



NETAPP TECHNICAL REPORT

## Snapmanager 5.0 for SQL Server: Best Practices Guide

Sourav Chakraborty, NetApp  
January 2009 | TR-3431

### EXECUTIVE SUMMARY

Innovative NetApp® technologies enable organizations to extract greater benefits out of their Microsoft SQL Server® deployments in the area of backup management of SQL Server deployments. The various technologies empower the database administrator to design a robust backup management strategy to protect the organization's SQL Server resources.

NetApp provides industry-leading solutions in the areas of complete data protection; thin storage provisioning; data de-duplication; file-based backups and instantaneous SQL Server database backups and restores.

The intent of this document is to discuss the best practices that empower the user to maximize the benefits from using SnapManager for SQL Server (SMSQL).

## TABLE OF CONTENTS

<b>1</b>	<b>INTRODUCTION .....</b>	<b>3</b>
1.1	PURPOSE AND SCOPE .....	3
1.2	INTENDED AUDIENCE .....	3
<b>2</b>	<b>SNAPMANAGER FOR SQL SERVER SIMPLIFIES SQL SERVER DATABASE MANAGEMENT .....</b>	<b>3</b>
2.1	SMSQL ARCHITECTURE: COMPONENTS USED BY SNAPMANAGER 5.0 FOR SQL SERVER .....	4
2.2	SMSQL 5.0 CONCEPTS .....	5
<b>3</b>	<b>INSTALLING &amp; CONFIGURING SMSQL 5.0 ON WINDOWS SERVER .....</b>	<b>6</b>
3.1	SIZING FOR SMSQL DATA-STORES.....	6
3.2	CONFIGURING SMSQL 5.0 – WHAT’S NEW .....	8
<b>4</b>	<b>LAYING OUT SQL SERVER DATABASES ON NETAPP STORAGE .....</b>	<b>12</b>
4.1	LIMITATIONS OF DATABASE LAYOUT .....	12
4.2	FRACTIONAL SPACE RESERVE SETTINGS .....	12
<b>5</b>	<b>BACKUP &amp; RESTORE WITH SMSQL 5.0 .....</b>	<b>14</b>
5.1	HOW DOES BACKUP WORK .....	14
5.2	BACKUP BEST PRACTICES .....	15
5.3	COMPRESSION OF TRANSACTION LOG BACKUP (SQL SERVER 2008).....	16
5.4	PROTECTING YOUR DATABASES.....	17
<b>6</b>	<b>DATABASE CLONING WITH SMSQL 5.0.....</b>	<b>18</b>
6.1	HOW DOES DATABASE CLONING WORK .....	19
6.2	DATABASE CLONING BEST PRACTICES .....	19
<b>7</b>	<b>DATASET INTEGRATION WITH SMSQL 5.0 .....</b>	<b>20</b>
<b>8</b>	<b>REMOTE BACKUP VERIFICATION WITH SMSQL 5.0 .....</b>	<b>27</b>
8.1	VERIFICATION OF SNAPMIRROR SECONDARIES.....	27
<b>9</b>	<b>SUMMARY .....</b>	<b>29</b>
	<b>APPENDIX A .....</b>	<b>30</b>
	<b>REFERENCES .....</b>	<b>31</b>

## 1 INTRODUCTION

In today's data-driven enterprise, business-critical SQL Server databases must be operational around the clock to facilitate decision making, e-commerce, and a myriad of other business processes. Rapid increases in data volume and database demands make it increasingly difficult to provide availability and protection of valuable data assets. To succeed, SQL Server database administrators need tools that will empower them to take regular backups with minimal impact, perform quick restores and recoveries, and create non-disruptive clones for development and test.

NetApp SnapManager for SQL Server (SMSQL) automates and simplifies the manual and time-consuming processes associated with the backup, recovery, and cloning of SQL Server databases. SnapManager leverages NetApp technologies like Snapshot, SnapRestore, and FlexClone while integrating with the latest SQL Server releases. SnapManager also integrates seamlessly with SQL Server's database management components such as Virtual Device Interface (VDI), SQL Server storage engine etc. to provide efficient backup and restore management for SQL Server databases. SMSQL supports both FC and iSCSI connections. In addition, the fast backup cycles, relatively small downtimes for database restores and SMSQL's ability to extend crucial storage management functionalities to SQL Server databases allow IT organizations to scale their storage infrastructure, meet increasingly stringent SLA commitments, and improve the productivity of database and storage administrators across the enterprise.

### 1.1 PURPOSE AND SCOPE

The objective of this paper is to provide best practices for deploying and using NetApp SnapManager 5.0 for SQL Server & NetApp SnapManager 5.0R1 for SQL Server to back up, recover, and clone SQL Server databases running on NetApp storage systems. The recommendations in this report are generic and not specific to any configuration.

### 1.2 INTENDED AUDIENCE

This report is intended for SQL Server database administrators, storage administrators, and architects implementing a backup, recovery, and cloning solution for SQL Server databases running on NetApp storage. Readers should ideally have a solid understanding of the architecture, administration, and backup and recovery concepts of SQL Server databases.

## 2 SNAPMANAGER FOR SQL SERVER SIMPLIFIES SQL SERVER DATABASE MANAGEMENT

SMSQL 5.0 and its previous versions are aimed at supporting the following major functions with respect to SQL Server databases: -

1. Space-efficient backup of SQL Server databases.
2. Quick restore of SQL Server databases from the backups.
3. Migration of SQL Server database files to NetApp storage with minimal effort and downtime.
4. Verification of backup-sets.
5. Mirroring of backup-sets to secondary locations for DR planning.

In addition, SMSQL 5.0 supports some very specialized functions including database cloning, moving backup-sets to secondary storage and advanced backup-set verification option. These are discussed further in this document.

This section discusses the SMSQL architecture and key concepts involved in understanding how SMSQL works. The discussion is done in two sub-sections wherein the first sub-section discusses the SMSQL architecture and the second sub-section discusses the key concepts that need to be understood in order to take full advantage of SMSQL's features.

## 2.1 SMSQL ARCHITECTURE: COMPONENTS USED BY SNAPMANAGER 5.0 FOR SQL SERVER

SnapManager 5.0 for SQL Server introduces a significant change in architecture over the previous versions. The SMSQL 5.0 software consists of two major parts: -

1. The Client-Side module that is responsible for displaying the GUI and interacting with the user.
2. The Server-Side module that houses all the services and DLL's responsible for executing the different functionalities of SMSQL 5.0. These services are housed in a single EXE named SnapMgrService.EXE and runs as a Windows service named "SnapManager Service".

This architecture allows SMSQL 5.0 to offer centralized management of multiple SQL Server hosts in an organization's infrastructure. Fig. 1 presents a pictorial view of the central management capability of SMSQL.

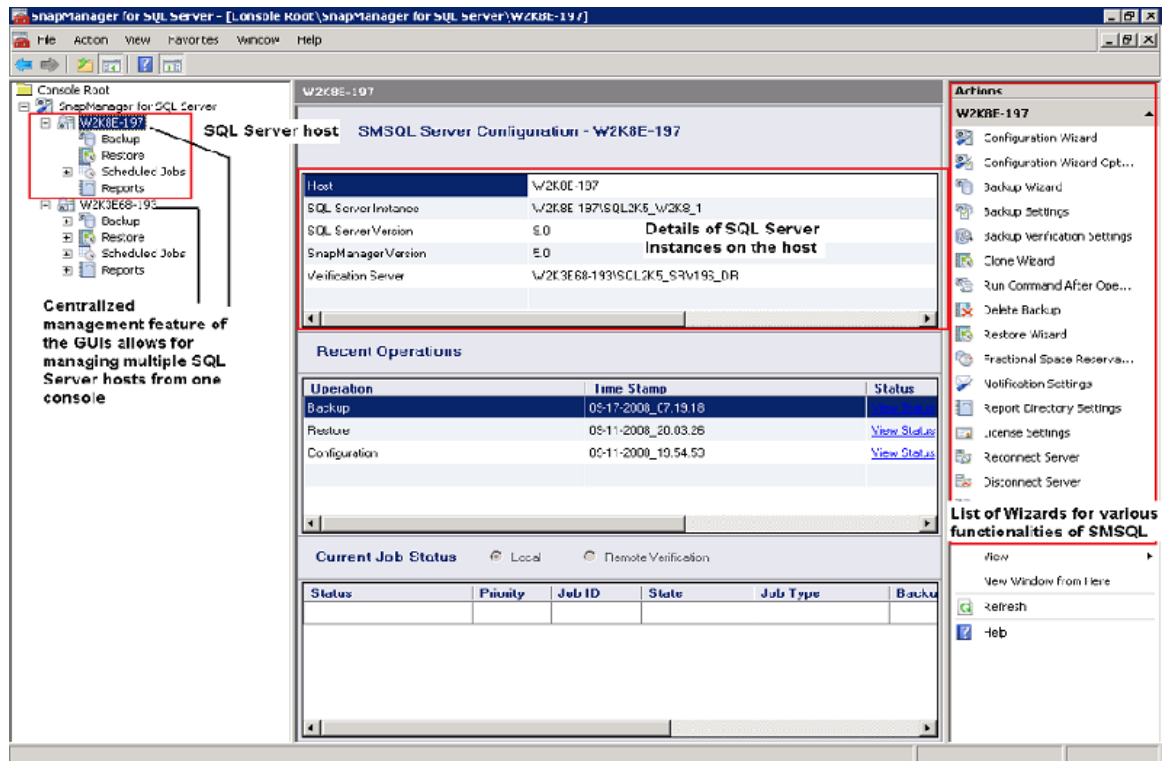


Figure 1) SMSQL 5.0 as a true central management console for SQL Server hosts

The Client and Server modules communicate amongst themselves using the Windows Communication Foundation (WCF) web services. This replaces the traditional use of RPC for inter-component communication. Furthermore, the WCF uses pure TCP communication for managing communications between the client and the server and the TCP traffic flows through Port 808 (by default). Note that this port is the default port prescribed by Microsoft to facilitate TCP/IP traffic for WCF based components. Thus, SMSQL 5.0 installer always checks for the .NET version and makes sure that it is version 3.5 SP1.

**TIP:** Using the SMSQL 5.0 GUI, it is possible to monitor SQL Server instances in a heterogeneous Windows environment. This means that it is possible to monitor all SQL Server instances running on various versions of Windows and on different platforms (x86, x64 and IA-64) from one single SMSQL GUI running on any platform and Windows OS. This implies that one can setup a true central management console to monitor a large SQL Server farm. The reason for this ability is the use of WCF

as a means of communication between client and server components of SMSQL.

