Paragon Drive Backup Enterprise Server Edition

Best Practices for MS SQL Server

Contents

1 Introduction

This paper addresses various aspects of a Microsoft SQL Server data protection by using Paragon Drive Backup Enterprise (DBE). It describes the concepts, limitations and best practices for Paragon Drive Backup Enterprise to protect "no downtime" operational solutions based on Microsoft SQL Server. All mentioned recommendations are generic and not specific for a certain SQL Server application.

1.1 About Drive Backup

Paragon Drive Backup Enterprise is a backup tool that implements the best of volume imaging techniques for reliable, fast and convenient data backup and restoration. The program includes end-user tools for building and automating recovery and replication procedures. It implements high-performance algorithms for intelligent data analysis and processing, provides an optimized manipulation for a large set of filesystems that covers all popular filesystems for Windows and Linux platforms and more features.

1.2 Backup Concepts

1.2.1 File-Level Backup and Volume Imaging

There are two concepts about backup subject. A *file-level backup* is oriented to store separate files. A *volume-level backup* or *imaging* is oriented to store whole filesystem of a volume.

A *file-level backup* naturally provides an intuitive and flexible way to select objects to store. File-oriented backup tools allow to choose any combination of both local and network accessible files. A file-level data restoration allows to selectively restore only damaged files without affecting other ones. However, there are important file-related system objects which are not files and usually cannot be stored, restored and even accessed from a file level.

A *volume imaging* can store files and any associated metadata including distribution information, security data, quotas, extended attributes, named streams, multiple hard and symbolic links and so on. Imaging tools generally provide higher backup performance because they do not involve filesystem drivers to the process. In addition, they can backup offline filesystems including ones not being supported by a host operating system. A data restoration generally does not require a host operating system to run, so that imaging technique is a perfect choice for system cloning and disaster recovery tools. Disadvantages of volume imaging are that it cannot be applied to remote resources and a general ineffectiveness of backup and restore of selective files within the volume-imaging framework.

1.2.2 Data Consistency

The fundamental requirement to backup is saving of *data consistency*. This means that if applications are stopped, and data are restored, and applications are restarted, they will run smoothly with restored data.

A *data consistency* is conditionally divided into a *physical* and *logical* consistency. A *physical consistency* means storing involved data files and file-related attributes in a state that is interpreted by applications as "integral", or "valid", or "auto-repairable on-the-fly". A *logical* consistency means an application-level correctness of stored business data. An automatic correction of minor inconsistency of business data is usually provided by the *transactions mechanism*, which is a basis of modern technologies of information processing.

1.2.3 Offline and Online Backup

As regards to the data consistency concept, offline data are in consistent state (with the only condition that an application or a system was shut down correctly). Offline data archiving is referred to as *offline backup*. Its advantages are ensured data consistency, increased backup speed (due to absence of concurrent data access), resource saving and low impact to a system performance.

The disadvantage is that offline backup is not applicable for "24x7 availability" systems. For continuously working systems *online backup* methods should be applied.

1.2.4 Snapshots

The great challenge to online backup is to provide a coherent state of all open files and databases involved in a backup, under the condition that applications may continue writing to a disk.

The "volume snapshot" concept has met that challenge. A *snapshot* is a point-in-time copy of a volume involved to a backup process. It must be quickly created at a period when applications do not write to disk. Once snapshot has been created, a backup utility copies data from it while applications continue working with an original volume. Modern snapshot technologies reduce *backup window* down to few seconds.

There are hardware and software based snapshot solutions. *Hardware snapshots* such as the *Split-Mirror* technique instantly provide an independent copy of a whole data set; they make no impact to the system performance and can participate in off-host backup solutions. Disadvantages are that hardware snapshots are naturally hardware-dependent, double requirements to storage resources and some time is required in order to re-synchronize a hardware snapshot after it has been used, with the actual disk data.

Instead, *software snapshots* are hardware-independent and resource-saving solutions. The disadvantage is that they cannot make a copy of a whole data set instantly, so that they run and make additional computing load to a host system until a backup routine is ended and a snapshot is destroyed.

Unlike hardware snapshots, types of software snapshots, which are used in practice, are not independent from original data. In case an original volume fails, all of its snapshots fail as well. Software snapshots do not really protect data; they are used as supplementary mechanisms in backup solutions.

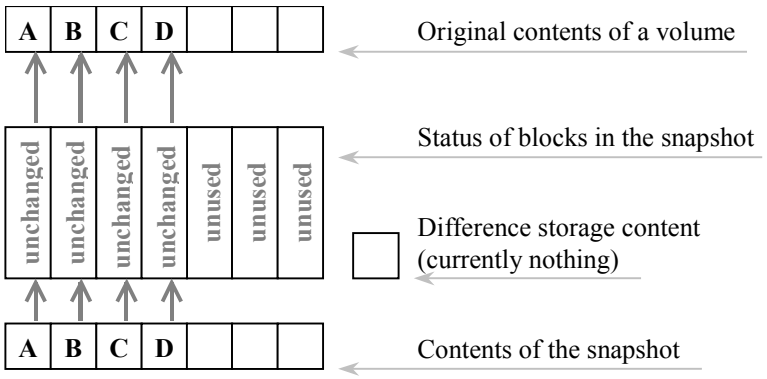
Paragon Drive Backup provides the original online backup technique (named *Paragon HotBackup*) and supports any snapshot technologies that can participate in the MS VSS framework.

1.2.5 Copy-On-Write

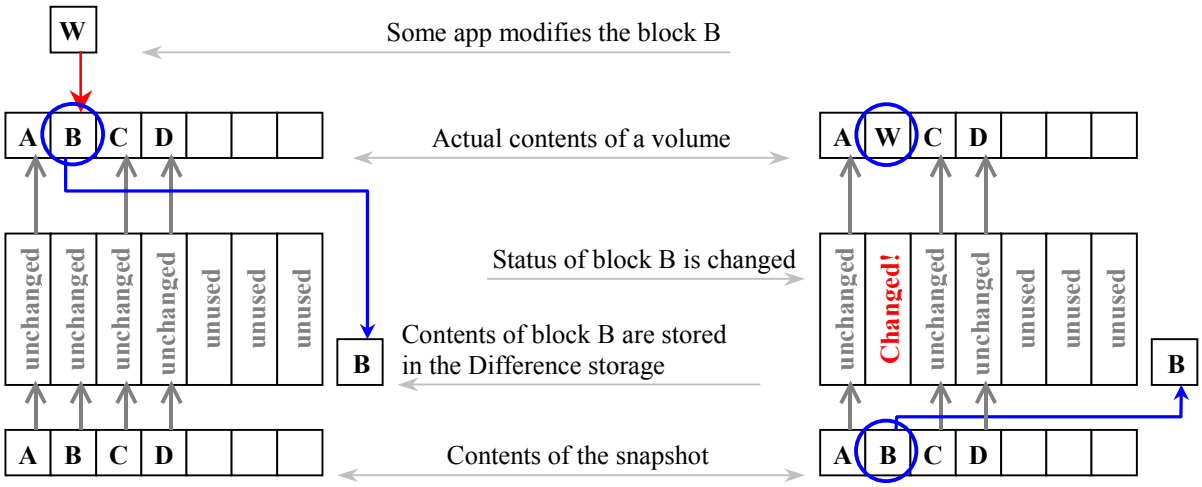
The *Copy-On-Write* (COW) method predominates among software snapshot techniques. It is based on the idea of preserving original contents of modified blocks before modifications are written to a disk, in a special *difference storage*. COW maintains an original volume in the actual state and stores its original content at the moment of snapshot creation in the difference storage.

A system-level agent, which is responsible for snapshot maintaining, must keep track of all changes being made on a volume. On first attempt to overwrite a block, the snapshot agent copies original block's content to the difference storage and then allows modifying that block. When a backup utility acquires some block from the snapshot, the agent determines its status and takes a "changed" block from the difference storage while "unchanged" block is taken from the original volume. The following illustrates COW basics:

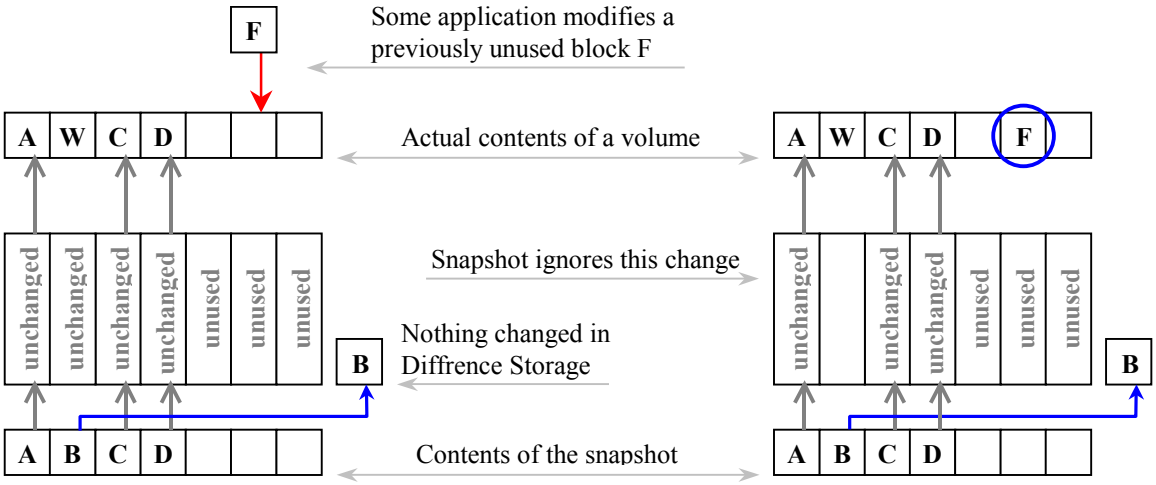
1. A volume was partially in use. Blocks A, B, C, and D contained data. A COW based snapshot was created for this volume. The snapshot provider watches only the used blocks of a volume. Initially they are marked as "unchanged".



2. Some application tried to modify the block B. Before modifications are made, the snapshot provider copies contents of the block B to the difference storage. The block is now marked as "changed". Then the block B is updated. The snapshot will redirect all queries to the block B to data stored in the difference storage. Queries to blocks A, C and D will be directed to appropriate blocks of the volume.



3. Some application tried to modify the block F, which was not originally in use. This block was not included in the snapshot. The snapshot provider does not take care of this change. Nothing is changed in the snapshot data.



1.2.6 Data Synchronization

Another problem for online backup functionality is that applications may temporarily hold open files in logically inconsistent state. The snapshot techniques do not solve that problem as it concerns solely to business application operation. The true reason of the problem is that applications are unaware about a backup routine running and do not synchronize their data.

Microsoft has made a great attempt to solve the problem. The snapshot backup framework referred to as Volume Shadow Copy Service (VSS) has been built in latest versions of Windows, exactly to Windows XP, Windows Server 2003 and Windows Vista. VSS includes mechanisms for notifying applications about a backup, interchanging of related information between VSS participants and synchronizing execution of software components involved to the process.

VSS has several significant limitations:

- Only VSS compliant applications can benefit from VSS framework.
- VSS is a local solution that works within a single host.
  Remote applications and distributed data systems aren't controlled by VSS.
- VSS currently works to its full capacity on Windows 2003 only.

1.2.7 Write Inactivity Paradigm

There are other (partial) solutions for synchronization problem that are applicable for non-VSS compliant software. There is a popular solution based on the paradigm of *Write Inactivity Period* (WIP) that was introduced by St.Bernard Software Company in the middle of 1990-th.

It is supposed that business applications for intensive information processing use transactions mechanisms. A *transaction* collects I/O writes into a compact group in order to reduce a chance of incomplete transaction committing if a failure of any type occurred. Data are in consistent state between transactions, so that a period when application(s) do not write to a disk is the best moment for a snapshot capture. This period is referred to as WIP – *Write Inactivity Period*.

2 Technology Overview

2.1 Paragon HotBackup Description

Paragon HotBackup is an online backup technology for Win'NT+ family operating systems. It was developed in 2001 and integrated to all company's backup solutions in 2002-2003. Currently it supports all versions of Windows NT4, 2000, XP and 2003 (including x86, IA64 and AMD64 versions).

2.1.1 Hotbackup Concepts

HotBackup is not a snapshot technology. However its concepts appear to be similar to ones used in software snapshot technologies. In particular, a sort of WIP observation and COW scheme are implemented.

During an online backup, Drive Backup uses the kernel mode driver HOTCORE.SYS in order to monitor and control write activity of applications and an operating system. The driver intercepts disk I/O requests and implements the most time-critical part of COW scheme while the Drive Backup utility includes disk data analysis and archiving functions.

The HOTCORE driver is installed during the standard Drive Backup setup procedure, and it is the reason why the system restart is required in order to complete the setup procedure. HOTCORE does nothing until is activated by the Drive Backup. In the idle mode the driver bypasses any calls, makes no impact to the disk subsystem performance and only takes few kilobytes of system memory.

2.1.2 How it Works

HOTCORE driver is activated only in case the online backup is performed. In an offline backup mode, the driver is not involved to the process.

- Drive Backup activates the HOTCORE driver in the beginning of the "physical" backup.
- HOTCORE waits for a pause between I/O writes on the targeted volume.
- When the pause is observed, the driver takes a "snapshot".

Within the framework of HotBackup, a "snapshot" is a map of blocks to be protected by the COW scheme against loosing their original contents. The process requires the close cooperation between the driver and the utility. Finally, the "protection area" of the snapshot includes only used blocks with optional exception of *excluded files* (e.g. PAGEFILE.SYS and HIBERFIL.SYS). The embedded module for filesystem analysis allows reducing this step down to few seconds or less.

- During the snapshot capture, the driver watches the volume against I/O writes. If a write occurred before "snapshot" is created, the step is repeated.
- Upon successful snapshot creation, the driver applies the COW scheme to the "protection area".
- The utility starts the backup process. Archived blocks are immediately excluded from the protection area, so that the area is shrinking during the backup.

- If a foreign application tried to modify a "protected" block, the driver preserves its original contents in a buffer. Initially, blocks are stored in a memory buffer. When it becomes near full, the utility moves blocks to temporary files (named like "X:\hb_nn.tmp", where X: is a drive defined in the Settings).
- Normally Drive Backup performs the fastest streamlined backup of used blocks. However, if temporary files grew very fast or became very large, the utility pauses the streamlined mode and begins emergency backup of buffered blocks.

In online backup the following time-related restrictions are used:

- A snapshot must be created within 60 seconds.
- Buffered data must be processed within 10 seconds.

