'.

(Fig. 1.13)

This is where you setup report paths for your maintenance reports to be stored. This report contains details of the steps performed in the maintenance plan including any error information.

**Maintenance
Plan History
Menu**

Database Maintenance Plan Wizard - LOCAL

Maintenance Plan History
Specify how you want to store the maintenance plan records.

Local server

Write history to the msdb.dbo.sysdbmaintplan_history table on this server

Limit rows in the table to: 1000 rows for this plan

Remote server

History is added to the msdb.dbo.sysdbmaintplan_history table on the remote server.
Windows Authentication is used to log on to the remote server.

Write history to the server: [] [...]

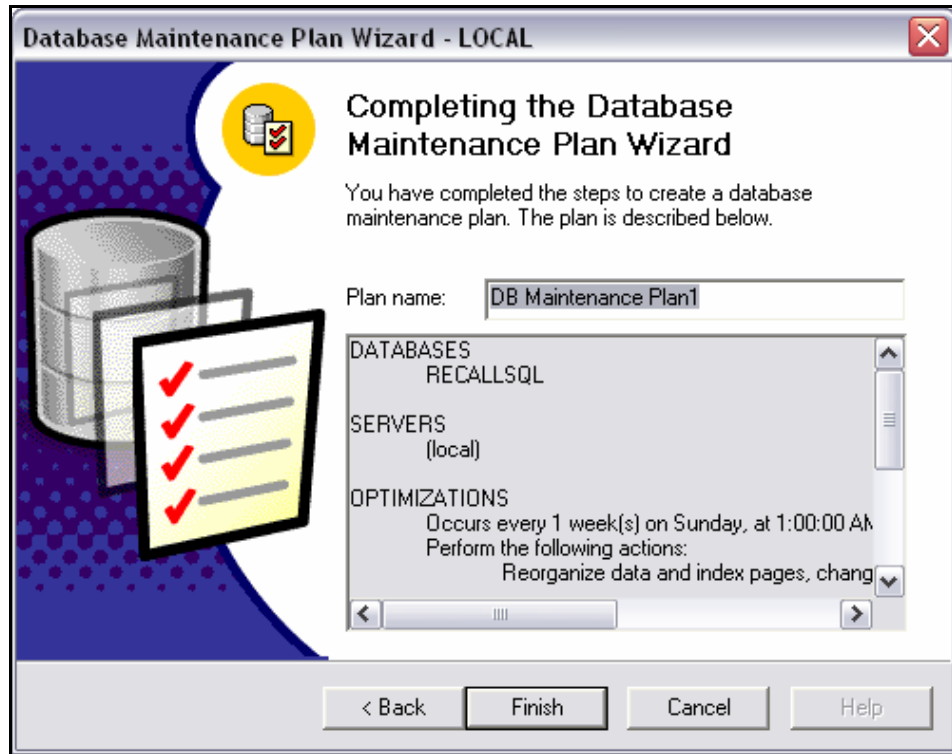
Limit rows in the table to: 10000 rows for this plan

< Back Next > Cancel Help

(Fig. 1.14)

This is where you setup the how maintenance plan records are stored. When enabled these options write a report as rows in the sysdbmaintplan_history table on either the local server or a remote server. The report contains the steps executed by the maintenance plan, including database name, activity, date, success or failure, and any error information.

**Completion
Menu**

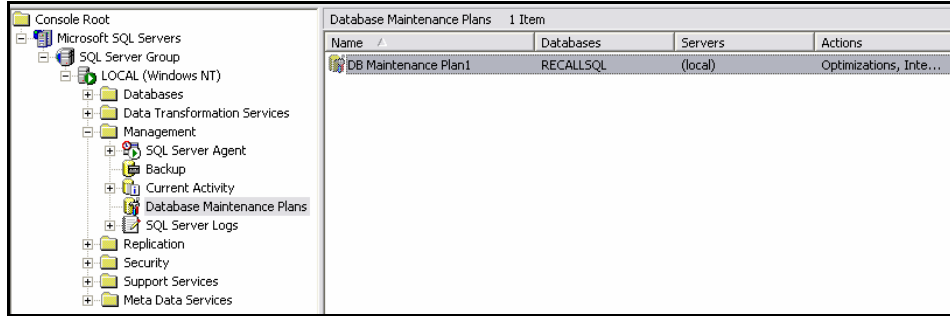


(Fig. 1.15)

This is the completion screen. You may verify the options you selected or rename the maintenance plan here.

Maintenance Plan Settings

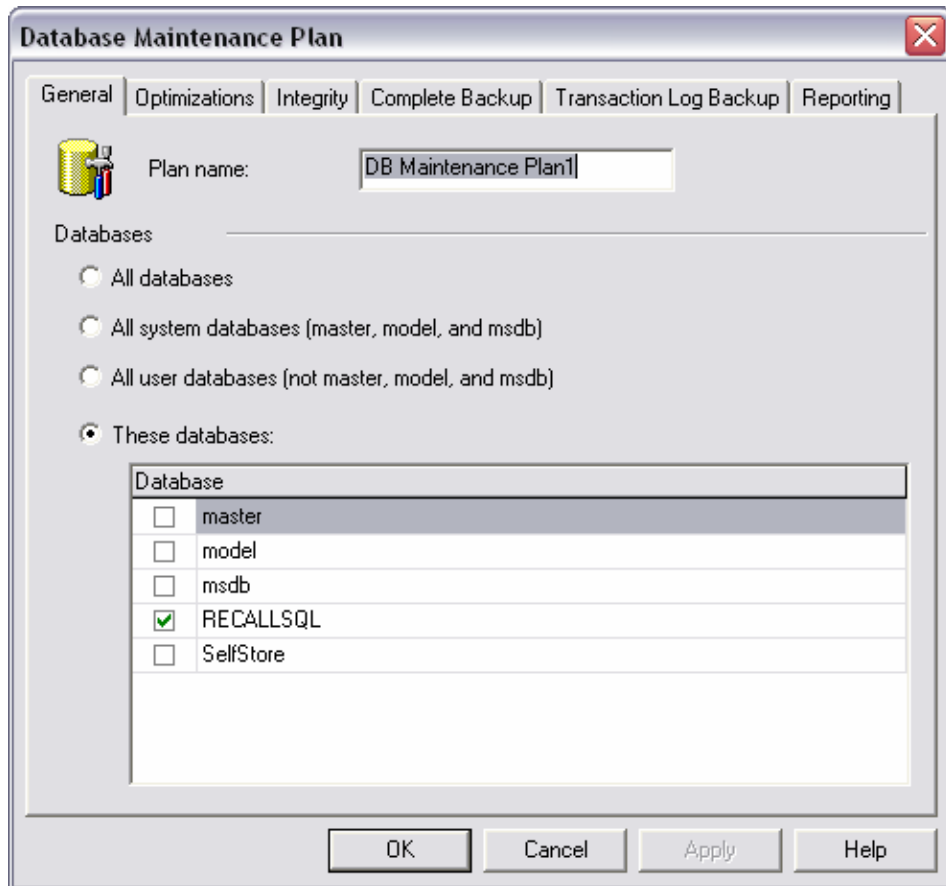
**DB
Maintenance
Plan1 Selected**



(Fig. 2.1)

By double clicking on a Maintenance Plan you can edit the options covered during the initial setup.