## 2.2 SMSQL 5.0 CONCEPTS

### SnapInfo Directory

The SnapInfo directory is the main repository of the SnapManager for SQL Server software. The main data items that the SnapInfo contains are as follows: -

- i. Metadata related to SMSQL configuration and backup sets.
- ii. Streamed full backup of system databases
- iii. Streamed full backup of user-defined databases that share a virtual disk with a system database
- iv. All backed up transaction logs (through SMSQL)

For a SQL Server host, there can be one or more SnapInfo directories. This is because SMSQL allows one or more SQL Server instances on a host to share a SnapInfo directory as well as each instance to have its own SnapInfo directory. This can be configured at any point of time by running the Configuration Wizard.

**TIP:** For large production systems one of two approaches can be followed to allocate SnapInfo directories: -

1. Give each highly active instance that sees a lot of backups, transactional activity and has greater criticality to the business operations a SnapInfo LUN and preferably volume of its own.
2. For SQL Server instanced in a production host that do not see too much activity and house non-critical databases, group these instances into groups of 2-3 instances and give these a SnapInfo LUN of their own. Note that do not group together instances that have databases with large TLOGs.

**TIP:** It is important to note that as a part of any backup process viz., full backup or Transactional Log backup, the SnapInfo directory is always snapshotted. Note that each SnapInfo directory must be placed on a LUN of its own and no other data file can be placed on this LUN. However, the volume can certainly be shared by other LUNs. Incase of very small non-production SQL Server hosts, the SnapInfo volume can be shared and used to put TLOGs, small databases etc.

### Important concepts for SQL Server backups

SMSQL primarily performs complete backup management for SQL Server databases. Thus the basic understanding of SQL Server's backup/restore model is very important to understand so you are clear about how SMSQL functions.

Firstly, we begin with the different recovery models available in SQL Server 2000, 2005 and 2008. Recovery models basically dictate the level to which the transactions in databases are logged into the TLOG. Hence, the recovery models govern the extent to which a database can be recovered. Table 1 shows a comparison of the 3 types of recovery models in SQL Server.

Recovery Model	Extent of Transaction Logging	Recoverability	Loss of Data
Simple	Minimal. TLOG is truncated at each checkpoint. TLOG cannot be backed up.	Uptill the last full backup.	Maximum
Bulk-Logged	Moderate. Does not log bulk operations such as Index creations, BULK COPY etc.	End of last known TLOG backup.	Moderate
Full	Maximal. All transactions are logged.	Point-In-Time	Minimum

**Table 1: Recovery Models in SQL Server.**

**TIP:** The recovery model of a database will directly impact the amount of space consumed by each TLOG backup. Note that the size of a TLOG backup equals the used portion of the TLOG and not the actual TLOG file size. The used portion's size may be found out by using the command: - DBCC SQLPERF('LOGSPACE').

**TIP:** It is common for SQL Server DBA's to switch between Full and Bulk-Logged Recovery especially during loading the database with bulk data. However, for DSS backend databases, the database mostly always remains in bulk-logged recovery mode. It is always best to assume FULL recovery mode for sizing SnapInfo. Adjustments can then be made if the database is actually in SIMPLE mode.

### 3 INSTALLING & CONFIGURING SMSQL 5.0 ON WINDOWS SERVER

Installation of SMSQL 5.0 is well documented in the Installation and Administration Guide for the product which may be downloaded from <http://now.netapp.com>. However, there are many issues around the installation that may be of interest. This section talks about some specific installation related topics.

#### 3.1 SIZING FOR SMSQL DATA-STORES

One of the most common questions while installing SMSQL is how to size volumes that are used by SMSQL to store data. The primary data that SMSQL 5.0 generates and stores in its own data-stores is the data related to the SnapInfo folders. As noted in section 2.2.1, the main data items that SnapInfo contains are as follows: -

- i. Metadata related to SMSQL configuration and backup sets.
- ii. Streamed full backup of system databases
- iii. Streamed full backup of user-defined databases that share a virtual disk with a system database
- iv. All backed up transaction logs (through SMSQL)

Of the above components, the following items are easy to size and thus you get a good idea of the amount of space that would be consumed: -

1. Metadata - Metadata per backup set is less than 10kb.
2. System databases – Total size of all system databases (that can be backed up with SMSQL 5.0) except TempDB which are MASTER, MODEL & MSDB = 15MB (at the maximum).

**TIP:** The only database that is subject to larger change rate is the MSDB since it stores job histories, backup histories, SQL Server jobs (including replication related) etc.

Amongst the remaining two items, the chances of having streamed backups of user databases is low since this happens only when the user databases are on the same LUN/volume as the system databases.

**TIP:** It is best to avoid a configuration where the system databases share the same volume as user databases. The reason is that SMSQL cannot snapshot volumes that have system databases on them because I/O cannot be suspended for system databases.

Therefore, the major consumers of space are the Transaction log backups of different databases. The main difficulty in determining the size of these backups arises from the following factors: -

1. The sizes of transaction log files vary and are subject to auto-growth.
2. The actual amount of log file space used varies greatly across all the databases in a SQL Server instance. It primarily depends on :-
  - i. Transactions per minute.
  - ii. Types of transactions. Note that SELECT statements are not logged as opposed to data manipulation statements such as INSERT, UPDATE and DELETE.
  - iii. Recovery model of the database. Refer table 1 for the impact.

The factors above make the sizing exercise for SnapInfo very cumbersome. In order to overcome these factors to a certain extent, the following approach is recommended: -

Step 1: For each SnapInfo Folder do:

Step 2: Based on recommendations in section 2.2.1, determine the number of SQL Server instances that will share the current SnapInfo folder.

Step 3: For each SQL Server instance do:

Step 4: Run the script listed in Appendix A. The script will generate values: -

- iv. Base Value: Sum of the current log file sizes of all the databases in the instance. Let this value be called BASE\_VAL.

Step 5: Now determine the maximum number of TLOG backups per database that will be kept in the SnapInfo folder. Let this be N.

Step 6: Therefore, the predicted LUN size =  $\text{BASE\_VAL} * N$ . Note that the LUN size is a cumulative total for all the SQL Server instances.

Step 7: Remember the LUN size for this iteration.

Step 8: Once steps 4-7 are run for each concerned SQL Server instance, we will get the target

LUN size needed for the SnapInfo folder.

Step 9: Note that each time SMSQL completes a Full or TLOG backup, it snapshots the SnapInfo volume. So now we need to determine the number of snapshots to be kept online. Let this value be M.

Step 10: From the list of LUN sizes that were obtained from steps 4-7, choose the highest value. It is safe to assume that the snapshot size will at most be this highest value. Lets name it HIGH\_VAL.

Step 11: Therefore the volume size for the SnapInfo folder = LUN Size obtained from step 8 + (HIGH\_VAL \* M).

For these steps it is assumed that the number of databases in the instance remain fairly constant.

Hence, we now have an approximation procedure to determine the size of the SnapInfo volume that takes into account: -

1. Average TLOG Sizes across all databases.
2. Number of TLOG backups to be kept online.
3. Number of SnapShots of the SnapInfo volume to be kept online.

**TIP:** By creating a schedule for regular TLOG backups, one can keep the max size of the TLOG file bounded to the value existing at the start of the above procedure. In BULK-LOG and FULL recovery modes, inactive portions of the TLOG file are made available to SQL Server for further transaction logging.

**TIP:** By default, SMSQL does not impose Fractional Space Reservation policies on the SnapInfo folder. However, by following the sizing exercise mentioned above, one can impose a lower value than of 20-30% for fractional space reservation. For low-activity non-production servers a value of 10% or lower may suffice too.

### 3.2 CONFIGURING SMSQL 5.0 – WHAT'S NEW

Essentially, the configuration path for SMSQL 5.0 remains unchanged from the earlier 2.1 version. It has the same settings that need to be configured and the definitions of those settings remain the same. Please refer to Chapters 6 & 7 of the Installation & Administration Guide for more details.

However, there are two changes in the additional tools that are available for configuration:

1. Control-File Based Configuration
2. SMSQL Legacy Job Upgrade (refer to page 67 of the Installation & Administration Guide)

#### Control-File Based Configuration

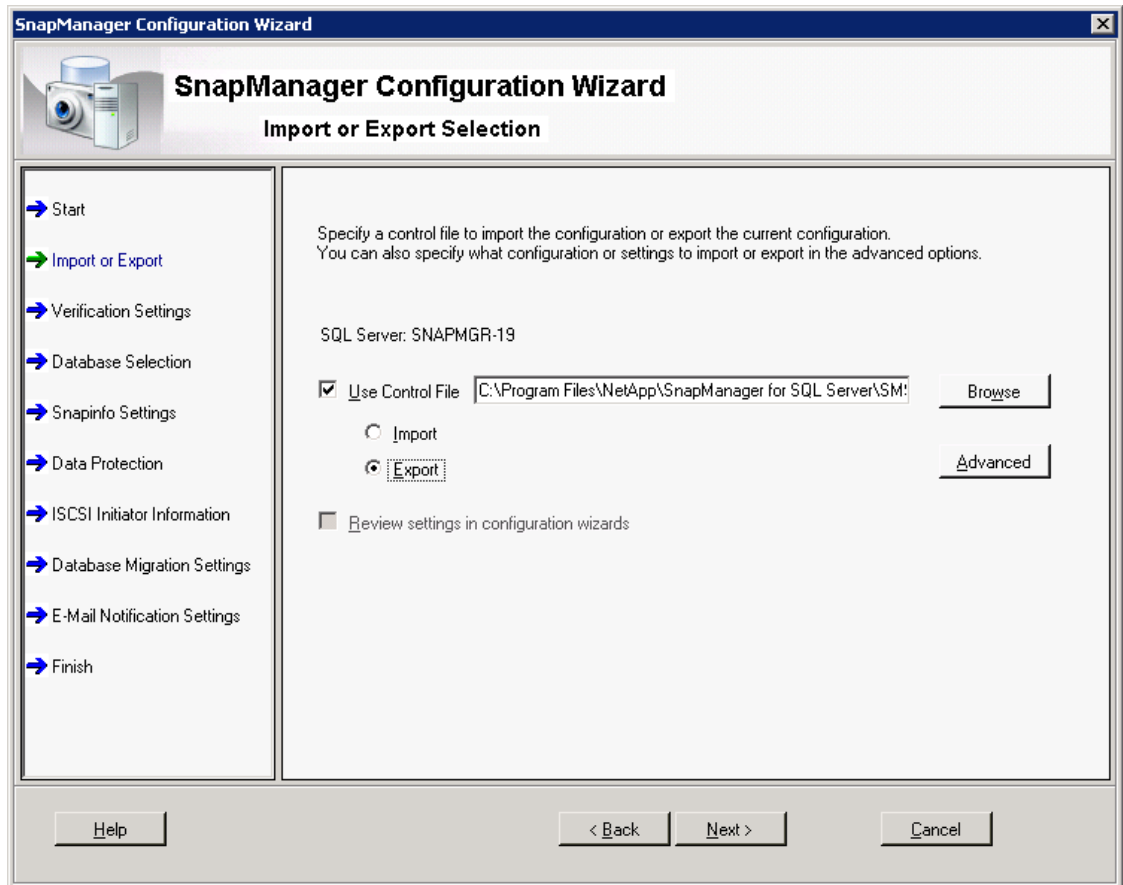
The control-file is basically an XML formatted file that stores all the configuration settings corresponding to the SMSQL 5.0 instance where it has been generated. Since this file is in XML format, it can be edited manually or programmatically but it is recommended that XML editors be used to avoid syntax errors in the file.