These restrictions are hard-coded and cannot be changed. If any of these restrictions were not satisfied, the online backup session fails. If the backup operation was aborted by a user or the utility failed, the driver automatically switches to the idle mode in 10 seconds.

2.2 MS VSS Basics

Volume Shadow Copy Service (VSS) is an open system-level framework for snapshot backup solutions. It was developed by Microsoft in close cooperation with leading vendors of backup solutions. VSS is included in Windows XP, Windows Server 2003 and Windows Vista.

VSS provides mechanisms for notifying applications about a backup, interchanging of related information between VSS participants and synchronizing execution of software involved to the process. These mechanisms ensure consistent backup of online data for VSS compliant applications.

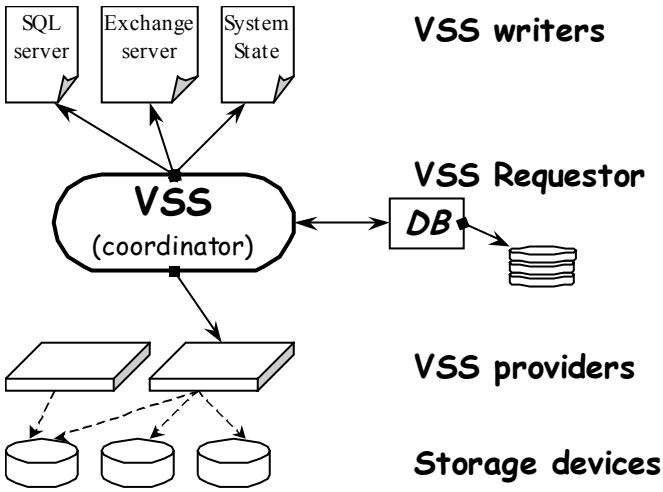
2.2.1 VSS Concepts

VSS is based on concepts of a snapshot and a volume shadow copy. Being invoked by a VSS aware backup utility, VSS creates snapshots for selected volumes and represents them as virtual read-only devices, which are referred to as *shadow copies of volumes*. Once shadow copies are created, a backup utility stores data from shadow copies while business applications continue writing to original volumes.

There are three kinds of software involved to the VSS framework:

- *Providers* (snapshot providers) – tools that create and maintain hardware or software snapshots.
- *Requestors* – utilities that acquire shadow copies, usually backup utilities.
- *Writers* – VSS compliant applications that hold open files on volumes, actual backup objectives.

Within the VSS framework, VSS writers are able to inform other VSS participants about files being in use, file grouping and restoration conditions. A group of files that constitute a whole entity in a business application and should be backed up together is named a *writer's component*. For example, all files that constitute a database in MS SQL Server are represented as a single writer's component. VSS writer can be an application itself or a special agent, which provides VSS-to-application interaction.



VSS itself only coordinates activity of providers, writers and requestors. A standard Windows XP/2003 distribution includes the VSS coordinator, the universal software provider (VOLSNAP), several VSS writers for

system components and the universal VSS writer for MS Desktop Engine (MSDEwriter). MSDEwriter provides integration of MS SQL Server 2000 to VSS framework.

2.2.2 How it Works

The comprehensive description of VSS can be found on appropriate Microsoft TechNet pages (e.g. see the descriptive topic "How Volume Shadow Copy Service Works",

http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/TechRef/2b0d2457-b7d8-42c3-b6c9-59c145b7765f.mspx). Below is only a brief description of MS VSS operation:

1. A VSS requestor (backup utility) invokes VSS to initialize the backup routine.
2. VSS acquires information from all VSS writers. As a result the *Writers Metadata Document* (WMD) is created. It contains distinctive names of applications and components, restoration parameters, list of files and other information.
3. A requestor receives WMD and makes a decision what to backup. It creates the *Backup Components Document* (BCD), which defines volumes, writers and components involved to the process and sends it to VSS. As an option, preferred VSS providers can be selected.
4. VSS commands to all involved VSS writers to finish running transactions and then "freeze". VSS waits until all involved writers complete that step.
5. Then VSS commands to appropriate VSS providers to create snapshots for selected volumes.
6. After snapshots are created, VSS allows writers to "thaw" and to write to disk. In addition, VSS asks writers if write operations were indeed suspended during the snapshot creation. If it was not the case, the whole backup procedure fails, snapshots are removed and the VSS requestor is notified.
7. Upon successful completion of above steps (within predefined time intervals), VSS creates *shadow copies* from snapshots and provides VSS requestor with appropriate references.
8. VSS requestor performs copying information from shadow copies….
9. Upon the process completion, the VSS requestor informs VSS about the backup completion status.
10. VSS informs all involved VSS writers about the backup completion status. VSS writers may use this notification to perform some specific actions (for example, Exchange truncates log files).

A shadow copy can be deleted immediately after the backup completes, or it can persist in the system. In the last case it can be mounted like an ordinary volume (this feature is available in Windows Server 2003 and Vista).

2.2.3 MS VSS Limitations

MS VSS has several significant limitations:

- Only VSS compliant applications can benefit from VSS framework.
- VSS is a local solution working within a single host. Remote applications aren't controlled by VSS.
- VSS currently works to its full capacity on Windows 2003 only.

2.3 How Drive Backup Integrates with MS VSS

Storage management frameworks obviously provide benefits and can simplify storage management activity. Now Paragon Drive Backup Enterprise is adapted to participate in VSS framework operation as a requestor.

2.3.1 Enabling Backup via VSS

To perform backup operations via VSS services, select the "Microsoft Volume Shadow Copy Service " option:

(menu:) General
↳ Settings…
↳ Hot processing options
↳ Hot processing technology
↳ Microsoft Volume Shadow Copy Service

Note that MS VSS service is available only in Windows XP, 2003 and Vista. In other operating systems, only Hotbackup technology is available for online backup.

Drive Backup provides a simplified VSS management, so that not all VSS options are controllable by a user. The program internally chooses only most reliable VSS operation modes.

2.3.2 How it Works

A VSS aware backup is performed in the following manner:

- A user chooses volume(s) to be backed up in the program's interface.
- When the "physical" backup operation begins, the program receives the compound WMD document from VSS (see #2 in the VSS description topic).
- Drive Backup determines VSS writers having components located on chosen volumes. These VSS writers are included to the BCD document (see #3). In other words, all VSS writers containing files on targeted volumes should participate in VSS operation at the snapshot creation (i.e. should be "frozen" and "thawed" by VSS).
- Drive Backup commands VSS to use the default order of VSS providers invocation: a hardware provider first (if available), a third-party software provider next (if available), the last is Microsoft's universal system provider VOLSNAP (always available).
- The program performs the volume-level backup of the created shadow copy set.
- After the operation completes, the shadow copy set is deleted.

Currently the program does not support restoration via VSS writers. In fact, VSS based restoration is generally just a file copying. Applications should be stopped, or appropriate components should be detached/dismounted manually in order to be restored.

2.4 Choosing between Online and Offline Backup

The reasons to prefer offline backup are:

- Online backup via Hotbackup or VSS option is slower than offline backup.
- VSS initialization is long and generally unstable under high IO traffic on a targeted volume.
- A resulting image produced by Hotbackup is slightly larger than one produced in offline mode, because of non-sequential image structure, see Hotbackup description.
- Neither of online backup options totally eliminates problems that are naturally inherent for online backup technologies in general.

VSS provide data consistency for VSS compliant applications only. Hotbackup does not guarantee 100% data consistency in any case, but only a very high probability of that. In fact, it provides perfect results for any applications that use transactions (e.g. SQL servers, MS Exchange).

Drive Backup provides some flexibility of choosing between offline and online backup mode. A user can choose between three modes: (a) "always offline backup", (b) "switch to online if a volume was in use" and (c) "always online backup". The differences between these modes are the following:

- (a) Always offline mode:
  - The program does not backup volumes being in use. It suggests to reboot in order to complete the operation in the "Startup Bluescreen" mode.
  - If a volume wasn't in use, the program switches to exclusive use of the volume. No applications can access any files on the volume until the backup procedure is finished.
- (b) Switch to online if a volume was in use:
  - If a volume was in use, the program performs the online backup. Other applications are allowed to access the volume during the operation.
  - If a volume wasn't in use, the program performs the offline backup. As it was described above, no applications can access any files on the volume until the backup procedure is finished.
- (c) "Always online backup":
  - The program unconditionally performs the online backup. Other applications are allowed to access the volume during the operation.

3 Protecting MS SQL Server

Microsoft SQL Server is a general-purpose relational database server, which can scale from hosting simple databases to mission critical business applications. It is the most popular relational database on Microsoft Windows platform. The more your business depends on SQL Server, the more important it is to protect it.

This version of Paragon Drive Backup Enterprise has no SQL Server specific functionality. For this reason, some considerations should be taken into account in order to provide reliable SQL Server databases backup.

The best practices presented in this guide are general principles, not guidelines for specific environments. This chapter discusses general requirements for both OLTP and DSS types of applications (OLTP – On-Line Transactions Processing applications are characterized by predominated use of rows insertions and modifications, which is equal to I/O write-after-reads on the disk level. DSS – Decision Support Systems are characterized by predominated use of rows selections, which is equal to massive I/O reads on the disk level).

3.1 Databases in MS SQL Server

Designing and implementing a SQL Server database is fairly intuitive, but it is important for database performance, maintenance and future growth. Understanding the relationship between database files, volumes and storage devices is essential for optimal use of Drive Backup Enterprise (DBE). This section discusses various layouts of MS SQL Server databases in relation to Drive Backup usage.

3.1.1 Database Physical Layout

There are two quite different types of database layouts in MS SQL Server:

1. A database can be placed onto a RAW partition.
   In-partition database layout provides some performance gain. However, these databases are not extensible and are rarely used for this reason. Drive Backup can be used for backing up of in-partition databases with no limitations.
2. A database can be placed into a set of files.
   It is a standard type of database layout. Database files can be located on any local volumes and network shared drives and can be spread over multiple volumes. Drive Backup can be used for backing up of file-based databases with some weak constraints.

Any database consists of *data storage* and *transaction log*. The first one is used for long-term storing of data, the second one is used for transaction managing and holding most recent data changes. In a file-based database both parts can be distributed over multiple files. Within the framework of MS SQL Server, database files usually have the following default extensions:

- **.MDF** – "main data file", the primary data file. The mandatory part of every database.
  It stores data and keeps database's servicing information including list of database files.
- **.NDF** – "next data file", secondary data file(s). An optional part of a database.
  Used for extending a database and sometimes for "multi-streaming" data access.
- **.LDF** – "log data file", the transaction log file(s). The mandatory part of every database.
  It keeps information for transaction management and recovery.

Data files are aggregated into *file groups* for better manageability. A file group consists of one or more data files, which by default are placed at same location.

For example, if your database was named "ABC" and placed into the "S:\ABC" folder, if consists of following files: the primary data file "S:\ABC\ABC_DATA.MDF", the transaction log "S:\ABC\ABC_DATA.LDF" and secondary data files "S:\ABC\ABC_DATA_1.NDF", "S:\ABC\ABC_DATA_2.NDF" etc.

3.1.2 System and User Databases

There are several *system databases* in MS SQL Server that keep structure, administration, maintenance and other servicing information about SQL Server databases:

master	Keeps system level server's settings such as accounts, locations of databases, startup info etc.
model	Is used as the template for all newly created databases.
msdb	Is used by the SQL Server Agent for storing maintenance jobs information.
tempdb	Is used for temporary storage needs. It is recreated every time SQL Server is started.

While user-defined databases are important for business, system databases are critical for SQL Server operation. All types of databases have same physical structure (see previous section for details).

3.1.3 Performance

There are multiple factors that affect database performance. The most significant of them are available memory and disk subsystem throughput. Available memory affects on read operations throughput, which is important for Decision Support Systems (DSS) performance. Disk subsystem throughput affects performance of both read and write operations. High write performance is essential for online transactions processing (OLTP) applications.

Most effective ways to increase disks throughput are using faster disks and using RAIDs. Striped RAID sets provide highest read/write performance, which is found nearly a multiple of single disk performance.

RAID 0	Striped Disk Array
RAID 0+1	Mirroring of striped segments (RAID 1 over RAID 0)
RAID 3	Striped Disk Array with Isolated Parity
RAID 5	Striped Disk Array with Distributed Parity
RAID 10	Striping of mirrored segments (RAID 0 over RAID 1)

It is a good practice to place your production databases on a high-performance RAID set.

3.1.4 Reliability

While database backup provides a disaster recovery option, there are ways to reduce the chance of a failure. A hardware-based method is to use fault-tolerant RAID configurations (for example, RAID-10 or RAID-5).

Additionally, system reliability can be increased by isolating databases from an operating system, from each other and from other intensively used file resources. It is a good practice to place your production database(s) on an isolated non-system volume, or even to place each database on a separated volume.

3.1.5 Backup

User-defined and system databases must be regularly backed up in order to protect business applications from disaster.

The only exclusion from this rule is **tempdb** system database. It is automatically re-created every time SQL Server is started from a clean copy, and its content are automatically dropped on disconnect. The SQL Server operates in such a way that **tempdb** contents never require to be saved. During a working session **tempdb** may grow large, so that it is reasonable to surely exclude **tempdb** contents from backups.

3.2 Backing up MS SQL's Databases with DBE

Some concerns should be taken into account in order to use Paragon Drive Backup Enterprise for successful backing up of MS SQL Server databases. The following circumstances should be considered:

- Is a database located on a mapped network drive(s)?
- Is a database spread over multiple volumes?
- Should a database be backed up online, or it is acceptable to temporarily hold it offline?
- Which operating system is running on the host?

The governing factors that limit Drive Backup adaptability for SQL Server backup are the following:

- This version of Drive Backup does not backup network drives.
- This version of Drive Backup does not include SQL Server specific agents. For this reason, databases are backed up like ordinary files. The WIP detection mechanism is used for database's data synchronization.
- Databases are always backed up in a mode that is effectively equal to the *full backup* (in terms of database backup types).

This version of Drive Backup is unaware of database logical structure. Drive Backup's "full" and "incremental" modes refer to volume backup modes at block level, not to database backup modes. At restoration, a whole database will be rolled back to the pre-backup state including data files and transaction log. This behavior is effectively equal to the full database backup-&-restore.

