ch07_233-258.qxd 11/28/01 4:45 PM Page 233

Chapter 7

Maintenance

Chapter at a Glance

This chapter focuses on the maintenance of your databases. This is the most important chapter in the book.

The Resources section contains references for:

- Jobs
- Tasks
- Other maintenance

Read Chapters 2, 3, and 6 before reading this chapter.

Overview

The most important and least glamorous tasks a DBA performs are those that involve maintenance. It's the kind of thing that if you do, nobody notices; but if you don't, *everybody* knows.

There are two types of maintenance tasks: tasks that keep the database going and tasks that keep the database going *fast*. I explain both types of tasks in this chapter, with a heavier focus on the first. The fast part I examine more closely in Chapter 9.

If you're already a DBA using another platform, you'll be pleased with how easy it is to maintain a database in SQL Server 2000. In fact, it's one of the main selling points for the product. Wizards are provided for almost everything you need to keep the database in shape.

The primary tool I'll describe for database maintenance is the Database Maintenance Wizard. This graphical tool walks you through setting up a scheduled plan that can include integrity checks, optimizations, and backups. These plans can be edited at any time.

The second set of tools are the T-SQL Database Consistency Check (DBCC) commands. They perform maintenance tasks on a database or an index, as well as on filegroups. There are four categories of DBCC commands: maintenance, miscellaneous, status, and validation statements. This chapter focuses on the maintenance commands.

Detail

You must understand the physical structure of the database engine to understand why you need to perform maintenance.

Physical Structure

Data is stored in files on the hard drive. Because the data is constantly modified, SQL Server 2000 sets aside an entire area of the hard drive to work with. Within this space, the engine places data in smaller sections called pages, which are 8K long.

As an illustration, you can act as a database engine. Think of a piece of paper eight lines long. You write information on the piece of paper as you take orders. Some orders are small, only a few lines long; and others are longer, perhaps several pages. After you enter an order, it can be modified.

Let's assume you've entered order number 1 on page 1, and it's two lines long, as shown in Table 7.1.

Next you enter order number 2, which is seven lines long. Looking at your order sheets, you now have two pages of information. On page 1 you have two lines from one order and six lines from order 2. The second page contains one line from the second order, as shown in Table 7.2.

Table 7.1 First Order, First Page

Order 1	Item 1	Client 1
Order 1	Item 2	Client 1

Table 7.2 Second Order, Second Page

Order 1	**Item 1**	**Client 1**
Order 1	**Item 2**	**Client 1**
Order 2	Item 1	Client 2
Order 2	Item 2	Client 2
Order 2	Item 3	Client 2
Order 2	Item 4	Client 2
Order 2	Item 5	Client 2
Order 2	Item 6	Client 2
Order 2	Item 7	Client 2

The client from order 1 calls back and adds two more items to her order. You dutifully record the items on the second page, where you have room, as shown in Table 7.3. RDBMS systems really work like this, and I'll use this description throughout the rest of the chapter.

Making a database fast is often a function not only of maintenance but also of design. Because you probably can't redesign your database, I'll focus on maintaining the internal structures of SQL Server 2000 to optimize the database.

The two primary tools for maintaining your databases are the *Maintenance Wizard* and the DBCC commands.

Table 7.3 First Order, Second Page

Order 1	**Item 1**	**Client 1**
Order 1	**Item 2**	**Client 1**
Order 2	Item 1	Client 2
Order 2	Item 2	Client 2
Order 2	Item 3	Client 2
Order 2	Item 4	Client 2
Order 2	Item 5	Client 2
Order 2	Item 6	Client 2
Order 2	Item 7	Client 2
Order 1	**Item 3**	**Client 1**
Order 1	**Item 4**	**Client 1**

The Maintenance Wizard

I've taught several classes on SQL Server, and I always tell my students about the SQL Server Maintenance Wizard. I've visited some of these same students' offices later, and in a few I've been shocked to see that even the simple step of setting up a maintenance plan using a wizard has been ignored. Invariably, after the wizard has been set up and run, the users see a dramatic difference in the speed of database access.

The Maintenance Wizard creates a group of Jobs and Schedules called a plan. This plan can be edited after you create it. You should set up one plan per database—even though you can choose more than one database in a single plan. If you create the plans separately, each of them can have separate schedules and tasks.

You should also create a plan for the system databases, especially master and msdb. You may recall from earlier chapters that the information in these databases involves logins and server configuration information (master) and the SQL Server Agent information (msdb). You should protect this information, and the wizard provides an easy way to do that.

The Maintenance Wizard automatically performs tasks in the following areas:

- Optimizations
- Integrity
- Backups
- Reporting

Let's look at the optimizations step first.

Optimizations

As data is entered into and deleted from the database, the defined space mentioned earlier becomes a bit fragmented, much like the files on your hard drive. The optimizations portion of the wizard examines three things in your database that affect this storage. The first item the wizard examines and changes is fragmentation. You also can select how much free space for new data will be left when the wizard is done defragmenting. How much space should you choose, and why? The explanation goes deeply into whether your database is data-entry-more-often versus read-more-often.

If your database is used primarily for data entry, it's best to leave a bit more space at the end of each page so that data stays together. This largely prevents the mixed data pages displayed in Table 7.3 earlier in the chapter.

If your database is composed of data that's read more often than it's written, such as a report server, leave a lower percentage of the page space free. In this case fewer pages have to be read into memory, thus saving time.

Both of these recommendations are a bit broad, but there's a good treatment on this topic in Books Online under DBCC SHOWCONTIG that explains how to monitor your database to determine a good fill factor. You can usually take the default on most of the questions asked when using the wizard, if you're not sure which value to select.

The second item the optimizations step performs for you is to update database statistics. Statistics are samples of data that assist the SQL Server 2000 search engine in locating data quickly. As data is added and deleted, these statistics become outdated.

You can keep statistics updated in two ways: by selecting this item while using the wizard or by setting a database option that automatically updates them. I explained this option in Chapter 4.

You pay a small performance penalty when the server takes time to update the statistics on the fly, and some programs have a problem with it. See the owner's manual for your software to see if it is mentioned. As a rule, I normally leave this option on for the database; and the wizard grays that item out, indicating that it's not necessary to check the statistics.

The last thing the optimizations step does is to remove any free space from the database. Recall that SQL Server 2000 allocates space in an operating system file and then uses space within that file. When data is deleted, space in the file isn't automatically reclaimed. This setting in the wizard handles shrinking the database at a specific recurring time. A database option can be set to autoshrink the database. The same caveats from earlier apply.

The next step in the wizard checks the integrity of the database.

Database Integrity

For the most part, SQL Server 2000 is one of the most stable RDBMS platforms you'll ever encounter. It's designed to recover from just about anything all by itself. Even so, it's always prudent to check the integrity of a database from time to time. This step offers to correct any minor errors for you, which means that only checks that don't change data are performed. This is normally a safe option to set.