Here is an excerpt of the type of data that resides in the control file: -

```
<VERIFICATION_CLIENT_SETTING>
  <VERIFICATION_SERVER>W2K8E-197</VERIFICATION_SERVER>
  <VER_SERVER_NTAUTH>true</VER_SERVER_NTAUTH>
  <VER_DBCC_NOINDEX>false</VER_DBCC_NOINDEX>
  <VER_DBCC_ALL_ERROR_MSG>false</VER_DBCC_ALL_ERROR_MSG>
  <VER_DBCC_NO_INFO_MSGS>true</VER_DBCC_NO_INFO_MSGS>
  <VER_DBCC_TABLOCK>false</VER_DBCC_TABLOCK>
  <VER_DBCC_PHYSICAL_ONLY>true</VER_DBCC_PHYSICAL_ONLY>
  <VER_DBCC_ATTACH_DB>false</VER_DBCC_ATTACH_DB>
  <VER_DBCC_BEFORE_MIGRATION>true</VER_DBCC_BEFORE_MIGRATION>
  <VER_DBCC_AFTER_MIGRATION>false</VER_DBCC_AFTER_MIGRATION>
  <VER_DELETE_DB_FILE_ORIG>true</VER_DELETE_DB_FILE_ORIG>
  <VER_RUN_UPDATE_STATISTICS>true</VER_RUN_UPDATE_STATISTICS>
</VERIFICATION_CLIENT_SETTING>
```

The default path where the configuration file is created is "C:\Program Files\NetApp\SnapManager for SQL Server". The configuration file can be created by choosing the EXPORT option in the configuration wizard. Fig.2 shows the screen-shot of this process.



**Fig.2: How to Export Control-File**

Data in the control-file is grouped into the following sections:

1. Storage layout
2. Notification settings
3. Verification settings
4. Report directory setting
5. Backup settings
6. Scheduled backup jobs
7. SnapMirror relationship settings

Thus, the benefits of the control file can be felt most during: -

1. Disaster Recovery scenarios where the same setting as the primary site exists at the DR site and SMSQL 5.0 is being configured at the DR site.
2. Configuration of SMSQL 5.0 instances installed on cluster nodes wherein all the configuration settings remain the same.

In order to take advantage of the control file, it has to be imported into the SMSQL instance via the configuration manager. Fig.3 shows the screen-shot of this process.

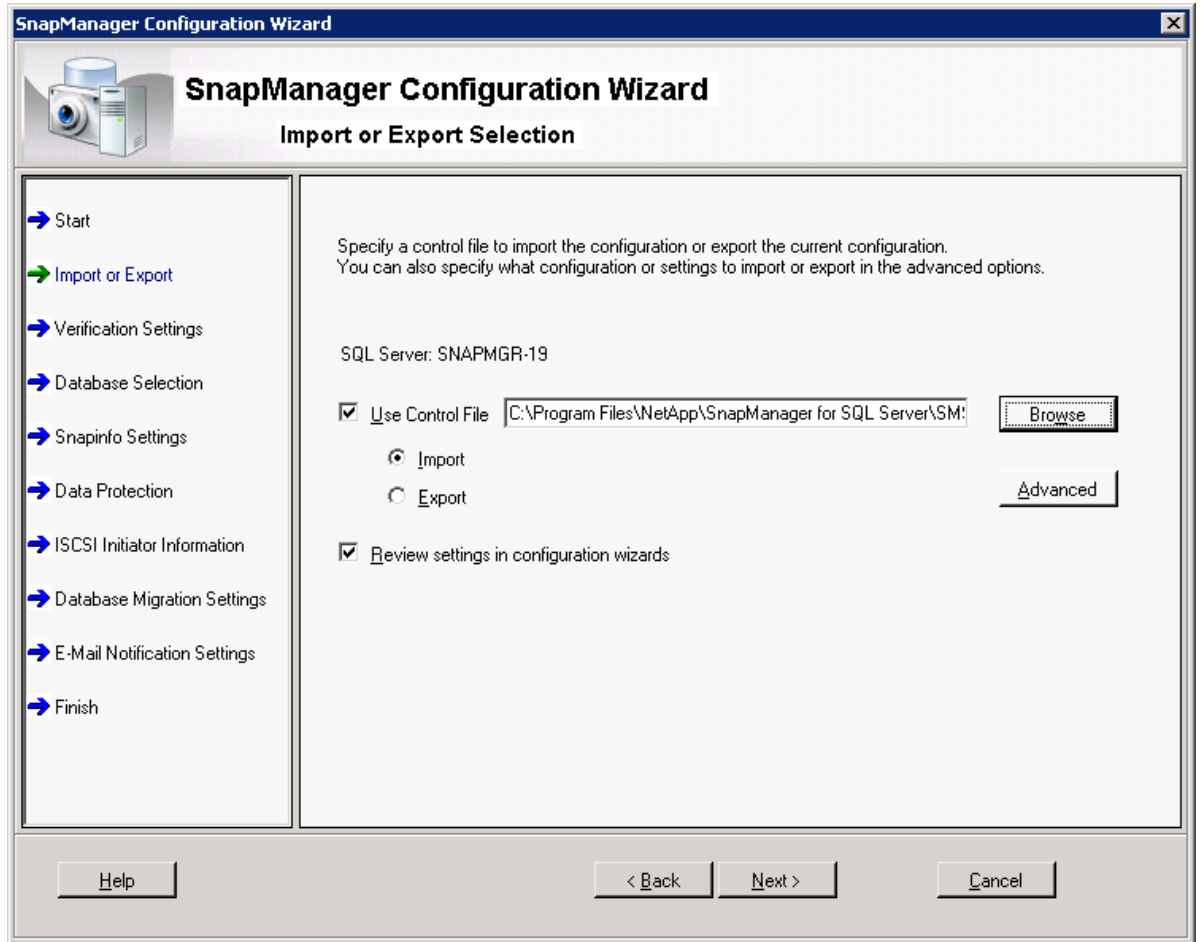


Fig.3: How to Import Control-File

However, there are number of prerequisites that must be kept in mind while attempting to import configuration settings from the control file: -

1. LUN Creation and Connection: The administrator has to create and connect to the LUNs on the host using the same drive letters or mount points as specified in the control file.
2. SnapMirror & Snapvault Relationships: The administrator has to establish all the relevant SnapMirror & SnapVault relationships as exists for the volumes of the original host where the control file was generated.

**Tip:** If the un-configured server, where the control file is being imported, has identical storage layout, same SQL Server instance names, database names, server names etc. then the import will work directly. Otherwise, one will have to edit the control file accordingly prior to import.

For more details on how to use control-files and the workflow, please refer to the “Understanding control-file based configuration” section in chapter 7 of the Installation & Administration Guide.

## 4 LAYING OUT SQL SERVER DATABASES ON NETAPP STORAGE

### 4.1 LIMITATIONS OF DATABASE LAYOUT

The limitations that are imposed by SMSQL 5.0 with respect to database layout remain unchanged from SMSQL 2.1.x except the following things that one should keep in mind: -

1. Ensure that those databases that need to avail the usage of Protection Manager datasets are placed on QTREE LUNs. There should only be one LUN per QTREE.
2. Please refer to the section titled "Maximum configurations supported by SnapManager" on page 95 of the Installation & Administration Guide. Although SnapManager does not prevent you from creating configurations that exceed these limits, such configurations are untested and unsupported. It is best that you do not exceed any of these limits.
3. SMSQL 5.0 also supports databases where the database files are spread across multiple storage controllers. This sort of a database configuration is widely used for VLDB's that act as a back end to DSS software and processes.
4. SMSQL 5.0 and earlier versions do not perform backups at a FILEGROUP level. Creating filegroups is purely a performance consideration and does not affect the way SMSQL backs up databases in any way. Section 5 of this document talks more about the internals of SMSQL backups.

Please refer [TR-3696](#) for additional details on supported database layouts on NetApp storage along with the merits and demerits of the same. Please refer to section "SQL Server database configuration restrictions enforced" on page 96 of the Installation and Administration Guide for details on additional limitations imposed by SMSQL 5.0 on database layout patterns.

### 4.2 FRACTIONAL SPACE RESERVE SETTINGS

SMSQL 5.0 takes advantage of fractional space reservation capabilities in Data ONTAP® 7.1 and above. Fractional Space reservation is the amount of space data ONTAP guarantees will be available for overwriting blocks in a LUN. In Data ONTAP 7.1 and later, it is adjustable between 0 and 100% of the size of the LUN.

**TIP:** Instead of opting for 100% space reservation that forces only half the volume to be available for LUN usage, careful sizing can reduce the space reservation to 10-20% of the volume size. This means that the actual volume needn't be any larger than 1.2 times the LUN size. Please refer to section 3.1 for an example of SnapInfo sizing that can lead to lower fractional space reservation values. Similar exercise may be followed for sizing database volumes as well.