3.2.1 Recommended Database Layouts

General recommendations:

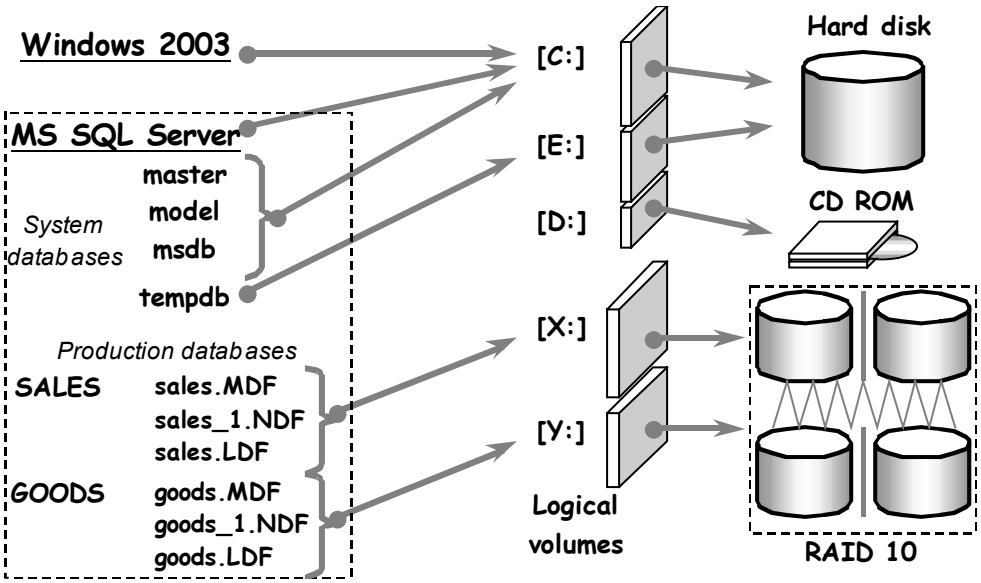
1. Place your production databases on local volumes.
2. Place your production databases on volumes located on high-performance RAID set(s).
3. Place your production databases on a dedicated volume (or each database on an isolated volume).
4. Place your production databases separately from an operating system.
5. Place the **tempdb** system database separately from production databases or on an isolated volume.

There are system-dependent recommendations:

6. In Windows Server 2003 and Vista, set up Drive Backup to use MS VSS for online backup of databases distributed over multiple volumes. Choose the "Microsoft Volume Shadow Copy" item in the "Hot processing technology" pull-down list in the program settings. VSS gives benefits for backing up MS SQL Server 2000 and 2005 and enhances the chance of successful backup of distributed databases.
7. Only MS SQL Server 2000 and 2005 are VSS compliant. For earlier SQL Server versions, there is no noticeable difference between Hotbackup and VSS options. VSS is available only in Windows XP, 2003 and Vista.
8. In Windows 2000 and NT 4.0 environments, place a whole database (data files and transaction logs) on a single volume. For these operating systems, only Hotbackup option is available. Hotbackup performs an asynchronous backup of multiple volumes, so that a distributed database may be archived in logically inconsistent state.
9. Distributed databases can be successfully backed up in offline mode. Either detach a database or take it offline, or stop the SQL Server before backing up volumes that contain distributed databases. For an offline database, both online and offline backup modes provide successful results.

A preferred system configuration could be like this:

- There is a high-performance RAID set that is dedicated for placing production databases. For example, a RAID 10 provides both high reliability and performance levels.
- There is a separate storage (hard disk or another RAID set) dedicated for placing the system partition, MS SQL Server binaries, and the **tempdb** database and for storing backup images.
- The system hard disk is partitioned in several volumes. The **tempdb** database is separated from an operating system and SQL Server binaries and is located on a dedicated volume.
- Backup images are stored on a non-system volume, probably on a volume containing **tempdb**.



3.2.2 Unsupported Database Layouts

This version of Drive Backup has some limitations in relation to backing up SQL Server databases:

1. A database that is entirely located on a mapped network drive cannot be backed up by using Drive Backup, because the program does not backup network drives.
2. A database that is partially located on a mapped network drive cannot be backed up by using Drive Backup, because not all database files can be stored.

Generally, a database partially located on a mapped network drive may be an obstacle for accurate data restoration for the whole volume. As Drive Backup is unable to backup completely such databases, it is unable to restore them correctly. To avoid possible problems caused by incomplete database restoration, do not apply a volume-level restoration of volumes containing distributed databases. Instead, a file-level restoration should be applied for such volumes. Use the "Image Explorer" utility for the file-level restoration from Drive Backup's backup images.

3. In Windows 2000 and NT 4.0 environments, distributed online databases cannot be successfully backed up in online mode (because of Hotbackup option synchronization limitations). Such databases can be accurately backed up in offline mode. Take offline or detach databases before backing up volumes containing distributed databases.

3.3 Restoring MS SQL's Databases with DBE

This version of Drive Backup does not apply online restoration for SQL Server databases. Offline restoration is the only available method. A database migration-at-restoration is not available as well.

A database can be restored in two ways. First, any database can be restored as a part of the volume restoration procedure. Second, database files can be manually extracted from a volume's image and then manually placed to their original location. In any case, a database being restored must be offline.

The difference between these methods is the following:

- At *volume-level restoration*, all contents of a targeted volume are rolled back. For example, if there were three databases on a volume, all of them are rolled back to the pre-backup state during the volume-level restoration. File system metadata are also restored and put into consistent state.
- The volume-level restoration is very fast and is able to recover volumes from scratch.
- At *file-level restoration*, one can extract from an image only required files. It provides selective data retrieval. Other volume's contents are not affected. However it is rather slow and requires a targeted volume to be healthy. File-level restoration cannot be used for recovering damaged volumes.

There are two ways to restore a damaged database and make it available again. The first one is to restore database files to their original location(s) and re-start SQL Server service(s), so that it will automatically attach the restored database. However, the SQL Server will be unavailable for users for some time, exactly for the period required to: disconnect all users, stop all databases, flush data to the disk, stop and re-start the SQL Server services and perform the so-called "soft recovery" procedure for all attached databases including the restored one. The whole process may take a while, and it will be the Server's blackout period.

Another way is to detach the damaged database from the SQL Server prior to restoration. It does not require to stop the SQL Server and thus allows avoiding blackouts for workable databases.

1. Run the SQL Server Enterprise Manager and select the database of interest in the tree view:

Console Root
↳ Microsoft SQL Servers
↳ *<SQL Server Group>*
↳ *<Server Instance Name>*
↳ Databases
↳ *<database name of interest>*
↳ (popup menu)
↳ All Tasks
↳ Detach Database…

The detached database should disappear from the list of databases.

2. In specific cases the SQL Server may display error messages in response to attempts to detach a damaged database. At first, ignore these error messages and try to close and re-run the SQL Server Manager. The detached database should disappear from the list.
3. If it was not the case, try to browse tables of the damaged database in order to force the SQL Server to update the database status to "Suspect". Then re-run the SQL Server Manager once again.
4. If all previous steps did not help, stop and re-start the SQL Server instance.

These actions are purposed to detach the damaged database without stopping the whole SQL Server. This step is required, because SQL Server does not allow duplicate names for registered databases regardless of their actual availability.

After the database is detached, it can be restored from a backup image. The restored database can be smoothly re-attached to the SQL Server:

Console Root
↳ Microsoft SQL Servers
↳ *<SQL Server Group>*
↳ *<Server Instance Name>*
↳ Databases
↳ (popup menu)
↳ All Tasks
↳ Attach Database…

3.4 Choosing between Hotbackup and VSS Online Backup Options.

Drive Backup provides two options for snapshot-based online backup in Windows XP, 2003 and Vista. In Windows 2000 and Windows NT4.0, only Hotbackup option is available.

Underlying technologies (Hotbackup and MS VSS) are different by their concepts and features. As regards to backing up SQL Server databases, these options exhibit different levels of operation stability and resulting data consistency. Following considerations can be taken into account when choosing between options:

- Only Microsoft SQL Server 2000 and Microsoft SQL Server 2005 are VSS compliant while earlier versions are not. For earlier versions of Microsoft SQL Server, the difference between online backup options is generally unimportant.
- VSS option allows to create coherent backup of multiple volumes. With Hotbackup option, Drive Backup always performs an asynchronous backup of multiple volumes.
  This feature can be determinative for backing up databases distributed over multiple volumes (regardless of SQL Server version).
- Hotbackup requires less memory and disk resources to operate. For example, latest updates of VSS (for Windows XP and 2003) need at least 300MB of disk space per every created shadow copy (at the moment of snapshot initialization), while Hotbackup will consume disk space only under noticeable disk I/O traffic. It will take 300MB only under high disk load.
- Both VSS and Hotbackup may fail to initialize or fail to maintain a created snapshot under conditions of high I/O traffic on a volume being backed up. VSS requires large amount of free space to maintain a snapshot, while Hotbackup mostly needs high backup performance compared to SQL server's load. The backup performance depends on (a) throughput of archived volumes and backup storage and (b) compression throughput, which in turn depends mostly on CPU and memory speed.
- Hotbackup initialization is more stable in comparison to VSS. As concerns to SQL servers, Hotbackup practically always initializes within predefined time intervals, while VSS may fail under high load of the SQL Server.
- Only latest versions of VSS are stable enough. It is highly recommended to install the latest service pack available and also check for latest hotfixes for Windows XP/2003 related to VSS.

The tests have revealed a general instability of VSS initialization under a "stressed load" of the SQL Server (e.g. under lengthy server's load greater than 1-1.5 thousand modified pages per second). To solve such a problem, either use the Hotbackup option or force checkpoint-ing for all involved databases.

- VSS option minimizes the portion of the transaction log that will be processed at the full database recovery, in same fashion as it is made by checkpoint-ing a database. Hotbackup option does not

optimize the log. The difference comes to light only after an image restoration: a database(s) backed up by the Hotbackup option may require more time to mount (attach) a database because of many transactions should be rolled forward.

The mentioned effect regarding the different roll forwarding periods is common for all non-VSS snapshot technologies. To reduce it, run the SQL Server's management console and set the *Recovery Interval* parameter to a positive value:

<SQL server instance>
↳ Properties
↳ Database Settings
↳ Recovery Interval (min).

The SQL Server will automatically generate checkpoints and thus dynamically reduce the portion of the transaction log to process.

3.5 Examples of Using Online Backup Options

This section demonstrates how to use Paragon Drive Backup for online backup of MS SQL databases and data restoration in case of a database corruption. The methods of retrieval of related database properties and their correlation to backup parameters are illustrated.

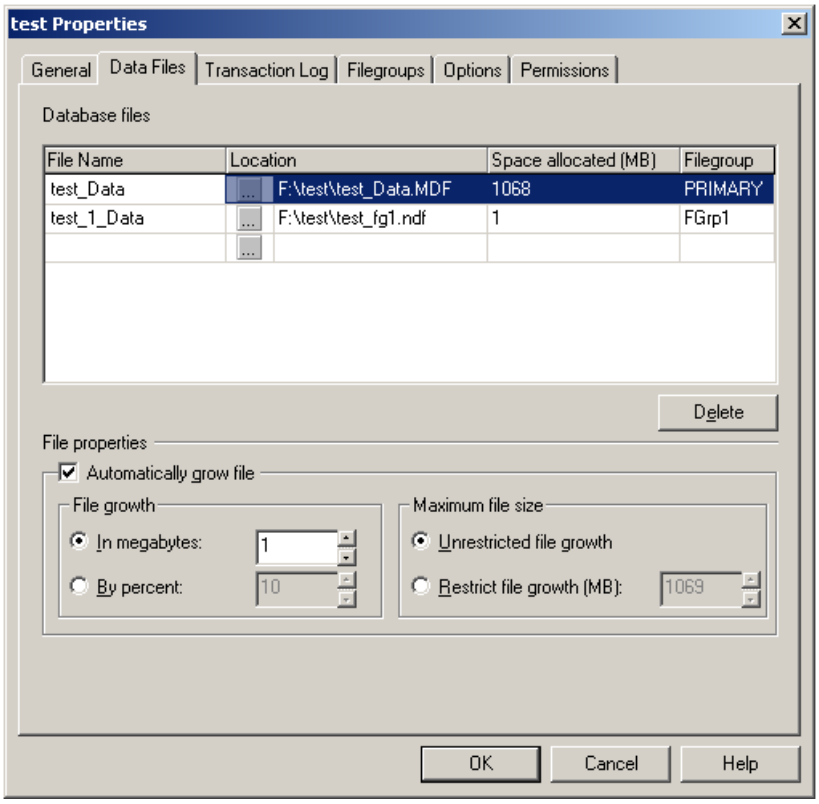
3.5.1 Example 1: Backup and Restore of an Isolated Database

Conditions: the production database (named "test") is placed on the dedicated volume [F:]. All database files (the main file **test_Data.MDF**, the log **test_log.LDF** and the secondary file **test_fg1.NDF**) are located on the volume F:. In practice, such database layouts are applicable only for small databases or under conditions of a tight budget.

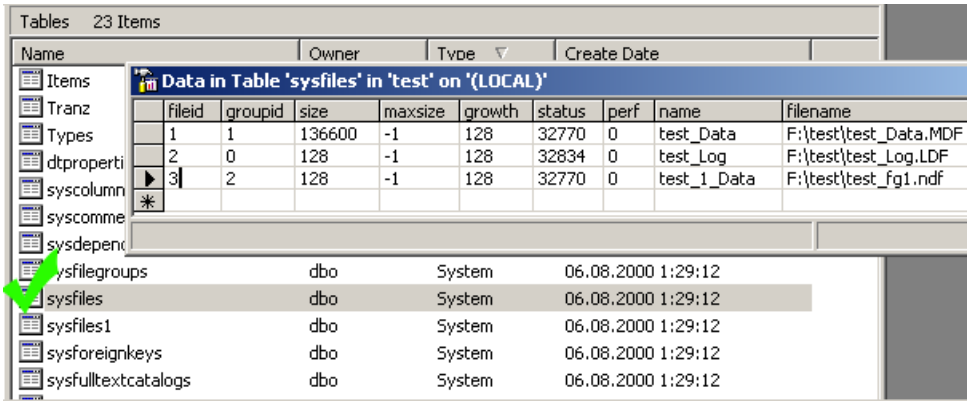
Purpose: backup and restore the "test" database.

This example demonstrates how to retrieve information about location of database files, how to backup the database and how to restore it after a failure.