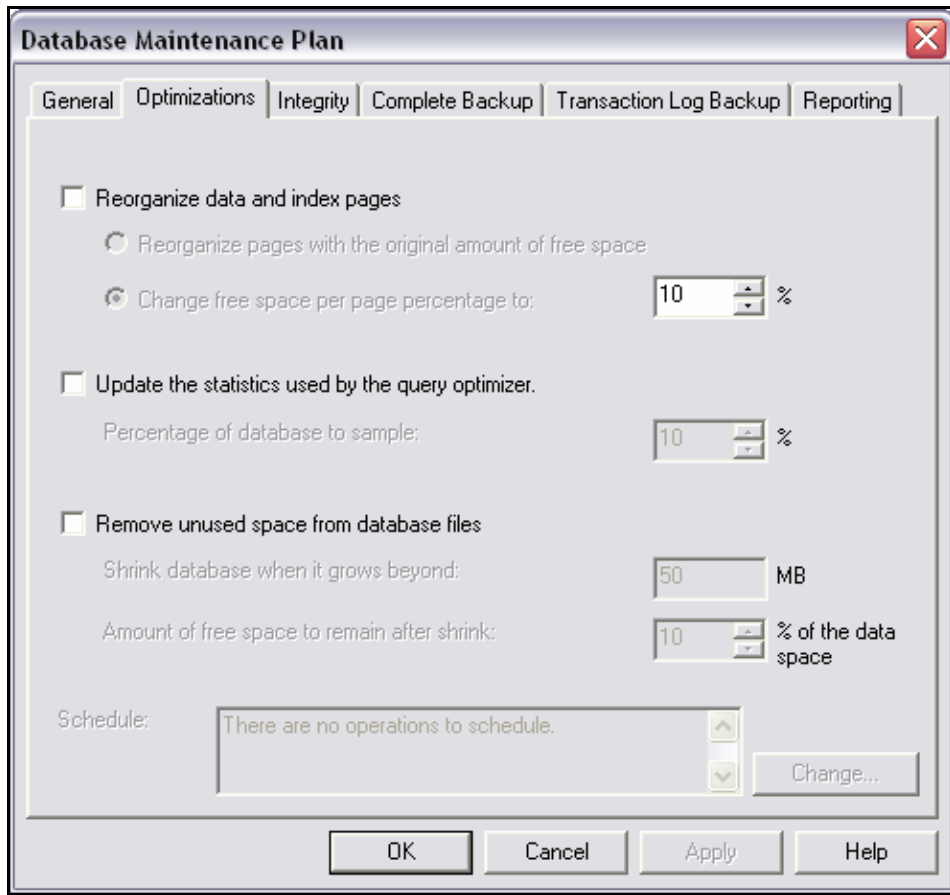
General Tab



(Fig. 2.2)

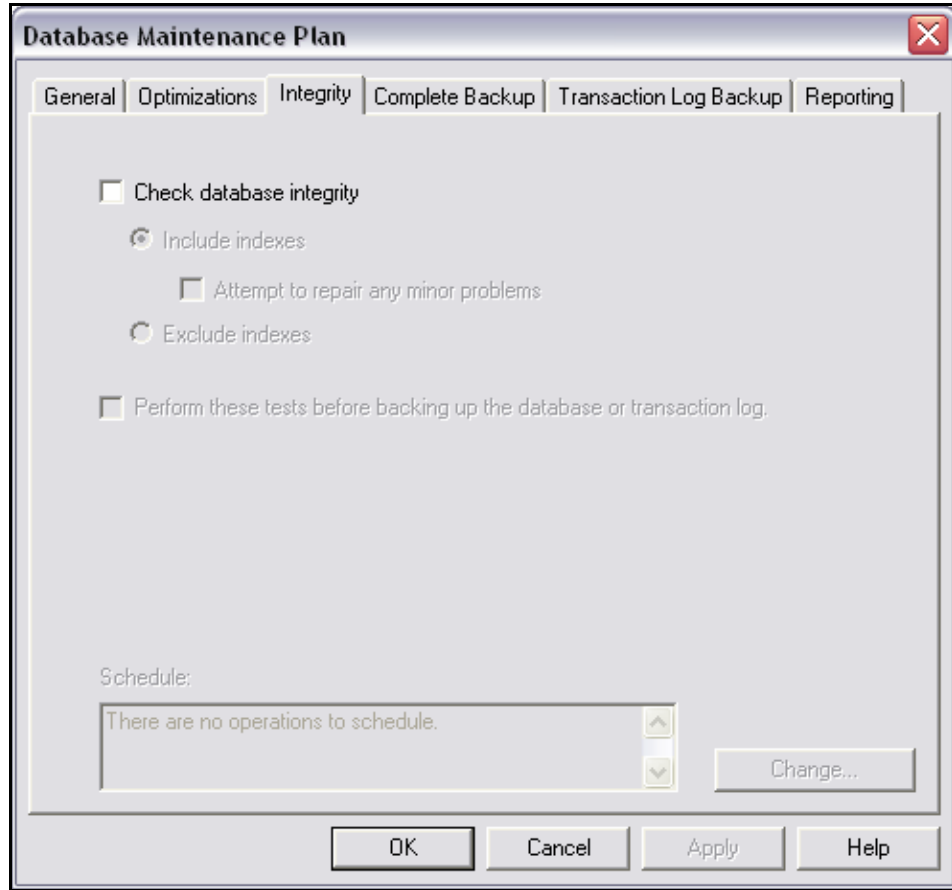
Corresponds with *figure 1.6* from the initial setup. You may also edit the Plan Name after completion here.

**Optimizations
Tab**



(Fig 2.3)

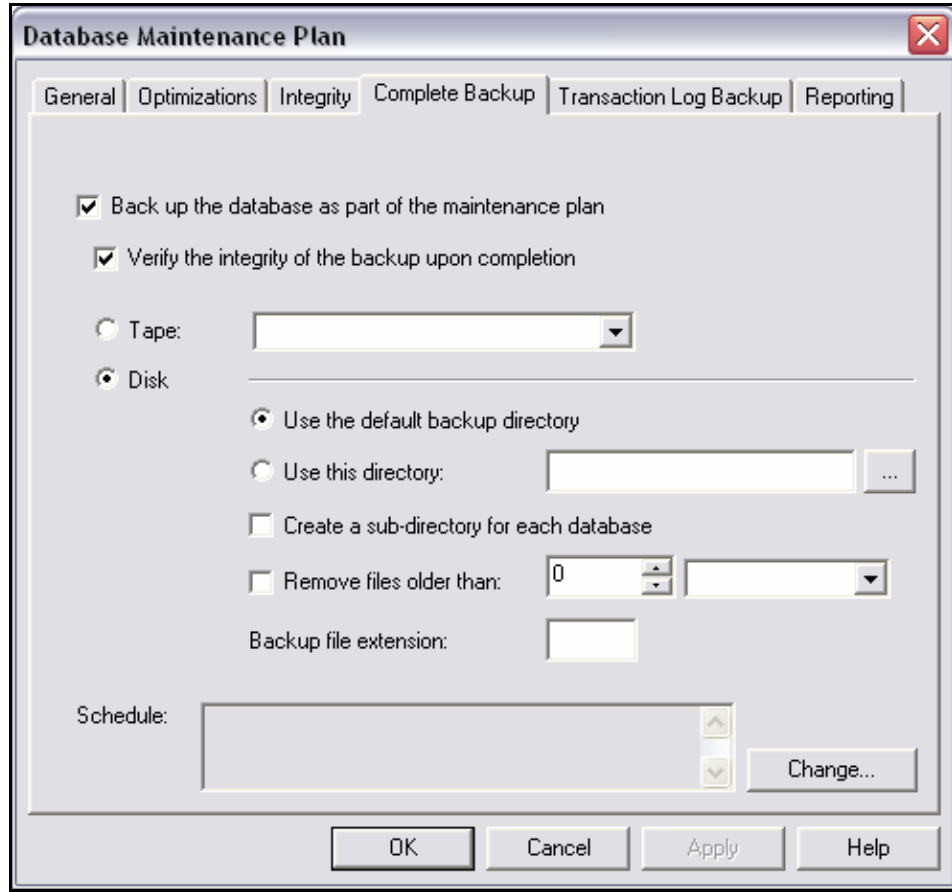
Corresponds with *figure 1.7* from the initial setup

Integrity Tab

(Fig. 2.4)

Corresponds with *figure 1.9* from the initial setup.