As part of the feature SMSQL implements two policies to guarantee that writes to a LUN do not fail: -

1. Automatic deletion of backup sets.
2. Automatic dismount of database.

These policies allow SMSQL to take appropriate actions when available space on a volume becomes limited. These configurable options give greater flexibility to administrators who are monitoring storage utilization very closely.

Space reservation is designed to ensure that protected files (or files that have space reservation turned on) always have the free space they expect so that a Snapshot copy of the LUN can complete even if 100% of the SQL Server database blocks have changed. Data ONTAP 7G provides for space reservation at the FlexVol, qtree, and file levels. Any file may be designated as protected and is always turned on by default for LUNs.

When creating a LUN, the NetApp storage system verifies there is enough available disk space to accommodate the specified size. By default, WAFL reserves blocks equal to two times the specified size of the LUN (when Snapshot is used) so that all future writes can be completed. If space reservation was not enabled, it would be possible to oversubscribe the available storage. If this occurs, SQL Server will, because its space is pre-allocated, interpret it to mean the storage device is bad, and it will mark affected databases as suspect.

SMSQL takes appropriate actions to prevent a LUN from becoming inaccessible as a result of no free space being available on the hosting volume. SMSQL automatically does the following when the appropriate user-configurable trigger is hit: -

1.Delete Backup Sets Policy:

- Will delete oldest backups first
- Will NOT delete the most recent backup on affected Lun.
- User can configure the number of backups to be retained.
- Will delete:
  - SMSQL created snapshots
  - Associated SnapInfo data

2.Dismount Databases Policy:

- This policy will be triggered, when the above policy fails to prevent the volume from reaching the end-of-space state. This stops the databases on the particular LUN from becoming corrupted.

**TIP:** Choose a backup retention level based on your SMSQL backup creation and verification schedule. If a Snapshot copy deletion occurs, you should ensure that a minimum of one verified backup remains on the volume. Otherwise, you run a higher risk of not having a usable backup set to restore from in case of a disaster.

SMSQL executes defined deletion policies on a per volume basis. If a backup set is spanned across multiple Volumes, you have to set similar or identical policies across those volumes. If this practice is not followed, you may end up with mismatching backup sets on different volumes. This may cause a backup set to be rendered useless and unable to be restored.

If automatically deleting backups cannot free enough space in the fractional overwrite reserve, SMSQL dismounts the databases when the reserve is 90% consumed. This prevents an out-of-space condition on LUNs hosting databases and transaction log files.

**TIP:** The threshold value for the policy "deletion of backup sets" should always be less than that of "dismount databases".

**TIP:** When a LUN is fully space reserved (fractional space reservation set to 100%), write operations to that LUN are guaranteed against failure caused by an out-of space condition due to Snapshot copy disk space consumption. If you do require a fractional space reservation policy, please work with your NetApp support representative. The representative can help implement a fractional space reservation policy that fits your environment. The fractional space reservation policy is set to 100% (disabled) by default. In order to leverage fractional space reservation policies, you must set the fractional reserve parameter to less than 100%.

## 5 BACKUP & RESTORE WITH SMSQL 5.0

In this section we focus on some of the key concepts of how databases are backed up by SMSQL 5.0. We will also focus on SMSQL 5.0's support for new backup features in SQL Server 2008.

### 5.1 HOW DOES BACKUP WORK

SMSQL 5.0 is based on the Virtual Devices Interface™ (VDI) technology from Microsoft that allows third party utilities to take consistent backups of SQL Server databases. For more information on VDI, please refer to the following link: -

<http://www.microsoft.com/downloads/thankyou.aspx?familyId=416f8a51-65a3-4e8e-a4c8-adfe15e850fc&displayLang=en>

The following are the broad steps that happen when a backup operation is triggered from SMSQL: -

1. Checks the SnapManager license.
2. Renames the most recent SnapInfo directory (if necessary).
3. Renames the most recent Snapshot copy (if necessary).
4. Creates a new directory in the SnapInfo directory for this backup. At the same time it also creates a .SML file that contains the metadata for this backup.
5. SMSQL enumerates all the databases that reside on the current NetApp volume even if they are in different LUNs.
6. SMSQL then initiates VDI backup using the BACKUP DATABASE....WITH SNAPSHOT command for each of the databases sharing the volume.
7. SQL Server receives the BACKUP commands and starts suspending the I/O activity on all databases that share the same NetApp volumes.
8. SQL Server returns an acknowledgement back to SMSQL confirming that I/O is suspended.
9. SMSQL now snapshots each of the involved volumes serially.
10. SMSQL returns acknowledgement to SQL Server about snapshot completion.
11. SQL Server unfreezes the I/O on the databases that were frozen in step 7.
12. SMSQL proceeds with further actions such as TLOG backup, SnapMirror update etc.

**TIP:** There are two important points that should be noted in the steps above: -

1. I/O for **all** databases in a volume are frozen irrespective of which database is being backed up.
2. If a database uses multiple volumes then these are snapshotted one by one. However, if it uses multiple LUNs on the same volume then all these LUNs are snapshotted together since snapshots are volume based.

## 5.2 BACKUP BEST PRACTICES

There are a few things that should be kept in mind with respect to SMSQL backup process: -

1. Microsoft recommends that databases should not be frozen for more than 10 seconds as a part of the snapshotting process. Please check the backup logs at C:\Program Files\NetApp\SnapManager for SQL Server\Report\Backup [Instance Name] to determine if the freeze window exceeded 10 seconds. Note that there are a couple of factors that affect the time taken to complete the snapshot operation: -
  - i. Size of the volumes on which the database LUNs reside and the rate of change in them. Note that SMSQL will snapshot volumes serially so if a database is spread across large number of volumes, the snapshotting will take more time.
  - ii. Number of databases sharing the same volume although they may be in different LUNs on the same volume. This is because NetApp snapshot is volume-wide and so also is the snapshot granularity from a VDI standpoint.
  - iii. How busy the storage controller is in terms of CPU usage, IOPS bandwidth etc.
  - iv. Space left on the volume. The fuller the volume gets (especially after 50-60% fullness) the slower will be the snapshots.
  - v. Network delays. This may cause delays in the acknowledgement from the storage controller reaching SMSQL and then SQL Server. Till then SQL Server will keep the databases frozen.
2. It is best not to put large databases on the same volume even if they are on separate LUNs inside the volume. This may have implications for the time taken to snapshot the volume.
3. Presently there is a limitation of 35 databases that can share a single volume. The reason for this is that in order to manage the freezing and un-freezing of I/O for each database, SQL Server has to use 4-5 worker threads per database. This may lead to a situation where SQL Server may run out of worker threads and fail in unexplained manner. Thus, Microsoft strongly recommends that no more than 35 databases should share a storage volume. More information on this issue may be found at: <https://now.netapp.com/Knowledgebase/solutionarea.asp?id=kb36996>

In order to overcome this limitation in SQL Server architecture, SMSQL 5.0 breaks up the total number of databases on a volume into batches of 35 databases and then proceeds to initiate a Snapshot of the volume for each batch. Thus at any given time only 35 databases in the volume will be frozen. However, it is strongly suggested that such a layout should only be done for small databases.

4. SMSQL 5.0 supports both synchronous and asynchronous mirroring which means that one can mirror the database volumes in any manner without any impact on SMSQL backups. If the database volumes are synchronously mirror then there is no need to select the "Update SnapMirror" option for the backup plan. Please refer the section "How SnapManager uses SnapMirror" on page 270 of the Installation and Administration Guide.
5. Data ONTAP has a limitation of 255 snapshots per volume so one must be careful about how many snapshots to be kept online. SMSQL 5.0 is integrated with NetApp Snapvault through Protection

Manager datasets. Hence, older snapshots should be archived as per their lifecycle and restore needs. Please refer section 8 of this paper for more details.

### 5.3 COMPRESSION OF TRANSACTION LOG BACKUP (SQL SERVER 2008)

SMSQL 5.0 supports compressed TLOG backups for SQL Server 2008 databases. The TLOG backup compression comes into play only if it has been set at the SQL Server level by the DBA. SMSQL 5.0 on its part does not use the WITH COMPRESSION qualifier with the TLOG backup statement. It may be enabled by checking the "Compress Backup" option in the Server Properties dialog box. Fig.4 shows the dialog box:

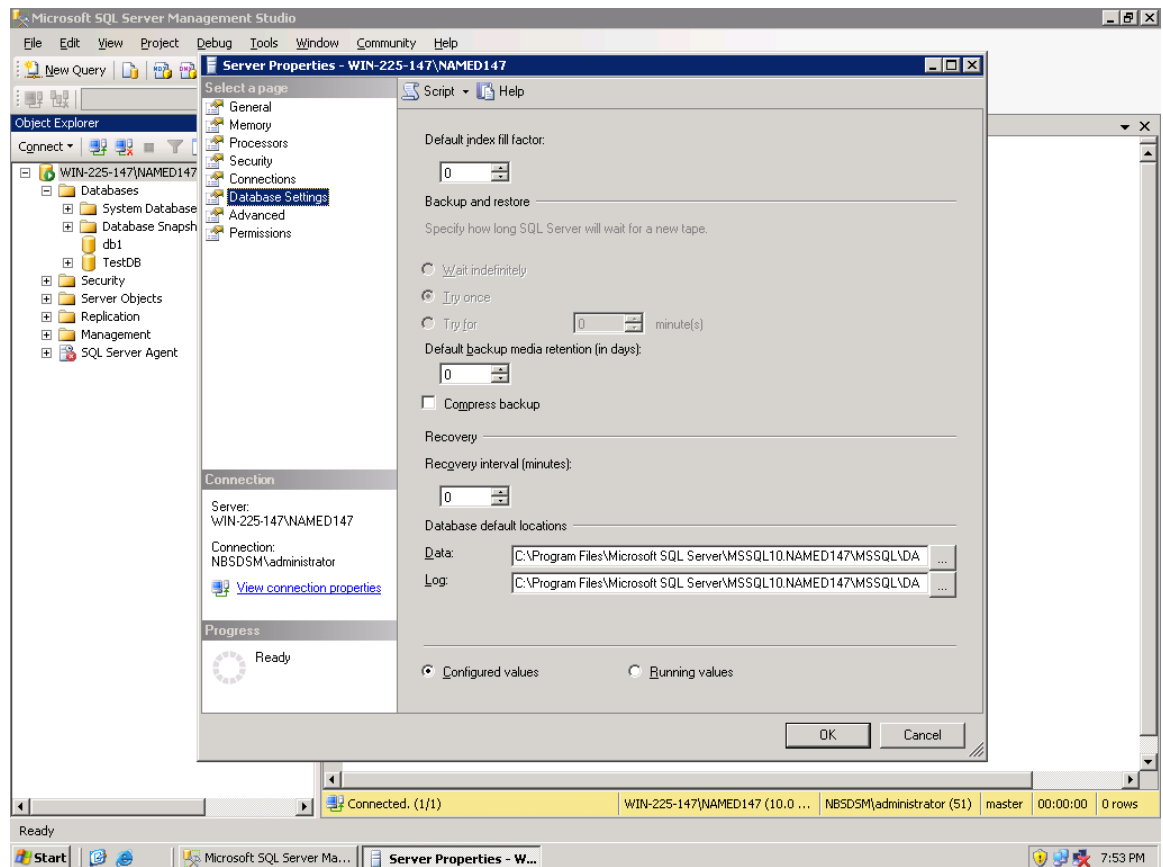


Fig. 4: TLOG Backup Compression in SQL Server 2008

The TLOG backup compression feature simply compresses the backup file that is generated as a result of the BACKUP LOG statement. Hence, in the case of SMSQL 5.0, if the option is set, then the .TRB file that is saved to the SnapInfo folder as a part of the TLOG backup is appreciably smaller than a regular TLOG backup. Since, the end result is just a smaller file on the SnapInfo LUN, there is no impact on the way snapshot-based backups work in SMSQL.