The next step involves backing up the database and potentially the logs.

Database Backups

You can back up your database or logs to tape or hard drive. The hard drive location can be broken into different directories for each database, and the wizard deletes backups from the hard drive that are older than a certain age.

The final step in the wizard involves recording all these activities and sending Notifications. The reports can be sent to disk, to a central server, to a history table, and to an operator, or all the above.

In a word, the Maintenance Wizard performs all the maintenance most systems need.

DBCC Commands

The next set of tools at your disposal are the DBCC commands, which are typed at the command line. As a matter of fact, the Maintenance Wizard uses many of these commands to do its magic. There are several maintenance commands, many of which can fix issues in addition to reporting them.

The DBCC commands I'll discuss are:

- `DBCC CHECKDB`
- `DBCC CHECKTABLE`
- `DBCC DBREINDEX`

The `DBCC CHECKDB` command examines an entire database for corruption. I'll describe the switches used with this command to indicate errors and perform various levels of repair.

DBCC CHECKDB

`DBCC CHECKDB` is the primary command that Microsoft recommends because it checks all the tables and indexes in a database. Although you can run this command in a diagnostic mode while people are using the system, the database must be set to single-user mode to correct any problems. I do that in the Examples section.

DBCC CHECKTABLE

The `DBCC CHECKTABLE` command does the same thing that `DBCC CHECKDB` does except against just one table. You can also use this command while people are using the system unless you're using the repair functions.

DBCC DBREINDEX

The `DBCC DBREINDEX` command updates the indexes on a server. Indexes point to the location of data, which speeds things up. Think of looking for the subject "backups" in this book without using the index in the back. How quickly could you find all the references to that subject?

Indexes are just as important in a database. They point to locations of data, and they are stored just like data in pages. These pages become fragmented over time, and the `DBCC DBREINDEX` command defragments them.

You should be aware that this defragmentation process can be quite long, depending on the number of indexes, the size of the index, and the load on the server at the time the command runs.

Graphical

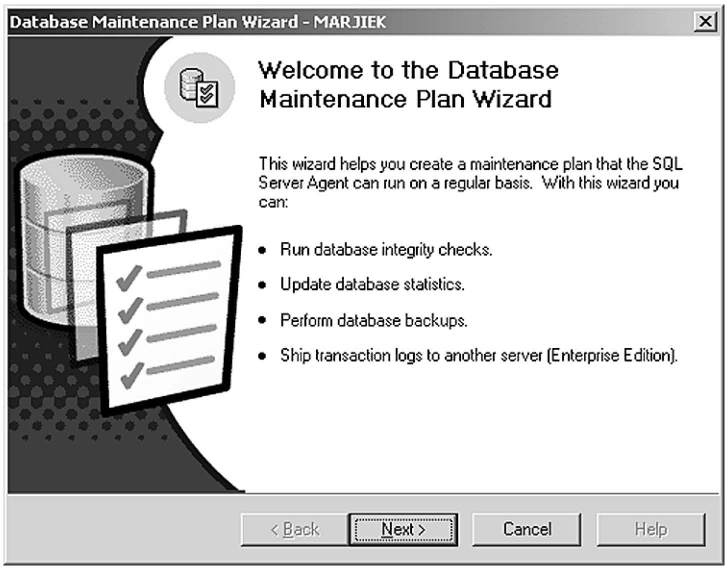
The Maintenance Wizard is the primary tool for maintaining your databases. It is so simple and takes so little time to implement that there is no excuse for not performing the steps on each of your databases.

The Maintenance Wizard

We examine this tool by setting up a live maintenance plan that you can leave in place on your test database. I have to steal a little thunder from the Examples section to describe the process.

Begin by opening Enterprise Manager, opening your server's name, and then drilling down to Maintenance, Database Maintenance Plans. Right-click that object and select New Maintenance Plan from the menu that appears. You'll see the screen shown in Figure 7.1. The introduction screen tells you what the wizard can do. Select Next, which brings up the screen in Figure 7.2.

Select your test database, but also notice the other choices. You have the option for the wizard to perform the same tasks on all databases. I don't like this choice because I often need different options per database, and a reported failure in one part of the wizard might be hard to spot. Notice as well that you can select just the system databases or all databases except the system databases.

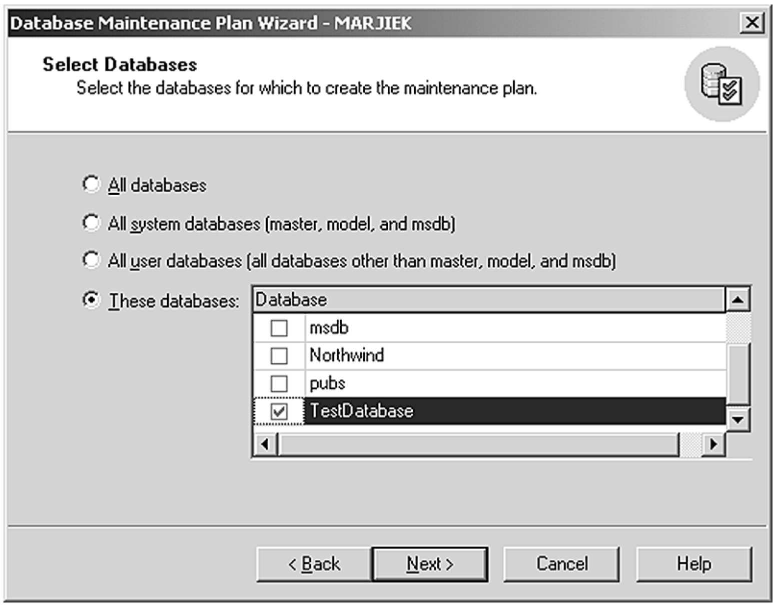

Database Maintenance Plan Wizard - MARJIEK

Welcome to the Database Maintenance Plan Wizard

This wizard helps you create a maintenance plan that the SQL Server Agent can run on a regular basis. With this wizard you can:

- Run database integrity checks.
- Update database statistics.
- Perform database backups.
- Ship transaction logs to another server (Enterprise Edition).

Figure 7.1 Creating a new maintenance plan


Database Maintenance Plan Wizard - MARJIEK

Select Databases
Select the databases for which to create the maintenance plan.

- ○ All databases
- ○ All system databases (master, model, and msdb)
- ○ All user databases (all databases other than master, model, and msdb)
- ● These databases:

	Database
☐	msdb
☐	Northwind
☐	pubs
☑	TestDatabase

Figure 7.2 Selecting the databases

As I mentioned earlier, you should make one plan per database, although you should create a plan for at least the two system databases master and msdb.