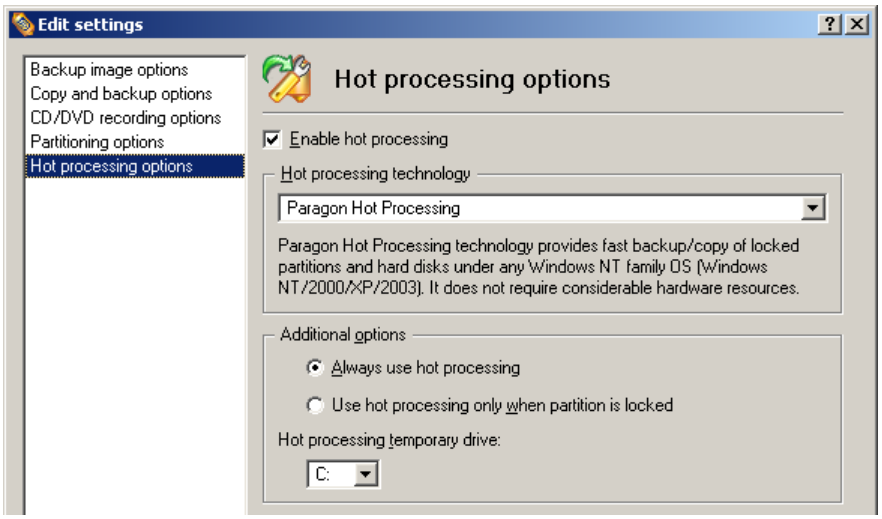
1. Inspect database properties to ensure that all database files are located on same volume. To do this, open the SQL Server Manager console, find the appropriate instance of SQL Server and open the "Databases" category. Find the appropriate database name in the list (in our example, it is the "test" database). Explore properties of this database. Location of files is determined on the "Data files" and "Transaction Log" tabs of the Properties window:



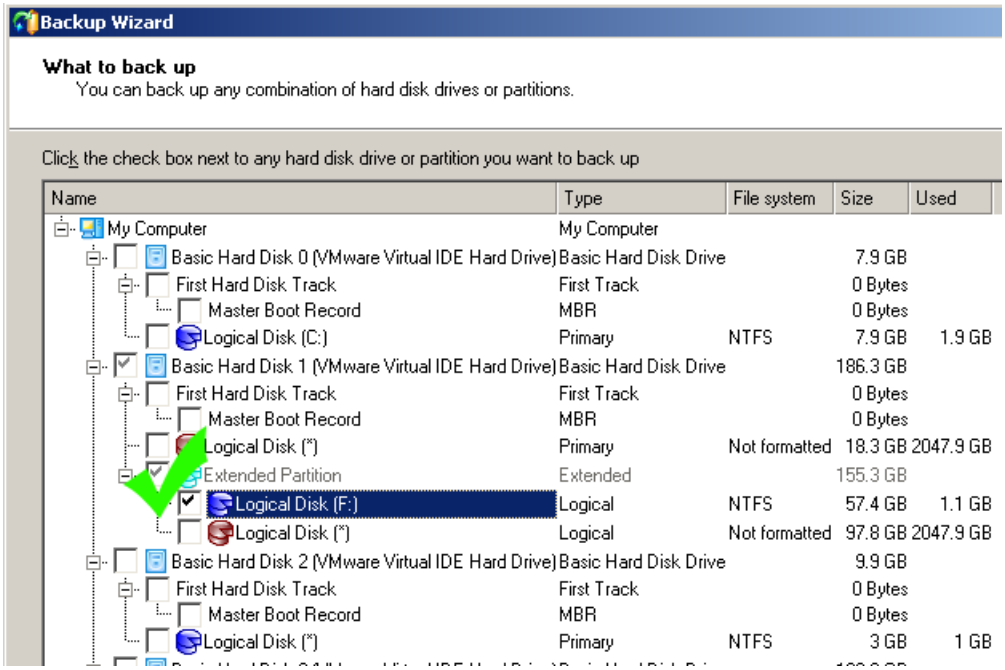
More comprehensive information about all database files can be retrieved from the "sysfiles" table of the selected database. Expand the node of the selected database, click on the "Tables" category. Then go to the right panel and find the "sysfiles" table in the list. Click right mouse button on the table and select the "Open table" → "Return all rows" item in the popup menu. The "sysfiles" table contents will be displayed. You can see file name and current size (expressed in 8KB-pages) and other information for every database file:



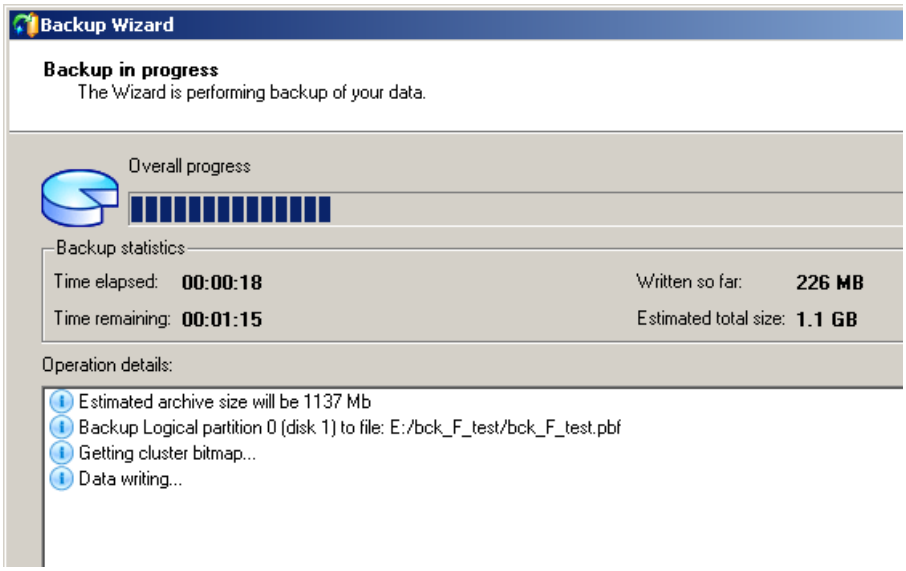
2. Run Drive Backup, open Settings dialog (menu: "File" → "Settings…"). Go to the "Hot processing options" page, enable online backup and choose an online backup technology to implement: Paragon Hotbackup or Microsoft Volume Shadow Copy:



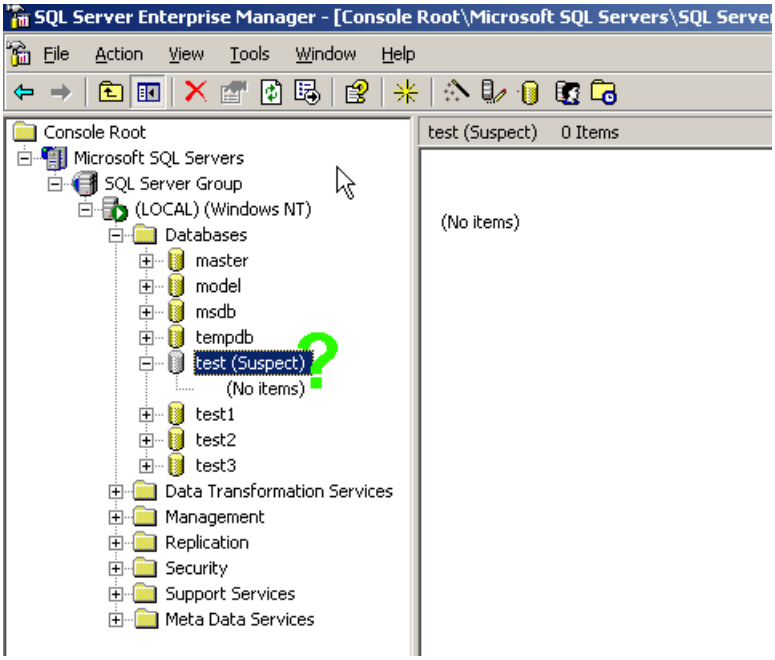
3. Run the Backup Wizard in order to set up the backup operation (no matter should it be a scheduled task or "run once" job). On the "What to back up" page, select the volume F: for backup:



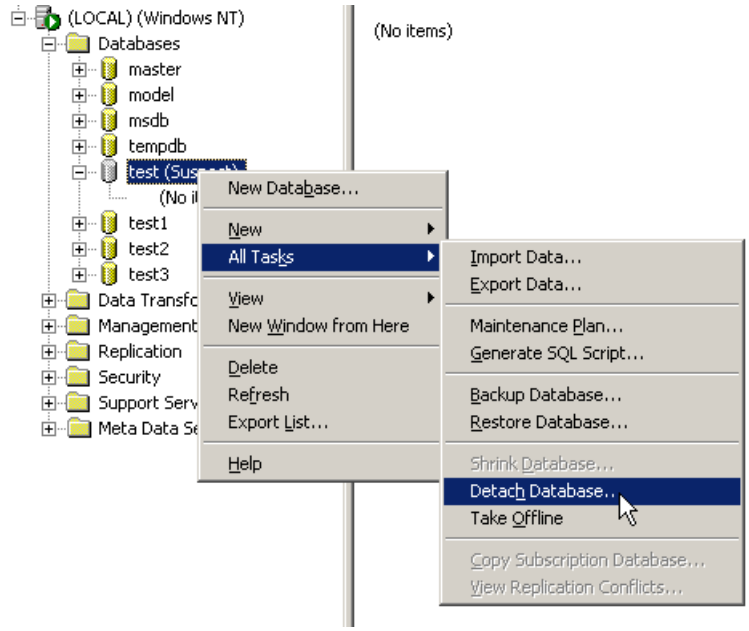
4. The program will make a backup image of the volume F:



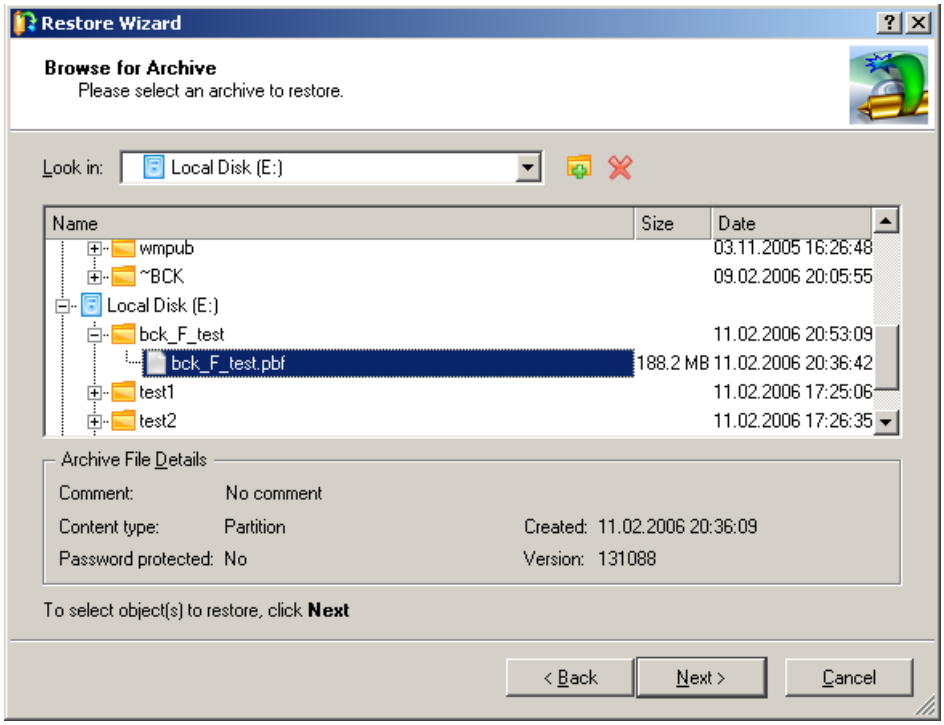
5. Now suppose the "test" database became corrupted (in our example, contents of the **test_Data.MDF** file were distorted, which resulted in system information corruption). The SQL Server cannot attach or use the "test" database. The database features are unavailable:



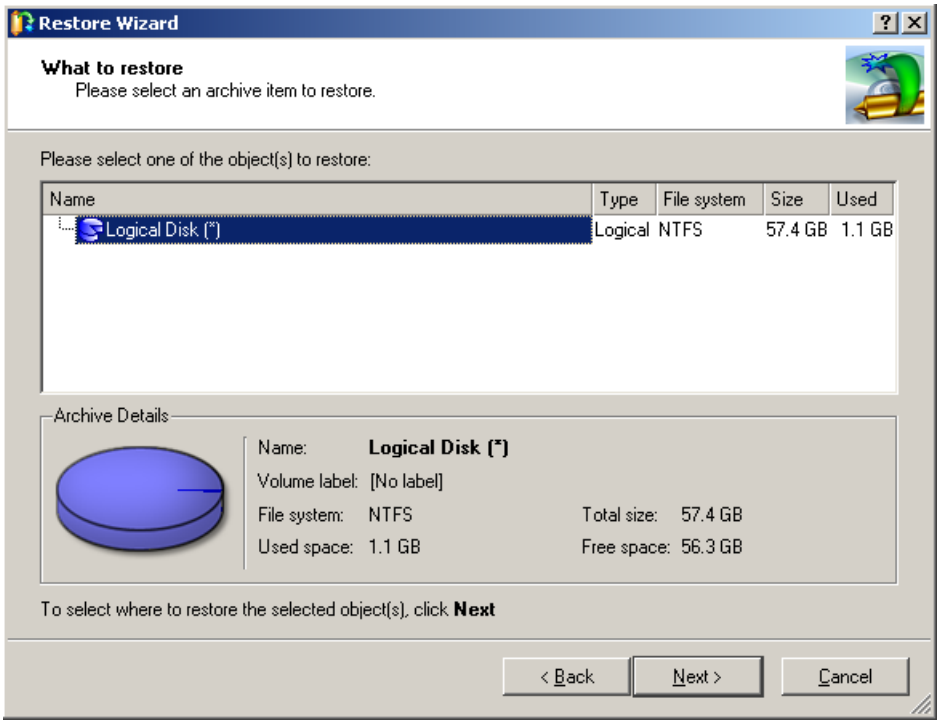
6. The database should be detached from the SQL Server and then restored from an image and then attached to the SQL Server again. To detach the "test" database, click right mouse button on the "test" database in the list and select the "All tasks" → "Detach database…" item in the popup menu:



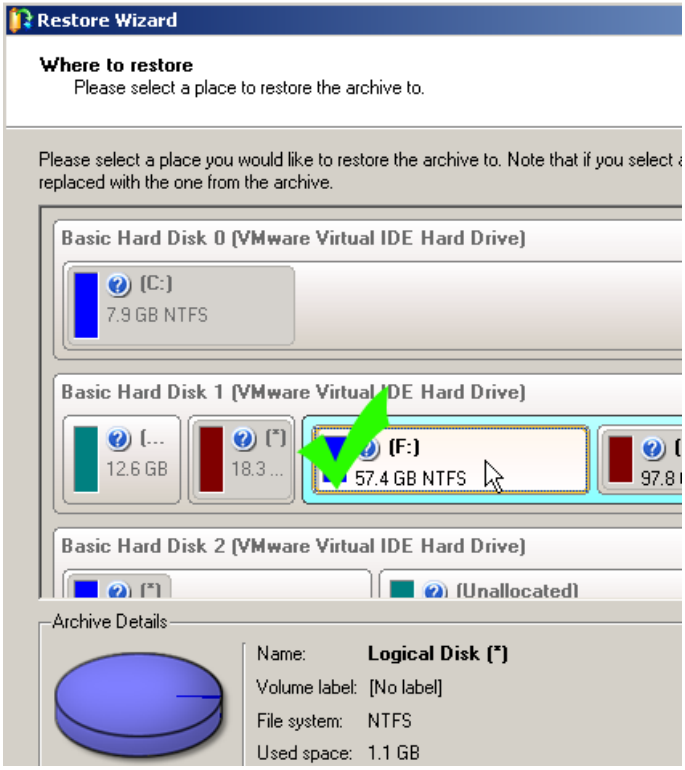
7. We assume there are no other databases and files on the volume F: than the "test" database, which is to be restored. Run Drive Backup and restore the entire volume F: from the appropriate image. To do this, run the Restore Wizard and select appropriate image file (**.PBF**):



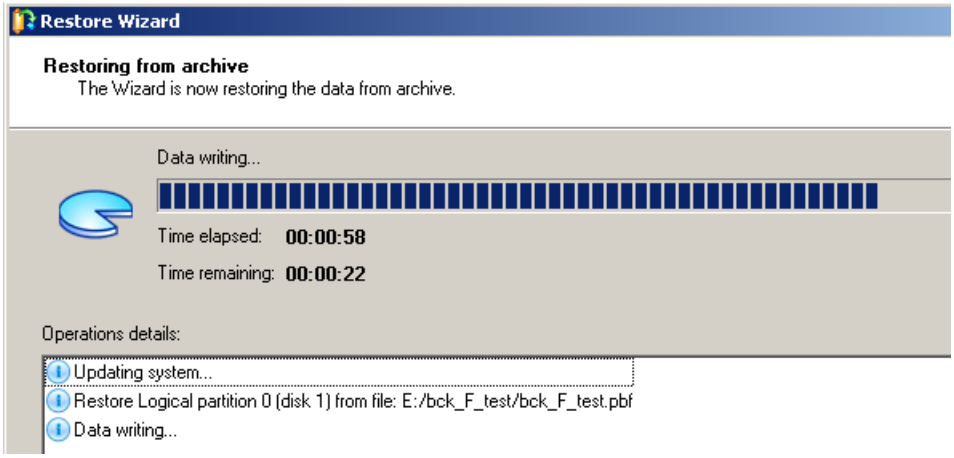
8. Select a volume stored in the image:



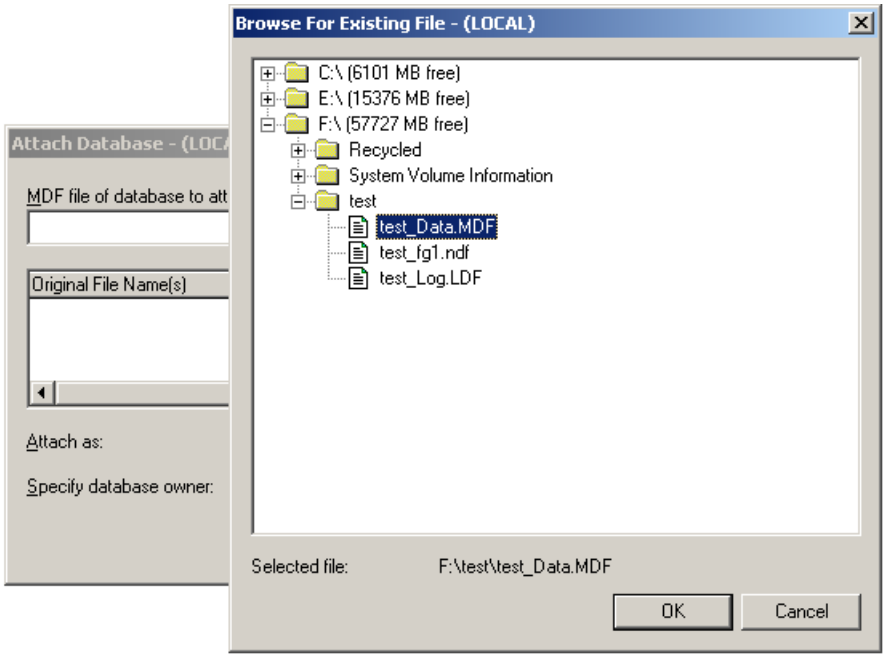
9. Select a volume on a disk where data should be restored:



10. Drive Backup will restore data of the volume:



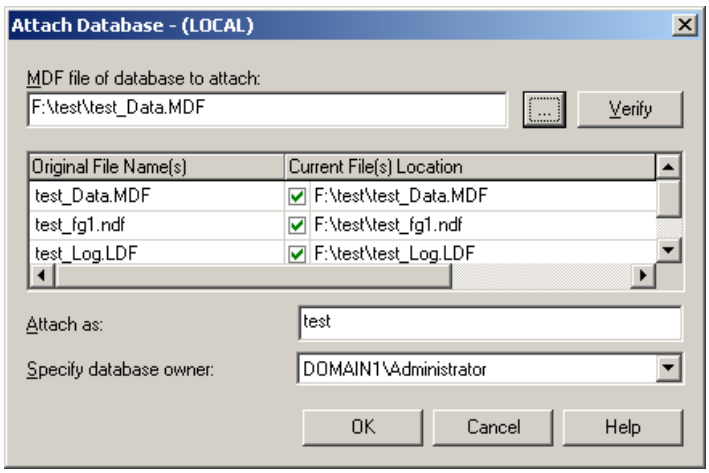
11. Re-attach the "test" database to the SQL Server. To do this, run the SQL Server Manager, find the appropriate SQL Server instance, select the "Databases" item and open the popup menu for it. In the popup menu, select the "Attach database" item. The program will display the "Attach database" dialog. Press the "..." button and find the **.MDF** file of the restored database (**test_Data.MDF** in our example):



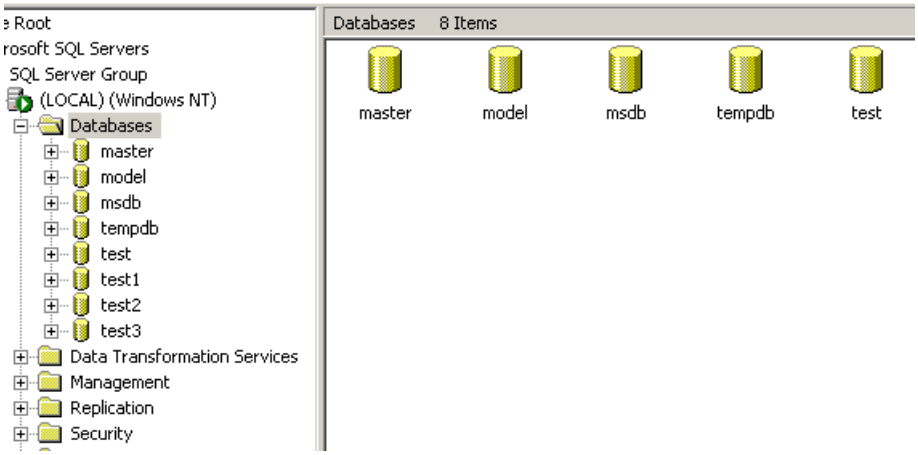
In most cases, SQL Server does not lock files of a corrupted database. So that it really is not required to detach a database prior to a database restoration. However, it may be required later (prior to a database re-attachment). The step 6 can be performed just before this step.

In case the SQL Server rejects to detach and attach the database, first check that the database was restored to a correct location. Then, try to stop and re-start the SQL Server service.

12. Inspect the next window and ensure that all files of the restored database are referred correctly. All database files must be marked with the green checkmark, it indicates that SQL Server has found all database files at their expected location. In this example, all database files must be located in same directory "**F:\test**":



13. The SQL Server will attach the database and update its status. It will become in workable state:



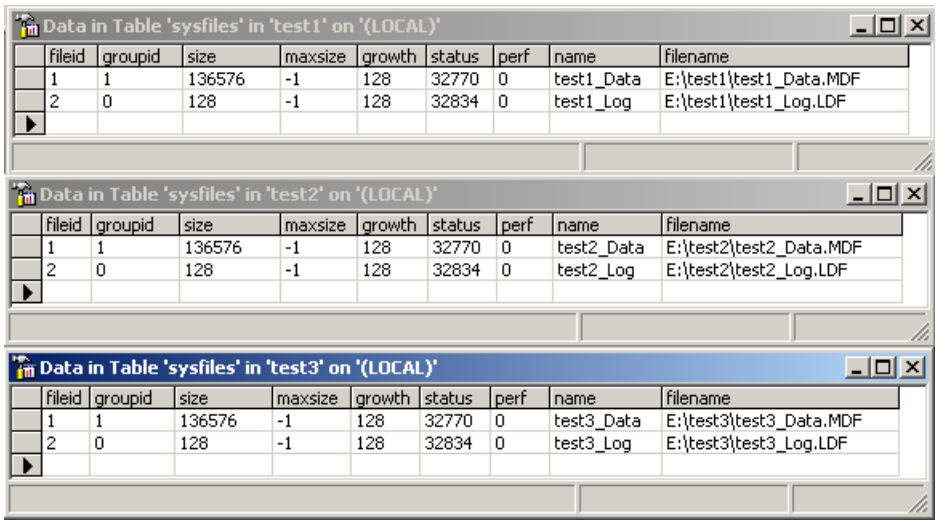
3.5.2 Example 2: Restore of a Single Database from an Image Containing Multiple Databases

Conditions: three production databases ("test1", "test2", "test3") are placed on the dedicated volume [E:]. All their database files are located on the volume E:.

Purpose: backup the "test2" database. Restore the "test2" database without affecting other two databases.

This example demonstrates how to retrieve information about location of database files, how to backup the database and how to restore a database on a file-by-file level.

1. The volume E: contains three databases (test1, test2, test3). You can inspect database properties in the SQL Server Manager. By opening "sysfiles" tables for all databases, you can see all of them at once:



Data in Table 'sysfiles' in 'test1' on '(LOCAL)'

fileid	groupid	size	maxsize	growth	status	perf	name	filename
1	1	136576	-1	128	32770	0	test1_Data	E:\test1\test1_Data.MDF
2	0	128	-1	128	32834	0	test1_Log	E:\test1\test1_Log.LDF

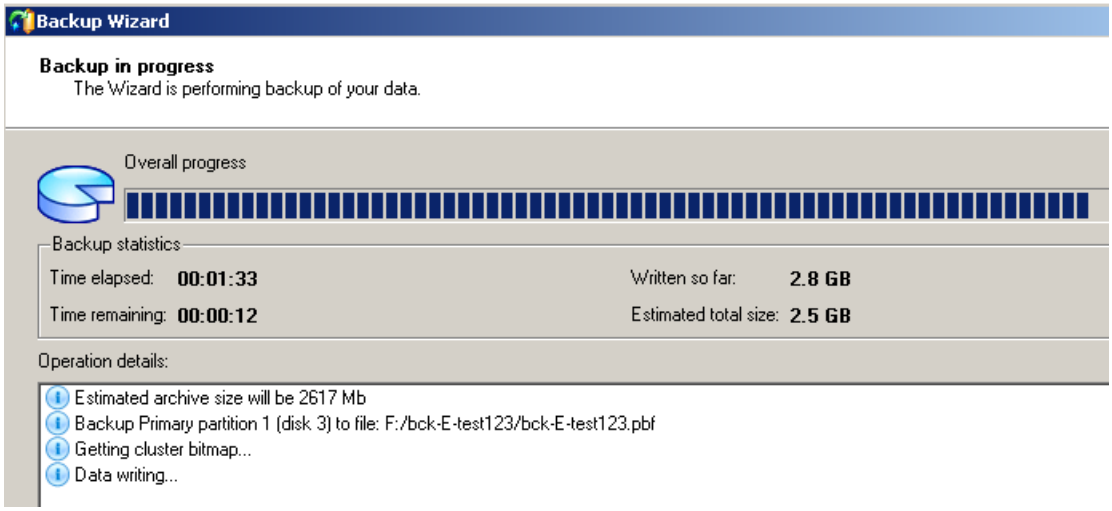
Data in Table 'sysfiles' in 'test2' on '(LOCAL)'

fileid	groupid	size	maxsize	growth	status	perf	name	filename
1	1	136576	-1	128	32770	0	test2_Data	E:\test2\test2_Data.MDF
2	0	128	-1	128	32834	0	test2_Log	E:\test2\test2_Log.LDF

Data in Table 'sysfiles' in 'test3' on '(LOCAL)'

fileid	groupid	size	maxsize	growth	status	perf	name	filename
1	1	136576	-1	128	32770	0	test3_Data	E:\test3\test3_Data.MDF
2	0	128	-1	128	32834	0	test3_Log	E:\test3\test3_Log.LDF

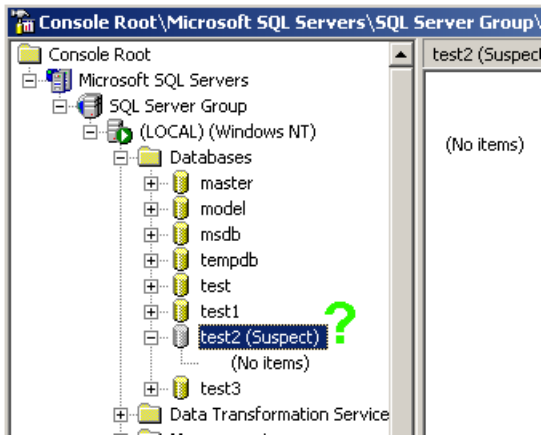
2. The entire volume E: should be backed up in online mode. Any of online backup options can be used for that purpose (either HotBackup or MS VSS):