## 5.4 PROTECTING YOUR DATABASES

The central piece for protecting SQL Server databases from catastrophic disasters at a data center revolves around the ability of SMSQL to mirror backups to remote locations. Please refer to the section “How SnapManager uses SnapMirror” on page 270 of the Installation and Administration Guide for more information on how to update snapmirror relationships from within SMSQL backup plans.

In addition, SMSQL 5.0 offers the ability to verify backups present at SnapMirror destination volumes. This feature is discussed in more details in section 9.1 of this paper. Furthermore, [TR-3604](#) discusses various solutions for planning DR for SQL Server user databases based on different RTO/RPO targets.

Nonetheless, is important to discuss some important concepts that must be kept in mind while planning how database volumes should be snapmirrored to DR site: -

1. **CHECKPOINT:** Apart from forcing dirty database pages on to disk, CHECKPOINT also alters the minimum recovery LSN for a database. The minimum recovery LSN or minLSN is the LSN from which SQL Server starts scanning a database's transaction LOG to perform a database recovery.
2. Database consistency: Note that taking NetApp snapshots outside of SMSQL for database volumes does not guarantee application consistent snapshot. Hence, any snapshots taken for database volumes must necessarily be initiated by SMSQL.
3. Network traffic generated by replication through SnapMirror updates. Note that SMSQL supports database volumes being both synchronously and asynchronously snapmirrored. However, if the underlying database volumes are synchronously mirrored then there is no need to check the “Update SnapMirror” option for the backup plan.

It is important to note that the following SnapMirror configurations do not work: -

1. Database volume asynchronously snapmirrored and log volume synchronously snapmirrored.
2. Database and log volumes asynchronously snapmirrored with snapmirror updates being done outside SMSQL. Note that even if the updates are scheduled to run at the same time, this arrangement will not work.
3. Database volume synchronously snapmirrored and TLOG volume asynchronously snapmirrored.

The reason for the above snapmirror configurations not working is due to the way database checkpointing and database recovery work. Basically, in the three scenarios mentioned above there is a mismatch between the last LSN that wrote to the database file before it got replicated and the minLSN. Specifically, the minLSN in most cases is further in time than the last LSN that wrote to the database. This causes a consistency issue during database recovery and hence, the recovery of the database from the replicated volumes fails.

## 5.5 PERFORMING TABLE-LEVEL RESTORES WITH SMSQL

Restoring particular tables from within backup-sets is a common requirement in many database environments. In the case of SQL Server, this requirement becomes more challenging because of the lack of support for table level restores from within SQL Server 2000 or 2005. However, SMSQL 5.0 provides an effective solution in this space with the use of the Database Cloning feature which is discussed in more detail in the next section.

The following steps may be used to perform table level restores from a database backup that is present online: -

1. Start the Clone Wizard from the SMSQL 5.0 GUI.
2. Choose the “Clone Databases from existing backup sets” option.

3. Follow the steps in the wizard and ensure that the database clone is created on the same SQL Server host (preferably) as the target database. Ensure that the name of the Clone is different from the name of the original database.

**TIP:** Depending on how much further in time should the database be recovered to, adequate number of TLOG backups (if they exist) may be chosen during the closing operation.

4. Once the clone of the database is ready and attached to a SQL Server instance, use BCP, SELECT INTO, SSIS packages etc, to import the table into the original instance.
5. Once the table is recovered in the original instance, run the cloning wizard again and choose the "Delete Clones" option and delete the clone that was created in step 3.  
TIP: It is advisable to use the "Delete Clone" option since it will perform a complete clean-up of the clone including LUNs that were mounted at the host.

Advantages of this method: -

1. No additional space needed since we will not be writing anything to the Clone.
2. Cloning of databases in SMSQL 5.0 is done either through FlexClones or read-write snapshots which does not call for additional space overhead unless that clone database is written to.
3. Cloning is far quicker and non-disruptive compared to a database restore.

## 6 DATABASE CLONING WITH SMSQL 5.0

One of the most important features that has been introduced in SMSQL 5.0 is the ability to create clones of a database. SMSQL 5.0 supports the following operations with respect to database clones: -

1. Clone Creation. The supported sources are: -
  - i. Active Databases.
  - ii. Database Backups.
2. Clone Deletion. This includes cleaning up of all clone-related resources including host-side LUN mappings.

Note that it is desirable to have FlexClone licensed on the storage system where the backup sets or active database is based. However, this is not a compulsory pre-requisite. In case FlexClone is not licensed, the LUN is completely backed by the snapshot.

**TIP:** In case one plans to use a database clone for a longer duration, it is recommended that FlexClone be licensed on the concerned filer/filers so if need be, the FlexClone can be split and made into a LUN of its own.

**TIP:** Note that cloning is a feature that does not depend on the SQL Server version since the underlying mechanism is the same. The only caution that must be exercised is that clones of databases or backup-sets must be created on similar SQL Server versions as the original database's SQL Server version.

In the following sub-sections we look at some of the internals of the cloning operations and discuss some best practices around database clones and how to use them optimally.

## 6.1 HOW DOES DATABASE CLONING WORK

In order to understand how Database Cloning can be used and to obtain the step-by-step process of creating a database clone please refer to chapter 11 of the Installation & Administration Guide. In this section we will discuss some of the internals of how database clones are created from within SQL Server: -

1. There are two scenarios of creating a database clone: -
  - a. From online backup sets of the database.
  - b. From active database that is present online.
2. In case it is chosen to create a clone of an active database, SMSQL 5.0 will first create a FULL backup of the database. The rest of the steps are similar.
3. SMSQL begins by contacting the SDW of the host on which the clone database will be attached to a SQL Server instance.
4. SMSQL now calls upon the services of SDW to create a clone of the database. For this, it passes the snapshot either as the one the user has created or the one created in step 2.
5. SDW goes ahead and begins a clone creation process at the storage level. It begins by checking for the presence of the FlexClone license. If FlexClone licensed then a flexclone of the snapshot is created. Else the snapshot is mounted as a read-write snapshot. SDW then proceeds to connect to the LUN inside the flexclone or the mounted snapshot.
6. Then the SMSQL instance on the originating host initiates a VDI-based restore of the database to the desired SQL Server instance on the desired host.

## 6.2 DATABASE CLONING BEST PRACTICES

Database cloning is a very useful feature and its usages can be best felt in the following scenarios: -

### 1. Creation of database clones for dev & test environment.

**TIP:** For such an environment it suffices to create clones from historical backup sets with limited number of logs played forward.

**TIP:** Multiple clones can be created from the same backup-set.

### 2. Creation of staging environments that acts as an exact replica of production environment.

**TIP:** One should back up the TLOG right after the FULL backup is complete. This may be done by checking the appropriate option in the backup wizard. Subsequently, select the latest TLOG backup to be applied as a part of the cloning process. This increases the currency of the cloned copy.

3. It is preferable to have FlexClone licensed for storage systems on which clones that are to be used for longer duration are created.

## 7 DATASET INTEGRATION WITH SMSQL 5.0

SMSQL 5.0 is integrated with NetApp Protection Manager datasets. This integration allows DBA's to archive database backups to a NetApp SnapVault secondary destination. Thus SMSQL 5.0 offers a complete disk-to-disk backup archival solution integrated with the product. The benefits of this feature are: -

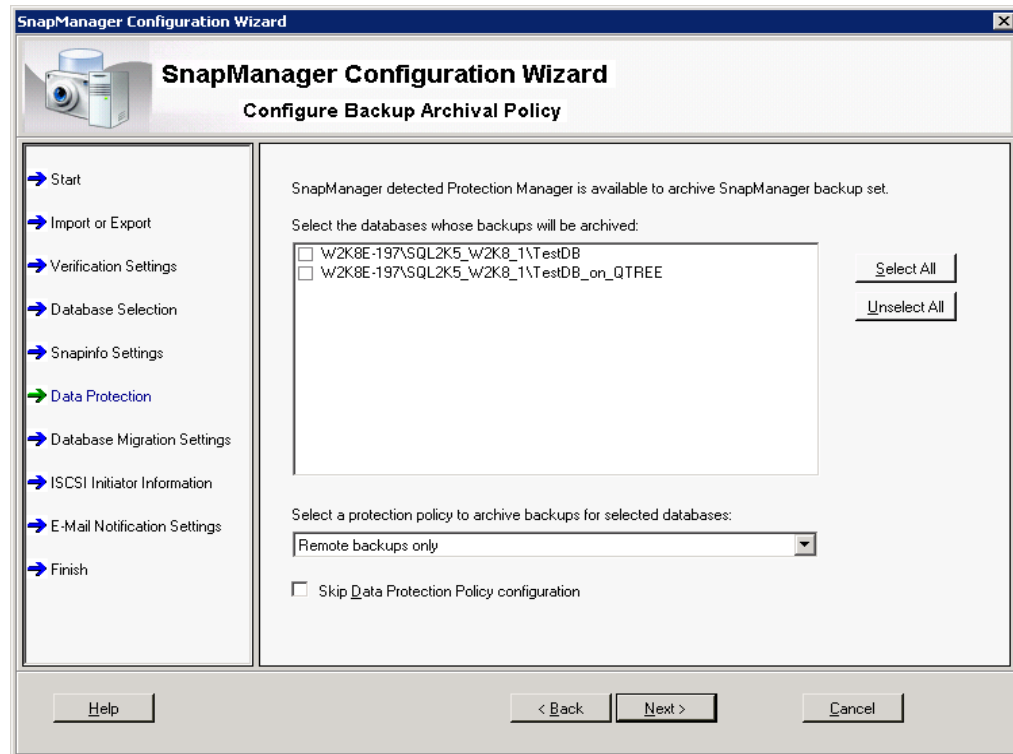
1. Allowing expensive primary storage space to be used by fewer online backups without losing older backup sets.
2. Planning added disaster recovery functionality.
3. Planning legal compliance strategies that require organizations to preserve older backups.