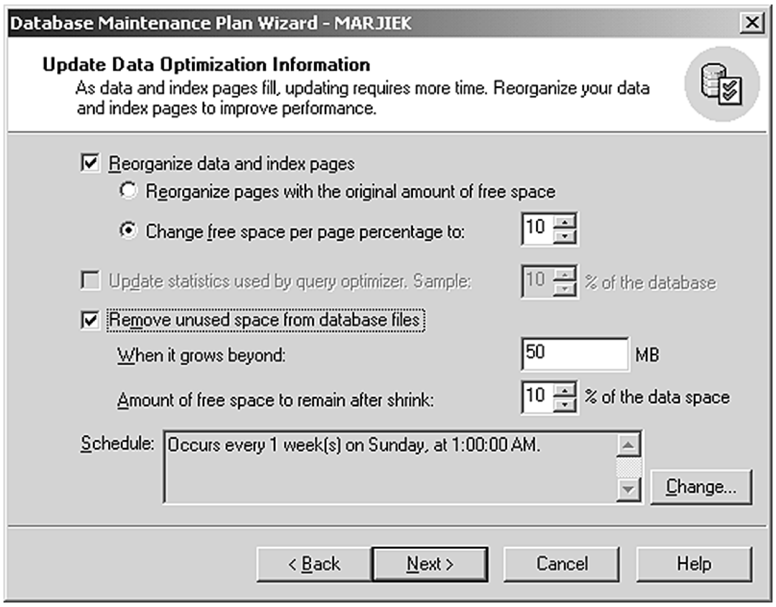
Select the test database you created. Then select Next to bring up the screen shown in Figure 7.3.

This panel sets the optimizations tasks. Notice that I've taken the defaults here, but you should change the schedule for this task. To do that, select the Change button to bring up the panel shown in Figure 7.4. In this panel you can see that the schedule is changed from a weekly recurrence to a nightly one. You shouldn't necessarily copy me in this choice; this part requires a little planning.

The issue involves the time it takes to perform the optimizations, as well as any other impacts, such as when users log in. On a small database, this isn't a big deal; but on a larger one, you need to monitor the time each step takes.

The other issue is how long the task takes to run. The logic goes like this: If the task runs once a week, it has a lot to do, and it takes a long time. If it runs more often, it doesn't have as much to do, and it doesn't take as long.

My suggestion is to set your schedule to your offpeak time and change it to offpeak days if the users are impacted. Make the changes and select OK to bring back the previous panel; then select Next to bring up the panel shown in Figure 7.5.


Database Maintenance Plan Wizard - MARJIEK

Update Data Optimization Information

As data and index pages fill, updating requires more time. Reorganize your data and index pages to improve performance.

☑ Reorganize data and index pages

○ Reorganize pages with the original amount of free space

● Change free space per page percentage to: 10

☐ Update statistics used by query optimizer. Sample: 10 % of the database

☑ Remove unused space from database files

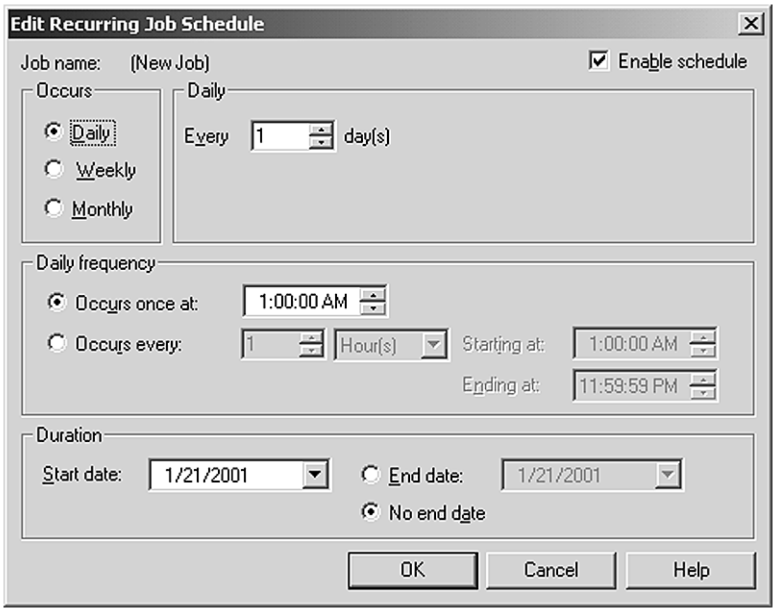
When it grows beyond: 50 MB

Amount of free space to remain after shrink: 10 % of the data space

Schedule: Occurs every 1 week(s) on Sunday, at 1:00:00 AM.

Figure 7.3
Optimization options

ch07_233-258.qxd 11/28/01 4:45 PM Page 242



Edit Recurring Job Schedule

Job name: (New Job) ☑ Enable schedule

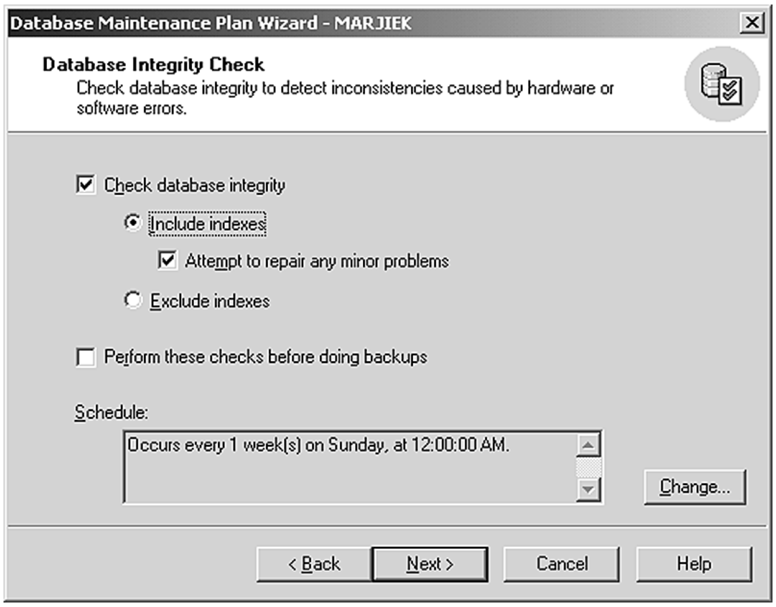
Occurs: [✓ Daily] Weekly Monthly

Daily: Every 1 day(s)

Daily frequency:
[✓ Occurs once at: 1:00:00 AM]
Occurs every: 1 Hour(s) Starting at: 1:00:00 AM Ending at: 11:59:59 PM

Duration:
Start date: 1/21/2001
End date: 1/21/2001 [✓ No end date]

Figure 7.4 Changing the schedule



Database Maintenance Plan Wizard - MARJIEK

Database Integrity Check
Check database integrity to detect inconsistencies caused by hardware or software errors.