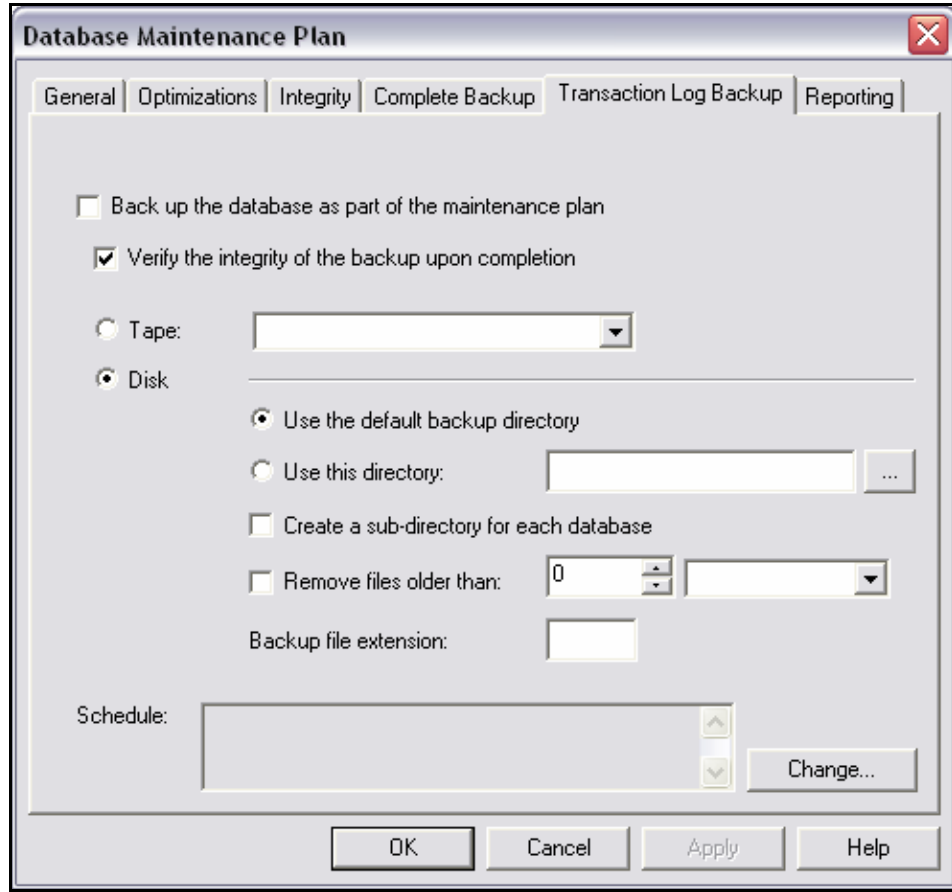
Complete Backup Tab



(Fig. 2.5)

Corresponds with *figures 1.10 and 1.11* from the initial setup.

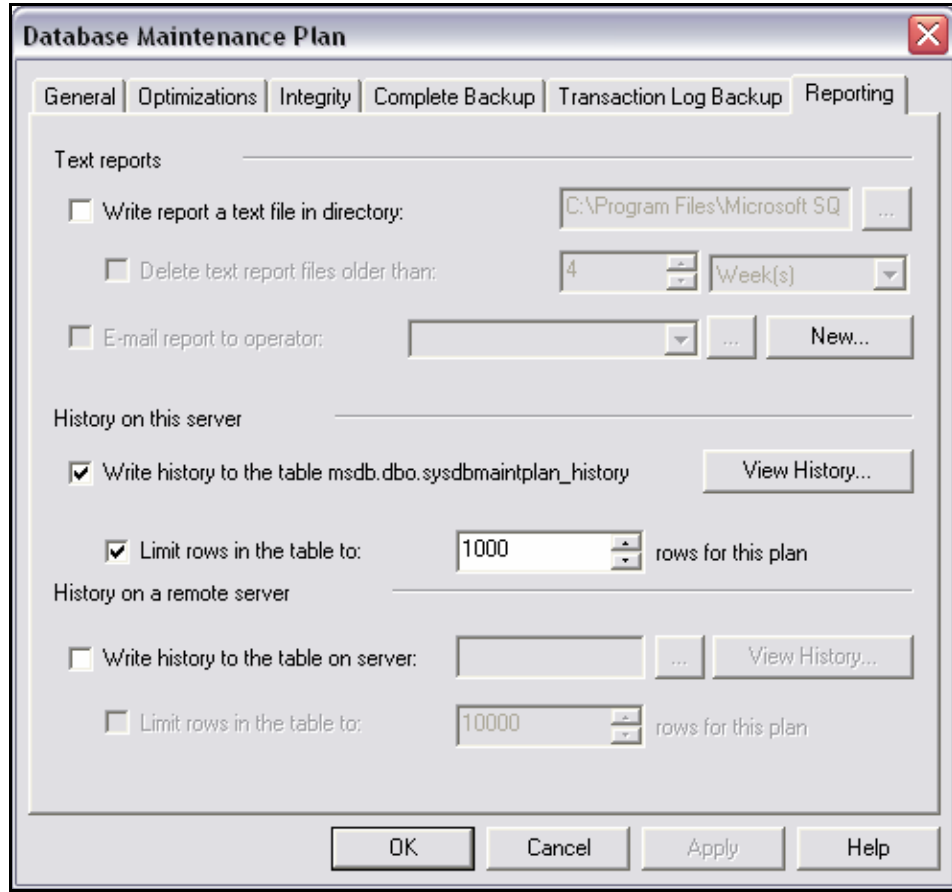
**Transaction
Log Backup
Tab**



(Fig. 2.6)

Corresponds with *figure 1.12* from the initial setup.

Reporting Tab

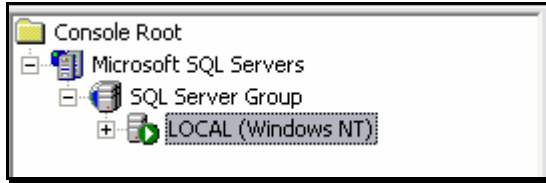


(Fig. 2.7)

Corresponds with *figures 1.13 and 1.14* from the initial setup.

Server Settings

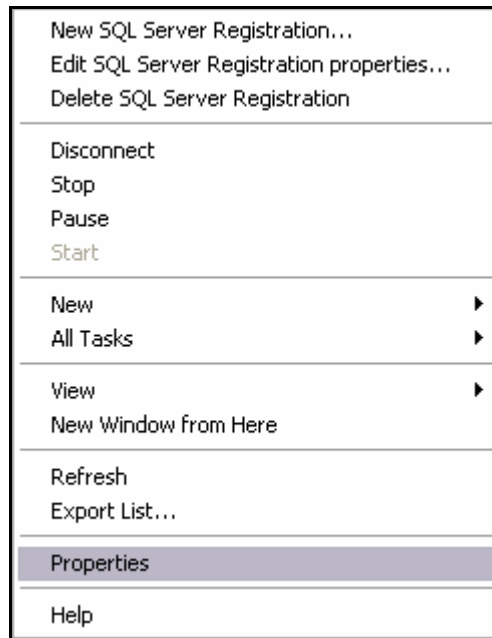
Root Menu /
Server
"LOCAL"
selected



(Fig. 3.1)

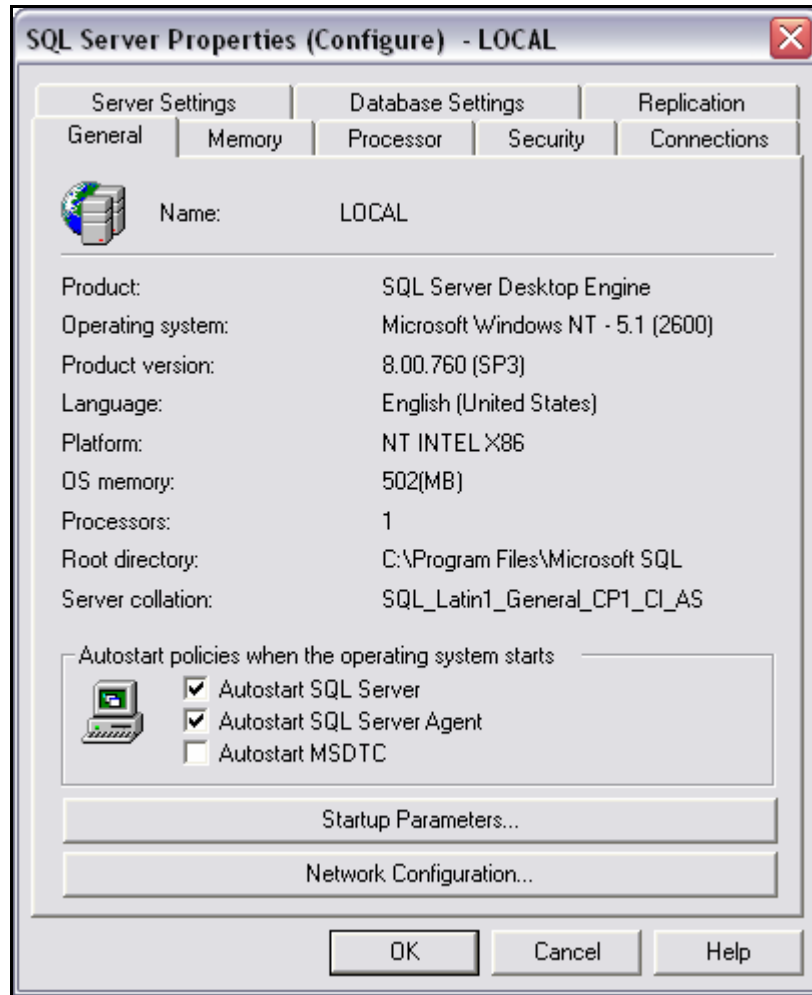
From the Root Menu right click your server to display the Server Menu in *figure 3.2*.