There are some prerequisites for planning disk-to-disk archival using Protection Manager datasets: -

1. For a database to make use of datasets, all its files should be placed on QTREE LUNs.
2. Ensure that SDW 6.0.1 is being used and has the integration with Protection Manager option checked. **CAUTION:** Note that this option can be selected only during installation. In case this option was not checked during installation, please run the installer, install again and check this option.

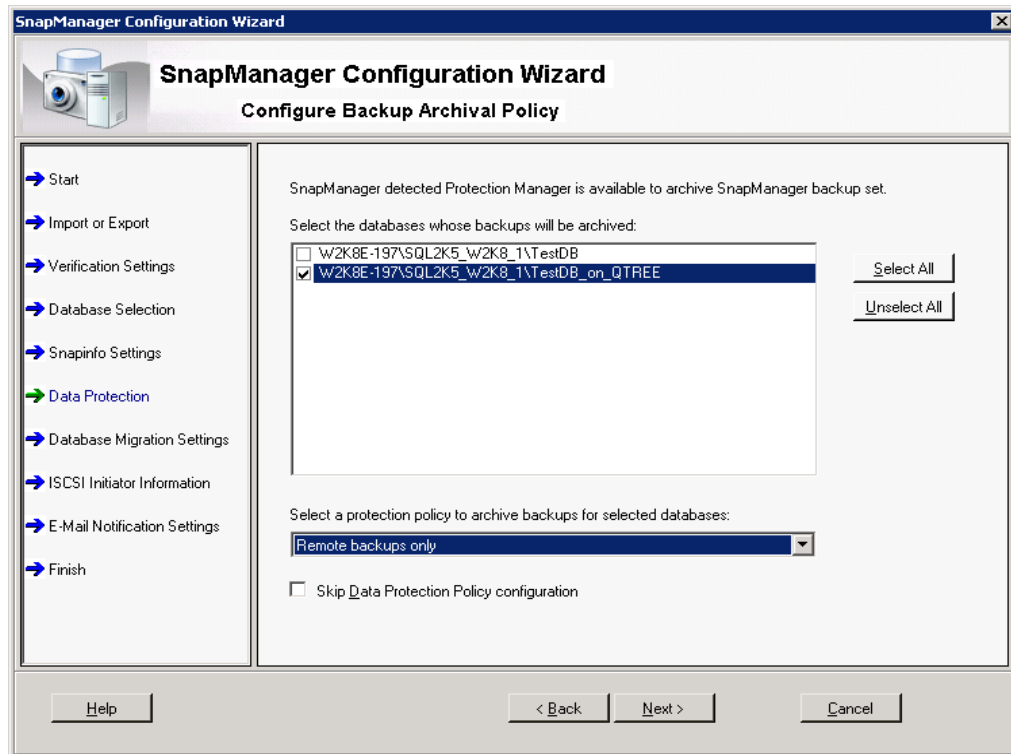
The following are the steps that need to be performed to ensure that the integration of SMSQL 5.0 with Protection Manager is complete: -

1. We begin by opening the "Configuration Wizard". Thereafter just keep clicking Next, without changing any option, until you reach the screen that says "Configure Backup Archival Policy": -



2. In our case we will proceed by choosing the “TestDB\_on\_QTREE” database. Make sure that the “Remote Backups Only” policy is selected from the drop-down list below. Click Next.

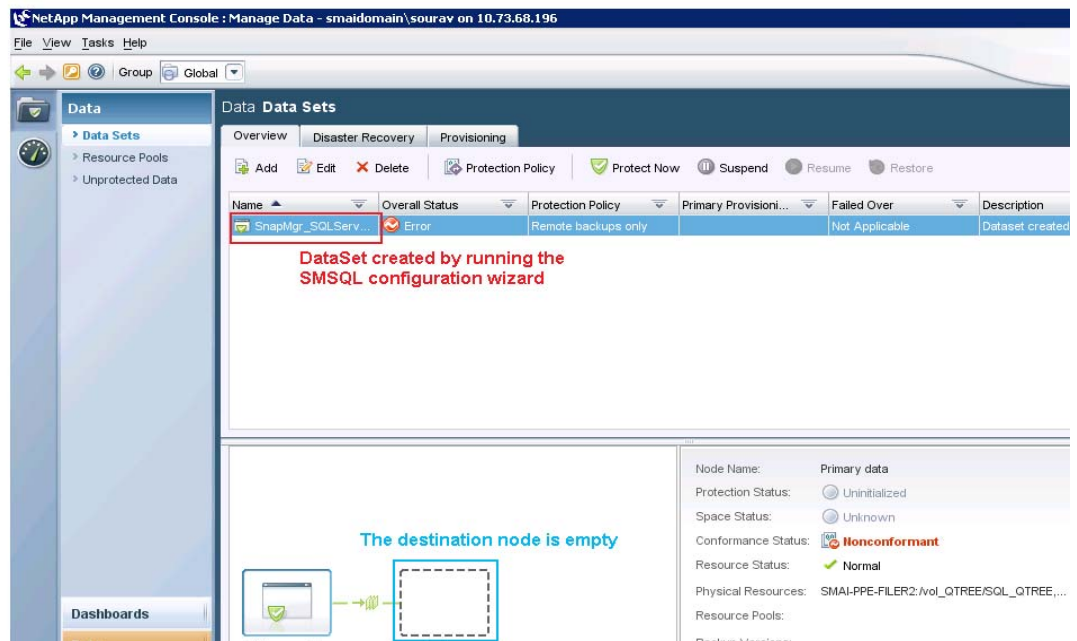
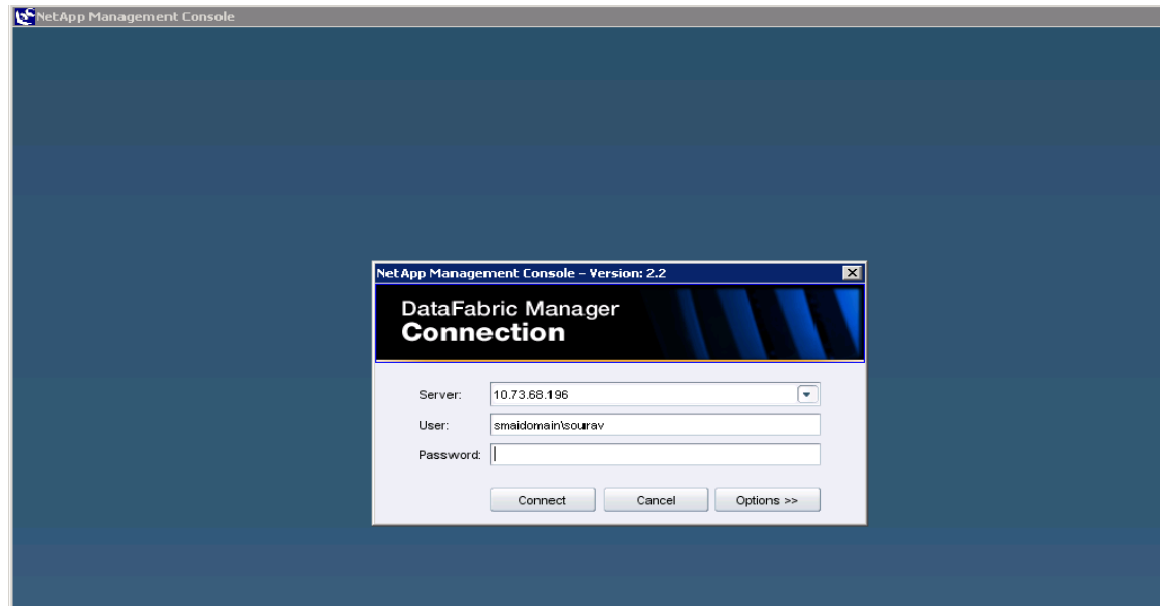
**TIP:** If the “Select All” button is pressed in the window above, SMSQL will select all the databases that are present on QTREE LUNs and hence qualify for dataset based protection.