☑ Check database integrity
[✓ Include indexes]
☑ Attempt to repair any minor problems
Exclude indexes

☐ Perform these checks before doing backups

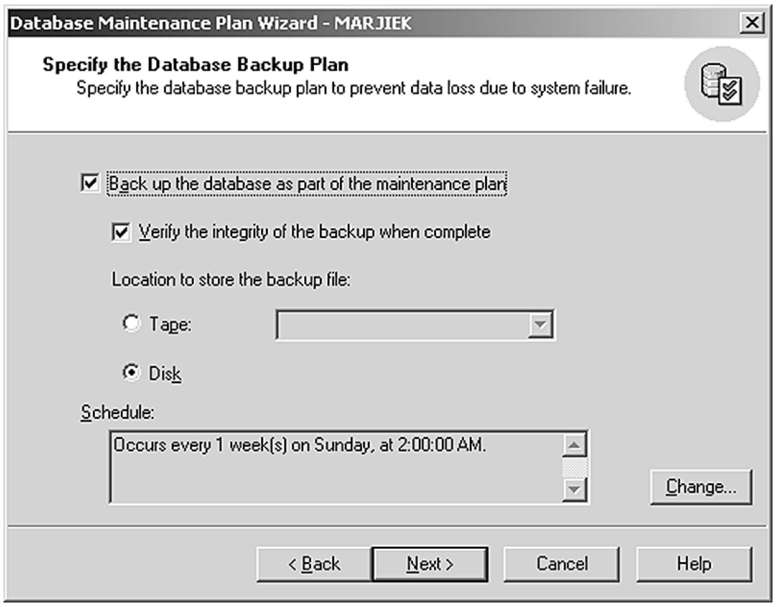
Schedule:
Occurs every 1 week(s) on Sunday, at 12:00:00 AM.

Figure 7.5 Database integrity checks

Here you select the integrity checks to run, which includes checking the integrity of indexes. You should tell the wizard to fix any minor problems it finds. Set this process to happen after the backups take place. This preserves a "pre-fixed" copy of the database in case the problem-correcting causes a problem. Change the schedule to happen each night rather than weekly. Now select OK and return to this panel; then select Next to move to the screen shown in Figure 7.6.

Here you select the backups to be placed on disk and that they will be verified when complete. I explain backup strategies fully in Chapter 8. This verification process just rereads the backup file and makes sure that the file is viable. This part takes a bit more time. Change the schedule to nightly, which brings you back to this panel. Select Next to bring up the panel that is shown in Figure 7.7.

Select the `C:\TEMP` directory for the backups and set the backups to be erased after one day. Why do this? If you can afford the space, it's nice to have the backup on a drive in case you need it quickly. Many sites send their tapes offsite each day for protection, so it takes time to get them back when needed. Leaving the file on the hard drive allows you to restore the database without


Database Maintenance Plan Wizard - MARJIEK

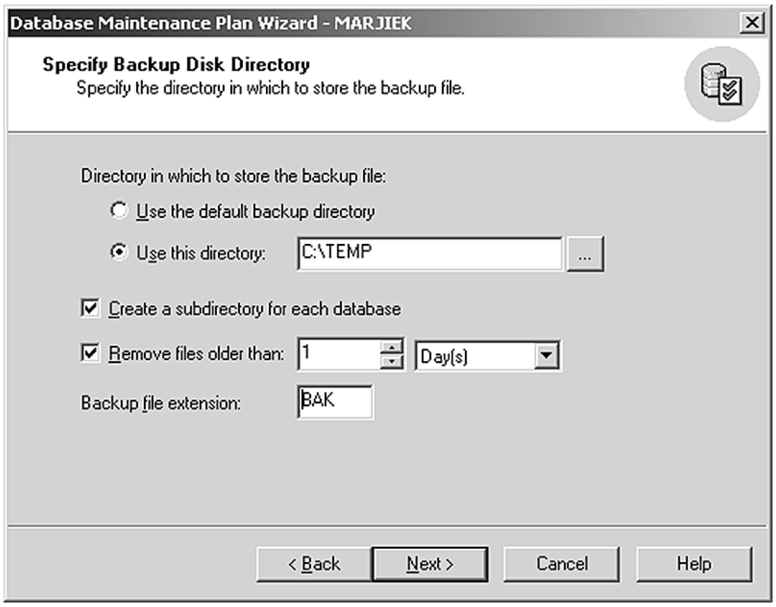
Specify the Database Backup Plan
Specify the database backup plan to prevent data loss due to system failure.

[✓] Back up the database as part of the maintenance plan
[✓] Verify the integrity of the backup when complete

Location to store the backup file:
○ Tape:
● Disk

Schedule:
Occurs every 1 week(s) on Sunday, at 2:00:00 AM.

Figure 7.6
Database backup


Database Maintenance Plan Wizard - MARJIEK

Specify Backup Disk Directory
Specify the directory in which to store the backup file.

Directory in which to store the backup file:
- ○ Use the default backup directory
- ● Use this directory: C:\TEMP

☑ Create a subdirectory for each database

☑ Remove files older than: 1 Day(s)

Backup file extension: BAK

Figure 7.7 Backup location

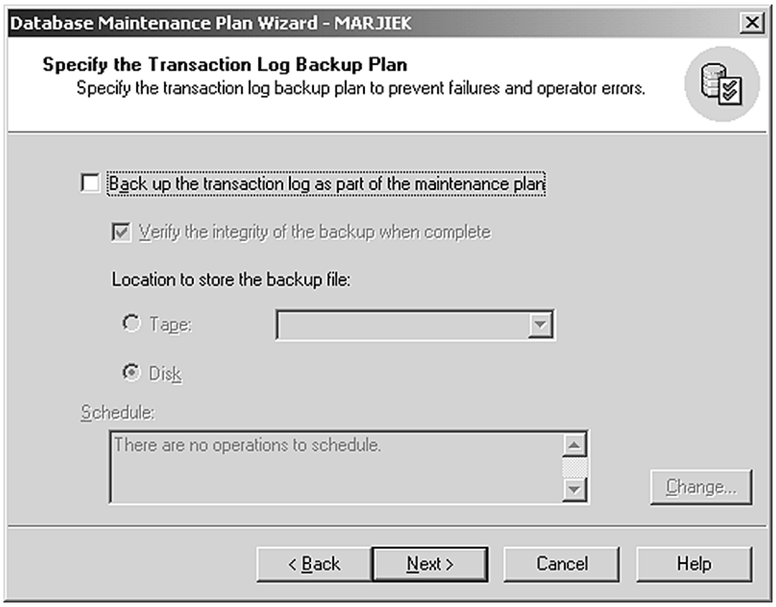
the tape. Notice also that you should select that the backups be placed in their own directories. This keeps things nice and tidy.

Select Next to bring up the panel shown in Figure 7.8. In this panel you can select the backup for the logs. Because your database is set to the Simple recovery model, you can't back up the logs; so changes are needed here. For more information on this setting, see Chapter 3.

Select Next to bring up the display shown in Figure 7.9. Here you select text reports, and you should have those deleted daily. You can have those reports mailed to you every day, because you set up that Mail profile in Chapter 2 and an Operator in Chapter 6.