Server Menu



(Fig. 3.2)

Select Properties.

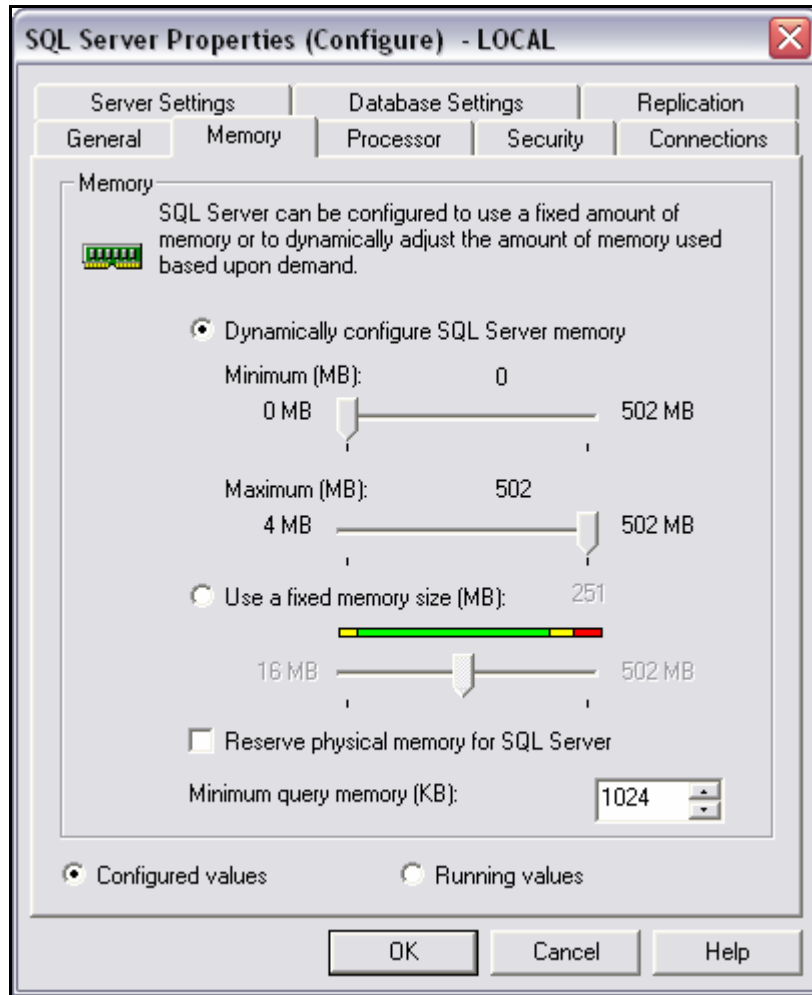
General Tab

(Fig. 3.3)

The General Tab displays your current settings and allows you to setup Autostart options for your server.

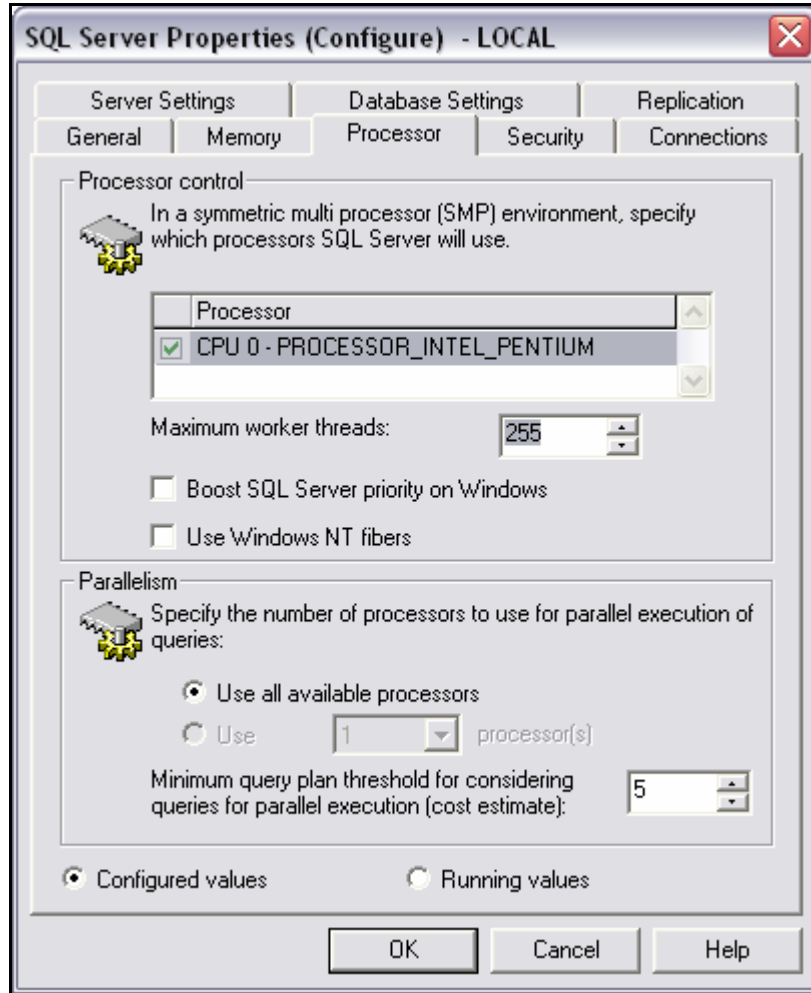
The “**Autostart SQL Server Agent**” does not default as selected. It is important that this is checked as TRSQL™ makes use of the advanced jobs system within the Server Agent to schedule recurring work orders or queries.

Memory Tab



(Fig. 3.4)

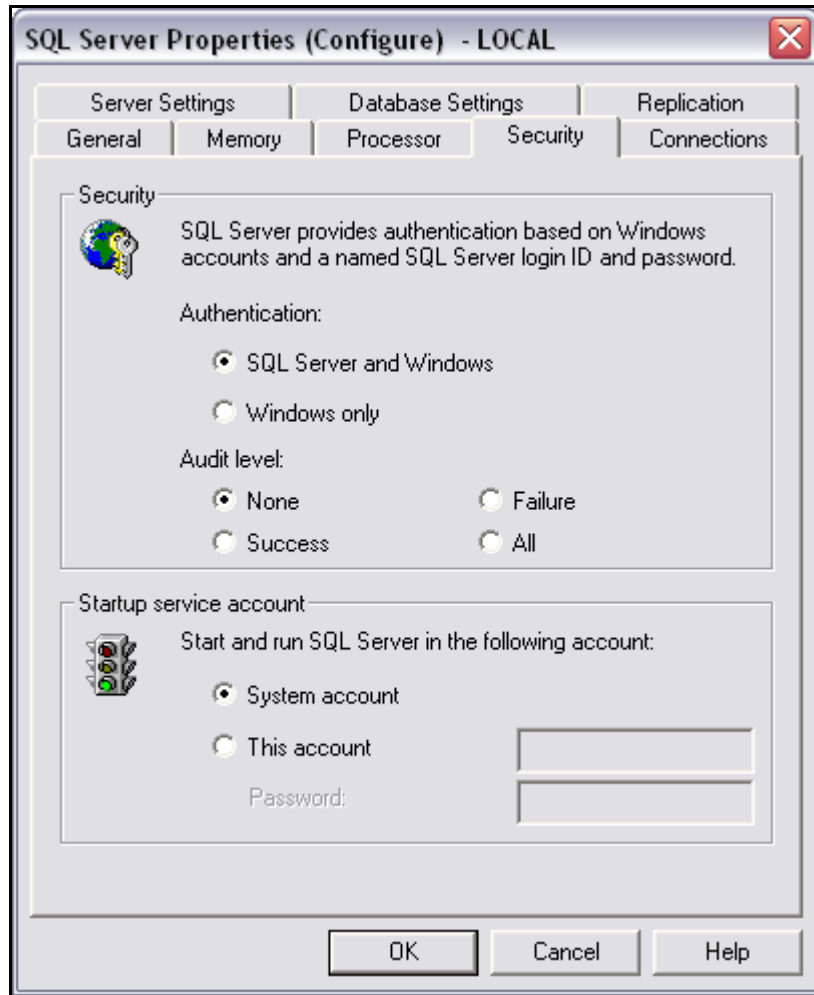
These are the server’s memory options. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