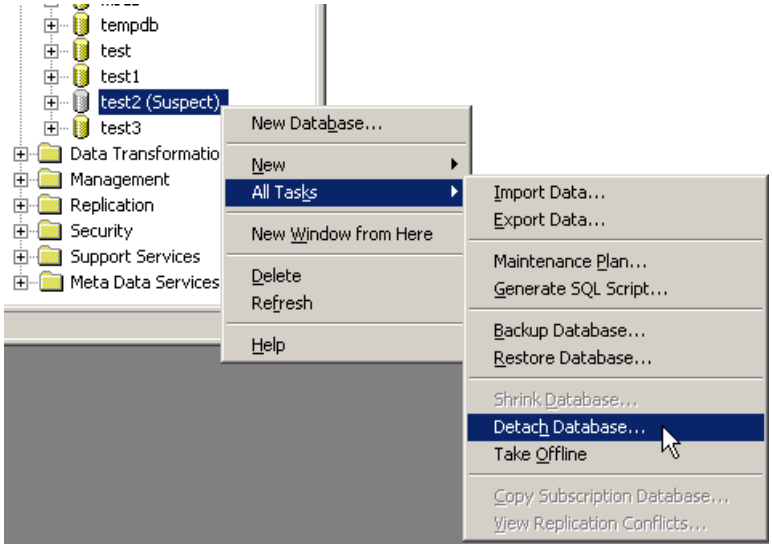
Performance:

- virtual machine system: P4 1.5 GHz, 512 MB RAM
- operating system: Windows Server 2003, Domain Controller
- virtual disk max.thoughput: 35 MB/sec
- average backup throughput: 26MB/sec, that is >90GB/hour
  (~74% of max.throughput)
  (it is 12× backup speed via SQL server API)
- final compression rate ("Best" level): ~ 1:6 (526MB image against 3.1GB original data)

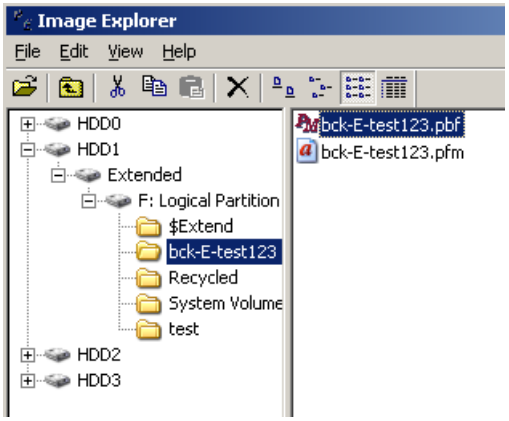
3. The "test2" database became corrupted (in this example, contents of the **test2_Data.MDF** file were distorted, which resulted in system information corruption). The SQL Server cannot attach or use the "test" database. The database features are unavailable:



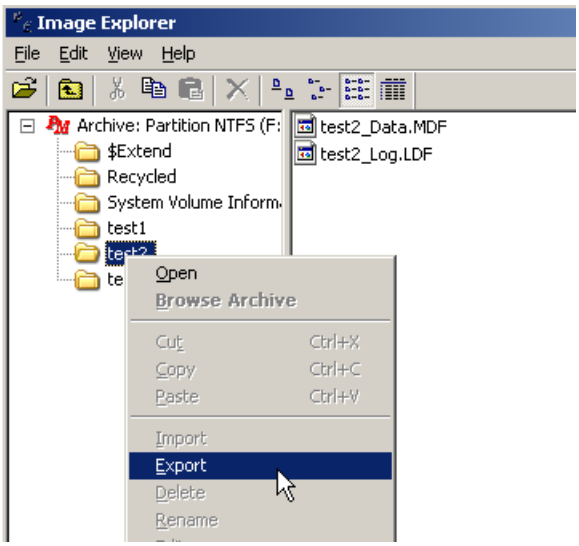
4. The database should be detached from the SQL Server and then restored from an image and then attached to the SQL Server again. To detach the "test2" database, click right mouse button on the "test2" database in the list and select the "All tasks" → "Detach database…" item in the popup menu:



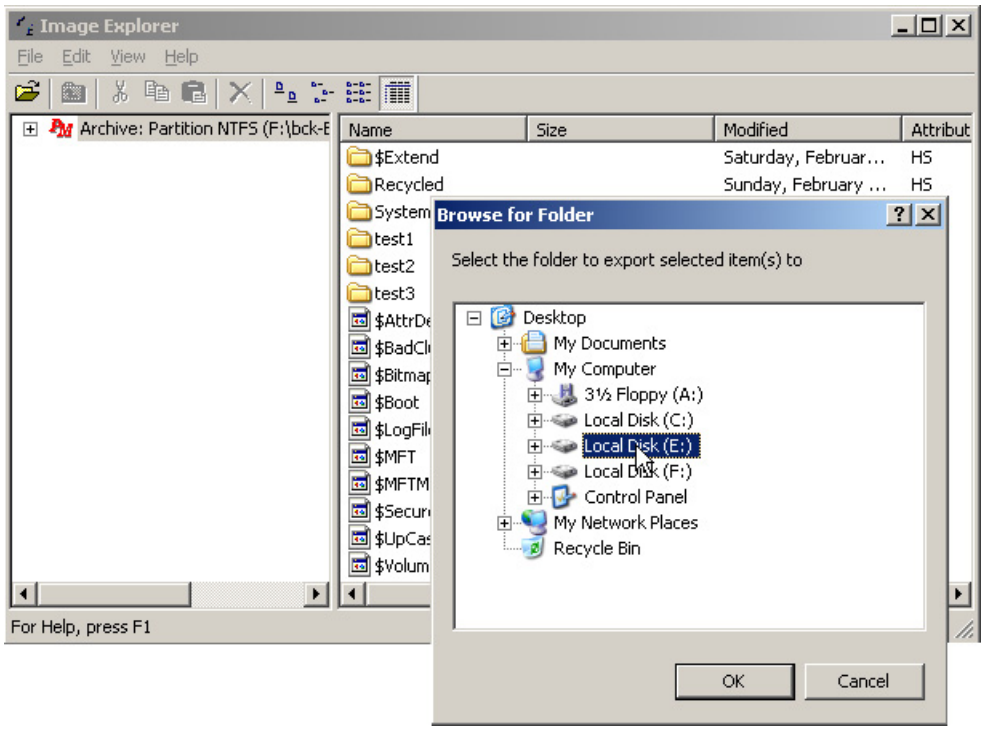
5. You should not restore the whole volume E: from the image, because it will back-off all three databases, including undamaged "test1" and "test3"! To avoid this trouble, you should restore the "test2" database in the file-by-file mode. You should open the image in the Image Explorer and manually copy files to their original location.
   Run Image Explorer and find the backup image of the volume E:. Open the image by double-clicking on the appropriate PBF-file. The Image Explorer will display contents of the image.



6. Now you should extract database files. In our example, the whole "test2" database was located in a single directory (**E:\test2**). So that the whole directory "test2" can be exported from the image. Open the popup menu for the directory stored in the image and select the "Export" menu item.

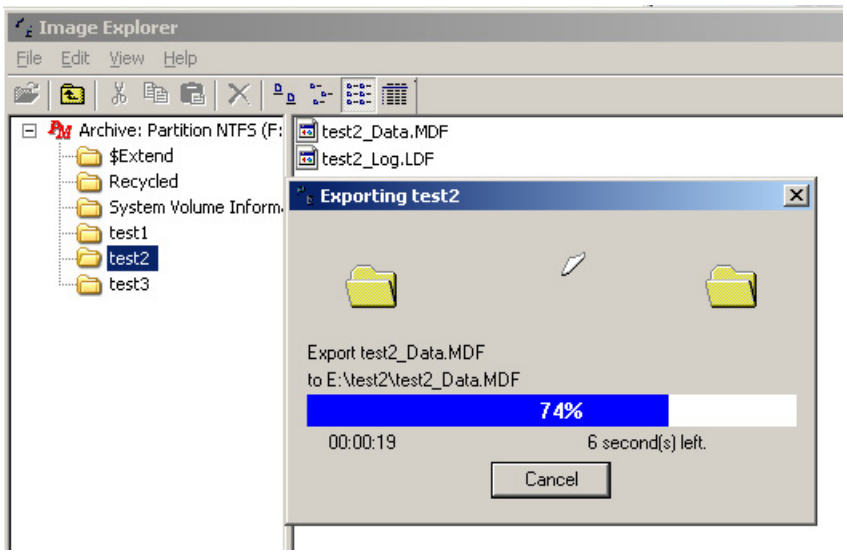


7. Now choose a **parent directory** where the "test2" directory should be placed. In this example, the database was originally located in the **E:\test2** directory. So that you should select the "E:\" directory as a target (to restore the database to its original place).

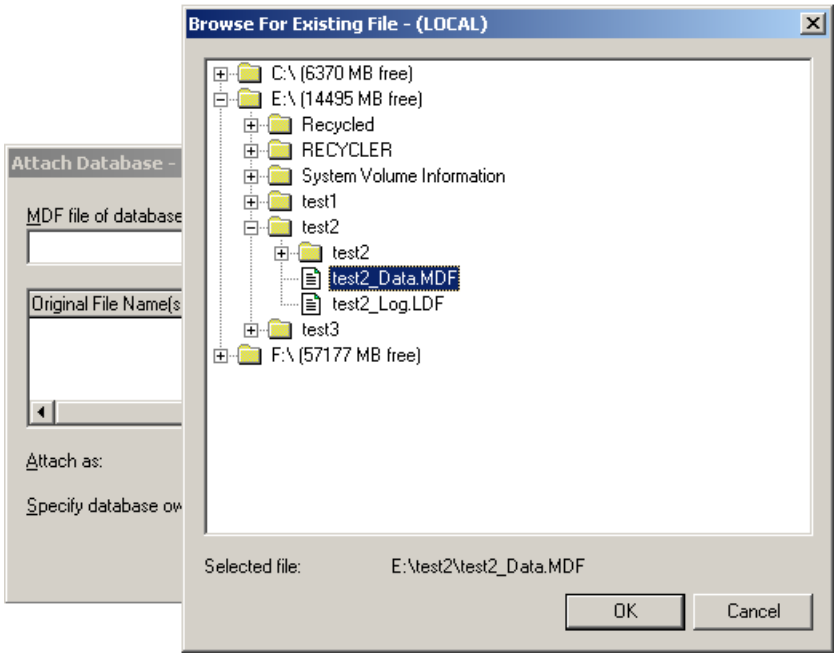


You can also restore a database to a new location. In this case, take care of correct filenames for all database files when attaching the restored database.

8. Image Explorer extracts selected files from a volume image:



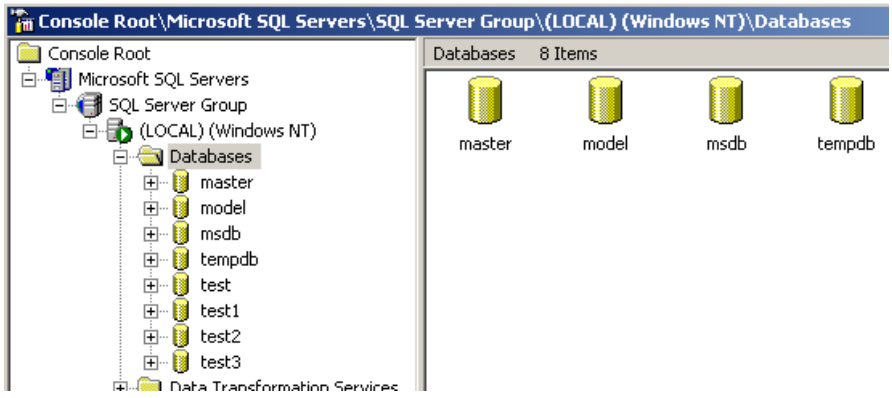
9. Re-attach the "test2" database to the SQL Server. To do this, run the SQL Server Manager, find the appropriate SQL Server instance, select the "Databases" item and open the popup menu for it. In the popup menu, select the "Attach database" item. The program will display the "Attach database" dialog. Press the "…" button and find the **.MDF** file of the restored database (**test2_Data.MDF**):



In most cases, SQL Server does not lock files of a corrupted database. So that it really is not required to detach a database prior to a database restoration. However, it may be required later (prior to a database re-attachment). The step 4 can be performed just before this one.

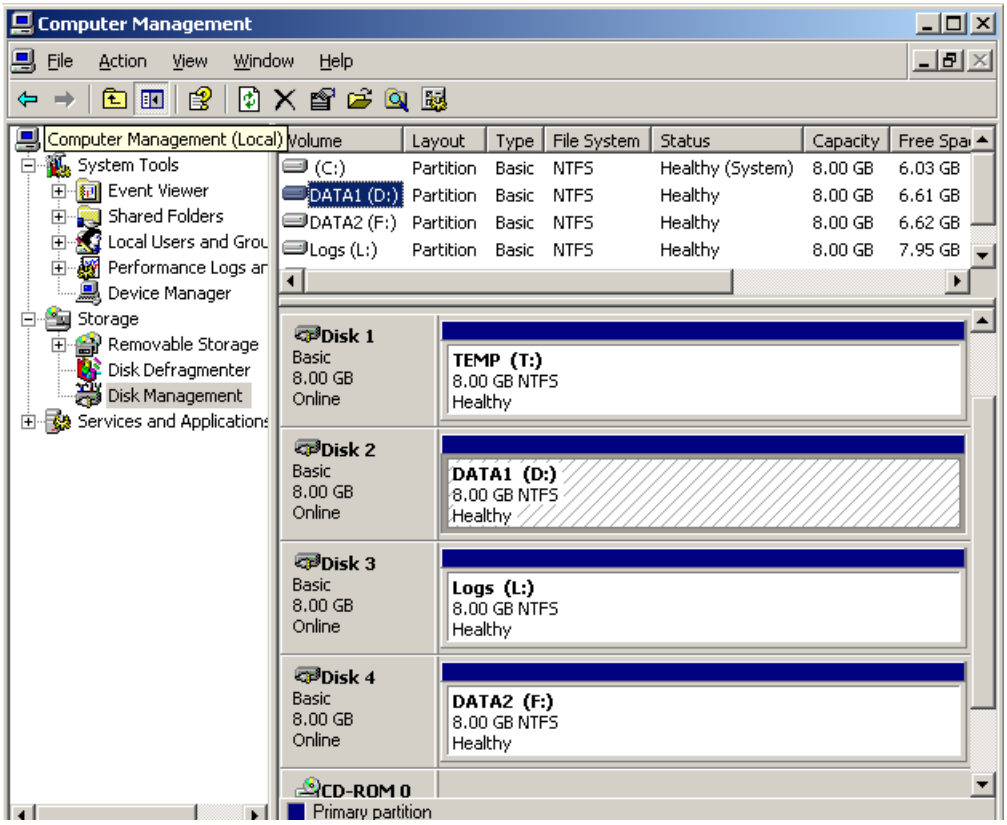
In case the SQL Server rejects to detach and attach the database, first check that the database was restored to a correct location. Then, try to stop and re-start the SQL Server service.