Select Next to bring up the panel shown in Figure 7.10. In this panel you can select the activities of the maintenance plan to record to a table on the server. Later in the Examples section, I show you how to access this table to see the status of each step in the plan. You can also send this data to a table in a central server.

ch07_233-258.qxd 11/28/01 4:45 PM Page 245



Database Maintenance Plan Wizard - MARJIEK

Specify the Transaction Log Backup Plan
Specify the transaction log backup plan to prevent failures and operator errors.

☐ 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:

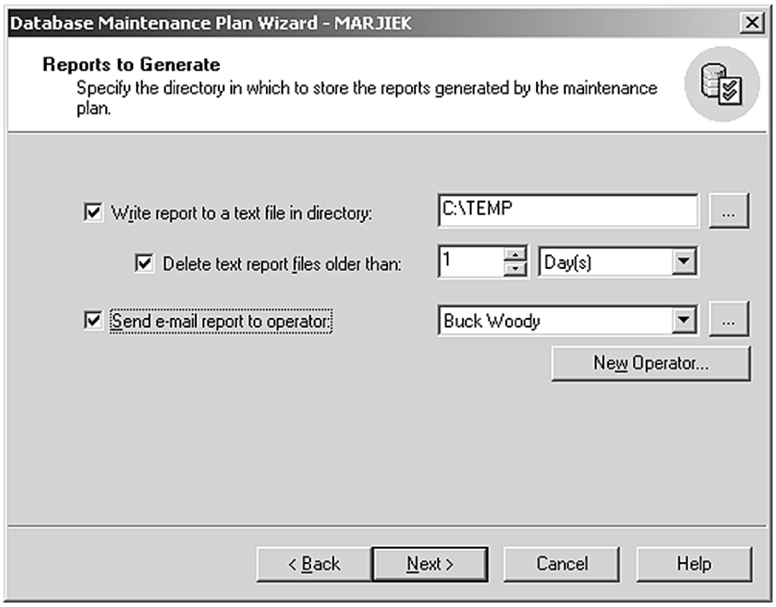
○ Tape:

● Disk

Schedule:

There are no operations to schedule.

Figure 7.8
Transaction log backups



Database Maintenance Plan Wizard - MARJIEK

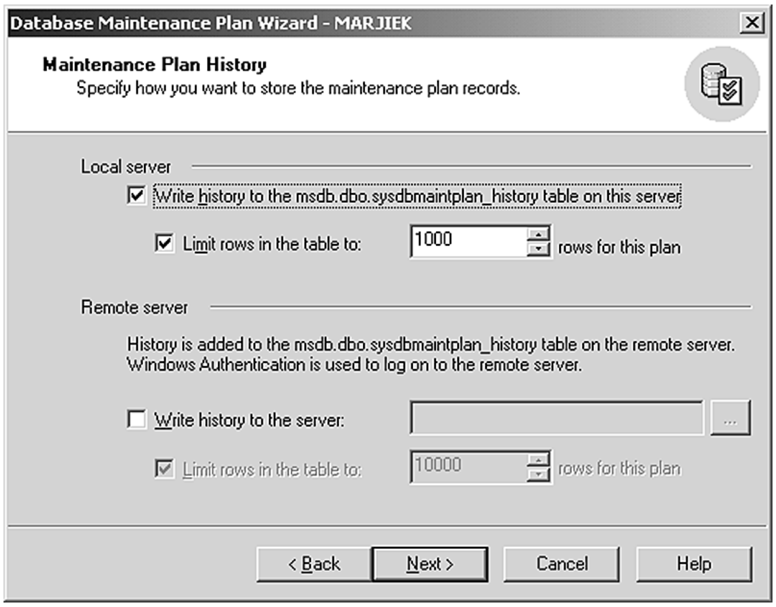
Reports to Generate
Specify the directory in which to store the reports generated by the maintenance plan.

☑ Write report to a text file in directory: C:\TEMP

☑ Delete text report files older than: 1 Day(s)

☑ Send e-mail report to operator: Buck Woody

Figure 7.9
Reports



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

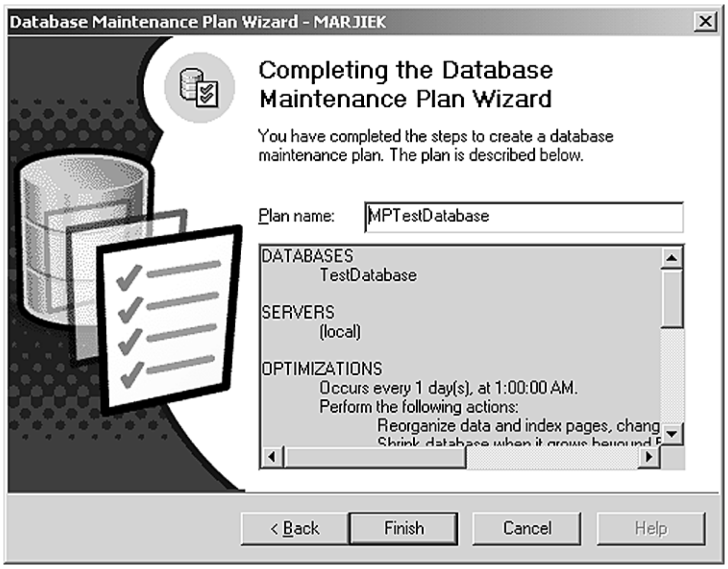
Remote server

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

Figure 7.10
Maintenance plan history



Completing the Database Maintenance Plan Wizard

You have completed the steps to create a database maintenance plan. The plan is described below.