Processor Tab

(Fig. 3.5)

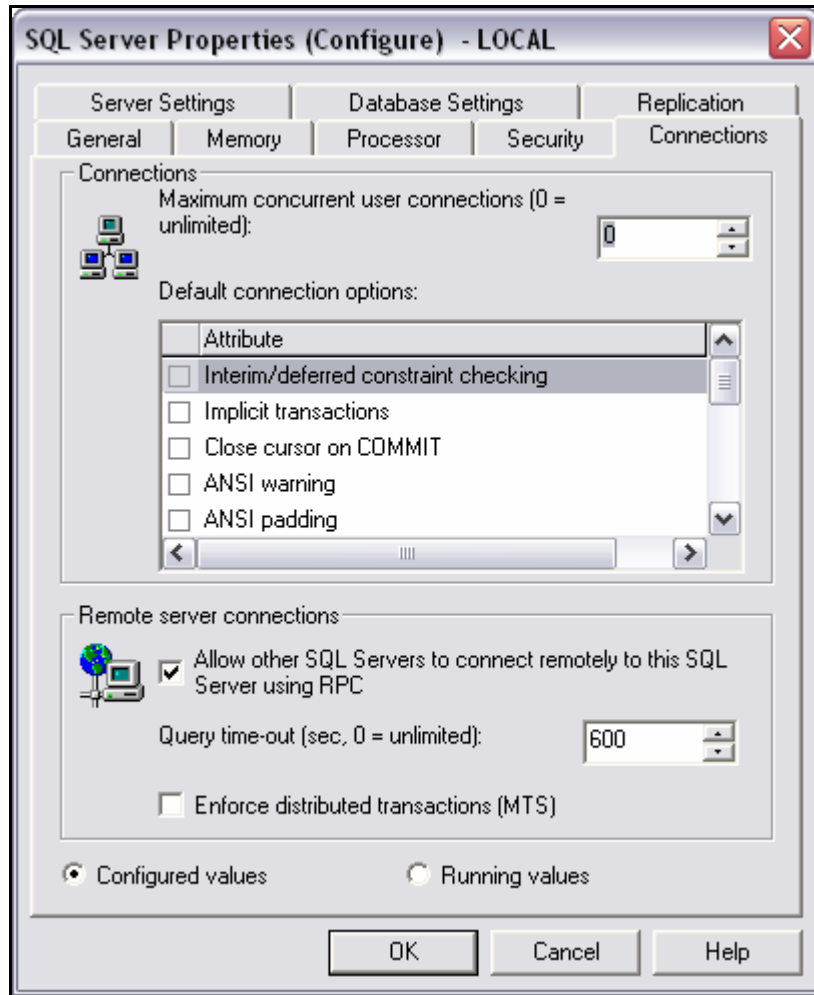
These are the Processor options for your server. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

Note: In the case of multiple processors it is recommended that you set the “Use Processors” option under Parallelism to 1. Limiting the # of processors here will prevent any one user from tying up too many of the server’s resources and causing slow downs for the rest of your users.

Security Tab*(Fig. 3.6)*

These are the security settings for your server. It is recommended that these options are setup by an advanced SQL Server technician / administrator to be configured to your needs or simply left as default.

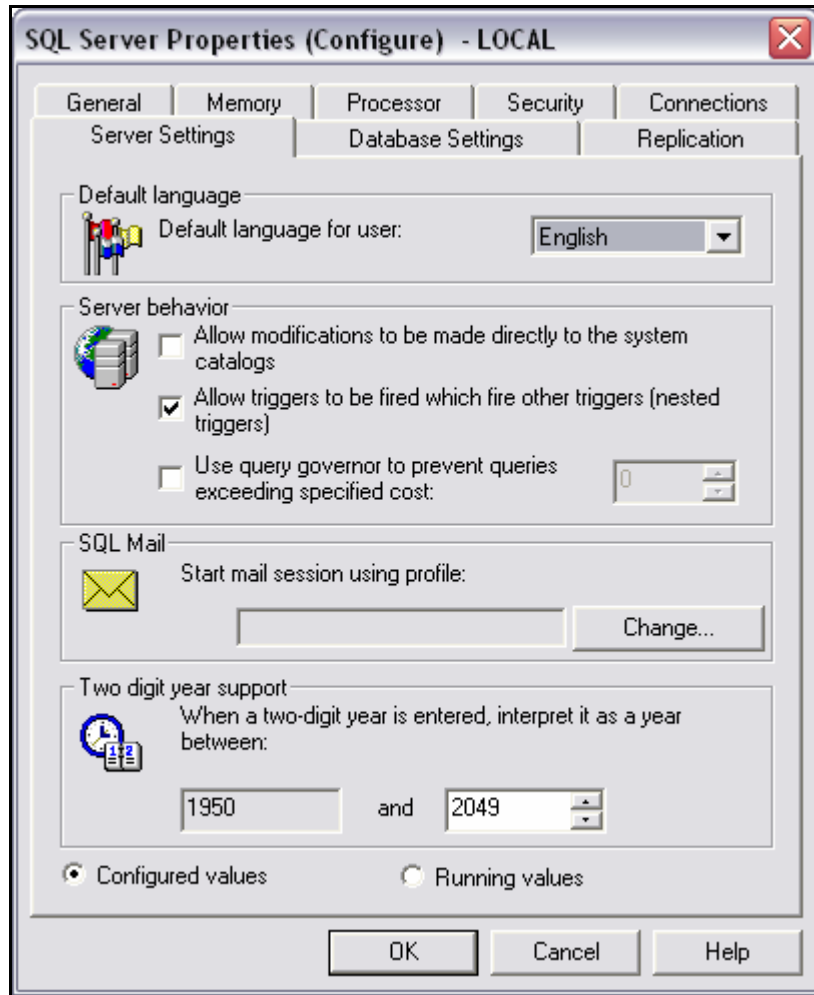
**Connections
Tab**



(Fig. 3.7)

These are the server's connection options. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

Server Settings
Tab

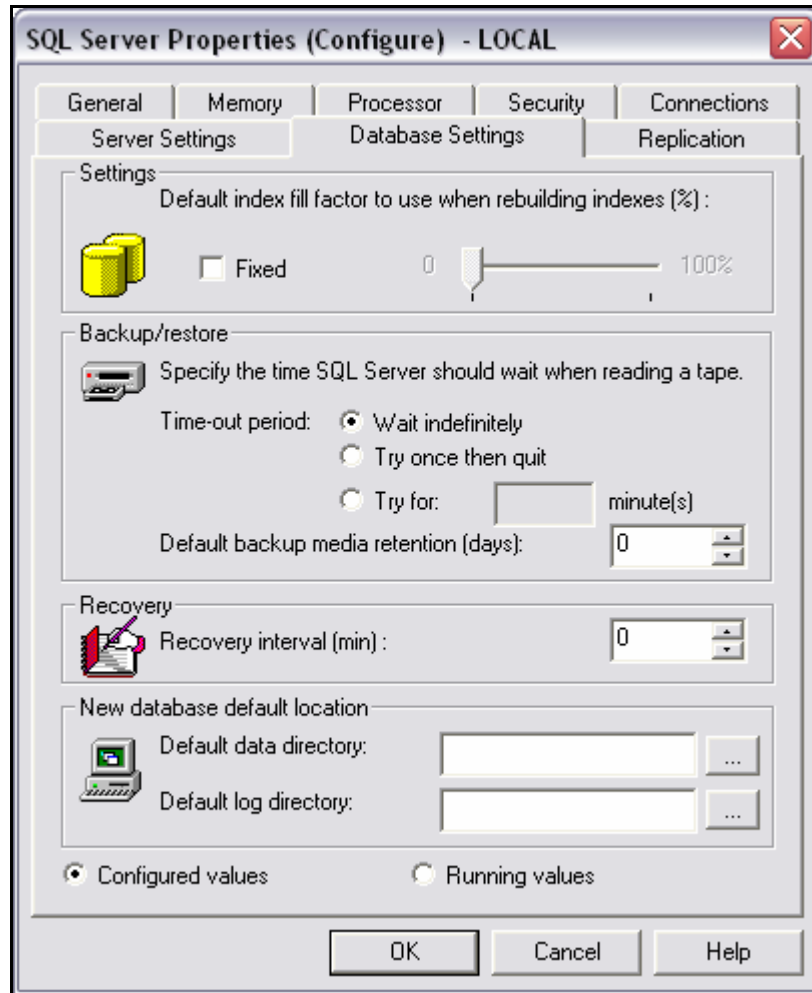


(Fig. 3.8)