10. The database is now in workable state



3.5.3 Example 3: Backup and Restore of a Single Database Distributed over multiple volumes

Conditions: the "Test" production database is placed on multiple volumes and is configured in accordance with Microsoft's recommendations for database performance acceleration. The database transaction log files are placed on the dedicated volume (L:). The data files are grouped in two file groups ("PRIMARY" and "SECOND"). The "SECOND" file group is active and includes all production tables; it consists of two database files, which are placed on different volumes. The "tempdb" database is isolated from production databases. And finally, all mentioned volumes are located on different hard disks:

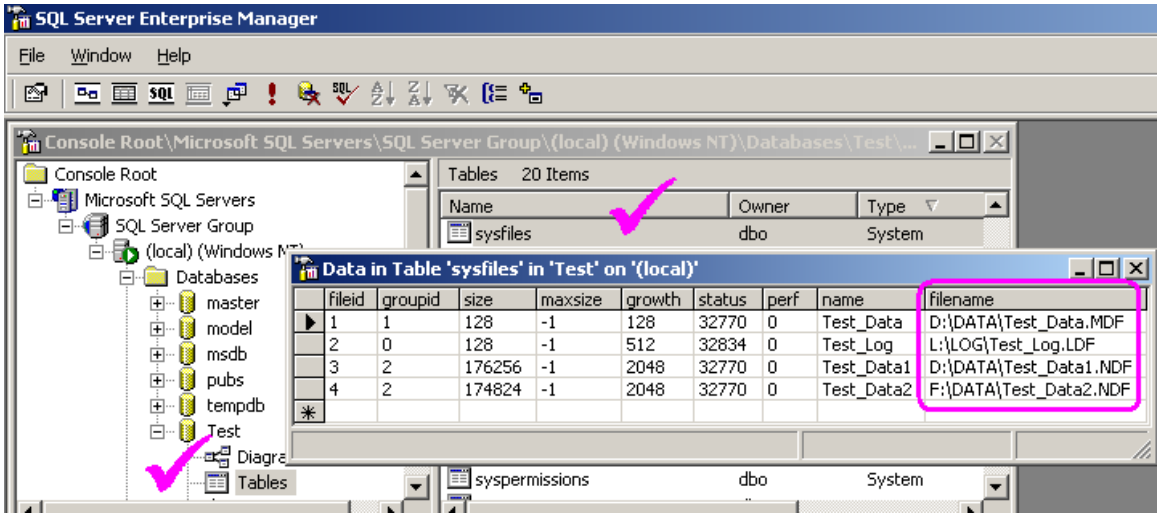


In case of such database layout, MS SQL Server supports striping of read and write operations for production tables. It allows to significantly accelerate database throughput due to parallel execution of I/O requests.

Purpose: backup and restore the "Test" database.

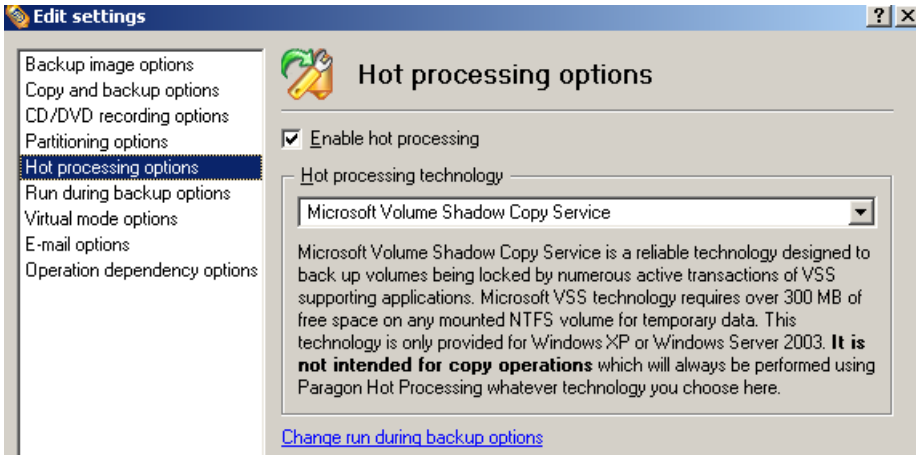
This example demonstrates how to backup distributed databases and how to deal with backup images of multiple volumes.

1. Inspect database properties in the SQL Server Manager in order to get know names of volumes that contain database files: expand the "Test" database in the tree view, select the "Tables" category, then select the "sysfiles" table in the right panel. Then click right mouse button on the table and select the "Open table" → "Return all rows" item in the popup menu.

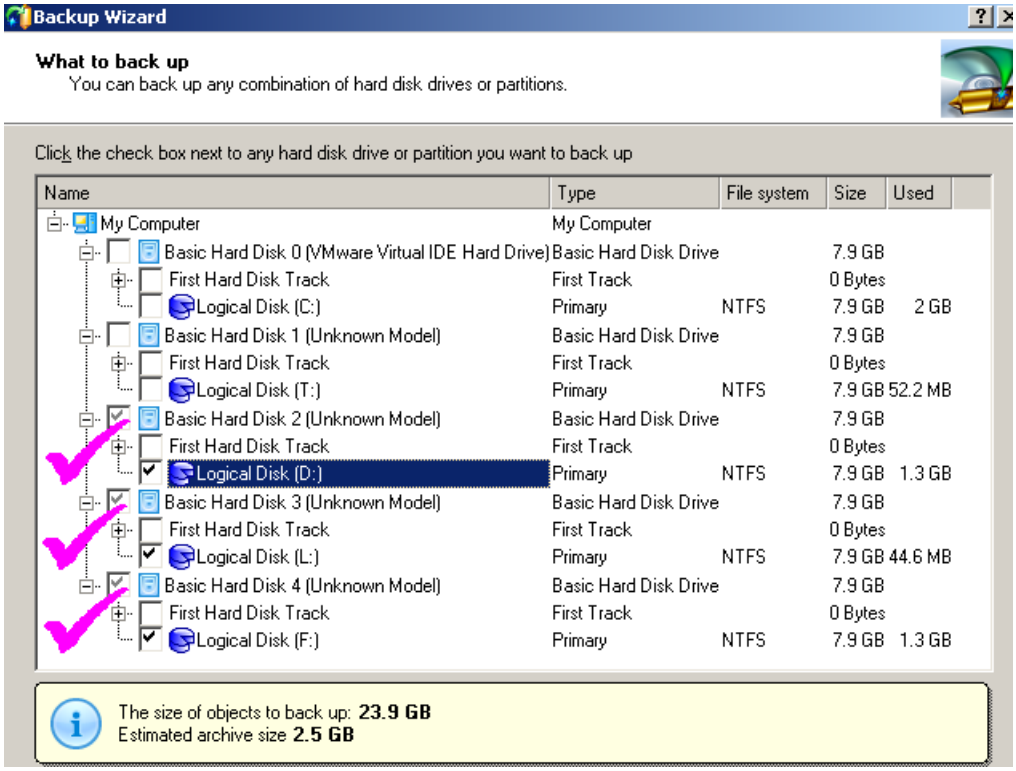


In this example, the "Test" database is spread over three volumes (D:, F: and L:).

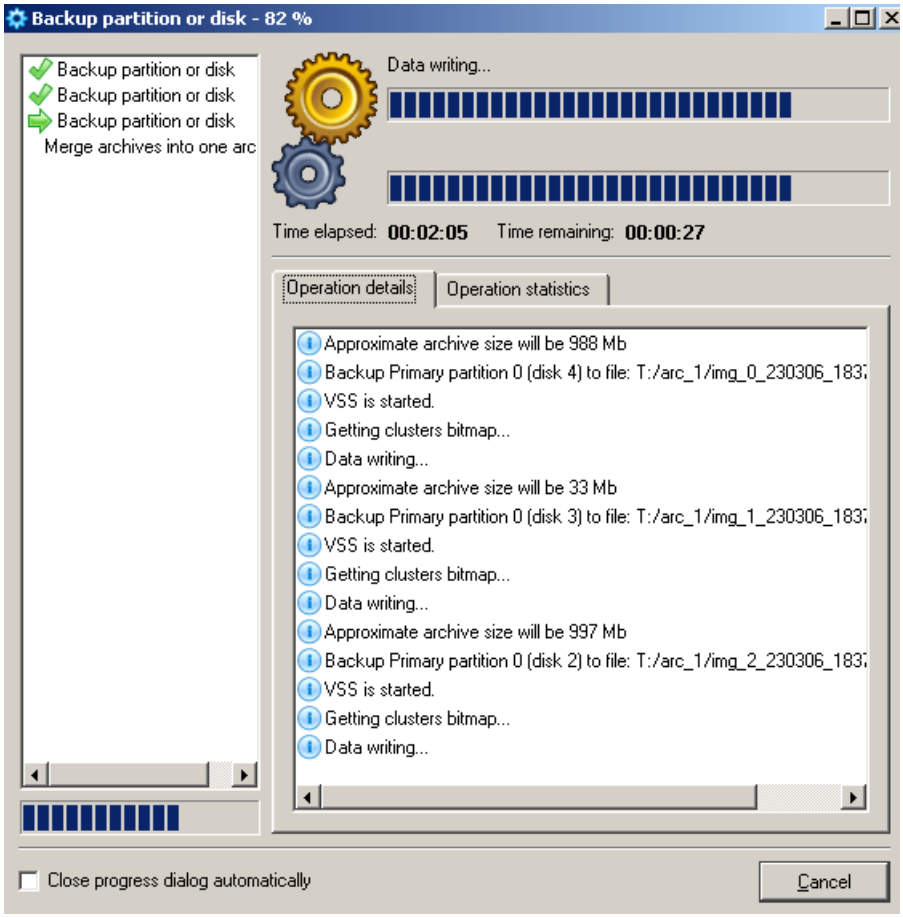
2. Run Drive Backup, open Settings dialog (menu: "File" → "Settings…"). Go to the "Hot processing options" page, enable online backup and choose the " Microsoft Volume Shadow Copy " online backup technology:



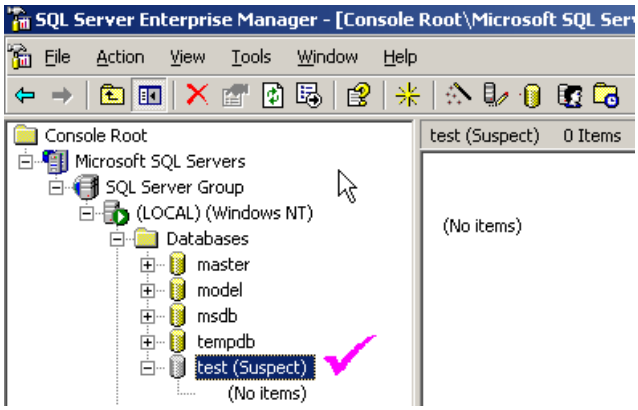
3. Run the Backup Wizard in order to set up the backup operation (no matter should it be a scheduled task or "run once" job). On the "What to back up" page, select the three volumes (D:, F: and L:) for backup:



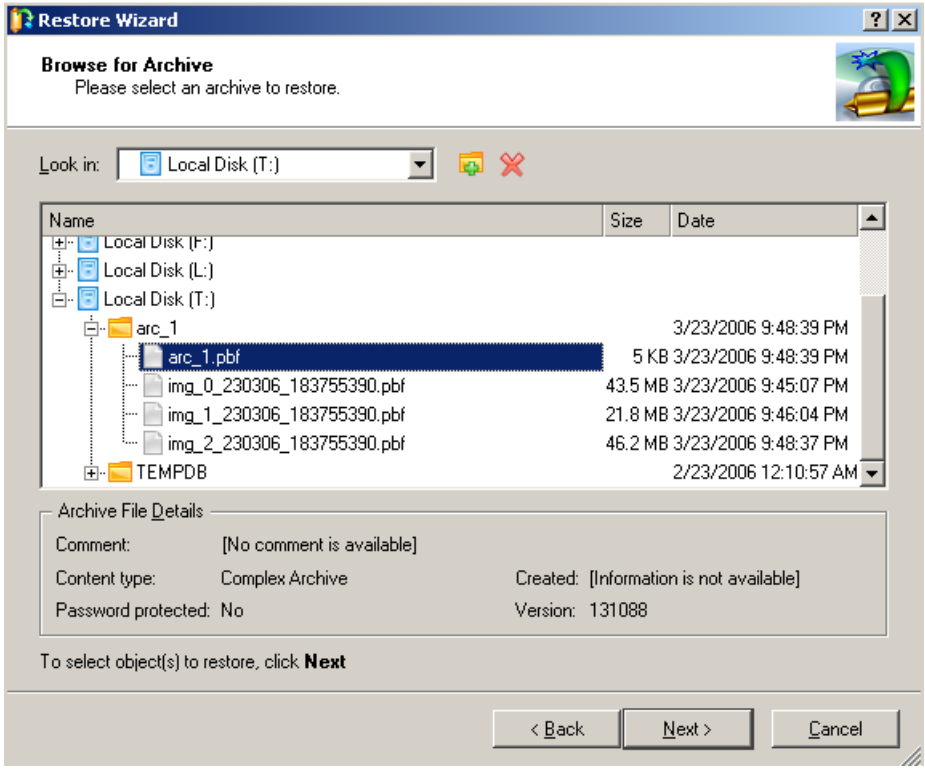
4. The program will make a backup image of three volumes:



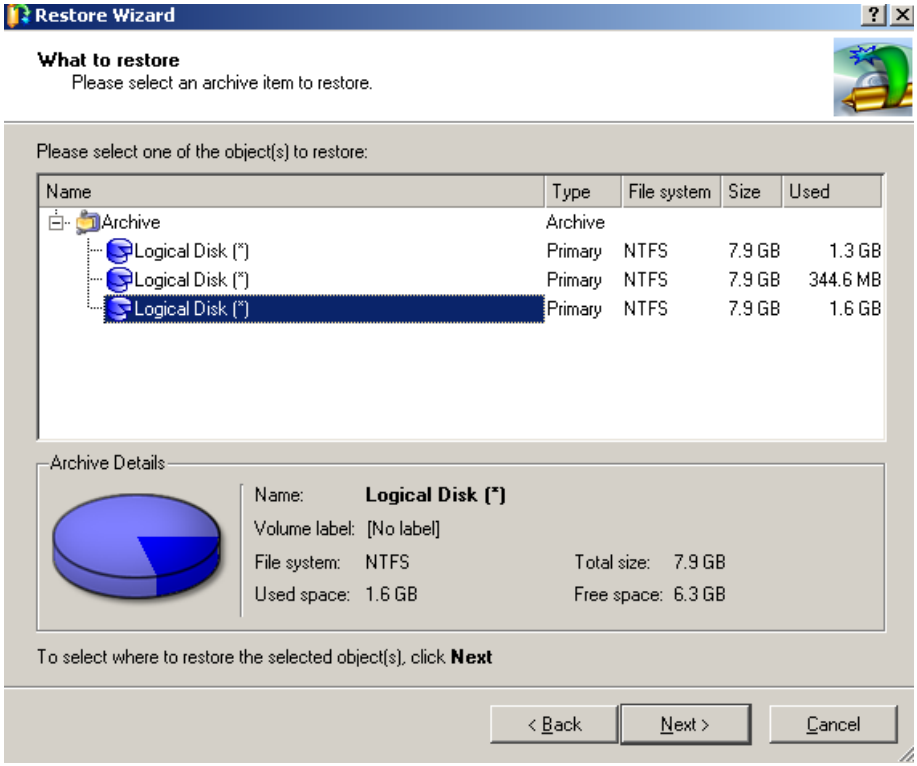
5. Now suppose the "test" database became corrupted. The SQL Server cannot attach or use the "test" database. The database features are unavailable:



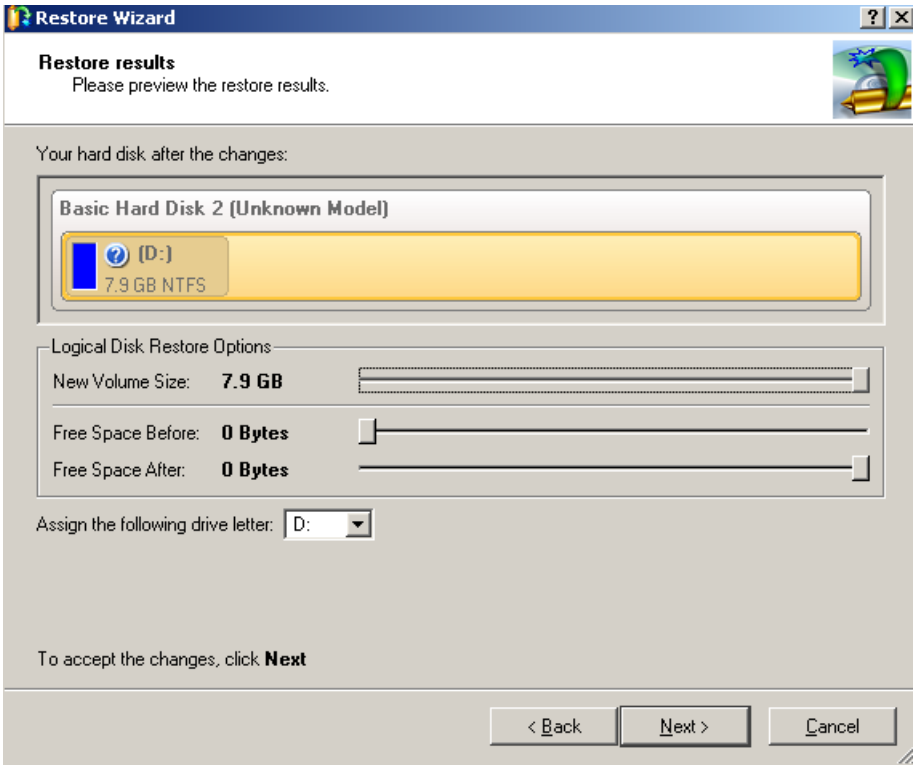
6. Detach the corrupted "Test" database from the SQL Server: click right mouse button on the "test" database in the list and select the "All tasks" → "Detach database…" item in the popup menu.
7. Let us assume there are no other databases and files on volumes D:, F: and L: other than the "Test" database files, which is to be restored. Under these conditions, a volume-level restore can be applied to the image data. Start Drive Backup, run the Restore Wizard and select the previously created backup image of volumes D:, F: and L:.



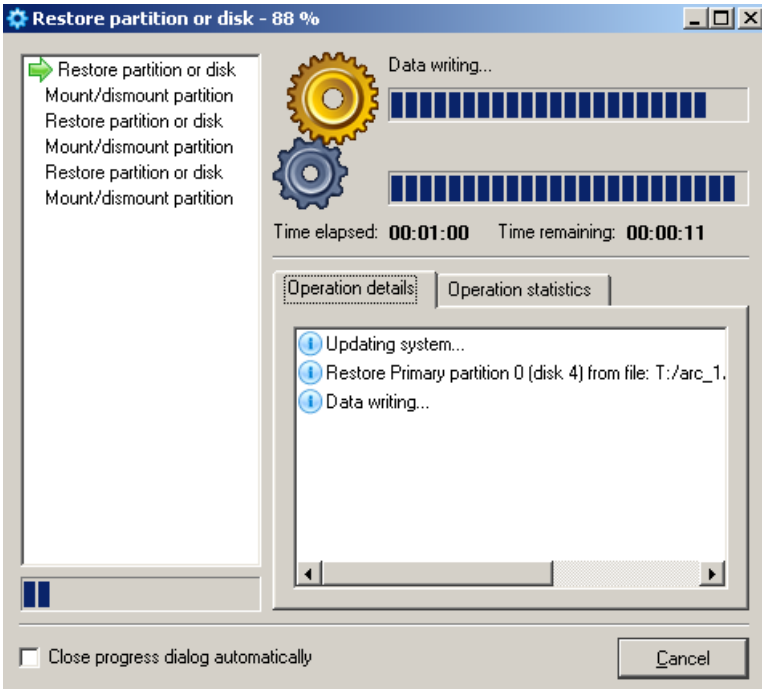
8. The program lists volumes stored in the backup image. You should select them one-by-one and restore data from each stored image.



9. For each restored volume, select a target where data should be restored. You are allowed to choose a location and size of a partition to be restored and a drive letter for the appropriate image. By default, the program suggests to restore the partition to its original place and a drive letter remains unchanged.



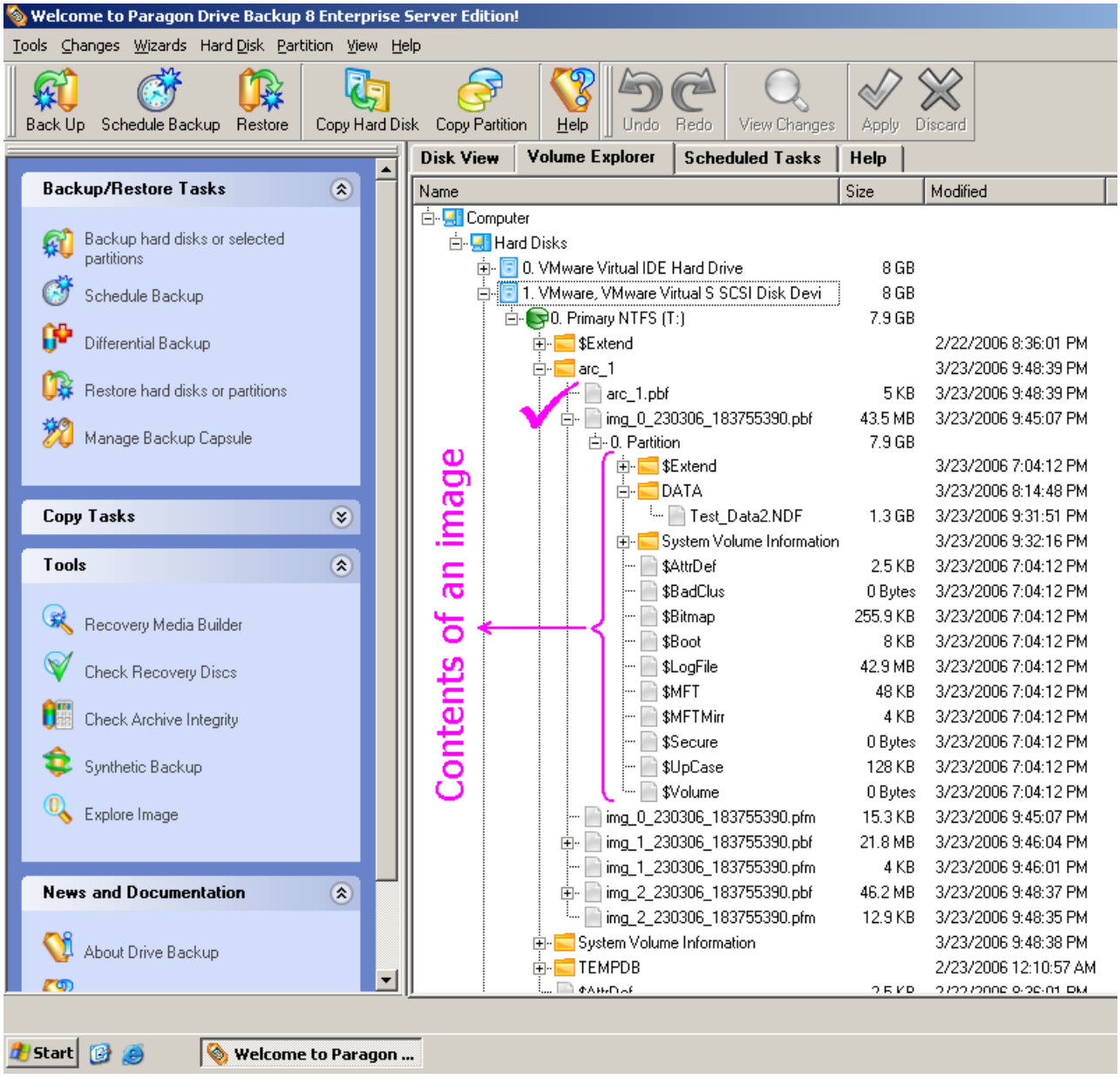
10. After all restore operations are defined, invoke the physical execution of the restore operation.



11. Re-attach the "Test" database to the SQL Server. Run the SQL Server Manager and select:
    Console Root
    ↳ Microsoft SQL Servers
    ↳ *<SQL Server Group>*
    ↳ *<Server Instance Name>*
    ↳ Databases
    ↳ (popup menu)
    ↳ All Tasks
    ↳ Attach Database…

The program will display the "Attach database" dialog. Press the "…" button and find the **.MDF** file of the restored database (in our example, it is the "**Test_Data.MDF**" file).

This example describes the volume-level data restoration, when all contents of the volume are "rolled back" to the state corresponding the moment of backup execution. It is the only acceptable restoration type in case of filesystem damage or disk hardware malfunction. However in some situations the volume-level restoration may lead to an undesired loss of most recent data modifications, such as corruption of one file or a database user's error. A file-level restoration can be implemented in these cases. The Image Explorer utility or the built-in Volume Explorer tool can be used for restoration of selected files from a backup image:

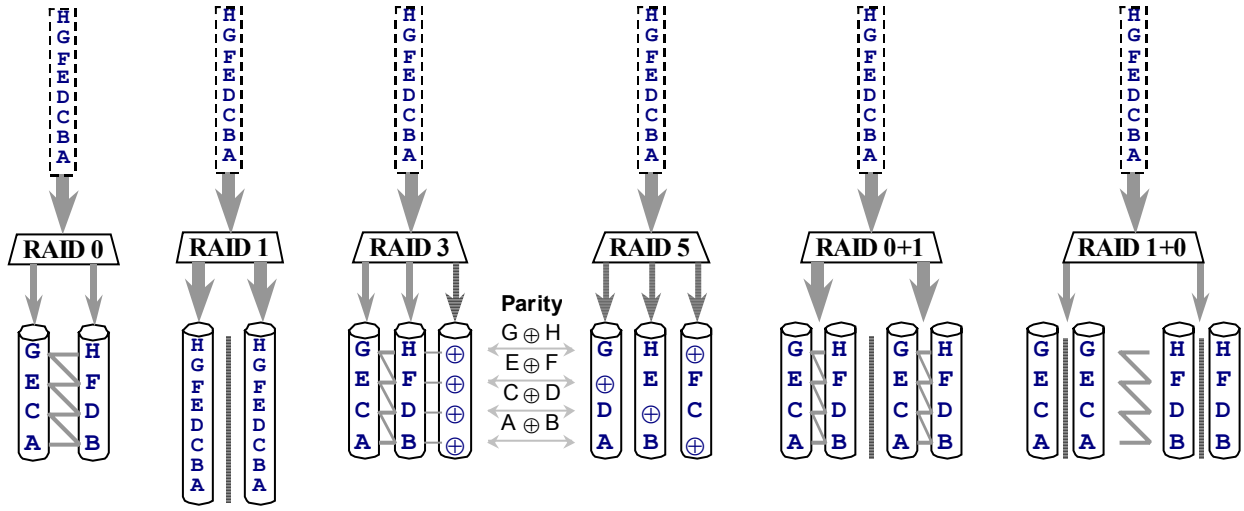


4 Appendix

4.1 RAID Levels

Use of RAIDs is the most effective way to increase throughput and provide a hardware-level fault tolerance of a disk subsystem. Striped RAID sets provide highest read/write performance that is found nearly a multiple of a single disk performance.

RAID level	Brief description	Amount of disks	Size factor	Performance factor	Fault tolerance
0	Striped Disk Array	N ≥ 2	× N	× N	No
1	Mirrored Disk Array	M ≥ 2	× 1	× 1	Yes
0+1	Mirroring of striped segments (RAID 1 over RAID 0)	M•N ≥ 4 M ≥ 2, N ≥ 2	× N	× N	Yes
3	Striped Disk Array with Isolated Parity	N+1 ≥ 3	× N	× N (for reads) × 1 (for writes)	Yes
5	Striped Disk Array with Distributed Parity	N+1 ≥ 3	× N	× N (for reads) × 1 (for writes)	Yes
10	Striping of mirrored segments (RAID 0 over RAID 1)	N•M ≥ 4 N ≥ 2, M ≥ 2	× N	× N	Yes



A more detailed characteristic of various RAID levels can be found in the Internet.