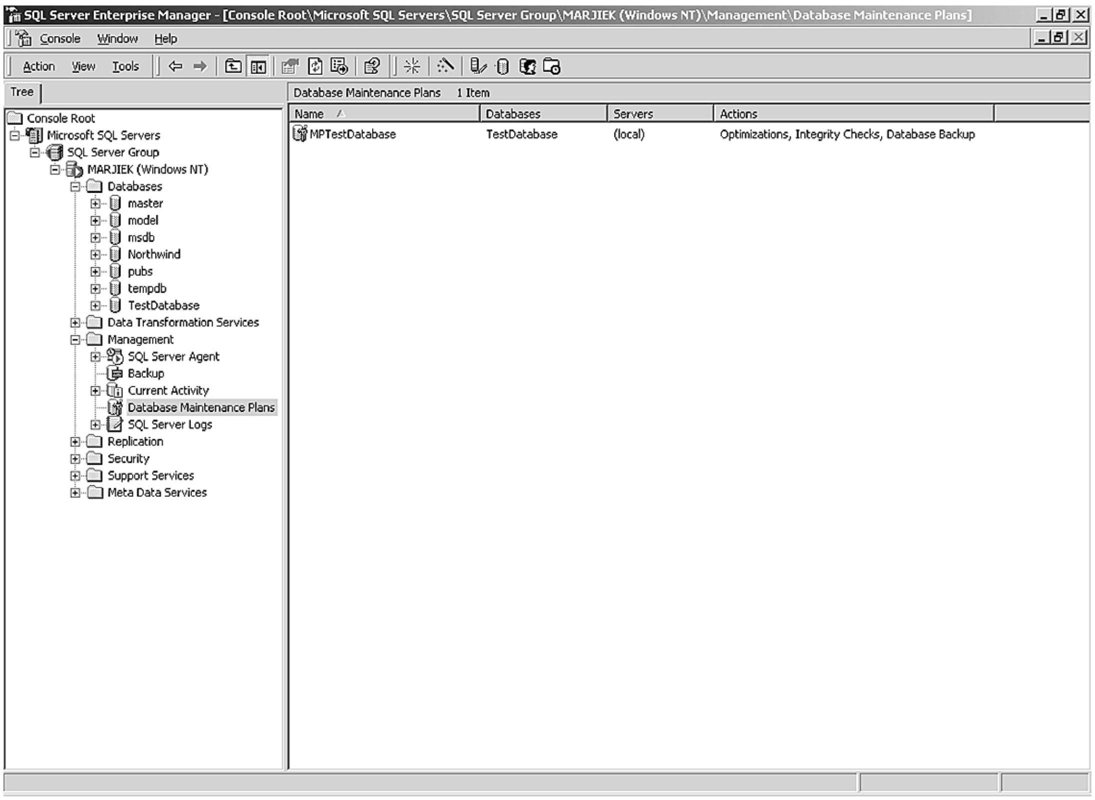
Plan name: MPTestDatabase

```
DATABASES
        TestDatabase

SERVERS
        (local)

OPTIMIZATIONS
        Occurs every 1 day(s), at 1:00:00 AM.
        Perform the following actions:
                Reorganize data and index pages, chang
```

Figure 7.11
Saving the plan



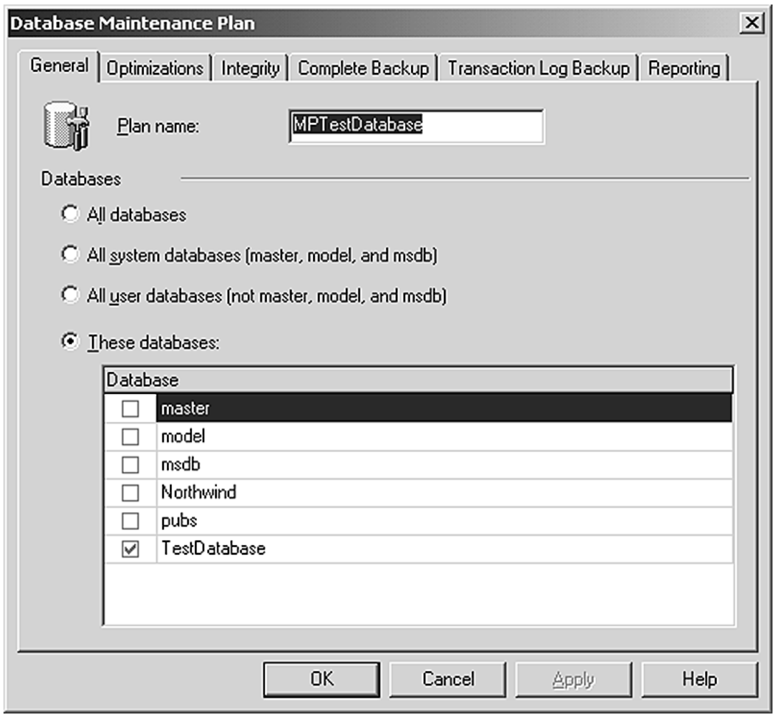
Name	Databases	Servers	Actions
MPTestDatabase	TestDatabase	(local)	Optimizations, Integrity Checks, Database Backup

Figure 7.12 *Maintenance plans object*

Pick Next to bring up the display shown in Figure 7.11. Name the maintenance plan, and then click Finish. To demonstrate the capability to edit the plan once it's complete, drill back down to the Maintenance Plans object, as shown in Figure 7.12.

Double-click the maintenance plan you created, and the display shown in Figure 7.13 appears. You can use the tabs to edit any of the items set up earlier. Once you've made your changes, select OK to save them. SQL Server 2000 takes care of the rest.

Set your plan to run in a few minutes from now; we'll display the history of the plan a bit later.


Database Maintenance Plan

Plan name: MPTestDatabase

Databases

- All databases
- All system databases (master, model, and msdb)
- All user databases (not master, model, and msdb)
- [✓ These databases:]

Database
☐ master
☐ model
☐ msdb
☐ Northwind
☐ pubs
☑ TestDatabase

Figure 7.13
The completed maintenance plan

Command Line

Although the Maintenance Wizard is the way to proactively maintain your database, sometimes running reactive commands may be necessary.

Recovering a Suspect Database

You'll run into these situations sometimes if a database server is shut down incorrectly or if there is some hardware failure on the server. I've not run into a lot of these errors, but they can happen.

Sometimes a database just gives up the ghost on its own. You'll see a grayed-out icon next to your database's name and the word "suspect" tacked on next to it in Enterprise Manager.

If this happens to you, don't panic. The first thing to do is to locate the last good backup and have it handy. This is a good reason for having the backups on a hard drive.

NOTE: If this is a mission-critical, time-sensitive database, it might be prudent to initiate a service call to Microsoft.

You can employ one of two fixes to correct a suspect database. The first is simply to make sure that your drives have enough space. If they don't, clear some space. It's the most common error you'll see.

The second thing to do is to reboot the server, or at least stop and start the SQL Server 2000 services. The reason for this is that when SQL Server 2000 starts, it automatically attempts to fix any problems it can.

The problem may have been caused by an OS error rather than an SQL one, so rebooting is often a good shotgun method to fix things. If that doesn't do it, your database may need more drastic corrections.

The complete list of emergency tactics to take against a problem can be found in Books Online under the topic Troubleshooting. We'll examine a couple of those involving the DBCC commands. You definitely should open the manuals and delve a little further into this topic before you experience this problem; then plan on paper the steps to take.

DBCC Commands

Many problems can be checked and even corrected by using the DBCC commands. These commands are the primary tool to use at the command line to diagnose and correct your ailing database. I'll focus on two of them.

DBCC CHECKDB

The `DBCC CHECKDB` command checks every object in a database for what SQL Server 2000 calls "structural" integrity. This means that everything is where SQL Server 2000 expects to find it and responds correctly. It does not mean that your data is OK—just that the database is *structurally* OK. Because this command affects everything in the database, it's not a good idea to run it in the middle of the day or when the server is in heavy use. You can—it's just not a good idea. Let's look at running this command in several flavors against your test database.

First, open Query Analyzer and run the simplest form against the database, as shown in Listing 7.1. Here I've told the `DBCC CHECKDB` command to display the results of checking the database. Listing 7.2 shows the long output of the command. It's worth reading at least once.