These are the server's configuration options. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

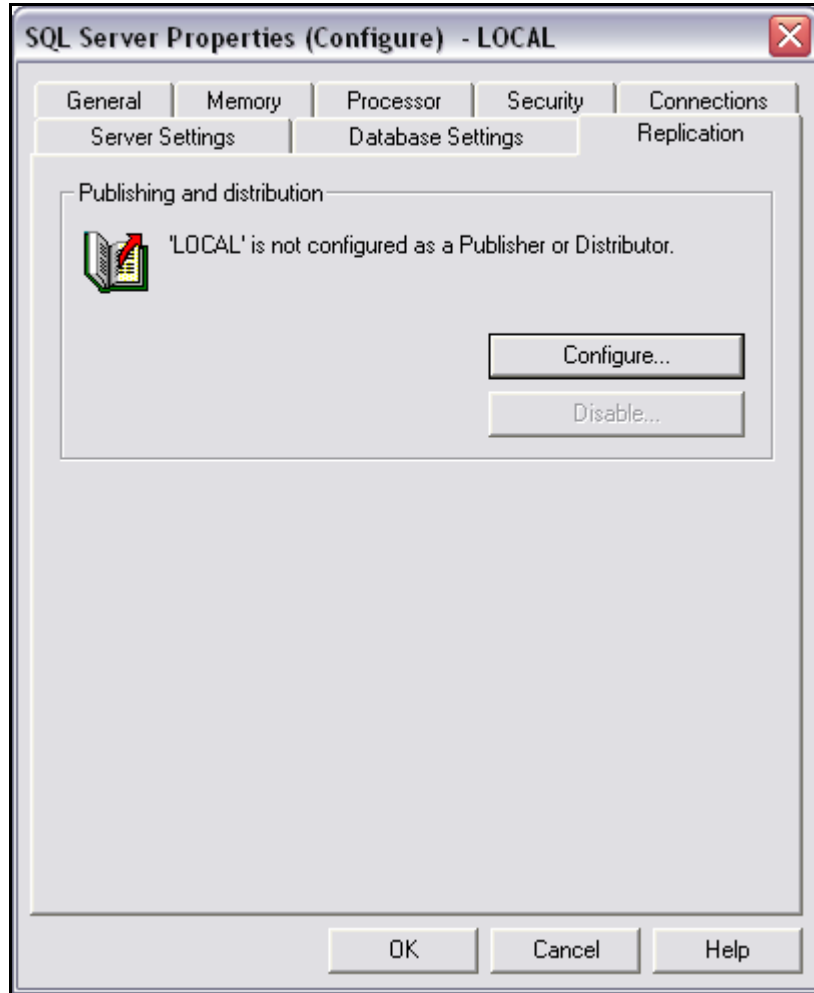
The “**Allow triggers to be fired which fire other triggers**” option is recommended to be checked as this allows TRSQL™ to perform many of its security options (Field History for example) fully.

**Database
Settings Tab**



(Fig. 3.9)

These are the server's database configuration options. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

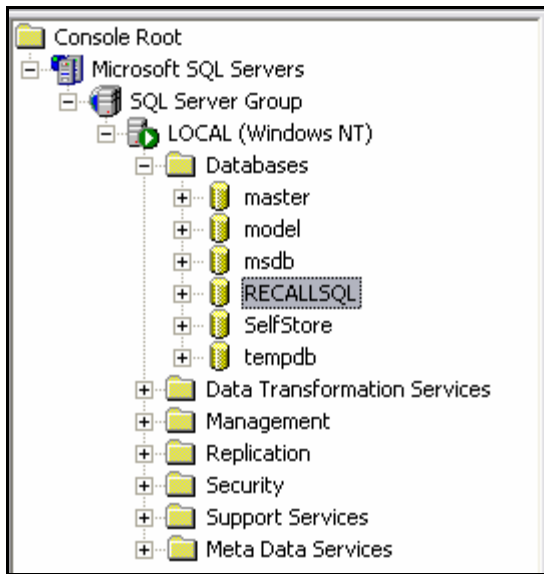
**Replication
Tab**

(Fig. 3.10)

These are the server's replication options. They may be configured by an advanced SQL Server technician to specify publishing and distribution on this server.

Database Settings

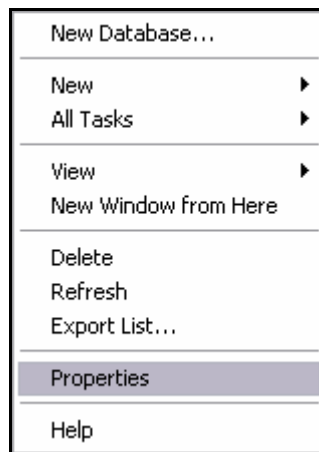
Root Menu /
Server /
Database /
RECALLSQL
selected



(Fig. 3.11)

From within the Root Menu, select your server; the Databases sub folder; and your TRSQL™ database. See *figure 3.11*. Right clicking a database will display the options shown in *figure 3.12*.

Database
Menu

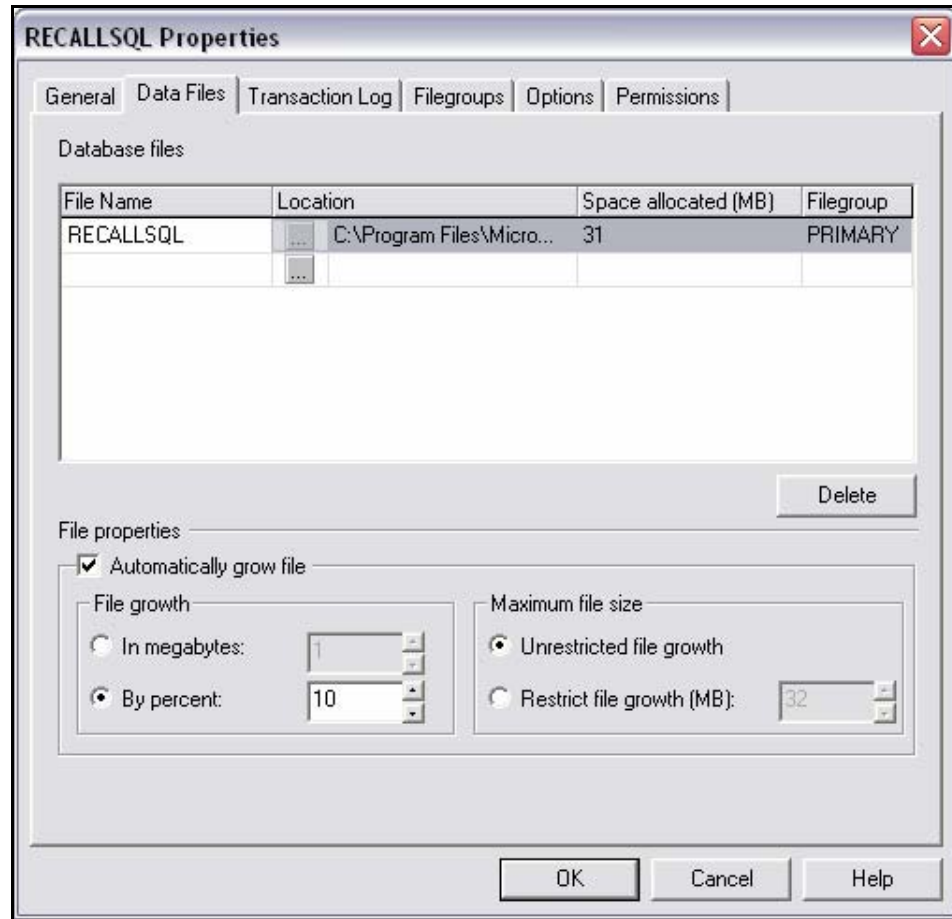


(Fig. 3.12)

Select the Properties.