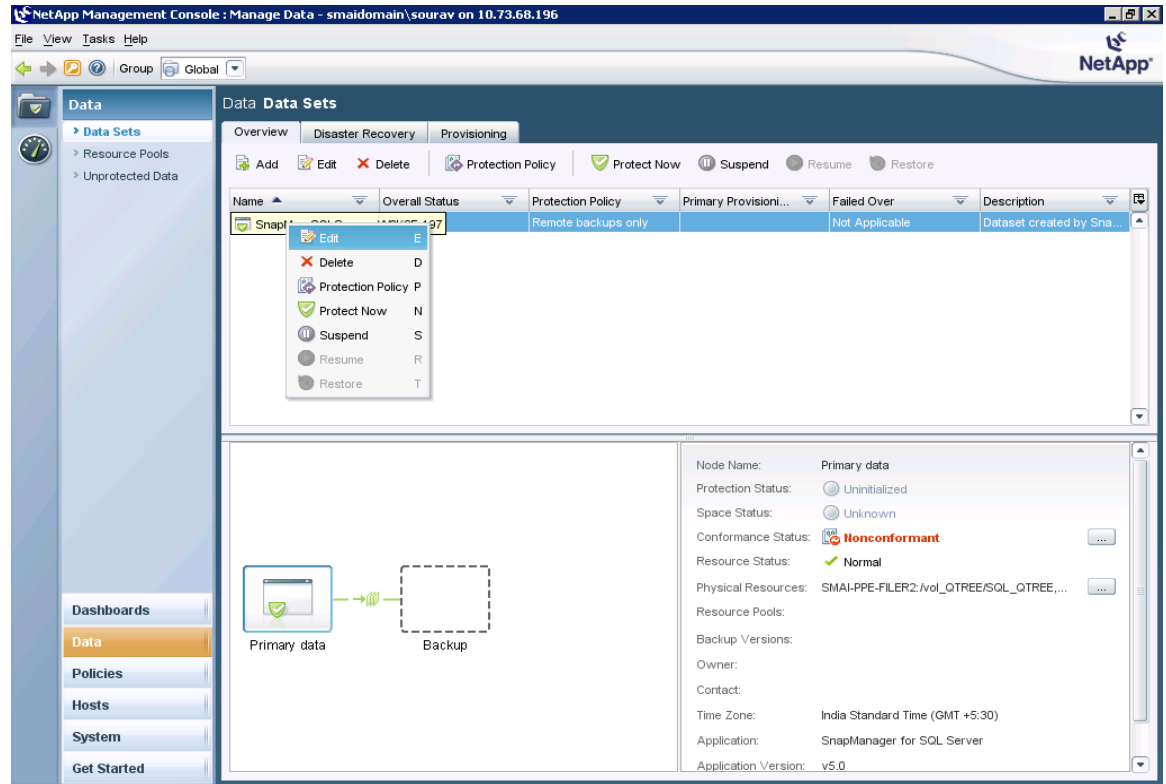
3. Now keep clicking Next, without changing any option till the end. SMSQL will create a dataset in Protection Manager. This dataset will have a name beginning with "SnapMgr\_SQLServer....". Note that this backup-set will be created with Snap-Vaulting as the backup mode but the destination node will be empty since SMSQL does not know which storage device you would choose as the SnapVault destination. Let's look at how to continue further.

**TIP:** It is best to inform the storage admin before you begin this operation. The storage admin in turn should be ready with the name of a destination storage device to be used as the SnapVault secondary. This device should be allocated to a resource pool.

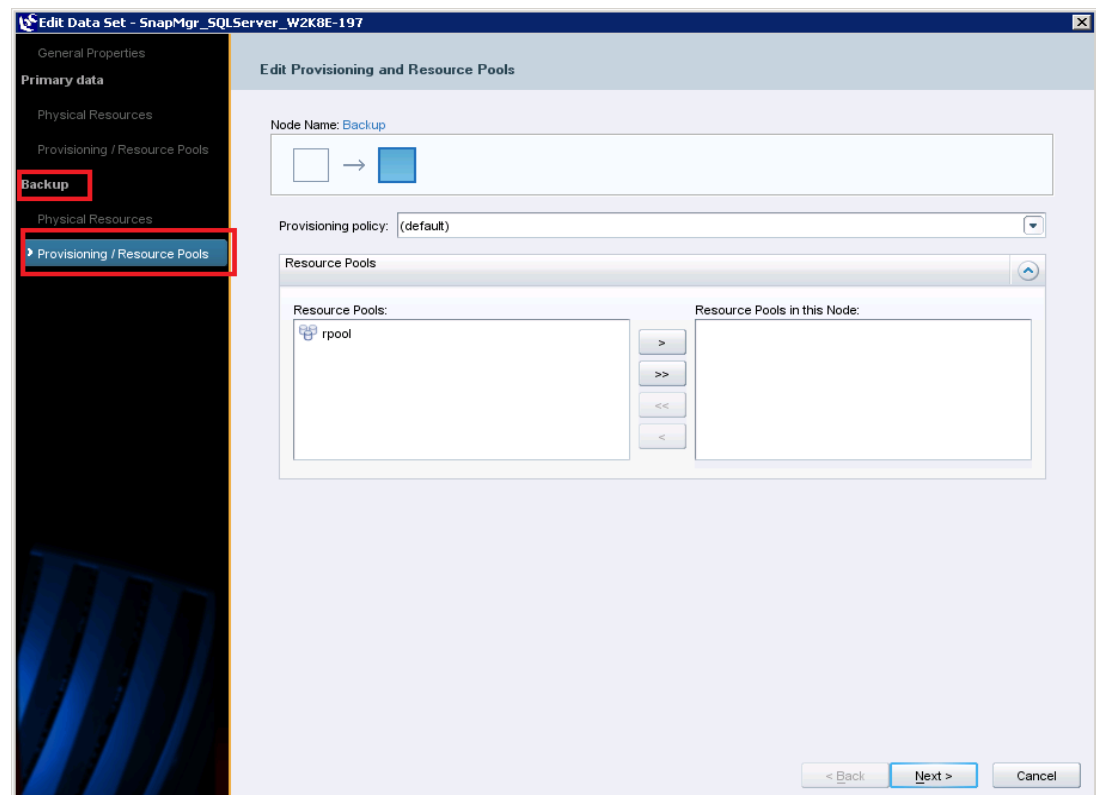
4. Now login into the NetApp Management console and go to the "Data" part that lists all the datasets that are created.



5. Right-click on the dataset and choose Edit.

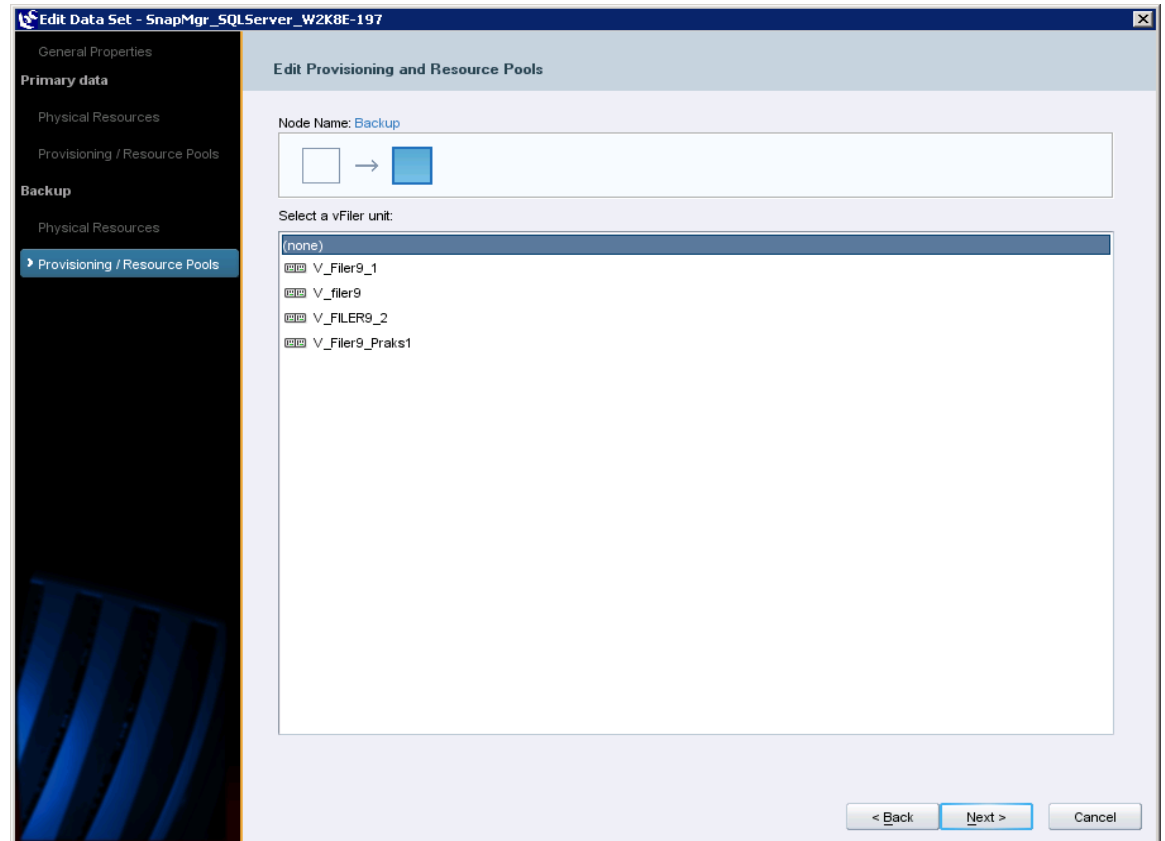


- In the “Edit Data Set -...” Dialog that opens, choose the option “Provisioning/Resource Pools” under the “Backup” section. Note that this will highlight the backup node in Blue.



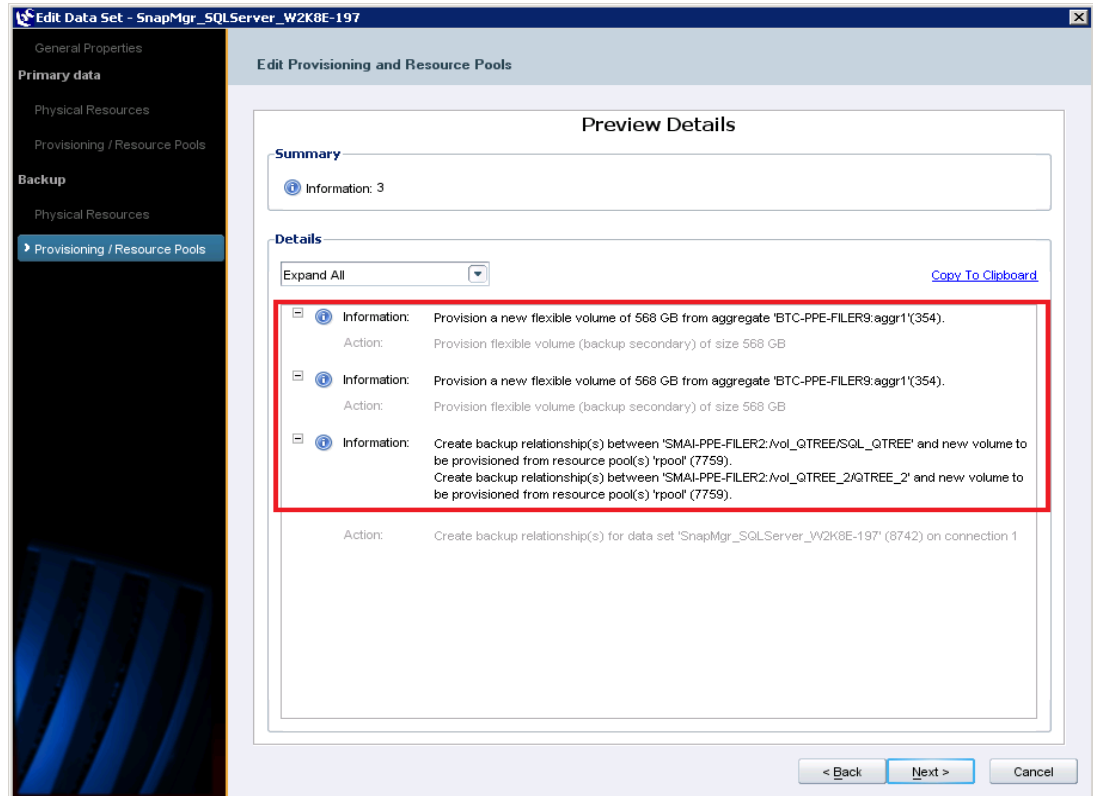


7. Choose the appropriate resource pool and the appropriate storage device in it that is configured to act as the SnapVault secondary. In our case we choose the resource pool named "rpool". Click Next.
8. Protection Manager will now ask you to select the appropriate Filer/vFiler in the resource pool: -