Listing 7.1 Simple DBCC CHECKDB

```
DBCC CHECKDB ('TestDatabase')
GO
```

Listing 7.2 DBCC CHECKDB output

```
DBCC results for 'TestDatabase'.
DBCC results for 'sysobjects'.
There are 21 rows in 1 pages for object 'sysobjects'.
DBCC results for 'sysindexes'.
There are 31 rows in 1 pages for object 'sysindexes'.
DBCC results for 'syscolumns'.
There are 256 rows in 4 pages for object 'syscolumns'.
DBCC results for 'systypes'.
There are 26 rows in 1 pages for object 'systypes'.
DBCC results for 'syscomments'.
There are 92 rows in 4 pages for object 'syscomments'.
DBCC results for 'sysfiles1'.
There are 2 rows in 1 pages for object 'sysfiles1'.
DBCC results for 'syspermissions'.
There are 18 rows in 1 pages for object 'syspermissions'.
DBCC results for 'sysusers'.
There are 12 rows in 1 pages for object 'sysusers'.
DBCC results for 'sysproperties'.
There are 0 rows in 0 pages for object 'sysproperties'.
DBCC results for 'sysdepends'.
There are 196 rows in 1 pages for object 'sysdepends'.
DBCC results for 'sysreferences'.
There are 0 rows in 0 pages for object 'sysreferences'.
DBCC results for 'sysfulltextcatalogs'.
There are 0 rows in 0 pages for object 'sysfulltextcatalogs'.
DBCC results for 'sysfulltextnotify'.
There are 0 rows in 0 pages for object 'sysfulltextnotify'.
DBCC results for 'sysfilegroups'.
There are 1 rows in 1 pages for object 'sysfilegroups'.
CHECKDB found 0 allocation errors and 0 consistency errors in database 'TestDatabase'.
DBCC execution completed. If DBCC printed error messages, contact your system administrator.
```

The rows in the pages that you see in the display conceptually correspond to Tables 7.1 through 7.3 that I showed earlier. Notice from the output that all the tables are checked. These include database system tables that SQL Server 2000 uses to track itself.

If the `DBCC CHECKDB` command finds a problem, you'll need to take steps to correct it. My standard operating procedure is to perform the less risky corrections first and then move to a more drastic command if that's necessary.

To do that, the server needs to be in single-user mode so that the users aren't able to access data. Notify your users to log out. If they don't or can't, you can use the T-SQL command `kill` to forcibly disconnect them. The process ID it needs can be derived from Enterprise Manager in the Current Activity object. Use the Properties menu item, Options tab, on a database using Enterprise Manager, and set the database to single user.

I recommend the command shown in Listing 7.3 in Query Analyzer. This method is a bit safer than using Enterprise Manager because Enterprise Manager can take multiple connections to the database.

Listing 7.3 Setting the database to single-user mode

```
ALTER DATABASE TestDatabase
SET SINGLE_USER
GO
```

Now that you've done that, you can go a bit further and allow the server to correct minor errors that it finds. The command to do this is shown in Listing 7.4.

Listing 7.4 DBCC CHECKDB with Repair Fast

```
DBCC CHECKDB ('TestDatabase', REPAIR_FAST)
GO
```

I won't bog you down with the output, which is the same as the output you saw in the earlier listing. The important thing is that the `REPAIR FAST` qualifier does the least amount of damage. As a matter of fact, this command is made to not lose any data. If that doesn't work, you escalate the `DBCC` routine to the command shown in Listing 7.5.

Listing 7.5 DBCC CHECKDB with Repair

```
DBCC CHECKDB ('TestDatabase', REPAIR_ALLOW_DATA_LOSS)
GO
```

This command is the most harmful but could save the overall database. When you run this command, SQL Server 2000 takes the gloves off and fixes the data any way it can. You get messages based on what was fixed, and you should print those so you can do a postmortem on the results. I recommend using this command only on a database that you're willing to restore from an earlier backup.

When you're finished, you need to put the database back in multiuser mode, or no one will be able to get in. The command to do that is shown in Listing 7.6.

Sometimes you don't need to fix a problem; you just want to update the indexes on a database. You do that with the next DBCC command, `DBREINDEX`.

Listing 7.6 Setting the database to multiuser mode

```
ALTER DATABASE TestDatabase
SET MULTI_USER
GO
```

DBCC DBREINDEX

The `DBCC DBREINDEX` command updates the indexes on any or all of the indexes in a database. In Listing 7.7 I've lifted the example from Books Online to demonstrate this command, since we didn't create any indexes on your test database.

Listing 7.7 DBCC DBREINDEX from Books Online

```
DBCC DBREINDEX ('pubs.dbo.authors', UPKCL_auidind, 80)
GO
```

In this example, I've selected the authors table in the pubs database and then specified a particular index on that table. All indexes have names that you can set; but if you don't, SQL Server 2000 sets them for you.

The number on the right specifies how full the index will be, which in turn indicates how much room is left for new index information. There's a whole section on choosing this number in Books Online; I cover that topic a bit more in Chapter 9.

Examples

As I mentioned earlier, I stole a little thunder from this section by placing an example of creating a plan in the Graphical section. Because your time is valuable and you've already seen the process once, I won't repeat that process here.

I discuss a neat query that shows the success of a maintenance plan, and I also show an example of correcting a corrupt database.

Checking the Success of a Maintenance Plan

Recall that you created a maintenance plan earlier in the chapter and you selected that the history of the plan be stored in a table in the local server. You can now use the query shown in Listing 7.8 to see the results of that history table. The output is shown in Listing 7.9.

Listing 7.8 Query to show the maintenance plan history

```
SELECT plan_name
, activity
, succeeded
, Duration
FROM
msdb.dbo.sysdbmaintplan_history
```

Listing 7.9 Maintenance plan query output

```
plan_name       Activity                      succeeded  duration
MPTestDatabase  Rebuild Indexes               1          1
MPTestDatabase  Shrink Database               1          1
MPTestDatabase  Check Data and Index Linkage  1          3
MPTestDatabase  Backup database               1          3
MPTestDatabase  Verify Backup                 1          0
MPTestDatabase  Rebuild Indexes               1          1
MPTestDatabase  Shrink Database               1          0
```

The "duration" column is in minutes. Notice also the column called "succeeded." This column returns a `1` for each successful part of the plan and a `0` for those that were not. Knowing that, you can tack on the line `WHERE succeeded = 0` at the bottom to display just the parts that didn't work. Combine this script with the Web page you created using the Web Assistant Wizard in Chapter 4, and you have an automatically updated Web page that shows the status of your maintenance plans.

Correcting a Suspect Database

This section demonstrates the correction steps I talked about earlier in the chapter. To do this, I had to run scripts to corrupt my database, but I won't show those here. The database I corrupted is called pubs2 and is a mirror copy of the pubs database. You'll learn how to make a mirror copy of a database in Chapter 8.