General Tab*(Fig. 3.13)*

The General Tab displays your current database (including size, status, available space, and users), your backup history, and maintenance plan history.

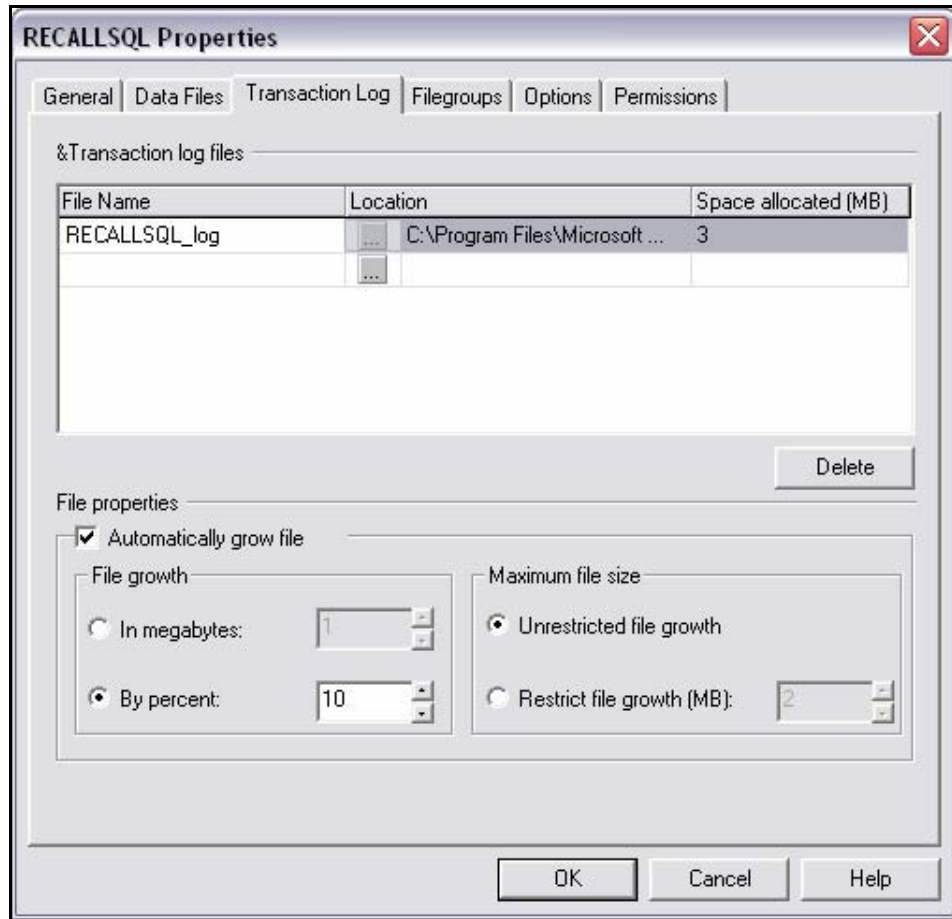
Data Files Tab

(Fig. 3.14)

The Data Files options control database growth and filegroup configurations. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

The “**Automatically grow file**” option allows the database to automatically increase in size as more space is needed. This may slow processes if the database growth is not optimized in a recurring maintenance plan (see *figure 1.7*). With proper planning and scheduled maintenance plans it is possible to prevent the database from automatically growing (slowing processes for your users), but it is recommended that the automatically grow file option is always selected in preparation for unexpected growth.

**Transaction
Log Tab**



(Fig. 3.15)

The Transaction Log options control the transaction log file configurations. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

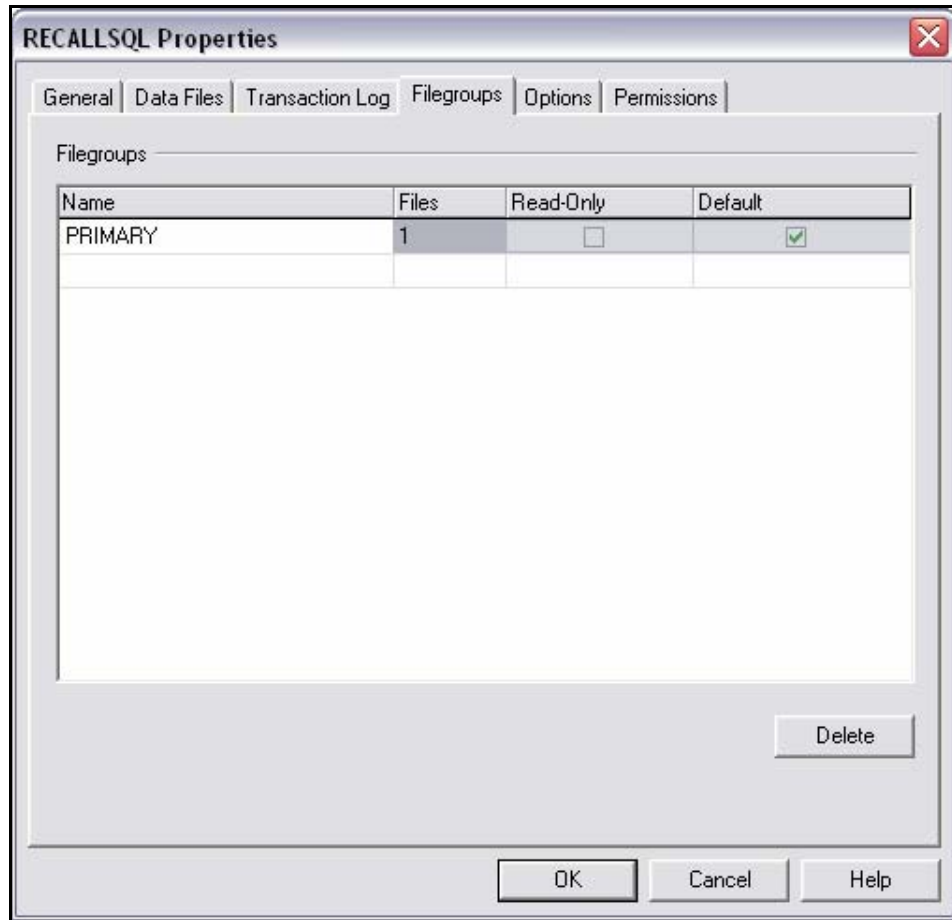
Please refer to the “Optimizing Transaction Log Performance” document (from SQL Server Books Online) on the next page for general recommendations from Microsoft for transaction log performance.

Optimizing Transaction Log Performance

General recommendations for creating transaction log files include:

- Create the transaction log on a physically separate disk or RAID (redundant array of independent disks) device. The transaction log file is written serially; therefore, using a separate, dedicated disk allows the disk heads to stay in place for the next write operation.
- Set the original size of the transaction log file to a reasonable size to prevent the file from automatically expanding as more transaction log space is needed. As the transaction log expands, a new virtual log file is created, and write operations to the transaction log wait while the transaction log is expanded. If the transaction log expands too frequently, performance can be affected.
- Set the file growth increment percentage to a reasonable size to prevent the file from growing by too small a value. If the file growth is too small compared to the number of log records being written to the transaction log, then the transaction log may need to expand constantly, affecting performance.
- Manually shrink the transaction log files rather than allowing Microsoft® SQL Server™ 2000 to shrink the files automatically. Shrinking the transaction log can affect performance on a busy system due to the movement and locking of data pages.