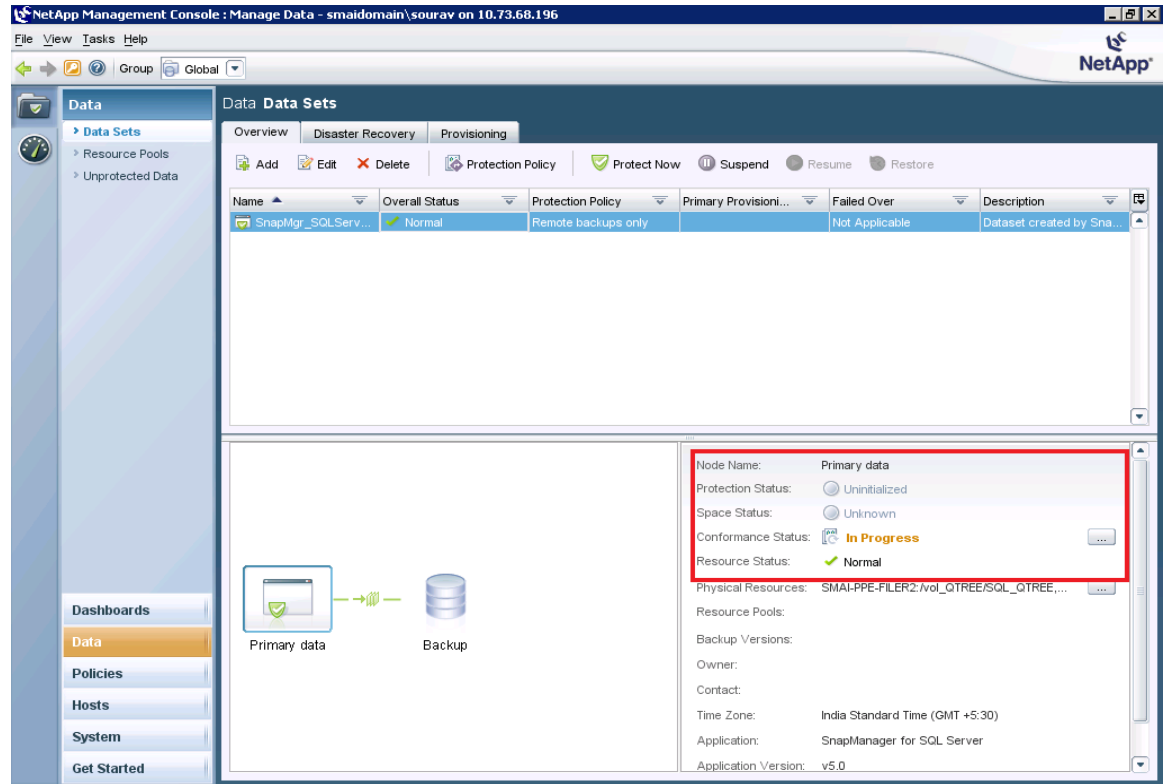


We will click Next here since we only have one physical filer in the resource pool.

9. Note that PM automatically created the required FlexVols on the SnapVault secondary. Click Next and Finish.



10. Close the “Edit Data Set” dialog box.
11. Note that Protection manager now tries to check the validity of the components chosen for the dataset. Note that the Conformation Status in the lower right hand says “In Progress”. In a while this says “Conformant”. At this point the configuration of the dataset is complete and the database in question is equipped to ship its backup-sets to a SnapVault secondary.



The steps to perform backup and restore to and from SnapVault destinations is well described in chapter 15 of the Installation and Administration Guide.

## 8 REMOTE BACKUP VERIFICATION WITH SMSQL 5.0

SMSQL 5.0 introduces an important feature with respect to backup verification. It now supports the verification of backup sets at SnapMirror and SnapVault destinations without breaking the relationships. In this section we present the details about this feature.

### 8.1 VERIFICATION OF SNAPMIRROR SECONDARIES

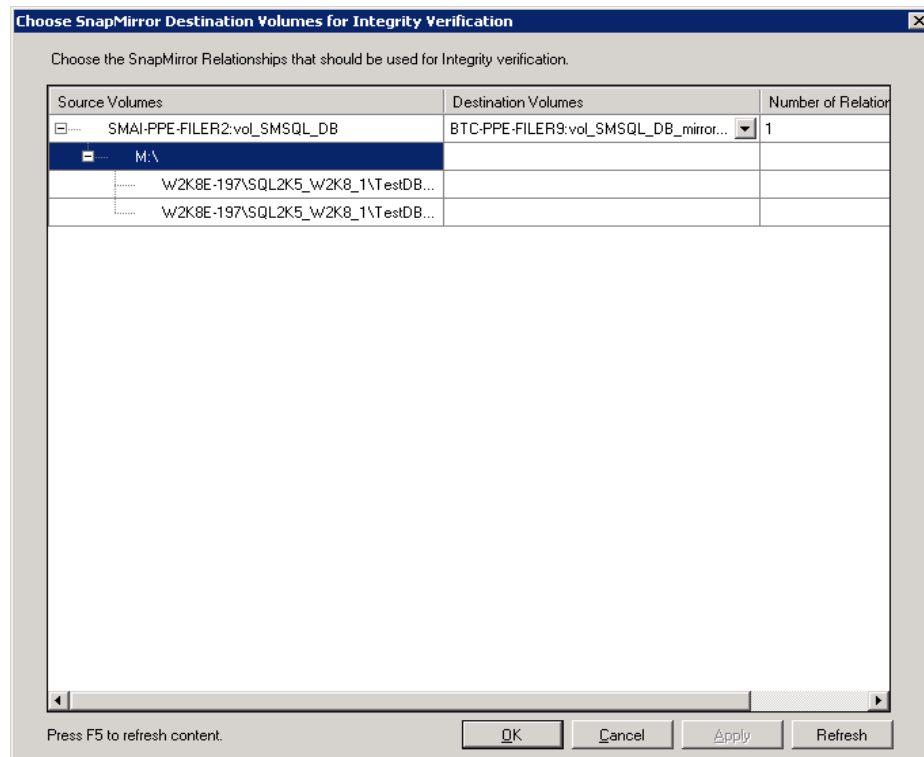
The pre-requisites for ensuring that SMSQL 5.0 is able to verify backup sets at the SnapMirror destination are: -

1. Data ONTAP version 7.1 or newer.
2. SnapMirror license on source filer.
3. FlexClone license on destination filers.
4. Enough disk space on destination filer to create a FlexClone volume.
5. In SQL Server host
  - a. Database(s) placed in volume(s) with SnapMirror relationships. Multiple SnapMirror destinations can be used.

## b. SnapDrive 6.0.1

**TIP:** In case the source database volumes have multiple snapmirror relationships, SMSQL 5.0 will verify the first default snapmirror destination that is returned by SDW.

**TIP:** SMSQL 5.0 provides a details dialog box that shows all the information such as source volume, destination volume etc. of a snapmirror relationship.



**TIP:** In order to mount the snapmirrored backups for verification, SMSQL firsts asks SDW to creates FlexClones of the SnapMirror destinations and mount the LUNs inside them to the verification server.

**TIP:** Verification Server must be kept as a non-production SQL Server host. For environments with large databases, it is preferable to have one dedicated SQL Server host as the verification server for each SMSQL instance.

**TIP:** DBCC CHECKDB is an extremely time consuming and resource-intensive operation. As far as possible use PHYSICAL ONLY option for DBCC CHECKDB to save time on optimizations. Using this option will instruct CHECKDB to only check physical consistency of a database. Note that there are

many other ways of verifying logical consistency including custom scripts. Additionally, while creating deferred verification jobs, one can maintain a window of last N-2 backups as being verified for logical consistency. This action will ensure that the verification window remains manageably small for rest of the backups in the verification window.

## 9 SUMMARY

SnapManager 5.0 (R1) for SQL Server is a powerful backup management tool for SQL Server databases. It has many powerful features that can be used to create numerous backup solutions for SQL Server databases.

## APPENDIX A

```
-- Create the table to accept the results

create table #output (dbname char(30),log_size real, usage real, status
int)

-- execute the command, putting the results in the table
insert into #output
    exec ('dbcc sqlperf(logspace)')

-- display the results
select *
from #output
go

--open cursor
declare output_cur cursor read_only for
select log_size
from #output

--make space computations
declare @v1 as real
declare @base_val as real

set @base_val = 0

open output_cur

FETCH NEXT FROM output_cur
INTO @v1

WHILE @@FETCH_STATUS = 0
BEGIN

    set @base_val = @base_val + @v1
```

```
    FETCH NEXT FROM output_cur
    INTO @v1
END

set @base_val = @base_val + 15
PRINT 'BASE_VAL = ' + cast(@base_val as nvarchar) + ' MB'

--clean-up
close output_cur
drop table #output
```

## REFERENCES

### **SnapManager for SQL**

[SnapManager for SQL 5.0 R1 Installation and Administration Guide](#)

### **SnapDrive for Windows**

[SnapDrive for Windows 6.0.1 Installation and Administration Guide](#)

### **Data ONTAP**

[Data ONTAP System Administration Guide](#)

[Data ONTAP Storage Management Guide](#)

### **NetApp SnapMirror**

[SnapMirror How-To-Guide](#)

[SnapMirror Best Practices Guide](#)

[Database Layout with Data ONTAP™ 7G](#)