In this example, the situation is that your users began to complain that a program that uses a SQL Server 2000 database has suddenly stopped working. You investigate the problem, and it seems that everyone is getting the same errors in the same location, making you suspicious that something endemic is wrong. You check the user's program files and middle tier, and everything seems to be in order. You begin to suspect that the database may have a problem.

The first step is to diagnose the error rather than blindly attempting to correct it, so you type the `DBCC CHECKDB` command and receive the output, both shown in Listing 7.10. I've removed the nonerror parts to save you some reading time.

Listing 7.10 Detecting a corrupt database

```
DBCC CHECKDB('pubs2')
GO
-------------------
DBCC results for 'pubs2'.
DBCC results for 'roysched'.
Server: Msg 7965, Level 16, State 2, Line 1
Table error: Could not check object ID 213575799, index ID 0 due to invalid
allocation (IAM) page(s).
Server: Msg 8968, Level 16, State 1, Line 1
Table error: IAM page (53764:0) (object ID 213575799, index ID 0) is out of the
range of this database.
Server: Msg 8996, Level 16, State 1, Line 1
IAM page (53764:0) for object ID 213575799, index ID 0 controls pages in filegroup
0, that should be in filegroup 1.
Server: Msg 2576, Level 16, State 1, Line 1
IAM page (0:0) is pointed to by the previous pointer of IAM page (1:110) object ID
213575799 index ID 0 but was not detected in the scan.
Server: Msg 2575, Level 16, State 1, Line 1
IAM page (53764:0) is pointed to by the next pointer of IAM page (0:0) object ID
213575799 index ID 2 but was not detected in the scan.
Server: Msg 7965, Level 16, State 1, Line 1
Table error: Could not check object ID 213575799, index ID 2 due to invalid
allocation (IAM) page(s).
Server: Msg 8968, Level 16, State 1, Line 1
Table error: IAM page (53764:0) (object ID 213575799, index ID 2) is out of the
range of this database.
Server: Msg 8996, Level 16, State 1, Line 1
IAM page (53764:0) for object ID 213575799, index ID 2 controls pages in filegroup
0, that should be in filegroup 1.
Server: Msg 2576, Level 16, State 1, Line 1
IAM page (0:0) is pointed to by the previous pointer of IAM page (1:134) object ID
213575799 index ID 2 but was not detected in the scan.
There are 0 rows in 0 pages for object 'roysched'.
CHECKDB found 5 allocation errors and 4 consistency errors in table 'roysched'
(object ID 213575799).
CHECKDB found 5 allocation errors and 4 consistency errors in database 'pubs2'.
DBCC execution completed. If DBCC printed error messages, contact your system
administrator.
```

Now that you've determined that the database has a problem, you check to determine what it is. Reading from Listing 7.10, you see that you have both allocation (structural) and consistency (data-related) errors.

You try the least harmful fix first, using the `DBCC CHECKDB` command with the `REPAIR_FAST` option. Before you do that, you need to get everyone out of the database, and make a backup. You also need to set the database to single-user. You can do that with the commands shown in Listing 7.11.

Listing 7.11 Setting the database to single-user mode

```
ALTER DATABASE pubs2
SET SINGLE_USER
GO
```

Now that you've done that, try the gentle fix. The commands are shown in Listing 7.12. I won't bore you with the output I got, but I can tell you it wasn't good!

Listing 7.12 DBCC command with least amount of damage

```
DBCC CHECKDB ('pubs2', REPAIR_FAST)
GO
```

Since we now know you could lose data, we make sure you have a current good backup. Type the next command, shown in Listing 7.13.

Listing 7.13 DBCC command with possible data loss

```
DBCC CHECKDB ('pubs2', REPAIR_FAST)
GO
```

You glance at the screen and notice that there are still errors in the output. You count them, and there are fewer. Repeat the command in Listing 7.13. This time you have no errors. You then check the data, and nothing is lost.

The method used was a bit of a "shotgun." We used the `DBCC CHECKDB` command instead of some of the other, more focused DBCC commands, which might have been more advisable. We were looking for the fastest solution to the problem, and the `DBCC CHECKDB` command checks the entire database. If this command doesn't fix the problem, you can always restore the backup you made before you began. You can then attempt to fix the database using the other DBCC commands. This approach isn't as useful if the database is large or time is of the essence. In that case, I would advise that call to Microsoft.

Notice also that we repeated the DBCC command. This is a common practice if the first run corrected some, but not all, errors. At other times, you'll run one type of DBCC command and then another type to fix the errors you'll get. If you get into that predicament, look up the DBCC commands in Books Online before you try your corrections.

Once you're done, comb through the logs, both Windows and SQL, to do a postmortem to find out how you got here. You should also document what happened so you (or someone) will know what to do if the problem reoccurs.

I've covered a very basic problem with databases. Countless problems can occur, but most are correctable. On the whole, SQL Server 2000 is incredibly stable, and I've not experienced any corruption problems with the 30-plus large databases I manage on as many servers. As a matter of fact, I had to go through quite a few gymnastics to corrupt a database for this example!

Rosetta Stone

You'll find that the commands for solving database problems are the one area that diverges the most in database platforms because each deals with that based on how it's put together. Therefore, the commands I parallel are fairly broad.

Oracle

Oracle doesn't have an automatic maintenance wizard in the 8i version I have, although there are some after-market products available to do the job.

Some tasks, such as backups, can be accessed through the graphical tools, but most of Oracle's tools for maintaining the database are command-driven. You can see these Commands compared with SQL Server 2000 in Table 7.4.

To be sure, there are many more commands in Oracle (and SQL Server, as you've seen) to maintain a database. Because of the difference in architectures, the two systems are maintained in ways that suit their particular physical implementation of an RDBMS. This isn't a cop-out for the lightness of the comparison, but the commands so distinctly diverge beyond what I've shown here that it's difficult to map them directly onto each other.

Table 7.4 Oracle and SQL Server maintenance commands

Task	Oracle Tool	Microsoft SQL Server 2000
Physical integrity check	DBVERIFY	DBCC commands
Reindex a current index	ALTER INDEX *index_name* REBUILD;	DBCC DBREINDEX
Back up and recover databases and logs	See Chapter 8	See Chapter 8

ch07_233-258.qxd 11/28/01 4:45 PM Page 257

Redo Logs, Key-Compressed Indexes, and other Oracle constructs are simply not correlated in SQL Server 2000.

Microsoft Access

With Microsoft Access, there aren't many maintenance or recovery commands. The closest thing you have in Access to all the commands involving maintenance is the Compact and Repair Database menu item.

Resources

SQL scripts involving maintenance:
http://www.swynk.com/sqlhome/maintenance7.asp
More on automating maintenance tasks:
http://www.microsoft.com/technet/SQL/manuals/admincmp/75517c06.asp
Jobs:
http://www.sqlmag.com/Articles/Index.cfm?ArticleID=5492