[©1988-2000 Microsoft Corporation. All Rights Reserved.](#)

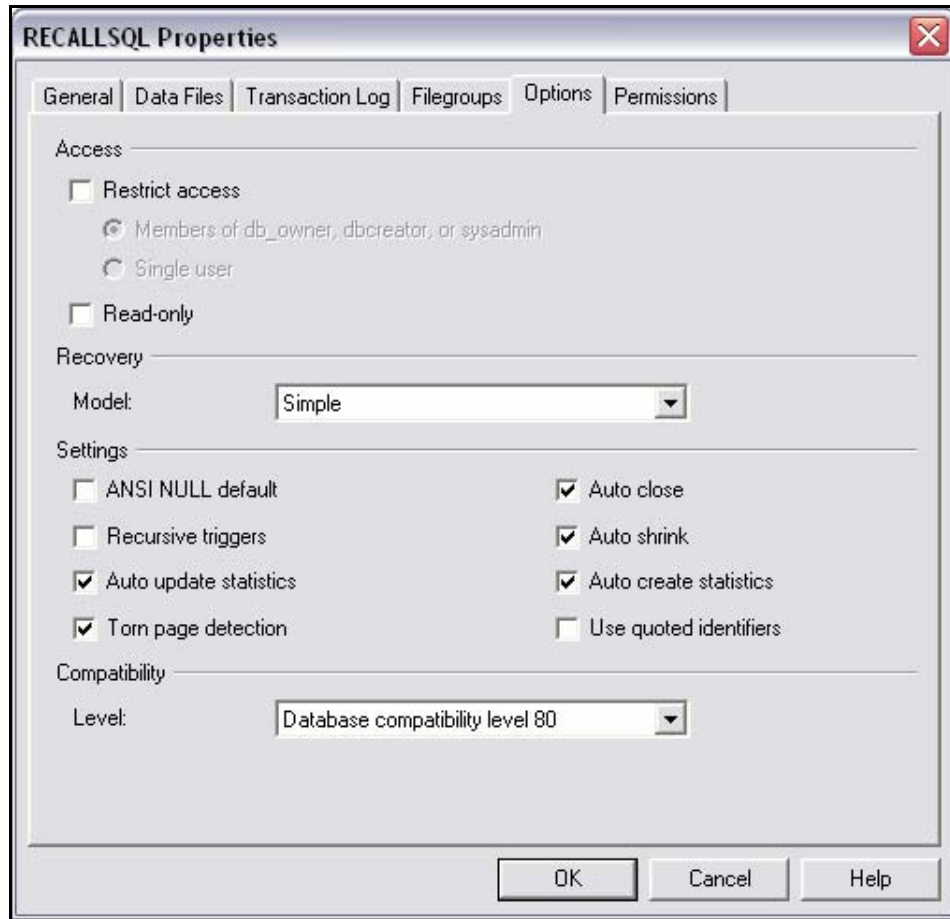
Filegroups Tab

(Fig. 3.16)

The Filegroups options control filegroup setup and maintenance. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

Note: When running on Windows NT, SQL Server performance can be improved further if the databases are created on disks formatted using NTFS and, specifically, 64-KB extent sizes. For more information about formatting an NTFS disk, see the Windows NT documentation.

Options Tab



(Fig. 3.17)

The Database Options control the access, recovery, and other settings for your database. They may be configured by an advanced SQL Server technician, but are recommended to be left as default.

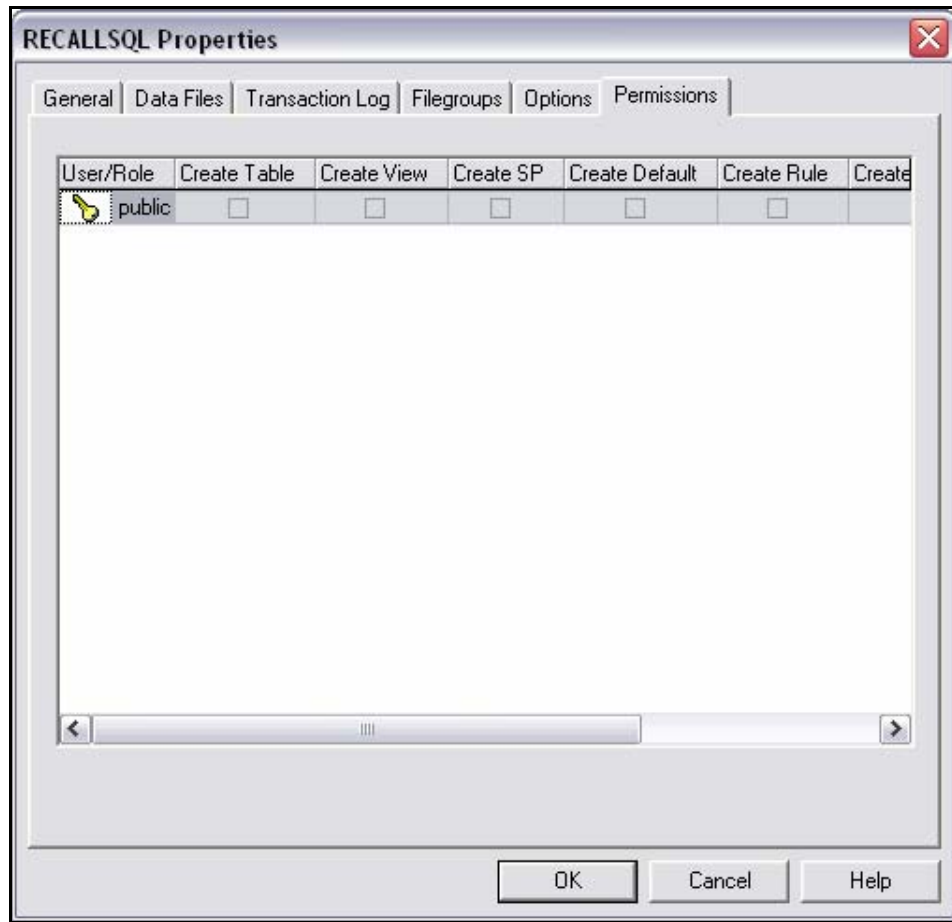
The “**Auto Shrink**” checkbox is recommended to be unchecked. This option is best controlled using the optimization from a maintenance plan. In this way the database shrinking can be controlled to have less of an impact on operations and process speed.

The “**Recovery Model**” option allows you to setup your backup / recovery procedures.

- Simple Recovery allows the database to be recovered to the most recent backup.
- Full Recovery allows the use of the Transaction Logs to recover the database to the point of failure (*this is required to properly perform transaction log backups*).
- Bulk Logged recovery allows the database to be recovered from minimal log space for large scale copy operations.

Note: See figure 1.12 for application and examples of each Recovery Model.

**Permissions
Tab**

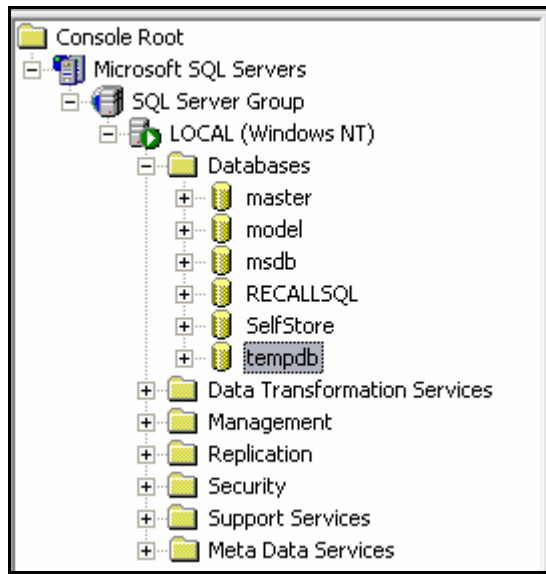


(Fig. 3.18)

The Permissions options control user's access and permissions. They may be configured by an advanced SQL Server technician to meet your security needs.

Tempdb Settings

Root Menu /
Server /
Database /
tempdb
selected



(Fig. 3.19)

From within the Root Menu, select your server; the Databases sub folder; and your TRSQL™ database. See figure 3.19. Right clicking a database will display the options shown in figure 3.20.

Database
Menu



(Fig. 3.20)

Select the Properties.

Please refer to the “Optimizing tempdb Performance” document (from SQL Server Books Online) on the next page for general recommendations from Microsoft for transaction log performance.

Optimizing tempdb Performance

General recommendations for the physical placement and database options set for the **tempdb** database include:

- Allow the **tempdb** database to automatically expand as needed. This ensures that queries that generate larger than expected intermediate result sets stored in the **tempdb** database are not terminated before execution is complete.
- Set the original size of the **tempdb** database files to a reasonable size to avoid the files from automatically expanding as more space is needed. If the **tempdb** database expands too frequently, performance can be affected.
- Set the file growth increment percentage to a reasonable size to avoid the **tempdb** database files from growing by too small a value. If the file growth is too small compared to the amount of data being written to the **tempdb** database, then **tempdb** may need to constantly expand, thereby affecting performance.
- Place the **tempdb** database on a fast I/O subsystem to ensure good performance. Stripe the **tempdb** database across multiple disks for better performance. Use filegroups to place the **tempdb** database on disks different from those used by user databases.

[©1988-2000 Microsoft Corporation. All Rights Reserved.](#)